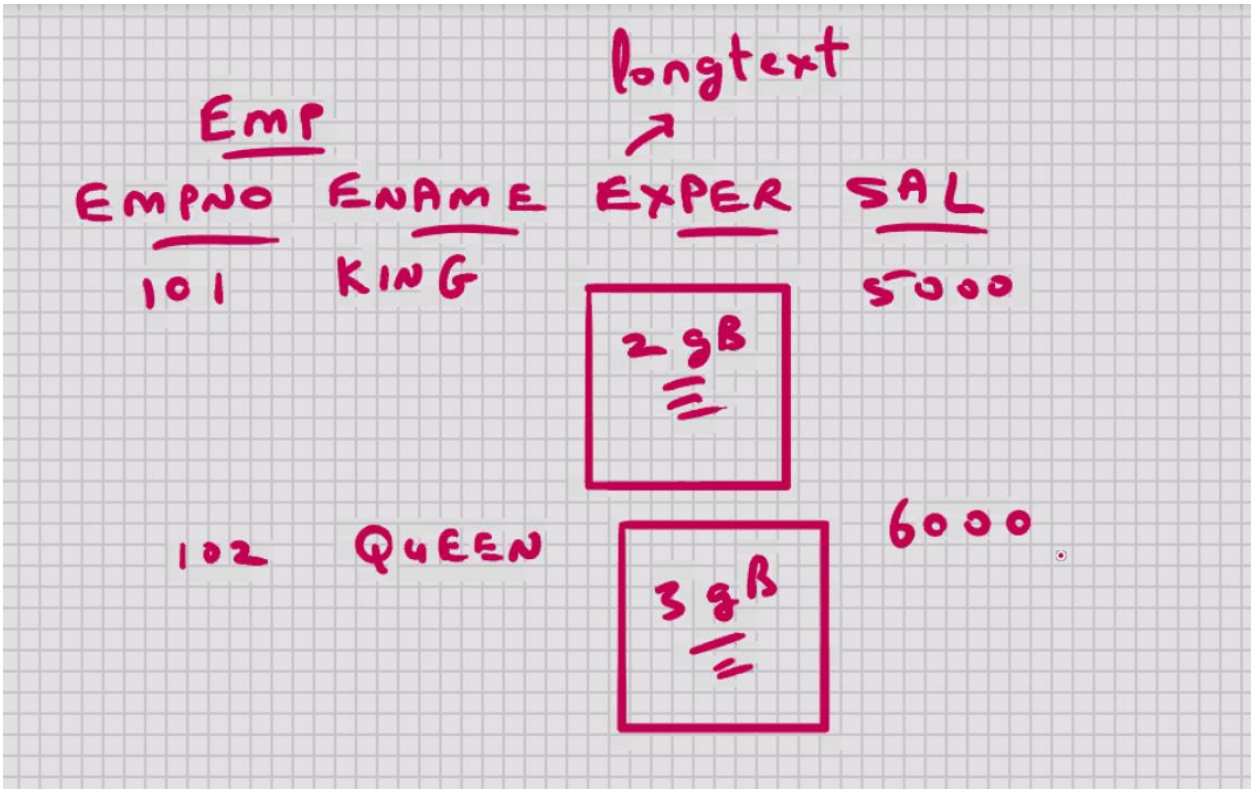
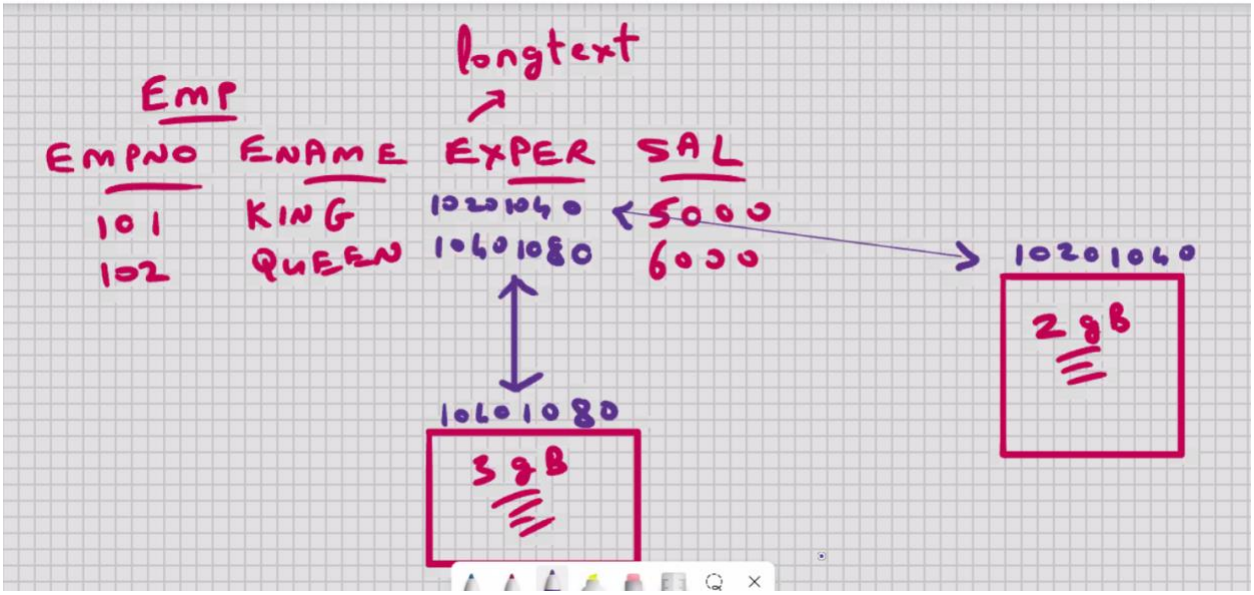
Longtext

* allows any character
* could be alpha-numeric also
* max 4,294,967,295 characters
* 4 Gd - 1
* eg. FEEDBACK,COMMENTS,REMARKS.POST,DIESCRIP,RESUME,EXPERIENCE etc.
* stored outside the table.
* stored outside the row.
* stored away from the table.
* stored away from the row.





Pointer --→ hard disk pointer →locator

- My sql maintances a LOCATOR (hd POINTER) from the longtext column to the longtext data
- Performance of other columns that would be used for searching
- Used for those columns that are only for storage and display purposes.
- Width does not have to be specified
- Eg . FEEDBACK longtext
- Variable length

Binary

- Fixed-length binary string
- Max upto 255 Bytes of binary data
- Eg:- BARCODES ,QR_CODES,PICTURE_CODES,SIGNATURES,FINGERPRINTS etc.
- Width need not be specified.

Varbinary

- Variable-length binary string
- Max upto 65,535 Byte if binary data
- E.g:- **ICONS,** GIFS WITH SOUND,THUMBNAIL,FAVICON etc.
- no default width, width has to be specified.

- **Both of the above are stored as character Strings of 1's and 0's.**

BLOB ->Binary Large Object

Longblob

- Max upto 4,294,967,295 Bytes of binary data
- 4Gb-1
- e. g:- PHOTOGRAPHS , CHATS, GRAPHS, MAPS, SOUNDS, MUSIC, VIDEOS etc.
- width does not have to be specified.
- E.g PHOTOGRAPH longblob
- Stored outside the table.
- Stored outside the row.
- Stored away from the table.
- Stored outside the row.
- My SQL maintains a LOCATOR(HD)

Integer Types(Exact values):-

= Singned or Unsigned

= by default it is Singned

Tinyint (1 Byte)

Smallint(2 Byte)

Mediumint(3 Byte)

Int (4 Byte)

Bigint(8 Byte)

Floating-Point Types(Approximate value):-

Float: == upto 7 decimal

Double == upto 15 Demical

++ used when it is important ti preserve the exact precision ,for example with monetary data

---

Boolean

+ logical datatype

True and false on 1-0

??????????

Date and Time datatype

Date

- 1st Jan 1000 AD to 31st Dec 9999 AD
- No problem of Y2K in MySQL
- "YYYY-MM-DD" is also suppored
- Year values in range 70-90 are converted to 1970-1999.
- Year values in range 00-69 are converted to 2000-2069.
  '2024-10-09' or '24-10-09'
  '1947-08-15'

  1970 is known as the year of Epoch

  Date1-date2→ return the number of days between the 2 dates

  e.g '2024-10-09' – '2024-10-08'

1st jan 1000AD-> 1

2nd jan 1000 AD->2

3rd jan 1000 AD ->3

…

9th Oct 2024 AD -> 1657218(number of days since 1st jan 1000 AD)

- Internally date is stored as a fixed length number and it occupies 7 bytes of storage

Time

- 'hh:mm:ss' or 'HHH:MM:SS'
- Time values can range from '-838:59:59' to '838:59:59'
- Time1-Time2

Datetime

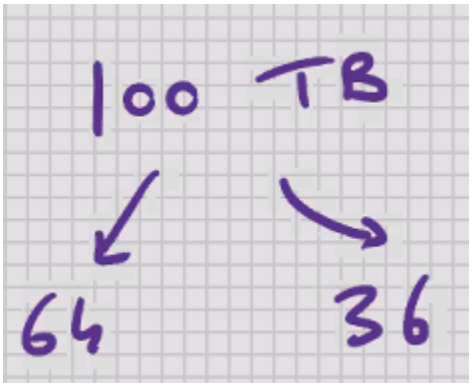
- Date and time is stored together
- 'YYYY-MM-DD hh:mm:ss' is the default datetime format in MySQL
- '1000-01-01 00:00:00' to '9999-12-31 23:59:59'
- datetime1-datetime2 -> return number of days ,remainder hours, minutes and second between the two

Year

- YYYY
- 1901 to 2155

In MySQL:-

- You can have max 4096 columns per table provided the row size<= 65,535.
- No limit on the number of row per table, provided the table size <= 64 Terabyte



- Largest table in the world->
  ORDERS table of amazon.com-> 100's of Terabytes daily
  Amazon uses oracle because no table size.

Programming ;


```
Create table EMP
(
Empno  char(4)
Ename varchar(25)
Sal float                                   float default range -   (-3.4 E+38 to -1.2 E38)
City varchar(15)
Dob date
);
```

Table created.

