SQL (STRUCTURED QUERY LANGUAGE)

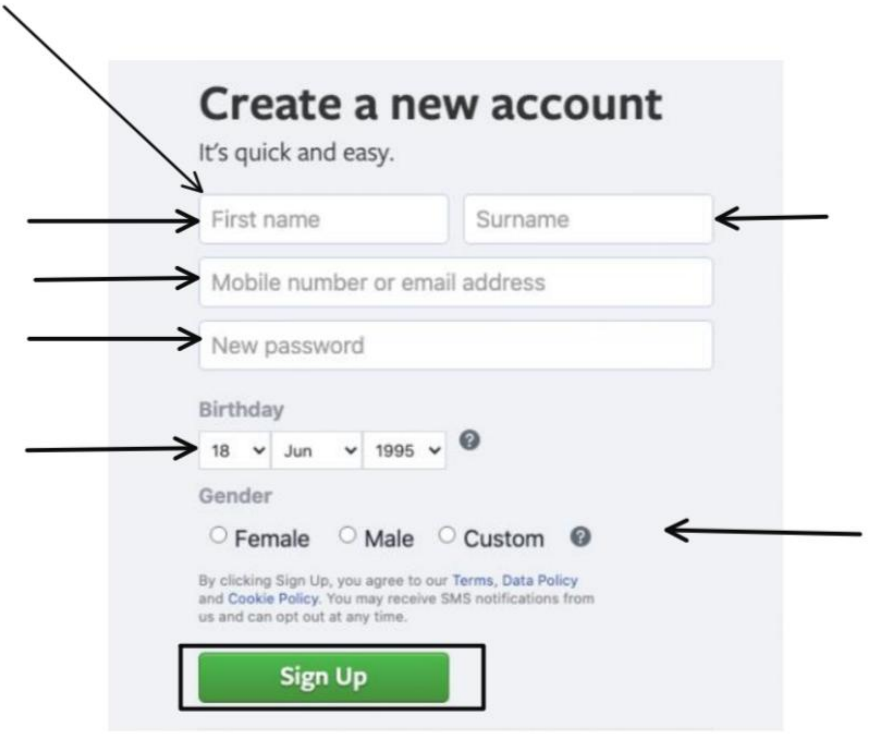
↓

DATABASE

What is DATA ?

"Data is a raw-fact which describes the attributes of an Entity ".

Properties or Attributes



I am creating an account for myself :

Attributes

First name : Rohan
Surname : Singh
Phone number : 9876543210
Password : Rohan@123
Dob : 14-MAY-199X
Gender : MALE

Laptop
Brand : *Dell*
RAM : *8gb*
Touch : *no*

Laptop
Brand : *Apple*
RAM : *16gb*
Touch : *yes*

Example : Water Bottle (*Entity*)

Height : 20cms
Color : blue
Capacity : 500ml

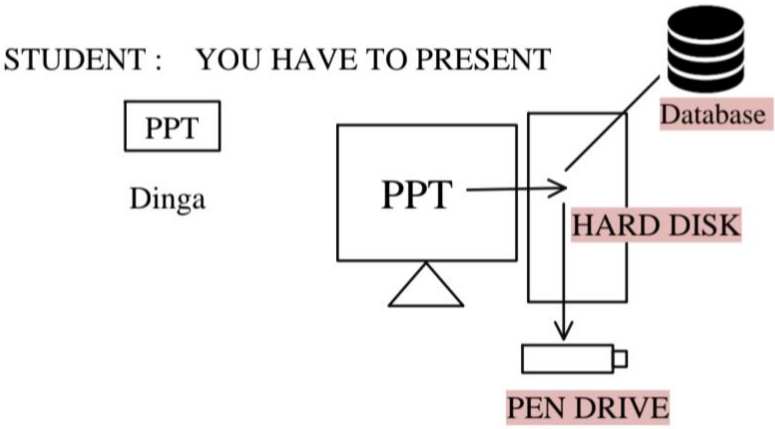
Attributes

Contacts

Name : *Dinga*
Phone : 108
DOB: *14-feb-95*

DATABASE :

"***Database is a place or a medium in which we store the data in a Systematic and organized manner*** "



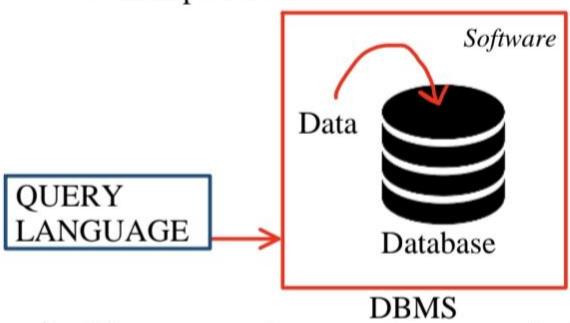
- The basic operations that can be performed on a database are
  - CREATE / INSERT
  - READ / RETRIEVE
  - UPDATE / MODIFY
  - DELETE / DROP

- These operations are referred as **"CRUD"** Operations .

