



**APPLICATION  
STORAGE  
MANAGER**  
for  
**S/390**

VERSION 2 RELEASE 5

**USER MANUAL**

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this is software or related software documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, the following notice is applicable:

U.S. GOVERNMENT RIGHTS Programs, software, databases, and related documentation and technical data delivered to U.S. Government customers are "commercial computer software" or "commercial technical data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, duplication, disclosure, modification, and adaptation shall be subject to the restrictions and license terms set forth in the applicable Government contract, and, to the extent applicable by the terms of the Government contract, the additional rights set forth in FAR 52.227-19, Commercial Computer Software License (December 2007). Oracle USA, Inc., 500 Oracle Parkway, Redwood City, CA 94065.

This software or hardware and documentation may provide access to or information on content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services.

**Second Edition (February 2011)**

Oracle is a registered trademark of Oracle Corporation and/or its affiliates. Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Copyright ©2001,2011, Oracle and/or its affiliates. All rights reserved.

This edition applies to Version 2 Release 5 of the ASM for S/390 product.

Second edition

February 2011

This page is intentionally left blank

---

# TABLE OF CONTENTS

---

<b><i>INTRODUCTION</i></b>	<b><i>1.1</i></b>
<b>1.0 Introduction.</b>	<b>1.3</b>
<b>1.1 New in version 2.5.</b>	<b>1.3</b>
<b>1.2 New in version 2.4.</b>	<b>1.5</b>
<b>1.3 New in version 2.3.</b>	<b>1.6</b>
<b>1.4 How to update this manual.</b>	<b>1.8</b>
<b>1.5 Page numbering.</b>	<b>1.9</b>
 <b><i>SYSTEM DESCRIPTION</i></b>	 <b><i>1.1</i></b>
<b>1.1 System Overview.</b>	<b>1.3</b>
<b>1.2 Software Environment.</b>	<b>1.4</b>
<b>1.3 NearArchive Databases.</b>	<b>1.4</b>
1.3.1 Identifying a NearArchive Object.	1.5
1.3.2 Components of a NearArchive database.	1.7
<b>1.4 Application Program Interface.</b>	<b>1.10</b>
1.4.1 Batch processing.	1.11
1.4.2 CICS processing.	1.12
<b>1.5 Online Access to NearArchive Databases.</b>	<b>1.13</b>
1.5.1 Online Response Times.	1.14
<b>1.6 NearArchive Utilities.</b>	<b>1.19</b>
 <b><i>INSTALLATION AND IMPLEMENTATION PROCEDURES</i></b>	 <b><i>2.1</i></b>
<b>2.1 Install distribution libraries.</b>	<b>2.4</b>
<b>2.2 Host system modification.</b>	<b>2.6</b>
2.2.1 Install user SVC.	2.6
2.2.2 Update Pageable Link Pack Area.	2.7
<b>2.3 Update NearArchive authorization code(s).</b>	<b>2.7</b>
<b>2.4 Perform TSO logon procedure changes.</b>	<b>2.8</b>
<b>2.5 Perform ISPF menu changes.</b>	<b>2.10</b>
<b>2.6 NearArchive database definition.</b>	<b>2.10</b>
<b>2.7 Batch implementation.</b>	<b>2.11</b>
<b>2.8 Online implementation.</b>	<b>2.12</b>
2.8.1 NearArchive control region.	2.12
2.8.2 Secondary NearArchive regions.	2.13
2.8.3 Multiple NearArchive Database Access.	2.15
2.8.4 CICS Definitions for a NearArchive Control Region.	2.15
2.8.5 CICS definitions for a secondary NearArchive region.	2.22

2.8.6 CICS JCL changes.	2.23
<b>2.9 Security user exit.</b>	<b>2.27</b>
<b>2.10 Conversion of pre-v2.5 databases.</b>	<b>2.30</b>
2.10.1 Database regression.	2.31
<b>2.11 Run installation verification procedures.</b>	<b>2.32</b>

## **NEARARCHIVE DATABASE DEFINITION AND MAINTENANCE** **3.1**

<b>3.1 Definition of a New NearArchive Database.</b>	<b>3.5</b>
3.1.1 Base Database Definition.	3.7
3.1.2 Storage Level 0 Definition.	3.12
3.1.3 Access Method Services Control.	3.21
3.1.4 System Dataset Sizing.	3.25
<b>3.2 Maintenance of a NearArchive Database.</b>	<b>3.28</b>
3.2.1 Base Database Maintenance.	3.29
3.2.2 Secondary Index Maintenance.	3.34
3.2.3 Storage Level Maintenance.	3.45
<b>3.3 Browse a NearArchive Database Definition.</b>	<b>3.49</b>
<b>3.4 Recovery Control Processing.</b>	<b>3.50</b>
3.4.1 Volume recovery/unavailable volume processing.	3.52
3.4.2 Set Disaster Recovery.	3.56
3.4.3 Unset Disaster Recovery.	3.57
<b>3.5 Pre-v2.5 Database Conversion Processing.</b>	<b>3.57</b>
3.5.1 Increase maximum primary index record size.	3.59

## **THE NEARARCHIVE PROGRAMMING INTERFACE** **4.1**

<b>4.1 Invoking NearArchive.</b>	<b>4.3</b>
<b>4.2 The NearArchive program interface.</b>	<b>4.4</b>
<b>4.3 NearArchive Request Codes.</b>	<b>4.7</b>
4.3.1 Request Code 40 – Connect to a NearArchive Database.	4.8
4.3.2 Request Code 41 – Query a NearArchive Index.	4.14
4.3.3 Request Code 42 – Retrieve an Object Record.	4.23
4.3.4 Request Code 43 – Retrieve a Complete Object.	4.28
4.3.5 Request Code 44 – Write a Data Record.	4.31
4.3.6 Request Code 46 – Identify Secondary Key.	4.34
4.3.7 Request Code 47 – Delete an Object.	4.37
4.3.8 Request Code 48 – Commit Updates.	4.39
4.3.9 Request Code 49 – Disconnect a NearArchive Database.	4.42

## **BATCH PROCESSING** **5.1**

<b>5.1 Invoking NearArchive from batch programs.</b>	<b>5.3</b>
5.1.1 JCL requirements.	5.3
5.1.2 Controlling Access to a NearArchive Database.	5.5

5.1.3 Database design considerations.	5.5
<b>5.2 Batch Archival Processing.</b>	<b>5.6</b>
5.2.1 Processing Overview.	5.6
5.2.2 Recovery and Restart.	5.8
5.2.3 Database Backup and Recovery.	5.12
<b>5.3 Batch Retrieval Processing.</b>	<b>5.21</b>
5.3.1 Processing Overview.	5.21
5.3.2 Operational considerations.	5.23
5.3.3 Logging of retrieval activity.	5.24

---

## ***CICS PROCESSING*** ***6.1***

<b>6.1 Application Programming Issues.</b>	<b>6.3</b>
6.1.1 Overview of Object Retrieval Processing.	6.3
6.1.2 Partial Object Retrieval.	6.6
6.1.3 TSQ stringing.	6.7
<b>6.2 Invoking NearArchive from CICS programs.</b>	<b>6.8</b>
<b>6.3 Operational Considerations.</b>	<b>6.9</b>
6.3.1 General Issues.	6.9
6.3.2 Concurrent Retrievals - Different Tape Cartridge Volumes.	6.10
6.3.3 Concurrent Retrievals - Same Tape Cartridge Volume.	6.11
6.3.4 Multiple NearArchive Database Support.	6.12
<b>6.4 Starting and Stopping NearArchive During CICS Operation.</b>	<b>6.12</b>
6.4.1 OT23 - Initializing a NearArchive Control Region.	6.13
6.4.2 OT24 - Quiescing a NearArchive Control Region.	6.13
<b>6.5 NearArchive Operator Interface.</b>	<b>6.14</b>
6.5.1 Invoking the Operator Interface in Line-Entry Processing Mode.	6.15
6.5.2 Invoking the Operator Interface in Full-Screen Processing Mode.	6.17
6.5.3 Display Status Summary.	6.19
6.5.4 Display Status Detail.	6.23
6.5.5 Alter NearArchive Configuration.	6.26
6.5.6 Exiting from operator interface processing.	6.29
<b>6.6 Logging of retrieval activity.</b>	<b>6.30</b>

---

## ***NEARARCHIVE UTILITIES*** ***7.1***

<b>7.1 OTASP010 - the NearArchive Index Print Utility.</b>	<b>7.6</b>
7.1.1 Functions.	7.6
7.1.2 JCL Requirements.	7.6
7.1.3 Condition Codes.	7.9
7.1.4 Running the Utility.	7.9
7.1.5 Utility Processing Reports.	7.9
7.1.6 Utility Failure and Restart Considerations.	7.11
<b>7.2 OTASP020 - the NearArchive Tape Scan Utility.</b>	<b>7.12</b>
7.2.1 Functions.	7.12

7.2.2 JCL Requirements.	7.12
7.2.3 Condition Codes.	7.14
7.2.4 Running the Utility.	7.15
7.2.5 Utility Processing Reports.	7.15
7.2.6 Utility Failure and Restart Considerations.	7.16
<b>7.3 OTASP030 - the NearArchive Database Maintenance Utility.</b>	<b>7.17</b>
7.3.1 Functions.	7.18
7.3.2 JCL Requirements.	7.20
7.3.3 Condition Codes.	7.24
7.3.4 Running the Utility.	7.25
7.3.5 Operator commands.	7.26
7.3.6 Utility Processing Reports.	7.27
7.3.7 Utility Failure and Restart Considerations.	7.27
<b>7.4 OTASP040 - the NearArchive Object Management Utility.</b>	<b>7.29</b>
7.4.1 Functions.	7.30
7.4.2 JCL Requirements.	7.31
7.4.3 Condition Codes.	7.35
7.4.4 Running the Utility.	7.36
7.4.5 Operator commands.	7.37
7.4.6 Utility processing reports.	7.37
7.4.7 Utility Failure and Restart Considerations.	7.39
<b>7.5 OTASP070 - the NearArchive Journal Maintenance Utility.</b>	<b>7.40</b>
7.5.1 Functions.	7.40
7.5.2 JCL Requirements.	7.41
7.5.3 Condition Codes.	7.45
7.5.4 Running the Utility.	7.45
7.5.5 Utility Processing Reports.	7.46
7.5.6 Utility Failure and Restart Considerations.	7.48
7.5.7 Journal Dataset Specification.	7.48
<b>7.6 OTASP080 - the NearArchive Secondary Index Update Utility.</b>	<b>7.52</b>
7.6.1 Functions.	7.53
7.6.2 JCL Requirements.	7.54
7.6.3 Condition Codes.	7.56
7.6.4 Running the Utility.	7.56
7.6.5 Utility Processing Reports.	7.57
7.6.6 Utility Failure and Restart Considerations.	7.57
<b>7.7 OTASP110 - the NearArchive Database Backup Utility.</b>	<b>7.58</b>
7.7.1 Functions.	7.59
7.7.2 JCL Requirements.	7.61
7.7.3 Condition Codes.	7.66
7.7.4 Running the Utility.	7.66
7.7.5 Utility Processing Reports.	7.68
7.7.6 Utility Failure and Restart Considerations.	7.68
<b>7.8 OTASP130 - the NearArchive Database Recovery Utility.</b>	<b>7.70</b>
7.8.1 Functions.	7.71



7.8.2 JCL Requirements.	7.72
7.8.3 Condition Codes.	7.74
7.8.4 Running the Utility.	7.75
7.8.5 Utility Processing Reports.	7.75
7.8.6 Utility Failure and Restart Considerations.	7.76

<b><i>MESSAGES AND CODES</i></b>	<b><i>8.1</i></b>
----------------------------------	-------------------

<b>8.1 NearArchive Request Codes.</b>	<b>8.4</b>
<b>8.2 NearArchive Return and Supplementary Codes.</b>	<b>8.6</b>
8.2.1 System File Identification.	8.22
<b>8.3 Index Print Utility Messages.</b>	<b>8.23</b>
<b>8.4 Tape Scan Utility Messages.</b>	<b>8.26</b>
<b>8.5 Database Maintenance Utility Messages.</b>	<b>8.29</b>
<b>8.6 Database Regression Utility Messages.</b>	<b>8.45</b>
<b>8.7 Object Management Utility Messages.</b>	<b>8.49</b>
<b>8.8 Journal Maintenance Utility Messages.</b>	<b>8.60</b>
<b>8.9 Secondary Index Update Utility Messages.</b>	<b>8.69</b>
<b>8.10 Database Backup Utility Messages.</b>	<b>8.74</b>
<b>8.11 Batch Archival Request Processor Messages.</b>	<b>8.87</b>
<b>8.12 Database Recovery Utility Messages.</b>	<b>8.89</b>
<b>8.13 Batch Archival Tape I/O Driver Messages.</b>	<b>8.101</b>
<b>8.14 Online Request Processor Messages.</b>	<b>8.102</b>
<b>8.15 Control Region Initialization Messages.</b>	<b>8.106</b>
<b>8.16 Online Parameter File (OTASS100) Processor Messages.</b>	<b>8.108</b>
<b>8.17 Control Region Quiesce Messages.</b>	<b>8.111</b>
<b>8.18 CICS Reader Task Messages.</b>	<b>8.112</b>
<b>8.19 CICS Operator Interface Messages.</b>	<b>8.114</b>
<b>8.20 CICS Housekeeping Processing Messages.</b>	<b>8.131</b>
<b>8.21 Batch Retrieval Request Processor Messages.</b>	<b>8.133</b>
<b>8.22 Database Initialization Messages.</b>	<b>8.134</b>
<b>8.23 General-purpose Component Messages.</b>	<b>8.137</b>
<b>8.24 SMF record format.</b>	<b>8.138</b>
8.24.1 Header section.	8.139
8.24.2 Record descriptor section.	8.140
8.24.3 Database section.	8.141
8.24.4 Request section.	8.142
<b>APPENDIX A: Installation Verification Procedure.</b>	<b>A.1</b>
<b>A.1 IVP Datasets.</b>	<b>A.2</b>
<b>A.2 IVP Database Initialization.</b>	<b>A.3</b>
A.2.1 Define and Initialize IVP Database.	A.3

A.2.3 Define Secondary Indexes.	A.5
<b>A.3 CICS Definitions.</b>	<b>A.7</b>
<b>A.4 Running the Batch IVP.</b>	<b>A.9</b>
A.4.1 The batch archival IVP.	A.9
A.4.2 The batch retrieval IVP.	A.10
<b>A.5 Running the online retrieval IVP.</b>	<b>A.11</b>
<b>APPENDIX B: Sample NearArchive Calls.</b>	<b>B.1</b>
<b>B.1 Batch Archival Processing.</b>	<b>B.3</b>
B.1.1 Connect to NearArchive Database.	B.3
B.1.2 Write Record to NearArchive Database.	B.5
B.1.3 Identify Secondary Key.	B.6
B.1.4 Commit Updates.	B.7
B.1.5 Disconnect Database.	B.8
<b>B.2 Batch Retrieval Processing.</b>	<b>B.9</b>
B.2.1 Connect Database.	B.9
B.2.2 Query Database.	B.11
B.2.3 Retrieve Data Record (Direct Retrieval).	B.13
B.2.4 Retrieve Data Record (Sequential Retrieval).	B.15
B.2.5 Delete NearArchive Object.	B.17
B.2.6 Disconnect Database.	B.18
<b>B.3 CICS Processing.</b>	<b>B.19</b>
B.3.1 Retrieve Archived Object.	B.19
B.3.2 Query Database.	B.21
B.3.3 Delete Object.	B.23
<b>APPENDIX C: Database Regression Utility</b>	<b>C.1</b>
<b>C.1 Functions.</b>	<b>C.1</b>
<b>C.2 JCL Requirements.</b>	<b>C.2</b>
<b>C.3 Condition Codes.</b>	<b>C.5</b>
<b>C.4 Operator commands.</b>	<b>C.6</b>
<b>C.5 Utility Processing Reports.</b>	<b>C.6</b>
<b>C.6 Utility Failure and Restart Considerations.</b>	<b>C.6</b>
<b>C.7 Post-utility processing</b>	<b>C.7</b>
<b>APPENDIX D: Pre-V2R4 API Specifications</b>	<b>D.1</b>
<b>D.1 The batch archival interface.</b>	<b>D.1</b>
D.1.1 Request Codes 10/11/12 - Open Database.	D.3
D.1.2 Request Code 13 - Write Data Record.	D.6
D.1.3 Request Code 14 - Identify Secondary Key.	D.8
D.1.4 Request Code 15 - Close Database.	D.10
D.1.5 Request Code 16 - Commit Updates.	D.12
<b>D.2 The CICS retrieval interface.</b>	<b>D.13</b>
D.2.1 Request Code 21 - Query Index.	D.14
D.2.2 Request Code 22 - Retrieve Object.	D.19
D.2.3 Request Code 23 - Delete Object.	D.21

<b>D.3 The batch retrieval interface.</b>	<b>D.23</b>
D.3.1 Request Code 30 - Open Database.	D.24
D.3.2 Request Code 31 - Query Index.	D.26
D.3.3 Request Code 32 - Read data record.	D.30
D.3.4 Request Code 33 - Delete Object.	D.34
D.3.5 Request Code 34 - Close database.	D.35
<b>APPENDIX E: Managing Disk ('K') Copy Capacity.</b>	<b>E.1</b>
<b>E.1 Disk ('K') Copy Capacity</b>	<b>E.1</b>
E.1.1 Conventional Physical Sequential (non-SMS)	E.1
E.1.2 Extended Format Data Sets (SMS)	E.1
E.1.3 Striped EFDS (SMS)	E.2
<b>E.2 Disk ('K') Copy Blocksize Considerations</b>	<b>E.2</b>
<b>E.3 SMS Requirements for EFDS/Striping</b>	<b>E.3</b>
E.3.1 Data Class Definition for EFDS/Striping	E.3
E.3.2 Sample ACS Routine for Data Class Selection	E.3
E.3.3 Storage Class Definition for EFDS/Striping	E.4
E.3.4 Sample ACS Routine for Storage Class Selection	E.4

This page is deliberately blank

---

# CHAPTER - I

---

## *Introduction*

This page is intentionally left blank

## **I.0 Introduction.**

---

This ASM for S/390 User Manual describes the implementation and usage of version 2.5 of Oracle's Application Storage Manager (ASM) for S/390 database management product. It is an updated version of the NearArchive version 2.4 User Manual.

ASM for S/390 was known in earlier releases as NearArchive. In order to provide continuity with previous releases, the user manual for this release will use the two names interchangeably. In general, references to the product itself will specify "ASM for S/390"; references to product components will specify "NearArchive". User interfaces supplied by this release of the software will continue to identify the product as "NearArchive".

As before, the ASM for S/390 user manual is intended for use by application system designers and developers to enable them to use the database management facilities provided by ASM for S/390 in order to incorporate the processing of archived data into existing or planned application systems.

Version 2.5 of ASM for S/390 will only execute with OS/390 version 1.1 or higher, and CICS/ESA version 3.3 or higher (for online database access). This manual does not contain any information which is dependent on the release of OS/390 or CICS which is in use on the ASM for S/390 host. Any such release-dependent information will be contained in the installation material sent to customers with the distribution tape.

## **I.1 New in version 2.5.**

---

The following enhancements have been introduced in version 2.5 of ASM for S/390:

- The maximum number of storage levels which may be defined for a NearArchive database has been increased from 2 to 8. To support this enhancement, modifications have been introduced to the object management utility (OTASP040) to allow direct migration between any two storage levels in the database. In addition, applications will now be able to store objects directly into any storage level in the database.

- Up to 256 disk ('K') copy datasets may now be created for each tape volume in a database storage level, for parallel storage of database objects on disk. This will enable more active management of disk copy datasets, and may substantially reduce the amount of disk space required to support this feature of the product.
- An extended database query application program interface has been introduced to return information on the range of tape datasets on which an object is stored (in addition to the information returned by the existing query interface), or to return volume serial numbers for the tape volumes containing those datasets.
- Full support has been introduced for the Sun Oracle Volsafe <sup>TM</sup> feature, to enforce usage or non-usage of Volsafe-enabled tape volumes for storage of objects in a database. This feature can be enabled separately for each tape copy within each storage level of a database. V2.5 also introduces full support for IBM 3590 tape devices, in addition to 3480/3490/3490E-type devices. This includes support for Sun Oracle's RedWood <sup>TM</sup>, 9840 and 9940 devices operating in both 3490E and 3590 emulation modes.
- Support for use of an external security user exit has been introduced. This will cause a user exit to be invoked at control points within the product to allow external security processing to be performed in order to establish whether the application is allowed to access a protected resource. Access can be permitted or denied by customer-dependent processing within the user exit.
- Support has been introduced for a timeout facility during CICS object retrieval processing. ASM for S/390 will now terminate a stalled retrieval request (eg. a request waiting for a tape mount to be satisfied, or some other external condition) when the time interval specified in a new CICS control region initialization parameter has expired.
- Two new batch utilities are supplied with the product. OTASP010 is used to produce a formatted report on the contents of the primary index of a NearArchive database. OTASP020 is used to produce a formatted report on the contents of a single tape volume in a NearArchive database. Both utilities are intended primarily for use in assisting with diagnostic procedures during problem analysis.



## **I.2 New in version 2.4.**

---

The following enhancements were introduced in version 2.4 of NearArchive:

- Year 2000 compliance for all external interfaces. Four digits are now used to identify year values in all external interfaces of the product (i.e.) the application program interface (API), CICS screens, TSO/ISPF panels, print reports and system messages. A new API is supplied to allow calling applications to identify archived objects using an 8-character archive date (in the format YYYYMMDD). The existing API (which identifies objects using a 6-character date in the format YYMMDD) will continue to be supported. V2.4 will only permit archive dates in the range 1950-2049 to be assigned to objects being stored in a NearArchive database – this restriction is enforced by default when using the earlier (6-digit) API.
- The database backup utility (OTASP110) may now be used to re-create a disk ('K') copy dataset from a primary tape copy.
- A disk ('K') copy dataset compression facility has been added to the functionality supplied by the database maintenance utility (OTASP030). This facility will allow disk copy datasets which contain lower than a user-supplied threshold of active disk copy objects to be compressed, allowing space occupied by expired disk copy objects to be released.
- The unit name used internally by NearArchive for creating temporary system disk datasets during product operation ('SYSALLDA') may now be overridden by the customer.
- Unused space assigned after creation of a disk ('K') copy dataset may now optionally be automatically released after closure of the dataset.
- Support for Double Byte Character Set (DBCS) in primary and / or secondary keys.

The following enhancements to CICS object retrieval processing are shipped with this release:

- Partial object retrieval. This feature allows CICS applications to retrieve a segment of a stored object to the target temporary storage queue, rather than the complete object. Object segments are designated by start record number and a count of the number of records in the segment.

- Retrieval prioritization. Calling applications may now optionally supply a priority (in the range 0-9, where 9 is the highest priority) to be assigned to an object retrieval request. This priority will be used by NearArchive when internally processing retrieval requests.
- Internal queuing of requests which cannot be processed at the time of submission. If a request cannot be processed due to the unavailability of one or more resources (e.g. tape drive, tape volume etc.), the request may now optionally be held internally by NearArchive for a customer-designated period of time. The retrieval request will be retried until it is satisfied, or until the designated period of time has expired. At this point it will be rejected with a 'resource unavailable' condition.
- SMF logging of batch and CICS retrieval activity may now be optionally enabled.
- All NearArchive CICS programs are now supplied with 31-bit addressing and residency mode. OS/390 CSA storage used by the NearArchive control region is now assigned in the extended CSA.
- A temporary storage queue (TSQ) 'stringing' facility is now supplied for CICS object retrieval processing, to allow NearArchive objects with greater than 32,767 records to be accessed successfully from a CICS application.
- The temporary storage queue identified in a CICS object retrieval request for storage of the retrieved object will be automatically deleted by NearArchive if it already exists, when the new v2.4 API is used. A retrieved object will continue to be appended to an existing temporary storage queue when the old (pre-v2.4) API is used.
- Modifications have been introduced in the way that the TSQDELETE initialization parameter is serviced. A value of 0 for this parameter will now force NearArchive to retrieve objects from tape even if the object exists in CICS temporary storage. The default value for this parameter has been changed to 1440 minutes. Refer to section 2.8.6 for more information on this enhancement.

## **I.3 New in version 2.3.**

---

The following enhancements were introduced in version 2.3 of NearArchive:

- Backup and recovery processing has been enhanced to provide an incremental backup/recovery facility. This will reduce backup volumes and execution times when processing large capacity primary

tape volumes. In addition, a new ‘speed’ option has been added to the backup utility, to reduce execution times for full-volume backup processing.

- Automatic detection of tape device type/tape volume capacity. This removes the need for explicit definition of tape device type during database definition processing, and ensures that tape duplexing and backup/recovery processing does not fail due to differing tape volume capacities.
- Dynamic load balancing. This feature (introduced in V2.2 via product service), when enabled, causes NearArchive to automatically access a designated backup copy when retrieval of an object from a primary copy has failed due to an unavailable resource (tape volume or tape device).
- Object storage on disk. This optional feature will enable objects within a NearArchive database to be stored on disk, in addition to tape, for a customer-designated period of time. Object retrievals will always be satisfied from the disk copy, if it exists. The disk copy will be deleted after the designated period, during execution of the NearArchive object management utility. Object retrievals after this time will be satisfied from the tape copy.
- Dynamic backout facility. This feature (introduce via product service for V2.2), when enabled, will cause NearArchive to backout all updates to a database during batch archival processing, after a ‘close database without commit’ call has been issued by an application. Dynamic backout will cause all primary and secondary index entries for uncommitted objects to be removed, and for tape storage occupied by those objects to be released. This storage will be over-written during further archival activity.

## **I.4 How to update this manual.**

---

Periodically you will receive documentation updates for this manual. Revised material will be indicated by a change bar in the left margin of the page.

A page may be revised due to a new software release, a fix to be applied to existing software, or an additional feature added to the software.

Where a documentation update requires insertion of additional pages, each additional page number will be suffixed by an insert number.

When you receive updates to the manual, insert the revised pages, and discard any outdated material. Note that this material is proprietary; please treat it as documentation confidential to your company.

## **I.5 Page numbering.**

---

The page numbering scheme for this publication works as follows:

Document and chapter titles appear on every page. The chapter title is printed on the page header, and the document title is printed on the page footer.

Pages are numbered consecutively within chapters, and each page number is preceded by the chapter number (e.g.) page 1.1 indicates Chapter 1, Page1.

This page is intentionally left blank

---

# CHAPTER 1

---

## *System Description*

This page is intentionally left blank



## 1.1 System Overview.

---

Oracle's ASM for S/390 product is a database management system intended to facilitate the archival of inactive data to tape (and optionally disk) storage, and to enable the retrieval of this archived data in a batch or online processing environment.

The product supplies a series of application program interfaces to support all the functions involved in writing and retrieving data to and from a NearArchive database, and supplies all the support facilities (via batch utilities and TSO/ISPF application dialogs) which are required for database access and administration functions. This includes support for the access of tape-resident objects in a CICS environment, which is required for online recall of data from a NearArchive database.

High-level interfaces are supplied for access to NearArchive databases, to enable archive database processing to be fully integrated into existing or planned batch or online applications. In this manner it is possible, using ASM for S/390, to extend existing system access to online data, via database management systems such as IMS/DB or DB2, to unlimited quantities of archived data stored on tape, while still retaining the ability to access this data online.

Typical applications which may be appropriate for ASM for S/390 implementation are:

- Customer billing. In a service environment, for example, customers may be billed quarterly or monthly. Current bills may be held in disk databases, allowing older inactive bills to be archived via ASM for S/390 to tape.

Inactive bills may be required to be held in the archive for a specific period for legal or other reasons. Using ASM for S/390, these bills may be stored on inexpensive tape media, thus releasing expensive disk storage for more active use. ASM for S/390 will continue to allow users/customers to access all archived data immediately, without the long delays typical of batch retrieval facilities. This may be useful for enabling improved customer service facilities (e.g.) immediate response to branch office or telephone queries.

- Periodic monitoring information (e.g.) meter readings. Information which is produced periodically and logged on disk by application processing may be archived via ASM for S/390 to tape and retained for an unlimited period, without impacting disk storage requirements. This information will still be available for immediate online access when required by CICS application users.
- Transaction details (e.g.. financial debit and credit transactions) may be archived on a periodic basis (e.g.. monthly), all transactions during that period being archived to a NearArchive database. All transactions for one particular account in any particular period may then be recalled from the database via a single request.

## **1.2 Software Environment.**

---

Batch archival and retrieval with ASM for S/390 operates using OS/390 as the system control program, and requires host support for 3480/3490 and/or 3590 devices. ASM for S/390 v2.5 will execute with versions of OS/390 from 1.1 upwards.

Online retrieval is performed using CICS as the transaction processing software. ASM for S/390 uses the standard CICS command-level interface, and will execute with releases of CICS/ESA from 4.1 upwards.

## **1.3 NearArchive Databases.**

---

ASM for S/390 stores data in one or more archival databases. Multiple databases may exist on each host system. Each NearArchive database must be defined and initialized prior to use, using the supplied ASM for S/390 TSO/ISPF database administration facility. All database administration functions are performed interactively using this facility. These functions include:

- Defining a new NearArchive database.
- Amending an existing NearArchive database.
- Browsing a NearArchive database definition.
- Controlling recovery options for a NearArchive database.

Data is held in a NearArchive database as a series of NearArchive 'objects'. Each object may consist of one or more variable-length logical records, each record containing up to 32760 bytes. There is no limit to the number of logical records in each object.

During the data archival procedure, each logical record within an object is passed sequentially to ASM for S/390 for insertion in the database. During ASM for S/390 object retrieval, each of its component logical records will be separately identifiable to the retrieving batch or online application.

### **1.3.1 Identifying a NearArchive Object.**

A NearArchive object is identified by a unique combination of primary key and archive date. Multiple objects with the same primary key may exist in the database, but the archive date for each of these objects must be different.

In addition, a NearArchive object may be indexed by one or more secondary keys, using the ASM for S/390 secondary indexing facility. Up to 99 separate secondary indexes may be defined for each NearArchive database, each secondary index containing entries which index NearArchive objects by secondary key. A secondary key may index more than one NearArchive object.

The primary and / or secondary keys may contain DBCS characters which should include all shift control bytes.

#### **Example:**

A NearArchive database is created to hold archived customer account information. Each NearArchive object will hold a customer's account transactions over a three-month period.

The primary key for this database will be the customer account number. Secondary index #1 is defined with a secondary key of customer surname. Secondary index #2 is defined with a secondary key of customer's date of birth.

Suppose an object has been archived to the NearArchive database for a customer with the following details:

<i>Account no:</i>	0123456789
<i>Surname:</i>	SMITH
<i>Date of birth:</i>	Jan 1, 1966

This object will contain all account transactions for this customer during the first three months of 1992.

This NearArchive object will be uniquely identified by a primary key value of '0123456789' and an archive date of '19980331', say (the last day in the archive period in YYYYMMDD format). It will also be identified by a secondary key value of 'SMITH' in secondary index #1, and of '19660101' (YYYYMMDD) in secondary index #2.

This object may be retrieved from online or batch processing via its unique primary key and archive data combination.

Alternatively, it may be retrieved by first specifying a secondary key value of 'SMITH' (for secondary index #1) or '19660101' (for secondary index #2). The required NearArchive object may then be identified from the multiple objects which may be indexed by either of these secondary keys, and retrieved by primary key as above.

Primary and secondary index entries are created by ASM for S/390 when an object is written to a NearArchive database. Primary index entries are always created. Entries will be automatically created for each secondary index which has been defined for that database. A secondary index entry will be generated for a NearArchive object under the following circumstances:

- when automatic entry creation has been defined for that secondary index. With this facility, the secondary key used in the entry will be located at character position 'a' in logical record number 'b' within a NearArchive object. Every secondary key within a single secondary index will have the same keylength 'c'. 'a', 'b' and 'c' are values which are specified for each secondary index using the NearArchive database administration facility.

Using these rules, ASM for S/390 will automatically identify the secondary key(s) to be used to index an object.

- by explicit request from a ASM for S/390 application. During object archival, an application may explicitly identify a secondary index and key to be used to index that object. The length of the secondary key must be equal to the secondary keylength defined for that secondary index.

Secondary index update processing may be performed synchronously during object archival processing (the default) or asynchronously using the secondary index update utility OTASP080.

### **1.3.2 Components of a NearArchive database.**

A NearArchive database consists of the following components:

- One or more single volume tape datasets containing archived objects.
- Optionally, one or more sequential disk datasets, containing copies of archived objects.
- A primary index dataset (VSAM KSDS).
- One or more secondary index datasets (VSAM KSDS).
- Between one and eight space management datasets (VSAM RRDS) (one per database storage level).
- A journal dataset (VSAM ESDS) (if audit trail facility enabled).
- A migration control dataset (VSAM ESDS) (if multiple storage levels defined).

All the above datasets are automatically created and maintained by ASM for S/390. Each database is identified in batch JCL via its primary index dataset. All other ASM for S/390 system datasets are dynamically allocated as required. Details of the sizing of each of the VSAM datasets specified above may be found in chapter 3 of this manual.

Each NearArchive database may consist of up to eight storage levels; these are referred to throughout this manual as storage level 0 (SL0) to storage level 7 (SL7). Data in each storage level is held in a series of single-volume sequential tape datasets. A maximum of 65535 datasets (and consequently 65535 tape cartridge volumes) may be present in each storage level. The amount of data held for each volume within a storage level may be restricted via the NearArchive database administration facility. Alternatively, the full capacity of a volume may be used if required. ASM for S/390 includes support for use of Sun Oracle's 3480, 3490, 3490E and 3590-compatible devices, and RedWood™, 9840 and 9940 high-capacity tape devices. Support is also provided for Sun Oracle's Volsafe™ tape data protection feature.

Each tape dataset in a database storage level may have up to 256 corresponding sequential disk datasets, if disk copy processing is enabled for that storage level. The size of each disk dataset is controlled via the storage level definition procedure, using the NearArchive ISPF database administration application. Disk copies of tape-resident objects may optionally be retained for a customer-designated period of time after objects are first archived in a database storage level.

A space management dataset will exist for each storage level in the database. No other additional disk dataset usage is required for multiple storage level support.

Storage levels within a NearArchive database are defined using the NearArchive database administration facility in TSO/ISPF. Each storage level is defined individually, enabling separate control of capacity and performance requirements by storage level. For example, data in storage level 0 may be held on half-filled 18-track tapes, while data in storage level 1 may be held on completely-filled 36-track tapes.

Objects may be automatically migrated between any one storage level and any higher storage level number, using the NearArchive object management utility (eg. directly from SL0 to SL7). Object migration is controlled by the NearArchive database administrator, on the basis of age of object. In this way, objects may reside in storage level 0 for the first period of their life in the NearArchive database (giving improved retrieval performance), and then be migrated to one or more higher storage levels for the remainder of their existence (giving successively higher-density storage with successively reduced retrieval performance).

The NearArchive database administrator may optionally define a retention period for objects within a NearArchive database. When this retention period is exceeded for an object, it will be automatically deleted by the NearArchive object management utility.

Index space occupied by deleted objects will be automatically re-used by ASM for S/390 during archival of new objects to that database. The NearArchive database maintenance utility must be executed to reclaim tape space occupied by deleted objects in a NearArchive database. This utility will recycle tape cartridge volumes whose level of active objects falls below a threshold set by the NearArchive database administrator. Recycled volumes will be released from the database and may be returned to the appropriate scratch tape cartridge pool.

All updates to a NearArchive database may optionally be logged to the System Management Facility (SMF) datasets. This facility is activated by the NearArchive database administrator. Customer processes may be used to produce statistical analyses from this information.

Up to four copies of all tape datasets in a NearArchive database may exist. These copies are identified within ASM for S/390 as follows:

1. 'A' copy. This is the primary dataset created by ASM for S/390 when objects are written to a database.

2. 'B' copy. This is a copy of the primary 'A' copy, created by the NearArchive database backup utility.
3. 'C' copy. This is a duplex copy of the primary 'A' copy, which may optionally be created concurrently with the 'A' copy when objects are written to a NearArchive database.
4. 'D' copy. This is a duplex copy of the 'B' backup copy, which may optionally be created concurrently with the 'B' copy by the NearArchive database backup utility.

All the above copies are created automatically by ASM for S/390. These copies may be used to implement database recovery processing in the following circumstances:

- loss of individual tape cartridge volume(s) from within a database. An individual volume may become unavailable for object retrieval due to media failure, physical loss, or for some other reason.
- loss of an entire database. All tape cartridges within a NearArchive database may become unavailable for object retrieval due to loss of a data center, or serious hardware malfunction.

In each of the above circumstances, ASM for S/390 provides facilities to enable access to a NearArchive database to continue without interruption, and without the need to physically restore any tape cartridge volumes. NearArchive data recovery functions are implemented via its database administration facility.

A NearArchive database recovery utility is also available to physically recover primary copies of a database if required.

Additionally, up to 256 disk datasets may exist for each primary tape dataset in a database storage level. These are known as the 'K' copy datasets. These datasets will contain copies of one or more objects from the corresponding tape dataset. Disk copies of objects are created at the same time as tape copies of objects during batch archival or storage level migration processing, if 'K' copy processing has been enabled for the storage level in which the objects are being stored. These disk copies will exist in addition to the normal tape copies; this means that the primary copy of each object in a database will always be held on tape, irrespective of whether a disk copy of that object exists or not.

Batch and online object retrieval processing will always select the disk copy of an object for retrieval if one exists. If no disk copy exists, an object will be retrieved from tape as normal. If any error is encountered during retrieval of the disk copy of an object, ASM for S/390 will automatically switch to retrieval of the tape copy.

Disk copies of objects are retained for a customer-designated period of time after first storage in a database storage level. They will be deleted during the next execution of the object management utility after the designated disk retention period has been exceeded. Retrieval of objects subsequent to this time will then be satisfied from tape.

Disk ('K') copy datasets will be deleted under the following conditions:

- all associated disk copy datasets will be deleted by the database maintenance utility when the corresponding primary ('A') tape copy has been recycled.
- by the object management utility after all objects in a single disk ('K') copy dataset have been deleted, due to the retention period for disk copies of objects having been exceeded, or due to the overall object retention period for the database having been exceeded for all objects in the dataset. Appropriate selection of the maximum size for each disk copy dataset will thus inhibit over-utilization of disk space when using high-capacity tape volumes for storage of primary copy tape datasets.

Disk copy datasets may be re-created after deletion or after loss of the dataset, through use of the NearArchive database backup utility (OTASP110).

## **1.4 Application Program Interface.**

---

ASM for S/390 supplies an application program interface (API) to allow applications to perform the following functions:

- storage of objects in a NearArchive database
- retrieval of objects from a NearArchive database
- deletion of objects from a NearArchive database
- querying the contents of a database
- creation of secondary keys for an archived object
- commit updates to a NearArchive database

A full description of the new application program interface may be found in chapter 4 of this manual.



### **1.4.1 Batch processing.**

An API is supplied to enable batch programs to write objects to any storage level in a NearArchive database. The batch program will identify the archive object by primary key and archive date. One or more secondary keys for each object may additionally be identified automatically or explicitly. This information is used by ASM for S/390 to maintain primary and secondary indexes for the database. These indexes are used to enable rapid retrieval of data in response to a batch or online object retrieval request.

Creation and maintenance of all components of a NearArchive database is performed by the product itself. No direct user access is required to any NearArchive system dataset.

The interface allows database updates to be explicitly committed at any point. Updates may also be implicitly committed when a database is closed for output processing. A restart facility is provided in order to facilitate rerun or restart of batch archival procedures after job failure, without the requirement of any prior data recovery operation.

The NearArchive API can be invoked from a batch program to retrieve objects from a NearArchive database. Objects to be retrieved are identified by primary key and archive date.

Each logical record within an identified object is returned sequentially to the calling application in response to a program call. An indication is given by ASM for S/390 when all logical records for the specified object have been retrieved. Direct or sequential retrieval of objects may be performed using the NearArchive batch retrieval interface.

Access to all components of a NearArchive database, including dataset allocation and deallocation, and data retrieval are supplied by ASM for S/390. No direct user access is required to any NearArchive system dataset.

Batch retrieval in this manner may be used for producing printed reports on archived data, re-printing archived bills etc. This batch process might be requested from an online application by users wishing to recall multiple NearArchive objects for a specific primary key, and who did not wish to wait for synchronous (online) retrieval to display all the required details on the user's terminal. The asynchronous (batch) retrieval could run as a background batch task, printing the requested details on a local printer while the online user continues to use the online system.

Batch programs may also use the API to delete objects from a NearArchive database, and to query the primary or secondary indexes of a database.

Refer to chapter 5 of this manual for more information on batch processing of a NearArchive database.

### **1.4.2 CICS processing.**

ASM for S/390 enables CICS users to retrieve archived data from a NearArchive database. This is invoked from user application processing using the high-level interface provided by the 'EXEC CICS LINK PROGRAM' command. In this way, recall of NearArchive objects can be integrated into existing online database retrieval systems, extending virtually without limit the amount of data that can be accessed online.

An object requested for retrieval by the user application program from a NearArchive database will be written to a CICS temporary storage queue (TSQ) identified by the calling program. There will be one entry in the TSQ for each record in the retrieved object. Multiple TSQs will be used to hold objects which contain greater than 32,767 records. All functions involving access to components of the NearArchive database, including allocation and de-allocation of tape datasets from within CICS, and retrieval of the archived object, will be provided by ASM for S/390. After control has been returned to the calling program, the temporary storage queue may be handled as appropriate for the particular application requirements. CICS applications may also retrieve a segment of an archived object into a TSQ (termed 'partial object retrieval'). An object segment is identified by two co-ordinates: the start record number (starting at 0), and the number of records in the segment.

The NearArchive control region, which processes all object retrieval operations in a CICS processing environment, requires initialization via an entry in the CICS post-initialization PLT in order to enable archive recall. This will enable online applications running within the control region itself, or from within any other CICS region which has a Multiple Region Operation (MRO) link to the control region, to retrieve data from one or more NearArchive databases.

Multiple database support is available with ASM for S/390 online processing (i.e.) it is possible to access many different NearArchive databases from within the same CICS system. The name of the database to be accessed is identified by the application program on request for object retrieval.

Use of ASM for S/390 for online recall from tape requires the installation of an automated tape processing system for satisfying specific mounts. Using tape automation, it will be possible for online users to access any tape-resident NearArchive object. The response time will be dependent on the tape hardware in use and the location of data on the tape. No impact is made on the response times of other CICS users in the system while tape retrieval is in progress.

In addition, the average time taken to retrieve any NearArchive object from tape is not dependent on the size of the NearArchive database in which the object is stored (i.e.) the number of tape cartridge volumes in the database. This means that huge quantities of data may be accessible online with consistent and predictable retrieval response times.

CICS programs may assign a priority to an object retrieval request. This priority will be honored by ASM for S/390 when adding a request to a tape reader task queue.

CICS programs may also use the API to delete objects from a NearArchive database, and to query the primary or secondary indexes of a database.

Refer to chapter 6 of this manual for more information on processing of a NearArchive database by CICS programs.

## **1.5 Online Access to NearArchive Databases.**

---

ASM for S/390 enables users to access very large quantities of data via online transaction processing facilities controlled by CICS. Special processing techniques are used by ASM for S/390 to bypass limitations imposed by CICS File Control processing to enable tape datasets to be accessed from within CICS applications, without impacting general user response times.

This means that only those users who have made archival retrieval requests (which may involve tape dataset access) will wait for tape I/O to be completed. There is no impact on response times for other CICS users who may be active at the same time. In addition, retrieval requests which are stalled (due to some external wait condition, such as an outstanding tape mount request) may optionally be terminated by ASM for S/390 if the time interval specified in the TIMEOUT initialization parameter has expired.

Coding techniques used to implement tape support from within CICS use standard OS macros and CICS commands. This means that there is no exposure due to the use of unpublished or release-dependent interfaces.

Access to one or more NearArchive databases in a CICS environment is performed by a NearArchive 'control' region. This region will service requests from NearArchive applications running within that region, or from within other CICS regions in the same OS/390 host which have been connected to the control region using standard CICS Multiple Region Operation (MRO) facilities.

The NearArchive control region will allow shared access to multiple NearArchive databases from all CICS regions within a single OS/390 host. If required, multiple NearArchive control regions may be defined per host system. Implementation of MRO is not a pre-requisite for ASM for S/390 operation in a CICS environment. The limits on the number of tape drives which ASM for S/390 may allocate simultaneously for online retrieval, and on the number of requests which may be queued per allocated drive, are set by the NearArchive CICS operator. The product has an overall limit of 256 simultaneous CICS retrieval requests per control region.

### **1.5.1 Online Response Times.**

A request for online retrieval of a tape copy of an archived object will generate the following sequence of events:

- a) Dynamic allocation of tape dataset, which in turn will cause:
  - Mount request for tape cartridge volume
  - Loading of volume onto drive
  - Label processing
- b) Dataset open.
- c) Location of block(s) containing requested object.
- d) Retrieval of data block(s).
- e) Dataset close which generates tape rewind.
- f) Dynamic de-allocation of tape dataset, which in turn will cause:
  - Unloading of tape cartridge volume.
  - Dismount request for volume.

Of the above sequence of events, items (e) and (f) are performed asynchronously while retrieved data is being processed by the requesting program, and will not therefore have an impact on transaction response time.

However, items (a) - (d) will all contribute to the overall response time. Of these, items (b) - (d) will all be of finite predictable duration.

In a manual tape processing environment, item (a) is dependent on the length of time it takes an operator to respond to a mount request. This means that in addition to requiring constant operator attendance during CICS operation, the time taken to respond to a mount request, and hence the CICS response time for archive retrieval, cannot be guaranteed. This is clearly unacceptable in an online environment.

For this reason, online retrieval by ASM for S/390 should only be considered where an automated tape processing subsystem (such as Sun Oracle's 4400 Automatic Cartridge System - ACS) is in operation. If this is the case, and if ASM for S/390 mount requests are given the highest priority within the ACS subsystem, then guaranteed levels of service for tape mounts can be provided. This in turn will generate a guaranteed level of response for CICS transactions which use ASM for S/390 to retrieve archived data, and consequently be acceptable for implementation in an online production environment.

An additional facility is supplied for online retrievals to prevent tape files being automatically closed and de-allocated when all retrievals have been completed for that dataset. This is set from the NearArchive ISPF database administration facility. For each NearArchive database, a range of ages of data may be identified, so that tape cartridge volumes which contain NearArchive objects whose age falls within the specified range will not be automatically closed and de-allocated after all outstanding retrievals for that volume have been completed.

Tape cartridge volumes containing objects which fall into this category will thus remain open and mounted on the drive. Tapes will be positioned directly after the last block retrieved. Any subsequent retrieval request for data from that cartridge will merely require repositioning of the tape. This will eliminate items (a) and (b) from the above search sequence, and reduce online response times accordingly.

However, when all drives are busy or loaded, a retrieval request for a tape cartridge volume not currently mounted will require the dismount of a volume which has been left on the drive after completion of its last

retrieval request as described above. This will impose an additional delay on the time taken to retrieve data for the new request.

If automated tape processing is not available, recall should be limited to batch processing systems. Note however that the requests for recall may still be made from an online system, via the online submission of batch retrieval jobs. Output from these jobs may be sent to the JES2 spool for output on the main system printer or any remote JES2 printer, or may be sent to a CICS print management facility, if such a system is available. In this latter case, the output may then be viewed or printed by the online user via facilities provided by the CICS print management software; this corresponds to asynchronous (background) retrieval of archived data as opposed to synchronous (foreground) retrieval which is the case with the ASM for S/390 online retrieval facility.

The above discussion is not applicable when retrieving the disk copy of an archived object. All such objects may be retrieved without any specific requirement for support of automated tape processing. If required, disk copies may be kept of **all** objects in a NearArchive database. This will remove altogether the requirement for automated tape processing facilities when accessing objects in an online processing environment.

In order to optimize performance of data retrieval, ASM for S/390 allows users to adjust online tape retrieval operation to match the requirements of any individual application. Users should pay particular attention to the following areas in order to optimize online tape processing performance:

a) **The amount of data held per tape cartridge volume (MAXBLKCT database initialization parameter).**

In general, reducing the amount of data held per tape volume will reduce the average time taken to locate a NearArchive object, and consequently reduce the average response time for online data retrieval. However, the number of tape library storage cells required to contain all volumes of a NearArchive database will increase proportionately.

b) **The maximum number of tape drives to be allocated by NearArchive for its own use (MAXDRIVE control region parameter).**

Increasing this number will allow ASM for S/390 to improve throughput for online tape retrieval operations, and reduce the likelihood of an 'archiving busy' condition being raised in response to a retrieval request. The value of MAXDRIVE which is in effect may be varied as required via the NearArchive CICS operator interface.

c) **The maximum number of requests which may be queued for any active tape volume (MAXQLEN control region parameter).**

All ASM for S/390 requests will be satisfied by retrieving data from a specific tape cartridge volume. However, due to the nature of tape processing, multiple concurrent requests for retrieval from a tape cartridge volume will be processed serially by ASM for S/390.

A second request, which arrives while a volume is being processed by another user, will be queued within ASM for S/390. In this case the second request will be delayed while the first request is being processed, and will then be subject to another delay, while the tape block which contains data for the second request is located. (Note that the overall response time for the second request may in fact be less than that for the first request, depending on its time of arrival).

Specifying a high value for MAXQLEN will allow many requests for retrieval of data from a single tape volume to be processed serially. This will reduce the likelihood of an 'archiving busy' condition being raised by ASM for S/390. However, allowing longer queue lengths (assuming that the recall rate is sufficiently high to cause long queues) will mean an increase in average response times for retrieval of data from a NearArchive database.

In order to avoid extended CICS response times during periods of high utilization, it is advisable to limit the queue length, so that users may be informed that all retrieval resources are currently busy, and that they should resubmit their request at a later time.

d) **Tape cartridge volumes to be retained on drive after use.**

Volumes containing objects whose age falls within a range specified by the database administrator for any NearArchive database will be retained on the tape drive after use. Other volumes will be automatically dismounted after all outstanding requests for retrieval from those volumes have been processed.

In order to evaluate the benefits of this facility, patterns of access for data recall should be examined and identified. If a high proportion of retrievals may be satisfied by a relatively small number of tape cartridge volumes, then those volumes should be retained on a drive after all outstanding retrievals have been processed.

Note that if patterns of access fall into this category, then it is actually more beneficial to maximize the amount of data held per tape (contrary

to the situation discussed in paragraph (a) above).

However, if all drives are currently occupied with tape cartridges, retrieval of data from a non-mounted volume will require the mounted volume which was least recently referenced to be dismounted. As this dismount processing will need to be completed before the new volume may be mounted, the response time for retrieval of data for the new request will be increased. Too high an incidence of this condition will have an adverse effect on general tape data retrieval performance.

Users should guard against an inappropriate specification of tape cartridge volumes to be retained on drive causing excessive 'dismount prior to mount' activity. In extreme cases, this may lead to a very high dismount /mount rate (in a manner analogous to page 'thrashing' in a constrained virtual storage environment). This will cause excessive robot accessor utilization, which in turn will have an adverse effect on tape retrieval response times.

**e) Improved Cartridge Recording Capability (ICRC) options.**

This will allow selective use of the ICRC hardware compaction feature to increase the amount of data which may be held per tape cartridge volume, or conversely to reduce the amount of storage (and consequently the length of time required to locate any item of data) used per tape volume to hold a given amount of data.

**f) CICS temporary storage queue housekeeping.**

Temporary storage queues (TSQs) used to store recalled objects from a NearArchive database may be automatically deleted by ASM for S/390 through use of the 'TSQDELETE' control region parameter. When this parameter is set to a non-zero value of 'n' (in the range 1-1440), ASM for S/390 will automatically delete each TSQ used for storage of a recalled NearArchive object 'n' minutes after that object was last accessed.

Any request to access a NearArchive object which has been previously recalled will be automatically satisfied from the TSQ used to store that recalled object, until such time as the TSQ is deleted by the NearArchive TSQ housekeeping facility. This facility will improve performance for repeated access to individual objects.

**g) Dynamic load balancing.**

When dynamic load balancing is enabled for a database storage level,



ASM for S/390 will automatically retry a retrieval request using a designated backup copy volume when access to a primary volume has failed due to a 'resource unavailable' condition. Use of this facility will improve performance and throughput for batch and online retrieval requests, where backup database copies are available for automated processing.

**h) Internal request queuing.**

If a request for retrieval of an object from tape cannot be immediately processed due to the unavailability of one or more resources, ASM for S/390 can optionally queue the request internally until sufficient resources are available to process the request or until a user-specified time interval has elapsed, at which point it will be rejected with a 'resource unavailable' condition. This facility is controlled using the 'TAPEWAIT' initialization parameter in the OTASS100 dataset.

ASM for S/390 will optionally record information about object retrieval requests to System Management Facility (SMF) datasets. If enabled, one SMF record is written for each CICS and/or batch object retrieval request. This information may be analyzed in order to produce appropriate statistics related to object retrieval processing (e.g.) distribution of tape retrieval requests during the online day, patterns of user access to NearArchive databases and tape volumes, average/maximum response times etc. This information may be of significance in assisting with the optimization of NearArchive object retrieval performance using the product and implementation features discussed in this section.

## **1.6 NearArchive Utilities.**

---

To assist in the control and maintenance of NearArchive database processing, a number of batch utilities are supplied with the product. Each of these utilities is intended to perform a specific function to ensure that data from a NearArchive database is always available for processing, in the most efficient manner possible.

The supplied utilities consist of:

- **OTASP010** – the index print utility.

The index print utility OTASP010 is a diagnostic aid which is intended for use when investigating problems associated with accessing objects from a NearArchive database. The utility will sequentially scan all or part of a NearArchive index, and produce a print report containing formatted details of all objects held in the scanned index.

- **OTASP020** – the tape scan utility.

Like OTASP010, the tape scan utility OTASP020 is also a diagnostic aid which is intended for use when investigating problems associated with accessing objects from a NearArchive database. The utility will sequentially scan a single tape dataset from within a NearArchive database, and produce a print report containing formatted details of all objects in the tape dataset.

- **OTASP030 - the database maintenance utility.**

Objects may be deleted from a NearArchive database via explicit delete calls from an application, or via automatic expiration processing. Tape space within the database which is occupied by deleted objects will then become redundant. Disk copies of objects may also be expired separately – space within a disk ('K') copy dataset occupied by these objects will also then become redundant. The database maintenance utility is used to reclaim redundant tape and disk space from within the database.

The utility is responsible for identifying all tape cartridge volumes within a NearArchive database which contain less than a user-supplied threshold of active objects. All such volumes are recycled by moving active objects from those volumes to a different location in the database. The recycled volumes are then released from the database and may be returned to the appropriate scratch tape pool. All existing disk ('K') copy datasets associated with a released volume will be automatically deleted by the utility.

The utility can also be used to identify disk copy datasets which contain less than a user-supplied threshold of active disk-resident objects. These datasets are compressed to remove unused space.

- **OTASP040 - the object management utility.**

The object management utility is responsible for maintaining the contents of a NearArchive database.

The utility will automatically delete all objects which have been resident in a database for longer than the retention period for that database (as specified by the NearArchive database administrator).

The utility will also migrate objects between storage levels, for those databases which have multiple storage levels. Control of the object migration process is also specified by the NearArchive database administrator via the supplied database administration TSO/ISPF

facility.

The object management utility also deletes the disk copy of an object for which the disk retention period has been exceeded.

- **OTASP070 - the journal maintenance utility.**

The journal maintenance utility allows analysis of journal entries for databases in which the audit trail facility has been enabled. It also optionally archives journal entries to the database's journal archive database. The utility can produce historical analyses of archived journal entries.

The contents of the journal dataset are cleared after successful execution of the journal maintenance utility.

- **OTASP080 – the secondary index update utility.**

The secondary index update utility is executed as the second phase in the two-phase asynchronous secondary index update processing. The first phase is executed during batch archival processing.

Asynchronous secondary index update processing is intended to reduce the amount of I/O activity performed during execution of a NearArchive batch archival job (thus improving job turnaround time). All asynchronous updates are then performed in the most efficient manner by the utility, thus minimizing the overall amount of I/O activity required to process secondary index updates.

- **OTASP110 - the database backup utility.**

The database backup utility is used to automatically create one or two full-volume backup copies of each tape dataset which has been updated during a batch archival process. The utility may alternatively be used to create an incremental backup dataset for each primary tape volume. The incremental backup dataset will contain all objects which have been added to the primary tape volume since the previous execution of the utility. This utility may also be used to create one or more disk ('K') copy datasets from a primary tape volume. All objects on the primary tape volume which are eligible for disk residency will be copied to disk.

- **OTASP130 - the database recovery utility.**

The database recovery utility is used to recreate one or more primary copy tape datasets in a NearArchive database from designated backup

copies. A combination of full-volume and incremental backup datasets may be used when reconstructing a primary volume dataset.

Refer to chapter 7 of this manual for a detailed description of the above utility programs.

---

## CHAPTER 2

---

# *Installation and Implementation Procedures*

This page is intentionally left blank

This chapter describes the activities required to install and implement ASM for S/390 for system use. On completion of these procedures, NearArchive databases may be defined (see chapter 3 for details of this procedure), and data may be archived and retrieved to and from these databases (see chapters 4 to 6 for details of database access from application programs).

The following checklists provide a list of activities which should be performed during initial ASM for S/390 installation and implementation or when upgrading from an earlier release of NearArchive. Use the appropriate checklist to control the implementation process. A full description of each activity is provided in the sections indicated in brackets.

#### **A) Initial installation of ASM for S/390**

- ☐ 1. Install distribution libraries with SMP/E (2.1).
- ☐ 2. Install user SVC (2.2.1).
- ☐ 3. Update PLPA (2.2.2).
- ☐ 4. Update NearArchive authorization code(s) (2.3).
- ☐ 5. Perform TSO logon procedure changes (2.4).
- ☐ 6. Perform ISPF menu changes (2.5).
- ☐ 7. Define CICS primary and secondary region table entries (2.8.4 & 2.8.5).
- ☐ 8. Perform CICS primary and secondary region JCL changes (2.8.6).
- ☐ 9. Install security user exit modules, if required (2.9)
- ☐ 10. Run installation verification procedures (2.10).

#### **B) Upgrading from an earlier release of NearArchive**

When migrating from an earlier release, carefully check the installation procedures to ensure that any changes to an earlier procedure are not overlooked. In particular, review the CICS table entry definitions specified in this chapter to ensure that any new entries required with the new release(s) have been correctly defined.

- \_\_\_ 1. Install distribution libraries with SMP/E (2.1).
- \_\_\_ 2. Install user SVC (2.2.1).
- \_\_\_ 3. Update PLPA (2.2.2).
- \_\_\_ 4. Update NearArchive authorization code(s) (2.3).
- \_\_\_ 5. Update TSO logon procedure changes (2.4).
- \_\_\_ 6. Update CICS primary and secondary region table entries (2.8.4 & 2.8.5).
- \_\_\_ 7. Update CICS primary and secondary region JCL (2.8.6).
- \_\_\_ 8. Install security user exit modules, if required (2.9)
- \_\_\_ 9. Migrate pre-v2.5 databases (2.10).
- \_\_\_ 10. Run installation verification procedures (2.11).

## 2.1 Install distribution libraries.

---

ASM for S/390 v2.5 is distributed on a standard-labeled magnetic tape cartridge in SMP/E RELFILE format. The cartridge has a serial number of OTS250.

The distribution tape will contain the following datasets:

File	Dataset name	Contents
1	SMPMCS	SMP/E modification control statements for installation of the product.
2	STK.SOTS250.F1	SMP/E JCLIN file.
3	STK.SOTS250.F2	Partitioned dataset in IEBCOPY UNLOAD format containing JCL for use during product installation.
4	STK.SOTS250.F3	ASM for S/390 distribution load library.
5	STK.SOTS250.F4	ASM for S/390 ISPF panel



File	Dataset name	Contents
		library.
6	STK.SOTS250.F5	ASM for S/390 ISPF message library.
7	STK.SOTS250.F6	ASM for S/390 ISPF skeleton JCL library.
8	STK.SOTS250.F7	ASM for S/390 sample source code library.
9	STK.SOTS250.F8	ASM for S/390 IVP load library.
10	STK.SOTS250.F9	ASM for S/390 IVP JCL library.
11	STK.SOTS250.F10	ASM for S/390 IVP test data.

To install ASM for S/390, perform the following steps:

1. Copy the installation JCL (file 3) from the distribution tape to disk. The following JCL may be used for this purpose:

```
//S1      EXEC PGM=IEBCOPY
//SYSPRINT DD SYSOUT=*
//SYSIN   DD DUMMY
//SYSUT1  DD DISP=(OLD,KEEP),UNIT=unit-name,
//          VOL=(,RETAIN,SER=OTS250),
//          LABEL=(3,SL,EXPDT=98000),
//          DSN=STK.SOTS250.F2
//SYSUT2  DD DISP=(NEW,CATLG),
//          DSN=NearArchive.install.JCL.library,
//          SPACE=(TRK,(5,5,10)),UNIT=unit-name,
//          VOL=SER=volser
```

On completion of the above step a partitioned dataset containing JCL to be used during the remainder of the installation process will have been created.

2. Edit member SMPPEDEF from the installation JCL dataset unloaded in the preceding step, as described within the dataset member itself. This job will define and initialize the SMP/E consolidated software inventory (CSI) dataset and create other SMP/E system

datasets and the ASM for S/390 target and distribution libraries. If any of these datasets are already in existence, delete them prior to running this job or amend the job to remove dataset creation or initialization processing, as required.

After editing the member, submit it. Verify that all job steps complete with condition code 0, and that all datasets are created and cataloged successfully.

3. Edit JCL member SMPERCV, as described within the member itself. This job performs SMP/E RECEIVE processing for the ASM for S/390 V2.5 function SYSMOD.

After editing the member, submit it. Verify that SMP/E processing has completed with condition code 0.

4. Edit JCL member SMPEAPP, as described within the member itself. This job performs SMP/E APPLY processing for the ASM for S/390 V2.5 function SYSMOD.

After editing the member, submit it. Verify that SMP/E processing has completed with condition code 0.

5. Optionally edit and submit JCL member SMPEACC. This job performs SMP/E ACCEPT processing for the ASM for S/390 V2.5 function SYSMOD. Processing of this job may be deferred until later. However, no service should be applied to the product until SMP/E ACCEPT processing has been performed successfully.

The ASM for S/390 product distribution libraries will have been successfully created on completion of the above steps. Continue with product installation processing at section 2.2.

## **2.2 Host system modification.**

---

This section gives details of the modifications required to the host OS/390 system, prior to ASM for S/390 implementation.

### **2.2.1 Install user SVC.**

ASM for S/390 requires installation of a type-3 or type-4 user SVC. This module is distributed as member 'OTASSVC' in the distributed NearArchive load library. To install the SVC, perform the following steps:

- a) Allocate a type-3 or type-4 SVC number from the available user SVCs on the system. This will be in the range 200-255. If there is no available SVC in this category, refer to IBM system documentation for details of how to define a new user SVC in the SVC table. Use default values for the APF and NPRMPT parameters in the SVC definition entry. The SVC module name ('OTASSVC') should be identified in the EPNAME parameter on the SVC Parm entry.
- b) Using IEBCOPY, or ISPF/PDF, copy member OTASSVC from the distributed NearArchive load library to 'SYS1.LPALIB', or any other library in the LPA list.
- c) The allocated SVC number should be used during definition of any new NearArchive database via the NearArchive ISPF database maintenance facility (see chapter 3 of this manual).

Note that a re-IPL with the CLPA option will be required to activate the new user SVC processing.

### **2.2.2 Update Pageable Link Pack Area.**

In order to prevent general access to the user SVC installed in the previous section, program OTASP055 from the NearArchive product library MUST be installed in the OS/390 pageable link pack area (PLPA). To do this, move member OTASP055 from the unloaded NearArchive distribution library to a library in the LPA list and re-IPL the system with the CLPA option.

Note that it is essential that OTASP055 be moved from the NearArchive product library and not merely copied (i.e.) OTASP055 must be removed from the NearArchive product library. Failure to observe this requirement will cause ASM for S/390 to fail with a system 047 abend. Also, note that installation of OTASP055 in the MLPA rather than the PLPA will cause ASM for S/390 to fail with an S047 abend.

## **2.3 Update NearArchive authorization code(s).**

Before using ASM for S/390, the product must be authorized to run on the customer's processor(s). One or more 8-byte authorization codes are supplied with the product in the accompanying documentation. Any update or refresh of these codes may be obtained by contacting your ASM for S/390 product support representative.

As ASM for S/390 is distributed on a processor license basis, one authorization code is supplied for each processor on a customer site which requires access to the NearArchive product library.

To authorize the product, IBM utility program AMASPZAP (Superzap) must be run to update the authorization module OTASP050 in the distributed load library.

Use the following parameters:

NAME	OTASP050	AUTHCODE
REP	nnnn	xxxxxxxx,yyyyyyyy

where nn = 0000 for authorization code 1  
0008 " " " 2  
0010 " " " 3  
0018 " " " 4 etc.

and 'xxxxxxxx,yyyyyyyy' is the supplied authorization code(s).

ASM for S/390 will accept a maximum of 10 authorization codes.

*NOTE: ASM for S/390 is authorized for execution on designated processors only. Any planned change or upgrade to a customer's host processor(s) should be communicated in advance to your ASM for S/390 product support representative, so that new/additional authorization codes may be supplied.*

## 2.4 Perform TSO logon procedure changes.

---

In order to enable NearArchive database administrators to define and maintain NearArchive databases via TSO/ISPF (see chapter 3 for details of this facility), the following changes should be made to the appropriate TSO logon procedure(s):

- a) Add the product load library to the ISPF load library concatenation (file ISpload or ISPLLIB), as follows:

//ISPLLIB	DD	.....
//	DD	DSN=OTAS.VvRr.LOADLIB,DISP=SHR

where 'OTAS.VvRr.LOADLIB' is the name given to the unloaded NearArchive load library during the product installation process (see section 2.1). Alternatively, this library may be added to the system linklist concatenation.

b) Add the product panel library to the ISPF panel library concatenation (file ISPPLIB), as follows:

```
//ISPPLIB DD .....  
//          DD DSN=OTAS.VvRr.ISPPLIB,DISP=SHR
```

where 'OTAS.VvRr.ISPPLIB' is the name given to the NearArchive ISPF panel library during the product installation process (see section 2.1).

c) Add the product message library to the ISPF message library concatenation (file ISPMLIB), as follows:

```
//ISPMLIB DD .....  
//          DD DSN=OTAS.VvRr.ISPMLIB,DISP=SHR
```

where 'OTAS.VvRr.ISPMLIB' is the name given to the NearArchive ISPF message library during the product installation process (see section 2.1).

d) Add the product skeleton JCL library to the ISPF skeleton library concatenation (file ISPSLIB), as follows:

```
//ISPSLIB DD .....  
//          DD DSN=OTAS.VvRr.ISPSLIB,DISP=SHR
```

where 'OTAS.VvRr.ISPSLIB' is the name given to the NearArchive ISPF skeleton JCL library during the product installation process (see section 2.1).

e) No additional amendments are required to the logon procedure(s) used by the TSO user(s) who will be responsible for performing NearArchive database administration functions.

If required, ISPF 'LIBDEF' processing may be used to dynamically add NearArchive libraries to the required ISPF library allocations on invocation of the NearArchive database administration function. However, the NearArchive product load library **must** be defined in the ISPLOAD or ISPLLIB concatenation as described above, as the LIBDEF facility does not support dynamic loading of program modules as performed by the NearArchive database administration process.

## 2.5 Perform ISPF menu changes.

---

In order to allow access to the NearArchive database administration dialogs in TSO/ISPF, an option should be added to the ISPF menu from which entry is to be granted.

The )PROC section of the panel definition for this menu should be amended, as shown in the example below:

```
&ZSEL = TRANS( TRUNC (&ZCMD, '.' )  
               /*****  
               /* Existing applications here          */  
               /*****  
               NA, 'PGM(OTASP400)' )
```

where 'NA' is the option chosen here for invoking the NearArchive database administration panels.

## 2.6 NearArchive database definition.

---

Before accessing a NearArchive database for the first time (by creating archived data), it is necessary to create and initialize the database. A NearArchive database can consist of the following components:

- Tape datasets. Data is stored in a NearArchive database in a series of single-volume sequential tape datasets. A NearArchive database may consist of up to eight storage levels, each storage level containing up to 65,535 individual tape datasets (i.e.) a maximum of 524,280 tape cartridge volumes per database.
- Sequential disk datasets. Up to 256 datasets may exist for each primary tape dataset, if disk ('K') copy processing has been enabled during object archival. Each dataset will contain a copy of one or more objects from the corresponding primary tape dataset. Disk datasets are automatically deleted by the NearArchive object management or database maintenance utilities when no longer required.
- A primary index. This is a VSAM KSDS, which contains one variable-length record for each unique primary key value in the database.

- One or more secondary indexes. Each secondary index is a VSAM KSDS. Up to 99 secondary indexes can be defined per NearArchive database.
- From one to eight space management datasets. The space management dataset is a VSAM KSDS containing information about tape volume utilization for an individual storage level within a NearArchive database. A maximum of eight space management datasets will be required per database. Each space management dataset will contain one variable-length record per tape cartridge volume in the corresponding storage level in the database.
- One database journal dataset. This is a VSAM ESDS which is used to log all updates to a NearArchive database if the audit trail facility is enabled for that database. The journal dataset is created when the audit trail facility is first enabled.

The minimum dataset configuration required for a version 2 NearArchive database consists of the following:

- One or more single-volume tape datasets.
- A primary index dataset.
- One space management dataset.

All the above VSAM datasets are created and initialized automatically via the NearArchive database administration facility in TSO/ISPF. Tape and sequential disk datasets will be created automatically by ASM for S/390 during the batch archival process.

A full description of the NearArchive TSO/ISPF database administration facility, and its use in defining and maintaining a NearArchive database, is given in chapter 3 of this manual.

## **2.7 Batch implementation.**

---

No special procedures are required for implementation of ASM for S/390 for batch archival and retrieval. The only requirement is that the NearArchive request stub 'OTASP100' is linked into each batch program that requires access to the database, in order to satisfy external references generated by the high-level CALL interface used for ASM for S/390 requests.

In addition, the NearArchive load library created in section 2.1 needs to be specified in the STEPLIB or JOBLIB concatenation for batch archival or retrieval jobs, or else to be present in the system linklist concatenation.

Programs link-edited with a prior version of OTASP100 do not in general require re-linking in order to run with the later release of ASM for S/390. However, modification of an existing ASM for S/390 application in order to implement some of the new API facilities shipped with V2.5 of the product will require the version of OTASP100 distributed with this release to be linked into the calling program. This will normally be done automatically after successful compilation of the modified application program.

## **2.8 Online implementation.**

---

ASM for S/390 runs under versions of CICS from 4.1 upwards. This section details all the CICS table definitions and other considerations that are required for ASM for S/390 operation.

### **2.8.1 NearArchive control region.**

ASM for S/390 uses a single CICS 'control region' to perform all object retrieval operations (i.e. allocation and processing of NearArchive tape or sequential disk datasets), as described in chapter 1 of this manual.

The internal limit to the number of tape drives that each NearArchive control region may allocate to satisfy simultaneous tape dataset retrieval operations is 256 - however, ASM for S/390 will obviously not be able to allocate more drives than are online to the host system.

User controls are provided (via the MAXDRIVE system initialization parameter) to enable users to limit the maximum number of drives which a NearArchive control region may allocate simultaneously.

An additional parameter (MAXDISK) is provided to set the number of disk reader tasks initiated in the control region for retrieval of disk-resident objects. This parameter is used to control performance of disk retrieval processing.

Additional controls are also provided (via the MAXQLEN system initialization parameter) to enable users to impose a limit on the number of requests which may be queued for an active disk or tape dataset. The maximum value for this limit is also 256.



ASM for S/390 will return an 'archiving busy' condition (return code 21) to any request which causes any of the above limits to be exceeded.

ASM for S/390 also allows temporary storage queues created during object retrieval processing to be automatically deleted after a customer-specified time interval has elapsed since object retrieval was completed. This facility is controlled via the TSQDELETE system initialization parameter.

Prior to use, the NearArchive control region must be initialized for online retrieval. This may be performed via PLTPI processing, or by running transaction OT23 at any time subsequent to CICS initialization (see chapter 6 of this manual for details of this transaction).

Similarly, the NearArchive control region may be quiesced for online retrieval via PLTSD processing, or via transaction OT24. No NearArchive database retrieval operations will be allowed subsequent to quiescing the NearArchive control region.

### **2.8.2 Secondary NearArchive regions.**

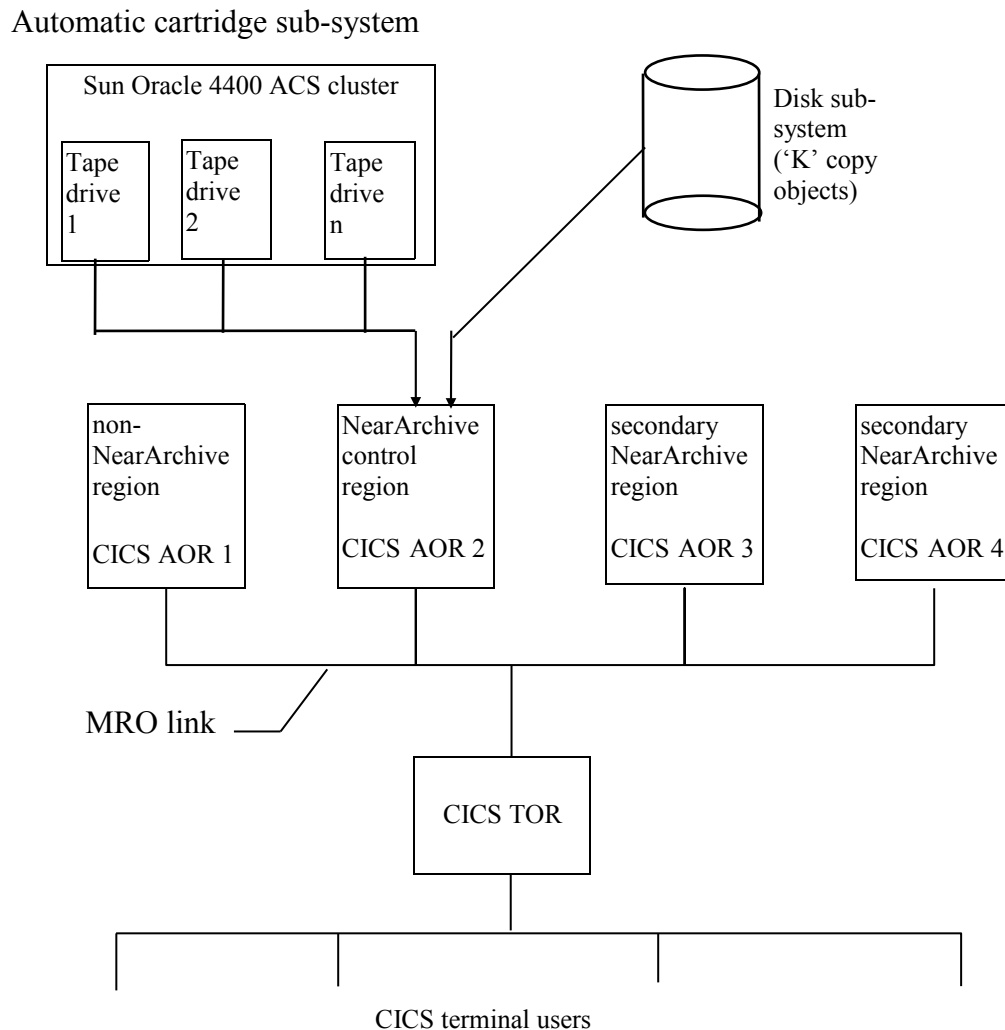
The NearArchive control region will process all ASM for S/390 retrieval requests (issued from application programs via the supplied online retrieval interface) from within the control region itself, and from any other CICS region which has a Multiple Region Operation) cross-memory or IRC link with the control region. These other regions in the MRO complex are known as 'secondary' NearArchive regions.

This means that a single NearArchive control region will be able to service NearArchive online retrieval requests from multiple CICS regions in an MRO complex. Note that all tape processing in this type of configuration will still be performed by the NearArchive control region. The limits to the maximum number of simultaneous requests which may be processed are those which are currently active in the NearArchive control region.

Each region in a CICS MRO configuration may be connected to only one initialized NearArchive control region. Any attempt to initialize a NearArchive control region which is connected via MRO to an application region which in turn is connected to another initialized NearArchive control region, will fail with message OTS23002. However, any number of secondary regions (from zero upwards) may be present in the MRO complex. Secondary NearArchive regions do not require any initialization prior to issuing ASM for S/390 online retrieval requests from these regions.

Note that if a host system contains more than one CICS MRO configuration (i.e. more than one group of CICS regions which are connected to each other via MRO), multiple NearArchive control regions may exist within that host system, each capable of performing object retrieval operations. However, as stated above, each single CICS MRO configuration may only contain one NearArchive control region.

Figure 2.1 illustrates a typical CICS MRO configuration, consisting of a single terminal-owning region (TOR) and four application-owning regions (AORs). One of the AORs is initialized as a NearArchive control region, and will perform all tape access for ASM for S/390 applications running in the control region itself, and in any of the attached secondary regions.



**Fig 2.1:** Sample CICS configuration for ASM for S/390

### **2.8.3 Multiple NearArchive Database Access.**

For online retrieval, it is possible to perform retrievals from multiple NearArchive databases from within the same task (i.e.) to retrieve objects from NearArchive databases which have been archived separately, and have separate indexes. The database identifier is specified by the user in the call interface parameter area, used when invoking ASM for S/390 from an application program. Multiple database access is discussed in more detail in chapter 6 of this manual.

A primary index file and space management file must be defined in the control region file control table (FCT) for each database to be accessed from a CICS system. The FCT entry names will have the format 'OTASxxxx' and 'OSM0xxxx' respectively, where 'xxxx' is the 4-character database name used by calling applications to identify the database from which retrieval is to be performed.

Additional control region FCT entries may be required for secondary indexes, storage level 1-7 space management datasets, and database journal dataset, as required.

Note that no FCT entries are required for tape or sequential disk datasets accessed by ASM for S/390 during object retrieval processing.

### **2.8.4 CICS Definitions for a NearArchive Control Region.**

The following entries will be required in a CICS region that is to be initialized as a NearArchive control region. Entries are supplied here using RDO (Resource Definition Online) formats. The corresponding CICS table definition macros may be used as required.

#### **Files (RDO)**

One group of entries per database is required. Each entry name has the format 'Oyyyxxxx' where 'yyy' is the file-type identifier, and 'xxxx' is a unique 4-character string which is used to identify the database.

Each group of file entries includes required and optional entries, as follows:

## Required entries

### f) OTASxxxx – the primary index entry.

```
DEFINE    File (OTASxxxx)
          DSName (primary.index.dataset.name)
          Lsrpoolid (m)
          STRings (nnn)
          Opentime (FIRSTREF)
          DIsposition (SHARE)
          RECORDFormat (V)
          Add (NO)
          BRowse (YES)
          DElete (NO)
          REAd (YES)
          Update (YES|NO)
          JOurnal (NO|1)
          JNLRead (NONE|UPDATEONLY)
          RECOVery (NONE|ALL)
          Fwdrecovlog (NO|1-99)
```

Update access to this dataset is only required if database objects are to be deleted from online processing applications. In this case, the 'JOurnal', 'JNLRead', 'RECOVery' and 'Fwdrecovlog' parameters should be set as required.

### g) OSM0xxxx – the storage level 0 space management dataset

```
DEFINE    File (OSM0xxxx)
          DSName ('hlq'.SPACEML0)
          Lsrpoolid (m)
          STRings (nnn)
          Opentime (FIRSTREF)
          DIsposition (SHARE)
          RECORDFormat (V)
          Add (NO)
          BRowse (YES)
          DElete (NO)
          REAd (YES)
          Update (NO)
```

where 'hlq' is the high-level qualifier defined for this database during the database definition and initialization procedure (see chapter 3). Update access to this dataset is not required.

### Optional entries

- a) OTnnxxxx – a secondary index entry.

One secondary index entry will be required for each secondary index that has been defined in a NearArchive database. Each entry should have the following format:

```
DEFINE      File (OTnnxxxx)
            DSName ('hlq'.SECIXnn)
            Lsrpoolid(m)
            STRings(nnn)
            Opentime(FIRSTREF)
            DIsposition(SHARE)
            RECORDFormat(V)
            Add(NO)
            BRowse(YES)
            DElete(NO)
            REAd(YES)
            Update(NO)
```

where 'hlq' is the high-level qualifier defined for this database during the database definition and initialization procedure, and 'nn' is the secondary index identifier (in the range 01-99). Update access to this dataset is not required.

- b) OSMnxxxx – the storage level n space management dataset.

These entries will only be required if additional storage levels have been defined in the NearArchive database.

```
DEFINE      File (OSMnxxxx)
            DSName ('hlq'.SPACEMLn)
            Lsrpoolid(m)
            STRings(nnn)
            Opentime(FIRSTREF)
            DIsposition(SHARE)
            RECORDFormat(V)
            Add(NO)
            BRowse(YES)
            DElete(NO)
            REAd(YES)
            Update(NO)
```

where 'hlq' is the high-level qualifier defined for this database during the database definition and initialization procedure, and 'n' is the storage level identifier (in the range 1-7). Update access to this dataset is not required.

c) OJRNxxxx – the journal dataset.

The journal dataset entry will only be required if the audit trail facility has been enabled for a NearArchive database.

```
DEFINE    File (OJRNxxxx)
          DSName ('hlq'.JOURNAL)
          Lsrpoolid (m)
          STRings (nnn)
          Opentime (FIRSTREF)
          DISposition (SHARE)
          RECORDFormat (F)
          Add (NO|YES)
          BRowse (YES)
          DElete (NO)
          REAd (YES)
          Update (NO)
          JOurnal (NO)
          RECOVery (NONE|ALL)
          Fwdrecovlog (NO|1-99)
```

where 'hlq' is the high-level qualifier defined for this database during the database definition and initialization procedure. Update access to this dataset is only required if database objects are to be deleted from online processing applications. In this case the 'Add' parameter should be set to 'Yes', and the 'RECOVery' and 'Fwdrecovlog' parameters set as required.

### **Programs (RDO)**

```
DEFINE    PROGRAM (OTASP050)
          LANGUAGE (ASSEMBLER)
          DATALOCATION (ANY)
          EXECKEY (CICS)

DEFINE    PROGRAM (OTASP200)
          LANGUAGE (ASSEMBLER)
          DATALOCATION (ANY)
          EXECKEY (CICS)

DEFINE    PROGRAM (OTASP205)
          LANGUAGE (ASSEMBLER)
          DATALOCATION (ANY)
          EXECKEY (CICS)

DEFINE    PROGRAM (OTASP210)
          LANGUAGE (ASSEMBLER)
          DATALOCATION (ANY)
          EXECKEY (CICS)
```

DEFINE	PROGRAM (OTASP215) LANGUAGE (ASSEMBLER) DATALOCATION (ANY) EXECKEY (CICS)
DEFINE	PROGRAM (OTASP220) LANGUAGE (ASSEMBLER) DATALOCATION (ANY) EXECKEY (CICS)
DEFINE	PROGRAM (OTASP230) LANGUAGE (ASSEMBLER) DATALOCATION (ANY) EXECKEY (CICS)
DEFINE	PROGRAM (OTASP235) LANGUAGE (ASSEMBLER) DATALOCATION (ANY) EXECKEY (CICS)
DEFINE	PROGRAM (OTASP240) LANGUAGE (ASSEMBLER) DATALOCATION (ANY) EXECKEY (CICS)
DEFINE	PROGRAM (OTASP260) LANGUAGE (ASSEMBLER) DATALOCATION (ANY) EXECKEY (CICS)
DEFINE	PROGRAM (OTASP261) LANGUAGE (ASSEMBLER) DATALOCATION (ANY) EXECKEY (CICS)
DEFINE	PROGRAM (OTASP262) LANGUAGE (ASSEMBLER) DATALOCATION (ANY) EXECKEY (CICS)
DEFINE	PROGRAM (OTASP263) LANGUAGE (ASSEMBLER) DATALOCATION (ANY) EXECKEY (CICS)
DEFINE	PROGRAM (OTASP264) LANGUAGE (ASSEMBLER) DATALOCATION (ANY) EXECKEY (CICS)
DEFINE	PROGRAM (OTASP270) LANGUAGE (ASSEMBLER) DATALOCATION (ANY) EXECKEY (CICS)

```

DEFINE      PROGRAM (OTASUX02)
            LANGUAGE (ASSEMBLER)
            DATALOCATION (ANY)
            EXECKEY (CICS)

```

If the supplied CICS user exit module OTASUX02 is being replaced with a customer-generated user exit module, then the LANGUAGE parameter in the definition for program OTASUX02 should be modified to correspond to the language used for creation of the customer-generated module.

## Mapsets (RDO)

```

DEFINE      MAPSET (OT26M00)

```

## Transactions (RDO)

```

DEFINE      TRANSACTION (OT22)
            PROGRAM (OTASP220)
            DTIMOUT (NO)
            TASKDATALOC (ANY)
            TASKDATAKEY (CICS)

```

```

DEFINE      TRANSACTION (OT23)
            PROGRAM (OTASP230)
            TASKDATALOC (ANY)
            TASKDATAKEY (CICS)

```

```

DEFINE      TRANSACTION (OT24)
            PROGRAM (OTASP240)
            TASKDATALOC (ANY)
            TASKDATAKEY (CICS)

```

```

DEFINE      TRANSACTION (OT26)
            PROGRAM (OTASP260)
            TASKDATALOC (ANY)
            TASKDATAKEY (CICS)

```

```

DEFINE      TRANSACTION (OT70)
            PROGRAM (OTASP270)
            TASKDATALOC (ANY)
            TASKDATAKEY (CICS)

```

```

DEFINE      TRANSACTION (O261)
            PROGRAM (OTASP261)
            TASKDATALOC (ANY)
            TASKDATAKEY (CICS)

```



```

DEFINE      TRANSACTION (O262)
            PROGRAM (OTASP262)
            TASKDATALOC (ANY)
            TASKDATAKEY (CICS)

```

```

DEFINE      TRANSACTION (O263)
            PROGRAM (OTASP263)
            TASKDATALOC (ANY)
            TASKDATAKEY (CICS)

```

Note the presence of the DTIMOUT parameter on the entry for OT22. This is required to prevent abnormal termination of a retrieval task by CICS, while the task is waiting for access to tape data.

The DTIMOUT parameter should also be coded as above for all customer-developed transactions which issue calls to ASM for S/390 for retrieval of data from a NearArchive database.

## LSR pools (RDO)

If defining a VSAM local shared resource pool (LSRpool) via RDO, ensure that the maximum keylength specified in the LSRpool definition ('Maxkeylength' field) is set to a value which is not less than:

1. (length of the longest NearArchive primary key) + 2
- and
2. the highest possible value for the combined length of the primary key + the longest secondary key in any NearArchive database for which one or more secondary indexes have been defined.

## Program List Table - post-initialization (PLTPI)

OTASP235 DFHPLT TYPE=ENTRY, PROGRAM=OTASP235
---

This entry should be made in the third phase of PLTPI processing (i.e.) after the DFHDELIM entry.

## Program List Table - shut-down (PLTSD)

OTASP240 DFHPLT TYPE=ENTRY, PROGRAM=OTASP240
---

This entry should be made in the first phase of PLTSD processing (i.e.) before the DFHDELIM entry.

## Destination Control Table (DCT)

The system transient data queue 'CSMT' must be present in the DCT to receive ASM for S/390 information messages.

### **2.8.5 CICS definitions for a secondary NearArchive region.**

The following entries will be required in each CICS region that is to be initialized as a secondary NearArchive region (i.e.) which contain the ASM for S/390 online interface processing only. Note that for v2.5, an entry is required for the primary index of each NearArchive database to be accessed from a secondary NearArchive region. This was not the case for earlier releases of the product.

#### **Files (RDO)**

One entry per database is required, as follows:

OTASxxxx – the primary index entry.

```
DEFINE          File (OTASxxxx)
                DSName (primary.index.dataset.name)
                Lsrpoolid (m)
                STRings (nnn)
                Opentime (FIRSTREF)
                DIsposition (SHARE)
                RECORDFormat (V)
                Add (NO)
                BRowse (YES)
                DElete (NO)
                REAd (YES)
                Update (NO)
```

Update access to this dataset from a secondary NearArchive region is not required.

#### **Programs (RDO)**

```
DEFINE          PROGRAM (OTASP200)
                LANGUAGE (ASSEMBLER)
                DATALOCATION (ANY)
                EXECKEY (CICS)

DEFINE          PROGRAM (OTASP205)
                LANGUAGE (ASSEMBLER)
                DATALOCATION (ANY)
                EXECKEY (CICS)
```

```

DEFINE      PROGRAM (OTASP210)
            LANGUAGE (ASSEMBLER)
            DATALOCATION (ANY)
            EXECKEY (CICS)

DEFINE      PROGRAM (OTASP215)
            LANGUAGE (ASSEMBLER)
            DATALOCATION (ANY)
            EXECKEY (CICS)

DEFINE      PROGRAM (OTASP270)
            LANGUAGE (ASSEMBLER)
            DATALOCATION (ANY)
            EXECKEY (CICS)

DEFINE      PROGRAM (OTASUX02)
            LANGUAGE (ASSEMBLER)
            DATALOCATION (ANY)
            EXECKEY (CICS)

```

If the supplied CICS user exit module OTASUX02 is being replaced with a customer-generated user exit module, then the LANGUAGE parameter in the definition for program OTASUX02 should be modified to correspond to the language used for creation of the customer-generated module.

## **Transactions (RDO)**

```

DEFINE      TRANSACTION (OT70)
            PROGRAM (OTASP270)
            TASKDATALOC (ANY)
            TASKDATAKEY (USER)

```

No other CICS entries are required for a secondary NearArchive region.

## **2.8.6 CICS JCL changes.**

### **2.8.6.1 NearArchive control region.**

The following CICS JCL changes are required for the CICS job which will be configured as the NearArchive control region:

- a) Add the distributed ASM for S/390 load library to the CICS DFHRPL concatenation.
- b) Add the distributed ASM for S/390 load library to either the STEPLIB concatenation for the DFHSIP step, to the CICS JOBLIB concatenation, or to the system linklist concatenation.

Note that if the ASM for S/390 load library is added to the STEPLIB concatenation, it must be APF-authorized to preserve authorization during CICS initialization processing.

However, authorization of the ASM for S/390 load library is not required for ASM for S/390 operation itself. If you do not wish to authorize the ASM for S/390 library, it should be added as an unauthorized library to the linklist concatenation.

- c) Add the following DD card to the CICS DFHSIP job-step:

```
//OTASS100 DD DSN=sequential.dataset.name,DISP=SHR
```

where 'sequential.dataset.name' is the name of a sequential disk dataset containing parameters which will control the operation of the ASM for S/390 online retrieval component after control region initialization. This dataset must contain one or more fixed length 80-byte records, and may be blocked or unblocked. OTASS100 parameters consist of a keyword followed by a keyword value. The keyword and its value are separated by a '=' sign and optionally one or more space characters. Parameters may begin in any column and may extend up to column 80, but cannot be continued from one line to the next. An asterisk '\*' may be placed in column 1 to indicate a comment card. Multiple keyword parameters may be contained on a line, each parameter separated by one or more spaces, and optionally a comma (',').

Keyword parameters are defined as follows:

**MAXDRIVE:** 1-3 numeric digits giving the maximum number of tape drives (must be greater than zero) which the NearArchive control region will allocate simultaneously to satisfy online retrieval requests for tape-resident objects (default value = 4).

**MAXQLEN:** 1-3 numeric digits (in the range 1-256), specifying the maximum number of concurrent requests which will be queued by the NearArchive control region for any one active tape dataset (default value = 4).

Any request which causes either of the above limits to be exceeded will be rejected by ASM for S/390 with return code 21, unless a non-zero value has been specified for the TAPEWAIT parameter (see below).

**MAXDISK:** 1-3 numeric digits giving the maximum number of disk reader tasks (must be greater than zero) which the NearArchive control region will initialize to satisfy online retrieval requests (default value = 4). This defines the maximum number of disk ('K') copy object retrieval requests that may be processed simultaneously. A new request for retrieval of a disk copy of an object will be added to the shortest existing task queue if all reader tasks are currently busy. There is no limit to the length of the task queue for each disk reader task.

**TSQDELETE:** 1-4 numeric digits (in the range 0-1440), specifying the number of minutes for which a temporary storage queue containing a retrieved object (in **any** CICS region in the MRO configuration containing the NearArchive control region) is to be retained after retrieval.. The maximum value for this parameter is 1,440 minutes (24 hours).

A value of 0 indicates that temporary storage queues containing retrieved objects are never automatically deleted by ASM for S/390.

For non-zero settings of the TSQDELETE parameter, ASM for S/390 will satisfy an object retrieval request from CICS temporary storage if a queue containing that object already exists. If the value of TSQDELETE is set to 0, all object retrieval requests will be satisfied from tape irrespective of whether the object has previously been retrieved to CICS temporary storage.

A default value of 1440 is used if this parameter is not specified.

**TAPEWAIT:** 1-4 numeric digits (in the range 0-1440), specifying the maximum number of minutes for which object retrieval requests are to be queued internally if there are insufficient resources available to immediately process the request.

The default value of 0 indicates that no internal queuing is to take place – this will cause ASM for S/390 to immediately reject requests which cannot

be processed because of insufficient resources, with a return code of '21'.

For non-zero values, a request which cannot be immediately processed will be queued internally until the necessary tape retrieval resources are available, or until the request has been queued for longer than the time specified in this parameter, at which point it will be rejected with a return code of '21', as above.

**TIMEOUT:** 1-4 numeric digits (in the range 0-1440), specifying the maximum number of minutes for which an object retrieval request is to remain in a stalled condition. A stalled condition occurs when the request is waiting for completion of an external event (e.g. completion of tape mount request, or for completion of a tape I/O request). ASM for S/390 will terminate a stalled retrieval request with a return code of '29' when the interval specified in this parameter has expired.

