DB2 - TRIGGERS

This chapter describes triggers, their types, creation and dropping of the triggers.

Introduction

A trigger is a set of actions, which are performed for responding to an INSERT, UPDATE or DELETE operation on a specified table in the database. Triggers are stored in the database at once. They handle governance of data. They can be accessed and shared among multiple applications. The advantage of using triggers is, if any change needs to be done in the application, it is done at the trigger; instead of changing each application that is accessing the trigger. Triggers are easy to maintain and they enforce faster application development. Triggers are defined using an SQL statement “CREATE TRIGGER”.

Types of triggers

There are three types of triggers:

1. BEFORE triggers

They are executed before any SQL operation.

2. AFTER triggers

They are executed after any SQL operation.

Creating a BEFORE trigger

Let us see how to create a sequence of trigger:

Syntax:

```
db2 create sequence <seq_name>
```

Example: Creating a sequence of triggers for table shopper.sales1

```
db2 create sequence sales1_seq as int start with 1 increment by 1
```

Syntax:

```
db2 create trigger <trigger_name> no cascade before insert on
<table_name> referencing new as <table_object> for each row set
<table_object>.<col_name>=nextval for <sequence_name>
```

Example: Creating trigger for shopper.sales1 table to insert primary key numbers automatically

```
db2 create trigger sales1_trigger no cascade before insert on
shopper.sales1 referencing new as obj for each row set
obj.id=nextval for sales1_seq
```

Now try inserting any values:

```
db2 insert into shopper.sales1(itemname, qty, price)
values('bicks', 100, 24.00)
```

Retrieving values from table

Let us see how to retrieve values from a table:

Syntax:

```
db2 select * from <tablename>
```

Example:

```
db2 select * from shopper.sales1
```

Output:

```
  ID       ITEMNAME       QTY
-------  ------------   ----------
    3      bicks            100
    2      bread            100

  2 record(s) selected.
```

Creating an AFTER trigger

Let us see how to create an after trigger:

Syntax:

```
db2 create trigger <trigger_name> no cascade before insert on
<table_name> referencing new as <table_object> for each row set
 <table_object>.<col_name>=nextval for <sequence_name>
```

Example: [To insert and retrieve the values]

```
db2 create trigger sales1_tri_after after insert on shopper.sales1
for each row mode db2sql begin atomic update shopper.sales1
set price=qty*price; end
```

Output:

```
//inseting values in shopper.sales1
db2 insert into shopper.sales1(itemname,qty,price)
values('chiken',100,124.00)
//output
ID     ITEMNAME        QTY          PRICE
----- -------------- ----------- -----------
    3 bicks          100         2400.00
    4 chiken         100         12400.00
    2 bread          100         2400.00

 3 record(s) selected.
```

Dropping a trigger

Here is how a database trigger is dropped:

Syntax:

```
db2 drop trigger <trigger_name>
```

Example:

```
db2 drop trigger slaes1_trigger
```