SQL command groups.

① DDL - (Data definition Language)
- creation of Objects
- create - Truncate - DROP.
- Alter - Rename

② DML - (Data manipulation language)
- manipulation of data
- insert - update - Delete.

③ DCL - (Data control Language)
(give acces or not)
- Grant
- Revok

④ TCL - (Transaction control Language)
- commit - save the work done
- Roll back - undo work
- save point - ~~pint~~ pin the work.

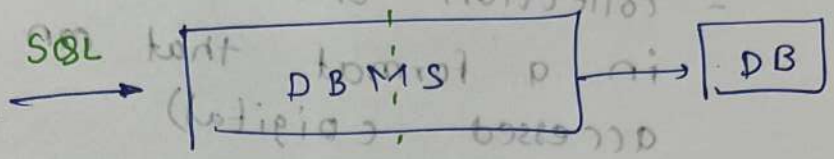
Data base -

- collection of data is Noting but database in a format that can be easily accessed (digital)

- आपल्या ~~कडे~~ lot's of big company where Google, facebook, instagram
- They have lot's of Data
- if we create account on instagram we have to fill email id, password name
- This all information is store in instagram's data base
- This nothing but collection data that we can accesse easily
- like add, search delet.
- data is store digitally in our computers.
- Database बरचन ~~राहु~~ अजुन गोष्ट समोर येते DBMS - Data base managment systems.

- A software application wed to manage our DB is called DBMS (Data base managment System)

- suppose जर मला (user ला) एक database accesse करायचा असेल तर



- we can not access database directly so we use DBMS to do it for you It is software application.

- ~~And we use~~ आणि आपण ~~is~~ ने SQL use ~~करतो~~ तो DBMS साठी.

Types of Database.

① Relational (RDBMS)
- Data stored in tables. like in school days we use to store our name presenty in Register.
- We use SQL to work with relational DBMS.

ex - My SQL
- SQL Server (Microsoft)
- Oracle
- postgre SQL

② Non- Relational (NO SQL)
- Data not stored in tables.
- Mongo DB.

What is SQL ?.

Structured Query Language.

- SQL is a programming language used to interact with relational database.
- It is used to perform CRUD Operations.
  - Create
  - Read
  - Update
  - Delete.

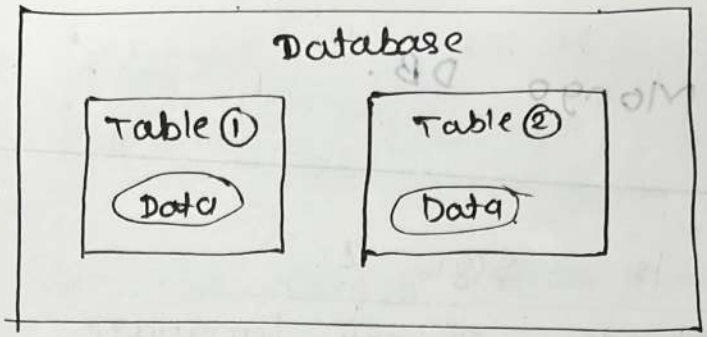
eg ↓
- Create account on insta
- Read names of id's password etc
- change password OR username
- Delete the instagram id from database.

SEQUEL
- structured
- English
- Query
- Language

SQL
- Structured
- Query
- Language

IBM

Database structure



- There is no necessary that every database contains only one table
- Database can contain more than 1 table always.

What is table -

- combination of rows & column.

- columns - structure / schema (design)
- Rows - Individual date.

Creating Our First Database :-

exploring the SQL Dashboard

- Administration
  - work of admin panel
  - start server, end server
  - give access revoke.
- schemas
  - how many database.
  - how many tables.

```
Create Database db_name;
```

- SQL query as simple as english lang.

```
Drop Database db_name;
```

- create database college;
- use college;

Creating Our First Table -

- use college
- create table table_name(
  column_name1 datatype (constraints.));

for eg

```
- CREATE TABLE student (
  std_id int primary key,
    name varchar(50),
    age int not null );
```

- datatype } we see in detailed
- constraint } in the next lectures.

