Project Report

BUS RESERVATION SYSTEM

MODULE TITLE: ADVANCED DATABASES

MODULE CODE: B9IS100

GROUP MEMBERS:

SINGUPURAM NAVIN KUMAR | 10612366

DIGVIJAY JATKAR | 10508676

TUBA TOSUNOGLU| 10625974

MODULE LEADER: DR SHAZIA A AFZAL

Table of Contents

TABLE OF FIGURES

1. ABSTRACT

Bus Reservation system -

Bus Reservation System focuses on providing consumers with bus rentals. Customers can view available buses, register, view profiles, and book a bus using this online system. Management of both the people and the vehicles is involved in this. The system unifies all tasks for managing the bus rental company's and its workers' workloads. All the data required for their management is already in the system. After making reservations, it can schedule your employees' full day.

2. SCOPE OF THE DATABASE

To accomplish the project's goals, research must be done in a variety of disciplines, from commercial principles to the realm of computing. The region includes

Reservations for buses: This involves an investigation into the operation of the bus reservation industry, the steps involved, and potential areas for improvement. the method via which the application was created technologically Both regular clients and employees of the organization will be able to use the system efficiently. Because the system is a web platform, it will always be accessible, barring any predicted infrequent transitory server issues.

1. Bus

Attribute Name	Data Type	Description
BusId	INT(10)	Id for uniquely identifier
BusName	VARCHAR(25)	Name designated to identify bus
BusEngineNumber	VARCHAR(25)	Engine number of a specific bus
BusChassisNumber	VARCHAR(25)	Chassis number of a bus designated
BusModelNumber	VARCHAR(25)	Model number of a bus
BusPrice	INT(10)	Rental price of a bus
BusFuelTypes	VARCHAR(10)	(Gasoline/Petrol/Diesel) type of fuel
BusMileage	INT(10)	Average(per Km bus)

BusPolicy	VARCHAR(25)	Policies attained
BusSeatingCapacity	INT(5)	Seating capacity for the persons
BusRegistrationDate	VARCHAR(20)	Registrated date
BusRating	INT(5)	User rating
CreatedAt	TIMESTAMP	The date on which the bus details added to DB

Primary Key: BusId

Required Attributes: BusId, BusName, BusEngineNumber, BusChassisNumber, BusModelNumber, BusPrice, BusFuelTypes, BusSeatingCapacity, BusPolicy BusRegistrationDate, BusRating, CreatedAt

Optional Attributes: BusMileage, BuFuelTankCC, BusRating

2. Customer

Attribute Name	Data type	Description
CustomerId	INT(10)	Identify customer uniquely
CustomerFirstName	VARCHAR(50)	First name of the customer
CustomerLastName	VARCHAR(50)	Last name of the customer
CustomerAddress	TEXT	Address of the customer
CustomerMobileNumber	VARCHAR(20)	Mobile number of the customer
CustomerRole	VARCHAR(10)	Role
CustomerDob	VARCHAR(15)	date of birth
CustomerEmail	VARCHAR(25)	email
CustomerPassword	VARCHAR(20)	password
CustomerLicense	VARCHAR(20)	license number
CreatedAt	TIMESTAMP	The date registered

Primary Key: CustomerId

Required Attributes: CustomerFirstName, , CustomerAddress, CustomerMobileNumber, CustomerDob, CustomerEmail , CustomerPassword, CustomerLicense, CreatedAt

Optional Attributes: CustomerLastName

3. Bus_type

Attribute Name	Data Type	Description
BusTypeId	INT(10)	Identifies every bus
BusId	INT(10)	Bus Id foreign key Bus Table
BusPowerSteering	Varchar(20)	Type of power steering
BusAirConditioner	Varchar(20)	Type of AC
BusPassengerAirbag	Varchar(20)	Type of Airbag passenger
BusDriverAirbag	Varchar(20)	Type of Airbag driver
BusSunRoof	Varchar(20)	Type of SunRoof
BusDriverSeat	Varchar(20)	Type of driver seat
BusFogLights	Varchar(20)	Type of bus fog lights
BusType	VARCHAR(20)	Type of bus
CreatedAt	TIMESTAMP	Date Registered

Primary Key: BusTypeId

Foreign Key: BusId
Attributes:BusPowerSteering, BusAirConditioner, BusPassengerAirbag, BusDriverAirbag, BusDriverSeat, BusType, BusType
Optional Attributes: BusSunRoof, BusFogLights

4. Billing

Attribute Name	Data type	Description
BillingId	INT(10)	Unique Identifier
ReservationId	INT(10)	Id for reservation details
BillingMode	VARCHAR(20)	Tyes of Billing Mode
BusTransactioAmount	INT(20)	Total amount of booked bus
AddPromocode	VARCHAR(20)	Promo code for special discount
AddTaxCharges	VARCHAR(10)	Addition tax charges
BusTransactionStatus	VARCHAR(10)	Transaction status
BusBillingCurrency	VARCHAR(20)	Currency of billing amt
CreatedAt	TIMESTAMP	Creatd Date of Billing
UpdatedAt	TIMESTAMP	Updatd Date of Billing

Primary Key: - BillingId
Foreign Key: - ReservationId
Required Attributes: - BillingMode, BusTransactioAmount, BusBillingCurrency, AddTaxCharges
Optional Attributes: - AddPromocode, CreatedAt, UpdatedAt

5. Bus Rental Type

Attribute Name	Data Type	Description
BusRentalTypId	INT(10)	uniquely identify
BusRentalTripSelection	VARCHAR(20)	Roundtrip or Oneway Booking
BusRentalSeaters	INT(10)	Seaters of total sittings
BusRentalFuelType	VARCHAR(10)	Fuel Type describes
BusRentalType	VARCHAR(20)	Type defines buses
BusRentalAutoPilot	INT	Auto Pilot option
BusRentalStatus	VARCHAR(10)	Check the particular bus available
BusRentalStartDt	VARCHAR(20)	Start date
BusRentalEndDt	VARCHAR(20)	End date

Primary Key : BusRentalTypId

6. Reservation

Primary Key: - ReservationId

Foreign Key: - BusId, CustomerId, BusRentalTypeId

Attributes: - ReservationStartLocation, ReservationEndLocation, ReservationDuration, ReservationTripAmount, BusKmStart, BusKmEnd, BusFuelStart, BusFuelEnd, ReservationStatus, CreatedAt, UpdatedAt

Attribute Name	Data Type	Description
ReservationId	INT(10)	Each reservation has unique Id
BusId	INT(10)	Unique identifier for a Bus
CustomerId	INT(10)	Customer Id for customer details
BusRentalTypeId	INT(10)	Identity for rental type
ReservationStartLocation	VARCHAR(25)	Start Location
ReservationEndLocation	VARCHAR(25)	end Location
ReservationDuration	INT(5)	duration
ReservationTripAmount	INT(20)	Total Amount chargeable
BusKmStart	INT(20)	Bus's Kilometres at the start
BusKmEnd	INT(20)	Bus's Kilometres at the end
BusFuelStart	INT(10)	Bus's Fuel Percentage at the start
BusFuelEnd	INT(10)	Bus's Fuel Percentage at the end
ReservationStatus	VARCHAR(20)	Reservation's status
CreatedAt	TIMESTAMP	creation date
UpdatedAt	TIMESTAMP	updating date

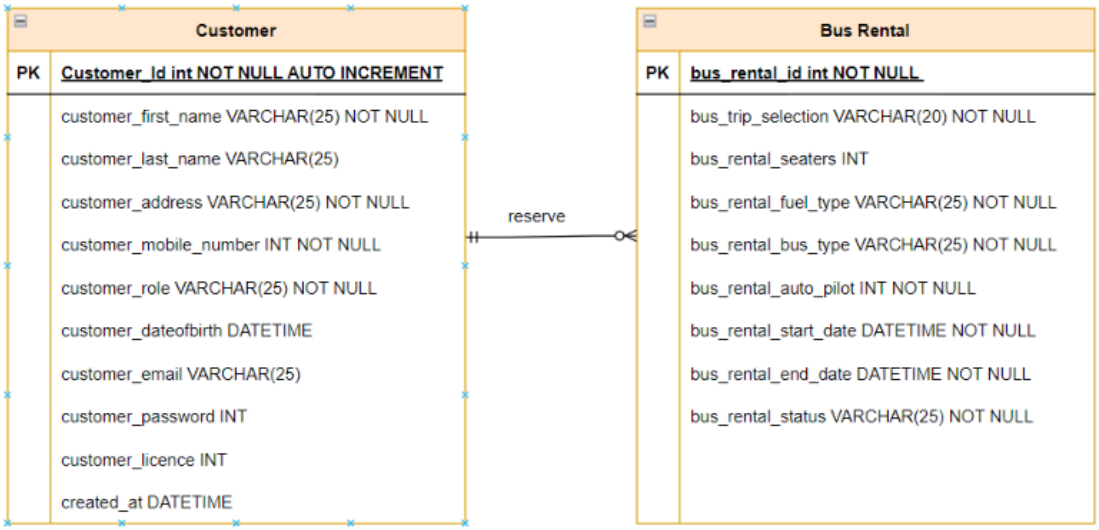
7. Reimbursement

Primary Key: - ReimbursementId Foreign Key: - ReservationId
RequiredAttributes: DeductionReimbursedamount, ReimbursementStatus, ReimbursementAmount, ReimbursementAccount, CreatedAt

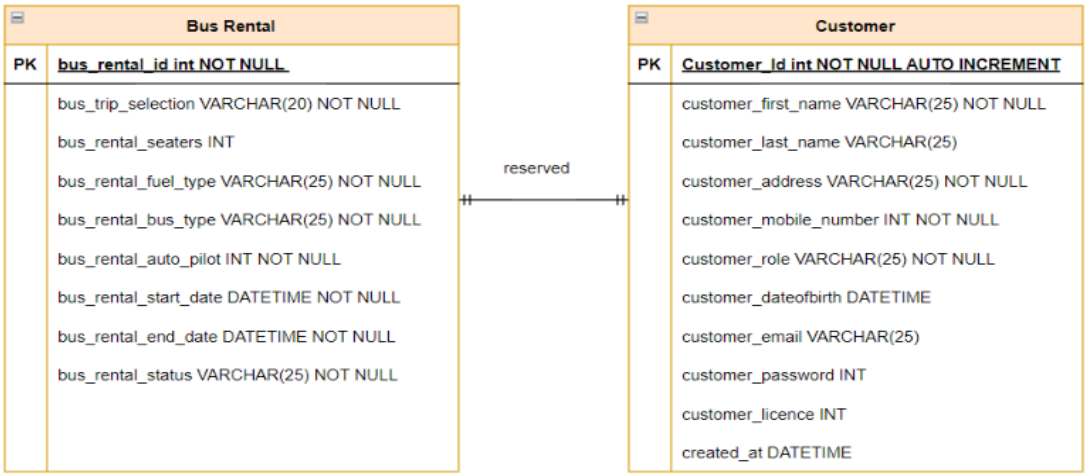
Attribute Name	Data Type	Description
ReimbursementId	INT(10)	Identify every unique reimbursement
ReservationId	INT(10)	Referred to the reservation
DeductionReimbursedamount	INT(10)	Reimbursed transaction amount
ReimbursementStatus	VARCHAR(20)	Status
ReimbursementAmount	INT(20)	Reimbursement amount
ReimbursementAccount	VARCHAR(20)	Amount of a reimbursement
CreatedAt	TIMESTAMP	Initiation date

