Assignment : 2

Q :(1) Explain relational algebraic operators.

→ - relational algebraic is the Procedural query language.

- Many Operations are Performed to get the output.

ex : (Student table)

Rno	Name	Gender	City	Per
101	Raj	M	Surat	88
102	SAVAN	M	Anand	60
103	Rahul	M	Bharuch	76
104	Parth	M	Surat	80
106	Rudra	M	Surat	85

(1) Select (σ)

- select Operation selects the tuples (rows) which satisfy the given conditions.

- syntax :

  $\sigma_p(r)$

- relational operations :

  $-, =, \neq, >, <, \leq, \geq$

ex: (1) σ Gender = "Male" (tblstudent)

Rno	Name	Gender	city	per
101	Raj	M	Surat	88
102	SAMAN	M	Anand	60
103	Rahu	M	Bharuch	70
104	Parth	M	Surat	80
106	Rudra	M	Surat	85

ex: (2) σ Gender = "Male" and city = "Surat" (tblstudent)

101	Raj	M	Surat
104	Parth	M	Surat
106	Rudra	M	Surat

ex: (3)

σ Per > 80 (tblstudent)

101	Raj	M	Surat	88
106	Rudra	M	Surat	85

(2) Project (π)

- Projection allows to display only specific attributes (Fields) data in place of all.

- Syntax :

  π attributes (relation)

ex:① π Rno, name (tblstudent)

Rno	Name
101	Raj
102	SAVAN
103	Rahul
104	Parth
106	Rudra

ex:② π Rno, name (σ gender = "Male" and city = "surat" (tblstudent))

Rno	Name
101	Raj
106	Rudra
103	Rahul

③ Union (U)

- If A and B are the two tables and if we apply union then it displays all the records of A and B.
- It removes duplicate records.
- Syntax :

A U B

ex:① π Rno, name (σ gender = "Male", (tblstudent)) U
π Rno, name (σ city = "surat" (tblstudent))

Rno	name
101	Raj
102	SAVAN
104	Parth
106	Rudra

(4) Intersection (∩)

- If A and B two tables and if we only intersection then it displays only those tuples which are available in both tables.
- It removes duplicate records.
- Syntax : A ∩ B

ex: (1)

π Rno, name (σ Gender = "male" (tblstudent)) ∩
π Rno, name (σ ~~Gender~~ city = "surat" (tblstudent))

Rno	Name
101	Raj
104	Parth
106	Rudra.

(5) Rename (ρ)

- Syntax :

ρname (expression)

ex: (1) ρstud_info (sno, sname)
(π Rno, Rname (tblstudent))

sno	sname
101	Raj
102	SAVAN
103	Rahul
104	Parth
106	Rudra

Q:② Discuss the data types available in SQL.

NO	DATA TYPE	Description	
1.	NUMBER (P, S)	It is used to store zero, negative fixed and floating point number. (p) determines the maximum length of the data types (S) determines the number of places to the right of the decimal.	
		number (6)	123456
		number (6,2)	1234.56
		number(6,-2)	123500
2.	CHAR(n)	It is used to store fixed length text up to 2000 bytes (characters).	
3.	VARCHAR (n)	It is used to store variable lenght text up to 2000 bytes (characters)	
4.	VARCHAR2 (n)	It is used to store variable length up to 4000 bytes (characters) - Internal datatypes	

5.	LOB data types	It is used to store and manipulate large block of unstructured data like text, image, audio or video.
	CLOB	(Character Large object) - It uses 1 byte / character. - It is stored in from of Ascii character set.
	NCLOB	(National character Large object) - It is Multiple bytes / character (ex: 2 or 3 bytes per charcter) - It is stored in from of unicode character set.
6.	LONG	- Normally it is used for internal purpose. - Within DATA Dictionary, it is used. - Prefer to the CLOB in place of LONG
7.	DATE	It is used to store data in format 'dd-mon-yyyy' or 'dd-mon-yy'
8.	RAW(n)	It is used to store hexadecimal values upto 32767 characters.
9.	LONG RAW RAW	It is used to store graphics, audio, text or array of binary data.