↓

limit of data type.

Insert into table

```
- INSERT INTO student VALUES (
    1, "AMAN", 26);
```

---

show everything

```
- select * from student;
```

SQL Data types

They define the type of values that can be stored in a column.

Data Type	Description
char CHAR(50)	string (0 - 255), can store characters of fixed length.
varchar VARCHAR(50)	string (0-255), can store characters up to given length. ∴ in char if we allocate is as char(50) it will by default give 50 size of string to the memory - if ~~me~~ ~~mi~~ ~~data~~ ~~kam~~ Abhi ~~jagah~~ ~~nahi~~ still it will take 50 size ~~of~~ string memory ∴ but varchar ~~jagah~~ ~~ki~~ 4 size string memory allowcate ~~karta~~.
BLOB BLOB(1000)	string (0 - 65535), can store binary large object. - long ~~string~~ upto 65535.
INT = 1 INT	integer (- 2,147,483,648 .To +)
TINYINT = 1	integer (-128 to 127)
BIGINT = 1	Integer (-9,223,372,036 854,7705,808 To .. +)

BIT BIT(2)	can store x-bit values. x can range from 1 to 64 BIT(1) → 0, 1 BIT(2) → (00, 01, 10, 11)
FLOAT	Decimal number - with precision to 23 digits. - we can not store decimal number in int
Double DOUBLE	Decimal number - with 24 to 53 digits Precision.
BOOLEAN	Boolean values 0 OR 1 True OR False
DATE	Date in format of YYYY-MM-DD ranging from 1000-01-01 to 9999-12-31
YEAR	Year in 4 digits format ranging from 1901 to 2155

SQL Data types (signed & unsigned)

TINYINT (-128 to 127) — actual range of tinyint

It can store from this range and it's sign data type that mean signed can be chang like (-) negative (+) positive

TINYINT UNSIGNED (0 to 255) — after removing negative data

- can use to increase the length.

- examples of unsigned data is
Age # salary
this value can't not be goes Negative
so we can use UNSIGNED.

37.09

Types of SQL commands :-

DDL - Data definition Language -
- creat, alter, rename, truncate & drop
table, column in SQL

DQL - Data Query Language -
- select

DML - Data manipulation Language -
- insert, update & delete

DCL - Data control Language
- grant & revoke permission to users.

- TCL - Transaction control language -
- start transaction, commit, rollback.

'A-P' - "%[A-P]%"

select * from
wher name like "A%." OR Like "%
OR Like "%.A-%."

- examples of unsigned data is
Age # salary
this values can not be goes Negative
so we can use UNSIGNED.

37.09

Types of SQL commands :-

DDL - Data definition Language -
- creat, alter, rename, truncate & drop table, column in SQL

DQL - Data Query Language -
- select

DML - Data manipulation Language -
- insert, ~~upd~~ update & delete

DCL - Data control Language
- grant & rpvoke permission to users.

- TCL - Transaction control language -
- Start transaction, commit, rollback.

'A-P' - "%[A-P]%"

select * from
wher name like "A%." OR Like "%.A)
OR Like "%.A-%."

Database Related Queries

- CREATE Database db_name ;
- create database if not exists db_name;

- drop database db_name;
- drop database if exists db_name;

- show databases ;
- show tables

Table Related Queries

- create
- CREATE TABLE table_name (
  column_name1 datatype constraint,
  column_name2 datatype constraint);

- create table student (
  rollno INT primary key,
  name varchar (50)
  );

Firstly - drop table student;

select & view All columns

- select * from table_name;
- select * from student;

Insert

~~Instert~~

- insert into student
  (rollno, name)
  values
  (101, "Abhishek"),
  (102, "sagar");

insert into student values (103, "Aman");

Practice Qs 1.

- Create a table inside this DB to store employee info (id, name and salary).
- Add following information in the DB:
  1, "adam", 25000
  2, "bob", 30000
  3, "casey", 40000
- Select & view all your table data.