3. BUSINESS RULES

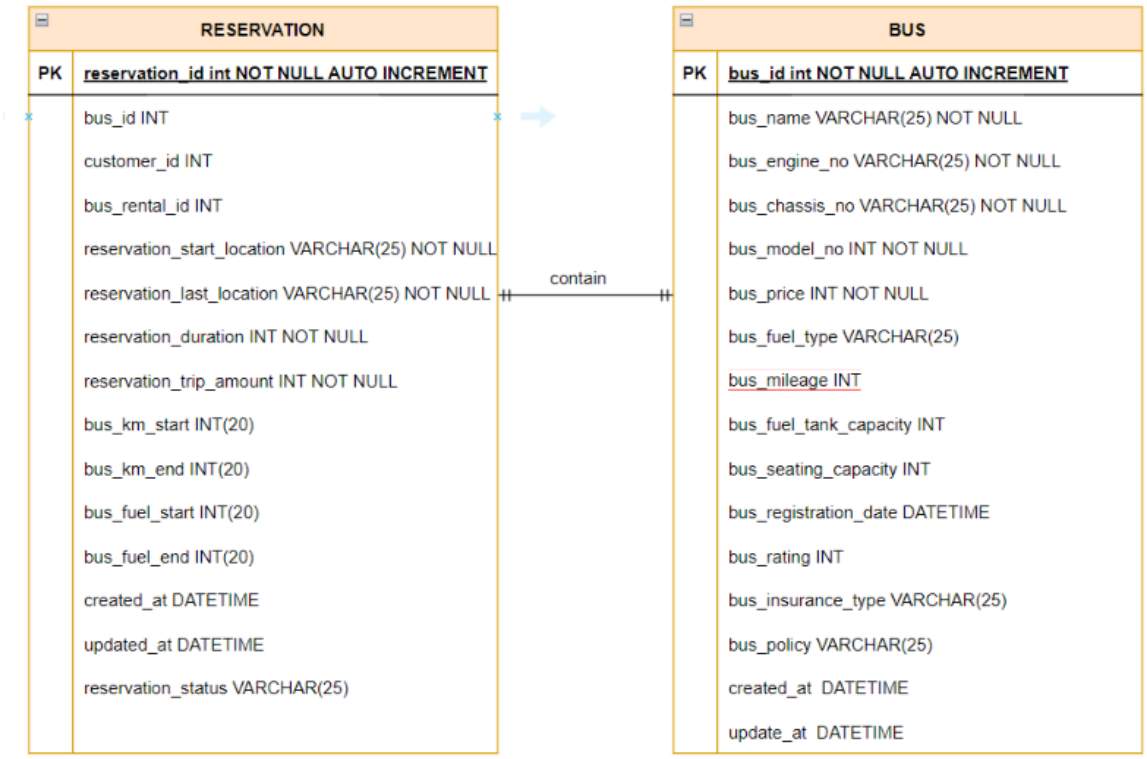
1. One to many dependencies from customer to bus rentals



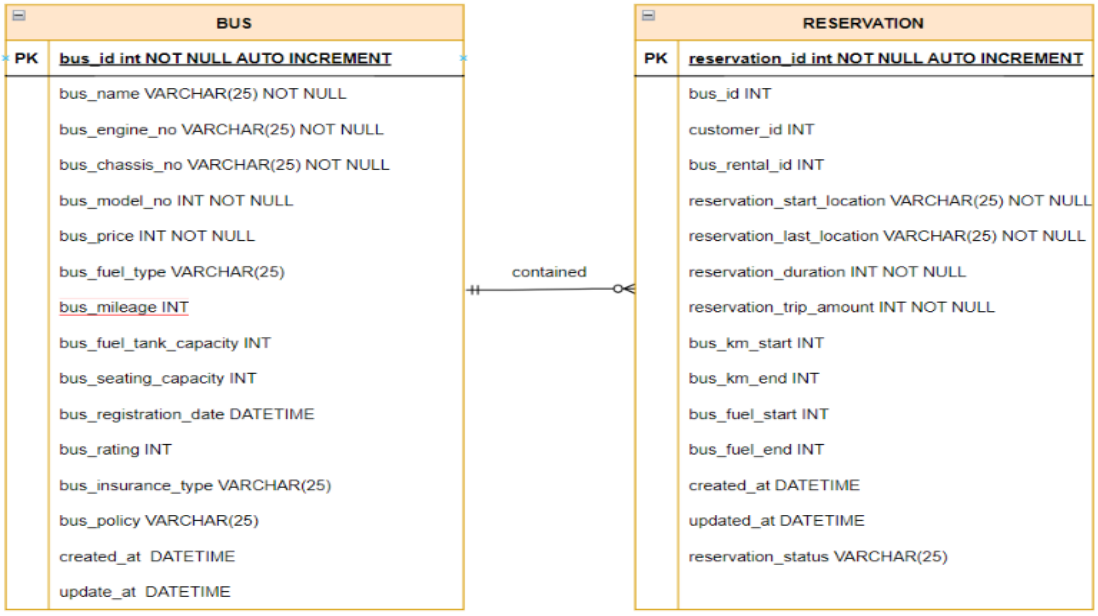
2. One to one mapping from bus rental to customer



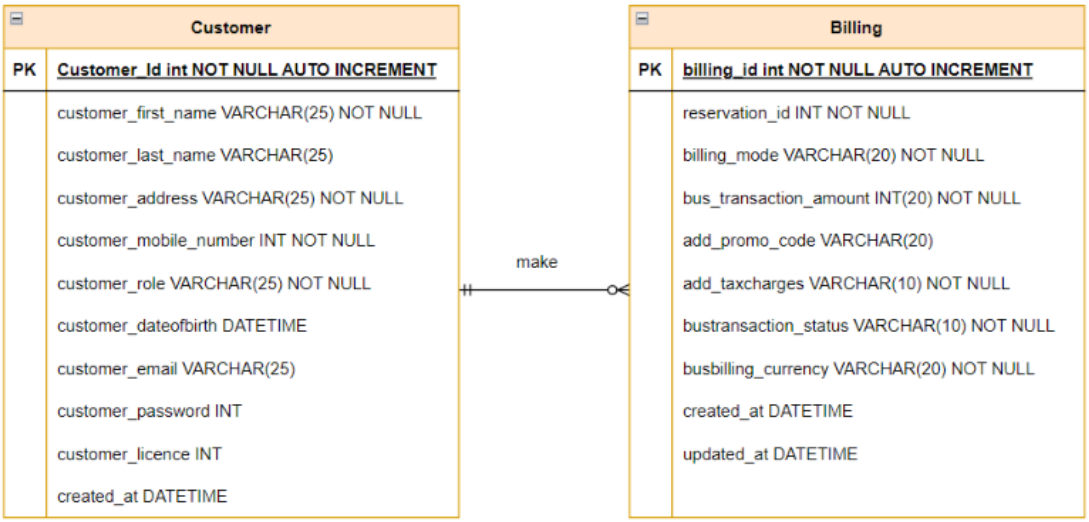
3. One reservation can contain one bus



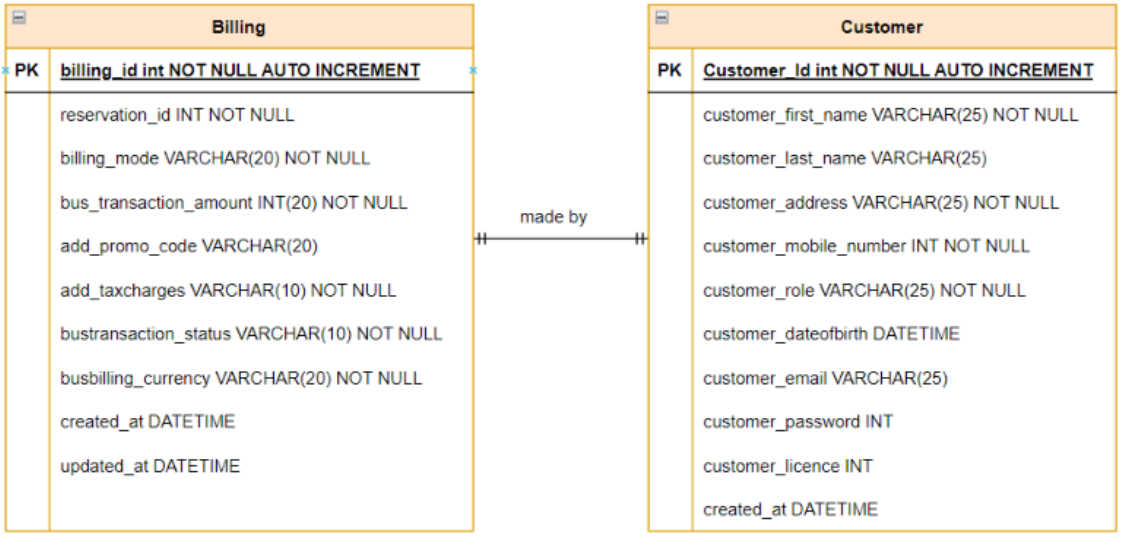
4 . One bus can be contained in many reservations



5. One customer can make many billing



6. One billing can be made by one customer



4. RELATIONAL SCHEMA FOR HYBRID IN 3NF

Data in a database can be organized using the normalization process. Reduced redundancy from a relation or collection of relations is achieved through normalization. Additionally, it aids in overcoming unfavorable traits like Insertion, Update, and Deletion Anomalies. The bigger table is split into smaller tables, and they are connected through relationships.

Below are the normalization forms used:

1. First Normalization Form (1NF) of an entity describes if only

   a. It holds multiple kind of values or data.

   b. Most of the values stored shold be atomic

2. Second Normalization Form (1NF) of an entity describes if only

   a. It should be in the first normalization form.

   b. Every non-key attribute needs to be reliant on the main key. (No partial reliance)

3. Third Normalization Form (1NF) of an entity describes if only

   a. It should be in the second normalization form.

   b. There should be no transitive dependencies in it.

1. Bus Table

a. Primary Key is defined (BusId).

b. There is an atomic value for each column

c. Groups should not repeatable.

Hence, the bus table is in the First Normalisation Form.

a. The table is in First Normalisation Form

b. The primary key controls every non-key attribute.

