



A faint BPMN diagram is visible in the background. It features several circular nodes, some with internal symbols like a lightning bolt or a circle with a dot. These nodes are connected by arrows, indicating a process flow. A thick horizontal bar is positioned across the middle of the page, partially obscuring the diagram.

FUEGO

Configuring the Fuego Archive Database

FuegoBPM Enterprise Server V5.5

May 17, 2005

Table of Contents

Introduction.....	3
Assumptions.....	3
Procedure Overview.....	3
Step 1: Define the Archive Database Configuration.....	3
Step 2: Copy the JDBC Driver JAR File to \lib.....	5
Step 3: Generate SQL Script to Create Archive Database.....	5
Step 4: Create the Archive Database	6
Step 5: Enable the Archive Viewer Web Application	6
Step 6: Create Archive Viewer WAR File.....	7
Step 7: Deploy Archive Viewer in the J2EE App Server	8
Step 8: Configure Archive Viewer Login Configuration	9
Appendix A: SQL Script to Create Archive Database.....	10

Introduction

The Fuego Archive Database provides the means of retaining a history for expired process instances. Any notes and documents associated with the expired instances are also archived.

Assumptions

This procedure assumes the FuegoBPM Enterprise Server is already installed and running.

A symbolic variable *FUEGO_HOME* (e.g. *C:\fuego5.5\enterprise*) will be used in the instructions to represent your specific installation directory. (This variable does not actually exist but you may set it if you wish.)

Procedure Overview

Steps 1-4 below are performed in all cases. After Step 4 there is a fork in the road, depending on whether the Archive Viewer web application will be deployed to a J2EE application server or will run within Fuego's embedded Tomcat server.

Step #	Description	Role
1	Define the Archive configuration	Fuego administrator
2	Copy the JDBC driver JAR file to the Archive Viewer web app lib directory.	Fuego administrator
3	Generate SQL script to create Archive database	Fuego administrator
4	Create the FuegoBPM Archive database	Database administrator
5	Enable Archive Viewer web app (note 1)	Fuego administrator
6	Create Archive Viewer WAR file (note 2)	Fuego administrator
7	Deploy Archive Viewer in app. server (note 2)	J2EE administrator
8	Configure Archive Viewer Login	Fuego administrator

Note 1: Performed only if not deploying the Archive Viewer to a J2EE app. server.

Note 2: Performed if deploying the Archive Viewer to a J2EE app. server.

As evident in the above table, there may be as many as three administrative roles needed in this procedure but the bulk of the effort will be expended by the Fuego resource:

1. Fuego Administrator – to enable the Archiving function with the FuegoBPM Enterprise Server.
2. Database Administrator – to create the Archive database used store expired instances.
3. J2EE Administrator – optional, to deploy the Archive View web application on a J2EE application server.

Step 1: Define the Archive Database Configuration

The Fuego administrator will use the FuegoBPM Web Console to enable the Archiving option and to create an Archive database configuration to be used by the FuegoBPM Server.

1. Open the FuegoBPM Server Web Console (e.g. <http://localhost:8585/webconsole>) where archiving is to be enabled.
2. Open on the *Services* tab and check the *Enable archiving* option:

Servers > Edit Server DocServer

Basic configuration Log Execution **Services** Networking Others

Disposer

Disposer latency	2 Days
Instance caducity	15 Days
Disposer starting time	February 21, 2005 1:20 PM
Dispose Participants	<input checked="" type="checkbox"/> Enabled 30 Days
Archiving	<input type="checkbox"/> Enable archiving

3. When the option is checked another option, *Create a new configuration* will display. Select this option.
4. In the *Add Configuration* dialog enter a name for the configuration name for the Archive database (not the database name itself but something meaningful to appear in the Configurations list), e.g. "Process Archives".

Servers > Edit Server Development > Add Configuration

Type

Name	Process Archives
Type	SQL Database
Subtype	MsSQL JDBC (i-net Driver)

Next Cancel Reset

5. Set the *type* to "SQL Database" and the *subtype* to an installed JDBC driver (e.g. *i-net*) to implement the Archive database.
6. Click the *Next* button to move to the last step to define the Archive database. Define the database specific properties (e.g. *Host*, *Port*, *Database* name, login *User* and *Password*).
7. The *Runtime* variables can be left at the default values or adjusted per recommendations set by Fuego administrator.

