

Alliance University

Alliance College of Engineering and Design

Assignment on Case Study

“Bank Database”

Database Management Systems: - Code No: CS504

V-Semester Section A – 2016 Batch

Under guidance of: Submitted by:

Prof.Bhoomika.A.P Abhishek .M

Reg.No: 16030141CSE006

Jay Dev Rai .M

Reg.No: 16030141CSE037

Kishan kumar Reddy

Reg.No: 16030141CSE043

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**CERTIFICATE**

This is to certify that the HOSPITAL DATABASE REPORT title STUDENT REGISTRATION was presented by **Abhishek.M** (Reg. No: 16030141CSE006) ,**Jay dev rai.M** (Reg. No: 16030141CSE037) &**Kishan Kumar Reddy** (Reg. No: 16030141CSE043) in partial fulfillment of the requirement for the Award of Degree of Bachelor of Technology in Computer Science and Engineering during the academic year 2018-2019 at Alliance University, Bangalore.

Faculty Signature:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**TABLE OF CONTENTS:**

1. **CHAPTER- INTRODUCTION**

1.1 INTRODUCTION TO DBMS

1.2 APPLICATION

1. **CHAPTER-DESIGN**

2.1 PROBLEM STATEMENT

2.2 REQUIREMENT ANALYSIS

2.3 ER DIAGRAM

2.4 DATABASE DESIGN

**3. CHAPTER-IMPLEMENTATION**

**4. CHAPTER-RESULT AND SNAPSHOTS**

**5.CHAPTER-CONCLUSION**

**6.CHAPTER-BIBLIOGRAPHY**

**INTRODUCTION**

**Database Management System (DBMS)**

A database management system (DBMS) is system software for creating and managing [databases](https://searchsqlserver.techtarget.com/definition/database). The DBMS provides users and programmers with a systematic way to create, retrieve, update and manage [data](https://searchdatamanagement.techtarget.com/definition/data).

A DBMS makes it possible for end users to create, read, update and delete [data](https://searchdatamanagement.techtarget.com/definition/data) in a database. The DBMS essentially serves as an interface between the [database](https://searchsqlserver.techtarget.com/definition/database) and end users or [application programs](https://searchsoftwarequality.techtarget.com/definition/application), ensuring that data is consistently organized and remains easily accessible.

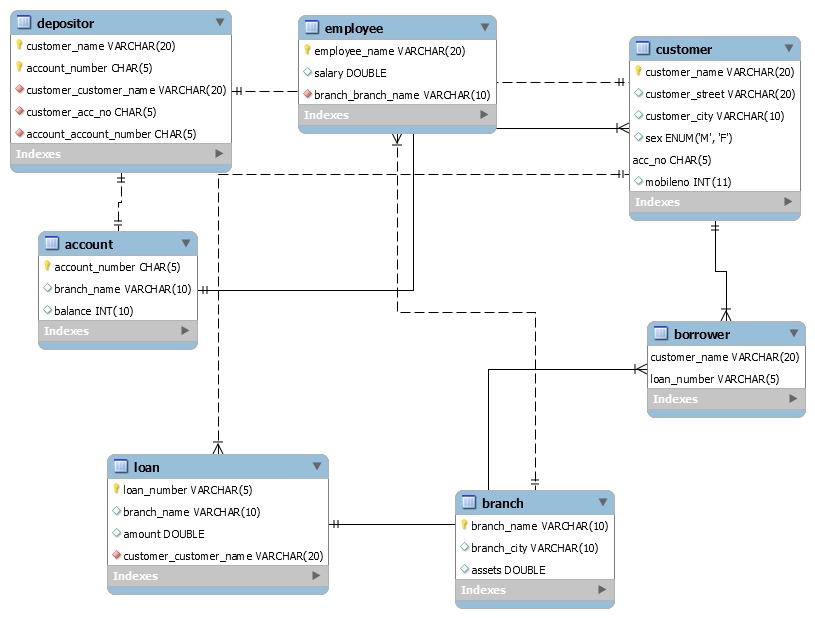
The DBMS can offer both logical and physical data independence. That means it can protect users and applications from needing to know where data is stored or having to be concerned about changes to the physical structure of data ([storage](https://searchstorage.techtarget.com/definition/storage) and hardware). As long as programs use the application programming interface ([API](https://searchmicroservices.techtarget.com/definition/application-program-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.