The default value of 0 indicates that stalled requests are never terminated by ASM for S/390. Requests in this condition will wait indefinitely until the external event causing the condition has completed.

Omission or invalid specification of any parameter will cause its default value to be used by ASM for S/390. If no OTASS100 dataset is present in the CICS JCL, ASM for S/390 will use default values for all parameters.

The values of MAXDRIVE, MAXQLEN, MAXDISK, TSQDELETE, TAPEWAIT and TIMEOUT set during NearArchive initialization may be adjusted during normal CICS operation via the NearArchive online operator interface (transaction OT26 - see chapter 6 for details of operation).

### Example:

0	0	1	1	2	2	3	3	4	4	5												
1	...	5	...	0	...	5	...	0	...	5	...	0	...	5	...	0	...	5	...	0	...	etc

```
MAXDRIVE = 8
MAXQLEN = 6
MAXDISK = 8
TSQDELETE = 10
TAPEWAIT = 5
TIMEOUT = 10
```

#### **2.8.6.2 Secondary NearArchive regions.**

The only JCL change required to CICS regions which will be used as secondary NearArchive regions, is to add the distributed ASM for S/390 load library to the CICS DFHRPL concatenation.

## **2.9 Security user exit.**

In order to allow customers to implement external security control for access to a NearArchive database, an external security module 'OTASUX01' (batch/TSO) or 'OTASUX02' (CICS) will be called at the following points:

1. opening a database for input or update processing.
2. retrieving an object
3. writing an object
4. deleting an object
5. browsing a database definition
6. updating a database definition
7. creating a database definition
8. display CICS control region configuration status
9. updating CICS control region configuration status

A standard user exit parameter area will be passed to the security exit program at each point. This parameter area will contain information about

the caller, the resource being accessed and the type of access required. Customer-specific code may be implemented in the exit program to determine whether access to the resource is to be allowed.

The exit program should return a response code of '00' to ASM for S/390 if access is to be permitted, and a response code of '08' if access is to be denied. If access of types (1) to (4) above is denied, a return code of '29' will be passed back to the application. If access of type (5) to (7) above is denied, an error message will be displayed on the appropriate ISPF panel.

Default user exit modules OTASUX01 and OTASUX02 are supplied with the product. These will always return a response code of '00' to ASM for S/390 (i.e.) access to protected resources is always permitted. These modules should be replaced with modules with identical names to implement customer-specific security processing for batch and CICS applications. This can be done during initial product installation, or at any time subsequent to that.

Request and supplementary codes will be passed to the user exit program indicating the type of request to be validated. These can take the following values:

Request code '01': Access a NearArchive database. This request will be passed to the user exit program during batch connect processing. The access type field will indicate whether the database is being opened for input or update processing. This will take the value 1 when opening the database for input processing, and 2 when opening the database for update processing. The 4-character database identifier in the user exit parameters will identify the database being accessed.

Request code '02': Access an object from a NearArchive database. This request will be passed to the user exit program prior to retrieving or deleting an object from a batch a CICS application, or storing an object from a batch application. The access type parameter field will be used to identify the type of access required. This will take the value 1 for object retrieval, 3 for object deletion, and 4 for object storage. The 4-character database identifier and the object primary key and archive date will identify the object being accessed.



Request code '03': Access a database definition from the NearArchive TSO/ISPF database administration application. This request will be passed to the user exit program when browsing, updating or creating a database definition. The access type parameter field will be used to identify the type of access required. This will take the value 1 when browsing a database definition, 3 when updating a database definition, and 4 when creating a new database definition. The 44-character primary index dataset name will be passed in the parameter list to identify the database definition being accessed.

Request code '04': Access the CICS control region configuration, via the supplied control region operator interface transaction 'OT26'. This request will be passed to the user exit program when displaying or modifying the status of the control region configuration. The access type parameter field will be used to identify the type of access required. This will take the value 1 when display the control region configuration status and 3 when modifying the control region status.

An 84-byte parameter area will be passed to the user exit at each invocation, to allow the user program to permit or deny access to the requested resource. The parameter area will have the following format:

Offset	Length	Type	Description
0	2	Char	Request code:  '01' – access NearArchive database '02' – access object '03' – access database definition '04' – access CICS control region configuration
2	2	Bin	Access type:  1      Read access 2      Update access 3      Alter access 4      Create access
4	2	Char	Response code:

Offset	Length	Type	Description
			'00' – allow access request '08' – reject access request  Any other response code will cause access to be rejected.
6	8	Char	Requester's jobname
14	8	Char	Requester's user identifier
22	4	Char	Requester's transaction identifier (CICS requests only)
26	4	Char	Database identifier (set for request codes '01'/'02' only)
30	44	Char	Database primary index dataset name (set for request code '03' only)
74	2	Bin	Primary key length (set for request code '02' only)
76	4	Ptr	Address of field containing object key (set for request code '02' only)
80	4	Ptr	Address of field containing object archive date (set for request code '02' only)

## 2.10 Conversion of pre-v2.5 databases.

NearArchive databases created using releases of the product prior to v2.5 must be converted for use with this release of the product.

ASM for S/390 v2.5 will not process unconverted databases. Any attempt to run the new release of the product against an unconverted database will have one of the following consequences:

- database open calls issued to NearArchive via the batch application program interface will be rejected with a return code of '91', a supplementary code 1 value of 0 and supplementary code 2 value of 1.

- application calls to NearArchive via the CICS application program interface will be rejected with a return code of '91', a supplementary code 1 value of 0 and supplementary code 2 value of 1.
- NearArchive CICS control region initialization (via transaction OT23) will fail if no v2.5 database primary index file is present in the file control table (FCT).
- NearArchive utilities will terminate with condition code 12 and issue an error message to the error report print file.
- database administration processing in TSO/ISPF will be terminated with message OTS49008 when a pre-v2.5 database is selected for processing.

Each database defined using a pre-v2.5 release of NearArchive should be converted using the following procedure:

Enter the name of the primary index of the database to be converted on the NearArchive database administration entry panel (OTSN4000) in TSO/ISPF, and select option 5. ASM for S/390 will then automatically convert the database for use by v2.5.

The database conversion process consists of reformatting control records in the primary index, and of converting the space management file(s) from a VSAM ESDS file to a VSAM KSDS with variable-length records. There is no requirement to access or convert tape datasets during the database conversion process.

Note that the length of time taken to convert a database will depend on the size of each space management file in the database. This in turn depends on the number of tape volumes in each storage level in the database.

Refer to section 3.5 of this manual for more information on the database conversion process.

### **2.10.1 Database regression.**

Conversion of version 2.2, 2.3 or 2.4 databases may be backed out, if required, by using the database regression utility OTASP032. This utility will process a version 2.5 database, and reformat the index and space management files back into a version 2.2, 2.3 or 2.4 format, as required.

Refer to appendix C of this manual for a description of the database regression utility.

## **2.11 Run installation verification procedures.**

---

After ASM for S/390 installation has been completed successfully, the installation verification procedures should be executed, to verify that all components required for ASM for S/390 operation have been installed and are functioning correctly.

Refer to appendix A of this manual for details on running the installation verification procedures.

---

## CHAPTER 3

---

# *NearArchive Database Definition and Maintenance*

This page is intentionally left blank

Specification and maintenance of a NearArchive database definition is performed using the NearArchive database administration TSO/ISPF dialog facility. This facility will enable all disk dataset components of a NearArchive database to be automatically defined and initialized from within a single TSO session.

The database definition and maintenance facility is designed for use by NearArchive database administration personnel. Access to this facility should be restricted to this category of user. If required, user exit OTASUX01 may be implemented to control database browse and update processes. Refer to section 2.9 for information on implementation of this facility.

The supplied TSO/ISPF database administration facility allows the following NearArchive database definition and maintenance procedures to be undertaken. Any of these functions may be invoked by entering the corresponding function number in the 'Option' field on the database administration entry panel OTSN4000. An example of this panel is given in figure 3.0.

1. Definition of a new NearArchive database.

This procedure allows the database administrator to specify initial values for database definition parameters. A new NearArchive database with a single storage level (storage level 0) will be defined during this procedure.

2. Maintenance of a NearArchive database.

This procedure allows the database administrator to perform the following functions:

- amend existing database definition parameters
- add a new storage level to the database
- add a new secondary index to the database
- amend an existing secondary index definition

OTSN4000	NearArchive V2.5	ENTER OPTION
Database Maintenance Facility		
COMMAND ==>		
-----		
-		
Primary index name ==> _____		
Option ==> ____		
1 - Browse database definition 2 - Update database definition 3 - Base database initialization 4 - Recovery control 5 - Pre-v2.5 database conversion		
Enter required option and press ENTER		
(C) Copyright StorageTek Corp. 1999-2001		

**Fig 3.0:** database administration entry panel OTSN4000

### 3. Browse a NearArchive database definition.

A NearArchive database administrator will be able to browse an existing NearArchive database definition, to display the following details:

- base database definition values
- definition of each storage level in the database
- definition of all secondary indexes defined in the database
- volume recovery entries in each storage level of the database.

### 4. Maintain recovery control entries for the database.

This procedure allows the database administrator to define, amend or delete a recovery control entry in any storage level in the database.

Recovery control entries are created by the database administrator when a tape cartridge volume becomes unavailable due to media failure or loss. These entries are used to specify backup copies to be



used for data retrieval and dataset recovery purposes.

This procedure also allows enabling or disabling of the NearArchive disaster recovery option. This option enables all tape datasets in a NearArchive database to be accessed from a specified backup copy identifier, in cases where an entire database is unavailable.

5. Convert a database created with an earlier release of the product for use with version 2.5.

V2.5 of ASM for S/390 will reject requests to process any pre-v2.5 database which has not been converted for use with this release. This option will allow a pre-v2.5 database to be automatically converted via the TSO/ISPF database administration dialog. Note that no tape dataset access or conversion is required during the database conversion process.

This chapter provides detailed information regarding each of the above procedures.

### **3.1 Definition of a New NearArchive Database.**

---

To enter the NearArchive database administration facility, select the appropriate option from your main ISPF application menu. Consult the person(s) responsible for installing and implementing ASM for S/390 on your system for precise details on how to invoke the NearArchive database administration facility on your TSO/ISPF system.

NearArchive database definition processing may be initiated by entering option '3' from the main database maintenance facility menu OTSN4000, and entering the name to be used for the database's primary index dataset. This dataset must not already exist. It will be automatically created and initialized during database definition processing.

The database definition procedure will display the following panels in sequence:

- OTSN4300 - base database definition panel
- OTSN4301 - storage level definition panel
- OTSN4302 - Access Method Services control panel for primary index definition
- OTSN4303 - Access Method Services control panel for journal dataset definition (if audit trail facility enabled)

OTSN4304 - Access Method Services control panel for journal archive database definition (if journal archive facility enabled).

Enter the appropriate details on each panel and press 'ENTER' to proceed to the next panel. When all panel entries have been accepted, the NearArchive database will be defined and initialized. Panel OTSN4000 will be redisplayed after successful definition of the database.

Press 'PF3' to terminate the database definition procedure at any point. NearArchive database definition will be abandoned, and panel OTSN4000 re-displayed. The database will not have been defined.

Default values will be used where no entry has been made for optional fields. Entries must always be made for mandatory fields. The description of individual fields below will identify whether an entry is mandatory or optional.

Online help information for each panel may be obtained by pressing 'PF1'.

OTSN4300	NearArchive V2.5	ENTER VALUES
Base Database Definition		
COMMAND ===>		
-----		
Primary index name ===> _____		
Length of primary key ===> ____		
NearArchive user SVC no ===> ____		
HLQ for this database ===> _____		
Data retention period ===> ____ yrs ____ days		
System dataset unitname ===> _____		
Online processing options:		
Retain tapes on drive for ===> ____ minutes		
Age range of tapes to be retained on drive ===> ____ to ____ days old		
Audit options:		Retrieval logging options:
Audit trail on? ===> _		SMF record id ===> ____
Journal archiving enabled? ===> _		Batch retrieval logging on? ===> _
		CICS retrieval logging on? ===> _

**Fig 3.1:** base database definition panel OTSN4300

### **3.1.1 Base Database Definition.**

After selecting option 3 from the database administration entry panel, the base database definition panel OTSN4300 will be displayed. This panel is illustrated in figure 3.1. The PF7 ('UP') and PF8 ('DOWN') keys may be used to scroll backwards and forwards through the panel display.

Panel OTSN4300 contains the following fields:

**Primary index name** This field gives the name of the database's primary index dataset, as entered on panel OTSN4000. This database will be created during database initialization processing. The contents of this field may not be amended.

**Length of primary key** Mandatory entry. Enter the primary keylength for the database being defined. This must be a numeric value in the range 1-254.

The maximum number of DBCS characters permitted is 126 plus the Shift-Out (SO) and Shift-In (SI) bytes. If using DBCS, enter the true count of bytes used and not the count of characters. For example, ((number of DBCS characters \* 2) + number of SO bytes + number of SI bytes) must be less than 254 bytes.

**NearArchive user SVC no** Mandatory entry. Enter the user SVC number allocated for NearArchive during the product installation procedure. This must be a numeric value in the range 200-255

Consult the person(s) responsible for installing ASM for S/390 on your system if you are unsure of this value.

**HLQ for this database** Mandatory entry. Enter the high-level database qualifier to be used by ASM for S/390 when creating system datasets (disk and tape) for this database. This field must contain from 1-30 alphanumeric characters, and be a valid dataset name.

**Data retention period** Optional entry. Enter the length of time (in years and days) for which a NearArchive object is to be

retained in the database. Enter a value of 0 years and 0 days if objects are to be retained indefinitely in the database (i.e.) no automatic expiry.

If no entry is made in this field, a default value of 0 years and 0 days is used (i.e.) objects will not be expired.

**System dataset  
unitname** Optional entry. Enter a valid system unit name or esoteric to be used internally by ASM for S/390 when allocating temporary disk datasets during database administration processing.

If no entry is made in this field, a default value of SYSALLDA will be used.

**Retain tapes on  
drive** Optional entry. Enter the number of minutes for which tape volumes in this database are to be retained on a drive during online retrieval (CICS) processing, after all retrievals have been processed. This must be a numeric value in the range 0-1439. A value of 0 (the default) indicates that tapes are not to be retained on a drive after use.

If a non-zero value is specified in this field, ASM for S/390 will not automatically dismount a tape volume which satisfies the age range retention criteria (as specified in the following field on the panel) after all outstanding retrievals to that volume have been completed. The volume will instead be automatically dismounted when the tape volume has been unreferenced for the length of time specified in this field.

**Age range of  
tapes retained  
on drive** Optional entry. This field is used in conjunction with the preceding field on the panel to control the disposition of a tape cartridge volume after all outstanding requests for retrieval from that volume have been completed during online retrieval (CICS) processing.

If a non-zero tape drive retention period has been specified in the preceding field, volumes containing one or more objects whose age (as

determined by object archive date) falls within the range specified here, will be retained on the tape drive after all outstanding retrieval requests have been completed. This will improve retrieval performance for subsequent accesses to that volume. Volumes not in this category will be automatically dismounted after use, irrespective of the tape drive retention value specified in the preceding field. Refer to section 6.4.1 of this manual for additional information on this topic.

Enter the range of ages (in days) which is to be used to control this processing. Each value must be a 1-5 character numeric identifier in the range 0-65535. The value specified for the end of the age range must be greater than or equal to the value specified for the start of the age range.

If omitted, a value of 0 is used for start and end values. This value will mean that no tape volumes in this database will be retained on a drive during online object retrieval, irrespective of the value specified in the preceding field.

**Audit trail on** Optional entry. Enter 'Y' to enable the audit trail facility for this database. This will cause all updates to the database to be logged in the database's journal dataset. This dataset will be automatically created during database definition processing if the audit trail facility has been requested, and the event will be recorded in the newly-created journal file. Enter 'N' to disable the audit trail facility for this database.

If omitted, a value of 'N' is used for this entry (audit trail facility disabled).

**Journal archiving enabled** Optional field. Enter 'Y' if the journal archiving facility is to be enabled for this database. This will cause all journal entries to be archived to a specially-defined NearArchive database, prior to the contents of the journal dataset being cleared by the journal maintenance utility OTASP070. This will enable historical analysis of journal entries to be performed.

The journal archive database will be automatically created during database definition processing, if this facility has been enabled, and the event will be recorded in the newly-created journal file.

Enter 'N' to disable the journal archiving facility for this database.

Refer to section 7.3 of this manual for additional information on journal processing.

If omitted, a value of 'N' is used for this entry (journal archiving disabled).

**SMF record id** Optional entry. Enter the 3-digit numeric value (in the range 128 to 255) of the SMF record identifier which is to be used for object retrieval logging. Enter a value of '0' to disable SMF logging for this database.

If omitted, a value of '0' is used for this entry (SMF logging disabled).

**Batch retrieval logging on** Optional entry. Enter 'Y' to enable SMF logging during object retrieval processing from batch application programs. Enter 'N' to disable SMF logging for batch retrievals. A value of 'Y' will only be accepted if the SMF record identifier is non-zero.

If omitted, a value of 'N' is used for this entry.

**CICS retrieval logging on** Optional entry. Enter 'Y' to enable SMF logging during object retrieval processing from CICS application programs. Enter 'N' to disable SMF logging for CICS retrievals. A value of 'Y' will only be accepted if the SMF record identifier is non-zero.

If omitted, a value of 'N' is used for this entry.

After all required entries have been made, press 'ENTER'. Any invalid entry will be rejected, and a corresponding message displayed in the top right hand corner of the panel. For additional information regarding the error, press PF1. To obtain help information regarding panel definition entries, press PF1 once more.

When all entries have been accepted, the storage level definition panel OTSN4301 will be displayed to allow storage level 0 to be defined for the NearArchive database.

OTSN4301		NearArchive V2.5		ENTER VALUES	
Storage Level Definition					
COMMAND ==>					
-----					
Primary index name ==>		OTAS.V2R5.IVP.INDEX			
Storage level ==>		0			
Database blocksize ==>		_____			
Maximum blockcount per volume ==>		_____			
Unit name for scratch allocn: A ==>		_____		B ==> _____	
C ==>		_____		D ==> _____	
Tape usage options:					
ICRC ==>		_____		Retention period ==> _____	
Dynamic load balancing ==>		_____		or Expiry date ==> _____	
Volsafe enabled: A ==>		_____		B ==> _____	
C ==>		_____		D ==> _____	
D/b maintenance options:			Duplexing options:		
Recycle threshold ==>		_____ %		Duplex during archival ==> _____	
AUTOUNCAT during recycle ==>		_____		Duplex during backup ==> _____	
Disk compress threshold ==>		_____ %			
Disk copy options:					
Unit name ==>		_____		Primary alloc ==> _____	
Retention ==>		_____ days		Secondary alloc ==> _____	
Release unused space ==>		_____		Dataset blocksize ==> _____	
				Maximum blockcount per segment ==> _____	

**Fig 3.2:** Storage level definition panel OTSN4301

### **3.1.2 Storage Level 0 Definition.**

The storage level definition panel OTSN4301 is displayed to allow definition of storage level 0 for the NearArchive database being defined. All new NearArchive databases are defined with a single storage level (level 0). Creation of additional storage levels is performed through the database maintenance dialog (option 2 from the main NearArchive menu).

Panel OTSN4301 is illustrated in figure 3.2, and contains the fields identified below. The PF7 ('UP') and PF8 ('DOWN') keys may be used to scroll backwards and forwards through the panel display.

**Primary index name** This field gives the name of the database's primary index dataset, as entered on panel OTSN4000. This dataset will be created during database initialization processing. The contents of this field may not be



amended.

**Storage level** This indicator identifies the storage level being defined. During initial database definition processing, this value will always be 0. The contents of this field may not be amended.

**Database  
blocksize** Mandatory entry. Enter the blocksize to be used by ASM for S/390 when creating tape or disk datasets for storage level 0 in this NearArchive database. This should be a numeric value the range 9-32760. As ASM for S/390 does not support spanned records, the blocksize entered here should not be less than the size of the largest logical record to be written in any NearArchive object.

**Maximum  
blockcount per  
volume** Optional entry. Enter the maximum number of blocks (of blocksize as entered in the previous field) which are to be written to each tape cartridge volume in storage level 0 of the NearArchive database. This should be a numeric value in the range 0-9999999.

Enter a value of 0 to indicate that maximum blockcount processing is not to be enabled. In this case, ASM for S/390 will fill tape cartridge volumes in this storage level to capacity.

Restricting the number of blocks written to each volume will enable control over the average time taken to locate any tape block during retrieval processing. Reducing the maximum block count will improve ASM for S/390 data retrieval response times.

If this entry is omitted, a value of 0 is used (i.e.) fill tape cartridge volumes to capacity.

**Unit name for  
scratch allocation** Values entered in these four fields are used by ASM for S/390 when dynamically allocating primary or backup copies of new tape datasets in storage level 0 in this NearArchive database. An entry for the 'A' copy is mandatory. Entries for the 'B', 'C' and 'D' copies are optional. If omitted, the 'A' copy unit name will be used for scratch

allocation of datasets in the appropriate database copy. Entries should consist of an 8-character alphanumeric identifier which identifies a valid esoteric unit name on the host system control program. These fields may be used to control the type or location of the device(s) used to process tape datasets in the appropriate copy of the NearArchive database. Refer to section 4.7 for a description of database backup processing.

**ICRC** Optional entry. Enter the option to be used for implementation of the Improved Cartridge Recording Capability (ICRC) tape data compaction feature for storage level 0 in this NearArchive database.

The available options are:

*ON* ICRC is always enabled when writing data to storage level 0 in this NearArchive database.

*OFF* ICRC is always disabled when writing data to storage level 0 in this NearArchive database.

*DEFAULT* The system default ICRC setting is used when writing data to storage level 0 in this NearArchive database.

If no entry is made in this field, the default value used is 'DEFAULT'. If ICRC is not available, specify a value of 'DEFAULT' in this field, or allow it to default.

**Dynamic load balancing** Optional entry. When dynamic load balancing is enabled, ASM for S/390 will automatically retry a batch or online retrieval request from a designated backup copy when a 'resource unavailable' condition has been detected (i.e., in conditions when ASM for S/390 would generate a return code of '21' if dynamic load balancing is not enabled - refer to the description of return code '21' in section 8.2 for a list of these conditions). If the

retry request subsequently fails with the same condition, ASM for S/390 will generate a return code of '21' as normal.

To enable dynamic load balancing, enter the identifier of the database copy to be used for automatic retry of the retrieval request ('B', 'C' or 'D'). Note that for performance reasons, the database copy specified here should be available for automated processing.

To disable dynamic load balancing, enter 'N'. This is the default value.

**Retention period** Optional entry. Enter the number of days (in the range 1-9999) to be assigned to the retention period for each new tape dataset created in the NearArchive database. Enter '0' (the default value) if no explicit retention period is to be assigned by NearArchive.

**Expiry date** Optional entry. Enter the expiry date to be set for each new tape dataset created in the NearArchive database, in the format 'YY/DDD' (20<sup>th</sup> century dates only) or 'YYYY/DDD'. Enter blanks (the default value) if no explicit expiry date is to be assigned by ASM for S/390.

Retention period or expiry date value may be entered for each storage level, but not both. If both entries are omitted a default retention period/expiry date will be allocated by the tape management product in use on the system, if present.

**Volsafe enabled** Optional entries. Values entered in these four fields are used by ASM for S/390 when dynamically allocating primary or backup copies of new tape datasets in storage level 0 in this NearArchive database to control whether a tape volume with the Sun Oracle Volsafe data protection feature is to be used for the new tape dataset.

Enter 'Y' if Volsafe is required for the corresponding tape copy, or 'N' if Volsafe is not required. ASM for S/390 will ensure that the

Volsafe feature is present or absent (depending on the value of the corresponding field) when allocating a scratch tape in storage level 0 in the database. Allocation will be rejected if the Volsafe status of the scratch volume does not match that requested. A default value of 'N' is used if no field entry is made.

**Recycle threshold** Optional entry. Enter a value giving the recycle threshold percentage of active objects on a tape cartridge volume in storage level 0 in this NearArchive database. This should be a numeric value in the range 0-99.

If the percentage of active objects on any volume falls below this level, that volume will be eligible for recycling by the NearArchive database maintenance utility OTASP030 (see section 7.2 for a description of this utility).

Entering a value of 0 in this field indicates that database maintenance is not to be performed on tape datasets in storage level 0 of this NearArchive database.

If omitted, a default value of 0 is used for this entry (i.e.) tape recycling disabled.

**Autouncat during recycle** Optional entry. Enter 'Y' if tape datasets are to be automatically uncataloged when released during recycle processing by the NearArchive database maintenance utility OTASP030 (see section 7.2 for a description of this utility). Enter 'N' if no automatic uncataloging is to be performed.

If omitted, a default value of 'N' is used for this entry.

**Disk compress threshold** Optional entry. Enter a value giving the disk compress threshold, expressed as the percentage of active objects on a disk copy dataset in storage level 0 in this NearArchive database. This should be a numeric value in the range 0-99.

If the percentage of active objects on any disk copy

dataset falls below this level, that dataset will be eligible for compression by the NearArchive database maintenance utility OTASP030 (see section 7.1 for a description of this utility).

Entering a value of 0 in this field indicates that database maintenance is not to be performed on disk copy datasets in storage level 0 of this NearArchive database.

Note that the threshold applies to individual disk copy datasets, and not to globally to all disk copy datasets associated with a single primary tape dataset. Disk copy datasets are treated individually for purposes of compression processing.

If omitted, a default value of 0 is used for this entry (i.e.) disk dataset compression disabled.

**Duplex during archival** Optional entry. Enter 'Y' if duplex ('C') copies of a NearArchive tape dataset are to be produced in addition to primary ('A') copies during batch archival processing.

Enter 'N' if no duplex copies are to be created during batch archival. If omitted, a default value of 'N' is used for this entry.

**Duplex during backup** Optional entry. Enter 'Y' if duplex full-volume backup ('D') copies of a NearArchive tape dataset are to be produced in addition to primary full-volume backup ('B') copies during default execution of the NearArchive database backup utility OTASP110 in full-volume backup mode. An entry of 'Y' in this field also indicates to the backup utility that duplex incremental backup ('J') copy datasets should be produced in addition to primary incremental backup ('I') copy datasets when the utility is executing in incremental backup mode.

Enter 'N' if no duplex backup copies (full-volume or incremental) are to be created during database backup processing. If omitted, a default value of

'N' is used for this entry.

### **Disk copy options :**

**Unit name** Optional entry. Enter the 1-8 character unit name to be used when creating disk ('K') copy datasets in this storage level. ASM for S/390 will automatically create a 'K' copy dataset when creating each new primary tape dataset during object archival processing. This dataset will be used to store disk copies of objects which have been stored in the corresponding primary tape dataset.

When a disk copy of an object exists, NearArchive batch and online retrieval processing will always attempt to retrieve that copy first. If disk retrieval fails for any reason, then retrieval of the tape copy of the object will continue as normal. If no entry is made in this field, then disk copy processing will not be enabled for the storage level being defined.

**Retention** Optional if disk unit name is not blank; must be blank otherwise.

Enter a 1-5 character numeric identifier in the range 0-36525. This value will control the length of time (in days) for which the disk copy of an object will be retained after initial storage in this storage level. The disk copy of the object will be automatically deleted by the object management utility when this retention period has been exceeded.

A value of 0 indicates that the disk copy is to be retained permanently. In this case, the disk copy will only be deleted when the entire object is expired by the object management utility (as controlled by the object retention period specified in base database definition).

If this entry is omitted, a default value of 0 is used (i.e. permanent disk retention).

**Release unused space** Optional if disk unit name is not blank; must be blank otherwise.

Enter 'Y' if unused disk space assigned when creating a new disk ('K') copy dataset in this storage level is to be released at the completion of database update processing.

Enter 'N' if the space allocated to the new disk copy dataset is to be retained at completion of database update processing.

If no entry is made in this field, a default value of 'N' is used.

**Primary alloc** Mandatory if disk unit name is not blank; must be blank otherwise.

Enter a 1-8 character numeric identifier in the range 1-16777215, giving the number of blocks to be allocated as the primary space allocation when ASM for S/390 creates a disk copy dataset in this storage level.

If this amount of contiguous disk storage is not available when creating a disk copy dataset during object archival processing, disk copy processing will be automatically disabled by ASM for S/390 until the next new tape dataset is being created.

**Secondary alloc** Mandatory if disk unit name is not blank; must be blank otherwise.

Enter a 1-8 character numeric identifier in the range 1-16777215, giving the number of blocks to be allocated for each secondary space allocation when ASM for S/390 creates a disk copy dataset in this storage level.

If this amount of contiguous disk storage is not available when extending a disk copy dataset during object archival processing, ASM for S/390 will close the current disk copy dataset and attempt to allocate a new disk copy dataset for storage of disk copies of objects being archived to the associated primary tape copy dataset.

**Dataset blocksize** Optional if disk unit name is not blank; must be blank otherwise.

Enter a 1-8 character numeric identifier in the range 9-32760, giving the blocksize that will be used when allocating new disk ('K') copy datasets in this storage level. This value does not need to be the same as that specified for tape blocksize in this storage group, nor does it need to be larger than the longest record to be archived in an object in this NearArchive database. This allows the most appropriate disk blocksize to be selected for the geometry of the disk volume on which the 'K' copy dataset will be created. If no entry is made in this field, a default value of 23,468 bytes (half-track blocking on a 3380 device) is used.

**Maximum blockcount per segment** Optional if disk unit name is not blank; must be blank otherwise.

Enter a 1-7 character numeric identifier in the range 0-9999999, giving the maximum number of data blocks to be written to any one disk ('K') copy dataset segment in this storage level of the database. A value of 0 in this field indicates that there is no maximum disk blockcount.

When this block count has been exceeded, ASM for S/390 will close the current disk dataset, and allocate a new disk copy dataset for storage of disk copies of objects being archived to the associated primary tape copy dataset. A maximum of 256 disk copy datasets may be created per primary tape copy dataset. If 256 disk copy dataset have already been created for the corresponding primary tape copy dataset, ASM for S/390 will disable disk copy processing until the next primary tape copy dataset is created.

An entry in this field is used to limit the overall size of each disk copy dataset. This is particularly important when high-capacity tape volumes are used for the primary tape copy datasets. During processing of the object management utility (OTASP040) a disk copy dataset will be deleted



(and the disk space occupied by the file released) only when the disk copies of all objects within that dataset have expired. Creation of multiple disk copy datasets per primary tape copy dataset will allow disk space to be released more actively by the object management utility and may reduce the overall amount of disk space required to support ASM for S/390 disk copy processing.

If no entry is made in this field, a default value of 0 will be used (i.e. no maximum blockcount). In this case ASM for S/390 will continue to write to a disk copy dataset until the primary tape dataset has been closed, or until there is no more space available in the disk dataset; in this latter case the disk dataset will be closed and a new disk copy dataset created.

After all required entries have been made, press 'ENTER'. Any invalid entry will be rejected, and a corresponding message displayed in the top right hand corner of the panel. For additional information regarding the error, press PF1. To obtain help information regarding panel definition entries, press PF1 once more.

When all entries have been accepted, the primary index Access Method Services panel OTSN4302 will be displayed to enable customization of the primary index definition parameters for the NearArchive database being defined.

### **3.1.3 Access Method Services Control.**

After all database definition and initialization parameters have been accepted, one or more Access Method Services (AMS) control panels will be displayed to allow customization of AMS parameters used by ASM for S/390 when defining system datasets for the database being defined.

AMS control panels will be displayed in the following sequence:

1. **OTSN4302** - the primary index AMS control panel. This panel is always displayed. Values entered on this panel will be used when creating the primary index dataset (VSAM KSDS) for the database being defined.

2. **OTSN4303** - the journal dataset AMS control panel. This panel will only be displayed if the audit control option was enabled on panel OTSN4300. Values entered on this panel will be used during creation of the journal dataset (VSAM ESDS) for the database being defined.

Note that panel OTSN4303 differs from the other AMS control panels in that only one 'CI size' parameter is specified. This is due to the fact that the journal dataset is an ESDS, and consequently does not contain an index component.

```
OTSN4302                      NearArchive 2.5

                                Access Method Services Control
COMMAND ===>
-----
Primary index name ===> _____

Primary index control:

    Volume serial number ===> _____
    Unit of allocation ===> _____
    Primary allocation value ===> _____
    Secondary allocation value ===> _____
    Data CI size ===> _____
    Index CI size ===> _____
    Delete before define? ===> _ (Y/N)

Press ENTER after all details have been entered correctly.

After all database definition parameters have been accepted,
the primary index will be allocated and initialized.
```

**Fig 3.3:** Access Method Services control panel OTSN4302

3. **OTSN4304** - the journal archive database AMS control panel. This panel will only be displayed if both the audit control facility and journal archiving were enabled on panel OTSN4300. Values entered on this panel will be used during creation of the journal archive database primary index dataset (VSAM KSDS).

Panel OTSN4302 is illustrated in figure 3.3. Each of the above panels will contain some or all of the following fields:

<b>Primary index name</b>	This field gives the name of the database's primary index dataset, as entered on panel OTSN4000. This dataset will be created during database initialization processing. The contents of this field may not be amended.
<b>Volume serial number</b>	Optional entry. Enter the serial number (1-6 alphanumeric, national or special characters) of the disk volume on which the dataset is to be created. This parameter may be omitted if the volume is to be automatically selected by the operating system.
<b>Unit of allocation</b>	Optional entry. Enter one of the following values, which is to be used to identify the unit of allocation specified in the primary and secondary allocation value fields:  <i>CYL</i> - allocation is to be performed in cylinders <i>TRK</i> - allocation is to be performed in tracks <i>REC</i> - allocation is to be performed in records.  If this entry is omitted, a default value of 'CYL' is used (i.e.) primary and secondary allocation units are specified in cylinders.
<b>Primary allocation value</b>	Mandatory entry. Enter the number of allocation units (as entered or defaulted in the preceding field) to be used for allocation of the primary data extent of the dataset. Refer to section 3.1.4 for information on choosing a primary allocation value.

**Secondary allocation value** Mandatory entry. Enter the number of allocation units (as entered or defaulted in the unit of allocation field) to be used for allocation of each secondary data extent of the dataset. Refer to section 3.1.4 for information on choosing a secondary allocation value.

**Data CI size** Optional entry. Enter the control interval CI size to be used for allocation of the data component of the dataset. This should be a valid VSAM control interval size.

If omitted, a control interval size of 16k is used for the data component.

**Index CI size** Optional entry. Enter the control interval size to be used for allocation of the index component of the dataset (KSDS only). This should be a valid VSAM control interval size.

If omitted, a control interval size of 2k is used for the index component.

Choosing unique control interval sizes for the various VSAM system datasets defined for a NearArchive database may improve performance of batch NearArchive operations, through the implementation of exclusive buffer pools for each dataset.

**Delete before define?** Optional entry. Enter the value 'Y' if ASM for S/390 should attempt to delete the dataset before it is defined. Enter 'N' if no deletion request should be issued.

The NearArchive database definition process will proceed as normal if a value of 'Y' is entered here and the dataset did not exist.

If omitted, a value of 'N' is used for this entry.

Press 'ENTER' after all required entries have been made on each panel. After all AMS control panel entries have been accepted, the database will be defined and initialized.

After successful completion of the database definition process, the NearArchive main menu panel OTSN4000 will be redisplayed. Further database administration activities may be performed as required.

If any error occurs during AMS processing, an error panel will be displayed, allowing viewing of AMS output messages. Error messages should be noted, and the appropriate corrective action taken. The main menu panel will then be redisplayed to allow retry of the database definition procedure.

### **3.1.4 System Dataset Sizing.**

Information in this section is provided as a guide for choosing the primary and secondary allocation values used during definition of the following NearArchive system datasets:

- primary index dataset
- journal dataset
- journal archive database primary index dataset

Primary and secondary allocation values for each category of system dataset may be chosen using the following guide:

#### **Primary index allocation:**

Each primary index dataset contains the following entries:

<b>Main index entries</b>	One main index entry will exist for each unique primary key value in the database. Each entry is a single VSAM record, consisting of a fixed prefix of length (k+4) bytes (where 'k' is the primary keylength), and a variable number of 32-byte suffixes. One suffix will exist for each NearArchive object held in the database for that primary key value. The maximum main index entry length is 65524 bytes.
<b>Supplementary index entries</b>	A main index entry may be accompanied by one or more supplementary index entries. These entries are used to retain information about secondary keys which index one or more objects with the associated primary key value.

For each main index entry, one or more

supplementary entries (up to a maximum of 255) may exist for each secondary index that has been defined in the database.

Each supplementary entry is a single VSAM record, consisting of a fixed prefix, of length  $(k+4)$  bytes where 'k' is the primary keylength), and a variable number of suffixes, each of length 's' bytes (where 's' is the length of the secondary key for that supplementary record). One suffix will be present for each secondary key value which indexes an object with the associated main index entry's primary key value.

As many supplementary records as are required to hold all secondary key values will be present in the index (up to a maximum of 255).

This information may be used to derive an estimate of the primary and secondary allocation values to be used when defining the primary index.

If necessary, an approximate estimate may be used during initial dataset allocation. By monitoring primary index usage, the primary and/or secondary allocation values may be amended during the life of the database.

The following formula may be used to estimate the total amount of space (in bytes) required for a primary index dataset:

**Main index entries:**  $(A * (k+4+(32*B)))$

Where A = total number of unique primary key values in database

B = average number of objects per primary key

and k = primary keylength.

**Supplementary index entries:**  $(A * (k+4+(s*C)))$   
(per secondary index)

where C = average number of secondary key values per primary key

and s = secondary keylength.

The supplementary index entry calculation should be repeated for each defined secondary index in the database.

An estimate for total primary index space over the life of the database may be derived by adding the main index entry requirements to the supplementary index entry requirements for each defined secondary index.

Primary and secondary allocation values may be chosen to allow gradual growth in the size of the index over the life of the database until it is fully populated. This point will be reached when data has been held in the database for longer than the database's retention period. This data will be expired and removed from the database by the NearArchive object management utility. Index space released by this process will be re-used, and consequently the size of the primary index dataset will then stabilise.

#### **Journal dataset allocation:**

One VSAM record will be created for each entry in the journal dataset. Each record will be (k+39) bytes long (where 'k' is the database primary keylength).

The contents of the journal dataset will be cleared after successful completion of the NearArchive journal maintenance utility OTASP070. The journal dataset should be large enough to contain the maximum number of journal entries created between successive executions of the journal maintenance utility.

#### **Journal archive database allocation:**

For journal archive allocation, one main index entry record will be created in the primary index for each day's data held in the journal archive database. Each record will be 40 bytes in length.

The maximum index space required to hold a year's worth of information in the journal archive database primary index will therefore be:

$$365 * 40 = 14600 \text{ bytes}$$

Supplementary index records are not used in the journal archive database.

Choose primary and secondary allocation values for the journal archive database primary index which will allow a gradual growth in the size of the dataset over the life of the database.

## 3.2 Maintenance of a NearArchive Database.

---

Maintenance of a NearArchive database is invoked from the main NearArchive database maintenance facility menu OTSN4000. Enter on this panel the name of the primary index dataset of the database on which maintenance is to be performed, and select option 2 ('Update database definition').

The database maintenance menu panel OTSN4200 will then be displayed (see figure 3.4). This allows selection of the following options:

1. Base database maintenance. This option enables maintenance of general database attributes which apply to all storage levels in a NearArchive database.
2. Secondary index maintenance. Choose this option to define new secondary indexes for the database, or to update or delete the definition of existing secondary indexes.
3. Storage level maintenance. Choose this option to add a new storage level to the database, or to maintain database attributes relating to an existing storage level.

```
OTSN4200                      NearArchive V2.5
                                Database Maintenance Menu
COMMAND ===>
-----
Primary index name ===> _____
                                Option ===> _
                                1 - Base database maintenance
                                2 - Secondary index maintenance
                                3 - Storage level maintenance

Enter required option and press ENTER
```

**Fig 3.4:** Database maintenance menu panel OTSN4200



Enter on panel OTSN4200 the option corresponding to the type of database maintenance to be performed. Sections 3.2.1 to 3.2.3 describe each of the above maintenance facilities.

### **3.2.1 Base Database Maintenance.**

The base database maintenance facility is invoked by entering option 1 on the database maintenance menu panel OTSN4200. This facility is used to maintain general database attributes which control processing in all storage levels of the database.

The following attributes may be amended via this facility (refer to section 3.1.1 for a detailed description of the usage of each attribute):

- NearArchive user SVC number. This may be amended if SVC numbers are re-allocated during the life of a database.
- Data retention period. The length of time that data is held in the database may be amended as required during the life of the database.
- System dataset unitname. The unit name used internally by ASM for S/390 when allocating temporary disk datasets during database administration processing.
- Age range of tapes to be retained on drive. Any change in processing requirements for this facility may be made via this panel.
- Audit trail on. The audit trail facility may be turned on or off, as required, at any stage during the life of a database. The database's journal dataset will be automatically allocated and initialized when this facility is enabled for the first time.
- Journal archiving enabled. Archival of journal entries by the NearArchive journal maintenance utility OTASP070 may be enabled or disabled. The database's journal archive database will be automatically allocated and initialized when this facility is enabled for the first time.

The following general database attributes may **not** be amended during the life of a database (i.e.) after a database has been defined and initialized:

- Primary keylength.
- High-level qualifier for system datasets.

The base database maintenance panel OTSN4210 will be displayed when option 1 has been selected from the database maintenance menu OTSN4200, giving details of the existing database definition. Make the appropriate changes and press 'ENTER' to update the database definition. Press PF3 ('END') to terminate the database maintenance procedure. Any changes will be ignored, and the database definition will not be updated. The PF7 ('UP') and PF8 ('DOWN') keys may be used to scroll backwards and forwards through the panel display.

OTSN4210	NearArchive V2.5	ENTER VALUES
Base Database Definition		
COMMAND ==>		
-----		
Primary index name ==> OTAS.V2R5.IVP.INDEX		
NearArchive user SVC no ==> _____		
Data retention period ==> _ yrs _ days		
System dataset unitname ==> _____		
Online processing options:		
Retain tapes on drive for ==> _____ minutes		
Age range of tapes to be retained on drive ==> _____ to _____ days old		
Audit options:		Retrieval logging options:
Audit trail on? ==> _		SMF record id ==> _____
Journal archiving enabled? ==> _		Batch retrieval logging on? ==> _
		CICS retrieval logging on? ==> _
Press ENTER when all details entered correctly		

**Fig 3.5:** base database maintenance panel OTSN4210

Panel OTSN4210 contains the following fields:

- |                                |   |
|--------------------------------|---|
| <b>Primary index name</b>      | This field gives the name of the database's primary index dataset, as entered on panel OTSN4000. The contents of this field may not be amended. |
| <b>NearArchive user SVC no</b> | Enter the new SVC number to be used for ASM for S/390 use. This must be a numeric value in the range 200-255.                                   |

<b>Data retention period</b>	Enter the length of time (in years and days) for which a NearArchive object is to be retained in the database. Enter a value of 0 years and 0 days if objects are to be retained indefinitely in the database (i.e.) no automatic expiry.
<b>System dataset unitname</b>	Enter a valid system unit name or esoteric to be used internally by ASM for S/390 when allocating temporary disk datasets during database administration processing.
<b>Retain tapes on drive</b>	<p>Enter the number of minutes for which tape volumes in this database are to be retained on a drive during online retrieval (CICS) processing, after all retrievals have been processed. This must be a numeric value in the range 0-1439. A value of 0 (the default) indicates that tapes are not to be retained on a drive after use.</p> <p>If a non-zero value is specified in this field, ASM for S/390 will not automatically dismount a tape volume which satisfies the age range retention criteria (as specified in the following field on the panel) after all outstanding retrievals to that volume have been completed. The volume will instead be automatically dismounted when the tape volume has been unreferenced for the length of time specified in this field.</p>
<b>Age range of tapes retained on drive</b>	<p>This field is used in conjunction with the preceding field on the panel to control the disposition of a tape cartridge volume after all outstanding requests for retrieval from that volume have been completed during online retrieval (CICS) processing.</p> <p>If a non-zero tape drive retention period has been specified in the preceding field, volumes containing one or more objects whose age (as determined by object archive date) falls within the range specified here, will be retained on the tape drive after all outstanding retrieval requests have been completed. This will improve retrieval performance for subsequent accesses to that volume. Volumes not in this category will be automatically dismounted after</p>

use, irrespective of the tape drive retention value specified in the preceding field. Refer to section 6.4.1 of this manual for additional information on this topic.

Enter the range of ages (in days) which is to be used to control this processing. Each value must be a 1-5 character numeric identifier in the range 0-65535. The value specified for the end of the age range must be greater than or equal to the value specified for the start of the age range.

**Audit trail on** Enter 'Y' to enable the audit trail facility for this database. This will cause all updates to the database to be logged in the database's journal dataset. This dataset will be automatically created during database maintenance processing if the audit trail facility is being enabled for the first time

Enter 'N' to disable the audit trail facility for this database. The database's journal dataset will not be deleted when this facility is disabled.

Any change in the status of the audit trail indicator will be logged in the database's journal file.

**Journal archiving enabled** Enter 'Y' if journal archiving is to be enabled for this database. This will cause all journal entries to be archived to a specially-defined NearArchive database, prior to the contents of the journal dataset being cleared by the journal maintenance utility OTASP070. This will enable historical analysis of journal entries to be performed.

The journal archive database will be automatically created and initialized during database maintenance processing, if this facility is being enabled for the first time.

Enter 'N' to disable the journal archiving facility for this database. The database's journal archive database will not be deleted when this facility is disabled.

Any change in the status of the journal archiving indicator will be logged in the database's journal file.

Refer to section 7.4 of this manual for additional information on journal processing.

- |                                   |   |
|-----------------------------------|---|
| <b>SMF record id</b>              | Enter the 3-digit numeric value (in the range 128 to 255) of the SMF record identifier which is to be used for object retrieval logging. Enter a value of '0' to disable SMF logging for this database.                                   |
| <b>Batch retrieval logging on</b> | Enter 'Y' to enable SMF logging during object retrieval processing from batch application programs. Enter 'N' to disable SMF logging for batch retrievals. A value of 'Y' will only be accepted if the SMF record identifier is non-zero. |
| <b>CICS retrieval logging on</b>  | Enter 'Y' to enable SMF logging during object retrieval processing from CICS application programs. Enter 'N' to disable SMF logging for CICS retrievals. A value of 'Y' will only be accepted if the SMF record identifier is non-zero.   |

After all required updates have been made, press 'ENTER'. Any invalid entry will be rejected, and a corresponding message displayed in the top right hand corner of the panel. For additional information regarding the error, press PF1. To obtain help information regarding panel definition

entries, press PF1 once more.

When all updates have been accepted, base database maintenance processing will continue as follows:

- If neither the audit trail or journal archiving facilities have been enabled, the database definition will be updated, and panel OTSN4200 re-displayed.
- If the audit trail facility has been enabled for the first time during database maintenance processing, the Access Method Services (AMS) control panel OTSN4303 will be displayed. This will allow customization of the AMS parameters used to define the database's journal dataset.

Refer to section 3.1.3 for a description of fields on this panel. Enter the appropriate details and press ENTER.

- If the journal archive facility has been enabled for the first time during database maintenance processing, the AMS control panel OTSN4304 will be displayed. This will allow customization of the AMS parameters used to define the database's journal archive database.

Refer to section 3.1.3 for a description of fields on this panel. Enter the appropriate details and press ENTER.

After all panel entries have been accepted, the journal dataset and journal archive databases will be automatically defined and initialized if either facility has been enabled for the first time during database maintenance processing, and the database definition will be updated.

Database maintenance menu panel OTSN4200 will then be re-displayed, and database maintenance processing may be continued as desired.

### **3.2.2 Secondary Index Maintenance.**

The NearArchive secondary index maintenance procedure is invoked by selecting option 2 from the database maintenance menu panel OTSN4200.

Secondary index maintenance menu panel OTSN4220 will then be displayed (see figure 3.6). This panel will allow the following secondary index update procedures to be performed:

1. Update secondary index definition. This will allow amendment of an existing secondary index definition.
2. Add new secondary index. This will allow the definition of a new secondary index for the NearArchive database currently being updated.
3. Delete a secondary index definition. This will allow an existing secondary index definition to be removed, if the secondary index does not contain any entries.

Select the appropriate option on panel OTSN4220. For each option, the identifier of the secondary index to be updated/created must be entered in the 'secondary index number' field. The identifier of a NearArchive secondary index is a numeric value in the range 1-99. This is the identifier used throughout ASM for S/390 to refer to that secondary index.

For options 1 and 3, the identified secondary index must already exist. For option 2, the identified secondary index must not already exist.

```

OTSN4220                      NearArchive V2.5
                             Secondary Index Maintenance
COMMAND ===>
-----
Primary index name ===> _____

                                Option ===> _
                                1 - Update secondary index definition
                                2 - Add new secondary index
                                3 - Delete secondary index

Secondary index number ===> _

```

**Fig 3.6:** Secondary index maintenance menu panel OTSN4220

### **3.2.2.1 Update Secondary Index Definition.**

```

OTSN4223                      NearArchive V2.5
                             Update Secondary Index Definition
COMMAND ===>
-----
Primary index name ===> _____
Secondary index no ===> _

Secondary key specification:

Start position in record ===> _____    0 - index disabled

Automatic indexing controls:

Record no within object ===> _____    0 - no automatic indexing
                                           ALL - index all records within
                                           archived object

```

**Fig 3.7:** Secondary index update panel OTSN4223

The secondary index update panel OTSN4223 will be displayed when option 1 is selected from the secondary index maintenance menu panel OTSN4220.

Existing values for the selected secondary index will be displayed on this panel. Refer to section 3.2.2.3 (add new secondary index) for a description of the displayed fields and their usage. Note that the secondary key length is not displayed on this panel. This value may not be amended after initial definition of the secondary index.

Amend these values as required and press ENTER. Any invalid update will be rejected, and a corresponding message displayed in the top right hand corner of the panel. For additional information regarding the error, press PF1. To obtain help information regarding panel definition entries, press PF1 once more.

When all updates have been accepted, the secondary index definition will be amended, and the secondary index maintenance menu panel OTSN4220 re-displayed.

### **3.2.2.2 Add new secondary index definition.**

OTSN4221		NearArchive V2.5	
		Add Secondary Index Definition	
COMMAND ==>			
-----			
Primary index name ==>		_____	
Secondary index no ==>		_	
Secondary key specification:			
Length ==>		_____	
Start position in record ==>		_____ 0 - index disabled	
Automatic indexing controls:			
Record no within object ==>		_____ 0 - no automatic indexing	
		ALL - index all records within archived object	

**Fig 3.8:** Secondary index definition panel OTSN4221

Panel OTSN4221 will be displayed when option 2 is selected from the secondary index maintenance menu panel OTSN4220.



Enter the appropriate details on this panel and press 'ENTER' to proceed. When all panel entries have been accepted, Access Method Services (AMS) control panel OTSN4222 will be displayed to allow customization of the AMS parameters used to define the secondary index dataset.

Press PF3 to terminate the secondary index definition procedure at any point. Definition of the secondary index will be abandoned, and secondary index maintenance menu panel OTSN4220 re-displayed. The secondary index will not have been defined.

Online help information for each panel may be obtained by pressing PF1.

Panel OTSN4221 contains the following fields. All field entries are mandatory for this panel.

- |                                 |  |
|---------------------------------|--|
| <b>Primary index name</b>       | This field gives the name of the database's primary index dataset, as entered on panel OTSN4000. The contents of this field may not be amended.  |
| <b>Secondary index no</b>       | The identifier of the secondary index to be updated, as entered on the secondary index maintenance menu panel OTSN4220. The contents of this field may not be amended.   |
| <b>Length</b>                   | <p>Mandatory field. Gives the secondary key length for this secondary index.</p> <p>This must be a numeric value in the range 1-252. However, the combined length of the primary and this secondary key cannot exceed 254 characters.</p> <p>The maximum number of DBCS characters permitted is 126 plus the Shift-Out (SO) and Shift-In (SI) bytes. If using DBCS, enter the true count of bytes used and not the count of characters!</p> <p>For example, (number of DBCS characters X 2) + number of SO bytes + number of SI bytes must be less than 254 bytes.</p> |
| <b>Start position in record</b> | Mandatory field. Enter the position of the start of the secondary key within logical record, for automatic secondary key generation during batch archival. This must be a numeric value in the range 0-32760. Note that the first character in a   |

logical record is identified by character position 1, not 0.

Enter '0' to disable the secondary index. Once disabled, no new entries will be created for a secondary index. Any existing entries may be used for object retrieval. A disabled secondary index may be enabled at any subsequent time, via the secondary index update panel OTSN4223.

This field is used in conjunction with the next field on the panel (record no within object) to control automatic creation of secondary index entries during object archival, by the NearArchive batch archival component. The first 's' characters, starting from the character position specified in this entry (where 's' is the secondary key length defined above) in a logical record of a NearArchive object (as specified in the 'record no within object field', or as explicitly identified by a calling application) will be used to automatically create a secondary index entry for that object.

**Record no within object** Mandatory field. Enter the sequence number of the logical record within a NearArchive object which contains the secondary key to be used to index that object.

This field is used in conjunction with the 'start position in record' field to control automatic indexing for this secondary index. See the description of this field above for additional information.

This entry can contain the following values:

*0* no automatic indexing is to be performed for this secondary index. Secondary index entries will be explicitly identified by calling applications during the batch archival procedure.

*1 to 9999999* specifies the sequence number (within NearArchive object) of

the logical record which contains the secondary key to be used when automatically creating a secondary index entry.

*ALL* specifies that all logical records within a NearArchive record will contain a secondary key to be used to automatically create a secondary index entry. Specification of this option should only be made after careful consideration, as it may potentially cause a separate secondary index entry to be created for each logical record within a NearArchive object.

Note that only one secondary index entry will be generated for a NearArchive object if multiple identical secondary keys are identified (automatically or explicitly) for that object.

After all required entries have been made, press 'ENTER'. Any invalid entry will be rejected, and a corresponding message displayed in the top right hand corner of the panel. For additional information regarding the error, press PF1. To obtain help information regarding panel definition entries, press PF1 once more.

When all entries have been accepted, the Access Method Services (AMS) control panel OTSN4222 will be displayed. This panel allows specification of AMS parameters to be used by the NearArchive database administration procedure when automatically defining the secondary index dataset. This is a key-sequenced VSAM dataset.

```

OTSN4222                NearArchive V2.5
                        Access Method Services Control
COMMAND ===>
-----
Primary index name ===> _____
Secondary index no ===> _
More:      - +

Secondary index dataset control:

    Volume serial number ===> _____
    Unit of allocation ===> _____
    Primary allocation value ===> _____
    Secondary allocation value ===> _____
    Data CI size ===> _____
    Index CI size ===> _____

Delete before define? ===> _ (Y/N)

Press ENTER after all details have been entered correctly.

After all secondary index definition parameters have been accepted,
the secondary index will be allocated and initialized.

```

**Fig 3.9:** Access Method Services control panel OTSN4222

Default values will be used where no entry has been made for optional fields on this panel. Entries must always be made for mandatory fields. The description of individual fields below will identify whether an entry is mandatory or optional. Online help information may be obtained by pressing PF1.

The AMS control panel for secondary index definition is illustrated in figure 3.9. It contains the following fields:

**Primary index name** This field gives the name of the database's primary index dataset, as entered on panel OTSN4000. The contents of this field may not be amended.

**Secondary index no** The identifier of the secondary index to be updated, as entered on the secondary index maintenance menu panel OTSN4220. The contents of this field may not be amended.

**Volume serial** Optional entry. Enter the serial number (1-6

**number** alphanumeric, national or special characters) of the disk volume on which the dataset is to be created.

This parameter may be omitted if the volume is to be automatically selected by the operating system.

**Unit of allocation** Optional entry. Enter one of the following values, which is to be used to identify the unit of allocation specified in the primary and secondary allocation value fields:

CYL - allocation is to be performed in cylinders  
TRK - allocation is to be performed in tracks  
REC - allocation is to be performed in records.

If this entry is omitted, a default value of 'CYL' is used (i.e.) primary and secondary allocation units are specified in cylinders.

**Primary allocation value** Mandatory entry. Enter the number of allocation units (as entered or defaulted in the preceding field) to be used for allocation of the primary data extent of the secondary index dataset. Refer to section 3.2.2.3 for information on choosing a primary allocation value.

**Secondary allocation value** Mandatory entry. Enter the number of allocation units (as entered or defaulted in the unit of allocation field) to be used for allocation of each secondary data extent of the secondary index dataset. Refer to section 3.2.2.3 for information on choosing a secondary allocation value.

**Data CI size** Optional entry. Enter the control interval size to be used for allocation of the data component of the dataset. This should be a valid VSAM control interval size.

If omitted, a control interval size of 16k is used for the data component.

**Index CI size** Optional entry. Enter the control interval size to be used for allocation of the index component of the dataset (KSDS only). This should be a valid VSAM control interval size.

If omitted, a control interval size of 2k is used for the index component.

Choosing unique control interval sizes for each secondary index that has been defined for a NearArchive database may improve performance during data archival and retrieval, through the use of exclusive VSAM buffer pools.

**Delete before define?** Optional entry. Enter the value 'Y' if ASM for S/390 should attempt to delete the secondary index dataset before it is defined. Enter 'N' if no deletion request should be issued.

The NearArchive database definition process will proceed as normal if a value of 'Y' is entered here and the dataset did not exist.

If omitted, a value of 'N' is used for this entry.

Press 'ENTER' after all required entries have been made on this panel. After all entries have been accepted, the secondary index will be defined and initialized. After successful completion of the secondary index definition process, the NearArchive secondary index maintenance menu panel OTSN4220 will be redisplayed.

If any error occurs during AMS processing, an error panel will be displayed, allowing viewing of AMS output messages. Error messages should be noted, and the appropriate corrective action taken. The secondary index maintenance menu panel will then be redisplayed to allow retry of the secondary index definition procedure.

### **3.2.2.3 Delete secondary index definition.**

```
OTSN4224                      NearArchive V2.5
                             Secondary Index Deletion
COMMAND ===>
-----
Primary index name ===> _____
Secondary index number ===> ____

    You have requested that the above secondary index definition is
    to be deleted from this NearArchive database.

    Press ENTER to confirm the deletion request.

    Enter END or Press PF3 to cancel the deletion request.

    On completion of the secondary index deletion process, the entry
    for that index will have been removed from the NearArchive database
    definition, and the corresponding secondary index dataset will have
    been physically deleted.
```

**Fig 3.10:** Deletion confirmation request panel OTSN4224

A secondary index definition may be deleted via option 3 on the secondary index maintenance panel OTSN4220, if it does not contain any entries (i.e.) there are no active objects in the database which are indexed by the selected secondary index.

The secondary index deletion confirmation panel (OTSN4224) will be displayed if a secondary index has been accepted for deletion. To continue with secondary index deletion processing, press ENTER. To cancel the secondary index request, enter an 'END' command in the command line and press ENTER, or press the PF3 key. The secondary index maintenance panel will then be re-displayed with an 'UPDATE ABORTED' message.

When secondary index deletion processing has been confirmed, ASM for S/390 will remove the secondary index definition from the database, and automatically delete the associated secondary index dataset, via an internal call to IDCAMS. On successful completion of this process, the secondary index maintenance panel will be re-displayed.

If an error occurs during deletion of the secondary index dataset, an error panel will be displayed to allow viewing of IDCAMS output messages. Error messages should be noted, and the appropriate corrective action taken before retrying the secondary index deletion request.

In these circumstances, it is likely that the secondary index definition will have been successfully removed from the database, but the associated secondary index dataset will not have been deleted. In this case, the secondary index maintenance panel will be re-displayed with a 'DELETION INCOMPLETE' message in the top right-hand corner of the screen.

To complete the secondary index deletion process, it will be necessary to explicitly delete the secondary index dataset, using standard IDCAMS processing or via a TSO 'DELETE' command, when the cause of the initial failure has been identified and rectified. The secondary index dataset will have the name:

```
'hlq'.SECIXnn
```

where 'hlq' is the database high-level qualifier, as assigned during NearArchive database definition processing, and 'nn' is the numeric identifier of the secondary index being deleted (in the range 01-99).

#### **3.2.2.4 Secondary Index Dataset Sizing.**

Each secondary index dataset will contain one record for each combination of secondary and primary keys in the database (e.g.) if secondary key A indexes objects with primary key B, C and D, then three secondary index records (AB, AC and AD) will exist in the secondary index dataset.

Each secondary index record consists of a fixed-length prefix, of length  $(k+s+4)$  ( where 'k' is the database's primary keylength, and 's' is the secondary keylength for that secondary index), and a variable length suffix. The length of the suffix will be 'n/8' bytes, rounded upwards (where 'n' is the number of objects in the database with the associated primary key value).

An approximate estimate may be used during initial secondary index dataset allocation. By monitoring secondary index usage, the primary and/or secondary allocation values may be amended during the life of the database.



### **3.2.3 Storage Level Maintenance.**

The NearArchive storage level maintenance procedure is invoked by selecting option 3 from the database maintenance menu panel OTSN4200. This facility should be invoked in order to perform the following database maintenance activities:

1. Update storage level definition. This will allow amendment of storage level-specific database definition parameters, for a single storage level.
2. Add new storage level. This will allow a new storage level to be defined in the NearArchive database.

On selecting this option, the storage level maintenance menu panel OTSN4230 will be displayed. This panel is illustrated in figure 3.11.

Select the option on panel OTSN4230 which corresponds to the storage level maintenance function to be performed. For each option, the storage level number (0 to 7) must be entered in the 'storage level number' field.

For option 1, the specified storage level must already exist. For option 2, the specified storage level must not already exist.

```
OTSN4230                      NearArchive V2.5
                               Storage Level Maintenance
COMMAND ==>
-----
Primary index name ==> _____
                               Option ==>  _
                                     1 - Update storage level definition
                                     2 - Add new storage level
Storage level number ==>  _
```

**Fig 3.11:** Storage level maintenance menu panel OTSN4230

OTSN4231	NearArchive V2.5	ENTER VALUES
Storage Level Amendment		
COMMAND ==>		
-----		
Primary index name ==> OTAS.V2R5.IVP.INDEX		
Storage level ==> n		
Maximum blockcount per volume ==> _____		
Unit name for tape alloc: A ==> _____ B ==> _____		
C ==> _____ D ==> _____		
Migrate from storage level after ____ yrs ____ days		
Tape usage options:		
ICRC ==> _____ Retention period ==> _____		
Dynamic load balancing ==> - or Expiry date ==> _____		
Volsafe enabled: A ==> - B ==> -		
C ==> - D ==> -		
D/b maintenance options:		
Recycle threshold ==> ____ %		
AUTOUNCAT during recycle ==> -		
Disk compress threshold ==> ____ %		
Duplexing options:		
Duplex during archival ==> -		
Duplex during backup ==> -		
Disk copy options:		
Unit name ==> _____ Primary alloc ==> _____		
Retention ==> _____ days Secondary alloc ==> _____		
Release unused space ==> - Dataset blocksize ==> _____		
Maximum blockcount per segment ==> _____		

**Fig 3.12:** Storage level amendment panel OTSN4231

### **3.2.3.1 Update Storage Level Definition.**

The storage level amendment panel OTSN4231 will be displayed when option 1 (update storage level definition) is selected from the storage level maintenance menu panel OTSN4230.

Existing values for the selected storage level will be displayed on this panel. These values may be amended as required.

Refer to section 3.1.2 for a description of the content and usage of the fields displayed on this panel. Note that the 'database blocksize' field is not present on this panel. The value of this attribute cannot be altered after initial storage level definition.

The following additional field is displayed on the storage level amendment panel:

**Migrate from storage level** Enter the age (in years and days) of objects which are to be eligible for migration from this storage level to the next storage level in the NearArchive database. The age of a NearArchive object is always calculated with reference to the archive date used to index that object.

Objects in this storage level which are older than the limit supplied in this field will be candidates for migration to the next storage level in the database by the NearArchive object management utility OTASP040. Note that when migration is being performed the sum of the values of the “Migrate from storage level” fields in all storage levels up to (but not including) the target storage level is used to determine whether an object is eligible for migration.

Enter a value of 0 years and 0 days if migration from this storage level to the next storage level in the hierarchy is to be disabled.

An entry in this field will only be accepted if the next storage level in the database is already in existence.

Amend the panel entries as required and press ENTER. Any invalid update will be rejected, and a corresponding message displayed in the top right hand corner of the panel. For additional information regarding the error, press PF1. To obtain help information regarding panel definition entries, press PF1 once more.

When all updates have been accepted, the specified storage level definition will be amended, and the storage level maintenance menu panel OTSN4230 re-displayed.

### **3.2.3.2 Add New Storage Level.**

To add a new storage level to a NearArchive database, select option ‘2’ from the storage level maintenance menu panel OTSN4230, specifying the storage level number to be created. The storage level definition panel OTSN4301 will be displayed. This panel will identify the NearArchive database primary index dataset and storage level number being defined.

Refer to section 3.1.2 for a description of the content and usage of fields displayed on this panel. Note that any entry made on this panel will apply only to the storage level being defined. The other storage level(s) in the database will retain existing definitions, and be unaffected by values defined for this new storage level. A new storage level may only be defined if the immediately preceding storage level has previously been defined.

After all required entries have been made, press 'ENTER'. Any invalid entry will be rejected, and a corresponding message displayed in the top right hand corner of the panel. For additional information regarding the error, press PF1. To obtain help information regarding panel definition entries, press PF1 once more.

If this is the first storage level defined for the NearArchive database (other than the base storage level 0), the Access Method Services (AMS) control panel OTSN4306 will be displayed, to allow customization of AMS parameters used to define the NearArchive database's migration control dataset.

```
OTSN4306                      NearArchive V2.5
                                Access Method Services Control
COMMAND ===>
-----
Primary index name ===> _____

Migration dataset control:

    Volume serial number ===> _____
    Unit of allocation ===> _____
    Primary allocation value ===> _____
    Secondary allocation value ===> _____
    CI size ===> _____

Delete before define? ===> _ (Y/N)

Press ENTER after all details have been entered correctly.
```

**Fig 3.13:** Access Method Services control panel OTSN4306

The AMS control panel for migration control dataset definition is illustrated in figure 3.13. The migration control dataset is automatically defined during definition of the first storage level for a NearArchive database, after the base storage level 0. This dataset is an entry-sequenced VSAM dataset.

A description of the content and usage of all fields displayed on panel OTSN4306 may be found in section 3.1.3.

The migration control dataset for a NearArchive database will consist of fixed-length records, each of length  $k+32$  bytes, where 'k' is the database's primary keylength. Each record is used to identify a single NearArchive object. The dataset must be large enough to hold as many records as the maximum number of NearArchive objects which are to be migrated between storage levels in a single execution of the NearArchive object management utility OTASP040.

Press 'ENTER' after all required entries have been made on this panel. After all entries have been accepted, the new storage level will be defined.

After successful completion of the storage level definition process, the NearArchive storage level maintenance menu panel OTSN4230 will be redisplayed.

### **3.3 Browse a NearArchive Database Definition.**

---

To browse an existing NearArchive database definition, enter the name of the database's primary index dataset on the main database maintenance facility menu OTSN4000, and select option '1' (browse database definition).

The following series of panels will be displayed:

**OTSN4100** - browse database definition. This panel will display base database definition parameters which apply to all defined storage levels in the NearArchive database.

**OTSN4101** - browse secondary index definition. One panel will be displayed for each secondary index in the NearArchive database. If no secondary indexes have been defined, no OTSN4101 panel will be displayed.

**OTSN4102** - browse storage level definition. One panel will be displayed for each storage level in the NearArchive database. These will give details of the definition of each storage level in the database.

**OTSN4103** - volume recovery control. One panel will be displayed after each storage level panel display, if there are one or more recovery control entries in existence for that storage level.

This panel will contain one row for each recovery control entry. Entries may be browsed backwards and forwards using the PF7 and PF8 keys.

All fields on each of the above panels will be protected from update. No amendment to database definition values may be made via the database browse facility.

Press ENTER to continue displaying the sequence of database definition panels. Press PF3 at any point to terminate the browse process. Press PF1 to obtain help information about the content of any individual panel.

### **3.4 Recovery Control Processing.**

---

The NearArchive database recovery control facility allows the following processing to be performed on a NearArchive database:

1. Identification of a tape sequence number, or range of sequence numbers, in any storage level in the database, which have become unavailable, either due to media failure or loss of the volumes for some other reason, for which retrieval requests are to be automatically satisfied from a designated backup copy.

A backup copy identifier ('B', 'C' or 'D') must be specified for each sequence number, or range of sequence numbers. See section 4.7 for a description of NearArchive tape cartridge backup identification.

After completion of recovery control entry processing, the appropriate backup tape cartridge volume will be automatically used to satisfy all requests for retrieval from the specified tape sequence number(s).

Recovery control entries are also used by the NearArchive database recovery utility OTASP130 to recover primary copies of a tape cartridge volume. Refer to section 7.6 for a description of this utility, and of its use of recovery control entries.

Recovery control entries may be manually deleted via this facility, if the primary copy of a tape cartridge volume becomes available once again. Tape sequence numbers are automatically deleted by the NearArchive database recovery utility after successful recovery of a primary tape cartridge volume.

2. Identification of a tape sequence number, or range of sequence numbers, which are to be marked as 'unavailable'. ASM for S/390 will return a 'resource unavailable' condition (return code = '21', supplementary code 1 = 2) to calling applications when a batch or online retrieval request is issued for an object contained on a volume in this status. To place a range of volumes into 'unavailable' status, a copy identifier of 'X' should be specified for that range.

3. Enabling/disabling of disaster recovery processing. See section 4.7 for a description of disaster recovery processing options for a NearArchive database.

This facility requires the specification of a backup copy identifier ('B', 'C' or 'D') during disaster recovery enablement. When enabled, all database retrieval operations will be automatically performed by ASM for S/390 from the database copy identifier specified during this process. Disabling disaster recovery will cause retrievals to be performed from the primary database copy as normal.

Recovery control processing is invoked by selecting option 4 (recovery control) from the main NearArchive database maintenance facility menu OTSN4000, and entering the name of the NearArchive database's primary index dataset. The recovery control panel OTSN4400 will then be displayed. This panel is illustrated in figure 3.14.

OTSN4400	NearArchive V2.5
	Recovery Control
COMMAND ===>	
-----	
Primary index name ===>	_____
Option ===>	_
	1 - Recover individual volume(s)
	2 - Set disaster recovery on
	3 - Unset disaster recovery
For option 1 only:	
Enter storage level in above	
database for which volume	
recovery is to be performed ===>	_

**Fig 3.14:** Recovery control panel OTSN4400

Select option 1 from this panel to invoke volume recovery or volume unavailable processing for a single tape sequence number, or range of sequence numbers. The storage level to be processed must also be entered on this panel.

Refer to section 4.7.1 for a description of tape dataset naming standards in a NearArchive database. This information may be used to identify the storage level in which tape cartridge volumes to be recovered are held.

Select options 2 or 3 to enable or disable disaster recovery processing for a NearArchive database.

**3.4.1 Volume recovery/unavailable volume processing.**

OTSN4401

NearArchive V2.5  
Volum Recovery Control

COMMAND ===>

-----

Primary index name ===> \_\_\_\_\_  
Storage level ===> \_

Sel	Start seq no	End seq no	Copy id
-	_____	_____	-
-	_____	_____	-
-	_____	_____	-
-	_____	_____	-
-	_____	_____	-
-	_____	_____	-
-	_____	_____	-
-	_____	_____	-
-	_____	_____	-
-	_____	_____	-
-	_____	_____	-
-	_____	_____	-
-	_____	_____	-
-	_____	_____	-

**Fig 3.15:** Volume recovery control panel OTSN4401

Individual volume recovery processing is invoked by entering option ‘1’ from the recovery control panel OTSN4400. Panel OTSN4401 (illustrated in figure 3.15) will be displayed. This panel will provide details of any existing volume recovery control entries for the NearArchive database storage level being processed. One line will be displayed for each current



entry. If no entries currently exist a single line containing underscores ('\_') will be displayed. PF7 and PF8 may be used to browse backwards and forwards through entries if more entries exist than can be displayed on a single panel. Panel OTSN4401 will contain the following fields:

**Primary index name** This field gives the name of the database's primary index dataset, as entered on panel OTSN4000. This identifies the NearArchive database being processed. The contents of this field may not be amended.

**Storage level** This indicator identifies the storage level being processed, as entered on recovery control panel OTSN4400. The contents of this field may not be amended.

**Sel** This field is used for identifying the volume recovery control entry(ies) which are to be processed. The following characters may be entered:

**D** delete a volume recovery control entry. Once deleted, data retrieval from all tape sequence numbers specified in that entry will be satisfied from the primary database copy.

**I** insert a volume recovery control entry. This will cause the next panel display to contain an additional entry of underscores ('\_'). Start and end sequence numbers in this entry may then be updated to add the new volume recovery control entry as desired.

Any other character entered in this column will be rejected.

Multiple entries per panel display may be selected for processing.

**Start seq no and end seq no** These fields display the start and end sequence numbers within storage level of the volumes in that recovery control entry.

All tape cartridge volumes within a single storage level in a NearArchive database contain a single

tape dataset. Each dataset is allocated a sequence number, starting at 1 and continuing in ascending numerical sequence up to a maximum value of 65535 (the maximum number of volumes per storage level). A dataset's sequence number is used when generating the dataset name.

NearArchive uses the following convention when allocating names for tape datasets within a NearArchive database:

'HLQ'.xy.Gnnnnn

where 'HLQ' is the high-level dataset name qualifier used for generating system dataset names for the NearArchive database (as defined during database definition processing)

'x' is the copy identifier ('A','B','C' or 'D')

'y' is the storage level identifier ('0' or '1')

and 'nnnnn' is the dataset sequence number (in the range 00001 - 65535).

Note that the 'xy' dataset name qualifier is not present for primary ('A') copy datasets in storage level 0 of a NearArchive database.

The start and end sequence numbers used in a volume recovery control entry may be derived from the name of the dataset which is contained on the tape cartridge volume for which recovery control processing is enabled.

Any update of start and end sequence numbers on this panel will be rejected by ASM for S/390. The only exception to this is for entries consisting of underscore ('\_') characters. A single entry of this type will be displayed:

- if there are no current volume recovery controls in this storage level of the database
- if a selection of 'I' (insert entry) has previously

been entered.

In this case, the underscore characters may be updated with the start and end sequence numbers of the new recovery control entry. Each sequence number must contain 1-5 numeric characters, and be in the range 1 to 65535.

If entered, the end sequence number must not be less than the start sequence number. If the end sequence number is not entered, it will be set equal to the start sequence number.

**Copy id** This field specifies the copy identifier ('B', 'C', 'D' or 'X') of the volume recovery control entry. For copy identifiers 'B', 'C' or 'D', this identifies the backup copy to be used for retrieval of data from the tape sequence numbers in the entry, and for recovery of primary database volumes. See section 4.7.1 for a description of NearArchive tape backup copies and their identifiers. For copy identifier 'X', this entry indicates that the associated range of volumes is to be placed into 'unavailable' status. ASM for S/390 will return a 'resource unavailable' condition to any application which issues a retrieval request for an object on any of these volumes.

Any update of copy identifier on this panel will be rejected by ASM for S/390. The only exception to this is for entries consisting of underscore ('\_') characters, as described for the start and end sequence number fields above.

As before, the underscore characters may be updated with the copy identifier of the new recovery control entry. The copy identifier must consist of the characters 'B', 'C', 'D' or 'X'.

After entering the appropriate selection character(s) to identify entries to be processed, and/or adding a new entry by updating the underscore characters, press ENTER to process the selections.

Any invalid selection or entry will be rejected, and a corresponding message displayed in the top right hand corner of the panel. The cursor

will be positioned on the line corresponding to the invalid entry. All requests made on this panel prior to the invalid entry will have been processed. For rejected entries, make the appropriate corrections and press ENTER.

After successful processing of all requests, the volume recovery control panel OTSN4401 will be redisplayed. Entries will be updated with any changes which have been made.

Note from the above description that volume recovery entries may only be deleted or inserted. Existing entries may not be updated (e.g.) by changing the range of sequence numbers or the copy identifier. To amend an existing entry, it must first of all be deleted (with selection character 'D'), and a new entry, containing the values required, inserted (with selection character 'I').

After all volume recovery control entry processing has been completed, press PF3 to return to the recovery control menu panel OTSN4400.

### **3.4.2 Set Disaster Recovery.**

To enable disaster recovery processing for a NearArchive database, select option 2 from the recovery control menu panel OTSN4400. The disaster recovery request panel OTSN4402 will then be displayed. This panel allows confirmation or cancellation of the disaster recovery enabling request.

To confirm the request, enter the identifier of the NearArchive database backup copy to be used during disaster recovery processing ('B', 'C' or 'D'), and press ENTER. Once accepted, disaster recovery processing will be enabled for the NearArchive database, and all object retrievals for that database will be satisfied from the database backup copy whose identifier was entered above. The recovery control menu panel OTSN4400 will then be re-displayed. Note that setting of disaster recovery processing for a database will over-ride recovery control entries for individual volumes in any storage level in that database.

To cancel the request from panel OTSN4402, press PF3. Disaster recovery processing will not be enabled for the NearArchive database, and the recovery control menu panel OTSN4400 will be re-displayed.

Refer to section 4.7 of this manual for a full description of NearArchive backup and disaster recovery processing.

### **3.4.3 Unset Disaster Recovery.**

To disable disaster recovery processing for a NearArchive database, select option 3 from the recovery control menu panel OTSN4400, and press ENTER. Disaster recovery processing must already be enabled for the database.

When the request is accepted, disaster recovery will be disabled for that database, and the recovery control menu panel OTSN4400 will be re-displayed.

After disabling disaster recovery for a NearArchive database, all object retrievals for that database will be satisfied from the primary database copy.

Refer to section 4.7 of this manual for a full description of ASM for S/390 backup and disaster recovery processing.

## **3.5 Pre-v2.5 Database Conversion Processing.**

All databases created with versions of NearArchive prior to v2.5 must be converted for use with this version of the product. Any attempt to access an unconverted database will be rejected by ASM for S/390 in the following manner:

- database open calls issued to NearArchive via the batch application program interface will be rejected with a return code of '91', a supplementary code 1 value of 0 and supplementary code 2 value of 1.
- application calls to NearArchive via the CICS application program interface will be rejected with a return code of '91', a supplementary code 1 value of 0 and supplementary code 2 value of 1.
- NearArchive CICS control region initialization (via transaction OT23) will fail if no primary index file created or converted for use with NearArchive v2.5 is present in the file control table (FCT).
- NearArchive utilities will terminate with condition code 12 and issue an error message to the error report print file.
- database administration processing in TSO/ISPF will fail with message OTS49008 when a pre-v2.5 database is selected for processing.

Conversion of a pre-v2.5 database is performed using the TSO/ISPF database administration dialog. To convert a pre-v2.5 database, the following sequence of actions should be performed:

1. Enter the primary index name of the database to be converted on the database administration entry panel OTSN4000, and select **option 5** (pre-v2.5 database conversion).

If the database has not already been converted, the database conversion confirmation panel OTSN4500 will then be displayed. An error message will be displayed in the top-right corner of panel OTSN4000 if a conversion request is entered for a database which has already been converted. Processing may then continue normally.

2. To abort the conversion process, press the PF3 key. The entry panel OTSN4000 will then be re-displayed, and processing may continue normally.

3. As the database space management dataset(s) will need to be re-created during the conversion process, enter on panel OTSN4500 the serial number of the disk volume on which the converted space management datasets are to be created. Leave this field blank if the datasets are to be DFSMS-managed.

4. To confirm the conversion process, press the ENTER key. The conversion procedure will then be executed. This will involve the following steps:

- A new space management file will be automatically allocated for each defined storage level in the database. This will be a VSAM KSDS (unlike the space management files for previous releases, which are VSAM ESDS files). The new space management file will have an identical name to that of the existing file, but the low-level qualifier will be changed to 'SPACENLx' (where 'x' is the storage level). The new space management file will have a variable record length.
- All records in each existing space management file will be copied to the corresponding new space management file. Records will be re-formatted during this process.
- The existing space management files will be renamed (by changing the low-level qualifier from 'SPACEMLx' to 'SPACEOLx'), and the new space management file will be renamed to that of the live file (by changing the low-level qualifier from 'SPACENLx' to 'SPACEMLx').

- Finally, control records in the primary index file will be reformatted, and the file updated.

On successful conclusion of this procedure the database will have been fully converted for use with v2.5 of ASM for S/390, and panel OTSN4000 will be re-displayed with a successful completion message. There are no other mandatory conversion requirements. However, if the maximum number of objects per unique primary key is likely to exceed 1015, an additional conversion step to increase the maximum primary index record size should be performed, as discussed in section 3.5.1 below.

Note that the old (pre-conversion) space management files will still remain in existence (with a low-level qualifier of 'SPACEOLx'). These files may be manually deleted at any subsequent time, when it has been determined that database conversion has completed successfully.

If the database conversion fails at any stage during the above process, an error message will be displayed on the terminal, and conversion processing will be aborted. In the case of failure during allocation or renaming of the space management file, the condition code generated from the internal IDCAMS request will be displayed on error panel OTSN4305. The user will then be given the option of browsing output from the IDCAMS processing.

After the cause of the failure has been identified and rectified, the database conversion process may be re-executed at any subsequent time. There are no special restart or recovery considerations.

### **3.5.1 Increase maximum primary index record size.**

If the maximum number of objects in the converted database which have the same primary key (but different archive dates) currently exceeds 1015 (or is likely to exceed this figure at some future date), the maximum record size of the database's primary index file will need to be increased.

Member DB25CONV in the distributed SAMPLIB dataset contains JCL to perform this conversion process. Edit the job by adding a job card, and replacing all occurrences of the character string 'index.name' in the job with the name of the primary index file for the database being converted. If the converted index is to be created on a different volume from that containing the original index file, remove the comment markers from the VOL parameter in the DEFINE CLUSTER command, and replace the string 'xxxxxxx' with the serial number of the volume on which the converted file is to be created. Then submit the job.

The following JCL is provided in SAMPLIB member DB25CONV:

```
/*  
//STEP1      EXEC  PGM=IDCAMS  
/*  
//SYSPRINT DD   SYSOUT=*  
/*  
//SYSIN      DD   *  
  
      DELETE index.name.NEW  
  
      SET LASTCC = 0  
  
      DEFINE CLUSTER (NAME(index.name.NEW) -  
                      MODEL(index.name) -  
                      /*      VOL(xxxxxx) -      */  
                      RECSZ(16380 65524) SPANNED)  
  
      IF MAXCC = 0 THEN -  
        REPRO IDS(index.name) -  
          ODS(index.name.NEW)  
      IF MAXCC = 0 THEN -  
        DELETE index.name  
  
      IF MAXCC = 0 THEN -  
      DO  
        ALTER  index.name.NEW      -  
          NEWNAME(index.name)  
        ALTER  index.name.NEW.INDEX -  
          NEWNAME(index.name.INDEX)  
        ALTER  index.name.DATA      -  
          NEWNAME(index.name.DATA)  
  
      END
```

On successful completion of the job, the database will be available for processing with version 2.5 of ASM for S/390. The maximum number of objects with the same primary key which may be stored in the converted database will range from 2039 (for a primary keylength of 254 characters ) to 2047 (for a primary keylength of 1 character).

Note that NearArchive databases used by Oracle's NearOAM product (for storage of migrated OAM objects) have unique primary keys for each object in the database (and hence only one object per primary key). It is therefore not necessary to perform the optional database conversion step described in this section for NearOAM storage group databases.



---

## CHAPTER 4

---

# *The NearArchive Programming Interface*

This page is intentionally left blank

ASM for S/390 supplies a high-level interface which enables customer application systems to invoke NearArchive functions from batch, TSO and CICS programs. This is activated via a standard CALL interface for batch or TSO programs, and a CICS command-level 'LINK' interface for online programs. A parameter block in the CALL interface or CICS communication area is used to communicate requests and responses.

A new application program interface (API) structure was shipped with version 2.4 of the product. This interface should be used when developing new ASM for S/390 applications, in order to avail of any enhanced product functionality and to ensure upward compatibility with future releases of the product. API enhancements shipped with version 2.5 can only be invoked via the new API.

The API shipped with releases of the product prior to version 2.4 continues to be supported in this release. ASM for S/390 applications coded to use this API will not need any modification to continue to function correctly with version 2.5 of the product. A specification of the previous API is supplied in appendix D of this manual for ongoing program maintenance requirements.

Note carefully when developing new ASM for S/390 applications, or modifying existing applications, that invocation of NearArchive services using a combination of old and new APIs should **not** be performed from within the same calling program. Unpredictable results may occur if this is done.

This chapter discusses the invocation of NearArchive using the standard application program interface.

## 4.1 Invoking NearArchive.

---

The following NearArchive services may be invoked by a batch or CICS application program:

- Connect to a NearArchive database
- Query the primary or secondary index of a NearArchive database.
- Retrieve an object from a NearArchive database.
- Store an object in a NearArchive database.
- Create a secondary key for a stored NearArchive object.
- Delete an object from a NearArchive database.
- Commit updates to a NearArchive database.

- Disconnect from a NearArchive database.

NearArchive services are invoked from batch or TSO programs by issuing standard CALL statements to the NearArchive module 'OTASP100'. This module is supplied on the distributed NearArchive load library, and must be linked in to the calling program at link-edit time, to satisfy the external reference. The CALL statement should pass the standard NearArchive parameter interface block, as specified in section 4.2.

NearArchive services are invoked from CICS programs using the CICS 'LINK' command, as follows:

```
EXEC CICS LINK PROGRAM('OTASP200')
          COMMAREA (PARMAREA)
          LENGTH (nn)
          END-EXEC.
```

where 'PARMAREA' is the user-defined storage area containing the parameter interface block used for communication between NearArchive and the calling application and 'nn' is the length of the parameter interface block passed to NearArchive. The communication area passed with this command should contain the standard NearArchive parameter interface block, as specified in section 4.2.

## 4.2 The NearArchive program interface.

The NearArchive application program interface (API) is used by all types of applications (batch, TSO and CICS) for invocation of ASM for S/390 services. The standard format of the application program interface parameter area is:

Offset	Length	Type	Description
0	2	char	Request code.
2	2	char	Return code.
4	2	bin	Supplementary request/return code 1.
6	2	bin	Supplementary request/return code 2.
8	4	char	Database name.
12	4	ptr	Pointer to field containing primary key of archived object.
16	8	char	Object archival date (in format YYYYMMDD).
24	8	char	Temporary Storage Queue (TSQ) name

Offset	Length	Type	Description
			to be used for data retrieval. (CICS Only)
32	4	bin	Object start position (partial CICS retrieval requests only).
36	4	bin	Object length (partial CICS retrieval requests only).
40	10	-	Reserved for future use
50	2	bin	Length (L) of the following object data buffer.
52	L	char	Object data buffer.

Any variations in field usage for specific request codes will be identified in the description of those request codes in the following sections.

The API may also be used for management of double-byte character set (DBCS) data objects. NearArchive DBCS support falls into three areas:

1. All data records written to the archive database may contain any combination of SBCS or DBCS characters with or without shift control bytes. ASM for S/390 does not process the contents of any record.
2. The program interface parameter area must always be specified in SBCS.
3. The primary and/or secondary keys may contain SBCS and/or DBCS values. To avoid problems printing our reports, your primary and/or secondary keys must include the necessary shift-out (SO) and shift-in (SI) control bytes.

Figures 4.1 to 4.3 illustrate the CALL statement coding required for Assembler, COBOL and PL/1 batch programs. Sample COBOL code for each type of NearArchive request is supplied in Appendix B.

Note that sufficient storage must be allocated in the data buffer field in the parameter area, either via GETMAIN processing or via pre-allocation in working storage, to hold the largest record contained in any NearArchive object that will be processed by the application. The illustrated examples are coded to handle a maximum record length of 256 bytes. Any unused field in the parameter interface area should be set to spaces (for character fields) or binary zeroes (for all other field types).

```

PARMBLK DSECT                                NEARARCHIVE PARAMETER AREA:
PREQCODE DS CL2                               REQUEST CODE
PRETCODE DS CL2                               RETURN CODE
PSUPPCD1 DS XL2                               SUPPLEMENTARY CODE 1
PSUPPCD2 DS 0XL2                             SUPPLEMENTARY CODE 2
PUPPER DS C                                  UPPER BYTE OF PSUPPCD2
PLOWER DS C                                  LOWER BYTE OF PSUPPCD2
PDBNAME DS CL4                               DATABASE NAME
PKEYADDR DS AL4                              --> PRIMARY KEY
PDATE DS CL8                                ARCHIVE DATE (YYYYMMDD)
PTSQNAME DS CL8                              TSQ NAME
POBJSTRT DS F                                OBJECT START RECORD
POBJRCNT DS F                                OBJECT RECORD COUNT
PRESVD1 DS CL10                             RESERVED
PLENGTH DS H                                DATA BUFFER LENGTH
PDATA DS CL256                              DATA BUFFER
...
USERPROG CSECT
...
L R5,@PARMBLK
CALL OTASP100,((R5))
...
@PARMBLK DS AL4

```

**Fig 4.1:** Assembler coding for 'OTASP100' call

```

WORKING-STORAGE SECTION.

01 OTAS-PARAMETER-BLOCK.
03 OTAS-REQ-CODE PIC 99.
03 OTAS-RET-CODE PIC 99.
03 OTAS-SUPP-CODE-1 PIC S9(4) COMP.
03 OTAS-SUPP-CODE-2 PIC S9(4) COMP.
03 OTAS-QUERY REDEFINES OTAS-SUPP-CODE-2.
05 OTAS-UPPER-BYTE PIC X.
05 OTAS-LOWER_BYTE PIC X.
03 OTAS-DATABASE-NAME PIC X(4).
03 OTAS-KEY-POINTER POINTER.
03 OTAS-ARCHIVE-DATE.
05 OTAS-ARCHIVE-DATE-YYYY PIC 9(4).
05 OTAS-ARCHIVE-DATE-MM PIC 99.
05 OTAS-ARCHIVE-DATE-DD PIC 99.
03 OTAS-TSQ-NAME PIC X(8).
03 OTAS-OBJECT-START PIC S9(8) COMP.
03 OTAS-OBJECT-REC-COUNT PIC S9(8) COMP.
03 OTAS-RESERVED-1 PIC X(10).
03 OTAS-BUFFER-LENGTH PIC S9(4) COMP.
03 OTAS-DATA-BUFFER PIC X(256).
...
PROCEDURE DIVISION.
.....
CALL 'OTASP100' USING OTAS-PARAMETER-BLOCK.

```

**Fig 4.2:** COBOL coding for 'OTASP100' call

```

DCL 01 OTAS-PARAMETER-BLOCK,
      03 OTAS-REQ-CODE          CHAR(2),
      03 OTAS-RET-CODE          CHAR(2),
      03 OTAS-SUPP-CODE-1       BIN FIXED(15),
      03 OTAS-SUPP-CODE-2       BIN FIXED(15),
      03 OTAS-QUERY DEF OTAS-SUPP-CODE-2,
            05 OTAS-UPPER        CHAR(1),
            05 OTAS-LOWER        CHAR(1),
      03 OTAS-DATABASE-NAME     CHAR(4),
      03 OTAS-KEY-POINTER       POINTER,
      03 OTAS-ARCHIVE-DATE,
            05 OTAS-ARCHIVE-DATE-YYYY CHAR(4),
            05 OTAS-ARCHIVE-DATE-MM   CHAR(2),
            05 OTAS-ARCHIVE-DATE-DD   CHAR(2),
      03 OTAS-TSQ-NAME          CHAR(8),
      03 OTAS-OBJECT-START       BIN FIXED(31),
      03 OTAS-OBJECT-REC-COUNT   BIN FIXED(31),
      03 OTAS-RESERVED-1         CHAR(10),
      03 OTAS-BUFFER-LENGTH     BIN FIXED(15),
      03 OTAS-BUFFER             CHAR(256);

DCL OTAS-PARAMETER-PTR          POINTER;
DCL OTAS-PARAMETER-STR CHAR(308) BASED(OTAS-PARAMETER-PTR);
      .....
      CALL OTASP100 (OTAS PARAMETER STR);

```

**Fig 4.3:** PL/I coding for 'OTASP100' call

## 4.3 NearArchive Request Codes.

The following NearArchive request codes are supported via the standard application program interface:

<b>Request Code</b>	<b>Function</b>
40	Connect to a NearArchive database for input, output or input-output processing (batch or TSO applications only).
41	Query the primary or secondary index for a NearArchive database. Supports single and double byte character sets.
42	Retrieve the first/next record from a NearArchive object (batch or TSO applications only).
43	Retrieve a complete NearArchive object (CICS applications only).
44	Write the first/next record in a data object to a NearArchive database (batch or TSO applications only).

- 46 Add a secondary index entry for a NearArchive object (batch or TSO applications only).
- 47 Delete a NearArchive object.
- 48 Commit updates to a NearArchive database (batch or TSO applications only).
- 49 Disconnect from a NearArchive database (batch or TSO applications only).

Each of the following types of request will be discussed individually in the remainder of this section of the manual. Issues which are specific to batch or TSO archival and retrieval applications are discussed separately in chapter 5 of the manual. Issues which are specific to CICS applications are discussed separately in chapter 6 of the manual.

#### **4.3.1 Request Code 40 – Connect to a NearArchive Database.**

A ‘database connect’ call must be issued by a batch or TSO application prior to accessing a NearArchive database for input, output or input-output processing. ‘Database connect’ calls never need to be issued by CICS applications. Connection to a NearArchive database is established for all CICS applications using CICS table entry definitions.

Call parameters should be set as follows:

**Request code:** This must be set to the value ‘40’ for all database connect requests.

**Return code:** Not used on entry to NearArchive. This field will contain a 2-character numeric value on return from NearArchive indicating the result of the call. Refer to chapter 8 of the manual for a description of all possible return codes.

**Supplementary code 1:** Must be set to 0 on entry to NearArchive. On return may contain additional information for an error return code.

**Supplementary code 2:** Must be set to 0 on entry to NearArchive. On return may contain additional information for an error return code.



