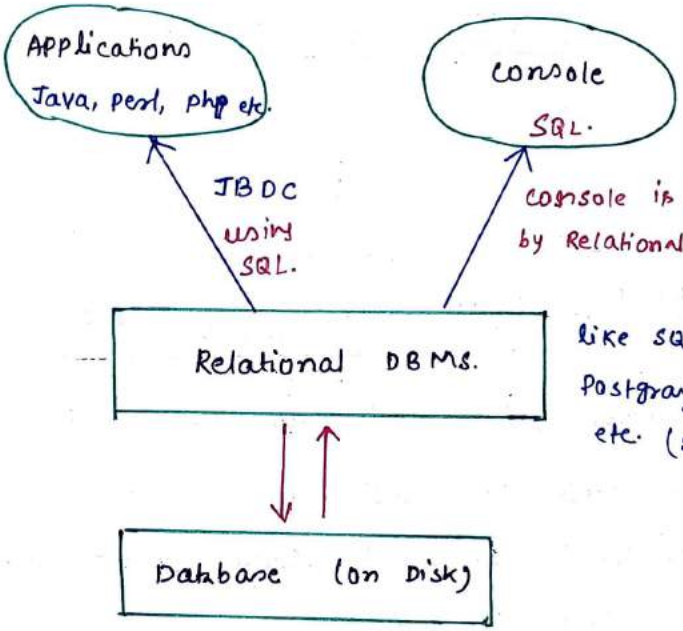
Date : 07/09/2021 | 01. SQL | GFG Placement Course | Abhishek Sharma NOTES.

SQL : Structured Query Language.

* you need to a programming Language that can be used to interact with Relational DBMS. That programming Lang. is SQL.

* we call it Structured Query Lang. bcz. it works on table which are structured data. you have a fixed schema and every rows follows the schema.

SQL ⇒




* you need a programming Lang. That can be used to interact with Relational DBMS.

That Programming Lang. is SQL.

* SQL is the Programming Lang. That can be used for interact with the Relational DBMS.

* As a programer you browse the data using console and you also write Application. and these Applications interact with your Relational DBMS using connections (JBDC Java Database connection) These applications are written in Java, perl, php etc

* That has the facility to interact with DBMS.

* using SQL. (Structured Query Language).

* console also run SQL Query.

* console allows you to Browse the data, Modify the data, to see the structure of data.

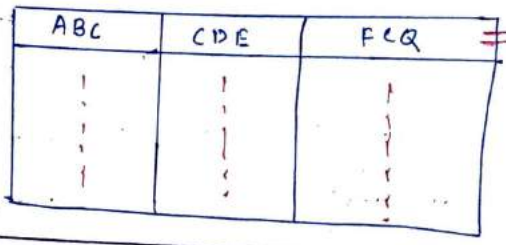
* SQL as a programming Lang. is standerised by ISO. Recent version came in 2019.

* SQL can be divided into Four Sub-languages.

① i) DDL (Data Definition Language) : create, Drop, ALTER

DDL is used to create or modify objects in your DBMS. Object can be a table, it can be a database or it can be other things.

* Example: In terms of table, it used to Modify or create the schema of the table or delete the schema also



⟹ Schema of table.

② ii) DQL (Data Query Language) : SELECT.

It is used for select all the statements.

Ex. you want to select all the rows.
you want to select some specific rows.
you want to select count of certain types of item.

All these tasks come in DQL. (Data Query Language)

③ iii) DML (Data Manipulation Language) : update, Insert, Delete

If you want to change the data, not change the schema. (bcz schema can be changed by DDL). That parts comes in DML.

Example: update the data, insert the data, Delete the data etc.

④ iv) DCL (Data control Language) : GRANT, REVOKE.

It used for Granting or Revoking permission on DBMS objects.

Ex. If you grant permission on a particular view to a particular person and do not want to allow other users to view.

* Example Syntax Of SQL.

Example of creating the Table.

```
CREATE TABLE student (
    student_id  INT,
    f_name      VARCHAR (20),
    l_name      VARCHAR (20),
);
```

(keyword)

Columns of the table

integer type of column.

will make this table.

| student_id | f_name | l_name |
|---|---|---|
| | | |

* If we want to create student_id as primary key we will write in the create Table section.

```
student_id  INT  PRIMARY KEY,
```

* If we want in f_name we don't want any NULL value.

```
f_name  VARCHAR (20)  NOT NULL,
```

Example of Inserting values in the table.

```
INSERT INTO student VALUES (1, "xyz", "abc");
INSERT INTO student VALUES (2, "pqr", "rst");
```

| student_id | f_name | l_name |
|---|---|---|
| 1 | xyz | abc |
| 2 | pqr | rst. |

* SQL is case insensitive. we can also write in ~~small~~ Lowercase alphabate (create table or insert into) but it is Highly Recomidable to use uppercase for keywords. (INSERT, DELETE, CREATE) and so on.

* We can also insert values in the table like this.

INSERT INTO student (student_id, f_name) VALUES (3, "cde");

| student_id | f_name | L_name |
|---|---|---|
| 3 | cde | NULL. |

Here we specify the column name in which we want to insert the data. We don't specify Here L_name so in the l_name ~~3rd~~ There is written NULL.

✓ This type to insert values is the Highly Recomdable. It is the better practice.

use this for insert.

* Example of Insert or Delete the ~~table~~ column from the Table.

For adding a column in the existing table:

ALTER TABLE

ADD address VARCHAR (500);

Our table will look like

| Student_id | f_name | L_name | Address |
|---|---|---|---|
| 1 | xyz | abc | NULL |
| 2 | pqr | mst | NULL |
| 3 | cde | NULL. | NULL |

Our new columns has been created.

Since we don't specify what to insert in Address area so that's why there will be NULL.

Date: 15/09/2021

02. SQL Insert statement:

GFG Placement 100 course
Abhishek Sharma NOTES.

**Insert:** Insert is a DML (Data Manipulation Language) command.

* Insert command is used to add one or more rows of data to the dababase table with specified column values.

* General Syntax:

```
INSERT INTO table_name (column1, column2, column3, ---)
VALUES (value1, value2, value3, -----);
```

First we will write Keyword that is INSERT INTO then we will write table name after that coulum name and then values. (Keyword) and in this we will insert what is value associated with colum 1 and column 2 and so on.

* Simple Insert:

* All the values should be provided to all the Columns.
* Inserting data into required Columns.

---

* Now we will see all the types of INSERT via Help of Example.

Example: 1.

Firstly we need to specify a Table.

code:

```
Select * from student
/* Student ID   Student Name   Branch   DOB   Score */
```

Output:

Student

| Student ID | Student Name | Branch | D.O.B | Score |
|---|---|---|---|---|
| | | | | |

Now we have a table "student" and we have also 5 columns in the "student" . Table.

Now we will insert data into this ~~insert~~ table.

Method : 1. Using Simple insert:

→ i) All the values should provided to all the columns.

Code:

```
insert into Student
values (66, 'Abhishek', 'computess', '07-Aug-2000', 98);
```

O/P: (If we write select * from Student) ↓ this displayed.

| Stu. ID | Stu. Name | Branch | D.O.B | Score |
|---|---|---|---|---|
| 66 | Abhishek | computess | 07-Aug-2000 | 98 |

* In this type of insert we haven't taken names of column. bcz we have to insert data in all the columns. That'why only values are specified and they are inserted as per the column is defined.

(if we insert "Abhishek" first instead of '66' i.e. Stu. ID) then it will show us ERROR . BCZ Student ID will be must of integer type) and so on

* All the character values are seperated by ',' comma.

* In this Method, we need to keep in mind All the columns and as per the columns we need to insert data.

* Date (DOB) datatype should be in 'DD - Mon - YYYY' Format

## METHOD - 2 : using Simple Insert.

### ii) Inserting Data into Required Columns.

previously we have inserted data into all the columns. But now if we want to insert data into some Required column. (Ex: we want to insert data in (Stu ID, Stu Name and Branch) column only).

Code:

```
insert into student ( student ID, student-name , branch) values
                                  (67, 'sharma', 'computers');
```

O/P: (If we write select * from student).

| Student ID | Student Name | Branch | DOB | Score. |
|---|---|---|---|---|
| 66 | Abhishek | computers | 07-Aug-2000 | 98 |
| 67. | sharma. | computers | - | - |

* Where there is no data available in the column there will be blank (-) like this.

* This is the two methods of insert in simple insert Method. Now we will see further the different types (Method) of Insert.

## Method : 3 Dynamic Insert (using ampersand &)

In this Method values are entered by the user at the execution time.

Previously we don't need to insert data., On the Time of execution it will take data from user. (Like in "cin" in C++, and C in "scanf") same.

code:

```
insert into student values (& student id, '& student name', '& branch',
                                                    & score );
```

O/P.