Configurations > Edit Configuration Process Archives

Type

Name	Process Archives
Type	SQL Database
Subtype	MsSQL JDBC (i-net Driver) Change subtype

Properties

Host	localhost
Port	1433
Database	FuegoProcess Archive
User	fuegoEngine Change Password

Runtime

Maximum Pool size	35
Connection Idle time (Mins)	15
Maximum opened cursors	3500

Save Cancel Reset

8. *Save* the configuration. The *Edit Configuration* dialog will close and return to the main page.



Remember the User Id, password and database names provided in this step as these properties will need in Step 8.

Step 2: Copy the JDBC Driver JAR File to \lib

The Archive Viewer web application must have access to the Archive database. To do this the relevant JDBC driver classes (contained in a JAR file) must be copied to:

`%FUEGO_HOME%\webapps\archivingviewer\lib`

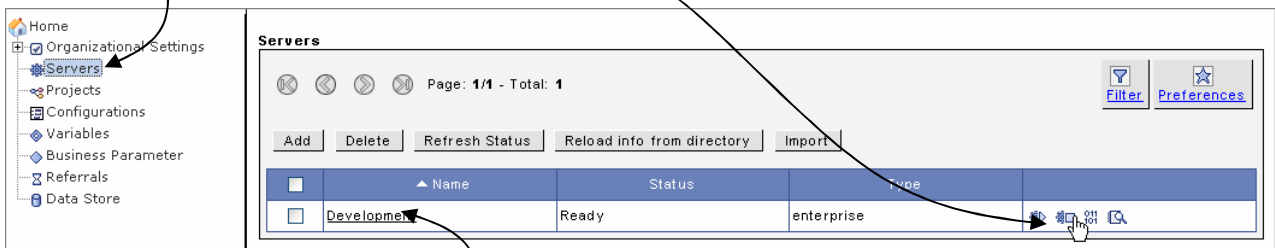
Step 3: Generate SQL Script to Create Archive Database

The Fuego administrator will create the SQL script for the Archive database whose configuration was defined in the previous step.



In this task it will be necessary to stop the FuegoBPM Server which may be an issue if there are users connected to it. (If you are using a J2EE FuegoBPM Server then this is not an issue.)

1. Open the FuegoBPM Server Web Console (e.g. <http://localhost:8585/webconsole>).
2. Select *Servers* from the menu tree view and *stop* the server (e.g. “Development”) where the archive database is to be created.



3. After the server is stopped, select the *server* to view its properties.
4. Select the drop/create database options as required for your installation. Check the *Create archiving data structure* option.

Archiving database creation	
Drop archiving database	<input type="checkbox"/>
Create archiving database	<input checked="" type="checkbox"/>
Create archiving data structure	<input checked="" type="checkbox"/>
User Name	<input type="text" value="sa"/>
User Password	<input type="password" value="*****"/>
<input type="button" value="OK"/> <input type="button" value="Cancel"/> <input type="button" value="Show SQL statements"/>	

5. Enter a User name and password (if you know it) for a user able to grant the following permissions, using SQL Server as an example:

```
CREATE DATABASE FuegoProcessArchives;
EXEC sp_addlogin 'fuegoEngine', 'password', 'FuegoProcessArchives';
USE FuegoProcessArchives;
EXEC sp_grantdbaccess 'fuegoEngine';
GRANT CREATE TABLE TO fuegoEngine;
GRANT CREATE VIEW TO fuegoEngine;
GRANT CREATE PROCEDURE TO fuegoEngine;
```

These values were defined in the Archive database configuration in the previous step.

6. Select the *Show SQL Statements* button to create the SQL script to be sent to the DBA. If the Fuego administrator has the necessary DBA permissions, click on the *Next* button to create the database via the *Manage Database* wizard.
7. If the DBA is creating the Archive database, then the Fuego administrator should coordinate with the DBA to ensure it is created. Once created, the Fuego Admin can proceed to the next step.