Hence, the table is in the Second Normalisation Form.

a. The table is in the Second Normalisation Form.

b. There are no transitive dependencies present.

Hence The table is in the Third Normalisation Form.

2. Customer Table

a. Primary Key is defined(CustomerId).

b. There is an atomic value for each column.

c. Groups should not repeat

Hence, The table is in the First Normalisation Form.

a. The table is in the First Normalisation Form

b. The primary key controls every non-key attribute.

Hence, the table is in the Second Normalisation Form.

a. The table is in the Second Normalisation Form.

b. There are no transitive dependencies present.

Hence The table is in the Third Normalisation Form.

3. Bus Type Table

a. Primary Key is defined(BusTypeId).

b. There is an atomic value for each column.

c. Groups should not repeat

Hence, The table is in the First Normalisation Form.

c. The table is in the First Normalisation Form

d. The primary key controls every non-key attribute.

Hence, the table is in the Second Normalisation Form.

b. The table is in the Second Normalisation Form.

b. There are no transitive dependencies present.

Hence The table is in the Third Normalisation Form.

4. Billing Table

a. Primary Key is defined(BillingId).

b. There is an atomic value for each column.

c. Groups should not repeat

Hence, The table is in the First Normalisation Form.

e. The table is in the First Normalisation Form

f. The primary key controls every non-key attribute.

Hence, the table is in the Second Normalisation Form.

c. The table is in the Second Normalisation Form.

b. There are no transitive dependencies present.

Hence The table is in the Third Normalisation Form.

5. Bus Rental Table

a. Primary Key is defined(BusRentalTypeId).

b. There is an atomic value for each column.

c. Groups should not repeat

Hence, The table is in the First Normalisation Form.

g. The table is in the First Normalisation Form

h. The primary key controls every non-key attribute.

Hence, the table is in the Second Normalisation Form.

d. The table is in the Second Normalisation Form.

b. There are no transitive dependencies present.

Hence The table is in the Third Normalisation Form.

6. Bus Reservation Table

a. Primary Key is defined(ReservationId).

b. There is an atomic value for each column.

c. Groups should not repeat

Hence, The table is in the First Normalisation Form.

i. The table is in the First Normalisation Form

j. The primary key controls every non-key attribute.

Hence, the table is in the Second Normalisation Form.

e. The table is in the Second Normalisation Form.

b. There are no transitive dependencies present.

Hence The table is in the Third Normalisation Form.

7. Reimbursement Table

a. Primary Key is defined(ReimbursementId).
b. There is an atomic value for each column.

c. Groups should not repeat

Hence, The table is in the First Normalisation Form.

a.The table is in the First Normalisation Form

b.The primary key controls every non-key attribute.

Hence, the table is in the Second Normalisation Form.

c. The table is in the Second Normalisation Form.

d. There are no transitive dependencies present.

Hence The table is in the Third Normalisation Form.

4.1. XML HYBRID DATABASE SCHEMA

In this instance, the outside system is considered. Refunds are processed, as well as a Xml data form is returned in response.

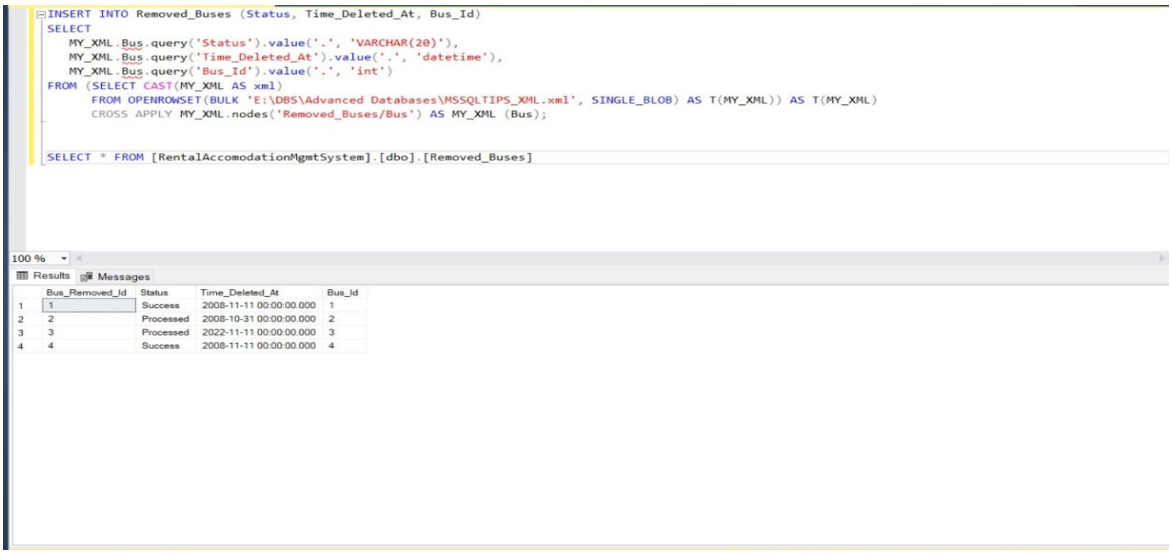
We employed insertion procedures that read the xml document below from a certain region, read it, and then entered the data into a table in order to establish a hybrid database.


```xml
<?xml version="1.0" encoding="utf-8"?>
<Removed_Buses>
  <Bus>
    <Status>Success</Status>
    <Time_Deleted_At>2008-11-11</Time_Deleted_At>
    <Bus_Id>1</Bus_Id>
  </Bus>
  <Bus>
    <Status>Processed</Status>
    <Time_Deleted_At>2008-10-31</Time_Deleted_At>
    <Bus_Id>2</Bus_Id>
  </Bus>
  <Bus>
    <Status>Processed</Status>
    <Time_Deleted_At>2022-11-11</Time_Deleted_At>
    <Bus_Id>3</Bus_Id>
  </Bus>
  <Bus>
    <Status>Success</Status>
    <Time_Deleted_At>2008-11-11</Time_Deleted_At>
    <Bus_Id>4</Bus_Id>
  </Bus>
</Removed_Buses>
```


Figure 1. XML DIAGRAM

In addition to rules governing data content and semantics, such as what fields an element can include, which sub elements it can contain, and how many items can be present, XML schema determines the shape, or structure, of an XML document. Additionally, it can specify the kind and range of values that can be assigned to each element or property. Facets are XML data restrictions that provide guidelines like minimum and maximum lengths.

```
INSERT INTO Removed_Buses (Status, Time_Deleted_At, Bus_Id)
SELECT
  MY_XML.Bus.query('Status').value('.', 'VARCHAR(20)'),
  MY_XML.Bus.query('Time_Deleted_At').value('.', 'datetime'),
  MY_XML.Bus.query('Bus_Id').value('.', 'int')
FROM (SELECT CAST(MY_XML AS xml)
    FROM OPENROWSET(BULK 'E:\DBS\Advanced Databases\MSSQLTIPS_XML.xml', SINGLE_BLOB) AS T(MY_XML)) AS T(MY_XML)
    CROSS APPLY MY_XML.nodes('Removed_Buses/Bus') AS MY_XML (Bus);
SELECT * FROM [RentalAccomodationMgmtSystem].[dbo].[Removed_Buses]
```



