PROJECT NAME:

OLA CAB BOOKING SYSTEM

NAME : KRITIKA SHIVAJI SINGH

INTRODUCTION

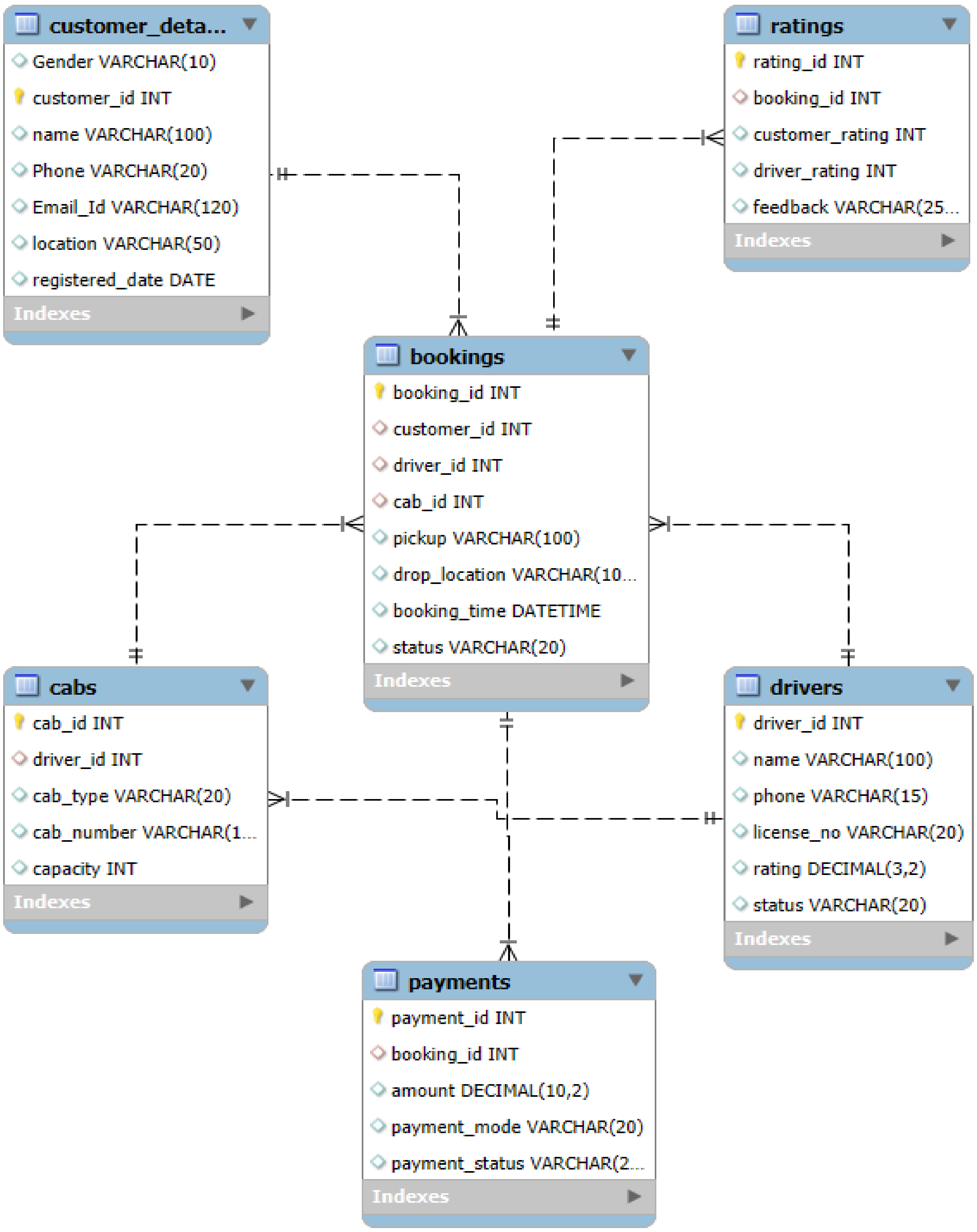
The Ola Cab Booking System is a digital platform designed to manage the process of booking rides between customers and drivers. With the growing demand for convenient and affordable transportation, ride-hailing services like Ola have become an essential part of urban life. This system ensures seamless coordination between customers, drivers, and vehicles, while also handling payments and customer feedback in an efficient way. By creating a structured database for the entire process, the project aims to replicate real-world cab booking operations in a simplified academic setting.

At its core, the system focuses on managing key entities such as Customers, Drivers, Cabs, Bookings, Payments, and Ratings. Customers can register and book rides, while drivers are assigned based on availability and cab type. Each booking is linked to a cab and a driver, and payments are processed through various modes such as UPI, cash, or cards. Additionally, customer and driver ratings are recorded to improve service quality and accountability. This relational model ensures that all critical data is interconnected and can be retrieved with simple SQL queries.

From a technical perspective, the project is developed using SQL concepts such as Data Definition Language (DDL), Data Manipulation Language (DML), and Data Query Language (DQL). It involves creating normalized tables, establishing relationships through primary and foreign keys, and executing queries for analysis such as revenue reports, popular cab types, top-performing drivers, and repeat customers. Advanced SQL operations like joins, subqueries, aggregate functions, and views are also implemented to provide deeper insights into system performance.

Overall, the Ola Cab Booking System project not only demonstrates database design and query skills but also reflects the real-world application of technology in the transportation sector. It highlights how structured data management improves efficiency, ensures transparency, and enhances user satisfaction in modern ride-hailing services. This project serves as a practical example of applying database management concepts to a familiar and highly relevant domain.

ER DIAGRAM

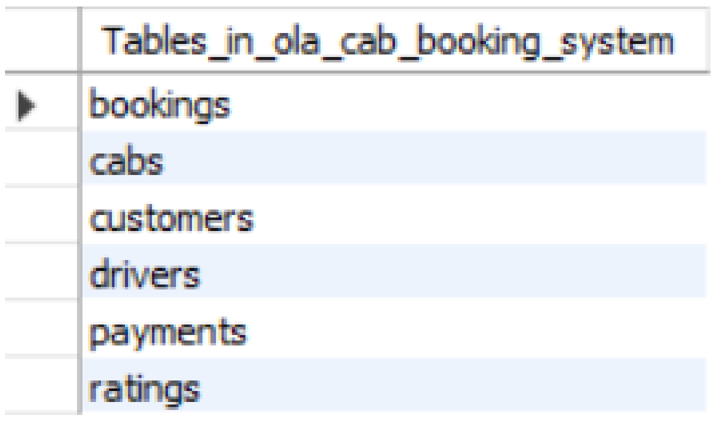


customer_deta...
Gender VARCHAR(10)
customer_id INT
name VARCHAR(100)
Phone VARCHAR(20)
Email_Id VARCHAR(120)
location VARCHAR(50)
registered_date DATE
Indexes
ratings
rating_id INT
booking_id INT
customer_rating INT
driver_rating INT
feedback VARCHAR(25...
Indexes
bookings
booking_id INT
customer_id INT
driver_id INT
cab_id INT
pickup VARCHAR(100)
drop_location VARCHAR(10...
booking_time DATETIME
status VARCHAR(20)
Indexes
cabs
cab_id INT
driver_id INT
cab_type VARCHAR(20)
cab_number VARCHAR(1...
capacity INT
Indexes
drivers
driver_id INT
name VARCHAR(100)
phone VARCHAR(15)
license_no VARCHAR(20)
rating DECIMAL(3,2)
status VARCHAR(20)
Indexes
payments
payment_id INT
booking_id INT
amount DECIMAL(10,2)
payment_mode VARCHAR(20)
payment_status VARCHAR(2...
Indexes

Databases:

Create database Ola_Cab_Booking_System
Use Ola_Cab_Booking_System

Tables in Ola_Cab_Booking_System Database:



Tables_in_ola_cab_booking_system
bookings
cabs
customers
drivers
payments
ratings

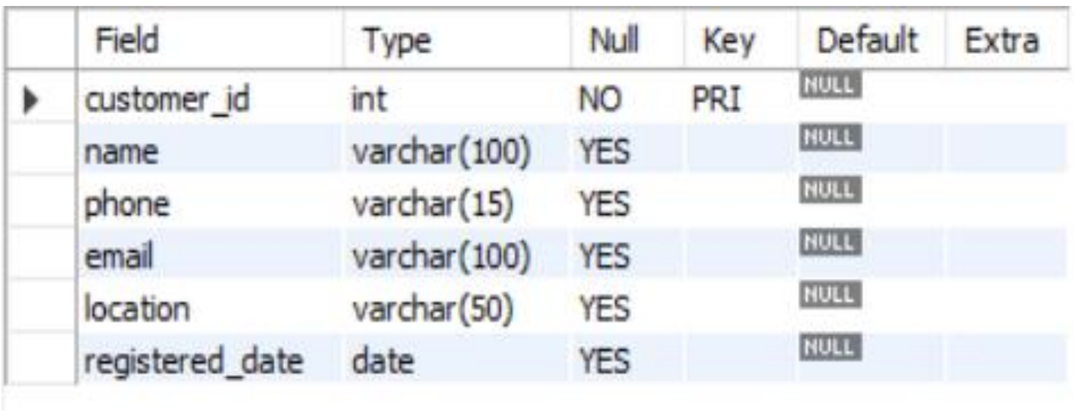
- **Data Definition language (DDL):**

  Creating Tables:

  A) Customers

  CREATE TABLE Customers (customer_id INT PRIMARY KEY, name VARCHAR(100), phone VARCHAR(15), email VARCHAR(100), location VARCHAR(50), registered_date DATE);

Desc Customers;

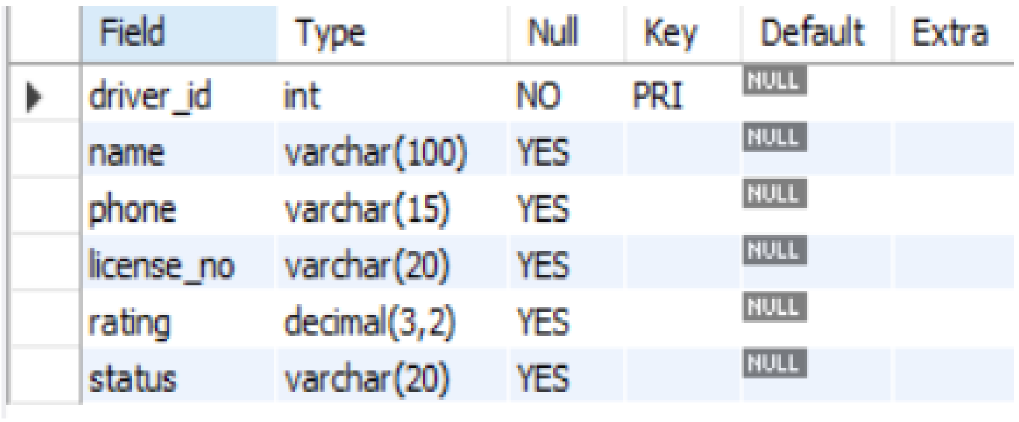


	Field	Type	Null	Key	Default	Extra
▶	customer_id	int	NO	PRI	NULL	
	name	varchar(100)	YES		NULL	
	phone	varchar(15)	YES		NULL	
	email	varchar(100)	YES		NULL	
	location	varchar(50)	YES		NULL	
	registered_date	date	YES		NULL	

B) Drivers

CREATE TABLE Drivers (driver_id INT PRIMARY KEY,name VARCHAR(100), phone VARCHAR(15), license_no VARCHAR(20), rating DECIMAL(3,2),status VARCHAR(20));

Desc Drivers;

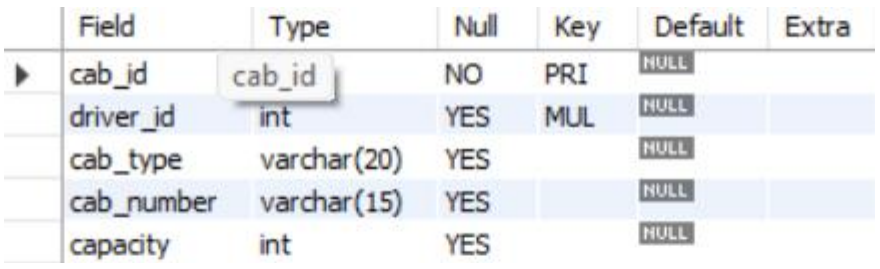


	Field	Type	Null	Key	Default	Extra
▶	driver_id	int	NO	PRI	NULL	
	name	varchar(100)	YES		NULL	
	phone	varchar(15)	YES		NULL	
	license_no	varchar(20)	YES		NULL	
	rating	decimal(3,2)	YES		NULL	
	status	varchar(20)	YES		NULL	

C) Cabs

```
CREATE TABLE Cabs (
 cab_id INT PRIMARY KEY,
 driver_id INT,
 cab_type VARCHAR(20),
 cab_number VARCHAR(15),
 capacity INT,
 FOREIGN KEY (driver_id) REFERENCES Drivers(driver_id));
```

Desc Cabs;

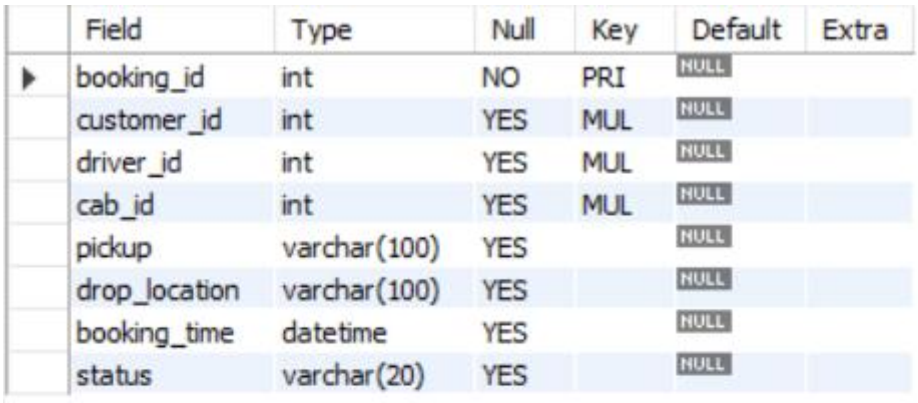


	Field	Type	Null	Key	Default	Extra
▶	cab_id	cab_id	NO	PRI	NULL	
	driver_id	int	YES	MUL	NULL	
	cab_type	varchar(20)	YES		NULL	
	cab_number	varchar(15)	YES		NULL	
	capacity	int	YES		NULL	

