Nagar Yuwak Shikshan Sanstha's
Yeshwantrao Chavan College of Engineering
(An Autonomous Institution affiliated to Rashtrasant Tukadoji Maharaj Nagpur University)
Hingna Road, Wanadongri, Nagpur - 441 110
NAAC A++
Ph.: 07104-237919, 234623, 329249, 329250 Fax: 07104-232376, Website: www.ycce.edu

Department of Computer Technology

Vision of the Department

To be a well-known centre for pursuing computer education through innovative pedagogy, value-based education and industry collaboration.

Mission of the Department

To establish learning ambience for ushering in computer engineering professionals in core and multidisciplinary area by developing Problem-solving skills through emerging technologies.

Session 2025-2026

Vision: Dream of where you want.	**Mission:** Means to achieve Vision

Program Educational Objectives of the program (PEO): (broad statements that describe the professional and career accomplishments)

PEO1	**Preparation**	**P: Preparation**	**Pep-CL abbreviation pronounce as Pep-si-lL easy to recall**
PEO2	**Core Competence**	**E: Environment (Learning Environment)**	
PEO3	**Breadth**	**P: Professionalism**	
PEO4	**Professionalism**	**C: Core Competence**	
PEO5	**Learning Environment**	**L: Breadth (Learning in diverse areas)**	

Program Outcomes (PO): (statements that describe what a student should be able to do and know by the end of a program)

Keywords of POs:
Engineering knowledge, Problem analysis, Design/development of solutions, Conduct Investigations of Complex Problems, Engineering Tool Usage, The Engineer and The World, Ethics, Individual and Collaborative Team work, Communication, Project Management and Finance, Life-Long Learning

PSO Keywords: Cutting edge technologies, Research

"I am an engineer, and I know how to apply engineering knowledge to investigate, analyse and design solutions to complex problems using tools for entire world following all ethics in a collaborative way with proper management skills throughout my life." *to contribute to the development of cutting-edge technologies and Research*.

Integrity: I will adhere to the Laboratory Code of Conduct and ethics in its entirety.

Name and Signature of Student and Date
(Signature and Date in Handwritten)
Nikhil Gourkar

Nagar Yuwak Shikshan Sanstha's
Yeshwantrao Chavan College of Engineering
(An Autonomous Institution affiliated to Rashtrasant Tukadoji Maharaj Nagpur University)
Hingna Road, Wanadongri, Nagpur - 441 110
NAAC A++
Ph.: 07104-237919, 234623, 329249, 329250 Fax: 07104-232376, Website: www.ycce.edu
Department of Computer Technology

Vision of the Department

To be a well-known centre for pursuing computer education through innovative pedagogy, value-based education and industry collaboration.

Mission of the Department

To establish learning ambience for ushering in computer engineering professionals in core and multidisciplinary area by developing Problem-solving skills through emerging technologies.

Session	**2024-25 (ODD)**	**Course Name**	**Lab : Java Stack**
Semester	**5**	**Course Code**	**CT**
Roll No	**59**	**Name of Student**	**Nikhil Gourkar**

Practical Number	**5**
Course Outcome	Proper Understanding of Basic Java programs covering loops , arrays and conditionals and implementation of programs.
Aim	Practical based on JDBC.
Problem Definition	Problem Definition: Create an application which creates a table in database for the file given by the user. Transfer complete data from file into the table. Display the metadata of the table like, No. of Columns, list of columns, No. of rows.
Theory (100 words)	JDBC stands for Java Database Connectivity. It is a standard Java API (Application Programming Interface) that defines how a Java application can interact with a database. Think of it as a bridge or a standard plug socket. • Your Java application is the "appliance" (like a lamp). • The database (like MySQL, PostgreSQL, Oracle) is the "power station." • The JDBC API is the standard "socket" design that all appliances use. • The JDBC Driver is the specific "adapter" or "plug" that connects your standard appliance to that specific power station. This design means your Java code doesn't need to know the specific details of how to talk to MySQL versus Oracle. It just "speaks" standard JDBC, and the driver handles the translation.
Procedure and Execution (100 Words)	Algorithm: Start. Connect: Establish a connection to the SQLite database file (jdbc_practical_5.db). Clean Slate: Drop the candidates table if it already exists. Create Table: Execute a CREATE TABLE command to make a new, empty candidates table with the 18 columns (using TEXT and REAL data types). Read File: Open the CSV file (Source Data_Pract 5 (1).xlsx - Sheet1.csv) with a BufferedReader. Skip Header: Read and discard the first line (the column headers). Start Transaction: Disable auto-commit on the database

connection to prepare for a fast batch insert.
Loop Through Data:

- Read one line at a time from the CSV file until the end.
- Split the line into an array of 18 string values.
- For each value, check its corresponding column:
  - If the column is numeric (REAL), try to parse the string as a Double. If it fails (e.g., the value is "N/A" or empty), set the value to NULL.
  - If the column is text (TEXT), set the value as a string (or NULL if empty).
- Add this complete INSERT statement to a batch.

