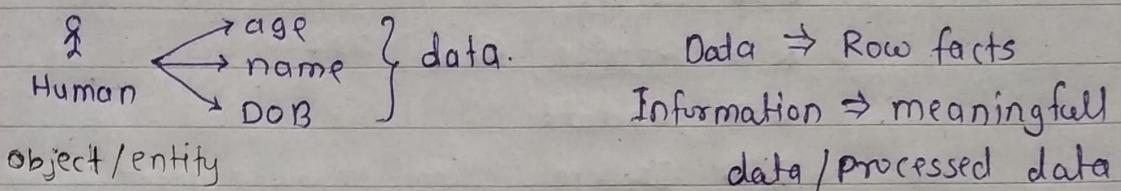


SQL Structured Query Language

- * **Data** :- Data is a row fact which describe the attribute of an entity. Piece of a information



- * **Information** :- Processed data is called information
eg. Qspider is a software training institute.

Data is classified into two types →

- ① Structured data ② Unstructured data

① **Structured Data** :- Data which is arranged in a structured manner/way.

eg. Data stored in a table.

Rows →				Column / Attribute / Fields
Tuples /	Records			
		cell	cell	cell
		first	second	third

② **Unstructured Data** :- Data which is not arranged in a specific structure.

eg. Data store in PDF, TextFile, word file.

- * **Database** :- Database is a place where we can store the data in organised way for data retrieval. (centralized location).

* DBMS :- Database Management System.

It is a system which is used to interact with the database using query language for data manipulation.

CRUD operation

C → Create

R → Retrive / Read

U → Update

D → Delete

* Data Manipulation :-

Create, Insert, Update, delete

DBMS is classified into four types :-

① Flat file DBMS

② Hierarchical DBMS

③ Network DBMS

④ RDBMS

① Flat file DBMS :- Here data will be stored in the form of files and folders.

Advantage - Simple to store data

Disadvantage - Duplication of data.

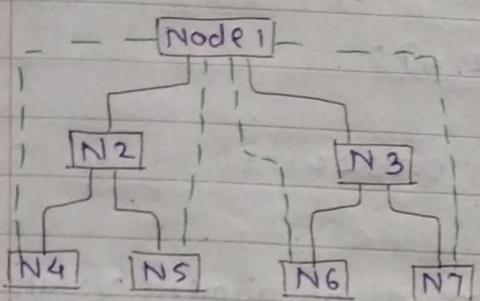
e.g. Data stored on drive of a computer.

② Hierarchical DBMS :- Here data will be stored in the form of parent-child relation or in a tree structure.

Advantage - Data duplication not allowed

Disadvantage - Retrieval time consuming

③ Network DBMS :- Here data will be stored in a new structure. All the nodes (data) are interlinked with another data.



Advantages - Retrieval data easy
Disadvantage - Complex

④ RDBMS :- Here data will be stored in the form of rows and columns. i.e. in a table structure.

- In RDBMS to perform some operations it should be done by using their relations.

* Data Integrity :- It is used to get accuracy, correctness and consistency of the column of the table and it can be achieved by datatype and constraints.

1) Datatypes :- It restricts particular type of data to be stored in the column of a table.

Types \Rightarrow ① Number [int
 float]

② char

③ Varchar

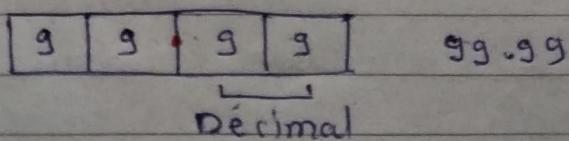
④ Date

① Number :- used to store numeric values/data.

e.g. 1) ID Number (2); - 99 to 99

It can store integers as well as float values.

case 2) ID Number (4,2);



case 3) ID Number (3,3);

1	2	3	0.123
---	---	---	-------

case 4) ID Number (2,4);

0	0	0	2	4	0.0024
---	---	---	---	---	--------

(B) Char :- Used to store alpha numeric values / data
eg. Name char (10);

V	A	S	U	D	H	A		
---	---	---	---	---	---	---	--	--

wasted memory block

Size of char is 2000

(C) Varchar :- varchar is used to store alpha numerical values / data.
eg. Name varchar (10);

V	A	S	U	D	H	A		
---	---	---	---	---	---	---	--	--

we can reuse these memory block.

Reuse / Free memory space

Size of varchar is 4000

⇒ Difference between char & varchar

char

varchar

- | | |
|--|--|
| 1) It's fixed length memory allocation | 1) It's variable length memory allocation |
| 2) Unused space will be wasted | 2) Unused spaces will be release / free |
| 3) In unused spaces blank space will be stored | 3) In unused space null will be stored. |
| 4) Size 2000 | 4) Size 4000 (varchar & varchar2 are same) |

① Date :- By default date format is -

DD - MON - YY

* Constraints :-

Constraints are the conditions we apply on a column of a table.

following are the types of constraints -

- | | |
|----------------|----------------|
| 1) Not null | 4) Foreign Key |
| 2) Unique | 5) Check |
| 3) Primary Key | 6) Default |

Only on a column name we can apply not on a table name.

① Not Null :-

1) Null \Rightarrow

① Null is nothing, neither zero nor blank space.

② It will not occupy any memory space.

③ Null represents unknown value.

④ Any arithmetic operation performed with null it results in null only.

e.g. 100 + null = null.

⑤ Two nulls are never same

ID	Name
1	A
2	
3	

Two nulls never same

- Not null will insure some value should be present in a column of a table.
- Values may be duplicate
- We can apply not null constrain on more than one columns of data.

Syntax :- column-name datatype Not Null;

e.g. ID number(2) Not null;

Not Null	ID	Name
	1	A
	2	B
	3	C
Not allowed	NULL	D

Not null constraints will not allow null value to be inserted in a column of a table but can accept duplicate value.

② Unique Key :-

- It will not allow duplicate value in a column of a table
- Unique column can take multiple null, but not duplicate.
- We can apply unique constraint on more than one column. (distinct value)

e.g.

ID number(2) unique;

unique	ID	Name	DE
	1	A	
	2	B	
	4	C	
Duplicate	5	D	E
Not allowed	5	F	

Two nulls are allowed because two null never same.

③ Primary Key :-

- It is a combination of unique + not null
 - We can apply primary key only on a single column of a table.
 - Primary key is used to uniquely identify the records.
 - Creation of primary key is not mandatory but it highly recommended to create primary key.
- eg. ID number (2) primary key ;

Primary Key	ID	Name
	1	A
Not allowed	2	B
	2	C
		D

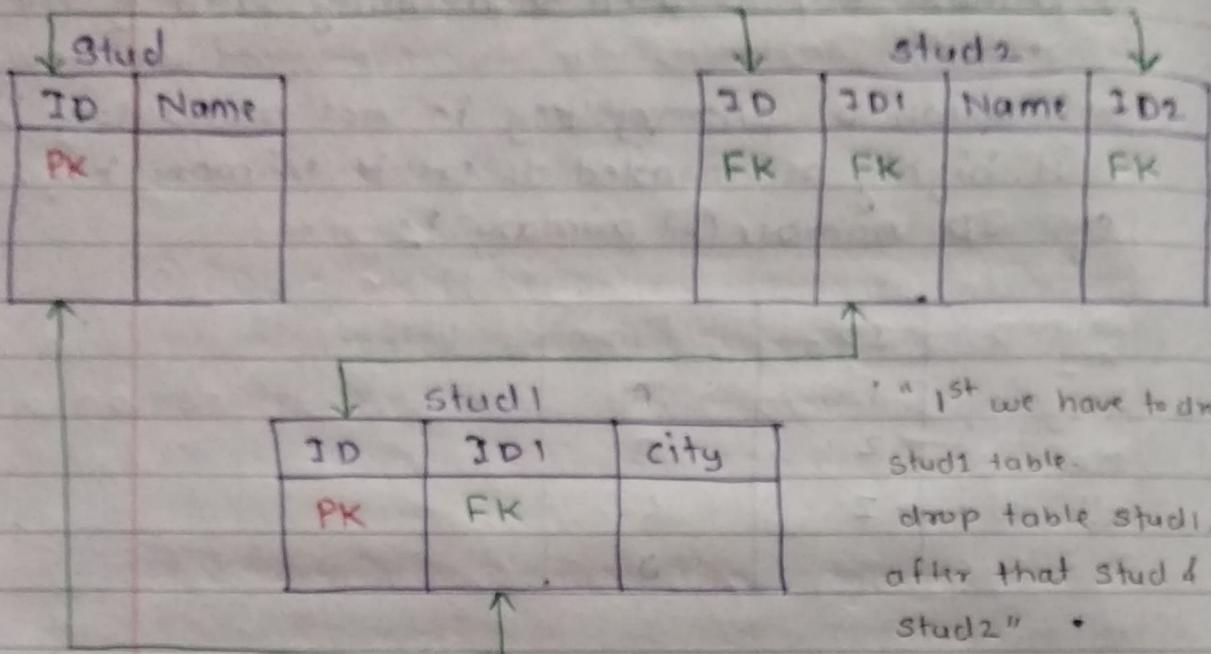
④ Foreign Key :-

- It is used to create relation between two tables.
- Foreign key is also called as referential integrity constraints.
- Foreign key is created on child table.
- To create a foreign key master table should have primary key defined on the common columns of a master table.
- We can have more than one foreign key.
- Foreign key is a combination of Null & duplicate.

create table stud (ID number (2) primary key,
Name varchar (10));

create table stud1 (ID number (2) primary key,
IDI number (2) references stud (ID),
city varchar (10));

```
create table stud2 (ID number(2) references
stud(20), ID1 number(2) references
stud1(ID), Name varchar(10),
ID2 number(2) references stud(20));
```



" 1st we have to drop
stud1 table.
drop table stud1;
after that stud &
stud2 "

- ⑤ Check :-
- Check is a user defined constraint. we can create for extra validation.
 - It will not allow invalid data to be get inserted which not satisfied check constraint.
- eg. create table stud (ID number(2), marks number(3) check(marks >=35 AND marks <=99));

stud	ID	Marks	\leftarrow marks $>= 35 \text{ & } < 99$
	1	35	allow
	2	99	allow
	3	75	allow
	4	34	\leftarrow Not allow -

using (a) primary key and check constraint
Or both constraints can have different
((a) condition part))

⑥ Default :-

- It is a user define constrain. If user will not provide any value for a column then default value can be consider./set.
- If user want to provide certain value user can provide the default value.

e.g. create table stud (ID number (2), city varchar (10)
default ('PUNE'));

ID	city
1	Mumbai
2	PUNE

By default it will add.

