

**CASE STUDY OF BANK DATABASE MANAGEMENT SYSTEM**

**IN RELATIONAL DATABASE DESIGN**

**CASE STUDY IN RELATIONAL DATABASE DESIGN**

**TITLE: BANK DATABASE MANAGEMENT SYSTEM**

Submitted to: Department of computer science

Batch (2020-2024)

STUDENT’S NAME: ANANYA JAIN (2010991473)

ANANYA MAHOTRA (2010991474)

ANGAD SINGH BEDI (2010991475)

GUIDE: Dr Vikas Solanki

**ABSTRACT**

The objective of this thesis is to develop a Bank Database Management System to keep a track on customers bank account details. The aim of designing and developing this bank management system primarily based engineering project is to provide secure and efficient net banking facilities to the bank customers over the internet. Apache Server Pages, MYSQL database used to develop this bank application where all banking customers can login through the secured web page by their account login id and password. Users will have all options and features in that application like get money from western union, money transfer to others, and send cash or money to inter banking as well as other banking customers by simply adding them as payees.

One case study “Bank Database Management System” is presented. Input for this case study is taken from its informal specification to a relational schema using entity-relationship modelling and its translation to relational model, to database schema, to implementation of the database, to interactive SQL querying of the installed database (SQL/Oracle).

**ACKNOWLEDGEMENT**

I would like to express my gratitude to all of those who made it possible to complete this thesis, in particular to my supervisor Dr Vikas Solanki. I would also like to thank my family for their understanding and continuous support.

**CONTENTS**

|  |  |
| --- | --- |
| Abstract | 2 |
| Acknowledgement | 3 |
| Chapter 1: Introduction | 5-10 |
| Chapter 2: Name of case study | 11-20 |
| 2.1 Case study informal description |  |
| 2.2 Terminologies and symbol of E-R diagram |  |
| 2.3 Case study Logical Model (ER Diagram) |  |
| 2.4 Case study physical schema |  |
| 2.5 Case study interactive queries |  |
| Chapter 3: Conclusion | 21 |
| Bibliography | 22 |

Annexure 23

**Chapter 1: Introduction**

* 1. **Database Management Systems**

A database management system (or DBMS) is essentially nothing more than **a computerized data- keeping system**. Users of the system are given facilities to perform several kinds of operations on such a system for either manipulation of the data in the database or the management of the database structure itself.

* 1. **Relational Database Management System**

A relational [database](https://searchdatamanagement.techtarget.com/definition/database) management system (RDBMS) is a collection of programs and capabilities that enable IT teams and others to create, update, administer and otherwise interact with a [relational database](https://searchdatamanagement.techtarget.com/definition/relational-database). RDBMS store data in the form of tables, with most commercial relational database management systems using [Structured Query Language](https://searchsqlserver.techtarget.com/definition/SQL) (SQL) to access the database. However, since SQL was invented after the initial development of the relational model, it is not necessary for RDBMS use.

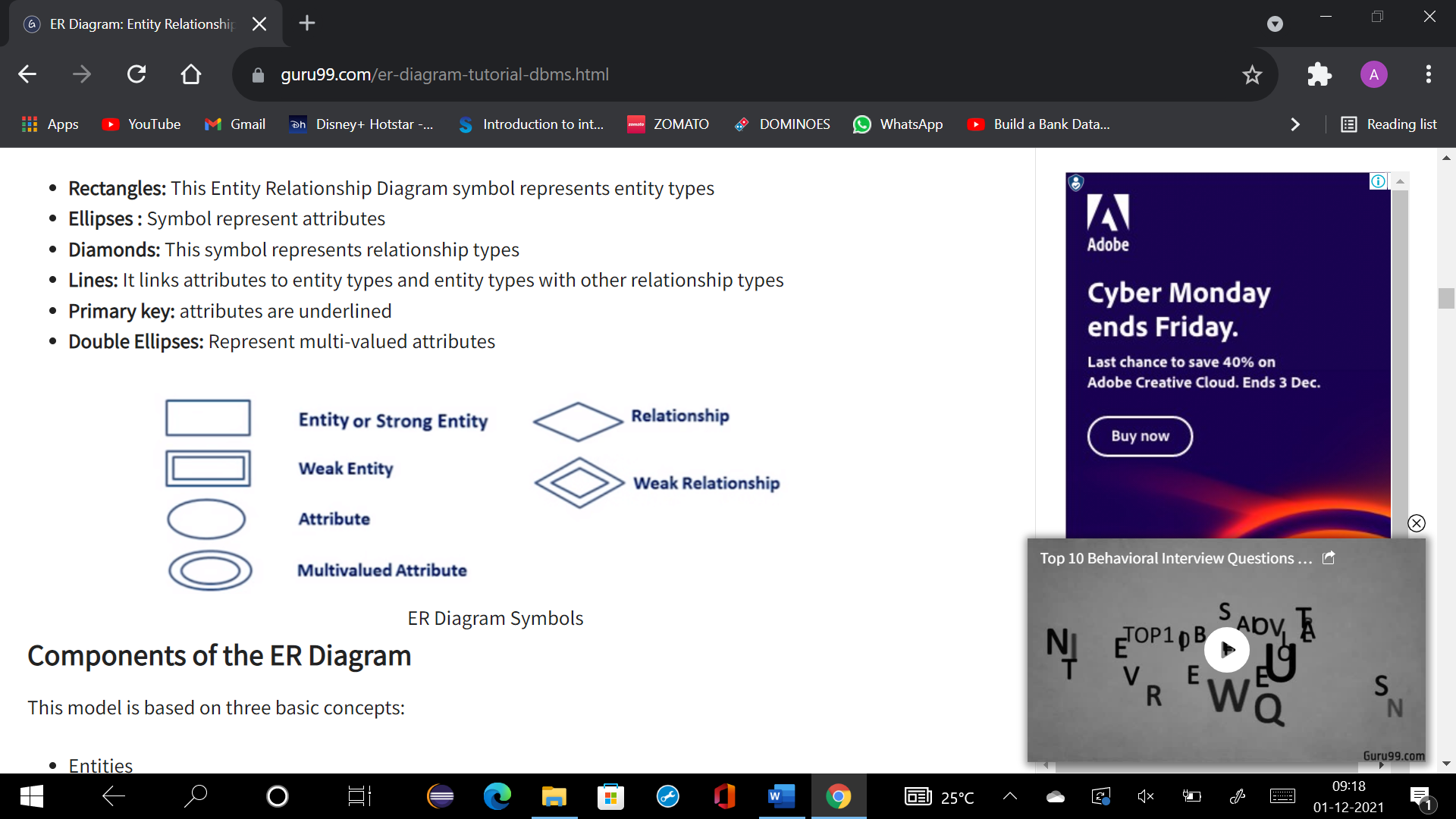
The RDBMS is the most popular database system among organizations across the world. It provides a dependable method of storing and retrieving large amounts of data while offering a combination of system performance and ease of implementation.