D) Bookings

```
CREATE TABLE Bookings (  booking_id INT PRIMARY KEY,customer_id INT,
driver_id INT, cab_id INT,pickup VARCHAR(100), drop_location
VARCHAR(100), booking_time DATETIME, status VARCHAR(20), FOREIGN KEY
(customer_id) REFERENCES Customers(customer_id), FOREIGN KEY (driver_id)
REFERENCES Drivers(driver_id),  FOREIGN KEY (cab_id) REFERENCES
Cabs(cab_id));
```

Desc Bookings;

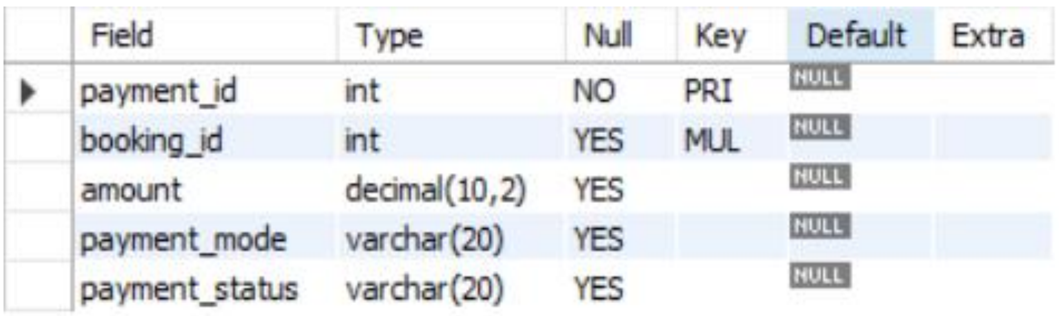


	Field	Type	Null	Key	Default	Extra
▶	booking_id	int	NO	PRI	NULL	
	customer_id	int	YES	MUL	NULL	
	driver_id	int	YES	MUL	NULL	
	cab_id	int	YES	MUL	NULL	
	pickup	varchar(100)	YES		NULL	
	drop_location	varchar(100)	YES		NULL	
	booking_time	datetime	YES		NULL	
	status	varchar(20)	YES		NULL	

E) Payments

CREATE TABLE Payments (payment_id INT PRIMARY KEY, booking_id INT, amount DECIMAL(10,2), payment_mode VARCHAR(20), payment_status VARCHAR(20), FOREIGN KEY (booking_id) REFERENCES Bookings(booking_id));

Desc Payments;

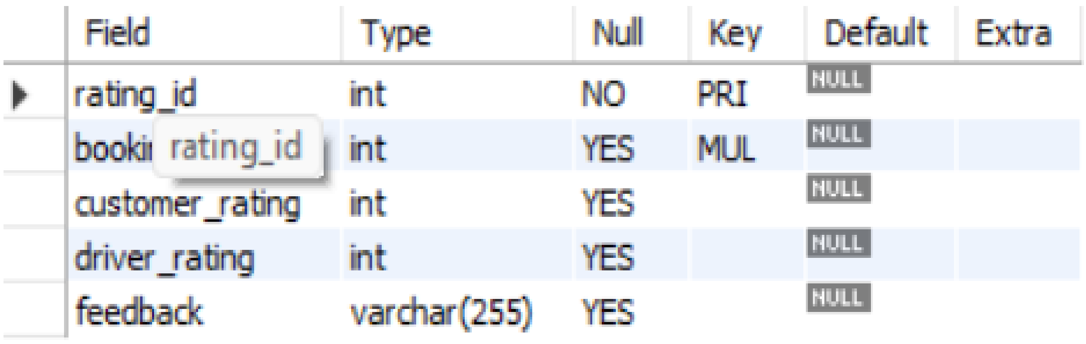


	Field	Type	Null	Key	Default	Extra
▶	payment_id	int	NO	PRI	NULL	
	booking_id	int	YES	MUL	NULL	
	amount	decimal(10,2)	YES		NULL	
	payment_mode	varchar(20)	YES		NULL	
	payment_status	varchar(20)	YES		NULL	

F) Ratings

CREATE TABLE Ratings (rating_id INT PRIMARY KEY, booking_id INT, customer_rating INT,driver_rating INT,feedback VARCHAR(255), FOREIGN KEY (booking_id) REFERENCES Bookings(booking_id));

Desc Ratings;

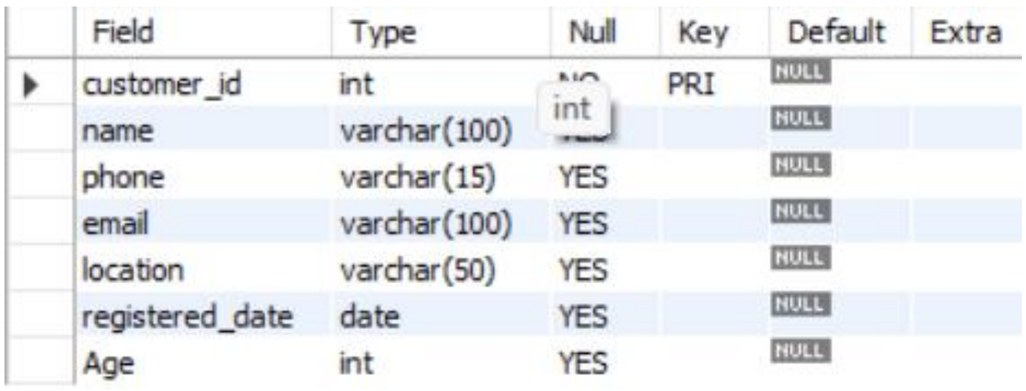


	Field	Type	Null	Key	Default	Extra
▶	rating_id	int	NO	PRI	NULL	
	booki rating_id	int	YES	MUL	NULL	
	customer_rating	int	YES		NULL	
	driver_rating	int	YES		NULL	
	feedback	varchar(255)	YES		NULL	

- **ALTER TABLE:**

  Add column

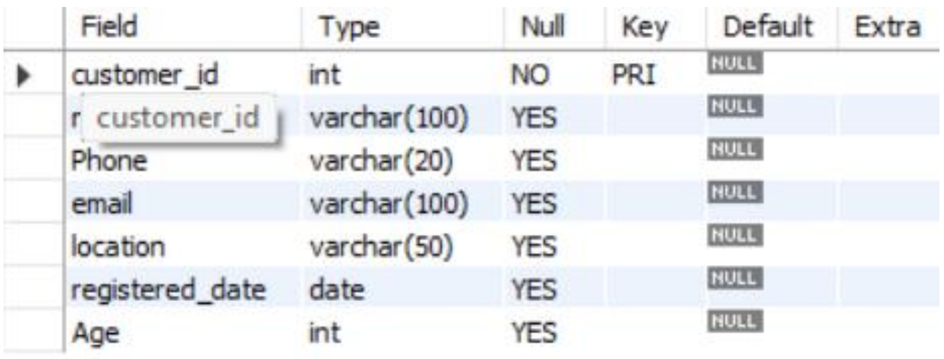
  ALTER TABLE Customers ADD Age INT;



Field	Type	Null	Key	Default	Extra
customer_id	int	NO	PRI	NULL	
name	varchar(100)	[illegible]		NULL	
phone	varchar(15)	YES		NULL	
email	varchar(100)	YES		NULL	
location	varchar(50)	YES		NULL	
registered_date	date	YES		NULL	
Age	int	YES		NULL	

Alter Table: Modify Column

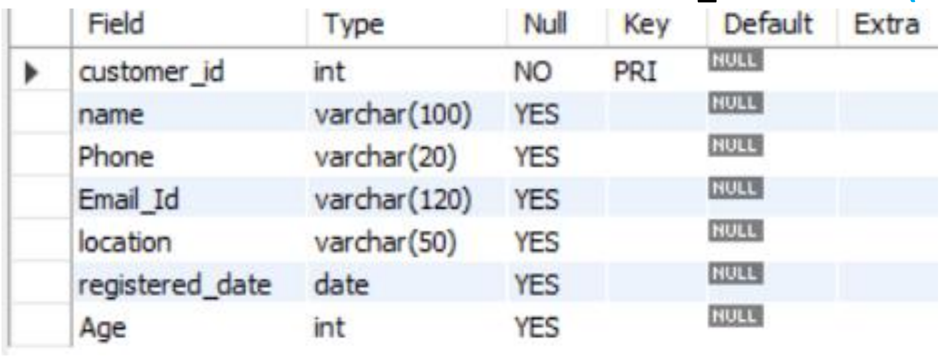
ALTER TABLE Customers MODIFY Phone VARCHAR(20);



Field	Type	Null	Key	Default	Extra
customer_id	int	NO	PRI	NULL	
[illegible]	varchar(100)	YES		NULL	
Phone	varchar(20)	YES		NULL	
email	varchar(100)	YES		NULL	
location	varchar(50)	YES		NULL	
registered_date	date	YES		NULL	
Age	int	YES		NULL	

Alter Table: Change

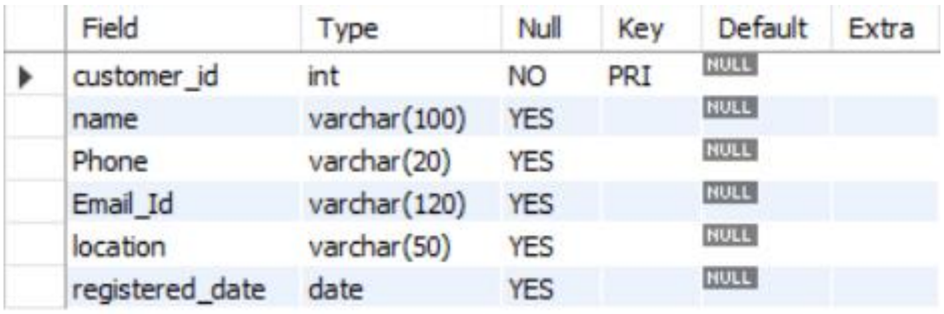
ALTER TABLE Customers CHANGE Email Email_Id VARCHAR(120);



Field	Type	Null	Key	Default	Extra
customer_id	int	NO	PRI	NULL	
name	varchar(100)	YES		NULL	
Phone	varchar(20)	YES		NULL	
Email_Id	varchar(120)	YES		NULL	
location	varchar(50)	YES		NULL	
registered_date	date	YES		NULL	
Age	int	YES		NULL	

Alter Table: Drop

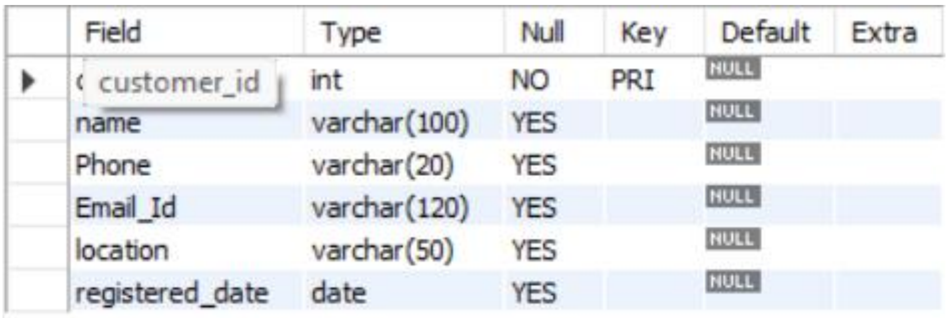
ALTER TABLE Customers DROP Age;



Field	Type	Null	Key	Default	Extra
customer_id	int	NO	PRI	NULL	
name	varchar(100)	YES		NULL	
Phone	varchar(20)	YES		NULL	
Email_Id	varchar(120)	YES		NULL	
location	varchar(50)	YES		NULL	
registered_date	date	YES		NULL	

Alter Table: Rename Table

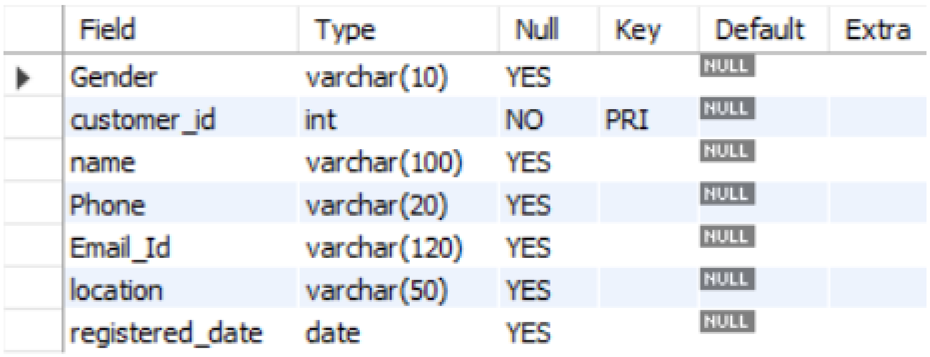
ALTER TABLE Customers RENAME TO Customer_Details;



Field	Type	Null	Key	Default	Extra
customer_id	int	NO	PRI	NULL	
name	varchar(100)	YES		NULL	
Phone	varchar(20)	YES		NULL	
Email_Id	varchar(120)	YES		NULL	
location	varchar(50)	YES		NULL	
registered_date	date	YES		NULL	

Alter Table: Add column at first position

ALTER TABLE Customer_Details ADD Gender VARCHAR(10) FIRST;

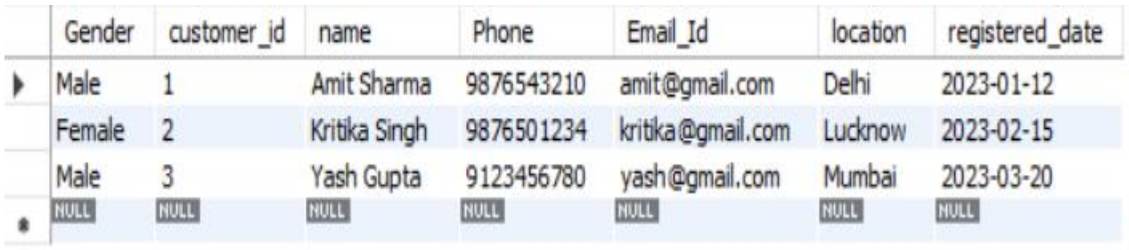


Field	Type	Null	Key	Default	Extra
Gender	varchar(10)	YES		NULL	
customer_id	int	NO	PRI	NULL	
name	varchar(100)	YES		NULL	
Phone	varchar(20)	YES		NULL	
Email_Id	varchar(120)	YES		NULL	
location	varchar(50)	YES		NULL	
registered_date	date	YES		NULL	

