YYC RENTALS

CAR RENTAL BUSINESS

Prepared by:
Jahnavi Sharma
https://www.linkedin.com/in/jahnavi-sh/

02

INTRODUCTION

In the competitive car rental industry, YYC Rentals, a Canadian car rental company, operates across multiple locations of major cities and airports, offering a variety of vehicles to meet the growing demand for short- and long-term rentals. Committed to providing reliable, accessible and secure services, the company focuses on customer satisfaction, convenience and competitive pricing. YYC Rentals requires an efficient and scalable database that manags customer, vehicle, rental and employee information, to support its operations.

This case study aims to build a relational database design to streamline YYC Rentals' operations. The database addresses that company's mission to offer diverse vehicles, affordable pricing, and exceptional client service while technology to make data-driven insights to improve operational efficiency. The database supports various tasks, including vehicle reservations, customer feedback, and employee management, ensuring that YYC Rentals remains competitive and meets the needs of its customers.

03

MISSION OBJECTIVES

Mission is to offer reliable, accessible and secure car rentals. The mission objectives ensure that YYC Rentals remains competitive in the market while continuing to grow and provide top-notch services

1. Wide selection of vehicles: The company's inventory is designed to cater to specific needs of different customer segments, ranging from economy, luxury to specialty vehicles.
2. Convenience and accessibility: The company strategically positions its branches in high-traffic areas like airports and central city locations, ensuring customer convenience and ease.
3. Customer satisfaction: Exceptional customer service is at the heart of YYC Rentals.
4. Data-driven decision making: In a era where data is key to success, the company aims to leverage technology to optimize its operations and customer interactions.
5. Security and reliability: The company is committed to investing in secure technologies to protect customer data and maintain integrity of all transactions.

04

DATABASE DESIGN

The database design of YYC Rentals is essential to the company's operational efficiency and scalability. The database consists of several key tables, each designed to capture crucial information about different aspects of the business. Below is a breakdown of these tables and their primary attributes:

1. Customer Table:

To store personal and contact details of all customers using YYC Rentals.

Column Name	Data Type	Constraints	Description
`user_id`	INT	PRIMARY KEY, AUTO_INCREMENT	Unique identifier for each customer
`first_name`	VARCHAR	NOT NULL	Customer's first name
`last_name`	VARCHAR	NOT NULL	Customer's last name
`email`	VARCHAR	NOT NULL, UNIQUE	Customer's email address (must be unique)
`phone_number`	VARCHAR	NOT NULL, UNIQUE	Customer's phone number
`password`	VARCHAR	NOT NULL	Encrypted password for account access
`address`	VARCHAR	NOT NULL	Customer's physical address
`created_at`	DATETIME	DEFAULT CURRENT_TIMESTAMP	Timestamp when the customer account was created

2. Vehicle table:
To store data about the vehicle available to rent.

Column Name	Type	Constraints	Description
`vehicle_id`	INT	PRIMARY KEY, AUTO_INCREMENT	Unique identifier for each vehicle
`model`	VARCHAR	NOT NULL	Vehicle make and model (e.g., Toyota Camry)
`vehicle_type`	VARCHAR	NOT NULL	Vehicle category (e.g., economy, luxury, SUV)
`daily_rate`	DECIMAL	NOT NULL	Daily rental price for the vehicle
`availability_status`	BOOLEAN	NOT NULL, DEFAULT TRUE	Indicates if the vehicle is available for rent (true/false)
`branch_id`	INT	FOREIGN KEY (branch_id) REFERENCES Branch(branch_id)	Links the vehicle to the branch where it is located
`insurance_id`	INT	FOREIGN KEY (insurance_id) REFERENCES VehicleInsurance(insurance_id)	Links the vehicle to its insurance policy
`year`	YEAR	NOT NULL	Year the vehicle was manufactured

3. Branch table:
To store information about the various rental locations of YYC Rentals.

Column Name	Data Type	Constraints	Description
`branch_id`	INT	PRIMARY KEY, AUTO_INCREMENT	Unique identifier for each branch
`branch_name`	VARCHAR	NOT NULL, UNIQUE	Name of the rental branch (e.g., "Toronto Airport")
`address`	VARCHAR	NOT NULL	Branch address
`city`	VARCHAR	NOT NULL	City where the branch is located
`state`	VARCHAR	NOT NULL	State or province where the branch is located
`zipcode`	VARCHAR	NOT NULL	Postal code for the branch
`phone_number`	VARCHAR	NOT NULL, UNIQUE	Contact phone number for the branch
`email`	VARCHAR	NOT NULL, UNIQUE	Contact email address for the branch

