Bank Management System

A COURSE PROJECT REPORT

By

Harsh Khanijo(RA211003010225)

Under the guidance of **Dr.Mathew Eliazer sir**
In partial fulfillment for the Course
18CSC303J-Database Management Systems

In

School of Computing

FACULTY OF ENGINEERING AND TECHNOLOGY SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

Kattankulathur, Chengalpattu District

APRIL 2024.

Acknowledgement

We would like to express our gratitude to our Professor, Dr. S. Babu who gave us the golden opportunity to do this wonderful project on the topic **"BANK MANAGEMENT SYSTEM"** which also helped us in doing a lot of research and we came to know about so many new things we are
really thankful to him.

We are also thankful to all the other faculty, teaching and non-teaching staff members of our department for their kind cooperation and help.

Lastly, we would also like to thank our friends who helped us a lot in finishing this project within the limited time. We are making this project not only for marks but to also increase our knowledge.

Index

CHAPTER-1
1.INTRODUCTION

A Database is a collection of logically coherent data, which come together to serve a particular purpose and they emulate some aspect of the real world.

Databases are undoubtedly utilized in corporate applications and financial activities. Database management systems store and retrieve (lots of) data. Banks manage a lot of data. A DBMS lets them store, manipulate, and retrieve data rapidly enough for them and their customers. Traditional relational databases will always be used by banks in their IT infrastructure, where they can serve as valuable systems of record. The major qualities covered by this database application include:

- Atomicity: When several modifications are part of a single transaction, they all fail or all succeed; never a portion. If I transfer money to you and the computer fails after debiting my account but before crediting yours, the RDBMS will guarantee that the rest of the transaction completes or (typically) the portion that was previously done is undone.
- Consistency: The RDBMS ensures that business rules stated in the database are never broken.
- Isolation: Each user seems to be the only user of the database; you won't "see" unfinished work by other users.
- Once a transaction is reported as complete, the RDBMS guarantees the changes are permanent.
- Redundancy Control: A data which is stored multiple times at different locations is a redundant data. DBMS avoids it and creates only one record.
- Security: Users should be granted access to only those data and reports which they specifically need. It restricts users from viewing or updating data which is not in their scope.

This project is one such application which includes creating a database that helps in managing a banking system.

CHAPTER-2

A bank management system for a DBMS involves designing a schema to manage entities like customers, accounts, and transactions. It ensures data integrity, security, and transaction concurrency control. Additionally, it facilitates reporting, user interface, backup, scalability, compliance, and performance optimization to meet banking requirements effectively.

2.1.2 Main features are:

1. Customer Management
2. Account Management
3. Transaction Processing
4. Reporting and Analytics
5. Security and Authentication
6. Loan Management
7. Compliance and Regulations
8. Audit Trail
9. Alerts and Notifications
10. User Management