- **;** is known as delimiter (also know as terminator) it indicates end of command
- SQL commands are case – insensitive

INSERT

Insert into emp

Values('1' , 'Amit' , 5000 , 'Mumbai' , '1995-01-15');

- When you are inserting the data it is case sensitive
- For char ,varchar, date, time, datetime, use ' '.
- 'YYYY-MM-DD' is the default date format in MySQL
  '1995-01-15' or '95-01-15'

Project Guideline

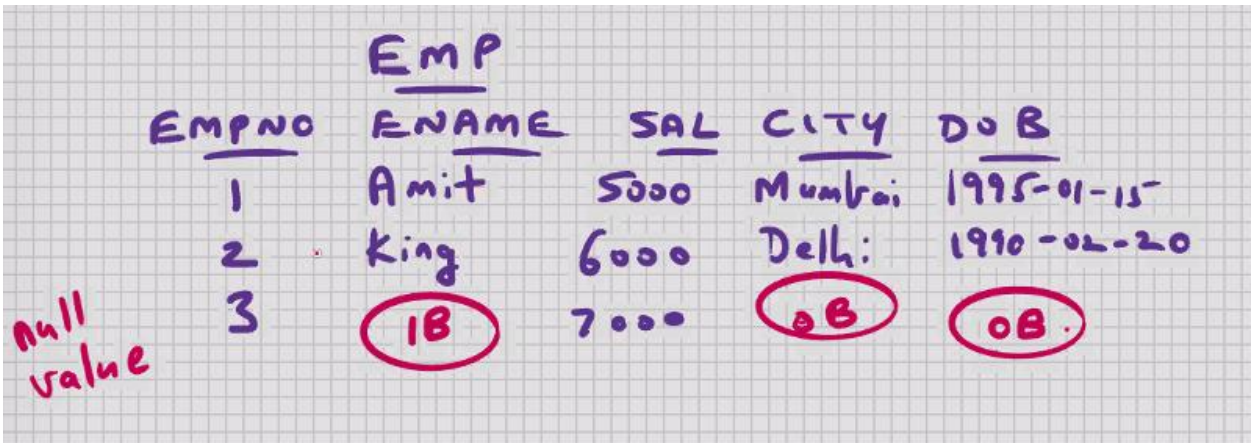
Recommended :-
1. Flexible
2. Readable
3.In future if you Alter the table and add a column ,then this Insert Statement will continue to work

Insert into emp(empno , sal, ename, city, dob)

Values('2' , 6000, 'King' , 'Delhi' , '1990-02-20');

Insert into emp(empno , sal)

Values('2' , 6000);



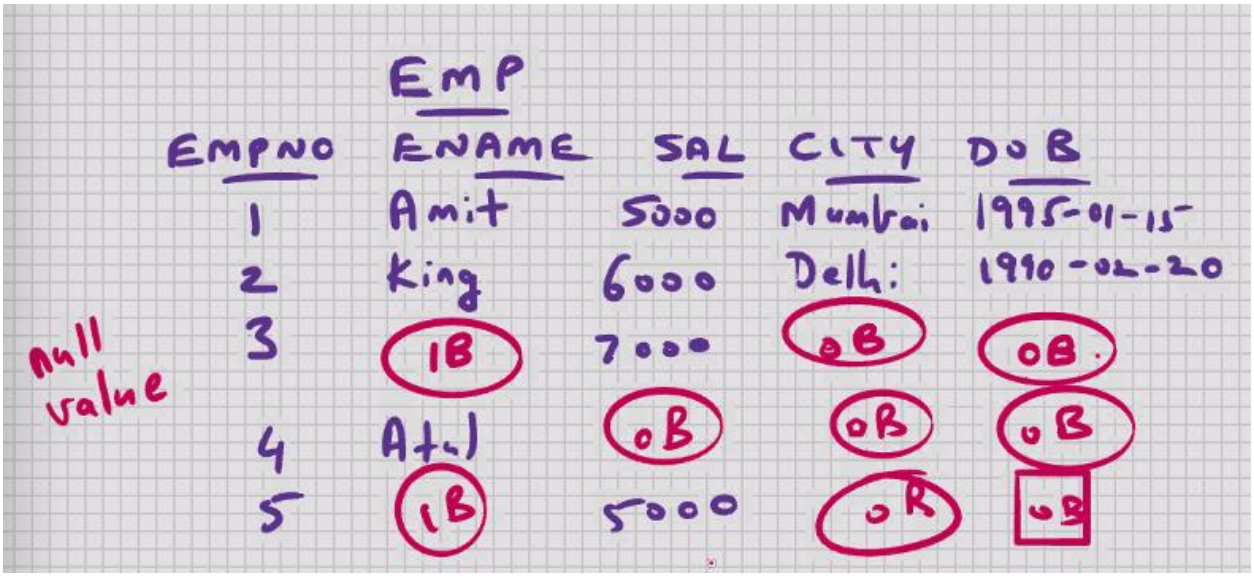
Null value

- Null means nothing
- Null values is having ASCII values 0
- Special treatment given ti null in all RDBMS
- Null values id independent of datatype
- Null values occupies only 1 byte of storage
- If you enter a null values for the last column of If the row is ending with null values ,then all those wont occupy any space.
- Those columns that are likely to have a large number of nulls its recommended that preferred that preferably they should be specified at the end of the table structure to converse on HD space.

```
insert into emp
Values('4' , 'Atul');          <= ERROR(not enough values)
```

```
insert into emp
Values('4' , 'Atul', null, null, null);
```

```
insert into emp
Values('5' , null, 5000, null, null);
```



To multiple rows in a table using a single INSERT statement;

Insert into emp values
('1' , 'Amit' , 5000, 'Delhi' , '1990-02-20'),
('1' , 'King' , 6000, 'Mumbai' , '1998-05-27'),
('1' , null , 5000, 'Delhi' , '1990-02-20'),
('1' , 'Amit' , null, null , null);

Insert into emp(empno ,sal) values
('1', 5000),
('2', 6000),
('3', 7000),
('4', 8000),

SELECT – print all the rows all the column in the table

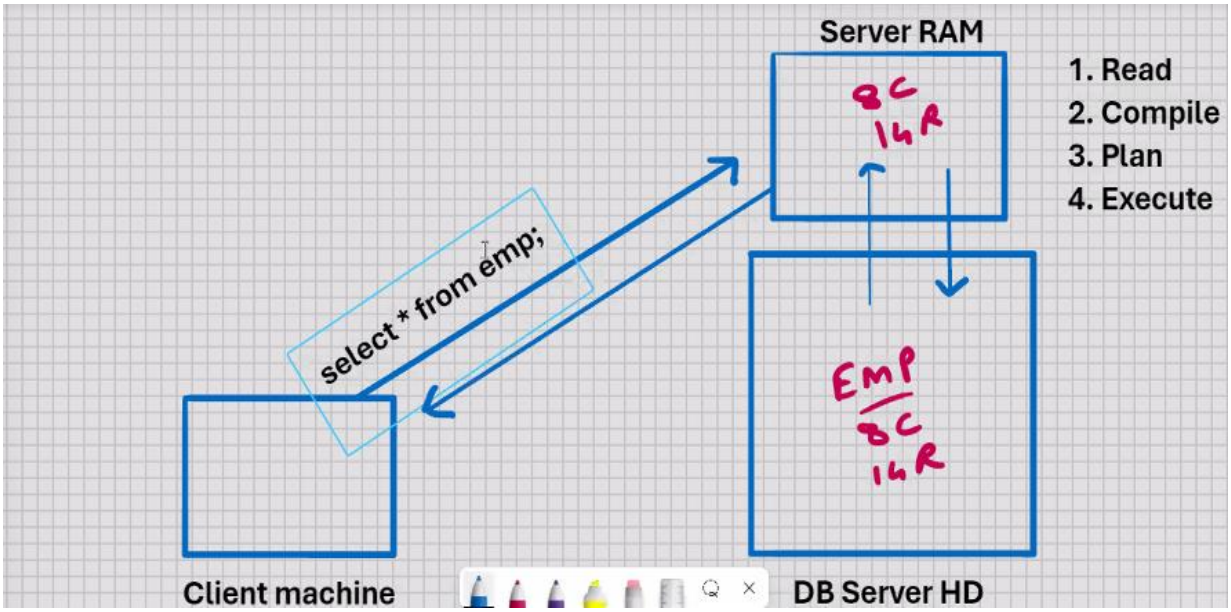
Select * from emp;
Select * from dept;

```
mysql> select * from emp;

     EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM     DEPTNO
---------- ---------- --------- ---------- ---------- ---------- ---------- ----------
      7839 KING       PRESIDENT            1981-11-17       5000                    10
      7698 BLAKE      MANAGER         7839 1981-05-01       2850                    30
      7782 CLARK      MANAGER         7839 1981-06-09       2450                    10
      7566 JONES      MANAGER         7839 1981-04-02       2975                    20
      7654 MARTIN     SALESMAN        7698 1981-09-28       1250       1400         30
      7499 ALLEN      SALESMAN        7698 1981-02-20       1600        300         30
      7844 TURNER     SALESMAN        7698 1981-09-08       1500          0         30
      7900 JAMES      CLERK           7698 1981-12-03        950                    30
      7521 WARD       SALESMAN        7698 1981-02-22       1250        500         30
      7902 FORD       ANALYST         7566 1981-12-03       3000                    20
      7369 SMITH      CLERK           7902 1980-12-17        800                    20
      7788 SCOTT      ANALYST         7566 1982-12-09       3000                    20
      7876 ADAMS      CLERK           7788 1983-01-12       1100                    20
      7934 MILLER     CLERK           7782 1982-01-23       1300                    10

14 rows selected.

mysql>
```

```
mysql> select * from dept;

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

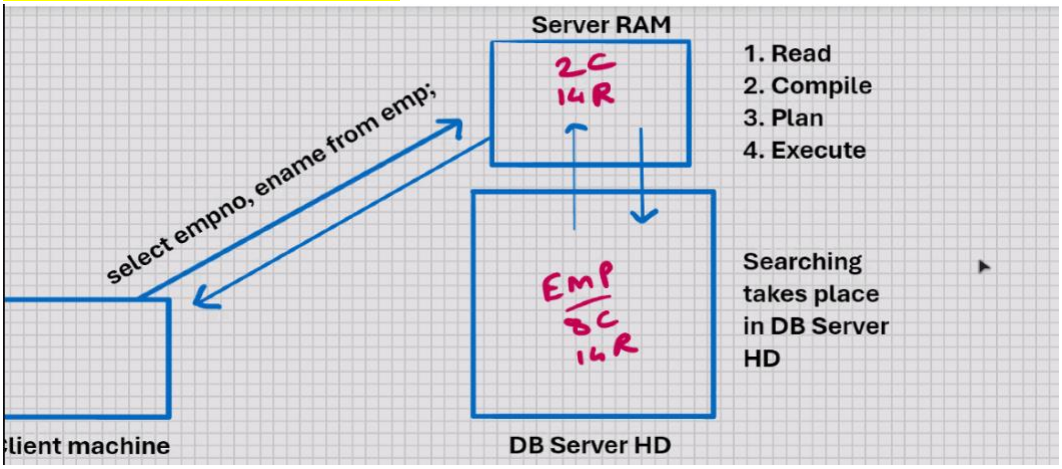
select * from emp;

1.Read
2.compile
3. Plan
4. Execute



--To restrict the columns;

Select empno, ename from emp;



Select deptno ,job, ename, sal, hiredate from emp;

- The position of columns in Select statement will determine the position of coloumns in the output.
- This you will write as per client requirement.

To restrict the rows :-
WHERE clause:-

