17/3/21

SQL

Data :→

Any meaningful thing which is present around us in the form of numbers, characters or physical objects is known as data.

Information :→

A collection of data is known as Information.

Subject	Marks
A	60
B	70
C	80
Average	80%

} Information data.

→ Information.

Database :→

It is a storage or a container in which we store a data in organised manner

↳ Based on storing the data into the database, the database is classified into 4 different modules,

* File System DBM
* Hierarchical DBM
* Network DBM
* Relational DBM

1. File System DBM :→

↳ This model was introduced during the year of 1950's & 1960's. In this model where all the data & information will be stored in the form of numerous physical files.

↳ Where the files had no relationship among themself due to which had the following drawbacks

* The data redundancy was present.
* The time taken to search for data or information will be more.

Note: The file-system DBMS is recommended only if the data to be maintained in minimum.)

2. Hierarchical DBMS :→

↳ In this model the data or information will be stored in the form of tree like a structure where there will be a partical relationship among the file due to which the data redundancy was reduced & searching became faster, this model was effecient only when compared with file-system DBMS

3. Network DBMS :→

↳ Just like Hierarchical DBMS, here also we store the data or information in the form of tree like structure where the files U be connected eachother over the network. Due to which the data redundancy was reduced max. & searching became faster when compared with previous models

↳ The only challenge of the model was to manage the network bcoz even if a single node fails, the entire model will be shutdown.

4. Relational DBMS :→

↳ This model was introduced by E.F codd during the year of 1970's

↳ The data or information will be stored in the form of numerous tables.

↳ The tables will be connected to eachother with the help of primary key & foreign key due to which the data redundancy was completely reduced & searching became faster & this

↳ This model was effectiective & effecitient model compared with all the previous model.

SQL :→

↳ The first name of SQL is Sequal Quere language.
↳ It also stands for Structured Quere Language.
↳ This language was introduced during the year of 1970's by Donald D Chamberlin & Raymond F Boyce with the help of the organizedion called as IBM.
↳ SQL is a language which is used to communicate with the database.

DBMS :→

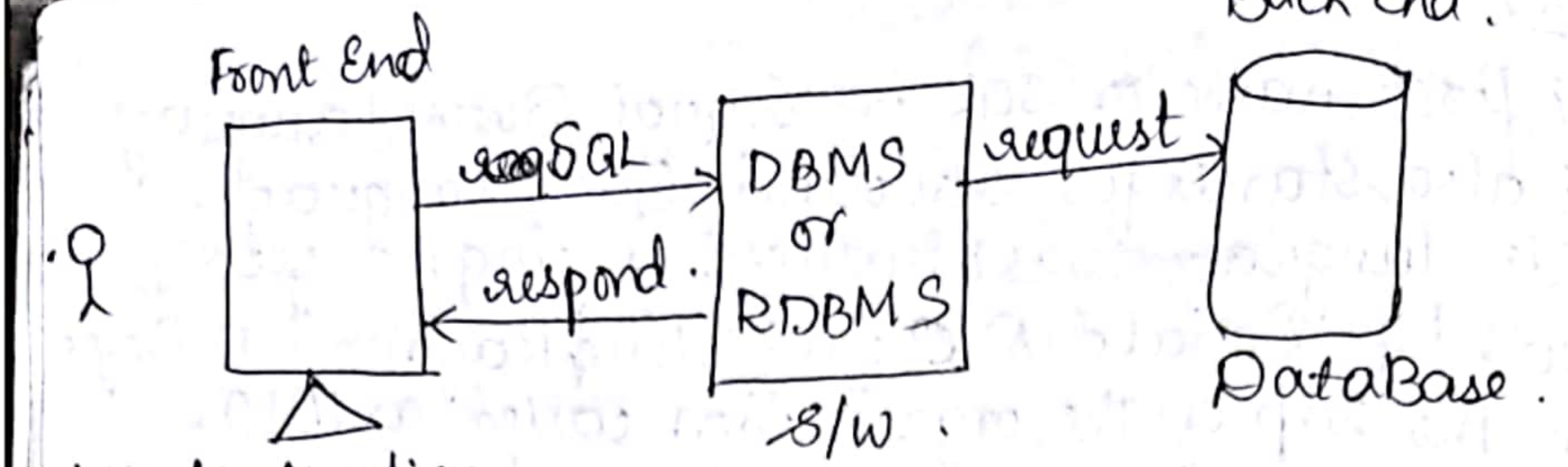
↳ DBMS stands for Data Base Management System.
↳ It is a software which is used to store & manage the data & also allows us to modify the data,
↳ It can be insertion, updation or deletion of the data.
↳ Examples Software of DBMS are,

* Foxpro,
* For Base
* D Base & so on.

RDBMS :→

↳ RDBMS stands for Relation
↳ It is a advanced version of DBMS which allows us to ~~edit~~ access the data more effeciently
↳ It manages to store the data in the form of tables compared with DBMS.
↳ The RDBMS has got enhanced security features, good performance & also it can store huge volume of data into the database.
↳ The Example Software of RDBMS are,

* Oracale
* My-SQL
* MS-access
* Mongo DB
* SQL Server
* Sy Base & so on.



18/3/21

Rows / Columns

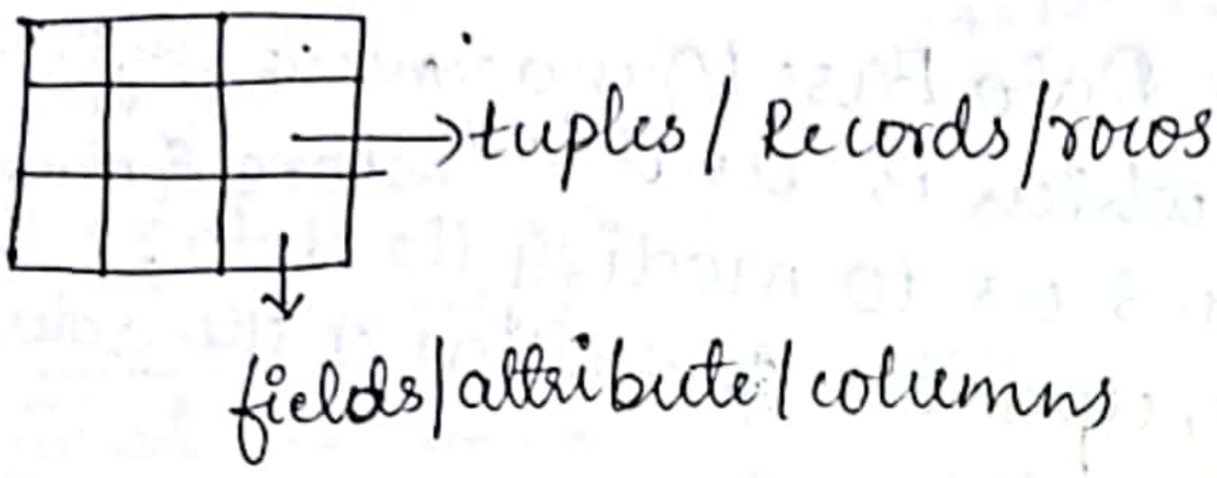


Table / Relation / Entity.

Data Integrity

↳ It is used to restrict the invalid data entering data entering into the table.

↳ We can achieve the data integrity by two ways

* Data types
* constraints.

↳ Table without data integrity

Employe Table:

Empno	Ename	Dno	Dname	Hiredata	Sal	Phno
01	Dinga	100	Testing	14-Feb-21	90,000	143
Dingi	02					

Invalid / Wrong data.

↳ Data not found.

(or)

↳ No rows selected.

Data Types

↳ Data types are used to specify the data to be stored in the each columns of the tables.

↳ The different types of Data types are,

1. Numeric Data type
2. Non-Numeric Data type.
3. Date Data type

1. Numeric Data type

↳ **Number**

It is a type of data type which is used to store numeric values without the precision for the selected columns of the table.

Syntax:

Number (size / length)

Ex: 9999 Number(4)

↳ **Number with precision**

It is a type of data type which is used to store numeric values along with the precision for the selected columns of the table.

Syntax:

Number(size , precision)

Ex:

99.999 Number(5,3

2. Non-Numeric Data type.

↳ **Char data**

It is fixed in length which is used to store alpha numeric values or only the character for the selected columns of the tables.

Syntax:

char(size / length)

↳ **Varchar / varchar2**

It is variable in size which is used to store. alpha numeric value or only the characters for th selected columns of the tables.

↳varchar datatype stores upto 2000 values
↳varchar2 datatype stores upto 4000 values

Syntax:

varchar/varchar2 (size/length).

3. Date Datatype

It is a type of datatype which is used to store date type of data into the selected columns of the table. The default format of date datatype is represented by,

DD-MON-YY. i.e., date-month-year

Employee

Empno	Ename	Dno	Dname	Hiredate	Sal	Pho	Panno
01	Dinga	100	Testing	14-Feb-21	90,000	143	
↓	↓	↓	↓	↓	↓	↓	↓
Number(5)	varchar(20)	Number(4)	varchar2(15)	Date	Number(7,2)	Number(10)	char(10

19/3/21

Constraints

↳Constraints are used to restrict the invalid data enting into the table.
↳The different types of constraints are,

* Not-Null
* Unique
* Primary key
* Foreign key
* Check constraints

*Not-Null Constraints

↳ It will ensure that some atleast some value must value as to be present in the selected colums of the table

*Unique Constraints

↳ It will ensure the duplication of values are not allowed in selected columns of the table.