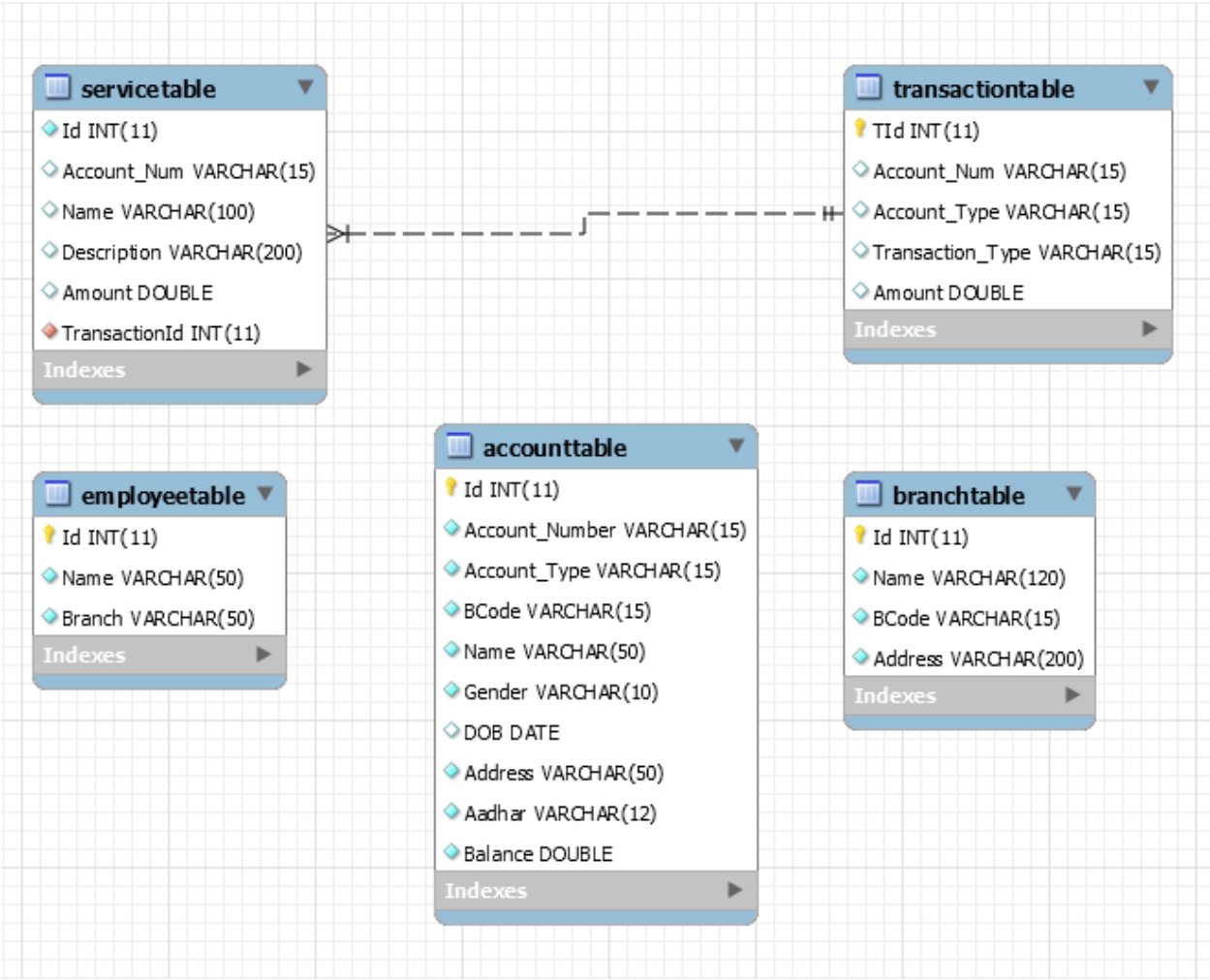
2.1.3 Objectives:

1. Streamlining customer interactions and transactions for improved service delivery.
2. Ensuring precise recording, tracking, and management of financial transactions and accounts.
3. Implementing robust security measures to safeguard customer data and prevent fraud and unauthorized access.
4. Ensuring adherence to banking regulations and compliance requirements, such as KYC, AML, and GDPR.
5. Optimizing banking processes to reduce manual efforts, errors, and operational costs.
6. Providing timely and accurate reports and analytics for informed decision-making and regulatory compliance.
7. Identifying, assessing, and mitigating risks associated with banking operations, including credit, market, and operational risks.
8. Designing the system to accommodate growth and changes in banking needs and technology advancements.
9. Integrating with other banking systems and third-party services for interoperability and enhanced functionality.
10. Focusing on delivering excellent customer experiences to build loyalty and retain customers in a competitive market.