```
Select*from emp
Where deptno=10;
```

- Where clause is used for searching
- Searching takes place is DB server HD
- WHERE clause is used to restrict the rows
- WHERE clause is used to retrive the rows from DB server to server RAM.

```
Select * from emp
Where sal>2000;
```

Relational Operators;-

1.) >
2.) >=
3.) <
4.) <=
5.) != or <>
6.) ==

```
Select * from emp
Where sal>2000 and sal<3000;
```

Logical Operator

1.) NOT
2.) AND
3.) OR

```
Select* from emp
Where deptno =10 or sal>2000 and sal<3000;
```

```
Select* from emp
Where (deptno =10 or sal>2000) and sal<3000;
```

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

Job is character column so we have to use single quote in manager.

for char, varchar, date, time and datetime use '

IN MYSQL

- When you are searching, queries are case-insensitive
- More user-friendly
-

IN other RDBMS

- When you are searching, queries are case sensitive
- More secure

```
Select * from emp
Where job ='MANAGER' and job ='CLERK';
```
→ nothing comes

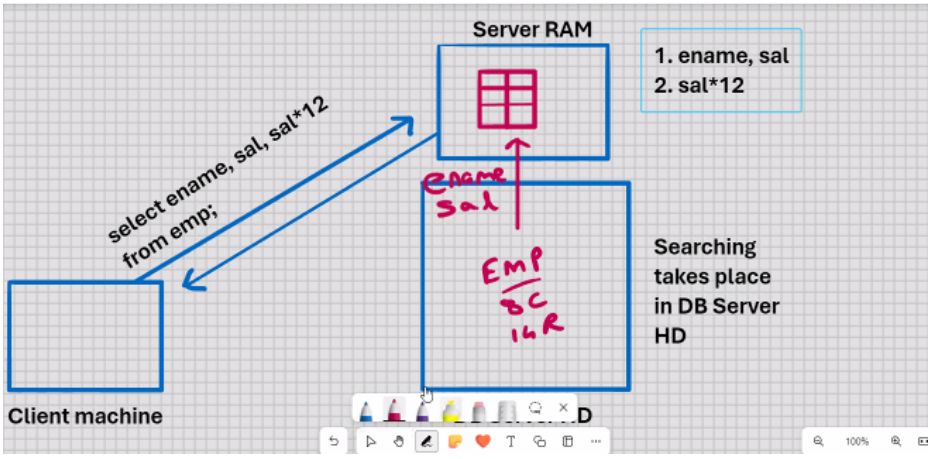
```
Select * from emp
Where job ='MANAGER' or job ='CLERK';
```
→printing all

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

Sal*12 is just for display not real coloum in our DB

Sal*12→ computed column , calculated column, virtual column, fake column, pseudo column

- Computed columns e.g yearly sal , are not saved in the table because that would be a wastage of HD space.
- As and when required you can SELECT sal*12



Arithmtic Operator

1. ( )
2. /
3. *
4. +
5. –

alias

Select ename, sal, sal*12 as “ANNUAL”

From emp;

Select ename, sal, sal*12 “ANNUAL”

From emp;

as -> ANSI SQL

as -> optional in MySQL

Select ename, sal, sal*12 annual

From emp;

If we remove double quote the anuual comes in ANNUAL (capital) in database .

Project Guideline

- Its recommended that you should always use double quotes ,the SELECT statement becomes more Readable.

```
Select ename, sal, sal*12 annual
From emp
Where sal*12 >300000;
```

```
Select ename “EMPNAME”
Sal “SALARY”
Sal*12 “ANNUAL”
From emp;
```

- You can specify alias for column of table also

```
Select ename “EMPNAME”
Sal “SALARY”,
Sal*12  “ANNUAL”,
Sal*12*0.4 “HRA”,
SAL*12 + sal*12*0.4 +sal *12*0.2 “NET”
From emp;
```

- You cannot use alias

"C:\Program Files\MySQL\MySQL Server 8.0\bin\mysql.exe" "--defaults-file=C:\ProgramData\MySQL\MySQL Server 8.0\my.ini" "-uroot" "-p"

```
mysql> use mysql;

When you install MySQL, few users are automatically created:-

1. root
*       DBA privileges (Database Administrator)
*       create database, drop database, alter database, configure
        database, create users, assign privileges, take backups,
        performance monitoring, performance tuning, performance planning,
        performance management, etc.
```

```
2. mysql.sys
*       most important user
*       owner of database and system tables
*       startup database, shutdown database, perform Recovery
```

10/10/2024

DSELECTION -> where you select specific columns from the table; it is known as selection.

Where Clause <- to view specific rows

PROJECTION -> When you select specific rows ;it is known as projection

- When you perform SSELECTION or PROJECTION , you are viewing a subset of the data

Relational, Logical , Arithmetic Operators

Connect to database using MYSQL command line client , show databases,

Create database/schema, Create user, Grant privileges

To see which tables are created;

Mysql> show tables;

To see structure of table :-

Mysql > desc emp; (describe)

Use <dbname>;

Select user from user; -- return username

Select * from user;

Documentation

MySQL Documentation
https://dev.mysql.com/doc/refman/8.4/en/

Oracle Documentation
https://docs.oracle.com

SQL

Select job from emp;

Duplicate remove only distinct – to supress the duplicate

Select distinct job from emp;

- Whenever you use DISTINCT sorting takes place internally in the server RAM.
- If you have large number of rows in the table, this SELECT statement will be slow.

Select distinct job , ename from emp;

- DISTINCT will work on combination of all the columns that are present in SELECT statement.
- (*) (ERROR)
  Select (distinct job),ename from emp;

Select deptno, job, ename, sal, hiredate from emp;

- In a DBMS , data is stored inside a file.
- Inside a file , row are stored sequentially.
- In Dbms ;there is concept of row numbering.
- In Rdbms table is not a file ; every row is a file.
- In RDBMS ,rows of the tables are not stored sequentially they are not stored in any specific order.
- Rows of the tables are scattered (Fragmented) all over the DB Server HD.
- The reason why RDBMS does this is to speed up the INSERT statement .
- In a multi-user environment when multiple users are inserting rows sequentially then the INSERT operation would be very slow.
- When you SELECT from a table, the searching is sequential.
- When you SELECT from a table the order of rows in the output depends on the row address; it will always be in ascending order of row address.
- Once you INSERT a row ,the row address is constant.

```
File  Edit  View
        depends on the row address; it will always be in ascending order
        of row address
*       Once you INSERT a row, the row address is constant
*       When you UPDATE a row, if the row length is increasing and the
        free space is not available, then the entire row is moved to some
        other address on the DB Server HD
*       Later when you SELECT from the table, you may see that row at some
        other location
*       It's only in the case of VARCHAR that the row length may increase
        or decrease
*       Hence it is not possible to see the first 'N' rows of a table, or
        the last 'N' rows of a table
*       this is common for all RDBMS
```

- There is a system table which stores all the table names and their respective row addresses.

n

```
*       When you SELECT from a table, MySQL will not search the entire
        DB Server HD; rather it will go to that system table, and retriev
        the rows of the SELECTed table accordingly
```

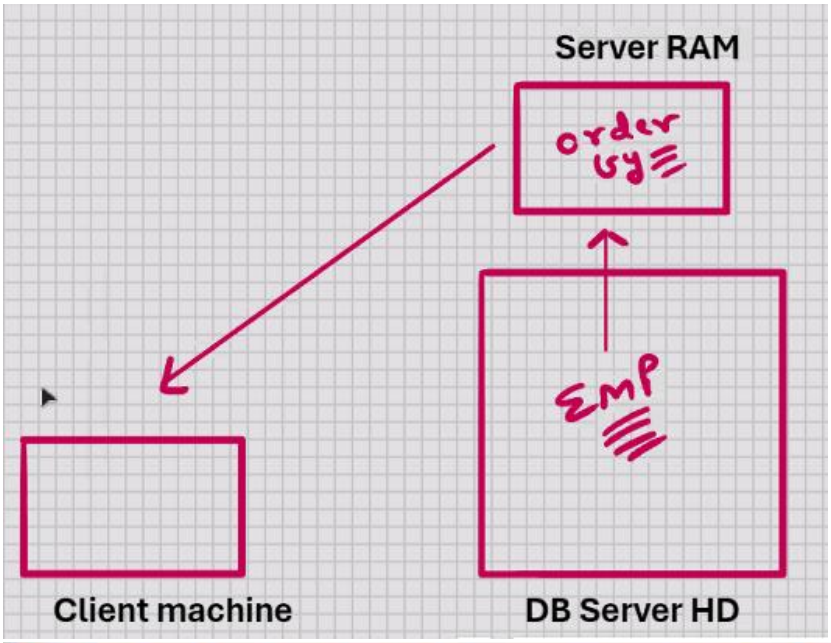
MySQL -SQL - order by clause

- Used for sorting
- Used for presentation /reporting purpose