* 1. **ER Diagram**

**ER Diagram** stands for Entity Relationship Diagram, also known as ERD is a diagram that displays the relationship of entity sets stored in a database. In other words, ER diagrams help to explain the logical structure of databases. ER diagrams are created based on three basic concepts: entities, attributes and relationships.

**Entity Relationship Diagram Symbols & Notations** mainly contains three basic symbols which are rectangle, oval and diamond to represent relationships between elements, entities and attributes. There are some sub-elements which are based on main elements in ERD Diagram. ER Diagram is a visual representation of data that describes how data is related to each other using different ERD Symbols and Notations.

* **Rectangles:**This Entity Relationship Diagram symbol represents entity types
* **Ellipses:**Symbol represent attributes
* **Diamonds:**This symbol represents relationship types
* **Lines:**It links attributes to entity types and entity types with other relationship types
* **Primary key:**attributes are underlined
* **Double Ellipses:**Represent multi-valued attributes



This model is based on three basic concepts:

* Entities
* Attributes
* Relationships

**Entities**

A real-world thing either living or non-living that is easily recognizable and nonrecognizable. It is anything in the enterprise that is to be represented in our database. It may be a physical thing or simply a fact about the enterprise or an event that happens in the real world.

An entity can be place, person, object, event or a concept, which stores data in the database. The characteristics of entities are must have an attribute, and a unique key. Every entity is made up of some ‘attributes’ which represent that entity.

Example: customer\_personal\_info , account\_details , transaction\_details , branch\_details

**Relationship**

Relationship is nothing but an association among two or more entities.

**Weak Entities**

A weak entity is a type of entity which doesn’t have its key attribute. It can be identified uniquely by considering the primary key of another entity. For that, weak entity sets need to have participation.

|  |  |
| --- | --- |
| Strong Entity Set | Weak Entity Set |
| Strong entity set always has a primary key. | It does not have enough attributes to build a primary key. |
| It is represented by a rectangle symbol. | It is represented by a double rectangle symbol. |
| It contains a Primary key represented by the underline symbol. | It contains a Partial Key which is represented by a dashed underline symbol. |
| The member of a strong entity set is called as dominant entity set. | The member of a weak entity set called as a subordinate entity set. |
| Primary Key is one of its attributes which helps to identify its member. | In a weak entity set, it is a combination of primary key and partial key of the strong entity set. |
| In the ER diagram the relationship between two strong Entity set shown by using a diamond symbol. | The relationship between one strong and a weak entity set shown by using the double diamond symbol. |
| The connecting line of the strong entity set with the relationship is single. | The line connecting the weak entity set for identifying relationship is double. |

**Attributes**

It is a single-valued property of either an entity-type or a relationship-type.

Types of Attributes

* Simple Attribute: Simple attributes can’t be divided any further. For example, a student’s contact number. It is also called an atomic value.
* Composite Attribute: It is possible to break down composite attribute. For example, a student’s full name may be further divided into first name, second name, and last name.
* Derived Attribute: This type of attribute does not include in the physical database. However, their values are derived from other attributes present in the database. For example, age should not be stored directly. Instead, it should be derived from the DOB of that employee.
* Multivalued Attribute: Multivalued attributes can have more than one values. For example, a student can have more than one mobile number, email address, etc.

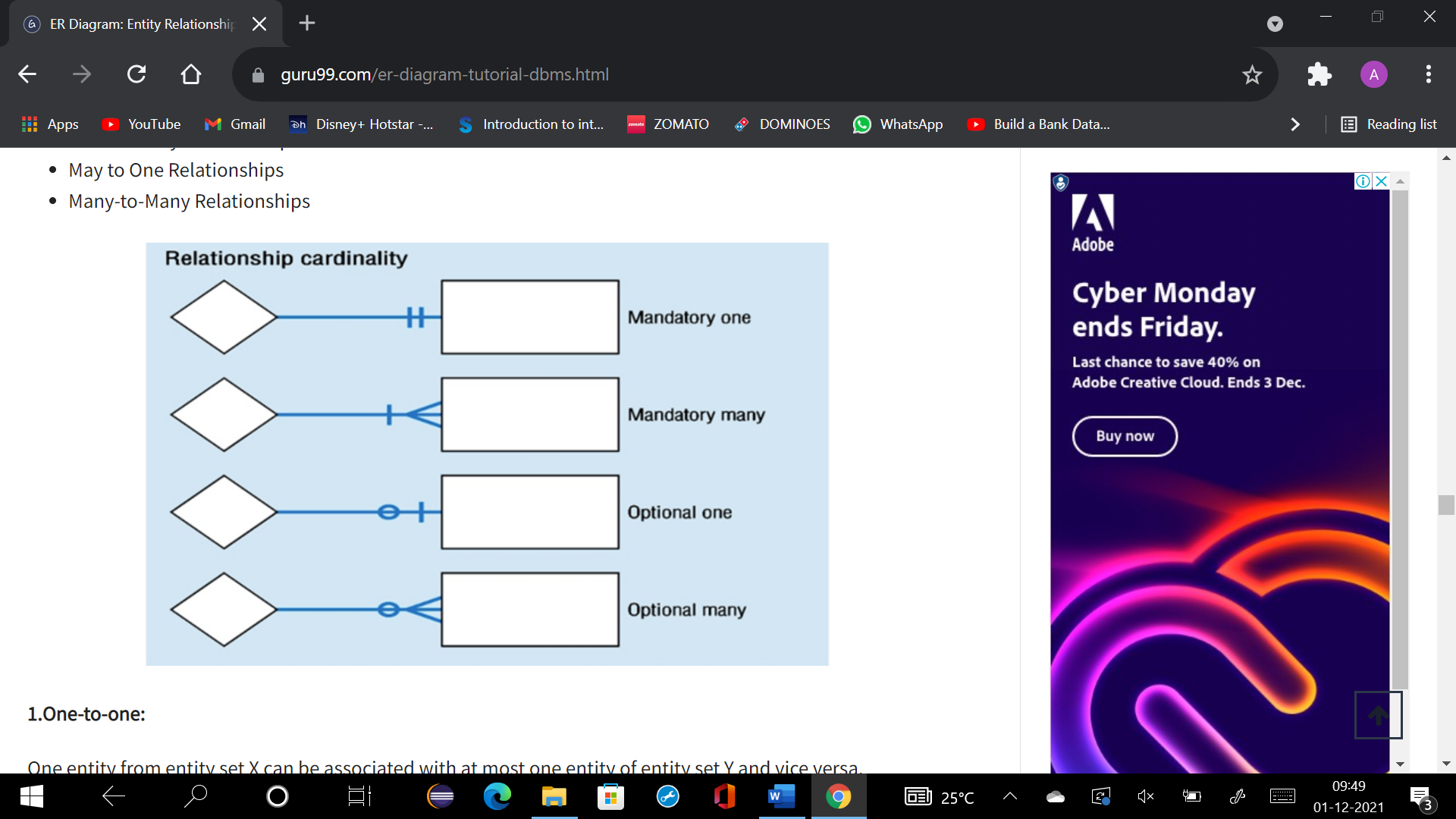
Example: CUSTOMER\_ID, CUSTOMER\_NAME, DATE\_OF\_BIRTH, GUARDIAN\_NAME, Address, CONTACT\_NO, MAIL\_ID, GENDER, MARITAL\_STATUS, IDENTIFICATION\_DOC\_TYPE, ID\_DOC\_NO, CITIZENSHIP Customer\_id, customer\_account\_number, account\_balance, transaction\_id , transation\_amount, transation\_status, CUSTOMER\_id, NAME\_OF\_BANK, IFSC\_CODE, LOCATION