*Primary-key Constraints

↳ It is a combination of not-null & unique constraints
↳ Primary key is used to identify the records uniquely
↳ Only one primary key can be created in a single table
↳ Creating a primary key is not mandatory but it is highly recommended to create
↳ The columns which are elegible to become as a primary key is called as condidate keys.
↳ The columns which are eligibel to become as a primary key column but not chooses as a primary key columns is called as alternative key

*↳ Check constraints

↳ Whenever the costumer ask for the additional validation in the requirement then we make use of check constraints.

Empno	Ename	Dno	Dname	Hiredate	Sal	Phno
01	Dinga	100	Testing	14-Feb-2021	90,000	143
02	Dingi	200	develop	01-Jan-2021	45,000	123
03	Monga	100	Testing	14-NOV-2021	85,000	456

U + NN

NN + U

NN

NN.

Check(sal >2000)

Check(length(Phno=10))

* Foreign key constraints

↳ It is also called as referencial integrity constraint.
↳ Foreign key is used to create the relationship b/w the table
↳ To create the relationship b/w the tables the common column as to be present in both the tables.
↳ More than one foreign key is allowed in the single table.
↳ Foreign key will accept duplication of ~~keys~~ values and null values

Employee

Eno	Ename	Dno	Dname	Hiredate	Sal	Phno
01	Dinga	100	Testing	14-Feb-2021	90,000	143
02	Ding-i	200	Develop	01-Jan-21	45,000	123
03	Manga	100	Testing	14-Nov-21	25,000	456

Check(Length (Phno=10))

U + NN. | NN. | Foreign key. | NN | Check(Sal>2000) | U + NN

Dept

Dno	Dname	Loc
100	Testing	Bloc
200	Dev	Blore
300	Testing	Blore

U + NN Primary key | U + NN.

SQL Statements/Languages.

* DQL (Data Query Language)
  - ↳ Select
* DDL (Data Definition Language)
  - ↳ Create
  - ↳ Alter
  - ↳ Rename
  - ↳ Truncate
  - ↳ Drop
* DML (Data Manuplication Language)
  - ↳ Insert
  - ↳ Update
  - ↳ Delete
* TCL (Transaction Control Language)
  (or) DTL (Data Transaction Language)
  - ↳ Commit
  - ↳ Rollback
  - ↳ Savepoint
* DCL (Data Control Language)
  - ↳ Grant
  - ↳ Revoke.

1. DQL (Data Query Language)

Select Statement:

↳ It is used to form the queries

↳ The capabilities of select statement are;

1. Projection - It is used to select the columns of the table.
2. Selection - It is used to select the rows of the table
3. Joins - It is used to fetch the data from the multiple tables.

Syntax;

Syntax :

Select * / {[Distinct] column_Name/Expression [Alias_Name]} from Table_Name ;

Basic Commands & Basic Queries.

(1)
SQL>SHOW USER;
USER is SCOTT
↳Show User : it shows which user is login

(2)
SQL>CONNECT or CONN.
Enter user-name : HR
Enter password : ***** (Tiger)
Connected..
SQL>SHOW USER;
USER is HR.
↳ This command is used to switch from one user to another user without exit.

(3). SQL> CL SCR.
↳ It is used to clear the screen.

(4). SQL> SELECT * FROM TAB;

TNAME	TABTYPE	CLUSTERID
END.		
DNO		

↳ It is used to check which all the data entered in the table

(5) SQL>DESC DEPT (DESC-Describe)
↳ It is used to know the structure of the table

Name	Null?	Type

⑥. SQL> SHOW PAGESIZE
pagesize 14
SQL> SHOW LINESIZE
linesize 80
SQL> SET PAGESIZE 100;
SQL> SET LINESIZE 100;
SQL> SELECT * FROM EMP;

*Write a query to display employee table:

SQL> SELECT * FROM EMP;

*Write a query to display all the employee's ename

SQL> SELECT Ename from emp;

*Write a query to display all the employee's name and job:

SQL> SELECT Ename, JOB from emp;

⑦. SQL> EDIT.
↳ This command is used to edit the previous command i.e., Edit (Ed)
↳ it will navigate to notepad, we can correct it there then close tab & save
↳ then run using "/"
↳ then command will run.

Literals

↳ Literals can be number, date or char literals
↳ date & character literals should be enclosed both single codes(' ')
↳ when we want to perform operations on independ literals then we need to make us of dummy table called as dual tables.

Examples.

SQL >SELECT 4+2 FROM DUAL;

4+2
6

SQL>SELECT 4+2 FROM EMP;

4+2
6
6
⋮
6

} 14 times.

14 rows selected.

SQL 7 SELECT 4+2 FROM DEPT;

4+2
6
6
6
6

4 rows selected.

SQL >SELECT ENAME, SQL, SQL*12 FROM EMP;

ENAME	SQL	SQL*12
SMITH	800	9600
⋮	⋮	⋮

14 rows selected.

Alias name :-

↳ It is a alternative name which is given for the column-name

↳ Between the column name & alias name we can make use of the optional keyword called "as"

↳ when we want to provide the space in the alias name then the alias name should be enclosed by double codes.

↳ alias name is used only for the user reference which is not going to effect the original column.

Examples:

SQL> SELECT MGR AS MANAGER FROM EMP;

SALARY

14 rows selected

SQL > SELECT ENAME, SAL, SAL * 12 "ANNALSALARY" FROM EMP.

ENAME	SAL	ANNAL SALARY.
SMITH	800	9600

14 rows selected.

Concatination Operator.

↳ It is used to merge the literalls or it can be the columns to concate them we make use of two vertical bars which is represented by ||

Ex:

* SQL > SELECT 'SQL || CLASS' FROM DUAL;

SQLCLASS

* SQL > SELECT ENAME || JOB FROM EMP;

ENAME||JOB.

SMITHCLERK

14 rows selected.

* SQL > SELECT ENAME || ' YOURSALARYIS' || SAL FROM EMP;

SMITH YOUR SALARY IS 800.

Operators

↳ Operators are used to perform specific kind of operations on the operands based on certain conditions. The different types of Operators are,

* Arithmetic Operator (+, -, *, /, %)
* Relational Operator (>, <, >=, <=, =, !=, <>)
* Logical Operator (and, or, Not)
* Special Operator (IS, like, between, In)

Rules for Precedince of Operator.

Order Evaluated	Operator
1.	Arithmetic
2.	concatenation
3.	Relational.
4.	Is, Like, in
5.	Between.
6.	Not
7.	and
8.	or.

WHERE clause

Syntax :

SELECT */{[DISTINCT] column-NAME / Expression [ALIAS-NAME]} from Table-NAME

Where column-name < condition > ;

Where clause is used to restrict the number of records to be displayed i.e., it gives the specific records as the output based on the certain condition.

↳ Display all the employees who are earning more than 3000

SQL> SELECT * FROM EMP WHERE SAL>3000;

EMPNO ENAME JOB MGR HIREDATE SAL C DEPTNO
KING

↳ Display all the employees who Job is manager.

SQL>SELECT * FROM EMP WHERE JOB='MANAGER';

↳ Display all the employees except who work for Deptno10

SQL>SELECT * FROM EMP WHERE DEPTNO != 10;

↳ list all the clerks in Depno=30.

SQL> SELECT * FROM EMP WHERE JOB='CLERK'
AND DEPTNO=30;

↳ Display all the salesman who is earning more than 1400 & belongs to Depno=30.

SQL> SELECT * FROM EMP WHERE JOB='SALESMAN'
AND SAL>1400 AND DEPTNO =30;

↳ Display all the employees who is working in Deptno=10, 20 or 30

SQL>SELECT * FROM EMP WHERE DEPTNO=10
OR DEPTNO=20 OR DEPTNO=30;

IN OPERATOR

IN Operator is used to evaluate the multiple values when we make use of the OR-operator more then one-time instead of that we can make use of the special-~~character~~ operator called as IN-operator

Display all the employes who is working in Deptno 10, 20 & 30.

SQL > SELECT * FROM EMP WHERE DEPTNO IN(10,20,30);

Display all the employe who is working as clerk or the manager

SQL > SELECT * FROM EMP WHERE ~~DEPTNO~~ JOB IN('CLERK','MANAGER');

Between Operator

↳ It is used to display the output for the range of the values.

↳ whenever we use Between Operator, it should be followed by and operator also.

Display all the employees who is earning the salary between the range of 1000 & 4000

SQL > SELECT * FROM EMP WHERE SAL BETWEEN 1000 AND 4000;

Syntax:

SELECT * from table_name where column_name operator_name V1 and V_2;

Display all the employees whose hireddate is between 17-Dec-80 & 3-Dec-81.

```
SQL>SELECT* FROM EMP WHERE HIREDATE
    BETWEEN '17-DEC-80' AND '3-DEC-81';
```

IS Operator

↳ It is used to compare with the null / not-null values

Display all the employees who don't have reporting manager.

```
SQL>SELECT * FROM EMP MGR IS NULL;
```

Display all the employees who are getting commission.

```
SQL>SELECT * FROM EMP COMM IS NOTNULL;
```

LIKE Operator

↳ It is used to perform pattern matching
↳ we can achieve the pattern matching with the wildcards i.e., % and _

↳ modulus(%) : It matches with single to n-charac
↳ underscore(_) : It matches with single character.