- Database name:** Not used for this request type.
- Primary key pointer:** Not used for this request type.
- Object archive date:** Not used for this request type.
- TSQ name:** Not used for this request type.
- Object start position:** Not used for this request type.
- Object length:** Not used for this request type.
- Data buffer length:** Mandatory for this request type. Indicates the number of entries in the database name list contained in the data buffer field.
- For version 2.5 this field must contain the value 1.
- Data buffer:** On entry, contains the database name list entries indicating the name of each database to which access is required and the type of access required. Each entry is 24 bytes in length. The number of entries in the list must be equal to the value set in the data buffer length field.

For version 2.5, the data buffer must contain a single database name list entry. This 24-byte entry should have the following format:

Offset	Length	Type	Description
0	4	Char	Name of database to be connected. This field specifies the last 4 characters of the DD name of the database primary index, as specified in the JCL used to execute the batch or TSO application (see chapter 5 for a description of batch and TSO JCL requirements). The first 4 characters of the DD name must always be set to the value 'OTAS'.

A default value of 'V100' (DD

Offset	Length	Type	Description
			name of OTASV100) is used for the database name if this field is set to spaces or binary zeroes.
4	2	Bin	<p>Access request type:</p> <ol style="list-style-type: none"> <li>1 Open for input processing. This request will only allow database query and object retrieval requests to be issued by the application.</li> <li>3 Open for input-output processing. This request will allow database query and object retrieval and deletion requests to be issued by the application.</li> <li>4 Open for normal output processing. This request will allow secondary key identification, commit and object storage and deletion requests to be issued by the application. It will cause NearArchive to start writing data to the last tape cartridge volume in the database.</li> <li>5 Open for normal output processing on a new tape cartridge volume. This request will allow secondary key identification, commit and object storage and deletion requests to be issued by the application. It will cause NearArchive to start writing data to a new tape cartridge volume in the database. Use this request code if it is required to control physical data location for retrieval performance</li> </ol>

Offset	Length	Type	Description
			reasons.
		6	Open for restart output processing. This request will allow secondary key identification, commit and object storage and deletion requests to be issued by the application. This request must be used to connect to a NearArchive database which is in an 'uncommitted' state due to the failure of an earlier update operation, or because the database was last closed without committing updates.  NearArchive will reject any other type of connect request if a database was not closed normally and all updates committed during the last update procedure.
6	2	Bin	Allocation recovery retry limit. This field should contain a binary value which sets the number of times tape allocation is retried after allocation recovery has been entered (due to a tape device being unavailable). A default value of 30 is used if this field is set to spaces or binary zeroes.
8	8	Char	Archive over-ride date. This field is only needed when connecting to a database with access code 6 (open for restart output processing). It is used to specify the date (in format YYYYMMDD) to be used to over-ride archive dates supplied in the application's call parameter interface for request code '44'

Offset	Length	Type	Description
			(write object data record) calls, during rerun processing (see usage note 3 below for a discussion of this facility).
			The NearArchive rerun archive over-ride date facility is not enabled if this field is set to spaces or binary zeroes. If specified, it must contain a valid date between 1 <sup>st</sup> January 1950 and 31 <sup>st</sup> December 2049.
16	4	Char	<p>This field is only needed when connecting to a database with access code 1 (input access) or 3 (input/output access). It is used to specify the last 4 characters of the DD name to be used by NearArchive when dynamically allocating a tape or disk dataset for object retrieval. The first four characters of the DD name are always set to 'OTAS'.</p> <p>A default value of 'T100' is used if this field is set to spaces or low-values.</p>
20	2	Bin	Storage level to be updated (update access only).
22	2	-	Reserved for future use.

### **Usage Notes**

1. Applications issuing a database connect call for output processing should be capable of establishing whether a normal or rerun open request is to be submitted (e.g. from an EXEC or other parameter value).
2. NearArchive will reject a normal output connect (access codes 4 or 5) request for a database which is in an uncommitted state, by setting a return code of '15'. Conversely, a rerun output connect (access

code 6) request for a database which was updated successfully during the previous archival procedure, will be rejected by NearArchive with a return code of '18' (see chapter 8 for a full description of NearArchive return codes).

3. Use the 'archive over-ride date' field in the rerun output connect request (access code 6), if the archive date specified by the calling application in the parameter interface area for 'write object record' calls (request code '44') during rerun processing differs from that specified for the same calls during initial (failed) processing.

This can occur, for example, if an application uses the system run date to identify NearArchive objects. Objects archived during a failed archival run will be identified by the run-date of the failed run; re-archiving these objects during rerun processing will cause them to be identified with a new archive date.

NearArchive will treat these objects as new archived items and generate new index entries for them leaving the old index entries, created during the failed archival run, in place. These old index entries will no longer be valid, and any attempt to access data via them will fail.

The archive over-ride date is provided to allow the date supplied by such applications in 'write object record' calls during rerun processing to be automatically over-ridden, so that the original archival date may be re-specified. This will ensure correct re-use of index entries during a rerun, and avoid errors due to redundant index entries.

4. An application may have a maximum of two active NearArchive connections at any one time – one for input processing and one for input/output or output processing. If more than one concurrent connection request is required, ensure that the database name and tape DD name fields in each connect request (at offsets 0 and 16 respectively in the database name list entry) are different.

Issuing more than one NearArchive connection request concurrently will mean that there must be one DD card in the JCL deck used to execute the batch or TSO procedure for each database connection. Each card must identify the primary index of the NearArchive database to be connected, using a DD name of 'OTASxxxx' (where 'xxxx' is the database name specified in the connection request).

Note that for concurrent database connections, the database being accessed need not be different for each connection. If the same

primary index dataset is specified in each of the database definition DD cards present in the JCL deck, then a single NearArchive database may be connected for input processing (to query or retrieve objects from a NearArchive database) and for output or input-output processing (to store or delete objects in the same NearArchive database).

5. The database storage level in which objects are to be stored is set during connect processing (in the 2-byte field at offset +20 in the database name list). This may take the value of any storage level currently defined in the database. Note that only one storage level may be updated during any single database connection. Storage of objects in another storage level in the database requires disconnection from the database and re-connection to the new storage level.

#### **4.3.2 Request Code 41 – Query a NearArchive Index.**

A 'query index' request is issued by a batch, TSO or CICS application to obtain a formatted list of index entries (primary or secondary) for a specified primary or secondary key value.

For batch or TSO applications, the formatted list is returned in the data buffer field of the NearArchive parameter interface area. The maximum number of formatted entries which may be accepted by the application is set in the 'data buffer length' field when the call is issued. On return from NearArchive, this field will hold the actual number of formatted entries present in the data buffer field in the parameter block. A return code of '00' and supplementary code 1 value of 4 is issued when there are more entries to be returned. Further query requests may then be issued, specifying the same query criteria, to retrieve the next set of formatted index entries for that query. Formatted index entries will be returned in the data buffer in the sequence requested by the calling application.

For CICS applications, formatted entries are returned in the auxiliary temporary storage queue (TSQ) specified in the call interface parameter area. One item will be present in the TSQ for each index entry which satisfied the query. Formatted index entries will be held in the TSQ in the sequence requested by the calling application.

An extended query interface has been introduced with version 2.5. This interface will return additional index information for an archived object, or may be used to return the serial number(s) of the tape volume(s) on which the object is stored. Invocation of the existing (compact) interface, extended interface or volume serial number query is controlled via the contents of the supplementary code 1 field in the interface.

Call parameters should be set as follows:

**Request code:** This must be set to the value '41' for all database index query requests.

**Return code:** Not used on entry to NearArchive. This field will contain a 2-character numeric value on return from NearArchive indicating the result of the call. Refer to chapter 8 of the manual for a description of all possible return codes.

**Supplementary code 1:** On entry, identifies the type of query, as follows:

- 0 - query primary index (compact format).
- nn - query secondary index 'nn' (compact format).
- 100 - query primary index (extended format)
- 1nn - query secondary index 'nn' (extended format)
- 200 - volume serial number query

On exit, this field may contain a value giving additional information after an exception condition.

**Supplementary code 2:** On entry, identifies the search key length and sort sequence in which formatted index entries are to be returned to the application, as follows:

The first byte is used when querying databases which have key values containing DBCS characters. In these cases, this byte should contain the length of the search key in bytes. If the length is zero or is equal to the defined key length, then a full-key query will be performed. If the length is non-zero but less than the defined key length, a generic query will be performed - all objects that match the indicated bytes will be returned. To retrieve all objects in the database, specify a length of -1 (x'FF'). Ensure that this field contains the number of bytes and not the number of characters to be searched.

The first byte should be set to low-values if the database does not have key values containing

DBCS characters.

In all cases the second byte contains the sort options:

x'00' - ascending date within ascending key

x'01' - descending date within ascending key

x'02' - ascending date within descending key

x'03' - descending date within descending key

On exit, this field may contain a value giving additional information after an exception condition.

**Database name:** Name of database to be queried.

For batch and TSO applications, this should be equal to the database name specified (or defaulted) in the corresponding 'connect database' request. A default value of 'V100' (DD name of OTASV100) is used for the database name if this field is set to spaces or binary zeroes.

For CICS applications, this should be the last four characters of the CICS file control table entry for the primary index of the database to be queried (the first four characters are always set to 'OTAS').

**Primary key pointer:** Pointer to the field containing the primary search key. This field is only required for primary index or volume serial number queries (supplementary code 1 = 0, 100 or 200).

The length of the primary key field addressed by this entry should be equal to the primary keylength specified during database initialization processing. A full or generic primary key value may be specified in this field.

Generic key specifications consist of a string of characters of length 0 to (k-1) bytes (where 'k' is the database primary keylength), followed by an



asterisk ('\*') and trailing spaces, if required. The total field length must always be equal to the database's primary keylength. NearArchive will return each object whose primary key value begins with the character string preceding the asterisk.

(e.g.) if a database has a primary key length of 8 bytes, a generic key length of x'05' could return the following primary key values:

'SMITH'  
'SMITHERS'  
'SMITHSON' etc.

A generic key specification consisting of a single asterisk followed by (k-1) spaces will cause NearArchive to return all objects in the database.

For databases with primary keys containing DBCS characters, the first byte of supplementary code 1 should be used to establish whether a full or generic key search is to be performed. See the description of the supplementary code 2 field above.

**Object archive date:** For volume serial number queries (supplementary code 1 = 200), this field should contain the archive date of the object to be queried.

Not used for other types of query.

**TSQ name:** For CICS applications, specifies the name of the temporary storage queue to be used to hold returned information. The information is returned in the same format as that described for the data buffer field below.

Not used for batch or TSO applications.

**Object start position:** Not used for this request type.

**Object length:** For CICS applications, gives the maximum number of entries to be returned in the target TSQ. On return from NearArchive this field will

contain the actual number of entries in the target TSQ.

Not used for batch or TSO applications.

**Data buffer length:** Not used for CICS applications.

For batch and TSO applications, gives the maximum number of formatted entries (each 'k+24' bytes in length for compact queries, 'k+32' bytes for extended queries or 6 bytes for volume serial number queries, where 'k' is the primary key length of the database) which the application can accept in the data buffer field.

On return from NearArchive, this field gives the actual number of formatted entries in the data buffer field.

**Data buffer:** On entry, contains the full or generic secondary key value to be queried (supplementary code 1 not = 0). Unused on entry for primary key queries. A full or generic secondary key value may be specified in this field.

Generic key specifications consist of a string of characters of length 0 to (s-1) bytes (where 's' is the secondary keylength), followed by an asterisk (\*) and trailing spaces, if required. The total field length must always be equal to the keylength of the secondary index specified in supplementary code 1.

(e.g.) if secondary index '05' has been defined for a database with key length of 6 characters, a genericsecondary key length of 3 could return different primary keys for the following secondary key values:

'ABC'  
'ABCD'  
'ABCDE'  
'ABCDEF' etc.

A generic key specification consisting of a single

asterisk followed by (s-1) spaces will cause NearArchive to return all objects in the database which are indexed in the specified secondary index.

For databases with secondary keys containing DCBS characters, the first byte of supplementary code 1 should be used to establish whether a full or generic key search is to be performed. See the description of the supplementary code 2 field above.

On exit for batch or TSO applications only, this field will contain one formatted index entry for each NearArchive object which satisfies the query, up to the limit specified in the preceding field in the parameter interface.

The data buffer field in the parameter interface area (for batch and TSO applications) or temporary storage queue identified in the parameter interface (for CICS applications) will contain one formatted entry for each NearArchive index entry which satisfies the query criteria.

Query responses may be in three formats – compact format (A), extended format (B) and a volume serial number response (C). The response format is determined from the contents of supplementary code 1.

***A) Compact format***

For compact queries (supplementary code 1 in the range 0-99) each entry is ‘k+24’ bytes in length (where ‘k’ is the primary keylength of the database) and has the following format:

Offset	Length	Type	Description
0	k	Char	Primary key of NearArchive object.  If a secondary index query was issued, this field will contain the primary key of a NearArchive object which is indexed by the specified secondary search key.
k	8	Char	Archive date of NearArchive object

Offset	Length	Type	Description
			(in format YYYYMMDD)
k+8	2	Bin	Number of tape blocks used to hold NearArchive object.
k+10	4	Bin	Identifier of first block on tape volume containing archived object.
k+14	2	Bin	Sequence number in database of tape dataset containing the NearArchive object.
k+16	1	Char	Hierarchical storage level in which object is stored (0 to 7).
k+17	1	Char	Object residency indicator: blank - tape only 'D' - tape + disk
k+18	4	Bin	Identifier of first block on disk ('K') copy dataset containing the NearArchive object (object residency = 'D' only). This field will contain binary zeroes if no disk copy of the object exists.
k+22	2	Bin	Number of disk blocks used to hold the object (object residency = 'D' only). This field will contain binary zeroes if no disk copy of the object exists.

### ***B) Extended format***

For extended queries (supplementary code 1 in the range 100-199) each entry is 'k+48' bytes in length (where 'k' is the primary keylength of the database) and has the following format:

Offset	Length	Type	Description
0	k	char	Primary key of archived object. If the query was based on primary key, this field will be identical for

Offset	Length	Type	Description
			all entries, and will contain the primary key value passed to NearArchive via the call parameter area.
			If the query was based on secondary key, this field will contain the primary key of an archived item which is indexed by the supplied secondary key.
k	8	date	Date of archived object (in format YYYYMMDD).
k+8	4	bin	Number of tape blocks used to hold archived object.
k+12	4	bin	Identifier of first block on volume containing archived object.
k+16	2	bin	Sequence number of tape dataset in database which contains the first block of archived object.
k+18	2	bin	Sequence number of tape dataset in database which contains the last block of archived object.
k+20	1	char	Hierarchical level in which data is stored (0 to 7).
k+21	1	char	Disk copy indicator: 'D' – disk copy of object exists ' ' - no disk copy exists
k+22	4	bin	Block-Id on K-Copy Dataset (in format 0TTR)
k+26	4	bin	Number of blocks in K-Copy dataset which are used for storage of the archived object
k+30	2	Bin	'K' copy dataset segment number
k+32	16	-	Reserved

### ***C) Volume serial number response***

If a volume serial number query has been requested (supp code 1 = 200), the response area will contain a series of one or more 6-character fields. These fields will specify the serial number(s) of the tape volume(s) containing the identified object, starting with the volume containing the first block of the object and for multi-volume objects will continue with the serial numbers of the other tape volumes containing the object, in sequence.

The Data Buffer Length field will contain the total number of volume serial numbers in the data area, on return from NearArchive.

### **Usage Notes**

1. For batch or TSO applications, the calling program should always set the 'data buffer length' field to the maximum number of formatted entries which it can receive in the data buffer field, prior to issuing a 'query index' call. Failure to do this may result in storage corruption due to NearArchive exceeding storage bounds within the calling program. Where the number of formatted index entries satisfying a query exceeds the maximum ('n') which a calling program has indicated that it will accept, NearArchive will place the first 'n' entries in the data area in the parameter interface block, and set a return code of '00', and supplementary code 1 value of 4. In these circumstances, the calling program should issue a further 'query index' request, specifying the same query criteria as in the first request. NearArchive will then return the next 'n' formatted entries in the data buffer field.

When all index entries which satisfy a query have been returned, NearArchive will set a return code of '00' and supplementary code 1 value of 0. Note that in this case, the number of returned entries may be less than 'n'; the data buffer length field will be reset to indicate the number of returned entries ('m') in the data buffer. In these circumstances, calling applications should only process the first 'm' entries; the contents of the remaining ('n-m') entries will be unchanged from the contents on entry to NearArchive. Submission of a repeat 'query index' request for a key after all entries for the previous request have been returned (and where there have been no intermediate query index calls), will result in the first 'n' items for that query being once again returned to the caller.

2. For CICS applications, the temporary storage queue which holds the returned index entries will be created by NearArchive. Any queue with the same identifier which existed prior to the call will be deleted. This means that applications do not need to be responsible for deletion

of temporary storage queues prior to use. In particular, repeated index query requests may be issued by an application, without requiring deletion of temporary storage queues between requests.

After a successful call, the number of items in the temporary storage queue will be equal to the number of primary or secondary index entries which satisfy the query request criteria.

3. Index entries will be returned by NearArchive in the sort sequence specified by the calling application in supplementary code
4. Note that no tape access is required to satisfy an index query request.
5. For CICS applications, an entry must exist in the CICS File Control Table (FCT) for the primary or secondary index being queried. Refer to section 2.8 for a description of NearArchive CICS control table entry requirements.

#### **4.3.3 Request Code 42 – Retrieve an Object Record.**

This call is issued by a batch or TSO application to retrieve the next data record from a NearArchive object. Supplementary code 1 is used to indicate whether a sequential or direct read is required.

For a sequential read, the next logical record stored physically on the tape is retrieved.

For a direct read, a change of primary key or archive date in the parameter area will cause that NearArchive object to be located and its first data record retrieved. Subsequent direct reads, using the same key and date values, will cause successive data records from that NearArchive object to be retrieved.

For direct reads, return code '30' is used to indicate that all records belonging to a NearArchive object have been retrieved. The application may then continue to issue direct or sequential reads as required.

Request code '42' should not be used by CICS applications. Request code '43' (read complete object) should be used instead.

Call parameters should be set as follows:

**Request code:** This must be set to the value '42' for all read object record requests.

**Return code:** Not used on entry to NearArchive. This field will

contain a 2-character numeric value on return from NearArchive indicating the result of the call. Refer to chapter 8 of the manual for a description of all possible return codes.

**Supplementary code 1:** On entry, identifies the type of read request, as follows:

- 0 direct read. For direct reads, the primary key and archive date of the NearArchive object to be retrieved are supplied by the application in the appropriate parameter fields.
- 1 sequential read. For sequential reads, the next record is retrieved from the database (in ascending physical sequence). The primary key and archive date of the NearArchive object to which the retrieved record belongs is returned by NearArchive in the corresponding parameter fields. A sequential read request will always be satisfied from tape, even if disk copy processing has been enabled for the database.

This field may contain a value on return, giving additional information after an exception condition.

**Supplementary code 2:** Must be set to 0 on entry to NearArchive. On return may contain additional information after an exception condition.

**Database name:** Name of database to be accessed. This should be equal to the database name specified (or defaulted) in the corresponding 'connect database' request.

A default value of 'V100' is used if this field contains spaces or low-values.

**Primary key pointer:** Pointer to the field containing the primary key of the NearArchive object to be accessed.



The length of the primary key field addressed by this entry should be equal to the primary key length specified during database initialization processing. For DBCS customers, this is the true number of bytes including all shift control bytes present when the object was archived.

For a direct read request (supplementary code 1 = 0), the primary key field addressed by this pointer value must contain the primary key of the NearArchive object to be retrieved.

For a sequential read request (supplementary code 1 = 1), the primary key field addressed by this pointer value will be set on exit by NearArchive with the primary key of the NearArchive object to which the returned record belongs.

**Object archive date:** The archive date of the NearArchive object to be accessed (in format YYYYMMDD).

For a direct read request (supplementary code 1 = 0), this field must contain, on entry, the archive date of the NearArchive object to be retrieved. Not used on entry for sequential reads.

For a sequential read request (supplementary code 1 = 1), on exit this field will contain the archive date of the NearArchive object to which the returned object belongs. Not used on exit for direct reads.

**TSQ name:** Not used for this request type.

**Object start position:** Not used for this request type.

**Object length:** Not used for this request type.

**Data buffer length:** On entry, gives the maximum record length which the application can accept in the following data buffer field.

On exit, gives the actual length of the data record in the following field. If the retrieved record is longer than the maximum specified on entry, it will be truncated and a return code of '04' set to indicate this condition. In this case, the contents of this field will give the truncated record length.

**Data buffer:** Contains the retrieved record on return from NearArchive after a successful call.

### **Usage Notes**

1. Applications can issue a combination of direct and sequential retrieval requests as required, in any order.
2. A direct retrieval request will cause repositioning in the NearArchive database on change of primary key or archive date. Further direct requests which specify the same primary key and archive date will cause successive records to be retrieved from the

identified object. A return code of '30' will be set by NearArchive when all logical records for a NearArchive object have been retrieved.

3. A sequential retrieval request will cause the next physical sequential record, from the current position, to be retrieved from the database. The primary key and archive date of the NearArchive object to which the retrieved record belongs will be returned by NearArchive in the call parameter area.

4. No indication is given by NearArchive that an object boundary has been crossed during sequential retrieval. It is the responsibility of the calling application to identify a change of primary key or date in the returned parameter area to establish that an object boundary has been crossed.

5. Current position for a sequential retrieval request may be established initially by a direct retrieval request. The current position is updated during sequential retrieval processing.

Where no current position has been established prior to the first sequential retrieval request being received by NearArchive after a database has been connected, the first active record in the lowest dataset sequence number in storage level 0 of the database will be returned.

6. During sequential processing, when the last record in a database storage level has been retrieved, the next sequential retrieval request will return the first record in the lowest dataset sequence number in the next storage level in the database. No special indication is given by NearArchive that a storage level boundary has been crossed during sequential retrieval.

If the next storage level is not defined for a database in the above circumstances, or if the last record in storage level 7 has been sequentially retrieved, NearArchive will set a return code of '30', and a supplementary code 1 value of 1 if another sequential retrieval request is submitted. This indicates an 'end of database' condition. If the application continues to submit sequential retrieval requests, NearArchive will start sequential processing from the beginning of the database again.

7. For direct processing requests, the disk ('K') copy of an object will always be retrieved if it exists. If disk retrieval fails for any reason, NearArchive will retrieve the object from tape as normal.

For sequential processing requests, NearArchive will always access the tape copy of an object, whether a disk copy exists or not. The

order in which objects are accessed during sequential processing will be governed by the physical order of storage of objects on tape.

If a direct retrieval request is satisfied from the disk copy of an object and is then followed by a sequential retrieval request, NearArchive will access the next object stored in physical sequence following the tape copy of the object which had been directly retrieved. Sequential processing will continue by retrieving tape copies of objects from that point onwards.

#### **4.3.4 Request Code 43 – Retrieve a Complete Object.**

A single ‘retrieve object’ request is issued by a CICS application in order to retrieve from a NearArchive database all logical records for a specified NearArchive object.

Logical records are returned by NearArchive to the application as separate items in the temporary storage queue whose identifier is supplied by the application in the NearArchive parameter interface area.

One item will exist in the temporary storage queue for each logical record which is contained in the specified NearArchive object. Logical records will be held in the temporary storage queue in the same sequence in which they are stored physically in the database; this is the sequence in which they were passed to NearArchive when stored in the database.

Call parameters should be set as follows:

**Request code:** This must be set to the value ‘43’ for all read complete object requests.

**Return code:** Not used on entry to NearArchive. This field will contain a 2-character numeric value on return from NearArchive indicating the result of the call. Refer to chapter 8 of the manual for a description of all possible return codes.

**Supplementary code 1:** On entry, used to identify the type of temporary storage queue to be used to hold the retrieved NearArchive object:

0 - auxiliary temporary storage queue

1 - main temporary storage queue.

On exit, this field may contain a value giving additional information after an exception

condition.

**Supplementary code 2:** On entry, indicates the priority to be given to the retrieval request. This must be a binary value in the range 0-9, where 0 is the lowest priority and 9 is the highest.

On return may contain additional information after an exception condition.

**Database name:** Name of database to be accessed. This four-character identifier should specify the last four characters of the CICS file control table entry for the primary index of the database to be accessed.

**Primary key pointer:** Pointer to the field containing the primary key of the NearArchive object to be retrieved.

The length of the primary key field addressed by this entry should be equal to the primary key length specified during database initialization processing. For DBCS customers, this is the true number of bytes including all shift control bytes present when the object was archived.

**Object archive date:** The archive date of the NearArchive object to be retrieved (in format YYYYMMDD).

If the archive date on entry is set to '00000000', NearArchive will return the oldest object with the specified primary key. If the archive date on entry is set to '99999999', NearArchive will return the newest object with the specified primary key value. In each of these cases, the archive date field on return from NearArchive will contain the actual archive date of the retrieved object.

**TSQ name:** The name of the temporary storage queue in which NearArchive will place the logical records from the retrieved object.

Note that any existing temporary storage queue with the same name will be automatically deleted

by NearArchive prior to creating the new queue.

**Object start position:** This field should contain a non-negative binary value which identifies the record number within the object at which retrieval is to start, where 0 is the number of the first record in the object. A value of 0 in this field indicates that retrieval is to start from the beginning of the archived object.

If the record number supplied in this field is beyond the end of the object, NearArchive will pass a return code of '04' and supplementary code 1 value of 1 back to the calling application. No object records will have been retrieved.

**Object length:** This field should contain a non-negative binary value which identifies the number of records to be retrieved from the object, starting at the record position identified in the preceding field. A value of 0 indicates that object retrieval is to continue until the end of the object is encountered.

For non-zero values, if the end of the object is encountered before the specified number of records has been read, NearArchive will pass a return code of '04' and supplementary code 1 value of 2 back to the calling application. The retrieval TSQ will contain all records retrieved from the object prior to the end of object condition being encountered.

**Data buffer length:** Not used for this request type.

**Data buffer:** Not used for this request type.

### **Usage Notes**

1. The temporary storage queue to be used by NearArchive to hold a retrieved object will be created by NearArchive if it did not exist prior to the request. If the temporary storage queue was already in existence, NearArchive will automatically delete it prior to retrieving the object.
2. NearArchive will automatically delete temporary storage queues containing retrieved objects if a non-zero value for the TSQDELETE initialization parameter has been specified (see section 2.8.6 for a

discussion of this facility). If the TSQDELETE parameter is set to zero, then it is the responsibility of the calling application to delete the temporary storage queue used to hold a retrieved NearArchive object.

3. All tape and disk access operations will be performed from the NearArchive control region. If an application is not executing in a NearArchive control region, its executing CICS region must be linked to a NearArchive control region using standard CICS Multiple Region Operation (MRO) linkage facilities. Refer to section 2.7 for further information regarding NearArchive and MRO. Note that use of MRO facilities is not mandatory for NearArchive operation.

4. The NearArchive control region must be initialized prior to retrieval requests being issued by applications from within that region, or from within any connected MRO region. Refer to chapter 6 for details regarding initialization of the NearArchive control region.

NearArchive will set a return code of '21' for retrieval requests which have been issued prior to NearArchive control region initialization.

5. NearArchive will set a return code of '04' and supplementary code 1 of 4 if the object being retrieved contains more than 32,767 records. In these circumstances, a string of temporary storage queues will have been created by NearArchive to hold all records from the archived object. Refer to chapter 6 for a discussion of the NearArchive temporary storage queue stringing facility.

6. If a disk copy exists for an object which is to be retrieved (in addition to a tape copy), then NearArchive online retrieval processing will always access the disk copy. If this access fails for any reason, NearArchive will automatically retrieve the tape copy of the object in the standard manner.

#### **4.3.5 Request Code 44 – Write a Data Record.**

This call is issued by a batch or TSO application to write the first or next data record in an object to a NearArchive database.

Call parameters should be set as follows:

**Request code:** This must be set to the value '44' for all write data record requests.

**Return code:** Not used on entry to NearArchive. This field will contain a 2-character numeric value on return from NearArchive indicating the result of the call. Refer to chapter 8 of the manual for a description

of all possible return codes.

**Supplementary code 1:** Not used on entry. On return, this field may contain a value giving additional information after an exception condition.

**Supplementary code 2:** Not used on entry. On return, this field may contain a value giving additional information after an exception condition.

**Database name:** Name of database to be updated. This should be identical to the database name specified (or defaulted) in the corresponding 'connect database' request. A default value of 'V100' (DD name of OTASV100) is used for the database name if this field is set to spaces or binary zeroes.

**Primary key pointer:** Pointer to the field containing the primary key of the NearArchive object of which the record being archived is a component.

The length of the primary key field addressed by this entry must be equal to the primary key length specified during database initialization processing. For DBCS customers, you must include the necessary shift control bytes to avoid problems printing reports.

**Object archive date:** The archive date of the NearArchive object of which the record being archived is a component, in the format YYYYMMDD. This date is used in conjunction with the primary key to uniquely identify a NearArchive object for retrieval from the database.

NearArchive will use any change in the value of this field during successive write requests to identify the start of a new object.

This field must contain a valid date in the range Jan 1<sup>st</sup> 1950 to Dec 31<sup>st</sup> 2049.

**TSQ name:** Not used for this request type.

**Object start** Not used for this request type.



**position:**

**Object length:** Not used for this request type.

**Data buffer length:** On entry, this field must contain the length of the data record to be archived.

**Data buffer:** On entry, this field should contain the data record to be archived.

### **Usage Notes**

1. A NearArchive object can consist of one or more logical records. One 'write data record' request must be issued for each record in a NearArchive object.

NearArchive uses the primary key and archive date referenced in the interface parameter area for this call to identify the NearArchive object for which the record being written is a component.

2. All records to be archived as part of a single NearArchive object must be presented to NearArchive contiguously (i.e.) in successive calls.

NearArchive will create a new primary index entry for a data object when it detects that all records in that object have been presented (i.e.) on change of primary key or archive date in the call interface parameter area. As primary index entries for a NearArchive object must be unique, any subsequent call which specifies the same primary key and archive date will be rejected by NearArchive with a return code of '17' (duplicate key).

3. The length of the data record to be archived must not exceed a value which is 8 bytes less than the maximum blocksize of the database, as specified during database initialization processing.

4. Specification of an archival date in the interface parameter area is mandatory; its omission will cause NearArchive to reject the call with a return code '14' (invalid archival date).

If the presence of an archival date is not required for an application (i.e. only one NearArchive object will ever exist for a single primary key value), a dummy date value may be specified in this field. This dummy date may then be specified explicitly during retrieval, or the NearArchive 'retrieve oldest/newest object' facility may be used to ensure that the single NearArchive object for a specified primary key

is retrieved (see section 4.3.3 and 4.3.4 for a description of NearArchive object retrieval requests). However, any dummy date used must still have a valid date value, in the format YYYYMMDD.

5. A return code of '00' with supplementary code 1 value of 4 will be returned to the calling application if disk ('K') copy processing has been enabled for the database storage level being updated, but NearArchive was not able to create a disk copy of the object. The object will have been successfully stored on primary (and duplex, if enabled) tape.

#### **4.3.6 Request Code 46 – Identify Secondary Key.**

This call is used by a batch or TSO application program to identify, implicitly or explicitly, a secondary key that should be used to index the data object currently being archived. See section 1.3.1 for a description of secondary indexing in a NearArchive database.

For implicit identification, the interface parameters will specify that automatic secondary key identification (as defined during secondary index initialization) should be used to extract the secondary key from the immediately preceding data record passed to NearArchive for archival.

For explicit identification, the interface parameters will identify the secondary key explicitly.

Call parameters should be set as follows:

**Request code:** This must be set to the value '46' for all 'identify secondary key' requests.

**Return code:** Not used on entry to NearArchive. This field will contain a 2-character numeric value on return from NearArchive indicating the result of the call. Refer to chapter 8 of the manual for a description of all possible return codes.

**Supplementary code 1:** On entry, must contain a half word binary value in the range 1-99 which specifies the identifier of the secondary index to be updated. This index must already have been created via the NearArchive database administration dialog in TSO/ISPF.

On return, this field may contain a value giving additional information after an exception

condition.

**Supplementary code 2:** On entry, must be set as follows:

- 0 - use automatic secondary key identification rules, as defined during secondary index initialization, to extract the secondary key from the immediately preceding data record that was presented to NearArchive for archival.
- 1 - the value of the secondary key to be used is supplied in the data area in the interface parameter area. This must be of length 's' characters (the key length for the specified secondary index, as defined during index initialization). For DBCS customers, this is the true number of bytes including all shift control bytes present when the object was archived.

Any other value will be rejected by NearArchive with a return code of '99'.

On return, this field may contain a value giving additional information after an exception condition.

**Database name:** Name of database to be updated. This should be identical to the database name specified (or defaulted) in the corresponding 'connect database' request. A default value of 'V100' (DD name of OTASV100) is used for the database name if this field is set to spaces or binary zeroes.

**Primary key pointer:** Not used for this request type.

**Object archive date:** Not used for this request type.

**TSQ name:** Not used for this request type.

**Object start position:** Not used for this request type.

**Object length:** Not used for this request type.

**Data buffer length:** Not used for this request type.

**Data buffer:** Not used if supplementary code 2 = 0.

If supplementary code 2 = 1, this field should contain the value of the secondary key to be used to index the object currently being archived.

### **Usage Notes**

1. The NearArchive object indexed via a secondary key entry generated from an 'identify secondary key' call is the object whose primary key and archive date are identified in the last 'write data' request which preceded the call. If no 'write data' request has yet been received by NearArchive after connecting to a database, the 'identify secondary key' call will be ignored.
2. Multiple 'identify secondary key' calls may be submitted contiguously to NearArchive. Each call should specify a different secondary index identifier. All contiguous calls will index the same NearArchive object (identified by the primary key and archive date specified in the preceding 'write data' request). Each call will generate a secondary key entry in the appropriate secondary index.
3. If a secondary key identified via this call has already been generated for the current NearArchive object (either through automatic secondary index processing, or through a previous type '14' request being received for that object), then the call will be ignored, and a return code of '00' generated. No special action is required by the application program in these circumstances.
4. Refer to section 1.3.1 of this manual for a description of the NearArchive secondary indexing facility.

### **4.3.7 Request Code 47 – Delete an Object.**

This call is used by a batch, TSO or CICS application to delete an object from a NearArchive database. For batch or TSO applications, the database must be connected for input-output or output access. For CICS applications, the entry for the database's primary index dataset in the CICS File Control Table (FCT) must specify update access.

Call parameters should be set as follows:

**Request code:** This must be set to the value '47' for all object deletion requests.

**Return code:** Not used on entry to NearArchive. This field will contain a 2-character numeric value on return from NearArchive indicating the result of the call. Refer to chapter 8 of the manual for a description of all possible return codes.

**Supplementary code 1:** Not used on entry. On return, this field may contain a value giving additional information after an exception condition.

**Supplementary code 2:** Not used on entry. On return, this field may contain a value giving additional information after an exception condition.

**Database name:** Name of database containing object to be deleted.

For batch and TSO applications, this should be identical to the database name specified (or defaulted) in the corresponding 'connect database' request. A default value of 'V100' (DD name of OTASV100) is used for the database name if this field is set to spaces or binary zeroes.

For CICS applications, this field should specify the last four characters of the CICS FCT entry for the database's primary index dataset.

**Primary key pointer:** Pointer to the field containing the primary key of the NearArchive object to be deleted.

The length of the primary key field addressed by this entry should be equal to the primary key

length specified during database initialization processing. For DBCS customers, this is the true number of bytes including all shift control bytes present when the object was archived.

**Object archive date:** The archive date of the NearArchive object to be deleted (in format YYYYMMDD).

**TSQ name:** Not used for this request type.

**Object start position:** Not used for this request type.

**Object length:** Not used for this request type.

**Data buffer length:** Not used for this request type.

**Data buffer:** Not used for this request type.

### **Usage Notes**

1. For batch and TSO applications, the NearArchive database must be connected for input-output or output processing prior to issuing an object deletion request. A return code of '89' will be set by NearArchive in response to a deletion request for a database which has been connected for input access only.

2. For CICS applications, the primary index of the NearArchive database being accessed must be available for update processing by the CICS application. This file will have a CICS File Control Table entry name of 'OTASxxxx', where 'xxxx' is the database name specified in the NearArchive call parameters (see section 2.8.4 for a description of CICS table entry requirements). A return code of '89' and supplementary code 1 value of X'0100' will be set by NearArchive in response to a deletion request for a database whose primary index is available for input processing only.

Additionally, if the NearArchive audit trail facility has been enabled for the database being accessed, then the database journal file must be available for output processing by the CICS application. This file will have a CICS File Control Table entry name of 'OJRNxxxx', where 'xxxx' is the database name specified in the NearArchive call parameters (see section 2.8.4 for a description of CICS table entry requirements). A return code of '89' and supplementary code 1 value

of X'0300' will be set by NearArchive in response to a deletion request for a database whose journal file is not available for output processing.

3. After successful deletion, a NearArchive object will no longer be available for retrieval. Space used by a deleted object's primary and secondary index entries will be reclaimed during execution of the NearArchive object management utility (OTASP040). Tape and disk space used by the deleted object's logical records will be reclaimed during execution of the NearArchive database maintenance utility (OTASP030).

#### **4.3.8 Request Code 48 – Commit Updates.**

This call is used by a batch or TSO application to commit all updates to a NearArchive database which has been connected for output processing. The commit request will write a 'commit' block to the tape dataset in current use. Tape update processing will restart from this physical location when an incomplete database archival process is resumed.

Call parameters should be set as follows:

**Request code:** This must be set to the value '48' for all commit requests.

**Return code:** Not used on entry to NearArchive. This field will contain a 2-character numeric value on return from NearArchive indicating the result of the call. Refer to chapter 8 of the manual for a description of all possible return codes.

**Supplementary code 1:** Not used on entry. On return, this field may contain a value giving additional information after an exception condition.

**Supplementary code 2:** Not used on entry. On return, this field may contain a value giving additional information after an exception condition.

**Database name:** Name of database for which updates are to be committed. This should be identical to the database name specified (or defaulted) in the corresponding 'connect database' request. A default value of 'V100' (DD name of OTASV100) is used for the database name if this field is set to spaces or binary zeroes.

**Primary key pointer:** Not used for this request type.

**Object archive date:** Not used for this request type.

**TSQ name:** Not used for this request type.

**Object start position:** Not used for this request type.



**Object length:** Not used for this request type.

**Data buffer length:** Not used for this request type.

**Data buffer:** Not used for this request type.

## **Usage Notes**

1. After a commit call has been successfully issued, if Volsafe support is not enabled for the primary or duplex dataset a subsequent database connect request for restart output processing (access code 6) will cause the NearArchive tape dataset to be positioned at the point at which this commit call was issued, for sequential addition of data records to the database. This means that all records added after this commit call (i.e. 'uncommitted' records) will be overwritten on tape storage, and index entries re-used (see chapter 5 for a description of database recovery and restart processing).

If Volsafe support has been enabled for the primary or duplex dataset, restart output processing will cause the tape dataset to be positioned at the volume trailer label, for sequential addition of data records to the database. Uncommitted data on the Volsafe volume will not be overwritten, but will be ignored by ASM for S/390, and will not be accessible via the ASM for S/390 API.

2. If the NearArchive dynamic backout facility is enabled, if Volsafe support is not enabled for the primary or duplex dataset NearArchive will reposition to the commit block on the tape dataset being updated when the last successful commit request was processed and write an end-of-file label, when a 'disconnect without commit' request is received. The next database connect request for output access will cause update of the NearArchive database to continue from that point. Space occupied by uncommitted objects will be re-used.

NearArchive will not reposition to the last commit point during dynamic backout processing if Volsafe support is enabled for the primary or duplex copy dataset. Uncommitted data on the Volsafe volume will not be overwritten during restart connect processing, but will be ignored by ASM for S/390, and will not be accessible via the ASM for S/390 API.

3. Applications which issue explicit NearArchive commit calls should ensure that updates to any other database which is being processed are also committed at the same time. An application should

have its own restart mechanism in the event of failure, so that processing may be restarted from the last commit point.

#### **4.3.9 Request Code 49 – Disconnect a NearArchive Database.**

This call is used by a batch or TSO application to disconnect from a NearArchive database after all input or output processing has been completed. All datasets and storage areas allocated during database connect processing will be released, and any tape dataset in use by the application will be dismounted from the appropriate tape device.

Call parameters should be set as follows:

**Request code:** This must be set to the value '49' for all disconnect requests.

**Return code:** Not used on entry to NearArchive. This field will contain a 2-character numeric value on return from NearArchive indicating the result of the call. Refer to chapter 8 of the manual for a description of all possible return codes.

**Supplementary code 1:** On entry, specifies the disposition of the tape dataset in use when the disconnect request was issued, as follows:

- 0 - unload the last tape cartridge volume used, after closing the database.
- 1 - leave the last tape cartridge volume used mounted at load point.
- 2 - leave the last tape cartridge volume used mounted at its current position.

On return, this field may contain a value giving additional information after an exception condition.

**Supplementary code 2:** Not used if a database is connected for input or input-output processing. For databases connected for output processing, indicates whether database updates are to be committed during disconnect processing, as follows:

0 - commit all updates during disconnect.

1 - do not commit updates during disconnect.

On return, this field may contain a value giving additional information after an exception condition.

**Database name:** Name of database to be disconnected. This should be identical to the database name specified (or defaulted) in the corresponding 'connect database' request. A default value of 'V100' (DD name of OTASV100) is used for the database name if this field is set to spaces or binary zeroes.

**Primary key pointer:** Not used for this request type.

**Object archive date:** Not used for this request type.

**TSQ name:** Not used for this request type.

**Object start position:** Not used for this request type.

**Object length:** Not used for this request type.

**Data buffer length:** Not used for this request type.

**Data buffer:** Not used for this request type.

## **Usage Notes**

1. Use supplementary code 1 to control handling of the tape cartridge volume in use when the disconnect request was issued. If the database is to be re-opened for further processing, it may be more efficient to suppress unloading of the tape cartridge volume, by setting a non-zero value in supplementary code 1.

When processing a database in input or input-output mode, subsequent access to a tape volume which has not been dismounted after use requires that the tape DD name specified in the database name list with the connect request should be identical to that used in the previous connect request.

2. When a database has been connected for output access, set supplementary code 2 to 0 to commit all updates during disconnect processing. This request should only be issued when output processing has been successfully completed. Rerun of an archival process is not possible after updates have been successfully committed when disconnecting the database.

Applications should set supplementary code 2 to a value of 1 if an error has occurred during the archival process which will necessitate a rerun. The application may then be rerun after the cause of the error has been rectified. If the NearArchive dynamic backout facility has not been enabled, the application should issue a connection request with rerun output access when next updating the database.

---

# CHAPTER 5

---

## *Batch Processing*

This page is intentionally left blank

ASM for S/390 supplies high-level interfaces which enable customer application systems to invoke NearArchive archival and retrieval functions from batch and online programs. These are activated via a CALL interface for batch programs, and a CICS command-level 'LINK' interface for online programs. A parameter block in the CALL interface or CICS communication area is used to communicate requests and responses.

This chapter discusses interface requirements for batch processing programs.

## **5.1 Invoking NearArchive from batch programs.**

---

Calls to NearArchive from batch processing programs are made by issuing standard CALL statements (as implemented for the programming language used to develop the calling application) to the NearArchive module 'OTASP100'. This module is supplied on the distributed ASM for S/390 load library, and must be linked in to the calling program at link-edit time in order to satisfy the external reference. Refer to section 4.1 for information on invoking NearArchive from a batch program using the new application program interface introduced with version 2.4. Refer to appendix D for information on invoking NearArchive using the pre-v2.4 application program interface.

### **5.1.1 JCL requirements.**

The JCL requirements for executing batch programs which invoke NearArchive are:

- a) Include the ASM for S/390 product load library in the JOBLIB or STEPLIB concatenation for the job, or ensure that this library is in the system linklist concatenation.
- b) Include a DD card with DDname OTASxxxx for each NearArchive database to be accessed during jobstep execution, where 'xxxx' is the database name as specified in the NearArchive 'connect database' call. Each DD card should identify the name of the database primary index dataset (e.g.):

```
//OTASxxxx DD DSN=database.primary.index.dsn,DISP=SHR
```

If no database name is supplied in the 'connect database' call, a default

value of 'V100' is used. In this case a DD card with DDname OTASV100 should be supplied.

- c) Add a REGION parameter to the JOB or EXEC card as required.
- d) If asynchronous secondary index update processing is to be performed, include a DD card with DDname OTnnxxxx for each secondary index which is to be updated asynchronously during batch archival processing to database 'xxxx' (where 'nn' is the 2-digit secondary index identifier, and 'xxxx' is the database identifier specified in the application's database open request). For example, to enable asynchronous updates to secondary index '01' in database 'TEST', the following DD card should be added to the JCL used to execute the archival job:

```
//OT01TEST DD DSN=sequential.dataset.name,DISP=SHR
```

where 'sequential.dataset.name' is the name of a sequential disk dataset. This may be allocated during execution of the job (DISP=NEW,CATLG), in which case the DCB attributes will be automatically set by NearArchive, or may be pre-allocated (DISP=SHR), in which case the dataset must be in variable blocked format, with a logical record length of 276 bytes (RECFM=VB, LRECL=276). Any appropriate blocksize may be chosen. The dataset will contain one record of length (k+s+10) bytes (where 'k' is the primary keylength and 's' is the secondary keylength) for each secondary index entry created during archival processing.

There should be one 'OTnnxxxx' DD card for each defined secondary index which is to be updated asynchronously during archival processing.

Following successful completion of the batch archival process, the secondary index update utility OTASP080 should be executed once for each secondary index which is being updated asynchronously (i.e.) once for each 'OTnnxxxx' DD card in the JCL used to executed the batch archival job. Refer to chapter 7 of this manual for a description of this utility. Asynchronous update processing for a secondary index is not complete until this utility has been executed successfully for that secondary index. Further archival to a database will not be allowed until all outstanding asynchronous secondary index updates to that database have been completed successfully.

If no 'OTnnxxxx' for secondary index 'nn' is supplied in the JCL, all updates to that secondary index will be performed synchronously during archival processing. This is the default process. No further



action is required following successful completion of the archival process. All updates to the appropriate secondary index will have been performed successfully at this time.

Remaining JCL requirements are merely those needed for running the program which invokes NearArchive services. Note that all tape datasets are dynamically allocated by NearArchive, and do not therefore require any JCL DD statements.

### **5.1.2 Controlling Access to a NearArchive Database.**

ASM for S/390 uses standard OS/390 macros to process a NearArchive database, and consequently access to a NearArchive database for object archival or retrieval may be controlled by a system security software product such as RACF.

To protect access to a NearArchive database at this level, security administrators should code a generic dataset protection rule, using the high-level qualifier of the dataset name as specified at database initialization time to control access to the database. Batch jobs which store or delete objects in a NearArchive database should be given UPDATE access authority to the appropriate dataset profile. Jobs which retrieve objects from a NearArchive database should be given READ access authority to the dataset profile.

Alternatively, the supplied user exit module OTASUX01 can be replaced to implement customer-specific security processing for accessing a database and its objects. Refer to section 2.8 for information on implementing external security processing via the security user exit.

### **5.1.3 Database design considerations.**

When modifying existing applications to interface with ASM for S/390, or developing new applications, the following points should be noted:

- If more than one data record is to be archived per NearArchive object, then it may be necessary to be able to identify types of data record on retrieval. It may be possible to perform this identification from the data content. However, if this is not possible, it will be necessary to add a prefix to the data prior to archival. This prefix should be passed to NearArchive as an integral part of the data to be archived, and the length of the data record as specified in the parameter block should include the length of the prefix.

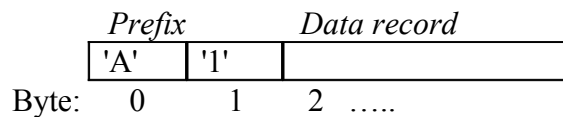
NearArchive will then treat both prefix and data as one data record -

on recall, the prefix will be presented to the caller along with the archived data in exactly the same format as that passed to NearArchive during the archival process.

- The record prefix should include at least two sections. The first should identify the record type (e.g.. code 'A' for record type 1, code 'B' for record type 2 etc.), in order to enable retrieval programs to handle recalled data correctly.

The second section should identify record level - this will allow for the modification of the format of the archived data record during the life of the application (e.g.) level code '1' could be used for the initial format at the time of system implementation. This numeric code can then be incremented each time a modification is made to the primary data record.

By making similar changes in the NearArchive archival and recall applications, it will be possible to cater for multiple data formats per record, in order to handle data correctly over the life of the ASM for S/390 application.



where 'A' = record code and '1' = record level

**Fig 5.1:** Example of record prefix for archived data.

## 5.2 Batch Archival Processing.

---

### 5.2.1 Processing Overview.

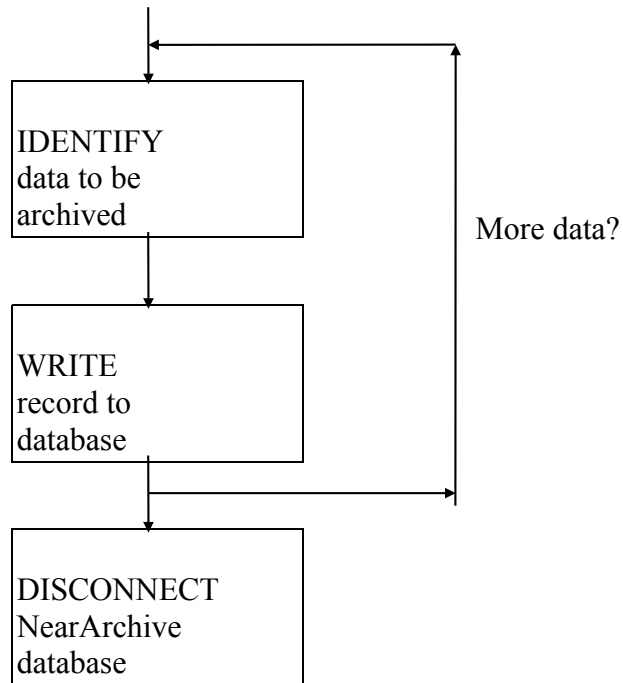
The process of archiving data to a NearArchive database from a batch application consists of the following steps:

- Connect to the NearArchive database to be updated. An ASM for S/390 application may only have one database connected for update at any one time. The database name is identified in the NearArchive open request. Multiple NearArchive databases may be updated from within a single invocation of a batch program, provided each is disconnected prior to opening the next database. The storage level in which objects are to be stored is set in the database name list on the connect request. Note that objects must all be stored in the same storage level during any single database connection. To change the storage level, the application should

disconnect from the database and reconnect, specifying a different storage level in the database name list.

- The archival application must locate and identify data to be archived. Identification of data eligible for archival is an application-specific function. It may be done on the basis of age of data, or by any other method appropriate to the archival application. The location of the data to be archived is similarly dependent on the nature of the archival application itself. ASM for S/390 does not have any built-in interfaces to specific disk database management systems. It is the responsibility of the calling application to process that data via the standard interface(s) supplied with the appropriate database management system product.
- Establish the unit of data which is to be archived as a single object in a NearArchive database. A NearArchive object can consist of multiple logical records. Each logical record within the NearArchive object will be stored in the same physical location, and will be indexed by a single primary index entry, consisting of primary key and archive date. All logical records belonging to the same NearArchive object may normally be retrieved (via batch or line processing) through one tape access operation. For DBCS customers, the primary and secondary keys should include all shift control bytes to avoid problems when printing reports.
- Pass each logical record sequentially to NearArchive for insertion in the NearArchive database, via NearArchive 'write record' call (see section 4.3). Each logical record belonging to the same NearArchive object should be identified by the same primary key and archive date.
- All logical records belonging to the same NearArchive object should be passed to NearArchive in one contiguous sequence. NearArchive will use a change of primary key and/or date in the call interface parameters to identify that all records belonging to a single NearArchive object have been passed by the application.
- When all objects to be archived have been written to the NearArchive database, the application should disconnect the database. This commits all changes to the database. Termination of application processing without disconnecting a database will prevent the database being opened for normal processing by any application. In these circumstances, an application must issue a 'rerun' connect request before being able to add data to that NearArchive database.

Figure 5.2 illustrates the standard application logic flow required to archive data to a NearArchive database.



**Fig 5.2:** Batch archival procedure

### **5.2.2 Recovery and Restart.**

Should the archive process fail for any reason, the job will need to be rerun from start of processing, or from the last commit point (if the application issues explicit NearArchive ‘commit data’ requests). The manner in which archival rerun processing is to be performed depends on whether the NearArchive dynamic backout facility has been enabled, as described in the following section.

#### **5.2.2.1 Recovery using dynamic backout.**

NearArchive will optionally backout all updates to a database during archival processing, back to the last commit point, when a ‘disconnect without commit’ request has been made by the calling application. This process is termed ‘dynamic backout’.

When dynamic backout processing is enabled, NearArchive will delete all index entries for uncommitted objects, and for non-Volsafe tape volumes will ensure that all tape space occupied by those objects is over-written during subsequent archival processing. Used tape space cannot be reclaimed for Volsafe tape volumes.

If dynamic backout processing has been successfully performed after an error condition causes the calling application to issue a 'disconnect without commit' request, then no special restart considerations need be observed. Objects backed out during the failed archival run may be re-archived at any subsequent time, after the cause of the original failure has been rectified.

The NearArchive dynamic backout facility is enabled during batch archival processing by including the following DD card in the JCL used to execute the procedure:

```
//OTASLOG1 DD DSN=&&OTASLOG1,DISP=NEW,UNIT=xxxxxxx,  
            SPACE=(CYL,(x,y))
```

Parameters in this card should be set as follows:

- **DDname** This must have the value 'OTASLOG1'.
- **DSN** Any temporary or permanent dataset name may be used.
- **DISP** Enter 'DISP=NEW' if a temporary dataset is being used. Otherwise enter 'DISP=SHR', if an existing permanent dataset is being used.
- **UNIT** This is required if 'DISP=NEW' is specified. This parameter should specify a disk unit esoteric name or device type.
- **SPACE** This is required if 'DISP=NEW' is specified. Enter values for the primary and secondary allocations ('x' and 'y') as appropriate.

The dynamic backout log dataset will contain one record for each object which has been successfully archived since the last commit point. Each record will be (k+6) bytes in length, where 'k' is the primary keylength of the database being updated. The log dataset should be large enough to contain the maximum number of objects which are likely to be archived between commit points.

### **5.2.2.2 Recovery without dynamic backout.**

If a batch archival process fails while the NearArchive database that is being updated is connected for update and the database is not disconnected normally, or is closed without committing updates (supplementary code 2 = 1 for request code '49') and dynamic backout is not enabled, then

uncommitted objects will remain in the database. The database will be marked by NearArchive as being in an 'uncommitted' state. Backout of these objects, and re-use of the tape space occupied by them (for non-Volsafe tape volumes), will take place during re-archival of uncommitted objects during restart of the archival process. NearArchive will prevent further updates to that database while it is an 'uncommitted' state, by rejecting any requests to connect that database for normal output processing (access codes 4 or 5).

In order to successfully connect to an uncommitted database for output, an application must connect for restart update processing (access code 6). On receipt of this request, for non-Volsafe tape volumes NearArchive will reposition in the database to the tape location at which the last successful commit request was issued during the failed archival run, or to the point at which the database was last opened for output processing if no explicit commit request was issued. For Volsafe tape volumes, NearArchive will position to the end of the tape dataset.

After successfully issuing the restart update connect request, the application may then continue to write data objects to the NearArchive database. For non-Volsafe tape volumes, storage occupied by objects which had been physically written to tape prior to the point of failure during the previous archival run will now be physically overwritten with newly-archived objects.

In addition, index entries for uncommitted objects which had been created during the failed run will be re-used when the same objects are re-archived during restart processing. This is in contrast to ASM for S/390 processing after a normal open request, where requests to archive an object will be rejected with a return code '17' if an index entry already exists for that object ('duplicate object' condition).

Note that index entries will only be re-used when uncommitted objects are re-archived after failure when the primary key **and** archive date of an object are identical in both runs. If it is possible that an archive date of an object may be different in the restart run from that used in the failed run (e.g., archive date is set equal to system run date), then the 'archive over-ride date' facility may be used to ensure that the dates remain the same. This facility is invoked with the restart update connect request (access code 6) and informs NearArchive that the over-ride date present in the database name list passed with the connect request should be used in preference to the archive date presented in a write request (request code '44'). Refer to section 4.3.1 for more details on this facility.

The NearArchive database should be disconnected normally with commit processing (supplementary code 2 = 0 for request code '49') following successful completion of the restarted archival process. This will allow subsequent archival processes to successfully connect to the NearArchive database for normal processing.

The following points regarding the above process should be noted:

- When an application detects a failure condition during an archival procedure, the NearArchive database should be disconnected without commit processing (supplementary code 2 = 1 for request code '49'). Disconnecting a NearArchive database in any other manner automatically commits all updates and will cause any subsequent restart connect request to be rejected (with return code '18').
- An application should ensure that all objects which were presented to NearArchive after the last commit point prior to the point of failure during a failed archival procedure should be re-presented during the restarted archival run. Tape storage and index entries for these objects will be re-used during the restarted process (effectively rolling back uncommitted database updates made during the failed run).

For non-Volsafe volumes, tape storage occupied by uncommitted objects will **always** be overwritten during a restart run. Failure to re-archive uncommitted objects will mean that the tape storage occupied by those objects will be overwritten, but that the index entries for uncommitted objects created during the failed run will remain in place. Attempts to retrieve any such object will fail with a return code '86'. This consideration is not relevant if Volsafe tape volumes are used for object storage.

Should any of the above issues raise problems regarding data integrity etc. during restart processing for any particular application, then the NearArchive database should be recovered after failure of an archival process and the entire archival process rerun normally, using the procedure discussed in the next section.

### **5.2.2.3 Asynchronous secondary index update processing.**

Asynchronous secondary index update processing is NOT permitted if the dynamic backout facility has been enabled (i.e. an 'OTASLOG1' DD card is present in the JCL used to execute the object archival process). A warning message will be issued to the job log during archival processing in these circumstances, and normal synchronous secondary index update processing will be performed.

There are no special considerations for restart of a failed object archival process where asynchronous secondary index update processing has been enabled. NearArchive does NOT require the secondary index update utility to be executed prior to restart of the object archival procedure. Existing entries in each secondary index update dataset (file(s) 'OTnnxxxx') will be appended with new entries created during rerun of the object archival job.

Following successful completion of archival processing, the secondary index update utility OTASP080 should be run as normal for each secondary index being updated asynchronously. Any duplicate entries in the input file to this utility which may result from restart of the archival process will not be processed by the utility.

### **5.2.3 Database Backup and Recovery.**

A NearArchive database consists of the following components:

- a) multiple single-volume tape datasets (maximum 65535 per storage level).
- b) optionally up to 256 disk copy datasets for each primary tape dataset
- c) a primary index dataset (VSAM KSDS).
- d) one space management dataset per database storage level (VSAM KSDS).
- e) one secondary index dataset (VSAM KSDS) for each secondary index that has been defined for a database.
- f) a database journal dataset (VSAM ESDS). Only created if the NearArchive audit trail facility has been enabled for a database.
- g) a migration control dataset (VSAM ESDS). Only created if multiple storage levels have been defined for a database.

System disk datasets (items (c) to (g) above) should be backed up after update using standard disk dataset backup procedures. Although no NearArchive dataset restore activity is required prior to rerunning a failed archival job (see section 5.2.2), these backups will be required in the event of loss of a dataset through accidental deletion or media failure.

Backup and recovery of a NearArchive database's tape and disk datasets (items (a) and (b) above), is controlled by ASM for S/390 itself, as described below.



### **5.2.3.1 Tape Dataset Backup.**

ASM for S/390 allows the creation of up to four full-volume copies of each tape dataset within a NearArchive database. These copies are identified as follows:

- a) Primary ('A') copy.

This copy is always present, and is the primary copy created and updated during standard data archival processing.

- b) Primary backup ('B') copy.

The 'B' copy is the primary backup copy created by the NearArchive database backup utility OTASP110.

- c) Duplex ('C') copy.

The duplex ('C') copy is created during batch archival processing concurrently with the primary 'A' copy, if duplexing has been enabled for the NearArchive database (see chapter 3). Creation of a 'C' copy requires the availability of a second tape cartridge drive unit during the archival procedure.

- d) Duplex backup ('D') copy.

The duplex backup ('D') copy is created by the NearArchive database backup utility OTASP110, concurrently with the 'B' copy, if duplexing during backup has been enabled for the NearArchive database (see chapter 3). Creation of a 'D' copy requires the availability of a third tape cartridge drive unit during the database backup procedure.

The dataset copy identifier is held as part of the standard NearArchive tape dataset name, which has the following format:

`'HLQ' . xy . Gnnnnn`

where 'HLQ' is the high-level dataset name qualifier used for generating system dataset names for the NearArchive database (as defined during database definition processing)

'x' is the copy identifier ('A','B','C' or 'D')

'y' is the storage level identifier ('0' to '7')

and 'nnnnn' is the dataset sequence number (in the range 00001 - 65535).

Note that the 'xy' dataset name qualifier is not present for primary ('A') copy datasets in storage level 0 of a NearArchive database.

Users may utilize this naming standard to (automatically) control tape cartridge creation and storage locations for each set of backup copies of a NearArchive database.

In addition, the NearArchive backup utility may be used to create one or more incremental backup datasets for each primary tape dataset. An incremental backup dataset will contain all objects stored on the corresponding primary tape dataset since last execution of the backup utility for that database. Up to 9999 incremental backup datasets may be created for each primary tape dataset. These datasets may be created on different tape devices from those used to create the corresponding primary dataset, and may be multi-volume if necessary. In addition, up to two copies (primary and duplex) of each incremental backup dataset may exist.

Incremental backup datasets will have a dataset name of the form:

`'HLQ'.Gnnnnn.xymmm`

where 'HLQ' is the high-level dataset name qualifier used for generating system dataset names for the NearArchive database (as defined during database definition processing)

'x' is the incremental copy identifier ('I' for primary copy, 'J' for duplex copy)

'y' is the storage level identifier ('0' to '7')

and 'mmmm' is the incremental backup dataset sequence number (in the range 0001-9999). Each execution of the utility will create a new incremental backup dataset which will be allocated a sequence number one greater than the previous incremental backup dataset for the corresponding primary dataset.

Incremental backup datasets may be used to reduce backup times for high-capacity primary tape volumes, by eliminating the need to perform full-volume backups each time a tape dataset has been updated during database update processing.

When a primary volume has been filled to its required capacity, a full-volume backup may be taken by the database backup utility; this process will automatically release and uncatalog all existing incremental backups for that primary dataset

A combination of full-volume and incremental backups may exist for a primary tape volume (e.g., an initial full-volume backup when a tape dataset was first created, followed by one or more incremental backups after subsequent update operations).

The following sample backup configuration may be used to ensure accessibility and availability of data within a NearArchive database:

- 'A' copies held onsite and available for automatic handling via the Automatic Cartridge System hardware. These copies will be used for normal database access requests.
- 'C' copies held onsite on shelf storage. These copies will be used to recover individual primary volumes due to media failure.
- 'B' copies held offsite. These copies will be used to provide database access when the primary copies are unavailable due to serious hardware failure, or unavailability of the prime data center (disaster recovery).

Incremental backups of the latest (partially-filled) volume in the database may be held offsite; this will avoid the requirement for repeated full-volume backups of the same primary tape volume after each database update operation, and of continual shipment of new full-volume backups offsite, and recall of old (uncataloged) full-volume backups from the offsite location. The offsite incremental backups will be automatically uncataloged when a full-volume backup is taken of the latest volume in the database (after it is filled to its required capacity); in this case the full-volume backup may be moved offsite and the uncataloged incremental backup volumes recalled from the offsite location.

ASM for S/390 provides standard facilities for access and recovery of data in the above circumstances.

Refer to section 7.5 of this manual for a description of the NearArchive database backup utility OTASP110. To produce 'B' and 'D' copy datasets, this utility should be automatically run after creation or update of a NearArchive database via a batch archival procedure.

If more than four copies of a NearArchive tape dataset are required, additional copies may be created using a physical (i.e., block-by-block) copy utility.

It is important that the manual copying of a tape dataset should be performed by a utility that operates at the physical (i.e., block) level rather than the logical (i.e., record) level, in order to preserve the internal blocking structure used by ASM for S/390. This can be achieved using

IEBGENER, with DCB over-rides in the JCL to ensure copying at the physical level, as follows:

```
//BACKUP    EXEC  PGM=IEBGENER
//SYSUT1    DD    DSN=primary.tape.dataset.name,DISP=OLD,
//           VOL=SER=xxxxxxx,UNIT=3480,
//           DCB=(RECFM=U,BLKSIZE=bbbbbb)
//SYSUT2    DD    DSN=backup.tape.dataset.name,
//           DISP=(NEW,CATLG),
//           VOL=SER=yyyyyyy,UNIT=3480,
//           DCB=(RECFM=U,BLKSIZE=bbbbbb)
//SYSPRINT  DD    SYSOUT=*
//SYSIN     DD    DUMMY
```

where 'bbbbbb' is the maximum block size of the database as specified during database initialization.

It is important to note that standard NearArchive duplexing and backup facilities will record details of all backup copies in the database index. These details are used during retrieval of data from backup copies after loss of a primary dataset, and in recovery of primary copies. 'Stand-alone' backup copies, as described above, will not be able to make use of these access and recovery procedures. In addition, the ability to retrieve data from such a copy, via standard NearArchive retrieval processing, is not guaranteed.

### **5.2.3.2 Tape Dataset Recovery.**

ASM for S/390 provides standard facilities for access and recovery of tape datasets within a NearArchive database, in the following circumstances:

- a) loss of individual tape volume(s) through media failure, mishandling, loss etc.
- b) loss of an entire database, through serious hardware failure, or loss of data center.

#### **a. Media Failure/Loss.**

When an individual tape volume becomes unavailable for any reason, the NearArchive database administrator should place the unavailable volume(s) in recovery status, using the supplied TSO/ISPF database maintenance dialogs.

To do this, the sequence number (or range of sequence numbers) for the unavailable volume(s) should be entered on panel OTSN4401. This panel is accessed via option 1 of the recovery control panel OTSN4400, which

in turn is accessed via option 4 from the main NearArchive menu panel OTSN4000 (refer to chapter 3 for a description of the NearArchive database definition and maintenance facility).

The sequence number ('nnnnn') of a database tape cartridge volume may be obtained from the low-level qualifier ('Gnnnnn') of the NearArchive tape dataset held on that volume.

Once a volume is placed in recovery status, all retrievals of data from that volume will be satisfied from the backup copy identified for that recovery control entry. Users should ensure that all such backup copies are available for automatic cartridge handling, as required, via a local or remote automatic cartridge system.

To restore a volume which is in recovery status, the NearArchive database restore utility (OTASP130) should be used (see section 7.6 for a description of this utility).

After a tape cartridge volume has been recovered using this utility, it will be automatically removed from recovery status. Subsequent retrievals of data from that volume will then be satisfied from the restored primary 'A' copy, which should be available for automatic cartridge handling.

#### **b. Loss of Database.**

If an entire NearArchive database becomes unavailable (due to serious hardware malfunction, or loss of data center), the NearArchive database administrator should invoke the 'disaster recovery' option for that database.

This option is enabled using the supplied TSO/ISPF database maintenance dialogs, by selecting option 2 from the recovery control panel OTSN4400 (which is invoked via option 4 from the main NearArchive menu panel OTSN4000. See chapter 3 for a description of this facility).

When the disaster recovery option is enabled for a NearArchive database, all retrievals will be satisfied, for all tape cartridge volumes within the NearArchive database, from the backup copy whose identifier was specified by the database administrator when enabling disaster recovery.

This means that all such copies should be available for access during disaster recovery processing, preferably using automated cartridge handling facilities.

If necessary, all primary volumes in the database may be recovered using the NearArchive database recovery utility OTASP130 (see section 7.6 for a description of this utility).

When a database has been fully recovered (i.e.. all primary tape datasets restored), the disaster recovery option may be unset using the NearArchive database maintenance dialogs. All subsequent retrievals will then be satisfied from the restored primary dataset copies.

Archival of data to a NearArchive database should be suspended while it is in disaster recovery status. Data archival procedures may be resumed when all primary dataset copies have been restored, and the disaster recovery option has been unset.

### **5.2.3.3 Disk Dataset Backup and Recovery.**

Backup of sequential disk ('K') copy datasets, if 'K' copy processing has been enabled for a database storage level, is not required. These datasets are themselves copies (partial or full) of their corresponding primary tape dataset. Loss of a disk dataset will not affect object access processing; in these circumstances ASM for S/390 will continue with object retrieval processing from tape as normal.

If a disk dataset needs to be re-created (subsequent to its normal deletion by the NearArchive object management utility, or after loss of the dataset for any other reason), it may be re-created from a primary tape dataset using the backup utility OTASP110. This utility will delete and re-create all existing disk copy datasets associated with a primary tape dataset, or create a new set of disk datasets if none currently exists, and copy all objects which are eligible for disk residency (based on the disk retention period set for the storage level being processed) from a designated primary tape volume to the new set of disk datasets. Up to 256 disk datasets may be created for each primary tape dataset. Disk residency rules should be set as required prior to executing the backup utility.

### **5.2.3.4 Database recovery after failure.**

#### **Backing up a NearArchive database after update.**

To be able to recover a NearArchive database following failure of an archival process, a backup copy of system datasets belonging to that database will be required. This backup should be taken after successful completion of a database update procedure.

The system datasets to be backed up after update of a NearArchive database consist of the following:

- The primary index dataset. This is a VSAM KSDS whose name is assigned during initial definition of the NearArchive database. A DD entry for the dataset must be specified in all NearArchive batch procedures, normally using DDname 'OTASV100'.
- All secondary index datasets, if secondary indexing has been implemented for the database. Each secondary index consists of a single VSAM KSDS, with dataset name:

`'hlq.SECIXnn'`

where 'hlq' is the system dataset high-level qualifier, as specified during definition of the NearArchive database, and 'nn' is a value in the range '00' to '99', identifying the secondary index.

- The space management dataset(s) for the storage level(s) updated by the job. Each space management dataset is a VSAM KSDS, with dataset name:

`'hlq.SPACEMLn'`

where 'hlq' is defined as above, and 'n' is the storage level number (in the range 0-7).

- The database's journal dataset, if the audit trail facility has been enabled for that database. This is a VSAM ESDS, with dataset name:

`'hlq.JOURNAL'`

where 'hlq' is defined as above.

Standard VSAM dataset backup procedures may be used to backup all the above datasets.

Backup of tape datasets in the NearArchive database may be made using the standard NearArchive tape duplexing/backup facilities (see section 5.2.3.1 for further details). Tape dataset backups are not normally required for recovery after failure of an archival procedure, unless that failure was caused by problems with the tape cartridge media.

In addition, any non-NearArchive datasets which were updated during the archival procedure (e.g.. VSAM datasets from which archived data was deleted), and for which updates were not rolled back at the time of failure

should also be backed up after successful completion of the archival procedure.

### **Restoring a NearArchive database.**

To restore a NearArchive database to its status prior to the start of a failed archival process, the following steps should be performed:

1. Restore the database's VSAM system datasets from backups made after completion of the previous successful archival procedure. These datasets will include:
  - the primary index dataset
  - all secondary index datasets defined for that database
  - the storage-level 0 space management dataset
  - the journal dataset (if the audit trail facility is enabled).
2. Recover any other non-NearArchive datasets which may have been updated during the archival process whose updates were not rolled back at the time of failure.

When these functions have been completed, the archival procedure may be rerun.

It is not necessary to recover NearArchive tape datasets prior to rerunning the archival procedure. NearArchive will start re-writing to the first tape dataset used during the failed procedure. All new tape datasets created during that failed procedure will be automatically uncataloged by NearArchive during rerun processing, and re-created using new scratch tapes.

However, it should be noted that ASM for S/390 will start writing data at the end of the first dataset, **not** at the position at which it was opened at the start of the failed procedure. This will mean that some physical tape storage will not be re-used during the rerun, causing the amount of active storage on that tape to be reduced. The unused storage will ultimately be released on execution of the NearArchive database maintenance utility.

If it is wished to avoid this condition, then the first tape dataset updated during the failed archival procedure should be recovered using the NearArchive database recovery utility OTASP130. To perform the recovery, place the tape dataset in recovery status, via option 4 of the TSO/ISPF NearArchive database maintenance dialog, and execute the recovery utility. On successful completion of this process, the archival job



may be rerun as described above. Refer to chapter 7 for details on use of the database recovery utility.

Note that only the first tape updated during a failed archival procedure will need to be recovered as described above. All subsequent tapes created during the failed procedure will be automatically uncataloged and re-created by ASM for S/390, and do not need to be recovered individually.

## **5.3 Batch Retrieval Processing.**

---

### **5.3.1 Processing Overview.**

Retrieval of data from a NearArchive database by a batch processing program involves the following steps:

- Connect to the NearArchive database for input or input-output processing (access code 1 or 3 in the database name list). The database should only be opened for input-output processing if the batch application intends to delete objects from the database.

The name of the database to be connected is specified in the NearArchive connect request. An application may only have one database connected for input-output processing at any one time.

Multiple NearArchive databases may be connected for input-output during execution of a batch application, but each must be disconnected prior to connecting the next database.

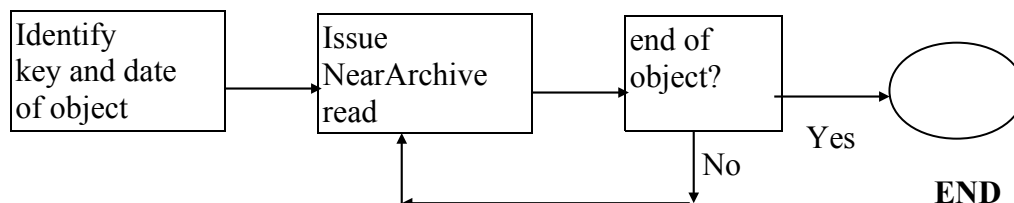
Multiple databases may be simultaneously connected for input processing simultaneously. In addition one database may be concurrently connected for input-output or output processing. Each connection request should specify a unique database name and DDname over-ride entries (offsets 0 and 16 in the database name list).

- Identify the primary key and archive date of the first NearArchive object to be retrieved from the database. This identification will normally come from a batch input document, or will be generated from processing of other datasets or databases.
- Issue a 'retrieve object record' request (code '42') for the object identified above. This request should specify the primary key and archive date of the object to be retrieved. The first logical record

belonging to that NearArchive object will be returned in the data area in the NearArchive parameter interface block.

- If a NearArchive object contains multiple logical records, issue repeated 'retrieve object record' requests for the object. Each request will return the next logical record for the identified object.
- For direct retrieval requests, when all logical records belonging to the NearArchive object have been returned by ASM for S/390, a return code of '30' will be passed back to the calling application in response to the next 'retrieve object record' request. In this case, the calling application should identify the next NearArchive object to be retrieved, and issue a 'retrieve object record' request, specifying the primary key and archive date of the new object. Retrieval of all logical records for the new objects may then be performed through repeated 'retrieve object record' calls, as specified above.
- For sequential retrieval requests, the first logical record of the next sequential NearArchive object in the database will be returned by ASM for S/390 in the above circumstances. In this case, the object's primary key and archive date will be updated in the NearArchive parameter area to correspond to that of the new object. The application may then continue to issue 'retrieve object record' requests to proceed with sequential processing of the database.
- Note that an application may switch from direct to sequential processing of a database, and vice-versa, by altering the parameters in the 'retrieve object record' call. This type of access is termed 'skip-sequential'.
- When all retrievals from a database have been completed, the database should be disconnected with a 'disconnect database' call (request code '49'). A database may be disconnected at any time, irrespective of whether the previous retrieval processing has completed successfully or not.

Figure 5.3 below illustrates the above processing sequence.



**Fig 5.3:** Batch retrieval procedure.

### **5.3.2 Operational considerations.**

NearArchive batch retrieval processing will dynamically allocate and de-allocate a single tape drive unit in response to each batch request processed. In this context, the following points concerning the NearArchive operational environment should be noted:

- a) There may be a substantial number of allocation and de-allocation requests made per batch run, with a similar number of tape mount and dismount requests. To avoid excessive manual intervention, and to provide a guaranteed level of service for these operations and for batch turnaround times, it is recommended that NearArchive batch retrieval be implemented in an automated tape processing environment.

Alternatively, tape mount/dismount activity may be reduced by sorting the batched retrieval requests prior to processing, so that all requests to the same volume are processed consecutively. The location (dataset sequence number and block identifier) of any NearArchive object may be identified from the formatted entries returned via a 'query index' (request code '41') call to NearArchive.

- b) During dynamic allocation of tape files for batch retrieval, it is possible that a required tape volume is already allocated for an online retrieval request, or to another batch retrieval job. In this case, the allocation request will fail. However, the NearArchive retrieval routines will recognize this occurrence and will wait for one minute before re-issuing the allocation request.

The number of allocation retries may be controlled by the application via the 'allocation retry limit' field in the database open request. The default for this value is 30. ASM for S/390 issues message OTS35201 to the system log for each failed allocation in these circumstances; this message will identify the number of retries left.

No user intervention is required during this procedure. However, it may mean that batch retrieval jobs will take a little longer to complete if this type of tape volume contention is encountered.

- c) Allocation recovery may occur when there are no available tape drives online to the system on which the batch retrieval program is running. This will be manifested through the display of an IEF238D operator reply message.

If this happens, the system operator should take one of the following actions:

1. Vary online another tape drive, and reply to the message with the drive's device number - note that this device number must have been listed in the preceding IEF247I message as an eligible device.

2. Reply 'CANCEL'.

For action (1), a mount request will be issued on the newly-online tape drive.

For action (2), the program will perform allocation retry processing after a wait of 1 minute, as discussed in (b) above. If a tape drive is now available, processing continues as normal. If a unit is still not available, allocation recovery re-occurs and the above process is repeated. This will continue until the maximum number of retries (as set by the application during database open processing) has been reached, or the operator cancels the job.

In order to avoid increased batch turnaround times in these circumstances, due to excessive waits for operator replies, it is advisable to use automated operator reply processing, if available, to reply 'CANCEL' to the above message.

### **5.3.3 Logging of retrieval activity.**

Retrieval of objects from an individual database by batch processing programs may optionally be logged by ASM for S/390, using the OS/390 System Management Facility (SMF). This facility is controlled using the "Retrieval logging options" on the base database definition and maintenance panels in the NearArchive database administration facility. Refer to chapter 3 for detailed information on this process.

To enable SMF logging of batch object retrieval requests from a database, set the following panel fields:

- "SMF record identifier" – set this to the value of the SMF record identifier to be used by ASM for S/390 when logging retrievals to the SMF datasets. This should be a value in the range 128-255 which is reserved for use by ASM for S/390. A value of 0 in this field indicates that no SMF logging is to be performed.
- "Batch retrieval logging on?" – set this to the value 'Y' to enable logging of batch retrieval requests.

Once enabled, ASM for S/390 will write one SMF record for each object retrieved from that database. Refer to chapter 8 for a description of the SMF record format. Customers may process this information as required

in order to produce statistics relating to object retrieval activity and performance.

For direct and sequential batch retrieval processing, the SMF record is written when the first record of an archived object has been retrieved and presented to the calling application, in response to a 'retrieve object record' request (request code '42'). Timestamps in the request section of this record will be set as follows:

- Timestamp 1 - the time that the request to retrieve the first record of an object was received by ASM for S/390.
- Timestamp 2 - this timestamp has no special significance for batch retrievals, and will contain a copy of timestamp1.
- Timestamp 3 - this timestamp has no special significance for batch retrievals, and will contain a copy of timestamp1.
- Timestamp 4 - the time that the first record in the object was passed back to the calling application.

This page is intentionally left blank

---

## CHAPTER 6

---

# *CICS Processing*

This page is intentionally left blank



ASM for S/390 provides a facility to enable online recall of archived data from CICS applications. Recall requests are passed to ASM for S/390 from user application programs via a command-level interface, using the CICS 'LINK' command. This is intended to enable integration of ASM for S/390 tape recall processing into standard CICS application processing, whether existing or planned, to provide a comprehensive data enquiry system, covering both standard and NearArchive databases.

ASM for S/390 will eliminate restrictions imposed by CICS File Control Program processing regarding support of tape datasets, within the context of ASM for S/390's own requirements. Controls are provided within the design of the product to ensure that accessing tape files from within a CICS region has no general impact on CICS response times. The performance of non-ASM for S/390 functions in the same CICS address space will not be affected.

## **6.1 Application Programming Issues.**

---

### **6.1.1 Overview of Object Retrieval Processing.**

To retrieve data from a NearArchive database, a CICS application program should perform the following steps:

- Identify the primary key and archive date of the NearArchive object to be retrieved, and the name of the database from which retrieval is to be performed. The primary key and date of the object to be retrieved may be obtained by requesting the user to input these details on the screen, or (more likely) from user selection of the object from a formatted display of all NearArchive objects for a specific primary or secondary database key.

In this latter case, a formatted list of NearArchive index entries for a primary or secondary key may be obtained by issuing a NearArchive 'query index' request. This may be used to construct a formatted list of archived items for user selection from a display screen.

For DBCS customers, the primary (and secondary) keys should include all shift control bytes to avoid problems when printing reports.

- The application program should then prime a parameter block with the values obtained in the previous step, the name of a CICS temporary storage queue which is to be used to hold the retrieved data,

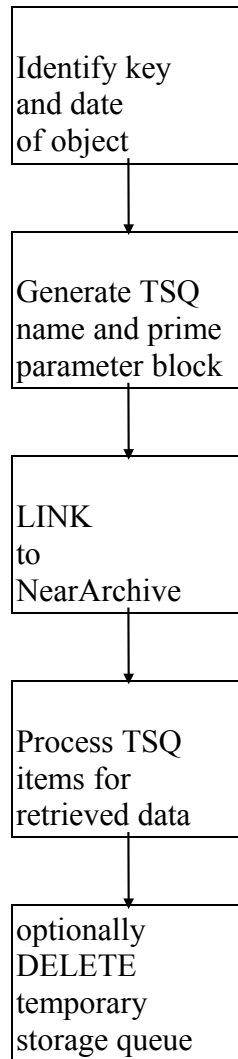
and indicate whether the queue is to be created in main or auxiliary temporary storage.

- Control is passed from the application program to ASM for S/390 via a CICS 'LINK' command to program OTASP200. The parameter interface block is passed in the CICS communication area. On return from ASM for S/390, a return code in the parameter interface block will indicate the result of the request.
- For successful requests, the specified temporary storage queue may be processed as desired by the application program (e.g.) retrieved data may be formatted and displayed on an enquiry screen. Each logical record within the retrieved NearArchive object will be held as a separate item in the temporary storage queue. Items will be held in the physical sequence in which they are stored in the NearArchive database.

Deletion of this temporary storage queue after completion of processing may be performed by the calling application, or alternatively may be deleted automatically by ASM for S/390, as controlled by the 'TSQDELETE' facility (see section 2.8.6).

Repeated retrieval requests to objects which have already been retrieved to a temporary storage queue will be satisfied from the retrieved temporary storage copy, until all temporary storage queues containing copies of that object have been automatically deleted by ASM for S/390, using 'TSQDELETE' processing.

The application processing sequence for recall of data from tape via ASM for S/390 is shown in Figure 6.1.



**Fig 6.1:** Online retrieval procedure

### **6.1.2 Partial Object Retrieval.**

ASM for S/390 supplies a facility to allow applications to retrieve a segment of an archived object. This facility is termed ‘partial object retrieval’. The segment to be retrieved is identified using two co-ordinates:

- Start record number. This is the number of the first record to be retrieved from within the object. Numbering of records starts at 0 and is incremented by 1 for each subsequent record in the object.
- The number of records to be retrieved. This is a count of the number of records in the segment. A count of 0 indicates that the segment continues from its starting location to the end of the object.

The partial object retrieval facility is intended for use by applications when retrieving data from a NearArchive database containing objects which hold more than one individual element of information.

For example, a database may have been defined for archival of financial transaction information, where each object contains details of 1,000 transactions with one record per transaction, each record being of size 20,000 bytes. Each object size in this case will be approximately 20 megabytes. If it is required that a CICS transaction should access information for a single transaction, then with full object retrieval the application would need to retrieve the 20Mb object to CICS temporary storage, and process it sequentially to locate the required transaction record. Using partial object retrieval, the application would identify the transaction to be retrieved (in terms of its segment co-ordinates). Only the 20kb for that transaction would be written to CICS temporary storage; the retrieved object may then be processed in its entirety by the application.

Where appropriate, partial object retrieval can give the following benefits over full object retrieval:

- Reduced NearArchive indexing requirements.
- Reduced data transfer time during object retrieval. Processing of an object by ASM for S/390 will terminate when all data in the requested segment has been retrieved.
- Reduced CICS temporary storage requirements.

Usage of the partial object retrieval facility by an ASM for S/390 application requires that information be held on the contents of each object in the database (i.e.) a 'sub-index' containing more detailed information about the contents of the database than that held in the NearArchive primary or secondary indexes. This information may be stored externally to the indexed object or the NearArchive database. Alternatively, the information on the contents of an object may be stored within the object itself (e.g.) a record at the start of an object identifying the segments held within the object, and the co-ordinates of each segment.

### **6.1.3 TSQ stringing.**

The NearArchive object retrieval process will copy a retrieved object to a CICS temporary storage queue (TSQ) in main or auxiliary storage (as indicated by the calling application). Each record in the object will be stored as a single entry in the queue. Records are stored in the TSQ in the order in which they appear in the object. However, a single CICS TSQ may only hold up to 32,767 entries. Objects with more than this number of records may consequently not be held in a single queue. To cater for objects with more than 32,767 records ASM for S/390 uses a facility called 'TSQ stringing'.

The TSQ stringing facility enables NearArchive objects containing more than 32,767 records to be processed by a CICS application. In these circumstances the TSQ identified by the calling application in its call parameters will contain the first 32,766 records from the object. The final (32,767<sup>th</sup>) entry in the TSQ will contain a pointer to the next TSQ in the string. Object storage will then continue from the start of the second TSQ. This pattern is repeated until all records from the retrieved object have been stored in CICS temporary storage. A maximum of 256 TSQs may be used to hold the retrieved object.

ASM for S/390 indicates to a calling application that stringing of TSQs has occurred in response to an object retrieval request by setting a return code of '04' and setting a value of 4 (for normal retrieval) or 6 (if an exception condition occurs during partial object retrieval) in supplementary code 1. Refer to chapter 8 for a detailed specification of these codes.

Applications which process databases where the size of an object is likely to exceed 32,767 entries should be coded to cater for the return and supplementary codes issued above. On receipt of this condition, the last entry in the TSQ identified in the application request will contain the following 16-byte record:

Offset	Length	Type	Description
0	8	Char	This field will always contain the character string 'NEXTTSQ='.
8	8	Char	The identifier of the next temporary storage queue in the string.

After processing all preceding entries in the queue as required, the identifier of the next TSQ in the string should be retrieved from the pointer record. Processing may then continue as required on the new TSQ.

This procedure should be repeated until all TSQs in the string have been processed. The last TSQ in the string will always contain less than 32,767 entries. There will be no pointer record at the end of this TSQ.

## 6.2 Invoking NearArchive from CICS programs.

---

CICS requests for access to a NearArchive database are issued using the CICS 'LINK' command, as follows:

```
EXEC CICS LINK PROGRAM('OTASP200')
      COMMAREA(PARMAREA)
      LENGTH(nn)
      END-EXEC.
```

where 'PARMAREA' is the user-defined storage area containing the parameter interface block used for communication between ASM for S/390 and the calling application and 'nn' is the length of the parameter interface block passed to ASM for S/390.

Refer to section 4.1 for more information on invoking ASM for S/390 from a CICS program using the new application program interface introduced with version 2.4. Refer to appendix D for information on invoking ASM for S/390 using the pre-v2.4 application program interface.

## 6.3 Operational Considerations.

---

### 6.3.1 General Issues.

- By nature of its online environment, a request for online retrieval via ASM for S/390 can occur randomly at any time during the online day. In a normal operational environment, it is possible that when such a request is received, there are no available tape drives on the OS/390 system from which the request was issued. When this happens, message IEF238D will be issued.

In order to satisfy the dynamic allocation request, an operator must vary online one of the devices identified in the preceding IEF247I message, and reply with that device number. A mount message will then be issued in the normal manner, and can be processed by the automatic tape library.

An operator may reply 'CANCEL' to the above message to deny the allocation request. This will be interpreted by ASM for S/390 as indicating that recall cannot take place at the moment. ASM for S/390 will then raise an 'archiving busy' condition (return code '21') to the caller.

In order to avoid any delays due to excessive waits from outstanding operator replies, it is recommended that some sort of automated operator reply processing should be implemented on the system under which ASM for S/390 is operating. This should be set to automatically reply 'CANCEL' to the IEF238D message. Users may then be informed immediately that NearArchive is currently busy and that the request should be re-submitted later.

- If a tape volume is unavailable for allocation at the time of the online retrieval request due to its being currently allocated to a batch retrieval request, ASM for S/390 will generate an 'archiving busy' condition (return code '21').
- Use of the online processing drive retention parameters (as specified during database initialization or maintenance processing) will enable users to identify tape cartridge volumes within a NearArchive database which are not to be automatically dismounted after all active retrieval requests have been processed

Subsequent requests for retrieval of data from this category of volume may then be processed more rapidly by eliminating the need to mount

and load a tape cartridge in order to access the requested data. See chapter 3 of this manual for a description of the definition and use of these parameters.

- In order to allow normal CICS termination of the NearArchive control region, it is essential that NearArchive is disabled in its control region before or during normal CICS shutdown processing, either by explicitly running transaction 'OT24' prior to shutdown, or by executing program OTASP240 in the shutdown PLT. Failure to perform either of these actions will result in the NearArchive CICS control region abnormally terminating with an 'A03' system abend during normal shutdown.

If a PLT entry is used to disable NearArchive, it is important that an 'immediate' shutdown is not performed, as this will bypass PLT processing (i.e.) use 'CEMT P SHUT' whenever possible to shutdown CICS processing.

- The 'TIMEOUT' control region initialization parameter may be used to terminate a stalled retrieval request. A retrieval request becomes stalled when it is waiting for completion of an external event (e.g. outstanding tape mount, I/O request to device etc.). Requests which are in this condition for longer than the time interval specified in the 'TIMEOUT' parameter will be terminated by ASM for S/390 and control returned to the calling application with a return code of '29'.

### **6.3.2 Concurrent Retrievals - Different Tape Cartridge Volumes.**

ASM for S/390 can handle multiple simultaneous requests for data retrieval from different tape volumes. All such requests are processed by a NearArchive control region, whether issued by applications running within the control region itself, or from any other region connected by the CICS Multiple Region Operation (MRO) facility.

The maximum number of tape volumes which may be processed concurrently by a NearArchive control region is set by the current value of the MAXDRIVE initialization parameter. This value is established during NearArchive control region initialization processing from the corresponding parameter value in the OTASS100 parameter file. The default value is 4.

The MAXDRIVE setting may be altered during CICS operation via the NearArchive CICS operator interface function (see section 6.6 for a description of the NearArchive CICS operator interface).



If the maximum number of tape drives to be used (as set by MAXDRIVE) is currently allocated for processing of other tape cartridge volumes, a request for retrieval from a new tape volume will be rejected with return code '21'.

If ASM for S/390 has allocated the maximum number of tape drives to be used (as set by MAXDRIVE), but one or more of the mounted tape cartridge volumes are not currently being processed (due to their being retained on the drive after last use, as specified by the online processing drive retention parameters for a database), then ASM for S/390 will dismount the least recently used volume in order to satisfy the new request. This request will proceed normally, but will experience some additional delay due to the need to dismount another tape cartridge volume before the volume it requires may be mounted.

### **6.3.3 Concurrent Retrievals - Same Tape Cartridge Volume.**

If more than one concurrent request for data retrieval from one particular tape cartridge volume is required, ASM for S/390 will queue the requests in the order received, up to the limit specified by the active value of the MAXQLEN control parameter. The value of MAXQLEN is established during NearArchive control region initialization processing from the MAXQLEN parameter in the OTASS100 parameter file. It may subsequently be amended via the NearArchive CICS operator interface function. See section 6.5 for details of the NearArchive CICS operator interface.

On receiving a retrieval request, ASM for S/390 first of all identifies the tape cartridge volume containing the requested data. The status of all retrieval tasks in the system is then examined to check if the required tape cartridge volume is currently mounted. If not, then a request is issued to a free retrieval task, which then allocates and processes the tape cartridge. If no free task is available, an 'archiving busy' condition is set (return code = '21').

If the required tape cartridge volume is currently being processed, a new request element is added to the retrieval task request queue. The length of this queue is limited by the active value of the MAXQLEN parameter. Any request above this maximum will generate an 'archiving busy' condition (return code = '21').

The second and subsequent requests in a queue will not be processed until the first request is complete. This means that these requests may receive longer response times. However, this is entirely dependent on when the request arrives in the queue. If, when a new request arrives, the time taken

to complete the preceding request is low, then the new request will be processed with only a short delay.

The only additional delay in processing second and subsequent requests will be for repositioning the tape at the required data block. There will be none of the delays for tape mounting and loading which may be associated with the first request.

Note that when a request has been processed, it will be removed from the queue, and the number of queue elements reduced by one. Any other request arriving will now be added to the end of the queue, and processed as above.

#### **6.3.4 Multiple NearArchive Database Support.**

Multiple NearArchive databases may be accessed from any one CICS system. The name of the database to be accessed is identified by the application program in the NearArchive call parameters.

A NearArchive database name is a maximum of four characters in length, and will be identical to the last four characters of the associated database primary index entry, as specified in the CICS File Control Table (FCT). The first four characters of the primary index entry are always set to 'OTAS'.

For each NearArchive database, a number of NearArchive system datasets must be defined in the FCT of the NearArchive control region. Refer to section 2.8 for a description of all CICS table definition requirements.

Note that a NearArchive CICS control region is enabled and disabled for NearArchive operation independently of any one particular database (i.e.) a control region need only be enabled once, irrespective of the number of different databases to be accessed.