```
select deptno, job, ename, sal, hiredate from emp
order by ename;

select deptno, job, ename, sal, hiredate from emp
order by ename desc;
```

Asc <- by default

Desc




```
select deptno, job, ename, sal, hiredate from emp
order by deptno;
```

- BUSINESS INTELLIGENCE /DATA SCIENCE . ARTIFICIAL INTELLIGENCE

```
mysql> select deptno, job, ename, sal, hiredate from emp
  2  order by deptno, job;
```

```
select deptno, job, ename, sal, hiredate from emp
order by deptno desc, job;

select deptno, job, ename, sal, hiredate from emp
order by deptno desc, job desc;
```

```
select ..................
order by country, state, city;

*       no upper limit on the number of columns in ORDER BY
        clause
```

```
*       if you have a large number of rows in the table, and
        a large number of columns in ORDER BY clause, then the
        SELECT statement will be slow, because that much
        sorting has to take place in Server RAM
```

Select deptno , job, ename, sal, hiredate from emp

Where deptno =10

Order by enme;

- SELECT statement execute from top to bottom and left to right
- Where clause has to specified BEFORE the ORDER BY clause
- WHERE clause is used for searching , searching takes place in the DB Server HD
- WHERE clause is used to restrict the rows
- WHERE clause is used to retrieve the rows from DB server HD to Server RAM
- ORDER BY clause sorting takes place in server RAM
- ORDER BY clause is the LAST clause in SELECT statement.

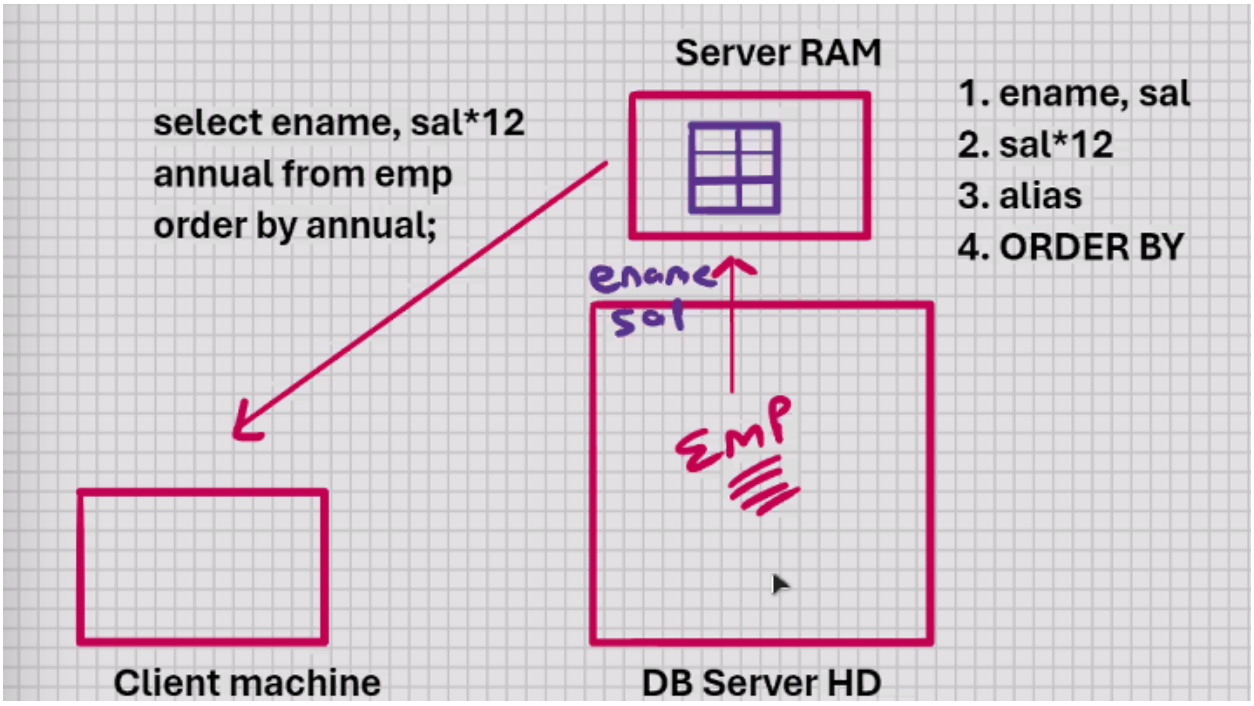
Select ename , sal*12 from emp;

Select ename , sal*12 from emp

Order by sal*12;

Select ename , sal*12 annual from emp

Order by annual;



```
select ename, sal*12 "Annual Salary" from emp
order by "Annual Salary";

select ename, sal*12 "Annual Salary" from emp
order by 2;
```

Select * from emp

Order by 2;

Select * from emp

Order by 1;

		EMP		
EMPNO	ENAME	SAL	CITY	DEPTNO
1	ADAMS	1000	Mumbai	10
2	BLAKE	2000	Delhi	10
3	ALLEN	2500	Mumbai	20
4	KING	3000	Delhi	25
5	FORD	4000	Mumbai	30

Select * from emp

Where ename> ’A’ and ename < ’B’;

		EMP		
EMPNO	ENAME	SAL	CITY	DEPTNO
1	ADAMS	1000	Mumbai	10
2	BLAKE	2000	Delhi	10
3	ALLEN	2500	Mumbai	20
4	KING	3000	Delhi	25
5	FORD	4000	Mumbai	30

```
ename varchar(20)
select * from emp
where ename > 'A' and ename < 'B';
```

ADAMS > AKKKK

ADAMS < BKKKK

Blank padded comparision semantics:-

When you compare 2 string of different length the shorter of 2 string is temporarily padded with blank spaces on RHS such that their length becomes equal then it will start the comparision character by character based on ASCII values.

```
Select * from emp
Where >='A' and ename < 'C'
```

MySql -sql –special operator(LIKE)

Wildcard (used fot pattern matching)

% ----any character and any number of characters

_ any 1 character

```
Select * from emp
Where ename ='A%';
```

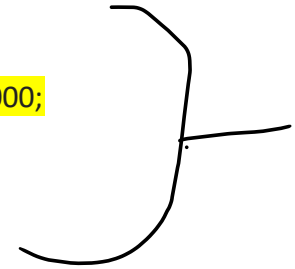
```
Select * from emp
Where ename like 'A%';
```

```
Select * from emp
Where ename like '%A';
```

```
Select * from emp
Where ename like '%A%';
```

```
Select * from emp
Where ename not like '%A%';
```

```
Select * from emp
Where ename like '__A%';

Select * from emp
Where ename like '_____';

Select * from emp
Where ename like '__I____';

Select * from emp
Where sal>=2000 and sal <=3000;
```



```
//between
Select * from emp
Where sal between 2000 and 3000;          <-inclusive
```

```
Select * from emp
Where sal not between 2000 and 3000;          <-excusive

Select * from emp
Where hiredate between '2023-01-01' and '2023-12-31';

Select * from emp
Where ename>='A' and ename <='F';

Select * from emp
Where ename between 'A' and 'F';
```

Select * from emp

Where deptno =10 or deptno =20 or deptno =30; <- most powerful

Select * from emp

Where deptno= any(10,20,30); <- faster

Select * from emp

Where deptno in(10,20,30); <-fastest

```
select * from emp
where deptno in(10,20,30);
select * from emp
where deptno not in(10,20,30);
select * from emp
where deptno =any(10,20,30);
select * from emp
where deptno !=any(10,20,30);
select * from emp
where deptno >any(10,20,30);
select * from emp
where deptno >=any(10,20,30);
select * from emp
where deptno <any(10,20,30);
select * from emp
where deptno <=any(10,20,30);
```

ANY -> logical OR

IN -> logical OR

```
*       ANY operator is overloaded (Operator Overloading)
*       IN operator is faster than ANY operator
*       ANY operator is more powerful than IN operator
*       with the IN operator, you can only check for IN and NOT IN
*       with the ANY operator, you can check for =ANY, !=ANY, >ANY, >=ANY,
        <ANY, <=ANY
*       if you want to check for equality or inequlaity, then use the
        IN operator
*       if you want to check for >, >=, <, or <=, then use the
        ANY operator
*       if your requirements are more complex, then use the
        Relational and Logical operators
```

```
MySQL - SQL - Special Operators (Like, Between, Any, In)


*       IN operator is supported by MySQL
*       ANY operator is supported by MySQL only when it is used with
        sub-query

select * from emp
city in('Mumbai','Delhi');
```

DDL- Create ,Drop

DML – Insert, Update, Delete

DQL -> Select

UPDATE

Update emp

Set sal=10000

Where empno=1;

Update emp

Set sal=sal+ sal*0.4

Where empno=1;

Update emp

Set sal =10000 ,city =’Pune’

Where empno=1;

Update emp

Set sal =10000, city=’Pune’

Where city =’Mumbai’;