5. DATA DIAGRAM

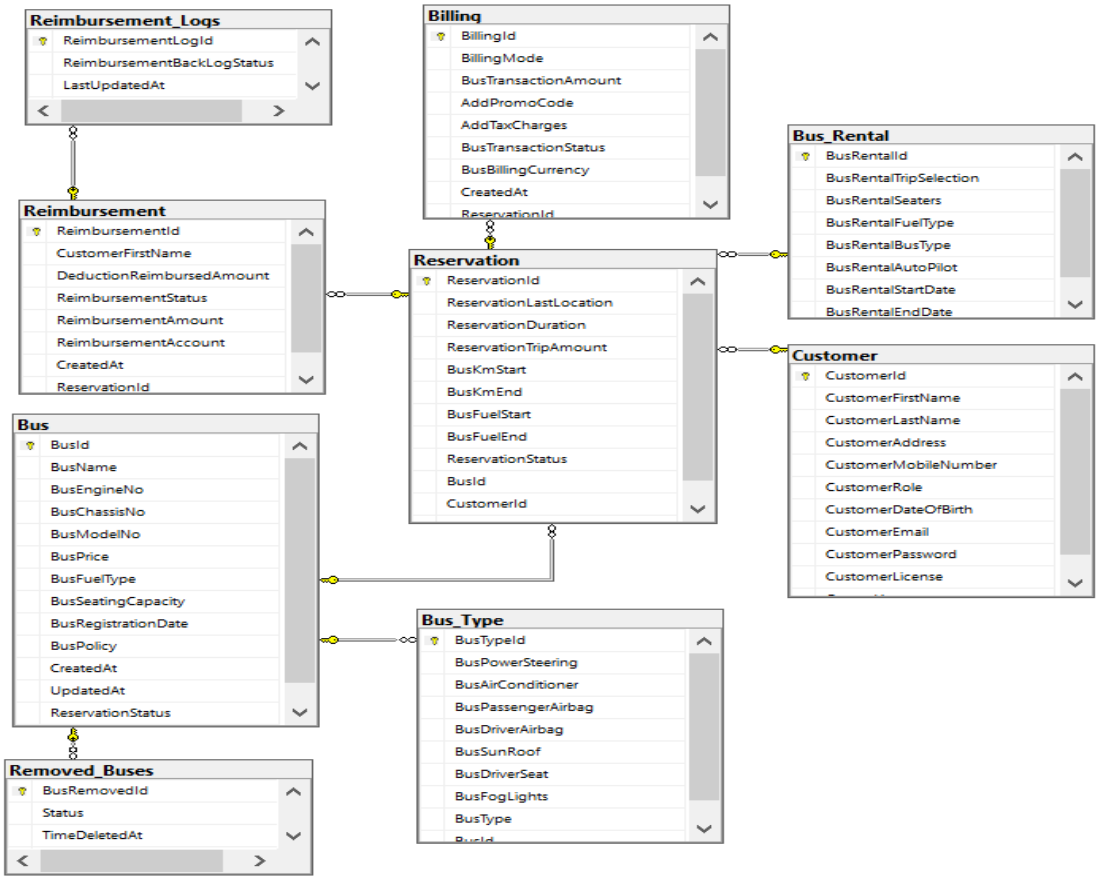


Figure 2 : Data Diagram

6. ENTITY RELATIONSHIP DIAGRAM

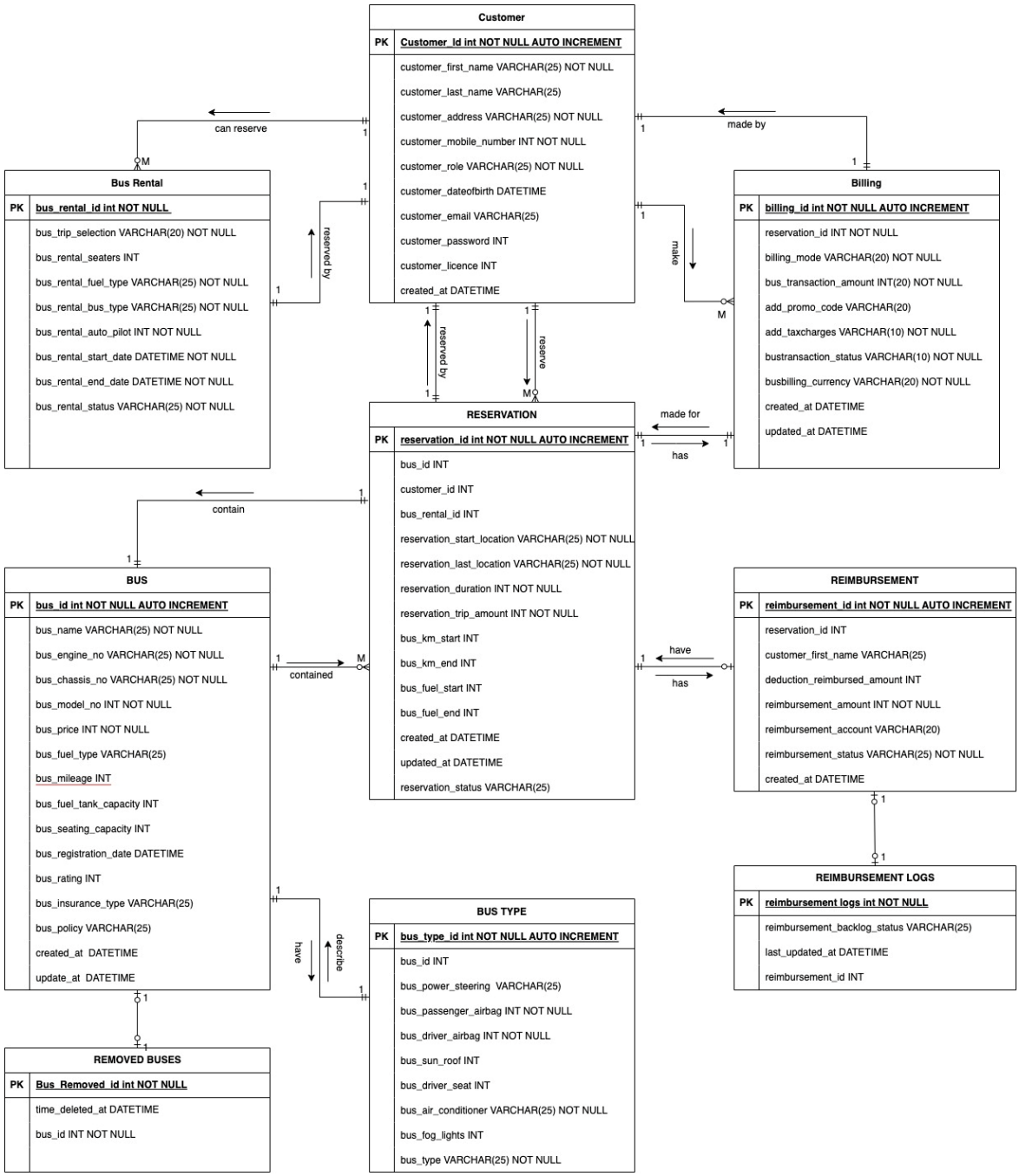


Figure 3 : Entity Relationship Diagram

7. IMPLEMENTATION IN SQL SERVER

In the system design we have added the following things where we the stored procedures and triggers, views are used to have different database functionalities.

7.1. STORED PROCEDURES

1) Design a process that allows users to look for buses that meet their criteria for ratings.

Here, the user's input is taken, and buses that meet or exceed that rating are displayed to the user:

Stored Procedure - 1

```
CREATE PROCEDURE [dbo].[SP_GetBillingByReservationId]
-- Add parameters sp
(@ReservationId INT)
AS
BEGIN

SET NOCOUNT ON;

-- Insert statements sp
SELECT Billing.Billing_Mode, Billing.Bus_Transaction_Amount,
Billing.Add_TaxCharges, Billing.Bus_Transaction_Status,
Reservation.Reservation_Last_Location, Reservation.Reservation_Duration,
Reservation.Reservation_status
from Billing
INNER JOIN Reservation
On Billing.Billing_Id = Reservation.Reservation_Id
where Reservation.Reservation_Id = 1
order by Reservation.Reservation_Duration
END
```

```
DECLARE @return_value int

EXEC    @return_value = [dbo].[SP_GetBillingByReservationId]
        @ReservationId = 1

SELECT  'Return Value' = @return_value

GO
```

.00 %

Results | Messages

	Billing_Mode	Bus_Transaction_Amount	Add_TaxCharges	Bus_Transaction_Status	Reservation_Last_Location	Reservation_Duration	Reservation_status
1	Per Usage	5	0.11	In	Dublin Airport	4	Confirm

	Return Value
1	0

Figure 4: Stored Procedure 1

Execute Procedure - [dbo].[SP_GetBillingByReservationId]

Select a page
General

Script | Help

Parameter	Data Type	Output Parameter	Pass Null V...	Value
@ReservationId	int	No	☐	1

Connection
Server:
DIGVIJAY
Connection:
root
View connection properties

Progress
Ready

OK | Cancel

Stored Procedure – 2

```
CREATE PROCEDURE [dbo].[SP_GetBusByBusName]
  -- Add parameters sp
  (@Bus_Name varchar(25))

  AS
  BEGIN

  SET NOCOUNT ON;

  -- Insert statements sp
  SELECT * from Bus
  where Bus_Name = @Bus_Name
  order by Bus_Id asc
  END
```

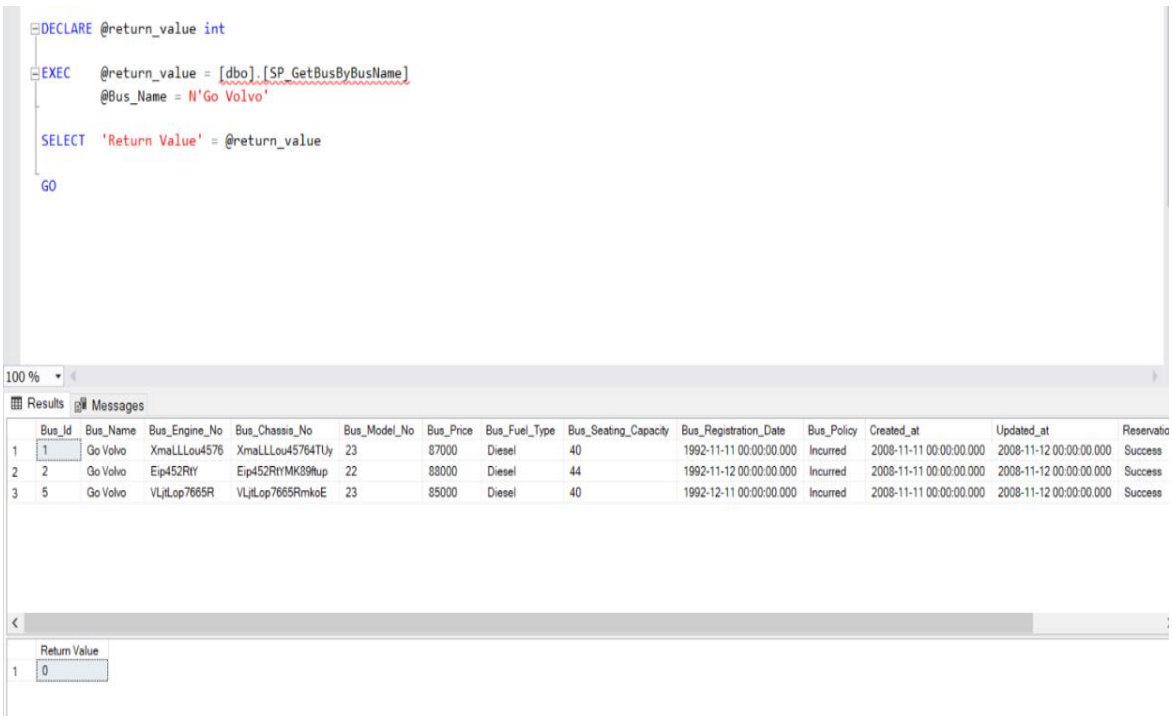
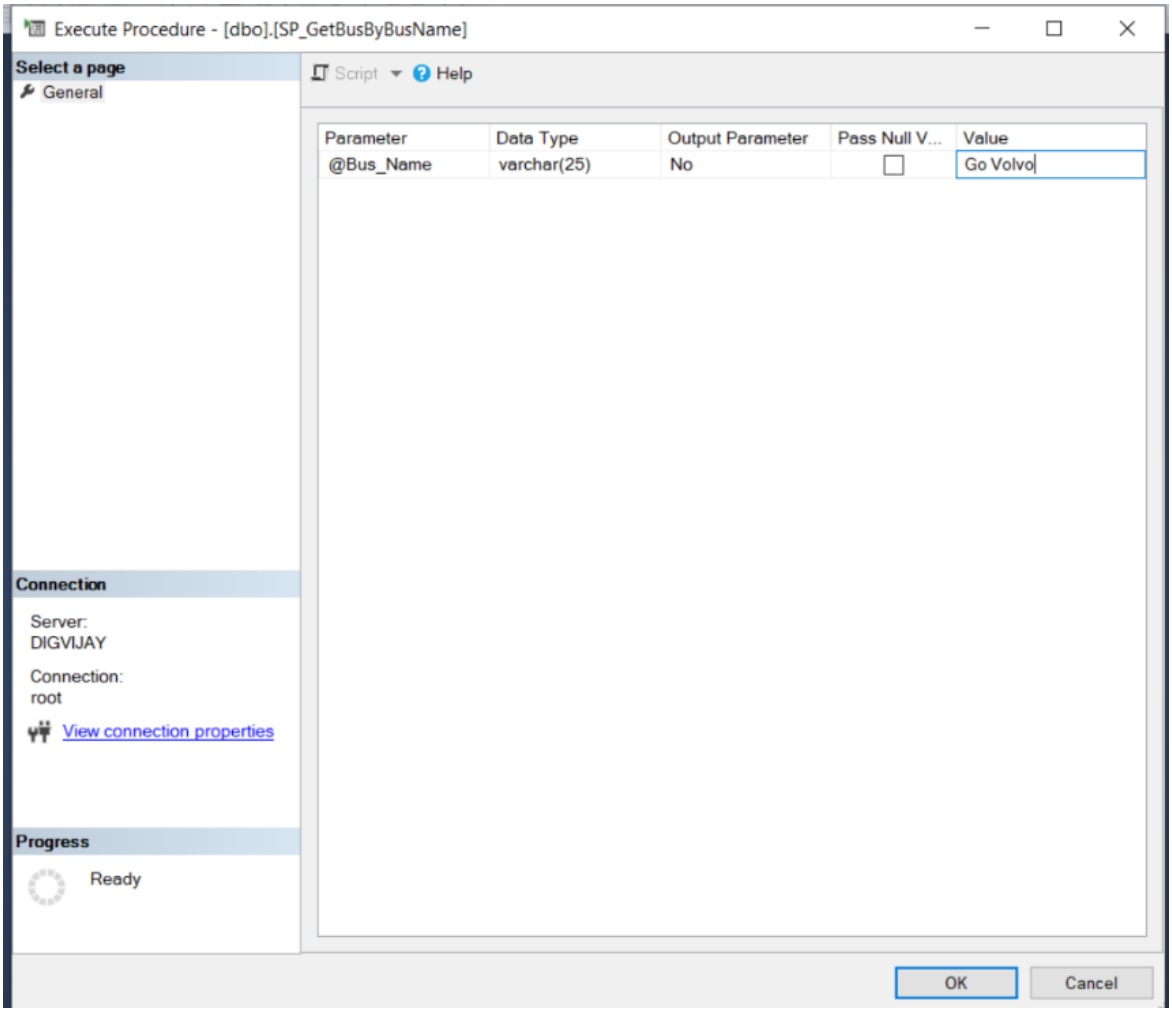