## **6.4 Starting and Stopping NearArchive During CICS Operation.**

---

The NearArchive CICS control region is normally initialized at CICS start-up time, and quiesced at CICS shut-down, via entries defined in the CICS PLT's. However, NearArchive control region processing may be stopped or started at any time during normal CICS operation.

This is performed via the supplied transactions OT23 and OT24. Issuing transaction identifier OT23 in a NearArchive control region will initialize that region for archive recall. Issuing transaction identifier OT24 will disable archive recall in that region.

If any changes are required in the configuration of any of the databases being accessed via ASM for S/390 (e.g.. changes in the value of the drive retention parameters), it is not necessary to quiesce the control region. It is sufficient to close the primary index for the database to be amended in the NearArchive control region, update the database definition via the NearArchive ISPF database administrator panels (see chapter 3), and reopen the primary database index in CICS.

However, this procedure should be undertaken with caution, as any transaction attempting to retrieve data from the database being amended will receive an error return code while the primary index is closed.

Note that enabling or disabling the control region for ASM for S/390 operation will enable or disable access to all NearArchive databases normally accessed from that region.

It should also be noted that secondary NearArchive regions (i.e. CICS regions which contain only the NearArchive user interface processing), do not require initialization in the manner discussed here.

#### **6.4.1 OT23 - Initializing a NearArchive Control Region.**

In order to initialize a NearArchive control region, an operator should enter the transaction identifier 'OT23' on a CICS terminal in that region. (Note that this will only be necessary if NearArchive has not been initialized via a PLTPI entry, or if NearArchive has been shutdown during CICS operation).

The NearArchive control region will now initialize, and will be available to service NearArchive retrieval requests from the control region itself, or from any other CICS region which is linked to the control region via a MRO cross-memory or IRC connection.

Refer to chapter 8 for a description of messages returned from this request.

#### **6.4.2 OT24 - Quiescing a NearArchive Control Region.**

In order to quiesce NearArchive processing in the control region, the operator should enter the transaction identifier 'OT24' on a CICS terminal.

NearArchive processing will then be unavailable in the control region, or from any other CICS regions which may be linked to the control region via a MRO cross-memory or IRC connection.

Refer to chapter 8 for a description of messages returned from this request.

## **6.5 NearArchive Operator Interface.**

---

In order to enable system operators to monitor the status of NearArchive tape retrieval processing, a CICS operator interface function is supplied with the product.

CICS operator interface processing runs within the NearArchive control region, and is invoked via the supplied transaction 'OT26'.

The NearArchive CICS operator interface is available in line-entry processing mode, or in full-screen processing mode.

In line-entry processing mode, the CICS operator interface provides the following facilities:

- The ability to dynamically increase or decrease the active value of the MAXDRIVE parameter during CICS operation.
- The ability to dynamically increase or decrease the active value of the MAXQLen parameter during CICS operation.
- The ability to dynamically increase or decrease the active value of the MAXDISK parameter during CICS operation.

In full-screen processing mode, the CICS operator interface provides the following facilities:

- A formatted summary of the status of all currently initialized tape and disk reader tasks within the NearArchive control region.
- A detailed formatted display of the status of any one of the currently initialized tape or disk reader tasks in the NearArchive region, including details on all requests queued for processing by that task.
- The ability to purge, or force purge, any one of the currently initialized tape or disk reader tasks, from the formatted task summary display screen.

- The ability to dynamically increase or decrease the active value of the MAXDRIVE parameter during CICS operation.
- The ability to dynamically increase or decrease the active value of the MAXQLEN parameter during CICS operation.
- The ability to dynamically increase or decrease the active value of the MAXDISK parameter during CICS operation.
- The ability to dynamically increase or decrease the active value of the TSQDELETE parameter during CICS operation.
- The ability to dynamically increase or decrease the active value of the TAPEWAIT parameter during CICS operation.
- The ability to dynamically modify the active value of the TIMEOUT parameter during CICS operation.

The NearArchive operator interface may only be invoked from within an initialized NearArchive control region. Any attempt to use this facility from another region, or from within a control region which has yet to be initialized for retrieval, will be rejected.

### **6.5.1 Invoking the Operator Interface in Line-Entry Processing Mode.**

To invoke the operator interface in line-entry processing mode, it is necessary to sign on to the CICS system being used as the NearArchive control region.

After successfully signing on, enter the transaction identifier 'OT26' in the top left hand corner of the screen, followed by the operator interface command, as described below. The command may be separated from the transaction identifier by zero or more spaces. The total length of the transaction identifier and command cannot exceed 80 characters.

The following commands may be entered:

#### **SET**

<pre>S(ET)      { MAXDRIVE = nnn             { MAXQLEN  = nnn             { MAXDISK  = nnn</pre>
--

The SET command is used to alter the current values of the NearArchive MAXDRIVE, MAXQLEN and MAXDISK parameters. The command may be abbreviated to the single character 'S'. 'nnn' must consist of 1-3 numeric digits in the range 1-256.

The values of MAXDRIVE, MAXQLEN and MAXDISK are initially set during NearArchive control region initialization from the corresponding parameters in the OTASS100 file in the CICS start-up deck (or from defaults if OTASS100 is not present).

Valid parameters for the SET command are:

**MAXDRIVE** - ASM for S/390 will alter the current setting of the MAXDRIVE parameter in this NearArchive control region to the value specified in the SET command.

If the value of MAXDRIVE is to be increased, ASM for S/390 will increase the number of initialized tape reader tasks, by restarting terminated tasks, and/or adding new tasks.

If the value of MAXDRIVE is to be decreased, ASM for S/390 will terminate a number of existing tape reader tasks equal to the decrease in the value of MAXDRIVE. These tasks will appear with the status TERM in any subsequent status display, and will not be available for object recall processing.

Note that processing of this command may take a little time for significant changes in the value of MAXDRIVE.

**MAXQLEN** - ASM for S/390 will alter the current setting of the MAXQLEN parameter to the value specified in the SET command.

**MAXDISK** - ASM for S/390 will alter the current setting of the MAXDISK parameter in this NearArchive control region to the value specified in the SET command.

If the value of MAXDISK is to be increased, ASM for S/390 will increase the number of initialized disk reader tasks, by restarting terminated tasks, and/or adding new tasks.

If the value of MAXDISK is to be decreased, ASM for S/390 will terminate a number of existing disk reader tasks equal to the decrease in the value of MAXDISK. These tasks will appear with the status TERM in any

subsequent status display, and will not be available for object recall processing.

Note that processing of this command may take a little time for significant changes in the value of MAXDISK.

Refer to chapter 8 for a description of messages returned by the NearArchive CICS operator interface facility.

Refer to section 2.8.6 for information on usage of the CICS control region initialization parameters.

Examples:

```
OT26 S MAXDRIVE=8
```

```
OT26 S MAXQLEN=8
```

```
OT26 S MAXDISK=16
```

### **6.5.2 Invoking the Operator Interface in Full-Screen Processing Mode.**

To invoke the operator interface in full-screen processing mode, it is necessary to sign on to the CICS system being used as the NearArchive control region.

After successfully signing on, enter the transaction identifier 'OT26' in the top left hand corner of the screen. The main NearArchive operator interface menu will then be displayed, as below:

OT26M00	NearArchive V2.5	YYYY/MM/DD
Online Retrieval Configuration		
TERMID: tttt		
USERID: uuuuuuuu		
Option:		
1 - Display tape retrieval status summary		
2 - Display tape retrieval status detail		
3 - Display disk retrieval status summary		
4 - Display disk retrieval status detail		
5 - Alter configuration		
OTS26000 SELECT OPTION AND PRESS ENTER		
PF3 - end		

Enter the option which corresponds to the function which it is wished to invoke:

- Option 1:** This will display a formatted summary of all currently initialized NearArchive tape reader tasks. A selection column will allow individual tasks to be selected for processing.
- Option 2:** This provides a formatted detail display of an individual NearArchive tape reader task. On entry to this option, tape reader task #1 will be automatically selected for display.
- Option 3:** This will display a formatted summary of all currently initialized NearArchive disk reader tasks. A selection column will allow individual tasks to be selected for processing.
- Option 4:** This provides a formatted detail display of an individual NearArchive disk reader task. On entry to this option, disk reader task #1 will be automatically selected for display.
- Option 5:** This option allows users to alter the current value of the MAXDRIVE, MAXQLEN, MAXDISK, TSQDELETE, and TAPEWAIT and TIMEOUT control parameters.



To exit from the main operator interface menu, press the PF3 or CLEAR keys.

### 6.5.3 Display Status Summary.

On selecting option 1 or 3 from the main menu, the following screen will be displayed. 'XXXX' in the header line will have the value 'Tape' if option 1 was selected, and 'Disk' if option 3 was selected:

[illegible]

For option 1, a summary of all currently initialized tape reader tasks will be formatted and displayed on this screen. For option 3, an identical display will be formatted for all currently initialized disk reader tasks.

The displayed fields will have the following contents:

**MAXDRIVE:**

This gives the currently active value of the MAXDRIVE initialization parameter. This sets the limit on the maximum number of tape drives

which ASM for S/390 will allocate to satisfy multiple simultaneous retrievals.

**MAXIMUM QLEN:**

This field gives the currently active value of the MAXQLEN initialization parameter. This determines the maximum number of concurrent requests which ASM for S/390 will queue for any one active tape volume.

**MAXDISK:**

This gives the currently active value of the MAXDISK initialization parameter. This sets the limit on the maximum number of disk reader tasks which ASM for S/390 will initiate to satisfy multiple simultaneous retrievals.

**TASK COUNT:**

The value in this field identifies the total number of reader task control blocks (tape for option 1, or disk for option 3) which have been allocated by ASM for S/390 for object retrieval processing. This value will be equal to the number of summary lines displayed for this option.

**Sel:**

This column is used by the operator to select a single task for processing (see below for details of available selection entries).

**Task:**

This column gives the identifier of each initialized NearArchive tape/disk reader task control block. This is a numeric value, in the range 1-256.

**Stat:**

This column identifies the status of each NearArchive retrieval task. It consists of two indicators, each consisting of two characters.

The first two characters can take the values:

**WT** - the task is currently waiting for work.

**BY** - the task is currently busy processing a retrieval request. The name of the dataset being processed by the task will appear in the 'Dataset in use' field.

**TM** - the task has been terminated, either by a reduction in the value of MAXDRIVE or MAXDISK, or by explicit termination from the operator interface.

The last two characters may take the values:

**SH** - task shutdown has been requested. This will be processed when all requests which were queued when the shutdown was requested have been completed.

**RT** - for a waiting tape reader task, this means that the last tape volume processed by this task has been retained on the drive. The 'Dataset in use' field identifies the name of the retained tape dataset, and the 'Last user' field identifies the time that the last retrieval from that tape volume was completed by ASM for S/390.

For a busy task, this indicates that the tape currently being processed by this task will be retained on the drive after all requests have been serviced.

This code will not occur for disk reader tasks.

**MB** - the task has paused during a multi-block retrieval request. Blanks in this field mean that none of the above conditions apply.

#### **Qlen:**

For busy tasks, this field gives the number of requests currently queued for that task. For waiting tasks, this value should always be 0.

#### **Dataset in use:**

This field identifies the name of the dataset currently being processed for busy tasks, and for waiting tape reader tasks for which the last tape used has been retained on the drive (status indicator 'WTRT').

#### **Last use:**

This field gives the time that the last ASM for S/390 retrieval operation was completed for waiting tasks which have not dismounted the last tape used (status indicator 'WTRT'). This time is taken from the timestamp that is used by ASM for S/390 to identify the least recently accessed tape when it is necessary to dismount a drive in order to satisfy a new request. This field is not used for disk reader tasks.

The last but one line of this display screen is used for displaying error and information messages.

The following keys may be used to control the operator interface procedure from this screen:

- PF3:** This will exit from the summary display screen, and cause the main operator interface menu to be re-displayed.
- PF7:** This will scroll back to display the previous set of NearArchive tasks, when there are more than 14 tasks initiated in the control region.
- PF8:** This will scroll forward to display the next set of NearArchive tasks, when there are more than 14 tasks initiated in the control region.
- ENTER:** This will process any selections made by the operator in the selection column. If no selections have been entered, the summary is updated and re-displayed.

A maximum of one selection is permitted per entry. Multiple selections will be rejected by ASM for S/390.

Selection characters may take the following values (in upper or lower case):

- S -** This selection is used to request a formatted display of processing details for the selected task. The task detail screen for that task is then displayed on the screen. This is an identical display to that provided by option 2 (for tape reader tasks) or option 4 (for disk reader tasks) from the main menu.
- P -** This is used to purge the selected task. The request will be rejected if the selected task is uninitialized or terminated, or if a purge request is already active for that task.

For waiting tasks, ASM for S/390 will purge the task before returning control to the operator. The MAXDRIVE count (for tape reader tasks) or MAXDISK count (for disk reader tasks) will be reduced by 1 after successfully purging a task.

For busy tasks, ASM for S/390 will queue a purge request, then return control to the operator. Any outstanding requests being processed by that task will be allowed to complete before the task is purged. The MAXDRIVE count (for tape reader tasks)

or MAXDISK count (for disk reader tasks) will be reduced by 1 when the task is eventually purged.

The last two characters of the task status field on the summary display will contain the value 'SH' to indicate that a purge request has been queued.

- F - This is used to force purge a task, when a normal purge request has not been successful. A force purge will only be allowed after a normal purge of a busy task has been requested, and will terminate the task immediately. Outstanding requests will not be allowed to complete, but will instead receive a return code '21' (archiving busy response).

6.5.4 Display Status Detail.

After selecting option 2 or 4 from the main menu, the following screen will be displayed. 'XXXX' in the header line will have the value 'Tape' if option 1 was selected, and 'Disk' if option 3 was selected:

OT26M02		NearArchive V2.5		YYYY/MM/DD
XXXX retrieval status - detail				
Task no: ZZ9				
Status: XXXX		Proc flag: XXXX	Current qlen: ZZ9	Last use: HH:MM:SS
DSN in use: XX				
Req	Block id	Block count	Term id	User id
ZZ9	XXXXXXXX	ZZZZ9	XXXX	XXXXXXXX
ZZ9	XXXXXXXX	ZZZZ9	XXXX	XXXXXXXX
ZZ9	XXXXXXXX	ZZZZ9	XXXX	XXXXXXXX
ZZ9	XXXXXXXX	ZZZZ9	XXXX	XXXXXXXX
ZZ9	XXXXXXXX	ZZZZ9	XXXX	XXXXXXXX
ZZ9	XXXXXXXX	ZZZZ9	XXXX	XXXXXXXX
ZZ9	XXXXXXXX	ZZZZ9	XXXX	XXXXXXXX
ZZ9	XXXXXXXX	ZZZZ9	XXXX	XXXXXXXX
ZZ9	XXXXXXXX	ZZZZ9	XXXX	XXXXXXXX
ZZ9	XXXXXXXX	ZZZZ9	XXXX	XXXXXXXX
ZZ9	XXXXXXXX	ZZZZ9	XXXX	XXXXXXXX
ZZ9	XXXXXXXX	ZZZZ9	XXXX	XXXXXXXX
XX				
PF3 - end		PF7 - backward	PF8 - forward	

Details of all the identified tasks will be formatted and displayed on this screen. The displayed fields will have the following contents:

**Task no:**

This is a numeric value, in the range 1-256, which is used to uniquely identify a NearArchive reader task. This field is unprotected, and may be updated with any numeric value prior to pressing the enter key, in order to display details of any other NearArchive reader task.

**Status:**

This field contains a code which identifies the current status of the task being displayed. This code can have the following values:

<b>NULL</b>	- the task has not yet been initialized
<b>WAIT</b>	- the task is waiting for work
<b>BUSY</b>	- the task is currently processing a request
<b>TERM</b>	- the task has been terminated

**Proc flag:**

This field gives additional information concerning the processing status of the identified task. It can take the following values:

<b>SHUT</b>	- the task will shut down after all active requests have been completed.
<b>MNTD</b>	- for a waiting tape reader task, this means that the tape dataset identified in the 'DSN in use' field was retained on the drive after the last retrieval request was completed. The next request for retrieval of data from this dataset may be processed by this task without any tape mount requirement. For a busy tape reader task, the tape currently being processed will not be dismounted after all queued requests have been completed.

This code will not occur for disk reader tasks.

<b>WTMB</b>	- the task is pausing during retrieval of a multi-block data item, to allow CICS temporary storage updating to take place.
-------------	--

**Current qlen:**

For busy tasks, this field gives the number of requests currently queued for that task. This number will be identical to the number of request details lines displayed via this option.

**Last use:**

For waiting tape reader tasks, which did not dismount the tape which was last used by that task (due to controls provided in the database definition), this field displays the time that the last request for retrieval from this dataset was completed.

This field is not used for disk reader tasks.

**DSN in use:**

For a busy task, this field identifies the name of the dataset which is currently being processed. For a waiting tape reader task with tape retained on drive (status 'WTRT') this field gives the name of the dataset which was last accessed by this task, and which was not dismounted after use. For all other tasks, this field will be blank.

**Req:**

This column contains a numeric value in the range 1-999 which identifies a request which has been queued for processing by this task. The request currently being processed will always appear first, and have the value 1. Subsequent requests will appear with ascending numeric identifiers, in the order in which they will be processed.

**Block id:**

This field gives the 8 hexadecimal-digit identifier of the tape or disk block which contains the start of archived data to be retrieved for this request. For tape reader tasks, it may be used to estimate the location on the tape of the data block to which ASM for S/390 will position when processing this request.

**Block count:**

This identifies the number of blocks on the tape or disk dataset containing data for this request.

**Term id and User id:**

These fields identify the CICS user making this request.

The last but one line of this display screen is used for displaying error and information messages. The following keys may be used to control the operator interface procedure from this screen:

**PF3:** This will exit from the detail display screen, and cause the main operator interface menu to be re-displayed.

**PF7:** This will scroll backward to display the previous set of requests for this task, when there are more than 12 requests queued.

**PF8:** This will scroll forward to display the next set of requests for this task, when there are more than 12 requests queued.

**ENTER:** If the task number displayed at the top of the screen has been updated, formatted details of the new task will be displayed. If the entered task number is invalid, or does not exist, an error message will be displayed.

If no task number has been entered, details of the existing task will be updated and redisplayed.

### **6.5.5 Alter NearArchive Configuration.**

After selecting option 5 from the main menu, the following screen will be displayed:

OT26M03	NearArchive V2.5	YYYY/MM/DD
Online configuration update		
Enter configuration changes below:		
Maximum length of request queue allowed : 999		
Maximum number of drives to be allocated : 999		
Maximum number of disk reader tasks : 999		
Retrieved object retention period (mins) : 9999		
Maximum wait time for tape requests (mins) : 9999		
Stalled request timeout value (mins) : 9999		
PF3 - end		



Current values of the MAXQLEN, MAXDRIVE, MAXDISK, TSQDELETE, TAPEWAIT and TIMEOUT control region initialization parameters will be displayed on entry to this option.

Any of these values may be amended by the NearArchive CICS operator. On pressing enter, the new values will be processed.

The invalid alteration of any of these parameters will be rejected, and an error message displayed on the last line but one of the display screen.

Valid alteration of either parameter value will cause ASM for S/390 to internally update these parameters, before returning control to the operator. The new values will be effective immediately, and will remain active until the NearArchive control region is quiesced. On next initializing the NearArchive control region, the values for these parameters supplied in the OTASS100 parameter file will once again become effective.

The displayed fields are as described below:

**Maximum length of request queue allowed :**

This gives the current value of the MAXQLEN parameter, which specifies the maximum number of concurrent requests which will be queued by the NearArchive control region for any one active tape dataset. A new value may be entered in order to amend the active MAXQLEN value. Valid entries must be numeric, and in the range 1-256.

Any request which causes the MAXQLEN value to be exceeded will be rejected by ASM for S/390 with return code 21.

**Maximum number of drives to be allocated :**

This gives the current value of the MAXDRIVE parameter, which specifies the maximum number of tape drives which the NearArchive control region will allocate simultaneously to satisfy online retrieval requests. A new value may be entered in order to amend the active MAXDRIVE value. Valid entries must be numeric, and in the range 1-256.

Note that any significant change in the value of this parameter may take a little time to be processed by ASM for S/390.

Any request which causes the MAXDRIVE value to be exceeded will be rejected by ASM for S/390 with return code 21.

**Maximum number of disk reader tasks :**

This gives the current value of the MAXDISK parameter, which specifies the maximum number disk reader tasks which the NearArchive control region will initialize to satisfy online retrieval requests. A new value may be entered in order to amend the active MAXDISK value. Valid entries must be numeric, and in the range 1-256.

This parameter controls the number of disk object retrieval requests which ASM for S/390 will process simultaneously. If all disk reader tasks are busy, any new disk retrieval request will be passed to the reader task which has the shortest request queue. This process will balance the length of the request queues for all disk reader tasks, and ensure optimum object retrieval performance.

**Retrieved object retention period (mins) :**

This gives the current value of the TSQDELETE parameter, specifying the number of minutes for which a temporary storage queue containing a retrieved object (in any CICS region in the MRO configuration containing the NearArchive control region) is to be retained after retrieval. A new value may be entered in order to amend the active TSQDELETE value. Valid entries must be numeric, and in the range 0-1440. A value of 0 disables the NearArchive temporary storage housekeeping facility and will cause all object retrieval requests to be satisfied from tape, irrespective of whether the object has previously been retrieved to CICS temporary storage.

**Maximum wait time for tape requests (mins) :**

This gives the current value of the TAPEWAIT parameter, specifying the maximum number of minutes for which object retrieval requests are to be queued internally if insufficient resources (e.g. tape drives) are available to process the request. ASM for S/390 will process requests queued internally in this manner when sufficient resources become available. A queued request will be rejected with a 'resource unavailable' condition (return code '21') if the maximum wait time (as specified in this parameter) is exceeded before sufficient resources become available.

A new value may be entered in order to amend the active

TAPEWAIT value. Valid entries must be numeric, and in the range 0-1440. A value of 0 disables internal queuing of requests.

**Stalled request timeout value (mins) :**

This gives the current value of the TIMEOUT parameter, specifying the maximum number of minutes for which object retrieval requests are to remain in a stalled condition (i.e. an indefinite wait on completion of an external event). A stalled request will be rejected with a return code of '29' if it remains in this condition for the time interval specified in this parameter.

A new value may be entered in order to amend the active TIMEOUT value. Valid entries must be numeric, and in the range 0-1440. A value of 0 disables timeout processing.

The following keys may be used to control the operator interface procedure from this screen:

**PF3:** This will exit from the configuration amendment screen, and cause the main operator interface menu to be re-displayed.

**ENTER:** This will process any changes made to either of the parameter values displayed on the screen. Any invalid changes will be rejected, and an error message will be displayed.

If no change to the displayed values has been made, the existing values will be redisplayed.

### **6.5.6 Exiting from operator interface processing.**

On exiting from operator interface processing, ASM for S/390 will normally clear the CICS screen, and display completion message OTS26008. The requesting user will then be left in native CICS mode, and may continue operation by entering any 4-character CICS transaction identifier.

ASM for S/390 also provides a mechanism to allow customers to automatically invoke other CICS transactions on exit from operator interface processing, if it is not required to leave the terminal operator in native CICS mode.

To invoke this mechanism, the customer should supply a NearArchive operator interface user exit program. This program must have the name 'OT26UXIT', be defined in the CICS processing program table (PPT) and be accessible from the standard CICS DFHRPL load library concatenation.

The NearArchive operator interface process will always attempt to transfer control to this program (via a CICS 'XCTL' command) before exiting. The user exit program OT26UXIT will then receive control, and may implement whatever processing is required by the customer (e.g.) starting another CICS transaction on the same terminal, linking to a system menu display program etc.

If the program OT26UXIT is not present in the CICS PPT, or is not accessible at execution time from the DFHRPL concatenation, the NearArchive operator interface process will continue with standard termination.

## **6.6 Logging of retrieval activity.**

---

Retrieval of objects from an individual database by CICS programs may optionally be logged by ASM for S/390, using the OS/390 System Management Facility (SMF). This facility is controlled using the "Retrieval logging options" on the base database definition and maintenance panels in the NearArchive database administration facility. Refer to chapter 3 for detailed information on this process.

To enable SMF logging of CICS object retrieval requests from a database, set the following panel fields:

- "SMF record identifier" – set this to the value of the SMF record identifier to be used by ASM for S/390 when logging retrievals to the SMF datasets. This should be a value in the range 128-255 which is reserved for use by ASM for S/390. A value of 0 in this field indicates that no SMF logging is to be performed.
- "CICS retrieval logging on?" – set this to the value 'Y' to enable logging of CICS retrieval requests.

Once enabled, ASM for S/390 will write one SMF record for each object retrieved from that database. Refer to chapter 8 for a description of the SMF record format. Customers may process this information as required in order to produce statistics relating to object retrieval activity and performance.

The SMF record is written after all records in the archived object have been written to CICS temporary storage in response to a 'retrieve object' request (request code '43'), immediately before control is returned to the calling application.

Timestamps in the request section of this record will be set as follows:

- Timestamp 1 - the time that the request to retrieve the object was received by ASM for S/390.
- Timestamp 2 - the time that the request was added to a tape or disk reader task queue.
- Timestamp 3 - the time that the request was read from its request queue by the tape or disk reader task.
- Timestamp 4 - the time that control was returned to the calling application, after all records in the retrieved object had been written to CICS temporary storage.

The intervals between successive timestamps in the SMF record will correspond to the following components in the CICS object retrieval process:

1. *(Timestamp 2) – (Timestamp 1)*. This is the time taken for ASM for S/390 to validate a request and add it to a tape or disk reader task queue.
2. *(Timestamp 3) – (Timestamp 2)* This is the time the request spent in the tape or disk reader task queue.
3. *(Timestamp 4) – (Timestamp 3)*. This is the time taken by the tape or disk reader task to locate the object and copy it to CICS temporary storage.
4. *(Timestamp 4) – (Timestamp 1)*. This interval gives the total response time for ASM for S/390 to process the object retrieval request.

This page is intentionally left blank

---

# CHAPTER 7

---

## *NearArchive Utilities*

This page is intentionally left blank



ASM for S/390 incorporates a number of batch utilities which provide a range of support facilities for controlling and maintaining a NearArchive database and its operating environment.

The supplied utilities, and the functions which they perform, are summarized below.

1. **OTASP010** – the NearArchive index print utility.

The NearArchive index print utility scans a NearArchive index and creates a print report containing formatted index entries for each object in the NearArchive database.

The utility is primarily intended to assist with diagnostic procedures after a problem has been detected accessing the database.

2. **OTASP020** – the NearArchive tape scan utility.

The NearArchive tape scan utility will scan a NearArchive tape dataset and create a print report containing formatted entries for each object in the dataset.

The utility is intended to assist with diagnostic procedures after a problems has been detected accessing the database.

3. **OTASP030** - the NearArchive database maintenance utility.

The NearArchive database maintenance utility processes a NearArchive database to reclaim tape dataset storage space made redundant due to NearArchive object deletion.

Tape cartridge volumes which contain less than a user-specified threshold of active objects will be recycled, active objects being moved from that volume to another location in the database. A recycled volume may then be returned to the appropriate scratch pool for re-use. All disk ('K') copy datasets associated with a recycled tape dataset will be deleted.

The utility will also optionally compress disk copy datasets which contain less than a user-specified threshold of active disk copy objects. This process will remove redundant space within those datasets occupied by objects that have been expired from the database, or whose disk retention period has expired.

4. **OTASP040** - the NearArchive object management utility.

The NearArchive object management utility processes a NearArchive database's primary index to identify objects to be expired, objects which are to be migrated from one storage level to another storage level in the database, and objects for which disk ('K') copies are to be deleted. Expired objects are deleted from the database. Migrated objects are moved from their existing location in the source storage level to a new location in the target storage level in the database. Disk copies of objects are deleted; an entire disk copy dataset will also be deleted when it does not contain any active object copies.

5. **OTASP070** - the NearArchive journal maintenance utility.

The NearArchive journal maintenance utility processes a NearArchive database's journal dataset (used if the audit trail facility has been enabled for that database), and produces a summary or detailed analysis of journal entries. The journal dataset is reset by the utility on successful completion of processing.

The NearArchive journal maintenance utility will also archive journal entries to a database's journal archive database, if journal archiving has been enabled for that database.

The utility can also produce an historical daily analysis (summary or detailed) of journal entries from the journal archive database, for a specified date interval.

6. **OTASP080** - the NearArchive secondary index update utility.

The NearArchive secondary index utility is used to perform the second phase of the two-phase asynchronous secondary index update facility.

The first phase of this process is performed during batch archival processing, when a sequential dataset is created containing all updates which are to be performed asynchronously to a specified database secondary index. This dataset is processed by the secondary index update utility, and all appropriate updates executed.

The asynchronous secondary index update procedure is complete after successful execution of the utility.

7. **OTASP110** - the NearArchive database backup utility.

The NearArchive database backup utility is executed after a NearArchive database has been updated via an application's batch archival procedure, or after execution of the NearArchive database maintenance or object management utilities.

The utility will create one or two full-volume copies of one or more primary NearArchive tape datasets, and maintain a record of the backups in the database's primary index. The utility may also be used to create one or two incremental backup datasets. These datasets will contain a backup of all objects which have been added to a primary tape dataset since the preceding execution of the backup utility.

Full-volume and incremental backup copies of a database which have been created by this utility will be available for use in recovery processing after loss of individual tape volumes, or loss of an entire data center.

The utility may also be used to create (or re-create) one or more disk ('K') copy datasets from a primary tape volume.

#### 8. **OTASP130** - the NearArchive database recovery utility.

The NearArchive database recovery utility is executed to recover one or more of a NearArchive database's primary tape cartridge volumes which are unavailable through media failure or loss.

It uses the database's recovery control entries (as entered by the NearArchive database administrator via the supplied ISPF dialogs) to identify all volumes in a database which are currently in recovery status.

The primary copy of each of these volumes will be re-created from a full-volume backup copy, from one or more incremental backup copies, or from a combination of both full-volume and incremental backup copies, and the volume automatically removed from recovery status.

Each of the above utilities is described in detail in the remainder of this chapter.

## **7.1 OTASP010 - the NearArchive Index Print Utility.**

The NearArchive index print utility will sequentially scan the primary index file for a NearArchive database and produce a print report containing one formatted entry for each object in the index. Execution parameters can be used to restrict scanning of the primary index to a range of primary key values, or to a maximum number of primary index entry records. A totals report giving details of the total number of live and deleted objects in the database is produced at end of processing.

The index print utility is primarily intended to be used for diagnostic purposes when a problem has been detected accessing objects in the database or with general database processing. The utility may also be run when a summary of the contents of a NearArchive database is required.

### **7.1.1 Functions.**

The index print utility performs the following functions:

- Sequentially processes the NearArchive primary index dataset specified in the JCL used to execute the utility. A formatted print entry is created for each object index entry. This may be for an active ('live') object, or for an object which has been logically deleted.
- The 'STARTKEY', 'ENDKEY' and 'COUNT' execution parameters are used to control the start and end points of the sequential scan process.
- At the end of sequential processing of the index, the utility will produce a report giving the total number of index entry records, live objects and deleted objects processed.

### **7.1.2 JCL Requirements.**

The following JCL may be used to execute the NearArchive index print utility.

```
//STEP1      EXEC PGM=OTASP010,REGION=4M,  
//          PARM=('COUNT=n',  
//              'STARTKEY=x',  
//              'ENDKEY=y',  
//              'NOSIE')  
//*  
//STEPLIB   DD    DSN=NearArchive.load.library,DISP=SHR  
//*
```

//OTASV100 DD	DSN=Database.primary.index.name, DISP=SHR
//SYSPRINT DD	SYSOUT=*

The following optional EXEC parameters may be specified:

**COUNT :** The COUNT parameter is used to restrict the number of main index entry (MIE) records processed during execution of the utility. The value ('n') of this parameter may consist of from 1 to 15 numeric digits, and specifies the maximum number of MIE records to be processed by the utility.

There will be one MIE record in the index for each unique primary key value in the database. An MIE record may index multiple objects sharing the same primary key value – each object will have a different archive date associated with it.

If this parameter is not specified, then processing of the utility will continue until the primary key value specified in the ENDKEY parameter has been processed, or until end of file is reached (if the ENDKEY parameter is not specified).

If the COUNT parameter is specified, then the ENDKEY parameter may not also be specified. A validation error is returned by the utility if both these parameters are present.

**STARTKEY:** The STARTKEY parameter is used to control the point in the index at which the sequential scan process is to start. The value ('x') specified in this parameter must consist of a string of characters equal in length to the primary keylength of the database being processed, and should identify the primary key value at which processing is to begin.

If specified, the utility will start processing the index at the main index entry (MIE) record containing this primary key value, or the MIE record containing the next highest primary key value, if the specified key is not present in the database.

If this parameter is omitted, processing will start from the beginning of the index file.

**ENDKEY:** The ENDKEY parameter is used to control the point in the index at which the sequential scan process is to end. The value ('y') specified in this parameter must consist of a string of characters equal in length to the primary keylength of the database being processed, and should identify the primary key value at which processing is to end.

If specified, the utility will end processing of the index after processing the main index entry (MIE) record containing this primary key value, or the MIE record containing the next lower primary key value, if the specified key is not present in the database.

If this parameter is not specified, then processing of the utility will continue until the number of MIE records specified in the COUNT parameter has been processed, or until end of file is reached (if the COUNT parameter is not specified).

If the ENDKEY parameter is specified, then the COUNT parameter may not also be specified. A validation error is returned by the utility if both these parameters are present. If the STARTKEY parameter is also specified, then the primary key value specified in the ENDKEY parameter must be greater than or equal to the primary key value specified in the STARTKEY parameter.

**NOSIE:** The NOSIE parameter is used to suppress printing of supplementary index entry (SIE) records from the index.

One or more SIE records will exist for each main index entry (MIE) record when secondary index processing has been enabled for the database. SIE records contain information which is used for cross-referencing purposes with a corresponding secondary index dataset during object deletion processing.

If the NOSIE parameter is omitted a print report entry will be created for each SIE record encountered during sequential scanning of the primary index file.

Dataset names should be set as follows for each DD entry:

**STEPLIB :** The name of the NearArchive product library, as unloaded during the product installation procedure (see chapter 2).

**OTASV100:** The name of the primary index of the NearArchive database to be processed by the utility. The utility will only process one NearArchive database for each execution.

**SYSPRINT:** The destination of the utility report file. If required, the report may be written to a sequential dataset, with record format = FBA, and a logical record length of 133 bytes.

### **7.1.3 Condition Codes.**

OTASP010 will set a condition code on completion. This code may take the following values:

- 0 - the utility has been executed successfully.
- 4 - a warning condition has been encountered during processing of the utility but execution of the utility continued. A warning message will have been written to the SYSPRINT dataset.
- 8 - an error condition has been encountered during processing of the utility but execution of the utility continued. An error message will have been written to the SYSPRINT dataset.
- 12 - a serious error has occurred during processing of the utility. Execution of the utility has been terminated. An error message will have been written to the SYSPRINT dataset.

### **7.1.4 Running the Utility.**

There are no special considerations relating to execution of the index print utility. The utility may process a primary index file which is being updated by another concurrent NearArchive process. However, in these circumstances the contents of the formatted print report may not be accurate.

### **7.1.5 Utility Processing Reports.**

The following reports will be written by the index print utility to the SYSPRINT dataset:

1. Execution parameter report. This will summarize the values of the execution parameters specified (or defaulted) for execution of the utility.

2. Formatted index entry report. This report will contain one entry for each object index entry encountered during sequential scanning of the index. The following information will be reported for each object:

- Primary key of object
- Archive date (in format YYYY-MM-DD)
- Sequence number of tape dataset containing start of object
- Sequence number of tape dataset containing end of object
- Number of tape blocks used to store object
- Identifier of first tape block of object
- Segment number (first byte of block id) of 'B', 'C' and 'D' copies of object. This information will only be present for pre-v2.5 format index entries.
- Storage level containing object
- Number of disk blocks used by object (zeroes if no disk copy of the object exists).
- Identifier of first disk block of object (zeroes if no disk copy of the object exists).
- Disk dataset segment containing start of object (not preset if no disk copy of the object exists).

If the NOSIE parameter has not been specified, the report will also contain one entry for each supplementary index entry (SIE) record encountered during sequential scan processing. This entry will identify the primary key and secondary index number with which the SIE record is associated. SIE records will only be present when secondary index processing has been enabled for the database. Refer to the description of the NOSIE parameter for additional information about supplementary index entry records.

3. Totals report. This report gives totals of records and objects processed during execution of the utility. The following totals are reported:

- Total number of main index entry (MIE) records processed
- Total number of live objects printed
- Total number of logically deleted objects printed
- Total number of supplementary index entry (SIE) records processed (not present if NOSIE is specified in the execution parameters, or if no SIE records have been processed)

Error or warning messages generated during execution of the utility will be written to the processing summary report. All messages will have the



message identifier 'OTS010xx', where 'xx' is a numeric value in the range 00-99. Refer to chapter 8 of this manual for a description of all messages issued by the index print utility.

#### **7.1.6 Utility Failure and Restart Considerations.**

There are no special considerations for rerunning the utility after an earlier failure. The job used to execute the utility may be resubmitted without modification when the cause of the original failure has been identified and rectified.

## **7.2 OTASP020 - the NearArchive Tape Scan Utility.**

The NearArchive tape scan utility will sequentially scan a tape dataset from within a NearArchive database, and create a report containing formatted details of each NearArchive object in the tape dataset.

The tape scan utility is primarily intended to be used for diagnostic purposes when a problem has been detected accessing objects in the database or with general database processing. The utility may also be run when a summary of the contents of an individual NearArchive tape volume is required.

### **7.2.1 Functions.**

The tape scan utility performs the following functions:

- Sequentially processes the NearArchive tape dataset specified in the JCL used to execute the utility. A formatted print entry is created for each object on the tape. A formatted entry will also be created for each commit block encountered on the tape.
- The 'STARTBLK', 'ENDBLK' and 'COUNT' execution parameters are used to control the start and end points of the sequential scan process.
- At the end of sequential processing of the tape, the utility will produce a summary report giving the total number of blocks, records and bytes on the tape.

### **7.2.2 JCL Requirements.**

The following JCL may be used to execute the NearArchive tape scan utility.

```
//STEP1      EXEC PGM=OTASP020,REGION=4M,
//          PARM=('COUNT=n',
//              'STARTBLK=x',
//              'ENDBLK=y',
//              'SUMMARY')
//          /*
//STEPLIB DD DSN=NearArchive.load.library,DISP=SHR
//          /*
//OTASV100 DD DSN=Database.primary.index.name,DISP=SHR
//OTAST100 DD DSN=Tape.dataset.name,DISP=SHR,UNIT=(, , DEFER)
//SYSPRINT DD SYSOUT=*
```

The following optional EXEC parameters may be specified:

**COUNT :** The COUNT parameter is used to restrict the number of tape blocks processed during execution of the utility. The value ('n') of this parameter may consist of from 1 to 10 numeric digits, and specifies the maximum number of tape blocks to be processed by the utility.

If this parameter is not specified, then processing of the utility will continue until the block number specified in the ENDBLK parameter has been processed, or until end of file is reached (if the ENDBLK parameter is not specified).

If the COUNT parameter is specified, then the ENDBLK parameter may not also be specified. A validation error is returned by the utility if both these parameters are present.

**STARTBLK:** The STARTBLK parameter is used to control the point in the tape dataset at which the sequential scan process is to start. The value ('x') specified in this parameter must be a numeric identifier of length 1-10 characters, and should identify the block number at which processing is to begin. Block numbering starts with the value 1 for the first block on the file, and is increased by 1 for each subsequent block.

If specified, the utility will start processing the tape dataset at the specified block number. If this parameter is omitted, processing will start from the beginning of the tape dataset.

**ENDBLK:** The ENDBLK parameter is used to control the point in the tape dataset at which the sequential scan process is to end. The value ('y') specified in this parameter must be a numeric identifier of length 1-10 characters, and should identify the block number at which processing is to end. Block numbering starts with the value 1 for the first block on the file, and is increased by 1 for each subsequent block.

If specified, the utility will end processing of the tape dataset after processing the specified block number, or at end of file if the dataset does not contain the number of blocks specified in this parameter.

If this parameter is not specified, then processing of the utility will continue until the number of blocks specified in the COUNT parameter has been processed, or until end of file is reached (if the COUNT parameter is not specified).

If the ENDBLK parameter is specified, then the COUNT parameter may not also be specified. A validation error is returned by the utility if both these parameters are present. If the STARTBLK parameter is also specified, then the block number specified in the ENDBLK parameter must be greater than or equal to the block number specified in the STARTBLK parameter.

**SUMMARY:** The SUMMARY parameter indicates that only summary reporting is required from utility execution. Printing of the formatted object report will be suppressed. If the SUMMARY parameter is omitted the formatted object report will be produced.

Dataset names should be set as follows for each DD entry:

**STEPLIB :** The name of the NearArchive product library, as unloaded during the product installation procedure (see chapter 2).

**OTASV100:** The name of the primary index of the NearArchive database to be processed by the utility. The utility will only process one NearArchive database for each execution.

**OTAST100:** The name of the tape dataset to be processed by the utility. The tape dataset specified in this entry must belong to the database whose primary index is specified in the OTASV100 DD entry. Unpredictable results may occur if this is not the case.

**SYSPRINT:** The destination of the utility report file. If required, the report may be written to a sequential dataset, with record format = FBA, and a logical record length of 133 bytes.

### **7.2.3 Condition Codes.**

OTASP020 will set a condition code on completion. This code may take the following values:

0 - the utility has been executed successfully.

- 4 - a warning condition has been encountered during processing of the utility but execution of the utility continued. A warning message will have been written to the SYSPRINT dataset.
- 8 - an error condition has been encountered during processing of the utility but execution of the utility continued. An error message will have been written to the SYSPRINT dataset.
- 12 - a serious error has occurred during processing of the utility. Execution of the utility has been terminated. An error message will have been written to the SYSPRINT dataset.

#### **7.2.4 Running the Utility.**

There are no special considerations relating to execution of the tape scan utility. However, the utility should not be executed if there are likely to be object retrieval or storage requests for the tape being processed while the utility is running. Such requests will be rejected until the utility has completed processing.

#### **7.2.5 Utility Processing Reports.**

The following reports will be written by the tape scan utility to the SYSPRINT dataset:

1. Execution parameter report. This will summarize the values of the execution parameters specified (or defaulted) for execution of the utility.
2. Formatted object report. This report will contain one entry for each object encountered during sequential scanning of the tape dataset. The following information will be reported for each object:
  - Primary key of object
  - Archive date (in format YYYY-MM-DD)
  - Object creation timestamp (time in format HH:MM:SS:th and Julian date in format CYYDDD)
  - Block number of first block containing object.
  - Total number of blocks used to store object.
  - Total number of logical records in object.
  - Length of largest block used to store object.
  - Length of longest logical record in object.

The report will also contain an entry for each commit block encountered on the tape. Commit blocks are written by ASM for

OS/390 when a database is first opened for update processing or in response to a commit request from a NearArchive application, and are used for re-positioning on tape during restart open processing or dynamic backout processing.

3. Summary report. This report gives totals of blocks, records and bytes processed during execution of the utility. The following totals are reported:

- Total number of bytes in tape dataset.
- Total number of blocks in tape dataset.
- Total number of records in tape dataset.
- Length of largest block in dataset.
- Length of longest logical record in dataset.

Error or warning messages generated during execution of the utility will be written to the processing summary report. All messages will have the message identifier 'OTS020xx', where 'xx' is a numeric value in the range 00-99. Refer to chapter 8 of this manual for a description of all messages issued by the index print utility.

#### **7.2.6 Utility Failure and Restart Considerations.**

There are no special considerations for rerunning the utility after an earlier failure. The job used to execute the utility may be resubmitted without modification when the cause of the original failure has been identified and rectified.

## 7.3 OTASP030 - the NearArchive Database Maintenance Utility.

---

When an object is deleted from a NearArchive database, or an object exceeds its retention period, its primary index entry will be modified to indicate that the object no longer exists. Index entries for deleted or expired objects in a NearArchive database will be physically removed during execution of the NearArchive object management utility OTASP040.

The NearArchive database maintenance utility OTASP030 must be executed to reclaim tape dataset storage within a NearArchive database that is occupied by a deleted or expired object. This utility will process all tape cartridge volumes within a single NearArchive database whose level of active objects has fallen below a user-specified threshold for that database.

All active objects from such a tape volume will be moved to another volume in the NearArchive database, and the recycled volume released from the database. Such recycled volumes may be returned to the appropriate scratch pool for re-use. Any disk ('K') copy datasets associated with a recycled volume will be automatically deleted after the volume has been released from the database. Note that if disk processing is enabled for a storage level which is being recycled, then disk copies will be created for all recycled objects. These disk copies will be deleted by the object management utility (OTASP040) when the age of the object (in days since first archival) exceeds the disk copy retention period, as specified during database administration processing.

The database maintenance utility may also be used to compress disk ('K') copy datasets which contain less than the compress threshold of active disk copy objects for the storage level being processed by the utility. Disk compress processing will remove unused space occupied by objects which have been expired from the database, or whose disk retention period has expired. The required compress threshold should be entered in the 'Disk compress threshold' field on the NearArchive database administration facility storage level update or definition panels.

The NearArchive object management utility (OTASP040) should always be executed prior to execution of the database maintenance utility so that primary index and space management file entries are correctly updated with details of expired and deleted objects (for both tape and disk copies).

This will ensure that all database control information is up-to-date prior to database maintenance being performed.

The database maintenance utility should be used at regular intervals to process databases from which objects have been deleted or have expired in order to reduce the overall tape cartridge holding.

The 'FORCE' execution parameter of the database maintenance utility may be used to force recycling of all current volumes in a database storage level and/or compression of all disk copy datasets, irrespective of the level of active objects on each tape or disk dataset. This option may be useful after a change has been made to a storage level's tape device type, in order to release all volumes created on the old device type, and to re-create these volumes using the new device type.

### **7.3.1 Functions.**

The database maintenance utility performs the following functions:

- Sequentially scans a NearArchive database's space management dataset, for one or all storage levels in the database, to identify tape cartridge volumes whose level of active objects has fallen below the recycle threshold defined for that database via the database administration ISPF dialogs and disk copy datasets whose level of active disk copy objects has fallen below the compress threshold defined for that storage level (see chapter 3).

The range of volumes to be scanned in each storage level in the database may be controlled through use of the STARTSEQ and ENDSEQ execution parameters.

- If the 'FORCE' execution parameter has been specified, **all** current datasets in a storage level will be selected for recycling and/or disk compression (depending on EXEC parameter).
- If 'RECYCLE=YES' has been specified (or defaulted) in the EXEC parameters, the utility will move all active objects from each tape dataset meeting the recycle criteria, and add them to the end of the NearArchive database.
- If 'COMPRESS=YES' has been specified (or defaulted) in the EXEC parameters, the utility will compress each disk dataset meeting the compress criteria, by removing unused space occupied by expired or deleted disk copy objects.



- The utility will produce a report giving details of tape volumes recycled and disk datasets compressed in the above manner. This report may be used by tape cartridge library administrators to control the status of recycled tape cartridge volumes.
- If the AUTOUNCAT feature has been enabled for the database being processed (specified via the database administration ISPF dialogs), the utility will automatically uncatalog a tape dataset when all active objects from that dataset have been successfully moved to another location in the database. This feature may be used in conjunction with a tape management software product, to automatically return the volume to scratch status.
- After a tape volume has been successfully released from the database, the utility will delete all disk ('K') copy datasets associated with the released tape dataset, if any exist.
- During disk compression processing, all disk allocation parameters defined for the storage level being processed (including primary and secondary allocation values and the release space indicator) will be honored. A compressed disk dataset will be re-created using these parameters.

Note that if a disk space failure (system abend 'X37') occurs, or the maximum disk blockcount is exceeded during compression processing, the compressed dataset will only contain disk copy objects processed up to that point. Active disk copy objects in the uncompressed dataset which have not been processed will no longer exist. NearArchive will satisfy future retrieval requests for these objects from tape storage.

To avoid disk space problems during compression processing, ensure that allocation parameters are set as required (via the NearArchive database administration facility) prior to execution of the database management utility.

- The utility will perform checkpoint processing after each selected tape or disk dataset has been successfully processed. Restart of the utility after failure or cancellation will cause processing to be resumed from the last successful checkpoint.
- After each execution of the utility, the NearArchive database backup utility should be executed to create backup copies of all tape datasets created or updated during utility processing.

### **7.3.2 JCL Requirements.**

The following JCL may be used to execute the NearArchive database maintenance utility.

```
//STEP1      EXEC PGM=OTASP030,REGION=4M,
//          PARM=('LEVEL=n,ALL',
//              'RESTART=YES|NO',
//              'FORCE',
//              'SCANTAPE=YES|NO',
//              'RECYCLE=YES|NO',
//              'COMPRESS=YES|NO',
//              'STARTSEQ=nnnnn',
//              'ENDSEQ=nnnnn')
//          *
//STEPLIB DD   DSN=NearArchive.load.library,DISP=SHR
//          *
//OTASV100 DD   DSN=Database.primary.index.name,DISP=SHR
//SYSPRINT DD   SYSOUT=*
//SYSOUT DD   SYSOUT=*
[/OTASS030 DD   DSN=Restart.dataset.name,DISP=SHR]
```

The following optional EXEC parameters may be specified:

**LEVEL :** Specify the LEVEL parameter to identify the database storage level to be processed by the utility. The parameter may take one of the following values:

**'n'** - process storage level 'n' only, where 'n' is a single numeric digit in the range 0-7.

**'ALL'** - process all storage levels in the database.

If this parameter is omitted, a default value of 'ALL' is used.

Where a database has only one storage level, specifying a value of 'ALL' for this parameter (or allowing it to default), will cause the utility to process storage level 0 only.

**RESTART:** Specify the RESTART parameter to control utility restart processing. The following parameter values may be specified:

**'NO'** - normal utility processing is to be invoked. This is the default value.

**'YES'** - restart utility processing is to be invoked. If the previous execution of the utility did not terminate successfully, the utility should be rerun with a RESTART parameter value of 'YES'. The utility will not execute after a failed run, unless 'RESTART=YES' is specified in the EXEC parameter. For normal running, the default value of 'NO' should be used.

**FORCE:** Specification of the FORCE parameter will cause the utility to recycle **all** current tape datasets in the storage level(s) being processed (if the 'RECYCLE=YES' EXEC parameter has been specified or defaulted), and/or compress **all** disk copy datasets in the storage level(s) being processed (if the 'COMPRESS=YES' EXEC parameter has been specified or defaulted). The percentage of active objects on each tape or disk dataset and the recycle or compress threshold specified for the storage level will not be significant when this option is used.

Omission of the FORCE parameter will implement standard execution of the utility (i.e.) recycle or compress datasets whose percentage level of active objects falls below the corresponding threshold supplied for the storage level being processed.

**SCANTAPE:** The SCANTAPE parameter controls processing of tapes to be recycled, when no active objects are registered for those tapes. The following parameter values may be specified:

**'YES'** - tapes selected for recycling for which no active objects are currently registered in the database's space management dataset should be scanned to perform fail-safe verification that all objects on the tape are inactive.

This is the default value, and should be used in all cases where there is no absolute certainty (from application considerations) that the tape to be recycled contains no active objects.

**'NO'** - tapes selected for recycling for which no active objects are currently registered in the database's space management dataset should not be scanned

to check for the presence of active objects. All such tape volumes will be released from the database (and optionally uncataloged) as normal.

Enabling this option may considerably reduce execution times for the database maintenance process, but should only be employed when there is absolute certainty that tape dataset of this type which are candidates for recycling will not contain any active objects.

**RECYCLE:** This parameter is used in conjunction with the 'COMPRESS' parameter to control the type of processing to be performed by the utility. The following parameter values may be selected:

'YES' - the utility is to perform tape dataset recycle processing. This is the default value.

'NO'- the utility is not to perform tape dataset recycle processing.

**COMPRESS:** This parameter is used in conjunction with the 'RECYCLE' parameter to control the type of processing to be performed by the utility. The following parameter values may be selected:

'YES' - the utility is to perform disk copy dataset compress processing. All disk copy datasets associated with each primary tape dataset in the range set (or defaulted) by the STARTSEQ and ENSEQ parameters will be processed to check if they are candidates for compression. 'COMPRESS=YES' may not be specified if 'RECYCLE=YES' has also been specified.

'NO'- the utility is not to perform disk copy dataset compress processing. This is the default value. 'COMPRESS=NO' may not be specified (or allowed to default) if 'RECYCLE=NO' has also been specified.

**STARTSEQ:** This parameter is used to specify the tape dataset sequence number at which scanning for eligible volumes/datasets for

recycling and/or compressing is to begin. If more than one storage level is being processed by the utility (as controlled via the LEVEL parameter) the start sequence number specified in this parameter will be honored for all storage levels. The utility will scan for eligible volumes/datasets up to the sequence number specified (or defaulted) in the ENDSEQ parameter.

‘nnnnn’ should be a 1-5 character numeric value in the range 0-65535, specifying the dataset sequence number (within each storage level) at which scanning is to start. If omitted, a default value of 0 is used. This indicates that scanning is to start from the beginning of each storage level being processed.

**ENDSEQ:** This parameter is used to specify the tape dataset sequence number at which scanning for eligible volumes/datasets for recycling and/or compressing is to finish. If more than one storage level is being processed by the utility (as controlled via the LEVEL parameter) the end sequence number specified in this parameter will be honored for all storage levels. The utility will scan for eligible volumes/datasets from the sequence number specified (or defaulted) in the STARTSEQ parameter.

‘nnnnn’ should be a 1-5 character numeric value in the range 0-65535, specifying the dataset sequence number (within each storage level) at which scanning is to end. If omitted, a default value of 65535 is used. This indicates that scanning is to continue to the end of each storage level being processed.

Dataset names should be set as follows for each DD entry:

**STEPLIB :** The name of the NearArchive product library, as unloaded during the product installation procedure (see chapter 2).

**OTASV100:** The name of the primary index of the NearArchive database to be processed by the utility. The utility will only process one NearArchive database for each execution.

**SYSPRINT:** The destination of the utility report file. If required, the report may be written to a sequential dataset, with record format = FBA, and a logical record length of 133 bytes.

**SYSOUT:** The destination of the print report issued by the internal sort performed during disk compress processing. This entry is only required if 'COMPRESS=YES' is specified.

**OTASS030:** This is an optional entry, which may be used to identify the dataset required by the utility for retaining restart formation during execution of a recycle process for the NearArchive database identified in the JCL. If specified, it must be a pre-defined disk dataset with the following attributes:

*Record format:* fixed blocked (FB)  
*Logical record length:* 24 bytes  
*Blocksize:* any multiple of 24 bytes.

The dataset must be large enough to contain a single 24-byte record. A unique restart dataset is required for each NearArchive database to be processed by the utility.

The contents of this dataset should not be modified in any way during or between executions of the database maintenance utility. The contents will be reset by the utility on successful completion of database maintenance processing. Care should be taken to ensure that any restart dataset supplied in the JCL used to execute the utility is uniquely assigned to the database being processed.

If this DD entry is omitted, the utility will dynamically allocate a disk dataset called 'hlq.RECYCLE', where 'hlq' is the high-level qualifier defined for the database being processed. This dataset will be allocated using the system unit name defined for the NearArchive database being processed, and will have a blocksize of 2400 bytes and primary and secondary allocation values of 1 block each. A dynamically created dataset will be automatically deleted at successful completion of database maintenance processing.

### **7.3.3 Condition Codes.**

OTASP030 will set a condition code on completion. This code may take the following values:

- 0 - the utility has been executed successfully.
- 4 - a warning condition has been encountered during processing of the utility but execution of the utility continued. A warning message will have been written to the SYSPRINT dataset.
- 8 - an error condition has been encountered during processing of the utility but execution of the utility continued. An error message will have been written to the SYSPRINT dataset.
- 12 - a serious error has occurred during processing of the utility. Execution of the utility has been terminated. An error message will have been written to the SYSPRINT dataset.

### **7.3.4 Running the Utility.**

Prior to executing the database maintenance utility, the following database definition parameters should be set as required:

- For tape recycle processing, the 'Recycle threshold' field in the database maintenance options section of the storage level definition panel (OTSN4301) or storage level amendment panel (OTSN4231) should be set as required.

This field should be set to a percentage value in the range 1-99 which establishes the recycle threshold for a database storage level. If the percentage of active objects on a tape dataset in that storage level falls below this threshold, the dataset will be eligible for recycling by the database maintenance utility.

A threshold value of 0 (the default value) indicates that no recycling is to be performed for the storage level.

Note that this entry will not be of significance when the 'FORCE' execution parameter is specified.

- 'Autouncat during recycle' in the database maintenance options section of the storage level definition or amendment panels.

If set to 'NO' (the default value), then recycled tapes are not automatically uncataloged by the database maintenance utility when released from the database.

If set to 'YES', recycled tapes will be automatically uncataloged when released from the database. This option may be used in conjunction

with a tape management software product, to automatically return released volumes to the appropriate scratch pool.

- For disk copy dataset compress processing, the 'Disk compress threshold' field in the database maintenance options section of the storage level definition amendment panels should be set as required.

This field should be set to a percentage value in the range 1-99 which establishes the compress threshold for a database storage level. If the percentage of active disk copy objects on a disk dataset in that storage level falls below this threshold, the dataset will be eligible for compression by the database maintenance utility.

A threshold value of 0 (the default value) indicates that no disk compression is to be performed for the database.

Note that this entry will not be of significance when the 'FORCE' execution parameter is specified.

Refer to chapter 3 for a description of the procedure required to control values of the above database maintenance parameters.

Execution of this utility for tape recycle processing will update tape datasets in the NearArchive database. The NearArchive database backup utility OTASP110 should therefore be executed after completion of the utility. This will create one or two backup copies of each tape cartridge volume created or updated during execution of the utility.

The database maintenance utility requires update access to the NearArchive database being processed, and consequently will not execute if the database is already open for update processing by another job. If the audit trail facility is enabled for the database being processed by the database maintenance utility, a journal entry will be created for each active object which has been moved from a recycled tape cartridge volume.

### **7.3.5 Operator commands.**

Execution of the utility may be terminated at any point by entering the following command through the system console:

```
(STO) P jobname
```

where 'jobname' is the name of the job used to execute the database maintenance utility.



The utility will acknowledge receipt of this command via message OTS03097. Execution of the utility will be terminated normally on completion of processing of the current input tape dataset (i.e. the tape dataset being processed when the stop command was entered) if recycle processing is being performed, or on completion of compression of the current disk copy dataset (i.e. the disk dataset being processed when the stop command was entered) if only compress processing is being performed. All database updates performed by the utility prior to termination will be committed. Utility processing may be continued at any subsequent time via normal job execution.

### **7.3.6 Utility Processing Reports.**

The following reports will be written by the database maintenance utility to the SYSPRINT dataset:

1. Execution parameter report. This report summarizes execution parameters, as specified on the JCL EXEC statement, or as defaulted.
2. Processing summary report. This report gives details of each tape volume released during recycle processing and/or each disk copy dataset compressed during compression processing. Totals of released volumes and/or compressed datasets are produced by storage level, and a total produced for all storage levels processed.

Error or warning messages generated during execution of the utility will be written to the processing summary report. All messages will have the message identifier 'OTS030xx', where 'xx' is a numeric value in the range 00-99. Refer to chapter 8 of this manual for a description of all messages issued by the database maintenance utility.

### **7.3.7 Utility Failure and Restart Considerations.**

If the database maintenance utility fails to complete successfully, either due to an error condition being encountered, or execution being cancelled by the system operator, it should be rerun, specifying an EXEC parameter of 'RESTART=YES'. Note that if execution was terminated early via entry of an operator 'STOP' command, the utility will terminate normally. For subsequent execution, the RESTART parameter should be set to 'NO', or allowed to default.

The utility will automatically checkpoint its activity after each volume to be recycled in the NearArchive database being processed has been successfully released from the database (if recycle processing is being performed) or after an eligible disk dataset has been compressed (if only

disk compression processing is being performed). During rerun of the utility, processing will be restarted from the last successful checkpoint. There is no requirement to restore any dataset prior to rerunning the utility.

The database maintenance utility has built-in processing to prevent normal processing by the utility of a NearArchive database, after failure of the previous database maintenance procedure for that database. In these circumstances, a parameter of 'RESTART=YES' **must** be present on the job's EXEC statement.

Note that as the database maintenance procedure may take some time to execute (depending on the number of tape cartridge volumes to be recycled or disk copy datasets to be compressed), execution may be terminated by the system operator at any time by entering a STOP command (see 7.1.5), in order to allow input or input-output access to the database from other jobs. Execution of the database maintenance procedure may subsequently be continued through normal execution of the utility, using an EXEC parameter of 'RESTART=NO' (or allowing this value to default).

Execution of the utility may also be controlled through use of the STARTSEQ and ENDSEQ execution parameters. These parameters may be used to limit the number of tape volumes or disk datasets to be processed in any one execution of the utility, or to control which section of a database storage level is to be recycled or compressed by the utility.

## 7.4 OTASP040 - the NearArchive Object Management Utility.

---

The NearArchive object management utility is used to automatically expire objects which have been stored in a NearArchive database for a period of time in excess of the retention period specified for that database.

For databases with multiple storage levels, the object management utility will also migrate objects from one storage level to another. Objects to be migrated are controlled by the migration periods that have been specified for all storage levels up to (but not including) the target storage level.

The utility also deletes the disk copy of objects which have been in the database for a length of time in excess of the disk retention period for the database storage level, as specified during database administration processing in TSO/ISPF. Any subsequent access to these objects will be satisfied from a tape copy of the object. Disk ('K') copy datasets which contain no active object copies at conclusion of utility processing will be automatically deleted.

The utility will process a single NearArchive database during each execution. The database to be processed is identified by specifying the name of its primary index dataset in the OTASV100 DD entry in the JCL used to execute the utility.

The object management utility should be run at regular intervals for databases which have become eligible for object expiration or for migration of objects between storage levels. The NearArchive database maintenance utility should be executed in conjunction with the object management utility in order to reclaim tape space occupied by expired or migrated objects.

The object management utility may be run in simulate mode to predict the object management actions (expiration or migration of objects) which will be performed by the utility. An option is available when running in this mode to simulate utility execution on any specific or relative date value.

### **7.4.1 Functions.**

The object management utility performs the following functions:

- Sequentially scans a NearArchive database's primary index, and identifies the following categories of objects:
  - a) objects which have been stored in the database for longer than the object retention period set for that database. These expired objects will be deleted from the database.
  - b) objects in the source storage level identified in the INLEVEL execution parameter, which have been stored in the database for longer than the sum of storage level migration periods for all storage levels up to (but not including) the target storage level identified in the OUTLEVEL execution parameter. All identified objects will be recorded on the database's migration control dataset (created automatically during storage level 1 definition via the NearArchive database administration ISPF dialogs). Note that migration will only be performed if the INLEVEL and OUTLEVEL execution parameters have been specified.
  - c) objects which have an active disk ('K') copy, but which have been in the database for a length of time exceeding the disk object retention period, as defined for the database storage level being processed. The disk copy of these objects will be deleted. All subsequent accesses to such objects will be satisfied from a tape copy.
- For databases with only one defined storage level, the utility will terminate after completing its scan of the database's primary index.
- If the INLEVEL and OUTLEVEL execution parameters have been specified the utility will continue by processing the objects recorded in the migration control dataset. The contents of this dataset will be sorted into the most efficient order for tape retrieval, and then processed sequentially.
- All objects from a single tape cartridge volume will be moved from their current location (in the source storage level) to a new location (in the target storage level). If duplexing is enabled for the NearArchive database, two copies of the target storage level tape dataset(s) will be created during this process.
- The utility will perform checkpoint processing after each input tape cartridge volume has been successfully processed during the migration procedure. Restart of the utility after failure or cancellation

will cause processing to be resumed from the last successful checkpoint.

- If storage level migration processing has taken place, the NearArchive database backup utility should be run after each execution of the object management utility to create backup copies of all target storage level tape datasets created or updated during object migration processing.
- Prior to terminating, the utility will automatically delete all disk ('K') copy datasets which no longer contain any active object copies.

### **7.4.2 JCL Requirements.**

The following JCL may be used to execute the NearArchive object management utility.

```
//STEP1      EXEC PGM=OTASP040,REGION=4M,
//          PARM=(' PRINT=SUMMARY|DETAIL',
//              'RESTART=NO|YES',
//              'SIMULATE',
//              'NEWTAPE=YES|NO',
//              'INLEVEL=n',
//              'OUTLEVEL=n',
//              'DATE=yyyy/mm/dd|+nnn|-nnn')
//*
//STEPLIB DD DSN=NearArchive.load.library,DISP=SHR
//OTASV100 DD DSN=Database.primary.index.name,DISP=SHR
//SYSPRINT DD SYSOUT=*
[/SORTOUT DD DSN=&&SORTOUT,DISP=NEW,UNIT=xxxxxxxx,]
[/          SPACE=(CYL,(y,z)) ]
```

The following optional EXEC parameters may be specified:

**PRINT :** Specify the PRINT parameter to identify the type of reports to be produced by the utility (see section 7.2.6 for a description of utility reports). The parameter may take one of the following values:

**SUMMARY** -produce summary expiration and migration processing reports.

**DETAIL** - produce detailed expiration and migration processing reports.

If this parameter is omitted, a default value of 'SUMMARY' is used.

**RESTART:** Specify the RESTART parameter to control utility restart processing. The following parameter values may be specified:

**'NO'** - normal utility processing is to be invoked. This is the default value.

**'YES'** - restart utility processing is to be invoked.

If the previous execution of the utility did not terminate successfully, the utility should be rerun with a RESTART parameter value of 'YES'. The utility will not execute after a failed run, unless 'RESTART=YES' is specified here.

For normal running, the default value of 'NO' should be used.

**SIMULATE:** If the 'SIMULATE' parameter is specified, the object management utility will run in simulate mode. Normal object management processing will be performed, and all standard utility reports issued, but no database update processing will take place.

This parameter may be used to predict the actions which will be performed by the utility prior to actual execution.

If this parameter is not specified, standard utility processing will be performed (i.e.) database updates will be performed as normal.

**NEWTAPE:** Specify the NEWTAPE parameter to control whether objects which are been migrated to a new storage level will be written to a new tape volume at the start of migration processing. The following parameter values may be specified:

**'NO'** - objects will be appended to the last existing tape volume in the storage level to which objects are being migrated. A new tape volume will only be allocated when the volume being updated is filled to its required capacity (as controlled by the maximum blockcount per volume setting for that storage level).

**'YES'** - a new tape volume will be allocated in the storage level to which objects are being migrated at the start of migration processing.

If this parameter is omitted, a default value of 'NO' is used.

**INLEVEL:** Specify the INLEVEL parameter when performing object migration processing, to identify the storage level from which objects are to be migrated. All objects in the storage level which are older than the sum of the migration periods specified for all storage levels up to (but not including) the target storage level will be moved from this storage level to the storage level specified (or defaulted) in the OUTLEVEL execution parameter.

If this parameter is specified, then the OUTLEVEL parameter must also be specified. The storage level specified in the INLEVEL parameter must have been defined in the database being processed, and must be lower in value than that specified in the OUTLEVEL parameter.

If omitted, the OUTLEVEL parameter must also be omitted. In this case no migration processing will be performed during execution of the utility.

**OUTLEVEL:** Specify the OUTLEVEL parameter when performing object migration processing, to identify the storage level into which objects are to be migrated. All objects in the source storage level (as specified on the INLEVEL parameter) which are older than the sum of the migration periods specified for all storage levels up to (but not including) the target storage level specified in this parameter will be moved from the source storage level to the this storage level.

If this parameter is specified, then the INLEVEL parameter must also be specified. The storage level specified in the OUTLEVEL parameter must have been defined in the database being processed, and must be higher in value than that specified in the INLEVEL parameter.

If omitted, the INTLEVEL parameter must also be omitted. In this case, no migration processing will be performed during execution of the utility.

**DATE:** The DATE parameter may only be specified if the SIMULATE parameter is also specified. It is used to simulate execution of the object management utility on a date other than the system run date. The parameter may take one of the following values:

**yyyy/mm/dd** - This specifies an explicit date for which object management processing is to be simulated. This entry may contain any valid date, in the format YYYY/MM/DD.

**+nnn or -nnn** - 'nnn' gives the number of days by which the system run date is to be increased ('+') or decreased ('-') to derive the date for which object management processing is to be simulated. 'nnn' must be a numeric value in the range 1 to 999.

If this parameter is omitted when running in simulate mode, the system run date will be used for object management simulation.

Dataset names should be set as follows for each DD entry:

**STEPLIB :** The name of the NearArchive version 2 product library, as unloaded during the product installation procedure (see chapter 2).

**OTASV100:** The name of the primary index of the database to be processed by the utility. The utility will only process one NearArchive database per execution.

**SYSPRINT:** The destination of the utility report file. If required, the report may be written to a sequential dataset, with record format = FBA, and a logical record length of 133 bytes.

**SORTOUT:** This is an optional entry which, if specified, identifies a temporary disk dataset to be used by the object management utility when sorting the list of objects which have been selected by the utility for migration between storage levels. If specified, it should be coded as illustrated in the sample JCL above, where:

'xxxxxxxx' is the unit name to be used for allocation of the dataset



'y' and 'z' are the primary and secondary allocations to be used when creating the temporary dataset. These should be large enough to hold information for all the objects which have been selected for migration in the current (or restarted) execution of the utility. There will be one record in this dataset for each selected object. Each record will be (k+32) bytes in length, where 'k' is the primary keylength as defined for the database being processed by the utility.

During processing of its migration cycle, the utility will perform an internal sort routine. To do this, it invokes the sort product available on the host system. The JCL example shown here does not use any sort work files; internal program storage will be used for sort work areas.

It may be necessary to add DD entries for sort work files if the number of entries to be sorted is sufficiently large. Refer to documentation supplied with the sort product in use on the host system for details on how to code these DD statements, and for any other considerations which should be taken into account when using the sort product.

Sort work files will not be required for execution of the object management utility with NearArchive databases for which only one storage level has been defined.

### **7.4.3 Condition Codes.**

OTASP040 will set a condition code on completion. This code may take the following values:

- 0 - the utility has been executed successfully.
- 4 - a warning condition has been encountered during processing of the utility, but execution of the utility has continue. A warning message will have been written to the SYSPRINT dataset.
- 8 - an error condition has been encountered during processing of the utility, but execution of the utility has continue. An error message will have been written to the SYSPRINT
- 12 - serious error has occurred during utility processing. Execution of the utility has been terminated. An error message will have been written to the SYSPRINT dataset.

#### **7.4.4 Running the Utility.**

Prior to executing the object management utility, the following database definition parameters should be set as required:

- 'Data retention period' in the base database definition panel (OTSN4300) or base database amendment panel (OTSN4210).

The length of time (in years and days) for which objects are to be retained in the NearArchive database should be entered in these fields. The object management utility uses these values to identify objects which are to be expired. Entering a retention period value of 0 years and 0 days (the default setting) will suppress automatic expiry of objects from the database.

- 'Migrate to next storage level' in the storage level amendment panel (OTSN4231) for all storage levels up to that specified in the OUTLEVEL execution parameter.

Enter the length of time (in years and days) for which objects are to be resident in each storage level of the database. The object management utility will use these values to identify objects which are to be migrated from the source storage level to the target storage level. Objects which are older than the sum of the migration periods for all storage levels up to (but not including) the target storage level will be migrated from source to target storage level.

- 'Disk copy retention period' in the storage level definition panel (OTSN4301) or storage level amendment panel (OTSN4231).

The length of time (in days) for which a disk copy of objects is to be retained in any particular database storage level should be entered in this field. The object management utility uses these values to identify objects for which disk copies are to be deleted. Entering a disk copy retention period value of 0 days (the default setting) will suppress deletion of disk copies of objects. These copies will be automatically deleted when the object itself is deleted (after exceeding the database's data retention period), or when the primary tape copy of the object is recycled.

Note that objects which are eligible for both migration and expiration will be expired by the object management utility.

Refer to chapter 3 for a description of NearArchive database definition and maintenance procedures.

If objects have been selected for migration during execution of the object management utility, then the NearArchive database backup utility OTASP110 should be executed after completion of the utility. This will create one or two backup copies of each target storage level tape volume created or updated during execution of the utility. Note that objects will only be selected for migration when the INLEVEL and OUTLEVEL execution parameters have been specified.

The object management utility requires update access to the NearArchive database being processed, and consequently will not execute if the database is already open for update processing by another job.

Inactive tape space within a NearArchive database which is occupied by objects which have been expired or migrated, or disk space occupied by objects whose disk copies have been expired may be reclaimed through execution of the NearArchive database maintenance utility OTASP030. This utility should be run periodically after completion of the object management utility, in order to minimize the overall tape and disk storage required by a NearArchive database.

#### **7.4.5 Operator commands.**

Execution of the utility may be terminated at any point by entering the following command through the system console:

```
(STO)P jobname
```

where 'jobname' is the name of the job used to execute the object management utility.

The utility will acknowledge receipt of this command via message OTS04097 or OTS04597, and perform normal termination processing. All database updates performed by the utility prior to termination will be committed. Utility processing may be continued at any subsequent time via normal job execution.

#### **7.4.6 Utility processing reports.**

The following reports will be written by the object management utility to the SYSPRINT dataset. These reports will be written whether running in 'actual' mode or 'simulate' mode:

1. Execution parameter report.

This report summarizes execution parameters, as specified on the JCL EXEC statement, or as defaulted.

## 2. Expiration processing report.

Two versions of this report may be produced; a summary report or a detailed report. The type of report produced is controlled by utility EXEC parameter.

The expiration processing summary report gives the total number of objects expired by the utility for the NearArchive database being processed.

The expiration processing detail report identifies each expired object by primary key, archive date, tape sequence number and storage level, and gives the total number of objects expired by the utility.

## 3. Migration processing report.

Two versions of this report may also be produced; a summary report or a detailed report. The type of report produced is again controlled by utility EXEC parameter.

The migration processing summary report gives the total number of objects migrated from the source storage level during this run of the utility, for the NearArchive database being processed.

The migration processing detail report identifies each migrated object by primary key, archive date, tape sequence number and storage level, and gives the total number of objects migrated from the source storage level to the target storage level. For DBCS customers, the primary key must contain the shift-out and shift-in control bytes to print correctly.

Error or warning messages generated during execution of the utility will be written to the expiration or migration processing summary report. All messages will have the message identifiers 'OTS040xx' or 'OTS045xx', where 'xx' is a numeric value in the range 00-99.

Refer to chapter 8 of this manual for a description of all messages issued by the object management utility.

If NearArchive objects have been migrated during execution of the utility, a sort report will be written to a SYSOUT dataset with DD name 'SYSOUT'. This dataset will be dynamically allocated by the utility.

### **7.4.7 Utility Failure and Restart Considerations.**

If the object management utility fails to complete successfully, either due to an error condition being encountered, or execution being cancelled by the system operator, it should be rerun, specifying an EXEC parameter of 'RESTART=YES'. Note that if execution was terminated early via entry of an operator 'STOP' command, the utility will terminate normally. For subsequent execution, the RESTART parameter should be set to 'NO', or allowed to default.

If execution of the utility failed during its expiration processing cycle, the restarted job will re-process the NearArchive database primary index, expiring all objects not successfully expired during the failed run, and rebuilding the migration control dataset (for databases where migration has been enabled).

The utility will automatically checkpoint its activity during its migration processing cycle, after all objects to be migrated from a single tape cartridge volume have been processed successfully. If execution of the utility failed during the migration processing cycle, processing will be restarted from the last successful checkpoint.

There is no requirement to restore any dataset prior to rerunning the utility.

The object management utility has built-in controls to prevent normal processing by the utility of a NearArchive database, after failure of the previous object management procedure for that database. In these circumstances, a parameter of 'RESTART=YES' **must** be present on the job's EXEC statement.

Archival of data to a NearArchive database should not be performed while completion of a cancelled object management procedure is still outstanding. It is also important that a NearArchive database's migration control dataset should not be amended in any way while an object management procedure is suspended in this manner.

Note that as the object migration procedure may take some time to execute (depending on the number of tape cartridge volumes to be processed), it may be terminated by the system operator at any time via entry of a 'STOP' command, in order to allow input or input-output access to the database from other jobs. Execution of the object management procedure may subsequently be continued, by re-submitting the job for normal execution.

## **7.5 OTASP070 - the NearArchive Journal Maintenance Utility.**

---

The journal maintenance utility is provided in order to enable analysis and archival of the entries in a NearArchive database's journal dataset, and to clear the contents of the dataset after analysis.

A NearArchive database's journal dataset is automatically created when the audit trail facility is first enabled for that database, via the NearArchive database administration ISPF dialogs. It is used to log all updates to the database. The audit trail facility may be enabled and disabled as required.

Journal entries are automatically created by NearArchive when updating a NearArchive database for which the audit trail facility has been enabled. Entries will be accumulated in the database's journal dataset. The journal maintenance utility will clear the contents of the journal dataset after successful execution.

If journal archiving has been enabled for a NearArchive database, the journal maintenance utility will archive all entries from the journal dataset to the database's journal archive database before clearing its contents. The journal archive database is a special NearArchive database which is used for archival of a database's journal entries. It is automatically defined and initialized when the journal archiving facility is first enabled for a NearArchive database via the NearArchive database maintenance ISPF dialogs.

The journal maintenance utility may be used to perform historical analyses of archived journal entries from the journal archive database.

### **7.5.1 Functions.**

The NearArchive journal maintenance utility performs the following functions:

- It sequentially processes a NearArchive database's journal dataset, and produces a summary or detailed analysis of journal entries.
- If journal archiving has been enabled for the NearArchive database, the utility will archive all entries from the journal dataset to the database's journal archive database.
- The contents of the journal dataset will be cleared on successful completion of the utility.

- The utility may be used to produce a summary or detailed analysis of archived journal entries from the database's journal archive database, for a specified date interval.
- When processing the live journal file, the utility will optionally sort all entries in the journal file to remove duplicate entries caused by cancellation or abnormal termination of a database update process.

### **7.5.2 JCL Requirements.**

The following JCL may be used to execute the NearArchive journal maintenance utility.

```
//STEP1      EXEC PGM=OTASP070,REGION=4M,
//          PARM=(' PRINT=SUMMARY|DETAIL',
//              'RESTART=NO|YES',
//              'DATE=yyyymmdd-yyyymmdd',
//              'NODUPS')
//          /*
//STEPLIB DD DSN=NearArchive.load.library,DISP=SHR
//          /*
//OTASV100 DD DSN=Database.primary.index.name,DISP=SHR
//SYSPRINT DD SYSOUT=*
//          /*
[/OTASSRT1 DD DSN=&&OTASSRT1,UNIT=SYSDA,DISP=NEW,    ]
[/          SPACE=(CYL,(a,b))                      ]
[/OTASSRT2 DD DSN=&&OTASSRT2,UNIT=SYSDA,DISP=NEW,    ]
[/          SPACE=(CYL,(a,b))                      ]
[/OTASSRT3 DD DSN=&&OTASSRT3,UNIT=SYSDA,DISP=NEW,    ]
[/          SPACE=(CYL,(a,b))                      ]
[/SYSOUT DD SYSOUT=*
```

The following optional EXEC parameters may be specified:

**PRINT :** Specify the PRINT parameter to identify the type of report to be produced by the utility (see section 7.3.5 for a description of the utility report). The parameter may take one of the following values:

**SUMMARY** - produce summary journal entry analysis report.

**DETAIL** - product detailed journal entry analysis report.

If this parameter is omitted, a default value of 'SUMMARY' is used.

**RESTART:** Specify the RESTART parameter to control utility restart processing. The following parameter values may be specified:

‘**NO**’ - normal utility processing is to be invoked. This is the default value.

‘**YES**’ - restart utility processing is to be invoked.

If the previous execution of the utility did not terminate successfully, the utility should be rerun with a RESTART parameter value of 'YES'. The utility will not execute after a failed run, unless 'RESTART=YES' is specified here.

For normal running, the default value of 'NO' should be used.

**DATE:** Specify the DATE parameter if the utility is to be used to produce an historical analysis from the database's journal archive database. If the DATE parameter is specified, the live journal dataset will not be processed. If no journal archive database exists, execution of the utility will be terminated.

Use the DATE parameter to specify the range of dates to be processed from the journal archive database. One or two date values may be specified (in the format YYYYMMDD), to indicate the start and end date (inclusive) of the range to be processed. The end date must not be less than the start date. The end date may be omitted (along with the '-' separator) if only one day's archived entries are to be processed.

The dates specified in this parameter refer to the dates on which journal entries were first created (i.e.) the date that the corresponding update to the NearArchive database was made.

If the DATE parameter is omitted, the utility will perform a live journal dataset analysis.



**NODUPS:** Specify the NODUPS parameter if the utility is to remove duplicate entries from the journal file during current journal analysis processing (i.e. no DATE execution parameter specified). Duplicate entries may be present if a database update procedure was cancelled or terminated abnormally, and was subsequently restarted to complete the update process.

Specification of this execution parameter will cause the utility to perform the following three additional processes prior to continuing with standard journal analysis processing:

- a) Sort the journal file into primary key/archive date sequence.
- b) Process the sorted file and remove entries for duplicate update actions.
- c) Sort the remaining journal entries back into ascending timestamp sequence.

This additional procedure will require three sort work files to be allocated in the JCL (see below). Note that the journal file itself is not modified by this additional processing. The journal utility may be restarted normally (as discussed in section 7.3.6) following execution failure.

If the NODUPS parameter is specified, the DATE parameter must not also be present in the execution parameters.

If the NODUPS parameter is omitted, the additional processing specified above will not be performed, and duplicate journal entries will not be removed. DD cards for the sort work files need not be specified in the JCL used to execute the utility.

Dataset names should be set as follows for each DD entry:

- STEPLIB :** The name of the NearArchive version 2.5 product library, as unloaded during the product installation procedure (see chapter 2).
- OTASV100:** The name of the primary index of the NearArchive database to be processed by the utility. The utility will only process one NearArchive database per execution.
- SYSPRINT:** The destination of the utility report file. If required, the report may be written to a sequential dataset, with record format = FBA, and a logical record length of 133 bytes.
- OTASSRTx:** These DD cards (where 'x' = 1, 2 and 3) are only required if the 'NODUPS' execution parameter has been specified. All three cards should be specified if required. They should each specify a separate permanent or temporary sequential disk dataset which is to be used as a sort work file by the utility. If a permanent dataset is to be used, it should be allocated with a variable spanned record format (RECFM=VBS) and a logical record length of (k+39) bytes, where 'k' is the primary keylength of the NearArchive database being processed by the utility.

The primary and secondary space allocations for each dataset ('a' and 'b') should be large enough to hold all records in the live journal file. There will be one journal record, of length (k+39) bytes, for each update action performed against the selected NearArchive database since the previous execution of the utility, or since it was first created if the utility is being executed for the first time.

- SYSOUT:** This entry is only required if the 'NODUPS' execution parameter has been specified. It gives the destination of the report files produced from internal sort requests. If required, the reports may be written to a sequential dataset, with record format = FBA, and a logical record length of 133 bytes.

### **7.5.3 Condition Codes.**

OTASP070 will set a condition code on completion. This code may take the following values:

- 0 - the utility has been executed successfully.
- 4 - a warning condition has been encountered during processing of the utility. A warning message will have been written to the SYSPRINT dataset. The utility has been executed successfully.
- 8 - an error condition has been encountered during processing of the utility. An error message will have been written to the SYSPRINT dataset. Execution of the utility has been terminated.
- 12 - a serious error has occurred establishing the runtime environment for the utility. An error message will have been written to the SYSPRINT dataset. Execution of the utility has been terminated.

### **7.5.4 Running the Utility.**

Prior to executing the journal maintenance utility, the following database definitions should be set as required:

- 'Audit trail on' in the audit options section of the base database definition panel (OTSN4300) and amendment panel (OTSN4210). The audit trail facility must already have been enabled for the NearArchive database being processed by the utility, in order that a journal dataset should exist. However, the audit trail facility does not have to be currently enabled when processing a NearArchive database with this utility.
- 'Journal archiving enabled' in the audit options section of the base database definition panel (OTSN4300) and amendment panel (OTSN4210). This facility should be enabled if it is required that the utility should archive all journal entries prior to clearing the contents of the NearArchive database's journal dataset.

If enabled, entries will be archived to the database's journal archive database. This is a special NearArchive database which is automatically defined and initialized when journal archiving is first enabled for a NearArchive database.

The journal maintenance utility may be used to perform either of the following two functions:

1. Perform analysis and maintenance of a NearArchive database's live journal dataset. The utility will produce a detailed and/or summary analysis of journal entries (depending on the PRINT parameter setting), and clear the contents of the journal dataset on successful completion. The journal dataset will then be reset and available for use during NearArchive database update operations.
2. Perform an historical analysis on a NearArchive database's archived journal entries, for a specified date range. This option is enabled through the use of the DATE keyword parameter in the EXEC card (see section 7.3.2 - JCL requirements).

The two processing options are mutually exclusive, and are controlled through specification of the DATE keyword in the job EXEC parameters.

### **7.5.5 Utility Processing Reports.**

The following reports will be written by the journal maintenance utility to the SYSPRINT dataset:

1. Execution parameter report.

This report summarizes execution parameters, as specified on the JCL EXEC statement, or as defaulted.

2. Journal entry analysis report.

A summary or detailed journal entry analysis report may be produced by the utility.

A summary report is produced if the PRINT=SUMMARY parameter is specified in the job EXEC parameters, or is allowed to default. This gives daily and overall totals for the number of journal entries in the following categories:

- archived objects
- objects deleted via batch request
- objects deleted via CICS request
- object deleted via dynamic backout during archival processing
- expired objects
- objects moved by the database maintenance utility

- objects migrated by the object management utility
- enabling/disabling of the audit trail facility
- enabling/disabling of the journal archive facility

A detailed report is produced if the PRINT=DETAIL parameter is specified in the job EXEC parameters. In addition to the summary details identified above, the detail report will also list the following information for each journal entry:

- date and time of entry - gives the date and time that the entry was created.
- type of operation - gives the type of operation which caused the entry to be created. The categories are as identified in the summary report description above.
- storage level - the database storage level in which the processed object was located.
- tape dataset sequence number - the sequence number of the tape dataset in the above storage level, in which the processed object resided.
- jobname - the name of the job (for batch operations), or VTAM application identifier (for CICS operations), which issued the logged request.
- user identification - the identification of the user issuing the logged request.
- archive date - the archive date of the processed object.
- primary key - the primary key of the processed object. Only the first 51 characters of the primary key will be printed for databases whose primary keylength exceeds this value.

Error or warning messages generated during execution of the utility will be written to the journal entry analysis report. All messages will have the message identifier 'OTS070xx', where 'xx' is a numeric value in the range 00-99.

Refer to chapter 8 of this manual for a description of all messages issued by the journal maintenance utility.

### **7.5.6 Utility Failure and Restart Considerations.**

If the journal maintenance utility fails to complete successfully, either due to an error condition being encountered, or execution being cancelled by the system operator, it should be rerun, specifying an EXEC parameter of 'RESTART=YES'.

The utility will detect conditions where the previous execution did not complete successfully. If 'RESTART=YES' has not been specified, the utility will terminate processing in these conditions.

There is no requirement to restore any dataset prior to rerunning the utility. Note that when processing a journal dataset (no DATE parameter specified), the contents of that dataset are only reset after successful completion of the utility.

### **7.5.7 Journal Dataset Specification.**

The journal dataset may be processed directly by customer applications if it is required to extend the analysis performed by the NearArchive journal maintenance utility. File and record specifications for this dataset are supplied in this section.

#### **A. Journal Dataset.**

A journal dataset is automatically created for each NearArchive database when the audit trail facility is first enabled for that database. The journal dataset is a VSAM entry-sequenced dataset, containing fixed-length records. Each record is (k+39) bytes long, where 'k' is the primary keylength of the NearArchive database, as defined during database definition processing.

A NearArchive database's journal dataset name will be:

`'hlq'.JOURNAL`

where 'hlq' is the high-level qualifier specified for the NearArchive database during database definition processing.

Each journal record has the following format:

Offset	Length	Type	Description
0	2	char	Type of operation:  '11' - inserted via batch archival '21' - deleted via batch delete '22' - deleted via CICS delete '23' - deleted via dynamic backout '24' - deleted via expiration utility '31' - moved by database maintenance utility '32' - moved by object management utility '41' - audit trail enabled '42' - journal archiving enabled '51' - audit trail disabled '52' - journal archiving disabled
2	k	char	Primary key of NearArchive object (length 'k' bytes)
k+2	4	pack	Archive date of NearArchive object in packed decimal format CYYMMDDF, where 'F' indicates an unsigned packed decimal field, and 'C' is the century indicator. This can take the following values:  '0' – date is in range 1950-2049 '1' – date is in range 2050-2149
k+6	2	bin	Number of tape blocks required to hold archived object. This is an unsigned half-word binary field. It will contain the value X'FFFF' if the number of blocks is 65535 or greater.
k+8	4	bin	Identifier of first tape block containing archived object
k+12	2	bin	Dataset sequence number
k+14	1	bin	Storage level
k+15	8	char	Jobname (from which request was

Offset	Length	Type	Description
			issued)
k+23	8	char	Userid of requester
k+31	8	bin	Timestamp:  bytes 0-3: time (in format X'hhmmssst') that request was processed  bytes 4-7: Julian date that request was processed, in packed decimal format 0cyydddF, where 'c' = 0 for 20 <sup>th</sup> century dates, and 1 for 21 <sup>st</sup> century dates.

## **B. Journal Archive Database.**

If journal archiving is enabled for a NearArchive database, the journal maintenance utility will archive each entry from a journal dataset to the journal archive database prior to resetting the dataset.

The journal archive database is a standard NearArchive database. It will be automatically defined and initialized when the journal archive facility is first enabled for a NearArchive database. The name of the journal archive database's primary index dataset is:

`'hlq'.J.INDEX`

where 'hlq' is the high-level qualifier specified for the primary NearArchive database during database definition processing.

Journal entries are archived to the journal archive database as separate logical records within an archived object. One NearArchive object will be created for each separate date value in the journal dataset. Each object will have the following identifier:

- Primary key - this is a 4-byte packed decimal field in format 'CYYMMDDF', where 'F' indicates an unsigned packed decimal field, and 'C' is the century indicator, containing the creation date of all journal entries in the NearArchive object. The century indicator can take the following values:



‘0’ – date is in range 1950-2049

‘1’ – date is in range 2050-2149

- Archive date - a dummy date, used to allow archival of multiple objects with identical primary key (journal entry date).

Each logical record within the NearArchive object will be of length  $(k+39)$  bytes, where ‘k’ is the NearArchive database's primary keylength, and will be in the journal dataset record format as described above.

Archived journal entries may be processed by a customer application using standard NearArchive batch or online interface requests.

## 7.6 OTASP080 - the NearArchive Secondary Index Update Utility.

---

The secondary index update utility is used to execute the second phase of the two-phase asynchronous secondary index update process. These phases consist of the following processing:

- Phase 1 – this process is executed during standard batch archival update of a database. Secondary indexes which are to be updated asynchronously are identified via the entry of the DD card ‘OTnnxxxx’ in the JCL used to execute the archival process, where ‘nn’ is the 2-digit secondary index identifier and ‘xxxx’ is the NearArchive database identifier (as specified in the archival application’s ‘open database’ call).

Secondary indexes identified for asynchronous update in this way are not updated directly during archival processing. NearArchive will instead save all secondary index updates in the sequential disk file identified in the ‘OTnnxxxx’ DD card.

Note that if an ‘OTnnxxxx’ DD card is not supplied for secondary index ‘nn’, then all updates will be performed synchronously during batch archival processing. This is the default process.

- Phase 2 – this process is performed by executing the secondary index update utility once for each secondary index which has been scheduled for asynchronous update processing, as described above.

The utility will process all updates for a single secondary index, as saved in file ‘OTnnxxxx’ during archival processing, sort the updates into the most efficient processing sequence, and perform all necessary secondary update processing. The asynchronous update process is complete for an individual secondary index on successful completion of phase 2 processing.

Implementation of asynchronous secondary index update processing may reduce the overall amount of I/O activity required to index objects during archival to a NearArchive database, depending on the structure of the objects, the order in which they are presented to NearArchive for archival, and the way in which secondary index entries are generated during archival processing. This may provide the following benefits:

- During archival processing, random I/O to the NearArchive secondary index (VSAM KSDS) for synchronous secondary index update processing will be replaced by sequential I/O to the asynchronous secondary index dataset (non-VSAM) for all secondary indexes which have been scheduled for asynchronous update. This is likely to reduce the overall I/O activity during batch archival processing, and improve job turnaround times.
- The secondary index update utility will sort all index updates in order to improve efficiency of updating the NearArchive secondary index by processing the file in sequential order. This may reduce the overall I/O activity required to perform index update processing compared with standard synchronous updates.

The secondary index update utility must be executed once for each secondary index which has been scheduled for asynchronous update during archival processing (i.e. once for each 'OTnnxxxx' DD card in the JCL used to execute this process). Further archival of objects to a database will not be permitted until all outstanding asynchronous secondary index updates have been performed successfully, as described above.

Refer to section 4.4 of this manual for additional information on asynchronous secondary index update processing.

### **7.6.1 Functions.**

The NearArchive secondary index update utility performs the following functions:

- It validates control information in the input sequential dataset, and verifies that there is an outstanding asynchronous update for the secondary index identified therein.
- After performing initial validation processing, it calls the system sort program to sort all entries in the input sequential dataset into ascending secondary key/primary key/archive date sequence. This is the sequence that entries are held in the NearArchive secondary index dataset.
- It then processes the sorted file sequentially, and performs NearArchive secondary index update processing.
- On successful completion of the above processing, the utility will remove the secondary index definition from outstanding asynchronous update status, and reset the input sequential dataset to empty.

- The utility will issue a summary report at successful or unsuccessful end of processing, giving totals of the number of records processed from the input sequential file, the number of secondary index update operations performed, and the number of exceptional conditions which may have occurred during processing.