Simple Commands →

1] set linesize 120;
set pagesize 20;

2] select * from tab; // check all tables / number of tables
in database

3] shift delete // clear screen.

4] desc emp; // describe table // check table details
// all constraints, datatype // display table structure.

5] create table demo (ID number (2), Name varchar (10));
create ↓ Duplicate Table

create table demo1 as select * from demo;

6] ed To modify any query
/ Enter.

- # SQL Structured Query Language (1970 → IBM)
- 1) SQL is used to perform manipulation operation on data which is stored in structured formats
 - 2) On unstructured data we can not apply SQL
 - 3) SQL is a query language is not a programming language.

SQL Statements ⇒

① DDL - Data Definition Language

- (A) create
- (B) alter
- (C) drop
- (D) Truncate
- (E) Rename

② DML - Data Manipulation Language

- (A) insert
- (B) update
- (C) delete

③ DCL - Data control language

- (A) Grant
- (B) Revoke

④ DQL - Data Query Language

- (A) Select

⑤ TCL - Transaction Control Language

- (A) Commit
- (B) Rollback
- (C) Save point

① DDL \Rightarrow

(A) Create \Rightarrow It is used to create a table, views in a database.

* How to create table? \Rightarrow

create table Table-Name;

create table Demo (ID number(2) primary key,
Name varchar(10) Not null, Contact varchar(10)
unique, City varchar(10) default ('PUNE'),
Marks number(2) check (Marks >= 35 AND
Marks <= 99));

ID	Name	Contact	City	Marks
PK	Not Null	Unique	Default	check

* How to create duplicate table? \Rightarrow

create table Demol as select * from Demo;

ID	Name	Contact	City	Marks
	Not Null			

- In duplicate table all the records, all the columns and only Not null constraint will get copy.
- After creating a duplicate tables both tables are independent.

(B) Alter :- Alter is used to add column, to rename a column, to modify a structure of a column, To delete a column.

Case 1 : ADD a column \Rightarrow

Syntax \Rightarrow

alter table table-name add col-name datatype(size);
--

e.g.

1> create table stud (id number(2));

2> desc table stud;

Name	Null?	Type
ID		number

3> alter table stud add name varchar(10);

4> desc table stud;

Name	Null?	Type
ID		number
Name	Yes	Varchar

Case 2 : Modify \Rightarrow Is used to modify the structure of a table, we can change the datatype,

also size of a column

Syntax \Rightarrow To modify size.

alter table table-name modify (col-name datatype(size));

Syntax \Rightarrow To modify datatype

alter table table-name modify (col-name newdatatype(size));
--

"column should be empty to modify column datatype"

Case 3 : Rename a column \Rightarrow

Syntax \Rightarrow

alter table table-name rename column old-col-name to new-col-name;

Eg.

alter table stud rename column ID to ID1;

Case 4 : To drop a column \Rightarrow

Syntax \Rightarrow

alter table table-name drop column col-name;
--

Case 5 : Add a constraint \Rightarrow Table should be empty.

Syntax \Rightarrow

Primary	alter table table-name add constraint constraint-name constraint (column-name);
Key	

Eg.

alter table stud add constraint pk primary key(id);

Syntax \Rightarrow

Foreign	alter table table-name add constraint constraint-name constraint (column-name) references table-name (column-name);
Key	

Eg. demo

ID	Name
(PK)	

stud

ID	Marks
(FK)	

1) alter table stud add constraint FK Foreign Key (ID)
references demo (ID);

Unique \Rightarrow

2) alter table stud add constraint UK unique (Name);
- after

c) Drop \Rightarrow Drop is used to drop the structure of the table and add records of table.

Syntax \Rightarrow

`drop table table-name ;`

Eg.

`drop table stud1 ;`

SQL > Show recyclebin;

all the tables after drop will moves to recyclebin.

To restore table from recyclebin \Rightarrow

Syntax \Rightarrow

`Flashback table table-name to before drop;`

To delete the table from recyclebin \Rightarrow

Syntax \Rightarrow

`purge table table-name ;` a table"

"permanently delete

D) Truncate \Rightarrow Is used to delete or clear only records of the table, Structure of table remains same.

Syntax \Rightarrow

`truncate table table-name ;`

E) Rename \Rightarrow Is used to rename the table name.

Syntax \Rightarrow

`rename old-table-name to new-table-name ;`

Eg. `rename stud to stud1 ;`

`rename stud to stud1 ;`

② DML : Data Manipulation language

a) insert :- insert is used to insert the records at new line.

Case1: insert record in all the columns.

insert into table-name values (value1, value2);

Case 2 :

insert into specific-column

insert into tablename (column-name) values (value1);

b) Update :- update is used to update the records

Case1: Update all records.

update table-name set column-name = value;

Case2 : Update specific records

update table-name set column-name = value where condition;

Eg.

1> update stud set city = 'PUNE' where ID=1 OR ID=2 ;

2> update stud set city = 'PUNE' where ID <= 2 ;

3> update stud set city = 'PUNE' where ID=1 OR name = 'B';

1> create ^{table}stud (ID number(2), Name varchar(10));

2> insert into stud values (1, 'A');

3> insert into stud values (2, 'B');

4> alter table stud add city varchar(10);

5> update stud set city = 'PUNE';

6> select * from stud;

ID	Name	city
1	A	PUNE
2	B	PUNE

stud

alter
update

c) delete : Delete is used to delete all the records from or specific records from table.

Case1 : To delete all records.

1> delete from table-name;	OR
2> delete table-name;	

Case2 : To delete specific records

1> delete from table-name where condition;	OR
2> delete table-name where condition;	

~~4/3/21~~ ③ * TCL - Transaction Control Language

- 1) Commit
- 2) Rollback
- 3) Savepoint

① Commit -

Commit is used to save DML changes permanently.

Syntax -

Commit;

② Rollback - Is used to undo DML changes.

Syntax -

Rollback;

e.g. 1) > create

> insert

> update

> delete

> Rollback

All statements will be Rollbacked

eg 2) > create

- > insert → DML - insert will be saved automatically because of alter(DDL).
- > Alter → DDL Auto save
- > update } Rollbacked.
- > delete }.
- > Rollback

Note ① DDL statements are auto committed, we need not to commit them.

② After commit their is no use of Rollback.

③ Savepoint - It acts as a break point.

- ~~Select~~ Statements before savepoint nigher committed noz rollbacked.

Syntax:

`Savepoint Savepoint-Name;`

eg ①

> create

> insert

> update

> Savepoint X;

> delete

> Rollback to X; ← only delete will be rollbacked

> Rollback; ← All DML will be rollbacked. error → Savepoint 'X' never established

No use of savepoint.

eg. ② > create

> insert

> Savepoint a;

> alter

> update

> Rollback to a; ← error

> Rollback;

Savepoint is save records for temporary.

④ DQL - Data Query language -

Select - Is used to display the records

Syntax -

Select * from table-name;

Select column-name from table-name;

Select column-name from table-name where condn;

* Dual :-

- ~~5/3/21~~
- 1) Dual is a dummy table which is used to perform some operations which are not present on any existing table.
 - 2) By default it have one row & one column.

* ALIA's :

- 1) ALIA's is a additional name provided to the columns.
- 2) ALIA's are applicable for single execution.
- 3) We have to provide ALIA's name after column name.

Eg. > Select sal+comm As total-sal from emp;

TOTAL-SAL

1900

1700

-

2650

4) 'AS' Keyword is not mandatory

Eg. > select sal+comm total-sal from emp;

TOTAL-SAL

1900

1700

-

2650

Eg. Select sal+comm "TOTAL SAL" from emp;

TOTAL SAL

1900

1700

-

Eg. Select sal + comm "Total Sal" from emp;

Total Sal

1900

1700

-

1

Eg. Select empname, sal + comm from emp where

sal + comm \geq 1500;

emp-name	sal + comm
A	1800

-

* SQL Operators :-

1) Arithmetic Operator - +, -, *, /

2) Logical - AND, OR, NOT

3) Relational - <, >, \leq , \geq , \neq , $=$, \neq

4) Special - IS, IN, BETWEEN, LIKE

5) Concat - ||

① Arithmetic Operator -

a) write a query to display (WAQTD) incremented sal of all the emp by RS 100.

b) WAQTD decrement the commition of all the emp by RS 50.

c) Display annual sal of all the emp.

d) Display quardarly sal of all the emp's

a) \Rightarrow SQL > select empname, sal + 100 "from emp;"

b) SQL > select comm - 50 "from emp;"

c) SQL > select empname, sal * 12 from emp;

d) SQL > select ename, sal, (sal * 12) / 4 "Quotsal" from emp;

e) increment the sal of all emp by 10%

e) Select ename, sal, sal + (sal * 10 / 100) "10% emp from emp;

$$\begin{cases} \text{Sal} + \text{Sal} \times 0.1 \\ \text{Sal} \times 11 / 10 \end{cases}$$

② Relational Operator -

- WAQTD emp who's sal is greater than 1250.
- WAQTD emp who's sal is less than 3000.
- WAQTD emp who are not working in dept 30.
- Display all the salesman (JOB)
- Display all the emp who earn at least 1250.
- Display all the emp who earn at most 3000.
- Display all the emp who are not clerk.