Figure 5: Stored Procedure 2



Stored Procedure – 3

```
CREATE PROCEDURE [dbo].[SP_GetBusInfoByLocation]
-- Add parameters sp
(@Reservation_Last_Location varchar(25))
AS
BEGIN

SET NOCOUNT ON;

-- Insert statements sp
```

```
SELECT Bus.Bus_Name, Bus.Bus_Engine_No, Bus.Bus_Chassis_No,
Reservation.Reservation_Last_Location, Reservation.Reservation_status from Bus
INNER JOIN Reservation
on Bus.Bus_Id = Reservation.Reservation_Id
where Reservation_Last_Location like '%' + TRIM(@Reservation_Last_Location) + '%'
order by Bus.Bus_Id
END
```

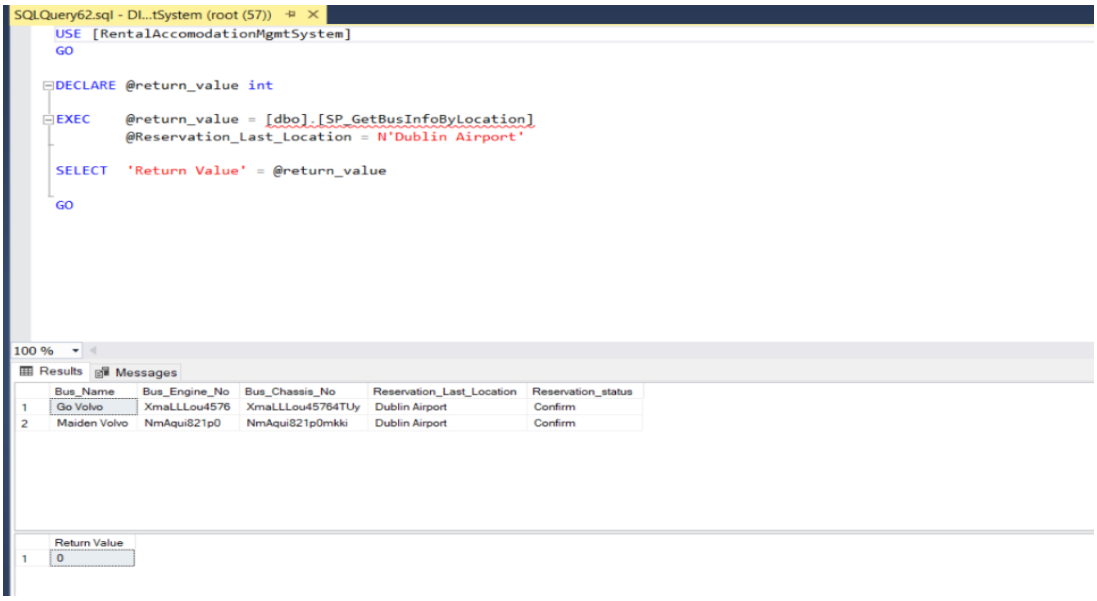
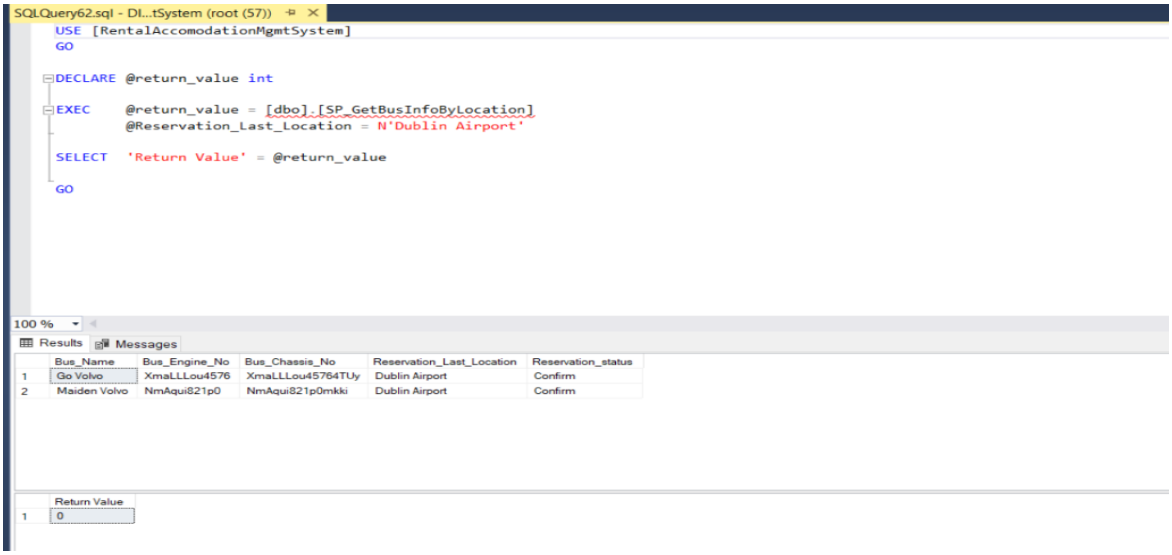


Figure 6 : Stored Procedure 3



Stored Procedure – 4

```
CREATE PROCEDURE [dbo].[SP_GetCustomerList]
-- Add parameters sp
(@Customer_Role varchar(25))
AS
BEGIN

SET NOCOUNT ON;

-- Insert statements sp
SELECT * from Customer
WHERE Customer_Role = @Customer_Role
ORDER BY Customer.Customer_First_Name;
END
```

```
DECLARE @return_value int

EXEC    @return_value = [dbo].[SP_GetCustomerList]
        @Customer_Role = N'Student'

SELECT  'Return Value' = @return_value

GO
```

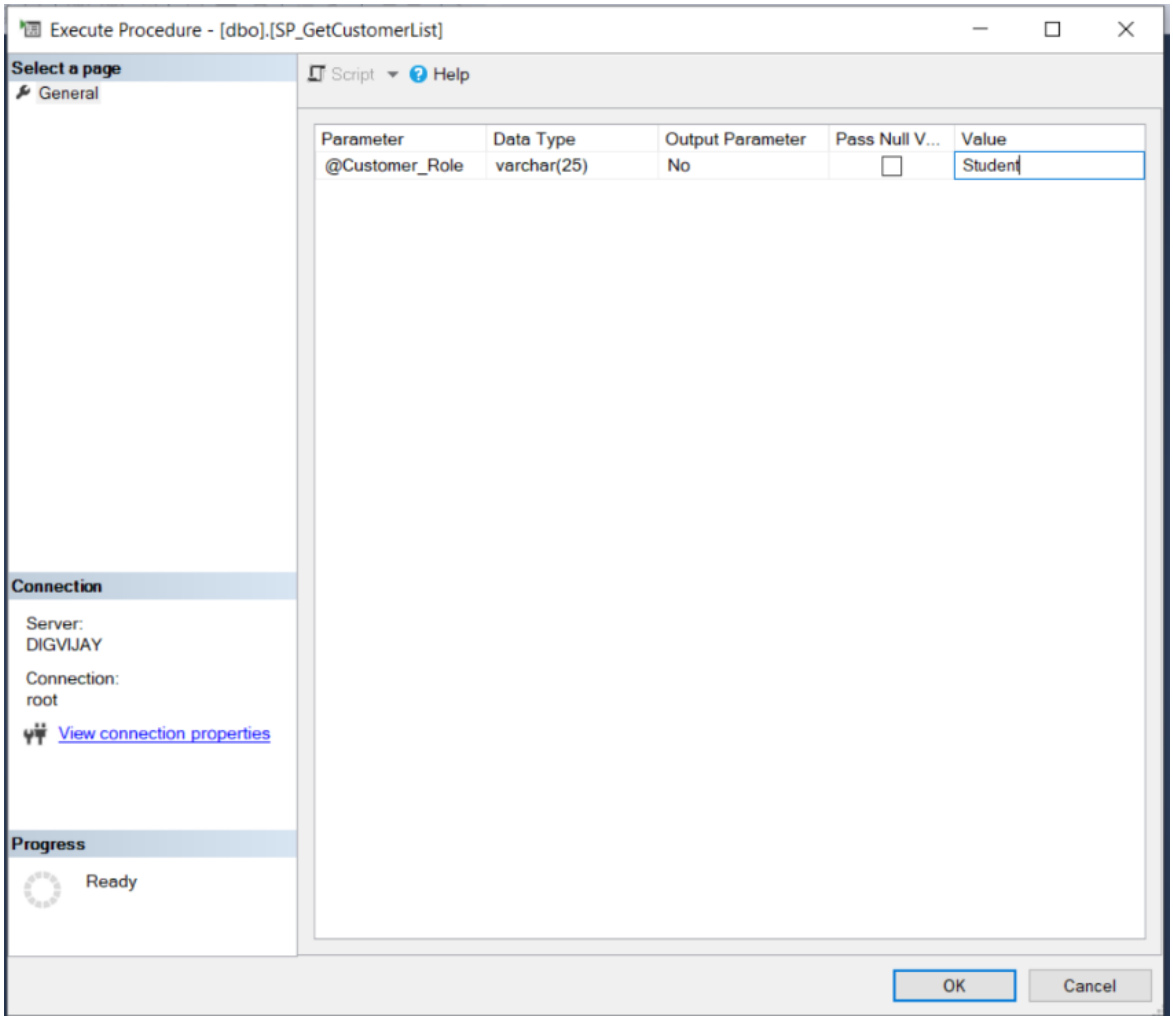
100 %

Results | Messages

	Customer_Id	Customer_First_Name	Customer_Last_Name	Customer_Address	Customer_Mobile_Number	Customer_Role	Customer_DateofBirth	Customer_Email	Customer_Password	Customer_License
1	5	Naresh	Shah	Ballsbridge, Dublin 04	+353 87 719 2344	Student	1990-04-24 00:00:00.000	maushah@gmail.com	Nshah@123	201212117899
2	1	Ramesh	Jadhav	Rathmines, Dublin 6	+353 89 097 7621	Student	1999-03-31 00:00:00.000	jadhavramesh@gmail.com	Nxxdih@123	201712117899
3	3	Swamima	Lele	D06 0242,Dublin 8	+353 87 221 4563	Student	1995-08-20 00:00:00.000	leleswamima@gmail.com	Polimn#%2	201812117899