**Database Applications:**

* Data abstraction and independence
* Data security
* A locking mechanism for concurrent access
* An efficient handler to balance the needs of multiple applications using the same data
* The ability to swiftly recover from crashes and errors, including restartability and recoverability
* Robust data integrity capabilities
* Logging and auditing of activity
* Simple access using a standard application programming interface (API)
* Uniform administration procedures for data

**DESIGN**

**ER diagram**



CREATE SCHEMA IF NOT EXISTS `bank` DEFAULT CHARACTER SET utf8mb4 COLLATE utf8mb4\_0900\_ai\_ci ;

USE `bank` ;

-- -----------------------------------------------------

-- Table `bank`.`account`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `bank`.`account` (

`account\_number` CHAR(5) NOT NULL,

`branch\_name` VARCHAR(10) NULL DEFAULT NULL,

`balance` DOUBLE NULL DEFAULT NULL,

PRIMARY KEY (account\_number));

-- -----------------------------------------------------

-- Table `bank`.`customer`

-- -----------------------------------------------------

CREATE TABLE accountIF NOT EXISTS `bank`.`customer` (

`customer\_name` VARCHAR(20) NOT NULL,

`customer\_street` VARCHAR(20) NULL DEFAULT NULL,

`customer\_city` VARCHAR(10) NULL DEFAULT NULL,

`sex` ENUM('M', 'F') NULL,

`acc\_no` CHAR(5) NOT NULL,

PRIMARY KEY (`customer\_name`, `acc\_no`),

INDEX `acc\_no\_idx` (`acc\_no` ASC) VISIBLE,

CONSTRAINT `acc\_no`

FOREIGN KEY (`acc\_no`)

REFERENCES `bank`.`account` (`account\_number`)

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB

DEFAULT CHARACTER SET = utf8mb4

COLLATE = utf8mb4\_0900\_ai\_ci;

-- -----------------------------------------------------

-- Table `bank`.`loan`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `bank`.`loan` (

`loan\_number` VARCHAR(5) NOT NULL,

`branch\_name` VARCHAR(10) NULL DEFAULT NULL,

`amount` DOUBLE NULL DEFAULT NULL,

`customer\_customer\_name` VARCHAR(20) NOT NULL,

PRIMARY KEY (`loan\_number`),

INDEX `fk\_loan\_customer1\_idx` (`customer\_customer\_name` ASC) VISIBLE,

CONSTRAINT `fk\_loan\_customer1`

FOREIGN KEY (`customer\_customer\_name`)

REFERENCES `bank`.`customer` (`customer\_name`)

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB

DEFAULT CHARACTER SET = utf8mb4

COLLATE = utf8mb4\_0900\_ai\_ci;

-- -----------------------------------------------------

-- Table `bank`.`borrower`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `bank`.`borrower` (

`customer\_name` VARCHAR(20) NOT NULL,

`loan\_number` VARCHAR(5) NOT NULL,

PRIMARY KEY (`customer\_name`, `loan\_number`),

INDEX `b\_idx` (`loan\_number` ASC) VISIBLE,

CONSTRAINT `a`

FOREIGN KEY (`customer\_name`)

REFERENCES `bank`.`customer` (`customer\_name`)

ON DELETE NO ACTION

ON UPDATE NO ACTION,

CONSTRAINT `b`

FOREIGN KEY (`loan\_number`)

REFERENCES `bank`.`loan` (`loan\_number`)