Step 4: Create the Archive Database

In the previous step the Fuego administrator created the SQL script for the Archive database. In this step the Database administrator will modify the SQL script to conform to database standards and will execute the script to create the Archive database and tables.

See *Appendix A* for a listing of the Archive database SQL script.

Step 5: Enable the Archive Viewer Web Application

The Fuego administrator should skip this step if the Archive Viewer is to be deployed to a J2EE app. server. It assumes the Archive Viewer will be configured to run in Fuego's embedded Tomcat web server.

1. Launch the FuegoBPM Enterprise Administration Center.

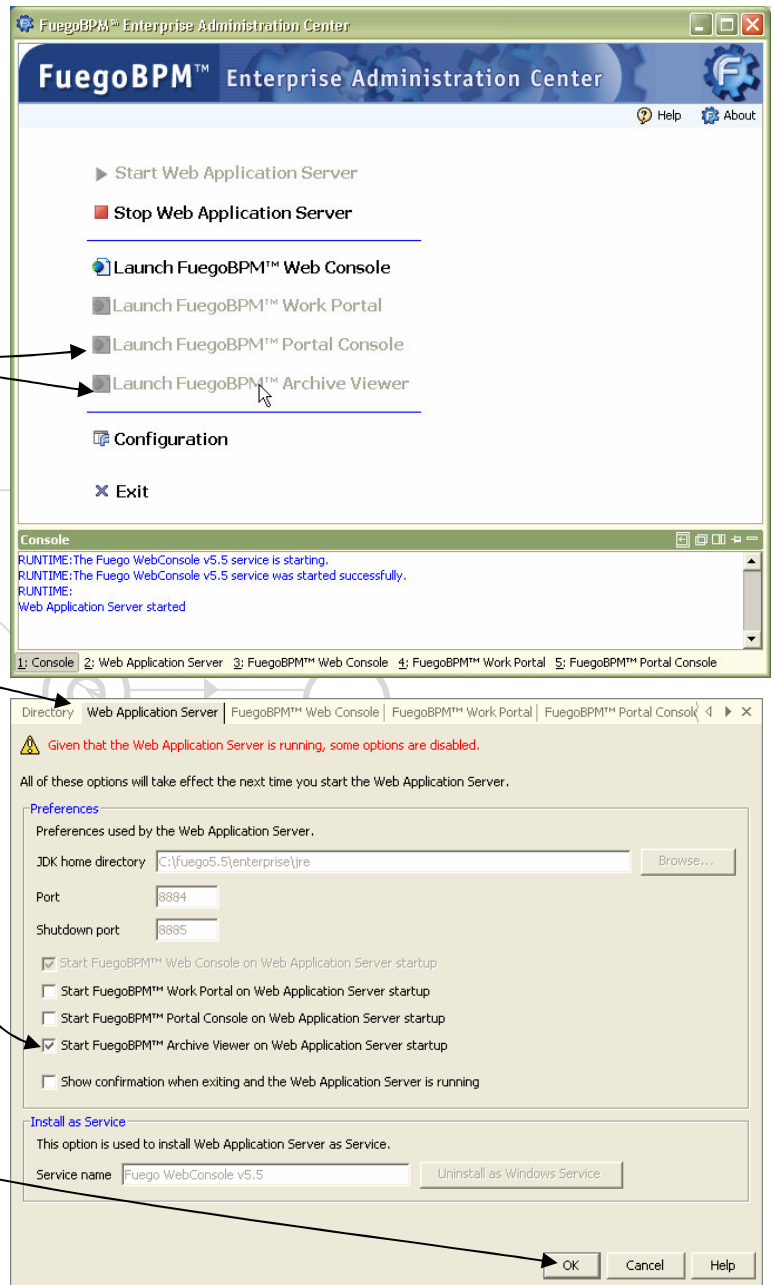
- If the *Launch FuegoBPM Archive Viewer* option is enabled then you can skip this step. Otherwise it will be necessary to complete the remaining tasks in this step.

- Select the *Configuration* option to open the Configuration dialog.

- In the *Configuration* dialog, select the *Web Application Server* tab and check the *Start FuegoBPM Archive Viewer on Web Application Server*.

