Report on

RatioDrive: A database for a ride sharing service

April 29, 2025

Prepared By:

Shadman Tahmid Arib, 2312201642
Md. Azmine Amin Mormo, 2312318042
Md. Arif Ali Robin, 2312760042

Course Instructor:

Prof. Dr. Kamruddin Nur

Contents

List of Figures

List of Tables

Project Overview

RatioDrive is a MYSQL database centralizing all the necessary data about drivers, users, cars, trips, and financial transactions, this MySQL-based database is designed to manage the operations of a carpooling platform efficiently. It ensures safety and accountability by protecting user information, emergency contacts, trip histories, driver licenses, national identification documents, vehicle ownership records, and login histories. Model, registration number, year of manufacture, odometer readings, and other pertinent documents, such as tax and fitness expiration dates, are all arranged methodically in the vehicle information. Financial reporting is made simple by the system's monitoring of revenue sources, including commissions, payment methods, and trip earnings. The database facilitates easy querying, improves fleet management, and fosters operational transparency by systematically arranging all operational and financial data.

Contributions

Table 1: Team Member's Contributions

ID	Name	Tasks	Contribution
2312318042	Md. Azmine Amin Mormo	• Project Plan • Project Overview • Conceptual Diagram • Logical Diagram • Physical Diagram • Data Population • Query	30%
2312760042	Md. Arif Ali Robin	• Project Objective • Project description • Scope Statement • Query • Table Creation • Conclusion	40%
2312201642	Shadman Tahmid Arib	• Project Deliverable • Roles and Responsibilities • Project Schedule Grant Chart • Scope Statement • Key Milestone • Query • Conclusion	30%

1 Project Title

RatioDrive: A database for a ride sharing service

2 Project Description

A database designed for the purpose of effectively managing a carpooling service's operations, the MySQL database includes vital information for drivers, users, and the business. The system tracks the user's journey history with information such as dates, places, distance, length, fares, and ratings for the driver. It also keeps track of user information, contact information, and emergency contacts. Driving licenses, national identification numbers, automobile ownership, and login histories and emergency contacts are also stored so that the app can ensure the highest degree of security for its users, guaranteeing that all activities pertaining to them are recorded and monitored. The information of all the cars, including model, registration number, year of manufacture, and odometer readings, as well as crucial papers like tax and fitness expiration dates, are also managed the database. The system also keeps track of revenue information, including commissions, payment methods, earnings from each trip, and other financial transactions. The system improves the capacity of the carpooling service to effectively manage drivers, passengers, and vehicles by centralizing and organizing all of this data. It also provides clear reporting on trip activity, financial performance, and fleet management. The project architecture prioritizes efficient and simple querying of all private user and financial data, while providing a convenient way for drivers and users to track journeys, book rides, and leave reviews. The end goal is to enable an application that offers a productive and safe platform for overseeing the whole carpooling ecosystem, enhancing financial control, operational transparency, and user experience.

3 Project Objective

The objectives of this projects are:

- **Ensure Rider and Driver Safety:** The system will maintain detailed and accurate records of all users and drivers, including all sorts of identification documents and emergency contacts.
- **Efficient Financial Tracking:** The system tracks payment methods, trip fares, driver earnings, and commissions. By maintaining transparent and organized financial records and graphs, it will create an easy to use and understand interface for the users.
- **Optimize Vehicle and Staff Management:** The platform manages vehicle details such as registration numbers, model, maintenance records, and expiration dates of documents. It also includes staff or admin portals to assign roles, monitor activities to keep the user safe.
- **Enhance Operational Efficiency:** By integrating functionalities such as ride scheduling, user management, financial reporting, vehicle tracking into one single app, it smooths out the difficulty of a user in going from point A to point B
- **Ensure Data Privacy and Security:** Data privacy is a core priority. However these concerns will not be handled by the database and will be handled in the app building stage.
- **Promote Sustainable Transportation and Community Engagement:** By encouraging ride-sharing, the platform supports environmental sustainability and reduces traffic congestion. It also fosters a sense of community as people interact with each other as they go from place to place.
- **Apply Database and Software Development Principles:** This project demonstrates the effective use of database development conventions and technology and how it scales to the real world.

4 Project Scope

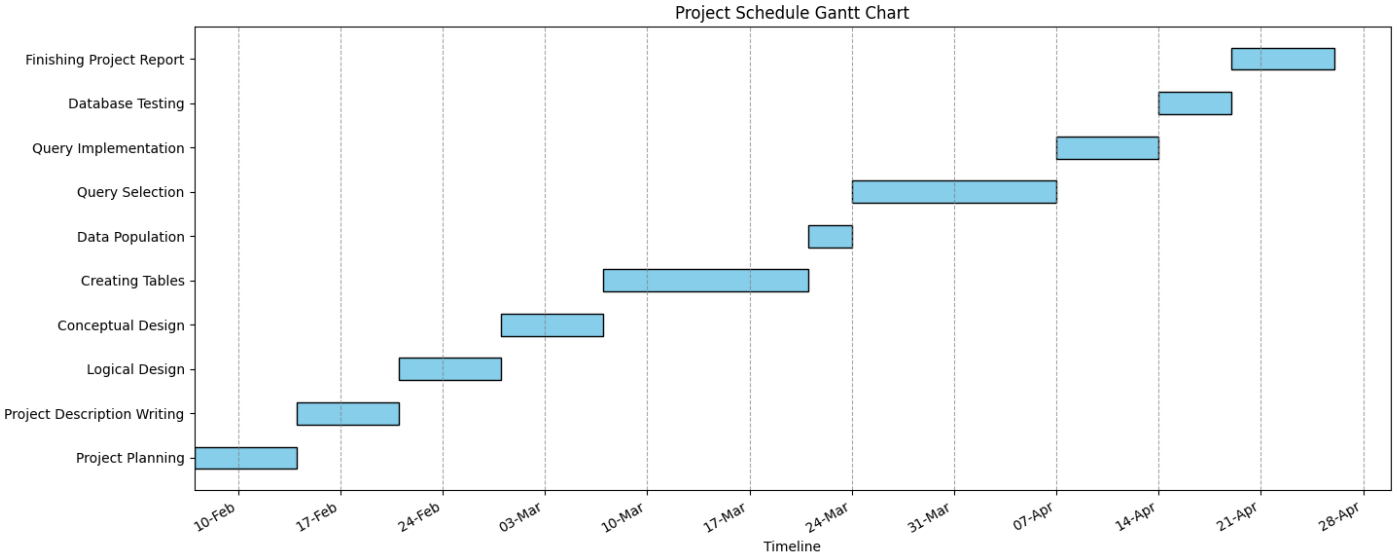
4.1 Scope Statement:

The goal of the **RatioDrive** project is to create a database that will be used to build a platform that will support a carpooling service's operations. Features such as vehicle management, payment and revenue tracking, trip booking and tracking modules, user and driver account management, and rating and review features will be included. Drivers and users will be able to maintain an easy-to-use profile on the system, which will include emergency contacts, driver's licenses, identification documents, and vehicle ownership. By matching drivers and passengers according to routes, preferences, and availability, it will make planning and booking rides easier. A dashboard will be created such that driver earnings, commissions, and payment options can be easily monitored. To guarantee operational compliance and vehicle safety, vehicle profiles, including registration information, maintenance logs, and important documents, will be managed through the platform. Users will be able to view past trips, fares, and driver ratings thanks to the system's maintenance of journey histories. Measures will be taken to prevent the use of the app strictly for professional taxi services. **RatioDrive** seeks to improve efficiency, guarantee passenger safety, and increase user trust by combining all the necessary carpooling functions into a single intuitive platform. The project will make use of contemporary technology to improve features over time, responding to the changing demands of the transportation sector and the carpooling community.

4.2 Roles and Responsibilities:

The Development Team is responsible for designing, developing, and testing the RatioDrive platform, ensuring that all core features are implemented effectively. The Project Manager oversees the project's execution, manages resource allocation, and ensures that the project is on track with deadlines. Stakeholders, including users, drivers, volunteers, and car repair garage owners, provide valuable feedback, participate in testing, and offer support throughout the project's lifecycle to ensure it meets the needs of the community.

4.3 Project Schedule:



5 Deliverable

- **User Portal:** An interface that allows the users to view the prices, book a car according to their budget, look into their driver and also even cancel the ride if need be.
- **Car Moderation Module:** Tools for viewing and managing a drivers cars details. It can allow customer support to remove and an algorithm to prevent a driver for signing up when certain criteria are not met. The cars information from here may also be used to determine the fare.
- **Admin Portal:** A centralized platform for the customer support crew to view all the Drivers and their cars within a certain region. It differs from the Moderation Module by restricting key details about the cars and is more finance centric where the information about drivers cars and how much they earn are briefly stored.
- **Offer Management Feature:** A system to plan, manage, and track offers that may allow a driver to get extra commision or a user to get a better deal for a limited period of time.
- **Payout Tracking System:** A feature that monitors and handles the payout to the drivers at the end of every week. It logs their earnings for the week and makes transfers all at once on every Sunday.
- **Driver AND User review Functionality:** Both passengers and drivers will be able to review each other which will be stored and visible for all of their further patrons to view.
- **Request Road Trip:** A future addition where a passenger may request road trips in advance and drivers may read through a board of such requests before they travel somewhere somewhere further than a short ride which is generally the intended purpose of the application.
- **Reporting and Analytics:** Dashboards and reports that provide real-time insights into how often a driver is driving, how much money they are making, how often a user is using the application. This data is then used to offer discounts, commissions or even penalize drivers that are trying to abuse the platform by offering taxi services.