During RunTime it will ask data from user.

## MEthod : 4 Using select in the INSERT Command.

### i) copying all columns of a table

Here we are copying data from one table to another table. we are taking the pre existing data.

In this Method, we will copy list all the ~~data~~ columns from a table to another table.

First we need to create 2 tables.

lets suppose. Sample 1 and Sample 2.

| x | y |
|---|---|
| 2 | 5 |
| 6 | 8 |
| 10 | 15 |

No Data.

Now we will copying all the data from Saple 1 table to Sample 2 table.

Code:

```
INSERT INTO SAMPLE 2 SELECT * FROM SAMPLE1 ;
```

O/P: (If we want to show sample 2 data : SELECT * FROM SAMPLE 2).

Sample 2

| x | y |
|---|---|
| 2 | 5 |
| 6 | 8 |
| 10 | 15 |

Previously it was null (no data).

## ii) Copying specific columns of a Table.

Let's suppose we have two ~~sets~~ Tables Sample 1 and sample 2. like this

| Name | Age |
|---|---|
| Abhishek | 21 |
| Rahul | 22 |
| Sunil | 23 |

Sample 1

| Name | Age. |
|---|---|
| Arya | 24 |
| Sachin | 25 |
| Giriraj | 26 |

Sample. 2

If we want to insert only name column from sample 1 to sample 2 then.

Code:

```
INSERT INTO sample 2 (name) SELECT name FROM Sample 1 ;
```

O/P.

Sample 2 will look like : ⟶

| Name | Age |
|---|---|
| Arya | – |
| Sachin | – |
| Giriraj | – |
| Abhishek | 21 |
| Rahul | 22 |
| Sunil. | 23 |

(Age में NULL value रहेगा)

## iii) Copying Specific Rows from a Table.

Example we have two tables Sample 1 and Sample 2. be like.

Sample 1

| Name | Age. |
|---|---|
| Riya | 21 |
| Priya | 21 |
| Priyanka | 20 |
| Sweta | 20 |
| Kriti | 21 |
| Puja | 21 |
| Duja | 19 |
| Chinky | 21 |

Sample 2.

| | |
|---|---|
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |

If we want to select only those name whose age is 21 from Sample 1 then.

code:

```
SELECT * FROM sample 1 WHERE age = 21;
```

if we want to copy this data ↑ in Sample 2 then.

code:

```
INSERT INTO Sample 2 (SELECT * FROM sample 1 WHERE age = 21)
```

O/P. Sample 2.

| Name | Age. |
|---|---|
| Riya | 21 |
| Priya | 21 |
| Kriti | 21 |
| Puja | 21 |
| chinky | 21 |

# 03. SQL SELECT statement.



**SELECT:** select is a DDL (Data Definition Language) command.

* No Data Manipulation Happens. Whatever we will perform using the select statement will not affect the actual Database.

* select Query is used to retrive data (records) from the tabl

* Basic Syntax.

```
SELECT column1, column2, ------
FROM table_name ;
```

What we will see in this Lecture:

Method : 1.

1. Displaying Data From Single Table.
   - i) Display all records present in the table using Select.
   - ii) Display only specific columns using select.
   - iii) Retrieve records based on some specified condit on

We will see all this one by one.

Let's suppose our Table.

Table name: emp.

| emp id | emp name | emp add | emp Salary | emp Phone | emp grade |
|---|---|---|---|---|---|
| 1 | a | j | 10 | 629090 | Excellent |
| 2 | b | k | 20 | 629091 | Good |
| 3 | c | l | 30 | 629092 | Excellent |
| 4 | d | m | 40 | £29093 | Excellent |
| 5 | e | n | 50 | 629094 | Good |
| 6 | f | o | 60 | 929095 | Excellent |

(i) i) Disply all Records present in the table.

Code:

```
Select * from emp ;
```

It will give us all the data present in the table emp.

Note: Whenever we use '*' we are not allowed to use any other operation or any other column name or function.

(ii) ii) Display only specific columns using SELECT.

Code:

```
Select emp name, emp salary, emp phone no. from emp;
```

If we want to select only some specific columns using select from emp table (Ex. emp name, emp salary, emp phone

(iii) iii) Retriere Records based on some specific condition.

we will use "WHERE" class.

~~sode~~ I want to details of all the employ'es whose grade is excellent then,

Code:

```
SELECT * FROM emp WHERE empgrade = Excellent ;
```

We will get all the rows from emp. table whose emp grade are excellent.

Method : 2. (2) performing some calculations on the column using SELECT.

* we can do simple Airthmetic operations (+, −, *, /) upon column data while displaying records.

Example: we need to find annual salary from the emp table. provided that annual salary table is not ~~present~~ exist in the table.

Code:

```
Select empsalary from emp;
```

// This will fetch us to the salary column.

⇓

we can multiply it by 12 and it will give us the annual sattary.

```
Select empid, empname, empsalary, empsalary *12 as anual salary
empgrade FROM emp;
```

This will create a another table named anual salary and is multiplied by 12 with empsal

if we dont mention empsal *12 as anualsal. Then Column name will be named as empsal *12 only.

O/P.

Emp.

| emp id | emp name | emp salary | Annual salary | Emp grade |
|---|---|---|---|---|
| | | | | |

## (3) Retriere Fields from multiple tables.

(data)

Let's suppose, we have two tables one named "emp" and other name "Dept".

emp.

| a | b | c | d |
|---|---|---|---|
| 1 | 4 | 7 | 10 |
| 2 | 5 | 8 | 11 |
| 3 | 6 | 9 | 12 |

dept

| e | f | g | h |
|---|---|---|---|
| 13 | 15 | 17 | 19 |
| 14 | 16 | 18 | 20 |

i) If we want to display both the table.

Code:

```
select * from emp;
Select * From dept;
```

ii) If we want to Select Some specific ~~table~~ columns from both the tables.

Code:

```
select a, b, c, g, h from emp, dept;
```

↳ It will give us all the possible combination of both the tables. from emp and dept.

Date: 15/09/2021

# 04. SQL INSERT ALL Statement.

Abhishek Sharma Reddy

It is the extention of Insert statement.

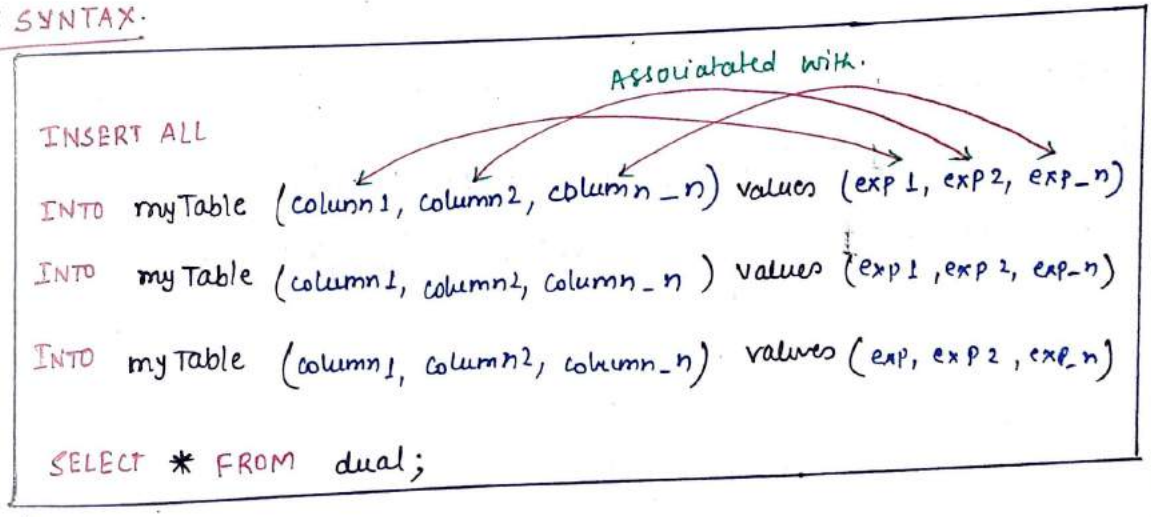
**Insert All:** Insert ALL statement is used to add multiple rows with a single Insert statement.

## * Limitations of insert statement:

At a time we can insert only a particular row.

* But with the help of insert All statement we can add multiple rows.
* By the help of insert all statement we can add n rows once a time.

## * SYNTAX.




↳ **Explanation:** we can add multiple rows in once. Here we are adding only 3 rows for example.

## * Approach - 1.

## * Insert into one Table.

INSERT INTO statement to insert multiple records into one table.

**Example.** Let's Assume we have a table called Sample 1 and it is empty.

* Now we will add 3 rows in this table Sample 1.

first we will see the normal insert method (one row by one) and then we will see how we can insert multiple row at once.