- Select the *OK* button to save your changes and close the *Configuration* dialog. The Archive Viewer launch option should now be enabled.

- Skip Steps 6 & 7 and go directly to Step 8.



Step 6: Create Archive Viewer WAR File

The Fuego administrator will perform this step only if the FuegoBPM Archive Viewer is to be deployed to a J2EE application server.

- The Fuego admin. should log into the server where the FuegoBPM Server has been installed.
- Open a *Command Prompt* window (*Start...Run...cmd*).

- Set the `FUEGO_HOME` variable (substitute your installation directory on the right side):

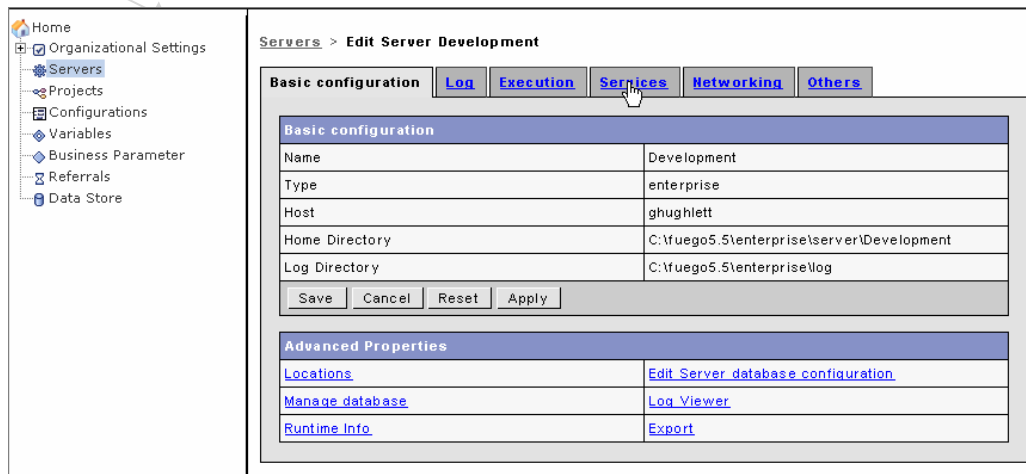
```
set FUEGO_HOME=C:\fuego5.5\enterprise
```

- Change to the archiving web app directory by issuing the following DOS command:

```
cd %FUEGO_HOME%\webapps\archivingviewer
```

- Create the WAR file by using the JDK `jar` command:

```
jar -cvf Archiveviewer.war *
```



For the `jar` command above to work, the `JAVA_HOME\bin` directory must be on the `PATH` environment variable of the FuegoBPM Server. If the JDK is not installed on the FuegoBPM Server host then the `jar` executable can be copied from a JDK to `FUEGO_HOME\bin` as a quick work around.

- After the `jar` command completes send the `ArchivingViewer.war` file to the app. server administrator.



The `ArchivingViewer.war` file is a compressed file in excess of 12MB.

Step 7: Deploy Archive Viewer in the J2EE App Server

In this step the J2EE will deploy the Archive Viewer WAR file created in the previous step. The WAR file must be deployed to an application server that supports Java version 1.4.2 or higher.

When this step is completed, the J2EE admin. should supply the Fuego admin. with the URL of the Archive Viewer application deployed on the application server, e.g. <http://localhost:7001/archivingviewer>.


Step 8: Configure Archive Viewer Login Configuration

This final step will test either of the deployment options for the Archive Viewer: in Fuego's embedded Tomcat server or deployed to a J2EE application server.