Ans.

```
- create database company;
- use company;
- create table employee (
      id int primary key,
      frame varchar(50),
      salary int );

- select * from employee
- insert into employee
  ( id , frame, salary)
  values ( 1, "Abhi", 10,000),
         ( 2, "sagar", 20,000);
```

- select * from employee.

keys

primat key

- main key
- not null, unique
- It is a column (or set of columns) in a table that uniquely identifies each row.
- There is only 1 P.K. & it should be not null.

Foreign key

- if one column is primary key in anther column that will foreign key in this table. for example.

Table ①

id	name	city-id	city
101	Abhi	1	pune
102	Sagar	2	Thane
103	sham	3	mumbai
104	Ram	1	pune

Table ②.

city_id	city-name.
1	pune
2	Thane
3	mumbai.

- Now in table ① id is primary key & in table ② city_id is primary key but city-id is also present in table ① so it is not table ones is foreign key.
- A foreign key is a column in a table that refers to the primary key in another table.

- There can be multiple FK's.
- FKs can have duplicate & null values.

---

constraints

SQL constraints are used to specify rules for data in a table.

NOT NULL - columns cannot have a null value
- col1 int NOT NULL

UNIQUE - all values in column are diff^n.
- col2 int unique.

PRIMARY KEY - makes a column unique & not null but used only for one.
- id int primary key;

we can also write like this

```
- create table temp (
  id INT,
  name VARCHAR (50),
  age INT,
  city VARCHAR (20),
  PRIMARY KEY (id)
  );
```

OR

Primary key (id, name)

- in this case name or id should be same but combination should be diffrent
- it means ~~name is rc~~ id 101 & 101 but name should be Ram & sita
- if name is same like RAM & RAM now id should be diff like 101 & 102.

Foreign key:

- prevent action that would destroy links between tables.
- Create table temp (
  cust_id int,
  foreign key (cust_id) refrences customer(id));
- in this temp table cust_id is foreign key & id is primary key from Customer table.

DEFAULT - sets the default value of a column.

salary int DEFAULT 25000

- Create table emp (
  id INT,
  Salary INT DEFAULT 25000);
- Insert into emp(id) values (101);

foreign

Check | CHECK

It can limit the values allowed in a column.

- create table city (
  id INT primary key,
  city varchar(50),
  age int
  constraint age_check check (age >= 18 & city = "Delhi"));

- create table newtab (
  age int check (age >= 18)
  );

DISTINCT

```
SELECT DISTINCT city from student;
```

if we want to know unique values from perticular column.

Where clause

To define some conditions

- select * from student WHERE marks 780;
- select * from student where city = "Mumbai";

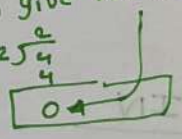
select col1, col2 . student where conditions ;

Select * from student where city = "Pune" ;

Select * from student
where marks 780 and city = "Mumbai";

Operators in where clause :-

Arithmetic operators - + , - , * , / division
% modulus. → give reminder



comparison Operators - equal to = ,
!= , > , >= , < , <=

Logical Operators - AND , OR , NOT , IN, BETWEEN,
ALL , LIKE , ANY

BITWISE Operators - & (Bitwise AND),
| (Bitwise OR)

- Select * from student
- where marks + 10 > 100 ;

AND operator

AND - to check for both conditions to be true.

select * from student where marks > 80
AND city = "Mumbai" ;

OR

OR - to check for one of the conditions to be true.

select * from student where marks > 80
OR city = "Mumbai" ;

Between - select for a given Range.

select * from student where marks
BETWEEN 80 AND 90;

IN - matches any value in the list.

Select * from student where city
in ("Delhi", "Mumbai") ;

NOT IN - to negate the given condition.

```
select * from studen where city NOT IN
("Delhi", "Mumbai")
```

Limit clause

sets an upper limit on number of rows to be returned.

```
select * From student LIMIT 3;
```

ORDER BY clause

- `select * from student order by city ASC;`
- `select * from student order by marks DESC Limit 3;`

ASC
DESC.