### **7.6.2 JCL Requirements.**

The following JCL may be used to execute the NearArchive secondary index update utility.

```
//STEP1      EXEC PGM=OTASP080,REGION=4M
//*
//STEPLIB   DD   DSN=NearArchive.load.library,DISP=SHR
//*
//OTASV100  DD   DSN=Database.primary.index.name,DISP=SHR
//SYSPRINT  DD   SYSOUT=*
//SYSOUT    DD   SYSOUT=*
//*
//OTASS080  DD   DSN=Async.update.dataset,DISP=SHR
//OTASS081  DD   DSN=&&OTASS081,UNIT=SYSDA,DISP=NEW,
//              SPACE=(CYL,(a,b))
//*
[//SORTWKnn DD   UNIT=SYSDA,SPACE=(CYL,(c,d))]
```

Dataset names should be set as follows for each DD entry:

- STEPLIB :** The name of the NearArchive version 2 product library, as unloaded during the product installation procedure (see chapter 2).
- OTASV100:** The name of the primary index of the NearArchive database to be processed by the utility. The utility will only process one NearArchive database per execution.
- SYSPRINT:** The destination of the utility report file. If required, the report may be written to a sequential dataset, with record format = FBA, and a logical record length of 133 bytes.
- SYSOUT:** The destination of the sort report file.
- OTASS080:** The name of the sequential dataset allocated to file ‘OTnnxxxx’ during batch archival processing, where ‘nn’ is the identifier of the secondary index to be processed during execution of the utility.

**OTASS081:** This entry should identify a disk dataset which is to be used to hold output from the internal sort of file OTASS080. A temporary dataset may be specified, as illustrated in the above JCL. Alternatively, a permanent dataset may be pre-allocated; in this case, the dataset must be in variable blocked format with a maximum logical record length of 276 bytes (RECFM=VB, LRECL=276) - any appropriate blocksize may be chosen.

The OTASS081 dataset must be at least as large as the input OTASS080 dataset. This dataset will contain one record for each secondary index update entry. Each record will be  $(k+s+10)$  bytes in length, where 'k' is the primary keylength of the database being processed, and 's' is the keylength for the secondary index being processed. The space allocation values ('a' and 'b' as illustrated in the above JCL) should be chosen to ensure that the dataset is large enough to contain all secondary index update entries created during execution of the corresponding batch archival process.

**SORTWKnn:** One or more SORTWKnn DD cards may be present in the JCL to specify external sort work files which are to be used during internal sort processing. Refer to documentation on the sort product in use on the host installation for detailed information on how to specify these entries.

If no SORTWKnn DD cards are present in the JCL, the sort program will perform an in-core sort. Sufficient virtual storage should be assigned to the region in order to allow the internal sort to execute successfully.

### **7.6.3 Condition Codes.**

OTASP080 will set a condition code on completion. This code may take the following values:

- 0 - the utility has been executed successfully.
- 4 - a warning condition has been encountered during processing of the utility. A warning message will have been written to the SYSPRINT dataset. Execution of the utility has terminated normally.
- 8 - an error condition has been encountered during processing of the utility. An error message will have been written to the SYSPRINT dataset. Execution of the utility has terminated normally.
- 12 - a serious error condition has been encountered during processing of the utility. An error message will have been written to the SYSPRINT dataset. Execution of the utility has terminated abnormally.

### **7.6.4 Running the Utility.**

The utility should be executed once for each outstanding asynchronous secondary index update, following successful completion of an object archival procedure (i.e.) once for each 'OTnnxxxx' DD card in the JCL used during object archival processing. Each separate execution of the utility should specify the name of the corresponding secondary index update dataset (as specified in the 'OTnnxxxx' DD card for batch archival) in the 'OTASS080' DD entry.

Following successful execution of the utility, the contents of the OTASS080 file will be deleted, and the file reset for use by the next object archival process for that database.

Further archival of objects to a database will not be permitted until all outstanding asynchronous secondary index updates have been completed via execution of the secondary index update utility.

It should be noted that use of the NearArchive dynamic backout facility is not permitted if one or more secondary indexes are to be updated asynchronously during object archival processing. See section 4.5.3 for further discussion of this issue.

### **7.6.5 Utility Processing Reports.**

The following reports will be written by the secondary index update utility to the SYSPRINT dataset:

A summary totals report will be issued at successful or unsuccessful end of processing, giving the following information:

- a) Total number of records read from file OTASS080.
- b) Number of secondary index entries updated.
- c) Number of duplicate entries read from file OTASS080. A duplicate entry may exist when object archival processing has been restarted following an earlier failure. Duplicate entries are ignored by the utility.
- d) Number of duplicate secondary index updates. A duplicate secondary index update may occur if the utility has been rerun following an earlier failure. No action is taken by the utility when a secondary index entry already exists.

The sum of totals (b), (c) and (d) should be equal to the total number of records read from file OTASS080 (a).

Error or warning messages generated during execution of the utility will be written to the SYSPRINT dataset. All messages will have the message identifier 'OTS080xx', where 'xx' is a numeric value in the range 00-99.

Refer to chapter 8 of this manual for a description of all messages issued by the secondary index update utility.

### **7.6.6 Utility Failure and Restart Considerations.**

There are no special restart considerations for rerunning the secondary index update utility after a failure has occurred. The job may be rerun as normal after the failing error condition has been rectified.

Any secondary index updates performed during the failed run will not be re-processed by the utility. Each occurrence of this condition will cause the 'number of duplicate secondary index updates' total to be incremented by 1. This total is displayed in the summary totals report produced at end of run.

Note that the secondary index update utility must be executed successfully for each secondary index being updated asynchronously before further object archival processing can take place for a database. A failed run of the utility will not remove this restriction.

## 7.7 OTASP110 - the NearArchive Database Backup Utility.

---

The NearArchive database backup utility (OTASP110) is used to create backup copies of tape datasets from within a NearArchive database. The utility can operate in one of two modes:

- Full-volume mode. When executing in full-volume mode, the utility will identify all primary tape datasets which have been updated since the preceding execution of the utility against the selected database. Each of these datasets will be backed up by copying the entire dataset to a full-volume backup ('B') copy dataset and/or a duplex backup ('D') copy dataset. The utility may also be used in full-volume mode to re-create the duplex ('C') copy of a primary copy dataset as created during database update processing, or to create/re-create a disk ('K') copy dataset.

Any existing full-volume backup tape dataset will be automatically uncataloged by the utility prior to creation of the new full-volume backup. Any existing disk copy dataset will be deleted prior to re-creation.

All tape backup volumes used by the utility should be of the same device type and capacity as that of the volume containing the corresponding primary copy dataset. The utility will terminate with an error message if incompatible devices are found. This error processing can be suppressed via EXEC parameter.

- Incremental backup mode. When executing in incremental backup mode, the utility will identify all primary tape datasets which have been updated since the preceding execution of the utility, and will copy all objects which have been added to the database since that execution to a primary ('I') copy incremental backup dataset, and optionally a duplex ('J') copy incremental backup dataset. There will be one set of incremental backup datasets for each corresponding primary copy dataset which has been processed by the utility.

Device types used for incremental backup datasets do not have to match those used for creating the primary copy dataset. Each incremental backup dataset may be a multi-volume file.

All incremental backup datasets (primary and duplex) for a primary copy dataset will be automatically uncataloged when a full-volume backup of the primary copy is created by the utility.



Mode of execution is controlled by the 'TYPE=' execution parameter.

The utility should be executed after a NearArchive database has been updated via batch archival application processing, or after execution of the NearArchive database maintenance or object management utilities. The tape datasets to be backed up are automatically identified by NearArchive.

Refer to section 5.2.3 of this manual for a general description of database backup processing.

### **7.7.1 Functions.**

The NearArchive database backup utility performs the following functions:

- Automatically identifies all tape datasets which need backing up for the database being processed. These will be tape datasets which have been updated or newly created since the last backup processing for the selected database.

When executing in full-volume mode, processing will continue as follows:

- The utility will construct the name(s) of the full-volume backup dataset(s) to be created for each tape dataset to be backed up.

This is done by adding the qualifier 'B0' (for primary backup copies), 'C0' (for re-creating duplex copies), 'D0' (for duplex backup copies) or 'K0' (when re-creating disk copy datasets) after the high-level qualifier of the primary dataset name.

- Uncatalogs any existing full-volume backup copies of a tape dataset. This condition will occur when the last tape cartridge volume in a NearArchive database was partially filled at the end of the previous archival or utility procedure.

The next execution of the procedure will continue writing to the partially-used volume until the maximum block count (as specified by the user during database initialization or amendment) has been exceeded, or until the volume is full.

This tape cartridge volume will be backed up in both executions of the archival or utility procedure. The first of the backups will be superseded by the second, and will be uncataloged and may be returned to scratch status.

- When performing disk copy backup processing, the utility will automatically delete all existing disk copy datasets associated with each primary tape dataset being processed.
- The utility verifies that the capacity of the full-volume backup volume(s) is not less than that of the primary volume. If it is, an error message is issued to the print file and processing of the utility terminates.
- Creates and catalogs the required full-volume backup dataset(s), as identified in the 'COPY=' execution parameter. If 'COPY=K' has been specified (create or re-create a disk copy dataset), the utility will only copy objects to the disk copy datasets if they are currently eligible for disk residency, as defined via the disk retention period parameters for the storage level being processed.
- If 'COPY=K' has been specified, multiple disk copy datasets may be created for each primary tape dataset processed by the utility. The disk processing parameters defined for the storage level containing the dataset being processed will be used to control the size of each individual disk copy dataset.
- After successfully creating a full-volume tape backup of the primary tape dataset, existing incremental backup datasets for that primary dataset will be automatically uncataloged.
- A processing report will be written to the SYSPRINT dataset, giving details of all full-volume backup tape or disk datasets which have been uncataloged or created, and all incremental backup datasets which may have been uncataloged during execution of the utility.

When executing in incremental backup mode, processing continues as follows:

- The utility will construct the name(s) of the incremental backup dataset(s) to be created for each tape dataset to be backed up.

This is done by adding the qualifier 'xymmmm' after the full primary dataset name, where 'x' is the incremental backup copy identifier ('I' or 'J'), 'y' is the storage level of the primary dataset, and 'mmmm' is a sequential numeric identifier (in the range 0001-9999, which is the incremental backup sequence number. This number is set to '0001' when the first incremental backup dataset is created for a primary dataset, and increased by 1 for each subsequent incremental backup dataset for that same primary dataset. One incremental backup dataset will be created for each execution of the utility in incremental backup mode.

- All objects which have been added to the primary dataset since the preceding execution of the utility will be copied to the primary ('I') copy incremental backup dataset, and optionally the duplex ('J') copy incremental backup dataset. Control of creation of the duplex copy is via the 'duplex during backup' field on the storage level creation/update panel during NearArchive database administration processing in TSO/ISPF.
- A processing report will be written to the SYSPRINT dataset, giving details of all incremental backup datasets which have been created during execution of the utility.

### **7.7.2 JCL Requirements.**

The following JCL may be used to execute the NearArchive database backup utility:

```
//STEP1      EXEC PGM=OTASP110,REGION=4M,
//          PARM=( 'TYPE=FULL|INCREMENTAL',
//                'COPY=B|C|D|K'
//                'LEVEL=n|ALL'
//                'UNITPRI=xxxxxxxx',
//                'UNITDUP=xxxxxxxx',
//                'SPEED|RECOVERY',
//                'NODEVCHECK')
//          *
//STEPLIB    DD    DSN=NearArchive.load.library,DISP=SHR
//          *
//OTASV100   DD    DSN=Database.primary.index.name,DISP=SHR
//SYSPRINT   DD    SYSOUT=*
```

The following optional EXEC parameters may be specified:

**TYPE:** This optional parameter is used to control execution of the utility. The following operands may be specified:

**FULL** The utility is to execute in full-volume backup mode. Full-volume backups will be taken of all primary tape datasets selected for backup processing. This is the default value.

**INCREMENTAL** The utility is to execute in incremental backup mode. A primary and (optionally) duplex incremental backup dataset will be created for each primary tape dataset selected for backup processing.

If no 'TYPE' parameter is specified, a default value of 'TYPE=FULL' is used.

**COPY:** This optional parameter is used to control which full-volume backup copy is to be created during execution of the utility in full-volume backup mode. This parameter is not permitted when 'TYPE=INCREMENTAL' has been specified. The following operands may be specified:

- B** The utility is to create or re-create a primary backup ('B' copy) dataset.
- C** The utility is to create or re-create a duplex ('C' copy) dataset.
- D** The utility is to create or re-create a duplex backup ('D' copy) dataset.
- K** The utility is to create or re-create disk ('K' copy) datasets. When this option is specified, identification of the tape datasets to be backed up must be supplied via the OTASBKUP DD entry. Also, the LEVEL parameter should not be used with this option. A warning message will be issued by the utility in these circumstances, and the EXEC parameter will be ignored.

If this parameter is omitted, the utility will create a primary full-volume ('B' copy) backup dataset, and optionally a duplex full-volume ('D' copy) backup dataset. Control over creation of the duplex backup dataset is via the 'duplex during backup' setting during database creation or update processing in NearArchive database administration processing under TSO/ISPF (see chapter 3). If this field is set to 'Y', the utility will automatically create a duplex backup dataset during full-volume backup processing, when no COPY parameter is supplied. If this field is set to 'N', no duplex backup dataset will be created under these conditions.

Note that the presence of the COPY execution parameter will over-ride the current 'duplex during backup' setting for the database storage level being processed by the utility.

**LEVEL :** Specify the LEVEL parameter to identify the database storage level to be processed by the utility. The parameter value 'n' may take one of the following values:

'n' - process storage level n only

'ALL' - process all storage levels in the database.

If this parameter is omitted, a default value of 'ALL' is used.

Where a database has only one storage level, specifying a value of 'ALL' for this parameter (or allowing it to default), will cause the utility to process storage level 0 only.

This parameter should not be coded when creating disk copy datasets ('COPY=K'). For this type of processing, the primary datasets to be backed up are identified via the OTASBKUP DD entry. A warning message will be issued by the utility in these circumstances, but execution will continue normally.

**UNITPRI:** This parameter is used to identify the unit name to be used when allocating a new primary ('I') copy incremental backup dataset. The operand should be a valid 1-8 character device type or esoteric name, as defined to OS/390.

This parameter is only permitted when 'TYPE=INCREMENTAL' has also been specified.

If omitted, the 'B' copy unit name as defined during storage level definition and update processing in TSO/ISPF will be used for allocation of the primary copy incremental backup dataset. See chapter 3 for a description of this setting.

**UNITDUP:** This parameter is used to identify the unit name to be used when allocating a new duplex ('J') copy incremental backup dataset. The operand should be a valid 1-8 character device type or esoteric name, as defined to

OS/390. This parameter is only permitted when 'TYPE=INCREMENTAL' has also been specified.

If omitted, the 'D' copy unit name as defined during storage level definition and update processing in TSO/ISPF will be used for allocation of the primary copy incremental backup dataset. See chapter 3 for a description of this setting.

**SPEED/RECOVERY:** The 'SPEED' and 'RECOVERY' parameters no longer have any impact on processing of the utility. They are supported for downward compatibility purposes, but will be ignored by the utility.

**NODEVCHECK:** This optional parameter is used to control processing when a volume capacity mismatch has been detected between the primary tape volume and a backup tape volume, during full-volume backup processing.

If this parameter is present, the utility will issue a warning message and generate a condition code of 4, but will continue with backup processing.

If this parameter is omitted, standard capacity checking will be performed during full-volume backup processing. Any mismatch will cause the utility to terminate with condition code 12.

Dataset names should be set as follows for each DD entry:

**STEPLIB:** The name of the NearArchive version 2.5 product library, as unloaded during the product installation procedure (see chapter 2).

**OTASV100:** The name of the primary index of the NearArchive database to be processed by the utility. The utility will only process one NearArchive database per execution.

**SYSPRINT:** The destination of the utility report file. If required, the report may be written to a sequential dataset, with record format = FBA, and a logical record length of 133 bytes.

**OTASBKUP:** This entry is mandatory when creating disk copy datasets ('COPY=K' specified in the EXEC

parameters), and is used to identify one or more primary tape datasets for which disk copy datasets are to be created. The file should contain one entry (as specified below) for each primary tape copy dataset to be processed by the utility. Each entry should identify the name of the primary tape dataset for which the disk copy dataset is to be created. Objects which are eligible for disk residency will be copied from this tape dataset and written to one or more disk copy datasets. The size of each dataset (and hence the number of disk datasets created per primary tape dataset) will be controlled by the disk copy parameters defined for the storage level containing the primary tape dataset being processed by the utility.

The entry is optional when creating full-volume or incremental tape backup datasets, and is used to override automatic identification of tape datasets to be backed up by the utility. If this DD entry is not present for tape backup dataset processing, the backup utility will automatically back up all tape datasets in the selected database which have been updated or newly created since the last execution of the backup utility for that database.

If this DD entry is in use, the dataset specified in the entry should be a sequential dataset with record format = FB and a logical record length of 80 bytes. One record must be present for each tape dataset to be backed up by the utility.

Each record must have the following format:

Offset	Length	Description
0	44	Name of primary tape dataset to be backed up. This name should be in the correct format for the primary copy of a tape dataset (i.e.)  'hlq'.Gnnnnn for datasets in storage level 0 of a database

Offset	Length	Description
		'hlq'.Ax.Gnnnn' for datasets in other storage levels
		where 'hlq' is the database high-level qualifier
		'x' is the stg.lev.
		'nnnnn' is the volume seq. no.
44	36	Unused

### **7.7.3 Condition Codes.**

OTASP110 will set a condition code on completion. This code may take the following values:

- 0 - the utility has been executed successfully.
- 4 - a warning condition has been encountered during execution of the utility, but processing has continued. A warning message will have been written to the SYSPRINT dataset.
- 8 - an error condition has been encountered during execution of the utility, but processing has continued. An error message will have been written to the SYSPRINT dataset.
- 12 - a serious error has occurred during execution of the utility. An error message will have been written to the SYSPRINT dataset. Execution of the utility has been terminated.

### **7.7.4 Running the Utility.**

Prior to executing the database backup utility, the following database definition parameter should be set as required:

- 'Duplex during backup' in the storage level definition panel (OTSN4301) or amendment panel (OTSN4231).

Backup duplexing should be enabled via one of the above panels if any of the following processing options is required:

1. When executing in full-volume backup mode, if no COPY execution parameter has been supplied a duplex backup ('D') copy



dataset will be created in addition to the standard primary backup ('B') copy dataset.

2. When executing in incremental backup mode, a duplex ('J') copy incremental backup dataset will be created in addition to the standard primary ('I') copy incremental backup dataset.

In both processing modes, if duplexing is enabled each copy may be directed to separate physical locations in order to provide the required level of recoverability and data availability in the event of loss of individual tape volumes from within the database, or loss of the entire database. In full-volume backup mode, this is achieved via the 'B' and 'D' copy unit name fields when creating or updating a database during database administration processing in TSO/ISPF. In incremental backup mode, this is achieved as described for full-volume backups, or optionally via the UNITPRI and UNITDUP execution parameters.

Note that each storage level in a NearArchive database has separate controls for duplexing of data during archival and backup.

- The following 'disk copy options' in the storage level definition panel (OTSN4301) or amendment panel (OTSN4231) should be set as required if creating or re-creating disk copy datasets ('COPY=K' specified in the EXEC parameters).
  - The retention period parameter will control the disk dataset creation process. Only objects which are eligible for disk residency (based on object age and the value of the disk copy retention parameter) will be copied to disk.
  - The primary and secondary dataset allocation values and the 'maximum blockcount' field should be set as required. These parameters will govern the size of each disk copy dataset, and hence the number of disk copy datasets created per primary tape copy dataset. A maximum of 256 disk copy dataset may exist per primary tape dataset.

The utility will close a disk dataset and create a new disk copy dataset if the disk dataset becomes full or if the maximum blockcount per dataset has been exceeded.

If creation of a new disk dataset fails for any reason, or the maximum number of 256 disk copy datasets per primary tape dataset has been reached, then execution will continue with the next dataset name entry in the OTASBKUP file.

Refer to chapter 3 for a description of NearArchive database definition and maintenance procedures.

In full-volume backup mode, the backup utility will always copy the complete contents of a tape dataset, from the start of the file. This is independent of the actual amount of data added to the file during execution of the batch archival or utility procedure. Any full-volume backup dataset already in existence (after previous execution of the backup utility) will be automatically uncataloged and may be returned to the appropriate scratch pool. In addition, any incremental backup datasets which were already in existence for the primary copy being backed up will be automatically uncataloged. The utility will report on all tape cartridge volumes released and/or uncataloged in this manner.

In incremental backup mode, the backup utility will copy all objects which have been added to the primary tape dataset since the preceding execution of the utility. A new incremental backup dataset is always created (by adding 1 to the sequence number of the latest preceding incremental backup dataset) during execution of the utility.

#### **7.7.5 Utility Processing Reports.**

The following reports will be written by the database backup utility to the SYSPRINT dataset.

- Backup processing report.

This report will print details of each full-volume or incremental backup dataset created during the backup procedure, and of each existing primary or incremental backup tape dataset which has been released, uncataloged or deleted during utility execution.

Error or warning messages generated during execution of the utility will be written to the backup processing report. All messages will have the message identifier 'OTS110xx', where 'xx' is a numeric value in the range 00-99.

Refer to chapter 8 of this manual for a description of all messages issued by the database backup utility.

#### **7.7.6 Utility Failure and Restart Considerations.**

If the database backup utility fails to complete successfully (return code greater than 4), refer to the processing report in the SYSPRINT dataset to

identify the cause of the problem. After correcting the error, the utility may be rerun.

There are no special restart requirements. Primary tape datasets successfully backed up during the failed run will not require processing during rerun of the utility. The full-volume or incremental backup dataset(s) being created when the failure occurred will be automatically uncataloged or deleted by the utility and recreated during the rerun.

If the failure was due to a media problem with the primary tape dataset, that dataset should be re-created by one of the following methods:

1. If a duplex ('C' copy) tape dataset was created during the batch archival or utility procedure, this dataset may be used to restore the primary ('A' copy) dataset.

This may be done using the NearArchive database recovery utility OTASP130. Refer to section 7.6 for details on the use of this utility.

After restoring the primary tape dataset, the database backup procedure may be rerun.

2. If no duplex copy of the primary tape dataset is available, it will be necessary to rerun the archival or utility procedure in order to recreate the primary dataset on a different tape cartridge volume.

In this case, all NearArchive system datasets must be restored to their status prior to running the archival or utility job, using standard disk dataset recovery software.

In addition, the latest tape dataset in the appropriate storage level of the NearArchive database must be restored from a previous backup copy. This should be done using the NearArchive database recovery utility OTASP130. Refer to section 7.6 for details on the use of this utility.

After successfully restoring the NearArchive database, the batch archival or utility job may be re-executed in order to recreate the primary tape dataset. Database backup procedures should be performed as normal after completion of the rerun archival or utility job.

## 7.8 OTASP130 - the NearArchive Database Recovery Utility.

---

The NearArchive database recovery utility is used to recover one or more primary tape datasets from within a single NearArchive database. Recovery is performed using full-volume and/or incremental backup tape datasets created by the NearArchive database backup utility OTASP110 (see section 7.5).

The tape datasets to be recovered are specified by their sequence numbers within the appropriate storage level of the NearArchive database. Specification of these datasets is made via the volume recovery facility in the supplied NearArchive database administration ISPF dialogs. Refer to chapter 3 of this manual for a description of database definition and maintenance procedures using these dialogs.

This volume recovery procedure is also used by default to identify which full-volume backup copy ('B' - primary backup, 'C' - duplex, or 'D' - duplex backup) is to be used for recovery purposes for each volume, or range of volumes. Default selection of full-volume backup copy may be over-ridden by execution parameter.

The database recovery utility will process all recovery control entries in one or both storage levels of a NearArchive database, and re-create the primary tape dataset(s) associated with each entry. The existing primary dataset(s) will be uncataloged prior to re-creation. Each volume will be removed from recovery status after being successfully restored. This means that any subsequent access to that dataset will be satisfied from the primary dataset copy. Note that volumes in recovery status which have been marked with an 'X' copy (i.e.. 'unavailable' volumes) will not be processed by this utility.

This utility should be used to recover individual tape cartridge volumes which have become unavailable due to media failure, or for some other reason, and also to recover all tape datasets in a NearArchive database, after loss of a complete database.

### **7.8.1 Functions.**

The NearArchive database recovery utility performs the following functions:

- Processes all recovery control entries in a single storage level or all storage levels of a NearArchive database.
- For each tape dataset which is in recovery status with a copy identifier of 'B', 'C' or 'D', the utility will recreate the primary dataset ('A') copy. The existing primary dataset will be uncataloged prior to recovery. The utility will recover from the backup copy identified in the recovery control entry, unless the 'COPYFULL=' execution parameter is specified, in which case it will use the backup copy identified in this parameter. In these circumstances, this backup copy will be used to recover all datasets which are in recovery status, irrespective of which copy has been designated for recovery control via the NearArchive database administration process in TSO/ISPF.
- The utility will recover a primary tape dataset using a combination of full-volume and incremental backup copies, in the following sequence:

a) Recovery will first be performed from a full-volume backup copy, using the copy identifier of the recovery control entry for the primary dataset (as set during recovery control processing using the NearArchive data administration process in TSO/ISPF), or from the copy identifier in the 'COPYFULL=' execution parameter, if specified.

If the designated backup copy does not exist (i.e.. is not cataloged) the utility will take the following actions, depending on the copy identifier selected (or specified via execution parameter):

'B' copy - if the 'B' copy does not exist, proceed with incremental recovery processing.

'C' copy - if the 'C' copy does not exist, issue error message OTS13041.

'D' copy - if the 'D' copy does not exist, check whether a 'B' copy exists for primary dataset being recovered. If it does, issue error message OTS13041; if it does not exist, proceed with incremental backup processing.

a) After all objects have been restored from the appropriate full-volume backup dataset (where one exists), or after skipping full-volume recovery processing (where a full-volume backup dataset does not exist), the utility will continue with recovery from incremental backup datasets.

If one or more incremental backup datasets exist for the primary tape dataset being recovered, the utility will process each of these datasets in ascending numeric sequence (as specified in the low-level qualifier of the incremental backup dataset; see section 7.4.1. for a description of the incremental backup dataset naming convention). All objects from each incremental backup dataset will be restored in sequence to the primary tape dataset. When all incremental backup datasets associated with the primary dataset have been processed, then recovery of that primary dataset is complete.

If no incremental backup dataset exists for the primary tape dataset being recovered, then no incremental backup processing will be performed, and recovery of that primary dataset is complete.

- After successful recovery of a primary tape dataset, it will be removed from recovery status.
- A processing report will be produced, giving details of all primary datasets recovered and uncataloged during execution of the utility.
- 'Unavailable' volumes (i.e. volumes in recovery status which have been marked with an 'X' copy) will not be processed by the utility.

### **7.8.2 JCL Requirements.**

The following JCL may be used to execute the NearArchive database recovery utility:

```
//STEP1      EXEC PGM=OTASP130,REGION=4M,
//          PARM=( 'COPYFULL=B|C|D',
//                'COPYINC=I|J'
//                'LEVEL=n|ALL'
//                'NODEVCHECK')
//          *
//STEPLIB    DD    DSN=NearArchive.load.library,DISP=SHR
//          *
//OTASV100   DD    DSN=Database.primary.index.name,DISP=SHR
//SYSPRINT   DD    SYSOUT=*
```

The following optional EXEC parameters may be specified:

**COPYFULL:** This optional parameter is used to control which full-volume backup copy is used during recovery processing for all primary datasets being recovered during execution of the utility. The following operands may be specified:

**B** The utility is to use a primary backup ('B' copy) dataset.

**C** The utility is to use a duplex ('C' copy) dataset.

**D** The utility is to use a duplex backup ('D' copy) dataset.

If this parameter is omitted, the utility will use the copy identifier associated with the recovery control entry containing the dataset being recovered. This is the default process.

**COPYINC:** This optional parameter is used to control which incremental backup copies are used during recovery processing for all primary datasets being recovered during execution of the utility. The following operands may be specified:

**I** The primary incremental backup ('I') copy should be used for recovery processing.

**J** The duplex incremental backup ('J') copy should be used for recovery processing.

If this parameter is omitted, the utility will use the primary incremental backup ('I') copy.

**LEVEL :** Specify the LEVEL parameter to identify the database storage level to be processed by the utility. The parameter value 'n' may take one of the following values:

'n' - process storage level n only

'ALL' - process all storage levels in the database.

If this parameter is omitted, a default value of 'ALL' is used.

Where a database has only one storage level, specifying a value of 'ALL' for this parameter (or allowing it to default), will cause the utility to process storage level 0 only.

**NODEVCHECK:** This optional parameter is used to control processing when a volume capacity mismatch has been detected between a full-volume backup tape volume and a primary tape volume, during full-volume recovery processing.

If this parameter is present, the utility will issue a warning message and generate a condition code of 4, but will continue with recovery processing.

If this parameter is omitted, standard capacity checking will be performed during full-volume recovery processing. Any mismatch will cause the utility to terminate with condition code 12.

Dataset names should be set as follows for each DD entry:

**STEPLIB:** The name of the NearArchive version 2 product library, as unloaded during the product installation procedure (see chapter 2).

**OTASV100:** The name of the primary index of the NearArchive database to be processed by the utility. The utility will only process one NearArchive database per execution.

**SYSPRINT:** The destination of the utility report file. If required, the report may be written to a sequential dataset, with record format = FBA, and a logical record length of 133 bytes.

### **7.8.3 Condition Codes.**

OTASP130 will set a condition code on completion. This code may take the following values:

- 0 - the utility has been executed successfully.
- 4 - a warning condition has been encountered during execution of the utility, but processing has continued. A warning message will have been written to the SYSPRINT dataset.



- 8 - an error condition has been encountered during execution of the utility, but processing has continue. An error message will have been written to the SYSPRINT dataset.
- 12 - a serious error has occurred during execution of the utility. An error message will have been written to the SYSPRINT dataset. Execution of the utility has been terminated.

#### **7.8.4 Running the Utility.**

Prior to executing the database recovery utility, the tape cartridge volume(s) to be recovered must be placed into recovery status. This is performed via the volume recovery function in the supplied NearArchive database administration ISPF dialogs. Refer to section 3.4 of this manual for a full description of how to perform this function.

Once a volume has been placed in recovery status, all accesses to that volume for retrieval of NearArchive objects will be satisfied from the backup copy identified in the corresponding volume recovery entry. This copy should be made available for retrieval processing as required. It must also be available for recovery processing by the database recovery utility.

After successful retrieval of a primary dataset, the corresponding backup copy will no longer be used for retrieving NearArchive objects, and may be returned to its normal location.

#### **7.8.5 Utility Processing Reports.**

The following reports will be written by the database recovery utility to the SYSPRINT dataset.

- Recovery processing report.

This report will print details of each primary tape dataset re-created during the recovery procedure, and of each existing primary tape dataset which has been released and uncataloged during utility execution.

Error or warning messages generated during execution of the utility will be written to the recovery processing report. All messages will have the message identifier 'OTS130xx', where 'xx' is a numeric value in the range 00-99.

Refer to chapter 8 of this manual for a description of all messages issued by the database recovery utility.

### **7.8.6 Utility Failure and Restart Considerations.**

The NearArchive database recovery utility may be rerun as normal after failure or cancellation of a recovery process. There are no dataset recovery requirements or any other special actions to be performed prior to rerunning the utility.

After successful recovery of a tape dataset, it will be removed from recovery status in the appropriate storage level in the NearArchive database being processed. All recovery processing for this dataset is then complete.

On rerunning the database recovery utility, any datasets successfully recovered during the failed run will have been removed from recovery status, and will consequently not be processed during the rerun of the utility.

Because a tape dataset is only removed from recovery status after successful recovery of its primary copy, failure or cancellation of the utility during recovery of a tape dataset will not alter that dataset's status. All retrievals for that dataset will continue to be satisfied from the identified backup copy after utility failure or cancellation.

During rerun of the utility, the partially restored primary copy dataset will be uncataloged prior to recovery, and the recovery process repeated for that dataset.

If there are many tape cartridge volumes in recovery status, or if the entire NearArchive database is being recovered, the database recovery procedure may take some time to execute. In these circumstances, the procedure may be cancelled by the system operator at any time, in order to allow access to the database from other jobs. Execution of the database recovery procedure may subsequently be continued as required.

---

# CHAPTER 8

---

## *Messages and Codes*

This page is intentionally left blank

NearArchive components communicate with the user through informational, warning and error messages. All NearArchive messages are accompanied by a message identifier which begins with the characters 'OTS', followed by a 5-character numeric identifier.

NearArchive messages may be displayed on the system log, in a NearArchive report file, in the CICS 'CSMT' transient data queue, or on a CICS display terminal. Messages displayed in the CICS 'CSMT' destination will include a date and timestamp after the message identifier.

NearArchive sets a two-character return code in response to each NearArchive request issued via the application program interface to indicate the result of the request. Additional error information for unsuccessful requests may be returned in two half-word supplementary code fields in the program interface.

This chapter documents all NearArchive request codes, all return and supplementary code values, and all messages, categorized by NearArchive component.

## 8.1 NearArchive Request Codes.

---

All requests to NearArchive for archival and retrieval of data are made using the NearArchive application program interface. Each request is identified by a two-character numeric value. Requests may be accompanied by additional values of one or two half-word supplementary code fields.

This section lists request and supplementary codes used by version 2.5 of NearArchive. Full details on request code usage are available in chapter 4 of this manual.

A '-' is used in the appropriate column to indicate that a supplementary request code is not required.

<b>Request code</b>	<b>Supp code 1</b>	<b>Supp code 2</b>	<b>Description</b>
40	-	-	Connect to a NearArchive database (batch applications only).
41	Query identifier (0-200)	Generic key length/Sort sequence (0-3)	<p>Query a NearArchive index (primary or secondary).</p> <p>Supplementary code 1 gives the query identifier (0/100 for primary index, 1-99/101-199 for secondary index with that identifier, 200 for volume serial number query).</p> <p>Supplementary code 2 identifies the number of bytes in the search argument and the sequence in which index entries are to be returned to the calling application.</p>
42	0 or 1	-	<p>Retrieve a record from a NearArchive object (batch applications only).</p> <p>Supplementary code 1 is set to 0 for direct retrieval and 1 for sequential retrieval.</p>
43	0 or 1	Request priority (0-9)	Retrieve a NearArchive object into a temporary storage queue (CICS applications only).

<b>Request code</b>	<b>Supp code 1</b>	<b>Supp code 2</b>	<b>Description</b>
			Supplementary code 1 is used to control the location of the temporary storage queue (0 for auxiliary, 1 for main).
44	-	-	Write an object record to a NearArchive database (batch applications only).
46	Secondary index id (1-99)	0 or 1	<p>Identify secondary key to be used to index object currently being archived (batch applications only).</p> <p>Supplementary code 1 gives the identifier of the secondary index to be used (in the range 1 to 99).</p> <p>Supplementary code 2 is set to 0 if the secondary key is to be generated automatically, or 1 if the secondary key is provided in the program interface area.</p>
47	-	-	Delete an object.
48	-	-	Commit database updates (batch applications only).
49	0 to 2	0 or 1	<p>Disconnect a NearArchive database (batch applications only).</p> <p>Supplementary code 1 is used to determine the disposition of the tape cartridge volume after disconnecting the database.</p> <p>Supplementary code 2 is used to control commit processing prior to disconnecting the database.</p>

## 8.2 NearArchive Return and Supplementary Codes.

The following return and supplementary codes will be returned by NearArchive in the program interface area after a processing request has been completed.

Return code	Supp code 1	Supp code 2	Description
00	0	0	Request completed successfully  <b>Action:</b> None.
	4	0	Request completed, but disk copy of object not written. A tape copy of the object has been successfully written.  <b>Action:</b> None. This is a warning code.
04	0	0	<u>Batch processing:</u>  A logical record has been retrieved from a NearArchive database, but its length exceeded the maximum set by the calling program in the program interface parameters. The record has been truncated.  <b>Action:</b> amend length value in call interface parameters.  <u>CICS retrieval processing:</u>  The maximum number of objects to be returned in response to a query request has been exceeded. The target temporary storage queue containing the results of the request will identify the first 'n' objects satisfying the query criteria, where 'n' is the maximum number set on entry.  <b>Action:</b> None. Processing may continue normally.
	1	0	No records retrieved during partial object retrieval, using the segment co-ordinates supplied in the request.



Return code	Supp code 1	Supp code 2	Description
	2	0	<p><b>Action:</b> Inform the user that no data meets the partial object retrieval requirements, or retry the request with the correct co-ordinates.</p> <p>End of object reached during partial object retrieval before processing the number of records specified in the segment co-ordinates supplied in the request.</p> <p><b>Action:</b> Inform the user that an incomplete segment has been retrieved, or retry the request with the correct co-ordinates.</p>
	4	0	<p>Multiple temporary storage queues have been used to contain the retrieved data because there were more than 32,767 records in the object or object segment being retrieved.</p> <p><b>Action:</b> The application should sequentially process all temporary storage queues in the string in order to access all data for the retrieved object or object segment. Refer to chapter 6 for information on the NearArchive temporary storage queue stringing facility.</p>
	6	0	<p>This is a combination of the conditions identified by supplementary code 1 settings of 4 and 2. Refer to the above description of both these values of the supplementary code 1 field for information on the required actions.</p>
12	0	0	<p>The record length passed during a batch archival request exceeded the maximum allowed for a database.</p> <p><b>Action:</b> Terminate processing. Amend application logic to reduce size of record passed to NearArchive.</p>

Return code	Supp code 1	Supp code 2	Description
13	file id	R15/R0	<p>An error has occurred retrieving the identifier of a data block from a NearArchive tape dataset.</p> <p>Supplementary code 1 gives the identifier of the file on which the error occurred (see description of file identifiers in section 8.2.1).</p> <p>Supplementary code 2 gives the contents of the low-order bytes from register 15 and register 0 on return from the NOTE macro.</p> <p><b>Action:</b> Terminate processing. Refer to IBM macro documentation for a description of the above values.</p> <p>Contact your NearArchive product support representative if there is no apparent reason for the error.</p>
14	1	0	Archival date in program interface area was not a valid date value.
	2	0	<p>Archive override date in program interface area for restart batch archival open request was not a valid date value.</p> <p><b>Action:</b> Terminate processing. Examine program logic or data to discover cause of error, and re-run job.</p>
15	0	0	<p>Normal archival open requested, but previous archival run ended abnormally.</p> <p><b>Action:</b> Amend application controls to request rerun processing, and rerun job.</p>
	1	Sec ix id	Normal archival open request, but an asynchronous update for the secondary index identified in supp code 2 is still outstanding.

Return code	Supp code 1	Supp code 2	Description
	2	Sec ix id	<p><b>Action:</b> Run the secondary index update utility to complete the asynchronous secondary index update process.</p> <p>An archival open request has been received, but both dynamic backout and asynchronous secondary index update have been enabled.</p> <p><b>Action:</b> Disable dynamic backout and/or asynchronous secondary index update processing and rerun the job.</p>
16	file id	R15/R0	<p>An error has occurred locating a block on a NearArchive tape dataset.</p> <p>Supplementary code 1 gives the identifier of the file on which the error occurred (see description of file identifiers in section 8.2.1).</p> <p>Supplementary code 2 gives the contents of the low-order bytes from register 15 and register 0 on return from the POINT macro.</p> <p><b>Action:</b> Terminate processing. Refer to IBM macro documentation for a description of the above values. Contact your NearArchive product support representative if there is no apparent reason for the error.</p>
17	0	0	<p>Duplicate NearArchive object identifier during batch archival. An object with the primary key and date identified in the call interface parameters already exists in the NearArchive database.</p> <p><b>Action:</b> Note duplicate record identification and proceed to next request. Check the program logic to identify cause of the duplicate request, and eliminate.</p>

Return code	Supp code 1	Supp code 2	Description
18	0	0	<p>Restart archival open requested, but previous archival run completed successfully.</p> <p><b>Action:</b> Terminate processing. Reset application controls to perform normal database open, and rerun job.</p>
19	X'01nn'	0	<p>Unable to add index entry (index record full), where 'nn' = '00' for the primary index, and a non-zero value in the range 01-99 to identify the corresponding secondary index. No further objects with the primary/secondary key value specified in the call interface parameters may be added to the database.</p> <p><b>Action:</b> Terminate processing. Run the NearArchive object management utility (OTASP040) to expire objects which have exceeded the database retention period.</p> <p>This will release space used in the database's primary/secondary index dataset, and allow objects to be added to the database.</p>
21	-  0  1	0	<p>One or more resources are not available to satisfy an object retrieval request. Supplementary code 1 contains a numeric value which identifies the unavailable resource:</p> <p>'MAXDRIVE' is currently at its maximum value (CICS retrieval only).</p> <p><b>Action:</b> The online retrieval request should be retried at a later time.</p> <p>'MAXQLLEN' is currently at its maximum value (CICS retrieval only).</p> <p><b>Action:</b> The online retrieval request should be</p>

Return code	Supp code 1	Supp code 2	Description
			retrieved at a later time.
	2		<p>Volume containing object is in 'unavailable' status ('X' copy recovery control status).</p> <p><b>Action:</b> None. Remove the volume from 'unavailable' status if access to this volume is required.</p>
	3		<p>'Volume unavailable' condition returned from dynamic allocation request. The volume containing the object is currently in use by another job.</p> <p><b>Action:</b> Retry the request when the volume has been freed by the other job.</p>
	4		<p>Allocation request cancelled by operator. There are no devices available to satisfy the retrieval request, and the operator has replied 'CANCEL' to the IEF238D message output by the system during allocation recovery processing.</p> <p><b>Action:</b> None.</p>
	5		<p>'Dataset unavailable' condition returned from dynamic allocation request. The tape dataset containing the object is currently in use by another job.</p> <p><b>Action:</b> Retry the request when the tape has been freed by the other job.</p>
	6		<p>NearArchive control region shutdown in progress (CICS retrievals only).</p> <p><b>Action:</b> Restart the control region before retrying the online retrieval request</p>
	7		<p>'Unit unavailable' condition returned from dynamic allocation request. All eligible tape devices are currently allocated to other jobs.</p>

Return code	Supp code 1	Supp code 2	Description
	8		<p><b>Action:</b> Retry the request when an eligible device becomes available.</p> <p>Request rejected because no resources have become available to process the request within the time period specified in the TAPEWAIT control region initialization parameter.</p> <p><b>Action:</b> Retry the request at a less busy time. The value of the TAPEWAIT parameter may need to be increased if this condition occurs frequently.</p>
23	0  C'XX'	0  C'YY'	<p>Abnormal condition detected during CICS retrieval.</p> <p>Abend 'XXYY' intercepted during CICS retrieval.</p> <p>A transaction dump will have been written to the CICS dump dataset for both the above conditions.</p> <p><b>Action:</b> Consult message in CSMT queue in NearArchive control region for details of the abnormal condition or abend.</p> <p>Take the appropriate action to ensure that the error does not re-occur.</p>
24	0	0	<p>NearArchive CICS transaction timeout.</p> <p><b>Action:</b> No action necessary, re-enter transaction.</p>
27	0	0	<p>NearArchive CICS control region not initialized.</p> <p><b>Action:</b> Enable the NearArchive control region for online retrieval by</p>

Return code	Supp code 1	Supp code 2	Description
			running transaction OT23 in that region.
28	1-99	0	Secondary index identified in supp code 1 not defined for this database (batch and CICS retrievals).  <b>Action:</b> Check the application logic to determine the cause of the error. Make the appropriate changes to ensure that error does not re-occur.
	file id	0	Specified database not defined correctly (CICS retrievals only). Supplementary code 1 gives the identifier of the database system file which has not been defined to CICS.  <b>Action:</b> Ensure that the system file identified in supplementary code 1 is correctly defined to CICS.
29	-	-	A request has been rejected by user exit processing, or the user exit module OTASUX01 (batch) or OTASUX02 (CICS) cannot be found. Supplementary code 1 identifies the type of request, as follows:
	1 2 3 4		Open database Retrieve object Write object Delete object
30	0	0	Last logical record in NearArchive object already retrieved (batch direct retrieval).
	1	0	End of NearArchive database encountered (batch sequential retrieval).  <b>Action:</b> None. Continue processing as required.

Return code	Supp code 1	Supp code 2	Description
31	0	0	Specified primary key does not exist in NearArchive database.
	1	0	NearArchive object with specified archive date does not exist for specified primary key.
	2	0	NearArchive object has been deleted.
	3	dataset seq no	The dataset containing the object is not cataloged (volume serial number query requests). Supp code 2 will contain the sequence number of the uncataloged dataset.  <b>Action:</b> None. Continue processing as required.
83	File id	-	Tape volume rejected during object storage processing:
		1	The Volsafe feature is not present on a new scratch tape, but Volsafe support is enabled for this tape copy in the storage level being updated.
		2	The Volsafe feature is present on a scratch tape, but Volsafe support is not enabled for the this tape copy in the storage level being updated.  <b>Action:</b> Close the database (with or without commit processing, as appropriate) and terminate processing of the application. Determine why a Volsafe / non-Volsafe tape has been mounted in response to the allocation request issued by NearArchive, and correct the problem. Then rerun the application.
85	0	0	Invalid data segment read when retrieving a disk copy of an object.
	1	0	No terminating segment when retrieving a disk



Return code	Supp code 1	Supp code 2	Description
			<p>copy of an object.</p> <p><b>Action:</b> Each of the above errors indicates a problem with the integrity of data in a disk ('K') copy dataset. Re-create the disk dataset using the NearArchive backup utility OTASP110. If this does not resolve the problem, contact your NearArchive product support representative.</p>
86	0	0	<p>Internal index error (object to be retrieved not in block identified in index entry).</p> <p><b>Action:</b> Terminate processing. Check that the correct dataset has been entered in the JCL, and that the primary index dataset has not been corrupted in any way.</p> <p>If it has, restore the primary index from the backup taken after the last database update procedure, and rerun the job.</p>
87	0	0	<p>NearArchive database has not been opened prior to access (batch requests only).</p> <p><b>Action:</b> Terminate processing. Examine program logic to determine cause of error. Make the appropriate changes to ensure that problem does not re-occur.</p>
88	0	0	<p>NearArchive database is already connected (batch requests only).</p> <p><b>Action:</b> Terminate processing. Examine program logic to determine cause of error. Make the appropriate changes to ensure that problem does not re-occur.</p>

Return code	Supp code 1	Supp code 2	Description
89	file id	0	<p>System file not open for update processing. An object deletion request has been issued by an application, but the NearArchive database has not been connected for input-output processing (batch), or the primary index or journal dataset has not been defined for update access in the File Control Table (CICS).</p> <p><b>Action:</b> Amend the database open call (batch) or File Control Table entries (CICS) and re-submit the request.</p>
90	file id	SVC99 error code	<p>An error has occurred attempting to dynamically allocate a NearArchive database system dataset.</p> <p><b>Action:</b> Terminate processing. Refer to IBM documentation on dynamic allocation for a description of the error code in supp code 2</p> <p>Take the appropriate corrective action and re-submit the request.</p> <p>If no apparent cause for failure can be detected, contact your NearArchive product support representative.</p>
91	file id	0	<p>NearArchive database not correctly initialized. The file identifier indicates which system dataset has not been properly set up.</p> <p><b>Action:</b> Terminate processing. Check that the correct NearArchive database index has been specified in the JCL.</p> <p>If no apparent cause for failure can be detected, contact your NearArchive product support</p>

Return code	Supp code 1	Supp code 2	Description
	0	1	<p>representative.</p> <p>NearArchive database being opened during batch archival or retrieval processing is at a pre-v2.5 level.</p> <p><b>Action:</b> The database must be converted for use with this release of the product, using the database conversion process in the data administration procedure under TSO/ISFP.</p> <p>Refer to chapter 3 of this manual for a description of this process.</p>
92	loc id	R15	<p>Failure trying to acquire virtual storage, or index record full when converting pre-v2.5 index entry.</p> <p>For batch applications, supp code 2 gives the return code received from the GETMAIN macro. Supp code 2 will always be 8 if an index entry conversion error has occurred.</p> <p>For CICS applications, the error occurred because insufficient storage was available in the CICS Dynamic Storage Area.</p> <p><b>Action:</b> For batch applications, if a pre-v2.5 index is being processed, then it is likely that a pre-v2.5 index entry record is being converted, and that the number of objects archived for the primary key now exceeds the maximum allowed. Refer to section 3.1.4 for information on main index entry sizing. Where possible, expire aged objects from the database to reduce the number of index sub-records, and re-archive the object.</p>

Return code	Supp code 1	Supp code 2	Description
			<p>If this condition is not true, refer to IBM system macro documentation for a description of the GETMAIN return code. If there is insufficient storage to satisfy the request, specify a larger REGION size, and rerun the job.</p> <p>For CICS applications, ensure that there is additional storage available in the Dynamic Storage Area, and resubmit the request.</p>
93	0	0	<p>Fatal error already received. NearArchive cannot continue because of a prior error condition.</p> <p><b>Action:</b> Terminate processing. Investigate the cause of the preceding error before rerunning the job.</p>
94	file id	ret/reas codes	<p>Error accessing system dataset.</p> <p>Supplementary code 1 gives the identifier of the file on which the error occurred (see description of file identifiers in section 8.2.1).</p> <p>Supplementary code 2 gives the return and reason codes from the failed VSAM I/O request.</p> <p><b>Action:</b> Terminate processing.</p> <p>Refer to IBM VSAM macro documentation for a description of the return and reason codes.</p> <p>Take the appropriate corrective action before resubmitting the job.</p> <p>If no apparent cause for failure can</p>

Return code	Supp code 1	Supp code 2	Description
			be detected, contact your NearArchive product support representative.
95	0	0	<p>Product not authorized.</p> <p><b>Action:</b> Check that NearArchive product authorization codes have been correctly applied, as described on page 2.7 of this manual.</p> <p>Contact your NearArchive product support representative if an authorization code update is required.</p>
96	0	0	<p>NearArchive module not found; program load error. A required NearArchive load module was not available at run-time.</p> <p><b>Action:</b> Terminate processing.</p> <p>Ensure that the NearArchive product distribution library is available to the job via the STEPLIB or JOBLIB DD statements, or from the system linklist concatenation.</p>
	1	0	<p>Error deleting NearArchive program module from virtual storage.</p> <p><b>Action:</b> None. This is a warning code. However, multiple instances of this condition during job execution may cause problems with virtual storage availability. If the problem persists, inform your NearArchive product support representative.</p>
97	file id	ret/reas	Error opening system file.

}

Return code	Supp code 1	Supp code 2	Description
		codes	<p>Supplementary code 1 gives the identifier of the file on which the error occurred (see description of file identifiers in section 8.2.1).</p> <p>Supplementary code 2 gives the hexadecimal return and reason codes (X'aabb') from a failed VSAM open processing request.</p> <p><b>Action:</b> Terminate processing.</p> <p>Refer to IBM VSAM macro documentation for a description of the return (X'aa') and reason (X'bb') codes, as described below:</p> <p>X'04' An error has occurred  X'08' issuing a VSAM OPEN  X'0C' macro  X'10' An error has occurred issuing a SHOWCAT macro</p> <p>X'14' An error has occurred issuing a BLDVRP macro  X'18' An error has occurred issuing a DLVRP macro.</p> <p>Take the appropriate corrective action before resubmitting the job.</p>
98	loc id	add. info.	<p>Internal NearArchive processing error. An internal logic error has been detected during NearArchive operation.</p> <p><b>Action:</b> Terminate processing. Retain any other information issued by the failing application, and report the error to your NearArchive product support representative</p>
99	0	0	Invalid request code received.
	1	0	Invalid supp code 1 received.

Return code	Supp code 1	Supp code 2	Description
	2	0	Invalid supp code 2 received.
	3	0	Database update request received but database was opened for secondary index update only.
	4	-	Invalid 'identify secondary index' request received:
		1	Supp code 2 = 2 (update specified object with secondary index entry) but database was not opened for secondary index update only.
		2	Supp code 2 = 0 or 1 (update secondary index for current object) but there is no associated object currently being archived. This can occur due to receipt of a 'commit' or 'delete' call immediately prior to the request.
	5	-	Invalid 'database connect' call (request code '40') received:
		1	Invalid data area length
		2	Invalid access request type
		3	Invalid database name list address
	6		
		0	Zero entry count for batch query call.
			<b>Action:</b> Terminate processing.  Examine application logic to identify the cause of the error. Re-submit the job after correcting the error.

### **8.2.1 System File Identification.**

Any error accessing/opening system files will be accompanied by a supplementary code 1 value of X'abcc', where the hexadecimal codes are defined as follows:

<b>a</b>	<b>b</b>	<b>cc</b>	
-	0	-	Tape dataset error.
0	0	00	Error occurred on primary tape dataset.
0	0	01	Error occurred on duplex tape dataset.
-	1	-	Index dataset error.
0	1	-	No record type identified.
0	1	00	Error occurred on primary index dataset.
0	1	non-zero	Error occurred on secondary index identifier X'cc' dataset.
1	1	-	Error occurred accessing processing control record.
2	1	stg lev	Error occurred accessing storage level control record. Bytes 2 and 3 identify the storage level
3	1	-	Error occurred accessing main index entry record.
4	1	-	Error occurred accessing supplementary index entry record.
5	1	-	Error occurred accessing secondary index record
-	2	-	Space management dataset error.
-	2	cc	cc = storage level identifier
-	3	-	Journal dataset error.
-	4	-	Migration control dataset error.
-	5	-	Disk dataset error. Error occurred accessing disk ('K') copy dataset.



## 8.3 Index Print Utility Messages.

---

### **OTS01000 - NEARARCHIVE INDEX PRINT UTILITY COMPLETED WITH CONDITION CODE xx**

Explanation: This informational message is issued when the index print utility has completed processing. 'xx' is the highest condition code encountered during this execution.

Action: No action necessary.

### **OTS01001 - ERROR X'xxyy' RECEIVED OPENING FILE aaaaaaaa DATASET bbbb....bbbb**

Explanation: A VSAM open error has occurred when opening the identified NearArchive dataset. 'xx' and 'yy' are the hexadecimal error and reason codes returned from the VSAM OPEN macro. 'aaaaaaa' is the DD name associated with the file, and 'bbbb....bbbb' is the name of the VSAM dataset. Processing is terminated.

Action: Refer to IBM documentation on VSAM macro processing for details of the return and reason codes. Take the appropriate corrective action and rerun the job.

### **OTS01002 - RETURN CODE xx RECEIVED READING JFCB FOR OTASV100**

Explanation: An error has been encountered locating the JFCB control block for the primary index file being processed by the utility. 'xx' gives the return code from the RDJFCB macro. Processing is terminated.

Action: Refer to IBM documentation on RDJFCB macro processing for details of the return code. Take the appropriate corrective action and rerun the job.

### **OTS01003 - DUPLICATE EXECUTION PARAMETER xxxxxxxxxx**

Explanation: The execution parameter identified in the message has been specified more than once in the execution parameter list. Processing of the utility is terminated.

Action: Remove the duplicate parameter entry and rerun the job.

**OTS01004 – PARAMETER LENGTH IS INCORRECT FOR EXECUTION  
PARAMETER xxxxxxxxxxxx**

Explanation: The length of the operand of the execution parameter identified in the message is greater than the maximum permitted value. Processing of the utility is terminated.

Action: Correct the invalid parameter entry and rerun the job.

**OTS01005 - ERROR X'xyy' READING FILE dddddddd eeee....eeee**

Explanation: An error has occurred reading a VSAM file. 'xx' and 'yy' are the hexadecimal error and reason codes returned from the processing request. 'ddddddd' is the DD name associated with the file, and 'eeee....eeee' is the name of the VSAM dataset. Processing is terminated.

Action: Refer to IBM documentation on VSAM macro processing for details of the return and reason codes. Take the appropriate corrective action and rerun the job.

**OTS01006 – INVALID CHARACTER SPECIFIED IN COUNT  
PARAMETER**

Explanation: A non-numeric character has been specified in the operand of the COUNT execution parameter. Processing of the utility is terminated.

Action: Correct the invalid parameter entry and rerun the job.

**OTS01007 - INSUFFICIENT STORAGE FOR WORKAREAS.**

Explanation: A request by the utility for main storage allocation has failed.

Action: Increase the REGION size specified for the job and resubmit.

#### **OTS01008 – INVALID EXECUTION PARAMETER SPECIFIED**

Explanation: A invalid parameter keyword has been detected when validating execution parameters. Processing of the utility is terminated.

Action: Correct the invalid parameter entry and rerun the job.

#### **OTS01009 – ENDKEY AND COUNT PARAMETERS CANNOT BOTH BE SPECIFIED**

Explanation: The ENDKEY and COUNT parameters are both present in the execution parameter list. This is an invalid combination. Execution of the utility is terminated.

Action: Correct the invalid parameter entry and rerun the job. Refer to section 7.1 for a description of the execution parameters which may be specified for the utility.

#### **OTS01010 – STARTKEY PARAMETER IS HIGHER THAN ENDKEY**

Explanation: The primary key value specified in the STARTKEY parameter is greater than that specified in the ENDKEY parameter. This is an invalid condition. Execution of the utility is terminated.

Action: Correct the invalid parameter entry and rerun the job. Refer to section 7.1 for a description of the execution parameters which may be specified for the utility.

#### **OTS01098 - UNABLE TO OPEN SYSPRINT DATASET**

Explanation: An error has occurred opening the SYSPRINT dataset.

Action: Refer to other available system messages to discover the cause of the open failure. Correct the error and resubmit the job.

## 8.4 Tape Scan Utility Messages.

---

### **OTS02000 - NEARARCHIVE TAPE SCAN UTILITY COMPLETED WITH CONDITION CODE xx**

Explanation: This informational message is issued when the tape scan utility has completed processing. 'xx' is the highest condition code encountered during this execution.

Action: No action necessary.

### **OTS02001 - ERROR X'xxyy' RECEIVED OPENING FILE aaaaaaaa DATASET bbbb....bbbb**

Explanation: A VSAM open error has occurred when opening the identified NearArchive dataset. 'xx' and 'yy' are the hexadecimal error and reason codes returned from the VSAM OPEN macro. 'aaaaaaa' is the DD name associated with the file, and 'bbbb....bbbb' is the name of the VSAM dataset. Processing is terminated.

Action: Refer to IBM documentation on VSAM macro processing for details of the return and reason codes. Take the appropriate corrective action and rerun the job.

### **OTS02002 - RETURN CODE xx RECEIVED READING JFCB FOR OTASV100**

Explanation: An error has been encountered locating the JFCB control block for the primary index file being processed by the utility. 'xx' gives the return code from the RDJFCB macro. Processing is terminated.

Action: Refer to IBM documentation on RDJFCB macro processing for details of the return code. Take the appropriate corrective action and rerun the job.

### **OTS02003 - DUPLICATE EXECUTION PARAMETER xxxxxxxxxx**

Explanation: The execution parameter identified in the message has been specified more than once in the execution parameter list. Processing of the utility is terminated.

Action: Remove the duplicate parameter entry and rerun the job.

**OTS02004 – PARAMETER LENGTH IS INCORRECT FOR EXECUTION  
PARAMETER xxxxxxxxxxxx**

Explanation: The length of the operand of the execution parameter identified in the message is greater than the maximum permitted value. Processing of the utility is terminated.

Action: Correct the invalid parameter entry and rerun the job.

**OTS02006 – INVALID CHARACTER SPECIFIED IN EXEC  
PARAMETER xxxxxxxxxxxx**

Explanation: A non-numeric character has been specified in the operand of the execution parameter identified in the message. Processing of the utility is terminated.

Action: Correct the invalid parameter entry and rerun the job.

**OTS02007 – UNABLE TO OPEN INPUT TAPE FILE**

Explanation: An error has occurred opening the tape file to be scanned by the utility. Execution of the utility is terminated.

Action: Refer to other system messages for additional information concerning the failure. Rectify the problem and rerun the job.

**OTS02008 – INVALID EXECUTION PARAMETER SPECIFIED**

Explanation: A invalid parameter keyword has been detected when validating execution parameters. Processing of the utility is terminated.

Action: Correct the invalid parameter entry and rerun the job.

**OTS02009 – ENDBLK AND COUNT PARAMETERS CANNOT BOTH BE  
SPECIFIED**

Explanation: The ENDBLK and COUNT parameters are both present in the execution parameter list. This is an invalid combination. Execution of the utility is terminated.

Action: Correct the invalid parameter entry and rerun the job.  
Refer to section 7.1 for a description of the execution parameters which may be specified for the utility.

#### **OTS02010 – STARTBLK PARAMETER IS HIGHER THAN ENDBLK**

Explanation: The value specified in the STARTBLK parameter is greater than that specified in the ENDBLK parameter. This is an invalid condition. Execution of the utility is terminated.

Action: Correct the invalid parameter entry and rerun the job.  
Refer to section 7.1 for a description of the execution parameters which may be specified for the utility.

#### **OTS02011 – STARTBLK VALUE MUST BE GREATER THAN ZERO**

Explanation: An invalid value has been specified in the STARTBLK parameter. Execution of the utility is terminated.

Action: Correct the invalid parameter entry and rerun the job.  
Refer to section 7.1 for a description of the execution parameters which may be specified for the utility.

#### **OTS02098 - UNABLE TO OPEN SYSPRINT DATASET**

Explanation: An error has occurred opening the SYSPRINT dataset.

Action: Refer to other available system messages to discover the cause of the open failure. Correct the error and resubmit the job.

## 8.5 Database Maintenance Utility Messages.

---

### **OTS03000 - NEARARCHIVE DATABASE MAINTENANCE UTILITY COMPLETED WITH CONDITION CODE xx**

Explanation: This informational message is issued when the database maintenance utility has completed processing. 'xx' is the highest condition code encountered during this execution.

Action: No action necessary.

### **OTS03001 - OPEN FAILED FOR dddddddd, RETURN CODE = X'xx' REASON CODE = X'yy'**

Explanation: A VSAM open error has occurred when opening the identified NearArchive dataset. 'xx' and 'yy' are the hexadecimal return and reason codes returned from the VSAM OPEN macro. Processing is terminated.

Action: Refer to IBM documentation on VSAM macro processing for details of the return and reason codes. Take the appropriate corrective action and restart the job.

### **OTS03002 - NEARARCHIVE CALL FAILURE REQUEST=aa RETURN CODE=bb SUP CODE-1=X'xxxx' SUP CODE-2=X'yyyy'**

Explanation: An internal NearArchive call issued by the utility has failed. 'aa' is the request code, 'bb' is the return code, 'xxxx' and 'yyyy' are the supplementary codes returned from NearArchive.

Action: Refer to NearArchive documentation for details of the return and supplementary codes. Take the appropriate corrective action and restart the job.

### **OTS03003 - INVALID EXEC PARM, xxxxxxxxxxxxxxxxxxxxxxxxx**

Explanation: An invalid parameter has been specified on the EXEC statement. Processing is terminated.

Action: Refer to NearArchive documentation for EXEC parameter specification. Make the appropriate corrections and restart the job.

**OTS03004 - UNABLE TO OPEN DATASET WITH DDNAME=dddddddd  
DCB OPEN FLAG=xx**

Explanation: An open error has occurred when opening the non-VSAM dataset 'dddddddd'. 'xx' is the DCB flag returned from the OPEN macro. Processing is terminated.

Action: Refer to IBM documentation for details of the OPEN error. Take the appropriate corrective action and restart the job.

**OTS03005 - DATABASE STORAGE LEVEL x DOES NOT EXIST**

Explanation: Storage level x does not exist in the NearArchive database. Processing is terminated.

Action: Examine the NearArchive database definition for the storage level specified. If no cause for the error can be identified, contact your NearArchive product support representative.

**OTS03006 - UNABLE TO GETMAIN STORAGE FOR xxxx BUFFER.**

Explanation: A request by the utility for main storage allocation has failed. 'xxxx' may contain the following values:

T000 - The storage was required for a tape data buffer.

KEY1 - The storage was required for a primary index control record buffer.

Action: Increase the REGION size specified for the job and resubmit.

**OTS03007 - POINT ERROR OTASV120 RETURN CODE=X'xx' REASON  
CODE=X'yyyyyyyy'**

Explanation: An error has occurred repositioning on the NearArchive space management dataset. 'xx' and 'yyyyyyyy' are the hexadecimal return and reason codes returned from the VSAM POINT macro. Processing is terminated.



Action: Refer to IBM documentation on VSAM macro processing for details of the return and reason codes. Take the appropriate corrective action and restart the job. If no cause of the error can be identified, contact your NearArchive product support representative.

**OTS03008 - PRE-VERSION 2.5 DATABASE FOUND - PROCESSING CANNOT CONTINUE**

Explanation: The NearArchive database maintenance utility cannot be run on databases which are in a pre-version 2.5 format. Processing is terminated.

Action: Ensure that the correct primary index dataset has been specified in the execution JCL. If necessary, correct the JCL and resubmit the job. Database indexes which were created with an earlier version of the product must be converted for use with the current release, via the NearArchive database administration facility in TSO/ISPF. Refer to section 3.5 of this manual for a description of this procedure.

**OTS03009 - UNABLE TO READ STORAGE LEVEL RECORD RETURN CODE = X'xx' FEEDBACK = X'yyyyyyyy'**

Explanation: An error has occurred reading storage level control information from the NearArchive database primary index dataset. 'xx' and 'yyyyyyyy' are the hexadecimal return and reason codes returned from the VSAM GET macro. Processing is terminated.

Action: Refer to IBM documentation on VSAM macro processing for details of the return and reason codes. Take the appropriate corrective action and restart the job.

**OTS03010 - UNABLE TO FREEMAIN STORAGE, LABEL=aaaaaaaa. RETURN CODE = X'xxxxxxxx' ADDRESS=X'bbbbbbbb' LENGTH=X'ccccccc'**

Explanation: An error occurred when the utility attempted to release a previously allocated main storage area. 'xxxxxxxx' is the return code from the FREEMAIN macro.

Action: Refer to IBM documentation for a description of the FREEMAIN return code. If no cause of the error can be found, contact your NearArchive product support representative.

**OTS03011 - WRITE ERROR ON OTASV120 RETURN CODE=X'xx'  
REASON CODE=X'yyyyyyy'**

Explanation: An error occurred when writing a record to the NearArchive database's space management dataset. 'xx' and 'yyyyyyy' are the hexadecimal return and reason codes respectively. Processing is terminated.

Action: Refer to IBM documentation on VSAM macro processing for details of the return and reason codes. Take the appropriate corrective action and restart the job.

**OTS03012 - INTERNAL PROCESSING ERROR AT LABEL aaaaaaaa.  
REASON=bbbb. PLEASE CONTACT NEARARCHIVE  
SUPPORT CENTRE.**

Explanation: A NearArchive internal error has occurred. Processing is terminated.

Action: Retain all information concerning the error, and contact your NearArchive product support representative.

**OTS03013 - UNABLE TO OPEN DATASET WITH DDNAME=OTAST000  
DCB OPEN FLAG=xx**

Explanation: An error has occurred when opening a NearArchive tape dataset during tape recycle processing. 'xx' is the DCB flag returned from the OPEN macro. Processing is terminated.

Action: Refer to IBM documentation for details of the OPEN error. Take the appropriate corrective action and restart the job.

**OTS03014 - aaaaaaaaaaaaaaaaaaaaaaaaaaaaaa vvvvvv  
bbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbb**

Explanation: This is an informational message where 'aaaa...aaaa' is a tape or disk dataset name, vvvvvv is a tape volume serial number and 'bbbbb...bbbb' is the action taken on that dataset. 'bbbbb...bbbb' may contain the following values:-

**\*\* NOT RECYCLED - BLK1 IS ACTIVE STRADLE ITEM**

Explanation: The first block on the tape cartridge volume contains data which is continued from the end of the previous tape dataset in the database. Volumes in this category cannot be recycled.

**VOLUME RELEASED**

Explanation: The volume vvvvvv has been released from the NearArchive database. It has not been uncataloged as the AUTOUNCAT facility is disabled for this database.

**VOLUME RELEASED AND UNCATALOGED**

Explanation: The volume has been released from the NearArchive database and has been automatically uncataloged by the utility.

**DISK DATASET UNCATALOGED AND DELETED**

Explanation: The identified disk copy dataset has been uncataloged and deleted after recycling of the corresponding tape volume.

Action: No action is necessary. Processing continues.

**MAXIMUM BLOCKCOUNT REACHED BEFORE END OF DSN**

Explanation: The maximum number of blocks to be written to a disk ('K') copy dataset has been reached during disk compression processing. The disk dataset is closed normally, and processing of the utility continues. Some objects from the uncompressed dataset may not be

present on the compressed dataset. Retrieval of these objects will be satisfied from tape.

Action: If it is required that more objects be held in the disk copy dataset, increase the maximum disk blockcount via the NearArchive database administration dialogs and recreate the disk copy dataset using the NearArchive database backup utility OTASP110.

### **DISK DATASET COMPRESSED**

Explanation: The identified disk dataset has been compressed successfully.

Action: No action is required. Processing continues normally.

### **OTS03015 - READ ERROR ON OTASV120 RETURN CODE=X'xx' REASON CODE=X'yyyyyyyy'**

Explanation: An error has occurred attempting to read a record from the NearArchive database's space management dataset. 'xx' and 'yyyyyyyy' are the hexadecimal return and reason codes returned from the VSAM GET macro. Processing is terminated.

Action: Refer to IBM documentation on VSAM macro processing for details of the return and reason codes. Take the appropriate corrective action and restart the job. If no cause of the error can be identified, contact your NearArchive product support representative.

### **OTS03016 - UNABLE TO READ PCR. GET RETURN CODE = X'xx' GET FEEDBACK = X'yyyyyyyy'**

Explanation: An error has occurred attempting to read the processing control record from the NearArchive database primary index dataset. 'xx' and 'yyyyyyyy' are the hexadecimal return and reason codes returned from the VSAM GET macro. Processing is terminated.

Action: Refer to IBM documentation on VSAM macro processing for details of the return and feedback codes. Take the appropriate corrective action and restart the job. If no cause of the error can be identified, contact your NearArchive product support representative.

**OTS03017 – I/O ERROR ON xxxxxxxx FILE: *a,b,c,d,e,f,g,h,i***

Explanation: An I/O error has occurred processing a disk or tape file used by the utility during tape recycle processing. 'xxxxxxx' identifies the file.

The following information about the error condition will be present in the message:

a: Job name  
b: Step name  
c: Device number  
d: Device type  
e: DD name  
f: Operation attempted  
g: Error description  
h: Block number  
i: Access method

Execution of the utility will be terminated.

Action: Review the error information in the message. Take the appropriate corrective action and rerun the job. If no apparent cause for the error can be determined, contact your Oracle product support representative.

**OTS03018 - DYNALLOC ERROR, REQ=aa RC=bb DD=ccccccc  
DSN=dddddddddddddddddddddddddddddddddddddd  
VOL=vvvvvv UNIT=uuuuuuuu**

**OTS03019 - INF=eeee ERR=ffff SYSC=g SYSD=hhhhhhhh DEFER=ii  
DISP=jjkkll SPACE=(pppppp,sssss)  
DCB=(mmmmmm,nnnnnn,oo)**

Explanation: An error has occurred dynamically allocating a dataset, and processing of the utility has been terminated.

Messages OTS03018 and OTS03019 are issued for problem determination purposes. Some or all of the following operands may appear:-

'aa' - internal NearArchive request code.  
'bb' - internal NearArchive return code

'ccccccc'	-	DDname of the dataset being processed
'ddd...ddd'	-	name of the dataset being processed.
'vvvvvv'	-	serial number of the volume containing the dataset being processed
'uuuuuuuu'	-	the unit name of the device on which the dataset is to be allocated
'eeee'	-	the dynamic allocation information code
'ffff'	-	the dynamic allocation error code
'g'	-	SYSOUT Class (for SYSOUT dataset allocation)
'hhhhhhh'	-	name of OUTPUT DDname (for SYSOUT dataset allocation)
'ii'	-	is the deferred mount attribute
'jjkkll'	-	status, normal disposition and conditional disposition indicator values
'pppppp'	-	primary allocation value
'sssss'	-	secondary allocation value
'mmmmmm'	-	block size
'nnnnnn'	-	logical record length
'oo'	-	record format indicator value.

Action: Refer to IBM documentation on dynamic allocation processing for a description of the error and information codes. Take the appropriate corrective action and restart the job.

**OTS03025 - RECYCLE THRESHOLD IS ZERO. NO VOLUMES WILL BE RECYCLED FOR THIS STORAGE LEVEL.**

Explanation: The recycle threshold specified in the NearArchive database definition is zero, indicating that tape recycle processing is disabled for this storage level.

Action: Examine the NearArchive database definition for the storage level specified. Specify a recycle threshold for the storage level if recycling is required and resubmit the job.

**OTS03026 - RESTART=NO INDICATED BUT RECYCLE DATASET EXISTS. RESTART=YES REQUIRED.**

**OTS03026 - RESTART=NO INDICATED BUT RESTART DATASET OTASS030 CONTAINS RESTART INFORMATION. RESTART=YES REQUIRED**

Explanation: A previous attempt to run this utility has failed abnormally but RESTART=YES has not been specified. The first message format is issued when no OTASS030 over-ride DD card is present in the JCL. The second message format is issued when an OTASS030 DD card has been specified.

Action: Ensure that the RESTART parameter is correctly specified and rerun the job.

**OTS03027 - RESTART DETECTED. STORAGE LEVEL xx NOT FOUND OR NOT SPECIFIED FOR RESTART.**

Explanation: An attempt to rerun this utility has failed because storage level xx is not found in the NearArchive database or an invalid storage level has been specified on the LEVEL parameter on the EXEC statement.

Action: Examine the output and the execution parameters specified to determine the cause of failure. Take the appropriate corrective action and restart the job with RESTART=YES specified on the EXEC statement.

**OTS03028 - ERROR IN NOTE PROCESSING. RETURN CODE = X'xx' LAST BLOCK ID OBTAINED = yyyyyyyy**

Explanation: An error has occurred during NOTE macro processing. 'xx' is the hexadecimal return code returned from the NOTE macro. yyyyyyyy is the last block ID successfully obtained. Processing is terminated.

Action: Refer to IBM system macro documentation for details of the return code. Take the appropriate corrective action and restart the job. If no cause of the error can be identified, contact your NearArchive product support representative.

**OTS03029 - RESTART REQUESTED/REQUIRED. RESTART DSN=  
'aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa' DOES NOT  
CONTAIN CONTROL RECORD.**

**OTS03029 - RESTART REQUESTED/REQUIRED. RESTART FILE  
OTASS030 DOES NOT CONTAIN CONTROL RECORD.**

Explanation: An attempt to rerun this utility has failed because the dataset used by the utility for restart processing does not contain the necessary control record. The first message format is issued when no OTASS030 over-ride DD card is present in the JCL. The second message format is issued when an OTASS030 DD card has been specified.

Action: Examine the output from the previous execution of the utility and determine if possible, the reason why the restart dataset does not contain the control record. If no cause of the error can be identified, contact your NearArchive product support representative.

**OTS03030 - \*\*\*\*\* RESTART REQUESTED AND REQUIRED. RESTART  
DATASET DID NOT EXIST AT START OF JOB. \*\*\*\*\***

Explanation: The utility has been restarted but the dataset used by the utility for restart processing did not exist as expected when the job started.

Action: Examine the output from the previous execution of the utility and determine if possible, the reason why the restart dataset did not exist when the utility was restarted. If no cause of the error can be identified, contact your NearArchive product support representative.

**OTS03031 - NORMAL PROCESSING REQUESTED BUT DATABASE IS  
IN UNCOMMITTED STATE - RESTART REQUIRED**

Explanation: A normal run of the database maintenance utility has been requested (RESTART=NO), but the NearArchive database being processed is in an uncommitted state. Processing of the utility is terminated.

Action: A NearArchive database cannot be opened for normal processing when it is in an uncommitted state. Identify



the database update process which terminated abnormally and left the database in this condition. This update process will need to be restarted.

If the database has been left in an uncommitted state after failure of the previous execution of the database maintenance utility, rerun the utility with an EXEC parameter of 'RESTART=YES'

#### **OTS03032 - PROCESSING TERMINATED BY OPERATOR REQUEST**

Explanation: Execution of the utility has been terminated after input of an operator STOP command.

Action: None. This is an informational message.

#### **OTS03034 - DATABASE IN UNCOMMITTED STATE – OBJECT ARCHIVAL/MIGRATION PROCESS INCOMPLETE**

Explanation: The database being recycled is currently in an uncommitted state due to an incomplete object archival or migration process (as indicated in the message). Processing of the utility is terminated.

Action: The recycle utility will not process a database when another incomplete update process is still outstanding. Ensure that the outstanding archival or migration process is run to completion before rerunning the recycle utility.

#### **OTS03035 - INTEGRITY FAILURE IN STORAGE LEVEL s SPACE MANAGEMENT FILE – RECORD FOR TAPE SEQUENCE nnnnn IS MISSING**

Explanation: During database maintenance processing, a missing record has been detected in the space management file for storage level 's'. 'nnnnn' gives the identity of the missing record. Processing of the utility is terminated with condition code 12.

Action: This message indicates that there is an integrity error in the space management file. This error must be corrected before processing of the utility can be restarted. Contact your Lifecycle Director product support representative to obtain information on the required corrective procedure.

**OTS03037 - CONFLICT ON OTASS030 FILE - REMOVE OTASS030 DD  
STATEMENT FROM THE JCL AND RERUN THE JOB**

Explanation: An OTASS030 DD card is present in the JCL, identifying a restart control dataset, but a system-generated restart control dataset also exists. This indicates that an incomplete recycle operation is still in progress for this database.

Action: Remove the OTASS030 DD card from the JCL and restart the incomplete recycle operation using the system-generated restart control dataset. This dataset will be deleted at successful conclusion of recycle processing. The OTASS030 DD card may then be used in subsequent executions of the utility.

**OTS03040 - RECYCLE=NO AND COMPRESS=NO SPECIFIED (OR  
DEFAULTED)**

Explanation: Both 'RECYCLE=NO' and 'COMPRESS=NO' have been specified in the utility EXEC parameter, or have been allowed to default to these values. This is an invalid combination of parameters. Execution of the utility terminates with condition code 12.

Action: Modify the EXEC parameters as required and rerun the job.

**OTS03041 - OTS03041 RESTART DETECTED. RESTART FLAG X'xx' IS  
NOT VALID**

Explanation: Restart of a failed execution of the utility has been requested, but the contents of the restart dataset are invalid. Execution terminates.

Action: If a DD entry for file OTASS030 is present in the JCL, ensure that the dataset specified in this entry is the same as that specified in the JCL for the previous failed execution of the utility. If it is, ensure that this dataset has not been modified in any way.

If no DD entry for file OTASS030 is present in the JCL, ensure that the restart dataset (named 'hlq.RECYCLE', where 'hlq' is the high-level qualifier defined for the NearArchive database being processed) has not been

modified in any way.

If no apparent cause for the error can be identified, contact your NearArchive product support representative.

**OTS03042 - DISK COPY PROCESSING NOT ENABLED IN STORAGE  
LEVEL FOR DSN = xxxx....xxxx**

Explanation: A disk dataset has been selected for compression, but disk copy processing has been disabled for the database storage level containing the dataset. Disk compression is not performed, and processing of the utility continues.

Action: Ensure that disk copy processing for the storage level containing the identified dataset is enabled (via the NearArchive database administration dialog) and rerun the utility at a later time.

**OTS03043 - BLOCK COUNT MISMATCH CREATING L COPY OBJECT  
FOR DISK DSN = xxxx....xxxx**

Explanation: The disk copy of an object being processed during disk compression processing does not contain the number of blocks identified in its index entry. Processing of the utility is terminated with condition code 12.

Action: This error indicates a possible data integrity problem with the disk copy dataset. The dataset should be re-created using the NearArchive database backup utility OTASP110. Execution of the database maintenance utility may then be restarted.

**OTS03044 - DISK COMPRESS TERMINATED FOR DSN = xxxx....xxxx.  
ABEND CODE = X'yyy'**

Explanation: System abend 'yyy' has occurred during disk compression processing. The disk dataset being processed is closed normally, and processing of the utility continues. Some objects from the uncompressed dataset may not be present on the compressed dataset. Retrieval of these objects will be satisfied from tape.

Action: Refer to IBM documentation for a description of the system abend code. If the failure occurs due to lack of

disk space during compress processing, make more disk space available for NearArchive use and recreate the disk copy dataset using the NearArchive database backup utility OTASP110.

**OTS03045 - RETURN CODE X'yy' FROM SORT ON DSN = xxxx....xxxx  
COMPRESS FAILED FOR THIS DATASET**

Explanation: Return code X'yy' was received from an internal invocation of the system sort utility. Database maintenance processing is terminated with condition code 12.

Action: Refer to the report issued by the sort utility (in spool file SYSOUT in the job output) for more information on the cause of the error. Take the appropriate corrective action and restart the database maintenance utility.

**OTS03046 - RETURN CODE = X'yy' FROM IDCAMS RENAMING DSN  
= xxxx....xxxx COMPRESS FAILED FOR THIS DATASET**

Explanation: Return code X'yy' was received from an internal invocation of the Access Method Services utility IDCAMS. Database maintenance processing is terminated with condition code 12.

Action: Refer to the report issued by IDCAMS (in spool file OTASS038 in the job output) for more information on the cause of the error. Take the appropriate corrective action and restart the database maintenance utility.

**OTS03048 - BLDVRP FAILED. RETURN CODE=X'xx'. PROGRAM  
EXECUTION TERMINATED**

Explanation: Return code X'xx' was received from a request to build a VSAM Local Shared Resources buffer pool. Processing of the utility is terminated with condition code 12.

Action: Refer to the IBM documentation on the BLDVRP macro for an explanation of the return code. Take the appropriate corrective action and restart the utility.

**OTS03049 - DLVRP FAILED. RETURN CODE=X'xx'. PROGRAM  
EXECUTION TERMINATED**

Explanation: Return code X'xx' was received from a request to delete a VSAM Local Shared Resources buffer pool. Processing of the utility is terminated with condition code 12.

Action: Refer to the IBM documentation on the DLVRP macro for an explanation of the return code. Take the appropriate corrective action and restart the utility.

**OTS03051 - EXEC PARAMETERS 'RECYCLE=YES' AND  
'COMPRESS=YES' ARE MUTUALLY EXCLUSIVE**

Explanation: EXEC parameters 'RECYCLE=YES' and 'COMPRESS=YES' have been specified for utility execution. This is an invalid combination.

Action: Correct the EXEC parameters to enable either tape recycle or disk compress processing, and resubmit the job.

**OTS03052 - PARAMETER 'RECYCLE|COMPRESS=YES' IS REQUIRED  
FOR THIS RESTART RUN**

Explanation: A restart of a failed database maintenance process is being executed, but the value of the RECYCLE or COMPRESS EXEC parameter (as identified in the message) has been changed from its setting during the failed execution.

Action: Change the value of the RECYCLE or COMPRESS EXEC parameter to 'YES' (to ensure that the EXEC parameter settings during rerun of the failed database maintenance procedure are the same as those specified during the failed procedure) and restart the job.

**OTS03096 - TAPE ALLOCATION RETRY ENTERED, RETRY LIMIT  
NOW nnnnn**

Explanation: The utility was unable to allocate a device for an output tape volume. Allocation will be retried after 1 minute. 'nnnnn' gives the number of retries remaining. The

utility will terminate if allocation is not successful after this number of retries.

Action: Make a tape device available for allocation by the utility prior to the retry limit being reached. The utility will then continue with normal execution. Otherwise, restart the utility when there are sufficient devices available for execution.

**OTS03097 - NEARARCHIVE DATABASE MAINTENANCE UTILITY  
WILL TERMINATE AFTER PROCESSING CURRENT  
INPUT VOLUME**

Explanation: The database maintenance utility has acknowledged receipt of an operator STOP command for early termination of processing.

Action: Execution of the utility will terminate when all objects on the tape volume being recycled when the command was entered have been processed.

**OTS03098 - UNABLE TO OPEN SYSPRINT DATASET**

Explanation: An error has occurred opening the SYSPRINT dataset.

Action: Refer to other available system messages to discover the cause of the open failure. Correct the error and resubmit the job.

**OTS03099 - INVALID ERROR MESSAGE ID xx ISSUED AT LOCATION  
yyyyyyyy**

Explanation: An unknown numeric message identifier ('xx') has been issued by the utility. 'yyyyyyyy' identifies the internal program location at which the error occurred.

Action: Contact your product support representative. Retain all information relating to the failed execution of the utility.

## 8.6 Database Regression Utility Messages.

---

### **OTS03200 - NEARARCHIVE DATABASE REGRESSION UTILITY COMPLETED WITH CONDITION CODE xx**

Explanation: This informational message is issued when the database regression utility has completed processing. 'xx' is the highest condition code encountered during this execution.

Action: No action necessary.

### **OTS03201 - ERROR X'xxyy' RECEIVED OPENING FILE aaaaaaaaa DATASET bbbb....bbb**

Explanation: A VSAM open error has occurred when opening the identified NearArchive dataset. 'xx' and 'yy' are the hexadecimal error and reason codes returned from the VSAM OPEN macro. 'aaaaaaaa' is the DD name associated with the file, and 'bbb....bbb' is the name of the VSAM dataset. Processing is terminated.

Action: Refer to IBM documentation on VSAM macro processing for details of the return and reason codes. Take the appropriate corrective action and restart the job.

### **OTS03202 - VALID EXEC PARAMETER NOT SPECIFIED. EXEC PARAMETER SHOULD SPECIFY VERSION=22, 23 OR 24**

Explanation: An invalid 'VERSION' parameter value has been specified on the EXEC statement. Processing is terminated.

Action: Refer to NearArchive documentation for EXEC parameter specification. Make the appropriate corrections and restart the job.

### **OTS03204 - UNABLE TO OPEN DATASET WITH DDNAME=ddddddd DCB OPEN FLAG=xx**

Explanation: An open error has occurred when opening the non-VSAM file 'ddddddd'. 'xx' is the DCB flag returned from the OPEN macro. Processing is terminated.

Action: Refer to IBM documentation for details of the OPEN error. Take the appropriate corrective action and restart the job.

**OTS03205 - ERROR X'xxxy' aaaaaaaa bbbb....bbbb IN FILE dddddddd  
eeee....eeee**

Explanation: An error has occurred processing a VSAM file. 'xx' and 'yy' are the hexadecimal error and reason codes returned from the processing request. 'ddddddd' is the DD name associated with the file, and 'eeee....eeee' is the name of the VSAM dataset.

'aaaaaaa' identifies the type of processing being performed when the error occurred. This may take the values 'READING', 'WRITING', 'UPDATING' or 'LOCATING'.

'bbbb....bbbb' identifies the type of record being processed. This may take the values 'PROCESSING CONTROL RECORD', 'STORAGE LEVEL CONTROL RECORD', 'START OF FILE' or 'SEQUENTIALLY'.

Processing is terminated.

Action: Refer to IBM documentation on VSAM macro processing for details of the return and reason codes. Take the appropriate corrective action and rerun the job.

**OTS03206 - INTERNAL ERROR RECEIVED AT LOCATION aaaaaaaa.**

Explanation: A NearArchive internal error has occurred. Processing is terminated.

Action: Retain all information concerning the error, and contact your NearArchive product support representative.

**OTS03207 - INSUFFICIENT STORAGE FOR WORKAREAS.**

Explanation: A request by the utility for main storage allocation has failed.

Action: Increase the REGION size specified for the job and resubmit.



### **OTS03208 - FILE OTASV100 IS NOT A VERSION 2.5 DATABASE**

Explanation: The NearArchive index specified in the DD entry for file OTASV100 is not in version 2.5 format. The NearArchive database regression utility cannot be run on databases which are in a pre-version 2.5 format. Processing is terminated.

Action: Ensure that the correct primary index dataset has been specified in the execution JCL. If necessary, correct the JCL and resubmit the job. If the correct index has been specified, then there is no need to perform regression processing, as the database is already in pre-v2.5 format.

### **OTS03209 - ERROR X'xxxx' INFO X'yyyy' DYNAMICALLY ALLOCATING DATASET zzzz....zzzz**

Explanation: An error has occurred dynamically allocating a dataset, and processing of the utility has been terminated. 'xxxx' and 'yyyy' give the error and information codes respectively, as returned from the SVC99 dynamic allocation request. 'zzzz....zzzz' gives the name of the dataset being allocated.

Action: Refer to IBM documentation on dynamic allocation processing for a description of the error and information codes. Take the appropriate corrective action and rerun the job.

### **OTS03210 – MORE THAN TWO STORAGE LEVELS DEFINED IN DATABASE – REGRESSION CANNOT BE PERFORMED**

Explanation: Additional storage levels (above the maximum of 2 supported by earlier releases of the product) have been defined in the database being regressed. Regression processing cannot be performed on this database.

Action: Restore the database to its state prior to definition of the new storage groups and rerun the regression job. Any changes made to the database since the new storage groups were added will be lost.

**OTS03216 - NEARARCHIVE DATABASE REGRESSION UTILITY HAS BEEN TERMINATED IN RESPONSE TO OPERATOR REQUEST**

Explanation: Execution of the utility has been terminated after input of an operator STOP command.

Action: None. This is an informational message.

**OTS03217 - IDCAMS ERROR, RETURN CODE xx. REVIEW JOBLOG AND/OR OTASIDCO OUTPUT FILE FOR ERROR MESSAGES.**

Explanation: An error has occurred during execution of the utility when deleting, defining or renaming a NearArchive VSAM dataset. 'xx' gives the condition code returned from the internal IDCAMS call.

Action: Refer to the print report issued by IDCAMS in file OTASIDCO for a description of the error. Take the appropriate corrective action and rerun the job.

**OTS03218 - ERROR X'xxxx' INFO X'yyyy' DYNAMICALLY DELETING DATASET zzzz....zzzz**

Explanation: An error has occurred dynamically deleting a dataset, and processing of the utility has been terminated. 'xxxx' and 'yyyy' give the error and information codes respectively, as returned from the SVC99 dynamic allocation/de-allocation request. 'zzzz....zzzz' gives the name of the dataset being allocated.

Action: Refer to IBM documentation on dynamic allocation processing for a description of the error and information codes. Take the appropriate corrective action and rerun the job.

**OTS03297 - NEARARCHIVE DATABASE REGRESSION UTILITY  
TERMINATING IN RESPONSE TO OPERATOR REQUEST**

Explanation: The utility is acknowledging receipt of an operator 'STOP' command for early termination of utility execution.

Action: The utility will checkpoint all current update activity and terminate processing normally. The database regression process may be continued at any subsequent time via normal execution of the database regression utility.

**OTS03298 - UNABLE TO OPEN SYSPRINT DATASET**

Explanation: An error has occurred opening the SYSPRINT dataset.

Action: Refer to other available system messages to discover the cause of the open failure. Correct the error and resubmit the job.

## **8.7 Object Management Utility Messages.**

---

**OTS04000 - NEARARCHIVE OBJECT MANAGEMENT UTILITY  
COMPLETED WITH CONDITION CODE xx**

Explanation: This informational message is issued when the object management utility has completed processing. 'xx' is the highest condition code encountered during program execution.

Action: No action necessary.

**OTS04001 - ERROR X'xxyy' RECEIVED OPENING FILE aaaaaaaa  
DATASET bbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbb**

Explanation: An open error has occurred opening the VSAM file aaaaaaaa. 'bbbbbb...bbbbbb' is the dataset name. 'xx' and 'yy' are the return and reason codes from the VSAM open macro. Processing is terminated.

Action: Refer to IBM documentation for details of the VSAM return and reason codes. Take the appropriate corrective action and restart the job.

#### **OTS04002 - INVALID 'RESTART' SETTING IN EXEC PARAMETERS**

Explanation: The RESTART parameter on the EXEC statement is invalid.

Action: Examine the operand specified for the RESTART parameter on the EXEC statement. Correct the parameter and rerun the job.

#### **OTS04003 - INVALID 'PRINT' SETTING IN EXEC PARAMETERS**

Explanation: The PRINT parameter on the EXEC statement is invalid.

Action: Examine the operand specified for the PRINT parameter on the EXEC statement. Correct the parameter and rerun the job.

#### **OTS04004 - INVALID EXEC PARAMETER FOUND**

Explanation: An unidentifiable parameter has been specified on the EXEC statement.

Action: Examine the parameters specified on the EXEC statement. Correct the parameters and rerun the job.

#### **OTS04005 - ERROR X'xxyy' aaaaaaaaa 'bbbbbbbbbbbbbbbbbbbb' IN FILE cccccc ddddddddddddddddddddddddddddddd**

Explanation: An error has occurred when processing a VSAM dataset. Some or all of the following operands may appear:-

'xx' - is the return code from the VSAM I/O macro

'yy' - is the reason code from the VSAM I/O macro

'aaaaaaaa' - is the process being performed. This may take the following values:-

'READING'  
'WRITING'  
'UPDATING'  
'DELETING'

'bbbb...bbbb' - is a description of the record being accessed.

'cccccccc' - is the file name.

'dddd...dddd' - is the dataset name.

Processing is terminated.

Action: Refer to IBM documentation for details of the VSAM return and reason codes. Take the appropriate corrective action and restart the job.

#### **OTS04006 - INTERNAL ERROR RECEIVED AT LOCATION yyyyyyyy**

Explanation: An internal error has occurred during processing of the utility.

Action: Retain all information relating to utility execution and contact your NearArchive product support representative.

#### **OTS04007 - INSUFFICIENT STORAGE FOR WORKAREAS**

Explanation: A request for virtual storage in the job's private area has been denied by the operating system.

Action: Increase the REGION size allocated for the job, or jobstep, and resubmit the job.

#### **OTS04008 - FILE OTASV100 IS NOT AT THE CORRECT RELEASE LEVEL**

Explanation: A pre-version 2.5 database is being processed by the utility. The NearArchive database maintenance utility cannot be run on databases prior to version 2.5. Processing is terminated.

Action: Ensure that the correct primary index dataset is specified under DD entry OTASV100 in the execution JCL. Correct and resubmit the job.

**OTS04009 - ERROR X'xxxx' INFO X'yyyy' DYNAMICALLY  
ALLOCATING DATASET aaaaaaaaaaaaaaaaaaaaaaaaaaaaaa**

Explanation: A dynamic allocation error has occurred during allocation of the identified dataset. Processing is terminated.

Action: Refer to IBM documentation on SVC99 processing for a description of the error and information codes. Take the appropriate corrective action and restart the job.