6 Diagrams

6.1 Conceptual Design Diagram

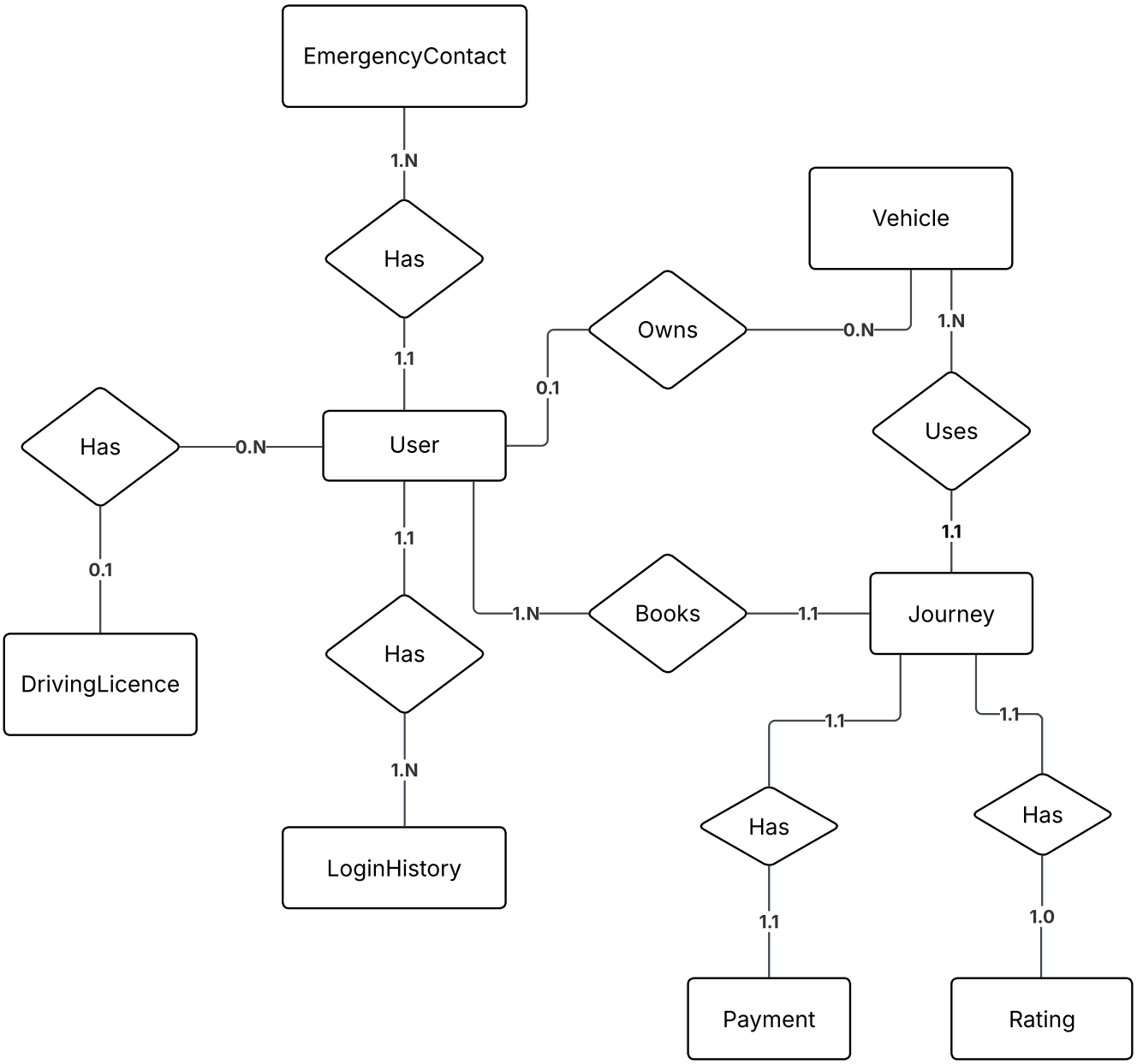


Figure 1: Conceptual design of the database

6.2 Logical Design Diagram

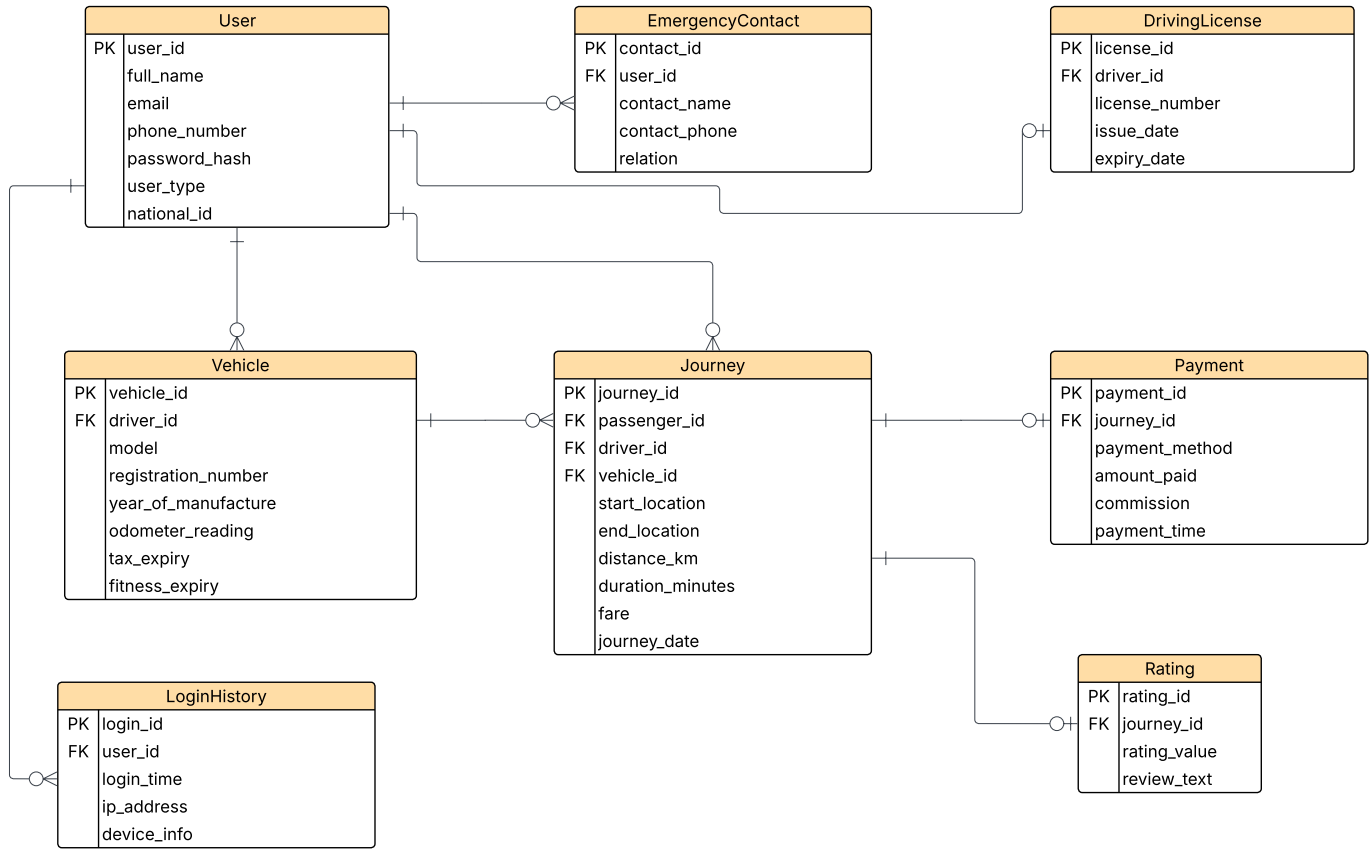


Figure 2: Logical design of the database

6.3 Physical Design Diagram

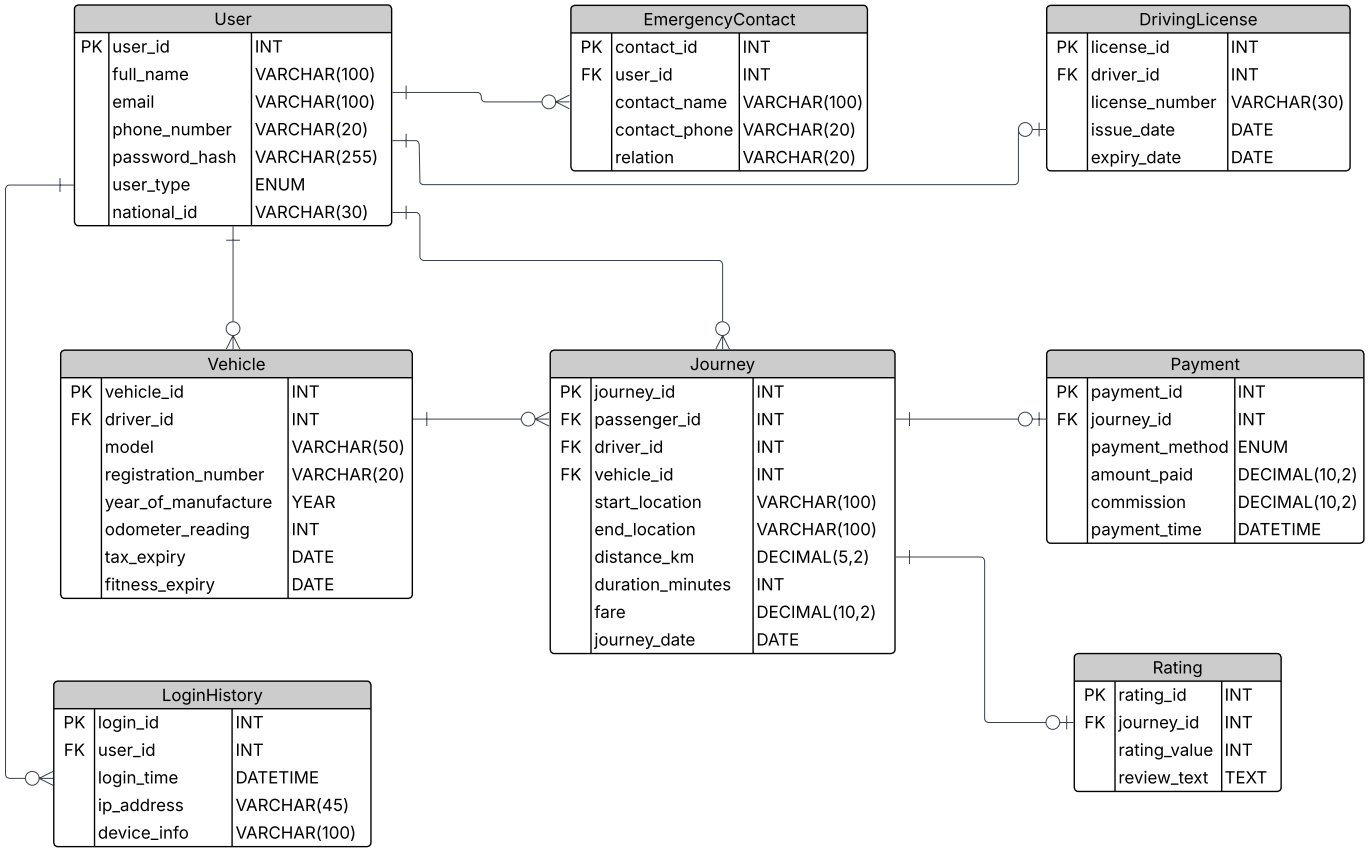


Figure 3: Physical design of the database

7 Table Creation

- **Users:**

```
    CREATE TABLE Users (
    user_id INT PRIMARY KEY AUTO_INCREMENT,
    full_name VARCHAR(100) NOT NULL,
    email VARCHAR(100) UNIQUE NOT NULL,
    phone_number VARCHAR(20),
    password_hash VARCHAR(255) NOT NULL,
    user_type ENUM('driver', 'passenger') NOT NULL,
    national_id VARCHAR(30) UNIQUE NOT NULL,
    created_at DATETIME DEFAULT CURRENT_TIMESTAMP
);

```

- **EmergencyContacts:**

```
    CREATE TABLE EmergencyContacts (
    contact_id INT PRIMARY KEY AUTO_INCREMENT,
    user_id INT NOT NULL,
    contact_name VARCHAR(100) NOT NULL,
    contact_phone VARCHAR(20) NOT NULL,
    relation VARCHAR(50),
    FOREIGN KEY (user_id) REFERENCES Users(user_id)
        ON DELETE CASCADE
);

```

- **Vehicles:**

```

    CREATE TABLE Vehicles (
    vehicle_id INT PRIMARY KEY AUTO_INCREMENT,
    driver_id INT NOT NULL,
    model VARCHAR(50),
    registration_number VARCHAR(20) UNIQUE NOT NULL,
    year_of_manufacture YEAR,
    odometer_reading INT,
    tax_expiry DATE,
    fitness_expiry DATE,
    FOREIGN KEY (driver_id) REFERENCES Users(user_id)
        ON DELETE CASCADE
);

```

- **DrivingLicenses:**

```
    CREATE TABLE DrivingLicenses (
    license_id INT PRIMARY KEY AUTO_INCREMENT,
    driver_id INT NOT NULL,
    license_number VARCHAR(30) UNIQUE NOT NULL,
    issue_date DATE,
    expiry_date DATE,
    FOREIGN KEY (driver_id) REFERENCES Users(user_id)
        ON DELETE CASCADE
);
```

- **Journeys:**

```
    CREATE TABLE Journeys (
    journey_id INT PRIMARY KEY AUTO_INCREMENT,
    passenger_id INT NOT NULL,
    driver_id INT NOT NULL,
    vehicle_id INT NOT NULL,
    start_location VARCHAR(100),
    end_location VARCHAR(100),
    distance_km DECIMAL(5,2),
    duration_minutes INT,
    fare DECIMAL(10,2),
    journey_date DATETIME,
    FOREIGN KEY (passenger_id) REFERENCES Users(user_id),
    FOREIGN KEY (driver_id) REFERENCES Users(user_id),
    FOREIGN KEY (vehicle_id) REFERENCES Vehicles(vehicle_id)
);