Aggregate Functions

Aggregate functions perform a calculation on a set of values, and return a single value.

- count() → count of values
- max() → get maximum value in a range
- MIN() → minimum value
- Sum() → sum of values
- Avg() → Avg of marks OR values.

- SELECT max(marks) from students;
- SELECT avg(marks) from student;

Group by clause

- Groups rows that have the same values into summary rows
- It collects data from multiple records and groups the result by one or more column.
- Generally we use group by with some aggregation function.

Q- count number of students in each city

- SELECT city , count (name)
  FROM student
  GROUP BY city ;

- write a tro Query to find avg marks in each city in ascending Order.

- select city , avg (marks)
  From student
  GROUP BY city
  Order by avg (marks) DESC ;

- select mode , count (customer)
  From ~~stud~~ payment
  Group By mode ;

- select grade , count (rollno)
  from student
  group by grade
  order by grade ;

Having clause

- We know that if we want to add any condition on rows we can use where clause
- But having clause is used to apply condn on group.
- Similar to where i.e. applies some condition on rows.
- used when we want to apply any condition after grouping.

---

count number of students in each city where max marks cross 90.

~~- select count(name), city~~

- select city, count (name)
  from student
  Group by city
  having max (marks) > 90;

not null and unique

General order of clauses.

```
SELECT   columns
FROM     table_name
WHERE    condition
GROUP BY columns
HAVING   condition
ORDER BY column(s) ASC
LIMIT    3 ;
```

Table related Queries

UPDATE (to update existing rows).

SET.

- UPDATE table_name
  SET col1 = val1 , col2 = val2
  where condition;

```
- UPDATE   student
  SET      grade = "EX"
  WHERE    grade = "A"
```

Now in this all the data in grade column where A is present now replaced with "EX".

```
SET SQL_SAFE_UPDATES = 0;
```

```
update student
set marks = marks + 1;
```

DELETE (to delete existing rows)

```
DELETE FROM student
WHERE marks < 33;
```

1:59:05

Revisiting foreign key.

- ```
  Create table dept (
    id INT primary key,
    name varchar(50)
    );
  ```
- ```
  Create table teacher (
    id INT primary key,
    name varchar(50),
    dept_id INT,
    Foreign key (dept_id) References dept(id)
    );
  ```

Table ① dept

id (PK)	name
101	science
102	English
103	Hindi

↑ parent table

F.K ⟷

Table ② teacher

id (PK)	name	dept_id (F.K)
101	Adam	101
102	Bob	102
103	Sagar	102
104	soham	103

↑ child table

Cascading in SQL insure that the change in the parent table reflect in the child table

Cascading for Foreign key < ON delete / ON update

2:06 : 20

ON Delete Cascade

When we create a foreign key using this option, it deletes the referencing rows in the child table when the referenced row is deleted in the parent table which has a primary key.

ON Update Cascade

When we create a foreign key using UPDATE CASCADE the Referencing rows are updated in the child table when the referenced row is updated in the parent table which has a primary key.

```
create table student (
    id INT primary key,
    courseID INT,
    foreign key (courseID) REFERENCES Course(id)
    ON DELETE CASCADE
    ON UPDATE CASCADE
    );
```

Table Related Queries

Alter (to change the schema)

ADD Column

- ALTER TABLE table_name
  ADD column column_name datatype constraint;

Drop Column

- ALTER TABLE table_name
  drop column column_name;

RENAME TABLE

- ALTER TABLE table_name
  Rename to new_table_name;

- Alter table student
modify column age varchar(2);
.... change data type

- Alter table student
change age stu_age INT;
.... change column name aswell d.T

TRUNCATE TABLE

TRUNCATE TABLE student;

.... it will only delete all the data from table, not the table.