Cardinality

Defines the numerical attributes of the relationship between two entities or entity sets.

Different types of cardinal relationships are:

* One-to-One Relationships
* One-to-Many Relationships
* May to One Relationships
* Many-to-Many Relationships



**1.One-to-one:**

One entity from entity set X can be associated with at most one entity of entity set Y and vice versa.

**2.One-to-many:**

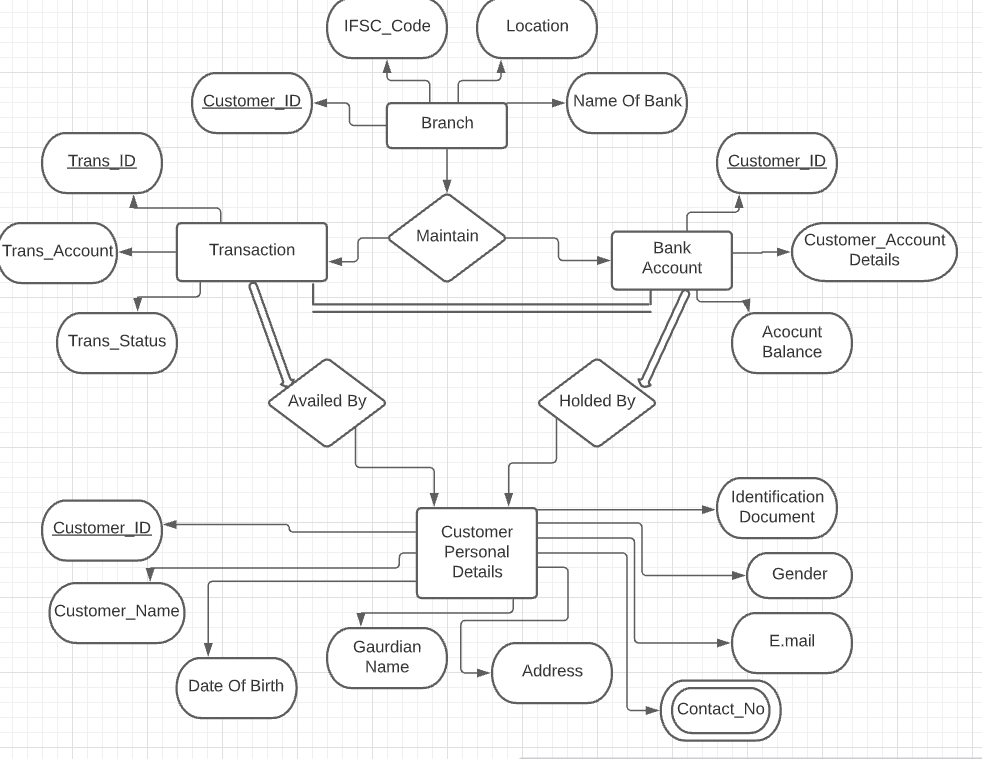
One entity from entity set X can be associated with multiple entities of entity set Y, but an entity from entity set Y can be associated with at least one entity.

**3. Many to One**

More than one entity from entity set X can be associated with at most one entity of entity set Y. However, an entity from entity set Y may or may not be associated with more than one entity from entity set X.

**4. Many to Many:**

One entity from X can be associated with more than one entity from Y and vice versa.

****

* 1. **Brief Introduction of case study**

The bank account management system is an application allowing customers to perform basic transactions from an automatic machine Bank, telephone, via a computer or with a smartphone via the Internet. The system allows the customer to create an account, deposit / withdraw money from his account, as well as view reports from all accounts present.

Customers can access the banks website to view their account details and perform the transactions on their account according to their requirements. The connection is made by secure access.

* 1. **Objective of the case study**

The main objectives of this bank database management system is to keep a track on customers bank account details. The aim of designing and developing this bank management system primarily based engineering project is to provide secure and efficient net banking facilities to the bank customers over the internet. Apache Server Pages, MYSQL database used to develop this bank application where all banking customers can login through the secured web page by their account login id and password. Users will have all options and features in that application like get money from western union, money transfer to others, and send cash or money to inter banking as well as other banking customers by simply adding them as payees.

**1.6 The structure of case study**

**1.61**

* MySQL Server- We have used My SQL Server Database for storing all the data.
* Operating System-We can run this project only on windows operating system.
* Database Portability (MS Access)-We can also setup this project with MS Access Database with some extra charges
* C++ The front end and the business logic section has been written in C++

**1.62** An entity relationship (ER) model is being constructed Microsoft Word.

**1.63** The oracle database schema is produced based on the ER model.

**1.64** Several SQL queries that can be used to query of the installed database are presented.

**Chapter 2: Bank Database Management System**

**2.1 Case Study Informal description**

This project on bank database management system is a web based project and it has been developed in MySql and we can manage Balance, accounts, savings accounts, current account, customer and employees from this project. The main objective to develop bank management system is to overcome the manual errors and make a computerized system.