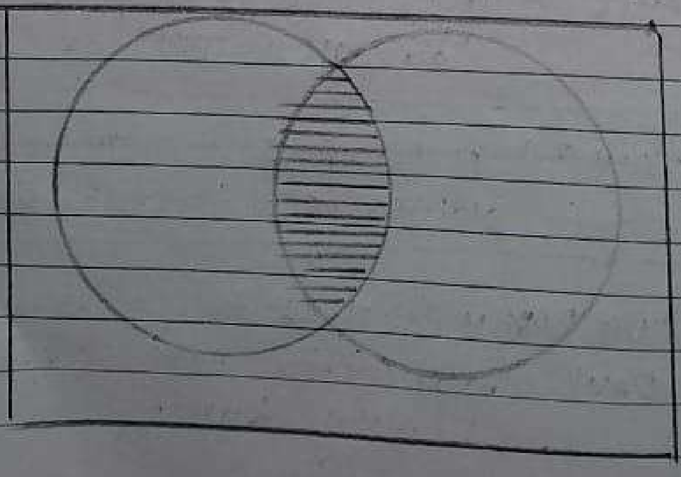
Q2③ What is the use OF Joins ? Discuss types OF joins.

→ Joins are used in SQL to combine data From two or more tables based on a related column. They help retrieve meaningfull information by merging tables in a relational database. Joins are essential for handling normlized databases, where data is stored across multiple tables to reduce redundancy.

→ Types of Joins :

① inner join :

- when the tables are combined using Inner Join then it returns all those records which are matched with both tables. It is also known as Equi Joins.



→ ANSI Style (only two tables can join)

Syntax :

```
select <column name1> <column n 2> <column n N>
From <Table name 1>
inner join <Table name 2>
ON <Table N 1>.<column Name 1> =
        <Table N 2>.<column Name 2>
Where
ORDER By <column n1>, <column n 2>,
                        <column n N>
```

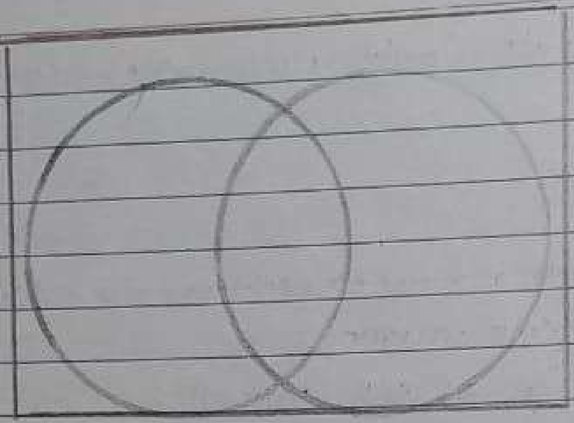
→ Theta style : (can join multiple tables

Syntax :

```
select <column n1> <column n 2> <column n N>
From <Table name 1> <Tablename 2>
where <Table name 1>.<column name 1> =
        <Table name 2>.<column name 2>
AND <condition>
ORDER By <column name 1>, <column name 2>,
                    <column name N>
```

② Outer join (Left , Right , Full)

- when the tables are combined using Left join then all the record of Left tables are displayed.
- If the related record's information is not available within another table then it returns Null to those fields.

