]>

Enabling/Disabling triggers in PL/SQL code

Description

Oracle suppports enabling/disabling triggers. DB2 can only CREATE or DROP them. Example:

```
EXECUTE IMMEDIATE 'ALTER TRIGGER TRG_LOAD_UPDATE disable';
EXECUTE IMMEDIATE 'DROP TRIGGER TRG_LOAD_UPDATE';
EXECUTE IMMEDIATE 'CREATE OR REPLACE TRIGGER TRG_LOAD_UPDATE BEFORE UPDATE ON TBL_LOAD FOR EACH ROW BEGIN ... '
```

Solution

Use a session variable that would control the enable/disable action of the trigger. Example:

```
IF (<check if global variable allows trigger to run>) THEN
   <perform the trigger action>
END IF;
```

This workaround allows control of the trigger on the session level, comparing to Oracle's global level. In many cases the session level is more advantageous.

Can be converted automatically: NO

Log-on or connection triggers

Description

DB2 does not support log-on or connection triggers.

Solution

DB2 9.7.3 provides a hook in the database configuration to execute a predefined procedure on connection. The procedure itself must not modify the database, but nothing prevents it from in turn invoking an autonomous procedure.

Here is an example:

```
SET SCHEMA ORAENV;
SET PATH = CURRENT PATH, ORAENV;

CREATE TABLE SECURITY.AUDIT(login VARCHAR(30), event VARCHAR(30), tstamp TIMESTAMP);

--#SET TERMINATOR @
CREATE OR REPLACE PROCEDURE CONNECT_PROC_MOD()
AUTONOMOUS
BEGIN
  INSERT INTO SECURITY.AUDIT VALUES(USER, 'CONNECT', CURRENT TIMESTAMP);
END
@

CREATE OR REPLACE PROCEDURE CONNECT_PROC()
BEGIN
  SET PATH = ORAENV, CURRENT PATH;
  CALL CONNECT_PROC_MOD();
END
@
--#SET TERMINATOR ;

UPDATE DB CFG USING CONNECT_PROC ORAENV.CONNECT_PROC;

CONNECT RESET;

CONNECT TO TEST;
SELECT * FROM SECURITY.AUDIT;
```

Can be converted automatically: NO