In this project, there are various type of modules available to manage customer details like accounts, balance. We can also generate reports for accounts, balance, current account, employees. Here the balance module manage all the operations of balance, accounts module can manage current account operations, customer module has been implemented to manage customer.

**2.2 Case study Logical Models**

* **Entities**

1. **Branch Entity-**Branch ID contains attributes…Customer \_ID, Name of Bank, Location, IFSC \_Code. Customer \_ID is the primary key.
2. **Bank Account Entity-**Bank account contains all the personal info of the customer for example- Customer \_ID, Customer \_Account Details, Account Balance.
3. **Transaction Entity-**Transaction entity contains all the transfer info of the customer. It contains attributes like Transaction \_ID, Transaction \_Account, Trans \_Status. Trans \_ ID is the primary key in the table.
4. **Customer Entity-**It contains attributes like Customer \_ID, Customer \_Name, Date of birth, Guardian Name, Address, Contact \_No, E-mail, Gender, Identification Document and many more.

* **Branch maintains bank account:** 1: N

One Branch can have many Accounts but one Account can not belong to many Branches, so the relationship between Branch and Account is one to many relationships.

* **Branch maintains all the transactions:** 1 : N

One Branch can have many transactions but transactions can not belong to many Branches, so the relationship between Branch and Loan is one to many relationship.

* **Bank Account held by Customers**: M : N

One Customer can have more than one Bank Account and also Bank One Account can be held by one or more Customers, so the relationship between Account and Customers is many to many relationships.

**2.3 Case study Physical Models, Queries, Output Screenshots**

Physical schema(customer\_personal\_info)

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute | Data Type | Size | Constraint |
| CUSTOMER\_ID | INT |  | PRIMARY KEY, NOT NULL |
| CUSTOMER\_NAME | VARCHAR | 15 | - |
| DATE\_OF\_BIRTH | DATE | - | - |
| GUARDIAN\_NAME | VARCHAR | 15 | - |
| ADDRESS | VARCHAR | 63 | - |
| CONTACT\_NO | INT | - | - |
| MAIL\_ID | VARCHAR | 17 | UNIQUE |
| GENDER | VARCHAR | 10 | - |
| MARITAL\_STATUS | VARCHAR | 10 | - |
| IDENTIFICATION\_DOC\_TYPE | VARCHAR | 20 | DEFAULT |
| ID\_DOC\_NO | INT | - |  |
| CITIZENSHIP | VARCHAR | 10 | CHECK |

create table customer\_personal\_info

(

CUSTOMER\_ID INT,

CUSTOMER\_NAME VARCHAR(15),

DATE\_OF\_BIRTH DATE,

GUARDIAN\_NAME VARCHAR (15),

Address VARCHAR(63),

CONTACT\_NO INT,

MAIL\_ID VARCHAR(17),

GENDER VARCHAR(10),

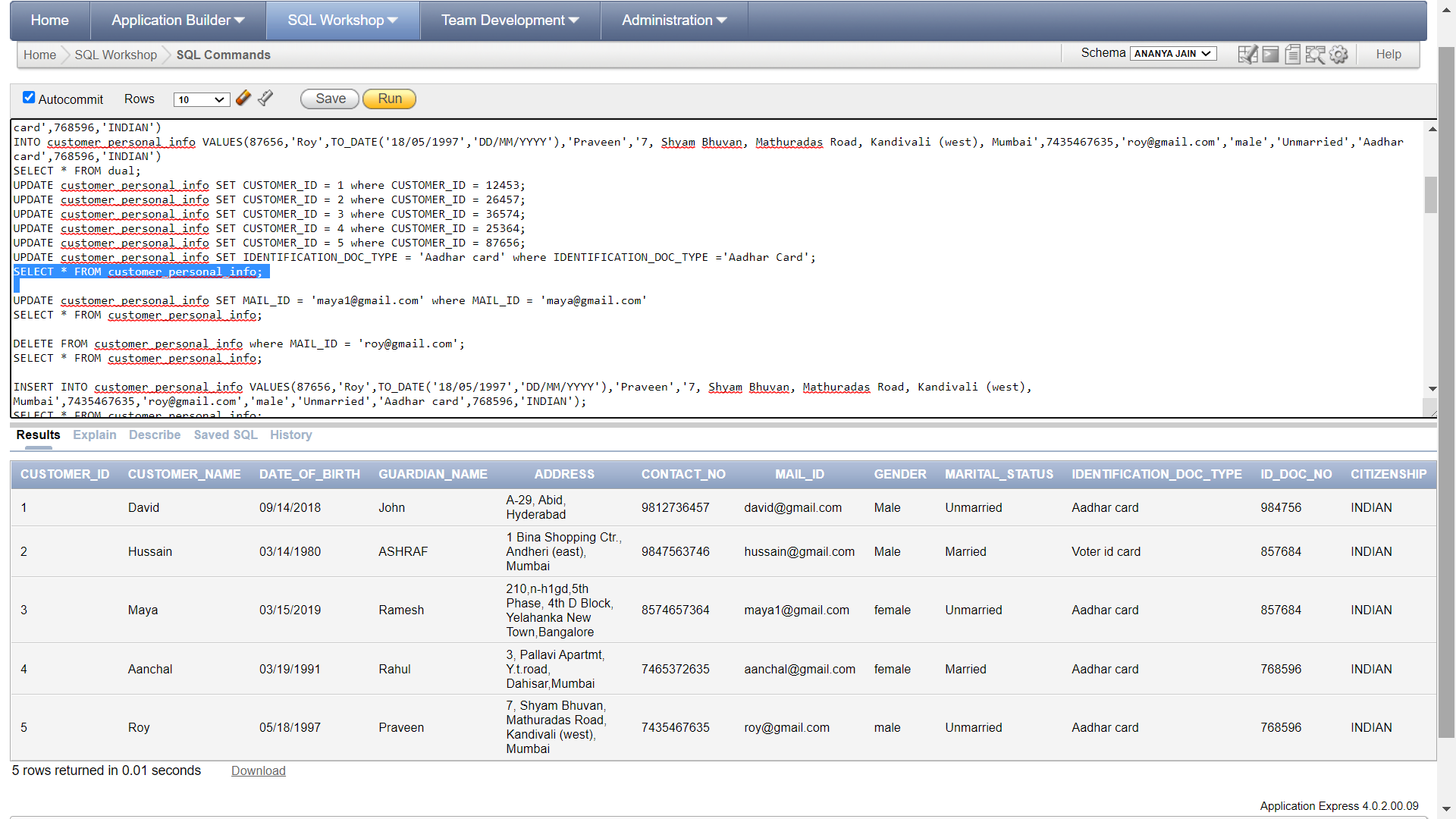
MARITAL\_STATUS VARCHAR(10),

IDENTIFICATION\_DOC\_TYPE VARCHAR(20),

ID\_DOC\_NO INT,

CITIZENSHIP VARCHAR(10)

