Inhouse Car Rental System

(Batch No 17)

Version Number	Date	Author/Owner	Description of Change
1.1	05.11.2021	Harshit Jain	Initial release

Contents

1. Team Members and their Responsibilities:

Member	Responsibility
Beenish 2021mt12234	ER Diagram, Relational Database Schema
Harshit Jain 2021mt12448	Database development, Application development
Nazia Sultana 2020mt93518	Normalization
Rajesh Kumar 2021mt12158	Problem statement, documentation
Ranjit Singh Gill 2020mt93742	SQL query, Object Model Diagram

2. Problem Statement & Requirements Definition

Inhouse Car Rental System:

Companies need to hire a third party for hiring taxi for the official visits of employees. Where a lot of dependencies exist on third party working like working days, staff availability, delays in response etc which are outside the control of the company. We propose an automated inhouse car rental system which will be completely managed by company administrators.

Inhouse Car Rental System is an application where the employee can book the car on rent. Within the company the employee can book the car from the available car which has been registered for the services and later return the car.

Database for cars registered for service and employee details will be maintained.

Overview and Functionality:

The Inhouse Car Rental System consists of two entities and two relations. Here's the brief introduction of each:

Car: it keeps the identifying attributes of a car that are car identification number, car registration number, model, make and car availability.

Customer: It represents the employee of the company for this system and has the following attributes : Customer identification number, name, address and contact number.

Rents: it defines the relationship between car and customer. It has the following attributes : rent identification number, start date, end date and fee.

Returns : it also defines the relationship between car and customer. It has the following attributes : return identification number, return date, fine and Elapsed time which is count of days from current date to rent_end date.

Some of the allowed operations are shown in following figures:

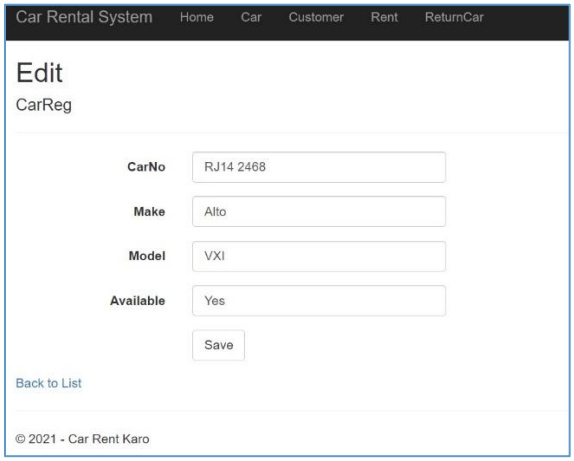


Fig 2.1

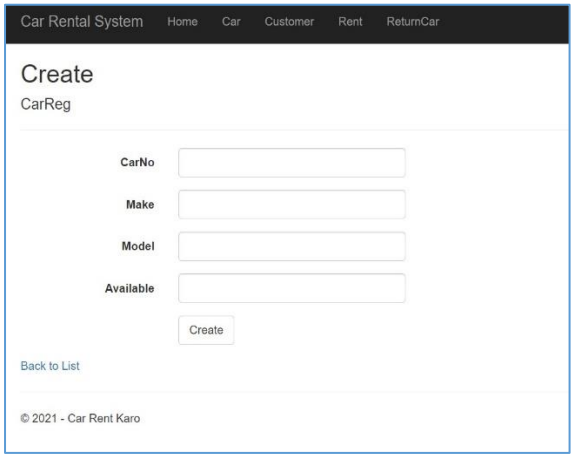


Fig 2.2

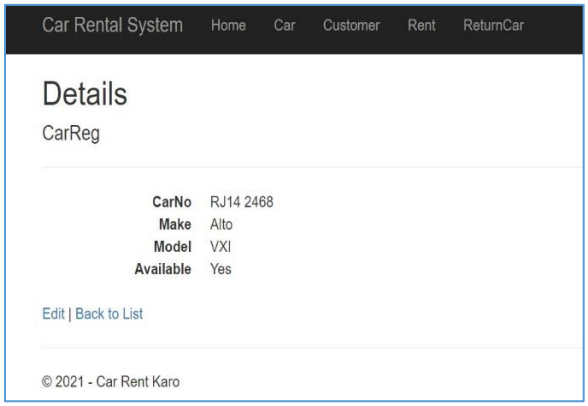


Fig 2.