**OTS04010 - NORMAL RUN REQUESTED BUT PREVIOUS RUN  
TERMINATED ABNORMALLY.**

Explanation: A previous attempt to run this utility has failed abnormally but RESTART=YES has not been specified for this run.

Action: Examine the output from the previous run to determine the cause of failure. Take the appropriate corrective action and restart the job with RESTART=YES specified on the EXEC statement.

**OTS04012 - ERROR LOADING PROGRAM OTASP045**

Explanation: An error has occurred when loading program OTASP045 the migration component of the object management utility. Processing is terminated.

Action: Check that module OTASP045 is present on the NearArchive load library. Contact your NearArchive product support representative for assistance if necessary.

**OTS04013 - ERROR 'xx' RECEIVED ACCESSING JOB/USER NAME**

Explanation: An error has occurred when retrieving the JOB name and USER name.

Action: This is an internal processing error. Retain all information regarding execution of the utility, and contact your NearArchive product support representative.

#### **OTS04014 - INVALID 'DATE' SETTING IN EXEC PARAMETERS**

Explanation: The DATE parameter specified in the job execution parameters is not correctly specified.

Action: Examine the operand specified for the DATE parameter on the EXEC statement. Correct the parameter and rerun the job.

#### **OTS04015 - 'DATE' PARAMETER MUST FOLLOW 'SIMULATE' PARAMETER**

Explanation: A DATE parameter has been specified in the job execution parameters, but no SIMULATE parameter has been specified. This is an invalid combination of execution parameters.

Action: Verify whether the utility is to be executed in simulation mode or not. If it is, add a SIMULATE parameter to the job execution parameter. If not, remove the DATE parameter.

#### **OTS04016 - NEARARCHIVE OBJECT MANAGEMENT UTILITY HAS BEEN TERMINATED IN RESPONSE TO OPERATOR REQUEST**

Explanation: This message is issued prior to completion of object management processing after execution of the utility has been terminated early via an operator STOP command.

Action: None. This is an information message.

#### **OTS04017 - INVALID VALUE SPECIFIED ON THE 'NEWTAPE' PARAMETER**

Explanation: The NEWTAPE parameter on the EXEC statement is invalid. The parameter value must be set to 'YES' or 'NO'.

Action: Examine the operand specified for the NEWTAPE parameter on the EXEC statement. Correct the parameter and rerun the job.

**OTS04018 - ERROR X'xxxx' INFO X'yyyy' DYNAMICALLY DELETING  
DATASET aaaaaaaaaaaaaaaaaaaaaaaaaaaaaa**

Explanation: A dynamic allocation error has occurred during deletion of the identified 'K' copy dataset. Processing will continue.

Action: Refer to IBM documentation on SVC99 processing for a description of the error and information codes. Take any appropriate corrective action to ensure the error does not re-occur.

The 'K' copy dataset identified in the message has been released from the database, but was not deleted by the utility. This dataset should be manually deleted after the reason for the error has been determined.

**OTS04019 – ASYNCHRONOUS UPDATE FOR SECONDARY INDEX nn  
STILL OUTSTANDING**

Explanation: An asynchronous update has been scheduled for secondary index 'nn' during batch archival processing, but the secondary index update utility OTASP080 has not been executed for that secondary index. Execution of the utility is terminated.

Action: Run the secondary index update utility OTASP080 using the asynchronous update dataset created during batch archival processing. Refer to chapter 7 for details of the asynchronous secondary index update process. The object management utility may be rerun following successful completion of this procedure.

**OTS04020 – INVALID VALUE SPECIFIED ON THE INLEVEL|  
OUTLEVEL PARAMETER OR STORAGE LEVEL DOES  
NOT EXIT**

Explanation: The storage level identifier specified in the INLEVEL/OUTLEVEL execution parameter does not correspond to a defined storage level.

Action: Examine the operand specified for the INLEVEL/OUTLEVEL parameter on the EXEC statement. Correct the parameter and rerun the job.



**OTS04021 – BOTH ‘INLEVEL’ AND ‘OUTLEVEL’ PARAMETERS  
MUST BE SPECIFIED OR BOTH OMITTED**

Explanation: One, but not both, of the INLEVEL or OUTLEVEL EXEC parameters has been specified. Both parameters must either be specified or both omitted.

Action: Correct the EXEC parameters and rerun the job.

**OTS04022 – THE ‘INLEVEL’ AND ‘OUTLEVEL’ PARAMETERS MUST  
NOT BE MODIFIED ON A RESTART RUN**

Explanation: A restart run is taking place, but the value of the INLEVEL or OUTLEVEL parameters specified on the EXEC statement or not the same as those specified for the original failing run.

Action: Ensure that the INLEVEL and OUTLEVEL parameter specifications on the EXEC statement are identical to those specified on the failing run. Correct the parameter(s) and rerun the job.

**OTS04097 - NEARARCHIVE OBJECT MANAGEMENT UTILITY  
TERMINATING IN RESPONSE TO OPERATOR REQUEST**

Explanation: The utility is acknowledging receipt of an operator ‘STOP’ command for early termination of utility execution.

Action: The utility will checkpoint all current update activity and terminate processing normally. The object management process may be continued at any subsequent time via normal execution of the object management utility.

**OTS04098 - UNABLE TO OPEN SYSPRINT DATASET**

Explanation: An error has occurred opening the SYSPRINT dataset.

Action: Refer to other available system messages to discover the cause of the open failure. Correct the error and resubmit the job.

**OTS04501 - ERROR X'xxyy' RECEIVED OPENING FILE aaaaaaaa  
DATASET bbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbb**

Explanation: An error has occurred opening the VSAM dataset aaaaaaaa. 'bbbb...bbbb' is the dataset name. 'xx' and 'yy' are the return and reason codes from the VSAM open macro. Processing is terminated.

Action: Refer to IBM documentation for details of the VSAM return and reason codes. Take the appropriate corrective action and restart the job.

**OTS04505 - ERROR X'xxyy' aaaaaaaa 'bbbbbbbbbbbbbbbbbbbb' IN  
FILE ccccccc ddddddddddddddddddddddddddddddd**

Explanation: An error has occurred when processing a VSAM dataset. Some or all of the following operands may appear:-

'xx' - is the return code from the VSAM I/O macro

'yy' - is the reason code from the VSAM I/O macro

'aaaaaaa' - is the process being performed. This may take the following values:-

'READING'  
'WRITING'  
'UPDATING'  
'DELETING'

'bbbb...bbbb' - is a description of the record being accessed.

'ccccccc' - is the file name.

'dddd...dddd' - is the dataset name.

Processing is terminated.

Action: Refer to IBM documentation for details of the VSAM return and reason codes. Take the appropriate corrective action and restart the job.

**OTS04507 - INSUFFICIENT STORAGE FOR WORKAREAS**

Explanation: A request for virtual storage in the job's private area has been denied by the operating system.

Action: Increase the REGION size allocated for the job, or jobstep, and resubmit the job.

**OTS04509 - ERROR X'xxxx' INFO X'yyyy' DYNAMICALLY  
ALLOCATING DATASET aaaaaaaaaaaaaaaaaaaaaa**

Explanation: A dynamic allocation error has occurred during allocation of the identified dataset. Processing is terminated.

Action: Refer to IBM documentation on SVC99 processing for a description of the error and information codes. Take the appropriate corrective action and restart the job.

**OTS04510 - NORMAL RUN REQUESTED BUT PREVIOUS RUN  
TERMINATED ABNORMALLY.**

Explanation: A previous attempt to run this utility has failed abnormally but RESTART=YES has not been specified for this run.

Action: Examine the output from the previous run to determine the cause of failure. Take the appropriate corrective action and restart the job with RESTART=YES specified on the EXEC statement.

**OTS04512 - ERROR LOADING PROGRAM OTASP352**

Explanation: An error has occurred when loading the tape I/O driver program OTASP352. Processing is terminated.

Action: Check that module OTASP352 is present in the NearArchive product load library. Contact your NearArchive product support representative for assistance if necessary.

**OTS04513 - CONDITION CODE xx RETURNED FROM INTERNAL  
SORT - EXAMINE SYSOUT DATASET FOR FURTHER  
INFORMATION**

Explanation: An error has occurred during internal sort processing. Processing is terminated.

Action: Refer to the documentation for the sort product in use on the host system for further information on the error. Take the appropriate corrective action and restart the job.

**OTS04514 - ERROR OPENING SORT DATASET OTASS045 - REFER TO JOBLOG FOR FURTHER INFORMATION**

Explanation: An error has occurred when opening the sort output dataset. Processing is terminated.

Action: Refer to the job log for a description of the error. Take the appropriate corrective action and restart the job.

**OTS04515 - RELOCATION ERROR ON OTASS045 DURING RESTART PROCESSING**

Explanation: An error has occurred during a RESTART run. The contents of the migration control dataset file are invalid. Processing is terminated.

Action: Determine, if possible, the reason why the contents of the migration control file are invalid. This dataset will have the name

'hlq'.MIGCNTL

where 'hlq' is the high-level qualifier defined for the database being processed.

Contact your NearArchive product support representative if there is no apparent reason for the error.

**OTS04516 - INTERNAL ERROR AT LOCATION xxxxxxxx ERROR CODE = X'yyyyyyyyy'**

Explanation: An internal error has occurred during execution of the utility.

Action: Retain all information regarding execution of the utility, and contact your NearArchive product support representative.

**OTS04517 - RETURN CODE 'aa' SUPP CODES X'bbbbcccc' RETURNED  
FROM OTASP352 REQUEST CODE 'dd' AT LOCATION  
eeeeeeee**

Explanation: An error has occurred during internal NearArchive processing when retrieving an object from tape during migration processing.

'aa', 'bbbb' and 'cccc' are the return and supplementary codes from the internal call. 'dd' is the request code.

Processing is terminated.

Action: Refer to NearArchive documentation for details of the request and return codes. Take the appropriate corrective action and restart the job.

If the cause of the error is not obvious, contact your NearArchive product support representative.

**OTS04518 - RETURN CODE 'aa' SUPP CODES X'bbbbcccc' RETURNED  
FROM NEARARCHIVE REQUEST CODE 'dd' AT  
LOCATION eeeeeeee**

Explanation: An error has occurred during internal NearArchive processing when writing an object to tape during migration processing. 'aa', 'bbbb', 'cccc' and 'dd' are the return, supplementary and request codes from the NearArchive call. Processing is terminated.

Action: Refer to NearArchive documentation for details of the request and return codes. Take the appropriate corrective action and restart the job. If the cause of the error is not obvious, contact your NearArchive product support representative.

**OTS04519 - DATABASE IN UNCOMMITTED STATE – {OBJECT  
ARCHIVAL|DATABASE MAINTENANCE} PROCESS  
INCOMPLETE**

Explanation: An object archival or database maintenance process (as identified in the message) has been executed on the NearArchive database being processed by the utility, but

this process has not completed successfully. Execution of the utility cannot be performed in these circumstances.

Action: Ensure that the outstanding archival or maintenance process is completed successfully before rerunning the object maintenance utility.

#### **OTS04597 - NEARARCHIVE OBJECT MANAGEMENT UTILITY TERMINATING IN RESPONSE TO OPERATOR REQUEST**

Explanation: The utility is acknowledging receipt of an operator 'STOP' command for early termination of utility execution.

Action: The utility will checkpoint all current update activity and terminate processing normally. The object management process may be continued at any subsequent time via normal execution of the object management utility.

## **8.8 Journal Maintenance Utility Messages.**

---

#### **OTS07000 - UNABLE TO OPEN SYSPRINT DATASET**

Explanation: An error has occurred opening the SYSPRINT dataset.

Action: Consult other available system messages for additional information on the error. Make the appropriate corrections, and resubmit the utility.

#### **OTS07001 - DD CARD MISSING IN JCL FOR OTASV100**

Explanation: No OTASV100 DD card was supplied in the JCL used to execute the utility. This DD card is required in order to identify the primary index of the database to be processed by the journal maintenance utility.

Action: Add a DD card for file OTASV100 to the JCL and rerun the job. Refer to chapter 7 for a description of JCL requirements for the journal maintenance utility.

**OTS07002 - OPEN FAILED FOR aaaaaaaa RETURN CODE=X'xx'  
REASON CODE=X'yy'**

Explanation: A error has occurred when opening the VSAM dataset aaaaaaaa. 'xx' and 'yy' are the hexadecimal return and reason codes from the VSAM open macro. Processing is terminated.

Action: Refer to IBM documentation on VSAM macro processing for details of the return and reason codes. Take the appropriate corrective action and restart the job.

**OTS07003 - UNABLE TO GETMAIN STORAGE FOR aaaaaaaa BUFFER.  
INCREASE JOB REGION SIZE AND RESTART THE  
UTILITY. PROGRAM LOCATION bbbb**

Explanation: A request for data buffer storage in the job's private area has been denied by the operating system. 'aaaaaaa' identifies the data buffer.

Action: Increase the REGION size allocated for the job or jobstep, and resubmit the job.

**OTS07004 - I/O ERROR READING PCR - FDBK= X'xxxxyyyy'**

Explanation: An error has occurred when reading the NearArchive processing control record from dataset OTASV100. 'xxxx' and 'yyyy' are the hexadecimal return and reason codes from the VSAM GET macro. Processing is terminated.

Action: Refer to IBM documentation on VSAM macro processing for details of the return and reason codes. Take the appropriate corrective action and restart the job. If no apparent cause of the error can be identified, contact your NearArchive product support representative.

**OTS07005 - NEARARCHIVE DATABASE IS NOT AT THE CORRECT  
LEVEL**

Explanation: The database index supplied in the OTASV100 DD entry is a pre-version 2.5 index. The NearArchive Journal maintenance utility cannot process pre-version 2.5 databases. Processing is terminated.

Action: Pre-version 2.5 databases must be converted for use with this release of the product. Database conversion is performed via the database administration process in TSO/ISPF. Refer to chapter 3 of this manual for information on the database conversion process.

**OTS07006 - ERROR DURING VALIDATION OF THE EXEC  
PARAMETER 'DATE'**

Explanation: An invalid 'DATE' parameter has been encountered in the EXEC parameters. Processing of the utility is terminated.

Action: Ensure that a correct value for the 'DATE' parameter is supplied in the EXEC parameters and rerun the job.

**OTS07007 - DATABASE OPEN CALL TO OTASP100 FAILED.  
RETURN CODE=aa SUPP CODE1=bbbb SUPP CODE2=cccc  
REQ=dd DBN=eeee**

Explanation: An internal NearArchive call has failed. 'aa' is the return code, 'bbbb' and 'cccc' are the supplementary codes returned from NearArchive. 'dd' is the request code and 'eeee' is the database name.

Action: Refer to section 8.2 of this manual for details of the return and supplementary codes. Take the appropriate corrective action and restart the job.

**OTS07008 - ERROR READING JOURNAL FILE RETURN CODE=X'xx'  
REASON CODE=X'yy'**

Explanation: An error has occurred when reading the NearArchive database's journal dataset. 'xx' and 'yy' are the hexadecimal return and reason codes from the VSAM GET macro. Processing is terminated.

Action: Refer to IBM documentation for details of the return and reason codes. Take the appropriate corrective action and restart the job.



**OTS07009 - OBJECT RETRIEVE REQUEST FAILED; RETURN  
CODE=aa SUPP CODE1=bbbb SUPP CODE2=cccc REQ=dd  
DATE=yy/mm/dd**

Explanation: An internal NearArchive object retrieval request has failed. 'aa' is the return code, 'bbbb' and 'cccc' are the hexadecimal supplementary codes returned from NearArchive. 'dd' is the request code. 'yy/mm/dd' is the archive date of the NearArchive object.

Action: Refer to section 8.2 of this manual for details of the return and supplementary codes. Take the appropriate corrective action and restart the job.

**OTS07010 - WRITE DATA RECORD CALL TO OTASP100  
FAILED.RETURN CODE=aa SUPP CODE1=bbbb SUPP  
CODE2=cccc REQ=dd DATE=yy/mm/dd**

Explanation: An internal NearArchive write data record request has failed. 'aa' is the return code, 'bbbb' and 'cccc' are the supplementary codes returned from NearArchive. 'dd' is the request code. 'yy/mm/dd' is the object's archive date.

Action: Refer to section 8.2 of this manual for details of the return and supplementary codes. Take the appropriate corrective action and restart the job.

**OTS07011 - DATABASE CLOSE CALL TO OTASP100 FAILED.  
RETURN CODE=aa SUPP CODE1=bbbb SUPP CODE2=cccc  
REQ=dd**

Explanation: An NearArchive database close call has failed. 'aa' is the return code, 'bbbb' and 'cccc' are the supplementary codes returned from NearArchive. 'dd' is the request code.

Action: Refer to section 8.2 of this manual for details of the return and supplementary codes. Take the appropriate corrective action and restart the job.

**OTS07012 - OPEN FAILED WITH ERROR CODE 'xxxxyyyy' FOR  
DATASET aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa**

Explanation: An open error has occurred for the identified VSAM dataset. 'xxxx' and 'yyyy' are the hexadecimal codes

returned from the VSAM open macro. Processing is terminated.

Action: Refer to IBM documentation for details of the return and reason codes. Take the appropriate corrective action and restart the job.

#### **OTS07013 - INVALID EXEC PARM SPECIFIED - RUN ABANDONED**

Explanation: An unidentifiable parameter has been specified on the EXEC statement.

Action: Examine the parameters specified on the EXEC statement. Correct the parameters and restart the job.

#### **OTS07014 - JOURNAL RECORD FOUND WITH INVALID OPERATION TYPE**

Explanation: An invalid record has been found on the NearArchive journal dataset.

Action: Check the contents of the database's journal dataset for invalid entries. This dataset will have the name

'hlq'.JOURNAL

where 'hlq' is the high-level qualifier defined for the NearArchive database being processed by the utility.

If no cause of the error can be identified, contact your NearArchive product support representative.

#### **OTS07015 - NO RECORDS RETRIEVED FOR SPECIFIED DATE(S)**

Explanation: There were no journal entries in the database's journal archive database, for the specified date range.

Action: Amend the dates if necessary and resubmit the job.

**OTS07016 - RESTART=NO SPECIFIED BUT PREVIOUS RUN DID NOT COMPLETE NORMALLY.**

Explanation: A previous attempt to run this utility has failed abnormally but RESTART=YES has not been specified for this run.

Action: Examine the output from the previous run to determine the cause of failure. Take the appropriate corrective action and restart the job with RESTART=YES specified on the EXEC statement.

**OTS07017 - NO RECORDS FOUND ON JOURNAL FILE**

Explanation: The journal dataset is empty.

Action: This is an information message. No action is necessary.

**OTS07018 - 'DATE TO' IS LESS THAN 'DATE FROM' IN THE EXEC PARAMETER**

Explanation: An error has been detected in the 'DATE=' EXEC parameter. The first date in the range is lower than the last date in the range. Processing is terminated.

Action: Correct the invalid execution parameter, and rerun the job.

**OTS07019 - INVALID EXEC PARM COMBINATION - 'DATE - NODUPS' SPECIFIED – RUN ABANDONED**

Explanation: The DATE and NODUPS execution parameters have both been specified. This is an invalid combination. Processing of the utility is terminated with a parameter validation error.

Action: Refer to chapter 7 for a description of execution parameter specification. Correct the parameter and re-submit the job.

**OTS07020 - ERROR X'xxyy' aaaaaaaa bbbb....bbbb IN FILE cccccccc**

Explanation: An error has occurred processing VSAM file 'ccccccc'. 'xx' and 'yy' give the hexadecimal error and reason



OTASP030) is  
incomplete

OBJECT MIGRATION an object migration  
process (utility  
OTASP040) is  
incomplete

OBJECT MANAGEMENT an object management  
process (utility  
OTASP040) is  
incomplete.

Processing of the utility is terminated.

Action: The journal utility cannot be used to process a database which is an uncommitted state. Ensure that the incomplete update operation (as identified in this message) has been completed successfully before re-running the journal maintenance utility.

**OTS07024 - DYNALLOC ERROR, REQ=aa RC=bb DD=ccccccc**  
**DSN=dddddddddddddddddddddddddddddddddddd**  
**VOL=vvvvvv UNIT=uuuuuuuu**

**OTS07025 - INF=eeee ERR=ffff SYSC=g SYSD=hhhhh DEFER=ii**  
**DISP=jjkkll SPACE=(pppppp,sssss)**  
**DCB=(mmmmmm,nnnnnn,oo)**

Explanation: An error has occurred during dynamic allocation of a dataset. Messages OTS07024 and OTS07025 are both displayed to assist with problem determination. Some or all of the following operands may appear:

'ccccccc' - is the DDname of the dataset to be allocated

'ddd...ddd' - is the name of the dataset to be allocated.

'eeee' - is the SVC99 information code

'ffff' - is the SVC99 error code

Processing is terminated.

Action: Refer to IBM documentation on SVC99 processing for a description of the error and information codes. Take the appropriate corrective action and restart the job.

**OTS07026 - CONDITION CODE nn RETURNED FROM INTERNAL  
SORT - EXAMINE SYSOUT DATASET FOR FURTHER  
INFORMATION**

Explanation: An internal sort request issued by the utility has failed with return code 'nn'. Execution of the utility is terminated.

Action: Refer to the sort processing report in the SYSOUT dataset for a description of the error condition. Take the appropriate corrective action and restart the utility.

**OTS07027 - ERROR OPENING SORT DATASET xxxxxxxx - REFER TO  
JOBLOG FOR FURTHER INFORMATION**

Explanation: An error has occurred opening the sequential sort work file 'xxxxxxx'. Processing of the utility is terminated.

Action: Refer to other available system information for additional information on the error condition which caused the failure. Take the appropriate corrective action and restart the utility.

**OTS07040 - NEARARCHIVE JOURNAL MAINTENANCE UTILITY  
COMPLETED WITH RETURN CODE = xx**

Explanation: This informational message is issued when the journal maintenance utility has completed processing. 'xx' is the highest condition code encountered during program execution.

Action: No action necessary.

## **8.9 Secondary Index Update Utility Messages.**

---

### **OTS08000 - NEARARCHIVE SECONDARY INDEX UPDATE UTILITY COMPLETED WITH CONDITION CODE xx**

Explanation: This informational message is issued when the secondary index update utility has completed processing. 'xx' is the highest condition code encountered during program execution.

Action: No action necessary.

### **OTS08001 - ERROR X'xxyy' RECEIVED OPENING FILE aaaaaaaaa DATASET bbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbb**

Explanation: An open error has occurred opening the VSAM file aaaaaaaaa. 'bbbb...bbbb' is the dataset name. 'xx' and 'yy' are the return and reason codes from the VSAM open macro. Processing is terminated.

Action: Refer to IBM documentation for details of the VSAM return and reason codes. Take the appropriate corrective action and rerun the job.

### **OTS08002 - CONDITION CODE xx RETURN FROM INTERNAL SORT – EXAMINE SYSOUT DATASET FOR FURTHER INFORMATION**

Explanation: An error condition has occurred during internal sort processing. 'xx' gives the condition code returned from the sort program.

Action: Refer to the sort report in the SYSOUT dataset for information on the cause of the error. Make the appropriate corrections and rerun the job.

**OTS08003 - ERROR X'xxxx' INFO 'yyyy' ACCESSING zzzzzzzz –  
DDNAME MISSING OR IN ERROR**

Explanation: An error has occurred querying the allocation of file 'zzzzzzzz'. 'xxxx' and 'yyyy' give the hexadecimal error and information codes returned from an SVC99 'query' request. Execution of the utility is terminated with condition code 12.

Action: Verify that file 'zzzzzzzz' has been allocated correctly in the JCL used to execute the utility. If it has, refer to IBM documentation on SVC99 processing for an explanation of the error and information codes displayed in the message, and take the appropriate action.

**OTS08004 - ERROR X'xxxx' BUILDING VSAM LSR POOL**

Explanation: An error has occurred building a VSAM local shared resources (LSR) buffer pool at start of execution. 'xxxx' gives the hexadecimal error returned from the LSR build request. Execution of the utility is terminated with condition code 12.

Action: Refer to IBM documentation on VSAM macro processing for a description of the error. Take the appropriate corrective action and rerun the job.

**OTS08005 - ERROR X'xxyy' aaaaaaaa 'bbbbbbbbbbbbbbbbbbbb' IN  
FILE ccccccc ddddddddddddddddddddddddddddddd**

Explanation: An error has occurred when processing a VSAM dataset. Some or all of the following operands may appear:

'xx' - is the return code from the VSAM I/O macro

'yy' - is the reason code from the VSAM I/O macro

'aaaaaaa' - is the process being performed. This may take the following values:-

'READING'

'WRITING'

'UPDATING'

'DELETING'



'bbbb...bbbb' - is a description of the record being accessed.

'ccccccc' - is the file name.

'dddd...dddd' - is the dataset name.

Processing is terminated.

Action: Refer to IBM documentation for details of the VSAM return and reason codes. Take the appropriate corrective action and rerun the job.

#### **OTS08006 - INTERNAL ERROR RECEIVED AT LOCATION yyyyyyyy**

Explanation: An internal error has occurred during processing of the utility.

Action: Retain all information relating to utility execution and contact your NearArchive product support representative.

#### **OTS08007 - INSUFFICIENT STORAGE FOR WORKAREAS**

Explanation: A request for virtual storage in the job's private area has been denied by the operating system.

Action: Increase the REGION size allocated for the job, or jobstep, and resubmit the job.

#### **OTS08008 - FILE OTASV100 IS NOT AT THE CORRECT RELEASE LEVEL**

Explanation: A pre-version 2.5 database is being processed by the utility. The NearArchive secondary index update utility cannot be run on databases prior to version 2.5. Processing is terminated.

Action: Ensure that the correct primary index dataset is specified under DD entry OTASV100 in the execution JCL. Correct and resubmit the job.

**OTS08009 - ERROR X'xxxx' INFO X'yyyy' DYNAMICALLY  
ALLOCATING FILE aaaaaaaa TO DATASET bbbb....bbbb**

Explanation: A dynamic allocation error has occurred during allocation of the identified dataset. Processing is terminated.

Action: Refer to IBM documentation on SVC99 processing for a description of the error and information codes. Take the appropriate corrective action and rerun the job.

**OTS08010 - NEARARCHIVE DATABASE IS IN UNCOMMITTED STATE  
– PROCESSING TERMINATED**

Explanation: The NearArchive database being processed by the utility is currently in an uncommitted state (i.e.) a database update job has terminated abnormally and has not yet been rerun. The secondary index update utility cannot process a database with this status. Processing of the utility is terminated with condition code 12.

Action: Ensure that the outstanding database update job is run to successful conclusion before resubmitting the secondary index update job.

**OTS08011 - EMPTY/INVALID SECONDARY INDEX UPDATE FILE  
OTASS080 ENCOUNTERED**

Explanation: The dataset allocated to file OTASS080 in the JCL is either empty or does not contain valid header information. Execution of the utility is terminated with condition code 12.

Action: Ensure that the correct dataset name has been specified in the DD entry for file OTASS080 in the JCL. Correct the DD entry and rerun the job.

If the DD entry is correct, ensure that the corresponding phase 1 batch archival procedure has been successfully run prior to running the secondary index update utility.

**OTS08012 - SECONDARY INDEX NUMBER nn NOT DEFINED IN  
NEARARCHIVE DATABASE**

Explanation: An asynchronous update to secondary index number 'nn' has been identified from the header information in file OTASS080, but this secondary index is not defined to the database whose primary index is specified in DD entry OTASV100. Execution of the utility is terminated with condition code 12.

Action: Ensure that the correct dataset names have been specified in the OTASV100 and OTASS080 DD entries in the JCL, and rerun the utility.

**OTS08013 - NO UPDATE OUTSTANDING FOR SECONDARY  
INDEX NUMBER nn**

Explanation: An asynchronous update to secondary index number 'nn' has been identified from the header information in file OTASS080, but there is no outstanding update scheduled for that secondary index. Execution of the utility is terminated with condition code 12.

Action: Ensure that the correct dataset names have been specified in the OTASV100 and OTASS080 DD entries in the JCL, and rerun the utility.

**OTS08019 - UNABLE TO OPEN xxxxxxxx DATASET**

Explanation: An error has occurred attempting to open the sequential dataset with filename 'xxxxxxx'. Execution of the utility is terminated with condition code 12.

Action: Refer to other available system messages for more information on the cause of the error. Make the appropriate corrections and rerun the job.

#### **OTS08098 - UNABLE TO OPEN SYSPRINT DATASET**

Explanation: An error has occurred opening the SYSPRINT dataset.

Action: Consult other available system messages for additional information on the error. Make the appropriate corrections, and rerun the utility.

### **8.10 Database Backup Utility Messages.**

---

#### **OTS11000 - NEARARCHIVE BACKUP UTILITY COMPLETED WITH RETURN CODE xxx**

Explanation: This informational message is issued when the database backup utility has completed processing. 'xx' is the highest condition code encountered during program execution.

Action: No action necessary.

#### **OTS11001 - OPEN FAILED FOR OTASV100. RETURN CODE= X'xx' ERROR CODE= X'yyyyyyyy'**

Explanation: An error has occurred when opening the NearArchive primary index OTASV100. 'xx' and 'yyyyyyyy' are the hexadecimal return and reason codes from the VSAM OPEN macro. Processing is terminated.

Action: Refer to IBM documentation on VSAM macro processing for details of the return and reason codes. Take the appropriate corrective action and restart the job.

#### **OTS11002 - UNABLE TO GETMAIN STORAGE FOR DATA BUFFER.**

Explanation: A request for data buffer storage in the job's private area has been denied by the operating system.

Action: Increase the REGION size allocated for the job or jobstep, and resubmit the job.

**OTS11003 - UNABLE TO READ aaa. GET RETURN CODE = X'xx' GET  
FEEDBACK = X'yyyyyyyy'**

Explanation: A VSAM error has occurred when reading from the NearArchive primary index. 'aaa' identifies the record being read. 'xx' and 'yyyyyyyy' are the hexadecimal return and reason codes from the VSAM GET macro. Processing is terminated.

Action: Refer to IBM documentation on VSAM macro processing for details of the return and reason codes. Take the appropriate corrective action and restart the job. If no apparent cause of the error can be identified, contact your NearArchive product support representative.

**OTS11004 - UNABLE TO OPEN DATASET WITH DDNAME=aaaaaaaa  
RETURN CODE = X'xx'**

Explanation: A open error has occurred when opening a NearArchive tape dataset during tape backup processing. aaaaaaaa is the DDname of the file being opened and may contain the following:

- OTASBKUP - Input control dataset
- OTAST100 - Primary archive volume
- OTAST200 - 'B' copy of primary volume
- OTAST201 - 'D' copy of primary volume -duplexing specified.

'xx' is the return code returned from the OPEN macro. Processing is terminated.

Action: Refer to IBM documentation for details of the OPEN error. Take the appropriate corrective action and restart the job.

**OTS11005 - DATASET NAME - aaaaaaaaaaaaaaaaaaaaaaaaaaaaaa - IS  
TOO LONG**

Explanation: The name of a NearArchive tape dataset in file OTASBKUP exceeds 41 characters. Processing is terminated.

Action: The NearArchive backup utility cannot process primary tape datasets which have more than 41 characters in the

dataset name.

This error should not occur during normal NearArchive processing. Check that the correct OTASBKUP dataset has been supplied in the JCL used to run the utility.

If no apparent cause of the error can be detected, contact your NearArchive product support representative.

**OTS11006 – OTASV100 IS A PRE-VERSION 2.5 DATABASE –  
PROCESSING IS TERMINATED**

Explanation: The database whose primary index is specified in the OTASV100 DD JCL statement has not been converted for use with version 2.5 of NearArchive. The utility cannot process this database.

Action: Ensure that the correct primary index name has been specified in the OTASV100 DD JCL statement. If it is, ensure that the database has been successfully converted for use with version 2.5 of the product before rerunning the job. Refer to section 3.5 for information on database conversion processing.

**OTS11009 - READ ERROR, FILE=aaaaaaaa RETURN CODE=X'xx'  
REASON CODE=X'yyyyyyyy'**

Explanation: An error has occurred reading a main index entry record from the NearArchive database's primary index dataset. 'xx' and 'yyyyyyyy' are the hexadecimal return and reason codes respectively from the VSAM GET macro. Processing is terminated.

Action: Refer to IBM VSAM documentation on VSAM macro processing for details of the return and reason codes. Take the appropriate corrective action and restart the job.

**OTS11010 - WRITE ERROR, FILE=aaaaaaaa RETURN CODE=X'xx'  
REASON CODE=X'yyyyyyyy'**

Explanation: An error has occurred writing a main index entry record to the NearArchive database's primary index dataset. 'xx' and 'yyyyyyyy' are the hexadecimal return and reason

codes respectively from the VSAM PUT macro.  
Processing is terminated.

Action: Refer to IBM VSAM documentation on VSAM macro processing for details of the return and reason codes. Take the appropriate corrective action and restart the job.

**OTS11011 - KEY=bbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbb**

Explanation: This message accompanies message OTS11010 to identify the primary key of the object being processed when the error occurred. Only the first 60 characters of the primary key are printed for databases whose primary keylength exceeds this value.

Action: See message OTS11010.

**OTS11014 – aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa bbbbbb  
cccccccccccccccccccccccccccccccc**

Explanation: This informational message is printed on completion of processing for each tape volume.

aaaaa...aaaaa - the name of the tape dataset being processed.

Bbbbbbb - the tape volume serial number.

cccccc...cccccc - the action taken. This may contain the following values:-

'\*\*\* OLD BACKUP DATASET - UNCATALOGED \*\*\*'

'\*\*\* OLD DUPLEX DATASET - UNCATALOGED \*\*\*'

'NEW BACKUP DATASET - CATALOGED '

'NEW DUPLEX DATASET - CATALOGED '

Action: No action necessary.

**OTS11019 - TAPE DATASET SPECIFIED IN OTASBKUP IS NOT A  
PRIMARY COPY. TAPE DSN = xxxxxxxxxxxxxxxxxxxxxxxxx**

Explanation: A tape dataset name specified in the optional OTASBKUP dataset is not that of a primary tape dataset

copy for the NearArchive database being processed. The invalid entry will be ignored for backup processing.

Action: Check that the entry specified in the OTASBKUP dataset specifies a valid primary copy dataset name. Refer to section 4.7 for a description of NearArchive tape dataset naming conventions. Rerun the utility in order to backup the specified primary dataset copy.

#### **OTS11020 - OTASBKUP DATASET IS EMPTY**

Explanation: This is a warning message indicating that the optional input control dataset OTASBKUP is empty. No tape datasets will be backed up.

Action: Verify that no tape datasets were created or updated during the preceding NearArchive batch archival procedure.

#### **OTS11021 - UNABLE TO READ SLC FOR STORLVL= aaa. GETSLC RETURN CODE = X'xx' GETSLC FEEDBACK = X'yyyyyyyy'**

Explanation: An error has occurred when reading the storage level control record from the NearArchive database's primary index dataset.

'aaa' is the storage level being read.

'xx' and 'yyyyyyyy' are the hexadecimal return and reason codes respectively from the VSAM GET macro.

Processing is terminated.

Action: Refer to IBM documentation on VSAM macro processing for details of the return and reason codes. Take the appropriate corrective action and restart the job. If no cause of the error can be identified, contact your NearArchive product support representative.

#### **OTS11022 - UNABLE TO UPDATE SLC FOR STORLVL= aaa. RETURN CODE = X'xx' FEEDBACK = X'yyyyyyyy'**

Explanation: An error has occurred when updating the storage level control record in the NearArchive database's primary



index dataset.

'aaa' is the storage level being read.

'xx' and 'yyyyyyy' are the hexadecimal return and reason codes respectively from the VSAM GET macro.

Processing is terminated.

Action: Refer to IBM documentation on VSAM macro processing for details of the return and reason codes. Take the appropriate corrective action and restart the job. If no cause of the error can be identified, contact your NearArchive product support representative.

**OTS11024 - DYNALLOC ERROR, REQ=aa RC=bb DD=ccccccc  
DSN=dddddddddddddddddddddddddddddddddddddd  
VOL=vvvvvv UNIT=uuuuuuuu**

**OTS11025 - INF=eeee ERR=ffff SYSC=g SYSD=hhhhhhhh DEFER=ii  
DISP=jjkkll SPACE=(pppppp,sssss)  
DCB=(mmmmm,nnnnn,oo)**

Explanation: An error has occurred dynamically allocating a dataset, and processing of the utility has been terminated.

Messages OTS11024 and OTS11025 are issued for problem determination purposes. Some or all of the following operands may appear:-

'ccccccc' - DDname of the dataset being processed.  
'ddd...ddd' - name of the dataset being processed.  
'eeee' - the dynamic allocation information code  
'ffff' - the dynamic allocation error code

Action: Refer to IBM documentation on dynamic allocation processing for a description of the error and information codes. Take the appropriate corrective action and restart the job.

#### **OTS11026 – END OF TAPE ON OUTPUT|DUPLEX FILE**

Explanation: An end-of-volume condition has occurred on a backup tape dataset. This condition indicates that the backup

tape volume had less capacity than the primary copy being backed up. Processing of the utility is terminated.

Action: Rerun the utility, ensuring that the scratch tape volume used for the backup copy is of sufficient capacity to hold all the data contained on the corresponding primary tape dataset copy.

There are no special restart requirements. Backup tape datasets created during the failed run will be automatically uncataloged during the rerun, and new tape volumes used for backing the primary tape dataset(s).

**OTS11028 – UNIT TYPE MISMATCH: INPUT DEVICE TYPE IS xxxx  
AND OUTPUT|DUPLEX DEVICE TYPE IS xxxx**

Explanation: There is a device type mismatch between the primary input volume and the primary or duplex backup volume. 'xxxx' and 'yyyy' identify the input and output tape device types respectively.

If the input device type is a 3490 and the output device type a 3590, a condition code of 4 will be generated and processing will continue normally. Otherwise execution of the utility will be terminated.

Action: Due to differences in block identification structure, a 3490-format tape volume may not be used as backup for a 3590-format tape volume. If this condition has occurred, ensure that volumes of the same device type are used for tape input and output, and rerun the job.

A 3590-format tape volume may be used as backup for a 3490-format volume, but performance may be degraded significantly when retrieving objects directly from the backup volume, when the primary volume has been placed in recovery status. To avoid this condition, ensure that volumes of the same device type are used for tape input and output.

**OTS11029 - CAPACITY MISMATCH: OUTPUT|DUPLEX xx GB  
VOLUME HAS LESS CAPACITY THAN INPUT yy GB  
VOLUME**

Explanation: There is a volume capacity mismatch between the primary input volume and the primary or duplex backup volume. 'xx' and 'yy' identify the input and output tape volume capacities respectively.

If the 'NODEVCHECK' execution parameter is specified, a condition code of 4 will be generated and processing will continue normally. Otherwise execution of the utility will be terminated.

Action: The NearArchive database backup utility requires that a tape volume used for the primary or duplex backup datasets is not of lower capacity than the volume containing the tape dataset being backed up.

Ensure that volumes of the same capacity are used for tape input and output, and rerun the job.

**OTS11031 - TAPE ALLOCATION RETRY ENTERED, RETRY LIMIT  
NOW nnnnn**

Explanation: A request for allocation of a tape unit has been denied by the operator. The backup utility will retry the request up to 30 times before terminating abnormally.

Action: None. This is an informational message. If necessary, the database backup job may be terminated via and operator CANCEL command and subsequently restarted when sufficient tape units are available.

**OTS11032 – TAPE USAGE MISMATCH FOR B|C|D-COPY: VOLSAFE IS|  
NOT REQUIRED BUT IS|NOT MOUNTED**

Explanation: The status of the Sun Oracle Volsafe feature on the scratch tape mounted for the 'B', 'C' or 'D' backup copy does not match the required status defined for that copy in the database storage level containing the tape being backed up.

Action: Modify tape management system controls to ensure that the scratch tape mounted in response to the allocation

request from the backup utility has the required Volsafe status, and rerun the job.

**OTS11033 - I/O ERROR ON INPUT|OUTPUT TAPE. SEE  
ACCOMPANYING ERROR MESSAGE IN SYSLOG -  
PROCESSING TERMINATED**

Explanation: An input-output error has been received during processing of an input or output tape dataset. Execution of the utility will be terminated.

Action: Review the system or job log for additional system messages regarding the I/O error. Take the appropriate corrective action and rerun the job. If no apparent cause for the error can be determined, contact your NearArchive product support representative.

**OTS11038 - INVALID PARM SPECIFIED ON EXEC STATEMENT -  
PROCESSING TERMINATED**

Explanation: A validation error was detected during execution parameter checking. Execution of the utility has been terminated.

Action: Refer to the parameter validation error report issued by the utility for information regarding the error(s) encountered during parameter validation processing. Take the appropriate corrective action and rerun the utility.

**OTS11041 - LOCATE-BLOCK ERROR. COMMAND CODE = X'aa'  
RETURN CODE = X'bb' STATUS FLAGS=X'cccc' SENSE  
CODES = X'dddd'**

Explanation: An error has been received attempting to locate a block on a tape volume. The following diagnostic information is supplied:

- 'aa' - the channel command code issued to the tape device
- 'bb' - the completion code returned from the sub-channel
- 'cccc' - unit and channel status flags on completion of the request

‘dddd’ - sense bytes from the IOB on completion of the request.

Execution of the utility is terminated

Action: Refer to documentation relating to the tape device on which the error was detected for an explanation of the diagnostic codes in the message.

Take the appropriate corrective action and rerun the job. If no reason can be identified for the error, contact your NearArchive product support representative.

#### **OTS11043 - INCREMENTAL BACKUP COUNT CANNOT EXCEED 9999. GOING TO NEXT RECORD.**

Explanation: A primary tape dataset has been selected for backup processing when TYPE=INCREMENTAL has been set in the execution parameters, but the maximum number of 9999 incremental backup datasets already exist for that primary dataset.

No backup processing will be performed for the selected primary dataset. A program condition code of 4 will be set, and processing will continue with the next primary dataset to be backed up.

Action: Full-volume backup processing must be performed on the database, in order to release all the incremental backup datasets that have been created for the selected primary copy dataset.

Re-run the utility with an execution parameter of ‘TYPE=FULL’ (or allow this to default). Incremental backup processing may then be resumed as normal.

#### **OTS11046 - OTASBKUP DATASET IS MISSING FOR K COPY REQUEST.**

Explanation: ‘COPY=K’ has been specified in the EXEC parameters but there is no OTASBKUP DD entry in the JCL used to execute the utility. Execution terminates.

Action: AN OTASBKUP DD card must be present in the JCL when creating or re-creating one or more disk ('K') copy datasets. Modify the JCL to include this entry and resubmit the job.

**OTS11047 - RECOVERY IN PARM FOR K COPY REQUEST - IGNORED.**

Explanation: Both 'COPY=K' and 'RECOVERY' have been specified in the EXEC parameters. This is an invalid combination. The 'RECOVERY' parameter is ignored, and execution continues normally.

Action: Remove the 'RECOVERY' entry from the EXEC parameters to suppress this message for further executions of the utility.

**OTS11048 - LEVEL IN PARM FOR K COPY REQUEST - IGNORED.**

Explanation: Both 'COPY=K' and 'LEVEL' have been specified in the EXEC parameters. This is an invalid combination. The 'LEVEL' parameter is ignored, and execution continues normally.

Action: Remove the 'LEVEL' entry from the EXEC parameters to suppress this message for further executions of the utility.

**OTS11050 - DISK COPY PROCESSING NOT ENABLED IN STORAGE LEVEL FOR DSN = xxxx....xxxx**

Explanation: Dataset 'xxxx....xxxx' has been specified in the OTASBKUP file for processing by the utility when the 'COPY=K' EXEC parameter is present. Disk copy processing is not currently enabled for the storage level containing the specified tape dataset. The OTASBKUP entry is ignored and processing continues with the next entry in the file.

Action: Enable disk copy processing for the appropriate storage level (via the NearArchive database administration dialog) and rerun the utility.

**OTS11053 - ERROR IN NOTE PROCESSING ON OTASK100. RETURN  
CODE = X'xx. LAST BLOCK-ID OBTAINED = yyyyyyyy**

Explanation: Error code 'xx' has been received retrieving a block identifier from the input tape volume. 'yyyyyyyy' gives the identifier of the last block successfully retrieved by the utility. Processing of the current dataset is terminated, and continues with the next entry in the OTASBKUP file.

Action: Refer to IBM documentation on the 'NOTE' macro for a description of the error code displayed in the message. Take the appropriate corrective action and rerun the backup request.

**OTS11054 - ERROR IN NOTE PROCESSING ON OTASK200. RETURN  
CODE = X'xx. LAST BLOCK-ID OBTAINED = yyyyyyyy**

Explanation: Error code 'xx' has been received retrieving a block identifier from the output disk dataset when 'COPY=K' has been specified in the EXEC parameters. 'yyyyyyyy' gives the identifier of the last block successfully retrieved by the utility. Processing of the current dataset is terminated, and continues with the next entry in the OTASBKUP file.

Action: Refer to IBM documentation on the 'NOTE' macro for a description of the error code displayed in the message. Take the appropriate corrective action and rerun the backup request.

**OTS11055 - BLOCK COUNT MISMATCH CREATING K COPY OBJECT  
FOR DISK DSN = xxxx....xxxx**

Explanation: The disk copy of an object being processed when 'COPY=K' has been specified in the EXEC parameters does not contain the number of blocks identified in its index entry. Processing of the current dataset is terminated.

Action: This error indicates a possible data integrity problem with the primary tape dataset. If necessary, the dataset should be re-created using the NearArchive database restore utility OTASP130.

**OTS11056 - DISK COPY TERMINATED FOR DSN = xxxx....xxxx.  
ABEND CODE = X'yyy'**

Explanation: System abend 'yyy' has occurred during backup processing when 'COPY=K' has been specified. The disk dataset being processed is closed normally, and processing of the utility continues with the next entry in the OTASBKUP file. Some objects from the primary tape dataset may not be present on the disk copy dataset. Retrieval of these objects will be satisfied from tape.

Action: Refer to IBM documentation for a description of the system abend code. If the failure occurs due to lack of disk space during disk copy processing, make more disk space available and rerun the utility.

**OTS11057 – UNABLE TO GET VOLSER FOR NEW {DUPLEX} BACKUP  
TAPE**

Explanation: An internal error has occurred during cataloging of a new backup or duplex backup tape dataset. Processing of the utility is terminated.

Action: This message indicates an internal NearArchive error condition. Retain any other messages issued during utility execution and report the error to your NearArchive product support representative.

**OTS11061 – TAPE ALLOCATION RETRY ENTERED, RETRY LIMIT  
NOW nnnnn**

Explanation: The utility was unable to allocate a device for an output tape volume. Allocation will be retried after 1 minute. 'nnnnn' gives the number of retries remaining. The utility will terminate if allocation is not successful after this number of retries.

Action: Make a tape device available for allocation by the utility prior to the retry limit being reached. The utility will then continue with normal execution. Otherwise, rerun the utility when there are sufficient devices available for execution.



## **8.11 Batch Archival Request Processor Messages.**

---

### **OTS12002 - ERROR OPENING OTASBKUP DATASET**

Explanation: An error has occurred opening the OTASBKUP sequential disk dataset, during batch archival processing.

Action: Consult any other available system messages for additional information about the error. NearArchive processing continues, but no dataset backup information will be recorded. The NearArchive database backup utility will be unable to execute after completion of this batch archival process. Take the appropriate actions to ensure that the error does not re-occur.

### **OTS12003 - ERROR OPENING OTASLOG1 DATASET - DYNAMIC BACKOUT DISABLED**

Explanation: An error has occurred opening the OTASLOG1 sequential dataset during database open processing. Dynamic backout processing has been disabled for the database update operation. Database open processing will continue normally.

Action: Consult any other available system messages for additional information about the error. NearArchive processing continues, but no dynamic backout processing will be performed if a 'close without commit' request is issued by the calling application.

Take the appropriate corrective action to ensure that the error does not re-occur during further database update operations. If no apparent cause for the error can be identified, contact your NearArchive product support representative.

### **OTS12004 - DISK COPY PROCESSING DISABLED**

Explanation: Disk copy processing has been disabled during NearArchive database update processing. An accompanying message will give additional information concerning this condition.

Action: Refer to the accompanying OTS12005 or OTS12006 message for additional information. Take whatever action is recommended in the message description in the user manual.

#### **OTS12005 - MAXIMUM BLOCK COUNT EXCEEDED**

Explanation: This message accompanies message OTS12004 when disk copy processing has been disabled during NearArchive database update processing. Disk copy processing has been disabled because the maximum blockcount per disk ('K') copy dataset, as specified by the NearArchive database administrator, has been reached.

Action: None. This is an informational message. Disk copy processing will be automatically resumed when the next tape dataset in the database is created.

#### **OTS12006 - RC=aa, SC1=X'bbbb', SC2=X'cccc', LOCATION=dddd**

Explanation: This message accompanies message OTS12004 when disk copy processing has been disabled during NearArchive database update processing. An error has been encountered processing a disk ('K') copy dataset. 'aa', 'bbbb' and 'cccc' provide the return code, supplementary code 1 and supplementary code 2 respectively from the object archival request. 'dddd' is the internal program location identifier where the error was received.

Action: Disk copy processing will be disabled for the remainder of the database update operation; tape dataset update processing will continue normally.

Refer to NearArchive return and supplementary code documentation for a description of the error condition. Take the appropriate corrective action to ensure that the error does not re-occur during further database update operations. If no apparent cause for the failure can be determined, contact your NearArchive product support representative.

**OTS12008 - ASYNCHRONOUS SECONDARY INDEX UPDATES  
CANNOT BE PERFORMED WHEN DYNAMIC BACKOUT  
IS ENABLED**

Explanation: A database open request has been received from an application, but both the dynamic backout and asynchronous secondary index update facilities have been enabled in the JCL used to run the application. The database open request is rejected with a return code of '15' with supplementary code 1 set to 2, and supplementary code 2 to the identifier of the secondary index which is to be updated asynchronously.

Action: Either disable the dynamic backout facility by removing the 'OTASLOG1' card from the JCL deck, or disable asynchronous secondary index update processing by removing all 'OTnnxxxx' DD cards from the JCL deck (where 'nn' is the 2-digit identifier of a secondary index, and 'xxxx' is the database name as specified by the application in its 'open database' request), and rerun the job.

## **8.12 Database Recovery Utility Messages.**

---

**OTS13000 - NEARARCHIVE DATABASE RECOVERY UTILITY  
COMPLETED WITH CONDITION CODE xxx**

Explanation: This informational message is issued when the database recovery utility has completed processing. 'xx' is the highest condition code encountered during program execution.

Action: No action necessary.

**OTS13001 - OPEN FAILED FOR OTASV100. RETURN CODE=  
X'xxxxxxxx'**

Explanation: An error has occurred when opening the NearArchive database's primary index dataset OTASV100. 'xxxxxxxx' is the hexadecimal reason code returned from the VSAM OPEN macro. Processing is terminated.

Action: Refer to IBM documentation on VSAM macro processing for details of the reason code. Take the appropriate corrective action and restart the job.

**OTS13002 - STORAGE LEVEL 'a' DOES NOT EXIST IN THIS DATABASE - PROCESSING TERMINATED**

Explanation: Storage level 'a' has been specified on the job's EXEC statement parameter, but does not exist in the NearArchive database.

Action: Check the LEVEL parameter on the EXEC parameter. Correct and rerun if necessary.

**OTS13003 - INVALID EXEC PARM 'aaaaaaaaaaaa'**

Explanation: An invalid parameter has been specified on the EXEC statement. Processing is terminated.

Action: Refer to chapter 7 of this manual for EXEC parameter specification. Make the appropriate corrections to the EXEC parameters and restart the job.

**OTS13004 - DD CARD MISSING IN JCL FOR OTASV100**

Explanation: The JCL DD statement for the NearArchive database's primary index dataset OTASV100 has not been specified in the execution JCL.

Action: Correct the JCL and resubmit the job.

**OTS13006 - UNABLE TO GETMAIN STORAGE FOR xxxx BUFFER**

Explanation: A request for buffer storage in the job's private area has been denied by the operating system. 'xxxx' may take the following values:

PCR - Processing Control Record  
SLC - Storage Level Control record  
MIE - Main Index Entry record  
TAPE - Tape data buffers

Action: Increase the REGION size for the job or jobstep, and resubmit the job.

### **OTS13007 - ERROR X'xxyy' READING PROCESSING CONTROL RECORD**

Explanation: An error has occurred reading the processing control record from the NearArchive database's primary index dataset. 'xx' and 'yy' are the hexadecimal return and reason codes respectively. Processing is terminated.

Action: Refer to IBM documentation on VSAM macro processing for details of the return and reason codes. Ensure that the NearArchive database has been defined correctly.

If no cause for the error can be identified, contact your NearArchive product support representative.

### **OTS13008 - OTASV100 IS A PRE-VERSION 2.5 DATABASE - RECOVERY UTILITY CANNOT EXECUTE**

Explanation: The NearArchive database being processed has not been converted for use by this release of the product. Execution of the utility is terminated.

Action: Database recovery processing is only permitted for databases which have been converted for use with v2.5. Execute the database conversion process using the NearArchive database administration procedure in TSO/ISPF before rerunning the job. Refer to chapter 3 of this manual for information about the database conversion process.

### **OTS13009 - ERROR X'xxyy' READING CONTROL RECORD FOR STORAGE LEVEL a**

Explanation: An error has occurred when reading the control record for storage level 'a' from the NearArchive database's primary index dataset. 'xx' and 'yy' are the hexadecimal return and reason codes respectively. Processing is terminated.

Action: Refer to IBM documentation on VSAM macro processing for details of the return and reason codes. Ensure that the NearArchive database has been defined correctly. If no cause of the error can be identified, contact your NearArchive product support representative.

### **OTS13010 - INTERNAL NEARARCHIVE PROCESSING ERROR**

Explanation: An internal error has occurred during execution of the utility. Processing is terminated.

Action: Retain all information regarding execution of the utility, and contact your NearArchive product support representative.

### **OTS13011 - ERROR X'xxyy' UPDATING CONTROL RECORD FOR STORAGE LEVEL a**

Explanation: An error has occurred when updating the control record for storage level 'a' in the NearArchive database's primary index.

'xx' and 'yy' are the hexadecimal return and reason codes respectively from the VSAM PUT macro. Processing is terminated.

Action: Refer to IBM documentation on VSAM macro processing for details of the return and reason codes. Take the appropriate corrective action and restart the job.

### **OTS13013 - UNABLE TO OPEN DATASET WITH DDNAME=aaaaaaaa DCB OPEN FLAG=bb**

Explanation: An error has occurred attempting to open the sequential file with DDname aaaaaaaa. aaaaaaaa may contain the following values:

OTAST100 - Primary archive volume  
OTAST200 - 'B' copy of primary volume

'bb' is the value in the DCB open flag after failure of the request.

Action: Refer to other system messages for additional information regarding the open failure. Take the appropriate corrective action and rerun the recovery utility.

**OTS13014 - DATASET 'aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa'**  
**VOLSER bbbbbb cccc ddddddddddddddddddddddddddd**

Explanation: This informational message is printed on completion of recovery processing for each tape volume. The following variables will appear in this message:

aaaaa...aaaaa - the name of the tape dataset being processed.

bbbbbb - the tape volume serial number.

cccc - ' - ' (normal completion)  
'\*\*\*\*' (error occurred)

dddddddddd - the action taken and may contain the following values:-

'OLD PRIMARY DATASET UNCATALOGED'  
'NEW PRIMARY DATASET RECOVERED AND  
CATALOGED'  
'NEW PRIMARY DATASET NOT CATALOGED'

Action: No action necessary.

**OTS13015 - ERROR X'xxyy' READING INDEX ENTRY RECORD**

Explanation: An error has occurred when reading a main index entry record from the NearArchive database's primary index dataset. 'xx' and 'yy' are the hexadecimal return and reason codes respectively. Processing is terminated.

Action: Refer to IBM documentation on VSAM macro processing for a description of the return and reason codes. Take the appropriate action and restart the job. If no cause of the error can be identified, contact your NearArchive product support representative.

**OTS13016 - PROCESSING STORAGE LEVEL(S) a bbbbbbbbbbbbbbbb**

Explanation: Informational message indicating the storage level(s) being processed.

bbbbbbbbbbbbbb may contain the following:-

(EXEC PARAMETER) - the storage level being processed is taken from the job EXEC parameter.

(DEFAULT) - all storage levels are being processed (the default value).

Action: No action necessary.

### **OTS13019 - END OF VOLUME ENCOUNTERED ON OUTPUT TAPE VOLUME**

Explanation: An end-of-volume condition has occurred on the output tape volume during dataset recovery processing. This is not permitted.

Action: Ensure that the tape volume to be used for the recovered primary dataset is of a similar specification to that of the backup dataset being used for data recovery. Rerun the recovery utility.

### **OTS13020 - ERROR X'xxyy' UPDATING INDEX ENTRY RECORD**

### **OTS13021 - PRIMARY KEY = aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa**

Explanation: An error has occurred when updating a main index entry record in the NearArchive database's primary index dataset. 'xx' and 'yy' are the hexadecimal return and reason codes respectively. 'aaaaaaaaaa' is the primary key of the NearArchive object being processed. Processing is terminated.

Action: Refer to IBM documentation on VSAM macro processing for details of the return and reason codes. Take the appropriate action and restart the job. If no cause of the error can be identified, contact your NearArchive product support representative.

### **OTS13023 - NO TAPE VOLUMES IN RECOVERY STATUS FOR REQUESTED STORAGE LEVEL(S)**

Explanation: Informational message indicating that no tape volumes are eligible for recovery.



Action: No action necessary.

**OTS13024 - DYNALLOC ERROR, REQ=aa RC=bb DD=ccccccc  
DSN=dddddddddddddddddddddddddddddddddddddd  
VOL=vvvvvv UNIT=uuuuuuuu**

**OTS13025 - INF=eeee ERR=ffff SYSC=g SYSD=hhhhhhhh DEFER=ii  
DISP=jjkkll SPACE=(pppppp,sssss)  
DCB=(mmmmmm,nnnnnn,oo)**

Explanation: An error has occurred dynamically allocating a dataset, and processing of the utility has been terminated.

Messages OTS03024 and OTS03025 are issued for problem determination purposes. Some or all of the following operands may appear:-

'ccccccc' - DDname of the dataset being processed  
'ddd...ddd' - name of the dataset being processed  
'eeee' - the dynamic allocation information code  
'ffff' - the dynamic allocation error code

Action: Refer to IBM documentation on dynamic allocation processing for a description of the error and information codes. Take the appropriate corrective action and restart the job.

**OTS13028 – UNIT TYPE MISMATCH: INPUT DEVICE TYPE IS xxxx  
AND OUTPUT DEVICE TYPE IS xxxx**

Explanation: There is a device type mismatch between the input volume and the output volume. 'xxxx' and 'yyyy' identify the input and output tape device types respectively.

If the input device type is a 3490 and the output device type is a 3590, a condition code of 4 will be generated and processing will continue normally. Otherwise execution of the utility will be terminated.

Action: Due to differences in block identification structure, a 3590-format tape volume may not be restored to a 3490-

format tape volume. If this condition has occurred, ensure that volumes of the same device type are used for tape input and output, and rerun the job.

A 3490 format tape volume may be restored to a 3590-format volume, but performance may be degraded significantly when retrieving objects directly from the backup volume, should the primary volume be placed in recovery status at some future date. To avoid this condition, ensure that volumes of the same device type are used for tape input and output.

**OTS13029 - CAPACITY MISMATCH: OUTPUT|DUPLEX xx GB  
VOLUME HAS LESS CAPACITY THAN INPUT yy GB  
VOLUME**

Explanation: There is a volume capacity mismatch between the backup volume being processed and the volume being restored. 'xx' and 'yy' identify the input and output tape volume capacities respectively.

If the 'NODEVCHECK' execution parameter is specified, a condition code of 4 will be generated and processing will continue normally. Otherwise execution of the utility will be terminated.

Action: The NearArchive database recovery utility requires that a tape volume used for the primary dataset being restored is not of lower capacity than the volume containing the backup tape dataset from which recovery is being performed.

Ensure that volumes of the same capacity are used for tape input and output, and rerun the job.

**OTS13030 - TOTAL VOLUMES PROCESSED FOR STORAGE  
LEVEL a = 999999**

Explanation: Informational message indicating the number of tape volumes processed for storage level a.

Action: No action necessary.

**OTS13031 - TOTAL VOLUMES PROCESSED FOR ALL STORAGE LEVELS = 999999**

Explanation: Informational message indicating the number of tape volumes processed for all storage levels

Action: No action necessary.

**OTS13032 – TAPE USAGE MISMATCH: VOLSAFE IS|NOT REQUIRED BUT IS|NOT MOUNTED**

Explanation: The status of the Sun Oracle Volsafe feature on the scratch tape mounted for the primary copy dataset being restored does not match the required status defined for the 'A' copy in the database storage level containing the tape being backed up.

Action: Modify tape management system controls to ensure that the scratch tape mounted in response to the allocation request from the backup utility has the required Volsafe status, and rerun the job.

**OTS13033 I/O ERROR ON INPUT|OUTPUT TAPE. SEE ACCOMPANYING ERROR MESSAGE IN SYSLOG - PROCESSING TERMINATED**

Explanation: An input-output error has been received during processing of an input or output tape dataset. Execution of the utility will be terminated.

Action: Review the system or job log for additional system messages regarding the I/O error. Take the appropriate corrective action and rerun the job. If no apparent cause for the error can be determined, contact your NearArchive product support representative.

**OTS13034 - UNABLE TO ALLOCATE INPUT INCREMENTAL BACKUP TAPE. - PROCESSING TERMINATED.**

Explanation: The utility was unable to allocate an incremental backup tape during recovery of a primary dataset. Execution of the utility has been terminated.

Action: Review the system or job log for additional system messages regarding the error. Take the appropriate

corrective action and rerun the job. If no apparent cause for the error can be determined, contact your NearArchive product support representative.

**OTS13035 - ERROR PROCESSING ITEM CONTROL RECORD. -  
PROCESSING TERMINATED.**

Explanation: An error has been encountered processing an object's control record from a full-volume or incremental backup tape dataset. Execution of the utility is terminated.

Action: Retain all information concerning the error condition, and report the error to your NearArchive product support representative.

**OTS13036 - UNABLE TO READ SPACE MANAGEMENT RECORD -  
PROCESSING TERMINATED.**

Explanation: An error has been encountered accessing a record from the database's space management file for the storage level being processed by the recovery utility. Execution of the utility is terminated.

Action: Retain all information concerning the error condition, and report the error to your NearArchive product support representative.

**OTS13037 - WRITE TO OUTPUT TAPE FAILED. - PROCESSING  
TERMINATED**

Explanation: An input-output error has been received during processing of an output tape dataset. Execution of the utility will be terminated.

Action: Review the system or job log for additional system messages regarding the I/O error. Take the appropriate corrective action and rerun the job. If no apparent cause for the error can be determined, contact your NearArchive product support representative.

**OTS13038 - INVALID PARM SPECIFIED ON EXEC STATEMENT -  
PROCESSING TERMINATED**

Explanation: A validation error was detected during execution parameter checking. Execution of the utility has been terminated.

Action: Refer to the parameter validation error report issued by the utility for information regarding the error(s) encountered during parameter validation processing. Take the appropriate corrective action and rerun the utility.

**OTS13039 - OUTPUT TAPE WAS NOT OPENED. - PROCESSING  
TERMINATED**

Explanation: An error has been received opening an output tape dataset. Execution of the utility will be terminated.

Action: Review the system or job log for additional system messages regarding the error. Take the appropriate corrective action and rerun the job. If no apparent cause for the error can be determined, contact your NearArchive product support representative.

**OTS13040 - ERROR READING INCREMENTAL BACKUP INPUT TAPE.  
- PROCESSING TERMINATED**

Explanation: An input-output error has been received during processing of an input incremental backup tape dataset. Execution of the utility will be terminated.

Action: Review the system or job log for additional system messages regarding the I/O error. Take the appropriate corrective action and rerun the job. If no apparent cause for the error can be determined, contact your NearArchive product support representative.

**OTS13041 - REQUESTED BACKUP COPY “x” NOT FOUND FOR THIS  
DATA SET. PROCESSING TERMINATED.**

Explanation: A primary tape dataset is currently in recovery control status in the NearArchive database being processed by the recovery utility, and has been selected for recovery

processing. However, the backup dataset copy identifier specified in the 'COPYFULL=' execution parameter for the utility does not exist. Processing of the utility is terminated.

Action: Check the reason that the identified full-volume backup dataset does not exist. If necessary, modify the 'COPYFULL=' execution parameter to specify the correct backup dataset identifier, and rerun the utility.

**OTS13042 - OLD DATASET REMAINS CATALOGED. (NEW TAPE IS EMPTY AND NOT CATALOGED.)**

Explanation: This message accompanies message OTS13041, when a backup dataset does not exist for recovery of a primary tape dataset.

Action: None. This message provides additional information concerning the error identified by message OTS13041. Refer to the description of that message for the action required.

**OTS13043 - NEW COPY OF DATASET RESTORED BUT NOT CATALOGED**

Explanation: An attempt to catalog a restore primary tape volume failed. The dataset has been restored but is not cataloged. Execution of the utility is terminated. This message is accompanied by message OTS13044.

Action: Review the system or job log for additional system messages regarding the error. Take the appropriate corrective action and rerun the job. If no apparent cause for the error can be determined, contact your NearArchive product support representative.

**OTS13044 - THE ORIGINAL COPY OF THIS DATASET IS STILL CATALOGED.**

Explanation: This message accompanies message OTS13043, when recataloging of a restored primary dataset has failed.

Action: None. This message provides additional information concerning the error identified by message OTS13043.

Refer to the description of that message for the action required.

#### **OTS13045 - ERROR PROCESSING RECOVERY CONTROL SECTIONS (RCS).**

Explanation: An error has occurred processing recovery control information in the NearArchive primary index file. Execution of the utility is terminated.

Action: Retain all information concerning the error condition, and report the error to your NearArchive product support representative.

### **8.13 Batch Archival Tape I/O Driver Messages.**

---

#### **OTS15201 - TAPE ALLOCATION DENIED. NEARARCHIVE WILL RETRY ALLOCATION AFTER 1 MINUTE**

Explanation: NearArchive has attempted to allocate a tape drive but no device is currently available. NearArchive will wait for one minute before making another attempt.

The number of allocation retries to be performed by NearArchive may be specified in the application's database open call. A default value of 30 is used if no value is specified by the application.

Action: If possible, make a tape drive available for NearArchive allocation, by varying online any devices which are currently offline. NearArchive will acquire any such device during allocation retry processing.

If no other devices are available, cancel another job in the system, or wait for another job to finish. Any tape drives allocated to the other job will become available when that job terminates, and may be acquired by NearArchive during allocation retry processing.

If no drive becomes available before the number of allocation retries has been exhausted, NearArchive will fail the application request with a dynamic allocation error code.

**OTS15202 - EOVS OCCURRED AT LOCATION xxxx: COMPLETION  
CODE = yy, STATUS CODE = zz**

Explanation: Logical end-of-volume has been detected on an output tape during batch archival processing. 'xxxx' gives an internal program location identifier, and 'yy' and 'zz' give the contents of the completion code and unit status fields from the IOB.

Action: None. This is an information message only. NearArchive will automatically create a new tape dataset and continue archival processing normally.

## **8.14 Online Request Processor Messages.**

---

**OTS20000 - EIBRESP 'xxx' RECEIVED FOR FUNCTION X'yyy'**

Explanation: A CICS command failure occurred during command execution. The response 'xxx' and function codes 'yyy' are as specified in the Exec Interface Block (EIB).

Action: Refer to CICS documentation for details of the function and response codes. Take appropriate corrective action. Contact your NearArchive support representative if there is no apparent reason for the error.

**OTS20001 - ABEND xxxx RECEIVED DURING NEARARCHIVE  
RETRIEVAL ROCESSING**

Explanation: An abend with code 'xxxx' has occurred during CICS retrieval processing.

Action: Refer to CICS documentation for details of the abend and corrective action. Contact your NearArchive support representative if there is no apparent reason for the error.

**OTS20002 – USER EXIT PROGRAM OTASUX02 NOT DEFINED TO  
CICS**

Explanation: A program entry for the user exit program OTASUX02 has not been defined to CICS.



Action: Ensure that the correct CICS table entries have been defined. Refer to section 2.8 for information on CICS table definitions.

#### **OTS20500 - EIBRESP 'xxx' RECEIVED FOR FUNCTION X'yyy'**

Explanation: A CICS command failure occurred during command execution. The response 'xxx' and function codes 'yyy' are as specified in the Exec Interface Block (EIB).

Action: Refer to CICS documentation for details of the function and response codes. Take appropriate corrective action. Contact your NearArchive support representative if there is no apparent reason for the error.

#### **OTS20501 - ABEND xxxx RECEIVED DURING NEARARCHIVE RETRIEVAL ROCESSING**

Explanation: An abend with code 'xxxx' has occurred during CICS retrieval processing.

Action: Refer to CICS documentation for details of the abend and corrective action. Contact your NearArchive support representative if there is no apparent reason for the error.

#### **OTS21000 - EIBRESP 'xxx' RECEIVED FOR FUNCTION X'yyy'**

Explanation: A CICS command failure occurred during command execution. The response 'xxx' and function codes 'yyy' are as specified in the Exec Interface Block (EIB).

Action: Refer to CICS documentation for details of the function and response codes. Take appropriate corrective action. Contact your NearArchive support representative if there is no apparent reason for the error.

#### **OTS21001 - ABEND xxxx RECEIVED DURING NEARARCHIVE RETRIEVAL PROCESSING**

Explanation: An abend with code 'xxxx' has occurred during CICS retrieval processing.

Action: Refer to CICS documentation for details of the abend and corrective action. Contact your NearArchive support representative if there is no apparent reason for the error.

#### **OTS21500 - EIBRESP 'xxx' RECEIVED FOR FUNCTION X'yyy'**

Explanation: A CICS command failure occurred during command execution. The response 'xxx' and function codes 'yyy' are as specified in the Exec Interface Block (EIB).

Action: Refer to CICS documentation for details of the function and response codes. Take appropriate corrective action. Contact your NearArchive support representative if there is no apparent reason for the error.

#### **OTS21501 - ABEND xxxx RECEIVED DURING NEARARCHIVE DELETE PROCESSING**

Explanation: An abend with code 'xxxx' has occurred during CICS object deletion processing.

Action: Refer to CICS documentation for details of the abend and corrective action. Contact your NearArchive support representative if there is no apparent reason for the error.

#### **OTS22000 - EIBRESP 'xxx' RECEIVED FOR FUNCTION X'yyy'**

Explanation: A CICS command failure occurred during command execution. The response 'xxx' and function codes 'yyy' are as specified in the Exec Interface Block (EIB).

Action: Refer to CICS documentation for details of the function and response codes. Take appropriate corrective action. Contact your NearArchive support representative if there is no apparent reason for the error.

#### **OTS22001 - ERROR 'xx' REASON X'yyyyzzzz' RECEIVED FROM TAPE|DISK READER SUB-TASK**

Explanation: An error has been received from a tape or disk reader sub-task by the NearArchive CICS online retrieval scheduler.

Action: This message supplies additional information when an error has occurred during online retrieval processing. Refer to the NearArchive return and supplementary code section in chapter 8 for a description of the return code 'xx' and supplementary codes 'yyyy' and 'zzzz'.

Contact your NearArchive support representative if there is no apparent reason for the error.

**OTS22002 - FILE 'OTASnnnn' DATASET'xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx'**

Explanation: This message accompanies message OTS22003 and supplies additional information about the file and dataset names being processed when a retrieval error occurred.

Action: See message OTS22003.

**OTS22003 - ABEND xxxx RECEIVED DURING NEARARCHIVE RETRIEVAL PROCESSING**

Explanation: An abend with code 'xxxx' has occurred during CICS retrieval processing.

Action: Refer to CICS documentation for details of the abend and corrective action. Contact your NearArchive support representative if there is no apparent reason for the error.

**OTS22004 – UNABLE TO OBTAIN LOCK DURING ERROR RECOVERY PROCESSING**

Explanation: The processor was unable to obtain a control block lock when trying to recover from an earlier error condition.

Action: Processing will continue normally. Inform your NearArchive support representative if this message is displayed persistently.

**OTS22005 – INVALID REQUEST ELEMENT CHAIN PROCESSING TAPE|DISK SUB-TASK nnn**

Explanation: An invalid request chain has been encountered when attempting to pass a new request to tape or disk sub-task 'nnn' (where 'nnn' is in the range 1-256).

Action: Processing of the request is rejected with a return code of '23'. Contact your NearArchive support representative if there are repeated instances of this condition.

## **8.15 Control Region Initialization Messages.**

---

### **OTS23001 - NEARARCHIVE V2.5 NOW INITIALIZED**

Explanation: The NearArchive control region has now been initialized for retrieval processing.

Action: No action necessary.

### **OTS23002 - NEARARCHIVE ALREADY INITIALIZED IN THIS MRO CONFIGURATION**

Explanation: A NearArchive control region has already been initialized in this CICS MRO configuration. This may be the region in which initialization has been attempted, or in another connected CICS region. The initialization request is terminated.

Action: No action necessary.

### **OTS23004 - NEARARCHIVE V2.5 INITIALIZATION FAILED**

Explanation: NearArchive initialization has failed in this region. The cause of failure will be detailed in an accompanying message.

Action: Determine why NearArchive initialization has failed from accompanying messages and take corrective action. Contact your NearArchive support representative if there is no apparent reason for the error.

### **OTS23005 - EIBRESP 'xxx' RECEIVED FOR FUNCTION X'yyyy'**

Explanation: A CICS command failure occurred during command execution. The response 'xxx' and function codes 'yyy' are as specified in the Exec Interface Block (EIB).

Action: Refer to CICS documentation for details of the function and response codes. Take appropriate corrective action.

Contact your NearArchive support representative if there is no apparent reason for the error.

#### **OTS23006 - INITIALIZATION ERROR xxxx**

Explanation: A general error has been received from initialization processing.

Action: Contact your NearArchive product support representative.

#### **OTS23007 - RETURN ERROR xxxx FOR ORAB ALLOCATION**

Explanation: Allocation of the NearArchive request anchor block in the OS/390 CSA has failed. The error code may take the following values:

4 - Internal request error.

Action: Contact NearArchive support.

8 - GETMAIN request has been denied by operating system.

Action: Increase the size of the OS/390 CSA, and retry the request.

12 - FREEMAIN request has been denied by the operating system.

Action: Contact NearArchive support.

16 - a) Module OTASP055 was not found during GETMAIN processing or

b) A system abend has been intercepted by the CSA allocation routines.

Action: Check system messages in the job log for details of the abend code. In particular, if abend S047 has been received from module OTASP055, ensure that this module does not reside outside the PLPA (e.g.) on the product distribution library (see installation instructions for a discussion of this consideration).

### **OTS23009 - NO INITIALIZED DATABASE IN THIS SYSTEM AT THE CORRECT RELEASE LEVEL**

Explanation: No database index has been found in the CICS File Control Table (FCT) for a version 2.5 database. Initialization processing is terminated unsuccessfully.

Action: Check that FCT entries and JCL are correct. If necessary convert any pre-version 2.5 databases which may already be defined in the FCT to the correct format, using the database conversion procedure in the NearArchive database administration dialog in TSO/ISPF. Refer to chapter 3 of this manual for information on this procedure.

### **OTS23010 - DATABASE INDEX NOT OPEN**

Explanation: No opened NearArchive database index has been found in this region.

Action: Ensure that the FILSTAT parameter on the FCT entry for the database index is correctly coded.

### **OTS23011 - RETURN ERROR xxxx FOR ORCB ALLOCATION.**

Explanation: Allocation of a NearArchive request control block in the OS/390 CSA has failed. See description of message OTS23007 for a list of possible codes.

Action: Follow actions outlined in message OTS23007.

## **8.16 Online Parameter File (OTASS100) Processor Messages.**

---

### **OTS23200 - aa**

Explanation: This message displays card images from the OTASS100 dataset processed during NearArchive control region initialization.

Action: No action necessary.

### **OTS23202 - PARAMETER ERRORS DETECTED**

Explanation: An error has been detected in the NearArchive control region initialization parameters in file OTASS100.

Action: Refer to chapter 2 for a description of the CICS initialization parameter formats. NearArchive continues with control region initialization, using default values for all initialization parameters. Correct the parameter in error before rerunning the control region initialization transaction.

### **OTS23203 - UNABLE TO OPEN OTASS100**

Explanation: An error has occurred when opening the NearArchive control region initialization parameter file OTASS100.

Action: NearArchive continues with control region initialization, using default values for all initialization parameters. Correct the parameter in error before rerunning the control region initialization transaction.

### **OTS23204 - FOLLOWING PARAMETERS READ FROM OTASS100:**

Explanation: This messages precedes the display of card images from the NearArchive control region initialization file OTASS100.

Action: No action necessary.

### **OTS23206 - NEARARCHIVE ONLINE CONFIGURATION IS:**

Explanation: This message precedes messages OTS23207 and OTS23209 which display the current NearArchive CICS control region configuration.

Action: No action necessary.

### **OTS23207 - MAXDRIVE = xxx**

Explanation: This message displays the maximum number of tape drives which are available for use by the NearArchive CICS control region at initialization.

Action: No action necessary.

**OTS23208 - MAXDISK = xxx**

Explanation: This message displays the maximum number of disk reader sub-tasks which are initialized for use by the NearArchive CICS control region at initialization.

Action: No action necessary.

**OTS23209 - MAXQLEN = xxx**

Explanation: This message displays the maximum number of concurrent requests which NearArchive will queue for any one tape volume.

Action: No action necessary.

**OTS23210 - TSQDELETE = xxxx**

Explanation: This message displays the value of the TSQDELETE control parameter set by the NearArchive CICS control region at initialization.

Action: No action necessary.

**OTS23211 - TAPEWAIT = xxxxx**

Explanation: This message displays the value of the TAPEWAIT control parameter set by the NearArchive CICS control region at initialization.

Action: No action necessary.

**OTS23212 - TIMEOUT = xxxxx**

Explanation: This message displays the value of the TIMEOUT control parameter set by the NearArchive CICS control region at initialization.

Action: No action necessary.



## **8.17 Control Region Quiesce Messages.**

---

### **OTS24001 - NEARARCHIVE V2.5 NOW QUIESCED**

Explanation: NearArchive processing has been terminated in the control region.

Action: No action necessary.

### **OTS24002 - NEARARCHIVE NOT INITIALIZED - REQUEST IGNORED**

Explanation: NearArchive is not currently initialized in this CICS region. The request is ignored.

Action: No action necessary.

### **OTS24003 - NEARARCHIVE QUIESCE FAILED**

Explanation: NearArchive has failed to quiesce in this region. An accompanying message will give details of the failure.

Action: Refer to the description of the accompanying message(s) for details of actions to be performed.

### **OTS24004 - xxxxxx RECEIVED FOR xxxxxx FUNCTION**

Explanation: A CICS command failure occurred during task execution. The response and function codes are as specified in fields EIBRESP and EIBFN in the EXEC Interface Block (EIB).

Action: Refer to CICS documentation for details of the function and response codes. Take appropriate corrective action. Contact your NearArchive product support representative if there is no apparent reason for the failure.

### **OTS24005 - SHUTDOWN REQUESTED FROM OUTSIDE CONTROL REGION - REQUEST REJECTED**

Explanation: OT24 has been issued from a secondary NearArchive region. This is not permissible.

Action: NearArchive shutdown must be performed from within the NearArchive control region.

## **8.18 CICS Reader Task Messages.**

---

### **OTS25001 - GETMAIN ERROR. SMF PROCESSING WILL BE DISABLED FOR TAPE READER TASK nnn**

Explanation: Insufficient virtual storage is available in the CICS region to satisfy a GETMAIN request by tape reader task 'nnn'. SMF recording of object retrieval activity by that task will be disabled until the task is restarted.

Action: Increase the amount of available below the 16Mb line in the CICS private area for OS/390 (non-CICS) storage requests by adjusting CICS initialization parameters and restarting CICS.

### **OTS25002 - RC=xx, SC1=X'yyyy', SC2=X'zzzz' FROM SMF WRITE - SMF RECORDING DISABLED**

Explanation: A request to write an SMF record by a tape reader task has been rejected by the NearArchive SMF processing component with the return and supplementary codes displayed in the message. SMF recording of object retrieval activity by that task will be disabled until the task is restarted.

If the return code = '14' an error was received from the SMF write request. In these circumstances, supplementary code 1 contains the error code from the SMFWTM macro.

Action: For return code '14', refer to IBM documentation on SMF for an explanation of the SMFWTM error code. Take the appropriate corrective action and stop and restart the reader task.

For other return codes, or if no reason can be found for the SMF write error, contact your NearArchive product support representative.

### **OTS25003 - TASK nnn MULTI-BLOCK REQUEST TIMEOUT**

**Explanation:** Tape reader sub-task 'nnn' has waited for a response from the NearArchive object retrieval driver program for longer than the time interval specified in the TIMEOUT initialization parameter. The request will be terminated.

**Action:** Consult any other available NearArchive system error messages for additional information regarding this condition. If no apparent cause for the timeout can be identified, contact your NearArchive customer support representative.

### **OTS25401 - GETMAIN ERROR. SMF PROCESSING WILL BE DISABLED FOR DISK READER TASK nnn**

**Explanation:** Insufficient virtual storage is available in the CICS region to satisfy a GETMAIN request by disk reader task 'nnn'. SMF recording of object retrieval activity by that task will be disabled until the task is restarted.

**Action:** Increase the amount of available below the 16Mb line in the CICS private area for OS/390 (non-CICS) storage requests by adjusting CICS initialization parameters and restarting CICS.

### **OTS25002 - RC=xx, SC1=X'yyyy', SC2=X'zzzz' FROM SMF WRITE - SMF RECORDING DISABLED**

**Explanation:** A request to write an SMF record by a disk reader task has been rejected by the NearArchive SMF processing component with the return and supplementary codes displayed in the message. SMF recording of object retrieval activity by that task will be disabled until the task is restarted.

If the return code = '14' an error was received from the SMF write request. In these circumstances,

supplementary code 1 contains the error code from the SMFWTM macro.

Action: For return code '14', refer to IBM documentation on SMF for an explanation of the SMFWTM error code. Take the appropriate corrective action and stop and restart the reader task.

For other return codes, or if no reason can be found for the SMF write error, contact your NearArchive product support representative.

### **OTS25403 - TASK nnn MULTI-BLOCK REQUEST TIMEOUT**

Explanation: Disk reader sub-task 'nnn' has waited for a response from the NearArchive object retrieval driver program for longer than the time interval specified in the TIMEOUT initialization parameter. The request will be terminated.

Action: Consult any other available NearArchive system error messages for additional information regarding this condition. If no apparent cause for the timeout can be identified, contact your NearArchive customer support representative.

## **8.19 CICS Operator Interface Messages.**

### **OTS26000 - SELECT OPTION AND PRESS ENTER**

Explanation: This message is displayed on the operator interface menu selection screen OTSN260.

Action: Enter a valid option (numeric character in the range 1-5) corresponding to the processing required, and press enter.

### **OTS26002 - PLEASE ENTER VALID OPTION**

Explanation: No option has been entered in the option field on the CICS operator interface menu selection screen OTSN260 before pressing enter.

Action: Enter a valid option (numeric character in the range 1-5) corresponding to the processing required, and press enter.

### **OTS26003 - INVALID OPTION SELECTED - TRY AGAIN**

Explanation: An invalid option has been entered in the option field on the CICS operator interface menu selection screen OTSN260.

Action: Enter a valid option (numeric character in the range 1-5) corresponding to the processing required, and press enter.

### **OTS26004 - NEARARCHIVE CONTROL REGION NOT INITIALIZED**

Explanation: The CICS operator interface facility has been invoked but the NearArchive control region has not yet been initialized. Operator interface processing cannot be performed.

Action: Initialize the NearArchive CICS control region using NearArchive CICS transaction 'OT23', before invoking the operator interface facility.

### **OTS26005 - STATUS DISPLAY MAY ONLY BE INVOKED FROM CONTROL REGION**

Explanation: The operator interface facility has been invoked from a secondary NearArchive region. This facility is only available from within a NearArchive control region.

Action: Sign-on to the NearArchive control region, and invoke the operator interface facility from there.

### **OTS26006 - CORRUPT NEARARCHIVE ANCHOR BLOCK**

Explanation: A control block used for NearArchive online database access is corrupt. Processing of the operator interface facility cannot continue.

Action: Disable the NearArchive control region by using NearArchive transaction 'OT24'. Re-enable the control region using transaction 'OT23'. If this does not resolve the problem, contact your NearArchive product support representative.

### **OTS26008 - NEARARCHIVE ONLINE CONTROL PROCESS ENDED**

Explanation: Execution of the NearArchive operator interface procedure has been terminated normally after PF3 has been entered from the selection menu screen OTSN260.

Action: None. This is an information message.

### **OTS26009 – USER EXIT PROGRAM OTASUX02 NOT DEFINED TO CICS**

Explanation: A program entry for the user exit program OTASUX02 has not been defined to CICS.

Action: Ensure that the correct CICS table entries have been defined. Refer to section 2.8 for information on CICS table definitions.

### **OTS26010 – AUTHORIZATION TO PERFORM REQUESTED FUNCTION DENIED BY USER EXIT**

Explanation: A request to change the status of the NearArchive control region has been rejected by the user exit program OTASUX02.

Action: None. The requested action cannot be performed by the current user.

### **OTS26100 - ONLY ONE SELECTION ALLOWED - TRY AGAIN**

Explanation: More than one task has been selected from the reader task summary screen OTSN261 for processing. A maximum of one selection may be entered on this screen.

Action: Select a single task for processing, and press enter.

### **OTS26101 - NEARARCHIVE NOT INITIALIZED**

Explanation: The CICS operator interface facility has been invoked but the NearArchive control region has not yet been initialized. Operator interface processing cannot be performed.

Action: Initialize the NearArchive CICS control region using NearArchive CICS transaction 'OT23', before invoking the operator interface facility.

#### **OTS26102 - STATUS DISPLAY MAY ONLY BE INVOKED FROM CONTROL REGION**

Explanation: The operator interface facility has been invoked from a secondary NearArchive region. This facility is only available from within a NearArchive control region.

Action: Sign-on to the NearArchive control region, and invoke the operator interface facility from there.

#### **OTS26103 - STATUS ERROR - CORRUPT ANCHOR BLOCK**

Explanation: A control block used for NearArchive online database access is corrupt. Processing of the operator interface facility cannot continue.

Action: Disable the NearArchive control region by using NearArchive transaction 'OT24'. Re-enable the control region using transaction 'OT23'. If this does not resolve the problem, contact your NearArchive product support representative.

#### **OTS26104 - STATUS ERROR - CORRUPT REQUEST HEADER**

Explanation: A control block used for NearArchive online database access is corrupt. Processing of the operator interface facility cannot continue.

Action: Disable the NearArchive control region by using NearArchive transaction 'OT24'. Re-enable the control region using transaction 'OT23'. If this does not resolve the problem, contact your NearArchive product support representative.

#### **OTS26105 - INVALID KEY HIT - TRY AGAIN**

Explanation: An invalid program function or attention key has been pressed during operator interface processing.

Action: Retry the request, using the correct key.

#### **OTS26106 - NO FURTHER BACKWARD SCROLLING POSSIBLE**

Explanation: A request has been made to scroll backwards through the list of reader tasks on the reader task summary screen OTSN261, using the PF7 key, but the first reader task is already displayed.

Action: None. The current OTSN261 screen is re-displayed.

#### **OTS26107 - NO FURTHER FORWARD SCROLLING POSSIBLE**

Explanation: A request has been made to scroll forwards through the list of reader tasks on the reader task summary screen OTSN261, using the PF8 key, but the last reader task is already displayed.

Action: None. The current OTSN261 screen is re-displayed.

#### **OTS26109 - INVALID SELECTION - TRY AGAIN**

Explanation: An invalid selection character has been entered against a reader task on the summary display screen OTSN261. Only the selection characters 'S', 'P' and 'F' (upper or lower case) are permitted.

Action: Re-enter the request using the correct selection character.

#### **OTS26110 - TASK NOT FOUND - REQUEST IGNORED**

Explanation: An error has occurred attempting to locate a task.

Action: None. The current OTSN261 screen is re-displayed.

#### **OTS26111 - TASK ALREADY BEING TERMINATED**

Explanation: A purge request has been entered for a task which is already being terminated in response to an earlier purge request.

Action: None. The request is ignored.



### **OTS26112 - TASK FORCE TERMINATED**

Explanation: A task has been successfully force terminated in response to an 'F' selection.

Action: The OTSN261 screen is re-displayed. The terminated task will be shown with a status code of 'TM'.

### **OTS26113 - TASK CANNOT BE FORCED PRIOR TO NORMAL TERMINATION REQUEST**

Explanation: A request has been made to force purge a task, but no purge request has yet been made for that task. A purge request must be entered against a task, prior to issuing a force purge request for that task.

Action: None. The request is ignored.

### **OTS26114 - SELECTED TASK NOT ACTIVE - REQUEST IGNORED**

Explanation: A task has been selected for purge processing, but that task is not currently active.

Action: None. The request is ignored.

### **OTS26115 - TASK TERMINATED**

Explanation: A waiting task has been successfully terminated in response to a purge request.

Action: The OTSN261 screen is re-displayed. The terminated task will be shown with a status code of 'TM'.

### **OTS26116 - TASK TERMINATION IN PROGRESS**

Explanation: A purge request has been accepted for a busy task.

Action: The task will be terminated when all queued requests for that task have been processed. The task will have a status indicator of 'BYSH' on redisplay of screen OTSN261.

### **OTS26118 – AUTHORIZATION TO PERFORM REQUESTED FUNCTION DENIED BY USER EXIT**

Explanation: A request to change the status of the NearArchive control region has been rejected by the user exit program OTASUX02.

Action: None. The requested action cannot be performed by the current user.

### **OTS26120 - EIBRESP 'xxx' RECEIVED FOR FUNCTION X'yyy'**

Explanation: An unexpected CICS command failure occurred during operator interface processing. The response 'xxx' and function codes 'yyy' are as specified in the Exec Interface Block (EIB).

Action: Refer to CICS documentation for details of the function and response codes. Take appropriate corrective action. Contact your NearArchive support representative if there is no apparent reason for the error.

### **OTS26121 - ABEND xxxx RECEIVED DURING NEARARCHIVE OPERATOR INTERFACE PROCESSING**

Explanation: An abend with code 'xxxx' has occurred during CICS operator interface processing.

Action: Refer to CICS documentation for details of the abend and corrective action. Contact your NearArchive support representative if there is no apparent reason for the error.

### **OTS26200 - REQUESTED TASK NOT IN EXISTENCE**

Explanation: The task number entered in on the reader task detail display screen OTSN262 does not exist.

Action: None. The request is ignored.

### **OTS26201 - NEARARCHIVE NOT INITIALIZED**

Explanation: The CICS operator interface facility has been invoked but the NearArchive control region has not yet been

initialized. Operator interface processing cannot be performed.

Action: Initialize the NearArchive CICS control region using NearArchive CICS transaction 'OT23', before invoking the operator interface facility.

#### **OTS26202 - STATUS DISPLAY MAY ONLY BE INVOKED FROM CONTROL REGION**

Explanation: The operator interface facility has been invoked from a secondary NearArchive region. This facility is only available from within a NearArchive control region.

Action: Sign-on to the NearArchive control region, and invoke the operator interface facility from there.

#### **OTS26203 - STATUS ERROR - CORRUPT ANCHOR BLOCK**

Explanation: A control block used for NearArchive online database access is corrupt. Processing of the operator interface facility cannot continue.

Action: Disable the NearArchive control region by using NearArchive transaction 'OT24'. Re-enable the control region using transaction 'OT23'. If this does not resolve the problem, contact your NearArchive product support representative.

#### **OTS26204 - STATUS ERROR - CORRUPT REQUEST HEADER**

Explanation: A control block used for NearArchive online database access is corrupt. Processing of the operator interface facility cannot continue.

Action: Disable the NearArchive control region by using NearArchive transaction 'OT24'. Re-enable the control region using transaction 'OT23'. If this does not resolve the problem, contact your NearArchive product support representative.

#### **OTS26205 - INVALID KEY HIT - TRY AGAIN**

Explanation: An invalid program function or attention key has been pressed during operator interface processing.

Action: Retry the request, using the correct key.

#### **OTS26206 - NO FURTHER BACKWARD SCROLLING POSSIBLE**

Explanation: A request has been made to scroll backwards through the list of queued requests on the reader task detail screen OTSN262, using the PF7 key, but the first queued request is already displayed.

Action: None. The current OTSN262 screen is re-displayed.

#### **OTS26207 - NO FURTHER FORWARD SCROLLING POSSIBLE**

Explanation: A request has been made to scroll forwards through the list of queued request on the reader task detail screen OTSN262, using the PF8 key, but the last queued request is already displayed.

Action: None. The current OTSN262 screen is re-displayed.

#### **OTS26208 - STATUS ERROR - CORRUPT REQUEST ELEMENT**

Explanation: A control block used for NearArchive online database access is corrupt. Processing of the operator interface facility cannot continue.

Action: Disable the NearArchive control region by using NearArchive transaction 'OT24'. Re-enable the control region using transaction 'OT23'. If this does not resolve the problem, contact your NearArchive product support representative.

#### **OTS26209 - INVALID ENTRY - TRY AGAIN**

Explanation: An invalid task number has been entered on the reader task detail display screen OTSN262.

Action: Enter a valid numeric identifier in the range 1-255 and press enter.

### **OTS26210 - EIBRESP 'xxx' RECEIVED FOR FUNCTION X'yyy'**

Explanation: An unexpected CICS command failure occurred during operator interface processing. The response 'xxx' and function codes 'yyy' are as specified in the Exec Interface Block (EIB).

Action: Refer to CICS documentation for details of the function and response codes. Take appropriate corrective action. Contact your NearArchive support representative if there is no apparent reason for the error.

### **OTS26211 - ABEND xxxx RECEIVED DURING NEARARCHIVE OPERATOR INTERFACE PROCESSING**

Explanation: An abend with code 'xxxx' has occurred during CICS operator interface processing.

Action: Refer to CICS documentation for details of the abend and corrective action. Contact your NearArchive support representative if there is no apparent reason for the error.