- **DATA MANIPULATION LANGUAGE (DML):**

Insert into table:

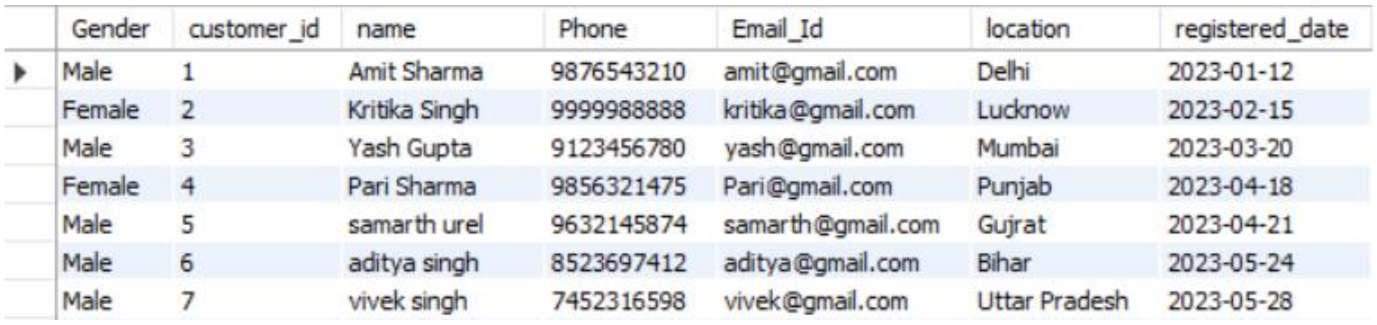
```
INSERT INTO `Customer_Details`(`Gender`, `customer_id`, `name`,
`phone`, `Email_ID`, `location`, `registered_date`)VALUES ('Male',
1, 'Amit Sharma',    '9876543210', 'amit@gmail.com',    'Delhi',
'2023-01-12'), ('Female', 2, 'Kritika Singh',  '9876501234',
'kritika@gmail.com', 'Lucknow', '2023-02-15'), ('Male',   3, 'Yash
Gupta',     '9123456780', 'yash@gmail.com',    'Mumbai', '2023-03-
20');
```



Gender	customer_id	name	Phone	Email_Id	location	registered_date
Male	1	Amit Sharma	9876543210	amit@gmail.com	Delhi	2023-01-12
Female	2	Kritika Singh	9876501234	kritika@gmail.com	Lucknow	2023-02-15
Male	3	Yash Gupta	9123456780	yash@gmail.com	Mumbai	2023-03-20
NULL	NULL	NULL	NULL	NULL	NULL	NULL

Update table: update customers phone number

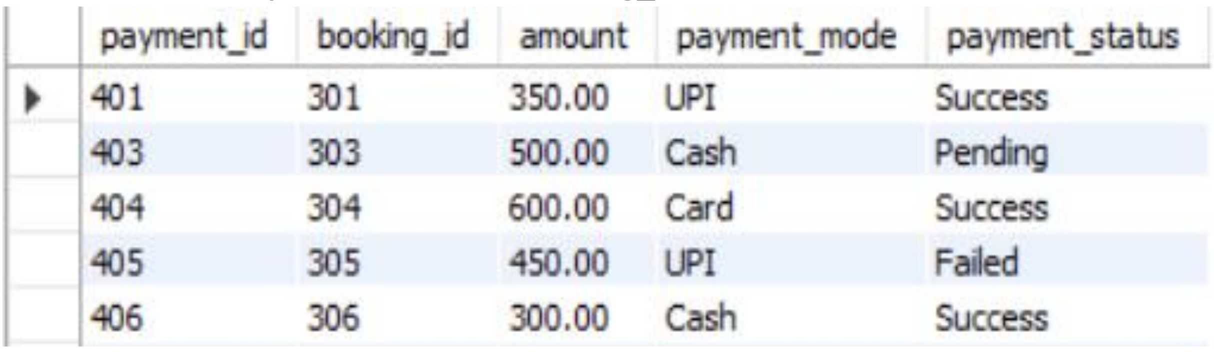
```
Update Customer_Details set phone=9999988888 where Customer_id=2;
```



Gender	customer_id	name	Phone	Email_Id	location	registered_date
Male	1	Amit Sharma	9876543210	amit@gmail.com	Delhi	2023-01-12
Female	2	Kritika Singh	9999988888	kritika@gmail.com	Lucknow	2023-02-15
Male	3	Yash Gupta	9123456780	yash@gmail.com	Mumbai	2023-03-20
Female	4	Pari Sharma	9856321475	Pari@gmail.com	Punjab	2023-04-18
Male	5	samarth urel	9632145874	samarth@gmail.com	Gujrat	2023-04-21
Male	6	aditya singh	8523697412	aditya@gmail.com	Bihar	2023-05-24
Male	7	vivek singh	7452316598	vivek@gmail.com	Uttar Pradesh	2023-05-28

Delete From Table:

```
DELETE FROM Payments WHERE Booking_id = 302;
```



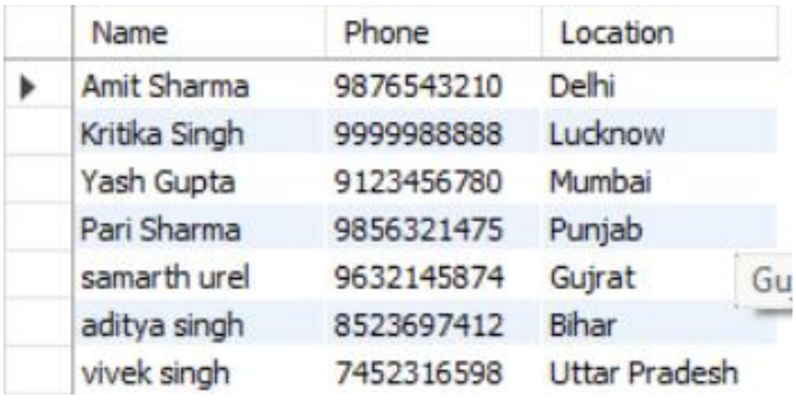
payment_id	booking_id	amount	payment_mode	payment_status
401	301	350.00	UPI	Success
403	303	500.00	Cash	Pending
404	304	600.00	Card	Success
405	305	450.00	UPI	Failed
406	306	300.00	Cash	Success

• DATA QUERY LANGUAGE (DQL):

1) Select:

```
Select * From Drivers;
```

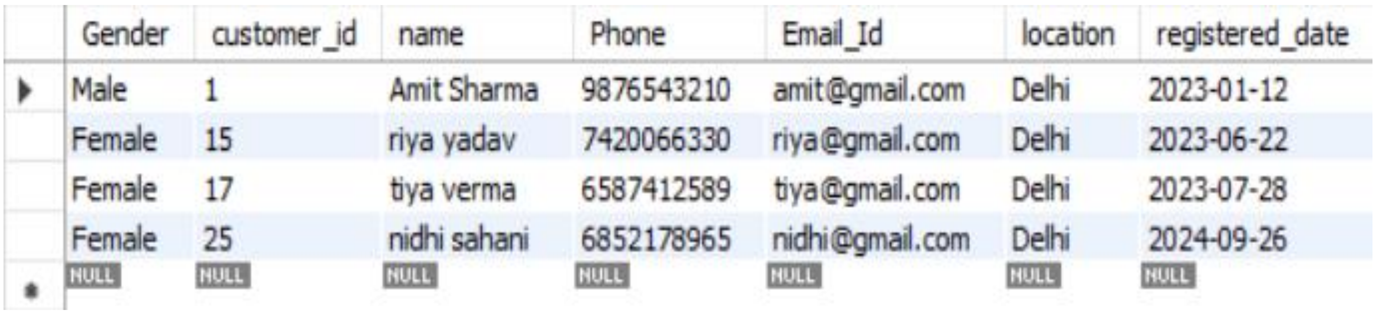
```
SELECT Name, Phone, Location FROM Customer_Details;
```



Name	Phone	Location
Amit Sharma	9876543210	Delhi
Kritika Singh	9999988888	Lucknow
Yash Gupta	9123456780	Mumbai
Pari Sharma	9856321475	Punjab
samarth urel	9632145874	Gujrat
aditya singh	8523697412	Bihar
vivek singh	7452316598	Uttar Pradesh

2) Where:

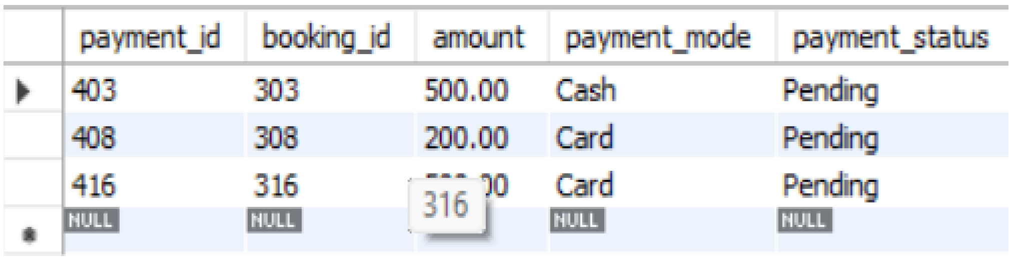
```
SELECT * FROM Customer_Details WHERE Location='Delhi';
```



Gender	customer_id	name	Phone	Email_Id	location	registered_date
Male	1	Amit Sharma	9876543210	amit@gmail.com	Delhi	2023-01-12
Female	15	riya yadav	7420066330	riya@gmail.com	Delhi	2023-06-22
Female	17	tiya verma	6587412589	tiya@gmail.com	Delhi	2023-07-28
Female	25	nidhi sahani	6852178965	nidhi@gmail.com	Delhi	2024-09-26
NULL	NULL	NULL	NULL	NULL	NULL	NULL

3) Where + Comparison Operators : Is Equals to (=)

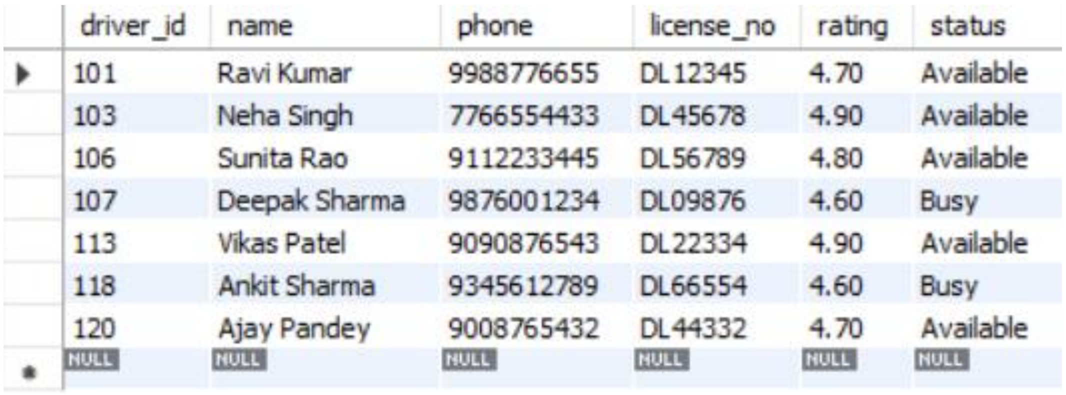
SELECT * FROM Payments WHERE Payment_Status='Pending';



	payment_id	booking_id	amount	payment_mode	payment_status
▶	403	303	500.00	Cash	Pending
	408	308	200.00	Card	Pending
	416	316	[illegible]	Card	Pending
●	NULL	NULL	316	NULL	NULL

Where + Comparison Operators : Greater Than (>)

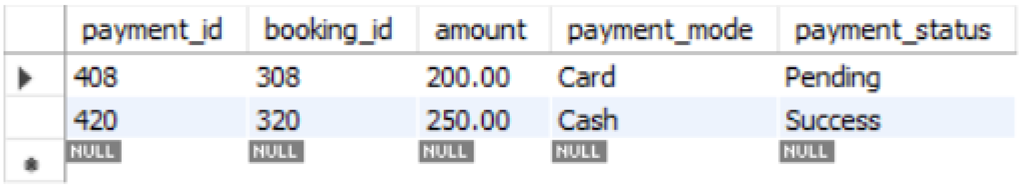
SELECT * FROM Drivers WHERE Rating > 4.5;



	driver_id	name	phone	license_no	rating	status
▶	101	Ravi Kumar	9988776655	DL12345	4.70	Available
	103	Neha Singh	7766554433	DL45678	4.90	Available
	106	Sunita Rao	9112233445	DL56789	4.80	Available
	107	Deepak Sharma	9876001234	DL09876	4.60	Busy
	113	Vikas Patel	9090876543	DL22334	4.90	Available
	118	Ankit Sharma	9345612789	DL66554	4.60	Busy
	120	Ajay Pandey	9008765432	DL44332	4.70	Available
●	NULL	NULL	NULL	NULL	NULL	NULL

Where + Comparison Operators : Less Than (<)

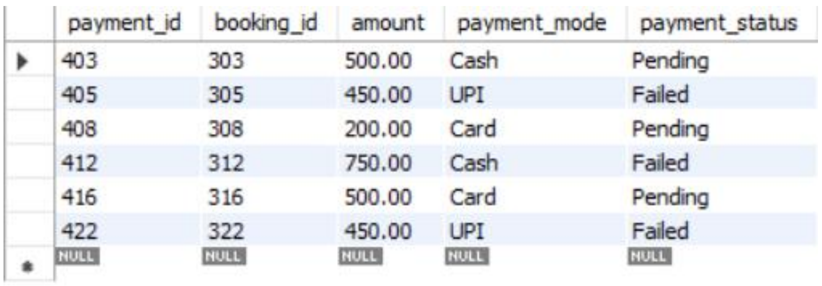
SELECT * FROM Payments WHERE Amount < 300;



	payment_id	booking_id	amount	payment_mode	payment_status
▶	408	308	200.00	Card	Pending
	420	320	250.00	Cash	Success
●	NULL	NULL	NULL	NULL	NULL

Where + Operators : Not Equals To (!=)

SELECT * FROM Payments WHERE Payment_Status != 'Success';