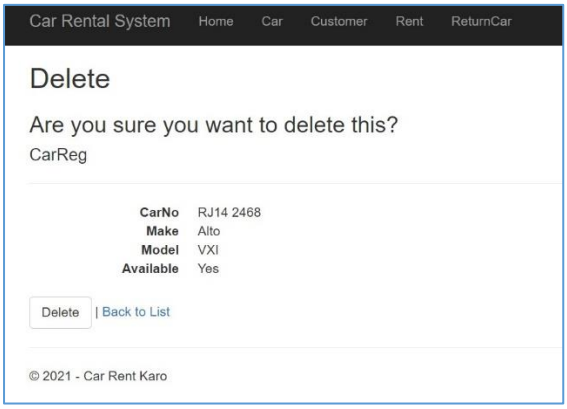


Fig 2.2

System Requirements:

Hardware Specification:

- Minimum PIV 2.8 GHz Processor
- Minimum RAM 2048MB
- Minimum HDD 20 GB Hard Disk Space

Software Specification:

- WINDOWS 10
- Visual Studio 2022 Community
- Visual Studio .Net Framework 4.5
- SQL Server 2019/SSMS 2018

3. Entity Relationship Model

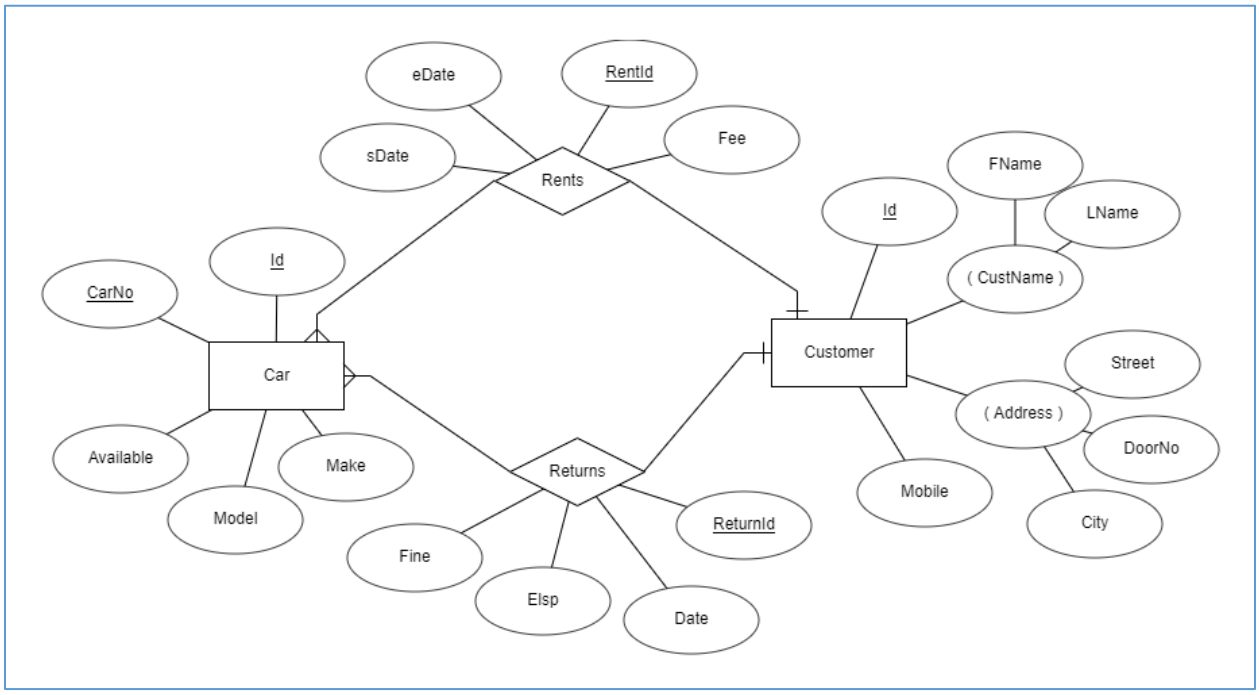


Fig 3.1

Entities

The identified entities along with their attributes in our car rental system are as follows:

1. **Car**
    a. Id – A unique identity number assigned to the car by our system, serves as a **Primary key.** ***Simple, single valued and stored attribute.***
    b. CarNo – Vehicle identification number (VIN), also unique hence serves as a **Candidate key.** ***Simple, single valued and stored attribute.***
    c. Available – To know if a car is available for rent. ***Simple, single valued and stored attribute.***
    d. Model – Model of the car. ***Simple, single valued and stored attribute.***
    e. Make – Brand of the car. ***Simple, single valued and stored attribute.***
2. **Customer**
    a. Id – A unique identity number assigned to the customer by our system, serves as Primary key. ***Simple, single valued and stored attribute.***
    b. CustName – Name of the customer. A **composite attribute** comprised of **Fname** and **Lname**. ***Composite, single valued and stored attribute.***
    c. Address – Address of the customer renting a car. A **composite attribute** comprised of **DoorNo**, **Street** and **City**. ***Composite, single valued and stored attribute.***

Relationships

1. **Rents –** Customer rents Car
    - **Relationship attributes:**
        i. **RentId:** When a car is rented, the event is noted and is assigned a unique identification number, serves as a **Primary key.**
        ii. **sDate:** Date on which customer rents the car

iii. **eDate:** Date on which customer commits to return the car.
iv. **Fee:** Based on eDate and sDate a fee is paid by customer before it can rent a car.

- **Cardinality Constraints:**
  1:N as a customer can rent multiple cars but a single car can be rented to a single customer only at a time.
- **Participation Constraints:**
  Partial from both the sides, as not all the customers will be renting a car. In the same manner, not all the cars will be rented by some or other customer.

2. **Returns** – Customer returns Car
   - **Relationship attributes**:
     i. **ReturnId**: When a car is returned, the event is noted and is assigned a unique identification number, serves as a **Primary key**.
     ii. **Date**: Date on which the car is returned.
     iii. **Fine**: If the actual return date exceeds the committed return date, a fine is imposed on the customer.
     iv. **Elsp**: Count of days
   - **Cardinality Constraints**
     1:N as a customer can return multiple cars but a single car will be returned by only one customer at a time.
   - **Participation Constraints:**

     Partial from both the sides, as not all the customers will be returning the car as well as not all the cars will be returned.

