

CHAPTER 1

INTRODUCTION

1.1 Database Management System

A database management system (DBMS)[1] is system software for creating and managing databases. The DBMS provides users and programmers with a systematic way to create, retrieve, update and manage data.

A DBMS makes it possible for end users to create, read, update and delete data in a database. The DBMS essentially serves as an interface between the database and end users or application programs, ensuring that data is consistently organized and remains easily accessible.

DBMS include change management, performance monitoring/tuning and backup and recovery. Many database management systems are also responsible for automated rollbacks, restarts and recovery as well as the logging and auditing of activity.

The DBMS can offer both logical and physical data independence. That means it can protect users and applications from needing to know where data is stored or having to be concerned about changes to the physical structure of data (storage and hardware). As long as programs use the application programming interface (API) for the database that is provided by the DBMS, developers won't have to modify programs just because changes have been made to the database. With relational DBMSs (RDBMSs), this API is SQL, a standard programming[2].

1.2 Introduction to MySQL

MySQL is a leading open source database management system. It is a multi-user, multithreaded database management system. MySQL is especially popular on the web. It is one of the parts of the very popular LAMP platform. Linux, Apache, MySQL and PHP. MySQL database is available on most important OS platforms. It runs on BSD Unix, Linux, Windows or Mac. Wikipedia, YouTube, Facebook use MySQL. These sites manage millions of queries each day. MySQL comes in two versions: MySQL server system and MySQL embedded system. The MySQL server software and the client libraries are dual-licensed: GPL version 2 and proprietary license.

The development of MySQL began in 1994 by a Swedish company MySQL AB. Sun Microsystems acquired MySQL AB in 2008. Sun was bought by Oracle in 2010[3].

1.3 Introduction to Java

Java is a general-purpose computer-programming language that is concurrent, class –based, object-oriented and specifically designed to have as few implementation dependencies as possible. It is intended to let application developers “write once, read anywhere”, meaning that compiled Java code can run on all platforms that support java without the need for recompilation.

James Gosling, Mike Sheridan and Patrick Naughton initiated the Java language project in June 1991[4].

1.4 Necessity of Project

Nearly everyone goes on a vacation, for this “Travel agency management system” would play a vital role in planning a perfect trip. The main purpose is to help tourism companies to manage its customers and trips. This motivated us to develop an application to ease the administrative activities for travel agencies.

1.5 Application and Advantages

This software can be used in by any travel agency to store their employee details, vehicle details, and the details of the trips booked by the customers.

The main advantage of this application is that it provides an user interface for the administrator. For managing the travel agency and reduces the time of recording the data. This is the efficient way of storing the data in database

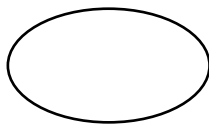
CHAPTER 2

DESIGN

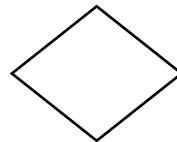
2.1 ER-diagram

An entity–relationship model describes interrelated things of interest in a specific domain of knowledge. The ER Diagram of Travel Agency Management System is shown in the figure:2.2.1.

Symbols in Entity Relationship:



Attribute



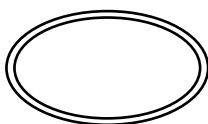
Relationship



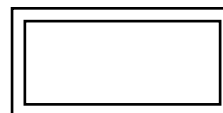
Entity



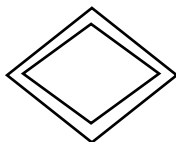
Derived attribute



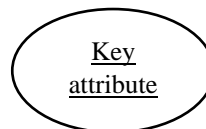
Multivalued
attribute



Weak entity



Weak
Relationship



Key attribute

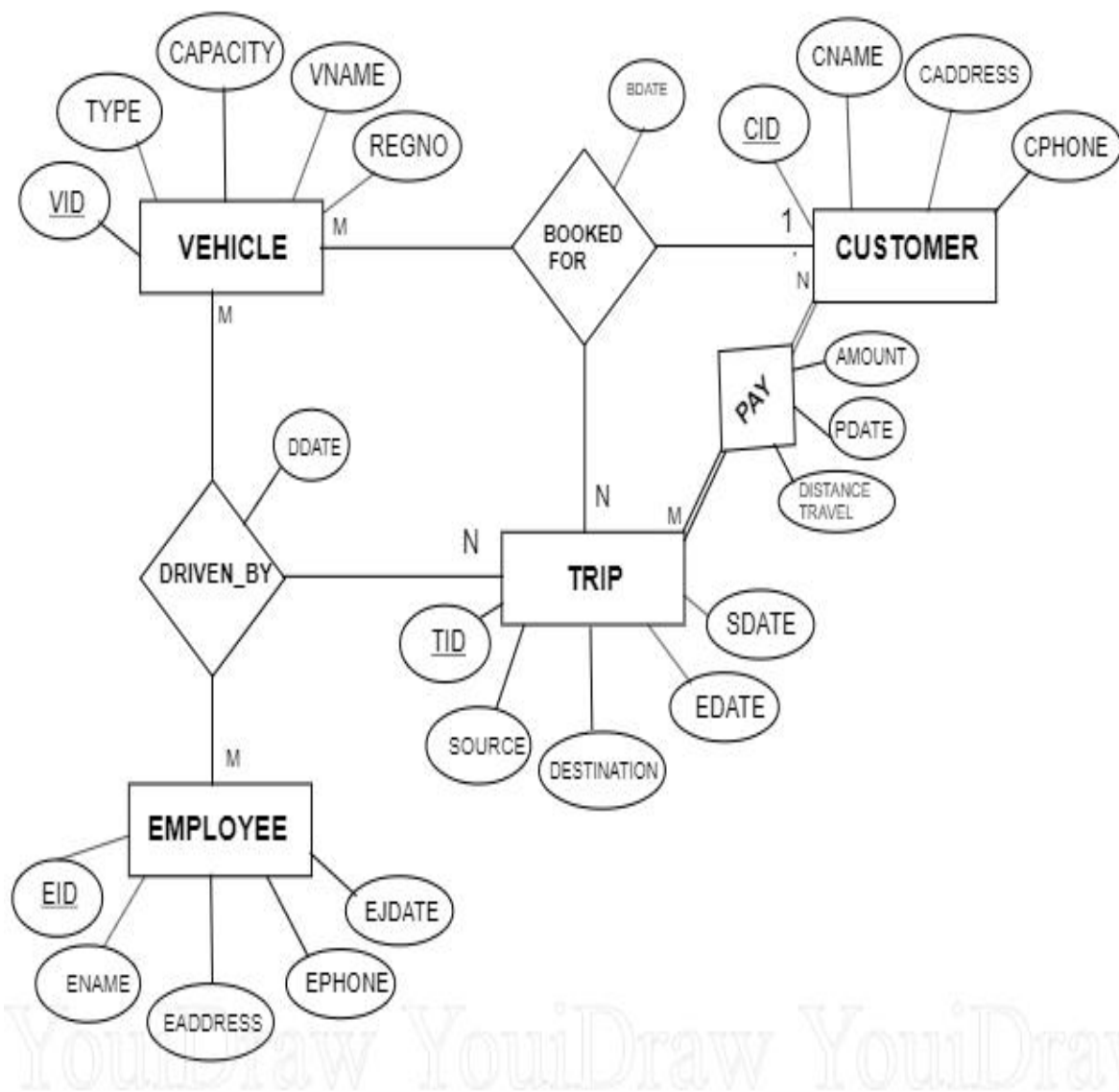


Figure:2.1.1 ER-Diagram of Travel agency Management System

2.2 Relational Schema

Relational Schema is the set of formulas(sentence) called integrity constraints imposed on database.

Step 1: Mapping of regular entity types

VEHICLE

<u>VID</u>	VNAME	CAPACITY	TYPE	REGNO
------------	-------	----------	------	-------

EMPLOYEE

<u>EID</u>	ENAME	EADDRESS	EPHONE	EJDATE
------------	-------	----------	--------	--------

CUSTOMER

<u>CID</u>	CNAME	CADDRESS	CPHONE
------------	-------	----------	--------

TRIP

<u>TID</u>	SOURCE	DESTINATION	SDATE	EDATE
------------	--------	-------------	-------	-------

Step 2: Mapping of weak entity types

The ERD of Travel Agency Management System does not contain any weak entity types

Step 3: Mapping of binary 1:1 relationship types

The ERD of Travel Agency Management System does not contain any binary 1:1 relationship types

Step 4: Mapping of binary 1: N relationship types

The ERD of Travel Agency Management System does not contain any binary 1: N relationship types

Step 5: Mapping of binary M:N relationship types**PAYMENT**

<u>CID</u>	<u>TID</u>	DISTANCE TRAVEL	AMOUNT	PDATE
------------	------------	-----------------	--------	-------

Step 6: Mapping of multivalued attributes

The ERD of Travel Agency Management System does not contain any multivalued attributes

Step 7: Mapping of N-array relationship types**BOOKED_FOR**

<u>CID</u>	<u>VID</u>	<u>TID</u>	BDATE
------------	------------	------------	-------

DRIVEN_BY

<u>EID</u>	<u>VID</u>	<u>TID</u>	DDATE
------------	------------	------------	-------

2.3 Schema diagram

The Schema diagram of Travel Agency Management System is shown in the figure:2.3.1.