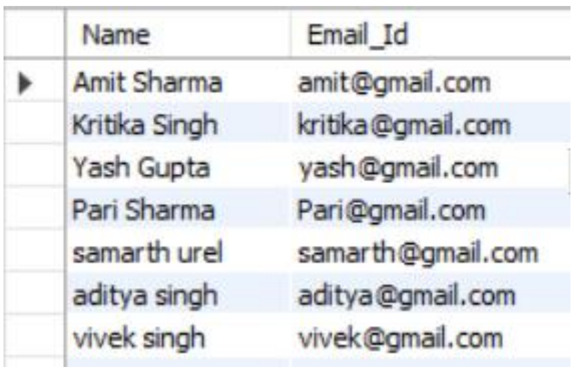


	payment_id	booking_id	amount	payment_mode	payment_status
▶	403	303	500.00	Cash	Pending
	405	305	450.00	UPI	Failed
	408	308	200.00	Card	Pending
	412	312	750.00	Cash	Failed
	416	316	500.00	Card	Pending
	422	322	450.00	UPI	Failed
●	NULL	NULL	NULL	NULL	NULL

- **Logical Operators**

1. **Not Null :**

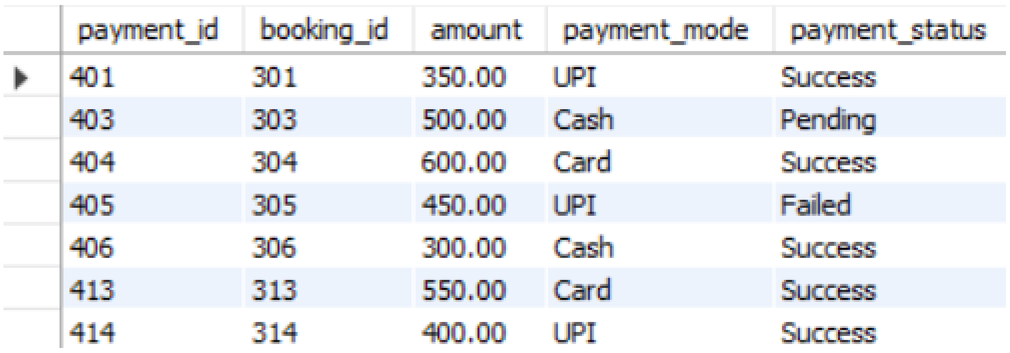
   SELECT Name, Email_Id FROM Customer_Details WHERE Email_Id IS NOT NULL;



	Name	Email_Id
▶	Amit Sharma	amit@gmail.com
	Kritika Singh	kritika@gmail.com
	Yash Gupta	yash@gmail.com
	Pari Sharma	Pari@gmail.com
	samarth urel	samarth@gmail.com
	aditya singh	aditya@gmail.com
	vivek singh	vivek@gmail.com

2. **Between :**

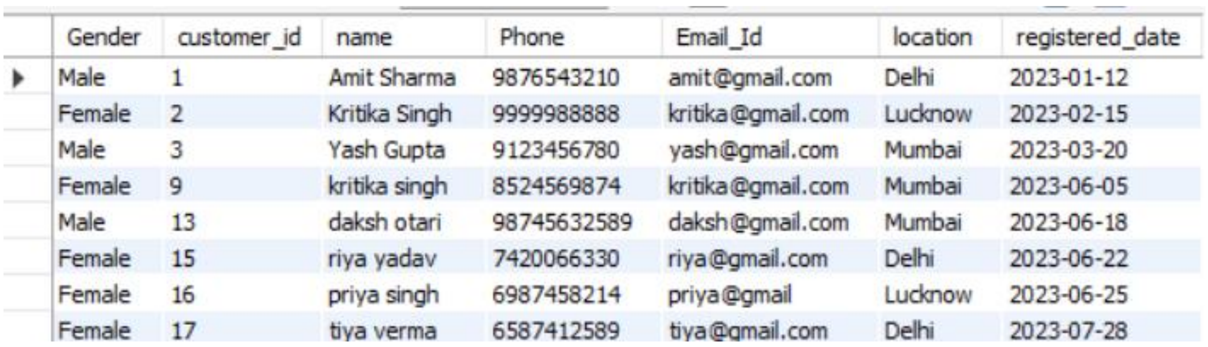
   SELECT *FROM Payments WHERE Amount BETWEEN 300 AND 600;



	payment_id	booking_id	amount	payment_mode	payment_status
▶	401	301	350.00	UPI	Success
	403	303	500.00	Cash	Pending
	404	304	600.00	Card	Success
	405	305	450.00	UPI	Failed
	406	306	300.00	Cash	Success
	413	313	550.00	Card	Success
	414	314	400.00	UPI	Success

3. **In :**

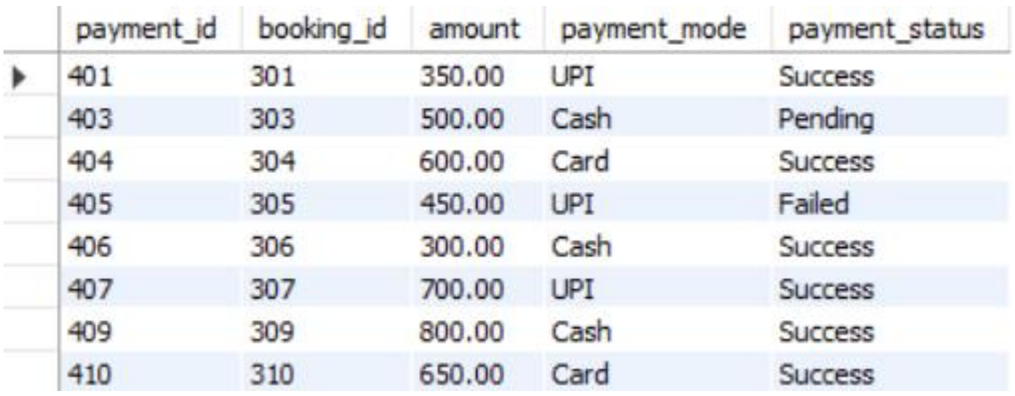
   SELECT *FROM Customers WHERE Location IN ('Delhi', 'Mumbai', 'Lucknow');



	Gender	customer_id	name	Phone	Email_Id	location	registered_date
▶	Male	1	Amit Sharma	9876543210	amit@gmail.com	Delhi	2023-01-12
	Female	2	Kritika Singh	9999988888	kritika@gmail.com	Lucknow	2023-02-15
	Male	3	Yash Gupta	9123456780	yash@gmail.com	Mumbai	2023-03-20
	Female	9	kritika singh	8524569874	kritika@gmail.com	Mumbai	2023-06-05
	Male	13	daksh otari	98745632589	daksh@gmail.com	Mumbai	2023-06-18
	Female	15	riya yadav	7420066330	riya@gmail.com	Delhi	2023-06-22
	Female	16	priya singh	6987458214	priya@gmail	Lucknow	2023-06-25
	Female	17	tiya verma	6587412589	tiya@gmail.com	Delhi	2023-07-28

4. **Any :**

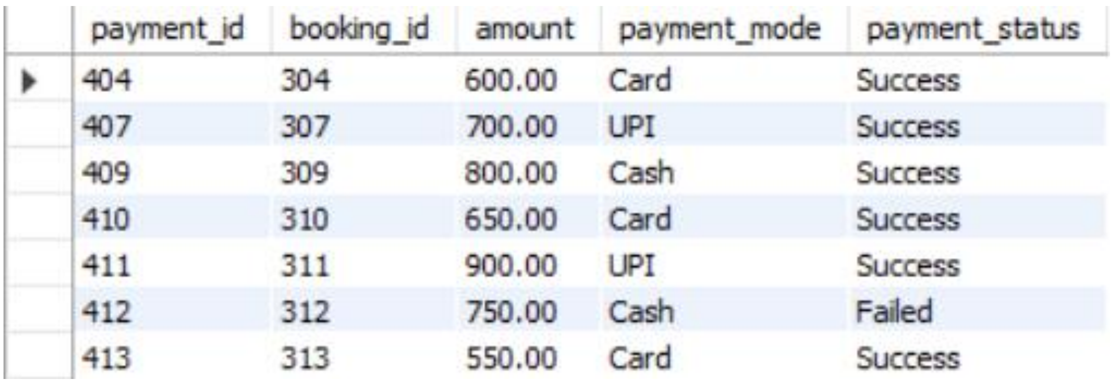
```
SELECT *FROM Payments WHERE Amount > ANY (SELECT Amount
FROM Payments WHERE Payment_Status='Pending');
```



payment_id	booking_id	amount	payment_mode	payment_status
401	301	350.00	UPI	Success
403	303	500.00	Cash	Pending
404	304	600.00	Card	Success
405	305	450.00	UPI	Failed
406	306	300.00	Cash	Success
407	307	700.00	UPI	Success
409	309	800.00	Cash	Success
410	310	650.00	Card	Success

5. **ALL :**

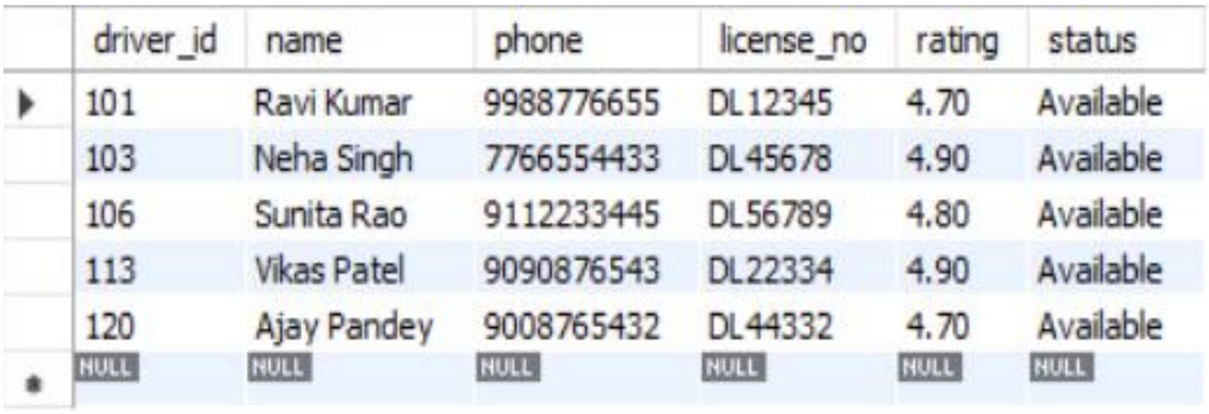
```
SELECT *FROM Payments WHERE Amount > ALL (SELECT Amount
FROM Payments WHERE Payment_Status='Pending');
```



payment_id	booking_id	amount	payment_mode	payment_status
404	304	600.00	Card	Success
407	307	700.00	UPI	Success
409	309	800.00	Cash	Success
410	310	650.00	Card	Success
411	311	900.00	UPI	Success
412	312	750.00	Cash	Failed
413	313	550.00	Card	Success

6. **AND :**

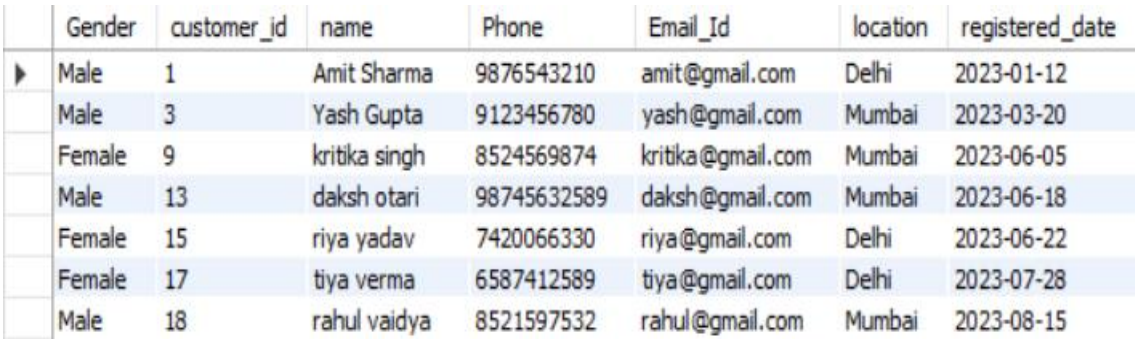
```
SELECT * FROM Drivers WHERE Rating > 4.5 AND Status =
'Available';
```



driver_id	name	phone	license_no	rating	status
101	Ravi Kumar	9988776655	DL12345	4.70	Available
103	Neha Singh	7766554433	DL45678	4.90	Available
106	Sunita Rao	9112233445	DL56789	4.80	Available
113	Vikas Patel	9090876543	DL22334	4.90	Available
120	Ajay Pandey	9008765432	DL44332	4.70	Available
NULL	NULL	NULL	NULL	NULL	NULL

7. **OR :**

SELECT * FROM Customers WHERE Location = 'Delhi' OR Location = 'Mumbai';

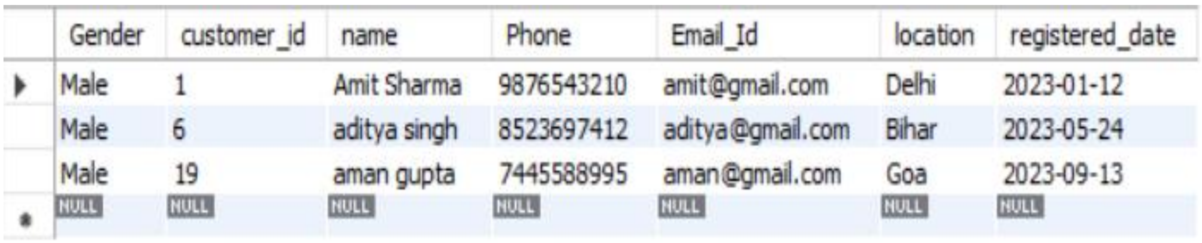


Gender	customer_id	name	Phone	Email_Id	location	registered_date
Male	1	Amit Sharma	9876543210	amit@gmail.com	Delhi	2023-01-12
Male	3	Yash Gupta	9123456780	yash@gmail.com	Mumbai	2023-03-20
Female	9	kritika singh	8524569874	kritika@gmail.com	Mumbai	2023-06-05
Male	13	daksh otari	98745632589	daksh@gmail.com	Mumbai	2023-06-18
Female	15	riya yadav	7420066330	riya@gmail.com	Delhi	2023-06-22
Female	17	tiya verma	6587412589	tiya@gmail.com	Delhi	2023-07-28
Male	18	rahul vaidya	8521597532	rahul@gmail.com	Mumbai	2023-08-15