4. Object Model Diagram

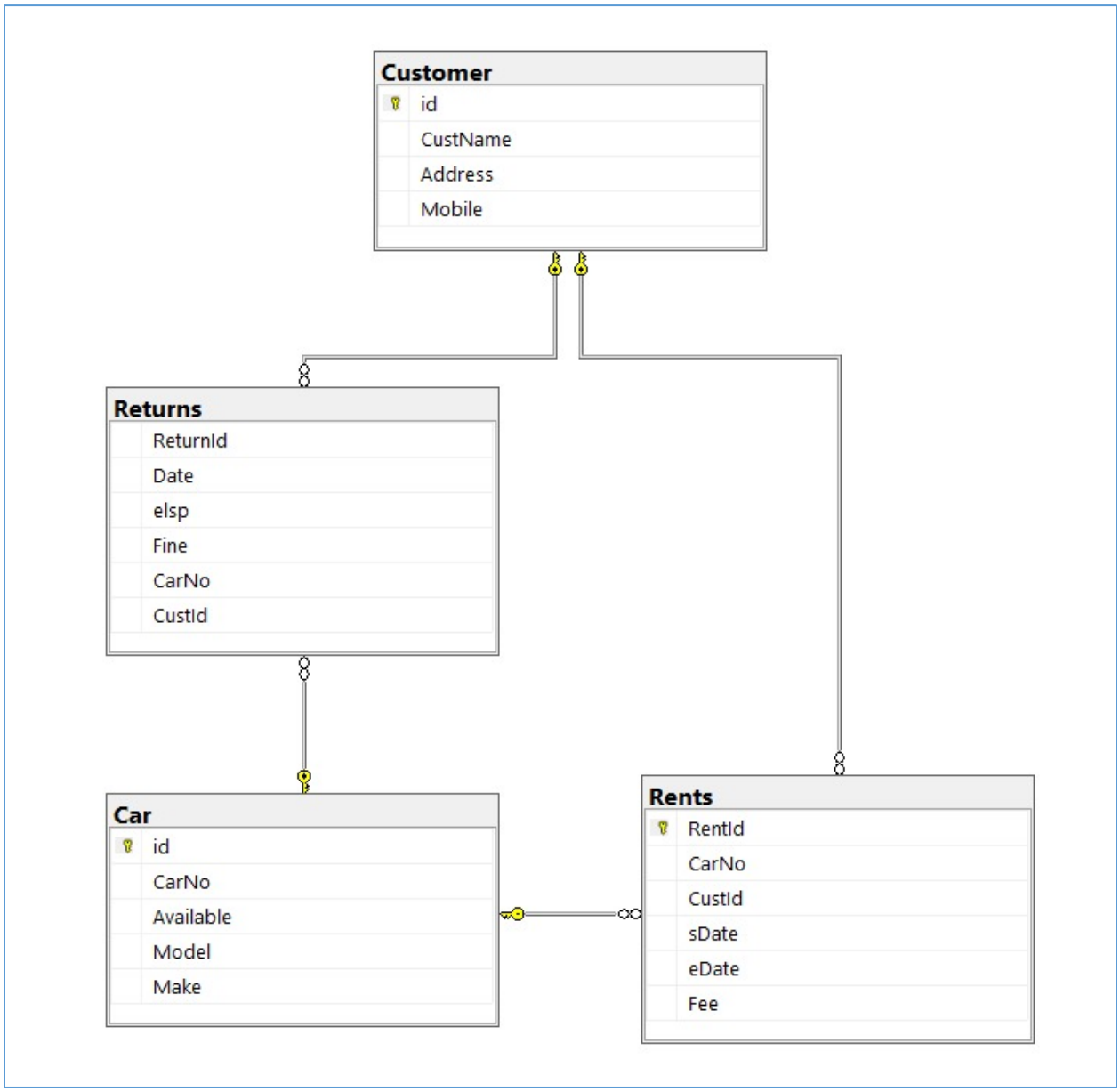


Fig 4.1

5. Relational Database Schema

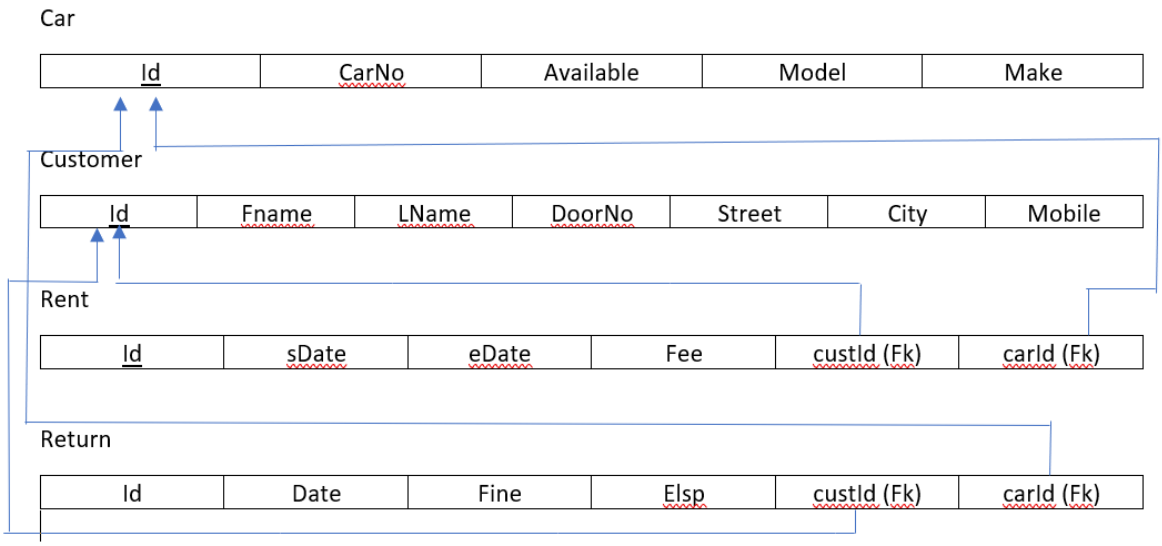


Fig 5.1

6. Normalization

(a) Details of schema refinement

We have divided the tables according to 3NF. **Returns**, **Rents** and **Car** is having custId as foreign key. **Customer** table is having the details about the customer, and its key has been used as a joint relation between the other tables.

There are no transitive functional dependencies, no multivalued attributes and no partial dependencies, hence our table is in 3NF.