```
*       you can UPDATE multiple rows and multiple columns simulataneously,
        but you can UPDATE only 1 table in a single command
*       if you want to UPDATE 2 or more tables, then a separate UPDATE
        command is needed for each table
```

Update emp

Set sal =10000

Where empno=1;

Update emp

Set sal =10000;

Update emp

Set sal=10000

Where empno=101;

DELETE ---------→ dml command

Delete from emp

Where empno=1;

Delete from emp

Where city =’Mumbai’;

Delete from emp;

DROP----------→ DDL command

Drop table emp;

Drop table emp, dept;

You cannot have a WHERE with DROP table (bcz DROP is DDL cmd).

TRANSECTIONAL PROCESSING

- Commit will save all the DML changes since the last commited statement commit work or commit.
- WORK is optional in My SQL
- When the user issues a commit, it is known as End of Transection Commit will make the Transection permanent.

Total work done =t1+t2+t3+t4+……+tn;

- Total work done =sum of work done in individual transections.
- When to issue the commit is decided by the end user , and it depends on the logical scope of work.
- Rollback will undo all the DML changes since the last committed state.
- once is committed cannot be rolled back

  Rollback work;
  Or
  Rollback;

- What is optimal in MySQL---→ROLLBACK
- Only the DML command are affected by Rollback and commit.
- Any DDL command automatically commit; not only will it commit itself, it will commit all DML changes before it.
- When you EXIT from SQL, the system automatically commits.
- Any kind of power failure ,network failure ,system failure, pc reboot, window close, improper exit from SQL , etc ; your last uncommitted Transection is automatically Rolled back.

- Savepoint is a point within your work
- Savepoint is similar to a Bookmark
- Savepoint is similar to a milestone within your Transection
- You can Rollback to a savepoint
- You Cant commit to a savepoint
- Commitwill save all the DML changes since the last committed state.
- When you rollback or commit ,the intermediate savepoint are cleared if you want to use then again, then you will have to reissue them in some new work

Rollback work to pqr;
Or
Rollback to pqr;

- Work is optional in my sql.

- Within a transaction you can have 2 savepoints within the same name.
- The latest savepoint supercedes the previous one.
- The older savepoint no longer exists.

Environment setting :-

Mysql> set commit=1;

Mysql> set commit=0;

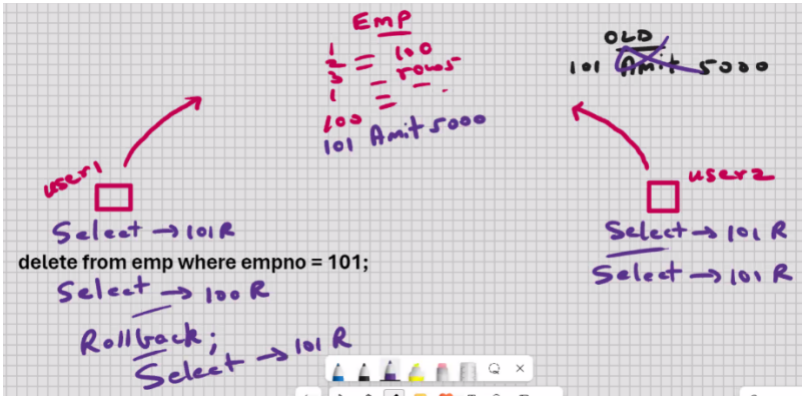
```
MySQL - Read and Write Consistency

*       in a multi-user environment
        when you SELECT from a table you can view
        ONLY the committed data of all users
        plus/union
        changes made by you
```

```
ACID Properties in RDBMS:-

A -> Atomicity
C -> Consistency
I -> Isolation
D -> Durability
```

```
ACID Properties in RDBMS:-

A -> Atomicity
*       the entire transaction takes place at once or doesn't happen at all
        (e.g. You withdraw Rs. 500/- from your Bank account ....)
C -> Consistency
*       the database must be consistent before and after the transaction
        (e.g. Same as above, Deposits/Withdrawals and Balances tables)
I -> Isolation
*       multiple transactions occur independently without interference
D -> Durability
*       the changes of a successful transaction are permanently, even if
        later the system failure occurs
```

Final project guideline

Have multiuser




```
Automatic row locking mechanism in MySQL:-

*       When you UPDATE or DELETE a row, that row is automatically "locked"
        for other users
*       ROW LOCKING IN MySQL IS AUTOMATIC
*       When you UPDATE or DELETE a row, that row becomes READ_ONLY
        for other users
*       Other users can SELECT from that table; they will view the old
        data "before" your changes
*       Other users can INSERT rows into that table
*       Other users can UPDATE or DELETE "other" rows from that table
*       No other user can UPDATE or DELETE your locked row till you have
        issued a Rollback or Commit
*       LOCKS ARE AUTOMATICALLY RELEASED WHEN YOU ROLLBACK OR COMMIT
```

```
OPTIMISTIC LOCKING:-
*       Automatic row locking mechanism in MySQL

PESSIMISTIC LOCKING:-
*       Lock the rows manually in advance "BEFORE" issuing UPDATE or DELETE

To lock the rows manually:-
*       you have to use SELECT statement with the FOR UPDATE clause

mysql> select * from emp
        where empno = 101
        for update;

*       LOCKS ARE AUTOMATICALLY RELEASED WHEN YOU ROLLBACK OR COMMIT
```

```
mysql> select * from emp
        where deptno = 10
        for update;                     <- (by default) it will WAIT
                                        in the Request Queue till the row
                                        becomes available

mysql> select * from emp
        where deptno = 10
        for update nowait;              <- if row is available, then lock
                                        it; else abort the operation

*       LOCKS ARE AUTOMATICALLY RELEASED WHEN YOU ROLLBACK OR COMMIT
```

```
mysql> select * from emp
        where deptno = 10
        for update wait 30;             <- seconds

*       if row is available, then lock it; wait in the Request for the
        specified time period; if row is still unavailable, then abort the
        operation

*       LOCKS ARE AUTOMATICALLY RELEASED WHEN YOU ROLLBACK OR COMMIT
```

Functions


```
MySQL - SQL - Character Functions

select * from emp;
select fname, lname from emp;

select concat(fname,lname) from emp;
```

Output:-

Arun Purun

TarunArun

SirunKirun

NutanPurun

Uses:-

a. For presentation/reporting purposes

Select concat (concate (fname ,' '), lname) from emp;

- Max upto 255 levels for function within function.
  (this limit of SQL can be exceeded with the help of Views)

Select upper(fname) from emp;

OutPut;

ARUN

TARUN

SIRUN

NUTAN

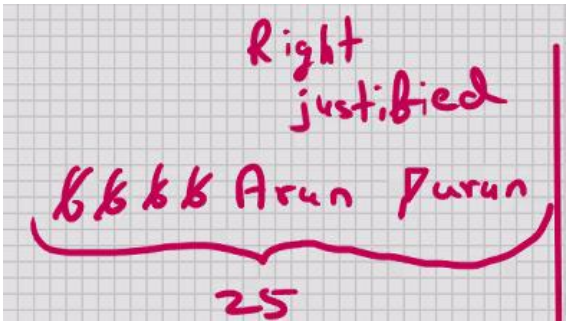
Update emp set fname = upper(fname);

Update emp set fname = upper(fname) where …….;

Select lower(fname) from emp;

Initacap-→ not in sql but implement using upper and lower and concate.

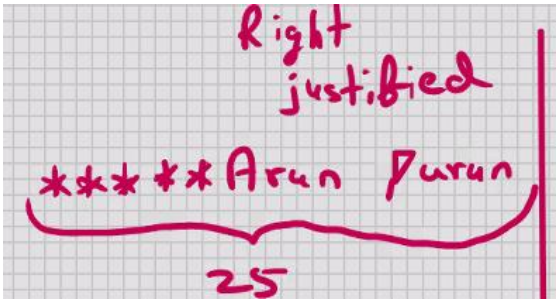
Select lpad(ename,25,’ ’)from emp;



Uses:-

a. Right Justification

sSelect lpad (ename,25,'*') from emp;



Uses:-

a. Right Justificaton
b. Cheque printing

Select rpad(ename,25,' ')from emp;

Uses:-

A. Left justification of numeric data.
B. Covert varchar to char (convert variable length to fixed- length).
C. Cheque printing
D. Center justification

Select lpad(rpad(ename,25.'*'),30,'*') from emp;

Select ltrim (ename) from emp;

Uses:

a. Left justification
   Eg. Rpad(itrim(……..)……);

Select rtrim (ename) from emp;

Uses:-

a. Convert char to varchar (convert fixed-length to variable length)
b. Right justification of char data
   Lpad(rtrim(……..)……);

Select trim(ename) from emp; <- remove blank spaces from both the sodes in Mysql

Select substr(ename,3) from emp;

3→ is starting position

o/p

Un Puran

Run Arun

Run Kirun

tan Puran

Select substr(ename,3,2) from emp;

o/p

Un

Ru

Ru

ta

substr('New Mumbai',1,3); o/p → 'New'

select substr (enum, -3) from emp;

select substr (ename ,-3,2) from emp;

uses:-

a. Used to extract a part of the string

Select replace (ename, 'un' , 'xy') from emp;

Select replace(ename , 'un' , '') from emp;

Select instr(ename,'un') from emp; (instring) <- return starting position

3

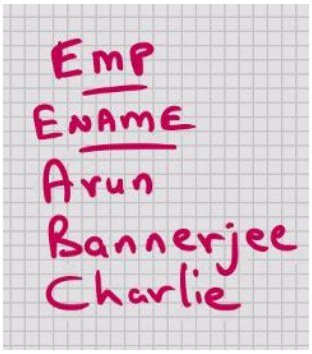
4

4

10

Uses:-

a. Used to check



Select length (ename) from emp;

4

9

7

Select ascii(ename) from emp; return ascii letter of 1st latter

65

66

67

Select ascii (substr(ename,2)) from emp;

Ascii value 'z'→ 122

Select ascii('z') from emp;

o/p:-

122

122

122

Select distinct ascii('z') from emp;

o/p

122

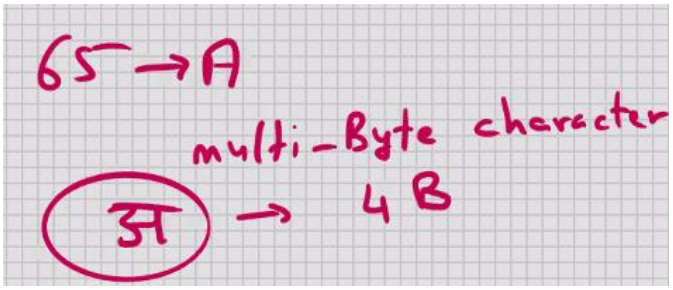
Select ascii('z') from dual;

122

- Dual is a system table
- It is automatically created when you install MySql
- Commom for all RDBMS
- DUAL contains inly 1 row and 1 column
- Dual is a dummy tables