- **String Functions :**

1. Like Operator :

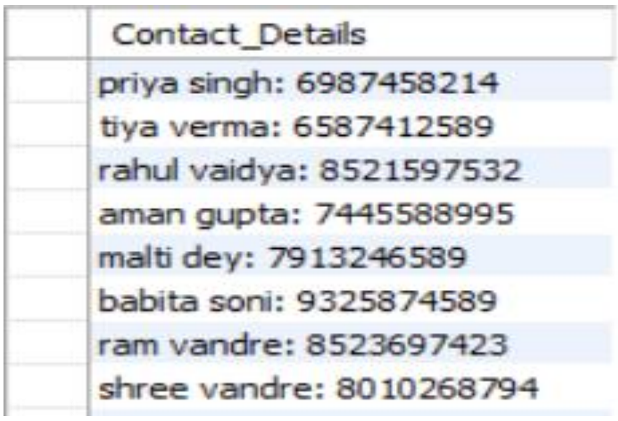
SELECT * FROM Customer_Details WHERE Name LIKE 'a%';



	Gender	customer_id	name	Phone	Email_Id	location	registered_date
▶	Male	1	Amit Sharma	9876543210	amit@gmail.com	Delhi	2023-01-12
	Male	6	aditya singh	8523697412	aditya@gmail.com	Bihar	2023-05-24
	Male	19	aman gupta	7445588995	aman@gmail.com	Goa	2023-09-13
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL

2. Concat Operator :

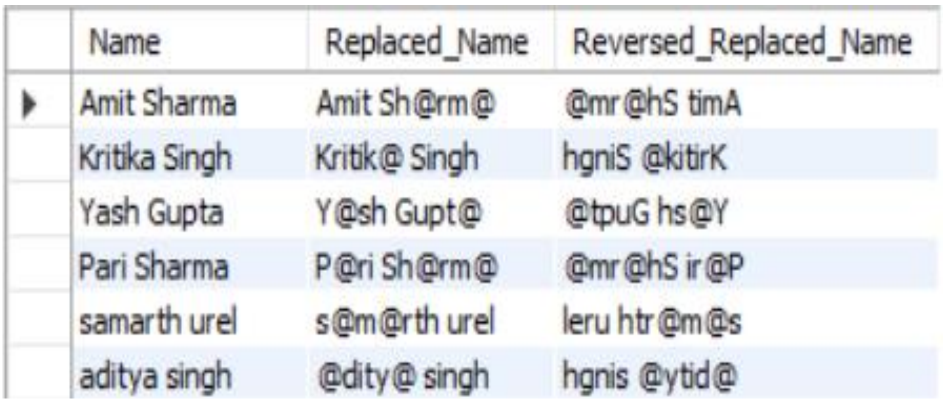
SELECT CONCAT(Name, ': ', Phone) AS Contact_Details FROM Customer_Details ;



	Contact_Details
	priya singh: 6987458214
	tiya verma: 6587412589
	rahul vaidya: 8521597532
	aman gupta: 7445588995
	malti dey: 7913246589
	babita soni: 9325874589
	ram vandre: 8523697423
	shree vandre: 8010268794

3. Replace & Reverse Operator :

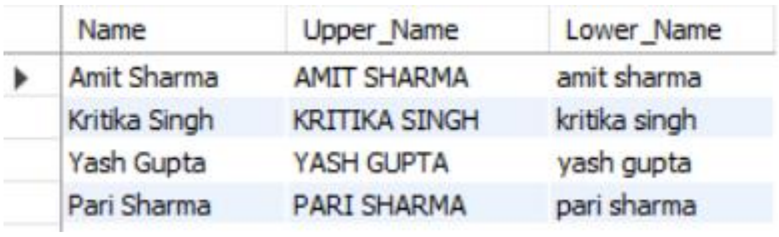
SELECT Name, REPLACE(Name, 'a', '@') AS Replaced_Name,REVERSE(REPLACE(Name, 'a', '@')) AS Reversed_Replaced_Name FROM Customer_Details;



	Name	Replaced_Name	Reversed_Replaced_Name
▶	Amit Sharma	Amit Sh@rm@	@mr@hS timA
	Kritika Singh	Kritik@ Singh	hgniS @kitirK
	Yash Gupta	Y@sh Gupt@	@tpuG hs@Y
	Pari Sharma	P@ri Sh@rm@	@mr@hS ir@P
	samarth urel	s@m@rth urel	leru htr@m@s
	aditya singh	@dity@ singh	hgnis @ytid@

4. Upper & Lower Operator :

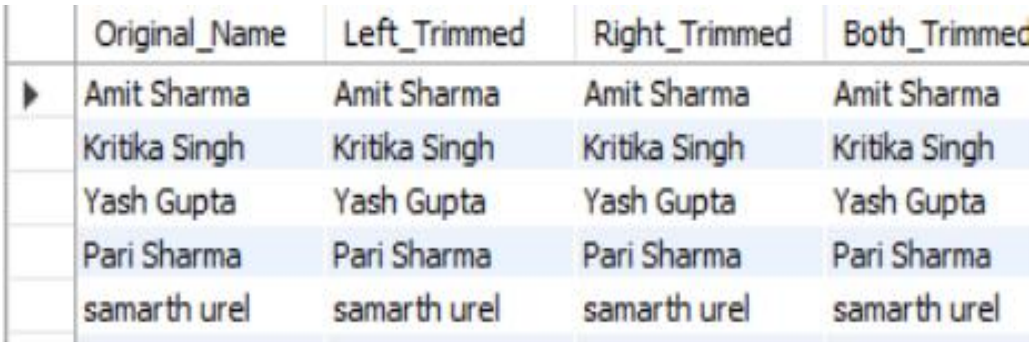
```
SELECT Name, UPPER(Name) AS Upper_Name, LOWER(Name) AS Lower_Name FROM Customer_Details;
```



Name	Upper_Name	Lower_Name
Amit Sharma	AMIT SHARMA	amit sharma
Kritika Singh	KRITIKA SINGH	kritika singh
Yash Gupta	YASH GUPTA	yash gupta
Pari Sharma	PARI SHARMA	pari sharma

- **SubString (Ltrim,Rtrim,Trim):**

```
SELECT Name AS Original_Name, LTRIM(Name) AS Left_Trimmed, RTRIM(Name) AS Right_Trimmed, TRIM(Name) AS Both_Trimmed FROM Customer_Details;
```

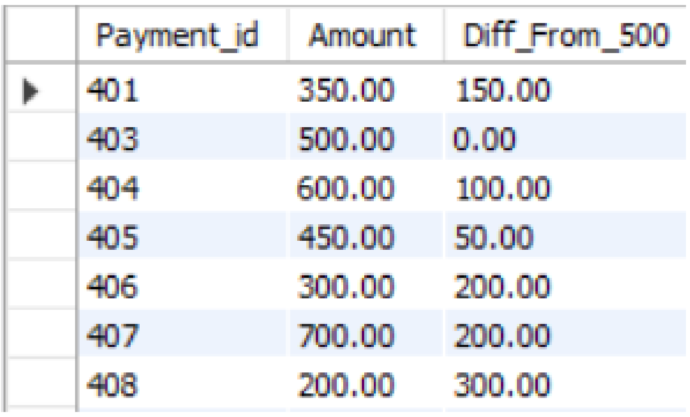


Original_Name	Left_Trimmed	Right_Trimmed	Both_Trimmed
Amit Sharma	Amit Sharma	Amit Sharma	Amit Sharma
Kritika Singh	Kritika Singh	Kritika Singh	Kritika Singh
Yash Gupta	Yash Gupta	Yash Gupta	Yash Gupta
Pari Sharma	Pari Sharma	Pari Sharma	Pari Sharma
samarth urel	samarth urel	samarth urel	samarth urel

- **Mathematical Functions :**

1. Abs() : Absolute Value

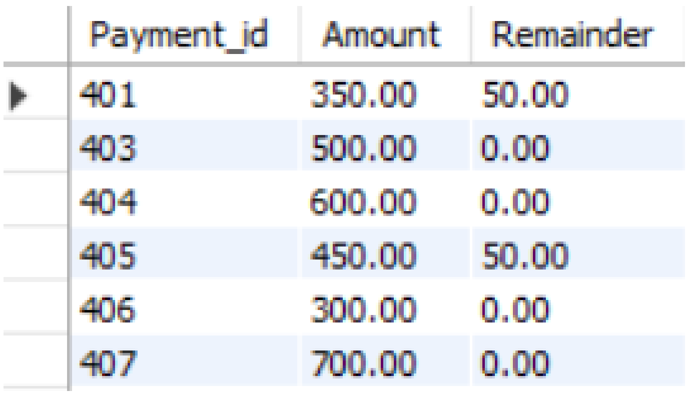
```
SELECT Payment_id, Amount, ABS(Amount - 500) AS Diff_From_500
FROM Payments;
```



Payment_id	Amount	Diff_From_500
401	350.00	150.00
403	500.00	0.00
404	600.00	100.00
405	450.00	50.00
406	300.00	200.00
407	700.00	200.00
408	200.00	300.00

2. Mod() : Remainder of division

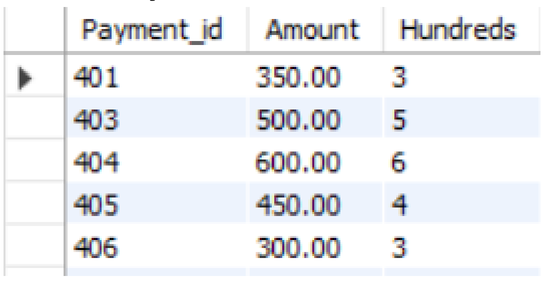
```
SELECT Payment_id, Amount, MOD(Amount, 100) AS Remainder
FROM Payments;
```



Payment_id	Amount	Remainder
401	350.00	50.00
403	500.00	0.00
404	600.00	0.00
405	450.00	50.00
406	300.00	0.00
407	700.00	0.00

3. Floor() : round down

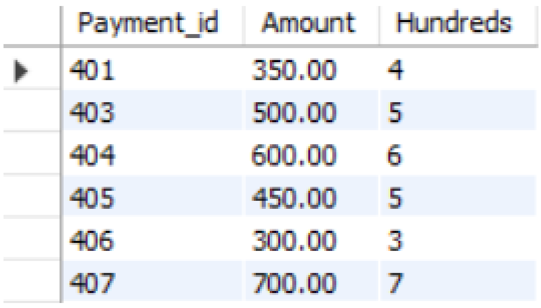
```
SELECT Payment_id, Amount, FLOOR(Amount/100) AS Hundreds
FROM Payments;
```



Payment_id	Amount	Hundreds
401	350.00	3
403	500.00	5
404	600.00	6
405	450.00	4
406	300.00	3

4. Ceil() : round up

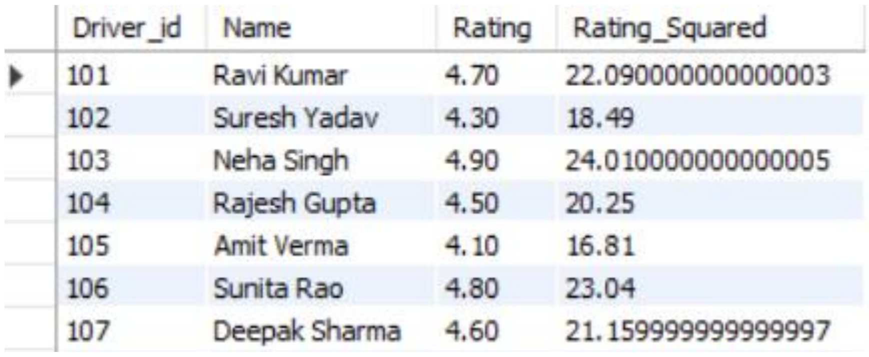
SELECT Payment_id, Amount, CEIL(Amount/100) AS Hundred FROM Payments;



Payment_id	Amount	Hundreds
401	350.00	4
403	500.00	5
404	600.00	6
405	450.00	5
406	300.00	3
407	700.00	7

5. Pow(x,y) : power

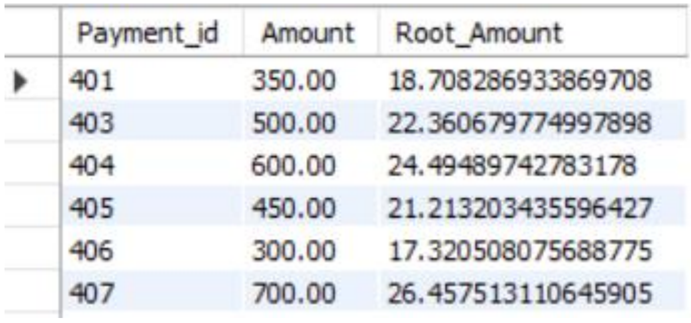
SELECT Driver_id, Name, Rating, POW(Rating, 2) AS Rating_Squared FROM Drivers;



Driver_id	Name	Rating	Rating_Squared
101	Ravi Kumar	4.70	22.090000000000003
102	Suresh Yadav	4.30	18.49
103	Neha Singh	4.90	24.010000000000005
104	Rajesh Gupta	4.50	20.25
105	Amit Verma	4.10	16.81
106	Sunita Rao	4.80	23.04
107	Deepak Sharma	4.60	21.159999999999997

6. Sqrt() : square root

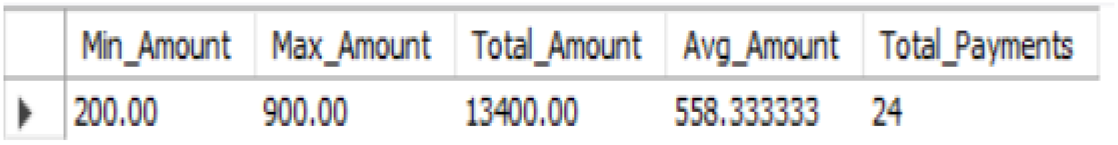
SELECT Payment_id,Amount,SQRT(Amount) AS Root_Amount FROM Payments;



Payment_id	Amount	Root_Amount
401	350.00	18.708286933869708
403	500.00	22.360679774997898
404	600.00	24.49489742783178
405	450.00	21.213203435596427
406	300.00	17.320508075688775
407	700.00	26.457513110645905