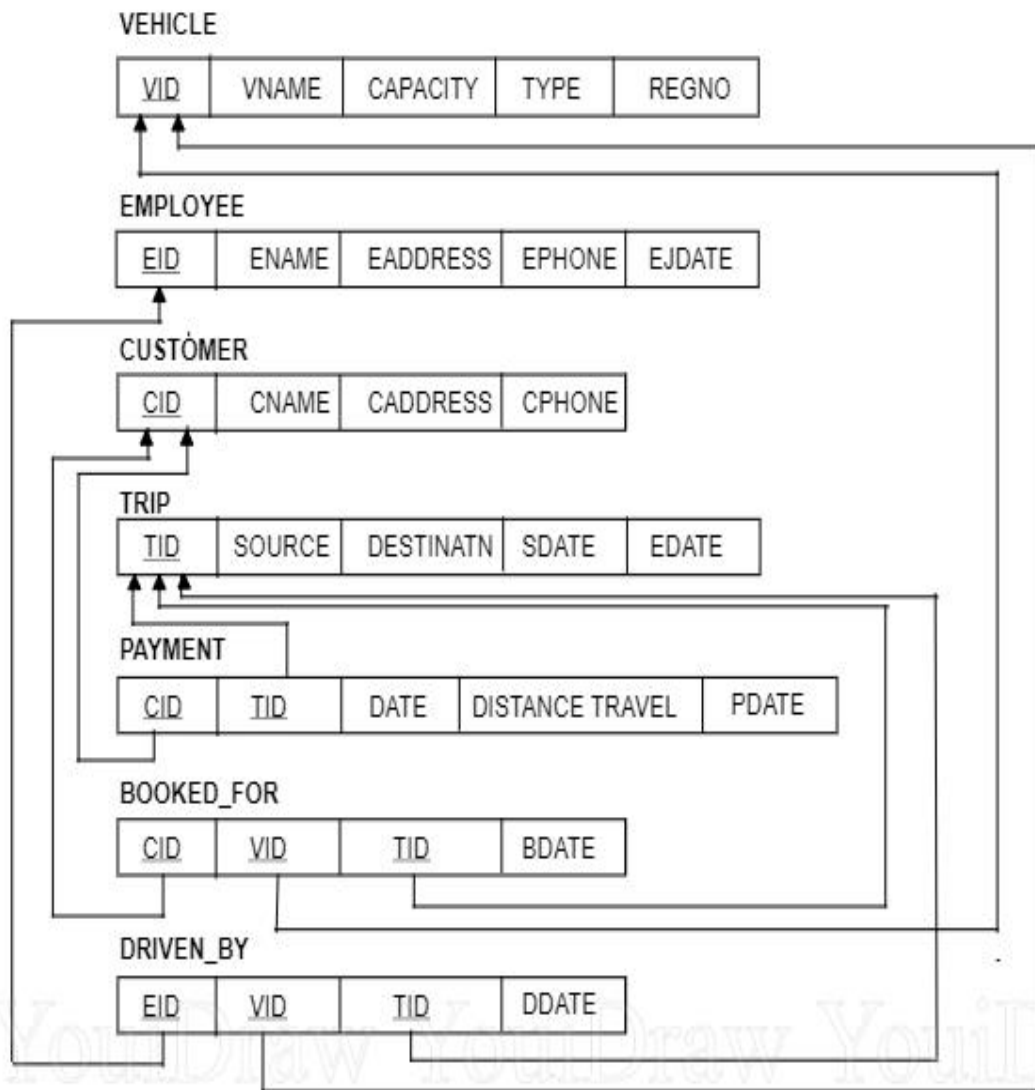


Figure 2.3.1 Schema Diagram of Travel Agency Management system

2.4 Normalization

Normalization is the process of decomposing unsatisfactory bad relations by breaking up their attributes into smaller relations.

First Normal Form (1NF)

The relations are said to be in 1NF if there are no multivalued attributes or nested relations.

Second Normal Form (2NF)

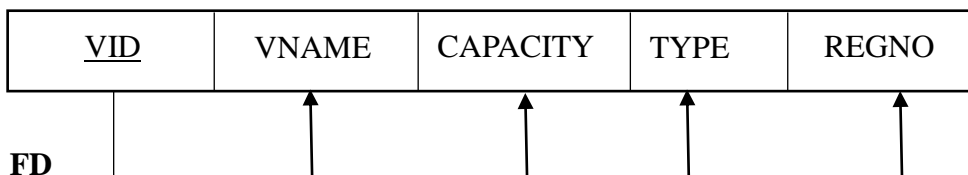
The relations are said to be in 2NF if it is in 1NF and no non-key attributes are functionally dependent on part of the primary key.

Third Normal Form (3NF)

The relations are said to be in 3NF if it is 2NF and no non-key attributes are functionally determined by another non-key attribute.

The FD's derived from the relation Vehicle are shown below

VEHICLE



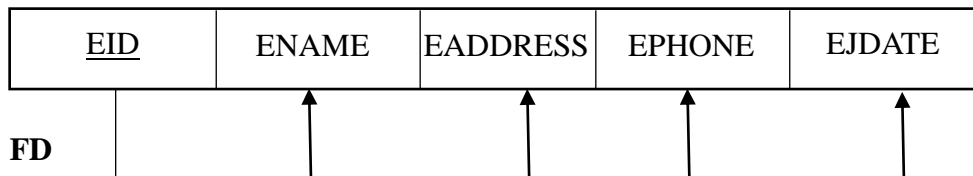
FD1: VID \rightarrow {VNAME, CAPACITY, TYPE, REGNO}

The vehicle relations are in first normal form since VID, VNAME, CAPACITY, TYPE, REGNO are atomic attributes.

The vehicle relations are in second normal form since VNAME, CAPACITY, TYPE, REGNO are functionally dependent on primary key VID.

The vehicle relation are in third normal form since there is no transitive dependency.

The FD's derived from the relation Employee are shown below.

EMPLOYEE

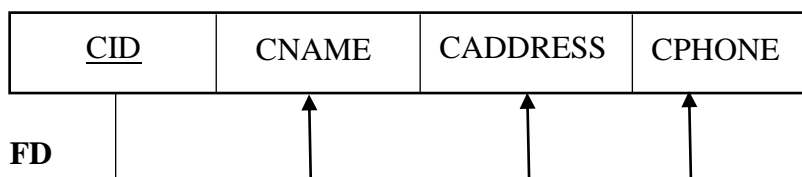
FD2: EID \rightarrow {ENAME, EADDRESS, EPHONE, EJDATE}

The employee relations are in first normal form since EID, ENAME, EADDRESS, EPHONE, EJDATE are atomic attributes.

The employee relations are in second normal form since ENAME, EADDRESS, EPHONE, EJDATE are functionally dependent on primary key EID.

The employee relation are in third normal form since there is no transitive dependency.

The FD's derived from the relation Customer are shown below.

CUSTOMER

FD3: CID \rightarrow {CNAME, CADDRESS, CPHONE}

The customer relations are in first normal form since CID, CNAME, CADDRESS, CPHONE are atomic attributes.

The customer relations are in second normal form since CNAME, CADDRESS, CPHONE are functionally dependent on primary key CID.

The customer relations are in third normal form since there is no transitive dependency.

The FD's derived from the relation Trip are shown below.

TRIP

<u>TID</u>	SOURCE	DESTINATION	SDATE	EDATE
FD				
	↑	↑	↑	↑

FD4: TID → {SOURCE, DESTINATION, SDATE, EDATE}

The trip relations are in first normal form since TID,SOURCE,DESTINATION,SDATE,EDATE are atomic attributes.

The trip relations are in second normal form since SOURCE,DESTINATION,SDATE,EDATE are functionally dependent on primary key TID.

The trip relations are in third normal form since there is no transitive dependency.

The FD's derived from the relation Booked_For are shown below.

BOOKED_FOR

<u>CID</u>	<u>VID</u>	<u>TID</u>	BDATE
FD			
			↑

FD5: {CID, VID, TID} → {BDATE}

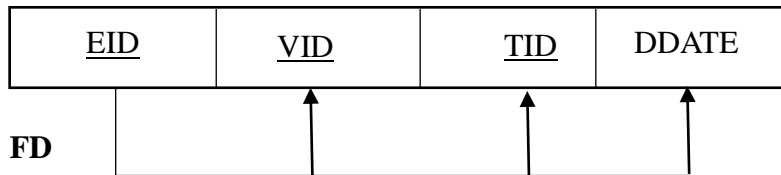
The booked_for relations are in first normal form since CID,VID,TID,BDATE are atomic attributes.

