Designing and Implementing a Relational DBMS on

"Car Insurance Management System"

Submitted to the

Savitribai Phule Pune University

In partial fulfillment for the award of the Degree of

Bachelor of Engineering
in

Information Technology
By

ANJALI GIRI

KALPESH KHAIRNAR

PRIYANKA TAKALE

RITU MAHAJAN

Under the guidance of

Prof. Bhawana Kanawade

Department Of Information Technology

International Institute of Information Technology, I²IT

Hinjwadi,Pune-41.

2020-2021

INDEX

ACKNOWLEDGEMENT

Before getting into the thick of the things, we would like to express our deep gratitude to the people who helped us during the course of this project. We are grateful to our project guide *Prof. Bhawana Kanawade* for her guidance throughout this project research and work

We also wish to thank all the faculty members of Information Technology and our respectable Head of Department for their constant help and efficient teaching procedures,

ABSTRACT

The Car Insurance Management System is a solution for organizations, which need to manage insurance for their cars, equipment, buildings, and other resources. Organizes and tracks insurance vendors and the policies provided under different coverage.

We are offering a robust web based insurance solution, which has the flexibility of customizations to match the specific needs of clients for achieving their business goal of good service and revenue generation.

Insurance policy administration system consists of a mathematical notation that captures the relationship between policies and objects and the entities that manage policies for those objects.

Hence there is need for an automated system, which can efficiently manage the company, records, provides instant access and one that improves the productivity. As a result of this automated system, the activities of the company are performed with in the stipulated time and the reliable and efficient service is ensured to its users.

The insurance company needs to keep track of details of its target companies, agents, policyholders, their premium payments and the various products that are available with it. Hence it is under tremendous pressure maintaining their day-to-day activities, which is currently being done manually.

Entire records have to be updated timely, even a slight mistake could complicate things. It is very difficult to handle bulk data since human memory is weaker than electronic counterpart. It is time consuming to summarize these details to produce the reports.

INTRODUCTION

SYSTEM DESCRIPTION:

The main objective of the Project on Car Insurance Management System is to manage the details of car, customer and accident. It manages all the information about Customer, Customer ID. Car Maker. The project is totally built at administrative end and thus only the administrator is guaranteed the access. The purpose of the project is to design a database system to reduce the manual work for managing the Customer, Insurance, Customer ID, Car Details, etc. It tracks all the details about the Car, Car name. Car Maker

The overall database contains 4 tables.

1. Car

2. Customer

3. Accident

4. Meet

1. Car - It contains the details of car such as car id, model name and company.

2. Customer - It contains the details of customer such as customer id, customer name, customer address, DOB, phone number and car id.