These tables cannot be further decomposed to attain higher normal form types of normalization in DBMS. In fact, it is already in higher normalization forms. Separate efforts for moving into next levels of normalizing data are normally needed in complex databases

(b) Soft copy details /screen shots of changes made

Customer

	Column Name	Data Type	Allow Nulls
🔑	id	int	☐
	custname	varchar(50)	☐
	address	varchar(50)	☐
	mobile	varchar(50)	☐
			☐

Fig 6.1

Car

	Column Name	Data Type	Allow Nulls
🔑	id	int	☐
	carno	varchar(50)	☐
	make	varchar(50)	☐
	model	varchar(50)	☑
	available	varchar(50)	☐
			☐

Fig 6.2

Returns

	Column Name	Data Type	Allow Nulls
🔑	id	int	☐
	carno	varchar(50)	☐
	custid	int	☐
	date	date	☐
	elsp	int	☐
	fine	int	☐
			☐

Fig 6.3

Rents

	Column Name	Data Type	Allow Nulls
🔑	id	int	☐
	carid	int	☐
	custid	int	☐
	fee	int	☐
	sdate	date	☐
	edate	date	☐
			☐

Fig 6.4

7. Table Definitions and Data Contents

Table Definitions: Fig 7.1 shows the definitions of the tables i.e. Customer, Rents, Car and Returns

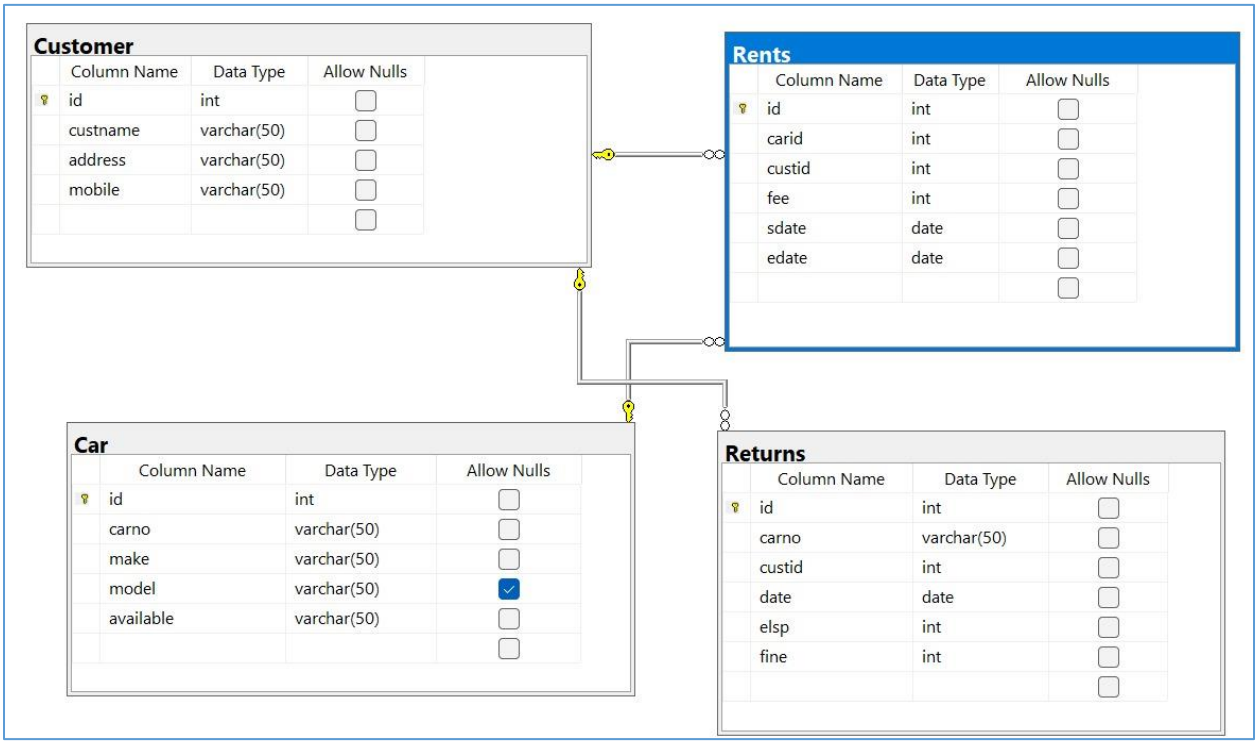
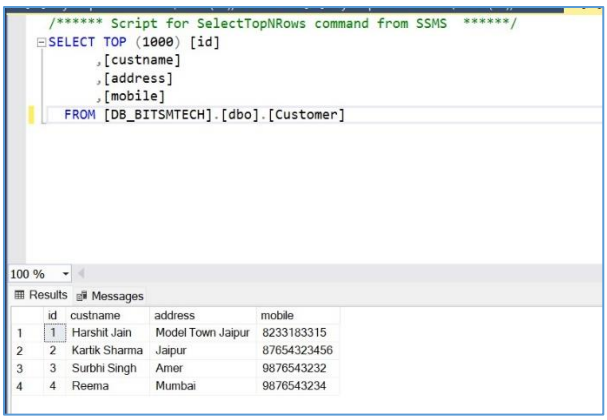