4. Vehicle reviews table:
To capture customer feedback on the vehicle that they had rented.

Column Name	Data Type	Constraints	Description
`review_id`	INT	PRIMARY KEY, AUTO_INCREMENT	Unique identifier for each review
`user_id`	INT	FOREIGN KEY (user_id) REFERENCES Customer(user_id), NOT NULL	Links the review to the customer
`vehicle_id`	INT	FOREIGN KEY (vehicle_id) REFERENCES Vehicle(vehicle_id), NOT NULL	Links the review to the vehicle being reviewed
`review_rating`	DECIMAL	NOT NULL, CHECK (review_rating BETWEEN 1 AND 5)	Rating given by the customer (e.g., on a scale of 1-5)
`review_text`	TEXT	NULL	Written feedback provided by the customer
`review_date`	DATE	NOT NULL	Date the review was submitted

5. Accident Reports:
To log any accidents involving rented vehicles.

Column Name	Data Type	Constraints	Description
`report_id`	INT	PRIMARY KEY, AUTO_INCREMENT	Unique identifier for each accident report
`reservation_id`	INT	FOREIGN KEY (reservation_id) REFERENCES Reservation(reservation_id), NOT NULL	Links the report to the reservation
`vehicle_id`	INT	FOREIGN KEY (vehicle_id) REFERENCES Vehicle(vehicle_id), NOT NULL	Links the report to the involved vehicle
`insurance_id`	INT	FOREIGN KEY (insurance_id) REFERENCES VehicleInsurance(insurance_id), NOT NULL	Links the report to the vehicle's insurance policy
`accident_date`	DATE	NOT NULL	Date when the accident occurred
`description`	TEXT	NULL	Brief description of the accident
`damage_cost`	DECIMAL	NULL	Estimated cost of damages
`is_resolved`	BOOLEAN	NOT NULL, DEFAULT FALSE	Indicates if the accident issue has been resolved (true/false)

6. Vehicle Insurance:

Column Name	Data Type	Constraints	Description
`insurance_id`	INT	PRIMARY KEY, AUTO_INCREMENT	Unique identifier for each insurance policy
`provider_name`	VARCHAR	NOT NULL	Name of the insurance provider
`policy_number`	VARCHAR	NOT NULL, UNIQUE	Unique policy number
`coverage_amount`	DECIMAL	NOT NULL	Coverage amount for the insurance policy
`valid_from`	DATE	NOT NULL	Start date of the insurance policy
`valid_until`	DATE	NOT NULL	Expiration date of the insurance policy

7. Employee table:
To store information about employees at YYC Rentals.

Column Name	Data Type	Constraints	Description
`employee_id`	INT	PRIMARY KEY, AUTO_INCREMENT	Unique identifier for each employee
`first_name`	VARCHAR	NOT NULL	Employee's first name
`last_name`	VARCHAR	NOT NULL	Employee's last name
`role`	VARCHAR	NOT NULL	Job role or position held by the employee (e.g., Manager)
`branch_id`	INT	FOREIGN KEY (branch_id) REFERENCES Branch(branch_id), NOT NULL	Links the employee to the branch where they work
`hire_date`	DATE	NOT NULL	Date when the employee was hired
`email`	VARCHAR	NOT NULL, UNIQUE	Employee's work email address
`phone_number`	VARCHAR	NOT NULL, UNIQUE	Employee's contact number

8. Reservations table:
To track all rental bookings made by customers

Column Name	Type	Constraints	Description
`reservation_id`	INT	PRIMARY KEY, AUTO_INCREMENT	Unique identifier for each reservation
`user_id`	INT	FOREIGN KEY (user_id) REFERENCES Customer(user_id), NOT NULL	Links the reservation to the customer
`vehicle_id`	INT	FOREIGN KEY (vehicle_id) REFERENCES Vehicle(vehicle_id), NOT NULL	Links the reservation to the rented vehicle
`reservation_date`	DATE	NOT NULL	Date when the reservation was made
`start_date`	DATE	NOT NULL	Start date of the rental period
`end_date`	DATE	NOT NULL	End date of the rental period
`total_price`	DECIMAL	NOT NULL	Total cost of the rental for the reservation period
`status`	VARCHAR	NOT NULL, CHECK (status IN ('active', 'completed', 'canceled'))	Status of the reservation (e.g., active, completed, canceled)

09 TABLE RELATIONSHIPS

This includes the key entities in the database, their relationships, and the foreign keys used to establish these connections.

1. Customer Entity and its relationships

Source Table	Target Table	Relationship Type	Foreign Key	Description
Customer	Reservations	One-to-Many	`user_id` in Reservations	One customer can make multiple reservations, but each reservation is linked to a single customer.
Customer	Vehicle Reviews	One-to-Many	`user_id` in Vehicle Reviews	One customer can leave multiple reviews for different vehicles.