	Return Value
1	0

Figure 7 : Stored Procedure 4



Stored Procedure – 5

```
CREATE PROCEDURE [dbo].[SP_GetReimbursementInfo]
-- Add parameters sp
(@Reimbursement_Status varchar(25))
AS
BEGIN

SET NOCOUNT ON;

-- Insert statements sp
select Reimbursement.Customer_First_Name, Reimbursement.Reimbursement_Amount,
```

```
Reimbursement_Logs.Reimbursement_BackLog_Status
From Reimbursement
INNER JOIN
Reimbursement_Logs on
Reimbursement_Logs.Reimbursement_Log_Id = Reimbursement.Reimbursement_Id
where Reimbursement_Status = @Reimbursement_Status;
END
```

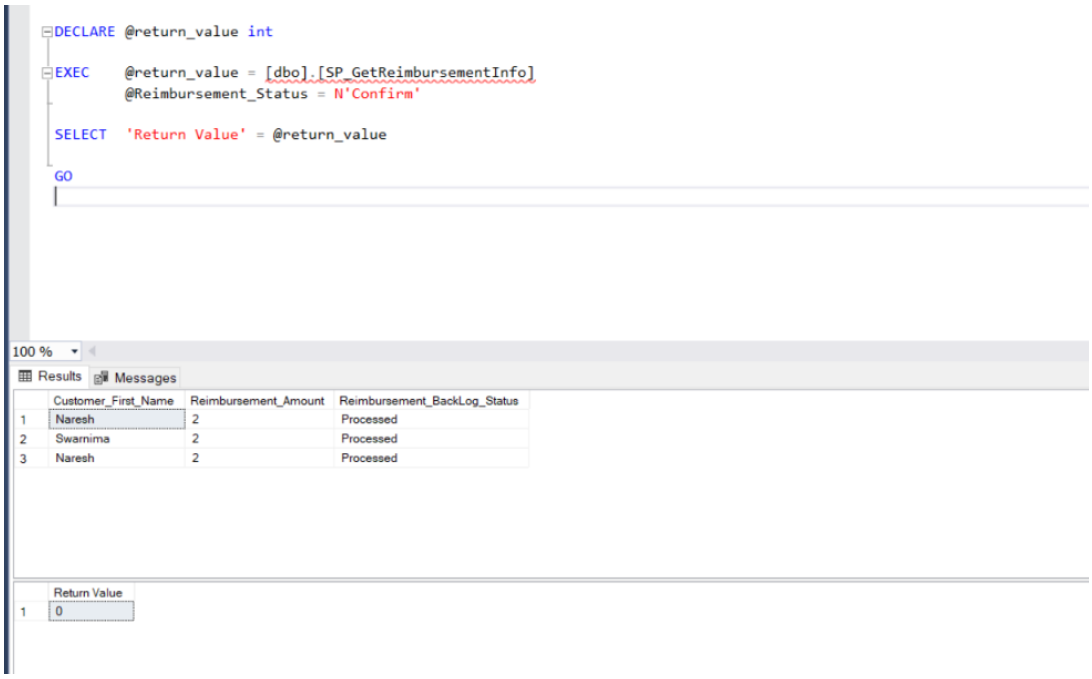
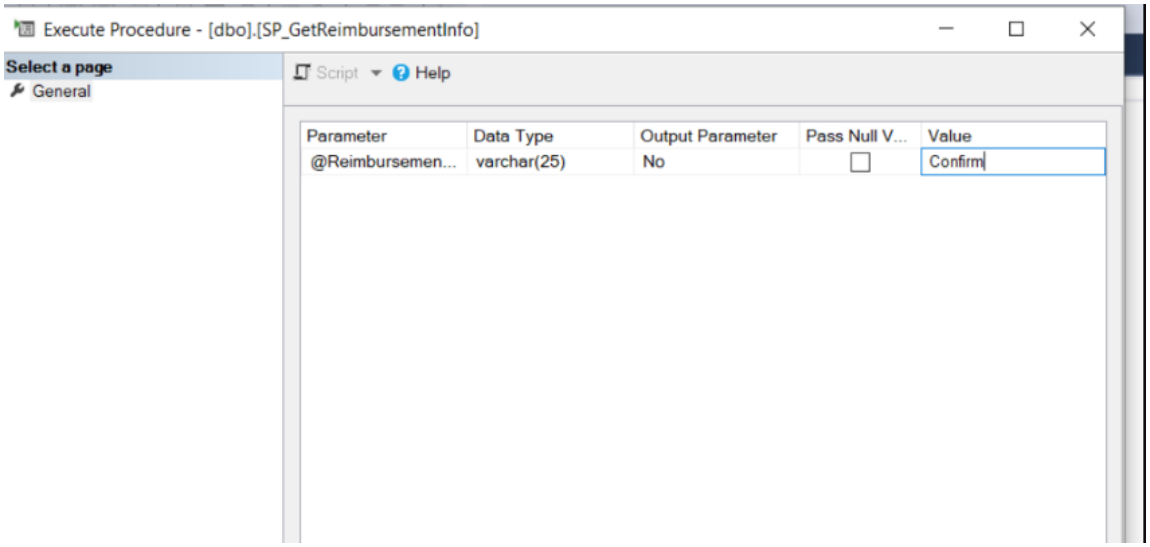


Figure 8: Stored Procedure 5



Stored Procedure – 6

```
CREATE PROCEDURE [dbo].[SP_GetRemovedBusesListByInsurance]
-- Add parameters sp
(@Bus_Insurance_Type varchar(25))
AS
BEGIN

SET NOCOUNT ON;

-- Insert statements sp
select Bus.Bus_Name, Bus.Bus_Engine_No, Bus.Bus_Chassis_No, Bus.Bus_Fuel_Type,
Removed_Buses.Bus_Removed_Id, Removed_Buses.Time_Deleted_At
From Bus
INNER JOIN
Removed_Buses on
Bus.Bus_Id = Removed_Buses.Bus_Removed_Id
where Bus_Insurance_Type like '%' + trim(@Bus_Insurance_Type) + '%';
END
```

```
DECLARE @return_value int

EXEC    @return_value = [dbo].[SP_GetRemovedBusesListByInsurance]
        @Bus_Insurance_Type = N'Silver'

SELECT  'Return Value' = @return_value

GO
```

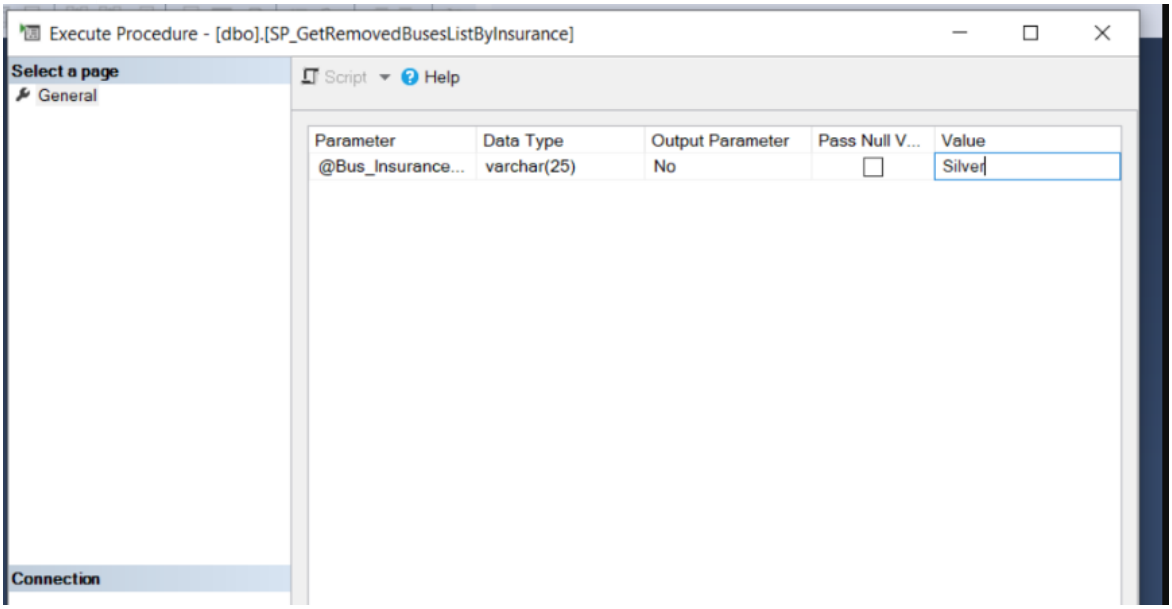
0 %

Results | Messages

Bus_Name	Bus_Engine_No	Bus_Chassis_No	Bus_Fuel_Type	Bus_Removed_Id	Time_Deleted_At
Go Volvo	XmaLLLou4576	XmaLLLou45764TUy	Diesel	1	2008-11-11 00:00:00.000
Go Volvo	Eip452RtY	Eip452RtYMK89ftup	Diesel	2	2008-10-31 00:00:00.000

Return Value
0

Figure 9: Stored Procedure 6



Stored Procedure – 7

```
CREATE PROCEDURE [dbo].[SP_GetReservationByReservationStatus]
-- Add parameters sp
(@Reservation_status varchar(20))
AS
BEGIN

SET NOCOUNT ON;

-- Insert statements sp
SELECT Customer.Customer_First_Name, Customer.Customer_Last_Name,
Customer.Customer_Email, Customer.Customer_Mobile_Number,
Reservation.Reservation_Duration, Reservation.Reservation_Last_Location,
Reservation.Reservation_Trip_Amount
from Customer
INNER JOIN
Reservation ON Customer.Customer_Id = Reservation.Reservation_Id
WHERE Reservation_status = @Reservation_status
/*'Confirm';*/
END
```

```
DECLARE @return_value int

EXEC    @return_value = [dbo].[SP_GetReservationByReservationStatus]
        @Reservation_status = N'Confirm'

SELECT  'Return Value' = @return_value

GO
```