- **Aggregate Functions (min, max, sum, avg, count):**

  SELECT MIN(Amount) AS Min_Amount, MAX(Amount) AS Max_Amount, SUM(Amount) AS Total_Amount, AVG(Amount) AS Avg_Amount, COUNT(*) AS Total_Payments FROM Payments;

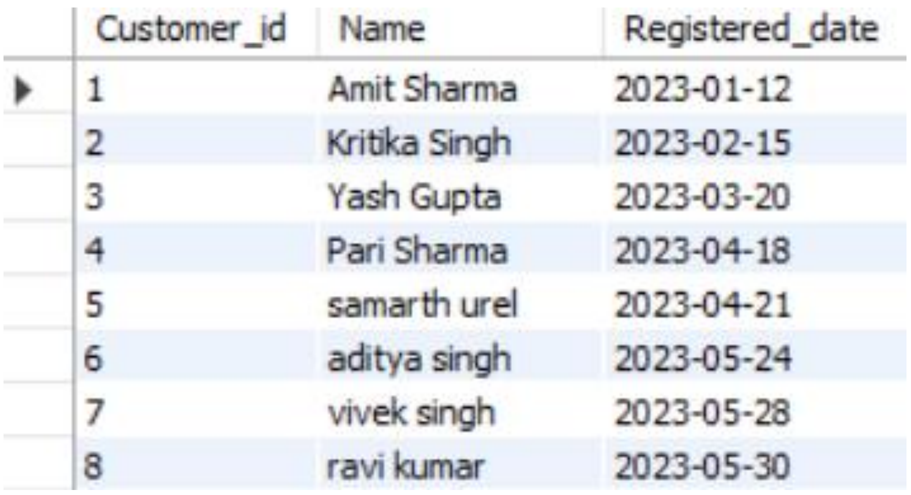


Min_Amount	Max_Amount	Total_Amount	Avg_Amount	Total_Payments
200.00	900.00	13400.00	558.333333	24

- **Date Function :**

  1. Curdate() : current date

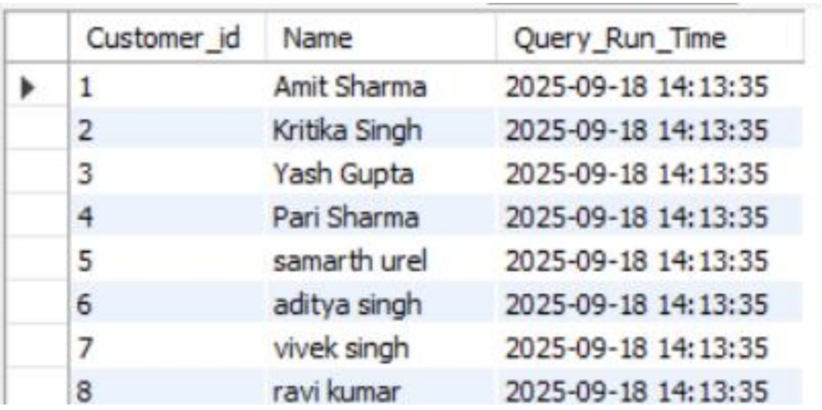
  SELECT Customer_id,Name, Registered_date FROM Customer_Details WHERE Registered_date < CURDATE();



Customer_id	Name	Registered_date
1	Amit Sharma	2023-01-12
2	Kritika Singh	2023-02-15
3	Yash Gupta	2023-03-20
4	Pari Sharma	2023-04-18
5	samarth urel	2023-04-21
6	aditya singh	2023-05-24
7	vivek singh	2023-05-28
8	ravi kumar	2023-05-30

2. Now() : current date and time

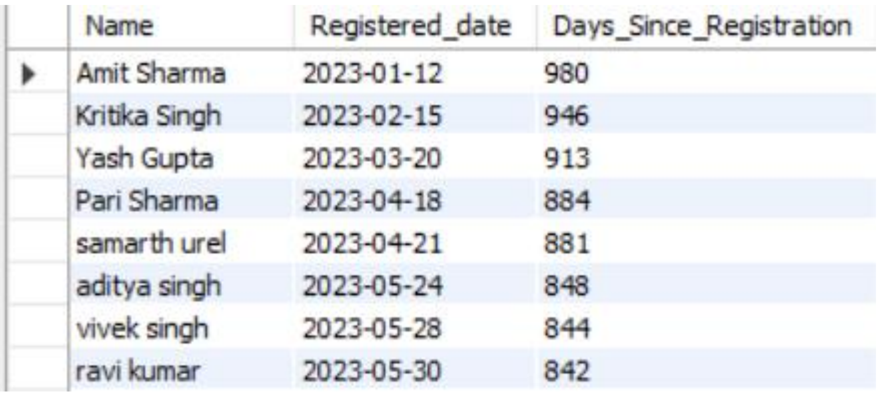
SELECT Customer_id, Name, NOW() AS Query_Run_Time FROM Customer_Details;



Customer_id	Name	Query_Run_Time
1	Amit Sharma	2025-09-18 14:13:35
2	Kritika Singh	2025-09-18 14:13:35
3	Yash Gupta	2025-09-18 14:13:35
4	Pari Sharma	2025-09-18 14:13:35
5	samarth urel	2025-09-18 14:13:35
6	aditya singh	2025-09-18 14:13:35
7	vivek singh	2025-09-18 14:13:35
8	ravi kumar	2025-09-18 14:13:35

3. Datediff() : difference between two dates (in days)

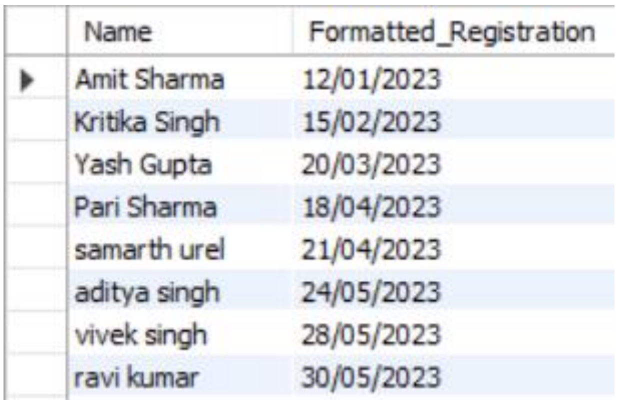
SELECT Name, Registered_date, DATEDIFF(CURDATE(), Registered_date) AS Days_Since_Registration FROM Customer_Details;



Name	Registered_date	Days_Since_Registration
Amit Sharma	2023-01-12	980
Kritika Singh	2023-02-15	946
Yash Gupta	2023-03-20	913
Pari Sharma	2023-04-18	884
samarth urel	2023-04-21	881
aditya singh	2023-05-24	848
vivek singh	2023-05-28	844
ravi kumar	2023-05-30	842

4. Date_format(): format the date into custom style

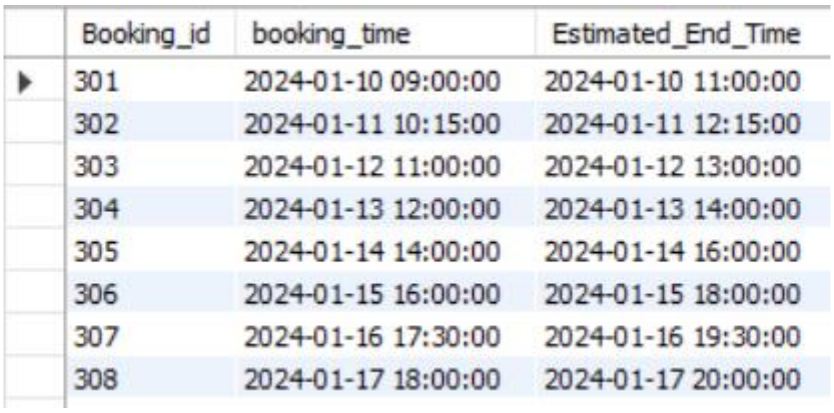
SELECT Name, DATE_FORMAT(Registered_date, '%d/%m/%Y') AS Formatted_Registration FROM Customer_Details;



Name	Formatted_Registration
Amit Sharma	12/01/2023
Kritika Singh	15/02/2023
Yash Gupta	20/03/2023
Pari Sharma	18/04/2023
samarth urel	21/04/2023
aditya singh	24/05/2023
vivek singh	28/05/2023
ravi kumar	30/05/2023

5. Date_add() : add interval to a date

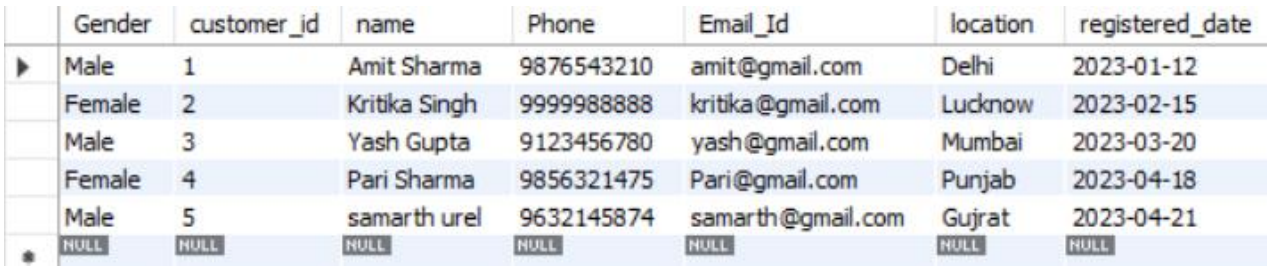
SELECT Booking_id, booking_time, DATE_ADD(booking_time, INTERVAL 2 HOUR) AS Estimated_End_Time FROM Bookings;



Booking_id	booking_time	Estimated_End_Time
301	2024-01-10 09:00:00	2024-01-10 11:00:00
302	2024-01-11 10:15:00	2024-01-11 12:15:00
303	2024-01-12 11:00:00	2024-01-12 13:00:00
304	2024-01-13 12:00:00	2024-01-13 14:00:00
305	2024-01-14 14:00:00	2024-01-14 16:00:00
306	2024-01-15 16:00:00	2024-01-15 18:00:00
307	2024-01-16 17:30:00	2024-01-16 19:30:00
308	2024-01-17 18:00:00	2024-01-17 20:00:00

• Limit Query :

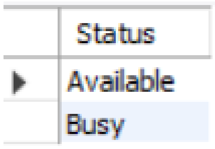
SELECT * FROM Customer_Details LIMIT 5;



Gender	customer_id	name	Phone	Email_Id	location	registered_date
Male	1	Amit Sharma	9876543210	amit@gmail.com	Delhi	2023-01-12
Female	2	Kritika Singh	9999988888	kritika@gmail.com	Lucknow	2023-02-15
Male	3	Yash Gupta	9123456780	yash@gmail.com	Mumbai	2023-03-20
Female	4	Pari Sharma	9856321475	Pari@gmail.com	Punjab	2023-04-18
Male	5	samarth urel	9632145874	samarth@gmail.com	Gujrat	2023-04-21
NULL	NULL	NULL	NULL	NULL	NULL	NULL

• Distinct Query:

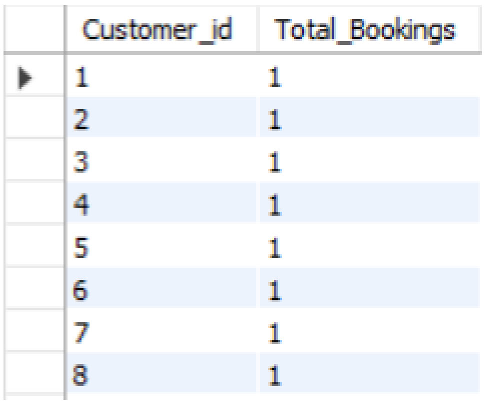
SELECT DISTINCT Status FROM Drivers;



Status
Available
Busy

• Group By Clauses :

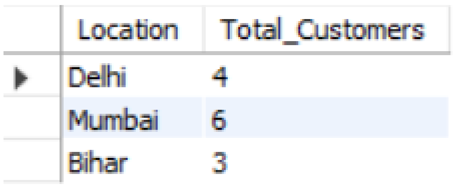
SELECT Customer_id, COUNT(*) AS Total_Bookings
FROM Bookings GROUP BY Customer_id;



Customer_id	Total_Bookings
1	1
2	1
3	1
4	1
5	1
6	1
7	1
8	1

• Having Clause:

SELECT Location, COUNT(*) AS Total_Customers
FROM Customer_Details GROUP BY Location HAVING COUNT(*) > 2;

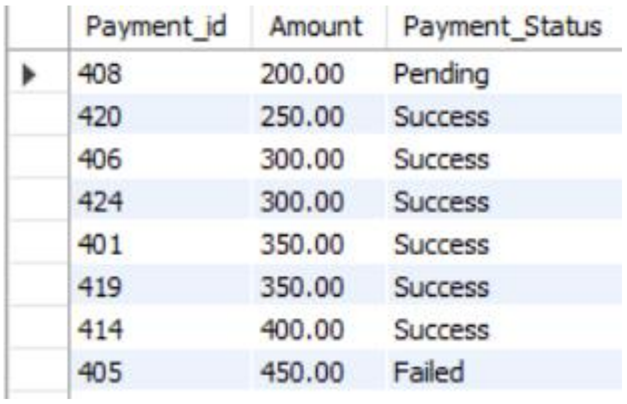


Location	Total_Customers
Delhi	4
Mumbai	6
Bihar	3

• Order By Clause:

1. Ascending

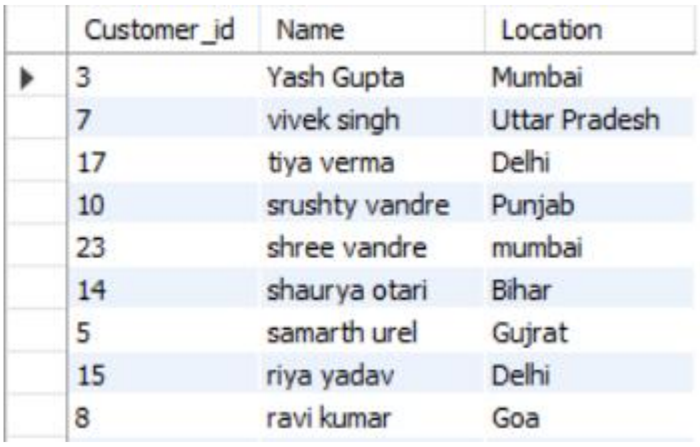
SELECT Payment_id, Amount, Payment_Status
FROM Payments ORDER BY Amount ASC;



