SQL IN ONE VIDEO

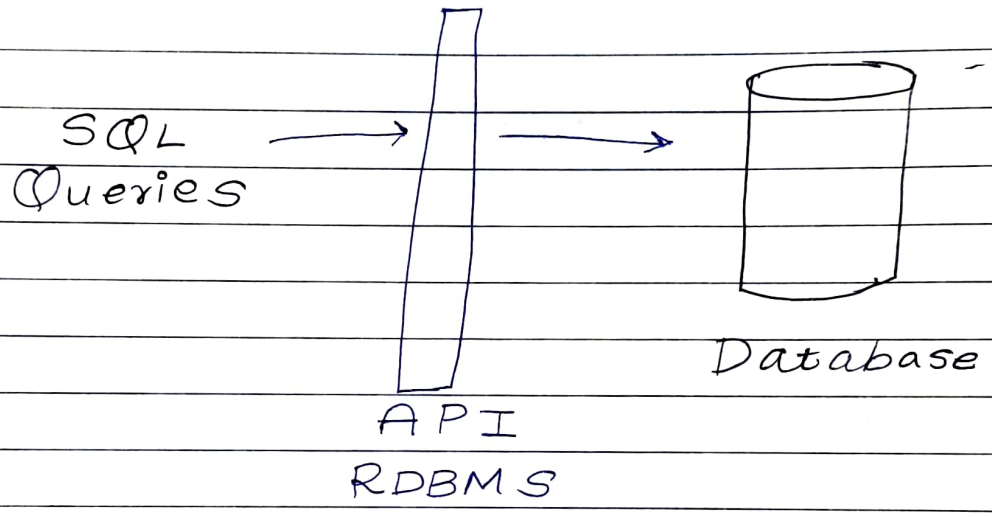
— By Lakshy Bhaiya

SQL

It is structured Query Language.

RDBMS Types

MySQL, Oracle, Msacess etc



CRUD Operations

C - create
R - Read
U - UPdate
D - Delete

SQL is just a language

RDBMS → 1. MySQL 2. MSSQL sever
3. Oracle
4. IBM

~~Language~~
Software

Above all are softwares

Difference between SQL & MySQL

SQL	MySQL
1. Query language.	1. MySQL itself a RDBMS
2. Way to access Data	2. CRUD done on it using SQL.

My first connection on workbench

```
create database newName;
show databases;
use newName;
CREATE TABLE student (
id INT PRIMARY KEY,
name VARCHAR (255)
);
INSERT INTO student VALUES (1,'Pratham');
SELECT * FROM student;
```

Data types in SQL

CHAR - For storing string
Fixed size

VARCHAR - Variable size

e.g →

VARCHAR(255) → [R|A|M]

CHAR(255) → [R|A|M]

BLOB – Audio, video

for more datatypes in detail read or refer original notes.

Refer W3school

signed, Unsigned

TINYINT → (-128 to 127)

UNSIGNED TINYINT → (0 to 255)

Advanced Datatype.

JSON – Key value pair type structure.

SQL Types of commands.

1. DDL (Data Definition Language)
   D R C A T
2. DML (Manipulation)
   I U D
3. DRL/DQL (Retrieval/query)
   1. SELECT
4. DCL (control)
   1. Grant
   2. Revoke
5. Transaction control language (TCL)
   1. Start transaction
   2. commit
   3. RollBack
   4. savepoint

Create DB

IF NOT EXISTS is a good practice.

SELECT
- To Retrieve data, all data, some specific data.
- order of executation is from Right to Left.

Can we use SELECT keyword without using FROM clause?

Using Dual table
Dual tables are dummy tables created by MySQL, help users. to do certain obvious actions without referring to user defined tables.

e.g → SELECT 55+11;
SELECT now();
SELECT ucase(); etc

WHERE
e.g →
SELECT * FROM worker WHERE DEPARTMENT = 'ADMIN';

Pattern Matching

'%' → Any number of characters
'_' → only one character.

Many more in original Notes.

Sorting

```
SELECT * FROM Worker ORDER BY
  salary;
```

Add desc at last for descending order.
Ascending is default.

Data Grouping

Find the no. of empolyees working in different department.

GROUP BY → Aggregation function
count, sum, etc

GROUP BY without Aggregation is similar to DISTINCT.

Having

GroupBy is there, then only we make use of Having.
similar to WHERE.

