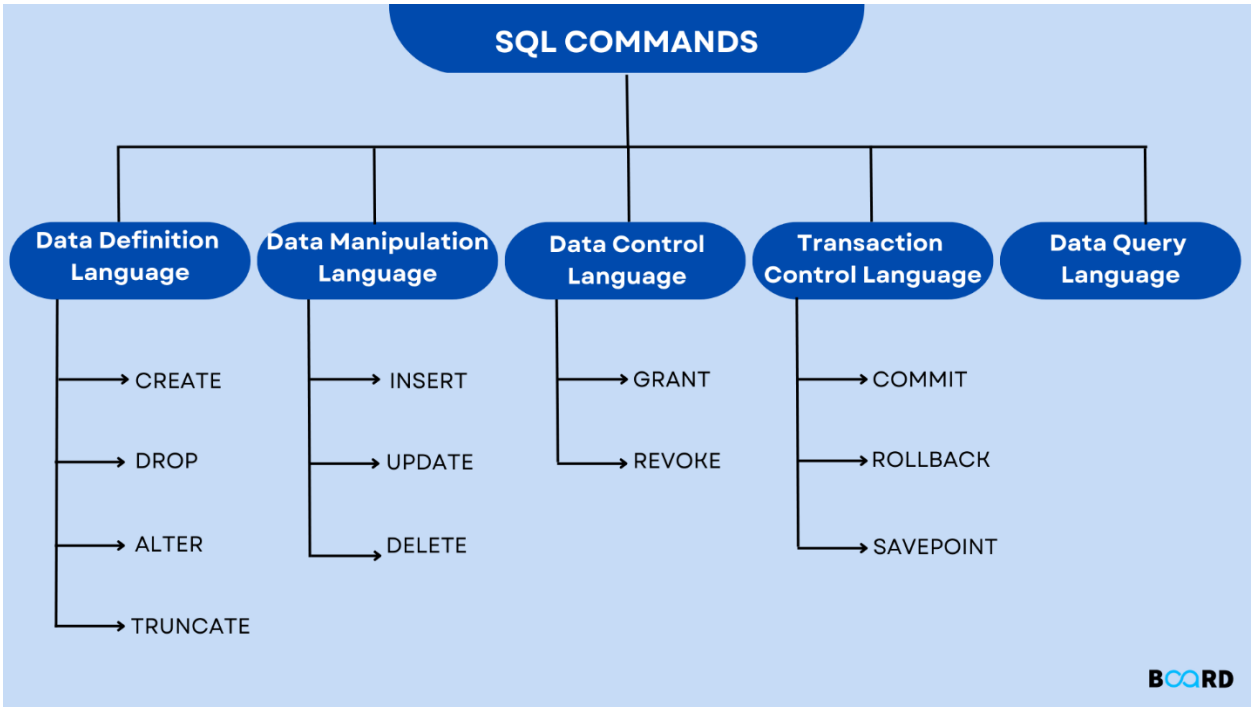
SQL Query

Theory:

1. SQL:
   Structured Query Language (SQL) is a specialized programming language for managing relational database data. It allows users to store, manipulate, and retrieve data efficiently in databases like MySQL, SQL Server, Oracle, and more.
2. SQL commands:

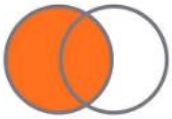


3. Aggregation Function

Function Name	Meaning	Example
SUM(column name)	Total sum of the values in a numeric column	`SUM(salary)`
AVG(column name)	Average of the values in a column	`AVG(salary)`
MAX(column name)	Largest value in a column	`MAX(salary)`
MIN(column name)	Smallest value in a column	`MIN(salary)`
COUNT(*)	Count of the number of rows selected	`COUNT(*)`

4. JOIN in SQL:

LEFT JOIN



Everything on the left
+
anything on the right that matches

```
SELECT *
FROM TABLE_1
LEFT JOIN TABLE_2
ON TABLE_1.KEY = TABLE_2.KEY
```

RIGHT JOIN

Everything on the right
+
anything on the left that matches

```
SELECT *
FROM TABLE_1
RIGHT JOIN TABLE_2
ON TABLE_1.KEY = TABLE_2.KEY
```

OUTER JOIN

Everything on the right
+
Everything on the left

```
SELECT *
FROM TABLE_1
OUTER JOIN TABLE_2
ON TABLE_1.KEY = TABLE_2.KEY
```

INNER JOIN

Only the things that match on the left AND the right

```
SELECT *
FROM TABLE_1
INNER JOIN TABLE_2
ON TABLE_1.KEY = TABLE_2.KEY
```

5. SQL LIKE OPERATOR

Pattern	Meaning
‘a%’	Match strings that start with ‘a’
‘%a’	Match strings with end with ‘a’
‘a%t’	Match strings that contain the start with ‘a’ and end with ‘t’.
‘%wow%’	Match strings that contain the substring ‘wow’ in them at any position.
‘_wow%’	Match strings that contain the substring ‘wow’ in them at the second position.
‘_a%’	Match strings that contain ‘a’ at the second position.
‘a_ _%’	Match strings that start with ‘a and contain at least 2 more characters.

