DBMS LAB WORK

DAY 1

```
create table employee(
   e_id number(4),
   e_name varchar2(10),
   designation varchar2(20),
   salary number(5),
   city varchar2(15)
);
```

Note --

varchar is more data saving than char, in normal char if we define the data to be of 10 in length and we inputted only 3 characters then other 7 remaining spaces will still be occupied, but in varchar the remaining spaces are deallocated making varchar more space conscious.

Numeric value is more preferred as a primary key.

Maximum no. of primary key 1, minimum is 0.

Whether two attributes are joined to make a primary key, that still will be called a single primary key. If asked then we'll say "there is 1 primary key, (attrib1 comma attrib2)." Concept of making a primary key from multiple attributes is called composite key.

Maximum 17 primary attributes can be joined to make a primary key (composite).

desc employee; // tells the description of the table.

drop table employee; // deletes the whole table.

```
create table employee(
   e_id num ber(4) primary key,
   e_name varchar2(10),
   designation varchar2(20),
   salary number(5),
   city varchar2(15)
);
```

This way we create a primary key in the table.

Making any key a primary key imposes two constraints on that attribute.

- ***Not Nullity***
- ***Uniqueness***

DAY 2

Primary key -> Not Null & Uniqueness are two main features of a primary key.

3 ways to make an attribute as primary key.

1. **During the creation of the table.**
   ```
   create table employee(
       e_id number(10) primary key
   );
   Also,
   create table employee(
       e_id number(10)
       primary key(e_id)
   );
   ```
2. We can have more than one primary attribute in the table, but the number of primary key will always be one.

   Primary key + any other attribute is a called a super key. Only primary key is also a super key.

   Creating multiple primary attributes in the table… **Composite Key**
   ```
   create table employee(
      e_id number(10),
      e_name varchar2(15),
      primary key(e_id, e_name)
   );
   ```
3. We can also assign primary key to the table by using alter keyword.

   using alter command

   ```
   alter table employee add primary key(e_id)
   for composite key
   alter table employee add primary key(e_id, e_name)
   ```

   After insertion of the data…

   if interviewer asks “can we include primary key now?”

   Ans must be **“depends”**

   First we’ll check the whether the attribute we want to make the primary holds true for not nullity and uniqueness, if it holds then only we can make it a primary key otherwise not.

How to remove a primary key from a table.

```
alter table employee drop primary key;
or
alter table employee drop primary key(e_id);
```

To add an attribute in the table.

```
alter table employee add(designation varchar2(15));
```

To remove an attribute from the table.

```
alter table employee drop(address);
```

To edit the datatype of an attribute.

```
alter table employee modify(salary number(6));
```

DAY 3

To know the date format in the system

select sysdate from dual;

And for date datatype we include it as… …, attribute_name date);

INSERT COMMANDS

To insert data into the table…

insert into employee(e_id, e_name, designation, city, salary, doj) values(3635, 'Raghav', 'Scrum Master', 'Noida', 85500, '21-JUL-13');

OR

If we are inserting **all the attributes at once** we can also use
insert into employee values(3635, 'Raghav', 'Scrum Master', 'Noida', 85500, '21-JUL-13');

The main difference between both is that in the first one we are defining the order of inputting data in the data and then inserting it, we can also change that order and even not write and attribute at all, while in the other one we have to insert all the attribute at once and also follow the exact manner defined at the time of table creation.

To retrieve the data from the table:

select * from employee;

here ***** symbolises for everything in that relation.

To retrieve particular column from table:

select e_id, e_name from employee;

To apply conditional selection:

select * from employee where designation = 'Analyst';

To Update a particular column:

For particular row…

update employee set(Blood_Group) = ('O+') where e_id = 3635;

For all rows (but same data)…

update employee set(Blood_Group) = ('O+');

To delete a row from table:

delete from employee where e_name = 'Nisha';

To remove a column from employee:

alter table employee drop(Blood_Group);

To remove content from the table:

To remove all data:

delete from employee;

To remove particular data:

delete from employee where e_id = 3635;

DAY 4

Foreign Key:

We can have a foreign key in the same table, it is technically possible, though it is rarest of the rare case

*Primary Key possesses -> **Integrity Constraint***

*Foreign Key possesses -> **Referential Integrity***

Keys in DBMS:

The attributes with the help of which we can identify a table uniquely.

1. **Super Key:** Prime Attribute or Prime Attribute + Any attribute in a relation.
2. **Candidate Key:** An attribute which can be a primary key, but is not a primary key.
3. **Alternate (Secondary) Key:** The remaining attributes in a table which can be a primary key, but is not.
4. **Unique Key:** This key can be left blank once, but they all must be unique.
5. **Primary Key:** The **(minimum set of)** attribute which can uniquely identify each tuple in a record.
6. **Foreign Key:** When the primary key of a table is related with the primary key attribute of another table, then it's called Foreign Key.
7. **Composite Key**: When two or more attribute together become the primary key of a relation then they are called composite.
8. **Artificial (Surrogate) Key:** When the DBA makes an attribute for their own for the better use of table, so they can access the data easily. This is artificial as it does not have any relation with the data whatsoever, when this key is made the primary key of the table, then it's called surrogate key.

For a Foreign Key:

-> It should be the primary key of the table it's coming from.

-> Their Domain (Data type) must be same in the table it has to be foreign key of.

-> The name must not be necessarily different, but we prefer different names to remove ambiguity.

Syntax:

-> ***To make foreign key during creation of the table.***

Table 1

```
create table person(
person_ID number(2) primary key,
last_nm varchar2(20),
first_nm varchar2(20),
age number(3)
);
```

Table 2

```
create table Det_order(
order_ID number(2) primary key,
order_quantity number(3),
p_ID number(2),
foreign key(p_ID) references person(person_ID)
);
```

-> ***To make foreign key after table creation***

alter table Det_Order add ***foreign key***(p_ID) ***references*** person(person_ID);

-> ***parent key not found***

We are allowed to enter the data into the parent table, but we can not enter the data into the table having an attribute of parent table as foreign key as we are bound with integrity constraint. We can not insert any data in the child table it's foreign key attribute is not present in the parent table.

Input:

```
insert into Det_order values(5,81,8);
```

Output:

integrity constraint, violated - parent key not found

-> ***child record found***

Like we can not insert the data in the child table, if parent record is not found, similarly we can not delete the data from the parent table, if we find a record of that instance in the child table, this situation gives rise to the message. *The reason being the record we want to remove is being referred to another table.*

Input:

delete from person where person_ID = 2;

Output:

integrity constraint, violated - child record found

Difference between Drop, Delete and truncate?