3. Accident - It contains the details of car accident such as report id, customer id, car id and date of accident.

4. Meet – This table represents the relationship, It contains the customer id, car id and report id.

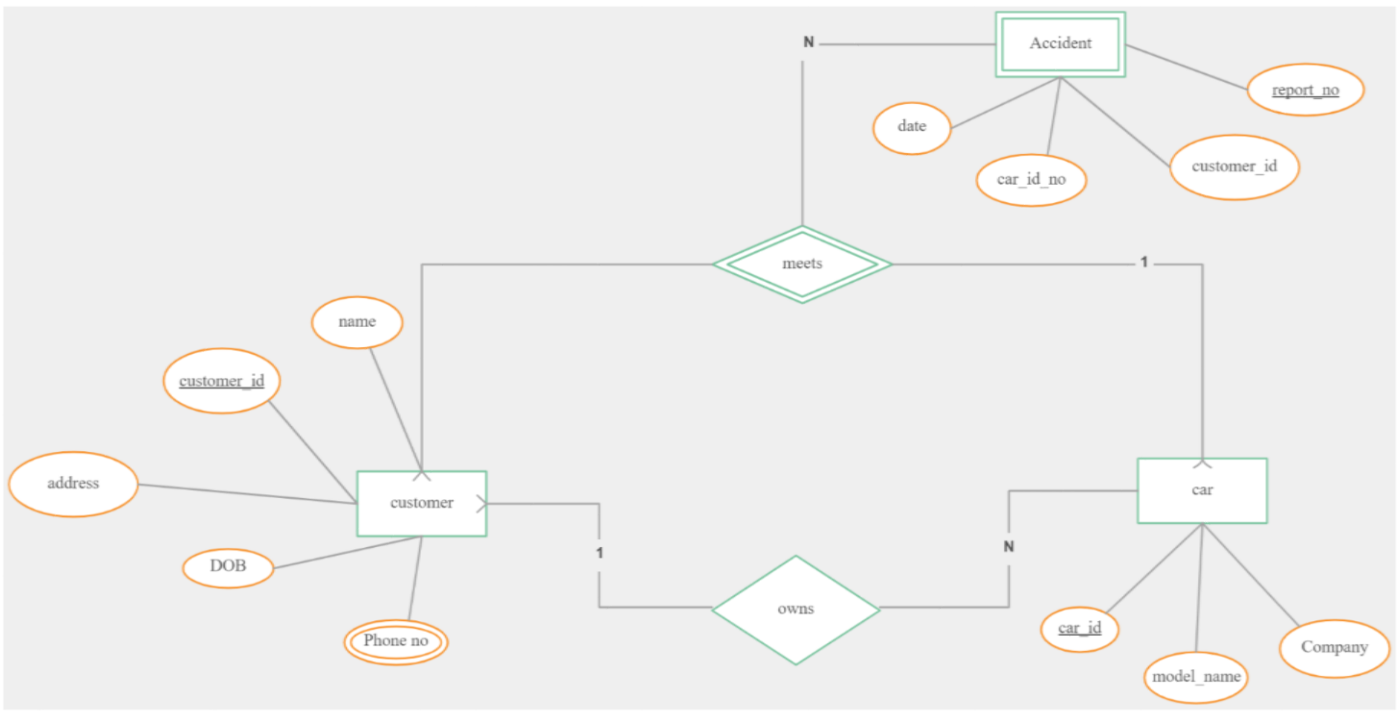
ER DIAGRAM

What is an Entity Relationship Diagram (ERD)?

An entity relationship diagram (ERD) shows the relationships of entity sets stored in a database. An entity in this context is an object, a component of data. An entity set is a collection of similar entities. These entities can have attributes that define its properties.

By defining the entities, their attributes, and showing the relationships between them, an ER diagram illustrates the logical structure of databases.

ER diagrams are used to sketch out the design of a database.