- \Rightarrow Select ename from emp where $sal > 1250;$
- \Rightarrow select ename from emp where $sal < 3000;$
- \Rightarrow select ename from emp where $deptno \neq 30;$
- \Rightarrow select ename, job from emp where job = 'salesman';
- \Rightarrow select ename from emp where $sal \geq 1250;$
- \Rightarrow select ename from emp where $sal \leq 3000;$
- \Rightarrow select ename from emp where $job \neq 'clerk';$
 $(job \neq 'clerk');$

③ Logical Operator -

- Display all the employees who are working as a clerk or manager.
- DATE who are salesmans and earning sal $> 1250.$
- DATE who are from dept 10 & 20.
- DATE who are not analist
- DATE who are analist & earning sal > 2500 or < 5000

- \Rightarrow select * from emp where job = 'CLERK' OR 'MANAGER';
- \Rightarrow select * from emp where job = 'salesman' AND $sal > 1250;$
- \Rightarrow select * from emp where deptno = 10 AND OR $deptno = 20;$
- \Rightarrow select * from emp where job NOT 'Analist';
~~not~~ job = 'ANALIST';

\Rightarrow select * from emp where job = 'Analyst' AND sal > 2500
OR sal < 5000;

Q) * Special Operator:- is, In, Between, like

i) Is: - It is used to compare null.

We cannot compare any other value with is.

Syntax:

```
select * / col-name from table-name
where col-name is null / is not null;
```

Q. Write a query to display employees who are not earning any commission.

\Rightarrow select * from emp where commission is null;

Q. WAQID who are earning some commission.

\Rightarrow select * from emp where commission is not null;

ii) In: Instead of using multiple OR we can use single in operator.

In operator works as OR (II) operator.

If we want single LHS with multiple RHS we use in operator.

Syntax:

```
select * / col-name from table-name where
col-name in (value1, value2, ...);
```

Q. WAQID the emp who are working as a clerk or analyst.

\Rightarrow select * from emp where job in ('clerk', 'analyst');

Q WAPTD the emp who are not working as a clerk or analyst.

⇒ select * from emp where job not in ('clerk', 'analyst');

iii) Between : Between operator selects the value within a given range.

Syntax :

Select * / col-name from table-name where col-name between value1 and value2;

Q WAPTD the emp whose salary between 1250 and 3000.

⇒ select * from emp where sal between 1250 and 3000;

Q WAPTD emp whose sal not between 1250 and 3000.

⇒ select * from emp where sal not between 1250 and 3000;

Q WAPTD emp whose name in between 'A' to 'S'.

⇒ select * from emp where ename between 'A' and 'S';

iv) like : Like operator is used for pattern matching.
we use two wildcard characters for pattern matching (like spe char)

Following are the two like char

a) % = It matches with 0, 1, or n number of characters.

b) _ = It matches with single character

escape : escape char are used in pattern string to indicate that any wildcard char that occurs after escape char, in a pattern string should be treated as a regular char.

- Q WITD the emp whose name starts with A
 ⇒ select * from emp where ename like 'A%',
pattern string

Note

- i) Name ends with A = ename like '%, A'
- ii) Name starts with A, ename like 'A %, A'
and ends with A
- iii) Whose name contains ename like '%, A%, '%
A anywhere in
the name
- iv) Whose name contains ename like '----'
exactly 4 char
- v) Name contains A at = ename like '_ A%, '
2nd position
- vi) Name contains A at = ename like '%, A '-'
2nd last position
- vii) Whose name contains ename like '%, A%, A%, '%
2 A

- Q1 List all the emp who don't have any reporting manager?
- ⇒ Select * from emp where mgr is null;
- Q2 List all the salesmans in dept 30 and having salary grater than 1500.
- ⇒ Select * from emp where job = 'Salesman' and DeptNo = 30 and sal > 1500;
- Q3 List all the emp whose name starts with 'S' or 'A'
- ⇒ Select * from emp where ename like 'S%.' OR 'A%.';
- Q4 List all the emp whose name does not starts with 'ES' or 'R'
- ⇒ Select * from emp where ename not like 'ES%.'
or 'R%.' ⇒ ename not like 'R%.'
- Q5 List all the emp with anual salary except those who are working in Dept 30.
- ⇒ Select * from emp
Select sal*12 As Anual-Sal from emp where dept is not 30;
- Q6 Display all the emp who are joined after year 81.
- ⇒ Select * from emp where hiredate >= '31-Dec-1981';
- Q7 Display all the emp who are joined in Feb.
- ⇒ Select * from emp where hiredate like '%.FEB%.';
- Q8 List the emp who are not working as a manager & clerk in dept 10 & 20 with the salary in a range of 1000 to 3000.

⇒ Select * from emp where job not in ('Manager', 'Clerk') and Deptno in (10, 20) And sal between 1000 And 3000;

Q9 List all the emp whose salary not in a range of 1000 to 2000 and working in dept 10, 20 or 30 except all salesmans.

⇒ Select * from emp where sal not between 1000 and 2000 and deptno in (10, 20, 30) and job not in ('Salesman');

Q10 Display all the emp whose job starts with man ..

⇒ Select * from emp where job like 'MAN%';

Q11 List the emp who are hired after 82 and before 87

⇒ Select * from emp

Q12 Display the emp whose commision is greater than sal

⇒ Select * from emp where comm > sal;

Q13 Display the emp whose annual sal ends with 0.

⇒ Select ename, sal, sal*12 'ANN-SAL' from emp where sal*12 like '%0';

Q14 Display the emp whose name starts with vowels

⇒ Select * from emp where ename like 'A%' or ename like 'E%' or ename like 'I%' or ename like 'O%' or ename like 'U%' ;

Q15 Display the emp who are hired in nov & Dec

⇒ Select * from emp where Hiredate like '%.Nov.' or hiredate like '%.Dec.';

Q.16 Display the emp whose job contain manager.

⇒ Select * from emp where job is 'manager' & like '%MAN%';

Q.17 Display all the emp who are having reporting manager in dept 10 also g with 10% high in the salary.

⇒ Select * from emp where select ename, sal, sal+sal*0.1 "Hike 10%" from emp where deptno=10 and mgr is not null;

Q.18 Display the emp whose name having exactly two L's.

⇒ Select * from emp where ename like '%.L%.L%';

Q.19 Display the name and designation of all the emp having reporting manager and also their names starts with 'S'.

⇒ Select ename, job from emp where mgr is not null and ename like '%.S%';

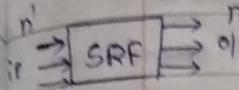
Q.20 Display all the emp who are having reporting manager in dept 10.

⇒ Select * from emp where deptno=10 and mgr is not null;

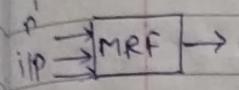
* SQL Functions :-

SQL functions are categorised into two types -

① Single Row function -

 Here we are providing n number of inputs, & for every input we will get separate output.

② Multi Row function -

 To multi row function we are providing n number of inputs and we will get only one output for multiple inputs.

* Multi Row function - (Aggregate function / Group function)

i) Multi row functions are :-

min()

max()

avg()

sum()

count()

② min() :- It returns minimum value or lowest character of given column. Syntax

Select min(col-name) from Table.name;

e.g.

> Select min(sal), min(hiredate), min(ename) from emp;

MIN(SAL)

MIN(HIREDATE)

min(ENAME)

800

17-DEC-80

ADAMS



ASCII value

> Select min(sal+comm) from emp;

min(sal+comm)

1500

(b) Max() It returns maximum value or largest character.

Syntax:

Select max(column-name), max(column-name) from table-name;

Eg.

> select max(sal), max(hiredate), max(ename), max(comm)
from emp;

max(sal) max(hiredate) max(ename) max(comm)

5000

~~10/3/21~~

* Average :- It returns average value of specified column.

Syntax:

Select avg(column-name) from Table-name;

Eg.

Select avg(sal) from emp;

Select avg(mgr) from emp;

Select avg(sal + comm) from emp;

* Sum: It returns summation of all the values in given column.

Syntax:

Select sum(column-name) from Table-name;

Eg.

Select sum(sal) from emp;

* Count: It returns number of rows in a given column.

Note Count function will not count null values.

Syntax:

Select count(column-name) from table-name;

Eg.

1> Select count(emphno) from emp;

→ 14

2) select count(comm) from emp;

→ 4

3) select count(*) from emps;

→ 14

4) select count(*) - count(comm) from emp;

→ 10

5) select count(*) from emps where comm is null;

→ 10

* distinct : It is a keyword. It is used to select unique record from the column.

Syntax :

select distinct column.name from Table-name;

eg. 1) select distinct deptno from emp;

→ 30

20

10

func.
nesting

function under function

2) select distinct dle

select count(distinct + (deptno)) from emp;

→ 3

* Single Row function:-

String Function

case

- lower
- upper
- initcap

char

- length
- replace
- reverse
- instr
- substr
- concat
- trim
 - LTrim
 - RTrim
- lpad
- rpad

Case manipulation function :-

① Lower:- It returns character of string in a lower case letter.

Syntax :

`Select lower (column-name) from Table-name;`

e.g.

> `Select lower ('SQL') from dual;`

→ SQL

eg. update the smithname in lowercase in emp table.

→ `update emp set ename = lower(ename) where ename = "SMITH";`

② Upper :- It is used to convert character of string in uppercase letter.

Syntax :

`Select upper (column-name) from Table-name;`

eg. `Select upper ('SQL') from dual;`

→ SQL

③ InitCap :- It returns first character of string in uppercase and rest every thing in lowercase.

Syntax :

`Select initcap (column-name) from emp;`

eg.

`Select initcap(ename) from emp;`

→ Smith

Allen

ward

Jones

eg.

Select lower(ename) "Lower", Upper(ename) "Upper",
initcap(ename) "Initcap" from emp;

→

lower	Upper	Initcap
Smith	SMITH	Smith
allen	ALLEN	Allen
ward	WARD	Ward

* Character Manipulation Function -

- ① length:- It returns number of character in a given string.

Syntax -

Select length(column-name) from table-name;

eg.

Select length(ename).pname from emp;

- ② Write a query to display an employee whose name is of 5 character.

→ Select ename from emp where length(ename)=5;

→ SMITH

ALLEN

JONES

BLACK

- ③ Concat : It is used to join two or more strings or columns.

Syntax:

Select concat(column-name1, column-name2)
from emp;

eg. Select concat ('QSP', 'Hadapsar') from dual;
 → QSPHadapsar

i) Nested concat :-

Syntax: select concat(column-name1, concat(column-name2, column-name3));

Eg.
 Select concat ('QSP', concat ('Pune', 'Hadapsar'));
 → QSPPuneHadapsar

~~11/3/21~~ (3) Reverse : It reverse the characters of a String.