```
select substr('New Mumbai',1,3) from dual;
select 3*12 from dual;
select 'Welcome to CDAC Mumbai' as "MESSAGE" from dual;
```

Select char(65 using utf8) from dual;



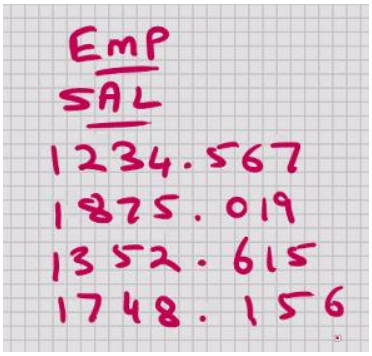
--→ where utf8 is the given character set for U.S. English ,else default is binary character set.

Project Guideline

Atleast one html page in hindi, marathi.

Select * from emp where soundex(ename)=soundex("Aruun");

Number Function



Select round(sal) from emp;

o/p

1235

1875

Etc.

Select round(sal,1) from emp;

o/p

1234.6

1875

1352.6

1748.2

Select round(sal,-2) from emp;

1200

1900

1400

1700

Select truncate (sal,0) from emp;

1234

1875

1352

1748

Uses:-

a. Age calc
b. Time calc

Select truncate (sal,1) from emp;

1234.5

1875

1352.6

1748.1

Select truncate (sal,-2) from emp;

1200

1800

1300

1700

Ceil-→ceiling

Select ceil(sal) from emp; <- if there is any value at all in the decimal it will add 1 to the whole number

1235

1876

1353

1749

Uses:-

a. Bill payment
b. Intrest payments
c. EMI payment

Select floor(sal) from emp;

1234

1875

1352

1748

Select truncate (3.6,0),floor (3.6),truncate(-3.6,0), floor(-3.6) from dual;

3 3 -3 -4

Round ,truncate ,ceil,floor

Select sign(-15) from dual; return sign of number

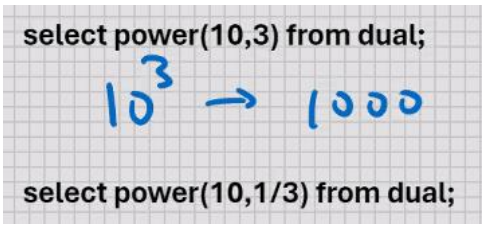
-1

Use:-

```
Uses:-
a. Check if number is positive or negative
b. sign(temperature)
c. sign(blood_group)
d. sign(sensex)
e. sign(time1 - time2)
f. sign(qty_in_stock)
g. sign(dist1 - dist2)
h. sign(acccceleration)
i. sign(customertraffic)
j. sign(velocity1 - velocity2)
k. sign(SP - CP)
```

Select mod(9,5) from dual;

Select power (10,3) from dual;



Select abs(-10) from dual;

o/p

10

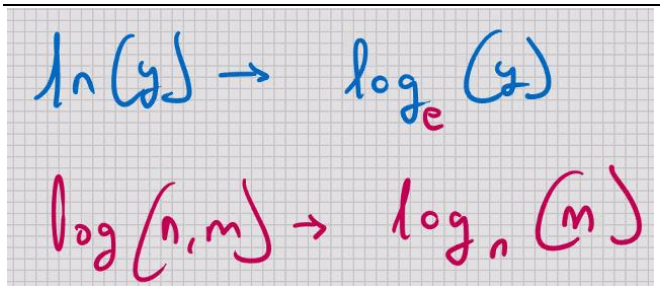
---

x-> radians

Sin(x)

Cos(x)

Tan(x)



Date and Time Function and Formats

```
MySQL - Date and Time Functions and Formats

1. Date, Time, Datetime, Year
2. 1st Jan 1000 AD to 31st Dec 9999 AD
3. No problem of Y2k
4. '-838:59:59' to '838:59:59'
5. 'YYYY-MM-DD' is the default date format in MySQL
6. 'YYYY-MM-DD' or 'YY-MM-DD'
7. 1970 is the year of the Epoch
8. 00:00:00 (12 am midnight is the default value for time)
9. date1-date2, time1-time2, datetime1-datetime2
10. Internally date is stored as a fixed-length number and it occupies
7 Bytes of storage
```

Select sysdate() from dual; <return date and time when the statement executed

2024-10-11 20:11:41

- Return DB Server machine date and time

Select now() from dual; <return date and time when the statement began to execute

2024-10-11 20:11:41

- Return DB Server machine date and time

Select sysdate(),now(), sleep(10), sysdate() ,now() from dual;

Sysdate()

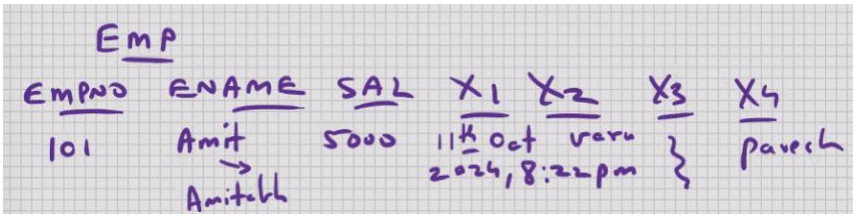
Uses;

A Date and time displays

Now()

Uses:-

A Maintain logs of user actions/operations.




```
select adddate(sysdate(),1) from dual;

2024-10-12 20:28:54

select adddate(sysdate(),1) from dual;
select adddate(sysdate(),2) from dual;
select adddate(sysdate(),7) from dual;
select adddate(sysdate(),-1) from dual;
```

```
select datediff(sysdate(),hiredate) from emp;

371
295
282
```

Return number of calender days between the two

- Decimal above is the remainder hours , minutes and seconds as a fraction of day

```
select date_add(hiredate, interval 2 month) from emp;

2023-12-15
2024-02-29
2024-03-15

select date_add(hiredate, interval -2 month) from emp;

select date_add(hiredate, interval 1 year) from emp;
select date_add(hiredate, interval -1 year) from emp;

select adddate(sysdate(),1/(24*60*60)) from dual;
```

Select last_day (hiredate) from emp;

Uses:-

a. Attendence_calc
b. Interest_calc

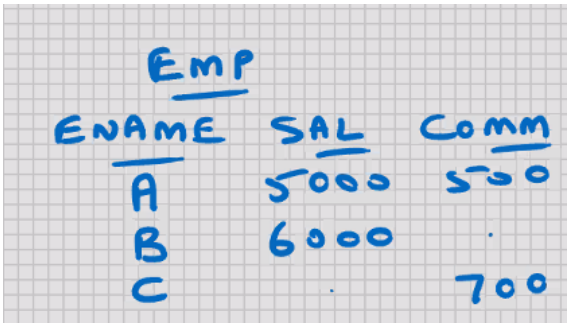
```
select last_day('2024-10-11') from dual;

select dayname(sysdate()) from dual;

Friday

select upper(dayname(sysdate())) from dual;

select substr(dayname(sysdate()),1,3) from dual;

select addtime('2010-01-15 10:00:00','1') from dual;
```

```
select addtime('2010-01-15 10:00:00','1') from dual;

2010-01-15 10:00:01
```

```
select addtime('2010-01-15 10:00:00','-1') from dual;

select addtime('2010-01-15 10:00:00','1:30:45') from dual;

2010-01-15 11:30:45
```

List Function

Independent of datatype



Select * from emp where comm=null; // not work

- Any comparision done with null ,return null.

Pessimistic Querying→ searching for null values.

Select * from emp where com is null;

Special operator→ is null.

Select * from emp where comm !=null; // not work

- Any comparition done with null, return null.

Select * from emp where comm is not null;

Select sal + ifnull(comm,0) from emp; if comm is null then return 0;

o/p else return comm;

5500 end if;

6000

Select if null(sal,0) + if null(comm,0) from emp;

5500

6000

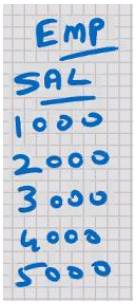
700

If null(comm,0);

If null(comm,100);

If null(city,'mumbai');

If null(orderdate,'2024-04-01');



Select greatest (sal,3000) from emp; → return greater of the two

o/p

3000

3000

3000

4000

5000

Uses:-

a. To set a lower limit on some value.
   Eg:-
   Bonus=10% sal,min bonus=300
   Select greatest (sal*0.1,300) from emp;

Select greatest (sal,col1,col2,col3,col4) from emp;

```
greatest(val1,val2,val3,....,val255)
greatest(col1,col2,col3,....,col255)
greatest('str1','str2','str3')
greatest('date1','date2','date3','date4')
greatest('time1','time2','time3')
greatest('datetime1','datetime2','datetime3')
```

Set x=greatest(a,b,c,d);

Select least(sal,3000) from emp;

1000

2000

3000

3000

3000

Use:-

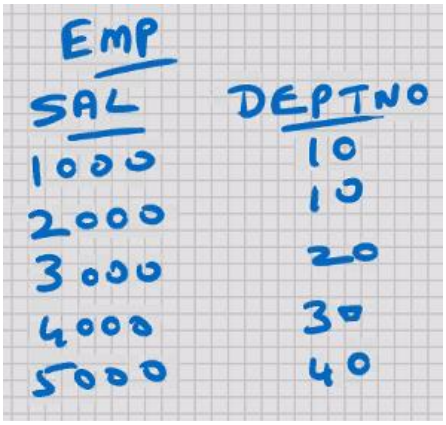
a. Used to set an upper limt on some value
   Eg:-
   Castback=10% amt,max cashback=300

Select least(amt*0.1,300) from orders;

Select least(sal,3000,4000,5000,10000) fom emp;

```
least(col1,col2,col3,....,col255)
least(val1,val2,val3,....,val255)
least(num1,num2,num3)
least('str1','str2','str3')
least('date1','date2','date3')
```

Set x=least(a,b,c,d);



Select

When deptno=10 then 'Training'

When deptno=20 then 'Exports'

When 'deptno=30' then 'Marketing'

Else 'Others'

End 'DEPTNAME'

From emp;

o/p:-

DEPTNAM

---

traning

training

exports

marketings

.

- If you don't specify else ,and if Deptno 40 is present in the table ,then it returns a null value

```
select
case
when deptno = 10 then 'Ten'
when deptno = 20 then 'Twenty'
when deptno = 30 then 'Thirty'
when deptno = 40 then 'Forty'
end "DEPTNUMBER"
from emp;
```

Problem is

```
if deptno = 10 then HRA = 40% annual
if deptno = 20 then HRA = 30% annual
if deptno = 30 then HRA = 25% annual
else HRA = 20% annual

select deptno, ename, sal, sal*12 "ANNUAL",
case
when deptno = 10 then sal*12*0.4
when deptno = 20 then sal*12*0.3
when deptno = 30 then sal*12*0.25
else sal*12*0.2
end "HRA"
from emp
order by 1;
```

order by 1 means coloumn 1 se order karega

Problem