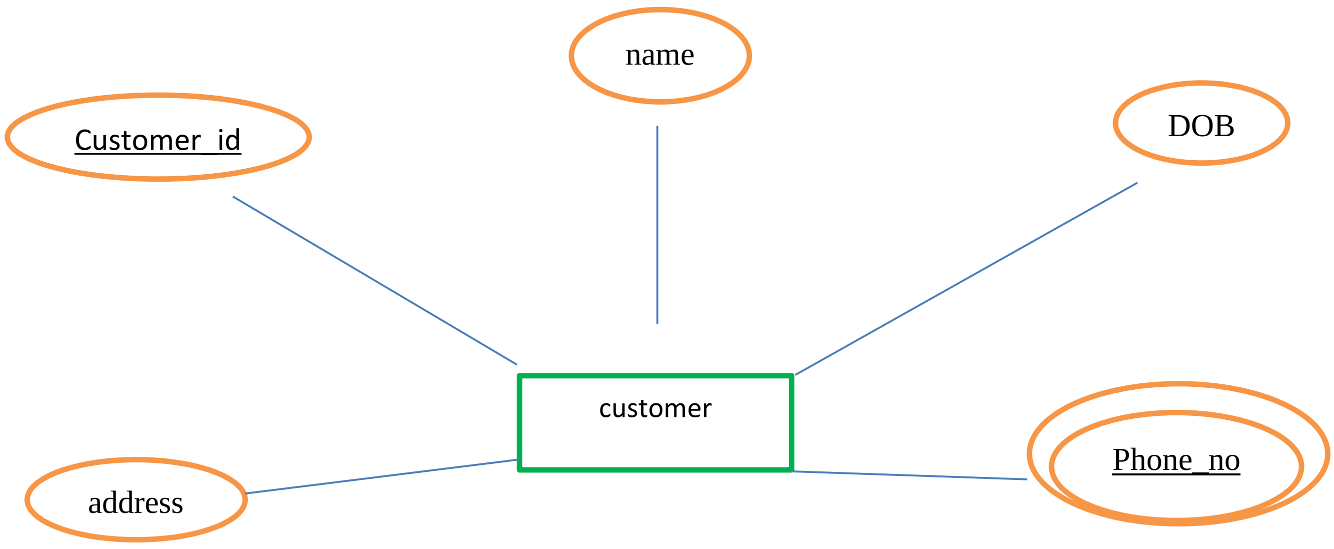


ER Diagram For Car Insurance Management System

CONVERTING ER DIAGRAM INTO TABLES

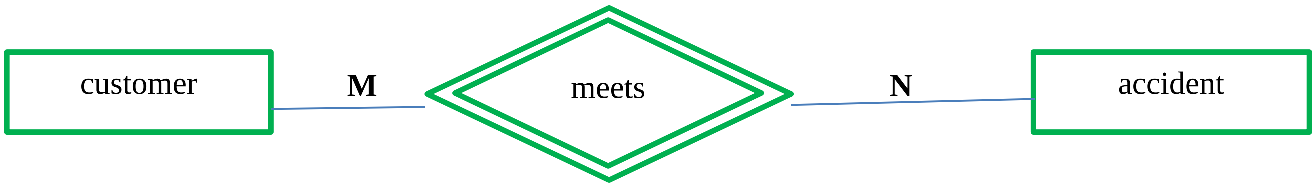
1. Converting strong entity types

- Each entity type becomes a table
- Each single valued attribute becomes a column
- Derived attributes are ignored
- Composite attributes are represented by components
- Multi-valued attributes are represented by a separate table
- Key attributes of the entity type is the Primary Key



2. Converting relationships

- Relationships are based on cardinalities and degree of the relation
    - **M : N**



- **1 : N**

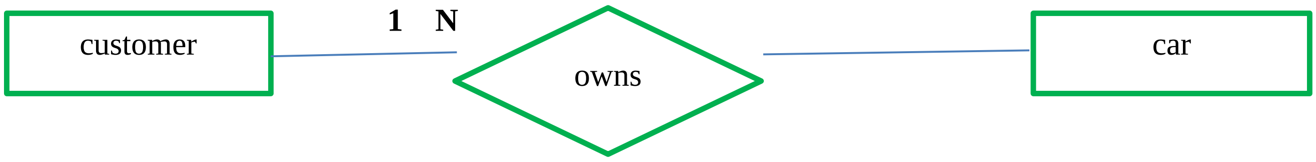
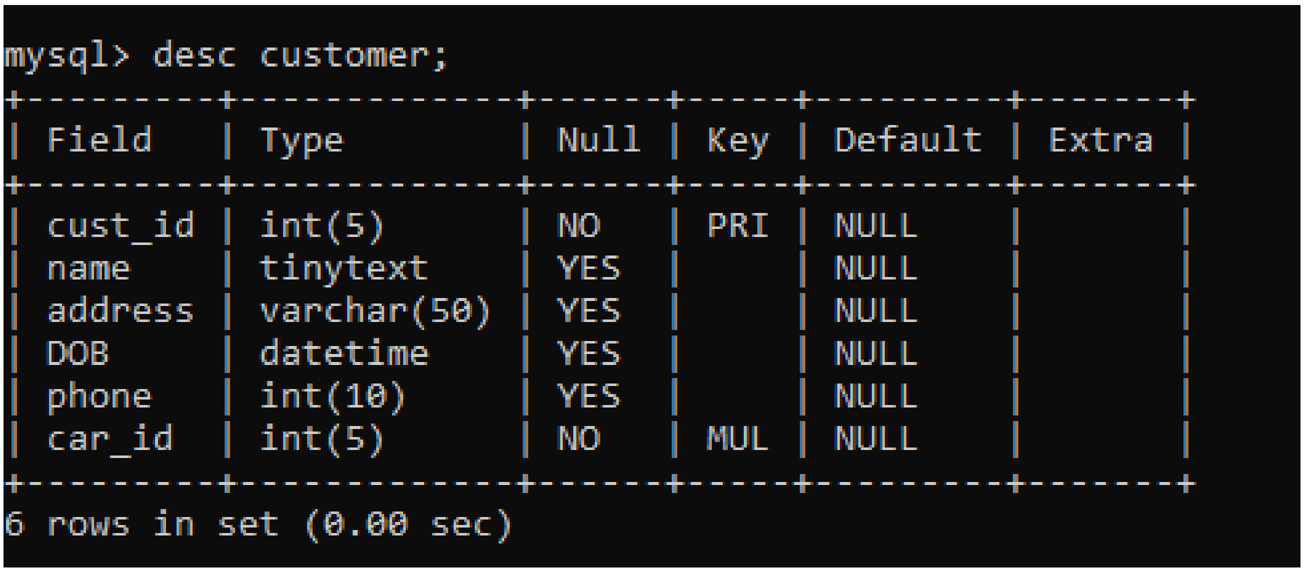