Display all the employees who name starts with S

```
SQL>SELECT * FROM EMP WHERE ENAME LIKE 'S%';
```

Display all the employees who name is having letter L as second character.

```
SQL>SELECT * FROM EMP WHERE ENAME LIKE
    '_L%';
```

Display all the Employees who salary is having exactly 4 values

```
SQL>SELECT *FROM EMP WHERE SAL LIKE '____';
```

Assignment Questions.

1. Display all the employees who are having employee number as 7902 or 7934.
2. Display all the employees who howe joined in the month of Jan.
3. Display all the employees working in Deptno 10, 20 or 30 & Employees working as Clerk, Salesman or analyst.
4. List all the employees who name not starts with letter 'A' & who's salary is more than 1000
5. List the department details which are having letter 'O' in their location as well as in there deptment name.
6. Display all the employees whose job as string man in it.
7. Display all the employees who are salesman having letter 'E' as the last but one character in employee name & salary should be howing exactly 4 values in it.
8. Display all the values employees who have joined after the year 81.
9. Display all the manager whose salary is ending with zero.
10. Display all the employees whose name starts with letter 'A' & ends with letter 'N'.

Employee table

Empno	Ename	Job	Mgr	Hiredate	SAL	COMM	Deptno
7369	Smith	Clerk	7902	17-Dec-80	800		20
7499	Allen	Salesman	7698	20-Feb-81	1600	300	30
7521	Ward	Salesman	7698	22-Feb-81	1250	500	30
7566	Jones	manager	7839	02-Apr-81	2975		20
7654	Martin	Salesman	7698	28-Sep-81	1250	1400	30
7698	Blake	manager	7839	01-May-81	2850		30
7782	Clark	manager	7839	09-Jun-81	2450		10
7888	Scott	analyst	7566	19-Apr-87	3000		20
7839	King	president		17-Nov-81	5000		10
7844	Turner	Salesman	7698	08-Sep-81	1500		30
7876	Adams	clerk	7788	23-May-87	1100		20
7900	James	clerk	7698	03-Dec-81	950		30
7902	Ford	analyst	7566	03-Dec-81	3000		20
7934	Miller	clerk	7782	23-Jan-82	1300		10

14 rows selected.

Dept table

Deptno	Dname	Loc
10	Accounting	NewYork
20	Research	Dallas
30	Sales	Chicago
40	Operations	Boston

① SQL>SELECT * FROM EMP WHERE EMPNO ~~IN(7902,~~ IN(7902,7934);

② SQL > SELECT * FROM EMP WHERE HIREDATE ~~LIKE '%JAN%';~~ LIKE '%JAN%';

③ SQL>SELECT * FROM EMP WHERE DEPTNO IN(10,20,30); AND

SQL >SELECT * FROM EMP WHERE JOB IN('CLERK','SALESMAN','ANALYST');

④ SQL> SELECT * FROM EMP WHERE ENAME LIKE 'A%';
AND
SQL> SELECT * FROM EMP WHERE SAL > 1000;

⑤ SQL> SELECT * FROM DEPT WHERE LOC LIKE 'O%' AND DNAME LIKE 'O%';

⑥ SQL> SELECT * FROM EMP WHERE JOB LIKE '%man%';

⑧ SQL> SELECT * FROM EMP WHERE HIREDATE >='01-JAN-82';
>31-Dec-82

⑩ SQL> SELECT * FROM EMP WHERE ENAME LIKE 'A%N';

⑨ SQL> SELECT * FROM EMP WHERE JOB LIKE 'MANAGER' AND SAL LIKE '%0';

⑦ SQL> SELECT * FROM EMP WHERE JOB='SALEMAN' AND ENAME LIKE '%E_' AND SAL LIKE '____';
ENAME, SAL*12 "ANNUAL SALARY"

⑪ SQL> SELECT * FROM EMP WHERE ENAME='SMITH';

⑫ SQL> SELECT * FROM EMP WHERE SAL=3000 AND JOB='ANALYST';

⑬ SQL> SELECT * FROM EMP WHERE JOB='MANAGER' AND MGR=7839;

⑭ SQL> SELECT * FROM EMP WHERE JOB NOT IN('SALESMAN','MANAGER');

11. Write a Query to display the annual salary of the employees who name is smith.
12. Write a Query to display all the employees who are earning 3000 as a salary and working as a analyst.
13. Write a Query to display all the employees who is working as a manager & reports to employees 7839
14. Write a Query to display all the employes except salesman & manager.
15. Write a Query to display details of the employees those who have hired during the year of 1981
16. Write a Query to display all the employees who have not hired during 1982
17. Write a Query to display all the employees those who are not getting salary.
(18) List the Employes those who are not working as manage & clerk in deptno 10,20 with the salary in the range of 1000 to 3000
19. Display all the employees who job is manager & who dont have reporting manager.
20. List all the employes who's name starts with 'S' or 'A'
21. Display all the employes who are getting 2500 and excess salary in Deptno-20
22. List all the employes who's name is having letter 'R' in the third position.
23. List all the employee who are having exactly 5 character in their job.
24. List all the employees who's name is having atleast two 'L' in it.

(15) SQL > SELECT * FROM EMP WHERE HIREDATE BETWEEN '01-JAN-81' AND '31-Dec-81

(16) SQL > SELECT * FROM EMP WHERE HIREDATE NOT BETWEEN '01-JAN-82' AND '31-DEC-82';

(17) SQL > SELECT * FROM EMP WHERE SAL IS NU

(18) SQL > SELECT * FROM EMP WHERE JOB NOT IN ('MANAGER', 'CLERK') AND DEPTNO (10, 20) AND SAL BETWEEN 1000 AND 3000;

(19) SQL > SELECT * FROM EMP WHERE JOB = 'MANAGER' AND MGR IS NOT NULL;

(20) SQL > SELECT * FROM EMP WHERE ENAME LIKE 'S%', 'A' OR ENAME LIKE 'A%';

(21) SQL > SELECT * FROM EMP WHERE DEPTNO = 20 AND SAL >= 2500;

(22) SQL > SELECT * FROM EMP WHERE ENAME LIKE '_ _R%';

(23) SQL > SELECT * FROM EMP WHERE JOB LIKE '_____';

(24) SQL > SELECT * FROM EMP WHERE ENAME LIKE '%LL%';

(25) (OR)

SQL > SELECT * FROM EMP WHERE HIREDATE LIKE '%81%';

SQL
23/

SQL Statements / Languages

② Data Defination Language

① Create Statement : This statement is used to create the table into database

Syntax:

Create TABLE TableName (Column1 DataType(size) constraint, column2 Data type (size) constraint,
:
column n Data type (size) constraint);

Examples

Create the following tables.

PRODUCTS
ProdID (PK)
ProdName (Not Null)
Qty (Chk>0)
Descriptsion.

ORDERS
ProdID (FK from products)
OrderID (PK)
Qty_sold (chk>0)
Price
Order_DT.

```
SQL> CREATE TABLE PRODUCTS (PRODID VARCHAR(20)
PRIMARY KEY, PRODNAME VARCHAR (25) NOT NULL,
  QTY NUMBER (5), DESCRIPTION CHR(50));
```

```
SQL> CREATE TABLE ORDERS (
        PRODID VARCHAR(20) REFERENCES PRODUCT
PRODID), ORDERID NUMBER(5) PRIMARY KEY,
QTY_SOLD NUMBER(5) CHECK (QTY_SOLD>0),
Price. NUMBER(8,2), ORDER_DT DATE);
```

② Customer.

↳ customer_id (PK)
↳ First_name
↳ last_name
↳ Email_address

Orders1.

↳ Order_id (PK)
↳ Customer_id (FK)
↳ Order_date.
↳ Order_states.

~~SQL~~ Health care

③ ~~Geo~~

```
SQL> create Table Patient
     (
      Patient_id varchar(10) Primary key,
      Patient_name varchar(15) not null,
      Address varchar(20),
      City varchar(10)
     );
SQL> Create Table Invoice.
     (
      Patient_id number(10) references patient
      invoice_no number(10) primary key, (patient_
     Amount number(10,5),
     Invoice_dt date);
```

④. Create the following table. 24/3/21

a). Table name = students.

regno (PK)
name (not null)
Semester
DOB
Phone

b) Table name : Books
bookno (PK)
bname
author

c) Table name : Library.
regno (FK from students)
Bookno (FK from books)
DOI (date of Issue)
DOR (date of return).

* Duplitating Table

Creating a table from another table

↳ It is used to duplicate all the records & characteristics of one table to an another table

Syntax:

CREATE TABLE NEW_TABLE_NAME AS SELECT * FROM EXISTING_TABLE_NAME ;

Example:

SQL> Create table Demo
AS
Select * From Dept ;

Table Created

SQL> DESC DEMO.

NAME	NULL?	Type
Deptno		
Dname.		
Loc		

SQL> Select * From Demo;

② Truncate Statement.

Truncate statement is used to remove all t
data permanently but the structure of the
table will remain same.

Syntax:

Truncate table table_name;

Ex:

SQL> Select * from Demo;

Name	Null?	Type
Deptno	Dname	Loc.
1		
2		
3		

SQL> DESC Demo.

Name	Null?	Type
Deptno		
Dname		
loc.		

SQL> Truncate table Demo;

Table truncated.

SQL> Select * from Demo;

no rows selected.

Drop Statement

↳ This statement is used to remove both the data as well as the structure of the table.

Syntax:

Drop Table Table_name ;

Ex: SQL> Create table test
as
select * from Dept;

Table Created.