The booked_for relations are in second normal form since BDATE is functionally dependent on primary keys CID,VID,TID.

The booked_for are in third normal form since there is no transitive dependency.

The FD's derived from the relation Driven_By are shown below

DRIVEN_BY



FD6: {EID, VID, TID} → {DDATE}

.

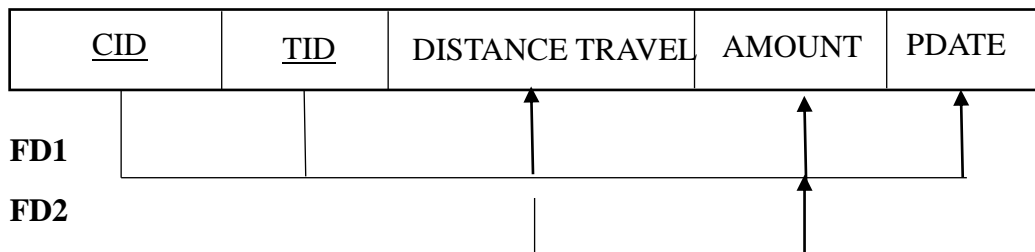
The Driven_by relations are in first normal form since EID,VID,TID,DDATE are atomic attributes.

The Driven_by relations are in second normal form since DDATE is functionally dependent on primary keys EID,VID,TID.

The Driven_by are in third normal form since there is no transitive dependency.

The FD's derived from the relation Payment are shown below.

PAYMENT



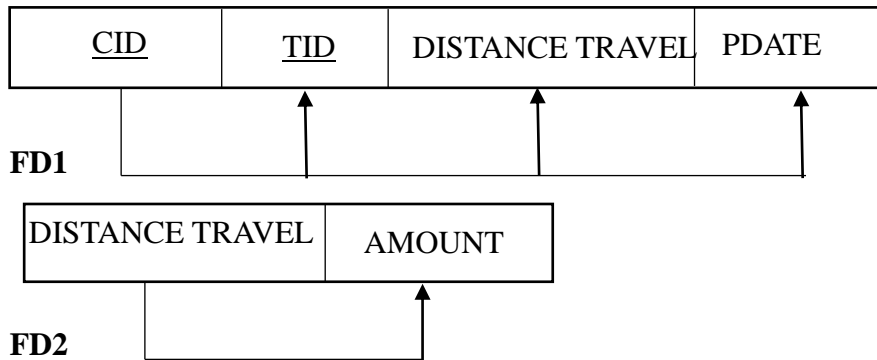
FD7: {CID, TID} → {DISTANCE TRAVEL, AMOUNT, PDATE}

FD8: DISTANCE TRAVEL → AMOUNT

The Payment relations are in first normal form since CID,TID,DISTANCE TRAVEL,AMOUNT ,PDATE are atomic attributes.

The Payment relations are in second normal form since DISTANCE TRAVEL,AMOUNT,PDATE is functionally dependent on primary keys CID,TID.

After converting the relation in 3NF.



The relations are in 3NF since no non-key attributes are functionally determined by another non-key attribute.

Therefore, the relations used in Travel Agency Management System are normalized and are in 1NF, 2NF and 3NF.

CHAPTER 3

IMPLEMENTATION

3.1 Software and Hardware Specification

3.1.1 Hardware Requirements

MySQL Workbench requires a current system to run smoothly. The minimum hardware requirements are:

CPU: Intel Core or Xeon 3GHz (or Dual Core 2GH) or equal AMD CPU

Cores: Single (Dual/Quad Core is recommended)

RAM: 4 GB (6 GB recommended)

Graphic Accelerators: NVidia or ATI with support of OpenGL 1.5 or higher

Display Resolution: 12801024 is recommended, 1024768 is minimum.

The minimum hardware requirements for NetBeans to run smoothly are:

OS: Microsoft Windows XP Professional SP3/Vista SPI/Windows 7 Professional/Windows10.

Processor: 800MHz Intel Pentium III or equivalent.

Memory: 512 MB.

Disk space: 650 MB of free disk space.

3.1.2 SOFTWARE REQUIREMENTS

Operating System: Windows 7/8/10.

Software's used: MySQL, NetBeans IDE 8.2, NetBeans IDE 11.1.

Libraries: Java JDK 8 (64 bit)

MySQL Connector Net 8.0.13.

MySQL Server 3.5.

MySQL Workbench 8.0 CE.

NetBeans IDE 8.2, NetBeans IDE 11.1.

3.2 Table Structure

3.2.1 Vehicle Relation

```
CREATE TABLE `DB`.`VEHICLE` (`VID` INT (3) NOT NULL, `VNAME` VARCHAR (20)
NULL, `CAPACITY` INT (3) NULL, `TYPE` VARCHAR(15) NULL, `REGNO` VARCHAR(15)
NULL, CONSTRAINT CPK_VID PRIMARY KEY (`VID`));
```

NAME	TYPE
VID	INT (3)
VNAME	VARCHAR (20)
CAPACITY	INT (3)
TYPE	VARCHAR (15)
REGNO	VARCHAR (15)

Figure 3.2.1 Vehicle Table Structure

3.2.2 Employee Relation

```
CREATE TABLE `DB`.`EMPLOYEE` (`EID` INT(3) NOT NULL, `ENAME` VARCHAR(20)
NULL, `EADDRESS` VARCHAR(25) NULL, `EPHONE` INT(10) NULL `EJDATE` DATE
NULL, CONSTRAINT CPK_EID PRIMARY KEY (`EID`));
```