Abhishek Sharma Notes.

```
insert into sample 1 (id, name) values (3, 'Mahi');
```

↳ one row inserted.

```
insert into sample 1 (id, name) values (4, 'Abhi');
```

↳ 2nd row inserted.

```
insert into sample 1 (id, name) values (5, 'Sonu');
```

↳ 3rd row inserted.

Now we will add all this 3 rows at once. via insert all statement.

```
insert ALL
into sample 1 (id, name) values (4, 'Sanjay')
into sample 1 (id, name) values (5, 'Prasad')
int sample 1 (id, name) values (6, 'Srivir')
Select * from dual;
```

## * Approach - 2.

## INSERT INTO MULTIPLE TABLE.

* INSERT ALL can insert multiple rows into more than one table in one command.

Ex. we have 2 tables one is Sample 1 and other is SAmple 2 we want to insert at a same time 2 rows in sample 1 and 2 rows in sample 2.

```
insert all
into sample 1 (id, name) values (4, 'Abhi')
into sample 1 (id, name) values (5, 'Ravi')
into sample 2 (id, name) values (6, 'Ram')
into sample 2 (id, name) values (7, 'Sari')
Select * from dual;
```

inserted Sample 1.

inserted in Sample 2.

# 05. SQL UPDATE STATEMENT.

Abhishek Sharma Note

* **UPDATE:** It is a DML command. i.e. It is used to manipulate the existing data of table.

Ex. In real word you tryiy to change your profile pic, your password, your email etc.

## * UPDATE GENERAL SYNTAX.

```
UPDATE table_name
SET column 1 = value 1 , column 2 = value 2, -----
WHERE condition;
```

### (1) Approach : 1. Updating a single column.

Let's Assume a Table first. called table name "emp".

| ID | name | Job | Date | Salary | comm |
|---|---|---|---|---|---|
| 1 | A | Eng. | 07-08-2021 | 5000 | — |
| 2 | B | ER. | 08-09-2021 | 2000 | — |
| 3 | C | DR, | 08-01-2020 | 3000 | — |
| 4 | D | ER. | 05-06-2024 | 1500 | — |
| 5 | E | ER. | 06-07-2024 | 1000 | — |

Update the comm to 500 Whose Salary is less than 3000.

Code:

```
Update emp
set comm = 500
Where salary < 3000 ;
```

O/P

id. 2, 4, 5 के comm वाले column में 500 आ गया।

It will be Reflected.

* We can specify multiple condition also (not only one condition)

Ex. We want to update the salary to 6000 and comm is 5000 and whose job is ER.

Code:

```
Update emp
set salary = 6000, comm = 5000
where job = 'ER';
```

* `Select * from emp` // This will give you our output.

3 * * Updating without where clause.

If we don't specify condition then all the column data will be updated. ↓ Particular.

Example: Make the salary of all employee to 9000.

Code:

```
update emp
set salary = 9000;
```

// All the rows have been updated with salary 9000.

4 * * update table with data from another table.

Here we fetch the data from another table and then we are updating.

Example: Let's suppose we have two tables. like table 1 and table 2.

Ex: update the ~~salary~~ job name of employee from "emp" table whose location is in another table "dept" is chicago.

We have 2 table Here → i) emp ii) Dept.

We saw emp table previously.

Here we will see "Dept" table.

Dept:

| id | Dept.job | Location |
|---|---|---|
| 1 | ER. | chicago |
| 2 | ER. | chicago |
| 3 | DR. | TOKYO |
| 4 | ER. | chicago. |

Code:

```
update emp
Set job = (select Location from dept
           where rownum = 1);
```

This will take Location name's 1st row name i.e. chicago from Dept table and update the job column for all the rows to chicago. in emp table.

Means: Job=(ER. DR. ER.) from emp table will now change to chicago.

Date: 16/09/2021

06. SQL BETWEEN Statement

Ph. 6290903490
Abhishek Sharma
By GFG, Complete placement

## BETWEEN in SQL.

* Delete the data in b/w a set of values (range of values)
* It can be used in a SELECT, INSERT, UPDATE or DELETE Statement.

### * (1) Using BETWEEN condition with NUMERIC values

Example: We have a table called "emp" and we want All Employee details whose salaries between 2000 to 3000.

NOTE:- Between will include the values i.e. 2000 and 3000 will also be included in our result.

Code:

```
select * from emp
WHERE Sal BETWEEN 2000 AND 3000;
```

It will give you the data of employee whose salary is between 2000 to 3000.

If we write.

```
WHERE sal BETWEEN 3000 And 2000;
```

Then no data will present (Provided that 2000 Above 3000 no. employee).

But in program it will run like

```
WHERE SAL => 2000 AND SAL <= 3000
```

(condition satisfied) ✓

wrong →

```
WHERE SAL => 3000 AND SAL <= 2000
```

✗✗

(2) * If we want to select Data between the data.

Emp:

| ID | Name | Hire date | Sal | job |
|---|---|---|---|---|
| – | – | – | – | – |

Code:

```
SELECT * FROM emp
WHERE HIREDATE BETWEEN TO_DATE ('2000/08/07', 'YYYY/MM/DD')
And TO_DATE ('2001/08/06', 'YYYY/MM/DD');
```

(3) * NOT Operator with the BETWEEN condition.

Example. All employee details whose salaries not in the range of 2000 to 3000.

Code:

```
SELECT * FROM Emp
WHERE SAL NOT BETWEEN 2000 AND 3000;
```

(4) * Like this we can also perform Insert operation as well.

Example: Assuming we have a table called Sample and we want insert data from Emp table to sample table those employee whose salary (sal) is 2000 to 3000.

Code:

```
INSERT INTO SAMPLE (SELECT * FROM EMP WHERE SAL BETWEEN 2000 AND 3000);
```

Provided that SAMPLE Table also have the same Column that in EMP table.

(5) We can also Delete Data from Table using BETWEEN.

Code:

```
DELETE from Emp
WHERE SAL NOT BETWEEN 1000 and 3000;
```

→ Delete the Data whose sal. is not b/w 1000 and 3000

(6) We can also ~~use~~ UPDATE the table using BETWEEN STATEMENT.

Ph. 6 29 0 90 34 90
Abhishek Sharma No Tes.

# 07. SQL DELETE statement.

## DELETE in DBMS.

Ex.

```
DELETE FROM EMP
WHERE CONDITION
```

Condition is optional if you don't give condition all the data will be deleted. (Whole table).

### (1) * Simple Delete.

Ex. We want to delete a row where name = Abhishek from emp table.

code:

```
~~Select~~ * from emp
Delete
WHERE name = 'Abhishek';
```

→ TOTAL ROW (complete Row) will be deleted.

### (2) * Specify Multiple Condition.

Ex. Assume these are Multiple Deparatment no. with same id.

Ex.

| Dept. No. |
|---|
| 10 |
| 10 |
| 20 |
| 10 |

We want to Delete all the rows who's Departmnt is 10. and job = Manager.

code:

~~Select~~ * ~~from~~ emp

```
Delete From emp
WHERE dept NO. = 10 and job = "Manag
```

if we want to both the conditions satisfied then we will write (AND) and if only one either of both will be condition satisfied we will write (OR).

~~bet~~ b/w the condition

(3) * Delete complete Data (All rows).

Code:
```
delete from emp;
```

* ↳ if we don't specify the condition all the rows will be deleted.

* we can't delete column via this delete fun$^{n}$. Only rows will be deleted.

## * Truncate command.

Truncate command is to use delete all the data.

Code:
```
truncate from emp;
```

### Difference b/w Delete and Truncate:

* If we use Delete command to delete the data it can be rolled back (data). If can be Recycleed.

* But If we use Truncate command to delete the data it can't be rolled back again (data).

* In delete, we can select some particular row and we can delete them.
But in Truncate we have to delete all the data.

This is the difference b/w Delete and Truncate command.

Ph. 6290903490.
Abhishek Sharma Notes.

# 08. SQL ALTER (ADD, DROP) command.

ALTER (ADD, DROP) : We can't use only ALTER command. We will use ALTER command with the combination of ADD ~~and~~ or DROP command.

Purpose: Purpose of this Alter command is to make changes at the column level.

(column level changes are not allowed in the previous all commands). Here via this Alter (Add, drop) command we can do column level changes.

We can do:
i) add columns
ii) delete columns
iii) Drop columns.
iv) Rename columns.
v) Changing datatype of particular column.
vi) " " size " " "

---

## (1) * ALTER - ADD.

used to add one or more new columns to the existing database tables.

Code:
```
ALTER TABLE table_name ADD (column_name datatype);
```

Directly we can't use Alter or Add we need to use it both (combination of Alter with add).

Ph. 6290903490.
Abhishek Sharma Notes.

