Lab 11

TRIGGERS:

Triggers are the SQL statements that are automatically executed when there is any change in the database. The triggers are executed in response to certain events (INSERT, UPDATE or DELETE) in a particular table. These triggers help in maintaining the integrity of the data by changing the data of the database in a systematic fashion.

SYNTAX

create trigger Trigger_name

(before | after)

[insert | update | delete]

on [table_name]

[for each row]

[trigger_body]

Characteristics

The following are the main characteristics that distinguish triggers from stored procedures:

A. We cannot manually execute/invoked triggers.

B. Triggers have no chance of receiving parameters.

C. A transaction cannot be committed or rolled back inside a trigger.

1**. CREATE TRIGGER:** These two keywords specify that a triggered block is going to be declared.

2**. TRIGGER_NAME:** It creates or replaces an existing trigger with the Trigger_name. The trigger name should be unique.

3. **BEFORE | AFTER:** It specifies when the trigger will be initiated i.e., before the ongoing event or after the ongoing event.

4. **INSERT | UPDATE | DELETE:** These are the DML operations and we can use either of them in a given trigger.

5**. ON[TABLE_NAME]:** It specifies the name of the table on which the trigger is going to be applied.

6. FOR EACH ROW: Row-level trigger gets executed when any row value of any column changes.

7**. TRIGGER BODY:** It consists of queries that need to be executed when the trigger is called.

Advantages of Triggers

Triggers provide a way to check the integrity of the data. When there is a change in the database the triggers can adjust the entire database.

Triggers help in keeking User Interface lightweight. Instead of putting the same function call all over the application you can put a trigger and it will be executed.

Disadvantages of Triggers

Triggers may be difficult to troubleshoot as they execute automatically in the database. If there is some error then it is hard to find the logic of trigger because they are fired before or after updates/inserts happen.

The triggers may increase the overhead of the database as they are executed every time any field is updated.

Types of triggers

A trigger defines a set of actions that are performed in response to an insert, update, or delete operation on a specified table. When such an SQL operation is executed, the trigger is said to have been activated. Triggers are optional and are defined using the CREATE TRIGGER statement.

Triggers can be used, along with referential constraints and check constraints, to enforce data integrity rules. Triggers can also be used to cause updates to other tables, automatically generate or transform values for inserted or updated rows, or invoke functions to perform tasks such as issuing alerts.

The following types of triggers are supported:

BEFORE triggers

Run before an update, or insert. Values that are being updated or inserted can be modified before the database is actually modified. You can use triggers that run before an update or insert in several ways:

• To check or modify values before they are actually updated or inserted in the database. This is useful if you must transform data from the way the user sees it to some internal database format.

• To run other non-database operations coded in user-defined functions.

BEFORE DELETE triggers

Run before a delete. Checks values (a raises an error, if necessary).

AFTER triggers

Run after an update, insert, or delete. You can use triggers that run after an update or insert in several ways:

• To update data in other tables. This capability is useful for maintaining relationships between data or in keeping audit trail information.

• To check against other data in the table or in other tables. This capability is useful to ensure data integrity when referential integrity constraints aren't appropriate, or when table check constraints limit checking to the current table only.

• To run non-database operations coded in user-defined functions. This capability is useful when issuing alerts or to update information outside the database.

INSTEAD OF triggers

Describe how to perform insert, update, and delete operations against views that are too complex to support these operations natively. They allow applications to use a view as the sole interface for all SQL operations (insert, delete, update and select).

• BEFORE triggers

By using triggers that run before an update or insert, values that are being updated or inserted can be modified before the database is actually modified. These can be used to transform input from the application (user view of the data) to an internal database format where desired.

• AFTER triggers

Triggers that run after an update, insert, or delete can be used in several ways.

• INSTEAD OF triggers

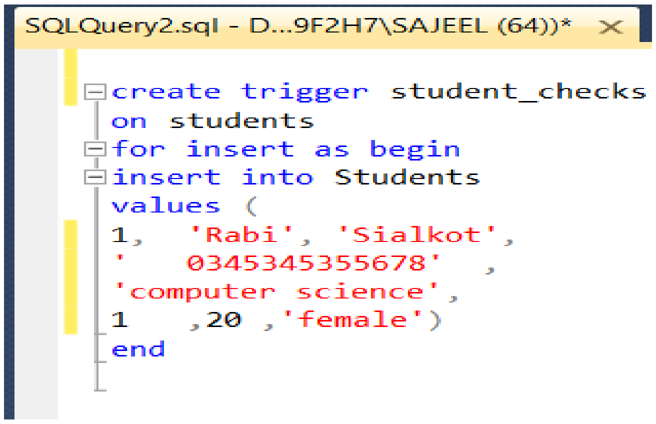
INSTEAD OF triggers describe how to perform insert, update, and delete operations against complex views. INSTEAD OF triggers allow applications to use a view as the sole interface for all SQL operations (insert, delete, update and select).