2. Vehicle Entity and its relationships

Source Table	Target Table	Relationship Type	Foreign Key	Description
Vehicles	Branches	Many-to-One	`branch_id` in Vehicles	Each vehicle belongs to one branch, but a branch can have many vehicles.
Vehicles	Reservations	One-to-Many	`vehicle_id` in Reservations	A vehicle can be reserved multiple times, but each reservation is for one specific vehicle.
Vehicles	Vehicle Reviews	One-to-Many	`vehicle_id` in Vehicle Reviews	One vehicle can receive multiple reviews from different customers.
Vehicles	Accident Reports	One-to-Many	`vehicle_id` in Accident Reports	One vehicle can be involved in multiple accident reports.
Vehicles	Vehicle Insurance	One-to-One	`insurance_id` in Vehicles	Each vehicle is linked to one insurance policy, but one policy may cover multiple vehicles.

3. Reservations Entity and its relationships

Source Table	Target Table	Relationship Type	Foreign Key	Description
Reservations	**Customer**	Many-to-One	`user_id` in **Reservations**	Each reservation is tied to one customer, but a customer can have multiple reservations.
Reservations	**Vehicles**	Many-to-One	`vehicle_id` in **Reservations**	Each reservation is for one vehicle, but a vehicle can have multiple reservations.
Reservations	**Accident Reports**	One-to-Many	`reservation_id` in **Accident Reports**	A single reservation can result in multiple accident reports.

4. Branch Entity and its relationships

Source Table	Target Table	Relationship Type	Foreign Key	Description
Branches	**Vehicles**	One-to-Many	`branch_id` in **Vehicles**	A branch can have many vehicles available for rent, but each vehicle belongs to one branch.
Branches	**Employees**	One-to-Many	`branch_id` in **Employees**	Each branch employs multiple staff members, but each employee works at only one branch.

5. Vehicle Reviews Entity and its relationships

Source Table	Target Table	Relationship Type	Foreign Key	Description
Vehicle Reviews	**Customer**	Many-to-One	`user_id` in **Vehicle Reviews**	Each review is submitted by one customer, but a customer can submit reviews for multiple vehicles.
Vehicle Reviews	**Vehicles**	Many-to-One	`vehicle_id` in **Vehicle Reviews**	Each review is for one vehicle, but a vehicle can receive multiple reviews from different customers.

6. Accident Reports Entity and its relationships

Source Table	Target Table	Relationship Type	Foreign Key	Description
Accident Reports	**Reservations**	Many-to-One	`reservation_id` in **Accident Reports**	Each accident report is tied to a single reservation, but one reservation can have multiple accident reports.
Accident Reports	**Vehicles**	Many-to-One	`vehicle_id` in **Accident Reports**	Each accident report involves one vehicle, but a vehicle can be part of multiple accident reports.
Accident Reports	**Vehicle Insurance**	Many-to-One	`insurance_id` in **Accident Reports**	Each accident report refers to one insurance policy, but multiple reports can reference the same policy.

7. Vehicle Insurance Entity and its relationships

Source Table	Target Table	Relationship Type	Foreign Key	Description
Vehicle Insurance	**Vehicles**	One-to-One	`insurance_id` in **Vehicles**	Each vehicle is linked to one insurance policy, but multiple vehicles can share the same insurance policy.

8. Employess Entity and its relationships

Source Table	Target Table	Relationship Type	Foreign Key	Description
Employees	**Branches**	Many-to-One	`branch_id` in **Employees**	Each employee works for one branch, but a branch can have many employees.

12

DATABASE DEVELOPMENT

We create a sample database for the rental car system. Below is the sample query run during the development. Moreover, the other queries are mentioned in Appendix B.

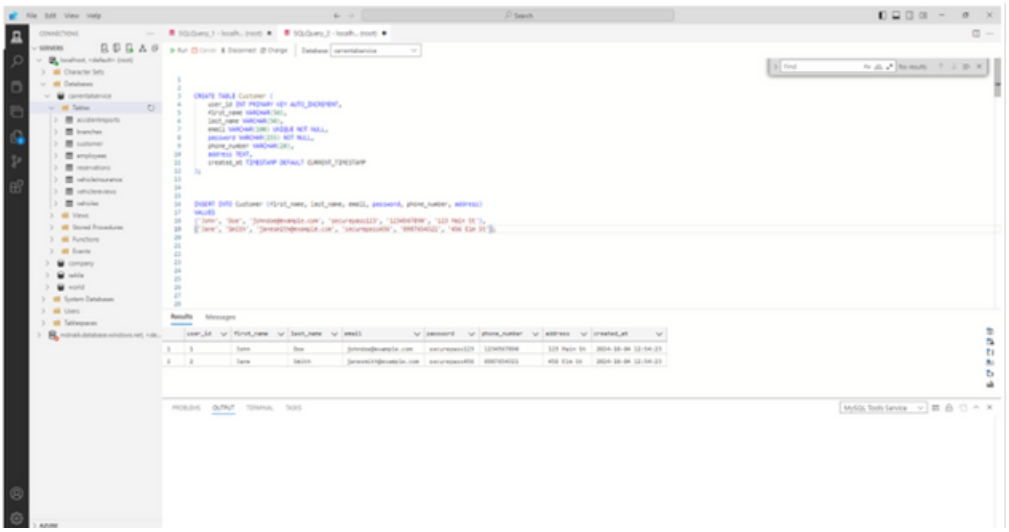
Create Database:

```
CREATE DATABASE CarRentalService;
```