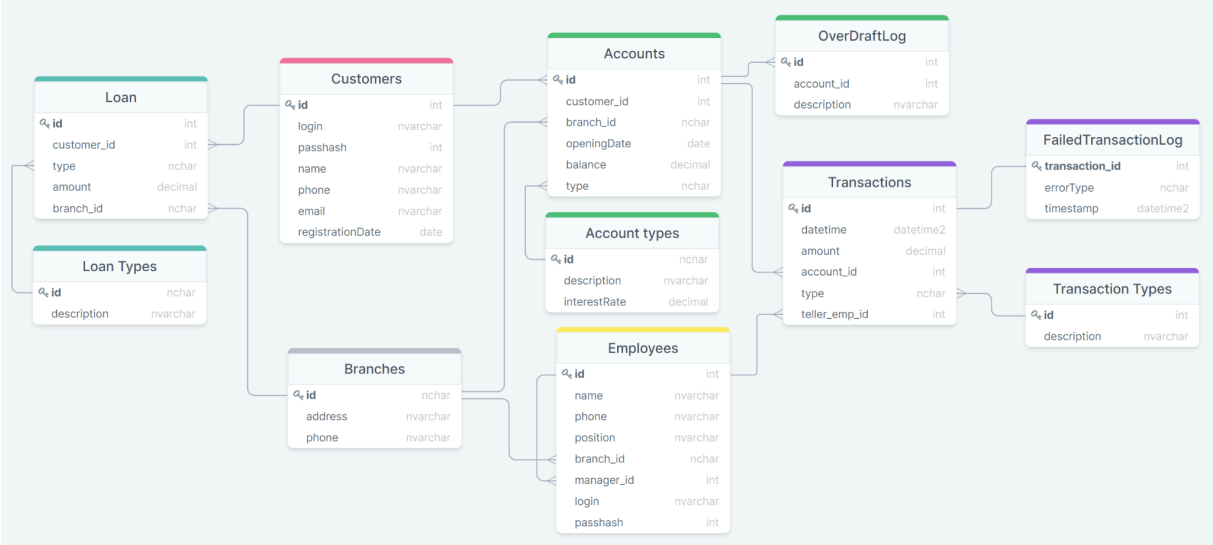
CHAPTER-3

3.1 BACK-END DESIGN

3.1.1 Conceptual Database Design(ER-Diagram)



3.1.2 Logical Database Design(ER Mapping)



- The entities are represented as tables.
- The tables contain the attributes.
- The attributes which are bold are referred to as primary keys.

3.2 FRONT-END DESIGN

3.2.1 Front-end web development details

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Bank Management System</title>
  <style>
    /* CSS styles */
    body {
      font-family: Arial, sans-serif;
      margin: 0;
      padding: 0;
      background-color: #f0f0f0;
    }
    .container {
      max-width: 800px;
      margin: 20px auto;
      padding: 20px;
      background-color: #fff;
      border-radius: 8px;
      box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
    }
    h1 {
      text-align: center;
    }
    form {
      margin-top: 20px;
    }
    input[type="text"], input[type="password"], input[type="number"] {
      width: 100%;
      padding: 10px;
      margin-bottom: 10px;
      border-radius: 4px;
      border: 1px solid #ccc;
      box-sizing: border-box;
    }
    button {
      width: 100%;
      padding: 10px;
      background-color: #007bff;
      color: #fff;
      border: none;
      border-radius: 4px;
      cursor: pointer;
    }
    button:hover {
      background-color: #0056b3;
    }
  </style>
</head>
<body>
  <div class="container">
    <h1>Bank Management System</h1>
    <form id="loginForm">
      <input type="text" id="username" placeholder="Username">
      <input type="password" id="password" placeholder="Password">
      <button type="submit">Login</button>
    </form>
    <div id="errorMessage" style="color: red; margin-top: 10px; display: none;"></div>
  </div>

  <script>
    // JavaScript code
    document.getElementById('loginForm').addEventListener('submit', function(event) {
      event.preventDefault();
      var username = document.getElementById('username').value;
      var password = document.getElementById('password').value;
      // Perform validation here if needed
      // Simulate login request (replace with actual login logic)
      if (username === 'admin' && password === 'password') {
        // Redirect to dashboard or perform other actions
        alert('Login successful!');
      } else {
        document.getElementById('errorMessage').innerText = 'Invalid username or password.';
        document.getElementById('errorMessage').style.display = 'block';
      }
```

```
  });
 </script>
</body>
</html>
```


3.2.2 Connectivity (front end and Back end):