* ~~Triggers~~ Triggers :- in SQL triggers are special procedures that are automatically triggers

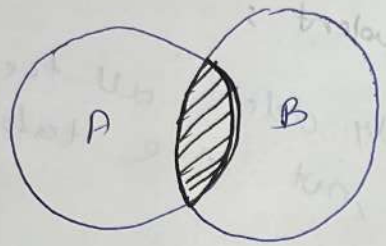
* Views :- views in SQL is a virtual table that is based on SQL queries.

* Case condition :- if else condition

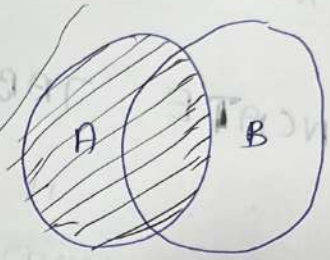
Joins in SQL

Join is used to combine rows from two or more tables, based on a related column between them.

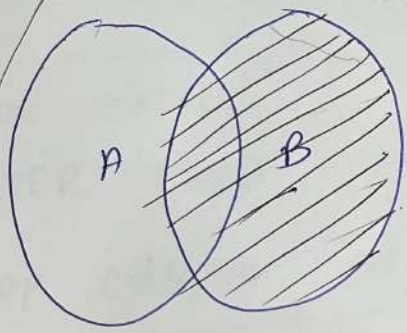
Types of Joins. (venn diagram)



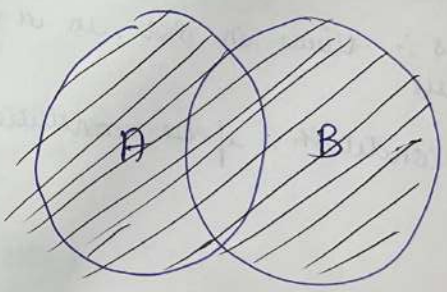
INNER JOIN



LEFT JOIN



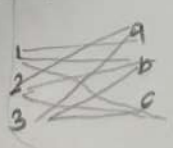
RIGHT JOIN



FULL JOIN

Outer join

cross join

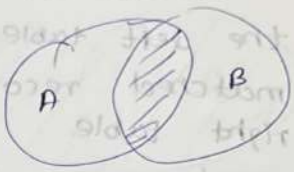


om two
column

N

INNER JOIN

Returns records that have matching values in both tables.



Syntax

```
SELECT columns(s)
from   table A
INNER JOIN table B
ON tableA.col_name = tableB.col_name;
```

-
```
select *
from student
Inner join course
on student.std_id = course.std_id;
```

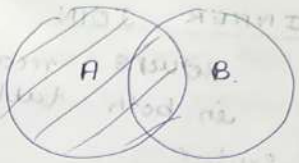
Alias

Alternative name.

```
select *
from student as s
Inner join course as c
on s.std_id = c.std_id ;
```

left join

Returns all records from the left table, and the matched records from the right table



syntax.

```
SELECT column(s)
From tableA as A
left Join tableB as B
on A.std_id = B.std_id ;
```

Right Join

Returns all records from the Right table, & the matched records from the left table

```
select column(s)
from table A as A
Right Join table B as B
on A.std_id = B.std_id ;
```

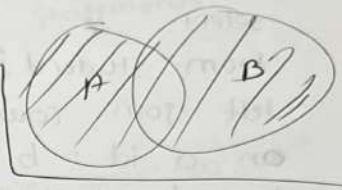
alter table employee add foreign key (dept-id) references dept cid)

Right Join

Join

full Join

Return all records when there is a match in either left or right table.

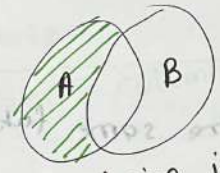


syntax in MySQL

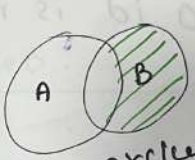
```
- select * from student as s
- LEFT join course as b
- ON a.id = b.id
- UNION
- select * from student as s
- Right join course as b
- on a.id = b.id ;
```

exclusive Join :-

Write SQL commands to display the right exclusive join.



left exclusive join



Right exclusive join.