```

- **Ratings:**

```
        CREATE TABLE Ratings (
    rating_id INT PRIMARY KEY AUTO_INCREMENT,
    journey_id INT NOT NULL,
    rating_value INT CHECK (rating_value BETWEEN 1 AND 5),
    review_text TEXT,
    FOREIGN KEY (journey_id) REFERENCES Journeys(journey_id)
        ON DELETE CASCADE
);
```

- **Payments:**

```
        CREATE TABLE Payments (
    payment_id INT PRIMARY KEY AUTO_INCREMENT,
    journey_id INT NOT NULL,
    payment_method ENUM('cash', 'card', 'mobile_banking') NOT NULL,
    amount_paid DECIMAL(10,2),
    commission DECIMAL(10,2),
    payment_time DATETIME DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (journey_id) REFERENCES Journeys(journey_id)
        ON DELETE CASCADE
);

```

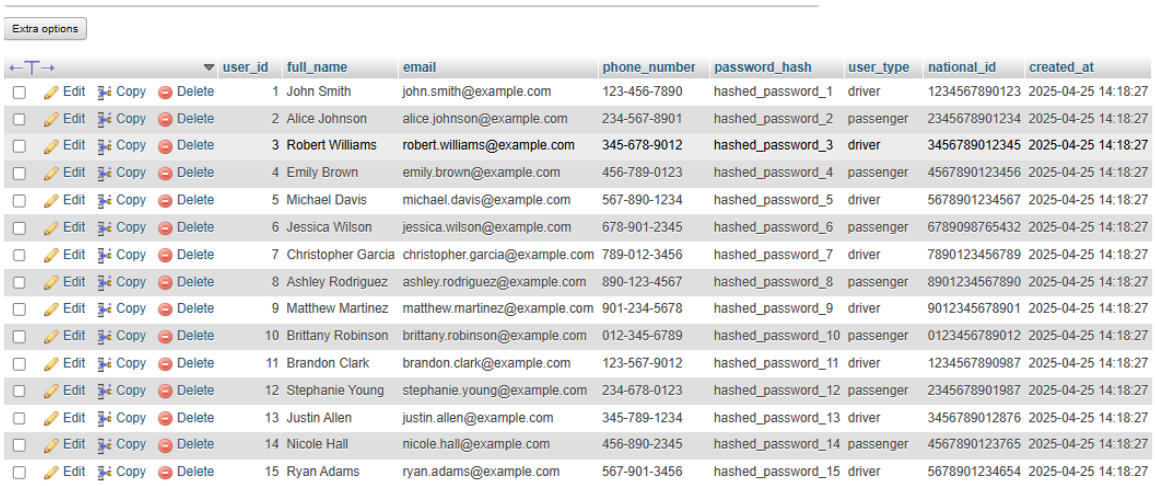
- **LoginHistory:**

```
       CREATE TABLE LoginHistory (
    login_id INT PRIMARY KEY AUTO_INCREMENT,
    user_id INT NOT NULL,
    login_time DATETIME DEFAULT CURRENT_TIMESTAMP,
    ip_address VARCHAR(45),
    device_info VARCHAR(100),
    FOREIGN KEY (user_id) REFERENCES Users(user_id)
        ON DELETE CASCADE
);
```

7.1 Data Population:

Users:

```
INSERT INTO Users (full_name, email, phone_number, password_hash, user_type,
    national_id) VALUES
('John Smith', 'john.smith@example.com', '123-456-7890', 'hashed_password_1', '
    driver', '1234567890123'),
('Alice Johnson', 'alice.johnson@example.com', '234-567-8901', '
    hashed_password_2', 'passenger', '2345678901234'),
('Robert Williams', 'robert.williams@example.com', '345-678-9012', '
    hashed_password_3', 'driver', '3456789012345'),
('Emily Brown', 'emily.brown@example.com', '456-789-0123', 'hashed_password_4',
    'passenger', '4567890123456'),
('Michael Davis', 'michael.davis@example.com', '567-890-1234', '
    hashed_password_5', 'driver', '5678901234567'),
('Jessica Wilson', 'jessica.wilson@example.com', '678-901-2345', '
    hashed_password_6', 'passenger', '6789098765432'),
('Christopher Garcia', 'christopher.garcia@example.com', '789-012-3456', '
    hashed_password_7', 'driver', '7890123456789'),
('Ashley Rodriguez', 'ashley.rodriguez@example.com', '890-123-4567', '
    hashed_password_8', 'passenger', '8901234567890'),
('Matthew Martinez', 'matthew.martinez@example.com', '901-234-5678', '
    hashed_password_9', 'driver', '9012345678901'),
('Brittany Robinson', 'brittany.robinson@example.com', '012-345-6789', '
    hashed_password_10', 'passenger', '0123456789012'),
('Brandon Clark', 'brandon.clark@example.com', '123-567-9012', '
    hashed_password_11', 'driver', '1234567890987'),
('Stephanie Young', 'stephanie.young@example.com', '234-678-0123', '
    hashed_password_12', 'passenger', '2345678901987'),
('Justin Allen', 'justin.allen@example.com', '345-789-1234', 'hashed_password_13
    ', 'driver', '3456789012876'),
('Nicole Hall', 'nicole.hall@example.com', '456-890-2345', 'hashed_password_14',
     'passenger', '4567890123765'),
('Ryan Adams', 'ryan.adams@example.com', '567-901-3456', 'hashed_password_15', '
    driver', '5678901234654');

```



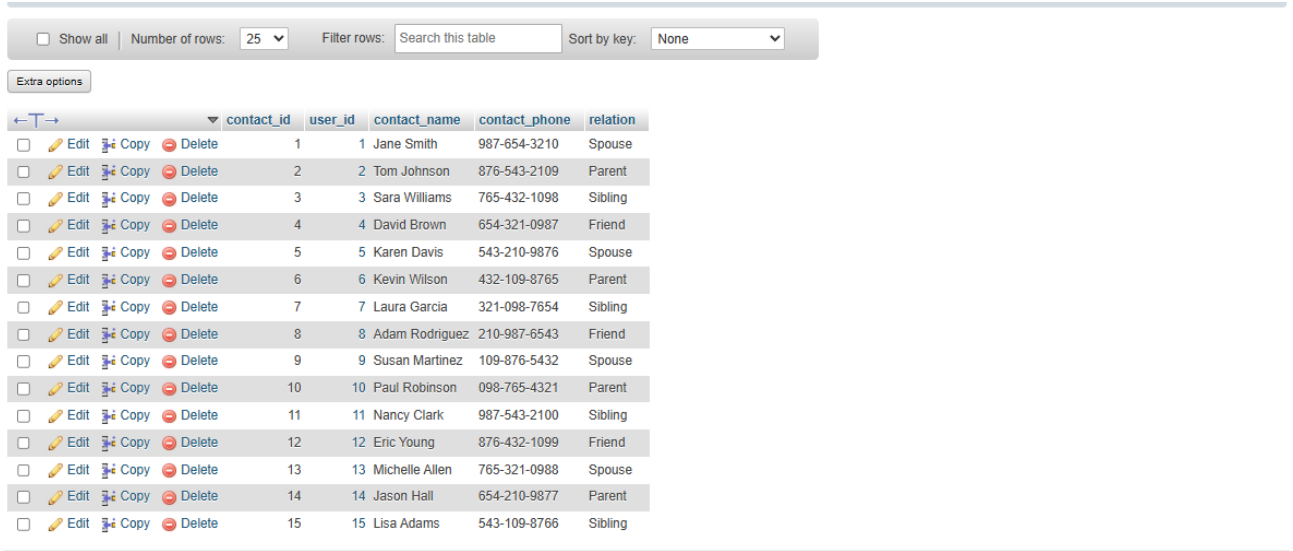
Extra options

	user_id	full_name	email	phone_number	password_hash	user_type	national_id	created_at
Edit Copy Delete	1	John Smith	john.smith@example.com	123-456-7890	hashed_password_1	driver	1234567890123	2025-04-25 14:18:27
Edit Copy Delete	2	Alice Johnson	alice.johnson@example.com	234-567-8901	hashed_password_2	passenger	2345678901234	2025-04-25 14:18:27
Edit Copy Delete	3	Robert Williams	robert.williams@example.com	345-678-9012	hashed_password_3	driver	3456789012345	2025-04-25 14:18:27
Edit Copy Delete	4	Emily Brown	emily.brown@example.com	456-789-0123	hashed_password_4	passenger	4567890123456	2025-04-25 14:18:27
Edit Copy Delete	5	Michael Davis	michael.davis@example.com	567-890-1234	hashed_password_5	driver	5678901234567	2025-04-25 14:18:27
Edit Copy Delete	6	Jessica Wilson	jessica.wilson@example.com	678-901-2345	hashed_password_6	passenger	6789098765432	2025-04-25 14:18:27
Edit Copy Delete	7	Christopher Garcia	christopher.garcia@example.com	789-012-3456	hashed_password_7	driver	7890123456789	2025-04-25 14:18:27
Edit Copy Delete	8	Ashley Rodriguez	ashley.rodriguez@example.com	890-123-4567	hashed_password_8	passenger	8901234567890	2025-04-25 14:18:27
Edit Copy Delete	9	Matthew Martinez	matthew.martinez@example.com	901-234-5678	hashed_password_9	driver	9012345678901	2025-04-25 14:18:27
Edit Copy Delete	10	Brittany Robinson	brittany.robinson@example.com	012-345-6789	hashed_password_10	passenger	0123456789012	2025-04-25 14:18:27
Edit Copy Delete	11	Brandon Clark	brandon.clark@example.com	123-567-9012	hashed_password_11	driver	1234567890987	2025-04-25 14:18:27
Edit Copy Delete	12	Stephanie Young	stephanie.young@example.com	234-678-0123	hashed_password_12	passenger	2345678901987	2025-04-25 14:18:27
Edit Copy Delete	13	Justin Allen	justin.allen@example.com	345-789-1234	hashed_password_13	driver	3456789012876	2025-04-25 14:18:27
Edit Copy Delete	14	Nicole Hall	nicole.hall@example.com	456-890-2345	hashed_password_14	passenger	4567890123765	2025-04-25 14:18:27
Edit Copy Delete	15	Ryan Adams	ryan.adams@example.com	567-901-3456	hashed_password_15	driver	5678901234654	2025-04-25 14:18:27

Figure 4: Users Table

EmergencyContacts:

```
1        INSERT INTO EmergencyContacts (user_id, contact_name, contact_phone,
     relation) VALUES
2  (1, 'Jane Smith', '987-654-3210', 'Spouse'),
3  (2, 'Tom Johnson', '876-543-2109', 'Parent'),
4  (3, 'Sara Williams', '765-432-1098', 'Sibling'),
5  (4, 'David Brown', '654-321-0987', 'Friend'),
6  (5, 'Karen Davis', '543-210-9876', 'Spouse'),
7  (6, 'Kevin Wilson', '432-109-8765', 'Parent'),
8  (7, 'Laura Garcia', '321-098-7654', 'Sibling'),
9  (8, 'Adam Rodriguez', '210-987-6543', 'Friend'),
10 (9, 'Susan Martinez', '109-876-5432', 'Spouse'),
11 (10, 'Paul Robinson', '098-765-4321', 'Parent'),
12 (11, 'Nancy Clark', '987-543-2100', 'Sibling'),
13 (12, 'Eric Young', '876-432-1099', 'Friend'),
14 (13, 'Michelle Allen', '765-321-0988', 'Spouse'),
15 (14, 'Jason Hall', '654-210-9877', 'Parent'),
16 (15, 'Lisa Adams', '543-109-8766', 'Sibling');
```



Show all | Number of rows: 25 | Filter rows: Search this table | Sort by key: None

