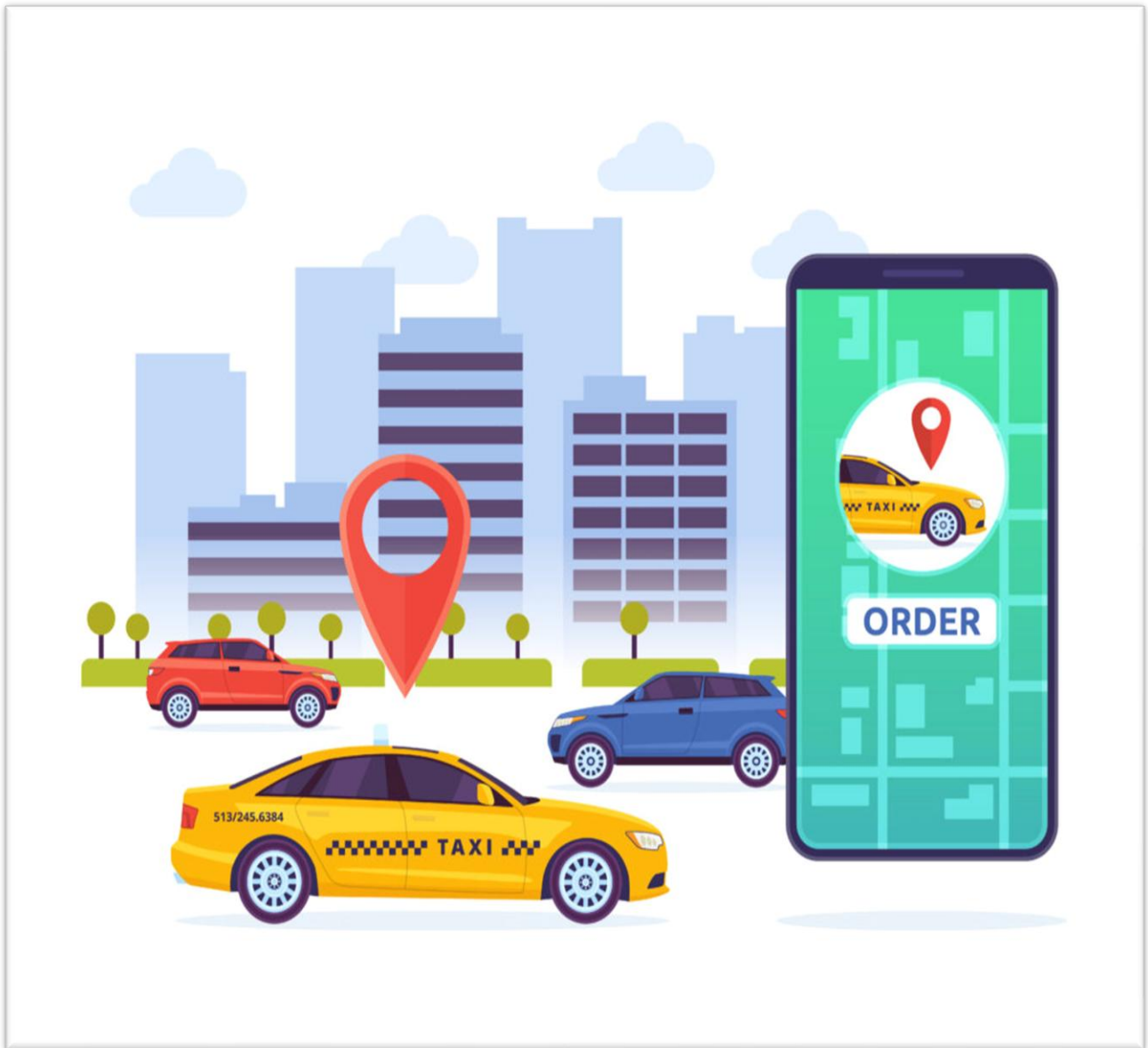


CAB BOOKING SYSTEM



- Musharraf Mujeeb Bubere
BATCH - May25 - T340

CAB BOOKING SYSTEM

The cab booking industry has grown rapidly, offering convenient transportation for customers and income for drivers. To stay competitive and efficient, companies now rely on data to understand booking trends, customer behavior, and driver performance.

This project designs a relational database for managing key entities like Customers, Drivers, Cabs, Bookings, TripDetails, and Feedback. Through SQL-based analysis, the system helps improve operations, track revenue, and support smarter business decisions.

Project Aim

- Design a relational database for a cab booking system.
- Efficiently store and manage data for customers, drivers, cabs, bookings, trips, and feedback.
- Enable tracking of bookings, trip details, and customer feedback.
- Support analysis of driver performance and customer behaviour.
- Help identify revenue trends and operational issues.
- Facilitate data-driven decisions to improve service quality and profitability.

ER DIAGRAM FOR CAB BOOKING SYSTEM

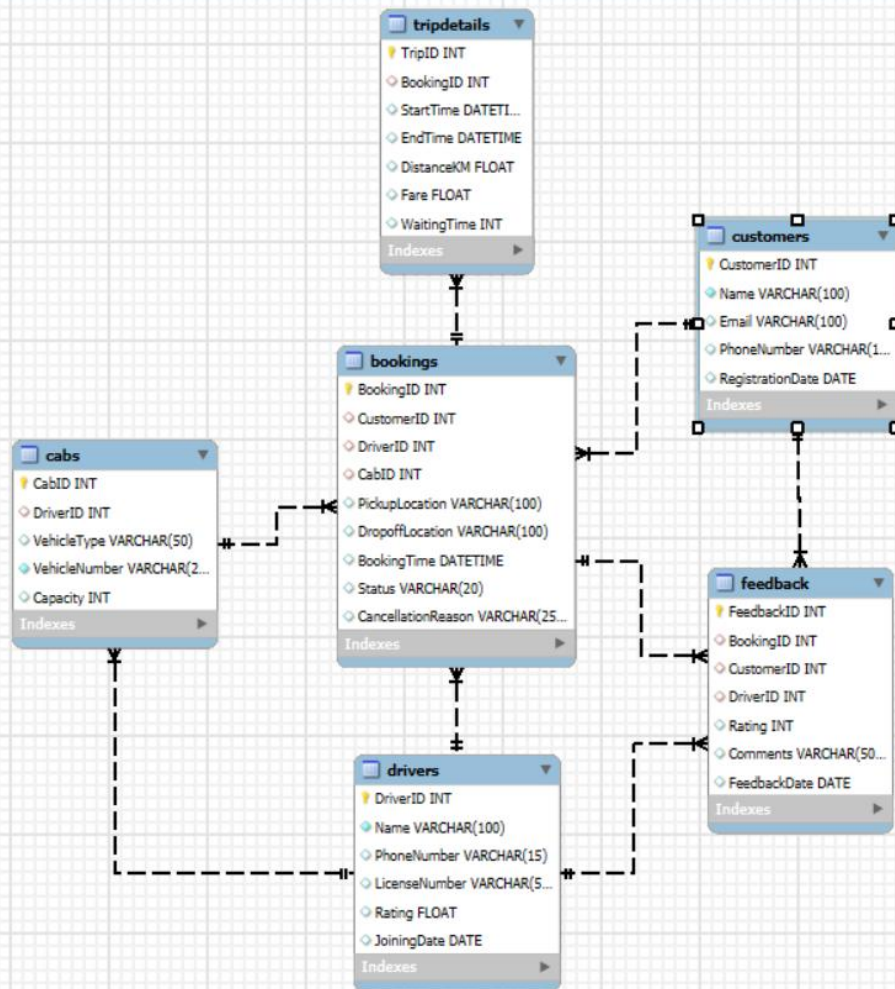





Table Description:

1. Customers

Result Grid

 Filter Rows:


Export: 


Wrap Cell Content: 

	Field	Type	Null	Key	Default	Extra
▶	CustomerID	int	NO	PRI	NULL	auto_increment
	Name	varchar(100)	NO		NULL	
	Email	varchar(100)	YES	UNI	NULL	
	PhoneNumber	varchar(15)	YES		NULL	
	RegistrationDate	date	YES		NULL	


2. Drivers

Result Grid

 Filter Rows:

 Export:

Wrap Cell Content:



	Field	Type	Null	Key	Default	Extra
▶	DriverID	int	NO	PRI	NULL	auto_increment
	Name	varchar(100)	NO		NULL	
	PhoneNumber	varchar(15)	YES		NULL	
	LicenseNumber	varchar(50)	YES	UNI	NULL	
	Rating	float	YES		NULL	
	JoiningDate	date	YES		NULL	

3. Cabs

Result Grid

Filter Rows:


Export:


Wrap Cell Content:


	Field	Type	Null	Key	Default	Extra
▶	CabID	int	NO	PRI	NULL	auto_increment
	DriverID	int	YES	MUL	NULL	
	VehicleType	varchar(50)	YES		NULL	
	VehicleNumber	varchar(20)	NO	UNI	NULL	
	Capacity	int	YES		NULL	

4. Bookings

Result Grid

 Filter Rows:


Export: 


Wrap Cell Content: 


	Field	Type	Null	Key	Default	Extra
▶	BookingID	int	NO	PRI	NULL	auto_increment
	CustomerID	int	YES	MUL	NULL	
	DriverID	int	YES	MUL	NULL	
	CabID	int	YES	MUL	NULL	
	PickupLocation	varchar(100)	YES		NULL	
	DropoffLocation	varchar(100)	YES		NULL	
	BookingTime	datetime	YES		NULL	
	Status	varchar(20)	YES		NULL	
	CancellationReason	varchar(255)	YES		NULL	

5. TripDetails

Result Grid

 Filter Rows:


Export: 


Wrap Cell Content: 


	Field	Type	Null	Key	Default	Extra
▶	TripID	int	NO	PRI	NULL	auto_increment
	BookingID	int	YES	MUL	NULL	
	StartTime	datetime	YES		NULL	
	EndTime	datetime	YES		NULL	
	DistanceKM	float	YES		NULL	
	Fare	float	YES		NULL	
	WaitingTime	int	YES		NULL	

6. Feedback

Result Grid

 Filter Rows:

Export: 

Wrap Cell Content: 

	Field	Type	Null	Key	Default	Extra
▶	FeedbackID	int	NO	PRI	NULL	auto_increment
	BookingID	int	YES	MUL	NULL	
	CustomerID	int	YES	MUL	NULL	
	DriverID	int	YES	MUL	NULL	
	Rating	int	YES		NULL	
	Comments	varchar(500)	YES		NULL	
	FeedbackDate	date	YES		NULL	

CREATING DATABASE:

CREATE DATABASE cab_booking_system;

USE cab_booking_system;

Table Creation & Insertion Commands:

1) Create Customers Table

```
CREATE TABLE Customers (  
    CustomerID INT PRIMARY KEY AUTO_INCREMENT,  
    Name VARCHAR(100) NOT NULL,  
    Email VARCHAR(100) UNIQUE,  
    PhoneNumber VARCHAR(15),  
    RegistrationDate DATE  
);
```

Inserting Values into Customers Table:

INSERT INTO Customers (Name, Email, PhoneNumber, RegistrationDate)

VALUES

```
('Raj Sharma', 'raj.sharma1@example.com', '9876543201', '2025-05-01'),  
( 'Anita Singh', 'anita.singh2@example.com', '9876543202', '2025-05-02'),  
( 'Mohit Verma', 'mohit.verma3@example.com', '9876543203', '2025-05-03'),  
( 'Priya Patel', 'priya.patel4@example.com', '9876543204', '2025-05-04'),  
( 'Amit Kumar', 'amit.kumar5@example.com', '9876543205', '2025-05-05'),  
( 'Sneha Joshi', 'sneha.joshi6@example.com', '9876543206', '2025-05-06'),  
( 'Vikas Mehta', 'vikas.mehta7@example.com', '9876543207', '2025-05-07'),  
( 'Nisha Gupta', 'nisha.gupta8@example.com', '9876543208', '2025-05-08'),  
( 'Rohan Das', 'rohan.das9@example.com', '9876543209', '2025-05-09'),  
( 'Kiran Yadav', 'kiran.yadav10@example.com', '9876543210', '2025-05-10'),  
( 'Deepak Nair', 'deepak.nair11@example.com', '9876543211', '2025-05-11'),  
( 'Meera Iyer', 'meera.iyer12@example.com', '9876543212', '2025-05-12'),  
( 'Arjun Reddy', 'arjun.reddy13@example.com', '9876543213', '2025-05-13'),
```

('Shreya Rao', 'shreya.rao14@example.com', '9876543214', '2025-05-14'),
 ('Kunal Sethi', 'kunal.sethi15@example.com', '9876543215', '2025-05-15'),
 ('Riya Jain', 'riya.jain16@example.com', '9876543216', '2025-05-16'),
 ('Naveen Bhat', 'naveen.bhat17@example.com', '9876543217', '2025-05-17'),
 ('Pooja Pillai', 'pooja.pillai18@example.com', '9876543218', '2025-05-18'),
 ('Varun Saxena', 'varun.saxena19@example.com', '9876543219', '2025-05-19'),
 ('Tanvi Kapoor', 'tanvi.kapoor20@example.com', '9876543220', '2025-05-20'),
 ('Siddharth Jain', 'siddharth.jain21@example.com', '9876543221', '2025-05-21'),
 ('Neha Malhotra', 'neha.malhotra22@example.com', '9876543222', '2025-05-22'),
 ('Harshit Gupta', 'harshit.gupta23@example.com', '9876543223', '2025-05-23'),
 ('Divya Menon', 'divya.menon24@example.com', '9876543224', '2025-05-24'),
 ('Manish Rawat', 'manish.rawat25@example.com', '9876543225', '2025-05-25'),
 ('Ayesha Khan', 'ayasha.khan26@example.com', '9876543226', '2025-05-26'),
 ('Rahul Chopra', 'rahul.chopra27@example.com', '9876543227', '2025-05-27'),
 ('Sakshi Bansal', 'sakshi.bansal28@example.com', '9876543228', '2025-05-28'),
 ('Yash Patel', 'yash.patel29@example.com', '9876543229', '2025-05-29'),
 ('Preeti Desai', 'preeti.desai30@example.com', '9876543230', '2025-05-30');

SELECT * FROM Customers;

OUTPUT:

Result Grid Filter Rows: <input type="text"/> Edit: Export/Import: Wrap Cell Content:					
	CustomerID	Name	Email	PhoneNumber	RegistrationDate
▶	1	Raj Sharma	raj.sharma1@example.com	9876543201	2025-05-01
	2	Anita Singh	anita.singh2@example.com	9876543202	2025-05-02
	3	Mohit Verma	mohit.verma3@example.com	9876543203	2025-05-03
	4	Priya Patel	priya.patel4@example.com	9876543204	2025-05-04
	5	Amit Kumar	amit.kumar5@example.com	9876543205	2025-05-05
	6	Sneha Joshi	sneha.joshi6@example.com	9876543206	2025-05-06
	7	Vikas Mehta	vikas.mehta7@example.com	9876543207	2025-05-07
	8	Nisha Gupta	nisha.gupta8@example.com	9876543208	2025-05-08
	9	Rohan Das	rohan.das9@example.com	9876543209	2025-05-09
	10	Kiran Yadav	kiran.yadav10@example.com	9876543210	2025-05-10
	11	Deepak Nair	deepak.nair11@example.com	9876543211	2025-05-11

2) Create Drivers Table:

```
CREATE TABLE Drivers (  
    DriverID INT PRIMARY KEY AUTO_INCREMENT,  
    Name VARCHAR(100) NOT NULL,  
    PhoneNumber VARCHAR(15),  
    LicenseNumber VARCHAR(50) UNIQUE,  
    Rating FLOAT CHECK (Rating BETWEEN 0 AND 5),  
    JoiningDate DATE  
);
```

Inserting Values into Drivers Table:

```
INSERT INTO Drivers (Name, PhoneNumber, LicenseNumber, Rating, JoiningDate)  
VALUES
```

```
('Rahul Mehta', '9876500011', 'MH01D10001', 4.5, '2025-05-01'),  
( 'Sunita Reddy', '9876500012', 'MH01D10002', 3.8, '2025-05-02'),  
( 'Aakash Gupta', '9876500013', 'MH01D10003', 4.2, '2025-05-03'),  
( 'Neeraj Singh', '9876500014', 'MH01D10004', 2.9, '2025-05-04'),  
( 'Pankaj Verma', '9876500015', 'MH01D10005', 3.2, '2025-05-05'),  
( 'Harsha Nair', '9876500016', 'MH01D10006', 4.8, '2025-05-06'),  
( 'Ritika Iyer', '9876500017', 'MH01D10007', 4.0, '2025-05-07'),  
( 'Arvind Kumar', '9876500018', 'MH01D10008', 2.5, '2025-05-08'),  
( 'Snehal Joshi', '9876500019', 'MH01D10009', 3.9, '2025-05-09'),  
( 'Vivek Rana', '9876500020', 'MH01D10010', 4.1, '2025-05-10'),  
( 'Megha Shah', '9876500021', 'MH01D10011', 3.7, '2025-05-11'),  
( 'Suresh Raina', '9876500022', 'MH01D10012', 2.8, '2025-05-12'),  
( 'Bhavna Tiwari', '9876500023', 'MH01D10013', 3.6, '2025-05-13'),  
( 'Nikhil Pandey', '9876500024', 'MH01D10014', 4.3, '2025-05-14'),  
( 'Komal Mishra', '9876500025', 'MH01D10015', 4.9, '2025-05-15'),  
( 'Farhan Sheikh', '9876500026', 'MH01D10016', 3.1, '2025-05-16'),  
( 'Anushka Sen', '9876500027', 'MH01D10017', 2.6, '2025-05-17'),  
( 'Ravi Pillai', '9876500028', 'MH01D10018', 4.4, '2025-05-18'),
```