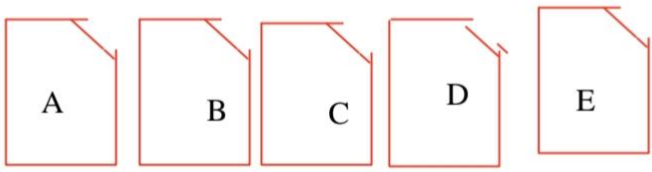
DATABASE MANAGEMENT SYSTEM (DBMS) :

"It is a software which is used to maintain and manage The database "

- **Security** and **authorization** are the two important features that DBMS provides .



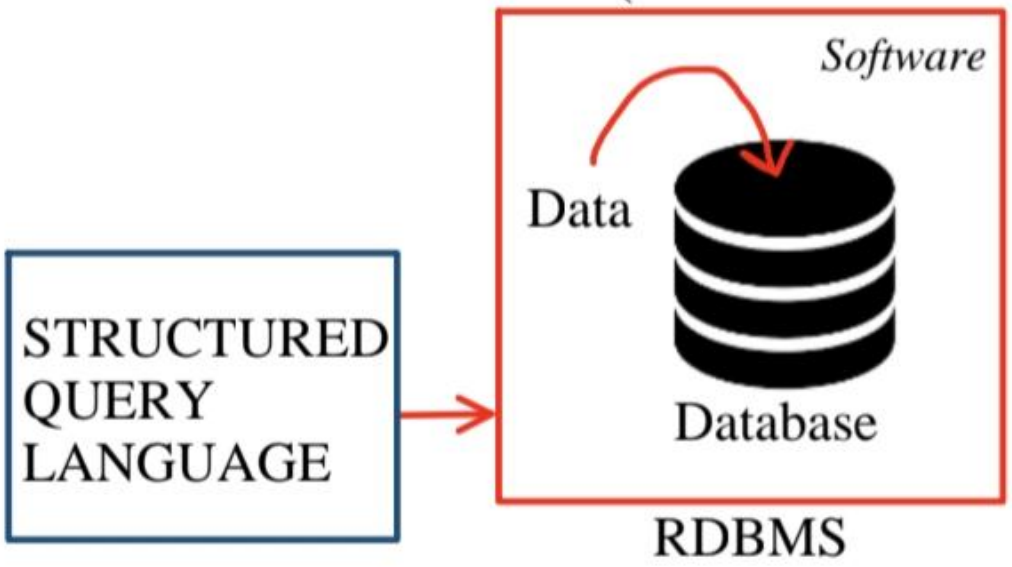
- We use query language to communicate or interact with DBMS
- DBMS stores the data in the form of *files* .



RELATIONAL DATABASE MANAGEMENT SYSTEM (RDBMS):

"It is a type of DBMS software in which we store the data

In the form of Tables (rows & columns) ".



- We use SQL to communicate or interact with RDBMS
- RDBMS stores the data in the form of *Tables*.

Example :

Names
A
B
C
D
E

DAY 2

Friday, 17 July 2020 8:59 AM

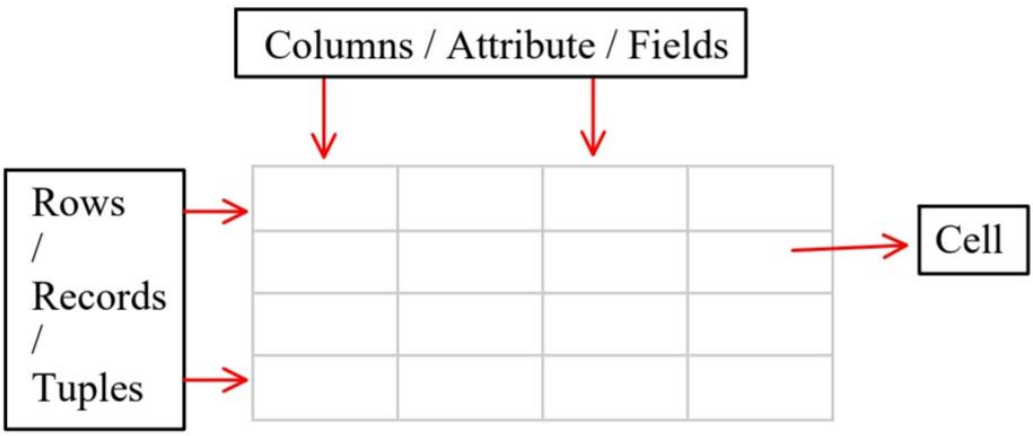
RELATIONAL MODEL :

Relational Model was designed by ***E.F CODD*** .
In Relational Model we can store the data in the from of *tables* .