```
mysql> desc clients;
+------------------+---------------+------+-----+---------+-------+
| Field            | Type          | Null | Key | Default | Extra |
+------------------+---------------+------+-----+---------+-------+
| acc_no           | int           | NO   | PRI | NULL    |       |
| acc_type         | enum('S','C') | NO   |     | NULL    |       |
| first_name       | varchar(15)   | NO   |     | NULL    |       |
| last_name        | varchar(15)   | NO   |     | NULL    |       |
| gender           | enum('M','F') | NO   |     | NULL    |       |
| birth_date       | date          | NO   |     | NULL    |       |
| acc_creation_date | date          | NO   |     | NULL    |       |
| mobile_no        | varchar(20)   | YES  |     | NULL    |       |
| email_id         | varchar(25)   | NO   |     | NULL    |       |
| pass             | varchar(8)    | NO   |     | NULL    |       |
+------------------+---------------+------+-----+---------+-------+
10 rows in set (0.00 sec)
```

Fig 3.2.2.1 Clients

```
mysql> desc current;
+-----------+------+------+-----+---------+-------+
| Field     | Type | Null | Key | Default | Extra |
+-----------+------+------+-----+---------+-------+
| acc_no    | int  | NO   | PRI | NULL    |       |
| balance   | int  | NO   |     | NULL    |       |
| overdraft | int  | NO   |     | NULL    |       |
+-----------+------+------+-----+---------+-------+
3 rows in set (0.00 sec)
```

Fig 3.2.2.2 Current

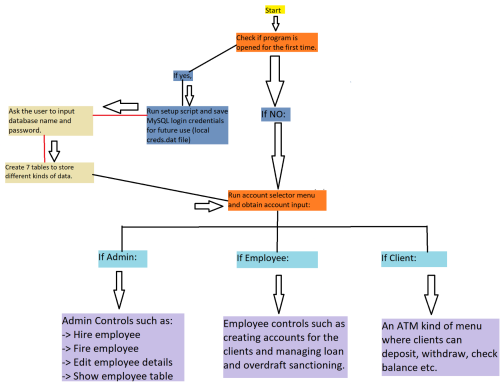


Fig 3.2.2.3 Structure

```
mysql> desc cash_in_hand;
+--------------+------+------+-----+---------+-------+
| Field        | Type | Null | Key | Default | Extra |
+--------------+------+------+-----+---------+-------+
| acc_no       | int  | NO   | PRI | NULL    |       |
| cash_in_hand | int  | NO   |     | NULL    |       |
+--------------+------+------+-----+---------+-------+
2 rows in set (0.00 sec)
```

Fig 3.2.2.4 cash_in_hand


```
mysql> desc empass;
+--------+------------+------+-----+---------+-------+
| Field  | Type       | Null | Key | Default | Extra |
+--------+------------+------+-----+---------+-------+
| emp_no | int        | NO   | PRI | NULL    |       |
| pass   | varchar(8) | NO   |     | NULL    |       |
+--------+------------+------+-----+---------+-------+
2 rows in set (0.00 sec)
```

Fig 3.2.2.5 empass

```
mysql> desc employees;
+------------+---------------+------+-----+---------+-------+
| Field      | Type          | Null | Key | Default | Extra |
+------------+---------------+------+-----+---------+-------+
| emp_no     | int           | NO   | PRI | NULL    |       |
| birth_date | date          | NO   |     | NULL    |       |
| first_name | varchar(15)   | NO   |     | NULL    |       |
| last_name  | varchar(15)   | NO   |     | NULL    |       |
| gender     | enum('M','F') | NO   |     | NULL    |       |
| hire_date  | date          | NO   |     | NULL    |       |
+------------+---------------+------+-----+---------+-------+
6 rows in set (0.01 sec)
```

Fig 3.2.2.6 employees

```
mysql> desc loan;
+----------------------+-------------------------------+------+-----+---------+-------+
| Field                | Type                          | Null | Key | Default | Extra |
+----------------------+-------------------------------+------+-----+---------+-------+
| acc_no               | int                           | NO   | PRI | NULL    |       |
| loan_type            | enum('PL','HL','EL','TL','BL') | NO   |     | NULL    |       |
| loan_amt             | int                           | NO   |     | NULL    |       |
| time_period_months   | int                           | NO   |     | NULL    |       |
| iterest_perc_per_annum | int                         | NO   |     | NULL    |       |
| amt-per-month        | int                           | NO   |     | NULL    |       |
| remaining_amt        | int                           | NO   |     | NULL    |       |
+----------------------+-------------------------------+------+-----+---------+-------+
7 rows in set (0.00 sec)
```

Fig 3.2.2.7 loan