## Dimensions of Alter - ADD.

i) Adding a single columns

ii) Adding a multiple columns.

iii) Adding Constraints. (constraints are conditions that are imposed like foreign Key, primary Key) .etc.

### (1) i) Adding a single column.

Let's suppose we have a table called "emp". like this

| Emp no. | EName | Job | MGR | SAL |
|---|---|---|---|---|
| | | | | |
| | | | | |
| | | | | |
| | | | | |

if we want to add one more column (Address). ↑ ex. in this table

Code:

```
ALTER TABLE EMP ADD ( ADDRESS VARCHAR2 (15)) ;
```

↑ Keyword ↑ Table name ↑ Key word ↑ Column name ↑ Datatype ↑ size.

O/P.

| Emp no. | EName | Job | MGR | SAL | Addres |
|---|---|---|---|---|---|
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |

### (2) Adding Multiple columns.

if we want to add DOB, Mothername, Fathername, & Allowances columns in the above table "EMP".

Code:

columns could be added

```
ALTER TABLE EMP ADD (DOB DATE , Mothername VARCHAR2 (12) ,
                     Fathername VARCHAR 2 (12) ,
                     Allowances NUMBER (4) ) ;
```

(size)

Abhishek Sharma Notes
Ph: 6290903490.

## (3) Add constraints

* We want to make a column as a primary key.

Code:

```
ALTER TABLE Emp
ADD PRIMARY KEY (Emp NO);
```

* If we want to set default all the rows of the column.

Ex. We want to set M (Male) in every row of column "Gender." of the table "emp".

Code:

```
ALTER Table EMP
ADD (gender char (1) DEFAULT 'M');
```

---

## (2) ALTER - DROP

* used to Delete one or more existing columns present in the table.

* we can also drop more than one column by seperating the column names with commas.

Code:

```
ALTER TABLE table_name DROP (column_name);
```

↳ Same functions similar to Alter Add command

Abhishek Sharma Notes.
Ph. 6290903490.

## Dimensions of Alter -DROP.

i) Droping a Sinyle column.

ii) Dropping multiple columns.

3) Dropping constraints.

---

### i) Dropping a Sinyle column.

Ex. Let's Assume we have to ~~delete~~ drop the address column.

Code:

```
alter Table Emp
DROP (Address);
```

// Address column is deleted now.

### ii) Dropping Multiple column.

Ex. Let's Assume we want to delete the columns of mother -name, father name and allowances.

Code:

```
Alter Table emp.
DROP (Mothername, Fathername, Allowances);
```

// Above mentainoned 3 columns have been deleted now.

if we want to crosscheck then Code: `SELECT * FROM EMP;`

### iii) DROPPING constraints.

Ex. If our constraint name is Primary ID then we will delete this via this method:

Code:

```
ALTER TABLE EMP
DROP CONSTRAINT PRIMARY_ID;
```

// o/p. constraints is deleted.

# 09. SQL ALTER (Rename, Modify Statement)

**ALTER- MODIFY:** We can change the datatype of existing column or the size of the data type of the existing column.

SYNTAX.

```
ALTER TABLE table_name MODIFY (column_name datatype);
```

## * Dimensions of Alter modify

i) Modifying a Single column.

ii) Modifying a multiple columns.

### i) Modifying a Single column.

Let's Assume we have the table called Emp and there is a column called Address But the Address size is 15 only. (i.e. we can insert only 15 characters to the address column).

* Now we want to change it to 75 characters.

Code:

```
ALTER TABLE EMP MODIFY (ADDRESS. VARCHAR2 (75));
```

### ii) Modifying Multiple columns.

Let's Assume we have 2 columns named Gender and Emp name to our table called "Emp" we want to change both datatype size. we can do it.

Abhishek Sharma NOTES.
ph. 6290903490.

code:

```
ALTER TABLE EMP
MODIFY (gender Varchar2 (2), Empname Varchar2(10));
```

if we want to change the datatype.

ex. if we want to change DOB datatype → (date to integer)
DOB ↑ Column Name; date ↑ Datatype; integer ↑ Datatype

then.

code:

```
ALTER TABLE EMP
MODIFY (DOB INT);
```

// previously it was date datatype
Now it is converted to int

→ // we can change it for only rows are empty all for this column.

* using this ALTER - MODIFY we can change the column size to any length and we can decrease it as well.

---

## * ALTER - RENAME

used to rename the existing column name of a table.

* NOTE: We can't rename multiple column name at once.
At a time we can do only one column name rename.

Syntax:

```
ALTER TABLE table_name RENAME Column
old-column_name TO new_column_name ;
```

Ex: If we want to change ~~Address~~ the column name from Address to Location in the given table Emp then.

code:

```
ALTER TABLE EMP RENAME COLUMN ADDRESS TO LOCATION;
```

→ Now column name has been modified ~~to~~ to from ADDRess to Location.

Abhishek Sharma NOTES.
Ph: 6290903490

# 10. SQL LIKE CLAUSE

**LIKE OPERATOR:** Like operator is used to search specified pattern in the data and retrieve the record when there is a pattern match as required.

Ex. यदि हमको A नाम से जो जो Person है उनको find करना है। या Z नाम से किसी person को खोजना है। या फिर किसी भी Pattern के हिसाब से search करना पड़े।

## GENERAL PATTERN.

| PATTERN | MEANING. |
|---|---|
| 'r%.' | Matches strings which start with 'r' |
| '%r' | Matches strings which end with 'r' |
| 'r%t' | Matches strings which contain the start with 'r' and end with 't'. |
| '%tri%' | Matches strings which contain the substring 'tri' in them at any position |
| '_tri%.' | Matches strings which contain the substring 'tri' in them at the 2nd position. |
| '_r%.' | Matches strings which contain 'r' at the 2nd position. |
| 'r__%.' ↑ 2 underscore है। | Matches strings which start with 'r' and contain at least 2 more characters. |

and so on.

Example of Like clause:

Abhishek Sharma NOTES
Ph: 6290903490.

Q) i) Display the employee whose name start with 'M'.

Code:

```
Select name from emp
Where name like 'M%' ;
```

// Here emp = table name.
name = column name.

↳ It will give names of all employees whose name starts with 'M'

Q) ii) Display the employee whose name ends with 'A'.

Code:

```
select name from emp
Where name like '%A' ;
```

Q) iii) Display the names of all employees having M in any position in their name.

Code:

```
Select name from emp
Where name like '%M%' ;
```

Q) iv) If we have a column named Hiredate and in it month of the Hiring is given.

We want to Display names and Hiredates for the employee joined in the Month December.

Code:

```
Select name, Hiredate From Emp
Where Hiredate Like '%DEC%' ;
```

Month वाली (MMM) के रूप में Represent होना है

3 word. like February = FEB
January = JAN.

Abhishek Sharma Noted
Ph: 6290903490

Q) v) Display names of all employee whose name contains exactly 4 letters.

Code:

```
SELECT Name from Emp
where name like '____';
```

We will give Here 4 underscore for 4 letter for 5 letter 5 underscore and so on.

Q) vi) Display names having 2nd letter with N.

Code:

```
SELECT NAME FROM EMP
WHERE NAME like '_N%';
```

One underscore means 1st letter will be Anything.

if we will give 2 underscore then 2 letters can be anything. and so on.

Q) viii) Display all the names starting with J and ending with s.

code:

```
SELECT NAME FROM EMP
WHERE NAME LIKE 'J%S';
```

Q) viii) Display those only names ~~those~~ those are free from Letter L.

Keep in mind.

code:

```
SELECT NAME FROM EMP
WHERE NAME NOT LIKE '%L%';
```

NOT LIKE

# 11. SQL CREATE STATEMENT

Abhishek Sharma Notes.
ph: 6290903490.

**CREATE:** create command is used to create a table in the database with the structure specified by user.

* This structure includes the number of columns to be present in the table and the data type of the column, size of data etc.

Basic Syntax

```
CREATE TABLE table_name
(
 column1 datatype (size),
 column2 datatype (size),
 column3 datatype (size),
 -----
 columnN datatype (size),
PRIMARY KEY (one or more columns)
);
```

// Primary key ek bhi ho sakte hai aur nahi bhi- Depends upon user.

Example:

```
CREATE TABLE employee (
emp_no      NUMBER (4,0),
ename       Varchar2 (10),
job         varchar2 (9),
hiredate    date,
sal         number (7,2),
dept no.    number (2,0),
comm        number (7,2)
);
```

How many Number
Decimal presision.

will give us Decimal Precission.

At the end don't do any comma.

O/P.

Employee.

| empno. | ename | job | hiredate | sal | deptno | comm |
|---|---|---|---|---|---|---|
| | | | | | | |

Abhishek Sharma Notes
Ph. 6290903490.

## * DESC COMMAND.

It is used to know the structure that we have created for our table.