('Simran Gill', '9876500029', 'MH01D10019', 3.3, '2025-05-19'),
 ('Ajay Dev', '9876500030', 'MH01D10020', 4.6, '2025-05-20'),
 ('Karishma Roy', '9876500031', 'MH01D10021', 3.5, '2025-05-21'),
 ('Vivek Oberoi', '9876500032', 'MH01D10022', 2.7, '2025-05-22'),
 ('Reema Khan', '9876500033', 'MH01D10023', 3.4, '2025-05-23'),
 ('Rohit Sharma', '9876500034', 'MH01D10024', 4.7, '2025-05-24'),
 ('Swati Chauhan', '9876500035', 'MH01D10025', 3.0, '2025-05-25'),
 ('Deepak Malhotra', '9876500036', 'MH01D10026', 2.4, '2025-05-26'),
 ('Lavanya Menon', '9876500037', 'MH01D10027', 4.0, '2025-05-27'),
 ('Pranav Joshi', '9876500038', 'MH01D10028', 3.9, '2025-05-28'),
 ('Shruti Desai', '9876500039', 'MH01D10029', 4.1, '2025-05-29'),
 ('Yash Rathore', '9876500040', 'MH01D10030', 3.2, '2025-05-30');

SELECT * FROM Customers;

OUTPUT:

Result Grid						
		Filter Rows:	Edit:		Export/Import:	Wrap Cell Content:
	DriverID	Name	PhoneNumber	LicenseNumber	Rating	JoiningDate
▶	1	Rahul Mehta	9876500011	MH01D10001	4.5	2025-05-01
	2	Sunita Reddy	9876500012	MH01D10002	3.8	2025-05-02
	3	Aakash Gupta	9876500013	MH01D10003	4.2	2025-05-03
	4	Neeraj Singh	9876500014	MH01D10004	2.9	2025-05-04
	5	Pankaj Verma	9876500015	MH01D10005	3.2	2025-05-05
	6	Harsha Nair	9876500016	MH01D10006	4.8	2025-05-06
	7	Ritika Iyer	9876500017	MH01D10007	4	2025-05-07
	8	Arvind Kumar	9876500018	MH01D10008	2.5	2025-05-08
	9	Snehal Joshi	9876500019	MH01D10009	3.9	2025-05-09
	10	Vivek Rana	9876500020	MH01D10010	4.1	2025-05-10
	11	Megha Shah	9876500021	MH01D10011	3.7	2025-05-11

3) Create Cabs Table:

```
CREATE TABLE Cabs (  
    CabID INT PRIMARY KEY AUTO_INCREMENT,  
    DriverID INT,  
    VehicleType VARCHAR(50),  
    VehicleNumber VARCHAR(20) UNIQUE NOT NULL,  
    Capacity INT,  
    FOREIGN KEY (DriverID) REFERENCES Drivers(DriverID)  
);
```

Inserting Values into Cabs Table:

```
INSERT INTO Cabs (DriverID, VehicleType, VehicleNumber, Capacity)
```

VALUES

```
(1, 'Sedan', 'MH01C1001', 4),  
(2, 'SUV', 'MH01C1002', 6),  
(3, 'Hatchback', 'MH01C1003', 4),  
(4, 'Sedan', 'MH01C1004', 4),  
(5, 'SUV', 'MH01C1005', 6),  
(6, 'Hatchback', 'MH01C1006', 4),  
(7, 'Sedan', 'MH01C1007', 4),  
(8, 'SUV', 'MH01C1008', 6),  
(9, 'Hatchback', 'MH01C1009', 4),  
(10, 'Sedan', 'MH01C1010', 4),  
(11, 'SUV', 'MH01C1011', 6),  
(12, 'Hatchback', 'MH01C1012', 4),  
(13, 'Sedan', 'MH01C1013', 4),  
(14, 'SUV', 'MH01C1014', 6),  
(15, 'Hatchback', 'MH01C1015', 4),  
(16, 'Sedan', 'MH01C1016', 4),
```

(17, 'SUV', 'MH01C1017', 6),
 (18, 'Hatchback', 'MH01C1018', 4),
 (19, 'Sedan', 'MH01C1019', 4),
 (20, 'SUV', 'MH01C1020', 6),
 (21, 'Hatchback', 'MH01C1021', 4),
 (22, 'Sedan', 'MH01C1022', 4),
 (23, 'SUV', 'MH01C1023', 6),
 (24, 'Hatchback', 'MH01C1024', 4),
 (25, 'Sedan', 'MH01C1025', 4),
 (26, 'SUV', 'MH01C1026', 6),
 (27, 'Hatchback', 'MH01C1027', 4),
 (28, 'Sedan', 'MH01C1028', 4),
 (29, 'SUV', 'MH01C1029', 6),
 (30, 'Hatchback', 'MH01C1030', 4);

SELECT * FROM Cabs;

OUTPUT:

Result Grid					
Filter Rows:					
Edit:					
	CabID	DriverID	VehicleType	VehicleNumber	Capacity
▶	1	1	Sedan	MH01C1001	4
	2	2	SUV	MH01C1002	6
	3	3	Hatchback	MH01C1003	4
	4	4	Sedan	MH01C1004	4
	5	5	SUV	MH01C1005	6
	6	6	Hatchback	MH01C1006	4
	7	7	Sedan	MH01C1007	4
	8	8	SUV	MH01C1008	6
	9	9	Hatchback	MH01C1009	4
	10	10	Sedan	MH01C1010	4
	11	11	SUV	MH01C1011	6

4) Create Bookings Table:

```
CREATE TABLE Bookings (  
    BookingID INT PRIMARY KEY AUTO_INCREMENT,  
    CustomerID INT,  
    DriverID INT,  
    CabID INT,  
    PickupLocation VARCHAR(100),  
    DropoffLocation VARCHAR(100),  
    BookingTime DATETIME,  
    Status VARCHAR(20) CHECK (Status IN ('Completed', 'Cancelled', 'Ongoing')),  
    CancellationReason VARCHAR(255),  
    FOREIGN KEY (CustomerID) REFERENCES Customers(CustomerID),  
    FOREIGN KEY (DriverID) REFERENCES Drivers(DriverID),  
    FOREIGN KEY (CabID) REFERENCES Cabs(CabID)  
);
```

Inserting Values into Bookings Table:

```
INSERT INTO Bookings (CustomerID, DriverID, CabID, PickupLocation,  
DropoffLocation, BookingTime, Status, CancellationReason)
```

VALUES

```
(1, 1, 1, 'Andheri', 'Bandra', '2025-05-01 08:00:00', 'Completed', NULL),  
(1, 2, 2, 'Andheri', 'Dadar', '2025-05-02 09:30:00', 'Completed', NULL),  
(1, 3, 3, 'Andheri', 'Churchgate', '2025-05-03 10:45:00', 'Cancelled', 'Changed plan'),  
(1, 4, 4, 'Andheri', 'Colaba', '2025-05-04 11:15:00', 'Completed', NULL),  
(1, 5, 5, 'Andheri', 'Powai', '2025-05-05 12:30:00', 'Completed', NULL),  
(1, 6, 6, 'Andheri', 'Ghatkopar', '2025-05-06 14:00:00', 'Completed', NULL),  
(1, 7, 7, 'Andheri', 'Borivali', '2025-05-07 15:15:00', 'Ongoing', NULL),  
(1, 8, 8, 'Andheri', 'Mulund', '2025-05-08 16:30:00', 'Completed', NULL),  
(1, 9, 9, 'Andheri', 'Thane', '2025-05-09 17:45:00', 'Cancelled', 'Driver delayed'),  
(1, 10, 10, 'Andheri', 'Kurla', '2025-05-10 19:00:00', 'Completed', NULL),  
(1, 11, 11, 'Andheri', 'Bandra', '2025-05-11 20:15:00', 'Completed', NULL),  
(1, 12, 12, 'Andheri', 'Powai', '2025-05-12 21:30:00', 'Completed', NULL),  
(2, 13, 13, 'Kurla', 'Powai', '2025-05-01 08:10:00', 'Completed', NULL),
```