Create table and Insert query for Customer table:

1. Customer Table

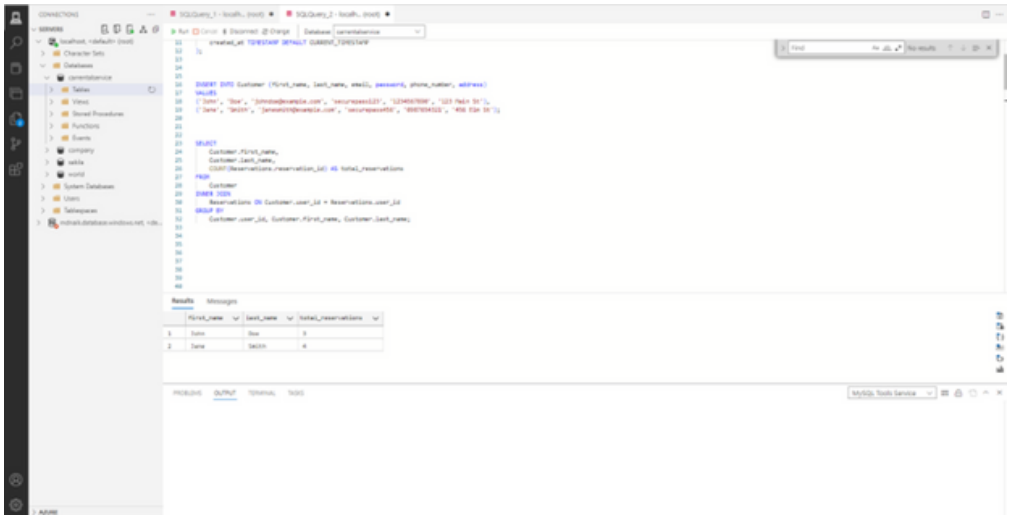
```
CREATE TABLE Customer (
    user_id INT PRIMARY KEY AUTO_INCREMENT,
    first_name VARCHAR(50),
    last_name VARCHAR(50),
    email VARCHAR(100) UNIQUE NOT NULL,
    password VARCHAR(255) NOT NULL,
    phone_number VARCHAR(20),
    address TEXT,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
);
INSERT INTO Customer (first_name, last_name, email, password,
phone_number, address)
VALUES
('John', 'Doe', 'johndoe@example.com', 'securepass123', '1234567890',
'123 Main St'),
('Jane', 'Smith', 'janesmith@example.com', 'securepass456',
'0987654321', '456 Elm St');
```



Inner join Query

We are trying to get the number of bookings by customer and displaying their first name and last name by matching the reservation id in reservation table and customer id from customer table.

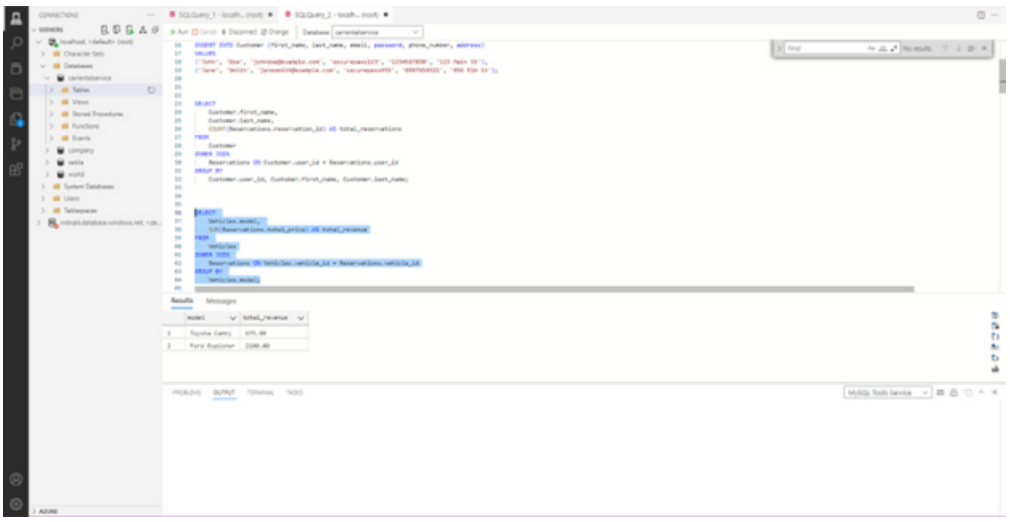
```
SELECT
    Customer.first_name,
    Customer.last_name,
    COUNT(Reservations.reservation_id) AS total_reservations
FROM
    Customer   INNER JOIN
    Reservations ON Customer.user_id = Reservations.user_id
GROUP BY
    Customer.user_id, Customer.first_name, Customer.last_name;
```



