Experiment No.___ *Date___/___/2022*

TITLE OF EXPERIMENT: - A JAVA Program to database application that uses any JDBC driver.

DIVISION:_____ BRANCH: _____

BATCH:_____ ROLL NO.: _____

PERFORMED ON DATE: _____

SIGNATURE OF TEACHING STAFF:

EXPERIMENT NO. 7

Aim: Write a database application that uses any JDBC driver.

Software:

1.	Command prompt
2.	JDK 8
3.	Eclipse neon3
4.	Mysql installer 8.0.28
5.	Mysql connector jar file

Theory:

What is JDBC Driver?

JDBC drivers implement the defined interfaces in the JDBC API, for interacting with your database server.

For example, using JDBC drivers enable you to open database connections and to interact with it by sending SQL or database commands then receiving results with Java.

The *Java.sql* package that ships with JDK, contains various classes with their behaviours defined and their actual implementaions are done in third-party drivers. Third party vendors implements the *java.sql.Driver* interface in their database driver.

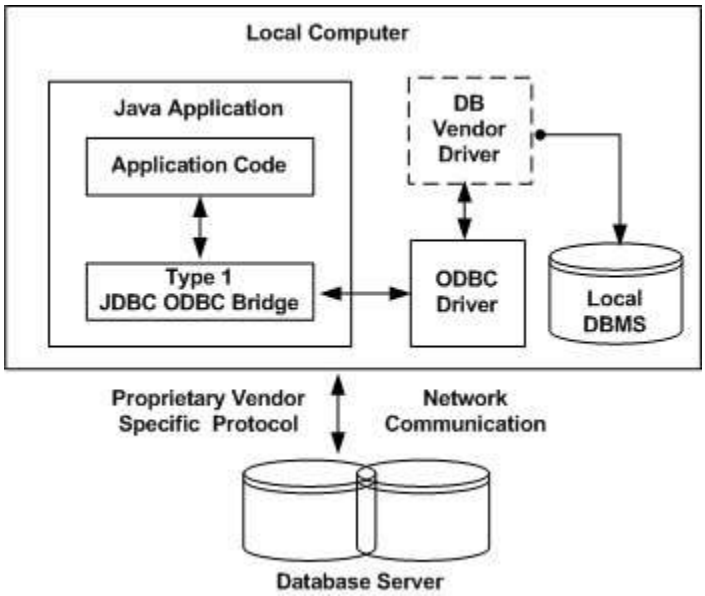
JDBC Drivers Types

JDBC driver implementations vary because of the wide variety of operating systems and hardware platforms in which Java operates. Sun has divided the implementation types into four categories, Types 1, 2, 3, and 4, which is explained below −

Type 1 − JDBC-ODBC Bridge Driver

In a Type 1 driver, a JDBC bridge is used to access ODBC drivers installed on each client machine. Using ODBC, requires configuring on your system a Data Source Name (DSN) that represents the target database.

When Java first came out, this was a useful driver because most databases only supported ODBC access but now this type of driver is recommended only for experimental use or when no other alternative is available.

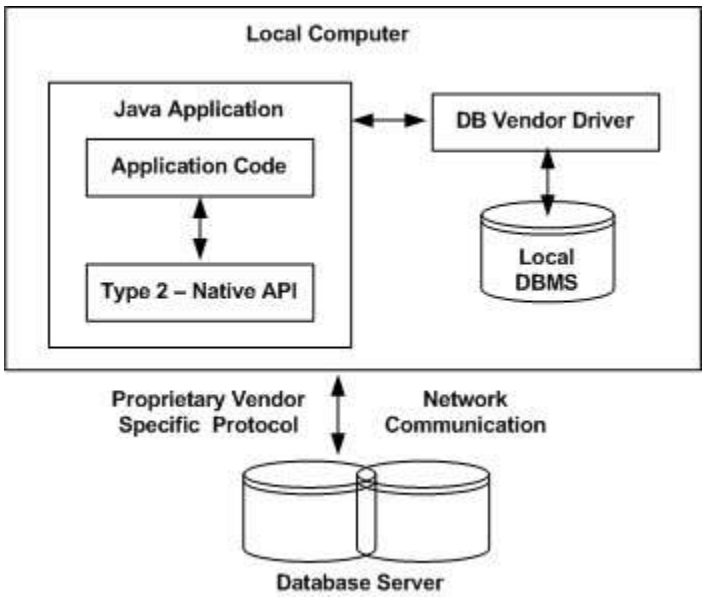


The JDBC-ODBC Bridge that comes with JDK 1.2 is a good example of this kind of driver.

