]>

DBMS_SQL

Description

1. Since DB2 does not support invocation context persistence between functions and procedures, the following routines are implemented as procedures instead of functions: OPEN_CURSOR, IS_OPEN, EXECUTE FETCH_ROWS, EXECUTE_AND_FETCH, and LAST_ROW_COUNT

2. Stored Procedure with DBMS_SQL.VARCHAR2_TABLE,DBMS_SQL.NUMBER_TABLE as output parameter.

Example:

```
FUNCTION split(list IN VARCHAR2, delimiter IN VARCHAR2) RETURN dbms_sql.varchar2_table AS...
```

3. Since DB2 does not allow overloading by parameter type, DBMS_SQL.COLUMN_VALUE DBMS_SQL.BIND_VARIABLE DBMS_SQL.DEFINE_COLUMN are not supported.

4. DBMS_SQL.Parse (nid, cmd, dbms_sql.native) statement does not automatically execute DDL statements like "truncate table".
In Oracle, the DBMS_SQL.Parse statement below is enough to parse and execute a DDL statement in the same time (there is no need to have the DBMS_SQL.Execute (nid)). In DB2 we need to explicitly specify the DBMS_SQL.Execute (nid) statement in order the DDL request to be executed.
In the following example, we do not get an error, but after executing the statement, the table did not get truncated.

```
nid := DBMS_SQL.OPEN_CURSOR;
cmd := 'TRUNCATE TABLE tab_name';
DBMS_SQL.Parse ( nid, cmd, dbms_sql.native);
DBMS_SQL.CLOSE_CURSOR(nid);
```

Solution

1. Rewrite such function calls to be procedure calls instead.

2. Create your own type equivalent.

TYPE VARCHAR2_TABLE IS TABLE OF VARCHAR2(2000) INDEX BY BINARY_INTEGER;
TYPE NUMBER_TABLE IS TABLE OF NUMBER INDEX BY BINARY_INTEGER;

3. Using DB2 equivalen for each datatype, for example DBMS_SQL.BIND_VARIABLE_VARCHAR.

Can be converted automatically: NO