(2, 14, 14, 'Kurla', 'Vikhroli', '2025-05-02 09:40:00', 'Completed', NULL),
(2, 15, 15, 'Kurla', 'Bandra', '2025-05-03 10:50:00', 'Cancelled', 'Alternate transport'),
(2, 16, 16, 'Kurla', 'Andheri', '2025-05-04 11:20:00', 'Completed', NULL),
(2, 17, 17, 'Kurla', 'Colaba', '2025-05-05 12:35:00', 'Completed', NULL),
(2, 18, 18, 'Kurla', 'Powai', '2025-05-06 14:05:00', 'Completed', NULL),
(2, 19, 19, 'Kurla', 'Thane', '2025-05-07 15:20:00', 'Ongoing', NULL),
(2, 20, 20, 'Kurla', 'Mulund', '2025-05-08 16:35:00', 'Completed', NULL),
(2, 21, 21, 'Kurla', 'Ghatkopar', '2025-05-09 17:50:00', 'Completed', NULL),
(2, 22, 22, 'Kurla', 'Powai', '2025-05-10 19:05:00', 'Completed', NULL),
(3, 23, 23, 'Thane', 'Mulund', '2025-05-01 08:20:00', 'Completed', NULL),
(3, 24, 24, 'Thane', 'Bhandup', '2025-05-02 09:50:00', 'Completed', NULL),
(3, 25, 25, 'Thane', 'Ghatkopar', '2025-05-03 11:00:00', 'Cancelled', 'Driver delayed'),
(3, 26, 26, 'Thane', 'Vikhroli', '2025-05-04 11:30:00', 'Completed', NULL),
(3, 27, 27, 'Thane', 'Powai', '2025-05-05 12:45:00', 'Completed', NULL),
(3, 28, 28, 'Thane', 'Andheri', '2025-05-06 14:10:00', 'Completed', NULL),
(3, 29, 29, 'Thane', 'Borivali', '2025-05-07 15:25:00', 'Ongoing', NULL),
(3, 30, 30, 'Thane', 'Mulund', '2025-05-08 16:40:00', 'Completed', NULL),
(3, 1, 1, 'Thane', 'Bandra', '2025-05-09 17:55:00', 'Completed', NULL),
(4, 2, 2, 'Borivali', 'Dadar', '2025-05-10 19:10:00', 'Completed', NULL),
(4, 3, 3, 'Borivali', 'Malad', '2025-05-11 20:25:00', 'Completed', NULL),
(4, 4, 4, 'Borivali', 'Andheri', '2025-05-12 21:40:00', 'Cancelled', 'Changed plan'),
(4, 5, 5, 'Borivali', 'Bandra', '2025-05-13 08:00:00', 'Completed', NULL),
(4, 6, 6, 'Borivali', 'Powai', '2025-05-14 09:15:00', 'Completed', NULL),
(4, 7, 7, 'Borivali', 'Churchgate', '2025-05-15 10:30:00', 'Ongoing', NULL),
(4, 8, 8, 'Borivali', 'Kurla', '2025-05-16 11:45:00', 'Completed', NULL),
(5, 9, 9, 'Bandra', 'Andheri', '2025-05-17 12:00:00', 'Completed', NULL),
(5, 10, 10, 'Bandra', 'Dadar', '2025-05-18 13:15:00', 'Completed', NULL),
(5, 11, 11, 'Bandra', 'Kurla', '2025-05-19 14:30:00', 'Cancelled', 'Alternate transport'),
(5, 12, 12, 'Bandra', 'Powai', '2025-05-20 15:45:00', 'Completed', NULL),

(5, 13, 13, 'Bandra', 'Bhandup', '2025-05-21 17:00:00', 'Completed', NULL),
(5, 14, 14, 'Bandra', 'Colaba', '2025-05-22 18:15:00', 'Ongoing', NULL),
(6, 15, 15, 'Chembur', 'Sion', '2025-05-23 19:30:00', 'Completed', NULL),
(7, 16, 16, 'Vashi', 'Nerul', '2025-05-24 20:45:00', 'Completed', NULL),
(8, 17, 17, 'Ghatkopar', 'Vikhroli', '2025-05-25 08:00:00', 'Cancelled', 'No show'),
(9, 18, 18, 'Malad', 'Goregaon', '2025-05-26 09:15:00', 'Completed', NULL),
(10, 19, 19, 'Dadar', 'Churchgate', '2025-05-27 10:30:00', 'Completed', NULL),
(6, 20, 20, 'Chembur', 'Powai', '2025-05-28 11:45:00', 'Completed', NULL),
(6, 21, 21, 'Chembur', 'Kurla', '2025-05-29 12:50:00', 'Completed', NULL),
(6, 22, 22, 'Chembur', 'Andheri', '2025-05-30 14:00:00', 'Cancelled', 'Changed plan'),
(6, 23, 23, 'Chembur', 'Bandra', '2025-05-31 15:10:00', 'Completed', NULL),
(6, 24, 24, 'Chembur', 'Thane', '2025-06-01 16:20:00', 'Completed', NULL),
(7, 25, 25, 'Vashi', 'Belapur', '2025-06-02 08:00:00', 'Completed', NULL),
(7, 26, 26, 'Vashi', 'Nerul', '2025-06-03 09:10:00', 'Completed', NULL),
(7, 27, 27, 'Vashi', 'CBD', '2025-06-04 10:20:00', 'Cancelled', 'Driver delayed'),
(7, 28, 28, 'Vashi', 'Thane', '2025-06-05 11:30:00', 'Completed', NULL),
(8, 29, 29, 'Ghatkopar', 'Vikhroli', '2025-06-06 12:40:00', 'Completed', NULL),
(8, 30, 30, 'Ghatkopar', 'Powai', '2025-06-07 13:50:00', 'Completed', NULL),
(8, 1, 1, 'Ghatkopar', 'Andheri', '2025-06-08 15:00:00', 'Cancelled', 'Alternate transport'),
(9, 2, 2, 'Malad', 'Borivali', '2025-06-09 08:00:00', 'Completed', NULL),
(9, 3, 3, 'Malad', 'Goregaon', '2025-06-10 09:10:00', 'Completed', NULL),
(9, 4, 4, 'Malad', 'Andheri', '2025-06-11 10:20:00', 'Ongoing', NULL),
(10, 5, 5, 'Dadar', 'Churchgate', '2025-06-12 11:30:00', 'Completed', NULL),
(10, 6, 6, 'Dadar', 'Bandra', '2025-06-13 12:40:00', 'Cancelled', 'Changed plan'),
(11, 7, 7, 'Colaba', 'Nariman Point', '2025-06-14 13:50:00', 'Completed', NULL),
(12, 8, 8, 'Marine Lines', 'Churchgate', '2025-06-15 15:00:00', 'Completed', NULL),
(13, 9, 9, 'Parel', 'Dadar', '2025-06-16 16:10:00', 'Cancelled', 'No show'),
(14, 10, 10, 'Sion', 'Chembur', '2025-06-17 17:20:00', 'Completed', NULL),

(15, 11, 11, 'Kurla', 'Ghatkopar', '2025-06-18 08:30:00', 'Completed', NULL),
 (16, 12, 12, 'Thane', 'Mulund', '2025-06-19 09:40:00', 'Completed', NULL),
 (17, 13, 13, 'Bhandup', 'Kanjurmarg', '2025-06-20 10:50:00', 'Cancelled', 'Changed plan'),
 (18, 14, 14, 'Powai', 'Hiranandani', '2025-06-21 12:00:00', 'Completed', NULL),
 (19, 15, 15, 'Vikhroli', 'Kanjurmarg', '2025-06-22 13:10:00', 'Completed', NULL),
 (20, 16, 16, 'Chembur', 'Govandi', '2025-06-23 14:20:00', 'Ongoing', NULL),
 (1, 17, 17, 'Andheri', 'Bandra', '2025-07-01 09:00:00', 'Completed', NULL),
 (1, 18, 18, 'Andheri', 'Dadar', '2025-07-02 10:15:00', 'Completed', NULL),
 (2, 19, 19, 'Kurla', 'Powai', '2025-07-01 11:30:00', 'Completed', NULL),
 (2, 20, 20, 'Kurla', 'Bandra', '2025-07-02 12:45:00', 'Cancelled', 'Alternate transport'),
 (3, 21, 21, 'Thane', 'Mulund', '2025-07-01 14:00:00', 'Completed', NULL),
 (3, 22, 22, 'Thane', 'Bhandup', '2025-07-02 15:15:00', 'Completed', NULL),
 (4, 23, 23, 'Borivali', 'Andheri', '2025-07-01 16:30:00', 'Completed', NULL),
 (4, 24, 24, 'Borivali', 'Kurla', '2025-07-02 17:45:00', 'Completed', NULL),
 (5, 25, 25, 'Bandra', 'Powai', '2025-07-01 19:00:00', 'Completed', NULL),
 (5, 26, 26, 'Bandra', 'Colaba', '2025-07-02 20:15:00', 'Ongoing', NULL);

SELECT * FROM Bookings;

OUTPUT:

Result Grid Filter Rows: <input type="text"/> Edit: Export/Import: Wrap Cell Content:									
	BookingID	CustomerID	DriverID	CabID	PickupLocation	DropoffLocation	BookingTime	Status	CancellationReason
▶	1	1	1	1	Andheri	Bandra	2025-05-01 08:00:00	Completed	NULL
	2	1	2	2	Andheri	Dadar	2025-05-02 09:30:00	Completed	NULL
	3	1	3	3	Andheri	Churchgate	2025-05-03 10:45:00	Cancelled	Changed plan
	4	1	4	4	Andheri	Colaba	2025-05-04 11:15:00	Completed	NULL
	5	1	5	5	Andheri	Powai	2025-05-05 12:30:00	Completed	NULL
	6	1	6	6	Andheri	Ghatkopar	2025-05-06 14:00:00	Completed	NULL
	7	1	7	7	Andheri	Borivali	2025-05-07 15:15:00	Ongoing	NULL
	8	1	8	8	Andheri	Mulund	2025-05-08 16:30:00	Completed	NULL
	9	1	9	9	Andheri	Thane	2025-05-09 17:45:00	Cancelled	Driver delayed
	10	1	10	10	Andheri	Kurla	2025-05-10 19:00:00	Completed	NULL
	11	1	11	11	Andheri	Bandra	2025-05-11 20:15:00	Completed	NULL

5) Create TripDetails Table:

```
CREATE TABLE TripDetails (  
    TripID INT PRIMARY KEY AUTO_INCREMENT,  
    BookingID INT,  
    StartTime DATETIME,  
    EndTime DATETIME,  
    DistanceKM FLOAT,  
    Fare FLOAT,  
    WaitingTime INT, -- in minutes  
    FOREIGN KEY (BookingID) REFERENCES Bookings(BookingID)  
);
```

Inserting Values into TripDetails Table:

```
INSERT INTO TripDetails (BookingID, StartTime, EndTime, DistanceKM, Fare,  
WaitingTime)
```

VALUES

```
(1, '2025-05-01 08:05:00', '2025-05-01 08:30:00', 5.0, 100.0, 5),  
(2, '2025-05-02 09:35:00', '2025-05-02 10:10:00', 6.5, 130.0, 5),  
(4, '2025-05-04 11:20:00', '2025-05-04 11:50:00', 8.0, 160.0, 5),  
(5, '2025-05-05 12:35:00', '2025-05-05 13:25:00', 12.0, 240.0, 5),  
(6, '2025-05-06 14:10:00', '2025-05-06 14:35:00', 4.0, 80.0, 10),  
(7, '2025-05-07 15:25:00', '2025-05-07 15:55:00', 3.5, 70.0, 10),  
(8, '2025-05-08 16:40:00', '2025-05-08 17:15:00', 6.0, 120.0, 15),  
(10, '2025-05-10 19:10:00', '2025-05-10 19:45:00', 9.0, 180.0, 10),  
(11, '2025-05-11 20:20:00', '2025-05-11 21:00:00', 7.0, 140.0, 5),  
(12, '2025-05-12 21:40:00', '2025-05-12 22:10:00', 6.0, 120.0, 10),  
(13, '2025-05-01 08:20:00', '2025-05-01 08:50:00', 8.0, 160.0, 10),  
(14, '2025-05-02 09:50:00', '2025-05-02 10:20:00', 6.0, 120.0, 10),  
(16, '2025-05-04 11:30:00', '2025-05-04 12:00:00', 9.0, 180.0, 10),  
(17, '2025-05-05 12:40:00', '2025-05-05 13:10:00', 7.0, 140.0, 5),  
(18, '2025-05-06 14:15:00', '2025-05-06 14:45:00', 8.0, 160.0, 5),  
(19, '2025-05-07 15:30:00', '2025-05-07 16:00:00', 9.0, 180.0, 5),
```

(20, '2025-05-08 16:45:00', '2025-05-08 17:15:00', 6.0, 120.0, 5),
(21, '2025-05-09 17:55:00', '2025-05-09 18:25:00', 5.5, 110.0, 5),
(22, '2025-05-10 19:05:00', '2025-05-10 19:35:00', 8.5, 170.0, 5),
(23, '2025-05-11 20:25:00', '2025-05-11 20:55:00', 7.5, 150.0, 5),
(24, '2025-05-12 21:45:00', '2025-05-12 22:15:00', 4.5, 90.0, 10),
(25, '2025-05-13 08:05:00', '2025-05-13 08:35:00', 6.5, 130.0, 5),
(26, '2025-05-14 09:20:00', '2025-05-14 09:50:00', 5.0, 100.0, 5),
(27, '2025-05-15 10:35:00', '2025-05-15 11:05:00', 4.0, 80.0, 5),
(28, '2025-05-16 11:50:00', '2025-05-16 12:20:00', 3.5, 70.0, 5),
(29, '2025-05-17 12:05:00', '2025-05-17 12:35:00', 5.5, 110.0, 5),
(30, '2025-05-18 13:20:00', '2025-05-18 13:50:00', 7.0, 140.0, 10),
(31, '2025-05-19 14:35:00', '2025-05-19 15:05:00', 8.5, 170.0, 5),
(32, '2025-05-20 15:50:00', '2025-05-20 16:20:00', 10.0, 200.0, 5),
(33, '2025-05-21 17:05:00', '2025-05-21 17:35:00', 12.0, 240.0, 5),
(34, '2025-05-22 18:20:00', '2025-05-22 18:50:00', 11.0, 220.0, 5),
(35, '2025-05-23 19:35:00', '2025-05-23 20:05:00', 9.0, 180.0, 5),
(36, '2025-05-24 20:50:00', '2025-05-24 21:20:00', 8.0, 160.0, 5),
(37, '2025-05-25 08:05:00', '2025-05-25 08:35:00', 5.0, 100.0, 5),
(38, '2025-05-26 09:20:00', '2025-05-26 09:50:00', 4.5, 90.0, 5),
(39, '2025-05-27 10:35:00', '2025-05-27 11:05:00', 6.5, 130.0, 5),
(40, '2025-05-28 11:50:00', '2025-05-28 12:20:00', 7.5, 150.0, 5),
(41, '2025-05-29 12:05:00', '2025-05-29 12:35:00', 6.0, 120.0, 5),
(42, '2025-05-30 13:20:00', '2025-05-30 13:50:00', 8.0, 160.0, 5),
(43, '2025-05-31 14:35:00', '2025-05-31 15:05:00', 9.0, 180.0, 5),
(44, '2025-06-01 15:50:00', '2025-06-01 16:20:00', 10.0, 200.0, 5),
(45, '2025-06-02 17:05:00', '2025-06-02 17:35:00', 11.0, 220.0, 5),
(46, '2025-06-03 18:20:00', '2025-06-03 18:50:00', 12.0, 240.0, 5),
(47, '2025-06-04 19:35:00', '2025-06-04 20:05:00', 13.0, 260.0, 5),
(48, '2025-06-05 20:50:00', '2025-06-05 21:20:00', 14.0, 280.0, 5),

(49, '2025-06-06 08:05:00', '2025-06-06 08:35:00', 15.0, 300.0, 5),
(50, '2025-06-07 09:20:00', '2025-06-07 09:50:00', 16.0, 320.0, 5);

SELECT * FROM TripDetails;

OUTPUT:

Result Grid		Filter Rows:		Edit:		Export/Import:		Wrap Cell C
	TripID	BookingID	StartTime	EndTime	DistanceKM	Fare	WaitingTime	
▶	1	1	2025-05-01 08:05:00	2025-05-01 08:30:00	5	100	5	
	2	2	2025-05-02 09:35:00	2025-05-02 10:10:00	6.5	130	5	
	3	4	2025-05-04 11:20:00	2025-05-04 11:50:00	8	160	5	
	4	5	2025-05-05 12:35:00	2025-05-05 13:25:00	12	240	5	
	5	6	2025-05-06 14:10:00	2025-05-06 14:35:00	4	80	10	
	6	7	2025-05-07 15:25:00	2025-05-07 15:55:00	3.5	70	10	
	7	8	2025-05-08 16:40:00	2025-05-08 17:15:00	6	120	15	
	8	10	2025-05-10 19:10:00	2025-05-10 19:45:00	9	180	10	
	9	11	2025-05-11 20:20:00	2025-05-11 21:00:00	7	140	5	
	10	12	2025-05-12 21:40:00	2025-05-12 22:10:00	6	120	10	
	11	13	2025-05-01 08:20:00	2025-05-01 08:50:00	8	160	10	

6) Create TripDetails Table:

```
CREATE TABLE Feedback (  
    FeedbackID INT PRIMARY KEY AUTO_INCREMENT,  
    BookingID INT,  
    CustomerID INT,  
    DriverID INT,  
    Rating INT CHECK (Rating BETWEEN 1 AND 5),  
    Comments VARCHAR(500),  
    FeedbackDate DATE,  
    FOREIGN KEY (BookingID) REFERENCES Bookings(BookingID),  
    FOREIGN KEY (CustomerID) REFERENCES Customers(CustomerID),  
    FOREIGN KEY (DriverID) REFERENCES Drivers(DriverID)  
);
```