TABLE STRUCTURES

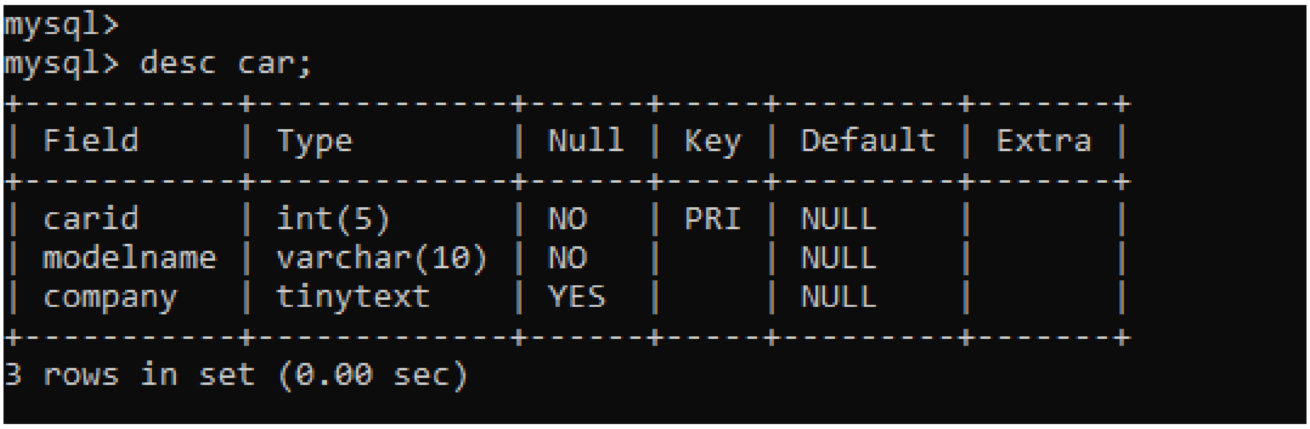
The structure of all tables included in the project is as under:-

1. Customer



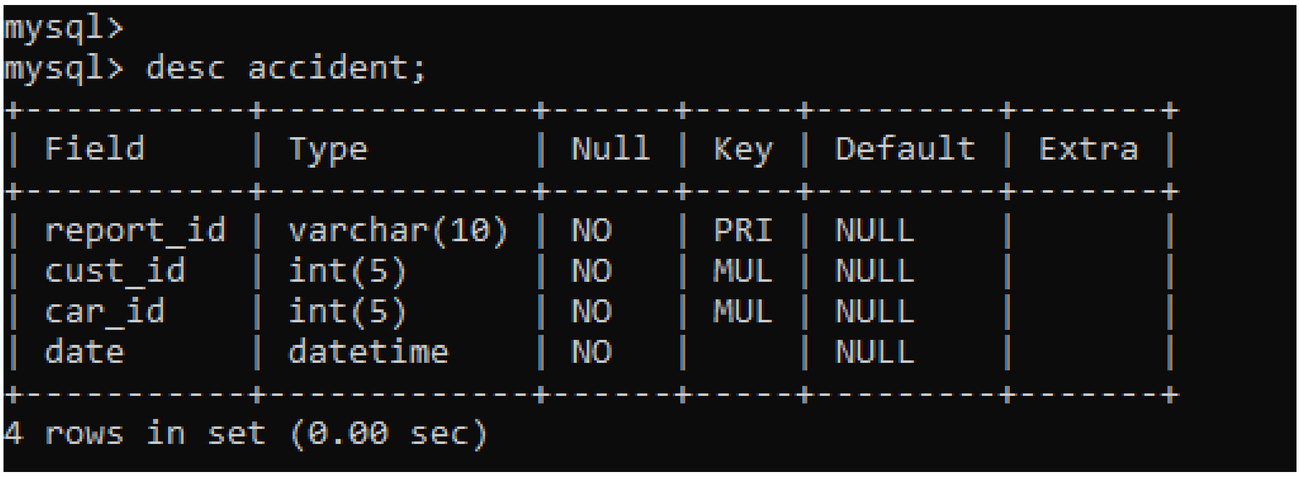
```
mysql> desc customer;
+----------+-------------+------+-----+---------+-------+
| Field    | Type        | Null | Key | Default | Extra |
+----------+-------------+------+-----+---------+-------+
| cust_id  | int(5)      | NO   | PRI | NULL    |       |
| name     | tinytext    | YES  |     | NULL    |       |
| address  | varchar(50) | YES  |     | NULL    |       |
| DOB      | datetime    | YES  |     | NULL    |       |
| phone    | int(10)     | YES  |     | NULL    |       |
| car_id   | int(5)      | NO   | MUL | NULL    |       |
+----------+-------------+------+-----+---------+-------+
6 rows in set (0.00 sec)
```