Generally it will Display

i) column names
ii) Datatype of column names
iii) size of data
iv) any constraints imposed the table.
v) default values for each column.
vi) whether NULL value allowed or not for each column

etc.

## * Creating a new table from an existing table.

### 1) By copying all columns from another table.

code:

```
create table emp1 as ( select * from emp) ;
```

Here we are coppied all the columns from emp table to emp1 table.

### 2) By copying selected columns from another table.

code:

```
create table emp2 as (select empNo, ename, sal
                      from emp);
```

Here we are copying selected columns (i.e. empNo, ename, sal) from emp table to emp2 table. (i.e. newly created).

Abhishek Sharma NOIDA
ph. 6290903490

3) By copying selected rows from another Table.

code:
```
create table emp3 as (select * from emp where deptNO = 10);
```

This statement 1st fetch the record from emp table where dept NO. column is 10. value.

This ↑ This will be coppied in emp 3 table.

4) copying selected columns from Multiple tables.

code:
```
create table emp4 as
(select ename, job, sal, loc, dname from emp, dept);
```

Here ename, job, sal ⇓ from table emp. loc, dname ⇓ from table dept.

we are selecting ↓ Multiple tables i.e. Here we are columns from

Selecting (ename, job, sal) ⇒ from emp table and

Selecting (loc, dname) ⇒ from dept. table.

column ~~from~~ and coping them to a new table called emp 4.

---

FAQ

Q) How can I create an Oracle table from another table without copying any values from the old table?

Ans: Here we need to create a table from another table such that only structure should be copied no data will copy. Table should be Empty.

Abhishek sharma Notes.
Ph. 6290903490.

if we write this code

```
create table emp5
as (select * from emp);
```

XX wrong way.

If we use this code then structure as well as data will be coppied from emp table to emp5 table.

Or we will write any false condition using where.

Ex.

```
create table emp5
as (select * from emp WHERE 1=2);
```

This condition fails.

and only structur will be copied from emp to emp5 table. No data will be copied.

Now.

* we have to write any false condition. for this

## 12. SQL AND - OR Statement.

If we have more than one statement present then we will use AND - OR Statement for precise filtration of data from the database tables. along with select, update and delete Queries.

AND Clause:-

AND CLAUSE

The AND results true only when all the conditions specified are true.

Abhishek Sharma Notes
Ph. 6290903490.

Syntax:

```
SELECT  column 1, column 2 . . . . .
FROM    table_name
WHERE   condition 1  AND  condition 2  AND  condition 3 . .
. . . ;
```

Example: Display the records of those employees who are working as a manager and getting salary greater than 2500/-

Ans if the employee is not a mayer or if his salary is not greater than 2500. Then no data will be displayed. Both the condition should be True for displaying the data from the given table.

```
SELECT  JOB, EName, Sal
FROM    EMP
WHERE   JOB = 'MANAGER'  AND  SAL > 2500 ;
```

## OR CLAUSE

Defn: Among multiple conditions specified in the WHERE clause the transaction is performed if any of the condition becomes true.

Syntax:

```
SELECT  column 1, column 2 . . . .
FROM    table - name
WHERE   condition 1  OR  condition 2  OR  condition 3 . . . ;
```

Abhishek Sharma Notes
Ph 6290903490

Ex. Display details of all analyst and managers.

Sol:

```
SELECT JOB, EName, SAL
FROM EMP
WHERE JOB = 'MANAGER' OR JOB = 'ANALYST';
```

All those who are Manager or Analyst should be represented by the database. (displayed)

* अगर कोई condition false होगी तो data Display नहीं होगा। अगर कोई भी condition True होगी तो Display होगा OR condition में।

---

## Combining AND and OR.

Q) Display the details of employee having a salary greater than 1500/- or if he is a manager which is exclusively from 10th ~~dept~~ dept and 30th Departments.

Ans:

```
SELECT * FROM EMP
WHERE (SAL > 1500 OR JOB = 'MANAGER') AND
      (DEPTNO = 10 OR DEPTNO = 30);
```

# 13. SQL DISTINCT clause

Abhishek Sharma Notes-
Ph. 6290903490.

Distinct statement is used to return only unique values present in a column or combination of columns.

```
SELECT DISTINCT column1, column2,
FROM table name ;
```

## * Distinct on a single column.

Let's suppose in our table emp one column is for job and in there 4 Mangers, 3, analyst, 2 clerk are there. we need to find only distint value in them (like only 1 manager, 1 analyst, 1 clerk will be displayed).

code:

```
SELECT DISTINCT job from emp ;
```

Distinct is ~~only~~ the command for only display purpose. It will not affect the data of our table.

## * Distinct on more than one column.

If one of the value will be unique it will be displayed.

Ex: If we want to display job and salary of our employee then. if one of the value will unique then it will be unique and will be displayed.

code:

```
SELECT DISTINCT JOB, SAL FROM EMP;
```

Abhishek Sharma Notes
Ph 6290903490

* If we want to know how many times a given thing is repeated.

Code:

```
SELECT job, count (*) from emp
group by job
having count (*) > 1 ;
```

↑ we are displaying values who are greater than 1.

O/P:

| JOB | count (*) | |
|---|---|---|
| Analyst | 2 | → (Analyst 2 times Repeated) |
| Manager | 3 | → (Manager 3 times Repeated). |

Q) Does the Distinct clause consider NULL to be a unique value in SQL?

Ans: Yes. (NULL values are also values).

if there are 4 NULL values then Distinct clause will show how many NULL values are there.

## 14. SQL IN CLAUSE

* The SQL IN condition allows you to easily test if an expression matches any value in a list of values.

* It is help to reduce the need for Multiple OR condition in a SELECT, INSERT, UPDATE, or DELETE Statement.

* Also used for manipulation of Data.

Abhishek Sharma Notes
Ph: 6290903490.

* i) using IN condition with character values.

Let's suppose in name we give 3 conditions if the name is Abhishek, sunil, or shubhankar (if any one name is found then data will be display).

code: we can use OR operation for doing this

```
SELECT * FROM EMP
WHERE NAME= 'Abhishek' OR NAME = 'SUNIL' OR NAME =
'SHUBHANKAR' ;
```

if any one of the condition is satisfy data will be displayed.

we can do it in simple by using In condition.

Code:

```
SELECT * FROM EMP
WHERE NAME IN ('Abhishek', 'SUNIL', 'SHUBHANKAR');
```

* ii) using IN condition with Numeric value.

Same as character value we can use In in the numeric value also. Let's suppose we want to display that data whose ID No. is 50, 52, and 59. If they are present then display otherwise ignore.

Code:

```
SELECT * FROM EMP
WHERE IDNo. IN (50, 52, 59);
```

Abhishek Sharma NO tes.
Ph. 6290903490.

iii) * Using IN condition with NOT Operator.

If we want to exclude some records then we can use this Statement.

Code:

```
SELECT * FROM EMP
WHERE ID NO. NOT IN (50, 52, 59);
```

→ After this statement all the values expect 50, 52, and 59 will be displayed.

iv) * performing DML operations using IN.

* If we want to delete some rows.

Code:

```
DELETE FROM EMP
WHERE NAME IN ('Abhishek', 'SUNIL');
```

* Those Rows who contain 'Abhishek and SUNIL' Delete the data of that ROW.

~~Hint~~

Same we can use IN in INSERT, update and select statements as well.

v) * using Sub Query as an Expression inside IN.

Example we have in salaries column there are many salaryies we need to find max. salary out of them. then we need to find 2nd max. salary out of them.

Abhishek Sharma Notes.
Ph. 6290903490.

## For finding max salary.

```
SELECT max (sal) FROM EMP;
```

// giving us max salary. from column named sal and table Emp.

## For finding 2nd Max salary.

* We will create a result set that is free from 1st highest salary. we will exclude 1st max salary then we can find again max salary out of them then we get the 2nd Max. Salary.

FAQ: (in interview)

Code:

```
SELECT SAL FROM EMP
WHERE SAL NOT IN (select Max (sal) from emp);
```

→ This will give us salary excluding the max sal.

## for finding 2nd Max sal.

Code:

```
SELECT max (sal) From Emp
WHERE SAL NOT IN (select Max (sal) from emp);
```

→ It will give us Salary (2nd Max).

# 15. Aggregate Functions and Group By Statement.

## Aggregate Functions.

- SUM
- MAX
- MIN
- COUNT
- AVG.

Students.

| NAME | Subject | Marks |
|---|---|---|
| ABC | DBMS | 80 |
| ABC | OS | 70 |
| PQR | OS | 80 |
| PQR | DBMS | 70 |
| XYZ | DBMS | 70 |
| BCD | OS | 100 |
| BCD | DBMS | 90 |

1) If we have to find maximum marks of Any student at any subject

code:

```
SELECT MAX (marks)
FROM student ;
```

O/P.