SQL> Select * from test;

Deptno Dname Loc.

SQL> DESC test;

Name Null? Type.

SQL> Drop table test;

Table dropped.

SQL> Select * from test; } does not Exist
SQL> DESC test;

Flashback:

↳ Syntax:

Flashback table table_name to Before Drop;

↳ Ex:

SQL> Flashback table test to Before Drop;

Flashback complete;

Purge:

Syntax: Purge Table Table_name ;

Ex: SQL> Purge Table Test;

Table pureged;

SQL> Flashback table test to Before Drop;

Error: Table not found in Recycle Bin

Note:

↳ functionality of Recycle-Bin was Introduced in Oracle. Thus although the table has been drop, we can still restore it using the flashback command & also we can permanent drop the table from the database using the Purg Command.

④ Alter Statement

⑤ Rename:

It renames a table name.

Syntax:

Rename table_name to new_table_name;

Ex:

SQL> Rename Sample to Sample1

table created;

SQL> Select * from Sample1;

14 rows created;

SQL > Select * from Sample;

error: doesn't exist.

26/3/2021

⑤ Alter Statement

↳ This statement is used to alter or do changes to the structure of the table that means it can be adding the columns/removing the columns or renaming the columns.

Syntax:

1. To add a new column.

ALTER TABLE TABLE-NAME ADD COLUMN_NAME

DATATYPE (SIZE);

2. To rename the column.

ALTER table table_name rename column column_name to new_column_Name.

3. To drop the column.

↳ALTER TABLE table_name DROP COLUMN column_table

↳ALTER TABLE table_name DROP (column_name1, column_table2);

Examples

```
SQL>CREATE TABLE SAMPLE1
    AS
    SELECT * FROM EMP;
TABLE CREATED
SQL> SELECT * FROM SAMPLE1;
SQL> ALTER TABLE SAMPLE1 ADD LOCATION var char(15);
SQL> ALTER TABLE SAMPLE1 RENAME COLUMN
        MGR to MANAGER;
SQL>ALTER TABLE SAMPLE1 DROP COLUMN LOCATION;
SQL> ALTER TABLE SAMPLE DROP(MGR, JOB);
```

③Data Manupulation Language (DML)

↳DML statements are used for managing data in database. DML statements are not auto-committed. It means changes made by DML command are not permanent to database. It can be called as tack.

↳①Insert : This table statement is used to insert data into the table.

Insert Syntax :→ Insert into table_name values (v1, v2,..);

Insert into table_Name (column1, column2,..) (or) Values (v1, v2,...);

Example:

```
SQL> SELECT * FROM PRODUCTS;
SQL> DESC PRODUCTS..
SQL> INSERT INTO PRODUCTS VALUES (1001, 'LAPTOP',
                                  10,'DELL');

SQL> SELECT * FROM PRODUCTS;
SQL> INSERT INTO PRODUCTS(PRODID, PRODNAME)
     VALUES(1002, 'CAMERA');
SQL> CREAT TABLE DEMO12
       AS
       SELECT * FROM DEPT;
SQL> SELECT * FROM DEMO12;
SQL> INSERT INTO DEMO12 VALUES(&V1, '&V2', &V3)
     /
Enter value for V1: 50
Enter value for V2: TEST
Enter value for V3: BNG
old 1 : INSERT INTO DEMO1 VALUES('&V1', &V2, &V3)
new 1 : INSERT INTO DEMO12 VALUES('50', 'TEST', 'BNG
1 row created
```

② Delete Command

↳ It is used to delete data from a table.

↳ Delete command can also be used with a condition to delete a particular row.

Syntax :→

```
DELETE FROM TABLE_NAME [WHERE
    CONDITION];
```

Example:

```
SQL>CREAT TABLE TEST
    AS
    SELECT * FROM EMP;
SQL>DELETE FROM TEST WHERE ENAME="MILLER";
SQL> DELETE FROM TEST ;
```

(3) Update Statement

↳ It is used to update one or more rows of the table.

Syntax:

```
Update table_name set column_name=value where
  condition;
```

1. Write a query to update salary by increasing it to rupess Rs. 200 & also give the commission of Rs. 100 where Employee no. shld be 7369.

```
SQL> UPDATE Geede table Test
    as
    Select * from Emp;
SQL>UPDATE TEST SET SAL=SAL+200,
COMM=100 WHERE EMPNO=7369;
```

```
SQL>UPDATE TEST SET DEPTNO=NULL WHERE
    ENAME='SCOTT';
```

2. Write a query to update Empname = 'Scott where dept no is null.
3. Write a query to update all the Emp Salary by 10%

```
SQL>UPDATE TEST SET SAL=Sal*0.1;
```

Truncate

1. Truncate is a DDL Command.
2. We cannot use where clause with the truncate.
3. Truncate statement is used to remove all the records from the table.

Delete

1. Delete is a DML Command
2. We can use where clause in delete command.
3. Delete statement is used to delete a row and also all the rows from the table.

29/3/24

④ Transaction Control Language (Data Transaction Language)

↳ Any DML change on a table is not a permanent one.

↳ We need to save the DML changes in order to make it permanent.

↳ We can also undo the ~~same~~ same DML changes on a table.

↳ The DDL changes cannot be undone as they are implicity used.

① ROLLBACK

Syntax : ROLLBACK ;

It undoes the DML changes performed on the table

② COMMIT

It saves the DML changes permanently to the database.

Syntax : COMMIT ;

Eg:

```
SQL> Create TABLE XYZ
     as
     Select * from EMP;
```

SQL>DELETE FROM XYZ;

14 rows deleted.

SQL> ROLLBACK;

Rollback completed.

SQL>commit;

commit completed.

SQL> Select * from XYZ;

no rows selected.

SQL> ROLLBACK;

Rollback completed.

SQL > DELETE*FRM XYZ;

14 rows deleted;

③. Savepoint

The SAVEPOINT statement names and marks the current point in the processing of transaction.

↳ With the ROLLBACK To statement, savepoints undo parts of a transaction instead of the whole transaction.

Syntax: savepoint savepoint_name;
rollback to savepoint savepointname;

SQL> DELETE FROM SAMPLE where ENAM = 'SMITH';

13 rows dilited.

SQL>SAVEPOINT A;

Savepoint complete;

SQL>Delete from sample where Ename = 'Scott';

SQL>savepoint B;

SQL>Delete from sample where Ename = 'James';

SQL>savepoint C;

SQL> Delete from sample where Ename = 'Miller';

SQL>Savepoint D;

SQL> ROLLBACK TO SAVEPOINT B;

SQL> Select * from sample;

5. Data Control Language

1. Grant :→ This statement is used to provide the access permission to different user i.e., to access the database.

Syntax :→

```
GRANT SELECT ON TABLE_NAME TO USER_NAME;
```

2. Rovoke :→ This statement is used to take back the given permission.

Syntax :→

```
REVOKE SELECT ON TABLE_NAME FROM
                USER_NAME;
```

Ex:

```
SQL> Show user.
User is scott.
SQL> GRANT SELECT ON EMP TO HR;
Grant Successful :
SQL > CONN.
Enter user_name : HR
Enter password : *****
Connected.
SQL> show User
User is HR.

SQL > CONN
Enter user_name : SCOTT
Enter password : *****
Connected.
SQL> REVOKE SELECT ON EMP FROM HR;
Revoke Successful.
SQL> CONN;
```

Order-By clause

Order-By clause is used to sort the records either in ascending or descending order
↳ Order-By clause by default sorts the records in ascending order & also we can make use of optional keyword called as ASC.
↳ If we want to sort the records in descending order we make use of DESC statement.

Syntax:

```
Select * / column-name from Table-name
order by column-name [Asc / Desc];
                      optional.
```

```
SQL> Select * from Emp order by Sal Asc;
SQL> Select Ename, Sal, Job from Emp order By 2;
SQL> Select * from Emp order By 7 desc;
```

Functions

Functions are important features of SQL which is used to perform calculation on data to modify individual data items, to manipulate the output for group of records & also it is used to format date & numbers. & so on.

Types of Functions

1. Single Row function.
2. Multi Row function / Group By-function / Aggregate function.

① Single Row Function.

- Character
  - Case-Manipulation Function.
    - Upper
    - Lower
    - InitCap.
  - Character Manuplation Function.
    - Instr
    - Sub-str
    - Trim.
    - Concate
    - Length
    - Replace.
- Date
  - To_char
  - To_Date
- General
  - NVL
  - NVL2.

② Multi Row / Group-By-Function / Aggrate function.

- Max
- Min
- Avg.
- Sum
- Count

31/3/21.

1. Single Row Functions

1. Character Functions

[a] Case Manipulation Functions :->

1. Upper Function:

This function is used to convert all the characters in a given string into a uppercase.

Syntax: Upper(literals/column name)

Ex: SQL> SELECT UPPER('sql') FROM DUAL;

```
UPP
---
SQL
```

SQL > SELECT UPPER(Ename) from EMP;

```
Upper(Ename
----------
SMITH
  :
  :
MILLER
```

2. Lower Function:

This function is used to convert all the characters in a given string into a lowercase.

Syntax: Lower (literals/column name)

Ex: SQL> SELECT Lower('sql') from DUAL;

```
LOW
---
sql
```

SQL > SELECT LOWER(Ename) From EMP;

```
lower(Ename)
------------
smith
  :
  :
miller
```

3. Initcap: This function is used to convert first char into upper case & rest of the case into lowercase.

Syntax: Initcap (literals / column_name)

Ex: SQL > Select initcap('SQL') from DUAL;

INI

Sql.