NAME	TYPE
EID	INT(3)
ENAME	VARCHAR(20)
EADDRESS	VARCHAR(25)
EPHONE	INT(10)
EJDATE	DATE

Figure 3.2.2 Employee Table Structure

3.2.3 Trip relation

```
CREATE TABLE `DB`.`TRIP` (`TID` INT (3) NOT NULL, `SOURCE` VARCHAR (20) NULL,
`DEST` VARCHAR (25) NULL, `SDATE` VARCHAR (45) NULL, CONSTRAINT CPK_TID
PRIMARY KEY (`TID`));
```

NAME	TYPE
TID	INT (3)
SOURCE	VARCHAR (20)
DESTINATION	VARCHAR (25)
SDATE	DATE
EJDATE	DATE

Figure 3.2.3 Trip Table Structure

3.2.4 Customer Relation

```
CREATE TABLE `DB`.`CUSTOMER` (`CID` INT NOT NULL, `CNAME` VARCHAR (20)
NULL, `CADDRESS` VARCHAR (25) NULL, `CPHONE` INT(10) NULL PRIMARY KEY
(`CID`));
```

NAME	TYPE
CID	INT(3)
CNAME	VARCHAR(20)
CADDRESS	VARCHAR(25)
CPHONE	INT(10)

Figure 3.2.4 Customer Table Structure

3.2.5 Booked_for Relation

```
CREATE TABLE `DB`.`BOOKED_FOR` ( `CID` INT(3) NOT NULL, `VID` INT(3) NOT NULL,
`TID` INT(3) NOT NULL, `BDATE` DATE NULL, CONSTRAINT CPK_CVT PRIMARY KEY
(`CID`, `VID`, `TID`, CONSTRAINT `CFK-AID` FOREIGN KEY (`VID`) REFERENCES
`DB`.`VEHICLE` (`VID`) ON DELETE CASCADE, ON UPDATE CASCADE, CONSTRAINT
`CFK_EID` FOREIGN KEY (`CID`) REFERENCES `DB`.`CUSTOMER` (`CID` ON DELETE
CASCADE,ON UPDATE CASCADE, CONSTRAINT `CFK-TID` FOREIGN KEY (`TID`)
REFERENCES `DB`.`TRIP` (`TID`)ON DELETE CASCADE,ON UPDATE CASCADE);
```

NAME	TYPE
CID	INT (3)
VID	INT (3)
TID	INT (3)
BDATE	DATE

Figure 3.2.5 Booked_for Table Structure

3.2.6 Driven-by Relation

```
CREATE TABLE `DB`.`DRIVEN_BY` ( `EID` INT(3) NOT NULL, `VID` INT(3) NOT NULL,
`TID` INT(3) NOT NULL, `DDATE` DATE NULL, CONSTRAINT CPK_CVT1 PRIMARY KEY
(`EID`, `VID`, `TID`), CONSTRAINT `CFK-AID1` FOREIGN KEY (`VID`) REFERENCES
`DB`.`VEHICLE` (`VID`) ON DELETE CASCADE, ON UPDATE CASCADE, CONSTRAINT
`CFK_EID` FOREIGN KEY (`EID`) REFERENCES `DB`.`EMPLOYEE` (`CID`) ON DELETE
CASCADE,ON UPDATE CASCADE, CONSTRAINT `CFK-TID` FOREIGN KEY (`TID`)
REFERENCES `DB`.`TRIP` (`TID`ON DELETE CASCADE, ON UPDATE CASCADE)
```


NAME	TYPE
EID	INT(3)
VID	INT(3)
TID	INT(3)
DDATE	DATE

Figure 3.2.6 Driven_by Table Structure

3.2.6 Payment Relation

```
CREATE TABLE `DB`.`PAYMENT` (`TID` INT NOT NULL, `CID` INT NOT NULL, `PDATE`
DATE NULL, `DISTANCE TRAVEL` INT(4) NULL, AMOUNT` DECIMAL (6) NULL,
PRIMARY KEY (`TID`, `CID`) ,CONSTRAINT `CFK_CID1` FOREIGN KEY (`CID`)
REFERENCES `DB`.`CUSTOMER` (`CID`) ON DELETE NO ACTION ON UPDATE NO
ACTION,CONSTRAINT `CFK_TID1` FOREIGN KEY (`TID`) REFERENCES `DB`.`TRIP`
(`TID`) ON DELETE NO ACTION ON UPDATE NO ACTION);
```

NAME	TYPE
TID	INT(3)
CID	INT(3)
DISTANCE_TRAVEL	INT(6)
PDATE	DATE
AMOUNT	DECIMAL(6)

Figure 3.2.2 Payment Table Structure

3.3 Functionalities used in Travel Agency Management System

3.3.1. Connecting to Database

The “Travel Agency Management System” has been developed in JAVA. It used the Oracle database for storing the data and it is connected by the following syntax:

```
Class.forName("com.mysql.jdbc.Driver");
```

```
Connection conn =DriverManager.getConnection("jdbc: mysql://localhost/test1", "root", "11606");
```

```
return conn;
```

3.3.2 Insert

Insert operation is used to add vehicle details, employee details, customer details, trip details and payment details into the database.

```
String query = "INSERT INTO EMPLOYEE
```

```
(ENAME,EPHONE,EADDRESS,EJDATE)VALUES(?,?,?,?);
```

```
try{
```

```
    pst =conn.prepareStatement(query);
```

```
    pst.setString(1,Eid.getText());
```

```
    pst.setString(1,Ename.getText());
```

```
    pst.setString(2,Ephone.getText());
```

```
    pst.setString(3,Eaddress.getText());
```

```
    pst.setDate(4, new java.sql.Date(Ejdate.getDate().getTime()));
```

```
    pst.execute();
```

```
    JOptionPane.showMessageDialog(null,"Employee added");
```

```
    fetch();
```

```
}
```

```
catch(Exception e){
```

```
    JOptionPane.showMessageDialog(null,e);
```

```
}
```