1. Open a browser to the started Archive Viewer web application, e.g. <http://localhost:8585/archivingviewer>.

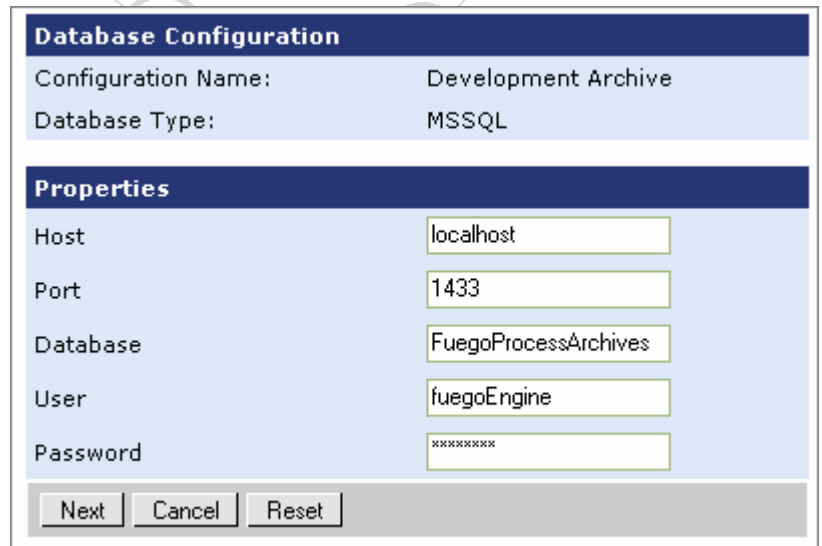
When run for the first time, the following dialog will appear.

2. Select a *Database Type* corresponding to the database configured in Step 2 and supply a name that is meaningful to the ultimate archive viewing end users.



3. Click on the *Next* button to provide additional database details about the Archive database configuration defined in Step 1.