SQL> SELECT ENAME, INITCAP(ENAME) FROM EMP

ENAME	INITCAP(ENAME)
SMITH	Smith
MILLER	Miller

SQL> SELECT UPPER('morning'), LOWER('MORNING'), INITCAP('MORNING') FROM DUAL;

Upper	LOWER	INITCAP
MORNING	morning	Morning.

[b] Characte Manipulation Function.

This functi

1. Replace function :→

This function is used to replace the substring from a given string

Syntax: Select Replace (String, substr to be replaced, New Substring)

Ex:

SQL> Select Replace ('Java', 'J', 'L') from Dual;

↳ Lava

SQL> Select Replace ('C Developer', 'C', 'Java') from Dual;

↳ Java Developer.

SQ2> Select Replace ('Java', 'x', 'x') from Dual;

↳ Java.

SQL> Select Replace ('Java', 'a', 'n') from Dual;

↳ Jana.

1. Write a Query to retrieve all the employees name & job while retrieving replace sales to spyder.
2. Write a Query to retrieve all the employees name while retrieving Replace char 'S' to '$'

① Select Replace (sales, 'sales', 'spyder') from JOB;

Select Replace Ename, Job, 'S', '$') from EMP;

Select Replace Ename, Replace ('S', 'S', '$') from EMP;
'Ename

2. sub-str function

This function is used to extract the substring from a given string.

↳ The extraction will happen from left to right.
↳ The length is Optional.

Syntax:

Substr(string, starting position, length)

Ex:→

Select Substr ('Spiders', 1, 3) from Dual; ↳SPI
Select Substr ('Spiders', 4, 4) from Dual; ↳DER.
Select Substr ('Spiders', 2, 10) from Dual; ↳PIDERS
Select Substr ('Spiders', 6) from Dual; ↳ERS
Select Substr ('Spiders', -5, 2) from Dual; ↳ID

1. Display all the employees name along with that display first char & last char from Employees name

2. Display all the employees who's name start with 'S' without using like Operator.

SQL> Select Ename, substr (Ename, 1, 1), substr (Ename, -1, 1) from EMP;

SQL> Select * from EMP WHERE SUBSTR (ENAME, 1, 1) = 'S';

Additional Working Questions

Select substr('Mahadevapura', 5, 5) from Dual;
↳ DEVAP.

Select substr('Mahadevapura', -8, 3) from Dual;
↳ DEV

Select substr('Mahadevapura', 10, 10) from Dual;
↳ URA

Select Replace('Nissan', 's', 'y') from Dual;
↳ Niyyan.

Select Replace('N Yash', 'N', 'Nis') from Dual;
↳ Nis Yash.

Select Replace('Euro', 'Z', 'y') from Dual;
↳ Euro.

01/11/21

Multi-Row/Group-By Function/Aggregate Function.

↳ This function operates on a set of records and returns one result.

① Write a query to display the total salary, highest salary, least salary, average salary & number of records.

```
SQL> SELECT SUM(SAL), MAX(SAL), MIN(SAL),
AVG(SAL), COUNT(*) FROM EMP;
```

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

② Display no. of Employees who is getting commissi

```
SQL> SELECT COUNT(*) FROM EMP WHERE
        COMM IS NOT NULL;
```

③ List the highest & lowest salary earned from by the salesman

SQL> SELECT MAX(SAL), MIN(SAL) WHERE FROM EMP WHERE JOB = SALESMAN.

④ List the no of clerks of dept-20

SQL> SELECT COUNT(*) FROM EMP WHERE JOB=CLERK AND DEPT=20.

⑤ Display the total salary in deptno=10

SQL>SELCT SUM(SAL) FROM EMP WHERE DEPT=10.

⑥ Display the oldest & latest eared date of EMP

SQL> SELECT MIN(HIREDATE), MAX(HIREDATE) FROM EMP;

⑦ Display the no. of employees those who are earning more than 2000.

SQL> SELECT COUNT(*) FROM EMP WHERE SAL > 2000;

⑧ Write a query to count unique deptno.

SQL> SELECT COUNT(distinct deptno) from emp;

count(distinct deptno)
3.

DISTINCT

↳ Distinct is a clause or a keyword which is used to remove duplicates from the result sets.

Ex

SQL> SELECT DISTINT DEPNO FROM EMP;

DEPTNO
10
20
30

SQL > SELECT DISTINT DEPNO, JOB FROM EMP;

DEPTNO	JOB
	Clerk
20	Salesman
30	Manger
20	Clerk
10	:
:	

Group-By Clause

↳ It is used in select statement to collect the data across the multiple records & group the result by one or more column

Syntax:

```
Select Column_name, Group_By function
from Table-name,
where <condition>
Group By Column_name
Having < condition will be based on Group By fun>
```

Points to remember :→

↳ The column which will be present in the select stmt shld be present in grp by clause also

↳ where clause is used to restrict the non-group data & it shld be present before the group by clause.

↳ Having clause is used to restrict the group data & it shld be present after the group-by clause

↳ where clause, having clause are optional & it is used only one certain condition.

↳ Order-By clause should be present always at the end of the query.

Ex. WAQ to display depart wise least salary?

```
SQL > SELECT DEPTNO, MIN(SAL) FROM EMP
  GROUP BY DEPTNO;
```

② WAQ to display Job wise total salary

```
SQL> SELECT JOB, SUM (SAL) FROM EMP
  GROUP BY JOB;
```

③ WAQ to display deptwise no of Employees.

SQL> SELECT DEPTNO, COUNT(*) FROM EMP
GROUP BY DEPTNO;

④ Display Jobwise highest salary only if the highest salary is more than 2500.

⑤ Display the average salary of all the Deptments

⑥ Display the maximum salary of each job.

⑦ Display jobwise highest salary only if the highest salary is more than 1500 excluding Deptno=30. Sort the Data based on highest salary in ascending order.

⑧ Display the deptno which are having more than four employees init.

④. Select Job, max(Sal) from Emp where Sal>8500
group by Job;

⑤. Select DeptNo, Avg(sal) from Emp Group By
Deptno;

⑥. Select Job, max(Sal) from Emp Group By
Job;

⑧. Select deptno, count(*) from Emp
Group By deptno
having count(deptno)>4;

⑦. Select Job, max(Sal)
from Emp
where deptno != 30
group by Job
having max(sal)>1500
Order by max(Sal ASC;

Questions

1. WAQ to display the Job whose minimum salary is less than 1000.
2. WAQ to display the deptno that contains more than 3 Employees.
3. WAQ to display deptno & there maximum salary for the ~~deptno~~ department, whose max salary is greater than or equal to 3000.
4. WAQ to display the Job & no of Employees who are working for it for the Job that contains more than 2 Employees.
5. WAQ to display the deptno & there avg salary for the Dept whose avg salary is greater than 200
6. WAQ to display the Job & total salary for the job whose total salary is < 4000.
7. WAQ to display the Deptno & there minimum salary for the dept whose minimum salary < 4000
8. WAQ to display the Deptno & no. of Employees working for it, for Deptno ~~whose avg salary~~ where it consist more than 5 Employees.
9. WAQ to display the maximum salary for each of the Job Excluding all the Employee whose name Ends with 'S'.

```
SQL>select JOB, MAX(SAL)
    from Emp
    where Ename NOT like '%S'
    group by Job;
```

1. Select Job, Min(Sal)
from Emp
where Sal < 1000
Group by Job;

Select Job, min(Sal)
from Emp
group by Job;
having min(Sal) < 1000;

2. Select Deptno, count(*)
from Emp
group by Deptno
having count(Deptno) > 3;

3. Select Deptno, max(sal)
from Emp
groups by Deptno
having max(sal) >= 3000;

(or) where clause

4. Select Tob, count(*)
from Emp
Group by Tob
having count (Job) > 2;

5. select Peptno, Avg (sal)
from Emp
group by deptno
Having avg (sal) > 2000;

6. Select Tob, Avg (Sal)
from Emp
group by Job
Having Avg (sal) < 4000;

Sum

7. Select Deptno, min(sal)
from Emp
group by Deptno
having min (sal) < 4000;

(8). Select Deptno, count(*)
from Emp
Group by deptno
having count (*) > 5;

4/6/21

Sub Query

↳ A Query within an another query is called as sub-queries

↳ It is also called as nested query.

↳ here, the inner query will be executed first & it will pass some input to ~~output~~ outer query, then outer query returns the final output.