);



ALTER TABLE customer\_personal\_info ADD PRIMARY KEY(CUSTOMER\_ID);

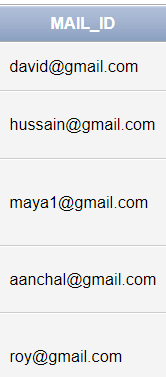
ALTER TABLE customer\_personal\_info MODIFY CUSTOMER\_ID NOT NULL;

ALTER TABLE customer\_personal\_info ADD UNIQUE (MAIL\_ID);

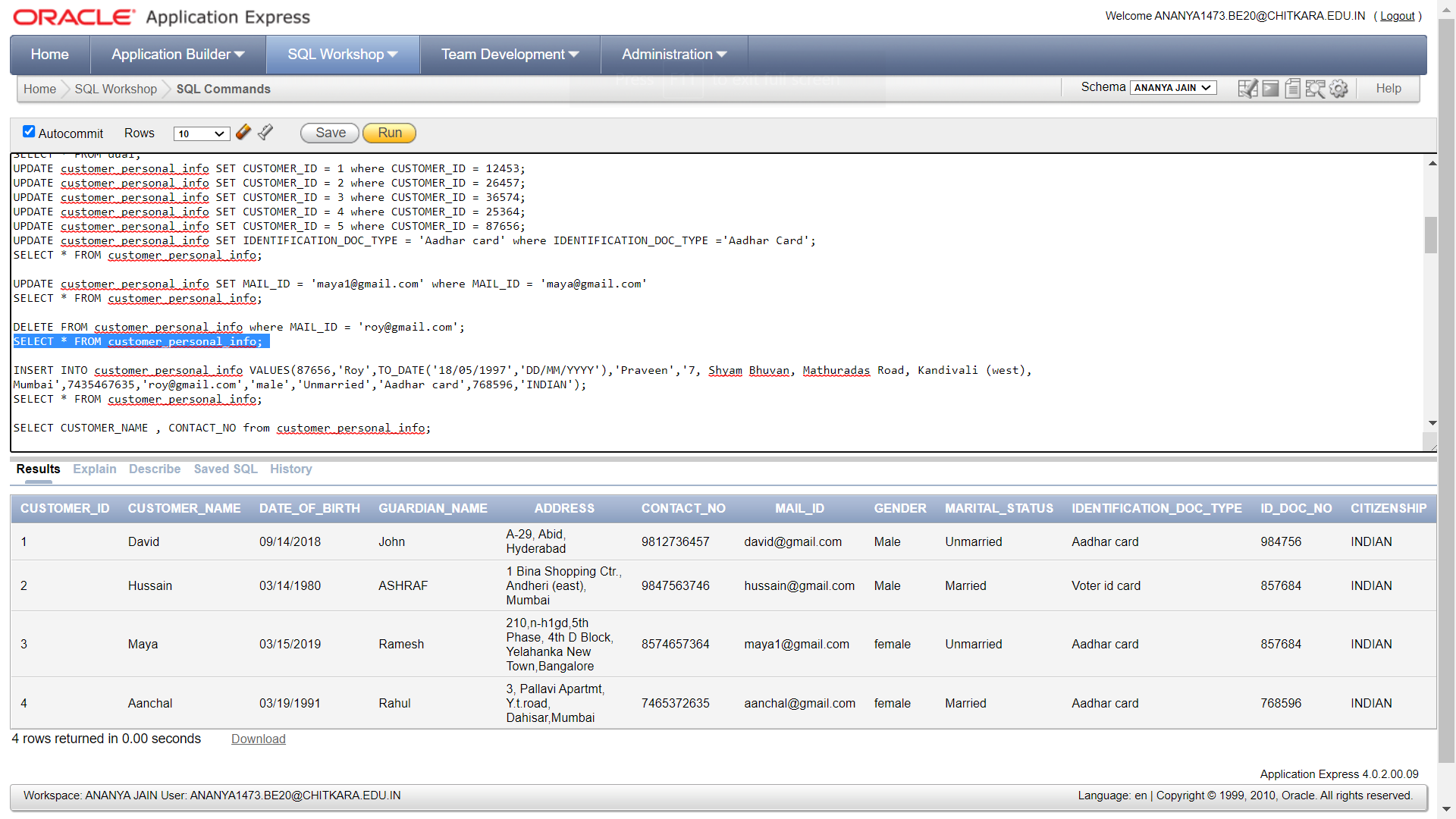
ALTER TABLE customer\_personal\_info MODIFY IDENTIFICATION\_DOC\_TYPE DEFAULT ‘Aadhar card’;

ALTER TABLE customer\_personal\_info ADD CONSTRAINT CITIZENSHIP CHECK (CITIZENSHIP =’INDIAN’);

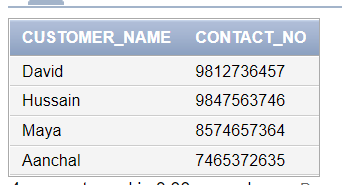
UPDATE customer\_personal\_info SET MAIL\_ID = ‘maya1@gmail.com’ where MAIL\_ID = ‘maya@gmail.com’