Syntax: select reverse(column-name) from Table-name;

Eg.
 Select reverse(ename) from emp;
 → HTHMS (SMITH)
 → NELLA (ALLEN)

If we want to reverse of number i.e 321 → 123
 then we should pass with single cot i.e
 '321' → 123

(4) Replace : It is used to replace the character of a String.

Syntax: select replace(string, 'char to be replace', 'char to be replace by') from table-name;

Eg. 1) select replace ('qspiders', 's', '\$') from dual;
 → q\$piders

Note - If we will not provide the 3rd parameter it will acts as remove.

eg. 2) select replace ('qspiders' , 'qspiders' , 'qsp')
from dual;

→ qsp

eg. 3) select replace ('ekata' , 'ekta' , 'e') from dual;
→ ekata

eg. 4) select replace ('qspiders' , 'pid' , '*') from dual;
→ qs*ers

eg. 5) select replace ('qspiders' , 's') from dual;
→ spider ↑
3rd parameter not provided
so it will acts as remove

⑤ Instring: It is used to identify position of character in given string.

Syntax:

`Instring('string' , 'char' , start position ,
no.of occurrences);`

- Start position & number of occurrences are optional
- By default start position will be 1 & number of occurrences also.
- If start position is negative number it will start search from end.
- Start position and number of occurrences cannot be zero (0);
- We cannot provide number of occurrences as a negative number.

eg.

- 1) select instr ('qspiders', 's', 1, 1) from dual;
→ 2
- 2) select instr ('qspiders', 's', 1, 2) from dual;
→ 8
- 3) select instr ('qspiders', 's', 3, 1) from dual;
→ 8
- 4) select instr ('qspiders', 's', 3, 2) from dual;
→ 0
- 5) select instr ('qspiders', 'z', 1, 1) from dual;
→ 0
- 6) select instr ('qspiders', 's', -1, 1) from dual;
→ 8
- 7) select instr ('qspiders', 's', -1, 2) from dual;
→ 2
- 8) select instr ('qspiders', 's', -2, 1) from dual;
→ 2
- 9) select instr ('qspiders', 's') from dual;
→ 2
- 10) select instr ('qspiders', 's', 0, 1) from dual;
→ 0
- 11) select instr ('qspiders', 's', \$, 0)
→ error (Number of occurrence can't be zero)

a) Substring := It display substring from main string.

Syntax:

`substr (string, start position, no. of char);`

- If start position is negative number then it will start from end.
- number of characters are option
- By default it will display the string till end

Eg. `Select ename, substr(ename, 1, 3), substr(ename, 2, 2) substr(ename, 3) from dual;`

ENAME	SUB	SU	SUBSTR
1) SMITH	SMI	MI	TH
2) ALLEN	ALL	LL	LEN
3) WARD	WAR	AR	RD

Eg. `Select ename, substr(ename, -3, 3), substr(ename, -3, 2) substr(ename, -1, 2), substr(ename, -1, 1) from emp;`

→

ENAME	SUB	SU	S	S
SMITH	SMITH	IT	H	H
ALLEN	LEN	LE	N	N
WARD	ARD	AR	D	D

Eg. 1) `Select substr('Qspiders', 0, 1) from dual;`

→ Q

2) `Select substr('Qspiders', 1, 1) from dual;`

→ Q

3) `Select substr('Qspiders', 1, 0) from dual;`

→ No output.

4) select substr ('@spiders', 1, -1) from dual;

→ \$ no output

5) select substr ('@spiders', 4, 1) from dual;

→ \$ no output

12/3/21

6) Display name of emp who's job starts with man;

→ select ename from emp where job like 'man%';
select ename, job from emp where substr(job, 1, 3) = 'MAN';

Ename

job

JONES

MANAGER

BLAKE

MANAGER

CLARK

MANAGER

⑥ Trim :- Is used to trim the character from given string.

Case 1 : To delete start char

Syntax:

select trim (leading 'char' from 'string')

Case 2 : To delete end char

Syntax

trim (trailing 'char' from 'string')

Case 3 : To delete start & end char

Syntax:

trim (both 'char' from 'string')

"To trim middle char we make use of replace query."

Eg.

- 1) Select trim (leading 'a' from 'akshaya') from dual;
→ TRIM (L
kshaya)
- 2) Select trim ('a' from 'akshaya') from dual;
→ TRIM (.
kshay)
- 3) Select trim ('a' from 'ekata') from dual;
→ TRIM
ekat
- 4) Select trim ('a' from 'akshaya') from dual;
→ TRIM
kshay
- 5) Select trim (leading 'o' from 'akshya') from dual;
→ TRIM
kshaya
- 6) Select trim (trailing 'a' from 'akshaya') from dual;
→ TRIM
akshay
- 7) Select trim (both 'a' from 'akshya') from dual;
→ TRIM (.
Kshay)
- 8) Select trim (leading 'ak' from 'akshya') from dual;
↑
→ Error → we can not take two char.

① Replace a - @ and a - # in string 'ekita'

Select ~~concat~~ replace ('EKita', 'a', '@', '#')

C

Ltrim : Used to ~~trim~~ the char from left side

Syntax :

`ltrim('string', 'substring')`

eg. 1) Select ltrim('eaeaaaaweaed', 'ae') from dual;

→ `ltrim`

weaed

2) Select ltrim('eaeaaaaweaed', 'wae') from dual;

→ `ltrim`

3) Select ltrim('eaeaaaaweaed', 'war') from dual;

`ltrim`

eaeaaaaweaed

It will trim substring upto new / ~~upper~~ character found.

It will starts from left side & check the 1st char present in substring or not.

Rtrim; Rtrim is used to trim the char from right side:

Syntax :

`rtrim ('string', 'substring')`

eg. 1) Select rtrim('eaeaaaaweaed', 'ae') from dual;

`RTRIM`

eaeaaaaw

2) select rtrim('eaewaaaweaed', 'ew') from dual;

→ R

3) select rtrim('eaewaaaweaed', 'ew') from dual;

RTRIM

eaewaaaweaed

4) select rtrim(ltrim('Qspider', 'Q'), 'r') from dual;

→ Spide

spider

sider

* Rpad :- To append the blank space.

eg. ***qsp***

→

Select rpad (lpad ('qsp', 16, '*'), 9, '*') from dual;

LPAD (RPAD)

qsp

Syntax : rpad ('string', 'number of char', 'Padded char').

lpad ('string', 'number of char', 'Padded char')

eg. select lpad ('qsp', 3, '*') from dual;

LPA

qsp

2) select lpad ('qsp', 1, '*') from dual;

→ q

3) select lpad ('qsp', -6, '*') from dual;

→ —

4) select lpad ('qsp', 0, '*') from dual;

→ —

* Number Functions -

ceil

floor

Round

Trunc

Abs

Mod

Power

Sqrt

③ Round :-

- eg. 1) select round(123.456, 3) from dual; 123.456
 ② select round(123.456, 2) from dual; 123.46
 ③ select round(123.425, 1) from dual; 123.4
 ④ select round(123.425, 0) from dual; 123
 ⑤ select round(123.541, 0) from dual; 124
 ⑥ select round(123.425, -1) from dual; 120
 ⑦ select round(123.345, 1.5) from dual; 123.3
 ⑧ select round(123.456, -1.5) from dual; 120
 ⑨ select round(123.456, -2) from dual; 100
 ⑩ select round(123.425, -3) from dual; 0
 ⑪ select round(123.425, -4) from dual; 0
 ⑫ select round(555.555, -3) from dual; 1000
 ⑬ select round(555.555, -4) from dual; 0

4) Trunc () : It will goes on only deleting value, It will not roundup the value.

- ① select trunc(123.456, 3) from dual; 123.456
 ② select trunc(123.456, 2) from dual; 123.45
 ③ select trunc(123.456, 1) from dual; 123.4
 ④ select trunc(123.456, 0) from dual; 123

-ve number - remove digits before decimal

+ve number - After decimal how many digits you want.

- ⑤ select trunc(123.456, -1) from dual; 120

⑤ Absolute Value :- `abs()`

convert -ve number to +ve
remove unwanted zero

① select `abs(10.01)` from dual; 10.01

② select `abs(-10.01)` from dual; 10.01

③ select `abs(-10.10)` from dual; 10.1

④ select `abs(10.0001)` from dual; 10.0001

⑤ select `abs(-01.10)` from dual; 1.1

⑥ Power () :- It will return power of no

① select `power(12,2)` from dual; 144

② select `power(15,2)` from dual; 225

⑦ Square `sqrt()`; It returns square root

① select `sqrt(625)` from dual; 25

② select `sqrt(1000)` from dual; 10

⑧ Mod () :

① select `mod(12,4)` from dual; 0

② select `mod(12,3)` from dual; 0

③ select `mod(12,5)` from dual; 2

* General functions:-

① `NUL(exp1,exp2)` or `(arg1,arg2)`

② `NUL2`

③ `decode`

① NVL :-

NVL(arg1, arg2)

- 1) If value of argument 1 is null then it returns value of argument 2.

NVL means null value.

eg.

Display salary of employee (sal + comm) if employees are not earning commission give commission rs.100
 > select ename, sal, comm, sal+nvl(comm, 100) from emp;

② NVL2 :-

Syntax:- nvl2(arg1, arg2, arg3)

null
not null

- If value of arg1 is null then it returns value of arg3.
- If value of argument1 is not null then it returns value of arg2

eg.

Display total sal of employee & give commission rs 100 to all employee.

> select ename, sal, comm, sal+nvl2(comm, comm+100, 100)
 from emp;

③ Decode :- It act as a if else statement.

eg.

- 1) Increment salary of CLERK by rs 50, Increment salary of all SALESMAN by rs 100 & remaining all employees salary should be incremented by 10.

> select ename, job, sal, decode(job, 'CLERK', sal+50, 'SALESMAN', sal+100, sal+10) from emp;

2) Select ename, job, decode(job, 'CLERK', sal+50,
 $\underbrace{\text{sal}+10}$) from emp;
 If statement part
 else part

3) If job is CLERK return CC & if job is SALESMAN
 return SS.