↳ To join the inner & outer query we make use of operators

↳ whenever there is a unknown value, independ value or depent values are present we make use of sub-queries

↳ There are two types of sub-queries,

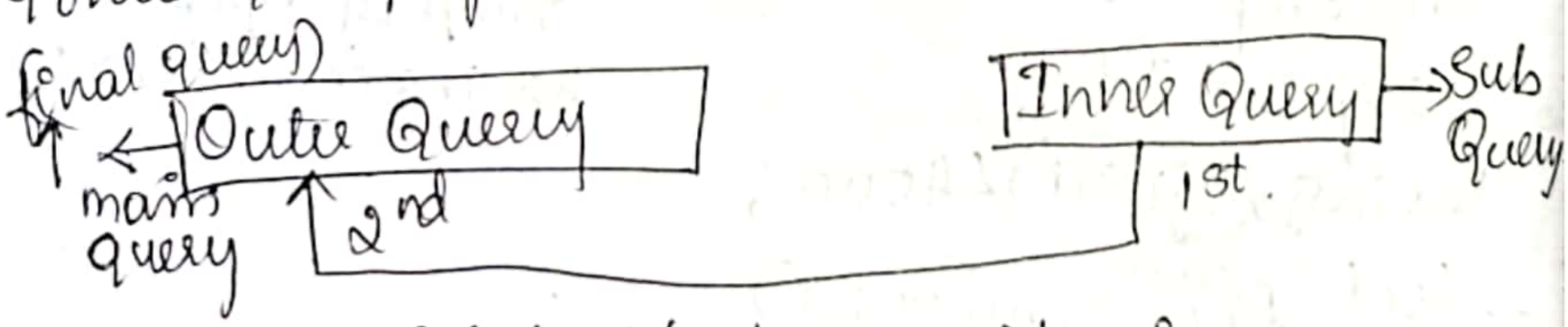
↳ Single-row sub-queries.

↳ Multi-row sub-queries

↳ Single-row subQueries :→ whenever the subque fetches single value as an i/p to outer quer in that case to join the inner & outer query generaly we make use of '=' operators.

↳ Multi-row subQueries :→ whenever the subQue fetches more than one value as an i/p to outer query so in that case to join the inne & outer query generally we make use of "in" operato

final query



Syntax :→ Select */column_Name.
from Table_name
where <conditi on> (Select */colum_na
from Table_name
where <condition>);

wAQ to display all the Emp's who job is same as james job.

Select * from Emp where Job = ('Select Job from Emp where Ename = 'James');

↳ 1st ; Select Job from Emp where Ename = 'James'

↳ 2nd ; Select * from Emp where Job.

2] WAQ to retrieve all the Emp's who salary is more miller salary.

SQL > Select SAL from Emp where ENAME = 'MILLER'

SQL > Select * from EMP WHERE sal > (Select SAL from Emp where ENAME = 'MILLER');

3] List all the Employees who is working for research deptment

SQL > select DeptNo from Dept where DNAME = 'Research';

SQL > Select * from Emp where Deptno = ('Select Deptno from Dept where Dname = 'Research');

4] List all the Employees who are working in the same dept of king.

SQL > Select Deptno from Emp where Ename = 'King';

SQL > Select * from Emp where Deptno = ('select Deptno from Emp where Ename = 'king');

5. List all the Employees who are located at NewYork

6. List all the Employees who is working for Sales & research department.

7. List the Deptartment name which are having salesman in it.

8. Display all the Employees whose sal is greater than avg sal of Deptno = 30

9. Display all the Employees whose location is having letter 'O' in it.

5] SQL> Select * from Emp where Deptno = (Select Deptno from Dept Where LOC = 'New York');

6] SQL> Select * from Emp where Deptno IN[Select Deptno from Dept where Dname IN['Sales', 'research'];

7] Select Dname from Dept where Deptno IN(Select Deptno from Emp where Job = 'salesman');

8) Select * from Emp where Sal > (Select Avg(Sal) from Emp where Deptno = 30 group by Deptno);

9] Select * from Emp where Deptno in (Select Deptno from Dept where Loc like '%O%');

7/4/21

1] Display all the Employees Job who Job is same as Scott & allen.

SQL> Select * from Emp where Job IN(Select Job from Emp where Ename IN('Scott', 'Allen

2] Display all the Employees who is Reporting to Scott.

SQL> Select * from Emp where MGR IN (Select MGR from Emp where Ename = 'Scott');

3] Select Employee details who belongs to dept no that contains more than 3 Employees in it.

SQL> Select * from Emp where Deptno In (Select Deptno from Emp where Ename > 3);

4] WAQ to display Employee details who is working for the Job where mon sal is less than 3000

SQL> Select * from Emp where Job IN (Select JOB from Emp group by Job having mon(sal) <3000);

5] WAQ to display Employee details who is working for Job whose min salary is > 1000

SQL> Select * from Emp where Job In (Select Job from Emp group by job having min(sal) > 1000);

6] WAQ to display Emp details who is getting salary more than total salary of Salesman.

SQL> Select * from Emp where sal > (Select sum (sal) from Emp where Job = 'Salesman');

7] WAQ to display Employee details who is getting maximum sal.

SQL> Select * from Emp where Sal IN (Select max(sal) from Emp);

8] WAQ to display the Employees details who getting min sal.

SQL > Select * from Emp where Sal IN (Select min(sal) from Emp;

9] WAQ all the dept name that are having atleast one Employee in it.

SQL> Select * from Dept where Deptno IN (Select Deptno from Emp groupby having count (Ename) >=1);

10] WAQ to display Emp details who is getting sal less than max salary of dept no 20.

SQL > Select * from Emp where sal < (Select max(sal) from Emp where Deptno = 20);

8/4/21

Joins

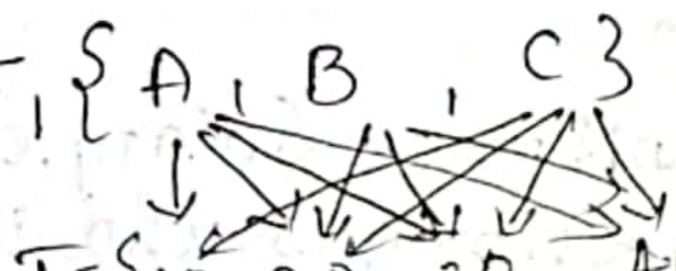
Joins are used to fetch the data from multiple tables. The different types of Joins are :->

1. cartesian / cross join.
2. Inner / Simple / Equi
3. Outer join.
   - left outer join
   - Right outer join
   - full outer join
4. self join.

1. Cartesian / Cross Join

def^:-> Where in each & Every records of one table directly matches write each & Every records of another table & it will display the o/p as valid as well as invalid records.

Ex: $T_1\{A, B, C\}$



$T = \{10, 20, 30, 40\}$.

To get:

o/p = {(A,10) (A,20) (A,30) (A,40)
(B,10) (B,20) (B,30) (A,40)
(C,10) (B,20) (C,30) (C,40)}

(A,10), (B,20), (C,30) -> valid record

=> Invalid records.

Q) Display Emp name along with the dept name

SQL> Select Ename, Dname from Emp, Dept;

Ename	Dname
Smith	Research
Allen	Research
Smith	Sales
Allen	Sales

Smith, Research, Allen, Research -> valid records

Smith, Sales, Allen, Sales -> Invalid records.

② Inner Joins

↳ Where in each & every records of one table is compared with each & every records of another table & it will display the o/p as matched records from both tables.

Syntax:→

Select Table 1 column, Table 2 column
from table 1, table 2
where table common column = Table 2. common column;

Q) T_1 | T_2

3 — 1
5 — 4
9 — 5
1 — 7

o/p: 1, 5

Q1) Display Emp name along with dept name.

SQL > Select Ename, Dname from Emp, Dept where Emp.Deptno = Dept.Deptno;

Q2) List the EmpName, Salary, dept name of all the Employees who is Earning more than 3000.

SQL > Select Ename, Job, Loc from Emp E, Dept D where E.Deptno = D.Deptno and Job = 'Manager';

Q3) Display Ename, Job, Dname, Loc of all the salesmen who are not located at DALLAS

SQL > Selec Ename, Dname, Job, Loc from Emp E, Dept D where E.Deptno = D.Deptno and Job = 'Salesman' and Loc = 'DALLAS';

Q4) Display Ename, Job, Deptname, loc of all Manager & check who works in accounting & sales dept

SQL > Select Ename, Job, Dname, Loc from Emp E, Dept D where E.Deptno = D.Deptno and Job IN ('Manager', 'Clerk') And Dname IN ('Accounting', 'Sales');

6] WAQ to display Ename, Dname of all the Employees whose name start with 'S'.
SQL> Select Ename, Dname from Emp E, Dept D where E.Deptno = D.Deptno AND Ename like 'S%';

7] WAQ to display Ename, Sal, Dname of all the Employees who all Earning more than miller;
SQL> Select Ename, Sal, Dname from Emp E, Dept D where E.Deptno = D.Deptno AND sal > (Select sal from Emp where sal = 'Miller');