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB

DEFAULT CHARACTER SET = utf8mb4

COLLATE = utf8mb4\_0900\_ai\_ci;

-- -----------------------------------------------------

-- Table `bank`.`branch`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `bank`.`branch` (

`branch\_name` VARCHAR(10) NOT NULL,

`branch\_city` VARCHAR(10) NULL DEFAULT NULL,

`assets` DOUBLE NULL DEFAULT NULL,

PRIMARY KEY (`branch\_name`))

ENGINE = InnoDB

DEFAULT CHARACTER SET = utf8mb4

COLLATE = utf8mb4\_0900\_ai\_ci;

-- -----------------------------------------------------

-- Table `bank`.`depositor`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `bank`.`depositor` (

`customer\_name` VARCHAR(20) NOT NULL,

`account\_number` CHAR(5) NOT NULL,

PRIMARY KEY (`customer\_name`, `account\_number`))

ENGINE = InnoDB

DEFAULT CHARACTER SET = utf8mb4

COLLATE = utf8mb4\_0900\_ai\_ci;

-- -----------------------------------------------------

-- Table `bank`.`employee`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `bank`.`employee` (

`employee\_name` VARCHAR(20) NOT NULL,

`branch\_name` VARCHAR(10) NOT NULL,

`salary` DOUBLE NULL DEFAULT NULL,

`branch\_branch\_name` VARCHAR(10) NOT NULL,

PRIMARY KEY (`employee\_name`, `branch\_name`),

INDEX `fk\_employee\_branch\_idx` (`branch\_branch\_name` ASC) VISIBLE,

CONSTRAINT `fk\_employee\_branch`

FOREIGN KEY (`branch\_branch\_name`)

REFERENCES `bank`.`branch` (`branch\_name`)

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB

DEFAULT CHARACTER SET = utf8mb4

COLLATE = utf8mb4\_0900\_ai\_ci;

**Code Implementation:**

package p1;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.util.ArrayList;

import java.util.Scanner;

public class a1 {

static int i=18;

public static void main(String[] args) throws Exception {

int deposit1,deposit2,deposit4,with1,with2,with4;

String deposit3,with3,get1,get2;

String name,street,gender,accno2;

int mob;

Scanner sc=new Scanner(System.in);

while(true) {

System.out.println("Enter your choice\n1.create account\n2.deposit\n3.withdraw\n4.customer details\n5.loan details\n6.exit\n");

int choice;

choice=sc.nextInt();

switch(choice) {

case 1:{

System.out.println("enter name : \n");

name=sc.next();

System.out.println("enter street : \n");

street=sc.next();

System.out.println("enter gender(M/F) : \n");

gender=sc.next();

System.out.println("enter mobile number : \n");

mob=sc.nextInt();

System.out.println("enter account number : \n");

accno2=sc.next();

createaccount(name,street,gender,mob,accno2);

}break;

case 2:{

System.out.println("enter account number : ");

deposit3=sc.next();

System.out.println("enter amount to deposit : ");

deposit1=sc.nextInt();

deposit2=getbalance(deposit3).get(0);

deposit4=deposit2+deposit1;

putbalance(deposit3,deposit4);

System.out.println("deposit Completed.\n");

}break;

case 3:{

System.out.println("enter account number : ");

with3=sc.next();

System.out.println("enter amount to withdraw : ");

with1=sc.nextInt();

with2=getbalance(with3).get(0);

if(with2>=with1) {

with4=with2-with1;

putbalance(with3,with4);

System.out.println("withdraw Completed.\n");}

else {System.out.println("\nInsufficient balance\n");}

}break;

case 4:{

System.out.println("enter account number : ");

get1=sc.next();

getdetails(get1);

}break;

case 5:{

System.out.println("enter loan number : ");

get2=sc.next();

loandetails(get2);

}break;

default:System.exit(0);sc.close();break;

}

}

}

public static void putbalance(String accno,int newbal) throws Exception{

try{

Connection con = getconnection();