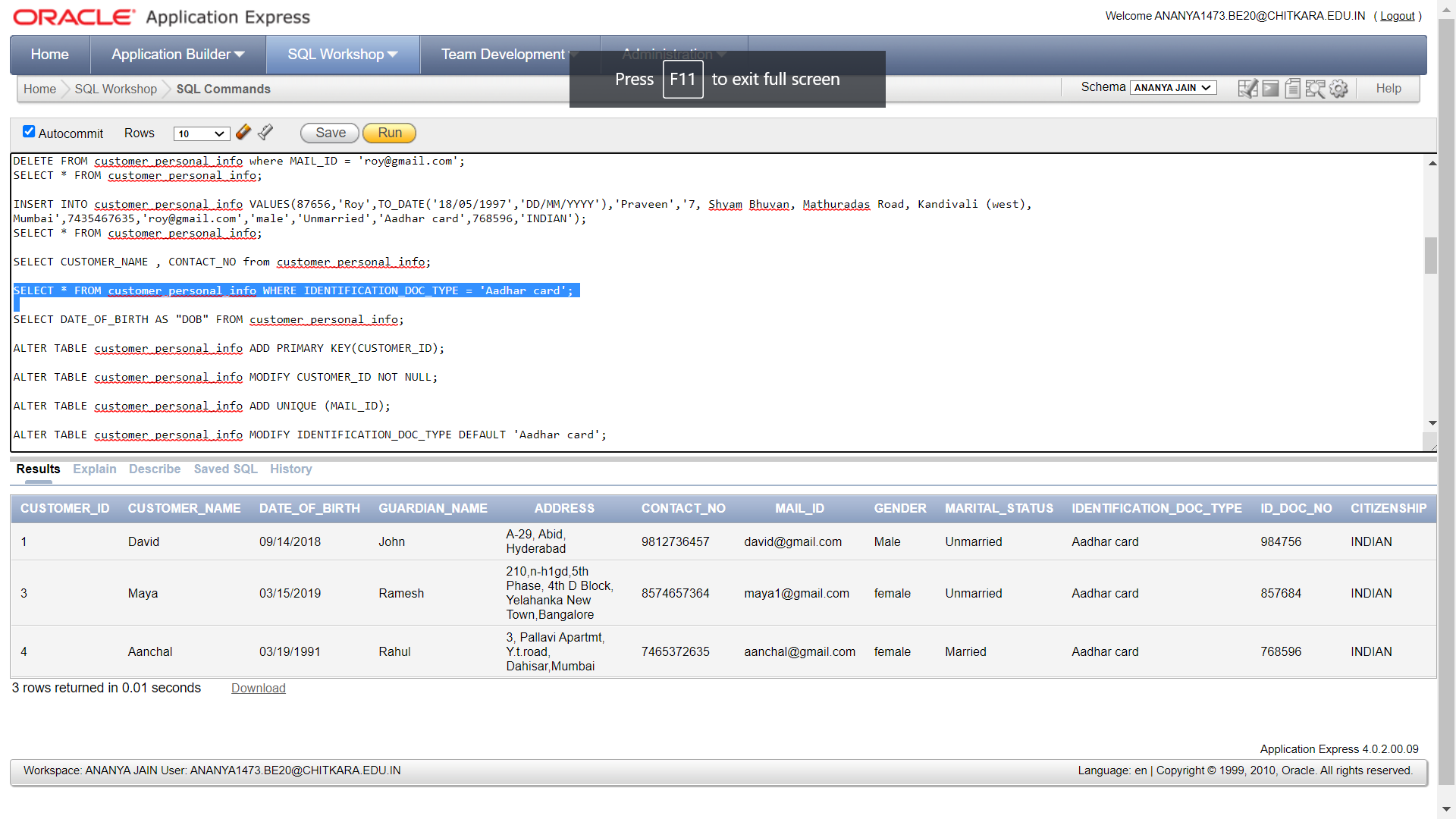
DELETE FROM customer\_personal\_info where MAIL\_ID = ‘roy@gmail.com’;



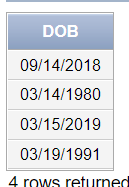
SELECT CUSTOMER\_NAME , CONTACT\_NO from customer\_personal\_info;



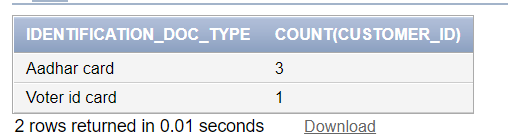
SELECT \* FROM customer\_personal\_info WHERE IDENTIFICATION\_DOC\_TYPE = ‘Aadhar card’;



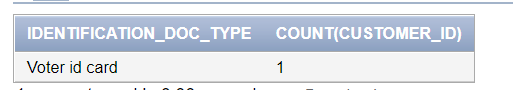
SELECT DATE\_OF\_BIRTH AS “DOB” FROM customer\_personal\_info;

****

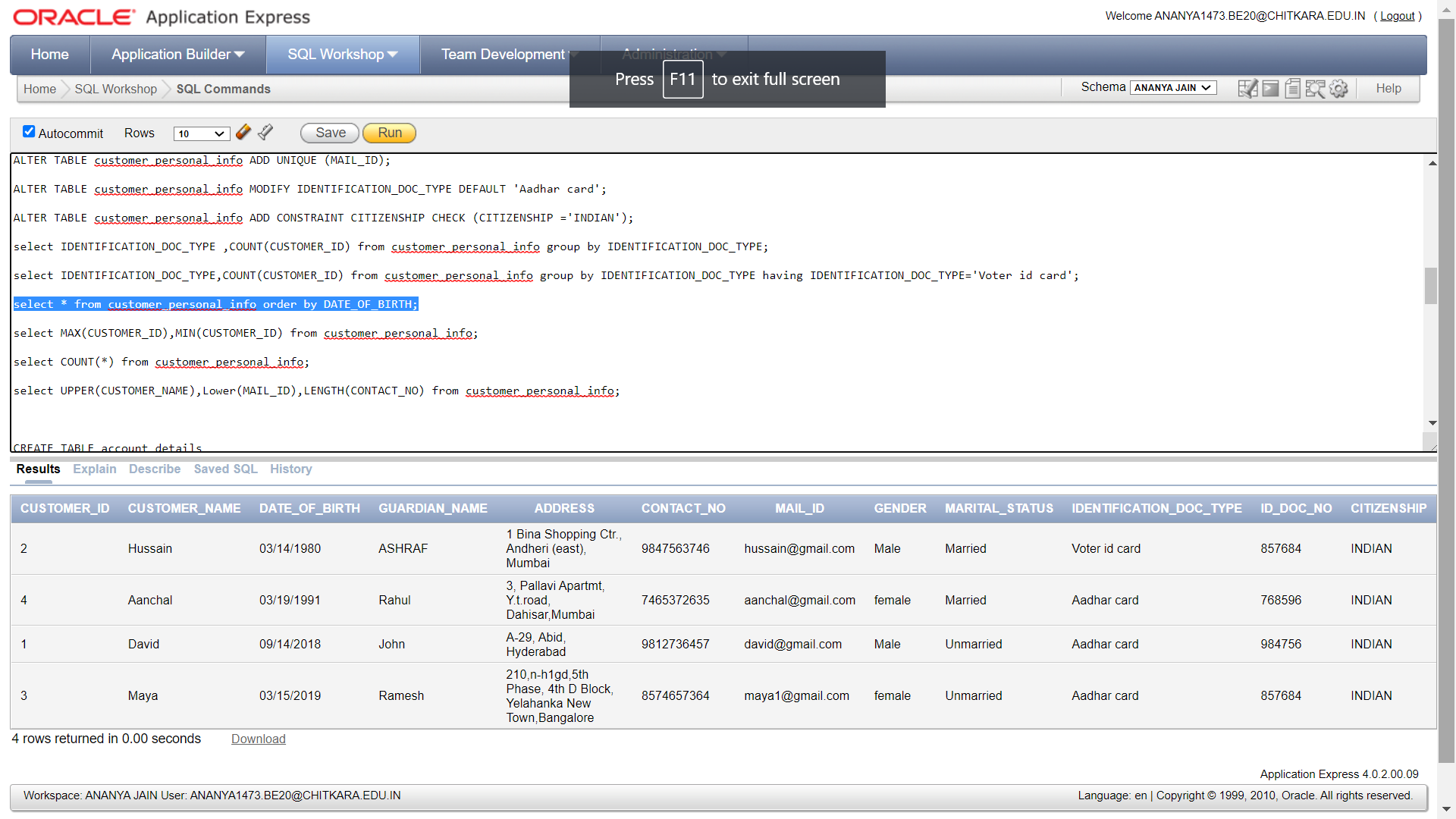
select IDENTIFICATION\_DOC\_TYPE ,COUNT(CUSTOMER\_ID) from customer\_personal\_info group by IDENTIFICATION\_DOC\_TYPE;