Department ~~Hav~~ count having more than 2 and more workers

```
SELECT DEPARTMENT, COUNT(DEPARTMENT)
  FROM Worker GROUP BY DEPARTMENT
  HAVING COUNT(DEPARTMENT) > 2;
```

WHERE VS HAVING

where - direct filtering
Having - filtering in groups

DDL constraints

1. Primary Key constraints
   1. NOT NULL
   2. Unsame
   3. only One P.K

Good Practice ⟶ Primary key should be INT.

For e.g ⇒

```
CREATE TABLE Customer C
    id INT PRIMARY KEY,
    branch-id INT,
    First_NAME CHAR(50));
```

1st method

OR

```
PRIMARY KEY (id)
```

2nd method

2. Foreign key
   1. Foreign key refers to the primary key of other.

```
e.g foreign key (Cust-ID) references Customer(id);
```

(Cust-ID: FK; Customer(id): Primary key)

3. Unique

```
Create table customer C
        Name VARCHAR(255) UNIQUE;
)
```

4 CHECK

```
CREATE table Account (
    :
    CONSTRAINT Acc_balance check
                (balance > 100);
);
```

5. Default

```
CREATE table Account (
        balance INT NOT NULL Default 0;
    :
);
```

ALTER operations in SQL

Add, Modify, Rename, Drop, Rename the table.

Desc command is used to describe that particular table.

Common syntax is

AITER TABLE Account .....

Detailed syntax are given in original notes.

DML (Data Manipulation Language)

Insert, to insert data
e.g

```
INSERT INTO customer (id, LName) VALUES
(2, 'PRATHAM');
```

```
UPDATE Customer SET Gender = 'F' where
id = 3;
```

Update multiple Rows

```
SET SQL_SAFE_UPDATES = 0;
UPDATE customer SET Gender = 'F';
```

We have to off safe mode before that because SQL finds this data change as fishy (malware) backed.

Here we are chaning all column data

DELETE
e.g →

```
DELETE FROM Customer WHERE id = 121;
```

Can delete table too, have to off the SQl safe mode.

Referential constraints

1. Insert
2. Delete constraints
   - → ON Delete cascade
   - → ON Delete set NULL

As a default,

RDBMS does not allow user or admin to delete parent entity

ON DELETE CASCADE

If we delete parent entity, the child entity corresponding to it, gets deleted.

ON DELETE SET NULL

If we delete parent entity, the child entity corresponding to it, sets as NULL.

REPLACE

It works into 2 ways
1. Data already present, Replace
2. Data not present, INSERT.

e.g. →

```
Replace INTO Customer(id, cName) values
  (001, 'Pratham');
```

If other attributes are not mentioned to be replaced then it is set as default or NULL.

REPLACE VS UPDATE

→ UPDATE, if data is not changed then other data is not affected.

→ If row is not present, Replace will add a new row while UPDATE will do Nothing.

JOINS

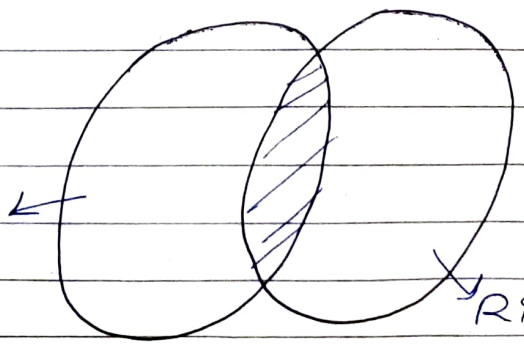
for fetching data, we use JOINS

INNER JOINS

To apply joins, there should be a common attribute

Syntax:

```
Select C.*, O.* From customer AS C
                    ↑
                (Aliasing)
INNER JOIN orders AS O
ON C.id = O.cust id;
```

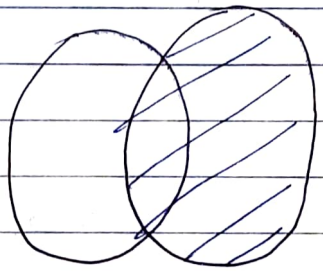


OUTER JOIN

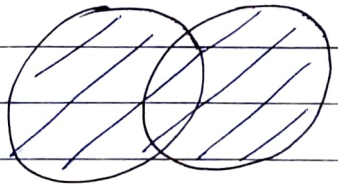
Syntax.

```
SELECT c.*, o.* FROM customer as c
LEFT JOIN orders as o ON c.id = o.cust_id
```

RIGHT JOIN



FULL JOIN



As keyword Full Join is not there in mysql
Hence we will emulate it
Left Join ∪ Right Join

Syntax

```
select * from LeftTable as l
  Left Join rightTable as r
On l.key = r.key
UNION
 Select * from lefttable as l
Right JOIN Right table as r
 ON l.key = r.key;
```

CROSS JOIN

Cartesian Product

for e.g →

T1	TR
5 rows	10 rows

Resultant :- 5 x 10 = 50 rows

- NO Industrial use.