> select ename, job, decode(job, 'CLERK', 'CC',
 'SALESMAN', 'SS') from emp;

4) Select ename, job, decode(job, 'CLERK', sal+nvl
 comm , 0) from emp;

* Conversion Function -

① to-number :-

eg. select to-number('123') from dual;
 Providing number in string format

⇒ to-number

123 ← in integer format

② to-date :- if date in 12-02-21 format then give
 the date format you want.

eg. 'mm-dd-yy'

If date in 12-MAR-2021 then it will print as
 it is no any format given by user.

eg.

1) Select to-date ('12-02-21', 'mm-dd-yy') from dual;
 mm dd yy

2) Select to-date ('02-MAR-21') from dual;

③ to-char:-

1> Select to-char (sysdate, 'dd-month-year') from dual;
OP \Rightarrow 20 - march - twenty twenty one

2> Select to-char (to_date ('01-FEB-21'), 'dd.spth-mon-year')
from dual;
OP \Rightarrow First - FEB - twenty twenty-one.

D - Day of week

w - week of month

DD - Day of month

WW - week of year

DDD - Day of year

DD \rightarrow print 2digit of date (20)

MM \rightarrow print 2digit of month (03)

YY \rightarrow print 2digit of year (21)

YYYY \rightarrow 4 digit of year (2021)

Fm \rightarrow Filter (0) zero remove unwanted zero

th \rightarrow It is used for third, nd, st

DDSP \rightarrow Spellout date (suppose DD = 20 then DDSP = twenty)

Mon \rightarrow Abbreviated name of month (eg - FEB, MAR etc)

Year \rightarrow year spell out

(suppose YYYY = 2021, twenty twenty-one)

Month \rightarrow Name of month

(suppose Mon = MAR, month = March)

Day - name of day (suppose 20 - Saturday)

Dy - Abbreviated name of day (20 - Sat)

3) to print 9th of sep

> Select to-char (to-date ('09-sep-21'), 'ddth, "of", mon')
from dual;

> Select to-char (to-date ('09-oct-21'), 'dd-mm-yy'),
('ddth "of" mon') from dual;

e.g. convert number into character

4) Select to-char (sal, '\$999.99') from emp;

⇒ \$ 800.00

\$ 1600.00

\$ 1250.00

5) Select to-char (sal, '\$999.99') from emp;

⇒ \$ 800.00

we can provide 3 digit number but in sal

#####

only 800.00 is a one 3 digit number other
are greater than 3 digit so their print the #.

6) Select to-char (sysdate, 'ts') from dual;

~~22/10/21~~ * Date Function it to move between date & time

- Sysdate

- Arithmetic with sysdate

- Systimestamp

- Add-months

- Months-between

- Next-day

- Last-day

1) `Sysdate`: It returns current system date.

eg. `select sysdate from dual;`
 ⇒ 22-MAR-21

2) Arithmetic with `Sysdate`:

eg. `select sysdate + 365 from dual;`
 ⇒ 22-MAR-22

3) `Systimestamp`:- It returns current system date & time in milliseconds also time zone

eg. `select systimestamp from dual;`

4) `Add-months`:- It returns a date after adding or substrating number of months.

eg. 1) `select add-months(sysdate, 2) from dual;`

⇒ 22-MAY-21

2) `select add-months('21-MAR-21', -2) from dual;`

⇒ 21-JAN-21

3) `select add-months(sysdate, -2) from dual;`

⇒ 22-JAN-21

4) `select add-months('21-MAR-2021', 2) from dual;`

⇒ 21-MAY-21

5) `Months-Between`:- It returns number of months between two given dates.

eg. `select months_between('22-MAR-2020', '22-MAR-2021')`

⇒ 12 months (1 year) → months between 12 months

2) `select month-between('22-MAR-22', sysdate) from dual;`

⇒ 12 months (1 year) → months between 12 months

3) `select months-between('22-Mar-21', '22-Mar-22') from dual;`

⇒ -12 months (1 year) → months between -12 months

⇒ -12

⑥ next-day :- It returns next day on upcoming day.

1) select next-day (sysdate, 'mon') from dual;

→ 29 - MAR - 21

2) select next-day (sysdate, 'sat') from dual;

→ 27 - MAR - 21

⑦ Last-Day : It returns last date of given month

1) select last-day (sysdate) from dual;

op → 31 - MAR - 21

2) select last-day ('21-FEB-2020') from dual;

op → 29 - FEB - 2020

* General Function :- Functions of database

4) nullif :- It returns null if both parameters are same.

e.g. 1) select nullif ('Mon', 'Mon') from dual;

→ null

2) select nullif ('Mon', 'Tues') from dual;

→ Mon ^{# arg point}

5) Coalesce :- It is similar to NVL but it accepts any number of parameters (at least 2) & it will returns 1st of them if it is not null.

- If it is null then it return second parameter

- If all of them are null then it returns null.

e.g. 1) select coalesce (null, 'A') from dual;

→ A

2) select coalesce ('A', null) from dual;

→ A

3) Select coalesce (null, null, 'A') from dual;

→ A

- 4) select coalesce (null, 'A', 'B') from dual;
- 5) select coalesce (null, null, null, 'A', 'B') from dual;
 > A
- 6) Select coalesce ('A', null, 'B') from dual;
 > A
- 7) Select coalesce (null, null, null) from dual;
 > null.

Queries

Page:

Date: 11/11

Use Functions Only

- 1) List employees whose name having 4 characters (without like).

⇒ Select ename from emp where length(ename)=4;

- 2) List employees whose job is having 7 characters

⇒ Select ename, job from emp where length(job)=7;

- 3) Find out how many times letter 's' occurs in 'Qspiders'.

⇒ Select length('Qspiders') - Length(Replace('Qspiders', 's')) from dual;

- 4) List the employees whose job is having last 3 characters as 'man'.

⇒ Select job from emp where substr(job, -3) = 'MAN';

- 5) List employees whose job is having first 3 char as 'man'.

⇒ Select job from emp where substr(job, 1, 3) = 'MAN';

- 6) Display all the names whose name is having exactly 1 'L' (without like)

⇒ Select ename from emp where length(ename) - length(Replace(ename, 'L')) = 1 ;

⇒ Select ename from emp where instr(ename, 'L', 1, 1) > 0 and instr(ename, 'L', 1, 2) = 0;

7) Display dept names which are having letter 'O'.

⇒ Select dname from dept where instr(dname, 'O', 1) > 0;

8) Display the output as shown below,

Scott working as a clerk earns 3000 in dept 20.

⇒ Select concat(ename, concat('working as a', concat('clerk
earns 3000 In deptno', deptno))) form emp where
ename = 'scott';

9) Calculate number of L in string 'HELLLLL'

⇒ Select length (HELLLLL)-length (Replace (HELLLLL, 'L'))
from dual;

10) Display all the employees whose job has a string
'MAN' (without like)

⇒ Select job from emp where instr (job, 'MAN', 1)>0;

11) Display all the employees whose job starts with
string 'MAN' (without like)

⇒ Select job from emp where substr (job, 1, 3)='MAN';

12) Display all the employees whose job ends with
string 'MAN' (without like)

⇒ Select job from emp where substr (job, -3) = 'MAN';

13) Display first 3 characters of ename in lowercase &
rest everything in uppercase. If ename is 'QSPIDERS'
then display this as 'qSPIDERS'

⇒ Select concat (lower (substr ('QSPIDERS', 1, 3)), substr
('QSPIDERS', 4)) from dual;

14) Display the result from emp table as below:

SMITH is a CLERK and gets salary 2000.

Here SMITH is ename column, CLERK is job and 2000 is SAL column and rest everything is literal strings.

⇒ Select concat(ename, concat('is a', concat(job, concat('and gets a salary of', 2000)))) from emp where ename = "Smith" and job = 'CLERK';

15) List the employees hired on a wednesday

⇒ Select ename, hiredate from emp where to_char(hiredate, 'DY') = 'WED';

16) List the employees hired on a sunday in the month of may.

⇒ Select ename, hiredate from emp where to_char(hiredate, 'DY') = 'SUN' and to_char(hiredate, 'MON') = 'MAY';

17) Display first half of name in lower and rest in reverse.

⇒ Select concat(lower(substr(ename, 1, length(ename)/2)), reverse(substr(ename, length(ename)/2+1))) from emp;

18) GADTD last 3 char of emp who are working as manager

⇒ Select substr(ename, -3) from emp where job = 'MANAGER';

19) GADTD name of emp if name starts with vowel (without like)

⇒ Select ename from emp where substr(ename, 1, 1) in ('A', 'E', 'I', 'O', 'U')

20) GIAQTD 1st half of emp name

⇒ Select ename , substr(ename,1,length(ename)/2) from emp;

21) GIAQTD 2nd half of emp name.

⇒ Select ename , substr(ename,1,length(ename)/2+1) from emp;

22) GIAQTD name of emp without 1st and last char.

⇒ Select ename , substr(ename,2,length(ename)-2) from emp;

23) GIAQTD name of the emp with char 'A' in name

⇒ Select ename from emp where instr(ename,'A',1,1)>0;

Subquery

- 1) Display all the employees whose department name ending 'S'.
 ⇒ Select ename from emp where deptno in (select deptno from dept where dname like '%.S');
- 2) Query to display the emp names who is having maximum salary in dept name 'Accounting'.
 ⇒ Select * from emp where sal = (select max(sal) from emp where deptno = (select deptno from dept where dname = 'Accounting'));
- 3) Query to display the dept name who is having highest commission.
 ⇒ select dname from dept where deptno = (select deptno from emp where comm = (select max(comm) from emp));
- 4) Query to display the employee names whose dept name has 2nd char as 'O'.
 (No answer provided)
- 5) Query to display all the emp's who's dept no is same as scott.
 ⇒ Select * from emp where deptno = (select deptno from emp where ename = 'SCOTT');
- 6) Query to display all the employees in 'Operations & Accounting' dept.
 ⇒ Select ename, deptno, from emp where deptno in (select deptno from dept where dname in ('Accounting', 'Operations'));

- 7) List the employees who has salary greater than miller
 ⇒ Select * from emp where sal > (select sal from emp where ename = 'MILLER');
- 8) Display the dname of employees who has no reporting man.
- 9) List all the employees who are reporting to jones manager.
 ⇒ Select * from emp where empno = (select mgr from emp where ename = 'Jones');
- 10) Display dname of emp whose name does not starts with 'S' and sal between 1500 and 3000
 ⇒ Select dname from dept where deptno in (select deptno from emp where ename not like 'S%' and sal between 1500 and 3000);
- 11) Display location of emp whose sal is min but sal > 2000
 ⇒ Select loc from dept where deptno = (select deptno from emp where sal in (select min(sal) from emp where sal > 2000));
- 12) Display the location of an employee in accounting dept.
 ⇒ Select loc from dept where deptno in (select deptno from dept where dname = 'Accounting Department');
- 13) WAQTD all the employee whose job not same as ALLEN and salary is greater than MARTIN.
 ⇒ Select ename, job, sal from emp where job != (select job from emp where ename = 'ALLEN') and sal > (select sal from emp where ename = 'MARTIN');

14) Display all the employees who is having location is same as ADAM's manager?