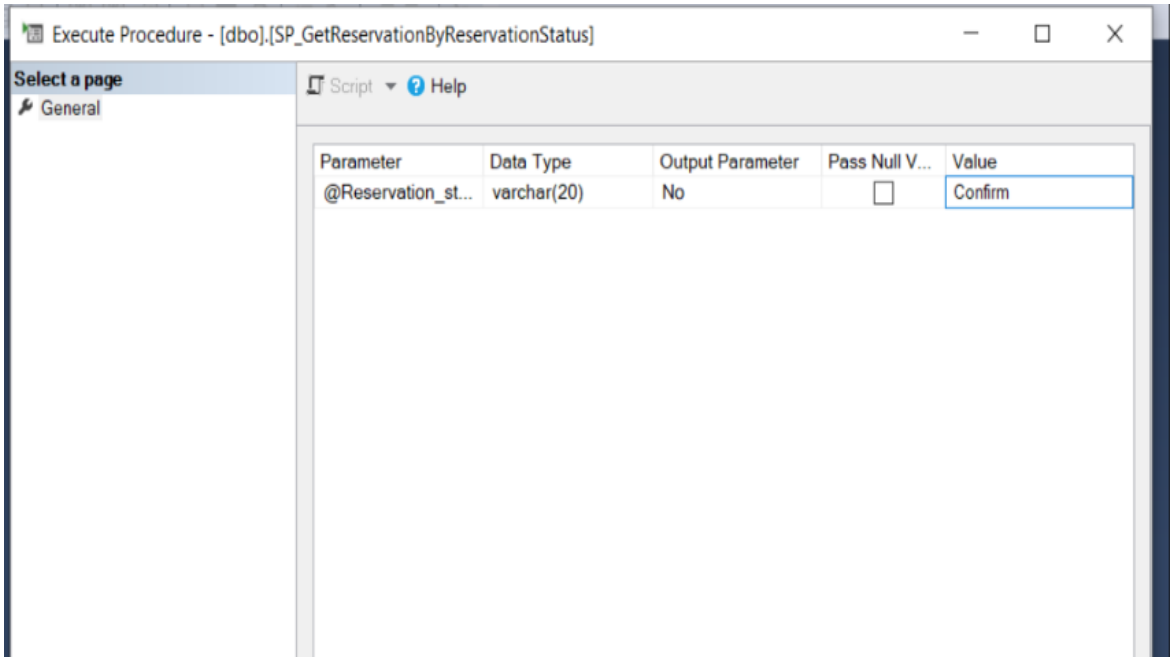
100 %

Results | Messages

	Customer_First_Name	Customer_Last_Name	Customer_Email	Customer_Mobile_Number	Reservation_Duration	Reservation_Last_Location	Reservation_Trip_Amount
1	Ramesh	Jadhav	jadhavramesh@gmail.com	+353 89 097 7621	4	Dublin Airport	14
2	Arif	Khan	khanarif@gmail.com	+353 89 921 2112	3	Dame Street	13
3	Swarnima	Lele	leleswarnima@gmail.com	+353 87 221 4563	2	Parnell Street	12
4	Joseph	Joshi	joshijosephjj@gmail.com	+353 89 054 7901	2	Dublin Airport	12
5	Naresh	Shah	maushah@gmail.com	+353 87 719 2344	2	Talbot Street	15

	Return Value
1	0

Figure 10: Stored Procedure 7



7.2. TRIGGERS

1) To store the information of the customer, when they try to register, login into the system and reserve any of the vehicle and the below table works, when the customer is inserted into the table, the trigger insertcustomer data into the insert customer trigger.

```
CREATE TRIGGER trInsertCustomer
ON Customer
FOR INSERT
AS
BEGIN
  Declare @Id int
  SELECT @Id = Customer_Id from inserted
  INSERT INTO Customer_Audit
  VALUES ('New Customer with Id = ' + CAST(@Id AS VARCHAR(10)) + ' is added at ' +
CAST(Getdate() AS VARCHAR(22)))
END
```

```
CREATE TRIGGER trInsertCustomer
ON Customer
FOR INSERT
AS
BEGIN
  Declare @Id int
  SELECT @Id = Customer_Id from inserted
  INSERT INTO Customer_Audit
  VALUES ('New customer with Id = ' + CAST(@Id AS VARCHAR(10)) + ' is added at ' + CAST(Getdate() AS VARCHAR(22)))
END
```

```
%
Messages
Commands completed successfully.

Completion time: 2022-12-28T12:38:29.1032439+00:00
```

Figure 11 : Trigger 1

```
SELECT * FROM [RentalAccomodationMgmtSystem].[dbo].[Customer_Audit]
```

%

Results | Messages

Id	Audit_Action
1	New customer with Id = 6 is added at Dec 28 2022 12:39PM

2) When a customer creates a reservation for a vehicle or a trip, insertreservation trigger is triggered and the data is inserted into the reservation_audit table.

```
CREATE TRIGGER trInsertReservation
ON Reservation
FOR INSERT
AS
BEGIN
  Declare @Id int
  SELECT @Id = Reservation_Id from inserted
  INSERT INTO Reservation_Audit
  VALUES ('New reservation with Id = ' + CAST(@Id AS VARCHAR(10)) + ' is added at ' +
CAST(Getdate() AS VARCHAR(22)))
END
```

```
CREATE TRIGGER trInsertReservation
ON Reservation
FOR INSERT
AS
BEGIN
  Declare @Id int
  SELECT @Id = Reservation_Id from inserted
  INSERT INTO Reservation_Audit
  VALUES ('New reservation with Id = ' + CAST(@Id AS VARCHAR(10)) + ' is added at ' + CAST(Getdate() AS VARCHAR(22)))
END
```

%

Messages

```
Commands completed successfully.

Completion time: 2022-12-28T12:52:56.1777510+00:00
```

Figure 12: Trigger 2

```
/****** Script for SelectTopNRows command from SSMS  ******/
SELECT TOP (1000) [Id]
      ,[Audit_Action]
  FROM [RentalAccomodationMgmtSystem].[dbo].[Reservation_Audit]
```

100 %

Results | Messages

	Id	Audit_Action
1	1	New reservation with Id = 7 is added at Dec 28 2022 1:01PM

7.3. VIEWS

1) Create View BusView AS

SELECT Bus_Id, Bus_Name, Bus_Engine_No, Bus_Chassis_No, Bus_Model_No, Bus_Price, Bus_Seating_Capacity, Bus_Fuel_Type, Bus_Insurance_Type FROM dbo.Bus

```
SELECT TOP (1000) [Bus_Id]
      ,[Bus_Name]
      ,[Bus_Engine_No]
      ,[Bus_Chassis_No]
      ,[Bus_Model_No]
      ,[Bus_Price]
      ,[Bus_Seating_Capacity]
      ,[Bus_Fuel_Type]
      ,[Bus_Insurance_Type]
  FROM [RentalAccomodationMgmtSystem].[dbo].[BusView]
```

%

Results | Messages

Bus_Id	Bus_Name	Bus_Engine_No	Bus_Chassis_No	Bus_Model_No	Bus_Price	Bus_Seating_Capacity	Bus_Fu
1	Go Volvo	XmaLLLou4576	XmaLLLou45764TUy	23	87000	40	Diesel
2	Go Volvo	Eip452RtY	Eip452RtYMK89ftup	22	88000	44	Diesel
3	Maiden Volvo	NmAqui821p0	NmAqui821p0mkki	22	88000	44	Diesel
4	Maiden Volvo	NmAqui821p0	NmAqui821p0mkki	22	88000	44	Diesel
5	Go Volvo	VLjtLop7665R	VLjtLop7665RmkoE	23	85000	40	Diesel

Figure 13: View 1

2) CREATE VIEW CustomerView AS

SELECT Customer_First_Name, Customer_Last_Name, Customer_Address, Customer_Mobile_Number, Customer_Email FROM dbo.Customer

```
/****** Script for SelectTopNRows command from SSMS  ******/
SELECT TOP (1000) [Customer_First_Name]
      ,[Customer_Last_Name]
      ,[Customer_Address]
      ,[Customer_Mobile_Number]
      ,[Customer_Email]
  FROM [RentalAccomodationMgmtSystem].[dbo].[CustomerView]
```

%

Results | Messages

Customer_First_Name	Customer_Last_Name	Customer_Address	Customer_Mobile_Number	Customer_Email
Ramesh	Jadhav	Rathmines, Dublin 6	+353 89 097 7621	jadhavramesh@gmail.com
Arif	Khan	Rathgar, Dublin 6	+353 89 921 2112	khanarif@gmail.com
Swarnima	Lele	D06 0242,Dublin 8	+353 87 221 4563	leleswarnima@gmail.com
Joseph	Joshi	D0734Y0, Dublin 7	+353 89 054 7901	joshijosephjj@gmail.com
Naresh	Shah	Ballsbridge, Dublin 04	+353 87 719 2344	maushah@gmail.com

Figure 14: View 2

8. Conclusion

In contrast to previous practice, where all activities related to the auto rental industry were restricted to a single physical place, the bus reservation industry has evolved with a new perk. The power of the internet has changed the nature of functions and how these tasks are accomplished, even while the physical place has not entirely disappeared.

Customers can now book buses online, rent automobiles online, and, after becoming a member, either have the bus delivered to their house. The web-based approach has provided a benefit to both customers and bus reservation Websites, allowing them to run their business and successfully and efficiently meet customer needs.

9. Bibiliography

Ms. Supriya Khaitan1,Shivalika Sisodia2, Shivam Jaiswal3, Abhinav Kabra4, 2021. “Online Reservation System” - Volume 7 (ISSN:1583-6258) – annalsofrscb

Oloyede, M.O.1 *, Alaya S.M.2 Adewole, K.S3, 2014. “Development of an Online Bus Ticket Reservation System for a Transportation Service” -Vol.5, No.12 – iiste

Mateusz Grzelak, Łukasz Napierała, Łukasz Napierała, Vincent Karovič & Iryna Ivanochko, 2019 – “Bus Ticket Reservation System Agile Methods of Projects Management” – INCoS

Nwakanma Ifeanyi Cosmas, Etus C, Ajere I.U. & Agomuo Uchechukwu Godswill ,2015. “Online Bus Ticket Reservation System” – Vol 1 – iiardonline

Ayman R. Mohammed, Sally S. Kassem, 2021. “UML Modeling of Online Public Bus Reservation System in Egypt” – IEEE

A. Ibrahim and A. Ta'a, "MOBILE - BASED BUS TICKETING SYSTEM IN IRAQ", *European Journal of Computer Science and Information Technology*, vol. 3, no. 5, pp. 42-55, Nov. 2015.