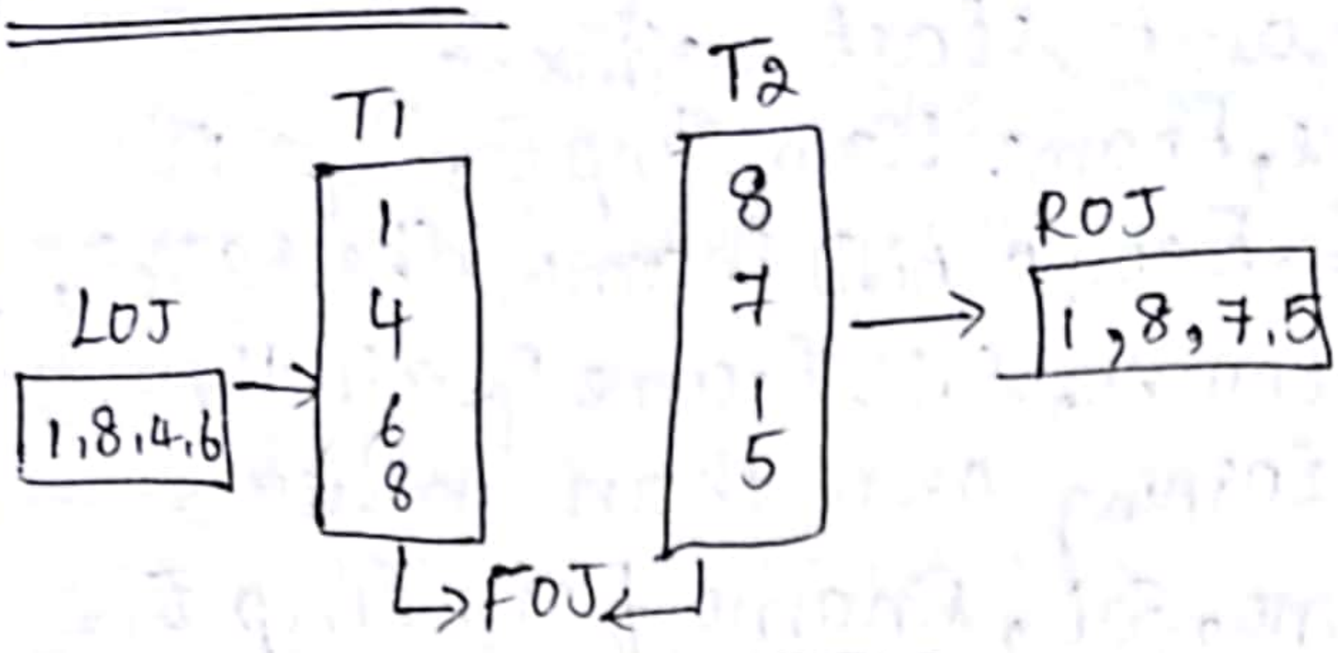
8] WAQ to display Ename, Hiredate, Dname of all the Employees, those who have hired before king.
SQL> Select Ename, Hiredate, Dname fore Emp E Dept D where E.Deptno = D.deptno AND Hiredate < (Select Hiredate from Emp where Ename = 'king');

***
8] WAQ to display Ename, Deptno, sal, Loc of all emp working in Deptno 20 & Earning more than 2300.

***
9] WAQ to display Ename, mgr, dname of all Emp who is having reporting Mgr.
SQL> Select Ename, Mgr, Dname from Emp E, Dept D where E.Deptno = D.Deptno AND Mgr is not null;

Outer Joins

9/6/21



Left Outer Join (LOJ) :→

Where in each & every records of one table is compared with each & every record of another table & it will display the o/p as matched records from both the tables & unmatched records only from the left table.

Syntax (Oracle std)

```
Select T1 column, T2 column
from T1, T2
where T1.CC = T2.CC(+);
```

Query :→

```
SQL> Select Ename, Dname
     from Emp, Dept
     where Emp.Deptno = Dept.Deptno(+);
```

Right Outer Join (ROJ) :→

Where in each & every records of one table is compared with each & every records of another table & it will display the output as matched records from both the tables & unmatched records only from the right table.

Syntax (Oracle std)

```
Select T1 column, T2 column
from T1, T2
where T1.CC(+) = T2.CC;
```

Query:

```
SQL> Select Ename, Dname
     from Emp, Dept
     where Emp.Deptno(+)=Dept.Deptno;
```

Full Outer Join :→

where in each & every records of one table compared with each & every records of another table & it will display the output as matched as well as unmatched records from both the tables.

Syntax (ANS, Std)

```
Select T1 column, T2 column
from T1 full outer join T2
on T1.cc = T2.cc;
```

Query :→

```
Select Ename, Dname
from Emp full outer join Dept
on Emp.Deptno = Dept.Deptno;
```

Self-Join :→

Joining a table to itself is called as self Join. Where in each & every records of one table is compared with each & every records of same table & it will display the o/p as matched records from the table

Example :→

Duplicated

Emp E → alias

Empno	Ename	mgs
01	A	04
02	B	01
03	C	02
04	D	03

Emp M → alias

Empno	Ename	mgs
01	A	04
02	B	01
03	C	02
04	D	03

Query:
Select E.Ename Employee_Name, M.Ename Manager_Name
from Emp E, Emp M
where E.mgs = M.Empno;

Rownum :→

It is a virtual column which is associated with the database table at the time of Execution, It is temp. column which is query based & which generates the number from 1.

Queries :→

SQL> Select Ename, Rownum from Emp;
SQL> Select Ename, Job, Rownum from Emp where Job = 'Manager';
SQL> Select * from Emp where Rownum = 1;
SQL> Select * from Emp where Rownum <= 5;
SQL> Select * from Emp where Rownum = 5;
no rows selected because rownum always starts with 1.

14/7/2021 Normalization.

"It is a database design technique which helps to organize a table".
↳ It divides large table into an smaller table & link them using primary key & foreign key
↳ It was introduced by Edger Codd who Proposed the thorey of Normalization with the introduction of 1st normal form & he continued to extend the theory with 2nd & 3rd NF. Later he joined with Raymond F. Boyce, together they introduced Boyce codd Normal Form (BCNF).

12/12/24 Media Library Database :→

Customer Info table (table without NF rules)

Name	Address	Cinema Rented	Salutation
Dinga	Hebbal	K.G.F F&F	Mr.
Dingi	M.G.Road	Bahubalii Conjury	Ms.
Dingi	Brigade	Master	Ms.

1st NF :→ ↳ Each cell of the table shld contain single value.

↳ Each records need to be unique.

Customer Info table :→ (table with NF rules)

Name	Address	Cinema Rented	Salutation
Dinga	Hebbal	K.G.F	Mr.
Dinga	Hebbal	F&F	Mr.
Dingi	M.G.Road	Bahubalii	Ms.
Dingi	M.G.Road	Conjury	Ms.
Dingi	Brigade	Master	Ms.

2nd NF :→

↳ The table as to be according to 1st NF rules

↳ Primary key & foreign key as to be provided for the tables.

Customer Info

CID	Name	Address	Salutation
01	Dinga	Hebbal	Mr.
02	Dingi	M.G.Road	Ms
03	Dingi	Brigade	Ms.

PK

Cinima Rented Info

CID	Cinema Rented
01	K.G.F
01	F&F
02	Bahubalii
02	Conjury
03	Master

FK

3rd NF :→

↳ The table as to be according to the 2nd NF rule
↳ If there is any transitive functional dependency then that as to be removed.

Customer Info.

CID	Nama	Address	SID
01	Dinga	Hebbal	200
02	Dingi	M.G.Road	100
03	Dingi	Brigade	100

PK

FK

Movie Rented Info.

CID	Movie Rented
01	M.G.F
01	K&F
02	Bahuballi
02	Conjury
03	Master

FK

Salutation Info

SID	Salut
100	Ms
200	Mr
300	Mrs
400	Dr.

PK

***

WAQ to display 2nd maximum salary (SMPT).

Emp E1

Emp no.	Ename	Sal
01	A	25,000
02	B	30,000
03	C	40,000
04	D	20,000
05	E	15,000

Emp E2.

Emp no.	Ename	Sal.
01	A	25,000
02	B	30,000
03	C	40,000
04	D	20,000
05	E	15,000

```
SQL> SELECT Distinct E1.SAL FROM EMP E1
WHERE 2=(SELECT COUNT (Distinct E2.SAL)
FROM EMP E2 WHERE E2.SAL >= E1.SAL);
```

e2.sal >= e1.sal.

25,000 >= 25,000 ✓
30,000 >= 25,000 ✓
40,000 >= 25,000 ✓
20,000 >= 25,000 X
15,000 >= 25,000 X

count = 3

e2.sal >= e1.sal

25,000 >= 30,000 X
30,000 >= 30,000 ✓
40,000 >= 30,000 ✓
20,000 >= 30,000 X
15,000 >= 30,000 X

count = 2.

WAQ to display 3rd minimum salary

SQL> Select Distinct E1.Sal from Emp E1
where 3 = (Select count (Distinct E2.sal) from
Emp E2 where E2.Sal <= E1.Sal);

WAQ to display top 3 maximum salary

SQL> Select Distinct E.1.sal from Emp E1
where 3 >= (Select Count (Distinct E2.sal) from
Emp E2 where E2.sal >= E1.sal);

WAQ to display top 4 minimum salary

SQL> Select Distinct E1.sal from Emp E1
where 4 >= (Select count (Distinct E2.sal) from
Emp E2 where E2.sal <= E1.sal);

WAQ to display fourth E1 2nd maximum salary

SQL> Select Distinct E1.sal from Emp E1 where
4 = (Select count (Distinct E2.sal) from Emp E2
where E2.sal >= E1.sal) OR 2 = (Select count
(Distinct E2.sal) from Emp E2 where E2.sal >= E1.sal);

15/4/21

Trim Function :->

↳ This function is used to remove the specified character from a given string.

↳ The different types of trim function are,

1. LTrim
2. RTrim
3. Trim → Leading
   → Trailing
   → Both

① LTrim function :->

↳ This function is used to remove the specified characters from the left hand side of a given string.

Syntax :-> LTrim (String, [Trim_string]);

Example :-> ~~Select~~

SQL> Select LTrim('SQL', 'S') from Dual;

→ QL

SQL> Select LTrim('QSpiders', 'SQ') from Dual;

→ piders

SQL> Select LTrim('QSpiders', 'SP') from Dual;

→ QSpiders.