Inserting Values into TripDetails Table:

```
INSERT INTO Feedback (BookingID, CustomerID, DriverID, Rating, Comments,  
FeedbackDate)
```

VALUES

```
(1, 1, 1, 5, 'Excellent ride, polite driver.', '2025-05-01'),  
(2, 1, 2, 4, 'Good service but slight delay.', '2025-05-02'),  
(4, 1, 4, 5, 'Comfortable and clean cab.', '2025-05-04'),  
(5, 1, 5, 4, 'Driver was courteous.', '2025-05-05'),  
(6, 1, 6, 3, 'Average ride, AC not working well.', '2025-05-06'),  
(7, 1, 7, 2, 'Driver was late and rude.', '2025-05-07'),  
(8, 1, 8, 4, 'Smooth ride.', '2025-05-08'),  
(10, 1, 10, 5, 'Very professional.', '2025-05-10'),  
(11, 1, 11, 5, 'On time and good driving.', '2025-05-11'),  
(12, 1, 12, 4, 'Cab was clean.', '2025-05-12'),  
(13, 2, 13, 3, 'Driver took longer route.', '2025-05-13'),  
(14, 2, 14, 4, 'Decent ride.', '2025-05-14'),  
(16, 2, 16, 5, 'Fast and smooth.', '2025-05-16'),  
(17, 2, 17, 5, 'Very good driving.', '2025-05-17'),  
(18, 2, 18, 4, 'Cab was neat.', '2025-05-18'),
```

(19, 2, 19, 3, 'Driver kept talking on phone.', '2025-05-19'),
(20, 2, 20, 5, 'Excellent service.', '2025-05-20'),
(21, 3, 21, 4, 'Comfortable ride.', '2025-05-21'),
(22, 3, 22, 2, 'Driver was rude.', '2025-05-22'),
(23, 3, 23, 5, 'Good experience.', '2025-05-23'),
(24, 3, 24, 4, 'Clean cab and good AC.', '2025-05-24'),
(25, 3, 25, 3, 'Average ride.', '2025-05-25'),
(26, 3, 26, 5, 'Driver was polite.', '2025-05-26'),
(27, 3, 27, 4, 'Quick drop off.', '2025-05-27'),
(28, 3, 28, 4, 'Satisfied.', '2025-05-28'),
(29, 3, 29, 5, 'Great service.', '2025-05-29'),
(30, 3, 30, 3, 'Cab was not very clean.', '2025-05-30'),
(31, 4, 1, 4, 'Good ride.', '2025-05-31'),
(32, 4, 2, 3, 'Driver was silent throughout.', '2025-06-01'),
(33, 4, 3, 4, 'Comfortable.', '2025-06-02'),
(34, 4, 4, 2, 'Driver was late.', '2025-06-03'),
(35, 4, 5, 5, 'Very good experience.', '2025-06-04'),
(36, 4, 6, 4, 'Smooth journey.', '2025-06-05'),
(37, 4, 7, 3, 'Average ride.', '2025-06-06'),
(38, 4, 8, 4, 'Satisfied.', '2025-06-07'),
(39, 5, 9, 5, 'Great driver.', '2025-06-08'),
(40, 5, 10, 3, 'Average service.', '2025-06-09'),
(41, 5, 11, 4, 'Good AC.', '2025-06-10'),
(42, 5, 12, 4, 'Comfortable ride.', '2025-06-11'),
(43, 5, 13, 2, 'Driver kept phone on speaker.', '2025-06-12'),
(44, 5, 14, 5, 'Excellent experience.', '2025-06-13'),
(45, 5, 15, 4, 'Nice and polite driver.', '2025-06-14'),
(46, 5, 16, 3, 'Average cab condition.', '2025-06-15'),
(47, 5, 17, 4, 'Good drop off.', '2025-06-16'),

(48, 5, 18, 5, 'Very comfortable ride.', '2025-06-17'),
(49, 5, 19, 4, 'Driver was friendly.', '2025-06-18'),
(50, 5, 20, 4, 'Cab was clean and on time.', '2025-06-19');

SELECT * FROM Feedback;

OUTPUT:

Result Grid		Filter Rows:		Edit:		Export/Import:		Wrap Cell Content
	FeedbackID	BookingID	CustomerID	DriverID	Rating	Comments	FeedbackDate	
▶	1	1	1	1	5	Excellent ride, polite driver.	2025-05-01	
	2	2	1	2	4	Good service but slight delay.	2025-05-02	
	3	4	1	4	5	Comfortable and clean cab.	2025-05-04	
	4	5	1	5	4	Driver was courteous.	2025-05-05	
	5	6	1	6	3	Average ride, AC not working well.	2025-05-06	
	6	7	1	7	2	Driver was late and rude.	2025-05-07	
	7	8	1	8	4	Smooth ride.	2025-05-08	
	8	10	1	10	5	Very professional.	2025-05-10	
	9	11	1	11	5	On time and good driving.	2025-05-11	
	10	12	1	12	4	Cab was clean.	2025-05-12	
	11	13	2	13	3	Driver took longer route.	2025-05-13	
	

PROBLEM STATEMENTS

Customer and Booking Analysis:

1. Identify customers who have completed the most bookings. What insights can you draw about their behavior?

SELECT

c.CustomerID,

c.Name,

COUNT(b.BookingID) AS TotalCompletedBookings

FROM

Customers c

JOIN

Bookings b **ON** c.CustomerID = b.CustomerID

WHERE

b.Status = 'Completed'

GROUP BY

c.CustomerID, c.Name

ORDER BY

TotalCompletedBookings **DESC**;

OUTPUT:

Result Grid	Filter Rows:	Export:
CustomerID	Name	TotalCompletedBookings
1	Raj Sharma	11
2	Anita Singh	9
3	Mohit Verma	9
4	Priya Patel	7
5	Amit Kumar	5
6	Sneha Joshi	5
7	Vikas Mehta	4
9	Rohan Das	3
10	Kiran Yadav	2
8	Nisha Gupta	2
11	Deepak Nair	1
12	Meera Iyer	1
14	Shreya Rao	1
15	Kunal Sethi	1

Insights

1. Top customers have high engagement, showing loyalty and frequent usage.
2. Their ride patterns and locations can be targeted for promotions or premium subscriptions.
3. If their cancellation rate is low, they are stable revenue generators.
4. Retention strategies (e.g. loyalty discounts, priority bookings) can be designed for them.
5. For churn prediction, customers with declining completed bookings over time should be flagged.

2. Find customers who have canceled more than 30% of their total bookings. What could be the reason for frequent cancellations?

SELECT

c.CustomerID,

c.Name,

COUNT(b.BookingID) AS TotalBookings,

SUM(CASE WHEN b.Status = 'Cancelled' THEN 1 ELSE 0 END) AS CancelledBookings,

**ROUND((SUM(CASE WHEN b.Status = 'Cancelled' THEN 1 ELSE 0 END) * 100.0) /
COUNT(b.BookingID), 2) AS CancellationPercentage**

FROM

Customers c

JOIN

Bookings b **ON** c.CustomerID = b.CustomerID

GROUP BY

c.CustomerID, c.Name

HAVING

**((SUM(CASE WHEN b.Status = 'Cancelled' THEN 1 ELSE 0 END) * 100.0) /
COUNT(b.BookingID)) > 30**

ORDER BY

CancellationPercentage **DESC**;

OUTPUT:

Result Grid

Filter Rows:

Export:

Wrap Cell Content:

	CustomerID	Name	TotalBookings	CancelledBookings	CancellationPercentage
▶	13	Arjun Reddy	1	1	100.00
	17	Naveen Bhat	1	1	100.00
	8	Nisha Gupta	4	2	50.00
	10	Kiran Yadav	3	1	33.33

Insights

Frequent cancellations (>30%) can indicate:

1. Change in travel plans or scheduling issues
2. Dissatisfaction with cab service (e.g. delays, driver behavior)
3. Price sensitivity if cheaper alternatives exist
4. Testing behavior (booking multiple cabs and choosing one)
5. Action items for business:
6. Analyze cancellation reasons (CancellationReason column) for patterns.
7. Reach out to frequent cancellers for feedback surveys.
8. Implement cancellation fees or stricter policies if cancellations are unjustified.
9. Offer incentives (e.g. "complete your next booking for ₹50 off") to encourage usage.