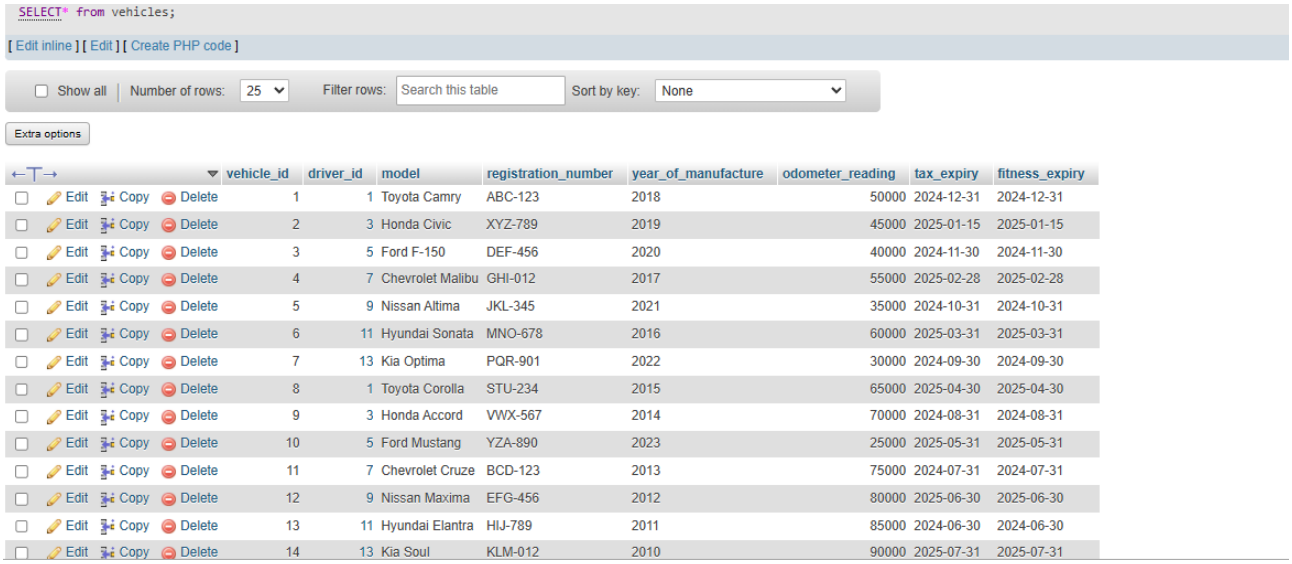
Extra options

	contact_id	user_id	contact_name	contact_phone	relation
Edit Copy Delete	1	1	Jane Smith	987-654-3210	Spouse
Edit Copy Delete	2	2	Tom Johnson	876-543-2109	Parent
Edit Copy Delete	3	3	Sara Williams	765-432-1098	Sibling
Edit Copy Delete	4	4	David Brown	654-321-0987	Friend
Edit Copy Delete	5	5	Karen Davis	543-210-9876	Spouse
Edit Copy Delete	6	6	Kevin Wilson	432-109-8765	Parent
Edit Copy Delete	7	7	Laura Garcia	321-098-7654	Sibling
Edit Copy Delete	8	8	Adam Rodriguez	210-987-6543	Friend
Edit Copy Delete	9	9	Susan Martinez	109-876-5432	Spouse
Edit Copy Delete	10	10	Paul Robinson	098-765-4321	Parent
Edit Copy Delete	11	11	Nancy Clark	987-543-2100	Sibling
Edit Copy Delete	12	12	Eric Young	876-432-1099	Friend
Edit Copy Delete	13	13	Michelle Allen	765-321-0988	Spouse
Edit Copy Delete	14	14	Jason Hall	654-210-9877	Parent
Edit Copy Delete	15	15	Lisa Adams	543-109-8766	Sibling

Figure 5: Emergency Contacts Table

Vehicles:

```
1       INSERT INTO Vehicles (driver_id, model, registration_number,
    year_of_manufacture, odometer_reading, tax_expiry, fitness_expiry) VALUES
2 (1, 'Toyota Camry', 'ABC-123', 2018, 50000, '2024-12-31', '2024-12-31'),
3 (3, 'Honda Civic', 'XYZ-789', 2019, 45000, '2025-01-15', '2025-01-15'),
4 (5, 'Ford F-150', 'DEF-456', 2020, 40000, '2024-11-30', '2024-11-30'),
5 (7, 'Chevrolet Malibu', 'GHI-012', 2017, 55000, '2025-02-28', '2025-02-28'),
6 (9, 'Nissan Altima', 'JKL-345', 2021, 35000, '2024-10-31', '2024-10-31'),
7 (11, 'Hyundai Sonata', 'MNO-678', 2016, 60000, '2025-03-31', '2025-03-31'),
8 (13, 'Kia Optima', 'PQR-901', 2022, 30000, '2024-09-30', '2024-09-30'),
9 (1, 'Toyota Corolla', 'STU-234', 2015, 65000, '2025-04-30', '2025-04-30'),
10 (3, 'Honda Accord', 'VWX-567', 2014, 70000, '2024-08-31', '2024-08-31'),
11 (5, 'Ford Mustang', 'YZA-890', 2023, 25000, '2025-05-31', '2025-05-31'),
12 (7, 'Chevrolet Cruze', 'BCD-123', 2013, 75000, '2024-07-31', '2024-07-31'),
13 (9, 'Nissan Maxima', 'EFG-456', 2012, 80000, '2025-06-30', '2025-06-30'),
14 (11, 'Hyundai Elantra', 'HIJ-789', 2011, 85000, '2024-06-30', '2024-06-30'),
15 (13, 'Kia Soul', 'KLM-012', 2010, 90000, '2025-07-31', '2025-07-31'),
16 (1, 'Toyota RAV4', 'NOP-345', 2009, 95000, '2024-05-31', '2024-05-31');
17
18
```



SELECT* from vehicles;

[Edit inline] [Edit] [Create PHP code]

Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

Extra options

←T→	vehicle_id	driver_id	model	registration_number	year_of_manufacture	odometer_reading	tax_expiry	fitness_expiry
Edit Copy Delete	1	1	Toyota Camry	ABC-123	2018	50000	2024-12-31	2024-12-31
Edit Copy Delete	2	3	Honda Civic	XYZ-789	2019	45000	2025-01-15	2025-01-15
Edit Copy Delete	3	5	Ford F-150	DEF-456	2020	40000	2024-11-30	2024-11-30
Edit Copy Delete	4	7	Chevrolet Malibu	GHI-012	2017	55000	2025-02-28	2025-02-28
Edit Copy Delete	5	9	Nissan Altima	JKL-345	2021	35000	2024-10-31	2024-10-31
Edit Copy Delete	6	11	Hyundai Sonata	MNO-678	2016	60000	2025-03-31	2025-03-31
Edit Copy Delete	7	13	Kia Optima	PQR-901	2022	30000	2024-09-30	2024-09-30
Edit Copy Delete	8	1	Toyota Corolla	STU-234	2015	65000	2025-04-30	2025-04-30
Edit Copy Delete	9	3	Honda Accord	VWX-567	2014	70000	2024-08-31	2024-08-31
Edit Copy Delete	10	5	Ford Mustang	YZA-890	2023	25000	2025-05-31	2025-05-31
Edit Copy Delete	11	7	Chevrolet Cruze	BCD-123	2013	75000	2024-07-31	2024-07-31
Edit Copy Delete	12	9	Nissan Maxima	EFG-456	2012	80000	2025-06-30	2025-06-30
Edit Copy Delete	13	11	Hyundai Elantra	HIJ-789	2011	85000	2024-06-30	2024-06-30
Edit Copy Delete	14	13	Kia Soul	KLM-012	2010	90000	2025-07-31	2025-07-31

Figure 6: Vehicles Table

Driving Licenses:

```
        INSERT INTO DrivingLicenses (driver_id, license_number, issue_date, expiry_date) VALUES
(1, 'DL12345', '2018-01-01', '2028-01-01'),
(3, 'DL67890', '2019-02-15', '2029-02-15'),
(5, 'DL11223', '2020-03-30', '2030-03-30'),
(7, 'DL44556', '2017-04-10', '2027-04-10'),
(9, 'DL77889', '2021-05-20', '2031-05-20'),
(11, 'DL22334', '2016-06-25', '2026-06-25'),
(13, 'DL55667', '2022-07-01', '2032-07-01'),
(1, 'DL88990', '2015-08-08', '2025-08-08'),
(3, 'DL33445', '2014-09-15', '2024-09-15'),
(5, 'DL66778', '2023-10-20', '2033-10-20'),
(7, 'DL99001', '2013-11-30', '2023-11-30'),
(9, 'DL44552', '2012-12-10', '2022-12-10'),
(11, 'DL77883', '2011-01-05', '2021-01-05'),
(13, 'DL22336', '2010-02-28', '2020-02-28'),
(1, 'DL55669', '2009-03-15', '2019-03-15');
```

	license_id	driver_id	license_number	issue_date	expiry_date
Edit Copy Delete	1	1	DL12345	2018-01-01	2028-01-01
Edit Copy Delete	2	3	DL67890	2019-02-15	2029-02-15
Edit Copy Delete	3	5	DL11223	2020-03-30	2030-03-30
Edit Copy Delete	4	7	DL44556	2017-04-10	2027-04-10
Edit Copy Delete	5	9	DL77889	2021-05-20	2031-05-20
Edit Copy Delete	6	11	DL22334	2016-06-25	2026-06-25
Edit Copy Delete	7	13	DL55667	2022-07-01	2032-07-01
Edit Copy Delete	8	1	DL88990	2015-08-08	2025-08-08
Edit Copy Delete	9	3	DL33445	2014-09-15	2024-09-15
Edit Copy Delete	10	5	DL66778	2023-10-20	2033-10-20
Edit Copy Delete	11	7	DL99001	2013-11-30	2023-11-30
Edit Copy Delete	12	9	DL44552	2012-12-10	2022-12-10
Edit Copy Delete	13	11	DL77883	2011-01-05	2021-01-05
Edit Copy Delete	14	13	DL22336	2010-02-28	2020-02-28
Edit Copy Delete	15	1	DL55669	2009-03-15	2019-03-15

Figure 7: Driving License Table

Journeys:

```
        INSERT INTO Journeys (passenger_id, driver_id, vehicle_id, start_location, end_location, distance_km, duration_minutes, fare, journey_date) VALUES
(2, 1, 1, 'Downtown A', 'Suburb B', 15.5, 30, 25.00, '2024-01-10 08:00:00'),
(4, 3, 2, 'Suburb C', 'Airport D', 25.0, 45, 40.00, '2024-01-10 09:30:00'),
(6, 5, 3, 'Residential E', 'Commercial F', 10.0, 20, 18.00, '2024-01-10 10:45:00'),
(8, 7, 4, 'Industrial G', 'Downtown H', 30.0, 50, 50.00, '2024-01-10 12:00:00'),
(10, 9, 5, 'Suburb I', 'Suburb J', 12.0, 25, 22.00, '2024-01-10 13:15:00'),
(12, 11, 6, 'Downtown K', 'Residential L', 18.0, 35, 30.00, '2024-01-10 14:30:00'),
(14, 13, 7, 'Commercial M', 'Airport N', 28.0, 48, 45.00, '2024-01-10 15:45:00'),
(2, 1, 8, 'Suburb O', 'Downtown P', 14.0, 28, 24.00, '2024-01-11 08:00:00'),
(4, 3, 9, 'Residential Q', 'Industrial R', 22.0, 40, 38.00, '2024-01-11 09:30:00'),
(6, 5, 10, 'Downtown S', 'Commercial T', 9.0, 18, 16.00, '2024-01-11 10:45:00'),
(8, 7, 11, 'Suburb U', 'Suburb V', 11.0, 23, 20.00, '2024-01-11 12:00:00'),
```

```
(10, 9, 12, 'Residential W', 'Downtown X', 17.0, 33, 29.00, '2024-01-11 13:15:00'),
(12, 11, 13, 'Commercial Y', 'Airport Z', 27.0, 46, 43.00, '2024-01-11 14:30:00'),
(14, 13, 14, 'Downtown AA', 'Suburb BB', 13.0, 26, 23.00, '2024-01-11 15:45:00'),
(2, 1, 15, 'Suburb CC', 'Residential DD', 21.0, 38, 36.00, '2024-01-11 17:00:00');