### **OTS26301 - NEARARCHIVE NOT INITIALIZED**

Explanation: The CICS operator interface facility has been invoked but the NearArchive control region has not yet been initialized. Operator interface processing cannot be performed.

Action: Initialize the NearArchive CICS control region using NearArchive CICS transaction 'OT23', before invoking the operator interface facility.

### **OTS26302 - STATUS DISPLAY MAY ONLY BE INVOKED FROM CONTROL REGION**

Explanation: The operator interface facility has been invoked from a secondary NearArchive region. This facility is only available from within a NearArchive control region.

Action: Sign-on to the NearArchive control region, and invoke the operator interface facility from there.

### **OTS26303 - STATUS ERROR - CORRUPT ANCHOR BLOCK**

Explanation: A control block used for NearArchive online database access is corrupt. Processing of the operator interface facility cannot continue.

Action: Disable the NearArchive control region by using NearArchive transaction 'OT24'. Re-enable the control region using transaction 'OT23'. If this does not resolve the problem, contact your NearArchive product support representative.

### **OTS26304 - INVALID MAXQLEN ENTERED - TRY AGAIN**

Explanation: An invalid entry has been made in the MAXQLEN update field on screen OTSN263. The update request is rejected.

Action: Enter a numeric value in the range 1-255 and retry the request.

### **OTS26305 - INVALID MAXDRIVE ENTERED - TRY AGAIN**

Explanation: An invalid entry has been made in the MAXDRIVE update field on screen OTSN263. The update request is rejected.

Action: Enter a numeric value in the range 1-255 and retry the request.

### **OTS26309 - ATTACH ERROR nnn - UPDATING TERMINATED**

Explanation: An error has been received attempting to attach a reader sub-task when increasing the value of MAXDRIVE or MAXDISK. 'nnn' gives the decimal error code received from the ATTACH macro request.

Action: Refer to IBM OS/390 macro documentation for a description of the error code. Take the appropriate corrective action before repeating the request. If no apparent reason for the failure can be identified, contact your NearArchive product support representative.

### **OTS26310 - GETMAIN ERROR nnn - UPDATING TERMINATED**

Explanation: An error has been received allocating OS/390 storage when increasing the value of MAXDRIVE or MAXDISK. 'nnn' gives the decimal error code received from the GETMAIN macro request.

Action: Refer to IBM OS/390 macro documentation for a description of the error code. Take the appropriate corrective action before repeating the request. If necessary, increase the amount of OSCOR storage allocated to the NearArchive control region. If no apparent reason for the failure can be identified, contact your NearArchive product support representative.

### **OTS26311 - INTERNAL ERROR IN REDUCING MAXDRIVE**

Explanation: An internal error has occurred reducing the value of the MAXDRIVE initialization parameter.

Action: Contact your NearArchive product support representative.

### **OTS26312 - REQUESTED UPDATE COMPLETED SUCCESSFULLY**

Explanation: The requested modification to the NearArchive control region initialization parameters has been completed successfully.

Action: None. This is an information message.

### **OTS26313 - INVALID TSQDELETE ENTERED - TRY AGAIN**

Explanation: An invalid entry has been made in the TSQDELETE update field on screen OTSN263. The update request is rejected.

Action: Enter a numeric value in the range 0-1440 and retry the request.

### **OTS26314 - INVALID MAXDISK ENTERED - TRY AGAIN**

Explanation: An invalid entry has been made in the MAXDISK update field on screen OTSN263. The update request is rejected.

Action: Enter a numeric value in the range 1-255 and retry the request.

#### **OTS26315 - INVALID FUNCTION KEY PRESSED**

Explanation: An invalid program function key has been pressed during operator interface processing.

Action: Retry the request, using the correct key.

#### **OTS26316 - INVALID TAPEWAIT ENTERED - TRY AGAIN**

Explanation: An invalid entry has been made in the TAPEWAIT update field on screen OTSN263. The update request is rejected.

Action: Enter a numeric value in the range 0-1440 and retry the request.

#### **OTS26317 - INVALID TIMEOUT ENTERED - TRY AGAIN**

Explanation: An invalid entry has been made in the TIMEOUT update field on screen OTSN263. The update request is rejected.

Action: Enter a numeric value in the range 0-1440 and retry the request.

#### **OTS26320 - EIBRESP 'xxx' RECEIVED FOR FUNCTION X'yyy'**

Explanation: An unexpected CICS command failure occurred during operator interface processing. The response 'xxx' and function codes 'yyy' are as specified in the Exec Interface Block (EIB).

Action: Refer to CICS documentation for details of the function and response codes. Take appropriate corrective action. Contact your NearArchive support representative if there is no apparent reason for the error.

#### **OTS26321 - ABEND xxxx RECEIVED DURING NEARARCHIVE OPERATOR INTERFACE PROCESSING**

Explanation: An abend with code 'xxxx' has occurred during CICS operator interface processing.



Action: Refer to CICS documentation for details of the abend and corrective action. Contact your NearArchive support representative if there is no apparent reason for the error.

#### **OTS26401 - NEARARCHIVE NOT INITIALIZED - REQUEST REJECTED**

Explanation: The CICS operator command entry facility has been invoked but the NearArchive control region has not yet been initialized. Operator command entry processing cannot be performed.

Action: Initialize the NearArchive CICS control region using NearArchive CICS transaction 'OT23', before invoking the operator command entry facility.

#### **OTS26402 - COMMAND MAY ONLY BE INVOKED FROM CONTROL REGION - REQUEST REJECTED**

Explanation: The CICS operator command entry facility has been invoked but the NearArchive control region has not yet been initialized. Operator command entry processing cannot be performed.

Action: Initialize the NearArchive CICS control region using NearArchive CICS transaction 'OT23', before invoking the operator command entry facility.

#### **OTS26403 - STATUS ERROR - CORRUPT ANCHOR BLOCK**

Explanation: A control block used for NearArchive online database access is corrupt. Processing of the operator command entry facility cannot continue.

Action: Disable the NearArchive control region by using NearArchive transaction 'OT24'. Re-enable the control region using transaction 'OT23'. If this does not resolve the problem, contact your NearArchive product support representative.

#### **OTS26404 - INVALID MAXQLEN VALUE ENTERED - REQUEST REJECTED**

Explanation: An invalid value has been supplied in the MAXQLEN parameter of the NearArchive CICS operator interface SET command. The update request is rejected.

Action: Enter a numeric value in the range 1-255 for the MAXQLEN parameter, and re-enter the SET command. Refer to section 6.5.1 of the manual for a description of valid SET command parameters.

#### **OTS26405 - INVALID MAXDRIVE VALUE ENTERED - REQUEST REJECTED**

Explanation: An invalid value has been supplied in the MAXDRIVE parameter of the NearArchive CICS operator interface SET command. The update request is rejected.

Action: Enter a numeric value in the range 1-255 for the MAXDRIVE parameter, and re-enter the SET command. Refer to section 6.5.1 of the manual for a description of valid SET command parameters.

#### **OTS26406 - INVALID OPERAND - REQUEST REJECTED**

Explanation: An invalid keyword parameter value has been supplied on the NearArchive CICS operator interface SET command. The update request is rejected.

Action: Enter a valid keyword parameter, and re-enter the SET command. Refer to section 6.5.1 of the manual for a description of valid SET command parameters.

#### **OTS26407 - INVALID MAXDISK VALUE ENTERED - REQUEST REJECTED**

Explanation: An invalid value has been supplied in the MAXDISK parameter of the NearArchive CICS operator interface SET command. The update request is rejected.

Action: Enter a numeric value in the range 1-255 for the MAXDISK parameter, and re-enter the SET command. Refer to section 6.5.1 of the manual for a description of valid SET command parameters.

### **OTS26409 - ATTACH ERROR nnn - UPDATING TERMINATED**

Explanation: An error has been received attempting to attach a reader sub-task when increasing the value of MAXDRIVE or MAXDISK. 'nnn' gives the decimal error code received from the ATTACH macro request.

Action: Refer to IBM OS/390 macro documentation for a description of the error code. Take the appropriate corrective action before repeating the request. If no apparent reason for the failure can be identified, contact your NearArchive product support representative.

### **OTS26410 - GETMAIN ERROR nnn - UPDATING TERMINATED**

Explanation: An error has been received allocating OS/390 storage when increasing the value of MAXDRIVE or MAXDISK. 'nnn' gives the decimal error code received from the GETMAIN macro request.

Action: Refer to IBM OS/390 macro documentation for a description of the error code. Take the appropriate corrective action before repeating the request. If necessary, increase the amount of OSCOR storage allocated to the NearArchive control region. If no apparent reason for the failure can be identified, contact your NearArchive product support representative.

### **OTS26411 - INTERNAL ERROR IN REDUCING MAXDRIVE**

Explanation: An internal error has occurred reducing the value of the MAXDRIVE initialization parameter.

Action: Contact your NearArchive product support representative.

### **OTS26412 - REQUESTED UPDATE COMPLETED SUCCESSFULLY**

Explanation: The requested modification to the NearArchive control region initialization parameters has been completed successfully.

Action: None. This is an information message.

### **OTS26413 - COMMAND UNRECOGNIZED - TRY AGAIN**

Explanation: An invalid command has been entered using the NearArchive CICS operator command entry facility. The update request is rejected.

Action: Re-enter the command, using a valid command keyword. Refer to section 6.5.1 of the manual for a description of valid commands.

### **OTS26414 - MISSING OPERAND - REQUEST REJECTED**

Explanation: No keyword parameter value has been supplied on the NearArchive CICS operator interface SET command. The update request is rejected.

Action: Enter a valid keyword parameter, and re-enter the SET command. Refer to section 6.5.1 of the manual for a description of valid SET command parameters.

### **OTS26415 – USER EXIT PROGRAM OTASUX02 NOT DEFINED TO CICS**

Explanation: A program entry for the user exit program OTASUX02 has not been defined to CICS.

Action: Ensure that the correct CICS table entries have been defined. Refer to section 2.8 for information on CICS table definitions.

### **OTS26416 – AUTHORIZATION TO PERFORM REQUESTED FUNCTION DENIED BY USER EXIT**

Explanation: A request to change the status of the NearArchive control region has been rejected by the user exit program OTASUX02.

Action: None. The requested action cannot be performed by the current user.

### **OTS26420 - EIBRESP 'xxx' RECEIVED FOR FUNCTION X'yyy'**

Explanation: An unexpected CICS command failure occurred during operator interface processing. The response 'xxx' and

function codes 'yyy' are as specified in the Exec Interface Block (EIB).

Action: Refer to CICS documentation for details of the function and response codes. Take appropriate corrective action. Contact your NearArchive support representative if there is no apparent reason for the error.

#### **OTS26421 - ABEND xxxx RECEIVED DURING NEARARCHIVE OPERATOR INTERFACE PROCESSING**

Explanation: An abend with code 'xxxx' has occurred during CICS operator interface processing.

Action: Refer to CICS documentation for details of the abend and corrective action. Contact your NearArchive support representative if there is no apparent reason for the error.

## **8.20 CICS Housekeeping Processing Messages.**

#### **OTS27000 - nnnnn TEMPORARY STORAGE QUEUES DELETED**

Explanation: This messages is written to the CICS 'CSMT' transient data queue on completion of a temporary storage housekeeping cycle. 'nnnnn' gives the number of temporary storage queues deleted during this housekeeping cycle.

Action: None. This is an information message. The NearArchive CICS temporary storage housekeeping cycle is automatically initiated every 5 minutes, when enabled, and will delete all temporary storage queues containing retrieved objects, which have been in existence for longer than the retention period specified in the TSQDELETE control region parameter.

#### **OTS27001 - EIBRESP 'xxx' RECEIVED FOR FUNCTION X'yyy'**

Explanation: A CICS command failure occurred during control region housekeeping processing. The response 'xxx' and function codes 'yyy' are as specified in the Exec Interface Block (EIB).

Action: Refer to CICS documentation for details of the function and response codes. Take the appropriate corrective action. Contact your NearArchive support representative if there is no apparent reason for the error.

**OTS27002 - ABEND xxxx RECEIVED DURING NEARARCHIVE RETRIEVAL PROCESSING**

Explanation: An abend with code 'xxxx' has occurred during control region housekeeping processing.

Action: Refer to CICS documentation for details of the abend and corrective action. Contact your NearArchive support representative if there is no apparent reason for the error.

**OTS27003 - NEARARCHIVE NOT INITIALIZED - HOUSEKEEPING PROCESSING TERMINATED**

Explanation: A control region housekeeping cycle has been initiated, but the control region is now disabled.

Action: None. This is an information message. Control region housekeeping processing will be automatically re-initiated when the control region is re-enabled.

**OTS27004 - INVALID REQUEST ANCHOR BLOCK - HOUSEKEEPING PROCESSING TERMINATED**

Explanation: A control block used for NearArchive online database access is corrupt. Control region housekeeping processing cannot continue.

Action: Disable the NearArchive control region by using NearArchive transaction 'OT24'. Re-enable the control region using transaction 'OT23'. If this does not resolve the problem, contact your NearArchive product support representative.

## 8.21 Batch Retrieval Request Processor Messages.

---

### **OTS32002 - RC=xx, SC1=X'yyyy', SC2=X'zzzz' FROM SMF WRITE - SMF RECORDING DISABLED**

Explanation: A request to write an SMF record has been rejected by the NearArchive SMF processing component with the return and supplementary codes displayed in the message. SMF recording of object retrieval activity will be disabled until the database is disconnected.

If the return code = '14' an error was received from the SMF write request. In these circumstances, supplementary code 1 contains the error code from the SMFWTM macro.

Action: For return code '14', refer to IBM documentation on SMF for an explanation of the SMFWTM error code. Take the appropriate corrective action and stop and restart the reader task.

For other return codes, or if no reason can be found for the SMF write error, contact your NearArchive product support representative.

### **OTS35201 - ALLOC-RETRY aaaa - DSN=bbbbbbbbbbbbbbbbbbbbbbbbbbbbbb**

Explanation: NearArchive has attempted to allocate a tape drive but no device is currently available. NearArchive will wait for one minute before making another attempt.

'bbbbbb...bbbbbb' identifies the tape dataset that NearArchive is attempting to allocate. 'aaaa' gives the number of times that NearArchive will retry the allocation request.

The number of allocation retries to be performed by NearArchive may be specified in the application's database open call. A default value of 30 is used if no value is specified by the application.

Action: If possible, make a tape drive available for NearArchive allocation, by varying online any devices which are

currently offline. NearArchive will acquire any such device during allocation retry processing.

If no other devices are available, cancel another job in the system, or wait for another job to finish. Any tape drives allocated to the other job will become available when that job terminates, and may be acquired by NearArchive during allocation retry processing.

If no drive becomes available before the number of allocation retries has been exhausted, NearArchive will fail the application request with a dynamic allocation error code.

## 8.22 Database Initialization Messages.

---

### **OTS49001 - RETURN CODE 'xx' RECEIVED FROM ISPF 'yyyyyyy' REQUEST ACCESSING RESOURCE aaaaaaaa**

Explanation: Return code 'xx' has been received from the ISPF dialog manager during execution of ISPF request yyyyyyy.

Action: Refer to IBM documentation on ISPF dialog manager processing for details of the return code. Take the appropriate corrective action and retry the ISPF request.

### **OTS49002 - ERROR X'xxyy' RECEIVED aaaaaaaa RECORD bbbbbbbbbbbbbbbbbbbbbbbbbbbbbb IN DATASET 'cccccccccccccccccccccccccccccccccccc'**

Explanation: A VSAM I/O error has occurred when processing a NearArchive system dataset. Some or all of the following operands may appear:-

'xx' - the return code from the VSAM I/O request

'yy' - the reason code from the VSAM I/O request

'aaaaaaa' - the process being performed. This may take the following values:

'READING'

'WRITING'



'UPDATING'

'DELETING'

'bbbb...bbbb' - describes the record on which the operation is being performed.

'cccc..cccc' - the dataset name.

Processing is terminated.

Action: Refer to IBM documentation on VSAM macro processing for details of the displayed return and reason codes. Take the appropriate corrective action and retry the ISPF request.

### **OTS49003 - ERROR X'xxxx' RECEIVED FROM aaaaaaaaa REQUEST**

Explanation: Return code 'xxxx' has been received while processing an operating system service request.

'aaaaaaaa' - can take the following values:

PROGLOAD - an error has occurred attempting to dynamically load a NearArchive program module.

GETMAIN - an error has occurred trying to obtain virtual storage in the TSO region.

Action: Refer to the appropriate IBM documentation for a description of the return code. Take the appropriate corrective action and retry the ISPF request.

### **OTS49004 - ERROR 'xx' REASON X'yyyyzzzz' ALLOCATING FILE 'aaaaaaaa' DATASET 'bbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbb'**

Explanation: An error has occurred dynamically allocating dataset 'aaaaaaaa'. 'xx' gives the return code, and 'yyyy' and 'zzzz' the error and information codes respectively from the dynamic allocation request.

Action: Refer to IBM documentation on dynamic allocation processing for a description of the error and information codes. Take the appropriate corrective action and retry the ISPF request.

**OTS49006 - ERROR 'xx' REASON X'yyyyzzzz' DE-ALLOCATING FILE  
'aaaaaaaa'**

Explanation: An error has occurred dynamically de-allocating dataset 'aaaaaaaa'. 'xx' gives the return code, and 'yyyy' and 'zzzz' the error and information codes respectively from the dynamic allocation request.

Action: Refer to IBM documentation on dynamic allocation processing for a description of the error and information codes. Take the appropriate corrective action and retry the ISPF request.

**OTS49008 - 'aa' HAS NOT  
BEEN CONVERTED FOR USE WITH THIS RELEASE OF  
NEARARCHIVE**

Explanation: The dataset identified in the message is a pre-version 2.5 NearArchive database primary index dataset.

Action: The NearArchive ISPF database administration dialog is unable to process pre-version 2.5 databases. All such databases must be converted for use with this release of the product, using the database conversion procedure from within the NearArchive database administration dialog in TSO/ISPF. Refer to chapter 3 of this manual for a description of this procedure.

**OTS49009 - ERROR X'xxxx' RECEIVED OPENING/CLOSING FILE  
'yyyy....yyyy'**

Explanation: An error has been encountered attempting to open or close a VSAM dataset. 'xxxx' gives the hexadecimal value of the error code returned from VSAM open/close processing. 'yyyy....yyyy' gives the name of the VSAM dataset being processed.

Action: Refer to IBM documentation on VSAM macro processing for a description of the displayed open/close error code. Take the appropriate corrective action and retry the request.

## **OTS49010 - NEARARCHIVE DATABASE ADMINISTRATION PROCESSING ALREADY INVOKED IN THIS TSO SESSION**

**Explanation:** An attempt has been made for nested invocation of the NearArchive database administration process in TSO. Multiple invocation of this procedure from within a single TSO session is only permitted in ISPF split-screen mode.

**Action:** None. If the NearArchive database administration process is currently invoked in TSO, enter ISPF split-screen mode for invocation of a second concurrent database administration process.

## **8.23 General-purpose Component Messages.**

---

### **OTS94000 – IOB = xxxxxxxx xxxxxxxx xxxxxxxx xxxxxxxx xxxxxxxx**

**Explanation:** An error has occurred executing a tape I/O channel command. Message OTS94000 issues a hexadecimal display of the first 20 bytes in the IOB used for the request. This message is always accompanied by message OTS94001, which gives details of the channel command in error.

**Action:** Refer to IBM documentation for an explanation of the codes displayed in the message. Take the appropriate corrective action and repeat the failed process or rerun the failing job.

### **OTS94001 – CCW = xxxxxxxx xxxxxxxx**

**Explanation:** An error has occurred executing a tape I/O channel command. Message OTS94002 issues a hexadecimal display of the channel command word producing the error. This message is always accompanied by message OTS94000, which gives the contents of the first 20 bytes of the IOB after completion of the channel program.

**Action:** Refer to IBM documentation for an explanation of the codes displayed in the message. Take the appropriate corrective action and repeat the failed process or rerun the failing job.

## 8.24 SMF record format.

---

NearArchive will optionally record batch and/or CICS retrieval activity using the OS/390 System Management Facility (SMF). This option is controlled for each database via the NearArchive database administration dialog in TSO/ISPF. Refer to chapter 3 of this manual for information on enabling and disabling SMF logging of batch and CICS retrievals for a NearArchive database.

When enabled, NearArchive will create an SMF record at the following processing points:

- Batch retrieval processing: when the first record of an archived object has been retrieved and returned to the calling application, in response to a 'retrieve object record' request (request code '42').
- CICS retrieval processing: after all records in the archived object have been written to CICS temporary storage in response to a 'retrieve object' request (request code '43').

Each SMF record will contain the following sections:

1. *SMF header section.* This contains standard system information and timestamps.
2. *Record descriptor section.* This section contains information about the remaining sections in the record.
3. *Database section.* This section contains information relating to the database being accessed during the retrieval request.
4. *Request section.* This section contains information specific to the retrieval request itself.

### **8.24.1 Header section.**

Each SMF record will have a standard SMF header section of length 18 bytes.

<i>Offset</i>	<i>Length</i>	<i>Format</i>	<i>Description</i>
0	2	bin	Record length. This is the length of the entire SMF record, including this field.
2	2	bin	Segment descriptor - binary zeroes.
4	1	bin	System indicator - binary zeroes.
5	1	bin	Record type (equal to the value of the SMF record identifier specified defined for the database being processed).
6	4	bin	Time since midnight, in hundredths of a second, that the SMF record was written.
10	4	packed	Date record was written, in the format 0ccyddF, where F is the sign.
14	4	char	System identification (from CVT).

### **8.24.2 Record descriptor section.**

The SMF header section is immediately followed by the record descriptor section. It has a fixed length of 38 bytes.

<i>Offset</i>	<i>Length</i>	<i>Format</i>	<i>Description</i>
18	6	bin	Reserved.
24	4	bin	Offset of database section. This is the offset from the start of the SMF record of the database section of the record.
28	2	bin	Length of database section. This is the length in bytes of a single database section.
30	2	bin	The number of database sections present in the record (always = 1).
32	4	bin	Offset of request section. This is the offset from the start of the SMF record of the first request section in the record.
36	2	bin	Length of request section. This is the length in bytes of a single section. This will be 'k+70' bytes, where 'k' is the primary keylength for the NearArchive database being processed, as specified in the database section.
38	2	bin	The number of request sections present in the record (always = 1).
40	16	-	Reserved for future use.

### **8.24.3 Database section.**

The database section starts at the offset specified in the record descriptor section of the NearArchive SMF record. It gives information about the database being processed for object retrieval. It has a fixed length of 64 bytes. Only one database section will be present per SMF record.

<i>Offset</i>	<i>Length</i>	<i>Format</i>	<i>Description</i>
	<i>h</i>		
0	1	bin	CICS tape or disk reader task Number. This will be 0 for batch retrieval requests, and non-zero for CICS retrieval requests
1	1	bin	Processing flag (CICS requests only). This gives information about the CICS reader task at the time of SMF record creation. This byte will always contain low-values for batch retrieval requests.
		<i>Flag</i>	<i>Meaning</i>
		X'80'	A task purge request has been issued
		X'20'	Tape to be retained on drive after all outstanding requests processed (tape reader tasks only).
2	4	char	Identifier of NearArchive database being processed.
6	2	bin	Primary keylength of NearArchive database being processed.
8	44	char	Name of tape or disk dataset being processed.
52	2	bin	Sequence number within database of tape or disk dataset being processed.
54	4	bin	Segment number of disk dataset being processed (in range X'0000' to X'00FF')
56	8	-	Reserved for future use

#### **8.24.4 Request section.**

The request section starts at the offset specified in the record descriptor section of the SMF record. It gives information about the request processed by the NearArchive batch or CICS retrieval processor. It has a variable length of 'k+70' bytes, where 'k' is the primary keylength of the database being processed, as specified in the preceding database section. The SMF record will always contain one request section.

<i>Offset</i>	<i>Length</i>	<i>Format</i>	<i>Description</i>
	<i>h</i>		
0	8	bin	<p>Request start timestamp. This gives the date and time that the request was received by NearArchive. It has the format:</p> <p>bytes 0-3: time in hundredths of a second since midnight that request processing started.</p> <p>bytes 4-7: date that request processing started, in packed decimal format 0cyydddF, where F is the sign.</p>
8	8	bin	<p>CICS queue timestamp. This gives the date and time that the retrieval request was added to a CICS reader queue (CICS retrievals only –this timestamp will be equal to the request start timestamp for batch retrievals). It has the format:</p> <p>bytes 0-3: time in hundredths of a second since midnight that the request was intercepted.</p> <p>bytes 4-7: date that the request was intercepted, in packed decimal format 0cyydddF, where F is the sign.</p>
16	8	bin	<p>CICS reader start timestamp. This gives the date and time that the request was first processed by the CICS reader task (CICS retrievals only – this timestamp will be equal to the request start timestamp for batch retrievals). It has the format:</p>



<i>Offset</i>	<i>Length</i>	<i>Format</i>	<i>Description</i>
			bytes 0-3: time in hundredths of a second since midnight that the request was first processed.
			bytes 4-7: date that the request was first processed, in packed decimal format 0cyydddF, where F is the sign.
24	8	bin	Request end timestamp. This gives the date and time that processing for this request was completed by NearArchive. It has the following format:  bytes 0-3: time in hundredths of a second since midnight that request processing ended.  bytes 4-7: date that request processing ended, in packed decimal format 0cyydddF, where F is the sign.
32	2	char	Return code.
34	2	bin	Supplementary return code 1
36	2	bin	Supplementary return code 2
38	4	bin	Identifier of tape or disk block containing object.
42	2	bin	Number of tape or disk blocks used to hold object. A value of X'FFFF' in this field indicates a block count >= 65535.
44	1	char	Residency indicator ('T' = tape, 'D' = disk)
45	1	bin	Storage level (0-7)
46	8	char	Requester's jobname
54	8	char	Requester's user identifier. For CICS retrievals, this will be the identifier of the user issuing the object retrieval request. For batch

<i>Offset</i>	<i>Length</i>	<i>Format</i>	<i>Description</i>
			retrievals, this will be the user identifier assigned to the batch job.
62	8	char	Archive date of retrieved object (in format YYYYMMDD).
70	k	char	Primary key of retrieved object (where 'k' is the primary keylength as specified in the database section).

---

# APPENDICES

---

This page is intentionally left blank

## **APPENDIX A: Installation Verification Procedure.**

---

After installation of the product has been successfully completed, as detailed in Chapter 2, installation verification procedure (IVP) processing may be executed.

This will execute a number of supplied routines which will verify that each of the ASM for S/390 component functions (batch archival, batch and online retrieval) has been installed and is executing correctly.

The instructions given in this appendix should be followed step-by-step, and the results of each stage of the IVP compared with the supplied sample listings, as described in the IVP instructions.

## A.1 IVP Datasets.

---

The files and libraries necessary for running the ASM for S/390 IVP are supplied on the standard ASM for S/390 distribution tape and are unloaded during ASM for S/390 installation (see Chapter 2). A brief description of the files and libraries is given below:

*File 9:     IVP load module library*

This is a partitioned dataset containing load modules required for the batch and online IVP routines.

*File 10:   IVP JCL library*

This is a partitioned dataset containing jobs to run the batch IVP routines.

*File 11:   IVP test data file*

This is a partitioned dataset containing test data for running the IVP routines.

## A.2 IVP Database Initialization.

---

ASM for S/390 must be installed for batch and online processing, as described in Chapter 2 of this manual, prior to executing the IVP procedures. To define and initialize the NearArchive IVP database, perform the following tasks in the sequence presented below. Refer to chapter 3 for a full description of each of the database administration panels displayed during this procedure.

### A.2.1 Define and Initialize IVP Database.

- From the appropriate TSO/ISPF menu invoke the NearArchive database administration option. The NearArchive panel OTSN4000 will be displayed.

- On panel OTSN4000 enter the following details:

**Primary index name:** OTAS.V2R5.IVP.INDEX  
(any other primary index name may be used as required).

Select option '3' and press ENTER. Panel OTSN4300 will be displayed for primary index definition.

- On panel OTSN4300 specify the following base database definition details (defaults will be taken for unspecified fields):

**Length of primary key:** 9  
**NearArchive user SVC no:** SVC number allocated for NearArchive use during product installation

**HLQ for this database:** OTAS.V2R5.IVP (any other high-level qualifier may be used as required)

**Audit trail on:** Y/N

If an audit trail is required during IVP processing specify 'Y', otherwise specify 'N' (default). If 'Y' is selected, all updates to the NearArchive database are recorded in the database journal dataset. These may subsequently be processed by the NearArchive journal maintenance utility OTASP070.

Press ENTER when completed. Panel OTSN4301 will be displayed when all entries have been accepted.

- On panel OTSN4301 specify the storage level 0 definition details (default options will be taken for unspecified fields):

**Database Blocksize:** 1024  
**Maximum blockcount per volume:** 10  
**Unit name for scratch allocn:** specify the unit name to be used when allocating scratch cartridges during batch archival.

Press ENTER when completed. Panel OTSN4302 will be displayed when all entries have been validated successfully

- On panel OTSN4302 specify the AMS parameters for the primary index dataset (default options will be taken for unspecified fields):

**Volume serial number:** serial number of disk volume on which primary index and space management files are to be resident (leave blank if this is to be system controlled)  
**Unit of allocation:** CYL  
**Primary allocation value:** 1  
**Secondary allocation value:** 1

Press ENTER when completed. After all fields have been validated successfully, panel OTSN4303 will be displayed if the 'AUDIT trail on' field on panel OTSN4300 was set to 'Y', otherwise the main NearArchive database administration menu panel OTSN4000 will be displayed after successful completion of the database definition process.

- On panel OTSN4303 specify the AMS parameters for the journal dataset (default options will be taken for unspecified fields).

**Volume serial number:** serial number of disk volume on which journal file is to be resident (leave blank if this is to be system controlled)  
**Unit of allocation:** CYL  
**Primary allocation value:** 1  
**Secondary allocation value:** 1

Press ENTER when completed. Panel OTSN4000 will be displayed after successful completion of the database definition process.



### **A.2.3 Define Secondary Indexes.**

- On panel OTSN4000 select option '2' and press ENTER. Panel OTSN4200 will be displayed.
- On panel OTSN4200 select option '2' and press ENTER. Panel OTSN4220 will be displayed to allow definition of secondary indexes.
- On panel OTSN4220 specify the following:-

**Option:** 2 (Add new secondary index)  
**Secondary index no:** 1

Press ENTER when completed. Panel OTSN4221 will be displayed.

- On panel OTSN4221 specify the secondary index definition details as follows:-

Secondary key specification:

**Length:** 5  
**Start position in record:** 1

Automatic indexing controls:

**Record no within object:** 1 (The record from which the secondary key will be extracted during archival).

Press ENTER when completed. Panel OTSN4222 will be displayed.

- On panel OTSN4222 specify the AMS parameters for secondary index 1 (default options will be taken for unspecified fields).

**Volume serial number:** serial number of disk volume on which secondary index file #1 is to be resident (leave blank if this is to be system controlled)

**Unit of allocation:** CYL

**Primary allocation value:** 1

**Secondary allocation value:** 1

Press ENTER when completed. Panel OTSN4200 will be re-displayed after the definition of secondary index 1 has completed successfully.

- Repeat the above procedure in order to define secondary indexes 50 and 99. Enter the following values for each index:

Index 50

Panel OTSN4220:

**Secondary index no:** 50

Panel OTSN4221:

Secondary key specification:

**Length:** 8

**Start posn in record:** 1

Automatic indexing controls:

**Record no within object:** 0

Index 99

Panel OTSN4220:

**Secondary index no:** 99

Panel OTSN4221:

Secondary key specification:

**Length:** 11

**Start posn in record:** 6

Automatic indexing controls:

**Record no within object:** ALL

- After the above definitions have been entered successfully, press PF3 to return from panel OTSN4200 to panel OTSN4000.
- On panel OTSN4000, press PF3 to return to the primary TSO/ISPF panel.
- All IVP database definition processing is now complete.

## A.3 CICS Definitions.

---

- To verify correct operation of the NearArchive online retrieval component, a CICS system should be initialized as a NearArchive control region (see Chapter 2 for details on setting up this region). The following OTASS100 parameters may be used:

MAXDRIVE = 2  
MAXQLEN = 2  
MAXDISK = 2  
TSQDELETE = 5  
TAPEWAIT = 0  
TIMEOUT=0

- The IVP load library should be available in the DFHRPL concatenation for CICS testing, or the IVP modules can be copied from the IVP load library to another library, already available in the DFHRPL.
- Add the following CICS PCT, PPT and FCT entries in the NearArchive CICS control region.

### **PPT entries:**

OTASD220	DFHPPT	TYPE=ENTRY, PROGRAM=OTASD220, LANG=ASSEMBLER
OTASM20	DFHPPT	TYPE=ENTRY, MAPSET=OTASM20

### **PCT entry:**

OIVP	DFHPCT	TYPE=ENTRY, TRANSID=OIVP, PROGRAM=OTASD220, *
		DTIMOUT=NO

Equivalent definitions may be added using CICS RDO.

### **FCT entries:**

The database identifier used during online IVP processing is 'TEST'. This should be used when generating the names of the IVP file control table entries. The following entries will be required:

OT01TEST - Secondary Index 1  
OT50TEST - Secondary Index 50  
OT99TEST - Secondary Index 99  
OTASTEST - NearArchive database

OSM0TEST - NearArchive Space Management dataset  
OJRNTTEST - NearArchive Journal dataset (if audit trail enabled  
during IVP database definition).

Refer to section 2.8.4 for a full description of the definitions required for these file control table entries.

Note that the high-level qualifier used to generated system dataset names for the IVP procedure is that entered during IVP database definition processing, as specified in section A.2.1.

In addition, update access will be required to files OTASTEST and OJRNTTEST, if archived objects are to be deleted during online IVP processing.

## **A.4 Running the Batch IVP.**

---

A series of eight batch jobs has been supplied to verify the installation of ASM for S/390. Each should be submitted in sequence, after the previous job has completed successfully.

Before submitting each job, update the JCL as follows:

- add a valid JOB card.
- amend the STEPLIB DD card to specify the product and IVP load libraries, as unloaded from the distribution tape during the product installation process.
- amend the OTASV100 DD card to specify the name of the primary index dataset defined during IVP database initialization processing (see A.2.1 above).
- where necessary, amend the SYSIN DD card to update the name of the corresponding IVP test dataset, as unloaded from the distribution tape during the product installation process.

After each job has completed, check that the call audit reports are identical to the sample reports given in figures A.1 to A.8.

### **A.4.1 The batch archival IVP.**

The first three jobs archive files of test data to the NearArchive database. Explicit secondary index calls are included in the each job for secondary index 50 which is defined with no automatic indexing. Secondary indexes 1 and 99 will be built automatically during archival of the test data.

A brief description of each job follows:-

**OTASD211** - This job archives IVP data file #1 to the test database using standard NearArchive calls via the new application program interface. The file contains variable length records.

**OTASD212** - This job archives IVP data file #2 to the test database using standard NearArchive calls via the old application program interface. The file contains fixed length records. A second tape dataset (sequence 2) will be created during this job.

**OTASD213** - This job archives IVP data file #3 to the test database using standard NearArchive calls via the new application program interface. The file contains fixed length records.

#### **A.4.2 The batch retrieval IVP.**

The following jobs have been supplied to verify NearArchive batch retrieval processing. Each job demonstrates a different method of retrieval.

**OTASD214** - This job retrieves archived data from the NearArchive database using direct retrieval requests.

**OTASD215** - This job retrieves archived data from the NearArchive database using sequential retrieval requests. All objects in the database are retrieved. After the last item in the database has been retrieved, a further retrieval request is issued which will cause the first record in the database to be re-read.

**OTASD216** - This job deletes a data object from the NearArchive database.

**OTASD217** - This job issues NearArchive primary database query requests. Three query response entries are returned on each request and the four possible sort retrieval request sequences are demonstrated.

**OTASD218** - This job issues NearArchive secondary index 01 query requests. Three query response entries are returned on each request and the four possible sort retrieval request sequences are demonstrated.

## A.5 Running the online retrieval IVP.

---

The online IVP routines are now ready for execution. Use the call audit reports produced from the batch archival (or retrieval) IVP to identify the primary and secondary keys of archived data in the NearArchive IVP database.

Sign on to the NearArchive CICS control region, and initialize the region by running transaction OT23. Message OTS23001 should be displayed on the terminal, indicating that the CICS control region has been successfully initialized. Ensure that the OTASTEST primary index dataset is open and enabled prior to running transaction OT23.

To run the NearArchive online IVP routines, enter transaction identifier 'OIVP' on the terminal. A NearArchive IVP data entry screen will be displayed. Data may then be retrieved by primary or secondary key.

To select data by primary key, simply type the key in the 'primary key' field and press 'Enter'. Any primary key from the call audit report may be selected.

To select by secondary index key, the 'secondary key' field and the 'sec index no' field should contain the secondary key and secondary index number respectively. The secondary indexes defined during the IVP process are as follows:-

Secondary Index 01:	Key length	-	5
	Start position	-	1
	Record number	-	1

Secondary Index 50:	Key length	-	8
	Start position	-	1
	Record number	-	0

Secondary Index 99:	Key length	-	11
	Start position	-	6
	Record number	-	ALL

The following secondary keys may be used during the IVP process. If required, secondary keys for each secondary index may be determined from the call audit report, using the above index definitions. Any of these may be selected for retrieval.

**Secondary index 01:**

'DATA'

**Secondary index 50:**

'SEC50#01 '  
'SEC50#05 '

**Secondary index 99:**

'SEGMENT #1 '  
'SEGMENT #2 '  
'SEGMENT #3 '  
etc...  
'SEGMENT #11 '  
'SEGMENT #12 '  
'SEGMENT #13 '

The IVP routines will display a screen giving the primary key and archive date of each archived object which meets the selection criteria.

Select any one item of archived data for retrieval, by entering an 'S' (upper or lower case) in the selection column next to the chosen date. The IVP will reject any invalid or multiple selection. The selected data will now be retrieved from the NearArchive database, and the first 80 bytes of each data record for that archived object will be displayed on the CICS terminal.

Check that the number and content of retrieved records is identical to the corresponding entries for the selected key and archive date in the relevant batch call audit report.

Sign on to a second terminal in the NearArchive CICS control region, and perform single and multiple simultaneous retrievals from one or both tape datasets to verify that online retrieval is functioning correctly.

Any item of archived data may be deleted from the NearArchive database by entering a 'D' (upper or lower case) in the selection column next to the chosen date. The screen will be refreshed to remove the deleted item.

To return from one IVP screen to the previous screen, press 'PF3'.

When all the above tasks have been performed successfully, the NearArchive IVP process is complete.



## OTASD210

## INSTALLATION VERIFICATION PROCEDURE - CALL AUDIT REPORT

REQ CDE	RET CDE	SUPP CODE1	SUPP CODE2	D/B NAME	KEY	DATE	REC LEN	RECORD AREA
40	00	x0000	x0000				1	TEST
44	00	x0000	x0000	TEST	123456789	19980401	80	DATA SEGMENT #1 KEY '123456789' DATE 01/04/1998'.....
46	00	x0000	x0000	TEST	123456789	19980401	0	SEC50#01
44	00	x0000	x0000	TEST	123456789	19980401	90	DATA SEGMENT #2 KEY '123456789' DATE '01/04/1998'.....
44	00	x0000	x0000	TEST	123456789	19980401	100	DATA SEGMENT #3 KEY '123456789' DATE '01/04/1998'.....
44	00	x0000	x0000	TEST	123456789	19980401	110	DATA SEGMENT #4 KEY '123456789' DATE '01/04/1998'.....
44	00	x0000	x0000	TEST	123456789	19980401	120	DATA SEGMENT #5 KEY '123456789' DATE '01/04/1998'.....
46	00	x0000	x0000	TEST			0	SEC50#05
44	00	x0000	x0000	TEST	123456789	19980401	130	DATA SEGMENT #6 KEY '123456789' DATE '01/04/1998'.....
44	00	x0000	x0000	TEST	123456789	19980401	140	DATA SEGMENT #7 KEY '123456789' DATE '01/04/1998'.....
44	00	x0000	x0000	TEST	123456789	19980401	150	DATA SEGMENT #8 KEY '123456789' DATE '01/04/1998'.....
44	00	x0000	x0000	TEST	123456789	19980401	160	DATA SEGMENT #9 KEY '123456789' DATE '01/04/1998'.....
44	00	x0000	x0000	TEST	123456789	19980401	170	DATA SEGMENT #10KEY '123456789' DATE '01/04/1998'.....
44	00	x0000	x0000	TEST	123456789	19980401	180	DATA SEGMENT #11KEY '123456789' DATE '01/04/1998'.....
44	00	x0000	x0000	TEST	123456789	19980401	190	DATA SEGMENT #12KEY '123456789' DATE '01/04/1998'.....
44	00	x0000	x0000	TEST	123456789	19980401	200	DATA SEGMENT #13KEY '123456789' DATE '01/04/1998'.....
48	00	x0000	x0000	TEST			16448	
44	00	x0000	x0000	TEST	123456789	20010821	210	DATA SEGMENT #1 KEY '123456789' DATE '21/08/2001'.....
46	00	x0000	x0000	TEST			0	SEC50#01
44	00	x0000	x0000	TEST	123456789	20010821	220	DATA SEGMENT #2 KEY '123456789' DATE '21/08/2001'.....
44	00	x0000	x0000	TEST	123456789	20010821	128	DATA SEGMENT #3 KEY '123456789' DATE '21/08/2001'.....
44	00	x0000	x0000	TEST	123456789	20010821	128	DATA SEGMENT #4 KEY '123456789' DATE '21/08/2001'.....
44	00	x0000	x0000	TEST	123456789	20010821	128	DATA SEGMENT #5 KEY '123456789' DATE '21/08/2001'.....
46	00	x0000	x0000	TEST			0	SEC50#05
44	00	x0000	x0000	TEST	123456789	20010821	128	DATA SEGMENT #6 KEY '123456789' DATE '21/08/2001'.....

Fig. A1 - job OTASD211

## OTASD210

## INSTALLATION VERIFICATION PROCEDURE - CALL AUDIT REPORT

REQ CDE	RET CDE	SUPP CODE1	SUPP CODE2	D/B NAME	KEY	DATE	REC LEN	RECORD AREA
44	00	x0000	x0000	TEST	123456789	20010821	128	DATA SEGMENT #7 KEY '123456789' DATE '21/08/2001'.....
44	00	x0000	x0000	TEST	123456789	20010821	128	DATA SEGMENT #8 KEY '123456789' DATE '21/08/2001'.....
44	00	x0000	x0000	TEST	123456789	20010821	128	DATA SEGMENT #9 KEY '123456789' DATE '21/08/2001'.....
44	00	x0000	x0000	TEST	123456789	20010821	128	DATA SEGMENT #10KEY '123456789' DATE '21/08/2001'.....
44	00	x0000	x0000	TEST	123456789	20010821	128	DATA SEGMENT #11KEY '123456789' DATE '21/08/2001'.....
44	00	x0000	x0000	TEST	123456789	20010821	128	DATA SEGMENT #12KEY '123456789' DATE '21/08/2001'.....
48	00	x0000	x0000	TEST			16448	
44	00	x0000	x0000	TEST	123456789	19980930	128	DATA SEGMENT #1 KEY '123456789' DATE '30/09/1998'.....
46	00	x0000	x0000	TEST			0	SEC50#01
44	00	x0000	x0000	TEST	123456789	19980930	128	DATA SEGMENT #2 KEY '123456789' DATE '30/09/1998'.....
44	00	x0000	x0000	TEST	123456789	19980930	128	DATA SEGMENT #3 KEY '123456789' DATE '30/09/1998'.....
44	00	x0000	x0000	TEST	123456789	19980930	128	DATA SEGMENT #4 KEY '123456789' DATE '30/09/1998'.....
44	00	x0000	x0000	TEST	123456789	19980930	128	DATA SEGMENT #5 KEY '123456789' DATE '30/09/1998'.....
46	00	x0000	x0000	TEST			0	SEC50#05
44	00	x0000	x0000	TEST	123456789	19980930	128	DATA SEGMENT #6 KEY '123456789' DATE '30/09/1998'.....
44	00	x0000	x0000	TEST	123456789	19981020	128	DATA SEGMENT #1 KEY '123456789' DATE '20/10/1998'.....
46	00	x0000	x0000	TEST			0	SEC50#01
44	00	x0000	x0000	TEST	123456789	19981020	128	DATA SEGMENT #2 KEY '123456789' DATE '20/10/1998'.....
44	00	x0000	x0000	TEST	123456789	19981120	128	DATA SEGMENT #1 KEY '123456789' DATE '20/11/1998'.....
46	00	x0000	x0000	TEST			0	SEC50#01
44	00	x0000	x0000	TEST	123456789	19981120	128	DATA SEGMENT #2 KEY '123456789' DATE '20/11/1998'.....
44	00	x0000	x0000	TEST	133456789	19980526	128	DATA SEGMENT #1 KEY '133456789' DATE '26/05/1998'.....
46	00	x0000	x0000	TEST			0	SEC50#01
44	00	x0000	x0000	TEST	223456789	19980605	128	DATA SEGMENT #1 KEY '223456789' DATE '05/06/1998'.....
46	00	x0000	x0000	TEST			0	SEC50#01

Fig. A1 - job OTASD211

## OTASD210

## INSTALLATION VERIFICATION PROCEDURE - CALL AUDIT REPORT

REQ CDE	RET CDE	SUPP CODE1	SUPP CODE2	D/B NAME	KEY	DATE	REC LEN	RECORD AREA
44	00	x0000	x0000	TEST	223456789	19980605	128	DATA SEGMENT #2 KEY '223456789' DATE '05/06/1998'.....
44	00	x0000	x0000	TEST	223456789	19980708	128	DATA SEGMENT #1 KEY '223456789' DATE '08/07/1998'.....
46	00	x0000	x0000	TEST			0	SEC50#01
44	00	x0000	x0000	TEST	223456789	19980708	128	DATA SEGMENT #2 KEY '223456789' DATE '08/07/1998'.....
44	00	x0000	x0000	TEST	223456789	19980708	128	DATA SEGMENT #3 KEY '223456789' DATE '08/07/1998'.....
44	00	x0000	x0000	TEST	223456789	19980708	128	DATA SEGMENT #4 KEY '223456789' DATE '08/07/1998'.....
44	00	x0000	x0000	TEST	233456789	19981012	128	DATA SEGMENT #1 KEY '233456789' DATE '12/10/1998'.....
46	00	x0000	x0000	TEST			0	SEC50#01
44	00	x0000	x0000	TEST	233456789	19981012	128	DATA SEGMENT #2 KEY '233456789' DATE '12/10/1998'.....
44	00	x0000	x0000	TEST	233456789	19981012	128	DATA SEGMENT #3 KEY '233456789' DATE '12/10/1998'.....
44	00	x0000	x0000	TEST	233456789	19980811	128	DATA SEGMENT #1 KEY '233456789' DATE '11/08/1998'.....
46	00	x0000	x0000	TEST			0	SEC50#01
49	00	x0000	x0000	TEST			0	

Fig. A1 - job OTASD211

OTASD210

## INSTALLATION VERIFICATION PROCEDURE - CALL AUDIT REPORT

REQ CDE	RET CDE	SUPP CODE1	SUPP CODE2	D/B NAME	KEY	DATE	REC LEN	RECORD AREA
10	00	x0000	x0000				16448	
13	00	x0000	x0000		123456789	930929	128	DATA SEGMENT #1 KEY '123456789' DATE '29/09/93'.....
13	00	x0000	x0000		123456789	930929	128	DATA SEGMENT #2 KEY '123456789' DATE '29/09/93'.....
13	00	x0000	x0000		143456789	930604	128	DATA SEGMENT #1 KEY '143456789' DATE '04/06/93'.....
14	00	x0000	x0000				0	SEC50#01
13	00	x0000	x0000		143456789	930604	128	DATA SEGMENT #2 KEY '143456789' DATE '04/06/93'.....
13	00	x0000	x0000		143456789	930709	128	DATA SEGMENT #1 KEY '143456789' DATE '09/07/93'.....
14	00	x0000	x0000				0	SEC50#01
13	00	x0000	x0000		143456789	930709	128	DATA SEGMENT #2 KEY '143456789' DATE '09/07/93'.....
13	00	x0000	x0000		143456789	930709	128	DATA SEGMENT #3 KEY '143456789' DATE '09/07/93'.....
13	00	x0000	x0000		153456789	930608	128	DATA SEGMENT #1 KEY '153456789' DATE '08/06/93'.....
14	00	x0000	x0000				0	SEC50#01
13	00	x0000	x0000		153456789	930608	128	DATA SEGMENT #2 KEY '153456789' DATE '08/06/93'.....
13	00	x0000	x0000		233456789	930110	128	DATA SEGMENT #1 KEY '233456789' DATE '10/01/93'.....
14	00	x0000	x0000				0	SEC50#01
16	00	x0000	x0000				16448	
13	00	x0000	x0000		233456789	930220	128	DATA SEGMENT #1 KEY '233456789' DATE '20/02/93'.....
14	00	x0000	x0000				0	SEC50#01
13	00	x0000	x0000		233456789	930220	128	DATA SEGMENT #2 KEY '233456789' DATE '20/02/93'.....
13	00	x0000	x0000		233456789	930220	128	DATA SEGMENT #3 KEY '233456789' DATE '20/02/93'.....
13	00	x0000	x0000		233456789	930220	128	DATA SEGMENT #4 KEY '233456789' DATE '20/02/93'.....
15	00	x0000	x0000				16448	

Fig. A2 - job OTASD212

OTASD210

## INSTALLATION VERIFICATION PROCEDURE - CALL AUDIT REPORT

REQ CDE	RET CDE	SUPP CODE1	SUPP CODE2	D/B NAME	KEY	DATE	REC LEN	RECORD AREA
40	00	x0000	x0000				1	TEST
44	00	x0000	x0000	TEST	123456789	19991231	128	DATA SEGMENT #1 KEY '123456789' DATE '31/12/1999'.....
44	00	x0000	x0000	TEST	123456789	19991231	128	DATA SEGMENT #2 KEY '123456789' DATE '31/12/1999'.....
44	00	x0000	x0000	TEST	123456789	19991231	128	DATA SEGMENT #3 KEY '123456789' DATE '31/12/1999'.....
44	00	x0000	x0000	TEST	123456789	20000101	128	DATA SEGMENT #1 KEY '123456789' DATE '01/01/2000'.....
44	00	x0000	x0000	TEST	123456789	20000101	128	DATA SEGMENT #2 KEY '123456789' DATE '01/01/2000'.....
44	00	x0000	x0000	TEST	123456789	20020310	128	DATA SEGMENT #1 KEY '123456789' DATE '10/03/2002'.....
44	00	x0000	x0000	TEST	123456789	20020310	128	DATA SEGMENT #2 KEY '123456789' DATE '10/03/2002'.....
44	00	x0000	x0000	TEST	143456789	19980414	128	DATA SEGMENT #1 KEY '143456789' DATE '14/04/1998'.....
44	00	x0000	x0000	TEST	143456789	19980414	128	DATA SEGMENT #2 KEY '143456789' DATE '14/04/1998'.....
44	00	x0000	x0000	TEST	323456789	19980130	128	DATA SEGMENT #1 KEY '323456789' DATE '30/01/1998'.....
44	00	x0000	x0000	TEST	323456789	19980130	128	DATA SEGMENT #2 KEY '323456789' DATE '30/01/1998'.....
44	00	x0000	x0000	TEST	323456789	19980130	128	DATA SEGMENT #3 KEY '323456789' DATE '30/01/1998'.....
44	00	x0000	x0000	TEST	323456790	19980130	128	DATA SEGMENT #1 KEY '323456790' DATE '30/01/1998'.....
44	00	x0000	x0000	TEST	323456790	19980130	128	DATA SEGMENT #2 KEY '323456790' DATE '30/01/1998'.....
44	00	x0000	x0000	TEST	323456790	19980130	128	DATA SEGMENT #3 KEY '323456790' DATE '30/01/1998'.....
44	00	x0000	x0000	TEST	323456791	19980130	128	DATA SEGMENT #1 KEY '323456791' DATE '30/01/1998'.....
44	00	x0000	x0000	TEST	323456791	19980130	128	DATA SEGMENT #2 KEY '323456791' DATE '30/01/1998'.....
44	00	x0000	x0000	TEST	323456791	19980130	128	DATA SEGMENT #3 KEY '323456791' DATE '30/01/1998'.....
44	00	x0000	x0000	TEST	323456792	19980130	128	DATA SEGMENT #1 KEY '323456792' DATE '30/01/1998'.....
44	00	x0000	x0000	TEST	323456792	19980130	128	DATA SEGMENT #2 KEY '323456792' DATE '30/01/1998'.....
44	00	x0000	x0000	TEST	323456792	19980130	128	DATA SEGMENT #3 KEY '323456792' DATE '30/01/1998'.....
44	00	x0000	x0000	TEST	323456793	19980130	128	DATA SEGMENT #1 KEY '323456793' DATE '30/01/1998'.....
44	00	x0000	x0000	TEST	323456793	19980130	128	DATA SEGMENT #2 KEY '323456793' DATE '30/01/1998'.....
44	00	x0000	x0000	TEST	323456793	19980130	128	DATA SEGMENT #3 KEY '323456793' DATE '30/01/1998'.....

Fig. A3 - job OTASD213

OTASD210

## INSTALLATION VERIFICATION PROCEDURE - CALL AUDIT REPORT

REQ CDE	RET CDE	SUPP CODE1	SUPP CODE2	D/B NAME	KEY	DATE	REC LEN	RECORD AREA
44	00	x0000	x0000	TEST	323456794	19980130	128	DATA SEGMENT #1 KEY '323456794' DATE '30/01/1998'.....
44	00	x0000	x0000	TEST	323456794	19980130	128	DATA SEGMENT #2 KEY '323456794' DATE '30/01/1998'.....
44	00	x0000	x0000	TEST	323456794	19980130	128	DATA SEGMENT #3 KEY '323456794' DATE '30/01/1998'.....
44	00	x0000	x0000	TEST	323456795	19980130	128	DATA SEGMENT #1 KEY '323456795' DATE '30/01/1998'.....
44	00	x0000	x0000	TEST	323456795	19980130	128	DATA SEGMENT #2 KEY '323456795' DATE '30/01/1998'.....
44	00	x0000	x0000	TEST	323456795	19980130	128	DATA SEGMENT #3 KEY '323456795' DATE '30/01/1998'.....
44	00	x0000	x0000	TEST	323456796	19980130	128	DATA SEGMENT #1 KEY '323456796' DATE '30/01/1998'.....
44	00	x0000	x0000	TEST	323456796	19980130	128	DATA SEGMENT #2 KEY '323456796' DATE '30/01/1998'.....
44	00	x0000	x0000	TEST	323456796	19980130	128	DATA SEGMENT #3 KEY '323456796' DATE '30/01/1998'.....
44	00	x0000	x0000	TEST	323456797	19980130	128	DATA SEGMENT #1 KEY '323456797' DATE '30/01/1998'.....
44	00	x0000	x0000	TEST	323456797	19980130	128	DATA SEGMENT #2 KEY '323456797' DATE '30/01/1998'.....
44	00	x0000	x0000	TEST	323456797	19980130	128	DATA SEGMENT #3 KEY '323456797' DATE '30/01/1998'.....
49	00	x0000	x0000	TEST			0	

Fig. A3 - job OTASD213

OTASD210

## INSTALLATION VERIFICATION PROCEDURE - CALL AUDIT REPORT

REQ CDE	RET CDE	SUPP CODE1	SUPP CODE2	D/B NAME	KEY	DATE	REC LEN	RECORD AREA
40	00	x0000	x0000				1	TEST
42	00	x0000	x0000	TEST	123456789	19980401	80	DATA SEGMENT #1 KEY '123456789' DATE '01/04/1998'.....
42	00	x0000	x0000	TEST	123456789	20010821	210	DATA SEGMENT #1 KEY '123456789' DATE '21/08/2001'.....
42	00	x0000	x0000	TEST	123456789	19980930	128	DATA SEGMENT #1 KEY '123456789' DATE '30/09/1998'.....
42	00	x0000	x0000	TEST	123456789	19981020	128	DATA SEGMENT #1 KEY '123456789' DATE '20/10/1998'.....
42	00	x0000	x0000	TEST	123456789	19981120	128	DATA SEGMENT #1 KEY '123456789' DATE '20/11/1998'.....
42	00	x0000	x0000	TEST	133456789	19980526	128	DATA SEGMENT #1 KEY '133456789' DATE '26/05/1998'.....
42	00	x0000	x0000	TEST	223456789	19980605	128	DATA SEGMENT #1 KEY '223456789' DATE '05/06/1998'.....
42	00	x0000	x0000	TEST	223456789	19980708	128	DATA SEGMENT #1 KEY '223456789' DATE '08/07/1998'.....
42	00	x0000	x0000	TEST	233456789	19981012	128	DATA SEGMENT #1 KEY '233456789' DATE '12/10/1998'.....
42	00	x0000	x0000	TEST	233456789	19980811	128	DATA SEGMENT #1 KEY '233456789' DATE '11/08/1998'.....
42	00	x0000	x0000	TEST	123456789	19930929	128	DATA SEGMENT #1 KEY '123456789' DATE '29/09/93'.....
42	00	x0000	x0000	TEST	143456789	19930604	128	DATA SEGMENT #1 KEY '143456789' DATE '04/06/93'.....
42	00	x0000	x0000	TEST	143456789	19930709	128	DATA SEGMENT #1 KEY '143456789' DATE '09/07/93'.....
42	00	x0000	x0000	TEST	153456789	19930608	128	DATA SEGMENT #1 KEY '153456789' DATE '08/06/93'.....
42	00	x0000	x0000	TEST	233456789	19930110	128	DATA SEGMENT #1 KEY '233456789' DATE '10/01/93'.....
42	00	x0000	x0000	TEST	233456789	19930220	128	DATA SEGMENT #1 KEY '233456789' DATE '20/02/93'.....
42	00	x0000	x0000	TEST	123456789	19991231	128	DATA SEGMENT #1 KEY '123456789' DATE '31/12/1999'.....
42	00	x0000	x0000	TEST	123456789	20000101	128	DATA SEGMENT #1 KEY '123456789' DATE '01/01/2000'.....
42	00	x0000	x0000	TEST	123456789	20020310	128	DATA SEGMENT #1 KEY '123456789' DATE '10/03/2002'.....
42	00	x0000	x0000	TEST	323456790	19980130	128	DATA SEGMENT #1 KEY '323456790' DATE '30/01/1998'.....
42	00	x0000	x0000	TEST	323456797	19980130	128	DATA SEGMENT #1 KEY '323456797' DATE '30/01/1998'.....
49	00	x0000	x0000	TEST			0	

Fig. A4 - job OTASD214

OTASD210

## INSTALLATION VERIFICATION PROCEDURE - CALL AUDIT REPORT

REQ CDE	RET CDE	SUPP CODE1	SUPP CODE2	D/B NAME	KEY	DATE	REC LEN	RECORD AREA
40	00	x0000	x0000				1	TEST
42	00	x0000	x0000	TEST	123456789	19980401	80	DATA SEGMENT #1 KEY '123456789' DATE '01/04/1998'.....
42	00	x0000	x0000	TEST	123456789	19980401	90	DATA SEGMENT #2 KEY '123456789' DATE '01/04/1998'.....
42	00	x0000	x0000	TEST	123456789	19980401	100	DATA SEGMENT #3 KEY '123456789' DATE '01/04/1998'.....
42	00	x0000	x0000	TEST	123456789	19980401	110	DATA SEGMENT #4 KEY '123456789' DATE '01/04/1998'.....
42	00	x0000	x0000	TEST	123456789	19980401	120	DATA SEGMENT #5 KEY '123456789' DATE '01/04/1998'.....
42	00	x0000	x0000	TEST	123456789	19980401	130	DATA SEGMENT #6 KEY '123456789' DATE '01/04/1998'.....
42	00	x0000	x0000	TEST	123456789	19980401	140	DATA SEGMENT #7 KEY '123456789' DATE '01/04/1998'.....
42	00	x0000	x0000	TEST	123456789	19980401	150	DATA SEGMENT #8 KEY '123456789' DATE '01/04/1998'.....
42	00	x0000	x0000	TEST	123456789	19980401	160	DATA SEGMENT #9 KEY '123456789' DATE '01/04/1998'.....
42	00	x0000	x0000	TEST	123456789	19980401	170	DATA SEGMENT #10KEY '123456789' DATE '01/04/1998'.....
42	00	x0000	x0000	TEST	123456789	19980401	180	DATA SEGMENT #11KEY '123456789' DATE '01/04/1998'.....
42	00	x0000	x0000	TEST	123456789	19980401	190	DATA SEGMENT #12KEY '123456789' DATE '01/04/1998'.....
42	00	x0000	x0000	TEST	123456789	19980401	200	DATA SEGMENT #13KEY '123456789' DATE '01/04/1998'.....
42	00	x0000	x0000	TEST	123456789	20010821	210	DATA SEGMENT #1 KEY '123456789' DATE '21/08/2001'.....
42	00	x0000	x0000	TEST	123456789	20010821	220	DATA SEGMENT #2 KEY '123456789' DATE '21/08/2001'.....
42	00	x0000	x0000	TEST	123456789	20010821	128	DATA SEGMENT #3 KEY '123456789' DATE '21/08/2001'.....
42	00	x0000	x0000	TEST	123456789	20010821	128	DATA SEGMENT #4 KEY '123456789' DATE '21/08/2001'.....
42	00	x0000	x0000	TEST	123456789	20010821	128	DATA SEGMENT #5 KEY '123456789' DATE '21/08/2001'.....
42	00	x0000	x0000	TEST	123456789	20010821	128	DATA SEGMENT #6 KEY '123456789' DATE '21/08/2001'.....
42	00	x0000	x0000	TEST	123456789	20010821	128	DATA SEGMENT #7 KEY '123456789' DATE '21/08/2001'.....
42	00	x0000	x0000	TEST	123456789	20010821	128	DATA SEGMENT #8 KEY '123456789' DATE '21/08/2001'.....
42	00	x0000	x0000	TEST	123456789	20010821	128	DATA SEGMENT #9 KEY '123456789' DATE '21/08/2001'.....
42	00	x0000	x0000	TEST	123456789	20010821	128	DATA SEGMENT #10KEY '123456789' DATE '21/08/2001'.....
42	00	x0000	x0000	TEST	123456789	20010821	128	DATA SEGMENT #11KEY '123456789' DATE '21/08/2001'.....

Fig. A5 - job OTASD215



OTASD210

## INSTALLATION VERIFICATION PROCEDURE - CALL AUDIT REPORT

REQ CDE	RET CDE	SUPP CODE1	SUPP CODE2	D/B NAME	KEY	DATE	REC LEN	RECORD AREA
42	00	x0000	x0000	TEST	123456789	20010821	128	DATA SEGMENT #12KEY '123456789' DATE '21/08/2001'.....
42	00	x0000	x0000	TEST	123456789	19980930	128	DATA SEGMENT #1 KEY '123456789' DATE '30/09/1998'.....
42	00	x0000	x0000	TEST	123456789	19980930	128	DATA SEGMENT #2 KEY '123456789' DATE '30/09/1998'.....
42	00	x0000	x0000	TEST	123456789	19980930	128	DATA SEGMENT #3 KEY '123456789' DATE '30/09/1998'.....
42	00	x0000	x0000	TEST	123456789	19980930	128	DATA SEGMENT #4 KEY '123456789' DATE '30/09/1998'.....
42	00	x0000	x0000	TEST	123456789	19980930	128	DATA SEGMENT #5 KEY '123456789' DATE '30/09/1998'.....
42	00	x0000	x0000	TEST	123456789	19980930	128	DATA SEGMENT #6 KEY '123456789' DATE '30/09/1998'.....
42	00	x0000	x0000	TEST	123456789	19981020	128	DATA SEGMENT #1 KEY '123456789' DATE '20/10/1998'.....
42	00	x0000	x0000	TEST	123456789	19981020	128	DATA SEGMENT #2 KEY '123456789' DATE '20/10/1998'.....
42	00	x0000	x0000	TEST	123456789	19981120	128	DATA SEGMENT #1 KEY '123456789' DATE '20/11/1998'.....
42	00	x0000	x0000	TEST	123456789	19981120	128	DATA SEGMENT #2 KEY '123456789' DATE '20/11/1998'.....
42	00	x0000	x0000	TEST	133456789	19980526	128	DATA SEGMENT #1 KEY '133456789' DATE '26/05/1998'.....
42	00	x0000	x0000	TEST	223456789	19980605	128	DATA SEGMENT #1 KEY '223456789' DATE '05/06/1998'.....
42	00	x0000	x0000	TEST	223456789	19980605	128	DATA SEGMENT #2 KEY '223456789' DATE '05/06/1998'.....
42	00	x0000	x0000	TEST	223456789	19980708	128	DATA SEGMENT #1 KEY '223456789' DATE '08/07/1998'.....
42	00	x0000	x0000	TEST	223456789	19980708	128	DATA SEGMENT #2 KEY '223456789' DATE '08/07/1998'.....
42	00	x0000	x0000	TEST	223456789	19980708	128	DATA SEGMENT #3 KEY '223456789' DATE '08/07/1998'.....
42	00	x0000	x0000	TEST	223456789	19980708	128	DATA SEGMENT #4 KEY '223456789' DATE '08/07/1998'.....
42	00	x0000	x0000	TEST	233456789	19981012	128	DATA SEGMENT #1 KEY '233456789' DATE '12/10/1998'.....
42	00	x0000	x0000	TEST	233456789	19981012	128	DATA SEGMENT #2 KEY '233456789' DATE '12/10/1998'.....
42	00	x0000	x0000	TEST	233456789	19981012	128	DATA SEGMENT #3 KEY '233456789' DATE '12/10/1998'.....
42	00	x0000	x0000	TEST	233456789	19980811	128	DATA SEGMENT #1 KEY '233456789' DATE '11/08/1998'.....
42	00	x0000	x0000	TEST	123456789	19930929	128	DATA SEGMENT #1 KEY '123456789' DATE '29/09/93'.....
42	00	x0000	x0000	TEST	123456789	19930929	128	DATA SEGMENT #2 KEY '123456789' DATE '29/09/93'.....
42	00	x0000	x0000	TEST	143456789	19930604	128	DATA SEGMENT #1 KEY '143456789' DATE '04/06/93'.....

Fig. A5 - job OTASD215

OTASD210

## INSTALLATION VERIFICATION PROCEDURE - CALL AUDIT REPORT

REQ CDE	RET CDE	SUPP CODE1	SUPP CODE2	D/B NAME	KEY	DATE	REC LEN	RECORD AREA
42	00	x0000	x0000	TEST	143456789	19930604	128	DATA SEGMENT #2 KEY '143456789' DATE '04/06/93'.....
42	00	x0000	x0000	TEST	143456789	19930709	128	DATA SEGMENT #1 KEY '143456789' DATE '09/07/93'.....
42	00	x0000	x0000	TEST	143456789	19930709	128	DATA SEGMENT #2 KEY '143456789' DATE '09/07/93'.....
42	00	x0000	x0000	TEST	143456789	19930709	128	DATA SEGMENT #3 KEY '143456789' DATE '09/07/93'.....
42	00	x0000	x0000	TEST	153456789	19930608	128	DATA SEGMENT #1 KEY '153456789' DATE '08/06/93'.....
42	00	x0000	x0000	TEST	153456789	19930608	128	DATA SEGMENT #2 KEY '153456789' DATE '08/06/93'.....
42	00	x0000	x0000	TEST	233456789	19930110	128	DATA SEGMENT #1 KEY '233456789' DATE '10/01/93'.....
42	00	x0000	x0000	TEST	233456789	19930220	128	DATA SEGMENT #1 KEY '233456789' DATE '20/02/93'.....
42	00	x0000	x0000	TEST	233456789	19930220	128	DATA SEGMENT #2 KEY '233456789' DATE '20/02/93'.....
42	00	x0000	x0000	TEST	233456789	19930220	128	DATA SEGMENT #3 KEY '233456789' DATE '20/02/93'.....
42	00	x0000	x0000	TEST	233456789	19930220	128	DATA SEGMENT #4 KEY '233456789' DATE '20/02/93'.....
42	00	x0000	x0000	TEST	123456789	19991231	128	DATA SEGMENT #1 KEY '123456789' DATE '31/12/1999'.....
42	00	x0000	x0000	TEST	123456789	19991231	128	DATA SEGMENT #2 KEY '123456789' DATE '31/12/1999'.....
42	00	x0000	x0000	TEST	123456789	19991231	128	DATA SEGMENT #3 KEY '123456789' DATE '31/12/1999'.....
42	00	x0000	x0000	TEST	123456789	20000101	128	DATA SEGMENT #1 KEY '123456789' DATE '01/01/2000'.....
42	00	x0000	x0000	TEST	123456789	20000101	128	DATA SEGMENT #2 KEY '123456789' DATE '01/01/2000'.....
42	00	x0000	x0000	TEST	123456789	20020310	128	DATA SEGMENT #1 KEY '123456789' DATE '10/03/2002'.....
42	00	x0000	x0000	TEST	123456789	20020310	128	DATA SEGMENT #2 KEY '123456789' DATE '10/03/2002'.....
42	00	x0000	x0000	TEST	143456789	19980414	128	DATA SEGMENT #1 KEY '143456789' DATE '14/04/1998'.....
42	00	x0000	x0000	TEST	143456789	19980414	128	DATA SEGMENT #2 KEY '143456789' DATE '14/04/1998'.....
42	00	x0000	x0000	TEST	323456789	19980130	128	DATA SEGMENT #1 KEY '323456789' DATE '30/01/1998'.....
42	00	x0000	x0000	TEST	323456789	19980130	128	DATA SEGMENT #2 KEY '323456789' DATE '30/01/1998'.....
42	00	x0000	x0000	TEST	323456789	19980130	128	DATA SEGMENT #3 KEY '323456789' DATE '30/01/1998'.....
42	00	x0000	x0000	TEST	323456790	19980130	128	DATA SEGMENT #1 KEY '323456790' DATE '30/01/1998'.....
42	00	x0000	x0000	TEST	323456790	19980130	128	DATA SEGMENT #2 KEY '323456790' DATE '30/01/1998'.....

Fig. A5 - job OTASD215

## OTASD210

## INSTALLATION VERIFICATION PROCEDURE - CALL AUDIT REPORT

REQ CDE	RET CDE	SUPP CODE1	SUPP CODE2	D/B NAME	KEY	DATE	REC LEN	RECORD AREA
42	00	x0000	x0000	TEST	323456790	19980130	128	DATA SEGMENT #3 KEY '323456790' DATE '30/01/1998'.....
42	00	x0000	x0000	TEST	323456791	19980130	128	DATA SEGMENT #1 KEY '323456791' DATE '30/01/1998'.....
42	00	x0000	x0000	TEST	323456791	19980130	128	DATA SEGMENT #2 KEY '323456791' DATE '30/01/1998'.....
42	00	x0000	x0000	TEST	323456791	19980130	128	DATA SEGMENT #3 KEY '323456791' DATE '30/01/1998'.....
42	00	x0000	x0000	TEST	323456792	19980130	128	DATA SEGMENT #1 KEY '323456792' DATE '30/01/1998'.....
42	00	x0000	x0000	TEST	323456792	19980130	128	DATA SEGMENT #2 KEY '323456792' DATE '30/01/1998'.....
42	00	x0000	x0000	TEST	323456792	19980130	128	DATA SEGMENT #3 KEY '323456792' DATE '30/01/1998'.....
42	00	x0000	x0000	TEST	323456793	19980130	128	DATA SEGMENT #1 KEY '323456793' DATE '30/01/1998'.....
42	00	x0000	x0000	TEST	323456793	19980130	128	DATA SEGMENT #2 KEY '323456793' DATE '30/01/1998'.....
42	00	x0000	x0000	TEST	323456793	19980130	128	DATA SEGMENT #3 KEY '323456793' DATE '30/01/1998'.....
42	00	x0000	x0000	TEST	323456794	19980130	128	DATA SEGMENT #1 KEY '323456794' DATE '30/01/1998'.....
42	00	x0000	x0000	TEST	323456794	19980130	128	DATA SEGMENT #2 KEY '323456794' DATE '30/01/1998'.....
42	00	x0000	x0000	TEST	323456794	19980130	128	DATA SEGMENT #3 KEY '323456794' DATE '30/01/1998'.....
42	00	x0000	x0000	TEST	323456795	19980130	128	DATA SEGMENT #1 KEY '323456795' DATE '30/01/1998'.....
42	00	x0000	x0000	TEST	323456795	19980130	128	DATA SEGMENT #2 KEY '323456795' DATE '30/01/1998'.....
42	00	x0000	x0000	TEST	323456795	19980130	128	DATA SEGMENT #3 KEY '323456795' DATE '30/01/1998'.....
42	00	x0000	x0000	TEST	323456796	19980130	128	DATA SEGMENT #1 KEY '323456796' DATE '30/01/1998'.....
42	00	x0000	x0000	TEST	323456796	19980130	128	DATA SEGMENT #2 KEY '323456796' DATE '30/01/1998'.....
42	00	x0000	x0000	TEST	323456796	19980130	128	DATA SEGMENT #3 KEY '323456796' DATE '30/01/1998'.....
42	00	x0000	x0000	TEST	323456797	19980130	128	DATA SEGMENT #1 KEY '323456797' DATE '30/01/1998'.....
42	00	x0000	x0000	TEST	323456797	19980130	128	DATA SEGMENT #2 KEY '323456797' DATE '30/01/1998'.....
42	00	x0000	x0000	TEST	323456797	19980130	128	DATA SEGMENT #3 KEY '323456797' DATE '30/01/1998'.....
42	30	x0001	x0000	TEST			226	
42	00	x0000	x0000	TEST	123456789	19980401	80	DATA SEGMENT #1 KEY '123456789' DATE '01/04/1998'.....
49	00	x0000	x0000	TEST			0	

Fig. A5 - job OTASD215

OTASD210

## INSTALLATION VERIFICATION PROCEDURE - CALL AUDIT REPORT

REQ CDE	RET CDE	SUPP CODE1	SUPP CODE2	D/B NAME	KEY	DATE	REC LEN	RECORD AREA
40	00	x0000	x0000				1	TEST
42	00	x0000	x0000	TEST	223456789	19980708	128	DATA SEGMENT #1 KEY '223456789' DATE '08/07/1998'.....
47	00	x0000	x0000	TEST	223456789	19980708	226	
42	31	x0001	x0000	TEST	223456789	19980708	226	
49	00	x0000	x0000	TEST			0	

Fig. A5 - job OTASD215

OTASD210

## INSTALLATION VERIFICATION PROCEDURE - CALL AUDIT REPORT

REQ CDE	RET CDE	SUPP CODE1	SUPP CODE2	D/B NAME	KEY	DATE	REC LEN	RECORD AREA			
40	00	x0000	x0000				1	TEST			
41	00	x0004	x0000	TEST	123456789		2	12345678919930929	0	12345678919980401	0
41	00	x0004	x0000	TEST	123456789		2	12345678919980930	0	12345678919981020	0
41	00	x0004	x0000	TEST	123456789		2	12345678919981120	0	12345678919991231	0
41	00	x0004	x0000	TEST	123456789		2	12345678920000101	0	12345678920010821	0
41	00	x0000	x0000	TEST	123456789		1	12345678920020310	0		
41	00	x0004	x0000	TEST	123456789		2	12345678920020310	0	12345678920010821	0
41	00	x0004	x0000	TEST	123456789		2	12345678920000101	0	12345678919991231	0
41	00	x0004	x0000	TEST	123456789		2	12345678919981120	0	12345678919981020	0
41	00	x0004	x0000	TEST	123456789		2	12345678919980930	0	12345678919980401	0
41	00	x0000	x0000	TEST	123456789		1	12345678919930929	0		
41	00	x0004	x0000	TEST	123456789		2	12345678919930929	0	12345678919980401	0
41	00	x0004	x0000	TEST	123456789		2	12345678919980930	0	12345678919981020	0
41	00	x0004	x0000	TEST	123456789		2	12345678919981120	0	12345678919991231	0
41	00	x0004	x0000	TEST	123456789		2	12345678920000101	0	12345678920010821	0
41	00	x0000	x0000	TEST	123456789		1	12345678920020310	0		
41	00	x0004	x0000	TEST	123456789		2	12345678920020310	0	12345678920010821	0
41	00	x0004	x0000	TEST	123456789		2	12345678920000101	0	12345678919991231	0
41	00	x0004	x0000	TEST	123456789		2	12345678919981120	0	12345678919981020	0
41	00	x0004	x0000	TEST	123456789		2	12345678919980930	0	12345678919980401	0
41	00	x0000	x0000	TEST	123456789		1	12345678919930929	0		
49	00	x0000	x0000	TEST			0				

Fig. A5 - job OTASD215

OTASD210

## INSTALLATION VERIFICATION PROCEDURE - CALL AUDIT REPORT

REQ CDE	RET CDE	SUPP CODE1	SUPP CODE2	D/B NAME	KEY	DATE	REC LEN	RECORD AREA			
40	00	x0000	x0000				1	TEST			
41	00	x0004	x0000	TEST			2	12345678919930929	0	12345678919980401	0
41	00	x0004	x0000	TEST			2	12345678919980930	0	12345678919981020	0
41	00	x0004	x0000	TEST			2	12345678919981120	0	12345678919991231	0
41	00	x0004	x0000	TEST			2	12345678920000101	0	12345678920010821	0
41	00	x0004	x0000	TEST			2	12345678920020310	0	13345678919980526	0
41	00	x0004	x0000	TEST			2	12345678920020310	0	12345678920010821	0
41	00	x0004	x0000	TEST			2	12345678920000101	0	12345678919991231	0
41	00	x0004	x0000	TEST			2	12345678919981120	0	12345678919981020	0
41	00	x0004	x0000	TEST			2	12345678919980930	0	12345678919980401	0
41	00	x0004	x0000	TEST			2	12345678919930929	0	13345678919980526	0
41	00	x0004	x0000	TEST			2	32345679719980130	0	32345679619980130	0
41	00	x0004	x0000	TEST			2	32345679519980130	0	32345679419980130	0
41	00	x0004	x0000	TEST			2	32345679319980130	0	32345679219980130	0
41	00	x0004	x0000	TEST			2	32345679119980130	0	32345679019980130	0
41	00	x0004	x0000	TEST			2	32345678919980130	0	23345678919930110	0
41	00	x0004	x0000	TEST			2	32345679719980130	0	32345679619980130	0
41	00	x0004	x0000	TEST			2	32345679519980130	0	32345679419980130	0
41	00	x0004	x0000	TEST			2	32345679319980130	0	32345679219980130	0
41	00	x0004	x0000	TEST			2	32345679119980130	0	32345679019980130	0
41	00	x0004	x0000	TEST			2	32345678919980130	0	23345678919981012	0
49	00	x0000	x0000	TEST			0				

Fig. A5 - job OTASD215

## **APPENDIX B: Sample NearArchive Calls.**

---

Appendix B provides examples of NearArchive calls from a COBOL application. The appendix contains three sections corresponding to the following three types of NearArchive invocation:

1. batch archival processing
2. batch retrieval processing
3. CICS processing

Each section provides annotated examples of commonly-used NearArchive requests for that type of processing. COBOL is used in these examples.

Although there is not necessarily any connection between each individual example provided in this appendix, the supplied code may be used as the basis for customer development of a NearArchive application.

## **NearArchive parameter interface definition.**

The following working storage area is used in each of the sample calls to hold the NearArchive parameter interface:

```
*****
*                W02-  NEARARCHIVE  PARAMETER  BLOCK                *
*****
01  W02-NEAR-PARMS.
    05  W02-REQUEST-CODE                PIC  X(2) .
    05  W02-RETURN-CODE                PIC  X(2) .                (1)
    05  W02-SUPP-CODE1                 PIC  S9(4)  COMP.          (1)
    05  W02-SUPP-CODE2                 PIC  S9(4)  COMP.          (1)
    05  W02-QUERY REDEFINES W02-SUPP-CODE2.
        10  W02-UPPER                  PIC  X.
        10  W02-LOWER                  PIC  X.
    05  W02-DATABASE-NAME              PIC  X(4) .
    05  W02-KEY-POINTER                POINTER.                  (2)
    05  W02-ARCHIVE-DATE.              (3)
        10  W02-DATE-YYYY              PIC  9(4) .
        10  W02-DATE-MM                PIC  9(2) .
        10  W02-DATE-DD                PIC  9(2) .
    05  W02-TSQ-NAME                   PIC  X(8) .
    05  W02-OBJECT-START               PIC  S9(8)  COMP.
    05  W02-OBJECT-REC-COUNT           PIC  S9(8)  COMP.
    05  W02-RESERVED                   PIC  X(10) .
    05  W02-DATA-LENGTH                PIC  S9(4)  COMP.          (4)
    05  W02-ARCHIVE-DATA.              (5)
        10  W02-DBNAME-ENTRY.          (6)
            15  W02-DBNAME              PIC  X(4) .
            15  W02-REQ-TYPE            PIC  S9(4)  COMP.
            15  W02-ALLOC-RETRY        PIC  S9(4)  COMP.
            15  W02-OVERRIDE-DATE      PIC  9(8) .
            15  W02-TAPE-OVERRIDE      PIC  X(4) .
            15  W02-STORAGE-LEVEL      PIC  S9(4)  COMP.
            15  FILLER                  PIC  X(2) .
        10  FILLER                     PIC  X(232) .
```

### **Notes:**

1. Returned by NearArchive on call completion.
2. Pointer to the primary key of object being archived.
3. Archival date in format YYYYMMDD.
4. Length of data record to be archived.
5. Data record to be archived. The maximum record length in this application is 256 bytes. The length of this field should be sufficient to hold the largest record that will be written to the archive database.



6. Database name list entry. This 24-byte area is used during database connect processing to pass information to NearArchive about the database to be connected.

## B.1 Batch Archival Processing.

---

### B.1.1 Connect to NearArchive Database.

```
B100-CONNECT-NEARARCHIVE-DATABASE SECTION.
*=====

      PERFORM      B200-INIT-NEARARCHIVE-PARMS.                                (1)

      MOVE         '40'          TO          W02-REQUEST-CODE.
      MOVE         1             TO          W02-DATA-LENGTH.
      MOVE         SPACES        TO          W02-DBNAME-LIST.
      MOVE         'V100'        TO          W02-DBNAME.
      MOVE         0             TO          W02-ALLOC-RETRY.
      MOVE         0             TO          W02-STORAGE-LEVEL.

      * CHECK FOR RESTART

      IF W04-NORMAL-RUN = 'Y'                                           (2)
        MOVE      4             TO          W02-REQUEST-TYPE.
      ELSE
        MOVE      6             TO          W02-REQUEST-TYPE.

      CALL         'OTASP100' USING      W02-NEAR-PARMS.

      * ERROR PROCESSING

      IF NOT (W02-RETURN-CODE = '00')                                   (3)
        MOVE      W02-RETURN-CODE TO          W99-ERR-MSG3-RETURN.
        MOVE      W02-SUPP-CODE1  TO          W99-ERR-MSG3-SUPP1.
        MOVE      W02-SUPP-CODE2  TO          W99-ERR-MSG3-SUPP2.
        MOVE      W99-ERR-MSG3    TO          W99-ERR-MSG.
        PERFORM   Z900-ERROR-PROCESSING.

      B199-EXIT.
      *=====
      EXIT.
      EJECT

      B200-INIT-NEARARCHIVE-PARMS SECTION.
      *=====

      MOVE      SPACES TO W02-NEAR-PARMS.
      MOVE      ZEROS  TO W02-SUPP-CODE1
                        W02-SUPP-CODE2
                        W02-OBJECT-START
                        W02-OBJECT-REC-COUNT
                        W02-DATA-LENGTH.
```

```
B299-EXIT.  
*=====
```

EXIT.  
EJECT

**Notes:**

1. Initialize NearArchive parameter block.
2. NearArchive processes 3 types of output connect database calls, depending on the value of the request type indicator in the database name list:
  - 4 - open for normal output processing. Issued prior to the first archival request during normal running.
  - 5 - open for normal output processing to a new tape volume. Issued to force NearArchive to write the first archival request to a new tape volume.
  - 6 - open for rerun output processing. Issued to connect the database during rerun processing (i.e.) after a NearArchive archival run has abnormally terminated.
3. On return to the calling program, NearArchive sets a 2-digit numeric code in the return code field of the parameter block to indicate the success or otherwise of the NearArchive call. Non-zero return codes may have additional error information in the supplementary code fields 1 and 2.

## **B.1.2 Write Record to NearArchive Database.**

```
* SET UP NEARARCHIVE ARCHIVE PARAMETERS

      PERFORM      B200-INIT-NEARARCHIVE-PARMS.                      (1)

      MOVE          '44'                      TO W02-REQUEST-CODE.    (2)
      MOVE          'V100'                    TO W02-DATABASE-NAME.
      SET           W02-KEY-POINTER TO ADDRESS OF PD-STAFF-NO.
      MOVE          PD-EMP-TO-DATE            TO W02-ARCHIVE-DATE.
      MOVE          W04-EX-REC-LENGTH         TO W02-DATA-LENGTH.
      MOVE          EXPENSES-RECORD           TO W02-ARCHIVE-DATA.

* CALL NEARARCHIVE TO ARCHIVE DATA

      CALL          'OTASP100'                USING      W02-NEAR-PARMS.

      IF NOT (W02-RETURN-CODE = '00')                      (3)
          MOVE      W02-RETURN-CODE            TO      W99-ERR-MSG3-RETURN.
          MOVE      W02-SUPP-CODE1             TO      W99-ERR-MSG3-SUPP1.
          MOVE      W02-SUPP-CODE2             TO      W99-ERR-MSG3-SUPP2.
          MOVE      W99-ERR-MSG3               TO      W99-ERR-MSG.
          PERFORM   Z900-ERROR-PROCESSING.
```

### **Notes:**

1. Initialize NearArchive parameter block (see B.1.1).
2. W02-REQUEST-CODE - '44', write a data segment to the database  
W02-DATABASE-NAME - Name of database to be updated  
W02-KEY-POINTER - Pointer to primary key of data object being archived. Note that the primary key field itself must be defined in the program linkage section. Refer to program OTASC100 in the distributed NearArchive sample code library for an example of this usage.  
W02-ARCHIVE-DATE - Archival date in format YYYYMMDD  
W02-DATA-LENGTH - Length of data record to be archived  
W02-DATA - Data record to be archived
3. On return to the calling program, NearArchive sets a 2 digit numeric code in the return code field of the parameter block to indicate the success or otherwise of the NearArchive call. Non-zero return codes may have additional error information in the supplementary code fields 1 and 2.

### **B.1.3 Identify Secondary Key.**

```
* SET UP NEARARCHIVE ARCHIVE PARAMETERS

      PERFORM      B200-INIT-NEARARCHIVE-PARMS.                (1)

      MOVE         '46'                                TO      W02-REQUEST-CODE.        (2)
      MOVE         49                                TO      W02-SUPP-CODE1.            (3)
      MOVE         1                                  TO      W02-SUPP-CODE2.            (4)
      MOVE         'V100'                             TO      W02-DATABASE-NAME.        (5)
      MOVE         W06-SEC-IX-KEY-49                   TO      W02-DATA.

* CALL NEARARCHIVE TO ARCHIVE DATA

      CALL         'OTASP100'                           USING      W02-NEAR-PARMS.

      IF NOT (W02-RETURN-CODE = '00')                    (6)
      MOVE         W02-RETURN-CODE                     TO      W99-ERR-MSG3-RETURN.
      MOVE         W02-SUPP-CODE1                       TO      W99-ERR-MSG3-SUPP1.
      MOVE         W02-SUPP-CODE2                       TO      W99-ERR-MSG3-SUPP2.
      MOVE         W99-ERR-MSG3                         TO      W99-ERR-MSG.
      PERFORM      Z900-ERROR-PROCESSING.
```

#### **Notes:**

1. Initialize NearArchive parameter block (see B.1.1).
2. W02-REQUEST-CODE - '46', identify secondary key for archived data.
3. W02-SUPP-CODE1 - Identify secondary index (01-99). In this example, secondary index number 49 is being used.
4. W02-SUPP-CODE2 - '1' indicates that the secondary key is in the data area.
5. W02-DATA - Secondary key. W06-SEC-IX-KEY-49 is a working storage area containing the secondary key to be used to index the object being archived. The length of this field will be equal to the secondary key length defined for index #49 in this NearArchive database.
6. On return to the calling program, NearArchive sets a 2 digit numeric code in the return code field of the parameter block to indicate the success or otherwise of the NearArchive call. Non-zero return codes may have additional error information in the supplementary code fields 1 and 2.

### **B.1.4 Commit Updates.**

```
* SET UP NEARARCHIVE PARAMETER BLOCK

      PERFORM      B200-INIT-NEARARCHIVE-PARMS.                (1)

      MOVE         '48'              TO      W02-REQUEST-CODE.  (2)
      MOVE         'V100'            TO      W02-DATABASE-NAME.

      CALL         'OTASP100'        USING    W02-NEAR-PARMS.

      IF NOT (W02-RETURN-CODE = '00')                (3)
      MOVE         W02-RETURN-CODE TO      W99-ERR-MSG3-RETURN.
      MOVE         W02-SUPP-CODE1  TO      W99-ERR-MSG3-SUPP1.
      MOVE         W02-SUPP-CODE2  TO      W99-ERR-MSG3-SUPP2.
      MOVE         W99-ERR-MSG3    TO      W99-ERR-MSG.
      PERFORM      Z900-ERROR-PROCESSING.
```

#### **Notes:**

1. Initialize NearArchive parameter block (see B.1.1).
2. A request code of '48' is required in the parameter block for a database commit call.
3. On return to the calling program, NearArchive sets a 2 digit numeric code in the return code field of the parameter block to indicate the success or otherwise of the NearArchive call. Non-zero return codes may have additional error information in the supplementary code fields 1 and 2.

### **B.1.5 Disconnect Database.**

```
* DISCONNECT NEARARCHIVE DATABASE

      PERFORM      B200-INIT-NEARARCHIVE-PARMS.                (1)

      MOVE         '49'              TO      W02-REQUEST-CODE.  (2)
      MOVE         'V100'            TO      W02-DATABASE-NAME.

      CALL         'OTASP100'        USING    W02-NEAR-PARMS.

      IF NOT (W02-RETURN-CODE = '00')                (3)
      MOVE         W02-RETURN-CODE TO      W99-ERR-MSG4-RETURN.
      MOVE         W02-SUPP-CODE1  TO      W99-ERR-MSG4-SUPP1.
      MOVE         W02-SUPP-CODE2  TO      W99-ERR-MSG4-SUPP2.
      MOVE         W99-ERR-MSG4    TO      W99-ERR-MSG.
      PERFORM      Z900-ERROR-PROCESSING.
```

#### **Notes:**

1. Initialize NearArchive parameter block (see B.1.1).
2. A request code of '49' is required in the parameter block for a database disconnect call. Supplementary codes 1 and 2 may optionally be used to specify close options for the tape dataset and commit options to be implemented during disconnect processing.
3. On return to the calling program, NearArchive sets a 2 digit numeric code in the return code field of the parameter block to indicate the success or otherwise of the NearArchive call. Non-zero return codes may have additional error information in the supplementary code fields 1 and 2.

## B.2 Batch Retrieval Processing.

---

### B.2.1 Connect Database.

```
* CALL NEARARCHIVE TO CONNECT DATABASE

      PERFORM      C100-INIT-NEARARCHIVE-PARMS.                (1)

      MOVE         '40'              TO      W02-REQUEST-CODE.  (2)
      MOVE         1                  TO      W02-DATA-LENGTH.
      MOVE         SPACES              TO      W02-DBNAME-LIST.
      MOVE         'V100'             TO      W02-DBNAME.
      MOVE         3                   TO      W02-REQ-TYPE.     (3)
      MOVE         10                  TO      W02-ALLOC-RETRY. (4)

      CALL         'OTASP100'          USING    W02-NEAR-PARMS.

      IF (W02-RETURN-CODE = '00')                (5)
          PERFORM D000-PROCESS-ARCHIVED-DATA.
      ELSE
          MOVE      W02-RETURN-CODE TO      W99-ERR-MSG1-RETURN.
          MOVE      W02-SUPP-CODE1  TO      W99-ERR-MSG1-SUPP1.
          MOVE      W02-SUPP-CODE2  TO      W99-ERR-MSG1-SUPP2.
          MOVE      W99-ERR-MSG1    TO      W99-ERR-MSG.
          PERFORM Z900-ERROR-PROCESSING.
          |
          |
          |
      C100-INIT-NEARARCHIVE-PARMS SECTION.
*=====
      MOVE      SPACES TO W02-NEAR-PARMS.
      MOVE      ZEROS  TO W02-SUPP-CODE1
                        W02-SUPP-CODE2
                        W02-OBJECT-START
                        W02-OBJECT-REC-COUNT
                        W02-DATA-LENGTH.

      C199-EXIT.
*=====
      EXIT.
      EJECT
```

**Notes:**

1. Initialize NearArchive parameter block.
2. W02-REQUEST-CODE - '40', Connect database request. This call must be issued once before retrieving data from a NearArchive database.
3. W02-REQ-TYPE - '3', Connect database for input-output processing. This will enable delete requests to be issued.
4. An allocation retry limit of 10 is set by this call. This overrides the default value of 30.
5. On return to the calling program, NearArchive sets a 2 digit numeric code in the return code field of the parameter block to indicate the success or otherwise of the NearArchive call. Non-zero return codes may have one or two supplementary codes set in fields 1 and 2.