Task:

Create trigger on project database.

QUERY:

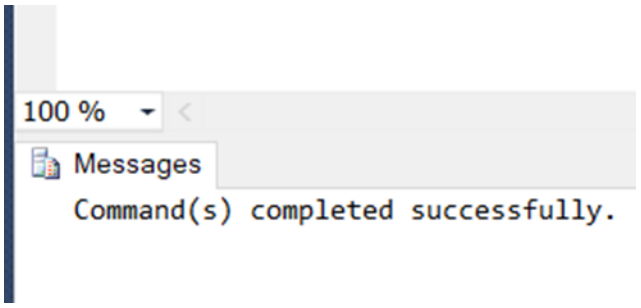
```
create trigger student_checks
on students
for insert as begin
insert into Students
values (
1,      'Rabi',        'Sialkot',
'      0345345355678'       ,
'computer science',
1      ,20     ,'female')
end
```



SQLQuery2.sql - D...9F2H7\SAJEEL (64))*

```
create trigger student_checks
on students
for insert as begin
insert into Students
values (
1,   'Rabi', 'Sialkot',
'    0345345355678'   ,
'computer science',
1    ,20 ,'female')
end
```

Output:



100 %

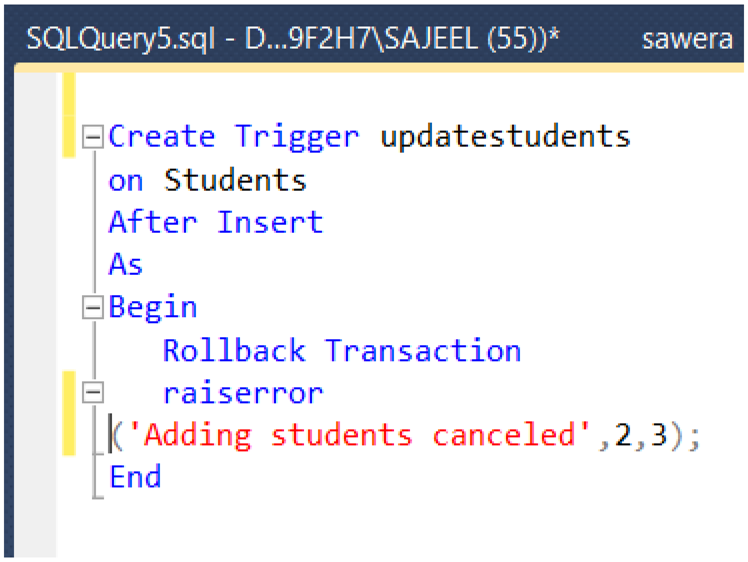
Messages

```
Command(s) completed successfully.
```

QUERY:

```
Create Trigger updatestudents
on Students
After Insert
```

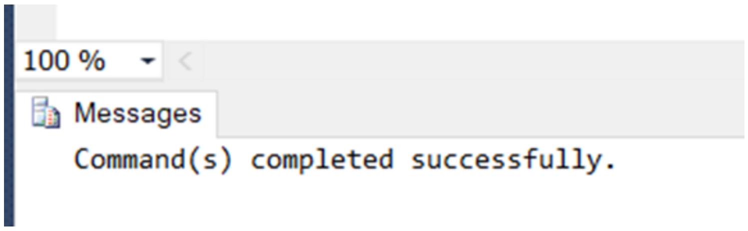
```
As
Begin
   Rollback Transaction
   raiserror
('Adding students canceled',2,3);
End
```



SQLQuery5.sql - D...9F2H7\SAJEEL (55))*   sawera

```
Create Trigger updatestudents
on Students
After Insert
As
Begin
    Rollback Transaction
    raiserror
('Adding students canceled',2,3);
End
```

Output:



```
Command(s) completed successfully.
```