Fig 7.1

Data Contents:



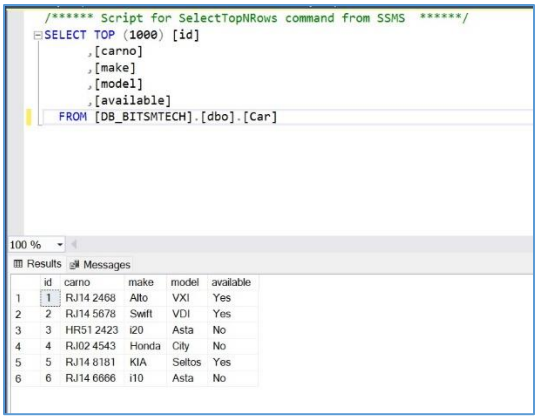
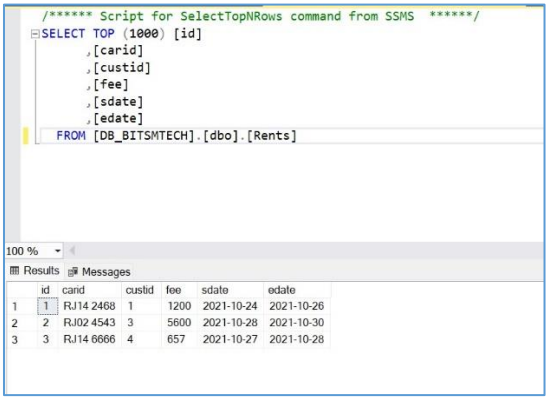


Fig 7.2 : Data contents for Customer

Fig 7.3 : Data contents for Car



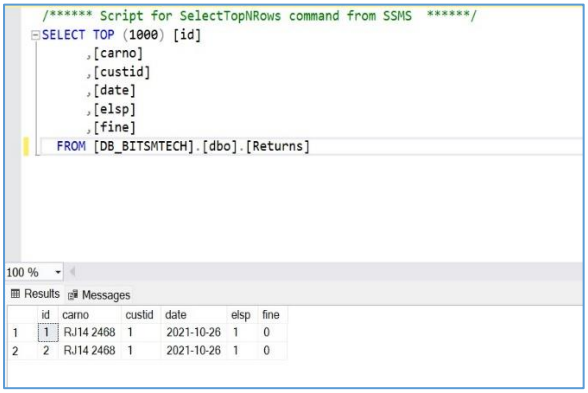
```
/****** Script for SelectTopNRows command from SSMS  ******/
SELECT TOP (1000) [id]
      ,[carid]
      ,[custid]
      ,[fee]
      ,[sdate]
      ,[edate]
  FROM [DB_BITSMTECH].[dbo].[Rents]
```

100 %

Results | Messages

	id	carid	custid	fee	sdate	edate
1	1	RJ14 2468	1	1200	2021-10-24	2021-10-26
2	2	RJ02 4543	3	5600	2021-10-28	2021-10-30
3	3	RJ14 6666	4	657	2021-10-27	2021-10-28

Fig 7.2 : Data contents for Rents



```
/****** Script for SelectTopNRows command from SSMS  ******/
SELECT TOP (1000) [id]
      ,[carno]
      ,[custid]
      ,[date]
      ,[elsp]
      ,[fine]
  FROM [DB_BITSMTECH].[dbo].[Returns]
```

100 %

Results | Messages

	id	carno	custid	date	elsp	fine
1	1	RJ14 2468	1	2021-10-26	1	0
2	2	RJ14 2468	1	2021-10-26	1	0