Any DBMS which follows Relational Model becomes RDBMS .

Any DBMS which follows rules of EF CODD becomes RDBMS .

TABLE : "It is a logical organization of data which consists of Columns &Rows .



Example :

Emp (Entity)
- Eid
- Ename
- Salary

Employee :

EID	ENAME	SALARY
1	SMITH	1000
2	ALLEN	1500
3	CLARK	2000

RULES OF E.F CODD :

1. The data entered into a cell must always be a *single valued data* .

   Example :

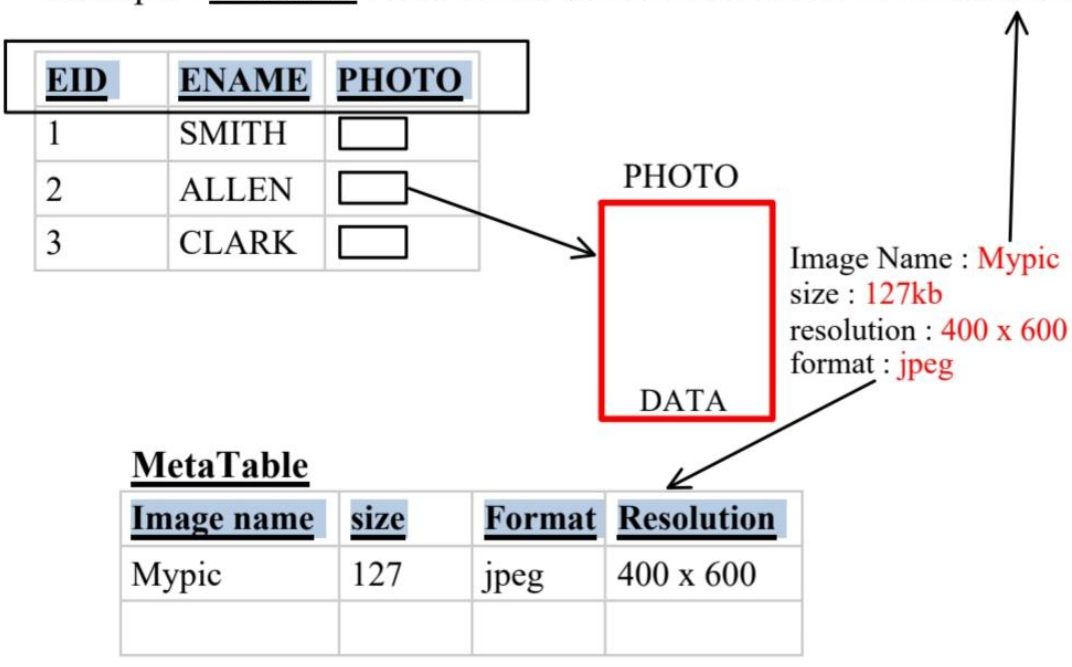
EID	ENAME	PHONE_NO
1	SMITH	101
2	ALLEN	102 , 202
3	CLARK	103

EID	ENAME	PHONE_NO	ALTERNATE NO
1	SMITH	101	
2	ALLEN	102	202
3	CLARK	103	

2. According to E.F CODD we can store the data in Multiple Tables , If needed we can establish a connection between the tables with the Help of *Key Attribute* .

3. In RDBMS we store everything in the from of tables including *Metadata* .

   Example : *Metadata* : The details about a data is knows as Metadata.



MetaTable

Image name	size	Format	Resolution
Mypic	127	jpeg	400 x 600

4. The data entered into the table can be validated in 2 steps .
    i. By assigning Datatypes .
    ii. By assigning Constraints .

   Datatypes are mandatory , whereas Constraints are Optional .

DATATYPES :

It is used to specify or determine the type of data that will be stored In a particular memory location .

Datatypes in SQL :

1. CHAR
2. VARCHAR / VARCHAR2
3. DATE
4. NUMBER
5. LARGE OBJECTS
    i. Character Large Object .
    ii. Binary Large Object .

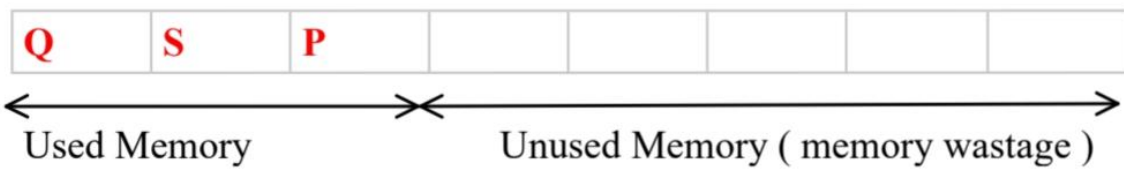
NOTE : SQL is not a Case Sensitive Language .