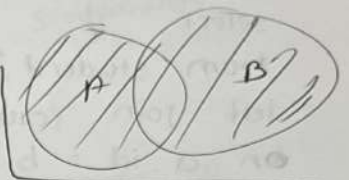
```
select *
from student as s
left join course as c
on s.id = c.id
where c.id is null;
```

```
select *
from student as s
Right join course as c
on s.id = c.id
where s.id is null;
```

full join

Return all records when there is a match in either left or right table.

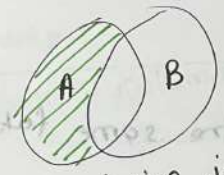
syntax in . MySQL



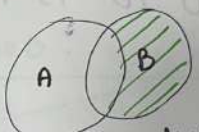

```
- select * from student as s
- LEFT join course as b
- ON a.id = b.id
- UNION
- select * from student as s
- Right join course as b
- on a.id = b.id ;
```

exclusive Join :-

Write SQL commands to display the right exclusive join:



left exclusive join



Righ exclusive join.

```
select *
from student as s
left join course as c
on s.id = c.id
where c.id is null;
```

```
select *
from student as s
Right join course as c
on s.id = c.id
where s.id is null;
```

full exclusive join

```
select *
from student as a
left join course as b
on a.id = b.id
where b.id is null
union
select *
from student as a
right join course as b
on a.id = b.id
where a.id is null;
```

self join

- find information in the same table.

```
select a.name as manager_name, b.name
as Name
from employee as a
join employee as b
on a.id = b.manager_id;
```

id	name	manager_id

Union -

- It is used to combine the result-set of two or more select statements.
- Gives UNIQUE records.

Rules to use it

- every select should have same NO. of columns
- columns must have similar data types.
- columns in every select should be in same order.

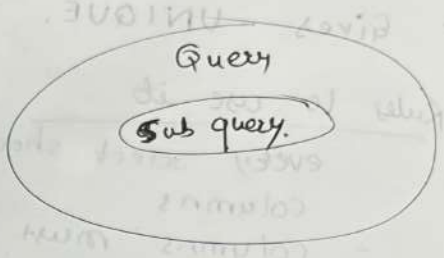
Syntax

```
select columns from tableA
union
select columns from tableB.
```

UNION ALL, will give duplicate values too.

SQL Sub Queries

- A subquery or inner query or a nested query is a query within another SQL query.
- It involves 2 selected statements.



Syntax

```
select columns
from table_name
where col_name operator
(subquery);
```

~~select name,~~

(1) select avg(marks) from student; ⟶ output - 87.666

(2) select name, marks from student where marks > 87.666;

(3) select name, marks from student where marks > (select avg(marks) from student);

Nested query

- select rollno * from student
  where rollno %2 = 0 ;
- select name
  from student
  where rollno in (102, 104, 106);

- select name
  from student
  where rollno in
  (select rollno from student where rollno %2=0)
  );

SQL Sub queries with from

- select * from student ;
- select MAX (marks)
  from (select * from student where
  city = "Delhi") as temp;

- 87.666

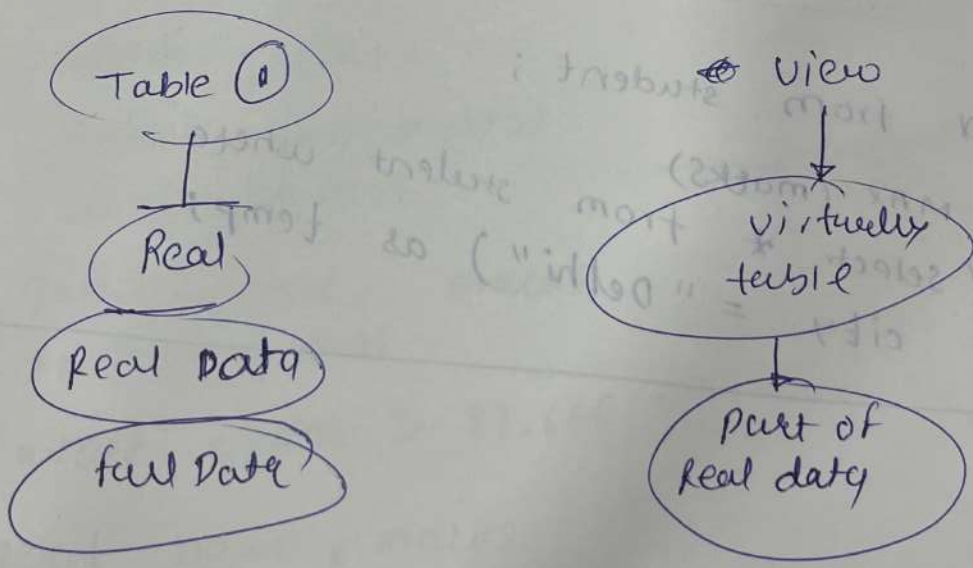
udent);