4. Supply the Archive database configuration details defined earlier in Step 1 again here.



5. Click the *Next* button to create the Login configuration.

6. The *Archive Viewer Database Login* dialog is the login dialog archive viewer end-users will see each time.



Appendix A: SQL Script to Create Archive Database

The Data Definition Language below is specific to FuegoBPM Enterprise Server V5.5, SP5.

```
-- You need to execute these sentences connected as dba user.
CREATE DATABASE FuegoProcessArchives;
EXEC sp_addlogin 'some_uid', 'ur_password', 'database_name';
USE database_name;
EXEC sp_grantdbaccess 'some_uid';
GRANT CREATE TABLE TO some_uid;
GRANT CREATE VIEW TO some_uid;
GRANT CREATE PROCEDURE TO some_uid;

-- You need to execute these sentences connected as the user created before (this user will be the
one used by Fuego to connect to this database).
CREATE TABLE INSTANCE
(
    CREATIONTIME DATETIME NOT NULL,
    FINALIZATIONTIME DATETIME NOT NULL,
    ORGANIZATIONIN DECIMAL (8, 0) NOT NULL,
    INSTANCEIN DECIMAL (8, 0) NOT NULL,
    PROCESSIN DECIMAL (8, 0) NOT NULL,
    PROCESSDEADLINE DATETIME NULL,
    PARENTTHREAD DECIMAL (5, 0) NOT NULL,
    DESCRIPTION VARCHAR ( 255) NULL,
    STATE INTEGER NOT NULL,
    THREADIN DECIMAL (5, 0) NOT NULL,
    PRIORITY INTEGER NOT NULL,
    AUTHOR DECIMAL (8, 0) NOT NULL,
    PRIMARY KEY (ORGANIZATIONIN,PROCESSIN,INSTANCEIN,THREADIN)
);
CREATE TABLE ATTACHMENT
(
    CREATIONTIME DATETIME NULL,
    ORGANIZATIONIN DECIMAL (8, 0) NOT NULL,
    PROCESSIN DECIMAL (8, 0) NOT NULL,
    REMARKS VARCHAR ( 255) NULL,
    CONTENT IMAGE NULL,
    VERSION DECIMAL (3, 0) NOT NULL,
    LASTVERSION INTEGER NOT NULL,
    INSTIN DECIMAL (8, 0) NOT NULL,
    CREATOR DECIMAL (8, 0) NOT NULL,
    DESCRIPTION VARCHAR ( 255) NULL,
    CONTENTSIZE DECIMAL (8, 0) NULL,
    REMOTEREERENCE VARCHAR ( 255) NULL,
    NAME VARCHAR ( 255) NULL,
    ATTACHMENTIN DECIMAL (8, 0) NOT NULL,
    CANBEREFERENCED INTEGER NOT NULL,
    OSINFO VARCHAR ( 60) NULL,
    PRIMARY KEY (ORGANIZATIONIN,PROCESSIN,INSTIN,ATTACHMENTIN,VERSION)
);
CREATE TABLE REMARK
(
    CREATIONTIME DATETIME NOT NULL,
    INSTIN DECIMAL (8, 0) NOT NULL,
    REMIN DECIMAL (4, 0) NOT NULL,
    ORGANIZATIONIN DECIMAL (8, 0) NOT NULL,
    PROCESSIN DECIMAL (8, 0) NOT NULL,
    ACTIVITYNAME VARCHAR ( 255) NULL,
    REMARK IMAGE NULL,
    PARTICIPANT DECIMAL (8, 0) NOT NULL,
    PRIMARY KEY (ORGANIZATIONIN,PROCESSIN,INSTIN,CREATIONTIME,REMIN)
);
CREATE TABLE VARIABLE
(
    ORGANIZATIONIN DECIMAL (8, 0) NOT NULL,
    VARIABLETYPE VARCHAR ( 255) NULL,
```

```

ID VARCHAR ( 255) NOT NULL,
PRIMARY KEY (ORGANIZATIONIN, ID)
);
CREATE TABLE BOOLEANVARIABLE
(
    INSTIN DECIMAL (8, 0) NOT NULL,
    VALUE INTEGER NOT NULL,
    ORGANIZATIONIN DECIMAL (8, 0) NOT NULL,
    PROCESSIN DECIMAL (8, 0) NOT NULL,
    THREADIN DECIMAL (5, 0) NOT NULL,
    ID VARCHAR ( 255) NOT NULL,
    PRIMARY KEY (ORGANIZATIONIN, PROCESSIN, INSTIN, THREADIN, ID)
);
CREATE INDEX BOOLEANID ON BOOLEANVARIABLE ( ORGANIZATIONIN, ID, VALUE );
;
CREATE TABLE DECIMALVARIABLE
(
    INSTIN DECIMAL (8, 0) NOT NULL,
    VALUE DECIMAL (25,6) NOT NULL,
    ORGANIZATIONIN DECIMAL (8, 0) NOT NULL,
    PROCESSIN DECIMAL (8, 0) NOT NULL,
    THREADIN DECIMAL (5, 0) NOT NULL,
    ID VARCHAR ( 255) NOT NULL,
    PRIMARY KEY (ORGANIZATIONIN, PROCESSIN, INSTIN, THREADIN, ID)
);
CREATE INDEX DECIMALID ON DECIMALVARIABLE ( ORGANIZATIONIN, ID, VALUE );
;
CREATE TABLE INTEGERVARIABLE
(
    INSTIN DECIMAL (8, 0) NOT NULL,
    VALUE INTEGER NOT NULL,
    ORGANIZATIONIN DECIMAL (8, 0) NOT NULL,
    PROCESSIN DECIMAL (8, 0) NOT NULL,
    THREADIN DECIMAL (5, 0) NOT NULL,
    ID VARCHAR ( 255) NOT NULL,
    PRIMARY KEY (ORGANIZATIONIN, PROCESSIN, INSTIN, THREADIN, ID)
);
CREATE INDEX INTEGERID ON INTEGERVARIABLE ( ORGANIZATIONIN, ID, VALUE );
CREATE TABLE REALVARIABLE
(
    INSTIN DECIMAL (8, 0) NOT NULL,
    VALUE REAL NOT NULL,
    ORGANIZATIONIN DECIMAL (8, 0) NOT NULL,
    PROCESSIN DECIMAL (8, 0) NOT NULL,
    THREADIN DECIMAL (5, 0) NOT NULL,
    ID VARCHAR ( 255) NOT NULL,
    PRIMARY KEY (ORGANIZATIONIN, PROCESSIN, INSTIN, THREADIN, ID)
);
CREATE INDEX REALID ON REALVARIABLE ( ORGANIZATIONIN, ID, VALUE );
CREATE TABLE STRINGVARIABLE
(
    INSTIN DECIMAL (8, 0) NOT NULL,
    VALUE VARCHAR (255) NOT NULL,
    ORGANIZATIONIN DECIMAL (8, 0) NOT NULL,
    PROCESSIN DECIMAL (8, 0) NOT NULL,
    THREADIN DECIMAL (5, 0) NOT NULL,
    ID VARCHAR ( 255) NOT NULL,
    PRIMARY KEY (ORGANIZATIONIN, PROCESSIN, INSTIN, THREADIN, ID)
);
CREATE INDEX STRINGID ON STRINGVARIABLE ( ORGANIZATIONIN, ID, VALUE );
CREATE TABLE TIMEVARIABLE
(
    INSTIN DECIMAL (8, 0) NOT NULL,
    VALUE DATETIME NOT NULL,
    ORGANIZATIONIN DECIMAL (8, 0) NOT NULL,
    PROCESSIN DECIMAL (8, 0) NOT NULL,
    THREADIN DECIMAL (5, 0) NOT NULL,
    ID VARCHAR ( 255) NOT NULL,
    PRIMARY KEY (ORGANIZATIONIN, PROCESSIN, INSTIN, THREADIN, ID)
);
CREATE INDEX TIMEID ON TIMEVARIABLE ( ORGANIZATIONIN, ID, VALUE );
CREATE TABLE PROCESS