Execute Batch: Send the entire batch of INSERT statements to the database at once.
Commit: Commit the transaction, saving all 3,796 rows to the database.
Get Metadata:

- Query the DatabaseMetaData to get and print the list of column names and the total column count (18).
- Execute a SELECT COUNT(*) FROM candidates query.

Display Results: Print the result of the COUNT(*) query (3796).
End.

Code:

```
1) import java.io.BufferedReader;
import java.io.FileReader;
import java.io.IOException;
import java.sql.Connection;
import java.sql.DatabaseMetaData;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.Set;
import java.util.HashSet;

public class CsvToDatabase {

    private static final String DB_NAME = "jdbc_practical_5.db";
    private static final String TABLE_NAME = "candidates";
    private static final String CSV_FILE_NAME = "Source Data_Pract 5
(1).xlsx - Sheet1.csv";
    private static final String CONN_STRING = "jdbc:sqlite:" +
DB_NAME;

    public static void main(String[] args) {
        try (Connection conn =
DriverManager.getConnection(CONN_STRING)) {
            if (conn != null) {
                System.out.println("Successfully connected to SQLite
database: " + DB_NAME);

                createTable(conn);
                transferData(conn);
                displayMetadata(conn);
```

```
            }
        } catch (SQLException e) {
            System.err.println("Database Connection Error: " +
e.getMessage());
            e.printStackTrace();
        }
    }

    private static void createTable(Connection conn) throws
SQLException {
        String sql = "CREATE TABLE IF NOT EXISTS " +
TABLE_NAME + " (\n"
                + "    Candidate_Id TEXT,\n"
                + "    Employee_Id TEXT,\n"
                + "    First_Name TEXT,\n"
                + "    Middle_Name TEXT,\n"
                + "    Last_Name TEXT,\n"
                + "    DegreePercentage REAL,\n"
                + "    \"12th_or_DiplomaPercentage\"  REAL,\n"
                + "    SSCPercentage REAL,\n"
                + "    University_Name TEXT,\n"
                + "    Native_State TEXT,\n"
                + "    \"BackgroundCS_NCS\" TEXT,\n"
                + "    Industrial_Training TEXT,\n"
                + "    TrainingExam1Percentage  REAL,\n"
                + "    Exam1Grade TEXT,\n"
                + "    TrainingExam2Percentage  REAL,\n"
                + "    Exam2Grade TEXT,\n"
                + "    IntoProduction REAL,\n"
                + "    Channel TEXT\n"
                + ");";

        String dropSql = "DROP TABLE IF EXISTS " + TABLE_NAME +
";";

        try (Statement stmt = conn.createStatement()) {
            stmt.execute(dropSql);
            stmt.execute(sql);
            System.out.println("Table '" + TABLE_NAME + "' created
successfully.");
        }
    }

    private static void transferData(Connection conn) throws
SQLException {
        String sql = "INSERT INTO " + TABLE_NAME + " VALUES
(?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?)";
        int rowCount = 0;
        String line;

        Set<Integer> realColumns = new HashSet<>();
        realColumns.add(5);
        realColumns.add(6);
        realColumns.add(7);
        realColumns.add(12);
        realColumns.add(14);
        realColumns.add(16);
```

```
        try (BufferedReader br = new BufferedReader(new
FileReader(CSV_FILE_NAME))) {

            System.out.println("File found. Transferring data from " +
CSV_FILE_NAME + "...");

            try (PreparedStatement pstmt = conn.prepareStatement(sql)) {

                br.readLine();
                conn.setAutoCommit(false);

                while ((line = br.readLine()) != null) {
                    String[] values = line.split(",", -1);

                    if (values.length == 18) {
                        for (int i = 0; i < values.length; i++) {
                            int paramIndex = i + 1;
                            String value = values[i];

                            if (value == null || value.isEmpty()) {
                                pstmt.setNull(paramIndex, java.sql.Types.NULL);

                            } else if (realColumns.contains(i)) {
                                try {
                                    double numValue = Double.parseDouble(value);
                                    pstmt.setDouble(paramIndex, numValue);
                                } catch (NumberFormatException e) {
                                    pstmt.setNull(paramIndex, java.sql.Types.NULL);
                                }
                            } else {
                                pstmt.setString(paramIndex,  value);
                            }
                        }
                        pstmt.addBatch();
                        rowCount++;
                    }
                }

                pstmt.executeBatch();
                conn.commit();

                System.out.println("Successfully transferred " + rowCount + "
rows.");

            } catch (SQLException e) {
                System.err.println("Database Error during transfer: " +
e.getMessage());
                conn.rollback();
                e.printStackTrace();
            } finally {
                conn.setAutoCommit(true);
            }

        } catch (java.io.FileNotFoundException e) {
            System.err.println("**********************************
****************************");
            System.err.println(">>> ERROR: FILE NOT FOUND");
            System.err.println(">>> Java could not find the file: " +
```

```
CSV_FILE_NAME);
            System.err.println(">>> " + new
java.io.File("").getAbsolutePath());
            System.err.println("**********************************
**************************");
        } catch (IOException e) {
            System.err.println("File I/O Error: " + e.getMessage());
        }
    }

    private static void displayMetadata(Connection conn) throws
SQLException {
        System.out.println("\n--- Table Metadata ---");

        DatabaseMetaData metaData = conn.getMetaData();

        try (ResultSet columns = metaData.getColumns(null, null,
TABLE_NAME, null)) {
            int columnCount = 0;
            System.out.println("List of Columns:");
            while (columns.next()) {
                String columnName =
columns.getString("COLUMN_NAME");
                System.out.println("  - " + columnName);
                columnCount++;
            }
            System.out.println("\nNumber of Columns: " + columnCount);
        }

        try (Statement stmt = conn.createStatement();
             ResultSet rs = stmt
```

