**CREATE\_TABLE**

create table department(

dep\_id number(5) primary key,

department\_name varchar2(30)

);

**CREATE\_TABLE BY ANOTHER TABLE**

create table department\_backup as select \* from department;

**CREATE\_TABLE\_WITH\_FOREIGN\_KEY**

create table employee(

emp\_id number primary key,

emp\_name varchar2(30),

mobile varchar2(15) unique,

salary number(7,2) check(salary > 10000),

joining\_date date default SYSDATE,

country varchar2(30) default 'BD',

dep\_id number(5), constraint dep\_emp\_fk FOREIGN KEY (dep\_id) REFERENCES department(dep\_id)

**SHOW TABLE STRUCTRES**

describe department;

**CREATE INSERT\_TRIGGER:**

CREATE or REPLACE TRIGGER employee\_after\_insert AFTER INSERT ON employee

FOR EACH ROW

DECLARE

BEGIN

insert into employee\_backup values(:new.emp\_id, :new.emp\_name, :new.mobile, :new.salary, :new.joining\_date, :new.country, :new.dep\_id);

DBMS\_OUTPUT.PUT\_LINE('Record Successfully Inserted Into employee\_backup Table');

END;

/

**CREATE UPDATE\_TRIGGER**

CREATE or REPLACE TRIGGER employee\_after\_update AFTER UPDATE on employee

FOR EACH ROW

DECLARE

BEGIN

update employee\_backup

set emp\_name = :new.emp\_name, mobile = :new.mobile, salary = :new.salary, joining\_date = :new.joining\_date, country = :new.country, dep\_id = :new.dep\_id

where emp\_id = :old.emp\_id;

DBMS\_OUTPUT.PUT\_LINE('Record Successfully Updated Into employee\_backup Table');

END;

/

**CREATE DELETE\_TRIGGER**

CREATE or REPLACE TRIGGER employee\_after\_delete

AFTER DELETE ON employee

FOR EACH ROW

DECLARE

BEGIN

Delete employee\_backup

where emp\_id = :old.emp\_id;

DBMS\_OUTPUT.PUT\_LINE('Record Successfully Deleted From employee\_backup Table');

END;

/

**CREATE SEQUENCE**

create sequence employee  
    start with 8  
    increment by 2  
    maxvalue 10000  
    nocycle  
    nocache;

**SHOW CREATED SEQUENCE LIST**

select sequence\_name from user\_sequences;

**SHOW INDEX:**

select index\_name from user\_indexes;

**CREATE INDEX**

CREATE INDEX emp\_INDX ON emp (empno, deptno) TABLESPACE index\_tbs;

OR

create index dep\_name\_idx on department(department\_name);

**CREATE INSERT PROCEDURE**

CREATE OR REPLACE PROCEDURE insertDepertment(

p\_id IN DEPARTMENT.dep\_id%TYPE,

p\_name IN DEPARTMENT.department\_name%TYPE)

IS

BEGIN

INSERT INTO DEPARTMENT(dep\_id, department\_name)

VALUES(p\_id, p\_name);

END;

/

**CALL/DATA INSERT BY PROCEDURE**

BEGIN

insertDepertment(dep\_id\_seq.nextval,'ADMIN');

END;

**CREATE UPDATE PROCEDURE**

CREATE OR REPLACE PROCEDURE updateCustomer(

p\_id IN CUSTOMER.id%TYPE,

p\_name IN CUSTOMER.name%TYPE)

IS

BEGIN

update CUSTOMER set name = p\_name where id = p\_id;

END;

/

**CALL/DATA UPDATE BY PROCEDURE**

Begin

updateCustomer(100, 'Mr. Mahbub');

end;

**CREATE DELETE PROCEDURE**

CREATE OR REPLACE PROCEDURE deleteCustomer(

p\_id IN CUSTOMER.id%TYPE)

IS

BEGIN

delete from CUSTOMER where id = p\_id;

END;

**CALL/DATA DELETE BY PROCEDURE**

Begin

deleteCustomer(100);

end;

**CREATE VIEW**

create view depv as select dep\_id, department\_name from department;

**SHOW VIEW LIST**

select view\_name from user\_viewes;

**ADD A COLUMN**

ALTER TABLE employee ADD(email VARCHAR2(30) unique);

**CREATE SYNONYM**

CREATE SYNONYM emp FOR SCOTT.EMP;