LAB WORK

1. Create a table named Vehicle with veh_number as primary key and the following attributes:
   veh_type, veh_brand, veh_year, veh_mileage, veh_owner, veh_photo, veh_price.

```
CREATE TABLE Vehicle(
veh_number int NOT NULL PRIMARY KEY,
veh_type varchar(50),
veh_brand varchar(50),
veh_year int,
veh_mileage varchar(50),
veh_owner varchar(50),
veh_photo varchar(50),
veh_price int,
);
SELECT * FROM Vehicle;
```

sults | Messages

veh_number	veh_type	veh_brand	veh_year	veh_mileage	veh_owner	veh_photo	veh_price

2. Enter a full detailed information of a vehicle.

```
SELECT * FROM Vehicle;
INSERT INTO Vehicle VALUES
(1,'Van','TATA', '2001','80km','Aayush','C:\Users\acer\Desktop\pexels-mikebirdy-116675.jpg','220000');
```

00 % | Results | Messages

	veh_number	veh_type	veh_brand	veh_year	veh_mileage	veh_owner	veh_photo	veh_price
1	1	Van	TATA	2001	80km	Aayush	C:\Users\acer\Desktop\pexels-mikebirdy-116675.jpg	220000

3. Increment vehicle's price by 10,000.

```
SELECT * FROM Vehicle;
UPDATE Vehicle
SET veh_price+=10000;
```

00 % | Results | Messages

	veh_number	veh_type	veh_brand	veh_year	veh_mileage	veh_owner	veh_photo	veh_price
1	1	Van	TATA	2001	80km	Aayush	C:\Users\acer\Desktop\pexels-mikebirdy-116675.jpg	230000

4. Remove all vehicle's records whose brand contains character 'o' in second position.

```
SELECT * FROM Vehicle;
DELETE FROM Vehicle
WHERE veh_brand LIKE '_o%';
```

Results Messages

veh_number	veh_type	veh_brand	veh_year	veh_mileage	veh_owner	veh_photo	veh_price
1	Van	TATA	2001	80km	Aayush	C:\Users\acer\Desktop\pexels-mikebirdy-116675.jpg	230000

5. Display the total price of all vehicles.

```
SELECT * FROM Vehicle;

SELECT SUM(veh_price) AS total_price from Vehicle;
```

Results Messages

total_price
230000

6. Create a view from above table.

```
SELECT TOP (1000) [veh_number]
      ,[veh_type]
      ,[veh_brand]
      ,[veh_year]
      ,[veh_mileage]
      ,[veh_owner]
      ,[veh_photo]
      ,[veh_price]
  FROM [library_db].[dbo].[vehicle_view]
```

Results Messages

veh_number	veh_type	veh_brand	veh_year	veh_mileage	veh_owner	veh_photo	veh_price
1	Van	TATA	2001	80km	Aayush	C:\Users\acer\Desktop\pexels-mikebirdy-116675.jpg	230000

7. Change data type of year to datetime.

```
ALTER TABLE Vehicle ALTER  COLUMN
veh_year datetime;
SELECT * from vehicle;
```

100 %

Results Messages

	veh_number	veh_type	veh_brand	veh_year	veh_mileage	veh_owner	veh_photo	veh_price
1	1	Van	TATA	1905-06-25 00:00:00.000	80km	Aayush	C:\Users\acer\Desktop\pexels-mikebirdy-116675.jpg	230000

8. Display details of vehicles ordering on descending manner in brand and by mileage when brand matches.

```
SELECT * from vehicle;
SELECT * FROM Vehicle
ORDER BY
veh_brand DESC, veh_mileage DESC;
```

esults Messages

veh_number	veh_type	veh_brand	veh_year	veh_mileage	veh_owner	veh_photo	veh_pri
1	Van	TATA	1905-06-25 00:00:00.000	80km	Aayush	C:\Users\acer\Desktop\pexels-mikebirdy-116675.jpg	230000

CONCLUSION:

In conclusion, the provided SQL tasks encompass a range of fundamental database operations, including table creation, data insertion, updates, deletions, aggregate functions, view creation, and data type modification. These operations are essential for effective database management and ensure the integrity, accessibility, and organization of data within a relational database system. By mastering these tasks, one can efficiently handle and manipulate data to meet various business and application requirements.