# Create/Drop/Describe Table

create table department (dept_id number(7) primary key, dept_name varchar2(50) unique, regi_date date default sysdate);

create table contact (con_id number(7) primary key, mobile number(15) unique, country varchar2(50) default 'Bangladesh')

create table employee (id number(7) primary key, emp_name varchar2(20) not null, salary number(7,2) check(salary>=25000), joining_date date check(joining_date<='01-Jan-2000'), dept_id number(7) not null, con_id number(7), constraint emp_dep foreign key (dept_id) references department, constraint emp_con foreign key (con_id) references contact);

create table department_backup as select * from department;

drop table dhaka;

desc dhaka;

# Create Role/User

create role super_user;

create user shuvo identified by shuvo;

# Grant Privilege

grant create session, create table, create view, create sequence, create procedure, create synonym, create trigger to super_user;

grant super_user to shuvo;

grant dba to shuvo;

# Grant Tablespace to User

grant unlimited tablespace to shuvo;

alter user olive quota 500m on system;

# Insert Data

insert into department values (dept_seq.nextval, 'Administration');

insert into contact (con_id, mobile, country) values(con_seq.nextval, 01712345678, 'Japan');

insert into employee (id, emp_name, salary, joining_date, dept_id, con_id) values(emp_seq.nextval, 'AAA', 25000, '01-Feb-1998', 10, 10);

# Table Privilege/Add Constraints

ALTER TABLE dhaka READ ONLY;

ALTER TABLE dhaka READ WRITE;

ALTER TABLE department ADD (job_id VARCHAR2(9));

ALTER TABLE department DROP COLUMN job_id;

alter table department modify (dept_name varchar2(55));

alter table employee add constraints emp_dept_id unique (con_id);

# Update Data

update dhaka set name='C' where income = 30000;

update employee set salary=35000, emp_name='BBB' where con_id=15;

update company set brand_id = (select brand_id from company where name='Google Inc.') where id=03;

# Create View/Sequence/Synonym/Index

create sequence dept_seq increment by 10 start with 10 maxvalue 5000 nocache nocycle;

create sequence con_seq increment by 5 start with 5 nocache nocycle;

create sequence emp_seq increment by 1 start with 1 nocache nocycle;

create view dha_view1 as select id, name, email from dhaka;

create or replace view empvu80 (id_number, name, sal, department_id) as select employee_id, first name|| ' ' || last_name, salary, department_id from employees;

CREATE OR REPLACE VIEW dept_sum_vu (name, minsal, maxsal, avgsal) AS SELECT d.department_name, MIN(e.salary), MAX(e.salary), AVG(e.salary) FROM employees e JOIN departments d ON (e.department_id = d.department_id) GROUP BY d.department_name;

CREATE INDEX emp_last_name_idx ON employees(last_name);

create synonym e20 for empvu80;

# Show View/Table/Sequence/Username

select view_name from user_views;

select table_name from user_tables;

select sequence_name from user_sequences;

select USERNAME from DBA_USERS;

## Autocommit

show autocommit;

set autocommit on;

## Delete Data

delete brand where id=04;

delete from department;

truncate table brand; (Works with DDL Statement)

## Spool On/Off

spool E:/quiz.txt;

spool off;

## Create Trigger for Insert

### Step-1: Create Two Tables

create table usa (id number (5), fname varchar2(55), lname varchar2(55));

create table usa_backup as select * from usa;

### Step-2: Create Trigger:

CREATE OR REPLACE TRIGGER usa_trigger AFTER insert ON usa (**NB: after update/before delete**)

FOR EACH ROW

DECLARE

BEGIN

insert into usa_backup values (:new.id, :new.fname, :new.lname);

dbms_output.put_line ('Data inserted successfully on usa_backup table');

END;

/

### Step-3: Insert Data & View from Backup Table

insert into usa values (01,'Mehedi','Hasan');

select * from usa_backup;

# Create Trigger for Update

## Step-1: Create Trigger

```
CREATE OR REPLACE TRIGGER usa_trigger_update AFTER update ON usa
FOR EACH ROW
DECLARE
BEGIN
UPDATE usa_backup
set id = :new.id, name = :new.name, lname = :new.lname
where id = :old.id or name = :old.name or lname = :old.lname;
dbms_output.put_line ('Data successfully updated into usa_backup table');
END;
```

## Step-2: Update Main Table & View from Backup Table

```
update usa set name = 'Hollywood' where id = 5010;
update usa set id = 5010 where name = 'Hollywood';
select * from usa_backup;
```

# Create Trigger for Delete

## Step-1: Create Trigger

```
CREATE OR REPLACE TRIGGER usa_trigger_delete AFTER delete ON usa
FOR EACH ROW
DECLARE
BEGIN
DELETE from usa_backup
where id = :old.id or name = :old.name;
dbms_output.put_line ('Data successfully deleted from usa_backup table');
END;
```

## Step-2: Delete from Main Table & View from Backup Table

```
delete from usa where name = 'Los Angeles';
delete from usa where id = 5010;
select * from usa_backup;
```

# Create Procedure for Insert

## Step-1: Create Table

```
Create Table Customer (id number(5) primary key, name varchar2(55) not null, mobile varchar2(55), country varchar2(55));
```

### Step-2: Create Insert Procedure

```
CREATE OR REPLACE PROCEDURE insertCustomer(
p_id IN customer.id%TYPE,
p_name IN customer.name%TYPE,
p_mobile IN customer.mobile%TYPE,
p_country IN customer.country%TYPE)
IS
BEGIN
INSERT into customer (id, name, mobile, country)
VALUES (p_id, p_name, p_mobile, p_country);
COMMIT;
END;
/
```

### Step-3: Calling Insert Procedure

```
BEGIN
insertCustomer(101, 'Shuvo', 01711000001, 'Bangladesh');
END;
/
```

## Create Procedure for Update

### Step-1: Create Update Procedure

```
CREATE OR REPLACE PROCEDURE updateCustomer(
p_id IN customer.id%TYPE,
p_name IN customer.name%TYPE,
p_mobile IN customer.mobile%TYPE,
p_country IN customer.country%TYPE)
IS
BEGIN
UPDATE customer SET name = p_name, mobile = P_mobile, country = p_country where id = p_id or name = p_name;
COMMIT;
END;
/
```

### Step-2: Calling Update Procedure

```
BEGIN
```

```
updateCustomer(102,'Masud',01711000002,'France');
END;
/
```

# Create procedure for delete

## Step-1: Create Update Procedure

```
CREATE OR REPLACE PROCEDURE deleteCustomer(
p_id IN customer.id%TYPE,
p_name IN customer.name%TYPE)
IS
BEGIN
DELETE from customer where id = p_id or name = p_name;
COMMIT;
END;
/
```

## Step-2: Calling Delete Procedure

```
BEGIN
deleteCustomer(103, 'Rafin');
END;
/
```