Triggers in mysql

Triggers are procedures which gets executed automatically.

Types of triggers

1. Database level(can be written in oracle but not in mysql)
    a. after startup
    b. before shutdown
    c. after login
    d. before logout

2. table level/statement level/DDL trigger(can be written in oracle but not in mysql)
    a. before create table
    b. after create table
    c. before/ after drop table
3. row level/DML triggers(can be written in both oracle and mysql)
    a. before /after insert/update/ delete

    where to use triggers
    1. if you want to monitor changes in a table
    2. to gather data for analysis purpose
    3. to manage complex views (called as instead of triggers, but can be written in oracle and not in mysql)
    4. to manage denormalization

To write a trigger

create a table to store information required for monitoring or analysis

create table dept_info(

dno int,

old_dname varchar(20),

new_dname varchar(20),

uname varchar(20),

changetime datetime,

action varchar(20))

syntax for trigger

delimiter $$

```
CREATE TRIGGER trigger_name trigger_time trigger_event
ON table_name
FOR EACH ROW
BEGIN
...
END$$
Delimiter ;
```

```
Delimiter $$
Create trigger dept_monitor before insert on dept
For each row
Begin
   Insert into dept_info(dno,new_dname,uname,changetime,action)
  Values(NEW.deptno,NEW.dname,current_user(),now(),'insert before')
End$$
Delimiter ;
```

```
Insert into dept values(11,'HR','Pune');
Old     ---- null
New   11,  'HR',   'Pune'
```

```
Delete from dept where deptno=10;
```

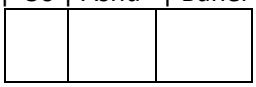
```
OLD  10    Purchase   Pune
NEW     ---- null
```

```
Update dept
Set dname='sales'
Where deptno=30;
```

```
OLD 30  marketing    Mumbai
```

```
NEW 30   sales      Mumbai
```

```
+-----+--------+---------+
| cid | cname  | address |
+-----+--------+---------+
|  20 | Rajesh | Pune    |
|  30 | Ashu   | Baner   |
```



```
Update customer
Set cname='Rajesh Joshi'
Where cid=20
```

```
Old  20 Rajesh   pune
New 20  Rajesh Joshi pune
```

```
Create trigger update_vehicle after update on customer
For each row
begin
Update vehicle
Set cname=NEW.cname
Where custid=OLD.cid;
End$$
Delimiter ;
```

Exception handling

```
delimiter $$
drop procedure insert_vehicle$$
create procedure insert_vehicle(pvid int,pvnm varchar(20),pmd varchar(20),pcn int,pcid int,pcnm va
rchar(20))
   begin

      #declare continue handler for SQLEXCEPTION select 'error occured' msg;
      declare exit handler for 1062 select 'duplicate entry';
      declare exit handler for SQLEXCEPTION select 'error occured' msg;
      insert into vehicle values(pvid,pvnm,pmd,pcn,pcid,pcnm);
      select * from vehicle;
   end$$

delimiter ;


create procedure myproc
begin
----- declare all variables
----- declare all specific exception
----- declare generalized exception
-----  write code
end;

delimiter $$

create procedure insert_testtable(pid int,pnm varchar(20),passnum int)
   begin
      declare exit handler for 1062 select 'duplicate entry' msg
                  ;
      declare continue handler for 1364 select 'null not allowed' msg;
      declare exit handler for SQLEXCEPTION select 'error occured' msg;
      insert into testtable(id,passport) values(pid,passnum);
      select * from testtable;
   end$$

delimiter ;
```

Top n analysis

To find 5 highly paid employees

```
select empno,ename,sal
from emp e
where 5 > (select count(*) from emp m
   where m.sal > e.sal);
```

To find 10 highly paid employees

```
select empno,ename,sal
```

```
from emp e
where 10 > (select count(*) from emp m
   where m.sal > e.sal);
```