SELF JOIN

- Emulated using 'INNER JOIN' and 'AS'.

e.g → Select e1.id, e2.id, e2.name.
FROM employee as e1
INNER JOIN
employee as e2
ON e1.id = e2.id;

LEFT JOIN and RIGHT JOIN can be interchangable.

Can we use JOIN w/o using JOIN Keyword? (Talking about INNER-JOIN)
yes
Syntax:
Select * From leftable, Righttable
WHERE Lefttable.id = Rightableid.

SET OPERATIONS

UNION - combines two or or more set operations

Syntax:

```
select * from Table 1 UNION
 select * from Table 2;
```

JOINS Vs UNION

UNION

Table 1

Col 1	Col 2
A	1
B	1
C	2

Table 2

Col 1	Col 2
A	1
B	2
D	3

UNION Result ⇒

Col 1	Col 2
A	1
B	1
C	2
D	3

JOIN Result ⇒

Col 1	Col 2	Col 1	Col 2
A	1	A	1
B	1	B	2
C	2	NULL	NULL
NULL	NULL	D	3

UNION is rowwise
Joins is columnwise

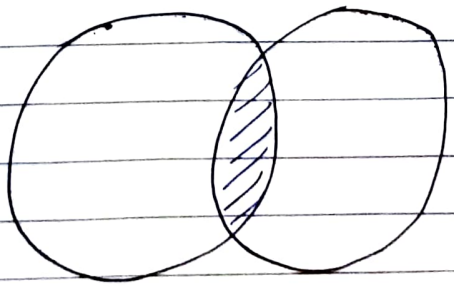
JOIN	SETOperations
1. Numbers of columns may or may not be same.	1. Number of columns must be same
2. Data types of two tables can be different.	2. Must be same
3. Can generate both distinct & duplicate rows	3. Only distinct
4. Combines results horizontally	4. Combines results Vertically.

INTERSECT

- Have to emulate

Syntax:

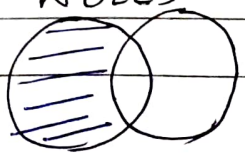
```
Select DISTINCT id from T1
( INNER JOINT T2 using (id);
```



MINUS

Syntax:

```
Select id from T1 LEFT JOIN T2
using (id) where T2.id is NULL;
```



Sub-Queries

- Alternative to JOINS
→ q1 (q2)
outer query depends on inner query.

e.g 1 → Select * from table
where id IN (select id From table
where name = 'Lak');

e.g 2. →
-- WHERE Clause different table
--emp details working in more than
1 Project.
Select * FROM employee where id in (
select empID FROM project GROUP By
empID having count (empID) > 1
);
e.g. 3 →
-- single value subquery
-- emp details having age > avg (age)
select * from Employee where age >
(select avg (age) From employee);

e.g 4 →
-- FROM clause
-- select Max age person whose
first name contains 'a'
SELECT max (age) from (select * from
employee where fname Like '%a%') As
temp;

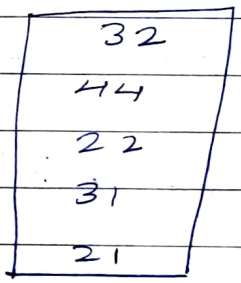
Corelated Queries

- Inner query that refers the outer query.

-- Find 3rd oldest employee

Code:

```
SELECT *
FROM Employee e1
WHERE 3 = (
  SELECT COUNT(e2.age)
  FROM Employee e2
  WHERE e2.age >= e1.age
);
```



Dry Run

(1) e1.age = 32
inner
⇒ 2 (How many entries bigger than this age + 1)

(2) e1.age = 44
⇒ 1

(3) 22
⇒ 4

(4) e1.age = 31
⇒ 3
(matches)

[For one outer query, INNER query loops]

JOINS	SUBQUERIES
1. Faster	1. Slower
2. calculation burden on DBMS	2. On user
3. Complex, difficult to understand	3. Easy
4. choosing optimal join for optimal use case is difficult	4. Easy

SQL VIEW

mysql provide view.

Define view → view only ↴

name	age

aliasing as → custom-view

If we apply this custom_view we can only see name & age.