Payment_id	Amount	Payment_Status
408	200.00	Pending
420	250.00	Success
406	300.00	Success
424	300.00	Success
401	350.00	Success
419	350.00	Success
414	400.00	Success
405	450.00	Failed

2. Descending :

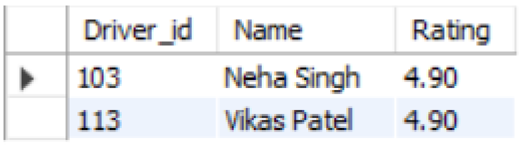
SELECT Customer_id, Name, Location
FROM Customer_Details ORDER BY Name DESC;



Customer_id	Name	Location
3	Yash Gupta	Mumbai
7	vivek singh	Uttar Pradesh
17	tiya verma	Delhi
10	srushty vandre	Punjab
23	shree vandre	mumbai
14	shaurya otari	Bihar
5	samarth urel	Gujrat
15	riya yadav	Delhi
8	ravi kumar	Goa

- **Sub-Query:**

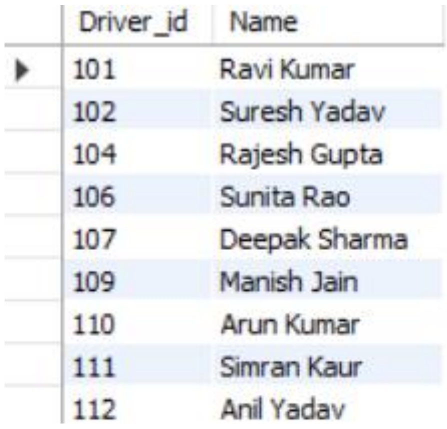
  SELECT Driver_id, Name, Rating FROM Drivers WHERE Rating = (SELECT MAX(Rating) FROM Drivers);



Driver_id	Name	Rating
103	Neha Singh	4.90
113	Vikas Patel	4.90

- **Multi-row subquery:**

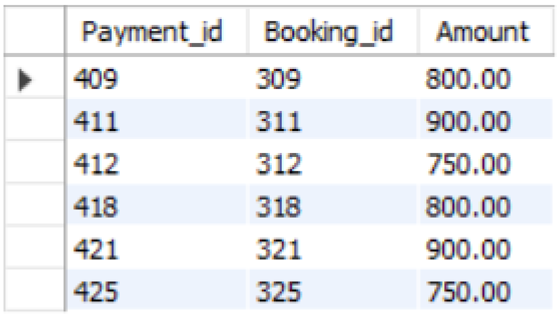
  SELECT Driver_id, Name FROM Drivers WHERE Driver_id IN (SELECT Driver_id FROM Bookings WHERE Status = 'Completed');



Driver_id	Name
101	Ravi Kumar
102	Suresh Yadav
104	Rajesh Gupta
106	Sunita Rao
107	Deepak Sharma
109	Manish Jain
110	Arun Kumar
111	Simran Kaur
112	Anil Yadav

- **Multi-column subquery:**

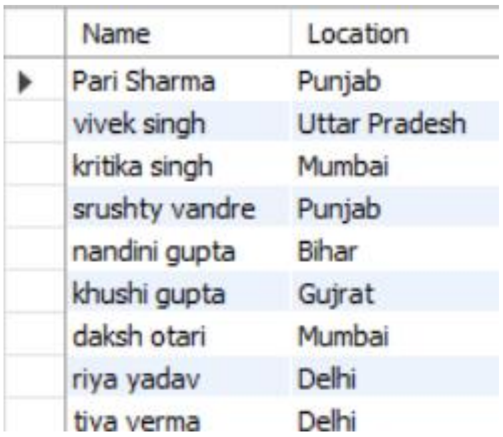
  SELECT Payment_id, Booking_id, Amount
  FROM Payments
  WHERE (Booking_id, Amount) IN (SELECT Booking_id, Amount FROM Payments WHERE Amount > 700);



Payment_id	Booking_id	Amount
409	309	800.00
411	311	900.00
412	312	750.00
418	318	800.00
421	321	900.00
425	325	750.00

- **Multi table subquery:**

  SELECT Name, Location FROM Customer_Details WHERE Customer_id IN (SELECT Customer_id FROM Bookings WHERE Booking_id IN (SELECT Booking_id FROM Payments WHERE Amount > 500));

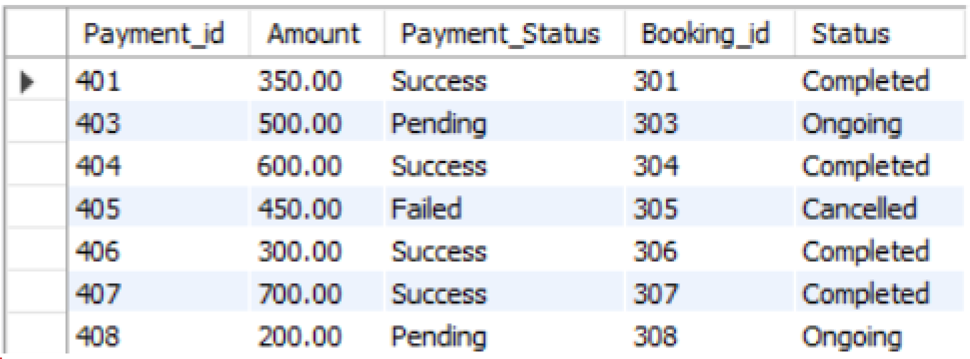


Name	Location
Pari Sharma	Punjab
vivek singh	Uttar Pradesh
kritika singh	Mumbai
srushty vandre	Punjab
nandini gupta	Bihar
khushi gupta	Gujrat
daksh otari	Mumbai
riya yadav	Delhi
tiya verma	Delhi

- **Joins:**

1. Inner join: (only matching rows)

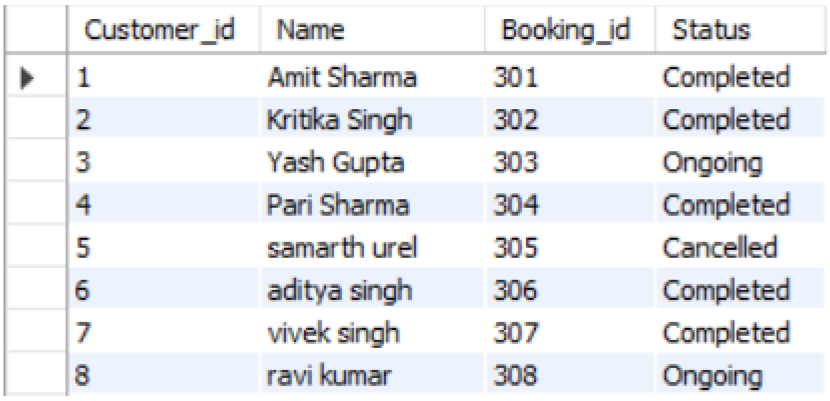
SELECT p.Payment_id, p.Amount, p.Payment_Status, b.Booking_id, b.Status FROM Payments p INNER JOIN Bookings b ON p.Booking_id = b.Booking_id;



Payment_id	Amount	Payment_Status	Booking_id	Status
401	350.00	Success	301	Completed
403	500.00	Pending	303	Ongoing
404	600.00	Success	304	Completed
405	450.00	Failed	305	Cancelled
406	300.00	Success	306	Completed
407	700.00	Success	307	Completed
408	200.00	Pending	308	Ongoing

2.Left Join : (all rows from left,matching with right)

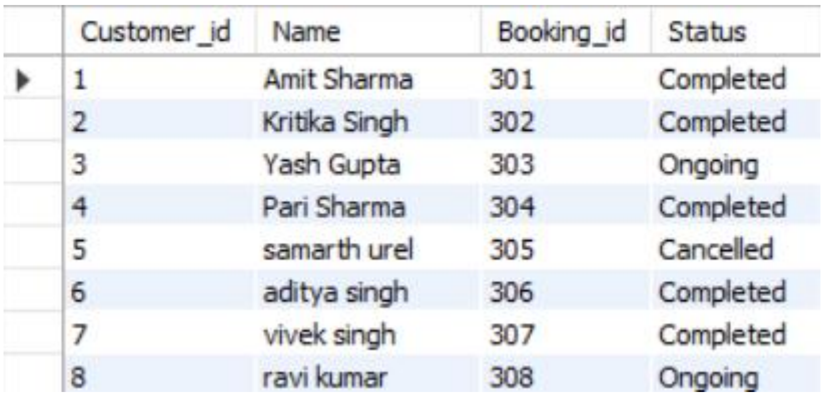
SELECT c.Customer_id, c.Name, b.Booking_id, b.Status FROM Customer_Details c LEFT JOIN Bookings b ON c.Customer_id = b.Customer_id;



Customer_id	Name	Booking_id	Status
1	Amit Sharma	301	Completed
2	Kritika Singh	302	Completed
3	Yash Gupta	303	Ongoing
4	Pari Sharma	304	Completed
5	samarth urel	305	Cancelled
6	aditya singh	306	Completed
7	vivek singh	307	Completed
8	ravi kumar	308	Ongoing

3. Full outer join : (all rows from both,using union)

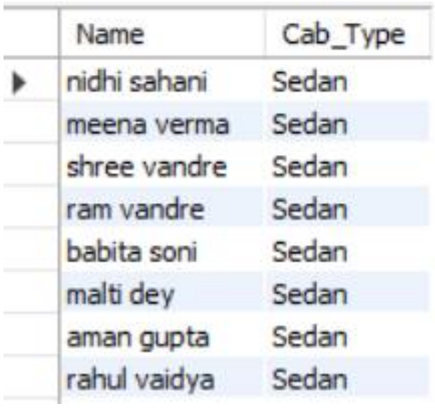
SELECT c.Customer_id, c.Name, b.Booking_id, b.Status FROM Customer_Details c LEFT JOIN Bookings b ON c.Customer_id = b.Customer_id UNION SELECT c.Customer_id, c.Name, b.Booking_id, b.Status FROM Customer_Details c RIGHT JOIN Bookings b ON c.Customer_id = b.Customer_id;



Customer_id	Name	Booking_id	Status
1	Amit Sharma	301	Completed
2	Kritika Singh	302	Completed
3	Yash Gupta	303	Ongoing
4	Pari Sharma	304	Completed
5	samarth urel	305	Cancelled
6	aditya singh	306	Completed
7	vivek singh	307	Completed
8	ravi kumar	308	Ongoing

4. Cross join: (cartesian product, all combinations)

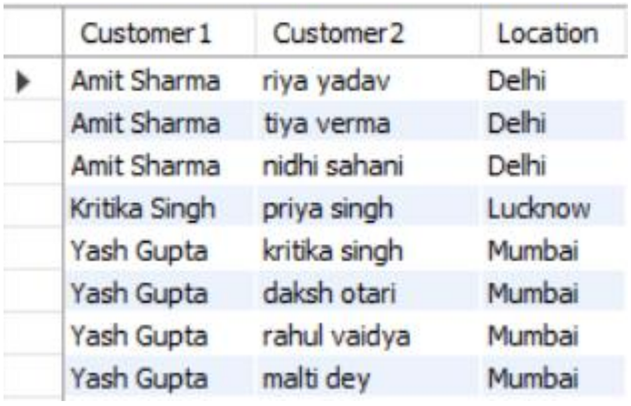
SELECT c.Name, cb.Cab_Type FROM Customer_Details c CROSS JOIN Cabs cb;



Name	Cab_Type
nidhi sahani	Sedan
meena verma	Sedan
shree vandre	Sedan
ram vandre	Sedan
babita soni	Sedan
malti dey	Sedan
aman gupta	Sedan
rahul vaidya	Sedan

5. Self join :

SELECT c1.Name AS Customer1, c2.Name AS Customer2, c1.Location FROM Customer_Details c1 INNER JOIN Customer_Details c2 ON c1.Location = c2.Location AND c1.Customer_id < c2.Customer_id;

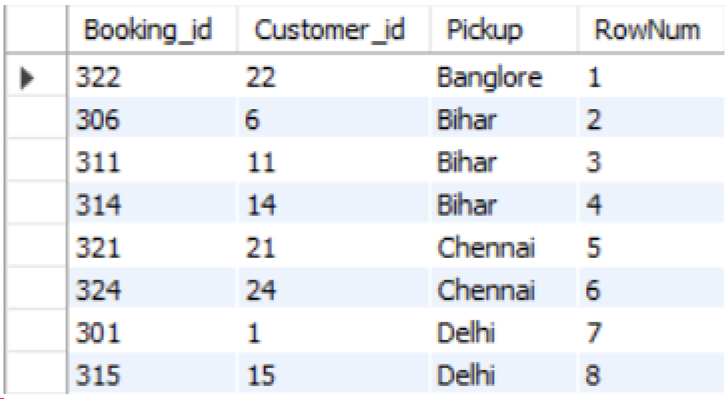


Customer1	Customer2	Location
Amit Sharma	riya yadav	Delhi
Amit Sharma	tiya verma	Delhi
Amit Sharma	nidhi sahani	Delhi
Kritika Singh	priya singh	Lucknow
Yash Gupta	kritika singh	Mumbai
Yash Gupta	daksh otari	Mumbai
Yash Gupta	rahul vaidya	Mumbai
Yash Gupta	malti dey	Mumbai

• Windows :

1. Row Number () : Without Partition

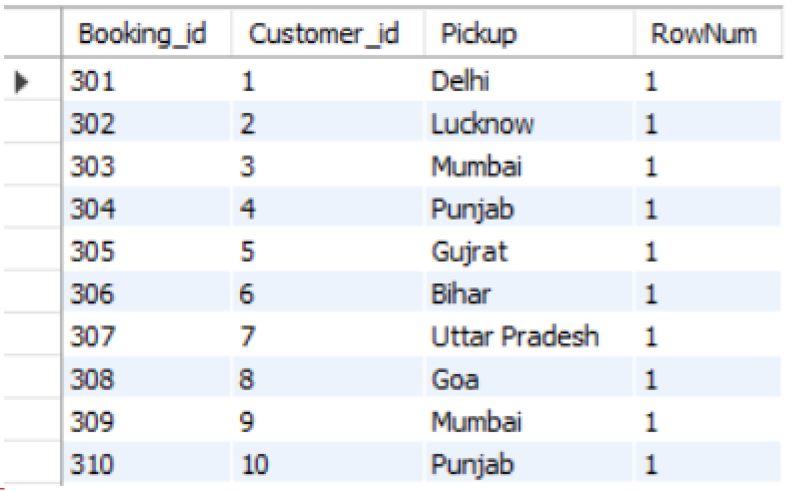
SELECT Booking_id, Customer_id, Pickup, ROW_NUMBER() OVER (ORDER BY Pickup) AS RowNum FROM Bookings;