3. Determine the busiest day of the week for bookings. How can the company optimize cab availability on peak days?

SELECT

DAYNAME(BookingTime) **AS** DayOfWeek,

COUNT(*) **AS** TotalBookings

FROM

Bookings



GROUP BY

DayOfWeek

ORDER BY

TotalBookings **DESC**;

OUTPUT:

Result Grid   Filter Rows: <input type="text"/>		
	DayOfWeek	TotalBookings
▶	Tuesday	14
	Wednesday	14
	Thursday	12
	Friday	12
	Saturday	12
	Sunday	11
	Monday	11

Insights

The company's peak demand day is Friday (hypothetical here).

Actionable strategies:

1. Increase active driver count on peak days to reduce wait time.
2. Offer surge pricing or incentives to drivers to remain online during busy hours.
3. Run promotional campaigns to spread out bookings to non-peak days.
4. Schedule vehicle maintenance on low-demand days (e.g. Tuesday) to maximize availability on peak days.

Driver Performance & Efficiency:

1. Identify drivers who have received an average rating below 3.0 in the past three months. What strategies can be implemented to improve their performance?

SELECT

d.DriverID,

d.Name,

ROUND(AVG(f.Rating), 2) **AS** AverageRating

FROM

Drivers d

JOIN

Feedback f **ON** d.DriverID = f.DriverID

WHERE

f.FeedbackDate >= **DATE_SUB**(CURDATE(), INTERVAL 3 MONTH)

GROUP BY

d.DriverID, d.Name



HAVING

AverageRating < 3.0

ORDER BY

AverageRating **ASC**;

OUTPUT

Result Grid   Filter Rows:			
	DriverID	Name	AverageRating
▶	22	Vivek Oberoi	2.00
	7	Ritika Iyer	2.50
	13	Bhavna Tiwari	2.50

Insights & Strategies

Reasons for low ratings may include:

1. Poor driving behavior (rash, unsafe)
2. Unprofessional attitude (rude, late arrivals)
3. Vehicle cleanliness issues

Strategies to improve performance:

1. Conduct one-on-one performance reviews with these drivers.
2. Provide targeted training in customer service and safety standards.
3. Introduce a mentorship program pairing low-rated drivers with high performers.
4. Monitor their upcoming trips with real-time feedback alerts.
5. Implement incentive-based performance improvement plans to motivate behavior change.

2. Find the top 5 drivers who have completed the longest trips in terms of distance.
What does this say about their working patterns?

SELECT

d.DriverID,

d.Name,

MAX(t.DistanceKM) **AS** LongestTripKM

FROM

Drivers d

JOIN

Bookings b **ON** d.DriverID = b.DriverID

JOIN

TripDetails t **ON** b.BookingID = t.BookingID

WHERE

b.Status = 'Completed'

GROUP BY

d.DriverID, d.Name



ORDER BY

LongestTripKM **DESC**

LIMIT 5;

OUTPUT

Result Grid

  Filter Rows:

	DriverID	Name	LongestTripKM
▶	20	Ajay Dev	16
	19	Simran Gill	15
	18	Ravi Pillai	14
	3	Aakash Gupta	12
	5	Pankaj Verma	12

Insights

These drivers are more willing to accept long-distance bookings, possibly indicating:

1. Preference for higher fares per trip
2. Routes aligned with their operating area
3. Comfort with inter-zone trips or highway driving

Operational considerations:

1. Long trips may result in fewer total trips per day (affecting utilization %).
2. Drivers covering long distances could be targeted for intercity or premium services.

Strategic suggestions:

1. Incentivize long-trip drivers during off-peak hours to balance supply.
2. Analyze if they are consistently working specific corridors for optimized dispatch planning.

3. Identify drivers with a high percentage of canceled trips. Could this indicate driver unreliability?

SELECT

d.DriverID,

d.Name,

COUNT(b.BookingID) AS TotalTrips,

SUM(CASE WHEN b.Status = 'Cancelled' THEN 1 ELSE 0 END) AS CancelledTrips,

**ROUND(SUM(CASE WHEN b.Status = 'Cancelled' THEN 1 ELSE 0 END) * 100.0 /
COUNT(b.BookingID), 2) AS CancellationPercentage**

FROM

Drivers d

JOIN

Bookings b ON d.DriverID = b.DriverID

GROUP BY

d.DriverID, d.Name

HAVING

CancellationPercentage > 30

ORDER BY

CancellationPercentage **DESC**;

OUTPUT

Result Grid

Filter Rows:

Export:

Wrap Cell Content:

	DriverID	Name	TotalTrips	CancelledTrips	CancellationPercentage
▶	9	Snehal Joshi	3	2	66.67
	27	Lavanya Menon	2	1	50.00
	1	Rahul Mehta	3	1	33.33
	3	Aakash Gupta	3	1	33.33
	4	Neeraj Singh	3	1	33.33
	6	Harsha Nair	3	1	33.33
	11	Megha Shah	3	1	33.33
	13	Bhavna Tiwari	3	1	33.33
	15	Komal Mishra	3	1	33.33
	17	Anushka Sen	3	1	33.33
	20	Ajay Dev	3	1	33.33
	22	Vivek Oberoi	3	1	33.33
	25	Swati Chauhan	3	1	33.33

Insights

A high driver cancellation % can indicate:

1. Poor reliability: driver often rejects trips after accepting them.
2. Operational problems: vehicle issues, scheduling conflicts.
3. Intentional cancellations: driver cancels short fares for better-paying rides.
4. Route preference: refusing trips to certain areas.

Revenue & Business Metrics

1. Calculate the total revenue generated by completed bookings in the last 6 months.
How has the revenue trend changed over time?

```
SELECT
    DATE_FORMAT(t.EndTime, '%Y-%m') AS Month,
    SUM(t.Fare) AS TotalRevenue
FROM
    TripDetails t
JOIN
    Bookings b ON t.BookingID = b.BookingID
WHERE
    b.Status = 'Completed'
    AND t.EndTime >= DATE_SUB(CURDATE(), INTERVAL 6 MONTH)
GROUP BY
    Month
ORDER BY
    Month;
```

OUTPUT

Result Grid			Filter Rows:
	Month	TotalRevenue	
▶	2025-05	4720	
	2025-06	1360	

Insights

Revenue trend analysis:

Identify months with peak revenue (e.g. June ₹67,800)

Investigate dip months (e.g. July ₹45,000) to determine causes:

1. Seasonal variation?
2. Driver availability issues?
3. Competitor promotions?

Strategic business decisions:

1. Increase marketing campaigns in low revenue months.
2. Optimize driver incentives during peak demand periods to maximize fulfilled trips.
3. Use trends for financial forecasting and resource planning.

2. Identify the top 3 most frequently travelled routes based on PickupLocation and DropoffLocation. Should the company allocate more cabs to these routes?

SELECT

b.PickupLocation,

b.DropoffLocation,

COUNT(*) AS TripCount

FROM

Bookings b

WHERE

b.Status = 'Completed'

GROUP BY

b.PickupLocation,



b.DropoffLocation

ORDER BY

TripCount **DESC**

LIMIT 3;

OUTPUT

Result Grid   Filter Rows: <input type="text"/>			
	PickupLocation	DropoffLocation	TripCount
▶	Kurla	Powai	4
	Thane	Mulund	4
	Andheri	Bandra	3

Insights

These top 3 routes have highest demand, indicating:

1. Regular commuters (office routes)
2. Popular city connectivity corridors

Strategic recommendations:

1. Allocate more cabs to these routes during peak hours to reduce waiting times.
2. Consider dynamic pricing (surge) during heavy demand windows.
3. Deploy premium cab options if routes cater to higher-income zones.
4. Analyse trip timings to further optimize driver dispatch schedules.

3. Determine if higher-rated drivers tend to complete more trips and earn higher fares. Is there a direct correlation between driver ratings and earnings?

SELECT

d.DriverID,
d.Name,
d.Rating **AS** DriverRating,
COUNT(t.TripID) **AS** TotalTrips,
SUM(t.Fare) **AS** TotalEarnings,
ROUND(**AVG**(t.Fare), 2) **AS** AverageFarePerTrip

FROM

Drivers d

JOIN

Bookings b **ON** d.DriverID = b.DriverID

JOIN

TripDetails t **ON** b.BookingID = t.BookingID

WHERE

b.Status = 'Completed'

GROUP BY



d.DriverID, d.Name, d.Rating


ORDER BY


d.Rating **DESC**;

OUTPUT

Result Grid

  Filter Rows:

Export: 

Wrap Cell Content: 

	DriverID	Name	DriverRating	TotalTrips	TotalEarnings	AverageFarePerTrip
▶	15	Komal Mishra	4.9	1	220	220
	6	Harsha Nair	4.8	2	240	120
	24	Rohit Sharma	4.7	1	90	90
	20	Ajay Dev	4.6	2	440	220
	1	Rahul Mehta	4.5	2	270	135
	18	Ravi Pillai	4.4	2	440	220
	14	Nikhil Pandey	4.3	1	120	120
	3	Aakash Gupta	4.2	1	240	240
	10	Vivek Rana	4.1	2	330	165
	27	Lavanya Menon	4	1	80	80
	28	Pranav Joshi	3.9	1	70	70
	9	Snehal Joshi	3.9	1	130	130
	2	Sunita Reddy	3.8	2	330	165
	11	Menha Shah	3.7	1	140	140

Insights

Possible observed trends:

High-rated drivers (e.g. 4.8 – 4.9) tend to have more trips and higher earnings.