2. Car



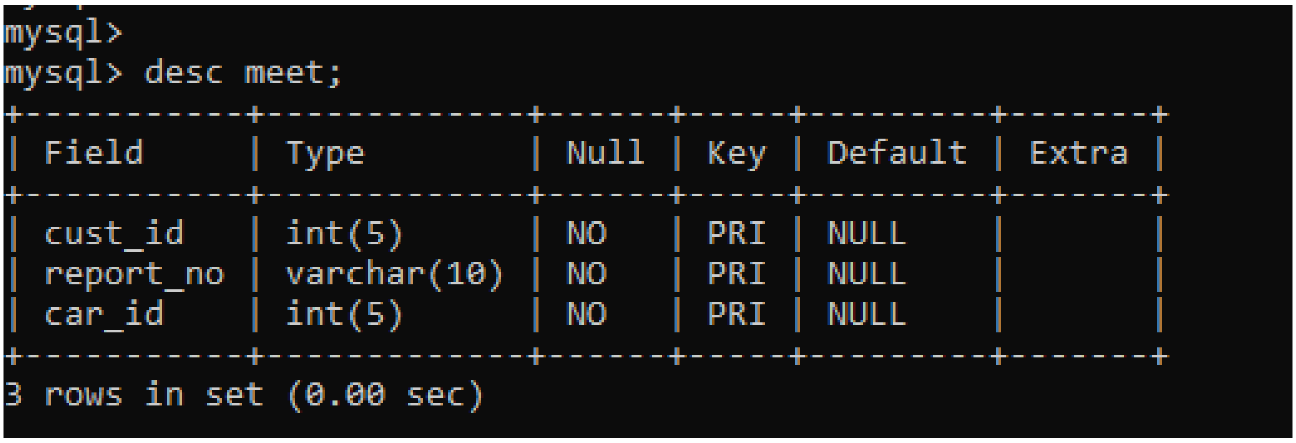
```
mysql>
mysql> desc car;
+-----------+-------------+------+-----+---------+-------+
| Field     | Type        | Null | Key | Default | Extra |
+-----------+-------------+------+-----+---------+-------+
| carid     | int(5)      | NO   | PRI | NULL    |       |
| modelname | varchar(10) | NO   |     | NULL    |       |
| company   | tinytext    | YES  |     | NULL    |       |
+-----------+-------------+------+-----+---------+-------+
3 rows in set (0.00 sec)
```

3. Accident



```
mysql>
mysql> desc accident;
+-----------+-------------+------+-----+---------+-------+
| Field     | Type        | Null | Key | Default | Extra |
+-----------+-------------+------+-----+---------+-------+
| report_id | varchar(10) | NO   | PRI | NULL    |       |
| cust_id   | int(5)      | NO   | MUL | NULL    |       |
| car_id    | int(5)      | NO   | MUL | NULL    |       |
| date      | datetime    | NO   |     | NULL    |       |
+-----------+-------------+------+-----+---------+-------+
4 rows in set (0.00 sec)
```

4. Meet



```
mysql>
mysql> desc meet;
+-----------+-------------+------+-----+---------+-------+
| Field     | Type        | Null | Key | Default | Extra |
+-----------+-------------+------+-----+---------+-------+
| cust_id   | int(5)      | NO   | PRI | NULL    |       |
| report_no | varchar(10) | NO   | PRI | NULL    |       |
| car_id    | int(5)      | NO   | PRI | NULL    |       |
+-----------+-------------+------+-----+---------+-------+
3 rows in set (0.00 sec)
```

Functional Dependencies

1] Customer Table: -

R(cust_id,cust_name,add,ph_no,DOB)={Cust_id->cust_name, cust_id->address, cust_id->ph_no, cust_id->DOB}

- ➢ Cust_id+ =>cust_id,cust_name,add,Ph_no,DOB

❖ **Union -**

Cust_id attribute uniquely identifies so cust_id is functionaly dependent on the customer_name ,address, phone_no, DOB.