```
mysql> desc overdraft;
+-----------------------------+------+------+-----+---------+-------+
| Field                       | Type | Null | Key | Default | Extra |
+-----------------------------+------+------+-----+---------+-------+
| acc_no                      | int  | NO   | PRI | NULL    |       |
| overdraft_amt               | int  | NO   |     | NULL    |       |
| od_with_interest_remaining  | int  | NO   |     | NULL    |       |
+-----------------------------+------+------+-----+---------+-------+
3 rows in set (0.00 sec)
```

Fig 3.2.2.8 overdraft

```
mysql> desc savings;
+---------+------------------+------+-----+---------+-------+
| Field   | Type             | Null | Key | Default | Extra |
+---------+------------------+------+-----+---------+-------+
| acc_no  | int              | NO   | PRI | NULL    |       |
| balance | int              | NO   |     | NULL    |       |
| loan    | enum('YES','NO') | NO   |     | NULL    |       |
+---------+------------------+------+-----+---------+-------+
3 rows in set (0.00 sec)
```

Fig 3.2.2.9 savings

CHAPTER-4

DATABASE CREATION

CODE:

- create the database "**bank**"

```
mysql> CREATE DATABASE bank;
```

- use the database bank:

```
mysql> USE bank;
```

- Creating table customer

```
CREATE TABLE customer
  (
     custid VARCHAR(6),
     fname VARCHAR(30),
     mname VARCHAR(30),
     ltname VARCHAR(30),
     city VARCHAR(15),
     mobileno VARCHAR(10),
     occupation VARCHAR(10),
     dob DATE,
    CONSTRAINT customer_custid_pk PRIMARY KEY(custid)
  );
```

- Creating table branch

```
CREATE TABLE branch
  (
   bid VARCHAR(6),
   bname VARCHAR(30),
   bcity VARCHAR(30),
   CONSTRAINT branch_bid_pk PRIMARY KEY(bid)
  );
```

- Creating table account

```
CREATE TABLE account
  (
    acnumber VARCHAR(6),
    custid  VARCHAR(6),
    bid VARCHAR(6),
    opening_balance INT(7),
    aod DATE,
    atype VARCHAR(10),
    astatus VARCHAR(10),
    CONSTRAINT account_acnumber_pk PRIMARY KEY(acnumber),
    CONSTRAINT account_custid_fk FOREIGN KEY(custid) REFERENCES customer(custid),
```

```
    CONSTRAINT account_bid_fk FOREIGN KEY(bid) REFERENCES branch(bid)
  );
```


- Creating table Tran details

```
CREATE TABLE trandetails
  (
   tnumber VARCHAR(6),
   acnumber VARCHAR(6),
   dot DATE,
   medium_of_transaction VARCHAR(20),
   transaction_type VARCHAR(20),
   transaction_amount INT(7),
   CONSTRAINT trandetails_tnumber_pk PRIMARY KEY(tnumber),
   CONSTRAINT trandetails_acnumber_fk FOREIGN KEY(acnumber) REFERENCES
account(acnumber)
  );
```

- Creating table loan

```
CREATE TABLE loan
 (

  custid VARCHAR(6),
  bid VARCHAR(6),
  loan_amount INT(7),
  CONSTRAINT loan_customer_custid_bid_pk PRIMARY KEY(custid,bid),
  CONSTRAINT loan_custid_fk FOREIGN KEY(custid) REFERENCES  customer(custid),
  CONSTRAINT loan_bid_fk FOREIGN KEY(bid) REFERENCES  branch(bid)
 );
```

Inserting Record:

```
INSERT INTO customer
VALUES('C00001','Ramesh','Chandra','Sharma','Delhi','9543198345','Service','1976-12-06');
INSERT INTO customer
VALUES('C00002','Avinash','Sunder','Minha','Delhi','9876532109','Service','1974-10-16');
INSERT INTO customer
VALUES('C00003','Rahul',null,'Rastogi','Delhi','9765178901','Student','1981-09-26');
INSERT INTO customer
VALUES('C00004','Parul',null,'Gandhi','Delhi','9876532109','Housewife','1976-11-03');
```

- Inserting records into the **branch** table:

```
INSERT INTO branch VALUES('B00001','Asaf ali road','Delhi');
INSERT INTO branch VALUES('B00002','New delhi main branch','Delhi');
INSERT INTO branch VALUES('B00003','Delhi cantt','Delhi');
```

INSERT INTO branch VALUES('B00004','Jasola','Delhi');

- Inserting records into the **account** table:

INSERT INTO account VALUES('A00001','C00001','B00001',1000,'2012-12-15','Saving','Active');
INSERT INTO account VALUES('A00002','C00002','B00001',1000,'2012-06-12','Saving','Active');
INSERT INTO account VALUES('A00003','C00003','B00002',1000,'2012-05-17','Saving','Active');
INSERT INTO account VALUES('A00004','C00002','B00005',1000,'2013-01-27','Saving','Active');

- Inserting records into the **Tran details** table:

INSERT INTO trandetails VALUES('T00001','A00001','2013-01-01','Cheque','Deposit',2000);
INSERT INTO trandetails VALUES('T00002','A00001','2013-02-01','Cash','Withdrawal',1000);
INSERT INTO trandetails VALUES('T00003','A00002','2013-01-01','Cash','Deposit',2000);
INSERT INTO trandetails VALUES('T00004','A00002','2013-02-01','Cash','Deposit',3000);
INSERT INTO trandetails VALUES('T00005','A00007','2013-01-11','Cash','Deposit',7000);

- Inserting records into the **Tran details** table:

INSERT INTO loan VALUES('C00001','B00001',100000);
INSERT INTO loan VALUES('C00002','B00002',200000);
INSERT INTO loan VALUES('C00009','B00008',400000);