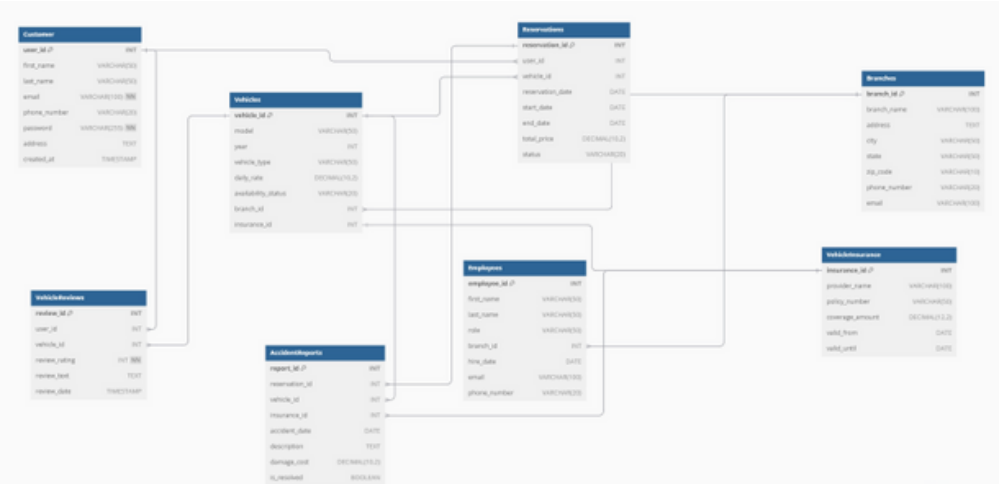
Groupby Query

We are trying to get the revenue generated by each vehicle using group by.

```
SELECT
    Vehicles.model,
    SUM(Reservations.total_price) AS total_revenue
FROM
    Vehicles
INNER JOIN
    Reservations ON Vehicles.vehicle_id = Reservations.vehicle_id
GROUP BY
    Vehicles.model;
```



ER DIAGRAM



CONCLUSION

In conclusion, the database is robust, secure, and aligned with YYC Rentals' core mission objectives. It ensures smooth operations, enhances customer satisfaction and provides a scalable platform for future growth. Additionally, the integration of security measures, scalability features, and efficient processes makes this database a valuable asset of YYC Rentals and it's continuous growth.

APPENDIX A

MISSION OBJECTIVES

1. ***Wide selection of vehicles***

The company's inventory is designed to cater to specific needs of different customer segments. It includes:

1. ***Economy vehicles for budget-conscious travellers who prioritise cost-effectiveness.***
2. ***Luxury vehicles for those seeking a more comfortable or premium experience***
3. ***Specialty vehicles such as SUVs and vans, for families or groups that may require more space and versatility.***

By offering a wide selection of vehicles, YYC Rentals is able to meet the demands of a broad customer base, including tourists, business travellers, and locals in need of temporary transportation.

To achieve this, the company regularly reviews its fleet composition and adds new models to stay up-to-date with market demands. By offering a wide range of vehicle types, YYC Rentals enhances its appeal and increases the likelihood of repeat business.

2. Convenience and accessibility

One of YYC Rentals' competitive advantage is convenience it provides to customers. The company strategically positions its branches in high-traffic areas like airports and central city locations. This ensures that customers can easily pick up and drop off vehicles at their convenience. This helps reduce the hassle associated with traditional car rental services. Whether a customer is arriving at the airport or booking a vehicle for a road trip, YYC Rental strive to provide a seamless experience.

APPENDIX A

3. Exceptional client service
Customer satisfaction is at the heart of YYC Rentals' business strategy. This is done by emphasizing:
a. *Competitive pricing: Company ensures that customers access to affordable rates, with a variety of packages and discounts available to suit different budgets. They provide competitive pricing that balances affordability with the quality of service provided.*
Exceptional client service: The company prioritizes a smooth rental process and offers support to customers at every step, from booking to vehicle return and maintenance. The company also provides 24/7 customer service to assist with any issues or inquiries, ensuring that help is always available when needed.