SQL> Select LTrim('QSpiders', 'RS') from Dual;

→ QSpiders

SQL> Select LTrim('MMMMMMS', 'M') from Dual;

→ S

SQL> Select LTrim('MMMAMMS', 'M') from Dual;

→ AMMS

SQL> Select LTrim('MMMMMM', 'M') from Dual;

→ null value

RTrim function:→
↳ This function is used to remove the specified characters from the right hand side of a given string
Syntax:→ RTrim(String, [Trim-string]);
Ex:→
SQL> Select RTrim('@Spiders', 'RS') from Dual; ⇒ @Spide
SQL> Select RTrim('@spiders', 'RDS') from Dual; ⇒ @Spide
SQL> Select RTrim('@spiders', '@') from Dual; ⇒ @spiders
SQL> Select RTrim('ANNNINNN', 'N') from Dual; ⇒ A.

Trim function:→
Syntax:→
Traim(Leading / Trailing / Both Trim_String from String)
Ex:→
SQL> Select Trim(Leading 'E' from EWELCOME') from Dual; ⇒ WELCOME.
SQL> Select Trim(Trailing 'E' from EWELCOME')
from Dual; ⇒ EWELCOM
SQL> Select Trim(Both 'E' from EWELOME')
from Dual; ⇒ WELOM

Concate function:→
↳ It is used to merge the literalls / columns & the concate function accepts only the two arguments in the parameter
↳ If we want to pass more than 2 arguments then we need to write a query in nested fun. that means a fun. within a another fun.
Syntax:→ Concat(string1, string2)
Concat(Concate(string1, string2), string3)
Ex:→ Select Concat('SQL', 'CLASS') from dual; ⇒ SQLclass
Select Concate('Ename', 'Job') from Emp;
Select Concat(Concat('Ename', 'Deptno'), 'Sal')
from Emp;

SQL> Select Concat(Concat(Ename, Your Salary), Sal)
from Emp;

Length function :=>

This function is used to find the length of a given string

Syntax :=> Length(literal / column);

Ex :=> Select length ('SQL') from Dual ; => 3
Select length ('SQL Class') from Dual ; => 9
Select length (Ename) from Emp ;
Select Ename, length (Ename) from Emp ;

① Display all the Employees whose name is having exactly five characters without using like operator.

SQL> Select Ename, length (Ename) where
Ename = 5 from emp ;
SQL> Select * from emp where length (Ename) = 5 ;

Instr function :=>

This function is used to find a position of a substr from a given string based on number of the occurance

Syntax :=> Instr (String, substr, starting Position, No of occurance)

Ex :=>
Select Instr('OSSPISSSDERSS', 'S', 1, 3) from dual => 6
Select Instr('OSSPISSSDERSS', 'S', 3, 7) from dual => 0
Select Instr('OSSPISSSDERSS', 'S', 6, 2) from dual => 7
Select Instr('OSSPISSSDERSS', 'S', -5, 2) from dual) =>
Select Instr('OSSPISSSDERSS', 'S', 5, -2) from dual)
Error occurs

16/4/2021.

Date functions :→

SQL> Select Sysdate from dual;

↳ Sysdate

16-APR-21

SQL> Select Current_date from dual;

↳ Current_date

16-APR-21

SQL> Select Systimestamp from dual;

↳ SystimeStamp

16-Apr-21 11.05.10500 AM +05:30

① To_char function :→

This fun is used to convert the given date into any other character format.

Example :→

SQL> Select To_char('sysdate', 'YYYY,MM,DD') from dual";

to_char

↳(YYYY,MM,DD)

2021-04-16

SQL> Select To_char('Hiredate', 'MM,YYYY,DD' from Emp;

↳ Hiredate ① Display only the year from the hiredate.

↳ SQL> Select To_char('Hiredate', 'YYYY') from Emp

② Display only the month from the hiredate

SQL> Select To_char('Hiredate', 'MM') from Emp;

③ Display all the Employees who are joined in the year of 81

1. SQL> Select * from Employee where To_char('Hiredate', 'YY')='81');

④ Display all the Emp those who are joined in the month of feb.

SQL> Select * from Employee where To_char('Hiredate', MM='Feb');

⑤ Display all the Emp those who have joined in the foureth month of the year.

SQL> Select * from Employee where To_char('Hiredate', MM=04);

⑥ Display all the Emp details those who have joined in the same year of Allen.

SQL> Select * ~~from~~
from Emp
where To-char(~~select~~ 'Hiredate', 'YY') = (Select * To_char(Hiredate, 'YY') from Emp where Ename = 'Allen');

② To_date function :→

This fun is used to convert date from chara -cter format to original date format.

Ex:→

SQL> Select To_date('2021/16/04', 'YYYY/MM/DD') from dual;

→ To_date('2021/16/04')

16-Apr-21

SQL> Select To_date('APR-2021-16', 'MM YYYY DD') from dual;

To_date

16-Apr-21.

General :→
This function is used to convert null values to actual values.

① NVL : → (Null Value Locator)

Syntax :→ NVL (Arg1, Arg2)

↳ If Arg1 is null, then it returns Arg 2
↳ If Arg1 is not null, then it returns itself.

Ex:
SQL> Select NVL('A','B') from Dual ↳ A (not null)
SQL> Select NVL(' ','B') from Dual ↳ B (null)

② NVL2 :→

Syntax :→ NVL2 (Arg1, Arg2, Arg3)
↳ If Arg1 is null, it returns arg3
↳ If Arg1 is not null, it returns arg2

Ex:
SQL> Select NVL2('A','B','C') from Dual ↳ B (not null)
SQL> Select NVL2(' ','B','C') from Dual ↳ C (null)

① WAQ to display all the Employees those who are getting the commission, increase there commission by Rs. 500. The Employees whose who are not getting Commission give Rs. 200

~~SQL> Select NVL(* from Emp where NVL(500,200)~~ comm

SQL> Select NVL(comm+500, 200) from Emp;

SQL> Select NVL2(comm, comm+500, 200) from Emp;

Co-related SubQuerys

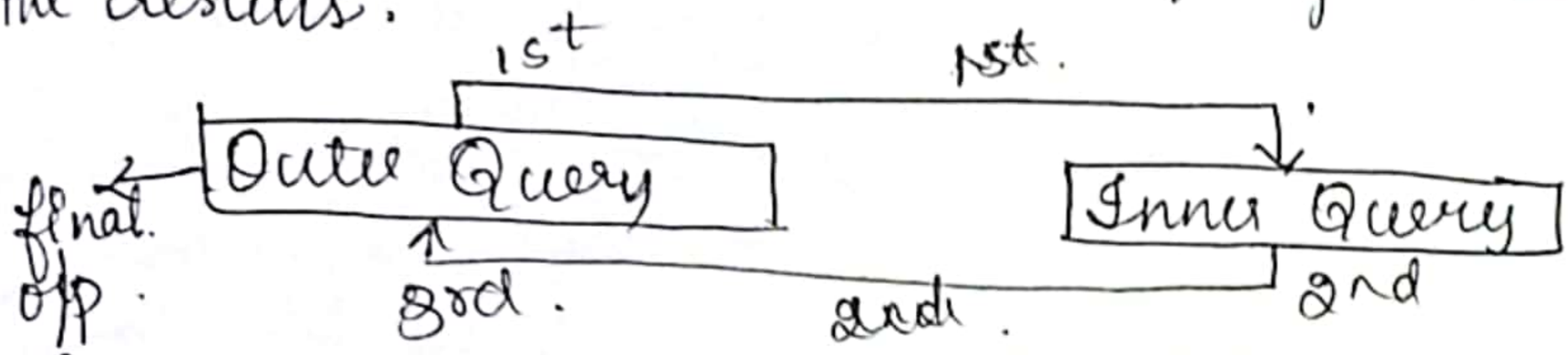
↳ There are special type of sub-queries.
↳ Here both inner & outer querys are inter-dependent
↳ for each & every records of outer query, the entire inner query will be executed.
↳ They work on the principle of both sub-query & Joins.
↳ This sub-querys uses two operators either exists/Not Exists

Exists :→

It returns true if a sub-query fetches atleast 1 value if it returns true, then outer query will display the results

Not-Exists :→

It returns true if a sub-query fetches no-values atall, if it returns true then outer query will display the results.



Syntax :→ Select T_1 column
from T_1
where Exists/Not Exists (Select T_1 colum
from T_2
where
$T_1.cc = T_2.cc$);

① Display dept name which is having atleast one Emp in it

SQL> Select Dname
from Dept
where Exists(Select ~~Dname~~ Empno
from Emp
where Dept.Deptno = Emp.Deptno);

② Display dept name whist is not having Emp in it.
Select Dname from Dept where Not Exists (Select Empno from Emp where Dept.Deptno = Emp.Deptno);

RESUME POINTS:

*GOOD UNDERSTANDING OF RDBMS CONCEPTS LIKE DATATYPES,CONSTRAINTS, NORMALIZATION(1NF,2NF,3NF),TABLES.

*EXCELLENT KNOWLEDGE OF WRITING SQL QUERIES.

*GOOD UNDERSTANDING OF SQL CONCEPTS LIKE OPERATORS,GROUP BY CLAUSE,SUB-QUERIES,FUNCTIONS.

*SOLID UNDERSTANDING OF SQL JOINS(INNER JOIN,OUTER JOIN,SELF JOIN).

*GOOD KNOWLEDGE OF SQL STATEMENTS (DATA DEFINITION LANGUAGE,DATA MANIPULATION LANGUAGE,TRANSACTION CONTROL LANGAUAGE).