Booking_id	Customer_id	Pickup	RowNum
322	22	Banglore	1
306	6	Bihar	2
311	11	Bihar	3
314	14	Bihar	4
321	21	Chennai	5
324	24	Chennai	6
301	1	Delhi	7
315	15	Delhi	8

2. Row_Number () : With partition

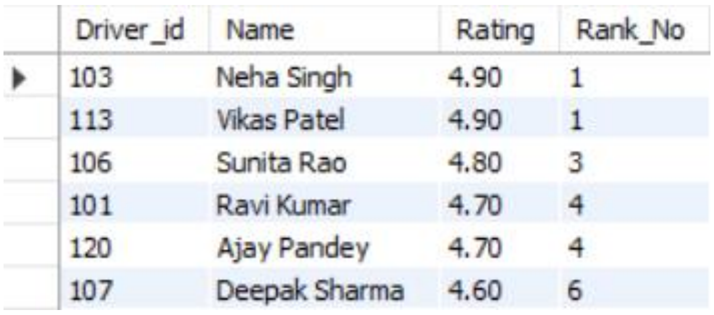
SELECT Booking_id, Customer_id, Pickup, ROW_NUMBER() OVER (PARTITION BY Customer_id ORDER BY Pickup) AS RowNum FROM Bookings;



Booking_id	Customer_id	Pickup	RowNum
301	1	Delhi	1
302	2	Lucknow	1
303	3	Mumbai	1
304	4	Punjab	1
305	5	Gujrat	1
306	6	Bihar	1
307	7	Uttar Pradesh	1
308	8	Goa	1
309	9	Mumbai	1
310	10	Punjab	1

3. Rank () : Without Partition

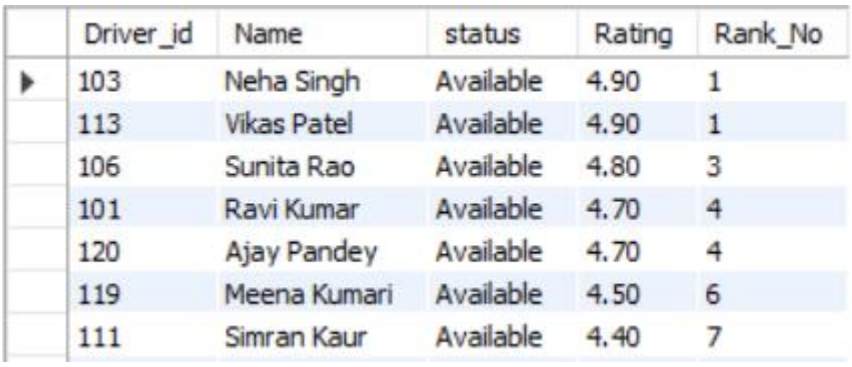
SELECT Driver_id, Name, Rating, RANK() OVER (ORDER BY Rating DESC) AS Rank_No FROM Drivers;



Driver_id	Name	Rating	Rank_No
103	Neha Singh	4.90	1
113	Vikas Patel	4.90	1
106	Sunita Rao	4.80	3
101	Ravi Kumar	4.70	4
120	Ajay Pandey	4.70	4
107	Deepak Sharma	4.60	6

4.Rank () : With Partition

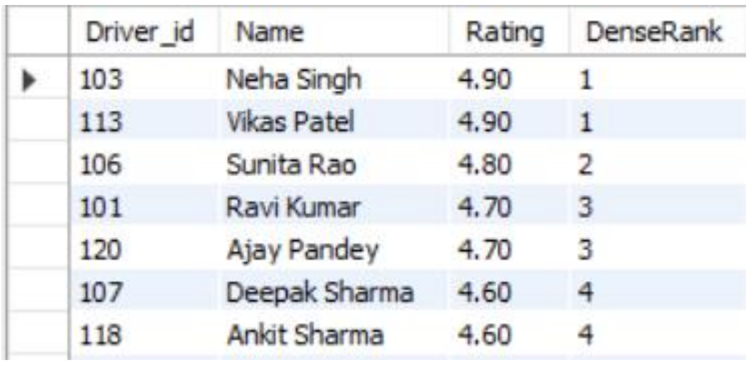
```
SELECT Driver_id, Name, status, Rating, RANK() OVER (PARTITION BY status ORDER BY Rating DESC) AS Rank_No FROM Drivers;
```



Driver_id	Name	status	Rating	Rank_No
103	Neha Singh	Available	4.90	1
113	Vikas Patel	Available	4.90	1
106	Sunita Rao	Available	4.80	3
101	Ravi Kumar	Available	4.70	4
120	Ajay Pandey	Available	4.70	4
119	Meena Kumari	Available	4.50	6
111	Simran Kaur	Available	4.40	7

5.Dense Rank (): Without Partition

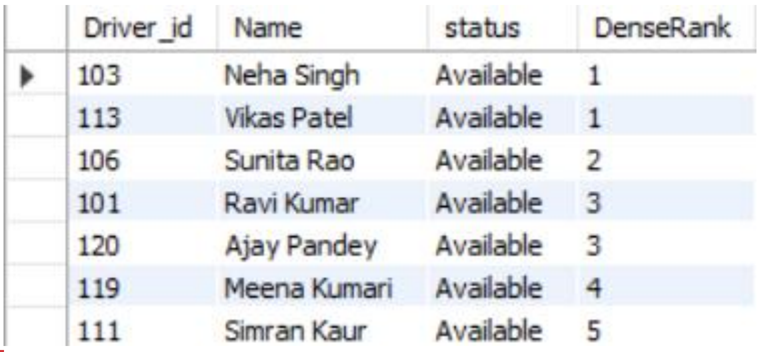
```
SELECT Driver_id,Name, Rating, DENSE_RANK() OVER (ORDER BY Rating DESC) AS DenseRank FROM Drivers;
```



Driver_id	Name	Rating	DenseRank
103	Neha Singh	4.90	1
113	Vikas Patel	4.90	1
106	Sunita Rao	4.80	2
101	Ravi Kumar	4.70	3
120	Ajay Pandey	4.70	3
107	Deepak Sharma	4.60	4
118	Ankit Sharma	4.60	4

6. Dense Rank (): With Partition

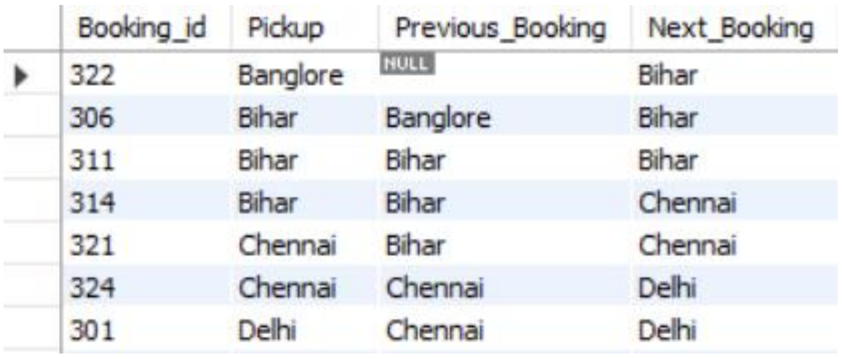
```
SELECT Driver_id,Name, status,  DENSE_RANK() OVER (PARTITION BY Status ORDER BY Rating DESC) AS DenseRank FROM Drivers;
```



Driver_id	Name	status	DenseRank
103	Neha Singh	Available	1
113	Vikas Patel	Available	1
106	Sunita Rao	Available	2
101	Ravi Kumar	Available	3
120	Ajay Pandey	Available	3
119	Meena Kumari	Available	4
111	Simran Kaur	Available	5

8. Lag & Lead :

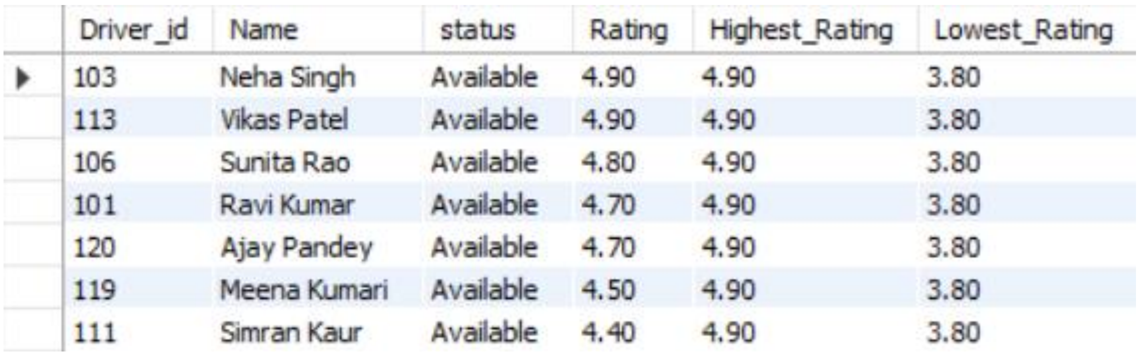
SELECT Booking_id, Pickup, LAG(Pickup, 1) OVER (ORDER BY Pickup) AS Previous_Booking, LEAD(Pickup, 1) OVER (ORDER BY Pickup) AS Next_Booking FROM Bookings;



Booking_id	Pickup	Previous_Booking	Next_Booking
322	Banglore	NULL	Bihar
306	Bihar	Banglore	Bihar
311	Bihar	Bihar	Bihar
314	Bihar	Bihar	Chennai
321	Chennai	Bihar	Chennai
324	Chennai	Chennai	Delhi
301	Delhi	Chennai	Delhi

9. First Value():

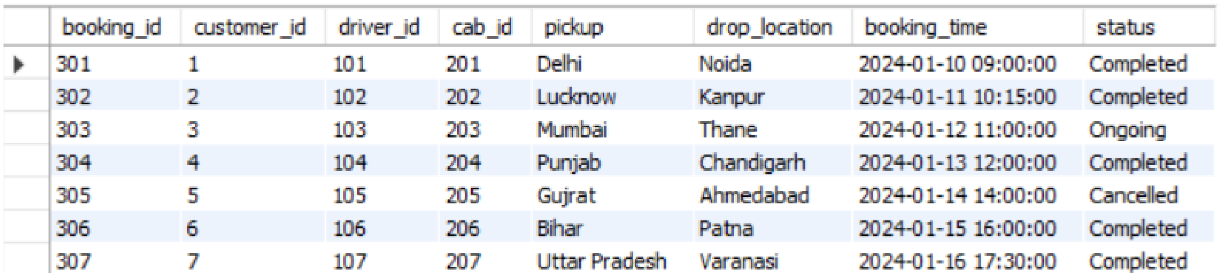
SELECT Driver_id,Name, status, Rating, FIRST_VALUE(Rating) OVER (PARTITION BY status ORDER BY Rating DESC) AS Highest_Rating, LAST_VALUE(Rating) OVER (PARTITION BY status ORDER BY Rating DESC ROWS BETWEEN UNBOUNDED PRECEDING AND UNBOUNDED FOLLOWING) AS Lowest_Rating FROM Drivers;



Driver_id	Name	status	Rating	Highest_Rating	Lowest_Rating
103	Neha Singh	Available	4.90	4.90	3.80
113	Vikas Patel	Available	4.90	4.90	3.80
106	Sunita Rao	Available	4.80	4.90	3.80
101	Ravi Kumar	Available	4.70	4.90	3.80
120	Ajay Pandey	Available	4.70	4.90	3.80
119	Meena Kumari	Available	4.50	4.90	3.80
111	Simran Kaur	Available	4.40	4.90	3.80

9. Nth Value():

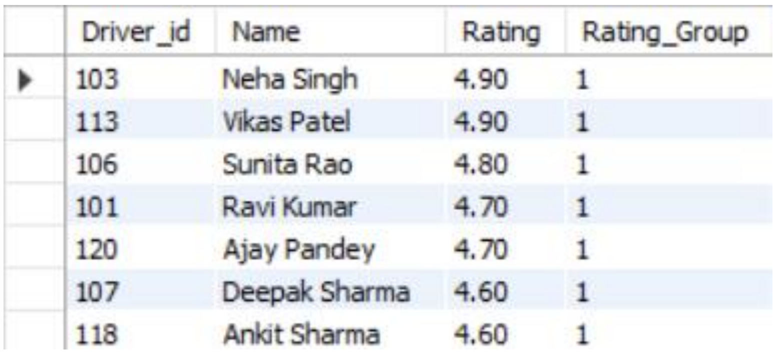
SELECT Customer_id, Booking_id, Pickup_Time,
NTH_VALUE(Pickup_Time, 2) OVER (PARTITION BY Customer_id
ORDER BY Pickup_Time ROWS BETWEEN UNBOUNDED PRECEDING
AND UNBOUNDED FOLLOWING) AS Second_Booking FROM
Bookings;



booking_id	customer_id	driver_id	cab_id	pickup	drop_location	booking_time	status
301	1	101	201	Delhi	Noida	2024-01-10 09:00:00	Completed
302	2	102	202	Lucknow	Kanpur	2024-01-11 10:15:00	Completed
303	3	103	203	Mumbai	Thane	2024-01-12 11:00:00	Ongoing
304	4	104	204	Punjab	Chandigarh	2024-01-13 12:00:00	Completed
305	5	105	205	Gujrat	Ahmedabad	2024-01-14 14:00:00	Cancelled
306	6	106	206	Bihar	Patna	2024-01-15 16:00:00	Completed
307	7	107	207	Uttar Pradesh	Varanasi	2024-01-16 17:30:00	Completed

10. Ntile():

SELECT Driver_id, Driver_Name, Rating, NTILE(3) OVER (ORDER BY
Rating DESC) AS Rating_Group FROM Drivers;



Driver_id	Name	Rating	Rating_Group
103	Neha Singh	4.90	1
113	Vikas Patel	4.90	1
106	Sunita Rao	4.80	1
101	Ravi Kumar	4.70	1
120	Ajay Pandey	4.70	1
107	Deepak Sharma	4.60	1
118	Ankit Sharma	4.60	1