Rdb Journal
technical corner

New EXCEPT, INTERSECT and MINUS Operators

A feature of Oracle Rdb

By Ian Smith
Oracle Rdb Relational Technology Group
Oracle Corporation

The examples in this article use SQL language from Oracle Rdb V7.1 and later versions.

New EXCEPT, INTERSECT and MINUS Operators

This release of Oracle Rdb adds three new operators the select expression syntax. The new operators EXCEPT, INTERSECT and MINUS are all forms of select table merge operations, and therefore have the same format as the UNION and UNION ALL operators.

EXCEPT (or EXCEPT DISTINCT)

The EXCEPT DISTINCT operator is used to create a result table from the first select expression except for those row values that also occur in the second select expression.

DISTINCT is an optional keyword, but EXCEPT and EXCEPT DISTINCT are identical operations. EXCEPT conforms to the ANSI and ISO SQL:1999 Database Language Standard.

The EXCEPT operator is not commutative. That is, A EXCEPT B may result in a different set of rows from B EXCEPT A. This is demonstrated by the examples below.

INTERSECT (or INTERSECT DISTINCT)

The INTERSECT DISTINCT operator is used to create a result table from the first select expression those row values that also occur in the second select expression.

DISTINCT is an optional keyword but INTERSECT and INTERSECT DISTINCT are identical operations. INTERSECT conforms to the ANSI and ISO SQL:1999 Database Language Standard.

In general the INTERSECT operator is commutative. That is, A INTERSECT B results in the same set of rows from B INTERSECT A. This is demonstrated by the examples below. However, care should be taken when using LIMIT TO within the different branches of the INTERSECT as this may make the result non deterministic because of different solution strategies employed by the Rdb optimizer.

MINUS

The MINUS operator is a synonym for the EXCEPT DISTINCT operator and is provided for language compatibility with the Oracle RDBMS SQL language.

UNION (or UNION DISTINCT)

Please refer to the existing Rdb documentation for detailed information on the UNION operator. Rdb 7.1 adds the optional keyword DISTINCT. UNION and UNION DISTINCT are identical operations. This is part of the ANSI and ISO SQL:1999 Database Language Standard.

CORRESPONDING

The UNION, EXCEPT, MINUS, and INTERSECT operators can be followed by the keyword CORRESPONDING. This causes the two select lists to be compared by name. Only column names that appear in both lists are retained for the resulting query table.

The name is either the column name, or the name provided by the AS clause. If there are no names in common, or a column name appears more than once in a select list then an error is reported.

Usage Notes

(a) The EXCEPT DISTINCT operator can be replaced by the NOT EXISTS predicate. The Rdb server currently implements EXCEPT DISTINCT by rewriting the query as described here. Consider this example:

```
SQL> select manager_id from departments
cont> except distinct
cont> select employee_id from employees;
```

This query could be rewritten as:

```
SQL> select manager_id
cont> from departments d
cont> where not exists (select *
cont>                   from employees e
cont>                   where e.employee_id = d.manager_id
cont>                      or (e.employee_id is null
cont>                            and d.manager_id is null));
```

This example shows that the EXCEPT format is easier to read. As the number of columns selected increases so does the complexity of the NOT EXISTS subquery.

(b) The INTERSECT DISTINCT operator can be replaced by the EXISTS predicate. The Rdb server currently implements INTERSECT DISTINCT by rewriting the query as described here. Consider this example that displays all managers that are also employees:

```
SQL> select manager_id from departments
cont> intersect distinct
cont> select employee_id from employees;
```

This query could be rewritten as:

```
SQL> select manager_id
cont> from departments d
cont> where exists (select *
cont>               from employees e
cont>               where e.employee_id = d.manager_id
cont>                    or (e.employee_id is null
cont>                        and d.manager_id is null));
```

This example shows that the INTERSECT format is easier to read. As the number of columns selected increases so does the complexity of the EXISTS subquery.