100 (Max Marks).

2) If we want to find minimum marks of any student at any sub.

code.

```
SELECT MIN (marks)
FROM student;
```

O/P.

70 (Minimum Marks).

3) If I want to count how many marks is there in the table.

code.

```
SELECT COUNT (marks)
FROM student;
```

O/P.

7 (Count How many marks).

4) If we want to find sum of total Marks.

Code.

```
SELECT SUM (Marks)
FROM student;
```

O/P.

560 (sum).

5) If we want to find Average Marks.

code:

```
SELECT AVG (marks)
FROM student;
```

O/P. 560/7 = Avg (Average).

Abhishek Sharma Notes.
Ph: 6290903490.

* If we have NULL value in the table student
↳ in marks column.

like:

Student1

| Name | Subject | Marks. |
|---|---|---|
| ABC | DBMS | 80 |
| ABC | OS | 70 |
| PQR | OS | NULL |
| XYZ | DBMS | 80 |

NULL value will be ignored by all of the Aggregate functions:
(Like, Sum, Max, Min, Count and Avg).

Ex. If we call Sum o/p. = 230
" " " Count o/p = 3
Avg o/p = 230/3.

* If we want to count NULL value also and do part in all operation then we will use astric (*).

Ex.

```
SELECT count (*)
From student1;
```

o/p. : 4

Ex.

## * GROUP BY.

We use Group By whenever we want to group rows accoudig to one or more attribute.

Ex. we want to find maximum marks in every subject from the above metioned table called "student".

Abhishek Sharma Notes.
ph. 6290.90.3490.

Student>

| Name | Subject | Marks. |
|---|---|---|
| ABC | DBMS | 80 |
| ABC | OS | 70 |
| PQR | OS | 80 |
| PQR | DBMS | 70 |
| XYZ | DBMS | NULL |
| BCD | OS | 100 |
| BCD | DBMS | 90. |
| XYZ | OS | 180 |

Q> 1> Find maximum Marks in every subject.

our expected o/p should be

DBMS = 90

OS = 100

Code:

```
SELECT Subject, MAX (marks)
FROM Students
GROUP BY subject ;
```

** 2. In Groupby, either Write Attributes which are part of Grop By or you can write Aggregate function.

Here we are grouping the row of the table according to the maximum marks in every subject.

* Now if we want to select only certain rows of the grouped data. (we can use Having).

Ex.

Q> 2> Find Sum of marks of the students having sum more than 150.

Ex. ABC = 80+70 = 150 ✗ (NOT More than 150)

PQR = 80+70 = 150 ✗ " " "

XYZ = 180 = 180 ✓ (More than 150)

BCD = 100+90 = 190 ✓ " " "

Abhishek sharma Notes.
Ph. 6290903490.

code:

```
SELECT Name, Sum(marks) AS msum
FROM student
GROUP BY Name
Having sum(marks) > 150;
```

O/P.

| Name | msum |
|---|---|
| XYZ | 180 |
| BCD | 190 |

## * Difference b/w WHERE and HAVING CLAUSE.

| | Having | Where. |
|---|---|---|
| purpose | Filters aggregate rows<br>Ex. SELECT Name, Max(marks)<br>FROM student<br>Group by Name<br>Having Name = 'PQR'<br>OR Name = 'BCD' | Filters unaggregate rows<br>Ex. SELECT Name, Max(marks)<br>from student<br>WHERE Name = 'PQR'<br>OR NAME = 'BCD'<br>Group by name. |
| uses. | Can be used with Select and Group By only. | can be used with Select, Insert, update and Delete. |
| Aggregate functions (like sum, Max, Min, Avg, count). | Can be used | Cannot be used unless it is a subquery contained in a Having clause or a select list and the column is being aggregate in the outer reference. |

Abhishek Sharma Notes
ph 6290903490.

# 16. SQL GROUP BY clause.

Let's assuming we have a table of Emp.

Q) Display number of employees working in each Dept.

A) Code:

```
Select deptno, count(*)
from emp
group by deptno;
```

All Data will be segrated according to their Dept no.
if 4 rows of Dept no. 30 is present in Zig Zag manner then they can be displayed in one by one. (group)

O/P.

| Dept. No. | count (*) |
|---|---|
| 30 | 6 |
| 10 | 3 |
| 20 | 5 |

30 Dept no. वाले 6 rows हैं and so on.

Q) Display number of employees in each job.

Code:

```
select job, count(*)
from emp
group by job;
```

O/P.

| Job | Count(*) |
|---|---|
| Salesman | 4 |
| Clerk | 5 |
| Manager | 3. |

Abhijeet Sharma Notes
ph. 6290903490

Q) Display the Highest paid and lowest paid salaries in each Department.

Code.

```
Select max(sal), min(sal), deptno.
from emp
group by deptno.;
```

Data is segrated according to their Dept. no. (Max sal. and Min sal).

| Max (Sal) | Min (Sal) | Dept No. |
|---|---|---|
| 2850 | 950 | 30 |
| 5000 | 1300 | 10 |
| 3000 | 800 | 20. |

Lets suppose we Have 3 Dept. 10, 20, 30 each Dept. Sal (Max, Min) has been Displayed.

## Having clause.

Q) Display dept. which are having more than 3 employees.

Code:

```
Select deptno., count(*)
from emp
  group by deptno
  having count(*) > 3;
```

Here we can't use Where clause. bcz we ~~with~~ have to filter the group summaries. we can use Having clause.

* In General if we have to filter any row we will use where clause but when we have grouped data in which we have to filter then we use having clause.

Abhishek Sharma Notes.
ph. 6290903490.

Q.) Display the number of analyst working in the company.

Code:

```
Select job, count (*)
from emp
group by job
having job = 'ANALYST';
```

Q) Display departments whose total salary paid is more than 9000.

Ans: Code:

```
select sum (sal), deptno
from emp
group by deptno,
having Sum (sal) > 9000;
```

Q) Display the Hiring year and number of employees hired in that year is more than one.

Ans: Code:

```
Select to_char (hiredate, 'YYYY') "YEAR", count(*)
from emp
Group by to_char (hiredate, 'YYYY');
having to_char (hiredate, 'YYYY') = '1981';
// having count (*) > 1;
```

This will give us the data of employee who has hired in perticular year.

This will give us Only those employee who are joined in 1981 year.

This will give Hiring year is more than 1.

# 17. SQL ORDER BY CLAUSE.

Abhishek Sharma Notes.
ph. 6290903490.

order by clause is used to arrange the fetched data from the database table in ascending or descending order of data values based on one or more column.

Ex: * Store records of Employees in increasing order of salaries.
* Student details as per increasing order of marks or attendance.
* Employee details based on seniority level.

~~SORT. Sorting~~

## Sorting according to a single column.

Q) Display the names, salaries of all employee based on decreasing order of their Salaries.

Code:

```
select name, sal
from emp
order by sal desc;
```

desc = decreasing.

if we don't specify anything it will be in increasing order

Q) Display the Department number and employee name as per increasing order of department numbers.

code:

```
select name, sal, deptno.
from emp
order by deptno;
```

↳ if we didn't specify anything it will in increasing order. by default.

Abhishek Sharma Notes.
Ph. 6290903490.

Q) Display the adl the details of employees as per increasing Order of i) Department numbers ii) names iii) Hire date.

Ans. Code:

```
Select * from ~~xxxxx~~ emp
order by name;
```

This will give us data sorted by their name (increasing)

```
Select * from emp
order by hiredate;
```

This will give increasing order of their Hiredate.

## * Sorting According to More than One Column.

First data will be sorted according to 1st column (given) then internally it will be sorted according to 2nd column and so on.

Q) Display the details of employees based on increasing order of departments and in each department salary should be further arrange in Highest to lowest order.

Ans. Code:

```
Select * from emp
Order by deptno , sal;
```

Q) Display the names, salary, jobs of employees who are working as managers in Hight to lowest order.

Ans. Code:

```
Select name, sal, job
from emp WHERE Job = 'MANAGER'.
order by sal desc;
```

# 18. SQL WHERE Clause

Abhishek Sharma Notes.
ph. 8290903490

Where clause is used for data filtering.

(sometimes a user is interstig in filtering data and retriving only specific data that meet certain conditions and requirments, which can be done by where clause).

Syntax.

```
SELECT column1, column2,
FROM table name
WHERE [condition];
```

Examples.

Ex.1. Display the records of those employees whose designation is 'MANAGER'.

Ans

Code

```
Select * from emp
Where Job = 'MANAGER';
```

O/P. only manager details will be displayed.

Ex2. Display name, job and the salary of that employee whose salary is greater than 2900.