⇒ Select ename/* from emp where deptno = (

Select deptno from dept where loc = (

Select loc from dept where deptno = (

Select deptno from emp where empno = (

Select mgr from emp where ename = 'ADAMS')));

15) Display the job, manager, number of employees who is working for jones?

⇒ Select job, mgr from emp where mgr = (

Select empno from emp where ename = 'JONES');

16) Display the employee names, hired date, commission of ford's manager?

⇒ Select empno, hiredate, comm /* from emp where empno = (Select mgr from emp where ename = 'FORD');

17) Display the number of employees who are getting salary less than the blake's manager.

⇒ Select count(*) from emp where sal < (

Select sal from emp where mgr = (

Select mgr from emp where ename = 'BLAKE'));

18) List employees who located in CHICAGO and their commision is zero.

⇒ Select ename from emp where comm in 0 and deptno in (Select deptno from dept where loc = 'CHICAGO');

19) List employees who work for sales department and their salary greater than average salary of their department.

→ Select ename from emp where deptno in (select deptno from dept where dname = 'SALES') and sal > (select avg(sal) from emp where deptno in (select deptno from dept where dname = 'SALES')) ;

20) List employees who are working in research dept and they are manager.

→ Select ename from emp where Job = 'Manager' and deptno in (select deptno from dept where dname = 'Research') ;

21) Display dept. name of the employees who earn commission.

→ Select dname from dept where deptno in (select deptno from emp where comm is not null) ;

Select dname from dept where deptno in (select deptno from emp where comm != null) ;

22) Display department name of the employee who earn maximum salary and have no reporting manager.

→ Select dname from dept where deptno in (Select deptno from emp where sal = (select max(sal) from emp where mgr is null)) ;

23) Display employee details who are reporting to blake and have commission without using null or not null .

→ Select * from emp where mgr in (select empno from emp where ename = 'BLAKE') and comm >= 0 ;

24) list all the deptname and loc of all the salesman manager's manager.

⇒ Select cname, loc from dept where deptno = (

Select deptno from emp where empno = (

Select mgr from emp where empno in (

Select mgr from emp where job = 'SALESMAN'));

25) Display 2nd max sal.

⇒ Select max(sal) from emp where sal < (select max(sal) from emp);

26) Display 2nd min sal.

⇒ Select min(sal) from emp where sal != (select min(sal) from emp);

27) Display max sal of Sales department

⇒ Select max(sal) from emp where deptno in (Select deptno from dept where dname = 'SALES');

28) Display all the emp whose job same as smith, dept same as Jones and sal > Turner

⇒ Select * from emp where job in (select job from emp where ename = 'SMITH') and dept in (select deptno from dept where ename = 'Jones') and sal > (select sal from emp where ename = 'Turner');

29) Display no of emp who comm is more than sal

30) Display the location of emp who earn max sal & have no reporting manager.

⇒ Select loc from dept where deptno in (select deptno from emp where mgr is null and sal = (select max(sal) from emp));

31) Display ename & empno of employee working as a CLERK and earn highest sal among all the clerks.

⇒ Select ename, empno from emp where job = 'CLERK' and sal in (select max(sal) from emp where job = 'CLERK');

32) Display all emp who are living in location which is having at least two 'o' in it.

⇒ Select ename from emp where location.deptno in (select deptno in (select deptno from dept where loc like '%.0%.0%'));

33) List the emp who has comm > max sal of all the salesmans and do not report to king.

⇒ Select ename from emp where comm > (select max(sal) from emp where job = 'SALESMAN') and mgr != (select empno from emp where ename = 'KING');

34) List all the dept name that are having salesmans.

⇒ Select dname from dept where deptno in (select deptno from emp where job = 'SALESMAN');

35) List all the emp who are earning sal more than any of the analyst.

⇒ Select ename, sal from emp where sal > Any (select sal from emp where job = 'Analyst');

1) Display all the employees who are joined before the last person?

⇒ Select * from emp where hiredate < (select max(hiredate) from emp);

2) Display all the employees who are earning more than any of the manager.

⇒ Select * from emp where sal > any (select sal from emp where job = 'MANAGER');

3) List employees who joined after 4 years of 1st emp of the company and less than Blake salary.

⇒ Select * from emp where hiredate > select add-months(min(hiredate), 4*12) from emp and sal < (select sal from emp where ename = 'BLAKE');

4) Display all the emp whose dept is sales and who is earning some comm (i.e. comm is not null or zero) and who is hired before the last person hired.

⇒ Select * from emp where comm is not null and hiredate < (select max(hiredate) from emp) and deptno in (select deptno from dept where dname = 'SALES');

5) Display all the dept names for ^fward's man's manager

⇒ Select dname, mgr from emp e, dept d where e.deptno = d.deptno and empno in (select mgr from emp where empno in (select mgr from emp where ename = 'WARD'));

6) Display dept name of the employee who earn minimum sal and have reporting manager.

⇒ Select dname from dept where deptno in (select deptno from emp where sal = (select min(sal) from emp where mgr is not null));

7) Display the dname of emp's whose salary is max salary but Lesser than 3000.

⇒ Select dname from dept where deptno in (select deptno from emp where sal in (select max(sal) from emp where sal < 3000));

8) Display last employee record according to empno

⇒ Select * from emp where empno = (select max(empno) from emp);

9) Select ename of emp who earns 2nd max salary & works for location 'DALLAS'

⇒ Select ename from emp e where e.deptno ≠
 Select ename, dname from emp e, dept d where
 e.deptno = d.deptno and e.sal in (select max(sal) from emp where sal < (select max(sal) from emp)) and
 loc = 'DALLAS';

10) List emp's who have commission greater than max sal of all the salesman & who do not report to King.

⇒ Select * from emp where comm > (select sal from emp where sal = (select max(sal) from emp where job = 'SALESMAN')) and mgr ≠ (select empno from emp where ename = 'KING');

Subqueries - Group By -

- 1) list deptname having atleast 3 salesman.

Select dname from dept where deptno in (select deptno from emp where job = 'SALESMAN' Group By deptno having count(*) >= 3);

- 2) list employees from research & accounting having at least 2 reporting.

Select * from emp where deptno in (select deptno from dept where dname in ('RESEARCH', 'ACCOUNTING')) and empno in (select mgr from emp group by mgr having count(*) >= 2);

- 3) Display the department location that is having greater than four employees in it.

Select loc from dept where deptno in (select deptno from emp group by deptno having count(*) > 4);

- 4) Display all the employees of dept 30, 20 with there annual salary and having at least 3 employees.

- 5) List the dept name that are having at least 3 emp's but not more than 5 employees in it

\Rightarrow Select dname from dept where deptno in (select deptno from emp Group By deptno having count(*) between 3 and 5);

6) List the dept names that are having at least 3 employees in it.

⇒ Select dname from dept where deptno in (select deptno from emp Group By deptno having count(*) >= 3);

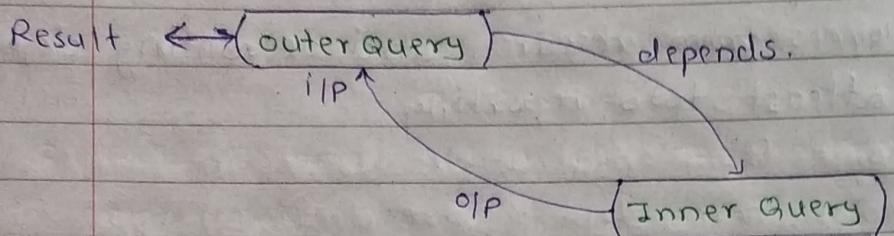
7) list employees from research and accounting dept having atleast two reporting

⇒ Select ename from emp where deptno in (select deptno from dept where dname in ('ACCOUNTING', 'RESEARCH')) and empno in (select mgr from emp Group By mgr having count(*) >= 2);

* Subqueries \Rightarrow

- 1) A queries which is written inside another query is called as a subqueries.

Working Principle of subqueries \Rightarrow



- 1) Inner query executes first and output of a inner query is given to outer query as a input.
- 2) Outer query executes completely and provide the result, therefore we can say outer query depends on a inner query.

Reasons to use Subqueries -

- 1) If we have unkown values in problem statements
- 2) when data is in one table and condition is in another table. #

Note = To compare two columns in subqueries
column should be same or datatype should be same

* Group By

1) Group by is used to group similar type of a data.

Emp			Emp			Emp		
ename	sal	deptno	ename	sal	deptno	ename	sal	deptno
A	100	10	A	100	10	A	100	10
B	200	20	B	200	20	C	300	10
C	300	10	C	300	10	B	200	20
D	400	20	D	400	20	D	400	20
E	500	30						

eg Select count(*), deptno from emp where deptno != 30
group by deptno;

Deptno	Count(*)
20	2 6 6
10	2 8 3

Syntax —

Select col-name, group function from table name where
<filter-condition> group by col-name;

order of execution

- 1) select
- 2) from
- 3) where
- 4) group by

② Any clause after group by clause nature of execution will be group by group only.

③ We have to mention a column name which is used in a group by in a select clause. Other than that it will not allow.

Q) We can use multirow function in a group by clause.

Nature of execution -

Where \Rightarrow Row by row execution

group by \Rightarrow Row by row execution

Select \Rightarrow Group by group execution \Rightarrow After group by all clauses will execute group by group only.

ex. Q1 WAQTD total sal needed to pay each job in the emp table.