```

```
SELECT* from journeys;
```

[Edit inline] [Edit] [Create PHP code]

Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

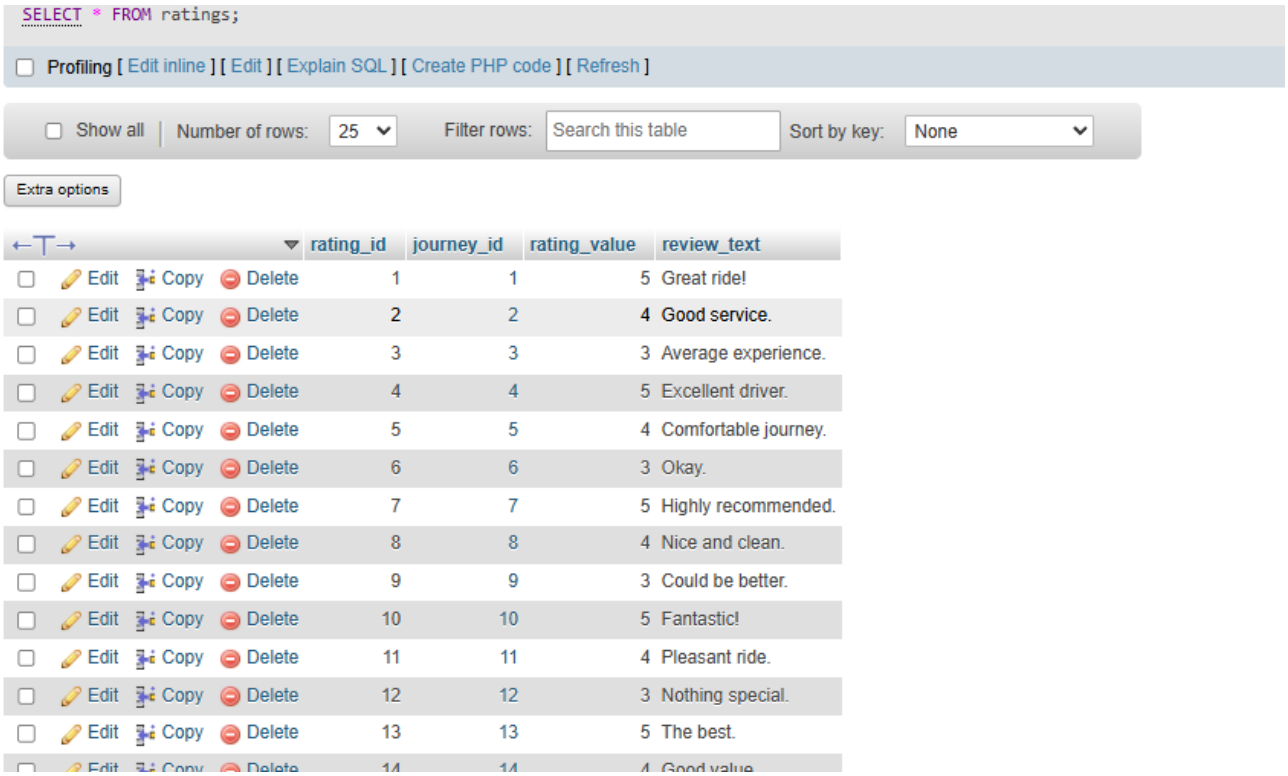
Extra options

	journey_id	passenger_id	driver_id	vehicle_id	start_location	end_location	distance_km	duration_minutes	fare	journey_date
Edit Copy Delete	1	2	1	1	Downtown A	Suburb B	15.50	30	25.00	2024-01-10 08:00:00
Edit Copy Delete	2	4	3	2	Suburb C	Airport D	25.00	45	40.00	2024-01-10 09:30:00
Edit Copy Delete	3	6	5	3	Residential E	Commercial F	10.00	20	18.00	2024-01-10 10:45:00
Edit Copy Delete	4	8	7	4	Industrial G	Downtown H	30.00	50	50.00	2024-01-10 12:00:00
Edit Copy Delete	5	10	9	5	Suburb I	Suburb J	12.00	25	22.00	2024-01-10 13:15:00
Edit Copy Delete	6	12	11	6	Downtown K	Residential L	18.00	35	30.00	2024-01-10 14:30:00
Edit Copy Delete	7	14	13	7	Commercial M	Airport N	28.00	48	45.00	2024-01-10 15:45:00
Edit Copy Delete	8	2	1	8	Suburb O	Downtown P	14.00	28	24.00	2024-01-11 08:00:00
Edit Copy Delete	9	4	3	9	Residential Q	Industrial R	22.00	40	38.00	2024-01-11 09:30:00
Edit Copy Delete	10	6	5	10	Downtown S	Commercial T	9.00	18	16.00	2024-01-11 10:45:00
Edit Copy Delete	11	8	7	11	Suburb U	Suburb V	11.00	23	20.00	2024-01-11 12:00:00
Edit Copy Delete	12	10	9	12	Residential W	Downtown X	17.00	33	29.00	2024-01-11 13:15:00
Edit Copy Delete	13	12	11	13	Commercial Y	Airport Z	27.00	46	43.00	2024-01-11 14:30:00
Edit Copy Delete	14	14	13	14	Downtown AA	Suburb BB	13.00	26	23.00	2024-01-11 15:45:00

Figure 8: Journeys Table

Ratings:

```
        INSERT INTO Ratings (journey_id, rating_value, review_text) VALUES
(1, 5, 'Great ride!'),
(2, 4, 'Good service.'),
(3, 3, 'Average experience.'),
(4, 5, 'Excellent driver.'),
(5, 4, 'Comfortable journey.'),
(6, 3, 'Okay.'),
(7, 5, 'Highly recommended.'),
(8, 4, 'Nice and clean.'),
(9, 3, 'Could be better.'),
(10, 5, 'Fantastic!'),
(11, 4, 'Pleasant ride.'),
(12, 3, 'Nothing special.'),
(13, 5, 'The best.'),
(14, 4, 'Good value.'),
(15, 3, 'Just fine.');
```

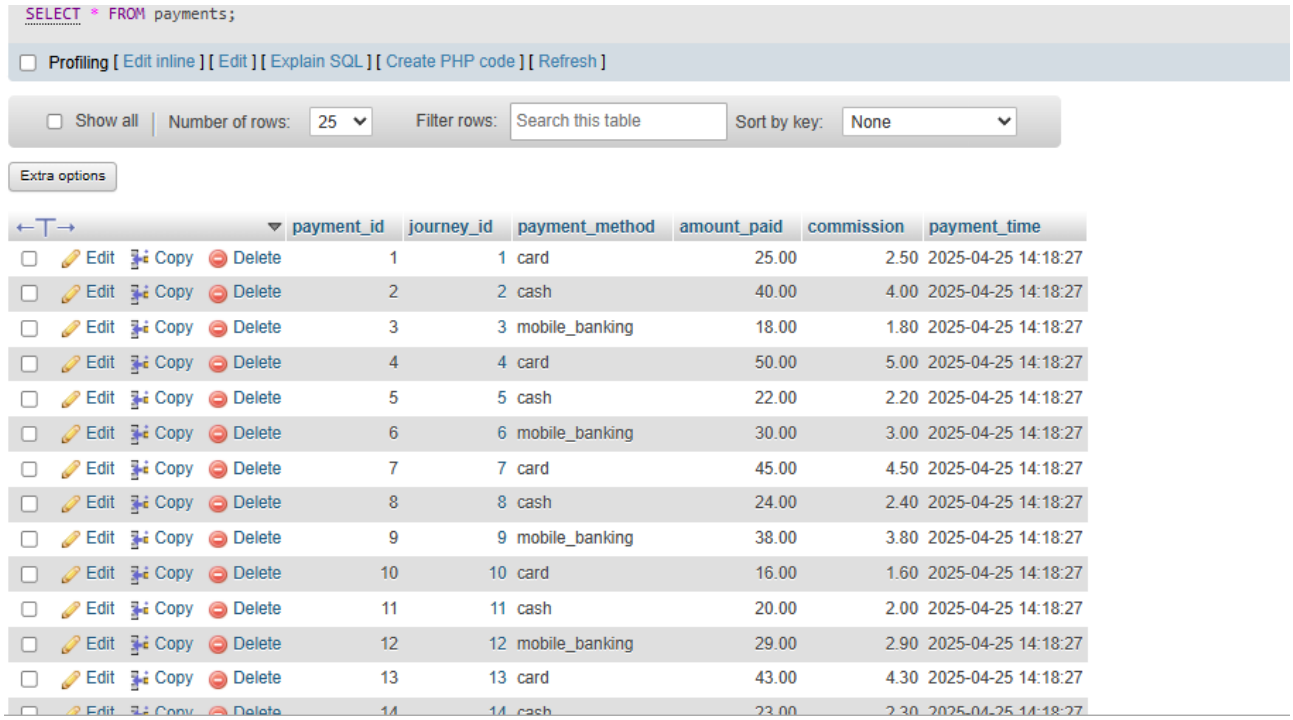


	rating_id	journey_id	rating_value	review_text
Edit Copy Delete	1	1	5	Great ride!
Edit Copy Delete	2	2	4	Good service.
Edit Copy Delete	3	3	3	Average experience.
Edit Copy Delete	4	4	5	Excellent driver.
Edit Copy Delete	5	5	4	Comfortable journey.
Edit Copy Delete	6	6	3	Okay.
Edit Copy Delete	7	7	5	Highly recommended.
Edit Copy Delete	8	8	4	Nice and clean.
Edit Copy Delete	9	9	3	Could be better.
Edit Copy Delete	10	10	5	Fantastic!
Edit Copy Delete	11	11	4	Pleasant ride.
Edit Copy Delete	12	12	3	Nothing special.
Edit Copy Delete	13	13	5	The best.
Edit Copy Delete	14	14	4	Good value

Figure 9: Ratings Table

Payments:

```
      INSERT INTO Payments (journey_id, payment_method, amount_paid, commission
    ) VALUES
(1, 'card', 25.00, 2.50),
(2, 'cash', 40.00, 4.00),
(3, 'mobile_banking', 18.00, 1.80),
(4, 'card', 50.00, 5.00),
(5, 'cash', 22.00, 2.20),
(6, 'mobile_banking', 30.00, 3.00),
(7, 'card', 45.00, 4.50),
(8, 'cash', 24.00, 2.40),
(9, 'mobile_banking', 38.00, 3.80),
(10, 'card', 16.00, 1.60),
(11, 'cash', 20.00, 2.00),
(12, 'mobile_banking', 29.00, 2.90),
(13, 'card', 43.00, 4.30),
(14, 'cash', 23.00, 2.30),
(15, 'mobile_banking', 36.00, 3.60);
```



```
SELECT * FROM payments;
```

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

Extra options

	payment_id	journey_id	payment_method	amount_paid	commission	payment_time
Edit Copy Delete	1	1	card	25.00	2.50	2025-04-25 14:18:27
Edit Copy Delete	2	2	cash	40.00	4.00	2025-04-25 14:18:27
Edit Copy Delete	3	3	mobile_banking	18.00	1.80	2025-04-25 14:18:27
Edit Copy Delete	4	4	card	50.00	5.00	2025-04-25 14:18:27
Edit Copy Delete	5	5	cash	22.00	2.20	2025-04-25 14:18:27
Edit Copy Delete	6	6	mobile_banking	30.00	3.00	2025-04-25 14:18:27
Edit Copy Delete	7	7	card	45.00	4.50	2025-04-25 14:18:27
Edit Copy Delete	8	8	cash	24.00	2.40	2025-04-25 14:18:27
Edit Copy Delete	9	9	mobile_banking	38.00	3.80	2025-04-25 14:18:27
Edit Copy Delete	10	10	card	16.00	1.60	2025-04-25 14:18:27
Edit Copy Delete	11	11	cash	20.00	2.00	2025-04-25 14:18:27
Edit Copy Delete	12	12	mobile_banking	29.00	2.90	2025-04-25 14:18:27
Edit Copy Delete	13	13	card	43.00	4.30	2025-04-25 14:18:27
Edit Copy Delete	14	14	cash	23.00	2.30	2025-04-25 14:18:27