INSERT INTO loan VALUES('C00010','B00009',500000)

NORMALIZATION OF DATABASE :

- In the Table customers the data is not arranged according to any order. therefore we use the sort function to arrange the records according to the date of birth allowing us to the easily identify the young and elderly customers of the bank
  Query:
- SELECT custid, fname, mname,dob
  FROM customer
  ORDER BY EXTRACT(year FROM dob), ASC;

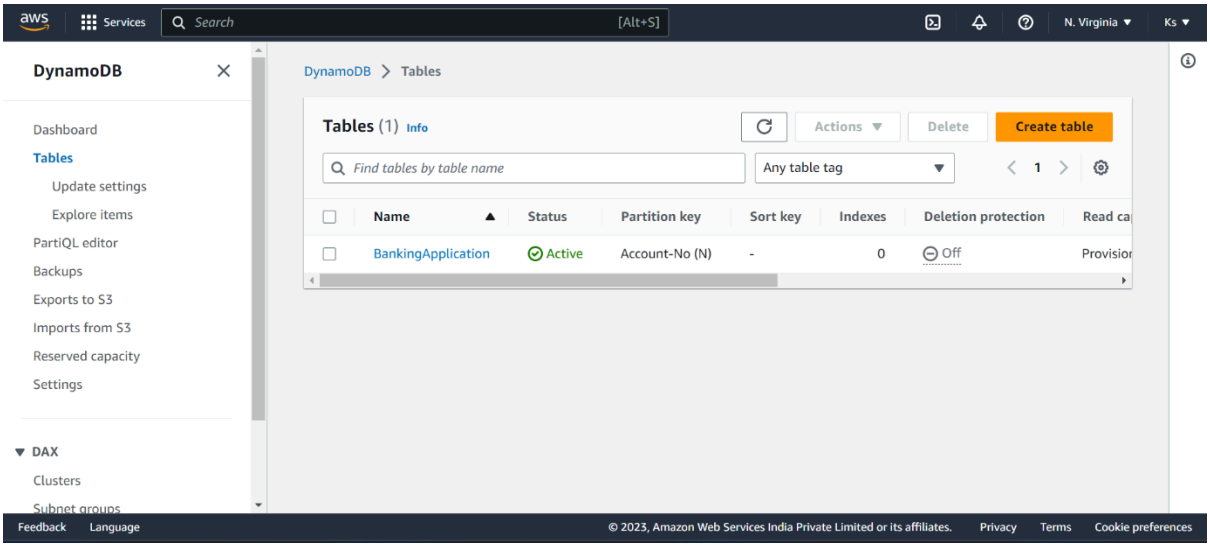
The above mentioned query displays the customer number, firstname, customer's date of birth and Displays in sorted order of date of birth year .

- For The customer's who don't have a middle name, for them we display the last name and Give the alias name as Cust_Name.
  Query:
- SELECT custid, fname, IF(mname IS NOT NULL, mname, ltname)
  AS Cust_Name

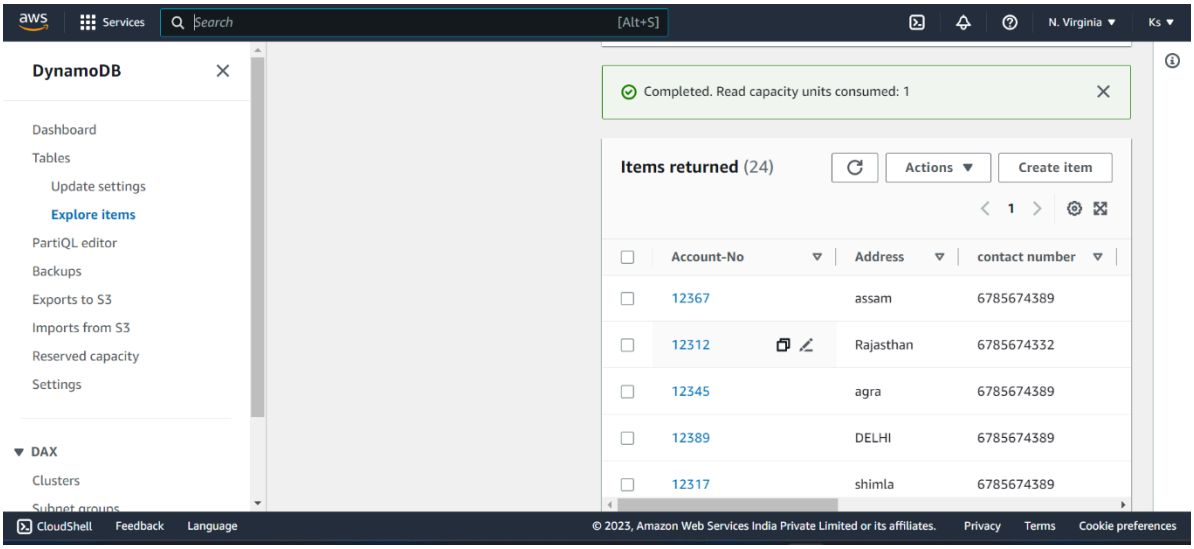
FROM customer;

IMPLEMENTATION USING DYNAMO-DB

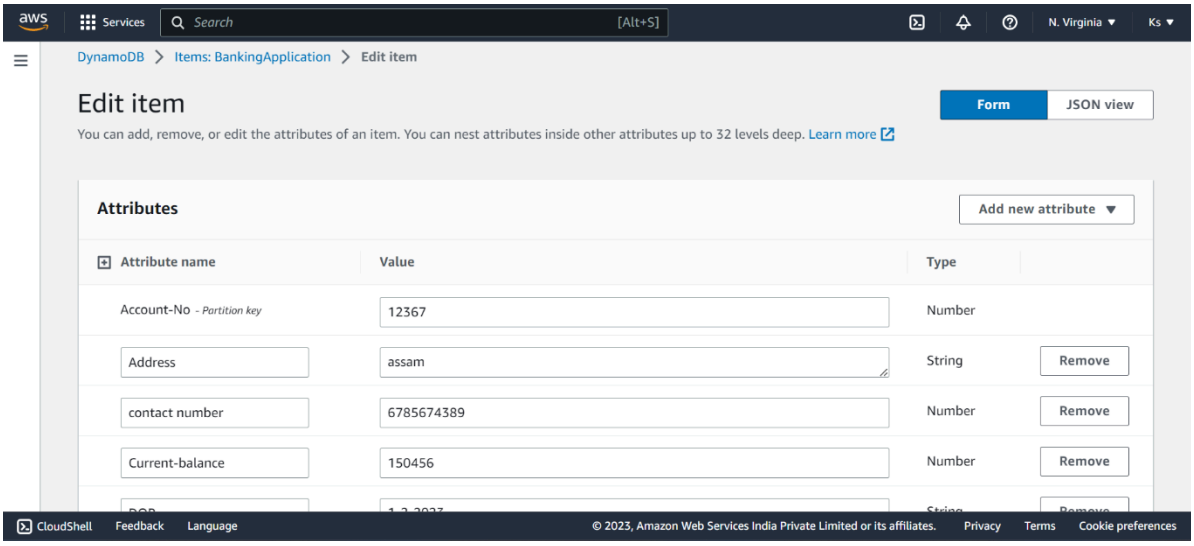
- Table created:

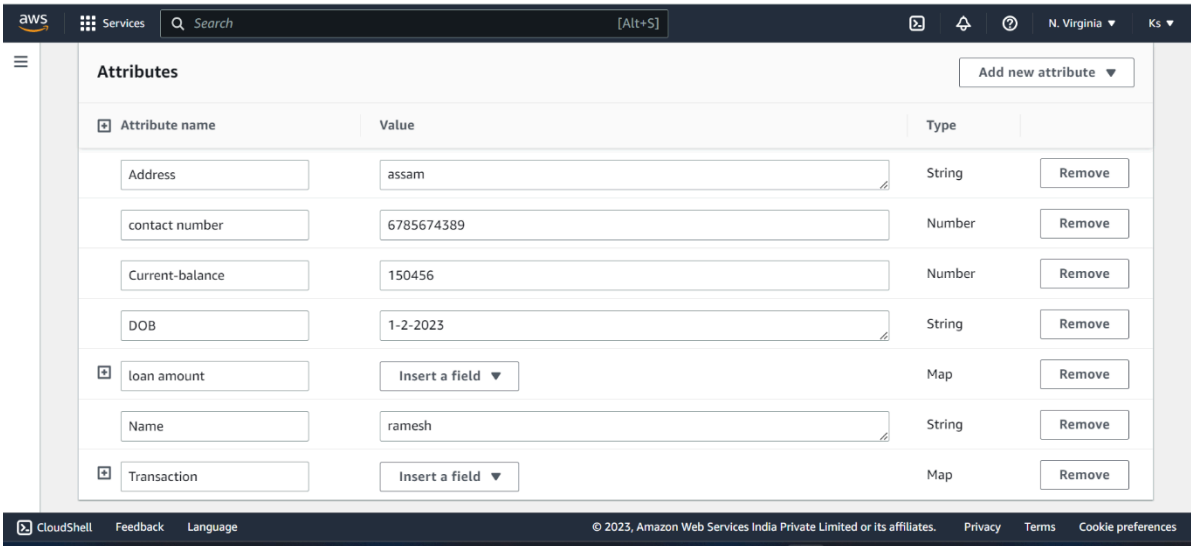


- Records entered:



- Attributes of the table:





CHAPTER-5

MODULES

- **ACCOUNT HOLDER:** As the name suggests, a record of customer details.
- **TRANSACTION:** Transactions to be made by the customer (credit amount, debit etc).
- **SERVICES:** Additional services that customers may want like (insurance, loan etc.).
- **BRANCH/EMPLOYEE:** Manager/Employee details of the concerned bank.

CHAPTER-6

APPLICATIONS

The bank management system caters to account holders by offering services like transaction management, account services, and access to branches or employees for assistance. It facilitates secure transactions, efficient account management, diverse banking services, and interaction with branches or employees for inquiries or assistance.

CHAPTER-7

CONCLUSION

In this project we created a database for banking system management. This application creates an environment that makes it easy to retrieve, update , and maintain the huge amount of data banks have to deal with.

This project resulted in the creation of database using MySql and amazon dynamodb

Overall we were successful in implementing a database that is a collection of logically coherent data, which come together to serve a particular purpose and they emulate some aspect of the real world.

CHAPTER-8

BIBLIOGRAPHY

It has been a matter of immense pleasure, honor and challenge to have this opportunity to take up this project and complete it successfully.

We have obtained information from various resources to design and implement our project. We have acquired most of the knowledge from the Internet.

The following are some of the resources:

- www.github.com
- www.tutorialspoint.com
- Youtube Tutorials.