## **B.2.2 Query Database.**

```
* SET UP NEARARCHIVE CALL PARAMETERS

      PERFORM      C100-INIT-NEARARCHIVE-PARMS.                (1)

      MOVE         '41'                TO      W02-REQUEST-CODE.  (2)
      MOVE         49                   TO      W02-SUPP-CODE1.   (3)
      MOVE         3                     TO      W02-SUPP-CODE2.   (4)
      MOVE         'V100'                TO      W02-DATABASE-NAME.
      MOVE         3                     TO      W02-DATA-LENGTH.   (5)
      MOVE         W06-SEC-IX-KEY-49     TO      W02-DATA.         (6)

      CALL         'OTASP100'            USING  W02-NEAR-PARMS.

      IF (W02-RETURN-CODE = '00')                (7)
        PERFORM D000-PROCESS-RETURNED-DATA.
      ELSE
        IF (W02-RETURN-CODE = '31')
          PERFORM D100-NO-RETURNED-DATA.
        ELSE
          MOVE      W02-RETURN-CODE      TO      W99-ERR-MSG2-RETURN.
          MOVE      W02-SUPP-CODE1       TO      W99-ERR-MSG2-SUPP1.
          MOVE      W02-SUPP-CODE2       TO      W99-ERR-MSG2-SUPP2.
          MOVE      W99-ERR-MSG2         TO      W99-ERR-MSG.
          PERFORM  Z900-ERROR-PROCESSING.
          |
          |
          |
```

### **Notes:**

1. Initialize NearArchive parameter block
2. W02-REQUEST-CODE - '41', query index.
3. W02-SUPP-CODE1 - Identifies the index to be queried. In this example, secondary index 49 is to be queried.
4. W02-SUPP-CODE2 - Specifies the sort sequence for the returned items. (x'03' = Descending date within descending key)
5. W02-DATA-LENGTH - Specifies the maximum number of formatted index entries to be returned by NearArchive.
6. W02-DATA - On entry to NearArchive, this specifies the secondary key to be queried for a secondary index call. In this example, W06-SEC-IX-KEY-49 contains the secondary key. This field must be the same length as that of the secondary key in index 49.

On return from NearArchive, W02-DATA will contain the formatted items returned by NearArchive. This field should be large enough to hold the number specified in the previous field.

The length of each formatted item is: (primary keylength + 24). In this example, the primary keylength is 22 characters; each formatted item will therefore be 46 bytes long. A maximum of 3 formatted items per data area may be accepted by the calling program.

7. On return to the calling program, NearArchive sets a 2 digit numeric code in the return code field of the parameter block to indicate the success or otherwise of the NearArchive call. Non-zero return codes may have one or two supplementary codes set in fields 1 and 2.

A return code of 00 and supplementary code 1 value of 4 indicates that not all index entries have been returned in the parameter area. Further query calls should be issued by the program to return remaining entries for the specified secondary key and date.

A return code of 31 indicates that the secondary key specified in the parameter block is not present in the NearArchive database.

### **B.2.3 Retrieve Data Record (Direct Retrieval).**

```
* SET UP NEARARCHIVE CALL PARAMETERS

      PERFORM      C100-INIT-NEARARCHIVE-PARMS.                (1)

      MOVE         '42'                TO      W02-REQUEST-CODE.  (2)
      MOVE         1                    TO      W02-SUPP-CODE1.    (3)
      MOVE         'V100'              TO      W02-DATABASE-NAME.
      SET          W02-KEY-POINTER     TO      ADDRESS OF W06-KEY.  (4)
      MOVE         W06-DATE             TO      W02-DATE.         (5)
      MOVE         131                 TO      W02-DATA-LENGTH.    (6)

      CALL         'OTASP100'          USING    W02-NEAR-PARMS.

      IF (W02-RETURN-CODE = '00')                (7)
        PERFORM D000-PROCESS-RETURNED-DATA.
      ELSE
        IF (W02-RETURN-CODE = '31')
          PERFORM D100-NO-RETURNED-DATA.
        ELSE
          MOVE      W02-RETURN-CODE     TO      W99-ERR-MSG3-RETURN.
          MOVE      W02-SUPP-CODE1      TO      W99-ERR-MSG3-SUPP1.
          MOVE      W02-SUPP-CODE2      TO      W99-ERR-MSG3-SUPP2.
          MOVE      W99-ERR-MSG3        TO      W99-ERR-MSG.
          PERFORM Z900-ERROR-PROCESSING.

      |                                     |
```

#### **Notes:**

1. Initialize NearArchive parameter block (see B.2.1).
2. W02-REQUEST-CODE - '42', retrieve data record.
3. W02-SUPP-CODE1 - 1 , direct read request.
4. W02-KEY-POINTER - Contains a pointer to a field containing the primary key of the NearArchive object to be processed. Note that the primary key field itself must be defined in the program linkage section. Refer to program OTASC200 in the distributed NearArchive sample code library for an example of this usage.
5. W02-DATE - Specifies the archive date of NearArchive object to be processed.
6. W02-DATA-LENGTH - Specifies the maximum length of record to be returned.

7. On return to the calling program, NearArchive sets a 2 digit numeric code in the return code field of the parameter block to indicate the success or otherwise of the NearArchive call. Non-zero return codes may have one or two supplementary codes set in fields 1 and 2.

A return code of 00 indicates that the first (or next) logical record has been retrieved for the specified NearArchive object. The program should process this record as required, and continue to issue retrieval request for the same object in order to process all its logical records.

A return code of 30 will be set when all logical records for the specified NearArchive object have been retrieved.

A return code of 31 indicates that the object whose key is specified in the parameter block does not exist in the NearArchive database.

## **B.2.4 Retrieve Data Record (Sequential Retrieval).**

```
* SET UP NEARARCHIVE PARAMETERS

      PERFORM      C100-INIT-NEARARCHIVE-PARMS.                (1)

      MOVE         '42'                TO      W02-REQUEST-CODE.  (2)
      MOVE         2                    TO      W02-SUPP-CODE1    (3)
      MOVE         'V100'              TO      W02-DATABASE-NAME.
      SET          W02-KEY-POINTER     TO ADDRESS OF W06-KEY.      (4)
      MOVE         W04-DATA-LENGTH     TO      W02-DATA-LENGTH.  (5)
      PERFORM      R100-RETRIEVE-DATA
                  UNTIL      W04-END-OF-DATA-IND = 'Y'.

                  |

R100-RETRIEVE-DATA SECTION.
*=====

* RETRIEVE NEARARCHIVE DATA

      CALL         'OTASP100'          USING      W02-NEAR-PARMS.

      IF (W02-RETURN-CODE = '30')                (6)
      MOVE         'Y'                    TO      W04-END-OF-DATA-IND.
      GO TO       R199-EXIT.

      IF NOT (W02-RETURN-CODE = '00')
      MOVE         W02-RETURN-CODE     TO      W99-ERR-MSG4-RETURN.
      MOVE         W02-SUPP-CODE1      TO      W99-ERR-MSG4-SUPP1.
      MOVE         W02-SUPP-CODE2      TO      W99-ERR-MSG4-SUPP2.
      MOVE         W99-ERR-MSG4        TO      W99-ERR-MSG.
      PERFORM      Z900-ERROR-PROCESSING.

      MOVE         W02-DATA             TO      W08-RECORD.      (7)

                  |

R199-EXIT.
*=====
      EXIT.
      EJECT
```

### **Notes:**

1. Initialize NearArchive parameter block (see B.2.1).
2. W02-REQUEST-CODE - '42', retrieve logical record.
3. W02-SUPP-CODE1 - 2, sequential read request.

4. W02-KEY-POINTER - Contains a pointer to a field containing the primary key of the NearArchive object to be processed. NearArchive will copy the primary key of the retrieved object to the field addressed by this pointer value after processing of the request. Note that the primary key field itself must be defined in the program linkage section. Refer to program OTASC200 in the distributed NearArchive sample code library for an example of this usage.
5. W02-DATA-LENGTH - The maximum record length allowed.
6. On return to the calling program, NearArchive sets a 2 digit numeric code in the return code field of the parameter block to indicate the success or otherwise of the NearArchive call. Non-zero return codes may have one or two supplementary codes set in fields 1 and 2

A return code of 00 indicates that a record has been successfully retrieved from the NearArchive database. The primary key and archive date fields in the parameter area will identify the NearArchive object of which the retrieved logical record is a component part.

A return code of 30 indicates that all objects in the database have been processed (end-of-database condition). Issuing another sequential retrieval request will cause the first object in the NearArchive database to be processed.

7. W02-DATA - is the field in the parameter block which contains the data record retrieved by the NearArchive call.

### **B.2.5 Delete NearArchive Object.**

```
* SET UP NEARARCHIVE CALL PARAMETERS

      PERFORM      C100-INIT-NEARARCHIVE-PARMS.                (1)

      MOVE         '47'                TO      W02-REQUEST-CODE. (2)
      MOVE         'V100'              TO      W02-DATABASE-NAME.
      SET          W02-KEY-POINTER     TO ADDRESS OF W06-KEY.    (3)
      MOVE         W06-DATE            TO      W02-DATE         (4)

      CALL         'OTASP100'          USING    W02-NEAR-PARMS.

      IF NOT (W02-RETURN-CODE = '00') (5)
      MOVE         W02-RETURN-CODE     TO      W99-ERR-MSG5-RETURN.
      MOVE         W02-SUPP-CODE1      TO      W99-ERR-MSG5-SUPP1.
      MOVE         W02-SUPP-CODE2      TO      W99-ERR-MSG5-SUPP2.
      MOVE         W99-ERR-MSG5        TO      W99-ERR-MSG.
      PERFORM      Z900-ERROR-PROCESSING.

      |                                     |
```

#### **Notes:**

1. Initialize NearArchive parameter block (see B.2.1).
2. W02-REQUEST-CODE - '47', delete NearArchive object.
3. W02-KEY-POINTER - Contains a pointer to a field containing the primary key of the NearArchive object to be processed. Note that the primary key field itself must be defined in the program linkage section. Refer to program OTASC200 in the distributed NearArchive sample code library for an example of this usage.
4. W02-DATE - Specifies the archive date of the object to be deleted.
5. On return to the calling program, NearArchive sets a 2 digit numeric code in the return code field of the parameter block to indicate the success or otherwise of the NearArchive call. Non-zero return codes may have one or two supplementary codes in fields 1 and 2.

### **B.2.6 Disconnect Database.**

```
* DISCONNECT NEARARCHIVE DATABASE

      PERFORM      C100-INIT-NEARARCHIVE-PARMS.                (1)

      MOVE          '49'              TO      W02-REQUEST-CODE.  (2)
      MOVE          'V100'            TO      W02-DATABASE-NAME.

      CALL          'OTASP100'        USING    W02-NEAR-PARMS.

      IF NOT (W02-RETURN-CODE = '00')                (3)
      MOVE          W02-RETURN-CODE    TO      W99-ERR-MSG6-RETURN.
      MOVE          W02-SUPP-CODE1     TO      W99-ERR-MSG6-SUPP1.
      MOVE          W02-SUPP-CODE2     TO      W99-ERR-MSG6-SUPP2.
      MOVE          W99-ERR-MSG6       TO      W99-ERR-MSG.
      PERFORM      Z900-ERROR-PROCESSING.
```

#### **Notes:**

1. Initialize NearArchive parameter block (see B.2.1).
2. A request code of '49' is the only mandatory entry in the parameter block for a database disconnect call. Supplementary code 1 may optionally be set to control tape cartridge volume handling during dataset close processing.
3. On return to the calling program, NearArchive sets a 2 digit numeric code in the return code field of the parameter block to indicate the success or otherwise of the NearArchive call. Non-zero return codes may have one or two supplementary codes set in fields 1 and 2.



## B.3 CICS Processing.

---

### B.3.1 Retrieve Archived Object.

```
* RETRIEVE ARCHIVED OBJECT
*
      MOVE      SPACES TO W02-NEAR-PARMS.
      MOVE      ZEROS  TO W02-SUPP-CODE1
                        W02-SUPP-CODE2
                        W02-OBJECT-START
                        W02-OBJECT-REC-COUNT
                        W02-DATA-LENGTH.

      MOVE      '43'          TO      W02-REQUEST-CODE.      (1)
      MOVE      'TEST'        TO      W02-DATABASE-NAME.    (2)
      SET       W02-KEY-POINTER TO      ADDRESS OF CA-ST-KEY.
      MOVE      CA-ST-DATE     TO      W02-ARCHIVE-DATE.
      MOVE      TSQ-NAME       TO      W02-TSQ-NAME.

      EXEC CICS LINK PROGRAM ('OTASP200')                    (3)
                        COMMAREA (W02-NEAR-PARMS)
                        LENGTH (52)
                        NOHANDLE

      END-EXEC.

      IF NOT (EIBRESP = DFHRESP(NORMAL))
      MOVE      'Y'           TO      W06-SERIOUS-ERROR-IND.
      MOVE      W99-MSG03     TO      W99-MSG00.
      GO TO     B099-EXIT.

*
* NO ARCHIVED DATA EXISTS
*
      IF W02-RETURN-CODE = '31'                                (4) .
      MOVE      W99-MSG04     TO      W99-MSG00.
      GO TO     B099-EXIT.

*
* ARCHIVING BUSY
*
      IF W02-RETURN-CODE = '21'
      MOVE      W99-MSG05     TO      W99-MSG00.
      GO TO     B099-EXIT.

*
* RETRIEVAL ERROR
*
      IF NOT (W02-RETURN-CODE = '00')
      MOVE      'Y'           TO      W06-SERIOUS-ERROR-IND.
      MOVE      W02-RETURN-CODE TO      W99-MSG06-RETURN-CODE.
      MOVE      W02-SUPP-CODE1 TO      W99-MSG06-SUPP-CODE1.
      MOVE      W02-SUPP-CODE2 TO      W99-MSG06-SUPP-CODE2.
      MOVE      W99-MSG06     TO      W99-MSG00.
      GO TO     B099-EXIT.
```

### **Notes:**

1. W02-REQUEST-CODE - 43; retrieve archived object. In this example, supplementary codes 1 and 2 are preset to 0. This indicates that retrieval will be into an auxiliary temporary storage queue, and that a priority of 0 is to be assigned to the request.
2. W02-DATABASE-NAME - Name of NearArchive database to be processed. This should match the last 4 characters of the CICS FCT entry for the database's primary index dataset.

W02-KEY-POINTER - Contains a pointer to a field containing the primary key of the object to be retrieved. Note that the primary key field itself must be defined in the program linkage section. Refer to program OTASC300 in the distributed NearArchive sample code library for an example of this usage.

W02-ARCHIVE-DATE - Archive date of the object to be retrieved, in the format YYYYMMDD.

W02-TSQ-NAME - Name of temporary storage queue to contain the archived object retrieved by NearArchive.

3. Link to NearArchive, passing the parameter block in the COMMAREA.
4. On return to the calling program, NearArchive sets a 2 digit numeric code in the return code field of the parameter block to indicate the success or otherwise of the NearArchive call. Non-zero return codes may have one or two supplementary codes set in reason code fields 1 and 2.

### **B.3.2 Query Database.**

```
* NEARARCHIVE QUERY CALL
*
      MOVE      SPACES TO W02-NEAR-PARMS.
      MOVE      ZEROS  TO W02-SUPP-CODE1
                        W02-SUPP-CODE2
                        W02-OBJECT-START
                        W02-OBJECT-REC-COUNT
                        W02-DATA-LENGTH.

      MOVE      '41'          TO      W02-REQUEST-CODE.      (1)
      MOVE      CA-SEC-IND-NO TO      W02-SUPP-CODE1.      (2)
      MOVE      CA-SORT-SEQUENCE TO      W02-SUPP-CODE2.    (3)
      MOVE      'TEST'        TO      W02-DATABASE-NAME.    (4)
      MOVE      CA-SEC-INDEX-KEY TO      W02-SEC-INDEX-KEY. (5)
      MOVE      TSQ-NAME       TO      W02-TSQ-NAME.

      EXEC CICS LINK PROGRAM ('OTASP200')                    (6)
                        COMMAREA (W02-NEAR-PARMS)
                        LENGTH (64)
                        NOHANDLE

      END-EXEC.

      IF NOT (EIBRESP = DFHRESP(NORMAL))
      MOVE      'Y'          TO      W06-SERIOUS-ERROR-IND.
      MOVE      W99-MSG03     TO      W99-MSG00.
      GO TO     B099-EXIT.

*
* NO ARCHIVED DATA EXISTS
*
      IF W02-RETURN-CODE = '31'                                (7)
      MOVE      W99-MSG07     TO      W99-MSG00.
      GO TO     B099-EXIT.

*
* OTHER ERROR
*
      IF NOT (W02-RETURN-CODE = '00')
      MOVE      'Y'          TO      W06-SERIOUS-ERROR-IND.
      MOVE      W02-RETURN-CODE TO      W99-MSG08-RETURN-CODE.
      MOVE      W02-SUPP-CODE1 TO      W99-MSG08-SUPP-CODE1.
      MOVE      W02-SUPP-CODE2 TO      W99-MSG08-SUPP-CODE2.
      MOVE      W99-MSG08     TO      W99-MSG00.
      GO TO     B099-EXIT.
```

### **Notes:**

1. W02-REQUEST-CODE - 41; database query request.
2. W02-SUPP-CODE1 - Identifier of index to be queried. In this example, a non-zero value is set in this field. This indicates that the query is to be performed on a secondary index.
3. W02-SUPP-CODE2 - Specifies the sort sequence for the returned items.
4. W02-DATABASE-NAME - Name of NearArchive database to be queried. This should match the last 4 characters of the CICS FCT entry for the database's primary index dataset.
5. W02-SEC-INDEX-KEY - Secondary key to be queried. In this example the length of the secondary key is 12 bytes, giving an overall parameter interface area of 64 bytes.
6. Link to NearArchive, passing the parameter block in the COMMAREA.
7. On return to the calling program, NearArchive sets a 2 digit numeric code in the return code field of the parameter block to indicate the success or otherwise of the NearArchive call. Non-zero return codes may have one or two supplementary codes set in reason code fields 1 and 2.

For a return code of 00, the temporary storage queue identified in the NearArchive parameter area will contain one formatted item for each index entry which satisfies the query criteria.

### **B.3.3 Delete Object.**

```
* NEARARCHIVE DELETE CALL
*
      MOVE      SPACES TO W02-NEAR-PARMS.
      MOVE      ZEROS  TO W02-SUPP-CODE1
                        W02-SUPP-CODE2
                        W02-OBJECT-START
                        W02-OBJECT-REC-COUNT
                        W02-DATA-LENGTH.

      MOVE      '47'          TO      W02-REQUEST-CODE.      (1)
      MOVE      'TEST'        TO      W02-DATABASE-NAME.    (2)
      SET       W02-KEY-POINTER TO ADDRESS OF CA-ST-KEY.      (3)
      MOVE      CA-ST-DATE     TO      W02-ARCHIVE-DATE.      (4)
      EXEC CICS LINK PROGRAM ('OTASP200')                      (5)
                        COMMAREA (W02-NEAR-PARMS)
                        LENGTH (52)
                        NOHANDLE

      END-EXEC.

      IF NOT (EIBRESP = DFHRESP(NORMAL))
      MOVE      'Y'          TO      W06-SERIOUS-ERROR-IND.
      MOVE      W99-MSG03     TO      W99-MSG00.
      GO TO     B099-EXIT.

*
* NO ARCHIVED DATA EXISTS
*
      IF W02-RETURN-CODE = '31'                                (6)
      MOVE      W99-MSG07     TO      W99-MSG00.
      GO TO     B099-EXIT.

*
* OTHER ERROR
*
      IF NOT (W02-RETURN-CODE = '00')
      MOVE      'Y'          TO      W06-SERIOUS-ERROR-IND.
      MOVE      W02-RETURN-CODE TO      W99-MSG09-RETURN-CODE.
      MOVE      W02-SUPP-CODE1 TO      W99-MSG09-SUPP-CODE1.
      MOVE      W02-SUPP-CODE2 TO      W99-MSG09-SUPP-CODE2.
      MOVE      W99-MSG09     TO      W99-MSG00.
      GO TO     B099-EXIT.
```

**Notes:**

1. W02-REQUEST-CODE      -    47; delete NearArchive object.
2. W02-DATABASE-NAME    -    Name of NearArchive database to be processed.  
This should match the last 4 characters of the CICS  
FCT entry for the database's primary index dataset.
3. Pointer to a field containing the primary key of object to be deleted. Note that the  
primary key field itself must be defined in the program linkage section. Refer to  
program OTASC300 in the distributed NearArchive sample code library for an  
example of this usage.
4. Archive date (in format YYYYMMDD) of object to be deleted.
5. Link to NearArchive, passing the parameter block in the COMMAREA.
6. On return to the calling program, NearArchive sets a 2 digit numeric code in the  
return code field of the parameter block to indicate the success or otherwise of the  
NearArchive call. Non-zero return codes may have one or two supplementary codes  
set in reason code fields 1 and 2.

## APPENDIX C: Database Regression Utility

---

Version 2.5 of ASM for S/390 requires that databases created using a pre-version 2.5 release of the product be converted prior to use with this release. Sections 2.9 and 3.5 in the main body of the manual discuss this requirement in detail.

Should it be required for any reason to revert to a previous release of the product after installing version 2.5, then any database which has been converted for use by this release will need to be regressed for use with the older release. This regression is performed by program OTASP032, the database regression utility.

Note that the regression utility will not be able to process a v2.5 database if more than 2 storage levels have been defined. If it is required to regress such a database, then the database primary index (and secondary indexes, if defined) and space management dataset(s) should be restored to their state prior to definition of the new storage levels, before running the regression utility on the database.

Following successful completion of the regression utility, an additional step may be required to reduce the primary index dataset maximum record length, if this was altered during product installation processing.

### C.1 Functions.

---

The database regression utility performs the following functions:

- Creates a new space management file (one for each storage level in the converted database) in pre-v2.5 format.
- Sequentially processes the primary index of the database being regressed, and converts the index back into v2.2, v2.3 or v2.4 formats (depending on EXEC parameter value).
- Initializes the new space management file(s) created above.
- Renames the existing space management file(s), then renames the newly-created space management file(s) to that of the renamed space management file(s).

## C.2 JCL Requirements.

---

The following JCL should be used to execute the NearArchive database regression utility.

```
//STEP1      EXEC PGM=OTASP032,REGION=4M,  
//           PARM='VERSION=22|23|24'  
//*  
//STEPLIB   DD    DSN=NearArchive.load.library,DISP=SHR  
//*  
//OTASV100  DD    DSN=Database.primary.index.name,DISP=SHR  
//OTASS032  DD    DSN=Restart.dataset.name,DISP=SHR  
//OTASIDC1  DD    DSN=AMS.parameter.file1,DISP=SHR  
//OTASIDC2  DD    DSN=AMS.parameter.file2,DISP=SHR  
//SYSPRINT  DD    SYSOUT=*  
//OTASIDCO  DD    SYSOUT=*
```

The following mandatory EXEC parameter must be specified:

**VERSION :** This parameter is used to determine which version of the NearArchive database is to be created during regression processing. The following values may be set:

- 22 - regress to a version 2.2 database.
- 23 - regress to a version 2.3 database.
- 24 - regress to a version 2.4 database.

The value of this parameter should be set to correspond to the version of NearArchive in use prior to installation of version 2.5 of the product.

Dataset names should be set as follows for each DD entry:

**STEPLIB :** The name of the ASM for S/390 product library, as unloaded during the product installation procedure (see chapter 2).

**OTASV100:** The name of the primary index of the NearArchive database to be processed by the utility. The utility will only process one NearArchive database for each execution

**OTASS032:** The name of the restart control dataset. This must be a sequential disk dataset, with record format FB (fixed blocked), and a logical record length of (k+8) bytes, where 'k' is the primary keylength of the database being



processed by the utility.

This file must be allocated prior to execution of the utility, or alternatively may be allocated by the JCL during initial execution of the utility. The file should be large enough to contain a single record.

**OTASIDC1:** The name of the first AMS parameter input file. This must be a sequential disk dataset, with record format FB (fixed blocked), and a logical record length of 80 bytes.

This file is used to supply AMS parameters to enable the utility to create a new space management file for each storage level in the database being regressed. The file must be allocated prior to execution of the utility, and should be initialized prior to use with the following AMS parameters:

```
DELETE hlq.SPACENL0
DEFINE CLUSTER -
  (NAME(hlq.SPACENL0) -
   SHR(2 3) -
   VOL(xxxxxx) -
   REC(1000 1000) -
   RECSZ(24 24) -
   NUMBERED)
```

where 'hlq' is the high-level qualifier defined for the database being regressed (as defined during initial NearArchive database definition), and 'xxxxxx' is the serial number of the disk volume on which the new space management file is to be created.

If storage level 1 has also been defined for the database being regressed, then the file should also contain the following AMS parameters:

```
DELETE hlq.SPACENL1
DEFINE CLUSTER -
  (NAME(hlq.SPACENL1) -
   SHR(2 3) -
   VOL(xxxxxx) -
   REC(1000 1000) -
   RECSZ(24 24) -
   NUMBERED)
DELETE hlq.MIGCNTLX
DEFINE CLUSTER -
  (NAME(hlq.MIGCNTLX) -
```

```
MODEL (hlq.MIGCNTL) -  
RECSZ (k+16 k+16)
```

where 'hlq' and 'xxxxxx' are as defined above, and 'k' is the database primary keylength.

**OTASIDC2:** The name of the second AMS parameter input file. This must be a sequential disk dataset, with record format FB (fixed blocked), and a logical record length of 80 bytes.

This file is used to supply AMS parameters to enable the utility to rename the existing space management file, and to rename the newly-created space management file to that of the existing file for each storage level in the database being regressed. The file must be allocated prior to execution of the utility, and should be initialized prior to use with the following AMS parameters:

```
DELETE old.sl0.space.mgmt.filename  
ALTER hlq.SPACEML0 -  
    NEWNAME (old.sl0.space.mgmt.filename)  
ALTER hlq.SM0.DATA -  
    NEWNAME (old.sl0.space.mgmt.filename.DATA)  
ALTER hlq.SM0.IX -  
    NEWNAME (old.sl0.space.mgmt.filename.IX)  
ALTER hlq.SPACENL0 -  
    NEWNAME (hlq.SPACEML0)  
ALTER hlq.SPACENL0.DATA -  
    NEWNAME (hlq.SPACEML0.DATA)
```

where:

'old.sl0.space.mgmt.filename' can be any unique dataset name and 'hlq' is the high-level qualifier defined for the database being regressed (as defined during initial NearArchive database definition).

If storage level 1 has also been defined for the database being regressed, then the file should also contain the following AMS parameters:

```
DELETE old.sl1.space.mgmt.filename  
ALTER hlq.SPACEML1 -  
    NEWNAME (old.sl1.space.mgmt.filename)  
ALTER hlq.SM1.DATA -  
    NEWNAME (old.sl1.space.mgmt.filename.DATA)  
ALTER hlq.SM1.IX -  
    NEWNAME (old.sl1.space.mgmt.filename.IX)
```

```

ALTER hlq.SPACENL1 -
      NEWNAME (hlq.SPACENL1)
ALTER hlq.SPACENL1.DATA -
      NEWNAME (hlq.SPACENL1.DATA)
DELETE old.migcntl.filename
ALTER hlq.MIGCNTL -
      NEWNAME (old.migcntl.filename.MIGCNTLY)
ALTER hlq.MIGCNTL.DATA -
      NEWNAME (old.migcntl.filename.MIGCNTLY.DATA)
ALTER hlq.MIGCNTLX -
      NEWNAME (hlq.MIGCNTL)
ALTER hlq.MIGCNTLX.DATA -
      NEWNAME (hlq.MIGCNTL.DATA)

```

where:

‘old.sl1.space.mgmt.filename’ and ‘old.migcntl.filename’ can be any unique dataset names and ‘hlq’ is the high-level qualifier defined for the database being regressed (as defined above).

**SYSPRINT:** The destination of the utility report file. If required, the report may be written to a sequential dataset, with record format = FBA, and a logical record length of 133 bytes.

**OTASIDCO:** The destination of the IDCAMS print report for execution of the AMS parameters supplied in files OTASIDC1 and OTASIDC2. If required, the report may be written to a sequential dataset, with record format = FBA, and a logical record length of 133 bytes.

## C.3 Condition Codes.

---

OTASP032 will set a condition code on completion. This code may take the following values:

- 0 - the utility has been executed successfully.
- 4 - a warning condition has been encountered during processing of the utility. A warning message will have been written to the SYSPRINT dataset. The utility has been executed successfully.
- 8 - an error condition has been encountered during processing of the utility. An error message will have been written to the SYSPRINT dataset.

- 12 - a serious error has occurred during processing of the utility, which prevented continued execution. An error message will have been written to the SYSPRINT dataset. Execution of the utility has been terminated

## **C.4 Operator commands.**

---

Execution of the utility may be terminated at any point by entering the following command through the system console:

```
(STO)P jobname
```

where 'jobname' is the name of the job used to execute the database regression utility.

The utility will acknowledge receipt of this command via message OTS03297. Commit processing will then be performed and execution of the utility terminated. Regression processing for the database may be continued at any subsequent time via a rerun of the regression job (see section C.6 for a description of rerun processing).

## **C.5 Utility Processing Reports.**

---

The following reports will be written by the database regression utility to the SYSPRINT dataset:

1. Processing summary report. This report gives details of the total number of main index entries processed from the selected database, and the total number of index entries which have been regressed.

Error or warning messages generated during execution of the utility will be written to the processing summary report. All messages will have the message identifier 'OTS032xx', where 'xx' is a numeric value in the range 00-99. Refer to chapter 8 of this manual for a description of all messages issued by the database regression utility.

## **C.6 Utility Failure and Restart Considerations.**

---

If the database regression utility fails to complete normally, either due to an error condition being encountered, execution being cancelled by the system operator, or execution of the utility being terminated early via entry of an operator 'STOP' command, then the utility may be rerun

normally, after any error condition which may have caused the initial failure has been investigated and rectified.

The utility will automatically checkpoint its activity after each 1,000 records have been processed from the database index, or immediately after receipt of an operator 'STOP' command. During rerun of the utility, processing will be restarted from the last successful checkpoint. There is no requirement to restore any dataset prior to rerunning the utility.

## C.7 Post-utility processing

---

After successful completion of the utility, if the optional step to increase the maximum record length of the primary index was performed during database conversion processing (as described in section 3.5.1), then the reverse step must now be performed to reduce the maximum record length of the primary index.

Member DB25REGR in the distributed SAMPLIB dataset contains JCL to perform this process. This job will create a new primary index file with reduced maximum record length and copy all records from the regressed primary index to the newly-created file.

Edit the job by adding a job card, and replacing all occurrences of the character string 'index.name' in the job with the name of the primary index file for the database being processed. If the new index is to be created on a different volume from that containing the regressed index file, remove the comment markers from the VOL parameter in the DEFINE CLUSTER command, and replace the string 'xxxxxx' with the serial number of the volume on which the new file is to be created. Then submit the job.

The following JCL is provided in SAMPLIB member DB25REGR:

```
//*  
//STEP1      EXEC PGM=IDCAMS  
//*  
//SYSPRINT DD  SYSOUT=*  
//*  
//SYSIN      DD  *  
  
    DELETE index.name.NEW  
  
    SET LASTCC = 0  
  
    DEFINE CLUSTER (NAME(index.name.NEW) -  
                   MODEL(index.name) -
```

```

/*          VOL (xxxxxxx) -          */
          RECSZ (16380 32760))

IF MAXCC = 0 THEN -
  REPRO IDS (index.name) -
    ODS (index.name.NEW)
IF MAXCC = 0 THEN -
  DELETE index.name

IF MAXCC = 0 THEN -
DO
  ALTER  index.name.NEW      -
    NEWNAME (index.name)
  ALTER  index.name.NEW.INDEX -
    NEWNAME (index.name.INDEX)
  ALTER  index.name.DATA      -
    NEWNAME (index.name.DATA)
END

```

The regressed database will be ready for processing by the appropriate version of NearArchive on successful completion of this job.

## APPENDIX D: Pre-V2R4 API Specifications

---

Version 2.4 of NearArchive introduced a new application program interface (API) for invocation of NearArchive services from a batch or CICS application program. All new applications which are being developed to use NearArchive for data storage and retrieval should be coded to use this new API.

ASM for S/390 will continue to support calls made using the pre-v2.4 API. Applications coded to use this API do not need modification to use version 2.5 of the product. Note, however, that some of the new product functionality introduced since version 2.3 will not be available to applications using the old API.

Appendix D provides a specification of the pre-v2.4 API.

### D.1 The batch archival interface.

---

Batch archival calls to NearArchive are made by issuing standard CALL statements to the NearArchive module 'OTASP100'. This module is supplied on the distributed ASM for S/390 load library, and must be linked in to the calling program at link-edit time, to satisfy the external reference.

The CALL should identify a parameter block, having the following general format:

<u>Offset</u>	<u>Length</u>	<u>Type</u>	<u>Description</u>
0	2	Char	Request code.
2	2	Char	Return code from NearArchive.
4	2	Bin	Supplementary request/return code 1
6	2	Bin	Supplementary request/return code 2
8	k	Char	Primary key of data to be archived (where 'k' is keylength specified at database initialization time)
8+k	6	Char	Archival date (in format YYMMDD).
14+k	2	Bin	Length (L) of data record to be archived.

16+k    L    Char   Data to be archived.

Any variations in field usage or interface layout for specific request codes will be identified in the description of those request codes in the following sections.

Note that sufficient storage must be allocated in the parameter block, either via GETMAIN processing, or via pre-allocation in working storage, to hold the largest record that will be written to the archive database.

Any unused field in the parameter interface area should be set to binary zeroes (binary fields) or spaces (character fields).

The following NearArchive request codes are available for the batch archival process:

<b>Code</b>	<b>Function</b>
-------------	-----------------

- |    |   |
|----|---|
| 10 | Open a NearArchive database for normal output processing. Data will be added to the last tape cartridge volume in the database. |
| 11 | Open a NearArchive database for rerun output processing after an archival procedure has terminated unsuccessfully.              |
| 12 | Open a NearArchive database for normal output processing. Data will be written to a new tape cartridge volume.                  |
| 13 | Write a data record to the database.  |
| 14 | Explicitly identify a secondary key to be used to index the object currently being archived.                                    |
| 15 | Close a NearArchive database for output processing.   |
| 16 | Commit all updates to a NearArchive database.   |



### **D.1.1 Request Codes 10/11/12 - Open Database.**

An 'open database' call must be issued prior to accessing a NearArchive database.

The interface format for open requests differs from the standard interface format, as described in section D.1. The amended format is as follows:

<u>Offset</u>	<u>Length</u>	<u>Type</u>	<u>Description</u>
0	2	Char	Request code
2	2	Char	Return code from NearArchive
4	2	Bin	Supplementary request/return code 1
6	2	Bin	Supplementary request/return code 2
8	2	Bin	Allocation recovery retry limit
10	4	Char	Database name
14	6	Char	Archive over-ride date (request code '11' only).

Call parameters should be set as follows:

**Request code:** '10': normal database open. This request will cause NearArchive to start writing data to the last tape cartridge volume in the database.

'11': rerun database open. This request must be used to open a NearArchive database following the failure of an archival procedure. NearArchive will reject any other type of open request if a database was not properly closed during the previous archival procedure.

'12': normal database open. This request will cause NearArchive to start writing data to a new tape cartridge volume in the NearArchive database. Use this request code if it is required to control physical data location for retrieval performance reasons.

<b>Return code:</b>	not required on entry. This field will contain a 2-character value on return from NearArchive, which indicates the result of the call (see chapter 8 for a list of possible return codes).
<b>Supplementary code 1:</b>	must be set to 1 (open for output) on entry (this is the only possible value for supplementary code 1 with these request codes).
<b>Supplementary code 2:</b>	must be set to zero on entry. May contain values on return, giving additional information after an exception condition.
<b>Allocation recovery retry limit:</b>	contains a binary value which sets the number of times tape allocation is retried after allocation recovery has been entered (due to unavailable tape drive). The default value of 30 is used if this field is set to spaces or binary zeroes.
<b>Database name:</b>	<p>name of the database to be opened. This field specifies the last 4 characters of the DD name of the database's primary index, as specified in the JCL used to execute the batch archival procedure. The first 4 characters of the DD name must always be set to the value 'OTAS'.</p> <p>The default value of 'V100' (DD name of OTASV100) is used for database name if this field is set to spaces or binary zeroes.</p>
<b>Archive over-ride date:</b>	<p>used for request code '11' only (restart open). Specifies the date (in format YYMMDD) to be used to over-ride archive dates supplied in the application's call parameter interface area for request code '13' (write data record) calls, during rerun processing (see usage note 3 below for a discussion of this facility).</p> <p>The NearArchive rerun archive date over-ride facility is not enabled if this field is set to spaces or binary zeroes.</p>

## **Usage Notes**

1. The application issuing the database open call should be capable of establishing whether a normal or rerun open request is submitted (e.g.. from EXEC or other parameter value).
2. NearArchive will reject a normal open request for a database which is in an uncommitted state, by setting a return code of '15'. Conversely, a rerun open request for a database which was updated successfully during the previous archival procedure, will be rejected by NearArchive with a return code of '18' (see chapter 8 for a full description of NearArchive return codes).
3. Use the 'archive date over-ride' field in the rerun open call (request code '11'), if the archive date specified by the calling application in the parameter interface area for 'write data record' calls (request code '13') during rerun processing differs from that specified for the same calls during initial (failed) processing.

This can occur, for example, if an application uses the system run date to identify NearArchive objects. Objects archived during a failed archival run will be identified by the run-date of the failed run; re-archiving these objects during rerun processing will cause them to be identified with a new archive date.

NearArchive will treat these objects as new archived items and generate new index entries for them, leaving the old index entries, created during the failed archival run, in place. These old index entries will no longer be valid, and any attempt to access data via them will fail.

The archive over-ride date is provided to allow the date supplied by such applications in 'write data record' calls during rerun processing, to be automatically over-ridden, so that the original archival date may be re-specified. This will ensure correct re-use of index entries during a rerun, and avoid errors due to redundant index entries.

### **D.1.2 Request Code 13 - Write Data Record.**

This call is issued to write a data record to the database. The standard interface format is used for this request. Interface parameters should be set as follows:

**Request code:** '13'

**Return code:** not required on entry. This field will contain a 2-character value on return from NearArchive, which indicates the result of the call.

**Supplementary codes 1 and 2:** not required on entry. May contain values on return, giving additional information after an exception condition.

**Primary key:** should contain the primary key of the NearArchive object of which the record being archived is a component. The length of this field must equal the primary keylength specified during database initialization. Generic key values are not permitted.

NearArchive will use any change in the value of this field during successive write requests, to identify the start of a new NearArchive object.

**Archival date:** should contain the archival date of the NearArchive object of which the record being archived is a component. This date is used in conjunction with the primary key to uniquely identify a NearArchive object for retrieval from the database.

NearArchive will use any change in the value of this field during successive write requests, to identify the start of a new data object.

The date should be a valid date in the format YYMMDD. Invalid dates will be rejected.

**Data length:** a half-word binary field which must contain the length of the data record being archived.

**Data area:** this should contain the data record to be archived to the database.

### **Usage Notes**

1. A NearArchive object can consist of one or more logical records. One 'write data record' request must be issued for each record in a NearArchive object.

NearArchive uses the primary key and archive date specified in the interface parameter area for this call to identify the NearArchive object for which the record being written is a component.

2. All records to be archived as part of a single NearArchive object must be presented to NearArchive contiguously (i.e.) in successive calls.

NearArchive will create a new primary index entry for a data object when it detects that all records in that object have been presented (i.e.) on change of primary key or archive date in the call interface parameter area. As primary index entries for a NearArchive object must be unique, any subsequent call which specifies the same primary key and archive date will be rejected by NearArchive with a return code of '17' (duplicate key).

3. The length of the data record to be archived must not exceed the maximum blocksize of the database, as specified during database initialization processing.

4. Specification of an archival date in the interface parameter area is mandatory; its omission will cause NearArchive to reject the call with a return code '14' (invalid archival date).

If the presence of an archival date is not required for an application (i.e. only one NearArchive object exists per primary key value), a dummy date value may be specified in this field. This dummy date may then be specified explicitly during retrieval, or the NearArchive 'retrieve oldest/newest object' facility may be used to ensure that the single NearArchive object for a specified primary key is retrieved. However, any dummy date used must still have a valid date value, in the format YYMMDD.

### **D.1.3 Request Code 14 - Identify Secondary Key.**

This call is used by an application program to identify, implicitly or explicitly, a secondary key that should be used to index the data object currently being archived.

For implicit identification, the interface parameters will specify that automatic secondary key identification (as defined during secondary index initialization) should be used to extract the secondary key from the immediately preceding data record passed to NearArchive for archival.

For explicit identification, the interface parameters will identify the secondary key explicitly. The standard interface format is used for this request. Interface parameters should be set as follows:

**Request code:** '14'

**Return code:** not required on entry. This field will contain a 2-character value on return from NearArchive, which indicates the result of the call.

**Supplementary**

**code 1:** on entry, will contain a 2-byte binary value in the range 1-99, which specifies the identifier of the secondary index to be updated.

On exit may contain additional information after an exception condition.

**Supplementary**

**code 2:** on entry, must be set as follows:

0 - use automatic secondary key identification rules, as defined during secondary index initialization, to extract the secondary key from the immediately preceding data record presented to NearArchive for archival.

1 - the value of the secondary key to be used is supplied in the data area in the interface parameter area. This must be of length 's' (the key length for the specified secondary index, as defined during index initialization).

Any other value will be rejected by NearArchive with a return code of '99'.

On exit may contain additional information after an exception condition.

**Primary key:** not used for this request code.

**Archival date:** not used for this request code.

**Data length:** not used for this request code.

**Data area:** supp code 2 = 0 - not used.

supp code 2 = 1 - should contain the value of the secondary key to be used to index the data object being archived.

### **Usage Notes**

1. The NearArchive object indexed via a secondary key entry generated from an 'identify secondary key' call, is the object whose primary key and archive date are identified in the last 'write data' request which preceded the call.

If no 'write data' request has yet been received by NearArchive after opening a database, the 'identify secondary key' call will be ignored.

2. Multiple 'identify secondary key' calls may be submitted contiguously to NearArchive. Each call should specify a different secondary index identifier. All contiguous calls will index the same NearArchive object (identified by the primary key and archive date specified in the preceding 'write data' request). Each call will generate a secondary key entry in the appropriate secondary index.

3. If a secondary key identified via this call has already been generated for the current NearArchive object (either through automatic secondary index processing, or through a previous type '14' request being received for that object), then the call will be ignored, and a return code of '00' generated. No special action is required by the application program in these circumstances.

### **D.1.4 Request Code 15 - Close Database.**

This call is issued after all data has been archived to the database. Failure to close the database properly at end of run will result in incorrect or missing index entries, and will prevent the database from being opened normally during the next archival run.

The standard interface format is used for this request. Interface parameters should be set as follows:

**Request code:** '15'

**Return code:** not required on entry. This field will contain a 2-character value on return from NearArchive, which indicates the result of the call.

**Supplementary**

**code 1:** on entry, specifies an option to be used for database close processing:

0 - unload the last tape cartridge volume used, after closing the database.

1 - leave the last tape cartridge volume used mounted at load point.

2 - leave the last tape cartridge volume used mounted at end-of-dataset.

On return, may contain additional information after an exception condition.

**Supplementary**

**code 2:** on entry, specifies whether updates are to be committed prior to closing the database:

0 - commit all database updates before closing the database.

1 - do not commit updates made since the last commit process, before closing database. If a database is closed without committing updates, the database must be next opened for rerun (request code '11'). Uncommitted objects will be backed out at this time.



On return, supplementary code 2 may contain additional information after an exception condition.

**Primary key:** no entry required for this call type.

**Archival date:** no entry required for this call type.

**Data length:** no entry required for this call type.

**Data area:** no entry required for this call type.

### **Usage Notes**

1. Use supplementary code 1 to control handling of the tape cartridge volume in use when the close request was issued. If the database is to be re-opened for further archival or retrieval processing, it may be more efficient to suppress unloading of the tape cartridge volume, by setting a non-zero value in supplementary code 1.

2. Set supplementary code 2 to 0 to commit all updates during close processing. This request should only be issued when archival processing has been successfully completed. Rerun of an archival process is not possible after updates have been successfully committed when closing the database.

Applications should set supplementary code 2 to a value of 1 if an error has occurred during the archival process which will necessitate a rerun. The application may then be rerun after the cause of the error has been rectified. Note that in this case the application should issue a 'rerun open' request (request code 11) to NearArchive when opening the database.

### **D.1.5 Request Code 16 - Commit Updates.**

This request is issued to commit all updates to a NearArchive database.

The standard interface format is used for this request. Interface parameters should be set as follows:

**Request code:** '16'

**Return code:** not required on entry. This field will contain a 2-character value on return from NearArchive, which indicates the result of call.

**Supplementary**

**code 1:** not used on entry. May contain values on return, giving additional information after an exception condition.

**Supplementary**

**code 2:** not used on entry. May contain values on return, giving additional information after an exception condition.

**Primary key:** no entry required for this call type.

**Archival date:** no entry required for this call type.

**Data length:** no entry required for this call type.

**Data area:** no entry required for this call type.

### **Usage Notes**

1. After a commit call has been successfully issued, any subsequent archival 'rerun open' request (call type 11) will cause the NearArchive tape dataset to be positioned at the point at which this commit call was issued, for sequential addition of data records to the database. This means that all records added after this commit call (i.e.. 'uncommitted' records) will be overwritten on tape storage, and index entries re-used.
2. Applications which issue explicit NearArchive commit calls should ensure that updates to any other database which is being processed are also committed at the same time. An application should have its own restart mechanism in the event of failure, so that processing may be restarted from the last commit point.

## D.2 The CICS retrieval interface.

---

CICS requests for retrieval of objects from a NearArchive database are issued using the CICS 'LINK' command, as follows:

```
EXEC CICS LINK PROGRAM('OTASP200')
              COMMAREA(PARMAREA)
              LENGTH(nn)
              END-EXEC.
```

where 'PARMAREA' is the user-defined storage area containing the parameter interface block used for communication between NearArchive and the calling application and 'nn' is the length of the parameter interface block passed to NearArchive.

The parameter interface block has the following format:

<u>Offset</u>	<u>Length</u>	<u>Type</u>	<u>Description</u>
0	2	Char	Request code.
2	2	Char	Return code from NearArchive.
4	2	Bin	Supplementary request/return code 1.
6	2	Bin	Supplementary request/return code 2.
8	4	Char	Database name ( = last 4 characters of CICS FCT entry for the database primary index dataset).
12	k	Char	Primary key of NearArchive object to be retrieved (where 'k' is keylength specified at database initialization time)
12+k	6	Char	Archive date of NearArchive object to be retrieved (in format YYMMDD).
18+k	8	Char	Name of temporary storage queue to be used to hold logical records from retrieved NearArchive object.
26+k	s	Char	Search key to be used in secondary index query request (where 's' is secondary keylength, as defined during secondary index initialization).

NearArchive accepts the following request codes for processing a NearArchive database from a CICS application:

<b>Code</b>	<b>Function</b>
21	Query database index (primary or secondary).
22	Retrieve NearArchive object into main or auxiliary temporary storage queue.
23	Delete NearArchive object from database.

### **D.2.1 Request Code 21 - Query Index.**

A 'query index' request is issued by an application to obtain a formatted list of index entries (primary or secondary) for a specified primary or secondary key value.

Formatted entries are returned in the auxiliary temporary storage queue specified in the call interface parameter area. One item will be present in the temporary storage queue for each index entry which satisfies the query. Index entries will be held in the temporary storage queue in the sequence requested by the calling application.

Interface parameters should be set as follows:

**Request code:** '21'

**Return code:** not required on entry. This field will contain a 2-character value on return from NearArchive, which indicates the result of the call.

**Supplementary code 1:** on entry, identifies the index to be queried, as follows:

0 - query primary index.

1-99 - query corresponding secondary index.

On exit, this field may contain a value giving additional information after an exception condition.

**Supplementary code 2:** on entry, identifies the sort sequence in which formatted index entries are to be written to temporary storage, as follows:

- 0 - ascending date within ascending primary key
- 1 - descending date within ascending primary key
- 2 - ascending date within descending primary key
- 3 - descending date within descending primary key

On exit, this field may contain a value giving additional information after an exception condition.

**Database**

**name:** Name of database to be queried. This four-character identifier should specify the last four characters of the CICS file control entry for the primary index of the database to be queried (the first four characters are always set to 'OTAS').

**Primary key:** primary search key (only required if supplementary code 1 = 0). The length (k) of this entry should be equal to the primary key length specified during database initialization processing. A full or generic primary key value may be specified in this field.

Generic key specifications consist of a string of characters of length 0 to (k-1) bytes (where 'k' is the database primary keylength), followed by an asterisk ('\*') and trailing spaces, if required. The total field length must always be equal to the database's primary keylength. NearArchive will return each object whose primary key value begins with the character string preceding the asterisk.

(e.g.) if a database has a primary keylength of 8 characters, a generic primary key specification of 'SMITH\*∇∇' could return the following primary key values:

'SMITH∇∇∇∇'  
 'SMITHERS'  
 'SMITHSON' etc.

A generic key specification consisting of a single asterisk followed by (k-1) spaces will cause NearArchive to return all objects in the database.

**Archival date:** this field is used by the query interface to control the maximum number of objects which may be returned in response to a query request. The field is sub-divided into two sub-fields, as follows:

Bytes 0-1: This is a half-word binary field, which should contain the maximum number of objects (i.e. maximum number of items in the target TSQ) to be generated by NearArchive in response to a query request.

This must be a binary value in the range 0-32767. If the field contains low-values or EBCDIC spaces, the default value of 32767 is used.

If the number of objects satisfying a query request exceeds the maximum specified in this field, NearArchive will generate a return code of '04' on return to the calling program. The target TSQ will contain the identifiers of the first 'n' objects satisfying the query request, where 'n' is the maximum value set on entry to NearArchive.

On return from NearArchive, for return codes '00' or '04', this field will contain the actual number of items in the target TSQ.

Bytes 2-5: The remainder of the archival date field is not used for this request code.

**TSQ name:** the name of the auxiliary temporary storage queue which NearArchive will create to hold the formatted index entries to be returned from the query request.

**Secondary key:** specifies the secondary search key, if supplementary code 1 is not = 0. Unused for primary key queries. A full or generic secondary key value may be specified in this field.

Generic key specifications consist of a string of characters of length 0 to (s-1) bytes (where 's' is the secondary keylength), followed by an asterisk (\*) and

trailing spaces, if required. The total field length must always be equal to the keylength of the secondary index identified in supplementary code 1. NearArchive will return each object whose secondary key value begins with the character string preceding the asterisk.

(e.g.) if secondary index '05' has been defined for a database with keylength of 6 characters, a generic secondary key specification of 'ABC\*VV' could return the following primary key values:

'ABCVVV'  
'ABCDVV'  
'ABCDEV'  
'ABCDEF' etc.

A generic key specification consisting of a single asterisk followed by (s-1) spaces will cause NearArchive to return all objects in the database which are indexed in the specified secondary index.

For successful 'query index' requests, a temporary storage queue, with an identifier as specified in the call interface parameters, will have been created by NearArchive. This will contain one item for each index entry satisfying the query criteria. Index entries will appear in the sequence specified by the calling application.

Each item will be (k+16) bytes in length, (where 'k' is the database primary keylength), and will have the following format:

<u>Offset</u>	<u>Length</u>	<u>Type</u>	<u>Description</u>
0	k	char	Primary key of NearArchive object. If the query was based on secondary key, this field will contain the primary key of a NearArchive object which is indexed by the specified secondary key.
k	6	char	Archive date of NearArchive object (in format YYMMDD).
k+6	2	bin	Number of tape blocks used to hold NearArchive object.

k+8	4	bin	Identifier of first block on volume containing NearArchive object.
k+12	2	bin	Sequence number of tape dataset in database which contains the NearArchive object.
k+14	1	bin	Hierarchical level in which object is stored (0 or 1).
k+15	1	char	Object residency indicator: blank - tape only 'D' - tape + disk

### **Usage Notes**

1. The temporary storage queue which holds the returned index entries will be created by NearArchive. Any queue with the same identifier which existed prior to the call will be deleted.

This means that applications do not need to be responsible for deletion of temporary storage queues prior to use. In particular, repeated index query requests may be issued by an application, without requiring deletion of temporary storage queues between requests.

2. After a successful call, the number of items in the temporary storage queue will be equal to the number of primary or secondary index entries which satisfy the query request criteria.

3. Index entries will be held in the temporary storage queue in the sort sequence specified by the calling application in supplementary code 2.

4. Note that no tape access is required to satisfy an index query request.

5. An entry must exist in the CICS File Control Table (FCT) for the primary or secondary index being queried.



### **D.2.2 Request Code 22 - Retrieve Object.**

A single 'retrieve object' request is issued by a CICS application in order to retrieve from a NearArchive database all logical records for a specified NearArchive object.

Logical records are returned by NearArchive to the application as separate items in the temporary storage queue whose identifier is supplied by the application in the NearArchive parameter interface area.

One item will exist in the temporary storage queue for each logical record which belongs to the specified NearArchive object. Logical records will be held in the temporary storage queue in the same sequence in which they are stored physically in the database; this is the sequence in which they were passed to NearArchive when written to the database.

Interface parameters should be set as follows:

**Request code:** '22'

**Return code:** not required on entry. This field will contain a 2-character value on return from NearArchive, which indicates the result of the call.

**Supplementary**

**code 1:** on entry, used to identify the type of temporary storage queue to be used to hold the retrieved NearArchive object:

0 - auxiliary temporary storage queue

1 - main temporary storage queue.

On exit, this field may contain a value giving additional information after an exception condition.

**Supplementary**

**code 2:** not used on entry. On exit, may contain a value giving additional information after an exception condition.

**Database name:** Name of database to be accessed. This four-character identifier should specify the last four characters of the CICS file control entry for the primary index of the database to be queried (the first four characters are always set to 'OTAS').

**Primary key:** primary key of NearArchive object to be retrieved. The length (k) of this entry should be equal to the primary key length specified during database initialization.

**Archival date:** archive date (in format YYMMDD) of NearArchive object to be retrieved.

If the archive date on entry is set to '000000', NearArchive will return the oldest object with the specified primary key. If the archive date on entry is set to '999999', NearArchive will return the newest object with the specified primary key value. In each of these cases, the archival date field on return from NearArchive will contain the actual archive date of the retrieved object.

**TSQ name:** the name of the temporary storage queue in which NearArchive will place the logical records from the retrieved NearArchive object.

**Secondary key:** not used for this request code.

### **Usage Notes**

1. The temporary storage queue to be used by NearArchive to hold a retrieved object will be created by NearArchive if it did not exist prior to the request. If the temporary storage queue was already in existence, NearArchive will append items to this queue for each logical record retrieved from the NearArchive database by the new request.
2. NearArchive will automatically delete temporary storage queues containing retrieved objects if a non-zero value for the TSQDELETE initialization parameter has been specified. If the TSQDELETE parameter is set to zero, then it is the responsibility of the calling application to delete the temporary storage queue used to hold a retrieved NearArchive object.
3. All tape and disk access operations will be performed from the NearArchive control region. If an application is not executing in a NearArchive control region, its executing CICS region must be linked to a NearArchive control region using standard CICS Multiple Region Operation (MRO) linkage facilities. Note that use of MRO facilities is not mandatory for NearArchive operation.

4. The NearArchive control region must be initialized prior to retrieval requests being issued by applications from within that region, or from within any connected MRO region.

NearArchive will set a return code of '21' for retrieval requests which have been issued prior to NearArchive control region initialization.

5. The maximum number of items which may be held in a CICS temporary storage queue is 32767. NearArchive will set a return code of '23' if a NearArchive object containing greater than this number of logical records is retrieved from a CICS application.

6. If a disk copy exists for an object which is to be retrieved (in addition to a tape copy), then NearArchive online retrieval processing will always access the disk copy. If this access fails for any reason, NearArchive will automatically retrieve the tape copy of the object in the standard manner.

### **D.2.3 Request Code 23 - Delete Object.**

This call is issued by a CICS application to delete an object from a NearArchive database.

The standard interface format is used for this request. Interface parameters should be set as follows:

**Request code:** '23'

**Return code:** not required on entry. This field will contain a 2-character value on return from NearArchive, which indicates the result of the call.

**Supplementary**

**code 1:** not used on entry. May contain a value on return, giving additional information after an exception condition.

**Supplementary**

**code 2:** not used on entry. May contain a value on return, giving additional information after an exception condition.

**Database name:** name of database to be accessed. This four-character identifier should specify the last four characters of the CICS file control entry for the primary index of the

database to be queried (the first four characters are always set to 'OTAS').

**Primary key:** primary key of NearArchive object to be deleted.

**Archival date:** date of NearArchive object to be deleted.

**TSQ name:** not used for this request code.

**Secondary key:** not used for this request code.

### **Usage Notes**

1. The primary index of the NearArchive database being accessed must be available for update processing by the CICS application. This file will have a CICS File Control Table entry name of 'OTASxxxx', where 'xxxx' is the database name specified in the NearArchive call parameters.

A return code of '89' and supplementary code 1 value of X'0100' will be set by NearArchive in response to a deletion request for a database whose primary index is available for input processing only.

2. If the NearArchive audit trail facility has been enabled for the database being accessed, then the database journal file must be available for output processing by the CICS application. This file will have a CICS File Control Table entry name of 'OJRNxxxx', where 'xxxx' is the database name specified in the NearArchive call parameters.

A return code of '89' and supplementary code 1 value of X'0300' will be set by NearArchive in response to a deletion request for a database whose journal file is not available for output processing.

3. After successful deletion, a NearArchive object will no longer be available for retrieval. Space used by a deleted object's primary and secondary index entries will be reclaimed during execution of the NearArchive object management utility (OTASP040). Tape and disk space used by the deleted object's logical records will be reclaimed during execution of the NearArchive database maintenance utility (OTASP030).

## D.3 The batch retrieval interface.

---

Calls to NearArchive for batch retrieval are identical in format to those for batch archival (i.e.) a standard program CALL is made to the NearArchive module 'OTASP100'. This module is supplied on the distributed NearArchive load library, and must be linked in to the calling program at link-edit time, to satisfy the external reference.

The standard batch retrieval CALL parameter block layout is similar to that for batch archival. It contains the following fields:

<u>Offset</u>	<u>Length</u>	<u>Type</u>	<u>Description</u>
0	2	Char	Request code.
2	2	Char	Return code from NearArchive.
4	2	Bin	Supplementary request/return code 1
6	2	Bin	Supplementary request/return code 2
8	k	Char	Primary key of data to be retrieved (where 'k' is keylength specified at database initialization time)
8+k	6	Char	Archival date (in format YYMMDD).
14+k	2	Bin	Length (L) of retrieved data record (set by NearArchive on return)
16+k	L	Char	Retrieved data record.

The block should be large enough to hold the longest data record on the archive database in the 'retrieved data record' field. Unused fields in the parameter interface area should be set to binary zeroes (for binary fields) or spaces (for character fields).

Any variations in field usage or interface layout for specific request codes will be identified in the description of those request codes in the following sections.

The following request codes are available for the batch retrieval process:

**Code    Function**

- 30    Open a NearArchive database for input or input-output processing.
- 31    Query database index (primary or secondary).
- 32    Retrieve data record (direct or sequential).
- 33    Delete NearArchive object from database.
- 34    Close a NearArchive database for input or input-output processing.

**D.3.1 Request Code 30 - Open Database.**

An 'open database' request must be issued prior to any retrieval or delete request to a NearArchive database.

The interface format for open requests differs from the standard interface format, as described in section D.3. The amended format is as follows:

<u>Offset</u>	<u>Length</u>	<u>Type</u>	<u>Description</u>
0	2	Char	Request code
2	2	Char	Return code from NearArchive
4	2	Bin	Supplementary request/return code 1
6	2	Bin	Supplementary request/return code 2
8	2	Bin	Allocation recovery retry limit
10	4	Char	Database name
14	4	Char	Tape DDname over-ride.

Call parameters should be set as follows:

**Request code:**    '30'

**Return code:**    not required on entry. This field will contain a 2-character value on return from NearArchive, which indicates the result of the call.

**Supplementary**

**code 1:** This must contain one of the following values:

1 - open database for input processing (no delete calls may be issued by the application).

2 - open database for input-output processing (delete calls may be issued by the application).

On exit, this field may contain supplementary error information.

**Supplementary**

**code 2:** must be set to binary zeroes on entry. May contain a value on return giving additional information after an exception condition.

**Allocation recovery**

**retry limit:** Contains a binary value which sets the number of times tape allocation is retried after allocation recovery has been entered (due to unavailable tape drive). The default value of 30 is used if this field is set to binary zeroes or spaces.

**Database name:** Name of the database to be opened. This field specifies the last 4 characters of the DD name of the database's primary index, as specified in the JCL used to execute the batch retrieval procedure. The first 4 characters of the DD name must always be set to the value 'OTAS'.

The default value of 'V100' (DD name of OTASV100) is used for database name if this field is set to spaces or binary zeroes.

**Tape DDname**

**over-ride:** Specifies the last four characters of the Ddname to be used by NearArchive when dynamically allocating a tape dataset for data retrieval. The first four characters of the DDname are always set to 'OTAS'.

The default value of 'T100' is used if this field is set to spaces or low-values.

**Usage notes**

1. An application should only issue an input-output open request (supplementary code 1 = 2), if it requires to delete objects from the NearArchive database. Unnecessary use of input-output processing may reduce batch job throughput, by enforcing serialisation of jobs which require update access to a NearArchive database.
2. A request to open a NearArchive database for input-output processing will fail if the database is already open for update by another job (batch or CICS). In these circumstances, the open request will receive a return code of '94', with supplementary codes 1 and 2 set to X'0100' and X'08A8' respectively.

If this occurs, the job should be re-submitted when the database is available for update processing.

3. An 'open database' request must be the first request issued when processing a NearArchive database. All other types of request will be rejected if a database has not been opened successfully.
4. The final three fields in the parameter interface area replace those fields specified with request code '35' in version 1 of the product. Request code '35' is no longer in use.

### **D.3.2 Request Code 31 - Query Index.**

A 'query index' request is issued by an application to obtain a formatted list of index entries (primary or secondary) for a specified primary or secondary key value.

The maximum number of formatted entries which may be accepted by the application is set in the 'data length' field when the call is issued. On return from NearArchive, this field will hold the actual number of formatted entries present in the data area field in the parameter block.

A return code of '00' and supplementary code 1 value of 4 is issued when there are more entries to be returned. Further query requests may then be issued, specifying the same query criteria, to retrieve the next set of formatted index entries for that query.

The standard interface format is used for this request. Interface parameters should be set as follows:

**Request code:** '31'

**Return code:** not required on entry. This field will contain a 2-character value on return from NearArchive, which indicates the result of the call.



### **Supplementary**

**code 1:** on entry, identifies the index to be queried, as follows:

- 0 - query primary index.
- 1-99 - query corresponding secondary index.

On exit, this field may contain a value giving additional information after an exception condition.

### **Supplementary**

**code 2:** on entry, identifies the sort sequence in which formatted index entries are to be returned, as follows:

- 0 - ascending date within ascending primary key
- 1 - descending date within ascending primary key
- 2 - ascending date within descending primary key
- 3 - descending date within descending primary key

On exit, this field may contain a value giving additional information after an exception condition.

**Primary key:** primary search key (only required if supplementary code 1 = 0). The length (k) of this field should be equal to the primary key length specified during database initialization processing. A full or generic primary key value may be specified in this field.

Generic key specifications consist of a string of characters of length 0 to (k-1) bytes (where 'k' is the database primary keylength), followed by an asterisk (\*) and trailing spaces, if required. The total field length must always be equal to the database's primary keylength. NearArchive will return each object whose primary key value begins with the character string preceding the asterisk.

(e.g.) if a database has a primary keylength of 8 characters, a generic primary key specification of 'SMITH\*VV' could return the following primary key values:

'SMITHVVV'  
'SMITHERS'

‘SMITHSON’ etc.

A generic key specification consisting of a single asterisk followed by (k-1) spaces will cause NearArchive to return all objects in the database.

**Archival date:** not required for this call type.

**Data length:** on entry, gives the maximum number of formatted items (each k+16 bytes in length) which the application can accept in the following field.

On exit, gives the actual number of formatted items in the following field.

**Data area:** On entry, specifies the secondary key value to be queried, if supplementary code 1 is not = 0. Unused on entry for primary key queries.

A full or generic secondary key value may be specified in this field.

Generic key specifications consist of a string of characters of length 0 to (s-1) bytes (where ‘s’ is the secondary keylength), followed by an asterisk (‘\*’) and trailing spaces, if required. The total field length must always be equal to the keylength of the secondary index identified in supplementary code 1. NearArchive will return each object whose secondary key value begins with the character string preceding the asterisk.

(e.g.) if secondary index ‘05’ has been defined for a database with keylength of 6 characters, a generic secondary key specification of ‘ABC\*VV’ could return the following primary key values:

‘ABCVVV’  
‘ABCDVV’  
‘ABCDEV’  
‘ABCDEF’ etc.

A generic key specification consisting of a single asterisk followed by (s-1) spaces will cause NearArchive to return all objects in the database which are indexed in the specified secondary index.

On exit, this field will contain one formatted index entry for each NearArchive object which satisfies the query, up to the limit specified in the preceding field.

Each entry is (k+16) bytes long (where 'k' is the database primary keylength), and has the following format:

<u>Offset</u>	<u>Length</u>	<u>Type</u>	<u>Description</u>
0	k	char	Primary key of NearArchive object. If the query was based on secondary key, this field will contain the primary key of a NearArchive object which is indexed by the specified secondary search key.
K	6	char	Archive date of NearArchive object (in format YYMMDD).
k+6	2	bin	Number of tape blocks used to hold NearArchive object.
k+8	4	bin	Identifier of first block on volume containing NearArchive object.
k+12	2	bin	Sequence number of tape dataset in database which contains the NearArchive object.
k+14	1	bin	Hierarchical level in which data is stored (0 or 1).
k+15	1	char	Object residency indicator: blank- tape only 'D' - tape + disk

### **Usage notes**

1. An application should always set the 'data length' parameter field to the maximum number of formatted entries which it can receive in the data area field, prior to issuing a 'query index' call. Failure to do this may result in storage corruption due to NearArchive exceeding storage bounds within the calling program.
2. Where the number of formatted index entries satisfying a query exceeds the maximum ('n') which a calling program has indicated that

it will accept, NearArchive will place the first 'n' entries in the data area in the parameter interface block, and set a return code of '00', and supplementary code 1 value of 4.

In these circumstances, the calling program should issue a further 'query index' request, specifying the same query criteria as in the first request. NearArchive will then return the next 'n' formatted entries in the supplied data area.

When all index entries which satisfy a query have been returned, NearArchive will set a return code of '00' and supplementary code 1 value of 0. Note that in this case, the number of returned entries may be less than 'n'; the data length will be reset to indicate the number of returned entries ('m') in the data area. In these circumstances, calling applications should only process the first 'm' entries; the contents of the remaining ('n-m') entries will be unchanged from the contents on entry to NearArchive.

3. Submission of a repeat 'query index' request for a key after all entries for the previous request have been returned (and where there have been no intermediate query index calls), will result in the first 'n' items for that query being once again returned to the caller.

### **D.3.3 Request Code 32 - Read data record.**

This call is issued by an application to retrieve the next data record from a NearArchive object. Supplementary code 1 is used to indicate whether a sequential or direct read is required.

For a sequential read, the next logical record stored physically on the tape is retrieved.

For a direct read, a change of key or date in the parameter area will cause that NearArchive object to be located and its first data record retrieved. Subsequent direct reads, using the same key and date values, will cause successive data records from that NearArchive object to be retrieved.

For direct reads, return code '30' is used to indicate that all records belonging to a NearArchive object have been retrieved. The application may then continue to issue direct or sequential reads as required.

The standard interface format is used for this request. Interface parameters should be set as follows:

**Request code:** '32'

**Return code:** not required on entry. This field will contain a 2-character value on return from NearArchive, which indicates the result of the call.

**Supplementary code 1:** on entry, identifies the type of read request

0 - direct read. For direct reads, the primary key and archive date of the NearArchive object to be retrieved are supplied by the application in the appropriate parameter fields.

1 - sequential read. For sequential reads, the next record is retrieved from the database (in ascending physical sequence). The primary key and archive date of the NearArchive object to which the retrieved record belongs is returned by NearArchive in the appropriate parameter fields.

This field may contain a value on return, giving additional information after an exception condition.

**Primary key:** primary key of NearArchive object to be accessed.

For a direct read request (supplementary code 1 = 0), this field must contain, on entry, the primary key of the NearArchive object to be retrieved. Not used on entry for sequential reads.

For a sequential read request (supplementary code 1 = 1), set on exit by NearArchive with the primary key of the NearArchive object to which the returned record belongs. Not used on exit for direct reads.

**Archival date:** date of NearArchive object to be accessed.

For a direct read request (supplementary code 1 = 0), this field must contain, on entry, the archive date of the NearArchive object to be retrieved. Not used on entry for sequential reads. If the archive date on entry

is set to '000000', NearArchive will return the oldest object with the specified primary key. If the archive date on entry is set to '999999', NearArchive will return the newest object with the specified primary key value.

For a sequential read request (supplementary code 1 = 1), set on exit by NearArchive with the archive date of the NearArchive object to which the returned record belongs. Not used on exit for direct reads.

**Data length:** on entry, gives the maximum record length which the application can accept in the following field.

On exit, gives the actual length of the data record in the following field. If the retrieved record is longer than the maximum specified on entry, it will be truncated, and a return code of '04' set to indicate this occurrence. In this case, this field will give the truncated record length.

**Data area:** contains the retrieved record on return from NearArchive after a successful call.

### **Usage notes**

1. Applications can issue a combination of direct and sequential retrieval requests as required, in any order.
2. A direct retrieval request will cause repositioning in the NearArchive database on change of primary key or archive date. Further direct requests which specify the same primary key and archive date will cause successive records to be retrieved from the identified object. A return code of '30' will be set by NearArchive when all logical records for a NearArchive object have been retrieved.
3. A sequential retrieval request will cause the next physical sequential record, from the current position, to be retrieved from the database. The primary key and archive date of the NearArchive object to which the retrieved record belongs will be returned by NearArchive in the call parameter area.
4. No indication is given by NearArchive that an object boundary has been crossed during sequential retrieval. It is the responsibility of the calling application to identify a change of primary key or date in the returned parameter area to establish that an object boundary has been crossed.

5. Current position for a sequential retrieval request may be established initially by a direct retrieval request. The current position is updated during sequential retrieval processing.

Where no current position has been established prior to the first sequential retrieval request being received by NearArchive after a database has been opened, the first active record in the lowest dataset sequence number in storage level 0 of the database will be returned.

6. During sequential processing, when the last record in storage level 0 of a database has been retrieved, the next sequential retrieval request will return the first record in the lowest dataset sequence number in storage level 1 of the database. No special indication is given by NearArchive that a storage level boundary has been crossed during sequential retrieval.

If storage level 1 is not defined for a database in the above circumstances, or if the last record in storage level 1 has been sequentially retrieved, NearArchive will set a return code of '30', and a supplementary code 1 value of 1 if another sequential retrieval request is submitted. This indicates an 'end of database' condition. If the application continues to submit sequential retrieval requests, NearArchive will start sequential processing from the beginning of the database again.

7. For direct processing requests, the disk ('K') copy of an object will always be retrieved if it exists. If disk retrieval fails for any reason, NearArchive will retrieve the object from tape as normal.

For sequential processing requests, NearArchive will always access the tape copy of an object, whether a disk copy exists or not. The order in which objects are accessed during sequential processing will be governed by the physical order of storage of objects on tape.

If a direct retrieval request is satisfied from the disk copy of an object, and is then followed by a sequential retrieval request, NearArchive will access the next object stored in physical sequence following the tape copy of the object which had been directly retrieved. Sequential processing will continue by retrieving tape copies of objects from that point onwards.

### **D.3.4 Request Code 33 - Delete Object.**

This call is issued by an application to delete an object from a NearArchive database. The database must be open for input-output processing for a delete request to be accepted. The standard interface format is used for this request. Interface parameters should be set as follows:

**Request code:** '33'

**Return code:** not required on entry. This field will contain a 2-character value on return from NearArchive.

**Supplementary code 1:** not used on entry. May contain a value on return, giving additional information after an exception condition.

**Supplementary code 2:** not used on entry. May contain a value on return, giving additional information after an exception condition.

**Primary key:** primary key of NearArchive object to be deleted.

**Archival date:** date of NearArchive object to be deleted.

**Data length:** not used for this request code.

**Data area:** not used for this request code.

### **Usage notes**

1. The NearArchive database must be opened for input-output processing (request code = '30', supplementary code 1 = 2) prior to issuing a batch deletion request.

A return code of '89' will be set by NearArchive in response to a deletion request for a database which has been opened for input processing only.

2. After successful deletion, a NearArchive object will no longer be available for retrieval. Space used by a deleted object's primary and secondary index entries will be reclaimed during execution of the NearArchive object management utility). Tape and disk space used by



the deleted object's logical records will be reclaimed during execution of the NearArchive database maintenance utility (OTASP030).

### **D.3.5 Request Code 34 - Close database.**

This call is issued by an application after all retrieval and update operations for a NearArchive database have been completed.

The standard interface format is used for this request. Interface parameters should be set as follows:

**Request code:** '34'

**Return code:** not required on entry. This field will contain a 2-character value on return from NearArchive, which indicates the result of the call.

**Supplementary code 1:** on entry, specifies an option to be used for database close processing:

0 - unload the last tape cartridge volume used, after closing the database.

1 - leave the last tape cartridge volume used mounted at load point.

2 - leave the last tape cartridge volume used mounted at end-of-dataset.

On return, may contain additional information after an exception condition.

**Supplementary code 2:** not used on entry. May contain values on return, giving additional information after an exception condition.

**Primary key:** no entry required for this call type.

**Archival date:** no entry required for this call type.

**Data length:** no entry required for this call type.

**Data area:** no entry required for this call type.

### **Usage notes**

1. Use supplementary code 1 to control handling of the tape cartridge volume in use when the close request was issued. If the database is to be re-opened for further retrieval processing from the point at which it was closed, it may be more efficient to suppress unloading of the tape cartridge volume by setting a non-zero value in supplementary code 1.
2. No further access to the NearArchive database will be possible after a close request has been successfully processed.

## **APPENDIX E: Managing Disk ('K') Copy Capacity**

---

This discussion of capacity applies **strictly** to disk ('K') copy files, as defined in the NearArchive database definition. It specifically does **not** apply to index, space management and other work files used by ASM for S/390.

Note also that the issue of extended disk dataset capacity discussed in this appendix may be avoided by creating multiple disk datasets for each corresponding primary tape volume. Refer to section 3.1 for information on restricting the size of each disk copy dataset and creating multiple disk datasets per primary tape volume. Up to 256 disk copy datasets may be created for each primary tape copy dataset.

### **E.1 Disk ('K') Copy Capacity**

---

ASM for S/390 uses conventional BSAM techniques to process disk ('K') copy data sets, with random-access repositioning via NOTE/POINT. The maximum capacity of a disk data set will depend upon whether the structure chosen is conventional non-SMS (Physical Sequential) or the enhanced functionality of SMS (Extended Format Data Sets and/or striping).

#### **E.1.1 Conventional Physical Sequential (non-SMS)**

PS disk ('K') copy data sets are restricted to a **single volume** and a maximum of 65,535 tracks. This is because the 3-byte TTR used by BSAM is reset to 000001 when volume switching occurs, in which case objects on the second and subsequent volumes are **not** retrievable. See table E.1 for examples of capacity for different devices and block sizes.

#### **E.1.2 Extended Format Data Sets (SMS)**

EFDS disk ('K') copy data sets are restricted to a **single volume**. This is because the 3-byte BLT used by BSAM is reset to 000001 when volume switching occurs, in which case objects on the second and subsequent volumes are **not** retrievable. See table E.1 for examples of capacity for different devices and block sizes. Implementation requires an SMS volume and ACS routines (Dataclas + Storclas) for 'K' copy allocation.

### **E.1.3 Striped EFDS (SMS)**

Striped EFDS disk ('K') copy data sets are restricted to 16 stripes (volumes) and a maximum of 16,777,214 blocks. See table E.1 for examples of capacity for different devices and block sizes. Implementation requires SMS volumes and ACS routines (Dataclas + Storclas) for 'K' copy allocation.

C A P A C I T Y (Gb)				
		28k blksize	1k blksize	
		(2/track)	(34/track)	
PS (non-SMS)	3390-3		2.8	1.7
	-9	3.7	2.2	
EFDS	3390-3	2.8	1.7	
	-9	<b>8.4</b>	<b>5.1</b>	
Striping	3390-3	<b>44.8</b>	<b>16.7</b>	
	-9	<b>134</b>	<b>16.7</b>	

**Table E.1:** Maximum disk ('K') copy capacity by file type, device type and blocksize.

## **E.2 Disk ('K') Copy Blocksize Considerations**

To maximize utilization of disk space, you should consider carefully the choice of disk blocksize for each NearArchive database. Under no circumstances should you exceed half-track blocksize (23,476 for 3380, 27,998 for 3390). Additionally, keep in mind that when written to disk, NearArchive blocks are **not** shared between archived objects. That is, each object will occupy 1 or more unique blocks. Therefore the optimum blocksize would be one that is slightly larger than average object size, not to exceed the upper limit of half-track blocksize.

Additional considerations apply to EFDS. While ASM for S/390 writes variable-blocked records logically, SMS writes fixed length blocks physically, expanding short blocks to the actual blocksize with padding bytes. For example, if blocksize were set to 1000 and an object of 1020 is archived, 2 blocks of 1000 each would be written to disk, essentially

“wasting” 980 bytes on the track. Therefore, choose blocksize carefully, based upon the characteristics of the majority of archived objects.

When choosing blocksize, be sure to remember that smaller blocksize also reduces the capacity of the track. Half-track blocking (2/track) retains 99% capacity of the track. But 2k blocks retain only about 68%, 1k blocks about 52% and .5k blocks only about 35%.

## **E.3 SMS Requirements for EFDS/Striping**

---

### **E.3.1 Data Class Definition for EFDS/Striping**

A new data class for EFDS/Striping must be established, if a suitable one does not already exist. When defining this data class with ISMF, on panel 3, the **DataSetNameType** field **must** be set to **EXT** and the **IfExtended** field **must** be set to **R**. To verify that the data set has been properly defined, view the newly allocated file with ISPF 3.4. The Dsorg field should show **PS-E** for EFDS.

If compression(compaction) is desired, it can be selected on Panel 2 by specifying **Y** for **Compaction**.

### **E.3.2 Sample ACS Routine for Data Class Selection**

In the following example, *SEQEXT* represents a previously defined EFDS data class and *EXAMPLE.HLQ* represents the HLQ as setup in the NearArchive database definition panels. *NEARARCHIVE* represents a variable within the ACS routine and may be chosen according to customer requirements.

Please note that ONLY the ‘K’ copy should be created as EFDS. The ACS routine must be written so that other NearArchive data sets are allocated normally.

```
PROC DATACLAS

FILTLIST NEARARCHIVE INCLUDE(EXAMPLE.HLQ.K%.**)

IF &DATACLAS NE ‘ ’ THEN EXIT

SELECT (&DSN)

      WHEN (&NEARARCHIVE) SET &DATACLAS = ‘SEQEXT’
```

```

        OTHERWISE                SET &DATACLAS = ‘ ’

    END

END

```

### **E.3.3 Storage Class Definition for EFDS/Striping**

If the multiple volume capacities of **Striping** are desired, a new storage class for striping must be established, if a suitable one does not already exist. When defining this storage class with ISMF, **SustainedDataRate** on panel 1 **must** be set **greater than 1**. The value specified must be set to a multiple of the desired number of stripes (the maximum number of stripes is 16). For 3380 the multiple is 3, for 3390 the multiple is 4. For example, a SustainedDataRate of 36 would yield 9 stripes on 3390, or 12 stripes on 3380. Please consult DFSMSdfp Storage Administration Reference for detailed information. To utilize EFDS only (without Striping), SustainedDataRate must be allowed to default to blank/0.

Please be aware that the disk (‘K’) copy primary space allocation, as specified on the NearArchive database definition panels, will be equally spread across the striped volumes. Additionally, secondary allocations will occur only on the original striped volumes.

### **E.3.4 Sample ACS Routine for Storage Class Selection**

In the following example, *STRIPED* represents a previously defined storage class and *EXAMPLE.HLQ* represents the HLQ as setup in the NearArchive database definition panels. *NEARARCHIVE* represents a variable within the ACS routine and may be chosen according to customer requirements.

Please note that ONLY the ‘K’ copy should be striped. The ACS routine must be written so that other NearArchive data sets are allocated normally.

```

PROC  STORCLAS

FILTLIST  NEARARCHIVE  INCLUDE(EXAMPLE.HLQ.K%.**)

IF &STORCLAS NE ‘ ’ THEN EXIT

SELECT (&DSN)

        WHEN (&NEARARCHIVE)  SET &STORCLAS = ‘STRIPED’

```

```
OTHERWISE          SET &STORCLAS = ‘  
END  
END
```

---

# INDEX

---

3  
3480 7  
3490 7  
3590 4, 7  
9  
9840 7  
A  
access authority 5  
Access Method Services 21  
age range of tapes retained on drive 8,  
31  
allocation recovery 11, 23  
allocation recovery retry limit 11, 23  
alter NearArchive configuration 26  
AMS 21, 33, 39, 48  
APF-authorized 24  
API 3  
application program 3  
application program interface 3, 4, 10  
application program interface, pre-V2R4  
1, 3  
archival 6  
archival application 7  
archive date 3, 7, 21, 33  
archive date over-ride 10, 11, 13  
archiving busy 9, 13  
Assembler 6  
audit trail 11, 18, 29, 38, 45  
authorization code 7  
automatic indexing 38  
autouncat 16, 19, 25  
auxiliary temporary storage 28  
B  
1.backup 50  
full-volume  
.....  
13, 58, 59, 61, 71  
incremental  
.....  
14, 58, 60, 61, 72  
backup copy 13, 18, 55, 56, 59, 70

backup utility 14  
batch archival 4, 11  
interface  
.....  
11  
tape I/O driver messages  
.....  
101  
batch processing 3  
batch retrieval 21, 23  
batch retrieval  
interface  
.....  
11  
C  
CI size 24  
CICS  
batch retrieval  
.....  
33  
CICS 'LINK' command 4  
CICS communication area 4  
CICS control region 12  
initializing  
.....  
13  
quiescing  
.....  
13  
CICS definitions 15  
journal  
.....  
18  
primary index  
.....  
16, 22  
secondary index  
.....  
17  
space management



.....  
 16, 17  
 CICS housekeeping messages 131  
 CICS operator interface 14  
 CICS operator messages 114  
 CICS processing 12  
 CICS reader task messages 112  
 CICS shutdown processing 10  
 COBOL 6  
 COBOL application 1  
 commit block 15, 39  
 commit request 10  
 commit updates 39, 44  
 compress threshold 16  
 concurrent retrievals 10, 11  
 connect database 8, 13  
 control interval size 24, 41  
 3.control region 12, 23, 31  
     initialization messages  
     .....  
     106  
     quiesce messages  
     .....  
     111  
     secondary  
     .....  
     13  
 copy identifier 55  
 D  
 data integrity 11  
 data retention period 29, 36  
 database 4  
     administration  
     .....  
     5  
     backup  
     .....  
     12  
     backup utility  
     .....  
     21, 58  
     blocksize  
     .....  
     13

browse definition  
 .....  
 49  
 connect  
 .....  
 8  
 conversion  
 .....  
 30, 57  
 definition  
 .....  
 5, 7, 10  
 definition and maintenance  
 .....  
 3  
 initialization  
 .....  
 7  
 initialization messages  
 .....  
 134  
 maintenance  
 .....  
 8, 16, 17, 28, 29  
 maintenance utility  
 .....  
 17, 18, 20, 39  
 maintenance utility messages  
 .....  
 23, 26, 29, 45  
 recovery  
 .....  
 18  
 recovery utility  
 .....  
 17, 21, 50, 70  
 restore  
 .....  
 20  
 database backup utility messages 74  
 database design 5  
 database name 9  
 database name list 9  
 database protection 5

database recovery utility messages 89  
 DB25CONV 59  
 DB25REGR 7  
 DBCS 3, 5, 7, 15, 25, 29, 32, 35, 37,  
 38  
 DCT 22  
 destination control table 22  
 direct read 24  
 direct retrieval 22, 26  
 direct to sequential processing 22  
 disaster recovery 5, 17, 51, 56  
 disaster recovery  
     disable  
     .....  
     57  
 disconnect database 42  
 disk block count 20  
 disk block identifier 20  
 disk compress threshold 17, 26  
 disk copy 7, 9, 18, 27, 31  
 disk copy dataset  
     recovery  
     .....  
     60, 62  
 disk dataset compression 17, 20, 22, 26  
 disk dataset recovery 21  
 disk retention period 36  
 disk space release 19  
 distribution tape 4  
 Double Byte Character Set 5  
 drive retention 9  
 DTIMOUT 21  
 duplex 13, 17, 66, 69  
 duplex backup 13  
 duplicate object 10  
 dynamic allocation 9, 23  
 dynamic backout 8, 41  
 Dynamic load balancing 14, 18  
 E  
 error information 3  
 esoteric 14  
 expiry date for tape dataset 15  
 F  
 FCT 15

file  
     RDO  
     .....  
     15  
 File Control Table 12, 23, 38  
 formatted index entry 3, 14, 19, 22  
 G  
 general purpose component messages  
     137  
 H  
 HLQ 7, 13, 29  
 I  
 ICRC 14, 18  
 IEBGENER 16  
 Improved cartridge recording capability  
     18  
 index entry report 10  
 index print utility 6  
 input processing 10, 21  
 input-output processing 10, 21  
 installation process 5  
 installation verification procedure 1  
 ISPF LIBDEF 9  
 IVP 1, 32  
     batch archival  
     .....  
     9  
     batch retrieval  
     .....  
     10  
 CICS definitions  
     .....  
     7  
 database initialization  
     .....  
     3  
 datasets  
     .....  
     2  
 online retrieval  
     .....  
     11  
 running batch

.....  
 9  
 J  
 JCL requirements 3  
 JOBLIB 12, 23  
 journal  
   archiving  
     .....  
     10, 45  
   dataset  
     .....  
     11, 12, 27, 38, 48  
   maintenance  
     .....  
     10, 32  
   maintenance utility  
     .....  
     21, 40  
   maintenance utility messages  
     .....  
     60  
 journal archive 29, 32  
   database  
     .....  
     27, 40, 50  
 journal record 49  
 L  
 link pack area 7  
 logical record 7, 33  
 logon procedure 8  
 loss of an entire tape database 16  
 loss of database 17  
 loss of individual tape volume(s) 16  
 LSRpool 21  
 M  
 main index entry 7, 10, 25  
 main temporary storage 28  
 MAXBLKCT 16  
 MAXDISK 12, 14, 16, 20, 25, 28  
 MAXDRIVE 10, 12, 14, 16, 19, 24, 27  
 maximum blockcount 13  
 MAXQLEN 11, 12, 14, 16, 17, 20, 24,  
   27  
 media failure 16

messages and codes 1  
 MIE 7, 10  
 migration 36, 47  
 migration  
   control dataset  
     .....  
     12, 48  
   of objects  
     .....  
     29  
 MLPA 7  
 MRO 31  
 Multiple database access 15  
 multiple NearArchive databases 12  
 Multiple Region Operation 10, 13, 31  
 N  
 NearArchive calls - sample 1  
 NearArchive messages 3  
 NEXTTSQ 8  
 NODUPS 43  
 O  
 object 7, 33  
   expiration  
     .....  
     29  
   identification  
     .....  
     5  
   migration  
     .....  
     8  
   object deletion 37  
   object length 30  
   object management utility 20, 29, 39  
   object management utility messages  
     49, 69  
   object residency indicator 20  
   object retrieval 3  
   object, newest 29  
   object, oldest 29  
   online parameter file messages 108  
   online request processor messages 102,  
     103  
   online retrieval 4, 9, 15, 24

online retrieval  
     interface  
     .....  
         12  
 operational considerations 9, 23  
 operator interface 15, 17  
     user exit  
     .....  
         30  
 OT22 21  
 OT23 11, 13  
 OT24 13  
 OT26 15  
 OT26UXIT 30  
 OTASBKUP 64  
 OTASLOG1 9, 11  
 OTASP010 3, 6  
 OTASP020 3, 12  
 OTASP030 3, 16, 17, 20, 39  
 OTASP040 4, 20, 29, 39  
 OTASP050 8  
 OTASP055 7  
 OTASP070 4, 10, 21, 29, 40  
 OTASP080 4, 21, 52  
 OTASP100 3, 4, 11  
 OTASP110 4, 21, 58  
 OTASP130 5, 21, 50, 70  
 OTASP200 4, 8  
 OTASP240 10  
 OTASS100 16, 24, 27  
 OTASSVC 6  
 OTASTEST 11  
 OTASUX01 3, 5, 27  
 OTASUX02 27  
 OTASV100 4, 10, 19  
 OTSN4000 5  
 OTSN4100 49  
 OTSN4101 49  
 OTSN4102 49  
 OTSN4103 49  
 OTSN4200 28  
 OTSN4210 30  
 OTSN4220 34  
 OTSN4221 36

OTSN4222 40  
 OTSN4230 45  
 OTSN4231 46  
 OTSN4300 5  
 OTSN4301 5, 12  
 OTSN4302 5, 21  
 OTSN4303 5, 22  
 OTSN4304 6, 22  
 OTSN4306 48  
 OTSN4400 51  
 OTSN4401 52  
 output processing 10  
 P  
 partial object retrieval 6, 12, 30  
 performance 16  
 PL/1 7  
 PLPA 7  
 PLTPI 13, 21  
 PLTSD 13, 21  
 primary index 5, 10, 12, 25, 30  
 primary index  
     file  
     .....  
         15  
     name  
     .....  
         7, 30  
 primary index file 6  
 primary index record size 59  
 primary key 3, 7, 21  
 primary keylength 7, 29  
 priority 29  
 program list table - post initialization  
     21  
 program list table - shut-down 21  
 Q  
 query index 14  
 query interface  
     compact  
     .....  
         19  
     extended  
     .....  
         14, 20

volume serial number  
 .....  
 14, 22  
 query sort sequence 15, 23  
 R  
 RACF 5  
 RDO 15  
 mapsets  
 .....  
 20  
 programs  
 .....  
 18  
 programs - 2nd NA region  
 .....  
 22  
 transactions  
 .....  
 20  
 transactions - 2nd NA region  
 .....  
 23  
 record count 6  
 record length 33  
 record length, maximum 26  
 record number 38  
 record prefix 6  
 recovery control 50, 56, 70  
 recovery control entries 4, 50  
 recovery status 16  
 recycle threshold 16, 25  
 RedWood 4, 7  
 regression  
 database  
 .....  
 1, 31  
 request code  
 10/11/12 - Open Database  
 .....  
 3  
 13 - Write Data Record  
 .....  
 6  
 14 - Identify Secondary Key

.....  
 8  
 15 - Close Database  
 .....  
 10  
 16 - Commit Updates  
 .....  
 12  
 21 - Ouery Index  
 .....  
 14  
 22 - Retrieve Object  
 .....  
 19  
 23 - Delete Object  
 .....  
 21  
 30 - Open Database  
 .....  
 24  
 31 - Query Index  
 .....  
 26  
 32 - Read data record  
 .....  
 30  
 33 - Delete Object  
 .....  
 34  
 34 - Close database  
 .....  
 35  
 40 - connect database  
 .....  
 8  
 41 - Query Index  
 .....  
 14  
 42 - Retrieve object record  
 .....  
 23  
 43 - Retrieve Object  
 .....  
 28

45 – Write Object Record	34
31	keylength
46 - Identify Secondary Key	37
34	maintenance
47 – Delete Object	34
37	number
48 – Commit Updates	37
39	update
49 – Disconnect Database	36
42	update utility
request codes 4, 7	4
request queueing 19, 25	secondary index update utility 21
Resource Definition Online 15	secondary key 34, 38, 39
response times 14	secondary key identification 35
restart 8	secondary region 13, 22, 27
restart connect 10	security 27
restart processing 9, 11, 41	sequence number 16, 54
retention period for tape dataset 15	sequential read 24
retrieval priority 13	sequential retrieval 22, 27
retrieve object record 21, 23	SIE 10
S	SMF 8, 19, 24, 30, 138
secondary index 5, 11, 12, 28	SMF
secondary index	batch retrieval
add new definition	33
36	batch retrieval logging
asynchronous update	11
4, 11	CICS retrieval logging
dataset sizing	11
44	SMF database section 141
deletion	SMF header section 139
43	SMF record
entry	format
6	138
identifier	SMF record descriptor section 140

SMF record identifier 10, 24, 30, 33  
 SMF request section 142  
 SMF timestamp 25, 31  
 SMP/E 4  
 SMPEACC 6  
 SMPEAPP 6  
 SMPPEDEF 5  
 SMPERCV 6  
 sort 35  
 space management  
     dataset  
         .....  
         11, 12, 15  
 stalled request 10  
 start position 37  
 start record number 6  
 status detail 23  
 STEPLIB 12, 23  
 5.storage level 4, 7, 12, 13, 14, 17, 53  
     add new level  
         .....  
         47  
     definition  
         .....  
         12  
     identifier  
         .....  
         13  
     maintenance  
         .....  
         28, 45  
     update definition  
         .....  
         46  
 supplementary index entries 25  
 supplementary index entry 10  
 SVC 7, 29, 30  
 SYSALLDA 8

system dataset sizing 25  
 system datasets 19  
 system file identification 22  
 System Management Facility 8  
 system security 5  
 system unit name 8, 31  
 T  
 tape block count 20  
 tape block identifier 20  
 tape capacity 60  
 tape dataset 12  
 tape dataset recovery 16  
 tape disposition 42  
 tape retrieval I/O driver messages 133  
 tape scan utility 12  
 tape sequence number 20  
 TAPEWAIT 15, 19, 25, 28  
 temporary storage queue 3, 7, 28  
 temporary storage queue name 29  
 temporary storage queue stringing 7,  
     31  
 TIMEOUT 10, 13, 15, 26, 29  
 TSO/ISPF 3, 10, 11  
 TSQ pointer record 8  
 TSQDELETE 4, 13, 15, 18, 25, 28, 30  
 U  
 uncommitted objects 10  
 unit name 13  
 user exit 5, 27  
 user exit parameter area 29  
 user SVC 6  
 utilities 19  
 V  
 Volsafe 4, 7, 15  
 volume recovery 52, 70  
 volume unavailable 51, 55, 70  
 W  
 write object record 31