Figure 10: Payments Table

LoginHistory:

```
1        INSERT INTO LoginHistory (user_id, login_time, ip_address, device_info)
    VALUES
2 (1, '2024-01-10 07:55:00', '192.168.1.100', 'Web browser'),
3 (2, '2024-01-10 09:25:00', '192.168.1.101', 'Mobile app'),
4 (3, '2024-01-10 10:40:00', '192.168.1.102', 'Web browser'),
5 (4, '2024-01-10 11:55:00', '192.168.1.103', 'Mobile app'),
6 (5, '2024-01-10 13:10:00', '192.168.1.104', 'Web browser'),
7 (6, '2024-01-10 14:25:00', '192.168.1.105', 'Mobile app'),
8 (7, '2024-01-10 15:40:00', '192.168.1.106', 'Web browser'),
9 (8, '2024-01-11 07:55:00', '192.168.1.107', 'Mobile app'),
10 (9, '2024-01-11 09:25:00', '192.168.1.108', 'Web browser'),
11 (10, '2024-01-11 10:40:00', '192.168.1.109', 'Mobile app'),
12 (11, '2024-01-11 11:55:00', '192.168.1.110', 'Web browser'),
13 (12, '2024-01-11 13:10:00', '192.168.1.111', 'Mobile app'),
14 (13, '2024-01-11 14:25:00', '192.168.1.112', 'Web browser'),
15 (14, '2024-01-11 15:40:00', '192.168.1.113', 'Mobile app'),
16 (15, '2024-01-11 16:55:00', '192.168.1.114', 'Web browser');
```

Extra options

	journey_id	passenger_id	driver_id	vehicle_id	start_location	end_location	distance_km	duration_minutes	fare	journey_date
Edit Copy Delete	1	2	1	1	Downtown A	Suburb B	15.50	30	25.00	2024-01-10 08:00:00
Edit Copy Delete	2	4	3	2	Suburb C	Airport D	25.00	45	40.00	2024-01-10 09:30:00
Edit Copy Delete	3	6	5	3	Residential E	Commercial F	10.00	20	18.00	2024-01-10 10:45:00
Edit Copy Delete	4	8	7	4	Industrial G	Downtown H	30.00	50	50.00	2024-01-10 12:00:00
Edit Copy Delete	5	10	9	5	Suburb I	Suburb J	12.00	25	22.00	2024-01-10 13:15:00
Edit Copy Delete	6	12	11	6	Downtown K	Residential L	18.00	35	30.00	2024-01-10 14:30:00
Edit Copy Delete	7	14	13	7	Commercial M	Airport N	28.00	48	45.00	2024-01-10 15:45:00
Edit Copy Delete	8	2	1	8	Suburb O	Downtown P	14.00	28	24.00	2024-01-11 08:00:00
Edit Copy Delete	9	4	3	9	Residential Q	Industrial R	22.00	40	38.00	2024-01-11 09:30:00
Edit Copy Delete	10	6	5	10	Downtown S	Commercial T	9.00	18	16.00	2024-01-11 10:45:00
Edit Copy Delete	11	8	7	11	Suburb U	Suburb V	11.00	23	20.00	2024-01-11 12:00:00
Edit Copy Delete	12	10	9	12	Residential W	Downtown X	17.00	33	29.00	2024-01-11 13:15:00
Edit Copy Delete	13	12	11	13	Commercial Y	Airport Z	27.00	46	43.00	2024-01-11 14:30:00
Edit Copy Delete	14	14	13	14	Downtown AA	Suburb BB	13.00	26	23.00	2024-01-11 15:45:00
Edit Copy Delete	15	2	1	15	Suburb CC	Residential DD	21.00	38	36.00	2024-01-11 17:00:00

Figure 11: Login History Table

8 Queries

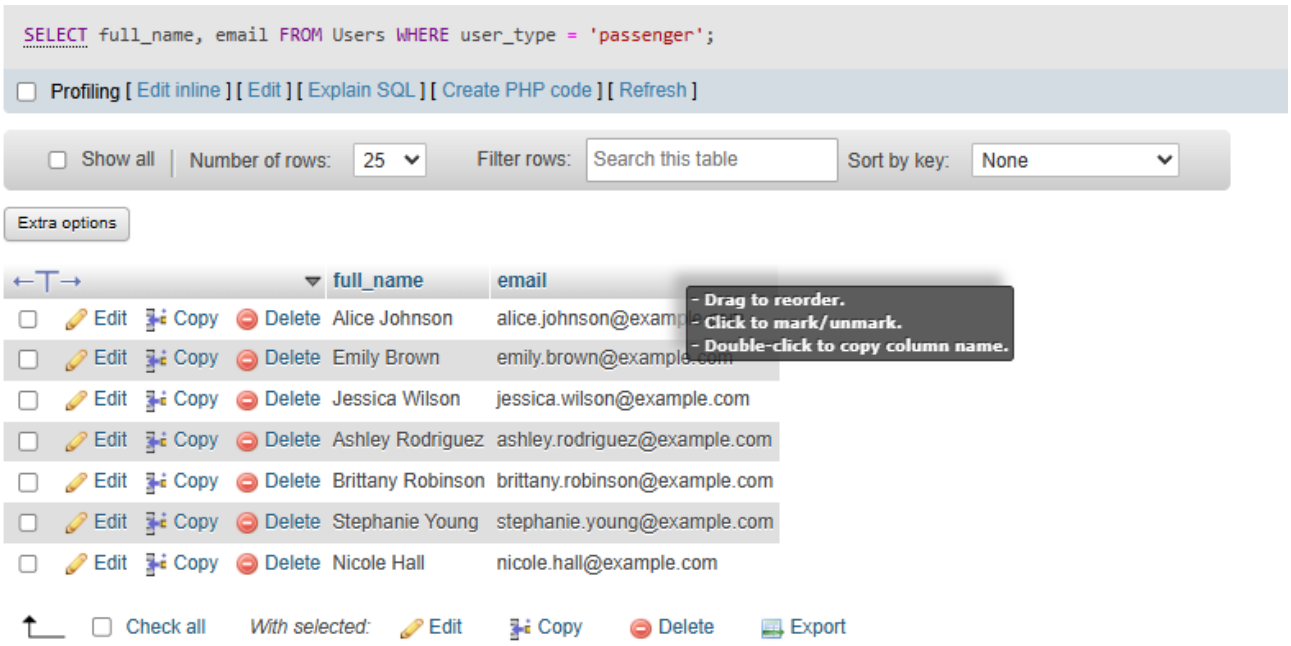
8.1 Query List

- Retrieve all passengers full names and their email addresses.
- List all drivers along with their vehicle models.
- Find all vehicles whose fitness expiry date is within the next 30 days.
- Count the number of journeys taken by each passenger.
- Find drivers who have not registered any vehicles yet.
- Retrieve journey details along with the driver's and passenger's names.
- List all emergency contacts for a given user.
- Retrieve all vehicles manufactured before 2015.
- Find the average rating received by each driver.
- List vehicles with expired tax or fitness certificates.
- Calculate total commission earned by the platform.
- Find the most popular payment method.
- List all journeys longer than 20 km.
- Count logins by device type.
- Find the busiest day for rides.
- List all emergency contacts for passengers.
- Calculate average fare per kilometer.
- Find drivers with multiple vehicles.
- Find the most recent login for each user.
- Calculate total distance traveled by each vehicle.
- List all 5-star ratings with their journey details.
- Calculate the average journey duration by driver.
- List all payments made via mobile banking with their journey details.
- Find drivers who haven't completed any journeys in the last month.
- Find passengers who spend more than average per journey.

8.2 SQL Query:

- **Retrieve all passengers full names and their email addresses**

```
        SELECT full_name, email
FROM Users
WHERE user_type = 'passenger';

```



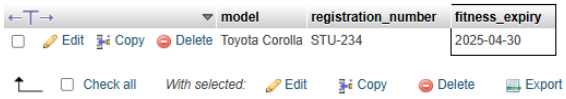
– List all drivers along with their vehicle models.

```sql
SELECT u.full_name, v.model
FROM Users u
JOIN Vehicles v ON u.user_id = v.driver_id;

```

Figure 12: Enter Caption

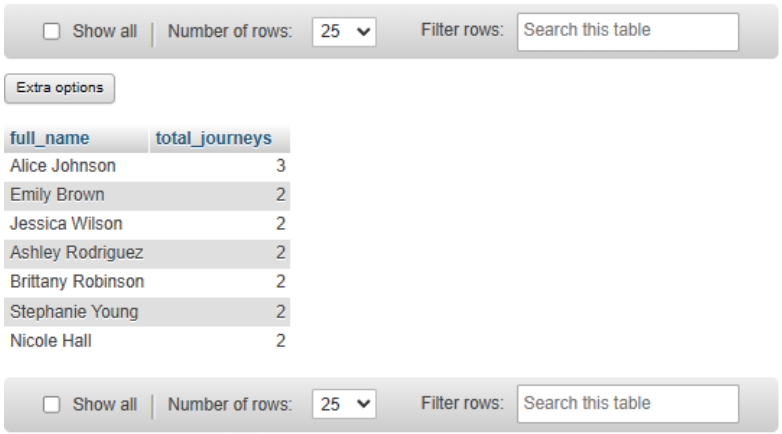
– Find all vehicles whose fitness expiry date is within the next 30 days.

```
    SELECT model, registration_number, fitness_expiry
FROM Vehicles
WHERE fitness_expiry BETWEEN CURDATE() AND DATE_ADD(CURDATE(), INTERVAL 30
    DAY);
```



– Count the number of journeys taken by each passenger.

```
    SELECT u.full_name, COUNT(j.journey_id) AS total_journeys
FROM Users u
JOIN Journeys j ON u.user_id = j.passenger_id
GROUP BY u.user_id;
```



– Find drivers who have not registered any vehicles yet.

```
    SELECT u.full_name
FROM Users u
LEFT JOIN Vehicles v ON u.user_id = v.driver_id
WHERE u.user_type = 'driver' AND v.vehicle_id IS NULL;
```

– Retrieve journey details along with the driver's and passenger's names.

```
    SELECT j.start_location, j.end_location, d.full_name AS driver, p.
    full_name AS passenger
FROM Journeys j
JOIN Users d ON j.driver_id = d.user_id
JOIN Users p ON j.passenger_id = p.user_id;
```

Extra options

start_location	end_location	driver	passenger
Downtown A	Suburb B	John Smith	Alice Johnson
Suburb C	Airport D	Robert Williams	Emily Brown
Residential E	Commercial F	Michael Davis	Jessica Wilson
Industrial G	Downtown H	Christopher Garcia	Ashley Rodriguez
Suburb I	Suburb J	Matthew Martinez	Brittany Robinson
Downtown K	Residential L	Brandon Clark	Stephanie Young
Commercial M	Airport N	Justin Allen	Nicole Hall
Suburb O	Downtown P	John Smith	Alice Johnson
Residential Q	Industrial R	Robert Williams	Emily Brown
Downtown S	Commercial T	Michael Davis	Jessica Wilson
Suburb U	Suburb V	Christopher Garcia	Ashley Rodriguez
Residential W	Downtown X	Matthew Martinez	Brittany Robinson
Commercial Y	Airport Z	Brandon Clark	Stephanie Young
Downtown AA	Suburb BB	Justin Allen	Nicole Hall
Suburb CC	Residential DD	John Smith	Alice Johnson