1. **CHAR :** In character datatype we can store 'A-Z' , 'a-z' , '0-9' And Special Characters($, & , @ , ! …) .

   - Characters must always be enclosed within single quotes ' '.
   - Whenever we use char datatype we must mention size
   - **Size** : it is used to specify number of characters it can store .
     - The maximum number of characters it can store is **2000ch**.
   - Char follows fixed length memory allocation .

   Syntax: CHAR (SIZE)

Example : CHAR (8)

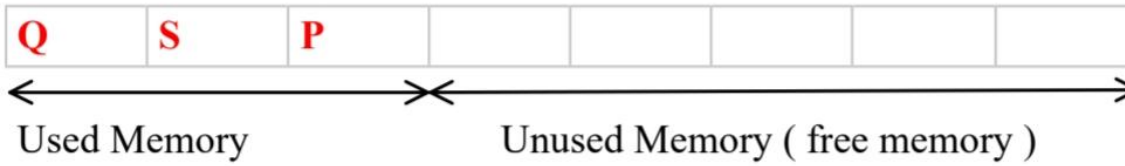


2. **VARCHAR :** In varchar datatype we can store 'A-Z' , 'a-z' , '0-9' And Special Characters($, & , @ , ! …) .

- Characters must always be enclosed within single quotes ' '.
- Whenever we use char datatype we must mention size
- **Size** : it is used to specify number of characters it can store .
  - The maximum number of characters it can store is **2000ch.**
- VarChar follows variable length memory allocation .

Syntax: VARCHAR (SIZE)

Example : VARCHAR (8)



NOTE : **VARCHAR2 :** it is an updated version of varchar where in We can store up to **4000Ch.**

Syntax: VARCHAR2(SIZE)

Example :

STUDENT

USN	SNAME	ADDRESS	PAN_NO
CHAR(4)	VARCHAR(10)	VARCHAR(10)	CHAR(10)
QSP1	DINGA	BANGALORE	ABC123XYZ1
QSP2	DINGI	MYSORE	ABC123XYZ2

ASSIGNMENT :

1. DIFFERENTIATE BETWEEN CHAR & VARCHAR

ASCII : [American Standard Code For Information Interchange]

3. **NUMBER :** It is used to store numeric values .

SYNTAX: **NUMBER** (Precision , [Scale])

[ ] - Not Mandatory .

Precision : it is used to determine the number of digits used To store integer value .

Scale : it is used to determine the number of digits used to store Decimal (floating) value within the precision .

- Scale is not mandatory , and the default value of scale Is zero (0) .

Example :	Number (3)	+/- 999
Example :	Number (5 , 0)	+/- 99999
Example :	Number (5 , 2)	+/- 999.99
Example :	Number (7 , 3)	+/- 9999.999
Example :	Number (4 , 4)	+/- .9999
Example :	Number (5 , 4)	+/- 9.9999
Example :	Number (3 , 6)	+/- .000999
Example :	Number (5 , 8)	+/- .00099999
Example :	Number (2 , 7)	+/- .0000099

EID	**PHONE_NO**	**SALARY**
Number(3)	Number (10)	Number (7 , 2)
101	9876543210	9000.85

4. **DATE :** it is used to store dates in a particular format .

It used *Oracle specified Format* .

'DD-MON-YY'	OR	'DD-MON-YYYY'
'22-JUN-20'		'22-JUN-2020'

SYNTAX: **DATE**

Example :

DOB	**Hiredate**	**Anniversary**
Date	Date	Date

'01-JAN-1945'	'20-JUN-20'	'15-APR-2008'

5. LARGE OBJECTS

1. Character large object (CLOB) :

It is used to store characters up to 4 GB of size .

2. Binary large object (BLOB) :

It is used to store binary values of images , mp3 , mp4 Documents etc …. Up to 4GB of size .

CONSTRAINTS :

It is a rule given to a column for validation .

Types of Constraints :

1. UNIQUE
2. NOT NULL
3. CHECK
4. PRIMARY KEY
5. FOREIGN KEY .

1. **UNIQUE** : "*It is used to avoid duplicate values into the column* ".

2. **NOT NULL** : *"It is used to avoid Null* ".

3. **CHECK** : "*It is an extra validation with a condition*
   If the condition is satisfied then the value is accepted else Rejected ".

4. **PRIMARY KEY** : "*It is a constraint which is used to identify a record Uniquely from the table* " .

   Characteristics of Primary key :
   - We can have only 1 PK in a table
   - PK cannot accept duplicate / repeated values .
   - PK cannot accept Null
   - PK is always a combination of Unique and Not Null Constraint.