Q2 Display emp working in each dept & its avg sal excluding all the emp whose sal is less than their commission.

Q3 Display number of emp only if they are working as manager or analyst and their annual sal should ends with zero in each dept.

Q4 Display number of clerks working in each dept

Q5 Display number of employee working in each dept except for those who are working in dept 10

Q6 Display number of employees getting commission in each dept

Q7 Display number of employees getting sal more than 1600 excluding all the managers in each dept.

Q8 Display avg sal needed to pay all the employee's who are having reporting manager in each job.

Q9 Display max sal given in each designation excluding those whose name starts with 'K'.

Ans \Rightarrow

① Select sum(sal) from emp where group by dept, job.

Select sum(sal), job from emp group by job;

② Select emp, avg(sal) from emp where sal < comm group by dept, job;

Select count(ename), avg(sal) from emp where sal > comm group by dept no;

③ Select count(*), deptno from emp where job in ('Manager', 'Analyst') and sal * 12 like '%.0' group by deptno;

④ Select count(job), deptno from emp where job = 'CLERK' group by deptno;

⑤ Select count(*), deptno
from emp
where deptno != 10
group by deptno;

⑥ select count(*), deptno
from emp
where comm is not null
group by deptno;

⑦ select count(*), deptno
from emp
where sal > 1600 and job != 'MANAGER'
group by deptno

⑧ Select avg(sal), job
from emp
where mgr is not null
group by job;

⑨ Select max(sal), job
from emp
where ename^{not} like 'K%'
group by job;

* Having Clause:

- 1) Having clause is similar to where clause but it is a group by filter condition. i.e. we can apply having by clause only on having groups.

Difference between having and where clause.

Having

- ① Use after group by clause
- ② Order of execution or nature of execution is
- ③ group by group
- ④ Used to filter the groups
- ⑤ Dependents on group by

Where

- ① Use before group by clause
- ② Order of execution or nature of execution is row by row.
- ③ used to filter the records
- ④ multirow functions are not allowed
- ⑤ Independent

Syntax = Select col-name, group function
 from table name
 where <filter-condition>
 group by col-name
 having <group filter condition>

Order of execution

- ⑤ select
- ① from
- ② where
- ③ group by
- ④ having

Nature of execution

- where → Row by row
 Group → Row by row
 having → Group by group
 Select → group by group

Questions →

- 1) GIAQTD hiredate on which at least 3 emp got hire
- 2) Display deptno which has more than 2 emp & total amt required to pay monthly sal of all the employees in that dept should be more than 5000.
- 3) Display the salaries which has repetition in the sal column of the emp table.
- 4) Display the name only if more than one person in the emp of the company has same name.
- 5) Display the number of employees hire into same date
- 6) Display number of employees getting same salary working in same dept.
- 7) Display number of employees whose name starts with vowel in each dept.

Ans. ↗

- ① Select hiredate , count(*) from emp group by hiredate having count(hiredate) ≥ 3 ;
- ② Select deptno , count(*) from emp group by deptno having count(*) > 2 and sum(sal)
- ③ Select count(sal) ^{sal} from emp group by sal having count(sal) > 1
- ④ Select ename from emp group by ename having count(*) > 1
- ⑤ Select count(*), hiredate from emp group by hiredate having count(hiredate) > 1
- ⑥ Select count(*), sal, deptno from emp group by deptno having count(sal) > 1

⑥ Select count(*), deptno, sal from emp group by deptno,
sal having count(sal) > 1;

⑦ Select count(*), ename, deptno from emp where substr(ename, 1, 1) in ('A', 'E', 'I', 'O', 'U') group by deptno, ename;

1.
Having

* Order by Clause

① Is used to arrange the records according to order (Ascending or descending)

② asc → used to display records in ascending order.

desc → used to display records in descending order

③ By default records will arranged in ascending order

example

① Select * from emp order by ename; //default ascending.

② Select * from emp order by ename asc; // Ascending order

③ Select * from emp order by ename desc; //descending

④ Select * from emp order by s; // used to display 5th column order in asc.

⑤ ~~Select~~ select * from emp order by comm asc;

* Joins :

Joins are used to join (combine) two or more tables.

- 1) Cartesian Join / cross Join
- 2) Inner Join
- 3) Outer Join
 - a) Left Outer Join
 - b) Right Outer Join
 - c) Full Outer Join
- 4) Natural Join
- 5) Self Join

① Cross Join / Cartesian Join :-

- 1) It is a type of a join where we get mashed records as well as unmashed records.
- 2) 1st record of one table is joined with all other records of another table.

Syntax :

Oracle - Select * from table1, table2;

ANSI - Select * from table1 cross join table2;

Example: \Rightarrow Select * from emp cross join dept;

\Rightarrow Select * from emp, dept;

EMP		Dept		ename	dname
ename	deptno	deptno	dname	A	Sales ✓
A	10	10	Sales	B	Sales X
B	20	20	Acc	A	Acc X
		30	Research	B	Acc ✓
				A	Research X
				B	Research X

(2) Inner Join :-

- 1) It is a type of join where we get only matched records of both the tables by specifying join cond'.

(→ replaced by inner join)

Syntax :-

(where) Replaced by where cond'

Orc

Select * from table1, table2 where {join cond'}

(Replaced by) →

ANSI

Select * from table1 inner join table2 on
{join cond'} where {filter conditions}

example: Select ename, dname from emp e, dept d where e.deptno

Orc \Rightarrow = d.deptno;

ename	dname
A	Acc
B	Sales

ANSI - Select ename, dname from emp e inner join dept d on
e.deptno=d.deptno;

Inner Join Queries

Q. ① Name of the emp and loc of all the employees.

⇒ Select ename, loc from emp e, dept d where
e.deptno = d.deptno;

Q. ② GIAQTD dname & sal for all the emp working in acc.

⇒ Select dname, sal from emp e, dept d where
e.deptno = d.deptno and dname = 'ACCOUNTING';

⇒ Select dname, sal from emp e inner join dept d
on e.deptno = d.deptno and where dname = 'ACCOUNTING'.

Q. ③ GIAQTD dname, and annual sal for all the employee
whose sal is more than 2340.

⇒ Select dname, sal * 12 from emp e, dept d where
e.deptno = d.deptno and sal > 2340;

⇒ Select dname, sal * 12 from emp e inner join dept d
on e.deptno = d.deptno where sal > 2340;

Q. ④ GIAQTD ename, dname for employees having character
A in their dname.

⇒ Select ename, dname from emp e, dept d where
e.deptno = d.deptno and dname like '%.A%';

⇒ Select ename, dname from emp e inner join dept d
on e.deptno = d.deptno where dname like '%.A%';

Q. ⑤ GIAQTD ename, dname for all the employees working
as salesman.

⇒ Select ename, dname from emp e, dept d where
e.deptno = d.deptno and job = 'SALESMAN';

Q. ⑥ CHAQTD dname & job for all the employees whose job and dname starts with character S.

⇒ Select dname, job from emp e, dept d where e.deptno = d.deptno and job like 'S%' and dname like 'S%';
 $\therefore \text{substr(job, 1, 1)} \text{ in ('S')}$

Q. ⑦ CHAQTD dname & mgr no for employees reporting to 7839.

⇒ Select dname, mgr from emp e, dept d where e.deptno = d.deptno and mgr = 7839

Q. ⑧ CHAQTD dname & hiredate for employee hiredate after 83 into accounting or research dept.

⇒ Select dname, hiredate from emp e, dept d where e.deptno = d.deptno and hiredate to_char(hiredate, 'yy') > 83 and dname in ('Accounting', 'Research');

Q. ⑨ CHAQTD ename & dname of the employees who are getting comm in dept 10, 30

⇒ Select ename, dname from emp e, dept d where e.deptno = d.deptno and comm is not null and e.deptno in (10, 30);

Q. ⑩ CHAQTD dname & empno for all the employees whose empno are (7839, 7902) and are working in loc new york.

⇒ Select dname, empno from emp e, dept d where e.deptno = d.deptno and empno in (7839, 7902) and loc = 'new york';

Q.11 GIAQTD loc & avg(sal) given for each location by excluding all the employees whose second char is A in their name.

⇒ Select loc, avg(sal) from emp e, dept d where e.deptno = d.deptno and group by loc and ename not like '_A%' group by loc;

Q.12 GIAQTD name of the emp and his loc if employee is working as manager and working under the employee whose empno is 7839

⇒ Select ename, loc from emp e, dept d where e.deptno = d.deptno and job = 'MANAGER' and mgr = 7839;

Q.13 GIAQTD dname and employee id's of all the employees who are clerks & having reporting managers.

⇒ Select dname, empno from emp e, dept d where e.deptno = d.deptno and job = 'CLERK' and mgr is not null;

Q.14 GIAQTD dname and total salary given to that dept if there are atleast 4 employees working for each dept.

⇒ Select dname, sum(sal) from emp e, dept d where e.deptno = d.deptno group by dname having count(*) >= 4;

Q.15 COUNT dname & number of employees working in each dept only if there are manager or clerks.

⇒ Select dname, count(*) from emp e, dept d where e.deptno = d.deptno and job in ('MANAGER', 'CLERK') group by dname;

2/04/21 * Outer Join

① Left Outer Join :- we will get matched records of both the table as well as unmatched records of the left table.

Syntax :-

ANSI - Select * from table1 left join table2 on table1.col-name = table2.col-name ;

ORC - Select * from table1, table2 where table1.col-name = table2.col-name (+);

emp				dept		Left Join	
ename	deptno	dname	deptno			ename	dname
A	10	CS	10			A	CS
B	20	IT	20	⇒		B	IT
C	30	Mech	30			C	Mech
D	50	Civil	40			D	

example ⇒ Select ename, dname from emp e, dept d where e.deptno = d.deptno (+);

ORC

Select ename, dname from emp e left join dept d on e.deptno = d.deptno ;

ANSI

② Right Outer Join :- In this we will get matched records of both the tables as well as unmatched records of right table.

Syntax :-

ANSI : Select * from table1 right join table2 on
table1.col-name = table2.col-name ;

Or : Select * from table1, table2 where
table1.colname ~~not~~ = table2.colname ;

Right Join

ename	dname
A	CS
B	IT
C	Mech
	Civil

example ⇒ Select ename, dname from emp1e, deptd . where
e.deptno ~~not~~ d.deptno

ANS1 →
Select ename, dname from emp1e Right join
deptd on e.deptno = d.deptno;

- ③ Full Outer Join :- we will get matched records of both the tables as well as unmatched records of both the table (right & left).