4. *Data-driven decision making*
I*n a era where data is key to success, the company aims to leverage technology to optimize its operations and customer interactions.*
The use of technology allows the company to streamline its operations such as vehicle tracking, maintenance scheduling and customer management, making the overall experience more efficient and reliable. Additionally, by understanding customer preferences and patterns, YYC Rentals can tailor its services to better meet customer needs.

5. *Secure and reliable platform*
As a company that is responsible for handling sensitive customer information, including personal details and payment, YYC Rentals places a strong emphasis on data security. The company is committed to investing in secure technologies to protect customer data and maintain integrity of all transactions. This not only helps build trust with customers but also ensures compliance with industry standards and regulations.

In addition to this, the company implements state-of-the-art security measure, including encryption technologies and robust data management practices. The reliability of the platform is also crucial, ensuring customers can easily access services without downtime or technical issues.

APPENDIX A

KEY CONSIDERAITIONS

While designing the database of YYC Rentals, several important considerations were taken into account, like:

1. *Data integrity:*

Maintaining data integrity helps ensure the accuracy and reliability of database

1. *Primary and Foreign keys: Each table has primary keys that uniquely identify each record. Foreign keys establish relationships between different tables, enforcing referential integrity. For example, the user_id in the Customer table serves as a primary key, while the same user_id serves as a foreign key in the Reservations table to link customer reservations to their corresponding profiles.*
2. *Constraints: Data constraints, such as NOT NULL, UNIQUE, and CHECK constraints, are applied to prevent invalid or duplicate data entries. For instance, fields like email and phone_number must be unique in the Customer table, ensuring that no two customers have the same contact information.*
3. *Validation Rules: Input validation ensures that data such as dates, email addresses, and prices conform to acceptable formats, reducing the risk of errors.*

2. Scalability:

As YYC Rentals expands to more locations, increases its fleet size and gains more customers, the database structure is built to accommodate this growth.

1. *Horizontal Scaling: The database can be extended to include new branches, employees, and vehicles without requiring changes to the database schema. New branches, for instance, can be easily added by inserting new records into the Branch table, and these branches can immediately be linked to vehicles and employees.*
2. *Efficient Indexing: Indexes on commonly queried fields, such as vehicle_id, user_id, and reservation_id, ensure that as the database grows, search and retrieval operations remain fast and efficient.*
3. *Modular Design: The database is modular, meaning that additional tables can be added in the future without disrupting existing tables and relationships.*

APPENDIX B

Following are some SQL queries for creating the database.

1. Customer table

```
INSERT INTO Customer (first_name, last_name, email, password, phone_number, address)
VALUES
('John', 'Doe', 'john.doe@example.com', 'hashed_password_1', '123-456-7890', '123 Elm
St, Springfield'),
('Jane', 'Smith', 'jane.smith@example.com', 'hashed_password_2', '234-567-8901', '456
Oak St, Shelbyville'),
('Michael', 'Johnson', 'michael.johnson@example.com', 'hashed_password_3',
'345-678-9012', '789 Pine St, Capital City'),
('Emily', 'Brown', 'emily.brown@example.com', 'hashed_password_4', '456-789-0123',
'321 Maple St, Ogdenville'),
('Daniel', 'Williams', 'daniel.williams@example.com', 'hashed_password_5',
'567-890-1234', '654 Cedar St, North Haverbrook'),
('Olivia', 'Jones', 'olivia.jones@example.com', 'hashed_password_6', '678-901-2345',
'987 Birch St, Brockway'),
('Matthew', 'Miller', 'matthew.miller@example.com', 'hashed_password_7',
'789-012-3456', '123 Ash St, Monorail City'),
('Sophia', 'Davis', 'sophia.davis@example.com', 'hashed_password_8', '890-123-4567',
'456 Fir St, Shelbyville'),
('David', 'Garcia', 'david.garcia@example.com', 'hashed_password_9', '901-234-5678',
'789 Oak St, Springfield'),
('Isabella', 'Martinez', 'isabella.martinez@example.com', 'hashed_password_10',
'012-345-6789', '321 Pine St, Capital City');
```

2. Vehicles table

```
INSERT INTO Vehicles (model, year, vehicle_type, daily_rate,
availability_status, branch_id, insurance_id) VALUES
('Toyota Camry', 2020, 'Sedan', 45.99, 'available', 1, 101),
('Ford Explorer', 2019, 'SUV', 65.50, 'rented', 2, 102),
 ('Chevrolet Silverado', 2021, 'Truck', 80.00, 'available', 1,
103),
 ('Honda Civic', 2018, 'Sedan', 40.00, 'maintenance', 3, 104),
('Tesla Model 3', 2022, 'Sedan', 120.00, 'available', 2, 105);
```