– List all emergency contacts for a given user.

```
   SELECT ec.contact_name, ec.contact_phone, ec.relation
FROM EmergencyContacts ec
WHERE ec.user_id = 1; -- Replace 1 with desired user_id
```

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

Show all | Number of rows: 25 | Filter rows: Search this table

Extra options

←T→	contact_name	contact_phone	relation
Edit Copy Delete	Jane Smith	987-654-3210	Spouse

Check all With selected: Edit Copy Delete Export

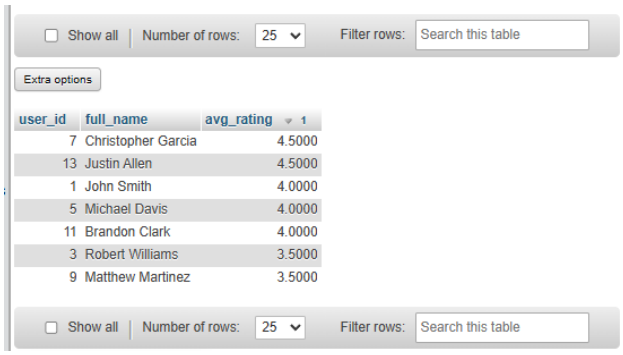
Show all | Number of rows: 25 | Filter rows: Search this table

– Retrieve all vehicles manufactured before 2015.

```
   SELECT model, registration_number, year_of_manufacture
FROM Vehicles
WHERE year_of_manufacture < 2015;
```

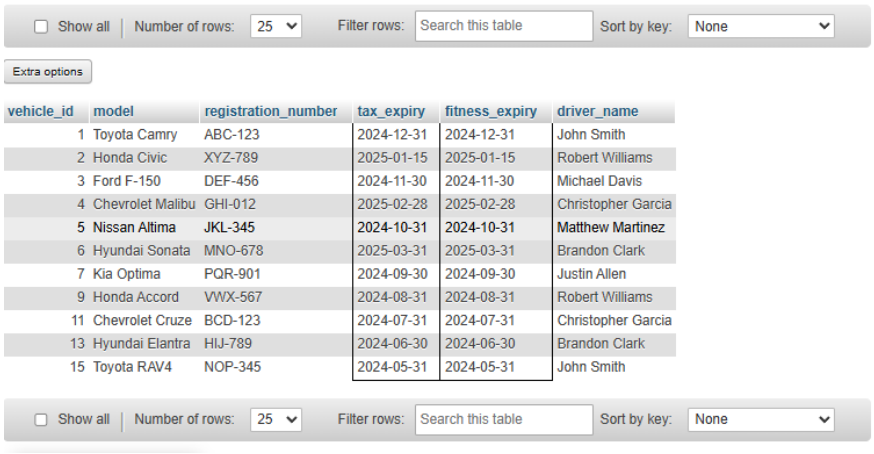
– Find the average rating received by each driver.

```sql
   SELECT d.user_id, d.full_name, AVG(r.rating_value) AS avg_rating
FROM Users d
JOIN Journeys j ON d.user_id = j.driver_id
JOIN Ratings r ON j.journey_id = r.journey_id
WHERE d.user_type = 'driver'
GROUP BY d.user_id, d.full_name
ORDER BY avg_rating DESC;

```



– List vehicles with expired tax or fitness certificates

```sql
    SELECT v.vehicle_id, v.model, v.registration_number,
       v.tax_expiry, v.fitness_expiry,
       u.full_name AS driver_name
FROM Vehicles v
JOIN Users u ON v.driver_id = u.user_id
WHERE v.tax_expiry < CURDATE() OR v.fitness_expiry < CURDATE();

```



Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

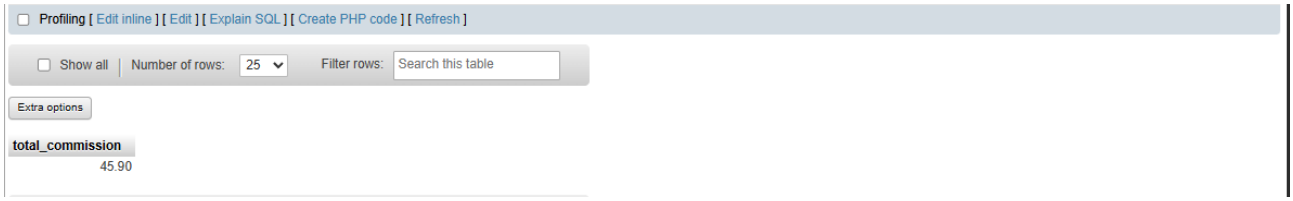
Extra options

vehicle_id	model	registration_number	tax_expiry	fitness_expiry	driver_name
1	Toyota Camry	ABC-123	2024-12-31	2024-12-31	John Smith
2	Honda Civic	XYZ-789	2025-01-15	2025-01-15	Robert Williams
3	Ford F-150	DEF-456	2024-11-30	2024-11-30	Michael Davis
4	Chevrolet Malibu	GHI-012	2025-02-28	2025-02-28	Christopher Garcia
5	Nissan Altima	JKL-345	2024-10-31	2024-10-31	Matthew Martinez
6	Hyundai Sonata	MNO-678	2025-03-31	2025-03-31	Brandon Clark
7	Kia Optima	PQR-901	2024-09-30	2024-09-30	Justin Allen
9	Honda Accord	VWX-567	2024-08-31	2024-08-31	Robert Williams
11	Chevrolet Cruze	BCD-123	2024-07-31	2024-07-31	Christopher Garcia
13	Hyundai Elantra	HIJ-789	2024-06-30	2024-06-30	Brandon Clark
15	Toyota RAV4	NOP-345	2024-05-31	2024-05-31	John Smith

Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

– Calculate total commission earned by the platform)

```sql
    SELECT SUM(commission) AS total_commission
FROM Payments;

```



Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

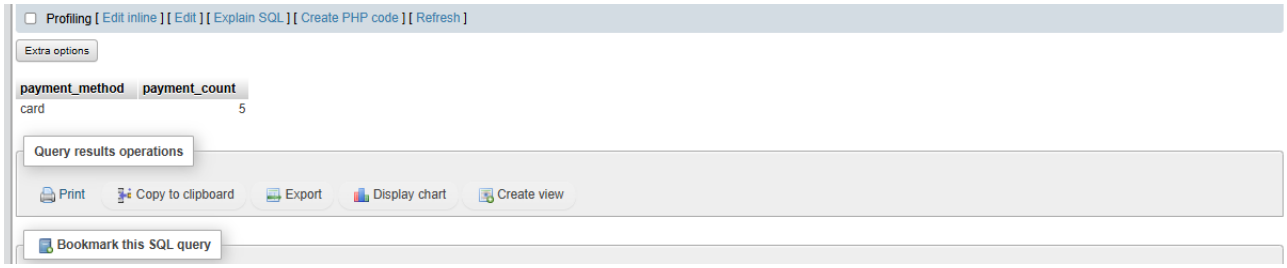
Show all | Number of rows: 25 Filter rows: Search this table

Extra options

total_commission
45.90

– Find the most popular payment method

```sql
    SELECT payment_method, COUNT(*) AS payment_count
FROM Payments
GROUP BY payment_method
ORDER BY payment_count DESC
LIMIT 1;


```



Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

Extra options

payment_method	payment_count
card	5

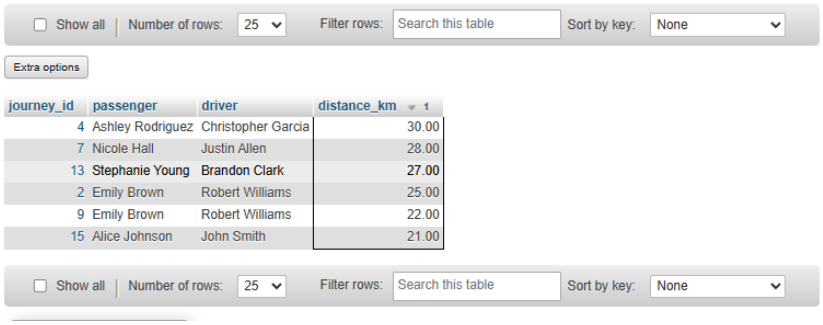
Query results operations

Print | Copy to clipboard | Export | Display chart | Create view

Bookmark this SQL query

– List all journeys longer than 20 km

```sql
    SELECT j.journey_id, p.full_name AS passenger,
       d.full_name AS driver, j.distance_km
FROM Journeys j
JOIN Users p ON j.passenger_id = p.user_id
JOIN Users d ON j.driver_id = d.user_id
WHERE j.distance_km > 20
ORDER BY j.distance_km DESC;

```

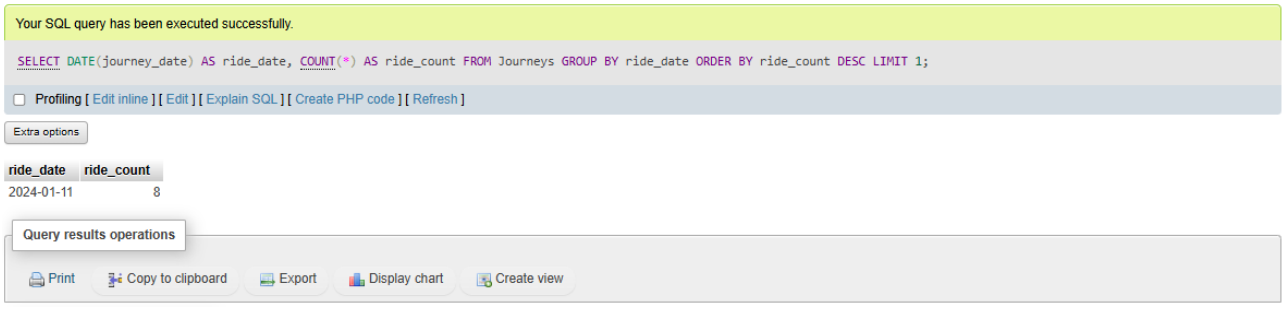


– Count logins by device type

```
    SELECT
    CASE
        WHEN device_info LIKE '%Mobile%' THEN 'Mobile'
        ELSE 'Web'
    END AS device_type,
    COUNT(*) AS login_count
FROM LoginHistory
GROUP BY device_type;

```

– Find the busiest day for rides

```
    SELECT DATE(journey_date) AS ride_date, COUNT(*) AS ride_count
FROM Journeys
GROUP BY ride_date
ORDER BY ride_count DESC
LIMIT 1;

```



– List all emergency contacts for passengers

```
   SELECT u.full_name AS passenger,
       ec.contact_name, ec.contact_phone, ec.relation
```

```
FROM Users u
JOIN EmergencyContacts ec ON u.user_id = ec.user_id
WHERE u.user_type = 'passenger';
```

Show all | Number of rows: 25 | Filter rows: Search this table | Sort by key: None

Extra options

passenger	contact_name	contact_phone	relation
Alice Johnson	Tom Johnson	876-543-2109	Parent
Emily Brown	David Brown	654-321-0987	Friend
Jessica Wilson	Kevin Wilson	432-109-8765	Parent
Ashley Rodriguez	Adam Rodriguez	210-987-6543	Friend
Brittany Robinson	Paul Robinson	098-765-4321	Parent
Stephanie Young	Eric Young	876-432-1099	Friend
Nicole Hall	Jason Hall	654-210-9877	Parent

Show all | Number of rows: 25 | Filter rows: Search this table | Sort by key: None

– Calculate average fare per kilometer

```
SELECT AVG(fare/distance_km) AS avg_fare_per_km
FROM Journeys
WHERE distance_km > 0;
```