5. **FOREIGN KEY** : "*It is used to establish a connection between the The tables* "

   Characteristics of Foreign key :
   - We can have only Multiple FK in a table
   - FK can accept duplicate / repeated values .
   - FK can accept Null
   - FK is not a combination of Unique and Not Null Constraint.
   - For an Attribute (column) to become a FK ,it is mandatory That it must be a PK in its own table .

Example :

EMP

Primary key				
		Check (Salary >		**Check**

		0)		(length(phone) = 10)
Not Null	Not Null	Not Null	Not Null	Not Null
Unique				Unique
EID	NAME	SALARY	DOJ	PHONE
Number(2)	Varchar(10)	Number(7,2)	Date	Number(10)
1	A	10000	'20-JUN-20'	9876543210
2	B	20000	'20-JUN-19'	9876543222
3	C	35000	'01-JAN-18'	9876543333
4	D	50000	'01-OCT-19'	9876511111

Example for Foreign Key :

Emp

EID	NAME	SALARY	DNO FK	CID FK
1	A	10000	20	2
2	B	20000	10	3
3	C	35000	20	1
4	D	50000	10	2

Child Table

Customer

CID	CNAME	CNO
1	X	1001
2	Y	2002
3	Z	3003

Parent Table

Dept

DNO	DNAME	LOC
10	D1	L1
20	D2	L2

Parent Table

ASSIGNMENT :

1. Differentiate between Primary key and Foreign key .

PRIMARY KEY	FOREIGN KEY
It is used to identify a records Uniquely from the table.	It is used to establish a connection Between the tables
It cannot accept Null	It can accept Null
It cannot accept duplicate values	It can accept duplicate values
It is always a combination of Not Null and Unique constraint	It is not a combination of Not Null and Unique constraint
We can have only 1 PK in a table	We can have Multiple FK in a table

NOTE : NULL

Null Is a *keyword* which is used to represent Nothing / Empty Cell.

Characteristics of Null :

- Null doesn't represent 0 or Space .
- Any operations performed on a Null will result in Null itself

- Null doesn’t Occupy any Memory .
- We cannot Equate Null .

OVERVIEW OF SQL *STATEMENTS* :

1. DATA DEFINITION LANGUAGE (DDL)
2. DATA MANIPULATION LANGUAGE (DML)
3. TRANSCATION CONTROL LANGUAGE (TCL)
4. DATA CONTROL LANGUAGE (DCL)
5. DATA QUERY LANGUAGE (DQL)

DATA QUERY LANGUAGE (DQL) :

" ***DQL is used to retrieve the data from the database*** " .

It had 4 statements :

1. SELECT
2. PROJECTION
3. SELECTION
4. JOIN

1. **SELECT :** "It is used to retrieve the *data* from the table and display it.

2. **PROJECTION :** "It is a process of retrieving the data by *selecting only the columns* is known as Projection " .
   - In projection all the records / values present in a particular column are by default selected .

3. **SELECTION :** "It is a process of retrieving the data by *selecting both the columns and rows* is known as Selection " .

4. **JOIN :**"It is a process of retrieving the data from *Multiple tables* simultaneously is known as Join " .

PROJECTION

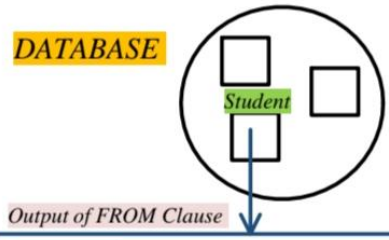
- "It is a process of retrieving the data by *selecting only the columns* is known as Projection " .
- In projection all the records / values present in a particular column are by default selected .

SYNTAX :
SELECT * / [DISTINCT] Column_Name / Expression [ALIAS]
FROM Table_Name ;

ORDER OF EXECUTION
1. FROM Clause
2. SELECT Clause

Example : Write a query to display names of all the students .

Example : Write a query to display names of all the students .




```
SELECT SNAME
FROM STUDENT ;
```

Output of FROM Clause

Student

SID	SNAME	BRANCH	PER
1	A	ECE	60
2	B	CSE	75
3	C	ME	50
4	D	ECE	80
5	C	CSE	75
6	E	CIVIL	95

Output of SELECT Clause

SNAME
A
B
C
D
C
E

NOTE :

- FROM Clause starts the execution .
- For FROM Clause we can pass Table_Name as an argument .
- The job of FROM Clause is to go to the Database and search for the table and put the table under execution .
- SELECT Clause will execute after the execution of FROM Clause
- For SELECT Clause we pass 3 arguments
  - *
  - Column_Name
  - Expression
- The job of SELECT Clause is to go the table under execution and select the columns mentioned .
- SELECT Clause is responsible for preparing the result table .
- Asterisk(*) : it means to select all the columns from the table .
- Semicolon : it means end of the query .