3.3.3 Modify

Modify operation is used to update the vehicle details, employee details, customer details, trip details and payment details into the database.

```

conn = Mysqlconnect.ConnectDB();
int row = Etable.getSelectedRow();
String cell = Etable.getModel().getValueAt(row,0).toString();
System.out.println(cell);

String modsql = "UPDATE EMPLOYEE SET ENAME =?,EPHONE =?,EADDRESS=? WHERE
EID =" +cell;

try{
    pst=conn.prepareStatement(modsql);
    pst.setString(1,Ename.getText());
    pst.setString(2,Ephone.getText());
    pst.setString(3,Eaddress.getText());
    pst.setDate(4, new java.sql.Date(Ejdate.getDate().getTime()));
    pst.execute();
    JOptionPane.showMessageDialog(null,"Table Modified");
    fetch();
}
catch(Exception e){
    JOptionPane.showMessageDialog(null,e);
}

```

3.3.4 Delete

Delete operation is used to delete vehicles, employee, customer and trip details from the database.

```

conn = Mysqlconnect.ConnectDB();
int row = Etable.getSelectedRow();
String cell = Etable.getModel().getValueAt(row,0).toString();
System.out.println(cell);

String delsql = "DELETE FROM EMPLOYEE WHERE EID = " + cell;

```

```

try{
    pst=conn.prepareStatement(delsql);
    pst.execute();
    JOptionPane.showMessageDialog(null,"Employee Deleted");
    fetch();
}
catch(Exception e){
}
}

```

3.3.5 Trigger

Trigger operation is used raise an error message if the entered phone number in employee table is less than 10 digits.

```

CREATE TRIGGER PHVAL
BEFORE INSERT ON EMPLOYEE
FOR EACH ROW
BEGIN
IF LENGTH (NEW.ephone) > 10 OR LENGTH(NEW.ephone) < 10
THEN
signal sqlstate '45000'
set message_text = "Phone Number` is not valid";
END IF;
END;

```

3.3.6 Stored Procedure

A procedure is created in our project to display the contents on booking table. The procedure is called by callable statement.

```

CREATE PROCEDURE DISP1()
BEGIN
select t.tid,c.cid,t.source,t.dest,sdate,edate,cname,caddress,cphone from trip
t,customerc,booked_for bwhere t.tid = b.TID and c.cid = b.CID;
END;

```

CHAPTER 4

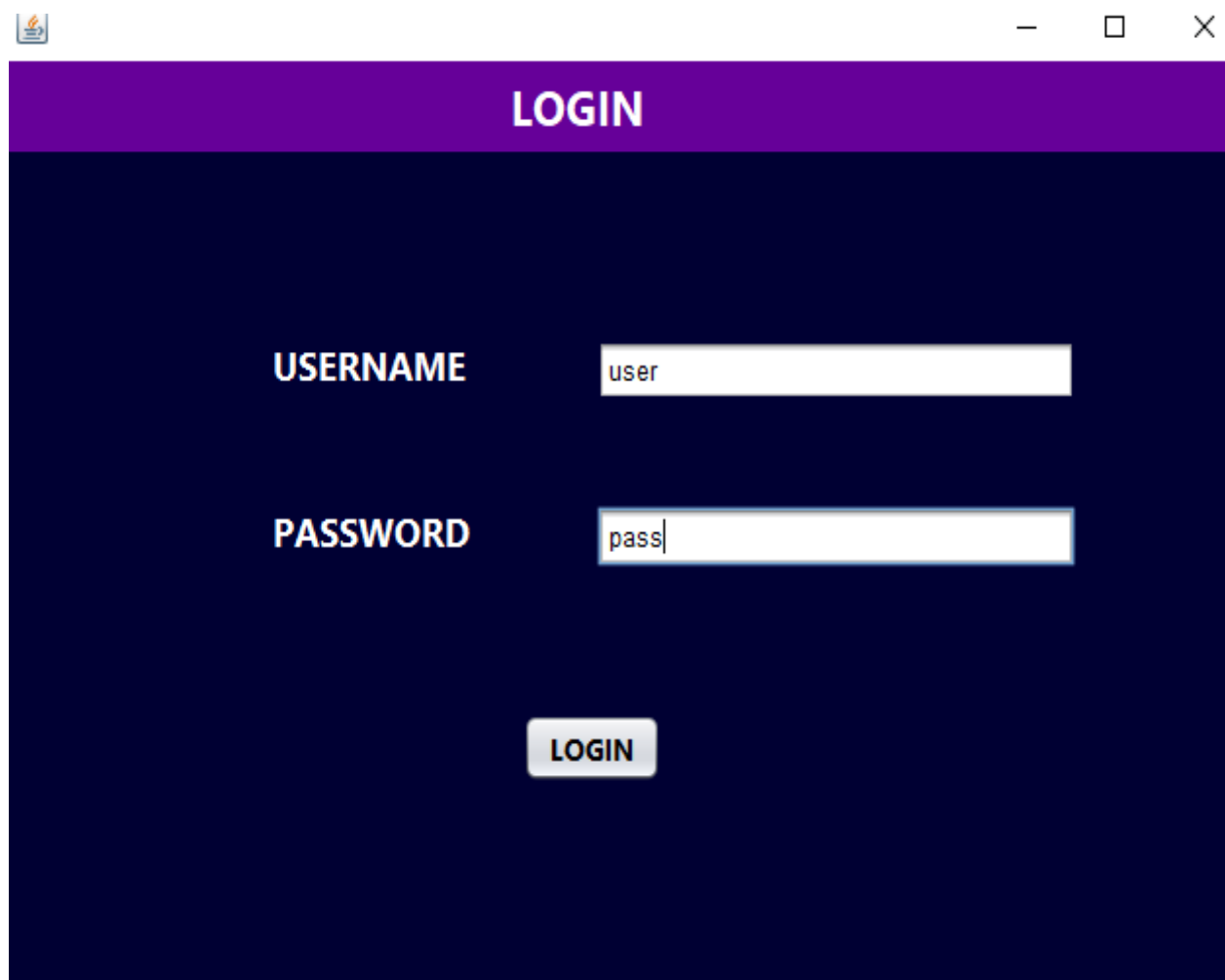
RESULTS

A complete Travel Agency Management database system, which allows the travel agencies to maintain the details of vehicle, employee, trip and the people booking a trip.