3. Reservations table

```
INSERT INTO Reservations (user_id, vehicle_id, reservation_date,
start_date, end_date, total_price, status)
VALUES
(1, 1, '2024-09-15', '2024-09-20', '2024-09-25', 229.95,
'confirmed'),
(2, 2, '2024-09-18', '2024-09-21', '2024-09-23', 131.00,
'completed'),
(3, 3, '2024-09-20', '2024-09-22', '2024-09-24', 160.00,
'cancelled'),
(4, 4, '2024-09-25', '2024-09-28', '2024-10-01', 240.00,
'confirmed'),
(5, 5, '2024-09-30', '2024-10-02', '2024-10-05', 360.00,
'confirmed');
```

4. Branches table

```
INSERT INTO Branches (branch_name, address, city, state, zip_code,
phone_number, email)
VALUES
('Downtown Branch', '123 Main St', 'Springfield', 'IL', '62701',
'217-555-1234', 'downtown@carrental.com'),
('Airport Branch', '456 Airport Rd', 'Springfield', 'IL', '62707',
'217-555-5678', 'airport@carrental.com'),
('Eastside Branch', '789 East Blvd', 'Shelbyville', 'IL', '62565',
'217-555-9876', 'eastside@carrental.com'),
('Westside Branch', '321 West Dr', 'Ogdenville', 'IL', '62298',
'618-555-4321', 'westside@carrental.com'),
('Capitol Branch', '654 Capitol Ave', 'Capital City', 'IL', '62702',
'217-555-8765', 'capitol@carrental.com');
```

5. Feedbacks and Ratings table

```
INSERT INTO VehicleReviews (user_id, vehicle_id, review_rating,
review_text)
VALUES
(1, 1, 5, 'Excellent car! Smooth ride and very comfortable. Highly
recommend!'),
(2, 2, 4, 'Great SUV for family trips. Good space but a bit thirsty on
gas.'),
(3, 3, 3, 'The truck was okay. Performance was decent, but it had some
scratches.'),
(4, 4, 5, 'Absolutely loved this car! Perfect for my weekend
getaway.'),
(5, 5, 4, 'The Tesla was amazing! Quick acceleration and great
technology. A bit pricey though.');
```

6. Accident Reports table

```
INSERT INTO AccidentReports (rental_id, vehicle_id, insurance_id,
accident_date, description, damage_cost, is_resolved)
VALUES
(1, 1, 101, '2024-09-21', 'Minor fender bender, no injuries
reported.', 250.00, 0),
(2, 2, 102, '2024-09-22', 'Side collision at an intersection.',
1500.50, 0),
(3, 3, 103, '2024-09-23', 'Rear-end collision, extensive
damage.', 3500.75, 1),
(4, 4, 104, '2024-09-24', 'Accident due to slippery road
conditions.', 800.00, 0),
(5, 5, 105, '2024-09-25', 'Vehicle rolled over, severe damage.',
5000.00, 1);
```

7. Employee table

```
INSERT INTO Employees (first_name, last_name, role, branch_id,
hire_date, email, phone_number)
VALUES
('Alice', 'Johnson', 'manager', 1, '2023-01-15',
'alice.johnson@carrental.com', '217-555-0011'),
('Bob', 'Smith', 'customer service', 2, '2023-03-10',
'bob.smith@carrental.com', '217-555-0012'),
('Charlie', 'Brown', 'mechanic', 1, '2023-05-20',
'charlie.brown@carrental.com', '217-555-0013'),
('David', 'Williams', 'customer service', 3, '2023-07-05',
'david.williams@carrental.com', '217-555-0014'),
('Eva', 'Davis', 'manager', 2, '2023-09-01',
'eva.davis@carrental.com', '217-555-0015');
```