- ➢ Cust_id->cust_name,address,ph_no,DOB

❖ **Transitive Dependency-**

- ➢ Cust_id->cust_name
  Cust_name->address
  Cust_id->address

❖ **Reflexivity –**

- ➢ Cust_id,cust_name->cust_id

❖ **Augmentation-**

- ➢ Cust_id->cust_name
  Cust_id ph_no-> cust_name ph_no

❖ **Decomposition-**

- ➢ Cust_id-> cust_name,address,ph_no,DOB
  Cust_id->cust_name , cust_id->ph_no , cust_id-> address and cust_id->DOB

2] Car table: -

R(car_id,model_name,company)={car_id->model_name, car_id->comapny}

- Car_id+ =>car_id,model_name,company   //minimum set
- Car_id,model_name+ =>car_id,model_name,company

- **Reflexivity-**
  - Car_id,model_name->model_name
- **Transitive-**
  - Car_id->model_name

    Model_name->company

In car table car_id is primary key attribute. Model_name and company are independent of each other but it depends on car_id. Car_id is functionaly dependent on model_name and company.

- Car_id->model_name
- Car_id->company

<u>3] Accident table:-</u>

R(Report_id,cust_id,car_id,date)

- **Transitivity:-**
  - Report_no->cust_id
    Cust_id->car_id
    Report_no->car_id
- **Reflexivity:-**
  - Report_id,date-> date
- **Augmentation:-**
  - Report_id ->cust_id
    Report_id date -> cust_id   date

NORMALIZATION

Normalization is a method for organizing data elements in a database into tables.

It keeps track to keep the database less vulnerable to some types of logical inconsistencies and anomalies. Tables can be normalized to varying degrees like first, second, third, BCNF, Greater the degree of normalization more is the protected from inconsistencies and anomalies.

The tables may be normalized under the following guideline:

1. **1NF**

   A relation schema is in 1NF if all of its attributes are:

   - Single valued
   - Restricted to assuming atomic values
   - Functionally dependent on the primary key

   Domain is atomic if its elements are considered to be indivisible units.

   Examples of non-atomic domains

   Set of names, composite attributes

   - Identification numbers like customer_id that can be broken up into parts
   - A relational schema R is in first normal form if the domains of all attributes of R are atomic.
   - All domains in our database are atomic since they are indivisible.
   - No Duplication of data ,Insert Anomaly ,Delete Anomaly ,Update Anomaly found, therefore our database clears the first normal form test.

2. **2NF**

   DEFINITION:

   A relational table is said to be in second normal form 2NF if it is in 1NF and every non-key attribute is fully functionally dependent upon primary key.

   The criteria for second normal form (2NF) are:

- The table must be in 1NF.
- Every non-key attributes of the table must be dependent upon the entire primary key.
- Tables customer, car, accident, meets are also in 2NF.
- Our database satisfies all the conditions of 2NF since the tables are in 1NF and every non-key attributes of the table must be dependent upon the entire primary key.

3. **3NF**

- A relation is in 3NF if and only if, it is in 2NF and there are no transitive functional dependencies.
- Transitive functional dependencies arise. When one non-key attribute is functionally dependent on another non-key attribute.
- Functional Dependency: non-key attribute -> non-key attribute.
- When there is redundancy in the database.
- The tables customer, car, accident, meets have been converted into 3NF.

By definition transitive functional dependency can only occur if there is more than one non-key field, so we can say that a relation in 2NF with zero or one non-key field must automatically be in 3NF.

CONCLUSION

A computerized insurance management system has been developed and the system was tested with sample data. The system results in regular timely preparations of required outputs. In comparison with manual system the benefits under a computer system are considerable in the saving of man power working hours and Effort.

Provision for addition, updation and deletion of customers is there in the system. It is observed that proper filing system has been adopted for future reference. The entire project runs on windows environments. The system can be used to make better management described at appropriate time. The user gets amount and timely information system.

Reference

- https://www.javatpoint.com/dbms-er-model-concept
- https://www.gatevidyalay.com/er-diagrams-to-tables
- https://www.javatpoint.com/dbms-reduction-of-er-diagram-into-table