For full outer join we don't have a oracle syntax.

Syntax :-

ANS2 - Select * from table1 full join table2 on table1.col-name = table2.col-name;

Full Outer Join

ename	dname
A	CS
B	IT
C	MECH
D	Civil

example Select ename, dname from emp e full join dept d on e.deptno = d.deptno;

ANS2

Natural Join

demo		
ename	deptno	loc
A	10	Pune
B	20	Mumbai
C	30	Pune
D	40	Nagpur

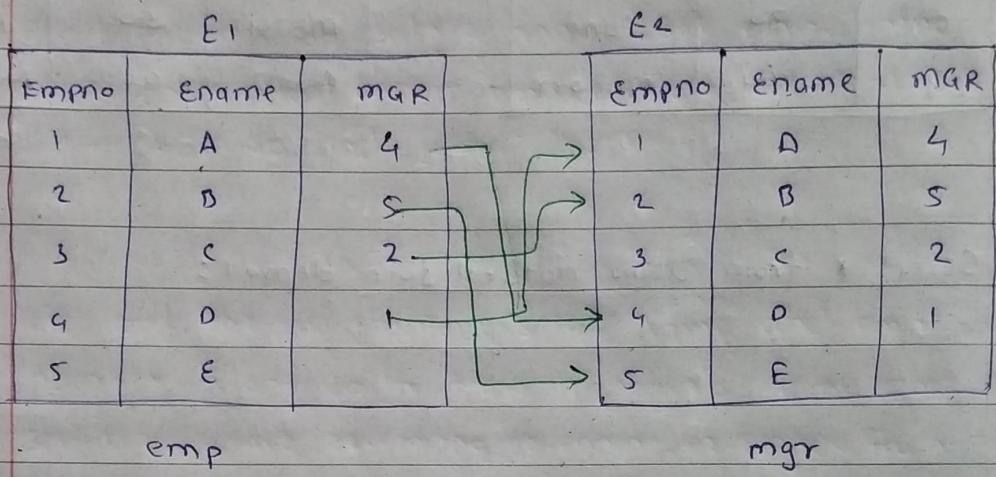
demo1			loc
dname	deptno	Address	loc
CS	10	PUNE	PUNE
IT	20	Mumbai	Mumbai
METH	30	PUNE	PUNE
Civil	40	Nagpur	Kolhapur

Select * from demo natural join demo1;

⑤ Self Join :- Joining - same table

> Select emp.* , sal+100 from emp;

> select e1.empno, e1.ename, e2.empno, e2.ename from emp e1, emp e2 where e1.mgr = e2.empno;



> Select e1.empno, e1.ename, e2.empno, e2.ename, e3.empno, e3.ename from emp e1, emp e2, emp e3 where e1.mgr = e2.empno and e2.mgr = e3.empno;

> Select e1.ename, e2.ename from emp e1, emp e2 where e1.mgr = e2.empno and e1.ename = 'BLAKE';

- 1] Join to same tables is called as a self join
- 2] Self join is majorly used to identify employee manager relation.

- ⑥ Display the name and salary of the employee who is working as a ~~placks~~ & blocks manager.

⇒

Select e1.ename, e2.sal, e2.ename from emp e1, emp e2 where e1.mgr = e2.empno and e1.ename = 'BLAKE';

Q Display name of the emp who are working under king.

⇒ Select e1.ename, from emp e1, emp e2

where e1.mgr = e2.empno and e2.ename = 'KING';

Q Display name of the emp as well as deptname
who is working as a Ford's manager.

⇒ Select e2.ename, dname .from emp e1, emp e2 ,dept
where e1.mgr = e2.empno and e2.deptno = d.deptno,
e1.ename = 'FORD';

Q Display details of the employee as well as its manager
managers location and sal.

⇒ Select e1.* , e2.loc , e3.sal from emp e1, emp e2, emp e3
from where e1.mgr = e2.empno and e2.mgr = e3.empno and
e3.deptno = d.deptno

Q Display details of 'ADAMS' manager & location of
managers manager

Q Display details of the emp who are working for
'SMITH'S managers manager.

⇒ select e4.* from emp e1, emp e2, emp e3, emp e4

where e1.mgr = e2.empno and e2.mgr = e3.empno and
e3.mgr = e4.empno and e1.ename = 'SMITH'

Q Display name and location of the employee who is
working as a 'BLAKE'S manager.

⇒ select e2.name, loc from emp e1, emp e2 ,dept d where
e1.mgr = e2.empno and e1.ename = 'BLAKE' and
e2.deptno = d.deptno;

Q Display details of the employees who was hired before Smith manager.

⇒ Select e3.* from emp e1, emp e2, emp e3 where e1.mgr = e2.empno and e2.mgr = e3.empno and e1.ename = 'SMITH' and e3.hiredate < e2.hiredate;

Assignment

Q List ename, job, annual sal, deptno, dname who earn 30000 per year and who are not clerks.

⇒ Select ename, job, sal * 12 annual_sal, emp.deptno, dname from emp, dept where emp.deptno = dept.deptno and sal * 12 > 30000 and job != 'CLERK';

Q List out the all employees by name and employee number along with their manager's name and empno.

⇒ Select e1.ename, e1.empno, e2.ename, e2.empno from emp e1, emp e2 where e1.mgr = e2.empno (+);

Q Display ename, dname even if there no employees working in a particular department

⇒ Select ename, dname from emp, dept where e.deptno (+) = d.deptno;

Q Display the department name along with total salary in each department.

⇒ Select dname, sum(e1.sal) from emp e1, dept d where e1.deptno = d.deptno group by dname;

Q Display employee name and department name for each emp.

⇒ Select e1.ename, d.dname from emp e1, dept d where e1.deptno = d.deptno;

- 6) Display location name of the emp who earn comm.
 ⇒ Select d.loc, e.ename from dept d, emp e where
~~e.deptno = d.deptno and comm is not null;~~
- 7) Display dept name of the emp who earn min salary and
 have no reporting manager.
 ⇒ Select d.dname from dept d, emp e where
~~e.deptno = d.deptno and sal = (select min(sal) from emp)~~
~~where mgr is null);~~
- 8) Display dept name, loc of all the employees who are
 reporting to smith.
 ⇒ Select d.dname, d.loc from dept d . where emp.deptno =
~~d.deptno and mgr = (select empno from emp where
 ename = 'SMITH');~~
 ~ No rows selected.
- 9) List all the dept name and location of all the salesman
 manager's manager.
 ⇒ Select dname, loc from dept d, emp e where d.deptno =
~~e.deptno and job in ('SALESMAN', 'MANAGER') and empno~~
~~in (select mgr from emp where job = 'MANAGER');~~
- 10) List employees who are working in research dept and they
 are manager.
 ⇒ Select ename from emp e, dept d where d.deptno = e.
~~deptno and dname = 'RESEARCH' and job = 'MANAGER'~~
- 11) Display the number of emp who are getting salary less
 than the blake's manager.
 ⇒ Select count(*) from emp e where sal < (select sal from
 emp where empno = (select mgr from emp where
 ename = 'BLAKE'));
 ~ 13

12) list the employee deptname and location of all the emp's who are analyst, reporting to BLAKE.

⇒ Select ename, dname, loc from emp e, dept d where e.deptno = d.deptno and job = 'ANALYST' and mgr = (Select empno from emp where ename = 'BLAKE');

13) Display the employee names, Hiredate, comm of FORD's manager.

⇒ Select ename, hiredate, comm from emp e where empno = (Select mgr from emp where ename = 'FORD')

14) Display ename, dname of all the employees whose sal less than avg sal of dept 30.

⇒ Select ename, dname from emp e, dept d where e.deptno = d.deptno and sal < (select avg(sal) from emp where deptno = 30 group by deptno);

15) Display ename, dname and loc of all the emp's who are working for jones.

⇒ Select ename, dname, loc from emp e, dept d where e.deptno = d.deptno and mgr = (select empno from emp where ename = 'JONES');

16) Display ename, dname of all the employees whose name starts with S.

⇒ Select ename, dname from emp e, dept d where e.deptno = d.deptno and ename like 'S%';

17) list the dname who are not having any employee in it.

⇒ Select dname from dept where deptno not in (select distinct deptno from emp);

⇒ Select dname from emp, dept where emp.deptno(+)=dept.deptno and ename is null;

18) Display all the dept names irrespective of any emp working in it or not, if an emp is working display his name.

⇒ select dname, ename from emp e right outer join dept d on e.deptno = d.deptno;

- 19) CWAGTD emp name, job, dname, loc of all employees who are working as actual managers and works at chicago.
- ⇒ Select ename, job, dname, loc from emp e, dept d where e.deptno = d.deptno and empno in (select mgr from emp) and loc = 'CHICAGO';
- 20) list the dept names in which the employees are hired betⁿ 1st of JAN 1981 and 31st Dec 1982 with salary more than 1800.
- ⇒ Select dname, hiredate from emp e, dept d where e.deptno = d.deptno and hiredate between '01-JAN-1981' and '31-DEC-1982' and sal > 1800;
- 21) Display 2nd least salary from emp table.
- ⇒ Select min(sal) from emp where sal > (Select min(sal) from emp);
- ⇒ Select min(e2.sal) from emp e1, emp e2, ~~emp e3~~ where e1.sal < e2.sal;
- 22) List the employees whose annual sal is greater than 1500 and who are joined before 1982 only.
- ⇒ Select ename, hiredate from emp where sal * 12 > 1500 and hiredate < '01-JAN-1982';
- 23) Display emp name along with their manager name.
- ⇒ Select e1.ename, e2.ename from emp e1, emp e2 where e1.mgr = e2.empno;
- 24) Display emp name and his dept name for the employees whose name starts with 'S'.
- 1) ⇒ Select ename, dname from emp e, dept d where e.deptno = d.deptno and substr(ename, 1, 1) = 'S';
- 2) ⇒ Select ename, dname from emp e, dept d where e.deptno = d.deptno and ename like 'S%';