```

```

(
    PROCESSID VARCHAR (255) NOT NULL,
    ORGANIZATIONIN DECIMAL (8, 0) NOT NULL,
    PROCESSIN DECIMAL (8, 0) NOT NULL,
    PUBLISHINGTIME DATETIME NULL,
    PROCESSXPDL IMAGE NOT NULL,
    PRIMARY KEY (ORGANIZATIONIN,PROCESSID,PROCESSIN)
);
CREATE INDEX IDENTNUMBER ON PROCESS ( ORGANIZATIONIN,PROCESSIN );
CREATE TABLE ACTIVITY
(
    ORGANIZATIONIN DECIMAL (8, 0) NOT NULL,
    PROCESSIN DECIMAL (8, 0) NOT NULL,
    ACTIVITYNAME VARCHAR ( 255) NOT NULL,
    ACTIVITYTYPE VARCHAR (15) NOT NULL,
    PRIMARY KEY (ORGANIZATIONIN,PROCESSIN,ACTIVITYNAME)
);
CREATE TABLE PARTICIPANT
(
    ORGANIZATIONIN DECIMAL (8, 0) NOT NULL,
    PARTICIPANTUID VARCHAR (255) NOT NULL,
    NAME VARCHAR (255) NOT NULL,
    PARTICIPANTIN DECIMAL (8, 0) NOT NULL,
    PRIMARY KEY (ORGANIZATIONIN,PARTICIPANTIN)
);
CREATE TABLE PROCESSIMAGE
(
    PROCESSID VARCHAR (255) NOT NULL,
    ORGANIZATIONIN DECIMAL (8, 0) NOT NULL,
    PROCESSIN DECIMAL (8, 0) NOT NULL,
    IMAGE IMAGE NULL,
    PRIMARY KEY (ORGANIZATIONIN,PROCESSID,PROCESSIN)
);
CREATE TABLE ORGANIZATION
(
    ORGANIZATIONIN DECIMAL (8, 0) NOT NULL,
    ID VARCHAR (255) NOT NULL,
    PRIMARY KEY (ID)
);
CREATE TABLE EVENT
(
    EVENTDATA VARCHAR ( 255) NULL,
    EVENTIN DECIMAL (8, 0) NOT NULL,
    INSTIN DECIMAL (8, 0) NOT NULL,
    ORGANIZATIONIN DECIMAL (8, 0) NOT NULL,
    PROCESSIN DECIMAL (8, 0) NOT NULL,
    ACTIVITYNAME VARCHAR ( 255) NULL,
    EVENTTIME DATETIME NOT NULL,
    THREADIN DECIMAL (5, 0) NOT NULL,
    EVENTTYPE INTEGER NOT NULL,
    PARTICIPANT DECIMAL (8, 0) NOT NULL,
    PRIMARY KEY (ORGANIZATIONIN,PROCESSIN,INSTIN,THREADIN,EVENTTIME,EVENTIN)
);

```