(c) For both EXCEPT and INTERSECT all duplicate rows are eliminated. For the purposes of these operators a row is considered a duplicate if each value in the first select list is equal to the matching column in the second select list, or if both these columns are NULL.

The duplicate matching semantics can be clearly seen in the rewritten queries which use NOT EXISTS and EXISTS.

Examples

The following section shows examples of these new operators.

Example 1: Using CORRESPONDING as a shorthand for the select list

This example derives results from tables with some column names in common. Here the RETIRED_EMPLOYEES table contains a subset of the columns from EMPLOYEES (EMPLOYEE_ID and LAST_NAME) as well as some new columns to describe the retired employee (e.g. RETIRE_DATE). CORRESPONDING is used to match the common columns names and produce this report.

```
SQL> select *, 'retired' as status from RETIRED_EMPLOYEES
cont> union corresponding
cont> select *, 'working' as status from EMPLOYEES
cont> order by status;
 EMPLOYEE_ID   LAST_NAME STATUS
 00207         Babbin         retired
 00173         Bartlett       retired
 ...etc...
```

Example 2: Changing a result name by applying the AS clause

This example show the use of the AS clause to provide the same name to the AVG statistical expression in each part of the UNION clause. CORRESPONDING will align these two renamed columns. Without the AS clause these column expressions would be eliminated from the UNION result table.

```
SQL> select pnum,
cont>     avg(weight) as AVG edit using 'ZZZZ99.99'
cont> from p
cont> group by pnum
cont> union corresponding
cont> select pnum,
cont>      avg(qty) as AVG
cont> from spj
cont> group by pnum;
 PNUM             AVG
 P1             12.00
 P1            333.33
 P2             17.00
 P2            150.00
 P3             17.00
 P3            388.89
 P4             14.00
 P4            650.00
 P5             12.00
 P5            450.00
 P6             19.00
 P6            325.00
12 rows selected
SQL>
```

Example 3: EXCEPT DISTINCT operator

Here we use UNION DISTINCT to derive the full set of EMPLOYEE_ID values. Since all managers are also employees this list should return the same as a query on EMPLOYEES. It is used here so show the differences between these similarly structured operators.

```
SQL> select manager_id from departments
cont> union distinct
cont> select employee_id from employees;
 MANAGER_ID
 00164
 00165
 00166
.
.
.
 00435
 00471
100 rows selected
```

Make sure that all managers are also employees. List all managers who are not employees. The results show that there are no managers in this list.

```
SQL> select manager_id from departments
cont> except distinct
cont> select employee_id from employees;
0 rows selected
```

List all employees who are not managers. Or, stated in a different way list all employees, except those that are managers.

Simply reversing the order of the select expressions from the previous query will perform this query. Note that we get quite a different result.

```
SQL> select employee_id from employees
cont> except distinct
cont> select manager_id from departments;
 EMPLOYEE_ID
 00165
 00167
 00169
.
.
.
 00416
 00435
74 rows selected
```

Example 4: INTERSECT DISTINCT operator

Show the managers who are also employees.

```
SQL> select manager_id from departments
cont> intersect distinct
cont> select employee_id from employees;
 MANAGER_ID
 00164
 00166
 00168
.
.
.
 00418
 00471
26 rows selected
```

INTERSECT DISTINCT is commutative, so reversing the select expressions will yield in the same result set. However, the same ordering of these rows is not guaranteed unless an ORDER BY clause is applied to the result.

```
SQL> select employee_id from employees
cont> intersect distinct
cont> select manager_id from departments;
 EMPLOYEE_ID
 00164
 00166
 00168
.
.
.
 00418
 00471
26 rows selected
```

Oracle Rdb
New EXCEPT, INTERSECT and MINUS Operators
May 2002

Oracle Corporation
World Headquarters
500 Oracle Parkway
Redwood Shores, CA 94065
U.S.A.

Worldwide Inquiries:
Phone: +1.650.506.7000
Fax: +1.650.506.7200
www.oracle.com

Oracle Corporation provides the software that powers the internet.