- WAQTD student id and student names for all the students.

```
SELECT SID , SNAME
FROM STUDENT ;
```

- WAQTD name and branch of all the students .

```
SELECT  SNAME , BRANCH
FROM STUDENT ;
```

- WAQTD NAME , BRANCH AND PERCENTAGE FOR ALL THE STUDENTS .

```
SELECT SNAME , BRANCH , PER
FROM STUDENT ;
```

```
SELECT *
FROM STUDENT ;
```

- WAQTD sname , sid , per , branch of all the students .

```
SELECT SNAME , SID , PER , BRANCH
FROM STUDENT ;
```

EMP Table :

EMPNO	ENAME	JOB	HIREDATE	MGR	SAL	COMM	DEPTNO
7369	SMITH	CLERK	17-DEC-80	7902	800		20
7499	ALLEN	SALESMAN	20-FEB-81	7698	1600	300	30
7521	WARD	SALESMAN	22-FEB-81	7698	1250	500	30
7566	JONES	MANAGER	02-APR-81	7839	2975		20
7654	MARTIN	SALESMAN	28-SEP-81	7698	1250	1400	30
7698	BLAKE	MANAGER	01-MAY-81	7839	2850		30
7782	CLARK	MANAGER	09-JUN-81	7839	2450		10
7788	SCOTT	ANALYST	19-APR-87	7566	3000		20
7839	KING	PRESIDENT	17-NOV-81		5000		10
7844	TURNER	SALESMAN	08-SEP-81	7698	1500	0	30
7876	ADAMS	CLERK	23-MAY-87	7788	1100		20
7900	JAMES	CLERK	03-DEC-81	7698	950		30
7902	FORD	ANALYST	03-DEC-81	7566	3000		20
7934	MILLER	CLERK	23-JAN-82	7782	1300		10

- **WAQTD name salary and commission given to all the employees .**

```
Select ename , sal , comm
From emp ;
```

- **WAQTD name of the employee along with their date of joining .**

```
Select ename , hiredate
From emp ;
```

DEPT :

DEPTNO	DNAME	LOC
10	ACCOUNTING	NEW YORK
20	RESEARCH	DALLAS
30	SALES	CHICAGO
40	OPERATIONS	BOSTON

- **WAQTD dname and location for all the depts .**

```
Select dname , loc
From dept ;
```

QUESTIONS ON EMP AND DEPT TABLE:

1.WRITE A QUERY TO DISPLAY ALL THE DETAILS FROM THE

DISTINCT Clause

" It is used to remove the duplicate or repeated values from the Result table " .

- Distinct clause has to be used As the first argument to select clause .
- We can use multiple columns As an argument to distinct clause, it will remove the combination of columns in which the records are duplicated .

Example :

Student

SID	SNAME	BRANCH	PER
1	A	ECE	60
2	B	CSE	75
3	C	ME	50
4	D	ECE	80
5	C	CSE	75
6	E	CIVIL	95

- SELECT SNAME
  FROM STUDENT ;

SNAME
A
B
C
D
C
E

- SELECT DISTINCT SNAME
  FROM STUDENT ;

SNAME
A
B
C
D
C
E

→

SNAME
A
B
C
D
E

- SELECT DISTINCT BRANCH
  FROM STUDENT ;

BRANCH
ECE
CSE
ME
ECE
CSE
CIVIL

→

BRANCH
ECE
CSE
ME
CIVIL

- SELECT DISTINCT PER
  FROM STUDENT ;

PER
60
75
50
80
75
95

→

PER
60
75
50
80
95

- SELECT DISTINCT BRANCH , PER

FROM STUDENT ;

BRANCH	PER
ECE	60
CSE	75
ME	50
ECE	80
CSE	75
CIVIL	95

BRANCH	PER
ECE	60
CSE	75
ME	50
ECE	80
CIVIL	95

EXPRESSION

"A statement which gives result is known as Expression ".

Expression is a combination Operand and Operator .

Operand : These are the values that we pass .

Operator : These are the Symbols which perform some Operation on The Operand .

Example : 5 * 10

EMP

EID	ENAME	SAL
1	A	100
2	B	200
2	C	100

1. WAQTD name and salary given to the employees .

   SELECT ENAME , SAL
   FROM EMP ;

2. WAQTD name and annual salary of the employees .

   SELECT ENAME , SAL * 12
3. FROM EMP ;

ENAME	SAL*12
A	1200
B	2400
C	1200

4. WAQTD all the details of the employee along with annual salary

   Select eid , ename , sal , sal*12
   From emp ;

   Select emp.* , sal*12
   From emp ;

5. WAQTD name and salary with a hike of 20% .

   Select ename , Sal + Sal*20/100
   From emp ;

Formulae to calculate percentage :

Sal + Sal * a / 100	Sal * 1.a

6. WAQTD name and salary of an employee with a deduction Of 10% .

Select ename , sal - sal * 10 /100
From emp ;

ALIAS

"It is an alternate name given to a Column or an Expression In the result table " .