****

select IDENTIFICATION\_DOC\_TYPE,COUNT(CUSTOMER\_ID) from customer\_personal\_info group by IDENTIFICATION\_DOC\_TYPE having IDENTIFICATION\_DOC\_TYPE=’Voter id card’;

****

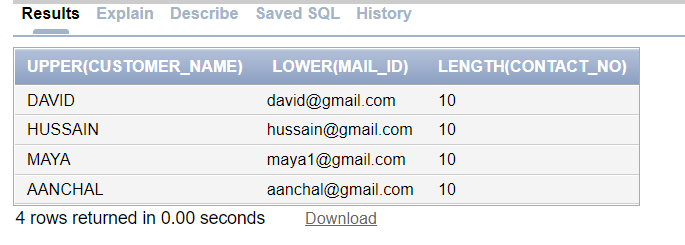
select \* from customer\_personal\_info order by DATE\_OF\_BIRTH;

****

select MAX(CUSTOMER\_ID),MIN(CUSTOMER\_ID) from customer\_personal\_info;

****

select UPPER(CUSTOMER\_NAME),Lower(MAIL\_ID),LENGTH(CONTACT\_NO) from customer\_personal\_info;

****

Physical schema (account\_details)

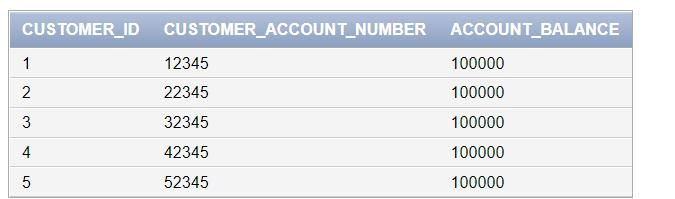
|  |  |  |  |
| --- | --- | --- | --- |
| Attribute | Data type | Size | Costraint |
| Customer\_id | INT | - | PRIMARY KEY |
| customer\_account\_number | INT | - | NOT NULL |
| account\_balance | INT | - | - |

CREATE TABLE account\_details(

Customer\_id INT PRIMARY KEY,s

customer\_account\_number INT NOT NULL,

account\_balance INT);

****

Physical schema (transaction\_details)

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute | Data type | Size | Constraint |
| transaction\_id | INT | - | PRIMARY KEY |
| transaction\_amount | INT | - | - |
| transaction\_status | VARCHAR | 10 | NOT NULL |