ER Diagram

```
Table Customer {
  user_id INT [pk, increment]
  first_name VARCHAR(50)
  last_name VARCHAR(50)
  email VARCHAR(100) [unique, not null]
  phone_number VARCHAR(20)
  password VARCHAR(255) [not null]
  address TEXT
  created_at TIMESTAMP [default: `CURRENT_TIMESTAMP`]
```

```
Table Vehicles {
  vehicle_id INT [pk, increment]
  model VARCHAR(50)
  year INT
  vehicle_type VARCHAR(50)
  daily_rate DECIMAL(10, 2)
  availability_status VARCHAR(20)
  branch_id INT [ref: > Branches.branch_id]
  insurance_id INT [ref: - VehicleInsurance.insurance_id]
}
```

```
Table Reservations {
  reservation_id INT [pk, increment]
  user_id INT [ref: > Customer.user_id]
  vehicle_id INT [ref: > Vehicles.vehicle_id]
  reservation_date DATE
  start_date DATE
  end_date DATE
  total_price DECIMAL(10, 2)
  status VARCHAR(20)
}
```

```
Table Branches {
  branch_id INT [pk, increment]
  branch_name VARCHAR(100)
  address TEXT
  city VARCHAR(50)
  state VARCHAR(50)
  zip_code VARCHAR(10)
  phone_number VARCHAR(20)
  email VARCHAR(100)
}
```

```
Table VehicleReviews {
  review_id INT [pk, increment]
  user_id INT [ref: > Customer.user_id]
  vehicle_id INT [ref: > Vehicles.vehicle_id]
  review_rating INT [note: '1-5 scale', default: 3, not null]
  review_text TEXT
  review_date TIMESTAMP [default: `CURRENT_TIMESTAMP`]
```

```
Table AccidentReports {
  report_id INT [pk, increment]
  reservation_id INT [ref: > Reservations.reservation_id]
  vehicle_id INT [ref: > Vehicles.vehicle_id]
  insurance_id INT [ref: > VehicleInsurance.insurance_id]
  accident_date DATE
  description TEXT
  damage_cost DECIMAL(10, 2)
  is_resolved BOOLEAN [default: `FALSE`]
}
```

```
Table Employees {
  employee_id INT [pk, increment]
  first_name VARCHAR(50)
  last_name VARCHAR(50)
  role VARCHAR(50)
  branch_id INT [ref: > Branches.branch_id]
  hire_date DATE
  email VARCHAR(100)
  phone_number VARCHAR(20)
}
```

```
Table VehicleInsurance {
  insurance_id INT [pk, increment]
  provider VARCHAR(100)
  policy_number VARCHAR(50)
  coverage_amount DECIMAL(12, 2)
  valid_from DATE
  valid_until DATE
}
```

JOIN QUERY

```
SELECT Customers.first_name, Customers.last_name,
Reservations.reservation_id, Reservations.start_date,
Reservations.end_date, Reservations.total_price FROM Customers INNER
JOIN Reservations ON Customers.user_id = Reservations.user_id;

SELECT
    Reservations.reservation_id,
    Vehicles.model,
    Vehicles.year,
    Reservations.start_date,
    Reservations.end_date,
    Reservations.total_price
FROM
    Reservations
INNER JOIN
    Vehicles ON Reservations.vehicle_id = Vehicles.vehicle_id;

SELECT
    Customers.first_name,
    Customers.last_name,
    Vehicles.model,
    Reservations.reservation_date,
    Reservations.start_date,
    Reservations.end_date,
    Reservations.total_price
FROM
    Customers
INNER JOIN
    Reservations ON Customers.user_id = Reservations.user_id
INNER JOIN
    Vehicles ON Reservations.vehicle_id = Vehicles.vehicle_id;

SELECT
    VehicleReviews.review_id,
    VehicleReviews.review_rating,
    VehicleReviews.review_text,
    Vehicles.model
FROM
    VehicleReviews
INNER JOIN
    Vehicles ON VehicleReviews.vehicle_id = Vehicles.vehicle_id;
```

25

GROUPBY QUERY

```
SELECT
    Customers.first_name,
    Customers.last_name,
    COUNT(Reservations.reservation_id) AS total_reservations
FROM
    Customers
INNER JOIN
    Reservations ON Customers.user_id = Reservations.user_id
GROUP BY
    Customers.user_id, Customers.first_name, Customers.last_name;

SELECT
    Vehicles.model,
    SUM(Reservations.total_price) AS total_revenue
FROM
    Vehicles
INNER JOIN
    Reservations ON Vehicles.vehicle_id = Reservations.vehicle_id
GROUP BY
    Vehicles.model;

SELECT
    Vehicles.model,
    AVG(VehicleReviews.review_rating) AS average_rating
FROM
    Vehicles
INNER JOIN
    VehicleReviews ON Vehicles.vehicle_id = VehicleReviews.vehicle_id
GROUP BY
    Vehicles.model;

SELECT
    Vehicles.model,
    COUNT(AccidentReports.report_id) AS total_accidents
FROM
    Vehicles
INNER JOIN
    AccidentReports ON Vehicles.vehicle_id =
AccidentReports.vehicle_id
GROUP BY
    Vehicles.model;
```

26

```sql
SELECT
    Branches.branch_name,
    COUNT(Reservations.reservation_id) AS total_reservations
FROM
    Branches
INNER JOIN
    Vehicles ON Branches.branch_id = Vehicles.branch_id
INNER JOIN
    Reservations ON Vehicles.vehicle_id = Reservations.vehicle_id
GROUP BY
    Branches.branch_name;
```