Code

```
Select name, salary, job
from emp
where Sal > 2900;
```

Abhishek Sharma Notes.
Ph. 6290903490

## * Operators that can be used in WHERE clause.

Basic operators can be used in where clause

*, ( >, <, <= , =>) and as so on.
!, !=

Q) Display the records of all employees other than mangers.

An:

Code:

```
Select name , sal, job
from emp
Where job ! = 'MANAGER' ;
```

Ex. of not equal to (!=) Operator.

Q) Display records of all employees whose salary is b/w 1000/- to 2000/-

An

Code:

```
Select name, sal, job
from emp
Where sal between 1000 and 2000 ;
```

Ex. of b/w operator.

(Always remember in b/w. always gives smaller value 1st then give larger value.

Ex. ( between 1000 and 2000)
↑ 1st given (smallest)
↑ 2nd given (largest)

Abhishek Sharma Notes
Ph. 6290903490.

* Deleting Records based on the condition specified in Where clause.

If the condition is satisfied the data will be deleted.

Ex.

```
delete from emp
Where sal = 2500 ;
```

O/p. That row Whose sal is 2500 gets Deleted. as the condition satisfied.

---

## 19. SQL DROP - TRUNCATE - RENAME Statement.

* i) DROP Command. // If we want to delete the table.

It delete the table existance completely.

Syntax.

```
DROP TABLE table_name ;
```

Ex.

```
Drop Table emp ;
```

// emp table will be deleted completely

---

* ii) FLASHBACK Command. // If we want to Restore from Recycle Bin.

It restores the ~~del~~ dropped table. (जैसे Recycle किया जाता है).
bin

Syntax.

```
flashback table table_name to before drop;
```

---

* iii) PURGE Command // If we want to delete completely from Recycle Bin also.

It will delete completely the table from Recycle Bin also.

Syntax.

```
drop Table table_name purge ;
```

Abhishek Sharma Notes.
Ph: 6290903490.

## iv) TRUNCATE COMMAND.

It will empty the table. i.e. all the table data will be deleted but the structure of table and database object is still alive. and the table can be reused normally.

Syntax.

```
TRUNCATE TABLE table_name;
```

Ex. `Truncate table emp.` ⇒ Data will be cleared but (table) database is alive now we can insert and reuse this table.

---

Difference b/w Delete and Truncate command.

| DELETE | TRUNCATE |
|---|---|
| i) Data can be rolled back. It is not permanently Deleted. | i) Data can't be rolled back. It is Permanently Deleted. But the table is still alive. |

---

## v) RENAME command.

It is used for renaming a table.

// For Renaming a Column we can use Alter Command But for renaming a table we can use RENAME Command.

Syntax.

```
rename old_table_name to new_table_name;
```

Ex.

```
rename emp to employeedata;
```

// After Renaming all the operations will be conducted through the new name.

# 20. Joins in SQL.

Abhishek sharma notes.
Ph. 6290903490.

If we want to join 2 tables then we use joins in SQL.

There are Multiple type of joins.

Joins in SQL.

- Cross Product
- Inner Join
  - Equi Join
    - Natural Join
- Outer join
  - Left Outer
  - Right outer
  - Full Outer
- Self join

---

## * Cross Join.

* Cross join is nothing But cross product of Both the table

Let's suppose we have two tables → employee and Department.

Employee

| Name | Dept ID. |
|---|---|
| ABC | 3 |
| XYZ | 1 |
| PQR | 1 |
| BCD | NULL |

Department.

| Dept-ID | Dept name. |
|---|---|
| 1 | Engineering |
| 2 | Sales |
| 3 | Marketing. |

Abhishek Sharma Notes
Ph. 6290903490.

## Cross join of Both the table

Code:

```
SELECT * FROM Employee, Department;

OR

SELECT * FROM Employee CROSS JOIN DEPARTMENT
```

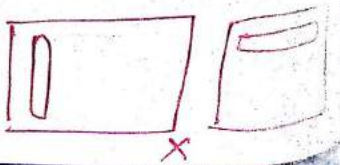
O/P.

Total ROWS will be 4 X 3 = 12.

| | Employee Name | Employee Dept-ID | Department Dept-ID | Department Dept-name |
|---|---|---|---|---|
| i) | ABC | 3 | 1 | Engineering |
| ii) | XYZ | 1 | 1 | Engineering |
| iii) | PQR | 1 | 1 | Engineering |
| iv) | BCD | NULL | 1 | Engineering. |
| v) | ABC | 3 | 2 | Sales |
| vi) | XYZ | 1 | 2 | Sales |
| vii) | PQR | 1 | 2 | Sales |
| viii) | BCD | NULL | 2 | sales |
| ix) | ABC | 3 | 3 | Marketing |
| x) | XYZ | 1 | 3 | Marketing |
| xi) | PQR | 1 | 3 | Marketing |
| xii) | BCD | NULL | 3 | Marketing. |

→ All the 4 rows of 1st ~~eat~~ Table will be multiplyed 4 time with the 1st row of 2nd table.

Like MATRIX MULTIPLICATION



Abhishek Sharma Notes.
ph 6290903490.

## * INNER Join.

There will be some condition specified. If the condition matches. Then only data will be displayed. Here we form cross product (join) first Then we Remove those rows which do not match the given condition.

Example.

```
SELECT Employee Name, Employee Dept ID, Department Dept-DD,
Department Dept_name FROM Employee INNER JOIN
Department ON Employee Dept_ID = Department Dept-ID.
```

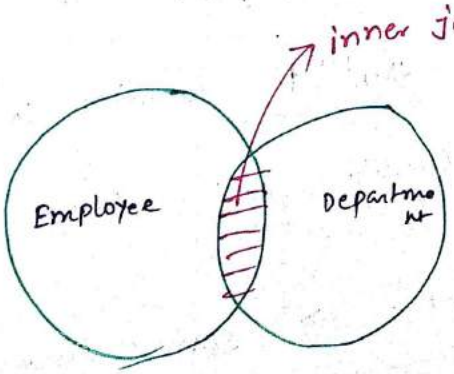
Condition.

O/P.

| ✓ | ✓ | ✓ | ✓ → from prev. Page. |
|---|---|---|---|
| ABC | 3 | 3 | Marketing |
| XYZ | 1 | 1 | Engineering |
| PQR | 1 | 1 | Engineering |

We can also understand INNER Join via set Theory.

Venn Diagram



Abhishek Sharma Notes.
Ph. 6290903490

## * Equi Join

It is a specia type of inner join. Here the constrains only is (=) operator. only Equal to operator ~~bdfff~~ we can used in equi join.

Ex:
↳ in Inner join we saw that example will also be valid for Equi join bcz. we give there (=) in the condition. ~~Wdd~~

## * NATURAL JOIN

It is a special case of Equi join. There Natural join is also a special case of INNER join also.

* Natural joins Happens in 2 tables when they have 1 or more than 1 same column names. for some attribute.

Ex: In our table employee and Department both have one common column name i.e. Dept_ID.

* If we do natural join then in the common column name it will apply ~~equal~~ equi join condition and returns the result.

Ex: In inner join Example. ⇒ same Example in Natural join Also.

Code:
```
Employee Name, Employee Dept ID,
Department. Dept-ID, Department Dept_name
FROM Employee NATURAL JOIN Department
```

O/P.

Abhishek Sharma Notes.
ph. 6290903490.

## * LEFT OUTER JOIN

All the rows of 1st table (Left table) come into result and those rows that match with the rows of the 2nd (Right) table have the matching entry of right table. If they do not match then their will be NULL NULL. entry.

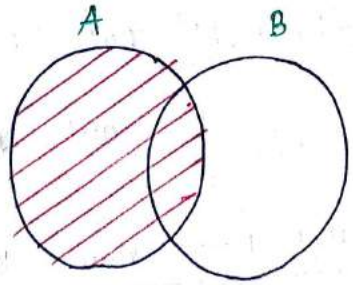
Code:

```
SELECT Employee Name, Employee Dept. ID , Department
Dept_ID , Department ,Dept_name FROM Employee
LEFT OUTER JOIN DEPARTMEN ON Employee.
Dept_ID = Department Dept ID.
```

O/P.

| ✓ | ✓ | ✓ | ✓ ⟹ Same as Previous |
|---|---|---|---|
| ABC | 3 | 3 | Marketing |
| XYZ | 1 | 1 | Engineering |
| PQR | 1 | 1 | Engineering |
| BCD | NULL | NULL | Engineering |

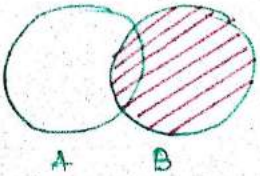
VEIN DIAGRAM :




## * Right OUTER JOIN

Same things opposite of Left outer join.