10. Appendix A: Create Table Queries

```
CREATE TABLE [dbo].[Billing](
    [Billing_Id] [int] IDENTITY(1,1) NOT NULL,
    [Billing_Mode] [varchar](20) NOT NULL,
    [Bus_Transaction_Amount] [int] NULL,
    [Add_Promo_Code] [varchar](20) NULL,
    [Add_TaxCharges] [varchar](20) NOT NULL,
    [Bus_Transaction_Status] [varchar](10) NOT NULL,
    [BusBilling_Currency] [varchar](20) NOT NULL,
    [Created_At] [timestamp] NOT NULL,
    [Reservation_Id] [int] NULL,
PRIMARY KEY CLUSTERED
(
    [Billing_Id] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON,
OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
GO

ALTER TABLE [dbo].[Billing]  WITH CHECK ADD FOREIGN KEY([Reservation_Id])
REFERENCES [dbo].[Reservation] ([Reservation_Id])
GO
```

```
CREATE TABLE [dbo].[Bus](
    [Bus_Id] [int] IDENTITY(1,1) NOT NULL,
    [Bus_Name] [varchar](25) NOT NULL,
    [Bus_Engine_No] [varchar](25) NOT NULL,
    [Bus_Chassis_No] [varchar](25) NOT NULL,
    [Bus_Model_No] [int] NOT NULL,
    [Bus_Price] [int] NOT NULL,
    [Bus_Fuel_Type] [varchar](25) NULL,
    [Bus_Seating_Capacity] [int] NULL,
    [Bus_Registration_Date] [datetime] NULL,
    [Bus_Policy] [varchar](25) NULL,
    [Created_at] [datetime] NULL,
    [Updated_at] [datetime] NULL,
    [Reservation_Status] [varchar](20) NULL,
    [Bus_Insurance_Type] [varchar](25) NULL,
PRIMARY KEY CLUSTERED
(
    [Bus_Id] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON,
ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
GO
```

```
CREATE TABLE [dbo].[Bus_Rental](
    [Bus_Rental_Id] [int] IDENTITY(1,1) NOT NULL,
    [Bus_Rental_Trip_Selection] [varchar](25) NULL,
    [Bus_Rental_Seaters] [varchar](25) NULL,
    [Bus_Rental_Fuel_Type] [varchar](25) NOT NULL,
    [Bus_Rental_Bus_Type] [int] NOT NULL,
    [Bus_Rental_Auto_Pilot] [int] NOT NULL,
    [Bus_Rental_Start_Date] [datetime] NULL,
    [Bus_Rental_End_Date] [datetime] NULL,
PRIMARY KEY CLUSTERED
(
    [Bus_Rental_Id] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON,
ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
GO
```

```
CREATE TABLE [dbo].[Bus_Type](
    [Bus_Type_Id] [int] IDENTITY(1,1) NOT NULL,
    [Bus_Power_Steering] [varchar](25) NULL,
    [Bus_Air_Conditioner] [varchar](25) NOT NULL,
    [Bus_Passenger_Airbag] [int] NOT NULL,
    [Bus_Driver_Airbag] [int] NOT NULL,
    [Bus_Sun_Roof] [int] NULL,
    [Bus_Driver_Seat] [int] NULL,
    [Bus_Fog_Lights] [int] NULL,
    [Bus_Type] [int] NULL,
    [Bus_Id] [int] NULL,
PRIMARY KEY CLUSTERED
(
    [Bus_Type_Id] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON,
ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
GO

ALTER TABLE [dbo].[Bus_Type]  WITH CHECK ADD FOREIGN KEY([Bus_Id])
REFERENCES [dbo].[Bus] ([Bus_Id])
GO

ALTER TABLE [dbo].[Bus_Type]  WITH CHECK ADD FOREIGN KEY([Bus_Id])
REFERENCES [dbo].[Bus] ([Bus_Id])
GO

CREATE TABLE [dbo].[Customer](
    [Customer_Id] [int] IDENTITY(1,1) NOT NULL,
    [Customer_First_Name] [varchar](25) NULL,
    [Customer_Last_Name] [varchar](25) NULL,
    [Customer_Address] [varchar](25) NOT NULL,
    [Customer_Mobile_Number] [varchar](50) NULL,
    [Customer_Role] [varchar](25) NOT NULL,
    [Customer_DateofBirth] [datetime] NULL,
    [Customer_Email] [varchar](25) NULL,
    [Customer_Password] [varchar](25) NULL,
    [Customer_License] [varchar](25) NULL,
    [Created_at] [datetime] NULL,
PRIMARY KEY CLUSTERED
(
    [Customer_Id] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON,
ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
GO
```

```sql
CREATE TABLE [dbo].[Reimbursement](
    [Reimbursement_Id] [int] IDENTITY(1,1) NOT NULL,
    [Customer_First_Name] [varchar](25) NULL,
    [Deduction_Reimbursed_Amount] [varchar](25) NOT NULL,
    [Reimbursement_Status] [varchar](25) NULL,
    [Reimbursement_Amount] [int] NOT NULL,
    [Reimbursement_Account] [int] NULL,
    [Created_At] [datetime] NULL,
    [Reservation_Id] [int] NULL,
PRIMARY KEY CLUSTERED
(
    [Reimbursement_Id] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
GO

ALTER TABLE [dbo].[Reimbursement]  WITH CHECK ADD FOREIGN KEY([Reservation_Id])
REFERENCES [dbo].[Reservation] ([Reservation_Id])
GO


CREATE TABLE [dbo].[Reimbursement_Logs](
    [Reimbursement_Log_Id] [int] IDENTITY(1,1) NOT NULL,
    [Reimbursement_BackLog_Status] [varchar](25) NULL,
    [Last_Updated_At] [datetime] NULL,
    [Reimbursement_Id] [int] NULL,
PRIMARY KEY CLUSTERED
(
    [Reimbursement_Log_Id] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON,
ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
GO

ALTER TABLE [dbo].[Reimbursement_Logs]  WITH CHECK ADD FOREIGN KEY([Reimbursement_Id])
REFERENCES [dbo].[Reimbursement] ([Reimbursement_Id])
GO
```

```
CREATE TABLE [dbo].[Removed_Buses](
    [Bus_Removed_Id] [int] IDENTITY(1,1) NOT NULL,
    [Status] [varchar](25) NULL,
    [Time_Deleted_At] [datetime] NULL,
    [Bus_Id] [int] NULL,
PRIMARY KEY CLUSTERED
(
    [Bus_Removed_Id] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON,
ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
GO

ALTER TABLE [dbo].[Removed_Buses]  WITH CHECK ADD FOREIGN KEY([Bus_Id])
REFERENCES [dbo].[Bus] ([Bus_Id])
GO
```

11. Appendix B: Insert Table Queries

```
INSERT INTO [dbo].[Billing]
 ([BillingMode]
 ,[BusTransactionAmount]
 ,[AddPromoCode]
 ,[AddTaxCharges]
 ,[BusTransactionStatus]
 ,[BusBillingCurrency]
B9IS100 Advanced Databases, Dublin Business School
40
 ,[ReservationId])
 VALUES
 ('Fixed',12,'FESTIVAL20',0.11,'In ','Euro',1) GO

INSERT INTO [dbo].[BusRental] ([BusRentalTripSelection]
    ,[BusRentalSeaters] ,[BusRentalFuelType] ,[BusRentalBusType]
    ,[BusRentalAutoPilot] ,[BusRentalStartDate] ,[BusRentalEndDate])
 VALUES ('Confirm',5,'Diesel',1,0,'2022-01-21','2022-01-27')
   GO

INSERT INTO [dbo].[Customer]
        ([CustomerFirstName]
        ,[CustomerLasName]
        ,[CustomerAddress]
        ,[CustomerMobileNumber]
        ,[CustomerRole]
        ,[CustomerDateofBirth]
        ,[CustomerEmail]
```

```
        ,[CustomerPassword]
        ,[CustomerLicense]
        ,[CreatedAt])
    VALUES
        ('Arif'
        ,'Khan'
        ,'Rathgar, Dublin 6'
        ,'+353 89 921 2112'
        ,'Student'
        ,'1994-07-31'
        ,'khanarif@gmail.com'
        ,'Nolva#21'
        ,'20150623117899'
        ,'2015-06-23')
GO

INSERT INTO [dbo].[ReimbursementLogs]
        ([ReimbursementBackLogStatus]
        ,[LastUpdatedAt]
        ,[ReimbursementId])
    VALUES
        ('Processed'
        ,'2022-12-25'
        ,5)
GO

INSERT INTO [dbo].[Reimbursement] ([CustomerFirstName]
    ,[DeductionReimbursedAmount] ,[ReimbursementStatus]
    ,[ReimbursementAmount] ,[ReimbursementAccount] ,[CreatedAt]
    ,[ReservationId])
VALUES ('Naresh','2','Confirm',2,2,'2022-11-12',1) GO

INSERT INTO [dbo].[Bus] ([BusName] ,[BusEngineNo] ,[BusChassisNo]
    ,[BusModelNo] ,[BusPrice] ,[BusFuelType] ,[BusSeatingCapacity]
    ,[BusRegistrationDate] ,[BusPolicy] ,[Createdat] ,[Updatedat]
    ,[ReservationStatus] B9IS100 Advanced Databases, Dublin Business
    School 41 ,[BusInsuranceType])
VALUES ('Go
    Volvo','XmaLLLou4576','XmaLLLou45764TUy',23,87000,'Diesel',40,'1992-
    11-11', 'Incurred','2008-11-11','2008-11-12','Success','Silver') GO
```

```
INSERT INTO [dbo].[BusType] ([BusPowerSteering] ,[BusAirConditioner]
    ,[BusPassengerAirbag] ,[BusDriverAirbag] ,[BusSunRoof]
    ,[BusDriverSeat] ,[BusFogLights] ,[BusType] ,[BusId])
VALUES ('HPS','Rolta',44,1,1,1,2,1,2) GO

INSERT INTO [dbo].[Reservation] ([ReservationLastLocation]
    ,[ReservationDuration] ,[ReservationTripAmount] ,[BusKmStart]
    ,[BusKmEnd] ,[BusFuelStart] ,[BusFuelEnd] ,[CreatedAt] ,[UpdatedAt]
    ,[Reservationstatus] ,[BusId] ,[CustomerId] ,[BusRentalId])
VALUES ('Dame Street',3,13,678,726,460,455,2022-10-08,2022-10-
    06,'Confirm',2,4,6) GO
```

12. Innovation

- Created stored procedures that make it easier for business owners to access the tables they regularly use.
- Primary key and foreign key restrictions are used to handle linked tables while removing and adding new data, in addition to processing standard data.
- Created views to simplify access to the tables that business owners utilize the most.
- We do not have deletion and updating since we implemented the data in third normal form, and have therefore developed stored methods for the anomaly.
- It is more safe and abstract since booking and payment information is stored in separate tables.