Type 2 − JDBC-Native API

In a Type 2 driver, JDBC API calls are converted into native C/C++ API calls, which are unique to the database. These drivers are typically provided by the database vendors and used in the same manner as the JDBC-ODBC Bridge. The vendor-specific driver must be installed on each client machine.

If we change the Database, we have to change the native API, as it is specific to a database and they are mostly obsolete now, but you may realize some speed increase with a Type 2 driver, because it eliminates ODBC's overhead.

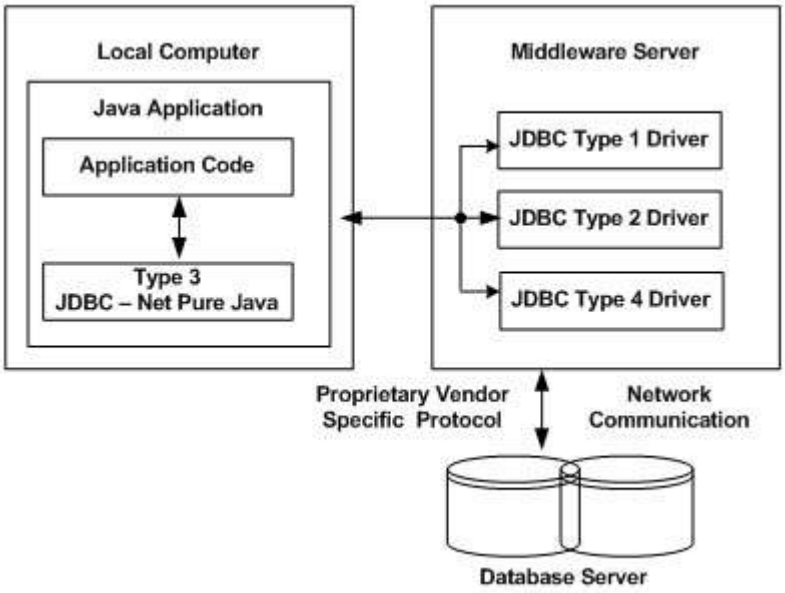


The Oracle Call Interface (OCI) driver is an example of a Type 2 driver.

Type 3 − JDBC-Net pure Java

In a Type 3 driver, a three-tier approach is used to access databases. The JDBC clients use standard network sockets to communicate with a middleware application server. The socket information is then translated by the middleware application server into the call format required by the DBMS, and forwarded to the database server.

This kind of driver is extremely flexible, since it requires no code installed on the client and a single driver can actually provide access to multiple databases.



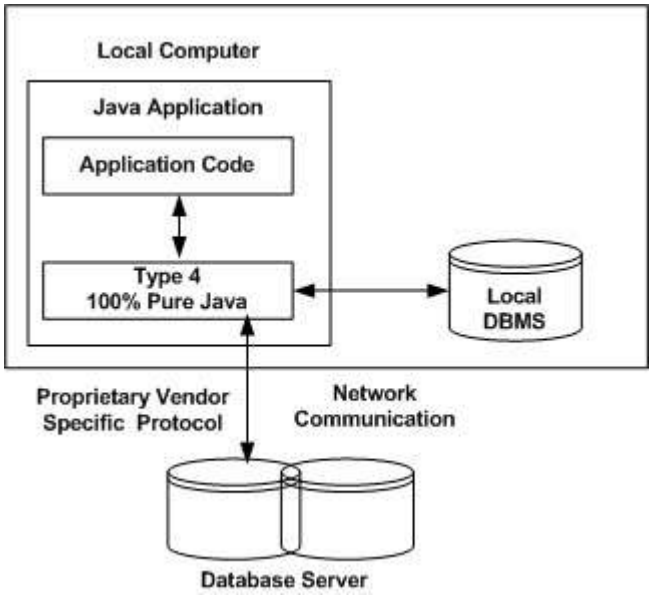
You can think of the application server as a JDBC "proxy," meaning that it makes calls for the client application. As a result, you need some knowledge of the application server's configuration in order to effectively use this driver type.

Your application server might use a Type 1, 2, or 4 driver to communicate with the database, understanding the nuances will prove helpful.