```
if sal < 3000, REMARK = 'Low Income'
if sal = 3000, REMARK = 'Middle Income'
if sal > 3000, REMARK = 'High Income'

select ename, sal,
case
when sign(sal-3000) = 1 then 'High Income'
when sign(sal-3000) = -1 then 'Low Income'
else 'Middle Income'
end "REMARKS"
from emp
order by 2;
```

Environment Function

Mqsql > use cdac Mumbai;

Mysql > select database() from dual;

Mysql> Select user() from dual;

```
*       Have 6 extra columns in every table
*       username, date and time when the row was inserted
*       username, date and time when the row was udpated

user(), sysdate(), now()
```

Show character set; -→what languages are available in your current installation

Group/Aggregate Function

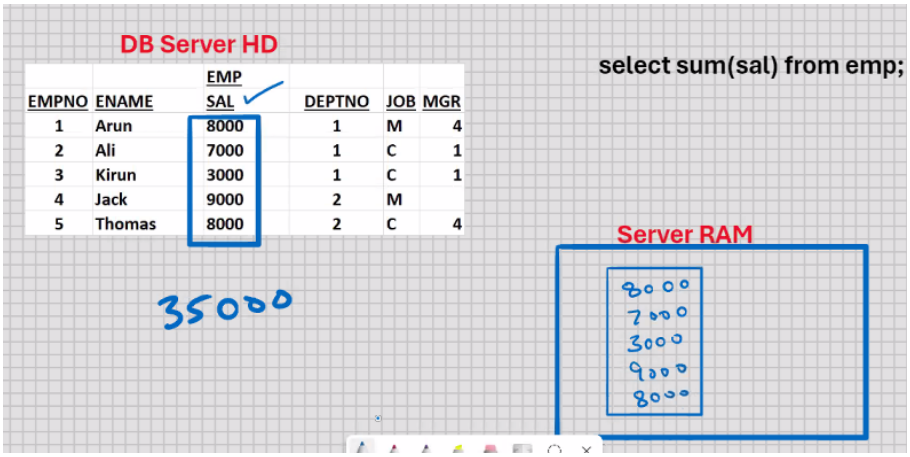
		EMP			
EMPNO	ENAME	SAL	DEPTNO	JOB	MGR
1	Arun	8000	1	M	4
2	Ali	7000	1	C	1
3	Kirun	3000	1	C	1
4	Jack	9000	2	M	
5	Thomas	8000	2	C	4

```
select
case
when job = 'M' then 'MANAGER'
when job = 'C' then 'CLERK'
end "JOB"
from emp;
```

```
Single-Row Functions:-
*       operate on 1 row at a time
*       Character, Number, Date and Time, List, Environment Functions
*       e.g. upper (ename), round(sal), etc.

Multi-Row Functions:-
*       operate on multiple rows at a time
*       Group Functions
*       e.g. sum(sal)
```

Select sum(sal) from emp;

How this statement work;



- Assumption ,4th row SAL is null:-
- Select sum(sal) from emp;
- 26000
- Null values are not counted by group funtions.
- Select sum(ifnull (sal,0)) from emp;
- 26000
- .
- Select avg(sal) from emp;
- 26000/4➔ 6200 null values are not counted by group function
- Select avg(ifnull (sal,0)) from emp;
- 26000/5➔5200
- ----------
- Select min(sal) from emp;
- 3000
- Select min(ifnull(sal,0))from emp;
- 0
- Select max(sal) from emp;
- 8000
- Select count(sal) from emp; ← return the count of number of rows where sal is not having null value
- 4
- Select count(*) from emp;← return a count of total number of row in the table.
- 5
-
```
select count(*) from emp;
select count(sal) from emp;
select count(*) - count(sal) from emp;
```
- Select max (sal)/min(sal) from emp;
- 8000/3000➔2.67
- ----------

- Select sum(sal)/count(*) from emp; ←----faster
- 26000/5
- Select avg(ifnull(sal,0)) from emp; ←----slow
- 26000/5

- Assumption 4th row sal is 9000:-

- Select sum(sal) from emp
- Where deptno=1;
- 18000
- Select avg(sal) from emp
- Where job='C';
- 6000
- Count -Query(counting the number of query hits):-
- Select count(*)from emp
- Where sal>7000;

```
sum(column)
avg(column)
min(column)
max(column)
count(column)
count(*)
stddev(column)
variance(column)
```

- Select count(*),min(sal),max(sal),sum(sal),avg(sal), from emp;

```
select count(*), min(sal), max(sal), sum(sal), avg(sal) from emp;

COUNT(*)   MIN(SAL)   MAX(SAL)   SUM(SAL)   AVG(SAL)
---------- ---------- ---------- ---------- ----------
    14         800        5000      29025       2073
```

- Known as summay :- SUMMRY REPORT-

3 restriction on group functions

```
select count(ename), max(sal) from emp;

select ename, max(sal) from emp;
```
<- ERROR

Restriction #1

You cannot select column of table along with the group function.

Restriction #2

```
select count(ename), max(sal) from emp;

select upper(ename), max(sal) from emp; <- ERROR
```

- You cannot select a single- row function alongwith a group function.

Restriction #3

```
Restriction #3:-
*       you cannot use Group function in the WHERE clause

select * from emp where sal > avg(sal);          <- ERROR
```

Group by clause (V.IMP)

- Used for grouping

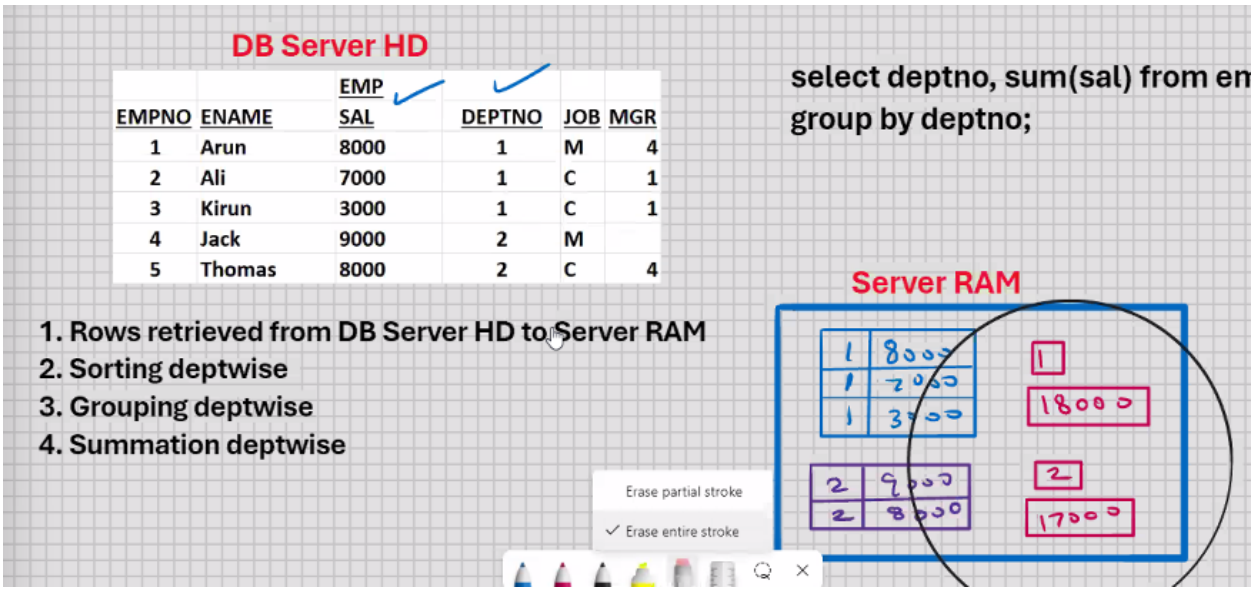
```
WHERE clause -> used for searching
ORDER BY clause -> used for sorting
FOR UPDATE clause -> used for locking the rows manually
GROUP BY clause -> used for grouping
```

```
select sum(sal) from emp;

35000

select sum(sal) from emp
where deptno = 1;
```

- 18000
- Want this:-

```
sum(sal) deptwise:-

deptno  sum(sal)
------  --------
1       18000
2       17000
```

- Select deptno , sum(sal) from emp group by deptno;




```
SELECT clause -> select deptno, sum(sal)
FROM clause -> from emp
GROUP BY clause -> group by deptno

select deptno, sum(sal) from emp
group by deptno;
```

Restrictions:-

Rule#1:-

```
Rule #1:-
*       Besides the Group function, Whichever column is present in SELECT
        clause, it has to be present in GROUP BY clause

select deptno, sum(sal) from emp;       <- ERROR
```

```
mysql> select deptno, sum(sal) from emp;
select deptno, sum(sal) from emp
       *
ERROR at line 1:
ORA-00937: not a single-group group function
Help: https://docs.oracle.com/error-help/db/ora-00937/


mysql> select deptno, sum(sal) from emp
  2  group by deptno;

    DEPTNO   SUM(SAL)
---------- ----------
        10       8750
        30       9400
        20      10875

mysql>
```

RULE #2

```
Rule #2:-
*       Whichever column is present in GROUP BY clause, it may or may not
        be present in SELECT clause

select sum(sal) from emp
group by deptno;

sum(sal)
--------
18000
17000
```

```
select deptno, sum(sal) from emp
group by deptno;

select deptno, max(sal) from emp
group by deptno;

select deptno, min(sal) from emp
group by deptno;

select deptno, avg(sal) from emp
group by deptno;

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

- 2D query→ Any select statement with a group by clause is know as a 2d query because you can plot a graph from the output.
- MS Excel, Oracle Graphics ,Oracle Apex etc.

```
select deptno, sum(sal) from emp
where sal > 7000                          <- used to retrieve the rows
group by deptno;

*       WHERE clause is specified BEFORE the GROUP BY clause
```

```
select deptno, sum(sal) from emp
where sal > 7000                          <- used to retrieve the rows
group by deptno;

*       WHERE clause is specified BEFORE the GROUP BY clause

select deptno, sum(sal) from emp
where sal > 7000 and sal < 10000
group by deptno;

select deptno, sum(sal) from emp
where job = 'C'
group by deptno;
```

```
select job, sum(sal) from emp
group by deptno;
```

```
select deptno, job, sum(sal) from emp
group by deptno, job;
```

- There is no upper limit on the number of column in group by clause
  Select…….
  Group by city ,state,city;

```
*       if you have a large number of rows in the table, and if you have
        a large number of columns in GROUP BY clause, then the SELECT
        statement will be very slow, because that much sorting has to take
        place in Server RAM
```

```
if you have 1 column in GROUP BY clause -> 2D query
if you have 2 columns in GROUP BY clause -> 3D query
if you have 3 columns in GROUP BY clause -> 4D query
etc.
known as Multi-dimensional queries (also known as Spatial queries)
```

```
select deptno, job, sum(sal) from emp
group by deptno, job;

select job, deptno, sum(sal) from emp
group by job, deptno;
```

Select ……..

Group by city, country, district, state; <- slow

Select ……..

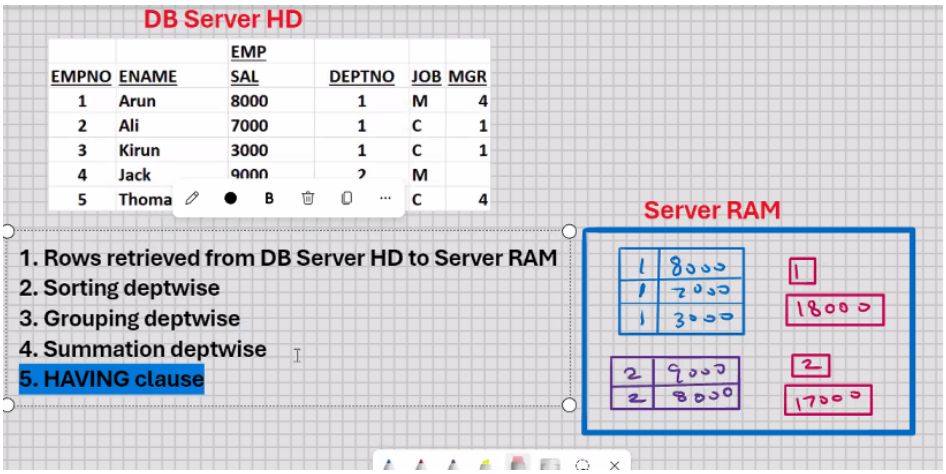
Group by country, state, district , city; <- fast

Select sum(sal) ,deptno, job from emp

Groupby job,deptno;

```
*       the position of columns in SELECT clause and the position of
        columns in GROUP BY clause need not be the same
*       the position of columns in SELECT clause will determine the
        position of columns in the output; this you will write as per User
        requirements
*       the position of columns in GROUP BY clause will determine the
        sorting order, grouping order, summation order, and hence the
        speed of processing; this you will write as per
        count(distinct(columname))
```

Select count(distinct deptno),count(distinct job) from emp;



```
select deptno, sum(sal) from emp
group by deptno;

deptno  sum(sal)
------  --------
1       18000
2       17000

select deptno, sum(sal) from emp
group by deptno
having sum(sal) > 17000;

deptno  sum(sal)
------  --------
1       18000
```

```
*       WHERE clause is used for searching
*       searching takes place in DB Server HD
*       WHERE clause is used to restrict the rows
*       WHERE clause is used to retrieve the rows from DB Server HD to
        Server RAM
*       HAVING clause works AFTER the summation is done
```

```
select deptno, sum(sal) from emp
group by deptno
having sal > 7000;      <- ERROR
```

```
select deptno, sum(sal) from emp
group by deptno
having deptno = 1;              <- WILL WORK BUT IT IS INEFFICIENT,
                                   INSTEAD SHOULD HAVE USED WHERE DEPTNO = 1

*       It's recommended that only the Group functions should be used
        in HAVING clause

select deptno, sum(sal) from emp
group by deptno
having sum(sal) > 17000;

select deptno, sum(sal) from emp
group by deptno
having sum(sal) > 17000 and sum(sal) < 25000;
```

```
select deptno, sum(sal) from emp
group by deptno
order by 2;

deptno  sum(sal)
------  --------
2       17000
1       18000
```

1. Rows retrieved from DB Server HD to Server RAM
2. Sorting deptwise
3. Grouping deptwise
4. Summation deptwise
5. HAVING clause
6. ORDER BY clause

- Order by clause is the last clause in SELECT statement in select statement

```
select .......... from ..........
where ..........
group by ..........
having ..........
order by ..........;
```

Matrix report

```
mysql> select deptno, count(*), min(sal), max(sal), sum(sal) from emp
  2  group by deptno
  3  order by 1;

    DEPTNO   COUNT(*)   MIN(SAL)   MAX(SAL)   SUM(SAL)
---------- ---------- ---------- ---------- ----------
        10          3       1300       5000       8750
        20          5        800       3000      10875
        30          6        950       2850       9400

mysql>
```

NOT WORK IN MYSQL

```
In Other RDBMS:-

select deptno, sum(sal) from emp
group by deptno;

deptno  sum(sal)
------  --------
1       18000
2       17000

select sum(sal) from emp
group by deptno;

sum(sal)
--------
18000
17000
```

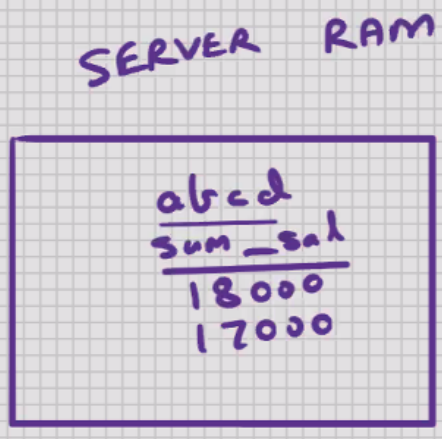
```
sum(sal)
--------
18000
17000

select max(sum(sal)) from emp          <- NESTING OF GROUP FUNCTIONS IS
group by deptno;                       SUPPORTED BY ORACLE

max(sum(sal))
-------------
18000
```

In MYSQL

```
In MySQL:-

select sum(sal) sum_sal from emp
group by deptno;

sum_sal
-------
18000
17000

select max(sum_sal) from
(select sum(sal) sum_sal from emp
group by deptno) abcd;

max(sum_sal)
------------
18000
```

EMP

EMPNO	ENAME	SAL	DEPTNO	JOB	MGR
1	Arun	8000	1	M	4
2	Ali	7000	1	C	1
3	Kirun	3000	1	C	1
4	Jack	9000	2	M	
5	Thomas	8000	2	C	4



```
select max(sum_sal) from
(select sum(sal) sum_sal from emp
group by deptno) abcd;

max(sum_sal)
------------
18000
```

JOINS

```
MySQL - SQL - JOINS (V. Imp)
*       all the data is stored in one table, data is stored in multiple
        tables
*       if you want to see the columns of 2 or more tables, then you will
        have to write a join
```

		EMP			
EMPNO	ENAME	SAL	DEPTNO	JOB	MGR
1	Arun	8000	1	M	4
2	Ali	7000	1	C	1
3	Kirun	3000	1	C	1
4	Jack	9000	2	M	
5	Thomas	8000	2	C	4

	DEPT	
DEPTNO	DNAME	LOC
1	TRN	Bby
2	EXP	Dlh
3	MKTG	Cal

Select dname, ename from emp, dept

Where dept.deptno=emp.deptno;

Dept.deptno→tablename.columnname

Emp.deptno→ tablename.columnname

Select dname, ename from emp, dept

Where dept.deptno=emp.deptno;

Dept→ driving tables

Emp→driven table

```
*       the position of columns in SELECT clause will
        determine the position of columns in the output;
        this you will write as per Client requirements

select ename, ename from emp, dept
where dept.deptno = emp.deptno;
-----------------------------------------------------------------
```

```
*       WHERE clause can be written in any sequence

select dname, ename from emp, dept
where dept.deptno = emp.deptno;
OR
select dname, ename from emp, dept
where emp.deptno = dept.deptno;
-----------------------------------------------------------------
```

```
dept -> driving table
emp -> driven table

                          <<-----
select dname, ename from emp, dept              <- FAST
where dept.deptno = emp.deptno;
-----------------------------------------------

emp -> driving table
dept -> driven table
                          <<-----
select dname, ename from dept, emp              <- SLOW
where dept.deptno = emp.deptno;

*       IN ORDER FOR THE JOIN TO WORK FASTER, PREFERABLY THE DRIVING TABLE
        SHOULD BE TABLE WITH THE LESSER NUMBER OF ROWS
```

```
*       the common column in both the tables is DEPTNO; the common column
        that is present in both the tables, the columnname need not be
        the same in both the tables, because the same column may have a
        different meaning in the other table
        e.g. EXPORT and IMPORT, PUCHASE and SALE, etc.

select dname, ename from emp, dept
where dept.x = emp.y;
```

```
*       the datatype of common column in both the tables should be the
        same and there has to be some logical reason on whose basis you
        are writing the join

select dname, ename from emp, dept
where dept.deptno = emp.empno;
```

```
select dname, ename from emp, dept
where dept.deptno = emp.deptno;
-----------------------------------------------

To make the output more presentable:-

select dname, ename from emp, dept
where dept.deptno = emp.deptno
order by 1;
-------------------------------------------------------------
```

```
*       It's good to display lots of columns in the output; it
        becomes richer and more meaningful

select dname, loc, ename, job, sal from emp, dept
where dept.deptno = emp.deptno
order by 1;
-------------------------------------------------------------
```

```
*       to display all the columns from both the tables

select * from emp, dept
where dept.deptno = emp.deptno
order by 1;
-----------------------------------------------------------------------
```

```
*       column ambiguously defined (if the common column had different
        column names in both the tables, then this problem would not
        have arisen)

        (*)
select deptno, dname, loc, ename, job, sal from emp, dept
where dept.deptno = emp.deptno
order by 1;
-------------------------------------------------
```

```
select dept.deptno, dname, loc, ename, job, sal from emp, dept
where dept.deptno = emp.deptno
order by 1;
         OR
select emp.deptno, dname, loc, ename, job, sal from emp, dept
where dept.deptno = emp.deptno
order by 1;
-----------------------------------------------
```

```
*       it's a good programming practice to use tablename.columnname for
        all the columns, becuase it makes the SELECT statement more
        Readble
select dept.deptno, dept.dname, dept.loc, emp.ename, emp.job, emp.sal
from emp, dept
where dept.deptno = emp.deptno
order by 1;
```