**TRIGERS**

**AIM: Implementation of Triggers & Assertions for Bank Database**

**INTRODUCTION:**

A trigger is a named PL/SQL block stored in the Oracle Database and executed automatically when a triggering event takes place. The event can be any of the following:

* A data manipulation language (DML) statement executed against a table e.g., INSERT, UPDATE, or DELETE. For example, if you define a trigger that fires before an INSERT statement on the customers table, the trigger will fire once before a new row is inserted into the customers table.
* A data definition language (DDL) statement executes e.g., CREATE or ALTER statement. These triggers are often used for auditing purposes to record changes of the schema.
* A system event such as start-up or shutdown of the Oracle Database.
* A user event such as login or logout.

The act of executing a trigger is also known as firing a trigger. We say that the trigger is fired.

Example: Given library book management database schema with student database schema. In these databases any student borrows a book from library then the count of that specified book should be decremented.

**SYNTAX:**

**create trigger [trigger name]**

**[before | after]**

**{insert | update| delete}**

**on [table\_name]**

**[for each row]**

**[trigger\_body]**

Breakdown of the terms mentioned above:

1. create trigger [trigger\_name]: Creates or replaces an existing trigger with the trigger\_name.
2. [before | after]: This specifies when the trigger will be executed.
3. {insert | update | delete}: This specifies the DML operation.
4. on [table\_name]: This specifies the name of the table associated with the trigger.
5. [for each row]: This specifies a row-level trigger i.e., the trigger will be executed for each row being affected.
6. [trigger\_body]: this provides the operation to be performed as trigger is fired.

\*\*\*\*Before and after of Trigger\*\*\*\*

Before triggers run the trigger action before the triggering statement is run.

After triggers run the trigger action after the triggering statement is run.

**QUERY:**

<BEFORE-INSERT>

* 1. student table creation

create table marks(

s\_id varchar(6) not null,

name varchar(20) not null,

subj1 number(5),

subj2 number(5),

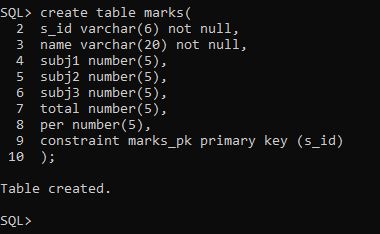
subj3 number(5),

total number(5),

per number(5),

constraint marks\_pk primary key (s\_id)

);



* 1. Sql Trigger to problem statement

create trigger marks

before INSERT on marks

for each row

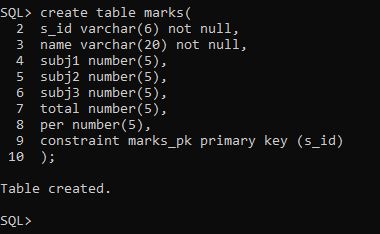
begin

:new.total := :new.subj1 + :new.subj2 + :new.subj3;

:new.per := :new.total / 300\*100;

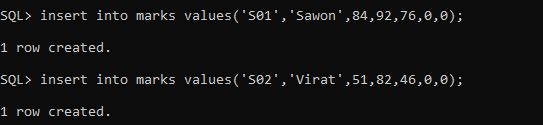
end;

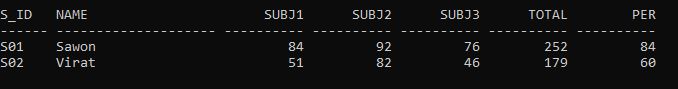
/



insert into marks values('S01','Sawon',84,92,76,0,0);

insert into marks values('S02','Virat',51,82,46,0,0);





<AFTER-INSERT>

1. book\_details

create table books(

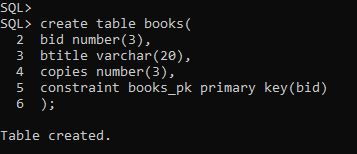
bid number(3),

btitle varchar(20),

copies number(3),

constraint books\_pk primary key(bid)

);



1. Book issue\_details

create table issue(

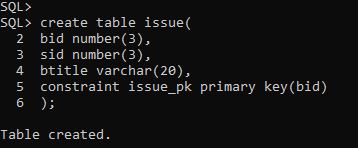
bid number(3),

sid number(3),

btitle varchar(20),

constraint issue\_pk primary key(bid)

);



1. Trigger creation

set verify off;

create or replace trigger book\_issue

after insert on issue

for each row

enable

begin

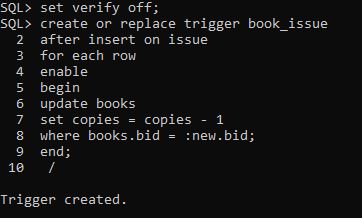
update books

set copies = copies - 1

where books.bid =: new.bid;

end;

/



1. Insert book details

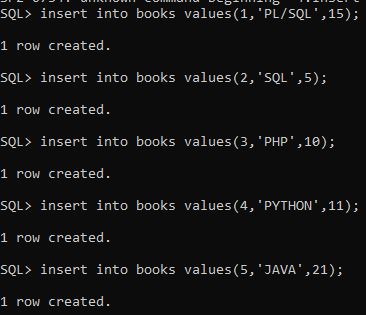
insert into books values(1,'PL/SQL',15);

insert into books values(2,'SQL',5);

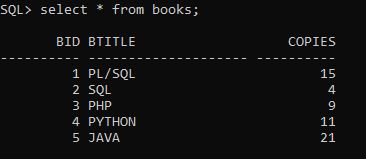
insert into books values(3,'PHP',10);

insert into books values(4,'PYTHON',11);

insert into books values(5,'JAVA',21);



#Initial Values in Books:



1. Insert book issue\_details

insert into issue values(5,4,'JAVA');