- We can assign alias name with or without using 'As' keyword .
- Alias names have to be a single string which is separated by An underscore or enclosed within double quotes .

Example :	ANNUAL_SALARY
	"ANNUAL SALARY"

➢ WAQTD annual salary for all the employees .

Select sal*12
From emp ;

SAL*12
1200
2400
1200

Select sal*12 Annual_Salary
From emp ;

Annual Salary
1200
2400
1200

Select sal + sal * 10 / 100 Hike
From emp ;

➢ WAQTD name and salary with a deduction 32% .

Select Ename , sal-sal*32/100 as deduction
From emp ;

OF 10%.
8.WAQTD TOTAL SALARY GIVEN TO EACH EMPLOYEE (SAL+COMM).
9.WAQTD DETAILS OF ALL THE EMPLOYEES ALONG WITH ANNUAL SALARY.
10.WAQTD NAME AND DESIGNATION ALONG WITH 100 PENALTY IN SALARY.

SELECTION :

"It is a process of retrieving the data by *selecting both the columns and rows* is known as Selection " .

SYNTAX :
SELECT * / [DISTINCT] Column_Name / Expression [ALIAS]
FROM Table_Name
WHERE <*Filter_Condition*> ;

ORDER OF EXECUTION

1. FROM
2. WHERE
3. SELECT

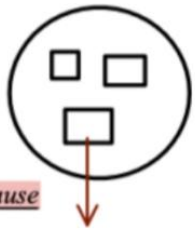
WHERE Clause

"Where clause is used to filter the records ".

Example :

- WAQTD names of the employees working in dept 20 .

Example :



3- SELECT ENAME
1- FROM EMP
2- WHERE DNO = 20 ;

Output of FROM Clause

EMP

EID	ENAME	SAL	DNO
1	SMITH	100	10
2	ALLEN	250	20
3	BLAKE	300	30
4	MILLER	400	10
5	JONES	250	20

Filter Condition
DNO = 20

1	SMITH	100	10
2	ALLEN	250	20
3	BLAKE	300	30
4	MILLER	400	10
5	JONES	250	20

Output of WHERE Clause

EID	ENAME	SAL	DNO
2	ALLEN	250	20
5	JONES	250	20

Output of SELECT Clause

ENAME
ALLEN
JONES

- WAQTD names of the employees getting salary More than 300 .

```
SELECT ENAME
FROM EMP
WHERE SAL > 300 ;
```

- WAQTD names and salary of the employees working in dept 10.

```
SELECT ENAME , SAL
FROM EMP
WHERE DEPTNO = 10  ;
```

- WAQTD all the details of the employees whose salary is Less than 1000 rupees .

```
SELECT *
FROM EMP
WHERE SAL < 1000 ;
```

EMP :

EMPNO	ENAME	JOB	HIREDATE	MGR	SAL	COMM	DEPTNO
7369	SMITH	CLERK	17-DEC-80	7902	800		20
7499	ALLEN	SALESMAN	20-FEB-81	7698	1600	300	30
7521	WARD	SALESMAN	22-FEB-81	7698	1250	500	30
7566	JONES	MANAGER	02-APR-81	7839	2975		20
7654	MARTIN	SALESMAN	28-SEP-81	7698	1250	1400	30
7698	BLAKE	MANAGER	01-MAY-81	7839	2850		30
7782	CLARK	MANAGER	09-JUN-81	7839	2450		10
7788	SCOTT	ANALYST	19-APR-87	7566	3000		20
7839	KING	PRESIDENT	17-NOV-81		5000		10
7844	TURNER	SALESMAN	08-SEP-81	7698	1500	0	30
7876	ADAMS	CLERK	23-MAY-87	7788	1100		20
7900	JAMES	CLERK	03-DEC-81	7698	950		30
7902	FORD	ANALYST	03-DEC-81	7566	3000		20
7934	MILLER	CLERK	23-JAN-82	7782	1300		10

- WAQTD name and hiredate of an employee hired on '09-JUN-1981'

```
SELECT ENAME , HIREDATE
FROM EMP
WHERE  DATE = '09-JUN-1981' ;
```

- WAQTD details of the employee whose name is 'Miller'

```
SELECT *
FROM EMP
WHERE ENAME ='MILLER'  ;
```