PreparedStatement updatebalance = con.prepareStatement("update account set balance ="+newbal+" where account\_number = '"+accno+"'; ");

updatebalance.executeUpdate();

} catch(Exception e){System.out.println(e);}

finally {

//System.out.println("deposit Completed.\n");

}

}

public static int accountnumber() {

i++;

return i;

}

public static void createaccount(String name,String street,String gender,int mob,String accno2) throws Exception{

try{

Connection con = getconnection();

PreparedStatement insertaccount = con.prepareStatement("insert into account(account\_number,branch\_name,balance) values('"+accno2+"','anekal','100');");

PreparedStatement insertcustomer = con.prepareStatement("insert into customer(customer\_name,customer\_street,customer\_city,sex,acc\_no,mobileno) values('"+name+"','"+street+"','bangalore','"+gender+"','"+accno2+"','"+mob+"');");

insertaccount.executeUpdate();

insertcustomer.executeUpdate();

} catch(Exception e){System.out.println(e);}

finally {

System.out.println(" account created.");

}

}

public static ArrayList<Integer> getbalance(String accno1) {

try{

Connection con = getconnection();

PreparedStatement statement = con.prepareStatement("SELECT balance FROM account where account\_number = '"+accno1+"'");

ResultSet result = statement.executeQuery();

ArrayList<Integer> array = new ArrayList<Integer>();

while(result.next()){

array.add(result.getInt("balance"));

}

return array;

}catch(Exception e){System.out.println(e);}

return null;

}

public static void getdetails(String accno1) {

try{

Connection con = getconnection();

PreparedStatement statement = con.prepareStatement("SELECT \* FROM customer where acc\_no = '"+accno1+"'");

ResultSet result = statement.executeQuery();

while(result.next()){

System.out.println("name : "+result.getString("customer\_name"));

System.out.println("street : "+result.getString("customer\_street"));

System.out.println("city : "+result.getString("customer\_city"));

System.out.println("gender : "+result.getString("sex"));

System.out.println("account number : "+result.getString("acc\_no"));

System.out.println("mobile number : "+result.getString("mobileno"));

}

}catch(Exception e){System.out.println(e);}

}

public static void loandetails(String accno1) {

try{

Connection con = getconnection();

PreparedStatement statement = con.prepareStatement("SELECT \* FROM loan where loan\_number = '"+accno1+"';");

ResultSet result = statement.executeQuery();

while(result.next()){

System.out.println("loan number : "+result.getString("loan\_number"));

System.out.println("branch name : "+result.getString("branch\_name"));

System.out.println("amount : "+result.getString("amount"));

System.out.println("customer name : "+result.getString("customer\_customer\_name"));

}

}catch(Exception e){System.out.println(e);}

}

public static Connection getconnection() throws Exception{

try {

String driver="com.mysql.jdbc.Driver";

String url="jdbc:mysql://localhost/bank";

String user="root";

String pass="jaydev@1";

Class.forName(driver);

Connection conn= DriverManager.getConnection(url,user,pass);

//System.out.println("connected");

return conn;

}catch(Exception e) {System.out.println(e);}

return null;