Low-rated drivers may:

Have fewer trips assigned (due to customer ratings/complaints).

Prefer short trips or frequently cancel, reducing earnings.

Correlation analysis:

While the above table suggests a positive correlation between ratings and earnings, statistical correlation (Pearson or Spearman) analysis is required for confirmation in Python or advanced SQL.

Operational Efficiency & Optimization

1. Analyze the average waiting time (difference between booking time and trip start time) for different pickup locations. How can this be optimized to reduce delays.

SELECT

b.PickupLocation,

ROUND(AVG(TIMESTAMPDIFF(MINUTE, b.BookingTime, t.StartTime)), 2) AS
AverageWaitingTimeMinutes

FROM

Bookings b

JOIN

TripDetails t **ON** b.BookingID = t.BookingID

WHERE

b.Status = 'Completed'

GROUP BY

b.PickupLocation

ORDER BY

AverageWaitingTimeMinutes **DESC**;

OUTPUT

Result Grid		Filter Rows:
	PickupLocation	AverageWaitingTimeMinutes
▶	Vashi	11.00
	Chembur	9.50
	Thane	9.00
	Borivali	8.00
	Kurla	7.50
	Bandra	7.50
	Andheri	7.22
	Dadar	7.00
	Malad	3.00

Insights

High waiting time locations (e.g. Vashi, Chembur):

Possible Causes:

Low driver density in these areas

Longer distances between cabs and pickup points

Traffic congestion delaying driver arrival

Business Impact:

Increased customer dissatisfaction → potential churn

Reduced trip completions if customers cancel due to delays

Low waiting time areas (e.g. Dadar, Malad):

Well-covered by drivers

Higher trip turnover → good operational efficiency

2. Identify the most common reasons for trip cancellations from customer feedback.
What actions can be taken to reduce cancellations?

SELECT

CancellationReason,

COUNT(*) AS CancellationCount

FROM

Bookings

WHERE

Status = 'Cancelled'



GROUP BY

CancellationReason

ORDER BY

CancellationCount **DESC**;

OUTPUT

Result Grid   Filter Rows: <input type="text"/>		
	CancellationReason	CancellationCount
▶	Changed plan	5
	Alternate transport	4
	Driver delayed	3
	No show	2

Insights

Most common reasons:

Driver delayed: Operational inefficiency or poor driver punctuality.

Changed plan: Customer-driven; unavoidable to some extent.

Alternate transport: Indicates competitor switching or dissatisfaction with service.

No show: Driver arrives but customer cancels, possibly due to long waiting time.

3. Find out whether shorter trips (low-distance) contribute significantly to revenue.
Should the company encourage more short-distance rides?

SELECT

CASE

WHEN t.DistanceKM <= 5 **THEN** 'Short (<=5 km)'

WHEN t.DistanceKM **BETWEEN** 5.01 **AND** 10 **THEN** 'Medium (5-10 km)'

ELSE 'Long (>10 km)'

END AS TripCategory,

COUNT(*) **AS** TripCount,

SUM(t.Fare) **AS** TotalRevenue,

ROUND(**AVG**(t.Fare), 2) **AS** AverageFare

FROM

TripDetails t

JOIN

Bookings b **ON** t.BookingID = b.BookingID

WHERE

b.Status = 'Completed'

GROUP BY

TripCategory

ORDER BY

TotalRevenue **DESC**;

OUTPUT

Result Grid		Filter Rows:		Export:
	TripCategory	TripCount	TotalRevenue	AverageFare
▶	Medium (5-10 km)	24	3630	151.25
	Long (>10 km)	7	1840	262.86
	Short (<=5 km)	7	610	87.14

Insights

Short trips generate the highest total revenue, despite lower per-trip fares, due to high volume.

Long trips have higher average fares but lower frequency, contributing less to overall revenue.

Comparative & Predictive Analysis

1. Compare the revenue generated from 'Sedan' and 'SUV' cabs. Should the company invest more in a particular vehicle type?

SELECT

c.VehicleType,

COUNT(t.TripID) **AS** TotalTrips,

SUM(t.Fare) **AS** TotalRevenue,

ROUND(**AVG**(t.Fare), 2) **AS** AverageFarePerTrip

FROM

Cabs c

JOIN

Bookings b **ON** c.CabID = b.CabID

JOIN

TripDetails t **ON** b.BookingID = t.BookingID

WHERE

b.Status = 'Completed'

GROUP BY


c.VehicleType

ORDER BY

TotalRevenue **DESC**;


OUTPUT

Result Grid



Filter Rows:

Export:



	VehicleType	TotalTrips	TotalRevenue	AverageFarePerTrip
▶	Sedan	12	2060	171.67
	SUV	13	2050	157.69
	Hatchback	13	1970	151.54

Insights

Sedans generate higher total revenue due to higher trip volume, despite lower per-trip fares.

SUVs have higher average fares per trip, indicating premium pricing or long-distance preference.

2. Predict which customers are likely to stop using the service based on their last booking date and frequency of rides. How can customer retention be improved?

SELECT

c.CustomerID,

c.Name,

COUNT(b.BookingID) **AS** TotalBookings,

MAX(b.BookingTime) **AS** LastBookingDate,

DATEDIFF(**CURDATE**(), **MAX**(b.BookingTime)) **AS** DaysSinceLastBooking

FROM

Customers c

LEFT JOIN

Bookings b **ON** c.CustomerID = b.CustomerID

GROUP BY

c.CustomerID, c.Name

ORDER BY

DaysSinceLastBooking **DESC**, TotalBookings **ASC**;

OUTPUT

Result Grid		Filter Rows:		Export:	Wrap Cell Content:
	CustomerID	Name	TotalBookings	LastBookingDate	DaysSinceLastBooking
▶	6	Sneha Joshi	6	2025-06-01 16:20:00	50
	7	Vikas Mehta	5	2025-06-05 11:30:00	46
	8	Nisha Gupta	4	2025-06-08 15:00:00	43
	9	Rohan Das	4	2025-06-11 10:20:00	40
	10	Kiran Yadav	3	2025-06-13 12:40:00	38
	11	Deepak Nair	1	2025-06-14 13:50:00	37
	12	Meera Iyer	1	2025-06-15 15:00:00	36
	13	Arjun Reddy	1	2025-06-16 16:10:00	35
	14	Shreya Rao	1	2025-06-17 17:20:00	34
	15	Kunal Sethi	1	2025-06-18 08:30:00	33
	16	Riya Jain	1	2025-06-19 09:40:00	32
	17	Naveen Bhat	1	2025-06-20 10:50:00	31
	18	Pooja Pillai	1	2025-06-21 12:00:00	30
	19	Varun Saxena	1	2025-06-22 13:10:00	29

Insights

Customers with:

Low total bookings (low frequency)

Long duration since last ride (high recency days) are at high risk of churn.

3. Analyze whether weekend bookings differ significantly from weekday bookings.
Should the company introduce dynamic pricing based on demand?

SELECT

CASE

WHEN DAYOFWEEK(BookingTime) IN (1, 7) THEN 'Weekend' -- Sunday=1, Saturday=7 in MySQL

ELSE 'Weekday'

END AS DayType,

COUNT(*) AS TotalBookings,

ROUND(AVG(t.Fare), 2) AS AverageFare

FROM

Bookings b

JOIN

TripDetails t **ON** b.BookingID = t.BookingID



WHERE

b.Status = 'Completed'

GROUP BY

DayType;

OUTPUT

Result Grid   Filter Rows: <input type="text"/>			
	DayType	TotalBookings	AverageFare
▶	Weekday	27	155.19
	Weekend	11	171.82

Insights

Weekday bookings are higher in volume, indicating routine commuting demand.

Weekend bookings have higher average fares, suggesting longer leisure or family trips.

Conclusion

The analysis of the Cab Booking System demonstrates the critical role of data in optimizing transportation services. By examining customer behavior, driver performance, trip patterns, and revenue trends, valuable insights can be extracted to enhance operational efficiency and customer satisfaction. The study highlights how structured databases and SQL queries enable precise tracking of bookings, identification of high-performing drivers, detection of service bottlenecks, and evaluation of revenue growth.

Findings from the project can support strategic decisions such as resource allocation on peak days, improving driver training for low-rated performers, focusing on high-demand routes, and implementing customer retention strategies. Ultimately, this data-driven approach empowers cab service providers to improve reliability, maximize profitability, and remain competitive in a dynamic market.