4.1 Snapshots

4.1.1 Login Page

Fig 4.1.1 shows user login system, where one can either login only if he already created an account.



The image shows a web browser window with a title bar containing a small icon and standard minimize, maximize, and close buttons. The main content area has a dark blue background. At the top, there is a purple header bar with the word "LOGIN" in white, bold, uppercase letters. Below this, the form consists of two labels, "USERNAME" and "PASSWORD", in white, bold, uppercase letters. To the right of each label is a white text input field. The "USERNAME" field contains the text "user", and the "PASSWORD" field contains the text "pass". Below the input fields is a white button with the word "LOGIN" in black, bold, uppercase letters.

Figure 4.1.1. Login Page

4.1.2 Home Page

Fig 4.1.2 shows Home page where the tables in the database can be selected to insert, delete, display, etc.

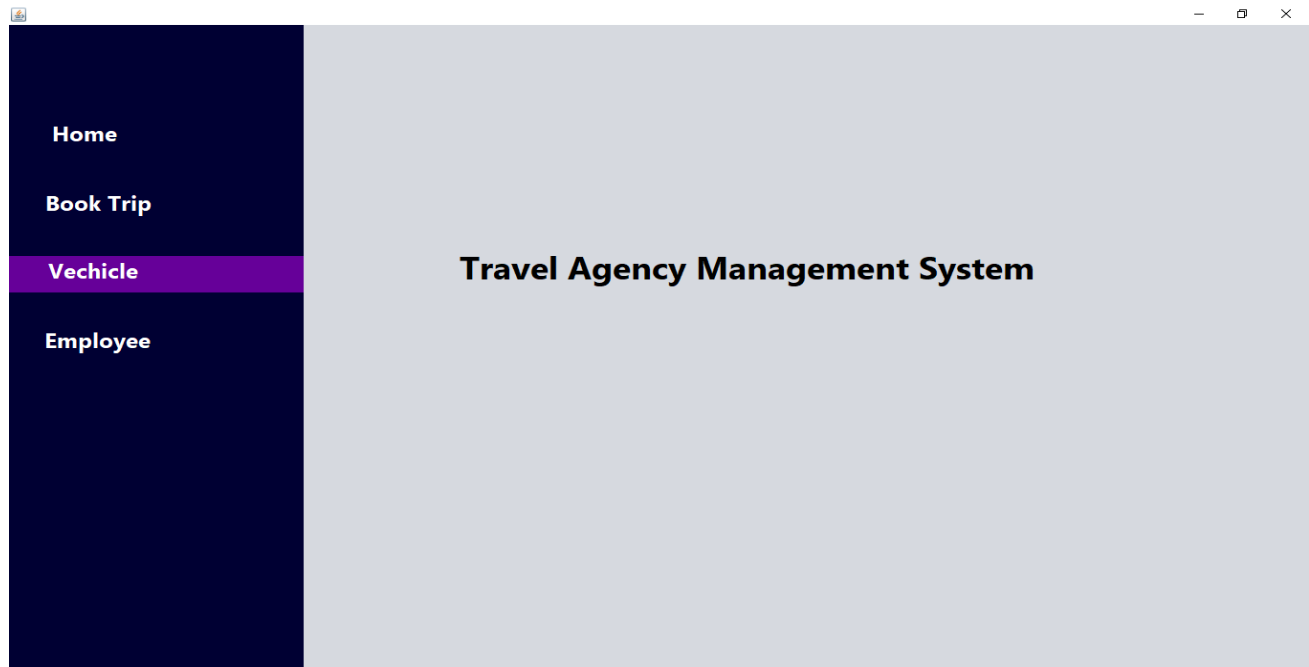


Figure 4.1.3. Home Page

4.1.3 Vehicle details

Fig 4.1.3 shows Vehicle page where we can insert, delete, modify and display the details of vehicles.

The screenshot shows a web application window titled "VEHICLE DASHBOARD". It features a form with four input fields and four buttons. The input fields are labeled "Vehicle Name", "Vehicle Type", "Capacity", and "Register Number". The buttons are labeled "ADD", "MODIFY", "DELETE", and "CLEAR". Below the form is a table with five columns: "vid", "vname", "capacity", "vtype", and "regno". The table contains three rows of data.

vid	vname	capacity	vtype	regno
36	SWIFT DEZIRE	5	CAR	KA 19 KP 2411
37	SWIFT DEZIRE	5	bus	KA 19 KP 2411
40	SWIFT	5	bus	KA 19 KP 2411

Figure 4.1.3. Vehicle Details

4.1.4 Employee details

Fig 4.1.4 shows Employee page where we can insert, delete, modify and display the details of employees.

EMPLOYEE DASHBOARD

Employee ID:

Employee Name:

Phone Number:

Address:

Joining Date:

Buttons: ADD, MODIFY, CLEAR, DELETE

eid	ename	ephone	eaddress	eJdate
84	joy	7894561234	bengaluru	2019-11-30
86	joy	7894561234	bengaluru	2019-11-30
87	joy	7894561234	bengaluru	2019-11-30
88	P raveen	7898456127	Manglore	1993-11-21

Figure 4.1.4. Employee Details

4.1.5 Booking details

Fig 4.1.5 shows Booking page where we can insert, delete, modify and display the details of customers of trips.