Type 4 − 100% Pure Java

In a Type 4 driver, a pure Java-based driver communicates directly with the vendor's database through socket connection. This is the highest performance driver available for the database and is usually provided by the vendor itself.

This kind of driver is extremely flexible, you don't need to install special software on the client or server. Further, these drivers can be downloaded dynamically.



MySQL's Connector/J driver is a Type 4 driver. Because of the proprietary nature of their network protocols, database vendors usually supply type 4 drivers.

Which Driver should be Used?

If you are accessing one type of database, such as Oracle, Sybase, or IBM, the preferred driver type is 4.

If your Java application is accessing multiple types of databases at the same time, type 3 is the preferred driver.

Type 2 drivers are useful in situations, where a type 3 or type 4 driver is not available yet for your database.

The type 1 driver is not considered a deployment-level driver, and is typically used for development and testing purposes only.

Execution with example:

- To execute the database application you can replace the *username* and *password* with your actual user name and password.
- Your MySQL or whatever database you are using is up and running.

Required Steps

The following steps are required to create a new Database using JDBC application −

- **Import the packages** − Requires that you include the packages containing the JDBC classes needed for database programming. Most often, using *import java.sql.** will suffice.
- **Register the JDBC driver** − Requires that you initialize a driver so you can open a communications channel with the database.

- **Open a connection** − Requires using the *DriverManager.getConnection()* method to create a Connection object, which represents a physical connection with a database server.
- **Execute a query** − Requires using an object of type Statement for building and submitting an SQL statement to insert records into a table.
- **Clean up the environment** try with resources automatically closes the resources.

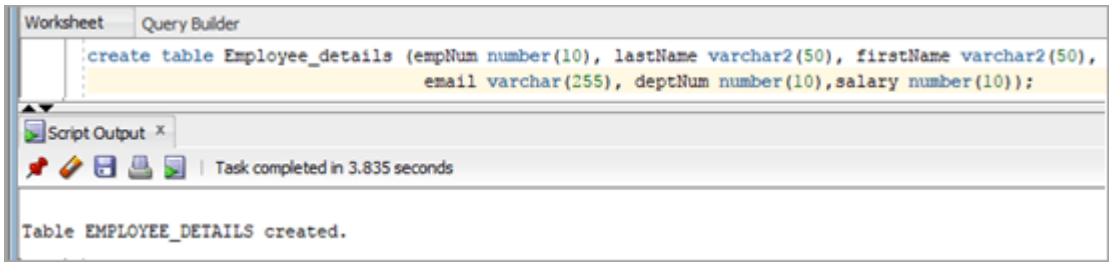
Create Table

Before that, first, create one table and add some entries into it.

Below is the SQL query to create a table.