MySQL views

A view is a virtual table based on the result-set of an SQL statement.

```
- create view view1 as
  select rollno, name from student;
  select * from view1;
```

A view always shows up-to-date data. The database engine recreates the view, every time a user queries it.

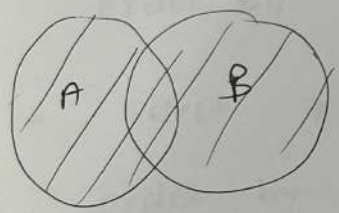


set Operators

- set operators allow you to combine the results of two or more select (queries) statements into a single result set based on set theory principles.
- These operators allow you to perform operations such as combining, interesecting, or subtracting sets of data.

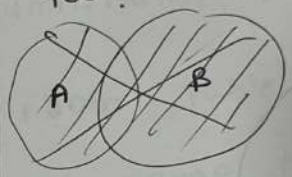
① union

- combine two or more sets
- removes duplicates

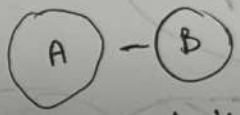


② union all

- combine two or more sets
- shows dupliates too.



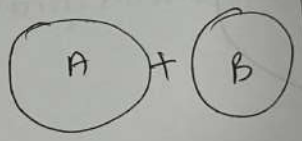
③ ~~Intersect~~ Except OR Minus



- removes dupliate

④ Minus OR Except

- select ..... from table
union
- select ... from table;



Rules

- every select statement must have the same No. of columns.
- The columns must also have similar data types.
- The columns in every select statement must also be in the same order.

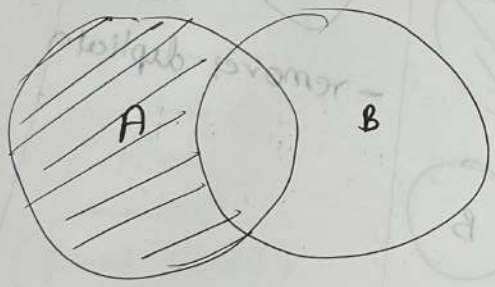
Union

```
- select * from employee
  UNION
  select * from manager ;
```

Union All

```
- select * from employee
  union all
  select * from manager ;
```

Except

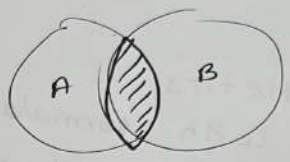



```
select * from employee

Except
Select * from manager ;
```

Intersect

- selekt * from employee
  Intersect
  select * from manager;



Last (select)

- select Top 1 * from table_name
  order by column-name Desc;

- select * from table-name
  order by column-name Desc
  Limit 1;

Topic to covet

SQL - Assignment.

1. 2. 3. 4.

- lec flow.

Introduction to statictics

- All theory with formulae.
- t (distribution), F (Distribution) (chi)^2 distribution using excel
  - calculation.
- and assessment. ∴
- lec flow.

Excel
- assignments
- assessments.
- lecture flow ✓

advanced excel & stat. ✓
- Assignments
- lecture flow (DE) ✓
- practice.