Fig 7.3 : Data contents for Returns

8. SQL Statements

Following script creates the table Customer :

```
USE [DB_BITSMTECH]
GO

/****** Object:  Table [dbo].[Customer]    Script Date: 10/20/2021 9:44:28 AM ******/
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO

CREATE TABLE [dbo].[Customer](
    [id] [int] NOT NULL,
    [custname] [varchar](50) NOT NULL,
    [address] [varchar](50) NOT NULL,
    [mobile] [varchar](50) NOT NULL,
 CONSTRAINT [PK_Customer] PRIMARY KEY CLUSTERED
(
    [id] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
```

script for creating table Car :

```
USE [DB_BITSMTECH]
GO

/****** Object:  Table [dbo].[Car]    Script Date: 10/20/2021 9:43:40 AM ******/
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO

CREATE TABLE [dbo].[Car](
    [id] [int] NOT NULL,
    [carno] [varchar](50) NOT NULL,
    [make] [varchar](50) NOT NULL,
    [model] [varchar](50) NULL,
    [available] [varchar](50) NOT NULL,
 CONSTRAINT [PK_CarReg] PRIMARY KEY CLUSTERED
(
    [id] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
```

script for creating table Rents :

```
USE [DB_BITSMTECH]
GO

/****** Object:  Table [dbo].[Rents]    Script Date: 10/20/2021 9:44:37 AM ******/
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO

CREATE TABLE [dbo].[Rents](
    [id] [int] NOT NULL,
    [carid] [int] NOT NULL,
    [custid] [int] NOT NULL,
    [fee] [int] NOT NULL,
    [sdate] [date] NOT NULL,
    [edate] [date] NOT NULL,
 CONSTRAINT [PK_Rents] PRIMARY KEY CLUSTERED
(
    [id] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
```

```
ALTER TABLE [dbo].[Rents]  WITH CHECK ADD  CONSTRAINT [FK_Rents_Car] FOREIGN KEY([carid])
REFERENCES [dbo].[Car] ([id])
GO

ALTER TABLE [dbo].[Rents] CHECK CONSTRAINT [FK_Rents_Car]
GO

ALTER TABLE [dbo].[Rents]  WITH CHECK ADD  CONSTRAINT [FK_Rents_Customer] FOREIGN KEY([custid])
REFERENCES [dbo].[Customer] ([id])
GO

ALTER TABLE [dbo].[Rents] CHECK CONSTRAINT [FK_Rents_Customer]
GO
```

script for creating table Returns :

```
USE [DB_BITSMTECH]
GO

/****** Object:  Table [dbo].[Returns]    Script Date: 11/9/2021 9:45:04 AM ******/
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO

CREATE TABLE [dbo].[Returns](
    [id] [int] NOT NULL,
    [carno] [varchar](50) NOT NULL,
    [custid] [int] NOT NULL,
    [date] [date] NOT NULL,
    [elsp] [int] NOT NULL,
    [fine] [int] NOT NULL,
 CONSTRAINT [PK_Returns] PRIMARY KEY CLUSTERED
(
    [id] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
```

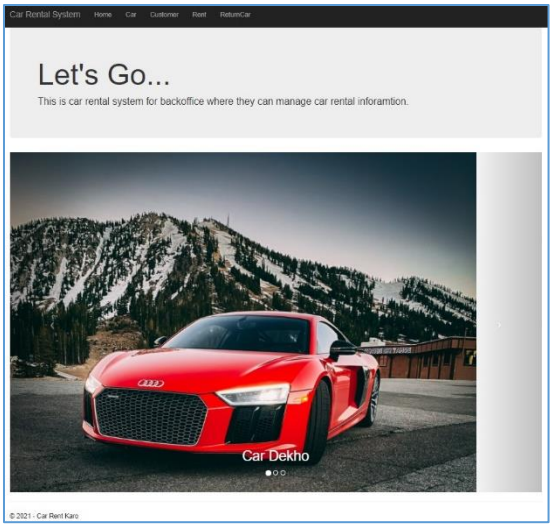
```
ALTER TABLE [dbo].[Returns]  WITH CHECK ADD  CONSTRAINT [FK_Returns_Customer] FOREIGN KEY([custid])
REFERENCES [dbo].[Customer] ([id])
GO

ALTER TABLE [dbo].[Returns] CHECK CONSTRAINT [FK_Returns_Customer]
GO
```