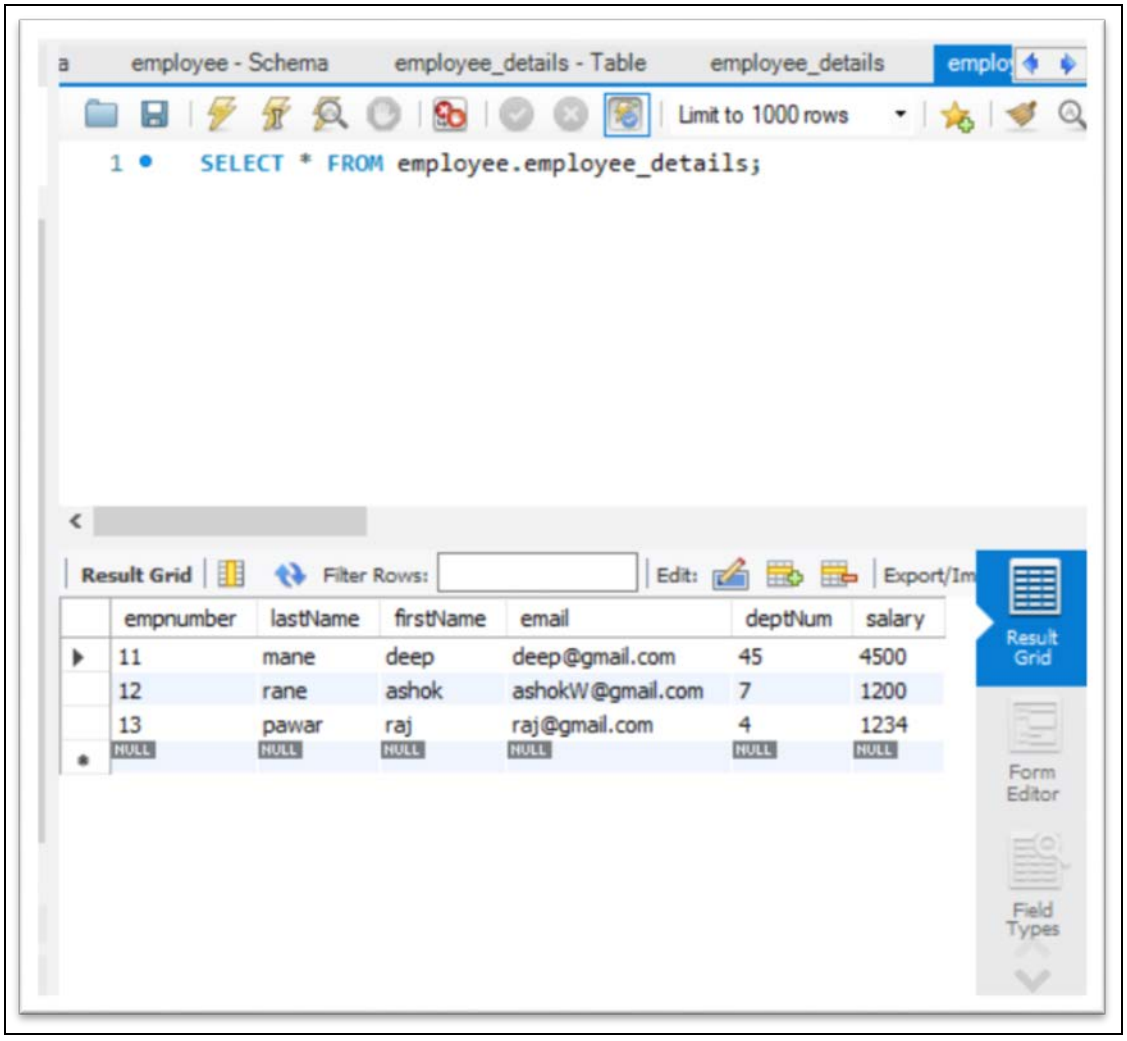
```
create table employee_details (empNum number(10), lastName varchar(50),
 firstName varchar(50), email varchar(255) , deptNum number(10), salary
number(10));
```

Created the “employee_details” table in Mysql DB.



Insert Data Into Table

Using the following queries, insert the data into the “employee_details” table.



Java Program

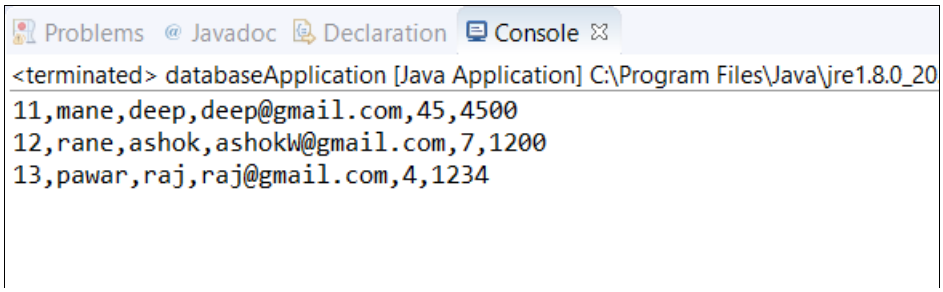
Download the JDBC jar file and import it into the Java project.

