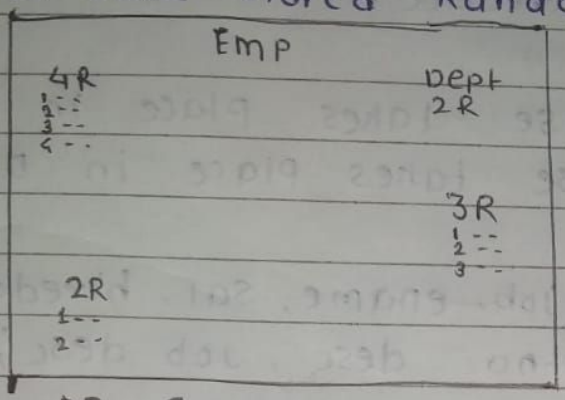
desc <tablename>;

e.g. desc emp

Date :- 22 Nov 2021

In DBMS Data stored sequentially.

In RDBMS data stored Randomly mixed with another data.



DB Server HD

* in RDBMS, table is not a file; every row is a file.
* in RDBMS, the rows of table are not stored sequentially; the rows of table are scattered all over the DB server HD.
* When you insert a row into table, wherever it find free space in DB server HD, it will store row there.
* When you UPDATE a row, if a row length is increasing row address may change

* later When you select from table you will see rows in some other order in output
* ORDER BY clause :→ (used for sorting)

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

order by ename; (by name)

order by ename desc; (descending)
asc -> by default

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

order by clause takes place in ServerRAM.
Where clause takes place in DB server HD.

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

* no upper limit on the number of columns in ORDER by clause.
* the Where clause is specified before the orderby clause

```
Select deptno, Job, ename, sal, hiredate from emp
Where deptno = 10
order by ename;
```

* ORDER BY clause is LAST clause in select statement

select ename, sal * 12 annual from emp order by annual;

in server RAM
1) ename
2) array create (2-D)
3) for loop sal * 12
4) annual = alias for sal * 12
5) order by annual; → ok

select ename, sal * 12 "Annual Salary" from emp order by 2; (2 is column no in select statement)

* New emp Table

EMP

EMPNO	ENAME	SAL	CITY	DEPT NO
1	ADAMS	1000	Mumbai	10
2	BLAKE	2000	Delhi	10
3	ALLEN	2500	Mumbai	20
4	KING	3000	Delhi	25
5	FORD	4000	Kolhapur	30

select * from emp
where ename > 'A' and ename < 'B'

* Blank - Padded comparision Semantics :-

When you compare 2 strings of different lengths the shorter of the 2 strings temporiarily padded on RHS with blank spaces such that their lengths are equal & then start comparision character by character based on ASCII value

select * from emp where ename >= 'A' and ename < 'B'

* Special operators (Like)

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

% :→ any character and any number of characters (could be 0 or characters)

Solution for case-insensitive Query in oracle
Where ename like 'A%' or ename like 'a%';

WildCards used for pattern matching
Select * from emp where ename = 'A%'
Select * from emp
Where ename like 'A%'; -- (Starting with A)
Where ename like '%A'; -- (ending with A)
Where ename like '%A%'; -- (containing A
Where ename like '__A%'; -- (Returns value containing A as 3rd letter)
Where ename like '____'; (Return value containing 4 letters)

'_' :→ any 1 character

* Select * from emp
Where ename not like 'A%' [Returns value not start with A]

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

Page No.
Date.

Between :->

Select * from emp where sal between 2000 and 3000;
(includes both values 2000 and 3000 [mutually inclusive])

* easier to Write
* Work faster
* A Readymade method by the name between is already present in database in compiled format the Plan etc is ready it directly executes

Select * from emp
where sal not between 2000 and 3000 [exclusive]
<- Recommanded to use

Select * from emp
1) Where hiredate between '2020-01-01' and '2020-12-31';
2) Where hiredate ~~between~~ >= '2020-01-01 and hiredata <= '2020-12-31';

First statement is faster as compared to second.

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

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

* Select * from emp
Where deptno = 10 or deptno = 20 or deptno = 30;
Where deptno = any (10, 20, 30) --> FASTER

ANY -> Logical OR
IN -> Logical OR

```
Select * from emp
Where deptno in (10, 20, 30);  <- FASTEST
```

IN operator is faster than ANY operator.
ANY operator is more powerful than IN operator.

* With IN You can check for IN and NOT IN Whereas With ANY, You can check for =ANY, !=ANY, >ANY, <ANY, >=ANY, <= ANY

* If You want to check for equality or inequality then use IN operator
* if You want to use <, >, <=, >= then use ANY operator.

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

* IN operator is supported by oracle and mysql.
* ANY operator is supported by ORACLE directly but not supported by MYSQL directly.

```
select * from emp
Where deptno = any (10,20);  <- Not Supported.
```

* ANY operator Works in MYSQL Provided it is used with Sub-query

* in MYSQL, You will have to use IN operator.

DDL -> create, drop
DML -> insert, update, delete
DQL -> select *, Select col1, col2 ---, where clause, Relational, Logical, Arithmetic operator, Computed column, Alias, Distinct, how rows are scattered in the DB server HD. ORDER by clause, asc / desc, string comparision, special operators

* UPDATE

```
update emp
    set sal= 10000          (column)
      where empno= 1;
```

```
update emp
    set sal = sal + sal* 0.4   (increase by 40%)
      where empno= 1;
```

```
update emp  set sal = 10000, city= 'pune'
      where empno = 1;  (update multiple
                         columns)
```

```
update emp  set sal= 10000, city= 'Nasik'
      where city = 'Mumbai';
```

(change salary as 10000 and city as Nasik at mumbai (all) locations)

* You can update multiple rows and multiple columns simultaneously but only 1 Table at a time
* Separate update command would be needed for every table.

```
update emp set sal = 10000
```
(operation on whole table)

:→ DELETE

```
delete from emp
       where empno = 1;
delete from emp
       where city = 'Mumbai';
delete from emp; --- (all rows delete but table remain)
```

:→ DROP

```
drop table emp
drop table emp, dept, customers; (delete multiple tables)
```

* You cannot specify WHERE clause with DROP Table.

:→ TRANSACTION PROCESSING

* Commit will save all the DML changes since the last commited stage
* When user issues a commit it is known as it is end of Transaction
* Commit will make the transaction Permanent

Commit Work; / Commit;
Work :→ optional in MYSQL and ORACLE
Work :→ ANSI SQL

Total Work done
$T_1 + T_2 + T_3 - - - - T_n$;
* Transaction is a unit of Work
* When to issue a commit, it depends on logical scope of Project (to be decided by user)

:→ ROLL BACK:-

rollback Work;
Rollback will undo the DML changes since the last commited state
* only DML commands are affected by rollback and commit
* any DDL command, it automatically commits
* When you exit from SQL* Plus (oracle client s/w) it automatically commits.
* Any kind of Power failure, network failure

Pc reboot, Window close, end of task etc in all such cases your last uncommited transaction, it automatically rolled back in mysql and oracle.

* in MySQL UPDATE and DELETE commands without WHERE clause will not be allowed in mysql Workbench
* To Try above commands
1) click on Edit (Menu at top)
2) Preferances -> SQL Editor
3) "safe updates" (checkbox)
4) uncheck it and click ok.
* To Reconnect without Logout
5) click on Query (menu at top)
6) Reconnect to server

* Savepoint:-
  * You can rollback to savepoint
  * Savepoint is a point within the work. (it is similar to bookmark)
  * savepoint is a sub-unit of Transaction

```
rollback Work to abc; [ both are fine
rollback to abc;                      ]
```

* When you rollback or commit, the intermediate savepoints will be cleared. If you want to use it again then reissue in another work.

Page No.
Date.

* You cannot commit to savepoint
* Commit will save all DML changes since last commited state
* For savepoint name (Max 30 characters)
* Within a transaction We can have same savepoint name but if we roll back it then newer savepoint overwrites older ; the older one no longer exists

* To try Rollback, comit & savepoint in MYSQL
Click on Query (Menu at top)
-) Auto comit transaction -) uncheck it