Output:

```
--- Table Metadata ---
List of Columns:
  - Candidate_Id
  - Employee_Id
  - First_Name
  - Middle_Name
  - Last_Name
  - DegreePercentage
  - 12th_or_DiplomaPercentage
  - SSCPercentage
  - University_Name
  - Native_State
  - BackgroundCS_NCS
  - Industrial_Training
  - TrainingExam1Percentage
  - Exam1Grade
  - TrainingExam2Percentage
  - Exam2Grade
  - IntoProduction
  - Channel

Number of Columns: 18
Number of Rows: 3796
```

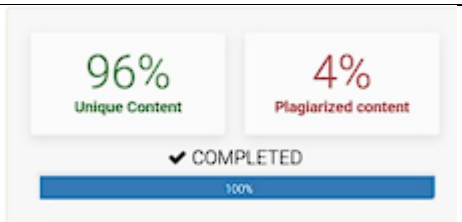
Output Analysis	Successfully connected to ... This confirms the Java code found the JDBC driver and was able to create the jdbc_practical_5.db file. Table 'candidates' created ... This shows the SQL CREATE TABLE command worked. The candidates table now exists with the correct 18 columns. File found. Transferring data ... This proves the program found your Source Data_Pract 5 (1).xlsx - Sheet1.csv file and started reading it. Successfully transferred 3796 rows. This is the most important line. It shows the code read all 3,796 data rows from the CSV, correctly parsed them (handling numbers and text), and successfully inserted all of them into the database table. Number of Columns: 18 and Number of Rows: 3796 This is the final verification. The program queried the database, which reported back that the table now contains 18 columns and 3,796 rows, proving the data transfer is complete and correct.
Link of student Github profile where lab assignment has	https://github.com/Nikhil07Gourkar/JavaLab

Vision of the Department

To be a well-known centre for pursuing computer education through innovative pedagogy, value-based education and industry collaboration.

Mission of the Department

To establish learning ambience for ushering in computer engineering professionals in core and multidisciplinary area by developing Problem-solving skills through emerging technologies.

been uploaded	
Conclusion	The practical was successfully completed. A Java application was built using JDBC to connect to a SQLite database. The program successfully created a table matching the structure of the provided CSV file, transferred all 3,796 data records, and displayed the final table metadata (18 columns and 3,796 rows), fulfilling all project requirements.
Plag Report (Similarity index < 12%)	
Date	06/11/25