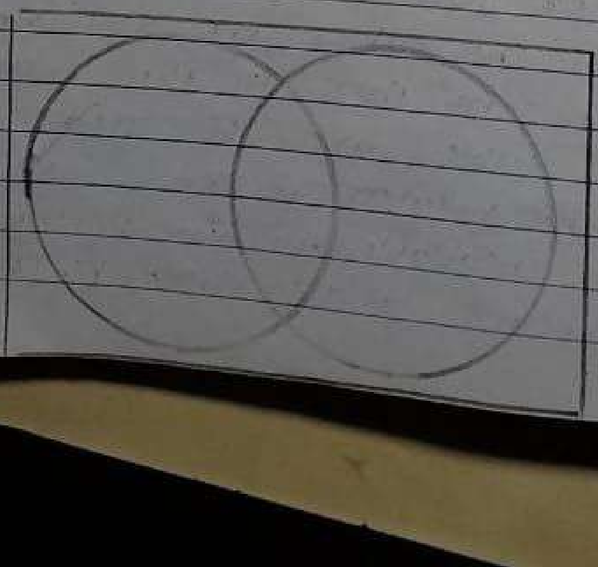


Syntax :

```
select <column Name 1> <column Name 2 >
                    <column Name N >
FROM <Table Name 1 >
LEFT join <Table Name 2 >
ON <Table Name 1 >. <Table column Name 1> =
      <Table Name 2 >. <column Name 2 >
```

→ Right :

- when the tables are combined using Right join then all the records of Right tables are displayed.
- If the related record's information is not available within another tables then it returns Null to those Fields.

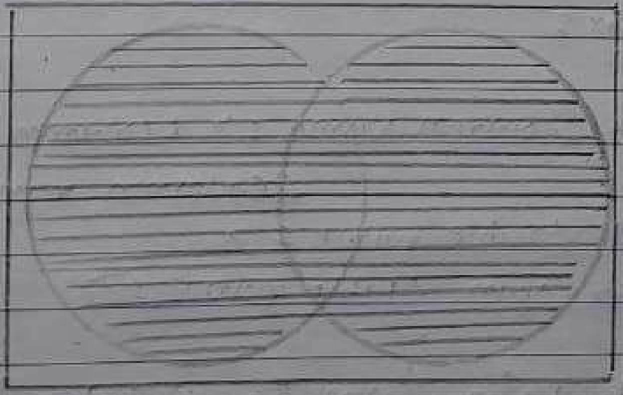


Syntax :

```
Select <column name1> <column name 2>,
                        <column name N>
FROM <Table name 1>
RIGHT JOIN <Table name 2>
ON <Table name 1>.<column name 1> =
    <Table name 2>.<column name 2>
```

→ Full Join

- When tables are combined using full join then it displays all the records of both the tables and returns Null to the fields where the respective information is missing.
- It gives combine result of LEFT and RIGHT joins.

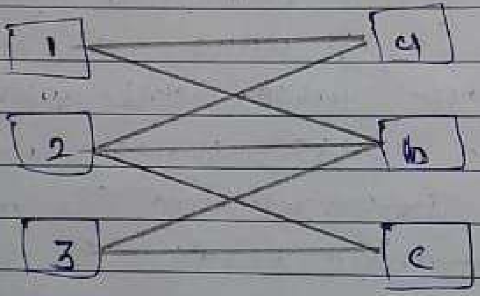


Syntax :

```
Select <column N1> <column N2>, <column N N>
FROM <Tablename 1>
Full JOIN <Table name 2>
ON <~~col~~ Table N>.<column N> = <Table N 2>
                          <column N 2>
```

③ Cross join

- When thable are combined with cross join then it displays all records.
- Each record of first table is joined with all records of another table
- If first table contains N records and second table contains M records then Cross join table displays N*M records.



Syntax:

```
Select <column Name 1> <column name 2>
                    <column name N>
FROM <Table name 1>
 Cross Join <Table name 2>
```

④ Self Join (Not in syllabus)

- Self join combines the ~~th~~ table with itself.
- same table is renamed as twice and then these two tables are combined with each other.

ex:

select A.RNO AS ARNO, B.Name AS BName,
~~A.Name~~ A.MARKS AS AMARKS FROM
Stud-info A, STUd-info Bwhere
A.MARKS < B. MARKS;

ARNO	BNAME	AMARKS
1	Mohit	35
1	Ankit	35
2	mayank	25
2	Mohit	25
2	Ankit	25
3	Ankit	45
4	mayank	10
4	Maya	10
4	Mohit	10
4	Ankit	10