Vein Diagram



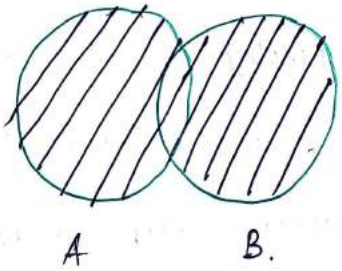

Hope समझ गए होंगे।

Abhishek sharma Notes.
Ph. 6290903490.

## FULL outer join

All the Attributes of table A and all the Attributes of table B that satisfies the condition.

Vein Diagram.




Example तो समझ ही गए होंगे


## SELF JOIN

Self join is a join where you perform join on the table itself. Here Left and Right Both tables are same. It can be inner join or can be outer join.

Ex: Print all Dept. ID's that have more than one Employees.

Employee.

| Name | Dept_ID |
|---|---|
| ABC | 3 |
| XYZ | 1 |
| PQR | 1 |
| BCD | NULL |

Abhishek Sharma Notes.
Ph. 6290903490.

we will join the Employee table with itself. We will use $E_1$ and $E_2$ for both the instances.

Code:

```
SELECT  E1 . Dept ID  FROM
Employee  As  E1  INNER  JOIN
Employee  As  E2  ON

    E1 . Dept ID  =  E2 . Dept ID  AND
    E1 . Name  <>  E2 . Name.
```

Different (not Equal)

## 21. SQL Subqueries.

Let's Assume we have a table called student and we have following columns and date in the table.

Student

| Name | Subject | Marks |
|---|---|---|
| ABC | OS | 70 |
| PQR | DBMS | 80 |
| ADC | DBMS | 90 |
| XYZ | OS | 80 |
| PQR | OS | 70 |
| XYZ | DBMS | 100. |

Abhishek Sharma Notes.
Ph. 6290903490.

① Q) Find name of the student who has maximum marks in any subject ?

Ans: If we write this code.

```
SELECT MAX (MARKS) FROM Student ;
```

O/P. (100.) Only marks no name

It will give me only ↓ marks. i.e. 100.
max

It will not give us name of the student.

So what we can do ?

Ans We can make the above Query as subQuery of another Query and from that we can find the name of the student also from our table Student.

Code:

```
SELECT Name, Marks FROM Student
WHERE Marks = (SELECT MAX (MARKS) FROM
                Student);
```

O/P.

| XYZ | 100 |
|---|---|

name as well as Marks. (max)

Here we used Above Query as ✓ SubQuery.

will be displayed.

Abhishek Sharma Notes.
ph. 6290903490.

Q.2) Print Marks of all students in every subject except the students that are suspended.

Ans We have 2 tables. One is called Student and other is suspended.

Student

| Name | Subject | Marks |
|---|---|---|
| ABC | OS | 70 |
| PQR | DBMS | 80 |
| ABC | DBMS | 90 |
| XYZ | OS | 80 |
| PQR | OS | 70 |
| XYZ | DBMS | 100 |

Suspended

| Name | Reason |
|---|---|
| PQR | Discipline |
| BCD | Attendance |

We want to print all the students marks but not those who are ~~ex~~. suspended. (we will exclude them).

Code:

```
SELECT * FROM Student
WHERE Name NOT IN
(SELECT Name FROM Suspended);
```

↳ we will use NOT in Here for excluding the suspended name. and its Marks.

O/P. Only PQR's Marks will not be displayed.

as it is in ~~&~~ Suspended. table

Abhishek sharma Notes.
ph. 6290903490.

## 22. SQL SUBQUERIES (SET-2):

* Now we will do Questions which was most asked in interviews from Sub Queries.

Q) Find Second Highest salary ?

Ans: We will use Here subQuery. In the inner Query we will find the Max salary and in the outer Query we will find the again max salary such that this sal. is smaller than the max salary. which is in the inner Query.

code:

```
SELECT MAX (salary) As Msal From
Employee WHere
Salary < (Select MAX (Salary) From employee);
```

Q.2) Find the name of the Employee having 2nd Highest salary ?

Ans We will use Here 3 Level of Sub Quiryies.

Previously we find the 2nd Highest salary NOW we will use that result as a subQuery. and find the name also.

Code:

```
SELECT Name FROM EMPLOYEE WHERE
SALARY = (SELECT MAX (salary) AS Msal From
Employee where Salary < (select MAX (salary) from
employee));
```

Q.3) Finding the nth Highest salary ?

Abhishek Sharma Notes.
ph. 6290903490.

Ans:- We will first sort the salary in decreasing order. After sorting we picked the TOP n rows to find the nth Highest salary (using TOP or LIMIT Query). We use this result as SubQuery. What every the result we get we further have the outer Query that finds the minimum of the results of the subQuery. Then we have more further Query that finds the employees whose salary is equal to this minimum salary.

Code :

```
SELECT * FROM Employee WHERE
SALARY = ( SELECT MIN (Salary) FROM
  Employee WHERE Salary IN ( SELECT
  DISTINT TOP 5 Salary From Employee
  ORDER BY DESC) );
```

↳ If we want to get 5th Highest salary.

Q.4) Find Details of all employee whose salary is greater than overall ~~avergs~~ average.

Sol code:

```
SELECT * FROM Employee WHERE
SALARY > ( SELECT AVG (Salary)
  From Employee);
```

Q.5) Find details of the employees whose salary is greater than average salaries of all departments.

Employee

| Name | Dept. | Salary. |
|---|---|---|
| ABC | Engineering | 80000 |
| PQR | Sales | 40000 |
| BCD | Sales | 30000 |
| RST | Engineering | 70000 |
| XYZ | Marketing | 50000 |
| WXY | Marketing | 40000 |

Logic:

* 'ALL' is used to represent all the values.
* 'Any' is used to represent any value in the subset.

if in the Ques. we find all the department then we will use "ALL" Keyword in our code and if we find any the department then we will use 'ANY' keyword in code.

code:

```
SELECT * FROM Employee WHERE
Salary > ALL (SELECT Avg (Salary)
             From Employee GROUP BY Dept);
```

# 23. Correlated SubQueries.

Abhishek Sharma Notes
ph. 6290903490.

* SubQuery uses value of the other Query.
* SubQuery is evaluated for each row of the order Query.

| Name | Dept_ID | Salary |
|---|---|---|
| ABC | 2 | 50000 |
| BCD | 2 | 30000 |
| CDE | 1 | 40000 |
| PQR | 3 | 80000 |
| LMN | 1 | 30000 |
| XYZ | 1 | 20000 |
| WXY | 3 | 70000 |

Q. Example: Find Name and Dept_ID of every employee whose salary is above average in the department.

Code:

```
SELECT Name, Dept_ID
FROM Employee E1 WHERE
SALARY > (SELECT AVG (Salary)
FROM Employee WHERE DEPT_ID = E1. Dept_ID);
```

(Renamed as E1)

O/p.

| Name | Dept. ID |
|---|---|
| ABC | 2 |
| CDE | 1 |
| PQR | 3 |

We will find avg. of the Salary and which is highest we will display the result.

Abhishek Bhaiya Notes
Ph 6290903490.

Ex: Find names of the departments that do not have any Employee.

| Name | Dept ID | Salary. |
|---|---|---|
| ABC | 2 | 50000 |
| BCD | 2 | 30000 |
| CDE | 1 | 40000 |
| PQR | 3 | 80000 |
| LMN | 1 | 30000 |
| XYZ | 1 | 20000 |
| WXY | 3 | 70000 |

Employee.

| Dept_ID | Name. |
|---|---|
| 1 | Engineering |
| 2 | Sales |
| 3 | Marketing |
| 4 | HR |
| 5 | Graphics. |

Department.

Code:

```
SELECT Name FROM
Department d WHERE
NOT Exist ( SELECT Dept_ID From
Employee.e WHERE e.Dept_ID =
d.dept_ID);
```

* We can use co-related SubQuerins in SELECT, UPDATE, DELETE clause and many other places.

Abhishek Sharma Notes.
Ph: 8290903490.

Ex. Print Employee names, their Salaries and their Department average salaries.

* We use only these Employee table.

Soln Code:

```
SELECT Name, salary,
(SELECT AVG (Salary) FROM
Employee e2 WHERE
e2.Dept_ID = e1.Dept_ID) As Average.
FROM Employee e1;
```

O/p.

| Name | Dept ID | Average. |
|---|---|---|
| ABC | 50000 | 40000 |
| BCD | 30000 | 40000 |
| ⋮ | ⋮ | ⋮ |
| All 7 rows. | ---- | ---- |

The end. of SQL.

Date: 18/09/2021.
10:30 PM.

From the Authors Hand:

Notes Written by Abhishek Sharma.

Thank you Guys for making the Group of 1000 members. Long more to go.

From GFG complete 100 Preparation course

Hope you guys loved it ☺

* If you read the notes and completed it Then go in the Group code warriors and write a Review about my notes.

* Happy coding ☺ Abhishek Sharma.