- WAQTD details of the employee hired after '01-JAN-1982'

```
SELECT *
FROM EMP
WHERE HIREDATE > '01-JAN-1982' >  ;
```

- WAQTD name sal and hiredate of the employees who were Hired before 1985 .

```
SELECT ENAME , SAL , HIREDATE
FROM EMP
WHERE HIREDATE < '01-JAN-1985' ;
```

- WAQTD name sal and hiredate of the employees who were Hired after 1985 .

```
SELECT ENAME , SAL , HIREDATE
FROM EMP
WHERE HIREDATE  > '31-DEC-1985' ;
```

OPERATORS IN SQL

1. ARITHEMATIC OPERATORS :- (+ , - , * , /)
2. CONCATENATION OPERATOR :- (||)
3. COMPARISION OPERATORS :- (= , != or <>)
4. RELATIONAL OPERATOR :- (> , < , >= , <=)
5.LOGICAL OP : (AND , OR , NOT)
6.SPECIAL OPERATOR :-
    1.IN
    2.NOT IN
    3.BETWEEN
    4.NOT BETWEEN
    5.IS
    6.IS NOT
    7.LIKE
    8.NOT LIKE
7.SUBQUERY OPERATORS:-
    1.ALL
    2.ANY
    3.EXISTS
    4.NOT EXISTS

CONCATENATION Operator :

" It is used to join the strings ".

Symbol : ||

Example :
```
SELECT ENAME
FROM EMP
WHERE JOB ='MANAGER' ;
```

Ename
ALLEN
MARTIN
SMITH

```
SELECT 'Hi ' || ename
FROM EMP
WHERE JOB ='MANAGER' ;
```

Ename
Hi ALLEN
Hi MARTIN
Hi SMITH

- WAQTD name and deptno of the employees hired After '01-JAN-87' .

```
SELECT ENAME , DEPTNO
FROM EMP
WHERE HIREDATE > '01-JAN-1987' ;
```

- WAQTD name and hiredate of the employees hired before 31-JUL-88

```
SELECT ENAME , HIREDATE
FROM EMP
WHERE HIREDATE < '31-JUL-88' ;
```

LOGICAL OPERATORS

1. AND
2. OR
3. NOT

We use logical operators to write multiple conditions .

1. WAQTD name and deptno along with job for the employee working in dept 10 .

```
SELECT ENAME , DEPTNO , JOB
FROM EMP
WHERE DEPTNO = 10 ;
```

2. WAQTD name and deptno along with job for the employee working as manager in dept 10 .

```
SELECT ENAME , DEPTNO , JOB
FROM EMP
WHERE JOB ='MANAGER' AND DEPTNO = 10 ;
```

3. WAQTD name , deptno , salary of the employee working in dept 20 and earning less than 3000 .

```
SELECT ENAME, DEPTNO , SAL
FROM EMP
WHERE DEPTNO = 20 AND SAL < 3000 ;
```

4. WAQTD name and salary of the employee if emp earns More than 1250 but less than 3000 .

```
SELECT ENAME , SAL
FROM EMP
WHERE SAL > 1250 AND SAL < 3000 ;
```

```
SELECT ENAME , DEPTNO
FROM EMP
WHERE DEPTNO = 10 OR DEPTNO = 20 ;
```

6. WAQTD name and sal and deptno of the employees If emp gets more than 1250 but less than 4000 and works in dept 20 .

```
SELECT ENAME , SAL , DEPTNO
FROM EMP
WHERE SAL > 1250 AND SAL < 4000 AND DEPTNO
=20 ;
```

7. WAQTD name , job , deptno of the employees working as a manager in dept 10 or 30 .

```
SELECT ENAME , JOB , DEPTNO
FROM EMP
WHERE JOB ='MANAGER' AND ( DEPTNO = 10 OR
DEPTNO = 20 ) ;
```

8. WAQTD name , deptno , job of the employees working in dept 10 or 20 or 30 as a clerk .

```
SELECT ENAME , JOB , DEPTNO
FROM EMP
WHERE JOB ='CLERK' AND ( DEPTNO = 10 OR
DEPTNO = 20 AND DEPTNO = 30 ) ;
```

9. WAQTD name , job and deptno of the employees working as clerk or manager in dept 10 .

```
SELECT ENAME , JOB , DEPTNO
FROM EMP
WHERE ( JOB = 'CLERK' OR JOB ='MANAGER' )
AND DEPTNO = 10 ;
```

10. WAQTD name , job , deptno , sal of the employees working as clerk or salesman in dept 10 or 30 and earning more than 1800 .

```
SELECT ENAME , JOB , SAL
FROM EMP
WHERE ( JOB ='CLERK' OR JOB ='SALESMAN')
AND ( DEPTNO = 10 OR DEPTNO = 30 ) AND SAL >
1800 ;
```