}

}

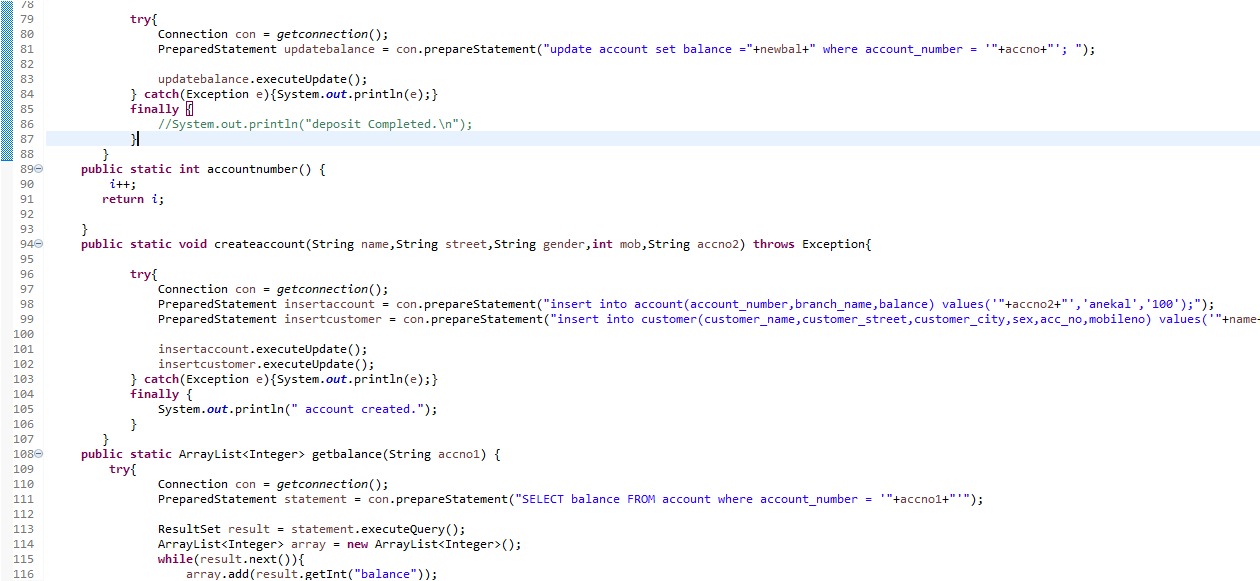
**IMPLEMENTATION**

**Connection**

**Inserting to the Database**

****

**Updating database**

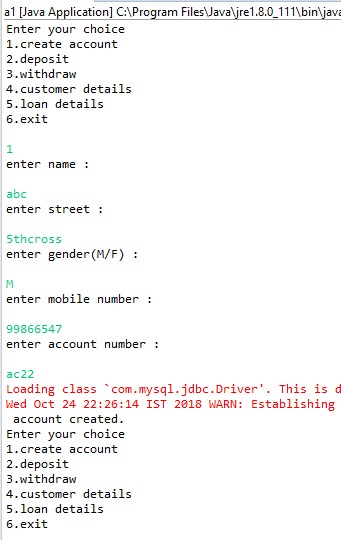
****

**Deleting from database**

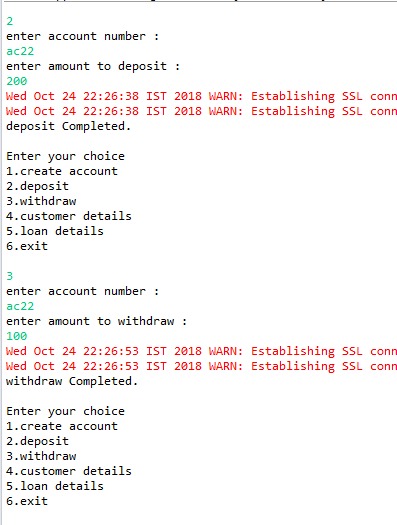
****

**RESULT AND SNAPSHOTS**

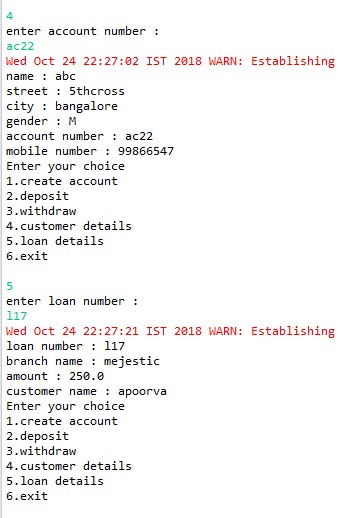
**Creating and Inserting to the Database**

****

**Deposit And Withdraw**

****

**Customer details and Loan Details**

****