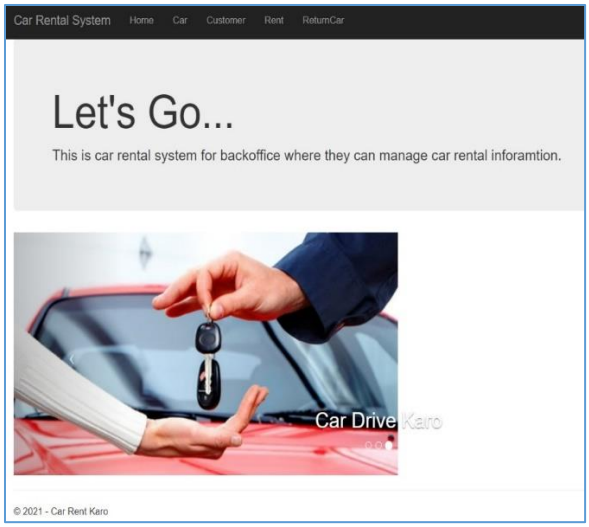
9. Stored Procedures/ Triggers

Not applicable

10. User Interface and Database connectivity

USER Interface:





Database connectivity:

Fig 10.2 shows the connectivity established with database with application.

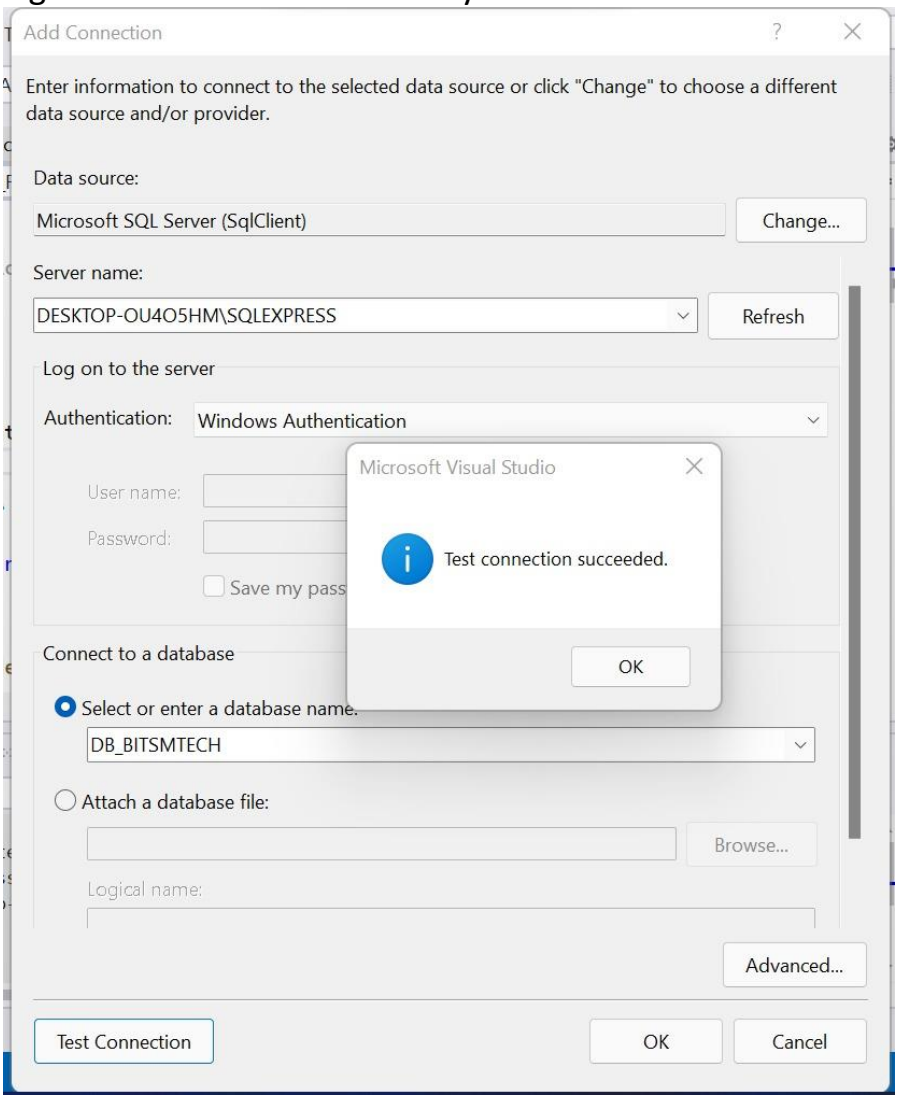


Fig 10.2

END OF DOCUMENT