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

Show all | Number of rows: 25 | Filter rows: Search this table

Extra options

avg_fare_per_km
1.7070814811

Show all | Number of rows: 25 | Filter rows: Search this table

– Find drivers with multiple vehicles

```
    SELECT u.user_id, u.full_name, COUNT(v.vehicle_id) AS vehicle_count
FROM Users u
JOIN Vehicles v ON u.user_id = v.driver_id
WHERE u.user_type = 'driver'
GROUP BY u.user_id, u.full_name
HAVING COUNT(v.vehicle_id) > 1;
```

– Find the most recent login for each user

```sql
  SELECT u.user_id, u.full_name, MAX(lh.login_time) AS last_login
FROM Users u
LEFT JOIN LoginHistory lh ON u.user_id = lh.user_id
GROUP BY u.user_id, u.full_name;

```

– Calculate total distance traveled by each vehicle)

```sql
SELECT v.vehicle_id, v.model, v.registration_number,
       SUM(j.distance_km) AS total_distance_km
FROM Vehicles v
LEFT JOIN Journeys j ON v.vehicle_id = j.vehicle_id
GROUP BY v.vehicle_id, v.model, v.registration_number
ORDER BY total_distance_km DESC;

```

Show all | Number of rows: 25 Filter rows: Search this table

Extra options

vehicle_id	model	registration_number	total_distance_km ▼ 1
4	Chevrolet Malibu	GHI-012	30.00
7	Kia Optima	PQR-901	28.00
13	Hyundai Elantra	HIJ-789	27.00
2	Honda Civic	XYZ-789	25.00
9	Honda Accord	VWX-567	22.00
15	Toyota RAV4	NOP-345	21.00
6	Hyundai Sonata	MNO-678	18.00
12	Nissan Maxima	EFG-456	17.00
1	Toyota Camry	ABC-123	15.50
8	Toyota Corolla	STU-234	14.00
14	Kia Soul	KLM-012	13.00
5	Nissan Altima	JKL-345	12.00
11	Chevrolet Cruze	BCD-123	11.00
3	Ford F-150	DEF-456	10.00
10	Ford Mustang	YZA-890	9.00

– List all 5-star ratings with their journey details

```sql
SELECT j.journey_id, p.full_name AS passenger,
       d.full_name AS driver, j.start_location, j.end_location,
       r.rating_value, r.review_text
FROM Journeys j
JOIN Ratings r ON j.journey_id = r.journey_id
JOIN Users p ON j.passenger_id = p.user_id
JOIN Users d ON j.driver_id = d.user_id
WHERE r.rating_value = 5;

```

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

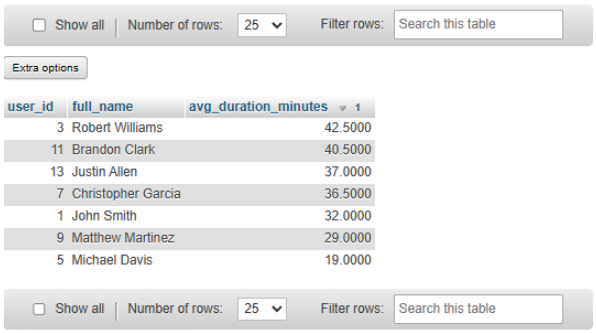
Extra options

journey_id	passenger	driver	start_location	end_location	rating_value	review_text
1	Alice Johnson	John Smith	Downtown A	Suburb B	5	Great ride!
4	Ashley Rodriguez	Christopher Garcia	Industrial G	Downtown H	5	Excellent driver.
7	Nicole Hall	Justin Allen	Commercial M	Airport N	5	Highly recommended.
10	Jessica Wilson	Michael Davis	Downtown S	Commercial T	5	Fantastic!
13	Stephanie Young	Brandon Clark	Commercial Y	Airport Z	5	The best.

Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

– Calculate the average journey duration by driver

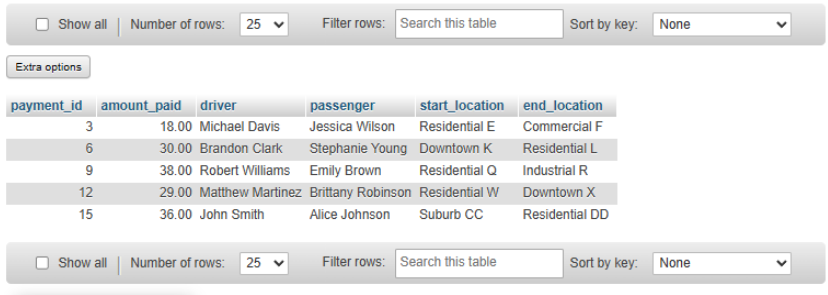
```sql
SELECT d.user_id, d.full_name, AVG(j.duration_minutes) AS
    avg_duration_minutes
FROM Users d
JOIN Journeys j ON d.user_id = j.driver_id
WHERE d.user_type = 'driver'
GROUP BY d.user_id, d.full_name
ORDER BY avg_duration_minutes DESC;


```



user_id	full_name	avg_duration_minutes
3	Robert Williams	42.5000
11	Brandon Clark	40.5000
13	Justin Allen	37.0000
7	Christopher Garcia	36.5000
1	John Smith	32.0000
9	Matthew Martinez	29.0000
5	Michael Davis	19.0000

– List all payments made via mobile banking with their journey details

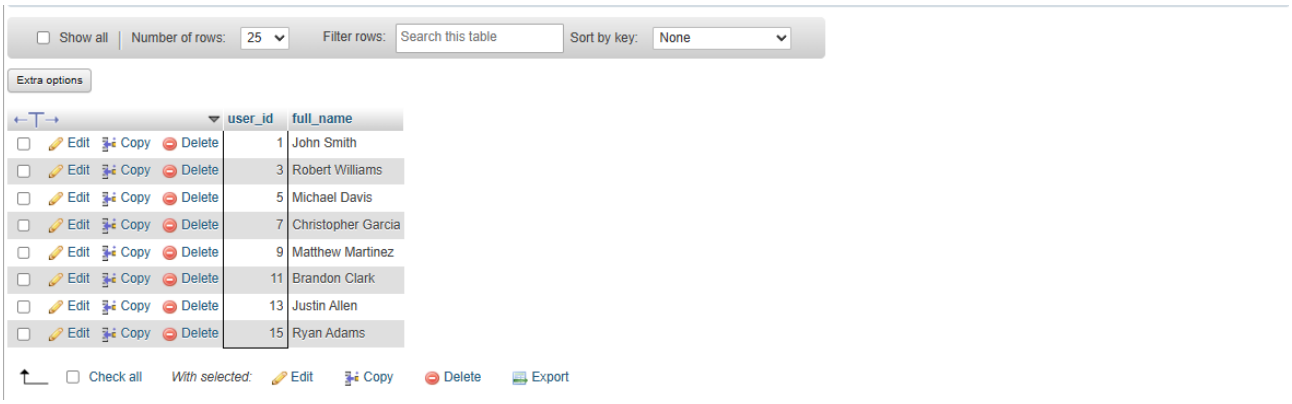
```sql
   SELECT p.payment_id, p.amount_paid,
       d.full_name AS driver, ps.full_name AS passenger,
       j.start_location, j.end_location
FROM Payments p
JOIN Journeys j ON p.journey_id = j.journey_id
JOIN Users d ON j.driver_id = d.user_id
JOIN Users ps ON j.passenger_id = ps.user_id
WHERE p.payment_method = 'mobile_banking';


```



payment_id	amount_paid	driver	passenger	start_location	end_location
3	18.00	Michael Davis	Jessica Wilson	Residential E	Commercial F
6	30.00	Brandon Clark	Stephanie Young	Downtown K	Residential L
9	38.00	Robert Williams	Emily Brown	Residential Q	Industrial R
12	29.00	Matthew Martinez	Brittany Robinson	Residential W	Downtown X
15	36.00	John Smith	Alice Johnson	Suburb CC	Residential DD

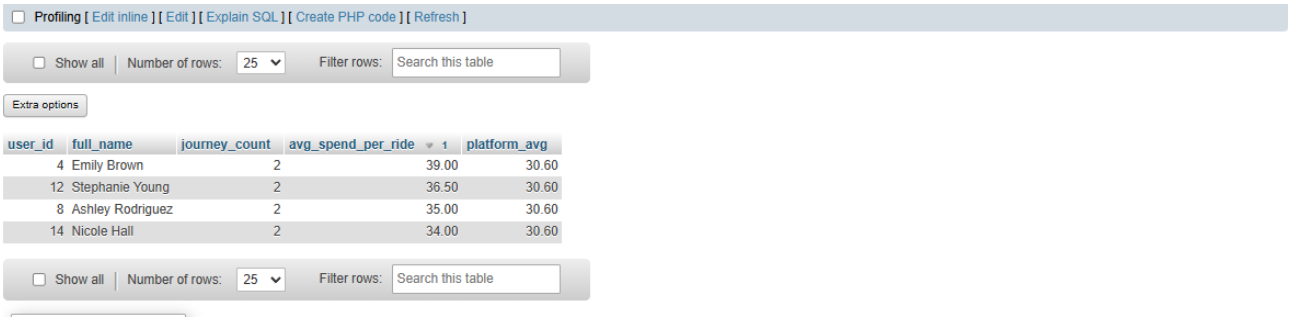
– Find drivers who haven't completed any journeys in the last month

```sql
    SELECT u.user_id, u.full_name
FROM Users u
WHERE u.user_type = 'driver'
AND u.user_id NOT IN (
    SELECT DISTINCT j.driver_id
    FROM Journeys j
    WHERE j.journey_date >= DATE_SUB(CURDATE(), INTERVAL 1 MONTH)
);

```



– Find passengers who spend more than average per journey

```sql
   SELECT
    p.user_id,
    p.full_name,
    COUNT(j.journey_id) AS journey_count,
    ROUND(AVG(j.fare), 2) AS avg_spend_per_ride,
    (SELECT ROUND(AVG(fare), 2) FROM Journeys) AS platform_avg
FROM Users p
JOIN Journeys j ON p.user_id = j.passenger_id
WHERE p.user_type = 'passenger'
GROUP BY p.user_id, p.full_name
HAVING AVG(j.fare) > (SELECT AVG(fare) FROM Journeys)
ORDER BY avg_spend_per_ride DESC;

```



9 Conclusions

A database designed for the purpose of effectively managing a carpooling service's operations, the MySQL database includes vital information for drivers, users, and the business. The system tracks the user's journey history with information such as dates, places, distance, length, fares, and ratings for the driver. It also keeps track of user information, contact information, and emergency contacts. Driving licenses, national identification numbers, automobile ownership, and login histories and emergency contacts are also stored so that the app can ensure the highest degree of security for its users, guaranteeing that all activities pertaining to them are recorded and monitored. The information of all the cars, including model, registration number, year of manufacture, and odometer readings, as well as crucial papers like tax and fitness expiration dates, are also managed the database. The system also keeps track of revenue information, including commissions, payment methods, earnings from each trip, and other financial transactions. The database improves the capacity of the carpooling service to effectively manage drivers, passengers, and vehicles by centralizing and organizing all of this data. It also provides clear reporting on trip activity, financial performance, and fleet management. The project architecture prioritizes efficient and simple querying of all private user and financial data, while providing a convenient way for drivers and users to track journeys, book rides, and leave reviews. The end goal is to enable an application that offers a productive and safe platform for overseeing the whole carpooling ecosystem, enhancing financial control, operational transparency, and user experience.

10 Acknowledgements

We wish to extend our profound thanks to our faculty advisor, Prof. Dr. Kamruddin Nur. We are grateful for the chance to develop this project under his supervision and for the insightful direction he provided. His knowledge, especially regarding MySQL, proved essential to the successful design and implementation of our Carpooling Management System.

Our sincere appreciation also goes to the members of our project group: Shadman Tahmid Arib, Md. Azmine Amin Mormo, and Md. Arif Ali Robin. Their unwavering commitment, cooperative approach, and valuable contributions were fundamental throughout the development lifecycle. Together, we successfully tackled obstacles and leveraged our collective expertise to build a comprehensive and effective database system for managing carpooling services.