BOOKING DASHBOARD

Trip ID:

Customer ID:

Source:

Destination:

Journey Date:

End Date:

Customer Name:

Address:

Phone Number:

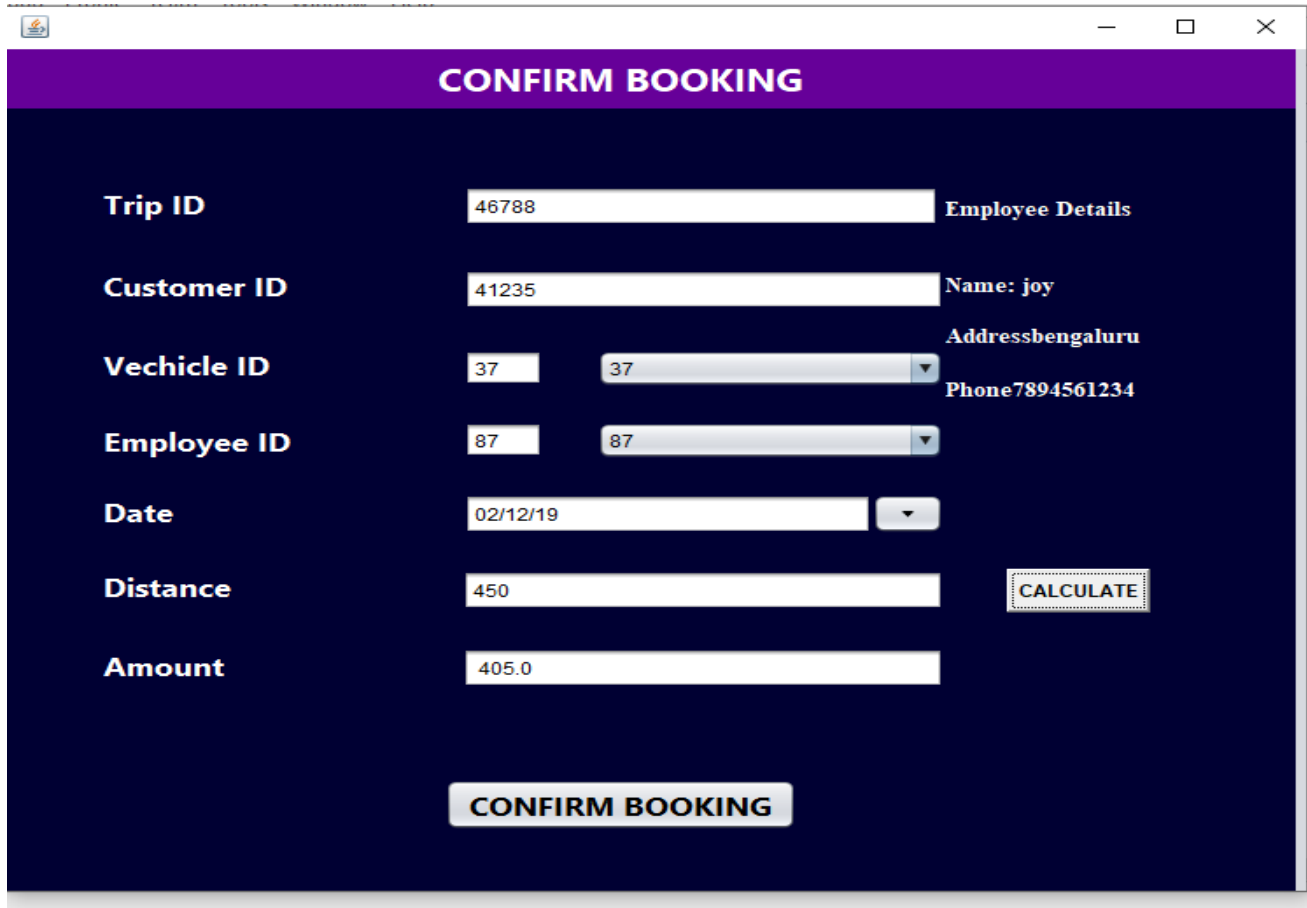
Buttons: BOOK, MODIFY, DELETE, CLEAR

tid	cid	source	dest	sdate	edate	cname	caddress	cphone
4647045	433844	MANGALURU	BANGLORE	2019-12-02	2019-12-04	RAMA	MANGALURU	6976941285

Fig 4.1.5 Booking Page

4.1.6. ConfirmBooking

Fig 4.1.6 shows Confirm booking page where we can insert the details of vehicle, employee and do payment.



CONFIRM BOOKING

Trip ID	<input type="text" value="46788"/>	Employee Details
Customer ID	<input type="text" value="41235"/>	Name: joy
Vehicle ID	<input type="text" value="37"/> <input type="text" value="37"/>	Address: bengaluru
Employee ID	<input type="text" value="87"/> <input type="text" value="87"/>	Phone: 7894561234
Date	<input type="text" value="02/12/19"/>	
Distance	<input type="text" value="450"/>	CALCULATE
Amount	<input type="text" value="405.0"/>	

CONFIRM BOOKING

Figure 4.1.6. Confirm Booking

CHAPTER 5

CONCLUSION

It has been a matter of immense pleasure, honour and challenge to have this opportunity to take up this project and complete it successfully. Our project maintains the entire database of a Travel agency. The software can be used in any Travel agency to maintain its vehicle details, employee details, trip details and customers' details. While developing this project we have learnt a lot about Travel Agency Management, we have also learnt how to make it user friendly (easy to use and handle) by hiding the complicated parts of it from users. During the development process we studied more about developing a software, how to implement the backend stored database in the real time system. We have tried to implement the project making it as user friendly and error free as possible. In future this project can be improved by recording the day-to-day activities taking place in a Agency.

REFERENCES

- [1] Ian Sommerville, "Software Engineering 8th edition". Pearson Education, 2008.
- [2] ElmasriNavathe, "Fundamental of Database System 3rd edition". Pearson Education, 2000.
- [3] We referred www.w3schools.in for MySQL.
- [4] We referred www.beginnersbook.com for Java.