Exception

```
delimiter //
create procedure insertempdept3(eid int,ejob varchar(20),did int)
begin
declare vdept int default 0
 declare continue handler for 1265 begin
 select "too long value";
 select 'xxx';
 select 'yyy';
 end;
 declare continue handler for 1062 begin
 if vdept=1 then
    select "duplicate entry in dept";
 else
    select "duplicate entry";
 end;
 declare exit handler for SQLEXCEPTION select "error occured";
 update emp
 set job=ejob
 where empno=eid;
 insert into emp values(eid,'xxx',ejob,123,'2020-11-11',45678,null,10);
 set vdept=1
 insert into dept values(did,'yyy','tttt');



end//
delimiter ;
```

2. write a procedure to insert record in product table, handle 1452 , for wong cid or sid , handle 3819 to display message ‘qty cannot be -ve’ , also handle the SQLEXCEPTION write all continue handler

```
delimiter //
create procedure insertupdateproduct1(ppid int,pcid int, psid int,pnm varchar(20),pqty int,
pprice float(9,2))
begin

  declare continue handler for 1452 begin
    select 'wrong sid or cid';
    select 'wrong values';
  end;
  declare continue handler for 3819 select 'qty cannot be -ve';
  declare continue handler for SQLEXCEPTION select 'error occured';
  insert into product values(ppid,pnm,pqty,pprice,pcid,psid);
  update product
  set cid=pcid+2
  where pid=ppid;
end//
delimiter ;
```

to raise user defined exception

3. write a procedure to display error message employee not found if empno is wrong,

if salary is -ve then display message salary cannot be -ve' using custom exception

following is syntax for user defined exception

```
SIGNAL SQLSTATE '42927' SET MESSAGE_TEXT = 'Error Generated';
DELIMITER //

CREATE PROCEDURE update_salary(
        IN p_employee_id INT,
   IN p_salary DECIMAL
)
BEGIN
        DECLARE employee_count INT;
```

```
  -- check if employee exists
  SELECT COUNT(*) INTO employee_count
  FROM emp
  WHERE empno = p_employee_id;

  IF employee_count = 0 THEN
                SIGNAL SQLSTATE '45000'
                SET MESSAGE_TEXT = 'Employee not found';
  END IF;

  -- validate salary
  IF p_salary < 0 THEN
                SIGNAL SQLSTATE '45000'
                SET MESSAGE_TEXT = 'Salary cannot be negative';
  END IF;

  -- if every is fine, update the salary
  UPDATE emp
  SET sal = p_salary
  WHERE empno = p_employee_id;

END //

DELIMITER ;

using custom exception
CREATE PROCEDURE p (pval INT)
BEGIN
 DECLARE specialty CONDITION FOR SQLSTATE '45000';
 IF pval = 0 THEN
```

```
  SIGNAL SQLSTATE '01000';
 ELSEIF pval = 1 THEN
  SIGNAL SQLSTATE '45000'
    SET MESSAGE_TEXT = 'An error occurred';
 ELSEIF pval = 2 THEN
  SIGNAL specialty
    SET MESSAGE_TEXT = 'An error occurred';
 ELSE
  SIGNAL SQLSTATE '01000'
    SET MESSAGE_TEXT = 'A warning occurred', MYSQL_ERRNO = 1000;
  SIGNAL SQLSTATE '45000'
    SET MESSAGE_TEXT = 'An error occurred', MYSQL_ERRNO = 1001;
 END IF;
END;
```

explanation :

If `pval` is 0, `p()` signals a warning because `SQLSTATE` values that begin with `'01'` are signals in the warning class. The warning does not terminate the procedure, and can be seen with `SHOW WARNINGS` after the procedure returns.
If `pval` is 1, `p()` signals an error and sets the `MESSAGE_TEXT` condition information item. The error terminates the procedure, and the text is returned with the error information.
If `pval` is 2, the same error is signaled, although the `SQLSTATE` value is specified using a named condition in this case.
If `pval` is anything else, `p()` first signals a warning and sets the message text and error number condition information items. This warning does not terminate the procedure, so execution continues and `p()` then signals an error. The error does terminate the procedure. The message text and error number set by the warning are replaced by the values set by the error, which are returned with the error information.

Triggers

Triggers are the procedures which are automatically gets executed, when user performs some action.

Triggers can be written at database level, table level or row level

database level tiggers—these trigger works in oracle/ but not in mysql

1. login /logout
2. server start/shutdown

Table level ------ these works I oracle but not in mysql
create/ drop tables

when we execute DML operation then we can execute some trigger
2 types trigger

1. statement level --- insert action, and if you inserted multiple rows by using single insert statement, then all the rows will get added in the table, but trigger will get executed only once, these triggers works in oracle but not in SQL.
2. row level ---- In mysql only row level trigger works , for all DML operations
    a. these trigger gets executed once for each affected row
3. insteadof ---- these triggers are used only for views, does not work in mysql, but works in oracle.

create view
myview
as
select * from
emp_india
union
select * from
emp_US
	insert into myview values(1,'xxx','india')
	
	create trigger inseremp insteadof on myview
	if new.location='india' then
	insert intp emp_india
	values(new.id,new.name.new.location);
	else
	insert intp emp_US
	values(new.id,new.name.new.location);

syntax

create trigger <name of the trigger> <timing> <DMLoperation> on <tablename>

for each row

statements

before	the trigger will get executed before the DML operation gets executed
after	the trigger will get executed after the DML operation gets executed
insteadof	the trigger will get executed, no DML operation gets executed

1. write a trigger to monitor insert , update and delete action on emp table
   if anyone changes name/sal/job store the changes

    1. create table emp_audit(
       empno int,
       oldname varchar(20),
       newname varchar(20),
       oldsal float(9,2),
       newsal float(9,2),
       oldjob varchar(20),
       newjob varchar(20),
       uname varchar(20),
       changes datetime,
       action varchar(20));

by default , every trigger receives 2 special values

	update	insert	delete
old	it will store the data existing in the table corresponding to empno.	null	the row which is in the table
new	it will store the data which gets stored in the table , after updation is done.	the data which is getting inserted in the table	null

```
------insert trigger
create trigger insertemp after insert on emp
for each row
  insert into emp_audit values(
        new.empno,null,new.ename,null,new.sal,null,new.job
        ,user(),now(),'insert')

-------update trigger
create trigger updateemp after update on emp
for each row
  insert into emp_audit values(
        new.empno,old.ename,new.ename,old.sal,new.sal,old.job,new.job
        ,user(),now(),'insert')

---delete trigger
create trigger deleteemp after delete on emp
for each row
  insert into emp_audit values(
        old.empno,old.ename,null,old.sal,null,old.job,null
        ,user(),now(),'delete')
```

2. write update, insert trigger for product table to monitor changes in qty and price columns in product(pid,pname,qty,price,catid,sid)

```
create table product_audit(
pid int,
pname varchar(20),
oldqty int,
newqty int,
oldprice float(9,2),
newprice float(9,2),
uname varchar(20),
changedt datetime,
action varchar(20));

create trigger updateproduct before update on product
for each row
```

```
    insert into product_audit
values(old.pid,old.pname,old.qty,new.qty,old.price,new.price,user(),now(),’upda
te’);

create trigger insertproduct before insert on product
for each row
    insert into product_audit
            values(new.pid,new.pname,null,new.qty,null,
            new.price,user(),now(),'insert');
```