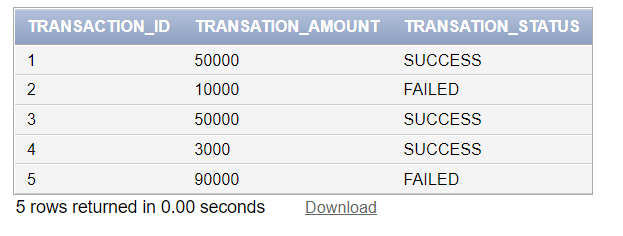
create table transaction\_details(

transaction\_id INT PRIMARY KEY,

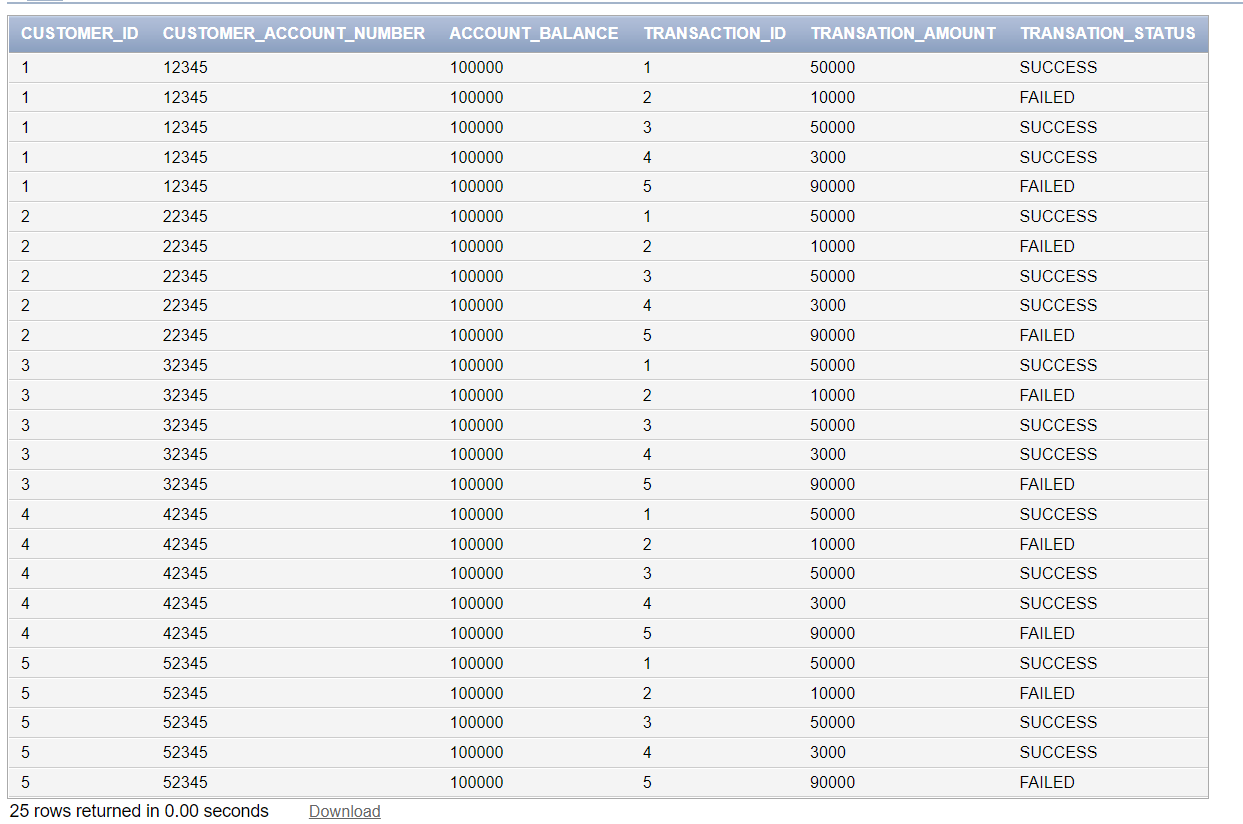
transation\_amount INT,

transation\_status VARCHAR(10) NOT NULL

);

****

select \* from account\_details cross join transaction\_details;



Physical schema (branch\_details)

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute | Data type | Size | Constraint |
| CUSTOMER\_id | INT | - | PRIMARY KEY |
| NAME\_OF\_BANK | VARCHAR | 50 | - |
| IFSC\_CODE | VARCHAR | 20 | NOT NULL |
| LOACTION | VARCHAR | 20 | - |

create table branch\_details(

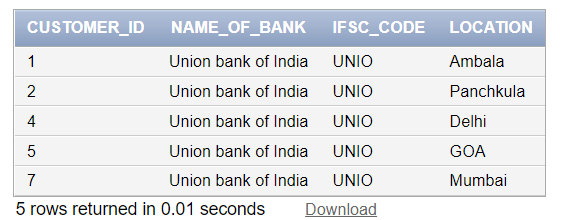
CUSTOMER\_id INT PRIMARY KEY,

NAME\_OF\_BANK VARCHAR(50),

IFSC\_CODE VARCHAR(20) NOT NULL,

LOCATION VARCHAR(20)

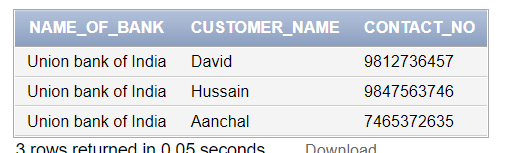
);



SELECT branch\_details.NAME\_OF\_BANK, customer\_personal\_info.CUSTOMER\_NAME , customer\_personal\_info.CONTACT\_NO FROM customer\_personal\_info

INNER JOIN branch\_details ON

customer\_personal\_info.CUSTOMER\_ID = branch\_details.CUSTOMER\_id;

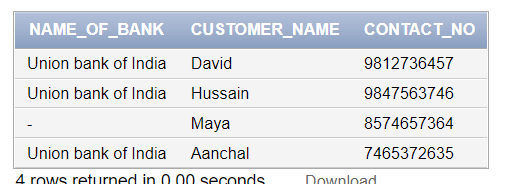


SELECT branch\_details.NAME\_OF\_BANK, customer\_personal\_info.CUSTOMER\_NAME , customer\_personal\_info.CONTACT\_NO

FROM customer\_personal\_info

LEFT JOIN branch\_details

ON customer\_personal\_info.CUSTOMER\_ID = branch\_details.CUSTOMER\_id;

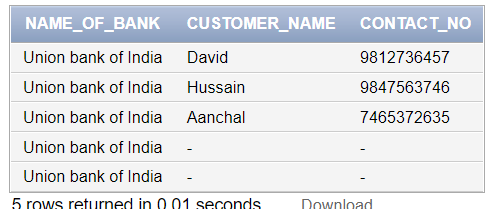


SELECT branch\_details.NAME\_OF\_BANK, customer\_personal\_info.CUSTOMER\_NAME , customer\_personal\_info.CONTACT\_NO

FROM customer\_personal\_info

RIGHT JOIN branch\_details

ON customer\_personal\_info.CUSTOMER\_ID = branch\_details.CUSTOMER\_id;

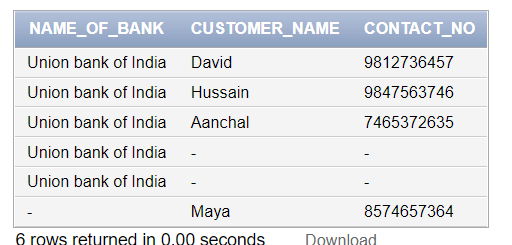


SELECT branch\_details.NAME\_OF\_BANK, customer\_personal\_info.CUSTOMER\_NAME , customer\_personal\_info.CONTACT\_NO

FROM customer\_personal\_info

FULL JOIN branch\_details

ON customer\_personal\_info.CUSTOMER\_ID = branch\_details.CUSTOMER\_id;



**Chapter-3**

**Conclusion and future work**

Bank management system is a virtualization of transactions in banking system. The banking system are used manual working but when we used online banking system it is totally virtualization process which avoid manual process and converts it in automatic process. If user can make a transaction in bank management system it is available in any were also user can link Aadhar with account. Bank management system is saving the time with accuracy than bank manual system.

**Bibliography**

* Code quotient
* W3schools
* Wikipedia
* Geeks for geeks
* Database system concepts by Abraham Silberschatz, Henry F. Korth, S. Sudarshan

**Annexure**

Work distribution table

|  |  |
| --- | --- |
| Student Name | Work Done |
| Ananya Jain (2010991473) | Coding |
| Ananya Malhotra (2010991474) | Summing up of Word File |
| Angad Singh Bedi (2010991475) | Research Work |