a) Java Database Connectivity with MySQL

```
// A database application that uses any JDBC driver.

// import sql package to use it in our program
import java.sql.*;

public class databaseApplication {

public static void main(String[] args) throws ClassNotFoundException, SQLException
{
    // store the SQL statement in a string
    String QUERY = "select * from employee_details";
    //register the oracle driver with DriverManager
    Class.forName("com.mysql.cj.jdbc.Driver");
    //Here we have used Java 8 so opening the connection in try statement
```

```
    try(Connection conn =
DriverManager.getConnection("jdbc:mysql://localhost:3306/employee","root","1234"))
    {
            Statement statemnt1 = conn.createStatement();
            //Created statement and execute it
            ResultSet rs1 = statemnt1.executeQuery(QUERY);
            {
                //Get the values of the record using while loop
                while(rs1.next())
                {
                    int empNumber = rs1.getInt("empNumber");
                    String lastName = rs1.getString("lastName");
                    String firstName = rs1.getString("firstName");
                    String email = rs1.getString("email");
                    String deptNum = rs1.getString("deptNum");
                    String salary = rs1.getString("salary");
                    //store the values which are retrieved using ResultSet and
print it
                System.out.println(empNumber + "," +lastName+ "," +firstName+ ","
+email +","+deptNum +"," +salary);
                }
            }
    }
    catch (SQLException e) {
        //If exception occurs catch it and exit the program
        e.printStackTrace();
    }
   }
}
```

Output:

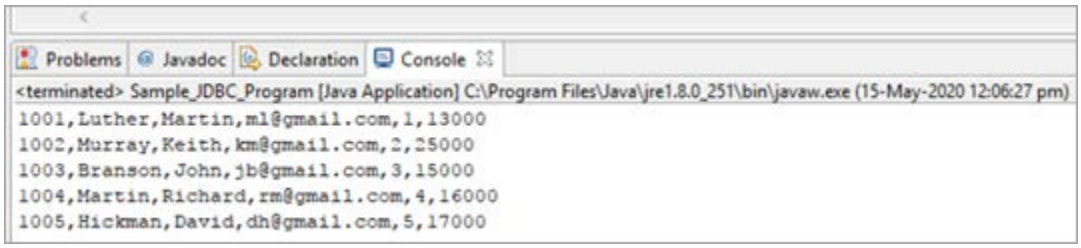


b) Java Database Connectivity with Oracle

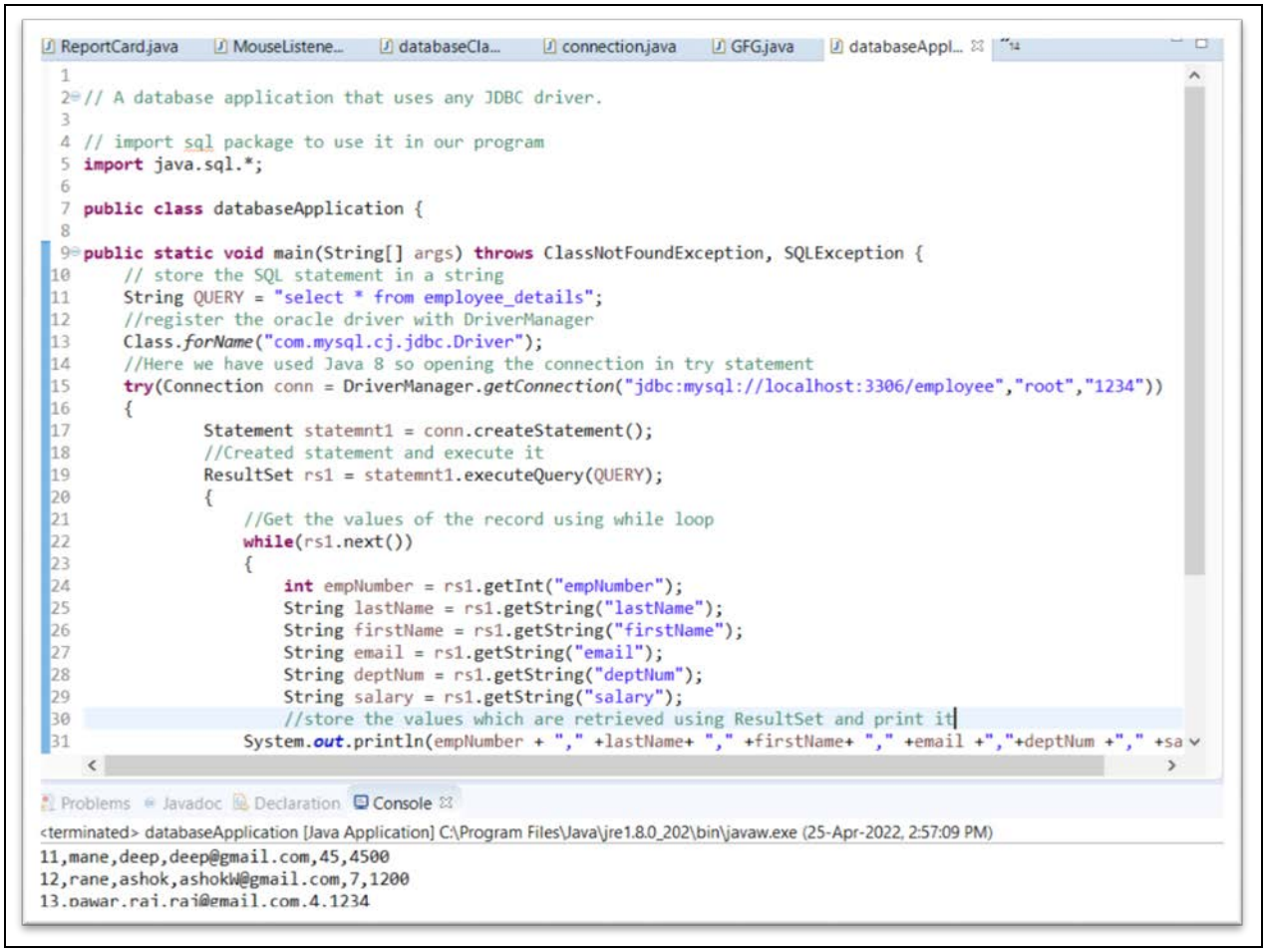
```
package com.STH.JDBC;
    // import sql package to use it in our program
    import java.sql.*;

    public class Sample_JDBC_Program {

    public static void main(String[] args) throws ClassNotFoundException, SQLException {
        // store the SQL statement in a string
        String QUERY = "select * from employee_details";
        //register the oracle driver with DriverManager
```

```
        Class.forName("oracle.jdbc.driver.OracleDriver");
        //Here we have used Java 8 so opening the connection in try statement
        try(Connection conn =
DriverManager.getConnection("jdbc:oracle:thin:system/pass123@localhost:1521:XE"))
        {
                Statement statemnt1 = conn.createStatement();
                //Created statement and execute it
                ResultSet rs1 = statemnt1.executeQuery(QUERY);
                {
                    //Get the values of the record using while loop
                    while(rs1.next())
                    {
                        int empNum = rs1.getInt("empNum");
                        String lastName = rs1.getString("lastName");
                        String firstName = rs1.getString("firstName");
                        String email = rs1.getString("email");
                        String deptNum = rs1.getString("deptNum");
                        String salary = rs1.getString("salary");
                        //store the values which are retrieved using ResultSet and
print it
                    System.out.println(empNum + "," +lastName+ "," +firstName+ ","
+email +","+deptNum +"," +salary);
                    }
                }
        }
        catch (SQLException e) {
            //If exception occurs catch it and exit the program
            e.printStackTrace();
        }
       }
    }
```

Output:



Screenshot's of Output:


ReportCard.java
MouseListene...
databaseCla...
connection.java
GFG.java
databaseAppl...
// A database application that uses any JDBC driver.
// import sql package to use it in our program
import java.sql.*;
public class databaseApplication {
public static void main(String[] args) throws ClassNotFoundException, SQLException {
// store the SQL statement in a string
String QUERY = "select * from employee_details";
//register the oracle driver with DriverManager
Class.forName("com.mysql.cj.jdbc.Driver");
//Here we have used Java 8 so opening the connection in try statement
try(Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/employee","root","1234"))
{
Statement statemnt1 = conn.createStatement();
//Created statement and execute it
ResultSet rs1 = statemnt1.executeQuery(QUERY);
{
//Get the values of the record using while loop
while(rs1.next())
{
int empNumber = rs1.getInt("empNumber");
String lastName = rs1.getString("lastName");
String firstName = rs1.getString("firstName");
String email = rs1.getString("email");
String deptNum = rs1.getString("deptNum");
String salary = rs1.getString("salary");
//store the values which are retrieved using ResultSet and print it
System.out.println(empNumber + "," +lastName+ "," +firstName+ "," +email +","+deptNum +"," +sa
Problems
Javadoc
Declaration
Console
<terminated> databaseApplication [Java Application] C:\Program Files\Java\jre1.8.0_202\bin\javaw.exe (25-Apr-2022, 2:57:09 PM)
11,mane,deep,deep@gmail.com,45,4500
12,rane,ashok,ashokW@gmail.com,7,1200
13,pawar,raj,raj@email.com,4,1234

Conclusion: