**JAVA SWING BASED- Animal Species Database- SQL CONNECTIVITY USING JDBC**

**DBMS Project Report Submitted in partial fulfilment of the**

**Requirements for the award of the Degree of**

**BACHELOR OF ENGINEERING**

IN

**INFORMATION TECHNOLOGY**

BY

*K SRIHAS REDDY (1602-21-737-057)*

Under the guidance of **Ms B. Leelavathy**



Department of Information Technology

Vasavi College of Engineering (Autonomous)

(Affiliated to Osmania University)

Ibrahimbagh, Hyderabad-31

2022

***DECLARATION BY THE CANDIDATE :***

K SRIHAS REDDY bearing hall ticket numbers, **1602-21-737-057** hereby declare that the project report entitled **“Animal Species Database”** Department of Information Technology, Vasavi College of Engineering, Hyderabad, is submitted in partial fulfilment of the requirement for the award of the degree of Bachelor of Engineering in Information Technology This is a record of bonafide work carried out by me and the results embodied in this project report have not been submitted to any other university or institute for the award of any other degree or diploma.

**K SRIHAS REDDY**

**1602-21-737-057**

**Vasavi College of Engineering (Autonomous)**

**Ibrahimbagh, Hyderabad-31**

**Department of Information technology**



## BONAFIDE CERTIFICATE

This is to certify that the project entitled **“ANIMAL SPECIES DATABASE”** being submitted by **K SRIHAS REDDY,** bearing **1602-21-737-057,** in partial fulfilment of the requirement for the course of **DATABASE MANAGEMENT SYSTEM LAB** in BE 2/4 (IT) IV- Semester is a record of bonafide work carried out by him under my guidance.

**Ms B. Leelavathy**

**Assistant professor, Professor & HOD, Internal Guide. Dept. of IT.**

**External Examiner**

Github link :- https://github.com/SRIHASREDDY/ANIMAL-SPECIES-DATABASE.git

***Abstract :***

The main objective of this project is to find which animal species the user is looking for and group the features and display them the user gives the features and what to identify for which animal this features belongs to. This project give us the animal name along with its features when the user specifies any of the features .

•The animal species database request to table the animal table, the features table and the “has” table which is formed with the relation of animal and features table.

•The attributes of animal table are aid (animal id) of Domain type number which is the primary key of the animal table and aname(animal name) of domain type varchar2 and atype (animal type) of domain type varchar2 .

•Attributes of feature tables are fname(feature name) of type varchar2 and fname(feature name) off type varchar2 and fdesc(feature description) of type varchar2.

* Attributes of “has” table hid which is a primary key of the “has ”

Table of Domain type number , aname of Domain type varchar2 ,fname of domain type varchar2 and fdesc of domain type varchar2 .These tables receive the data from the animal table and features table respectively.

***AIM AND PRIORITY OF THE PROJECT :***

To create a Java GUI based management system for projects made by students in a particular college. All the values are to be updated and handled in the database using JDBC connectivity .

**ARCHITECTURE AND TECHNOLOGY USED :**

VSCode , Oracle 11g Database, Java SE version 8, MYSQL Developer. MYSQL developer is an Oracle Database utility, commonly used by users, administrators and programmers.

The interface of SQL Developer is used for creating the database. DDL and DML commands are implemented for operations being executed. The details of “animal “,”features” and “has” and their values are stored in the form of tables in the database.

VSCode is an integrated development environment (IDE) used in computer programming. It contains a base workspace and an extensible plug-in system for customizing the environment. VSCode its primary use is for developing java applications, but it may also be used to develop applications in other programming languages via plug-ins, including Erlang, JavaScript etc.

The front-end application code is written in “Java” using VSCode. The portal for front end application is designed through VSCode, runs and has the capacity to connect with the database which has data inserted using MYSQL.

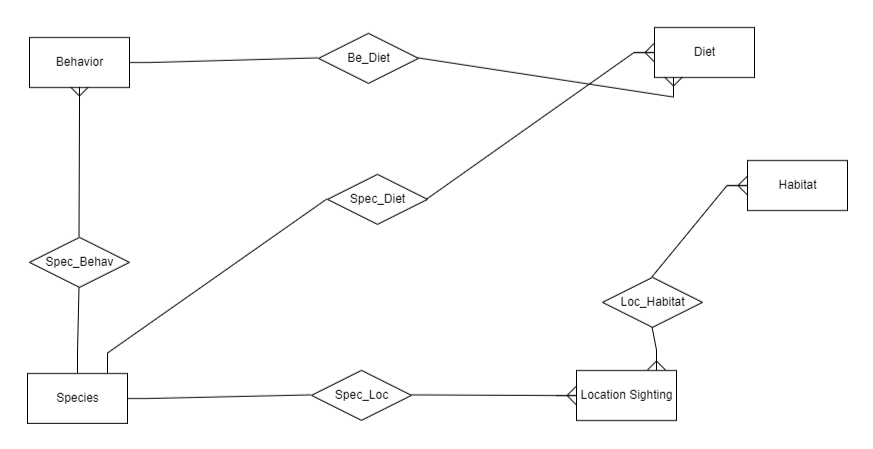
**Java Swing** tutorial is a part of Java Foundation Classes (JFC) that is used to create window-based applications. It is built on the top of AWT (Abstract Windowing Toolkit) API and entirely written in java. Unlike AWT, Java Swing provides platform-independent and lightweight components. The javax.swing package provides classes for java swing API such as JButton, JTextField, JTextArea, JRadioButton, JCheckbox, JMenu, JColorChooser etc.

**Java-SQL Connectivity using JDBC :**

Java Database Connectivity (JDBC) is an application programming interface (API) for the programming language Java, which defines how a client may access a database. It is a Java-based data access technology used for Java database connectivity. It is part of the Java Standard Edition platform, from Oracle Corporation. It provides methods to query and update data in a database and is oriented towards relational databases.

**DESIGN :**

* ***Entity Relationship Diagram :***



* ***Database design :***

CREATE TABLE LocationSighting (

sighting\_id INT PRIMARY KEY,

species\_id INT NOT NULL,

location VARCHAR(255) NOT NULL,

date\_time DATETIME NOT NULL,

observer\_name VARCHAR(255) NOT NULL,

FOREIGN KEY (species\_id) REFERENCES Species(species\_id)

);



CREATE TABLE Behavior (

behavior\_id INT PRIMARY KEY,

species\_id INT NOT NULL,

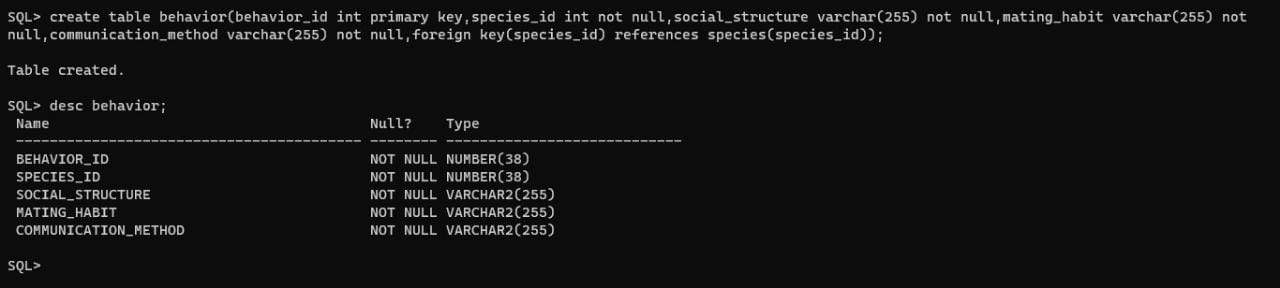
social\_structure VARCHAR(255) NOT NULL,

mating\_habit VARCHAR(255) NOT NULL,

communication\_method VARCHAR(255) NOT NULL,

FOREIGN KEY (species\_id) REFERENCES Species(species\_id)

);



CREATE TABLE Diet (

diet\_id INT PRIMARY KEY,

species\_id INT NOT NULL,

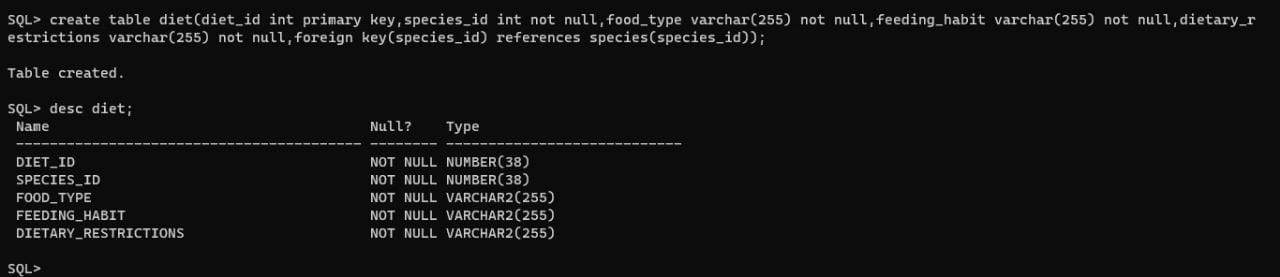
food\_type VARCHAR(255) NOT NULL,

feeding\_habit VARCHAR(255) NOT NULL,

dietary\_restrictions VARCHAR(255) NOT NULL,

FOREIGN KEY (species\_id) REFERENCES Species(species\_id)

);



CREATE TABLE Habitat (

habitat\_id INT PRIMARY KEY,

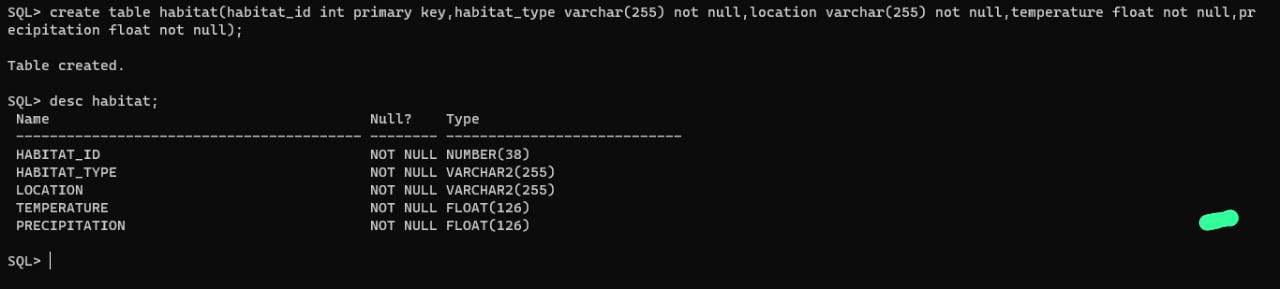
habitat\_type VARCHAR(255) NOT NULL,

location VARCHAR(255) NOT NULL,

temperature FLOAT NOT NULL,

precipitation FLOAT NOT NULL

);



CREATE TABLE Species (

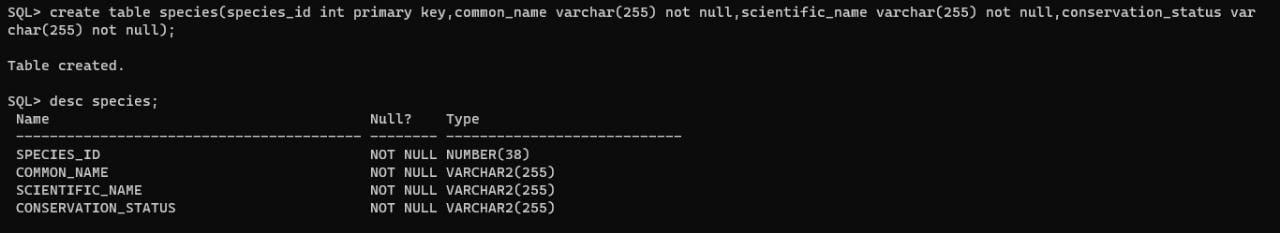
species\_id INT PRIMARY KEY,

common\_name VARCHAR(255) NOT NULL,

scientific\_name VARCHAR(255) NOT NULL,

conservation\_status VARCHAR(255) NOT NULL

);



INSERT INTO Behavior (behavior\_id, species\_id, social\_structure, mating\_habit, communication\_method)

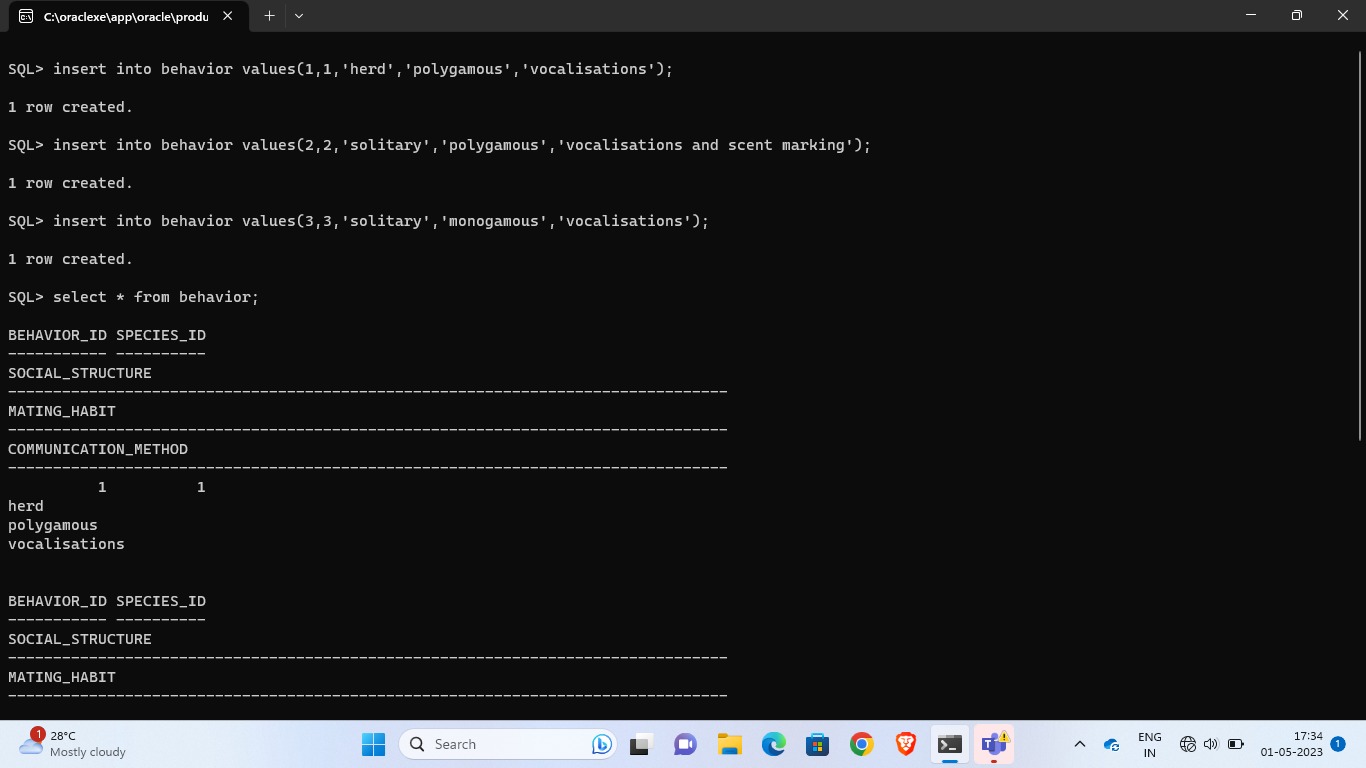
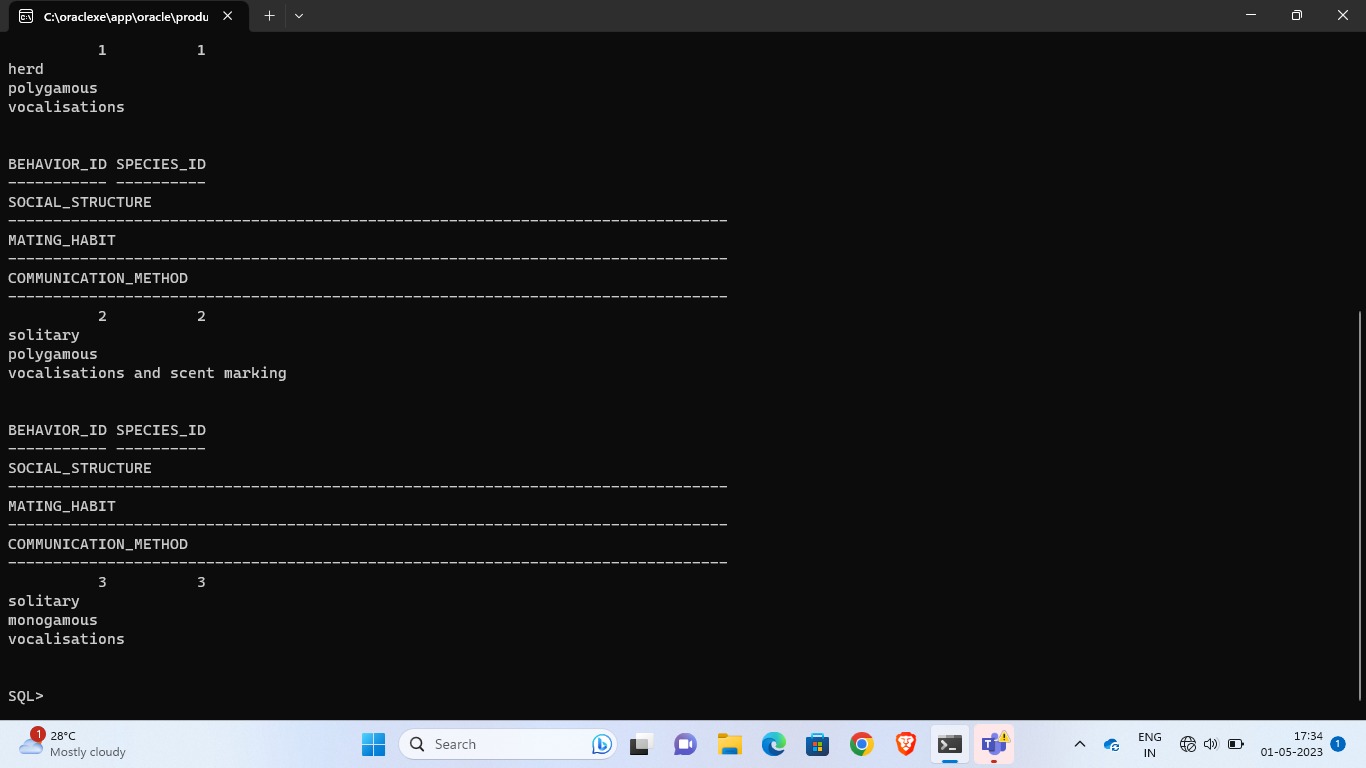
VALUES (1, 1, 'Herd', 'Polygamous', 'Vocalizations'),

(2, 2, 'Solitary or territorial', 'Polygamous', 'Vocalizations and scent marking'),

(3, 3, 'Solitary', 'Monogamous', 'Vocalizations'),

(4, 4, 'Solitary or in small groups', 'Unknown', 'Songs and calls'),

(5, 5, 'Solitary or territorial', 'Polygamous', 'Vocalizations and scent marking');



INSERT INTO Diet (diet\_id, species\_id, food\_type, feeding\_habit, dietary\_restrictions)

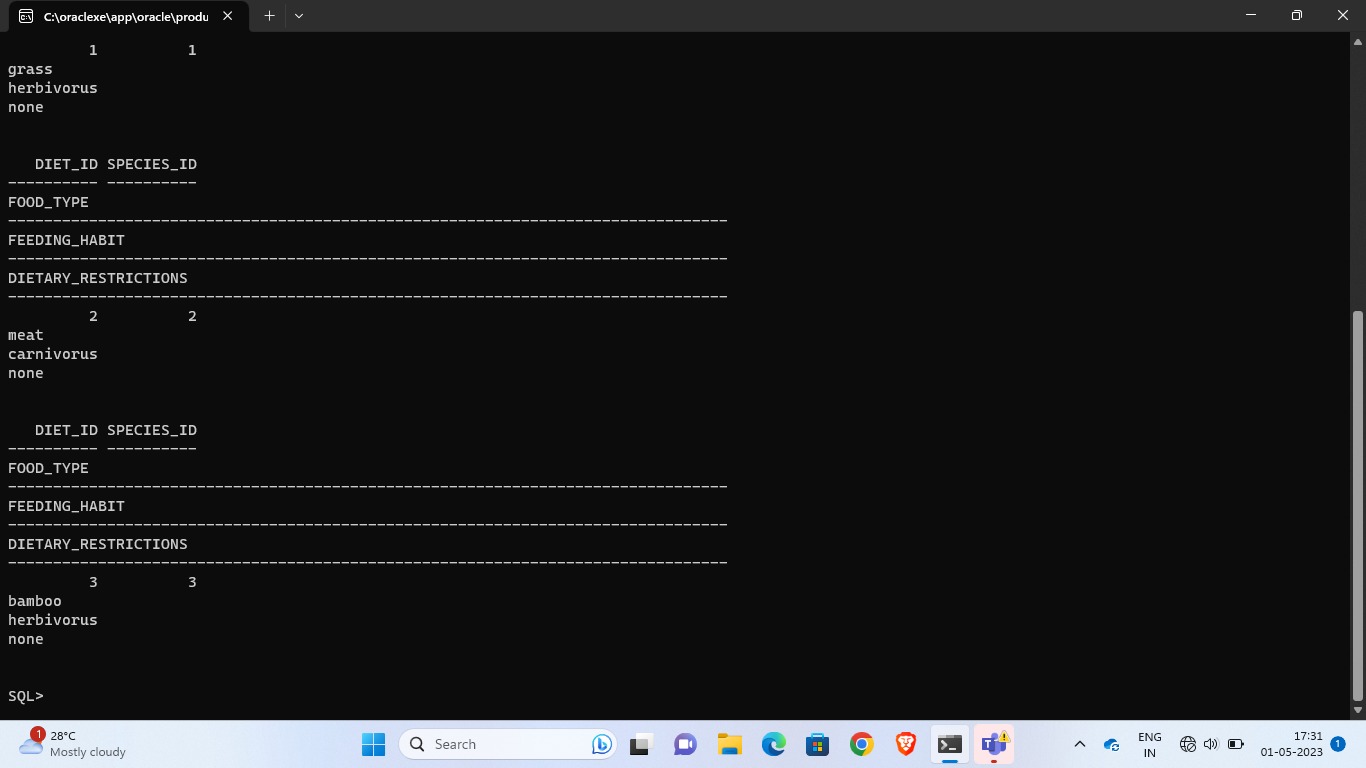
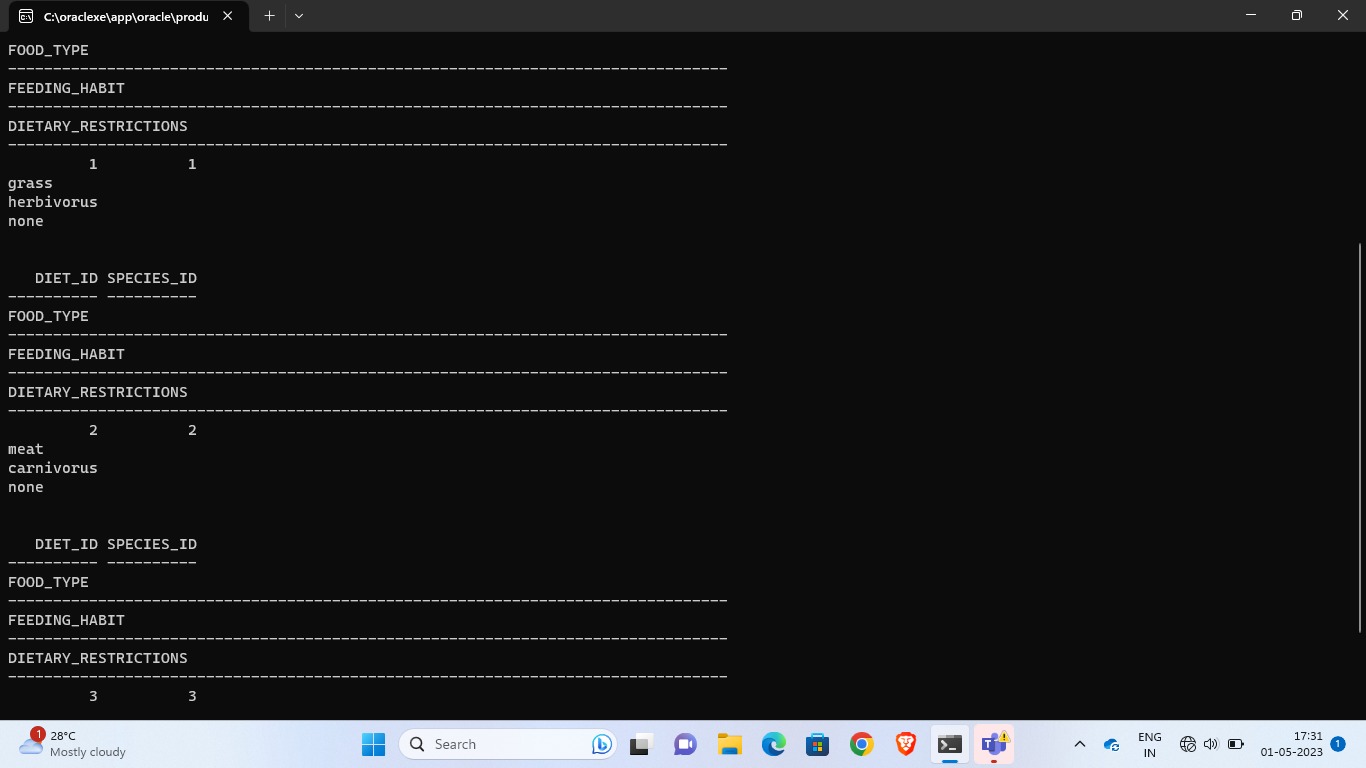
VALUES (1, 1, 'Grass', 'Herbivorous', 'None'),

(2, 2, 'Meat', 'Carnivorous', 'None'),

(3, 3, 'Bamboo', 'Herbivorous', 'None'),

(4, 4, 'Krill', 'Filter feeder', 'None'),

(5, 5, 'Wild goats, sheep, and ibex', 'Carnivorous', 'None');



INSERT INTO Habitat (habitat\_id, habitat\_type, location, temperature, precipitation)

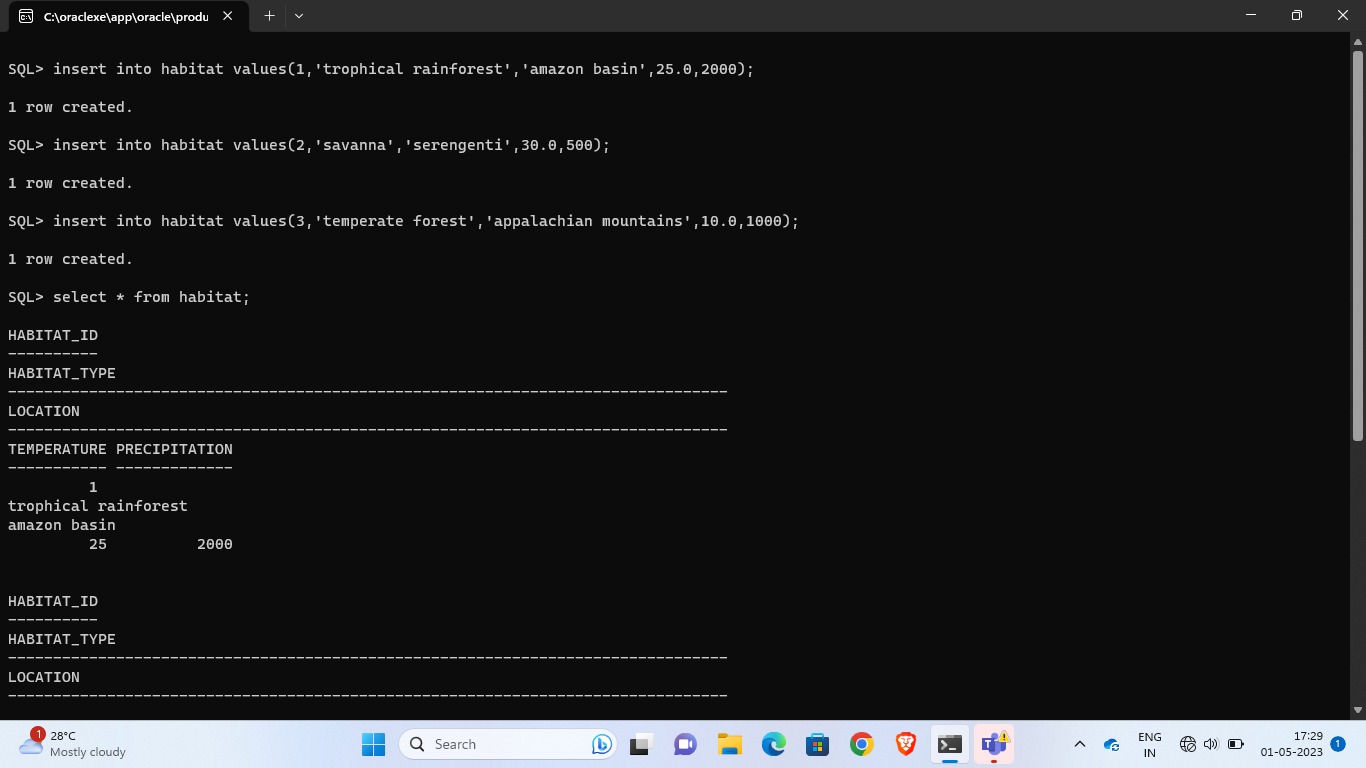
VALUES (1, 'Tropical rainforest', 'Amazon Basin', 25.0, 2000.0),

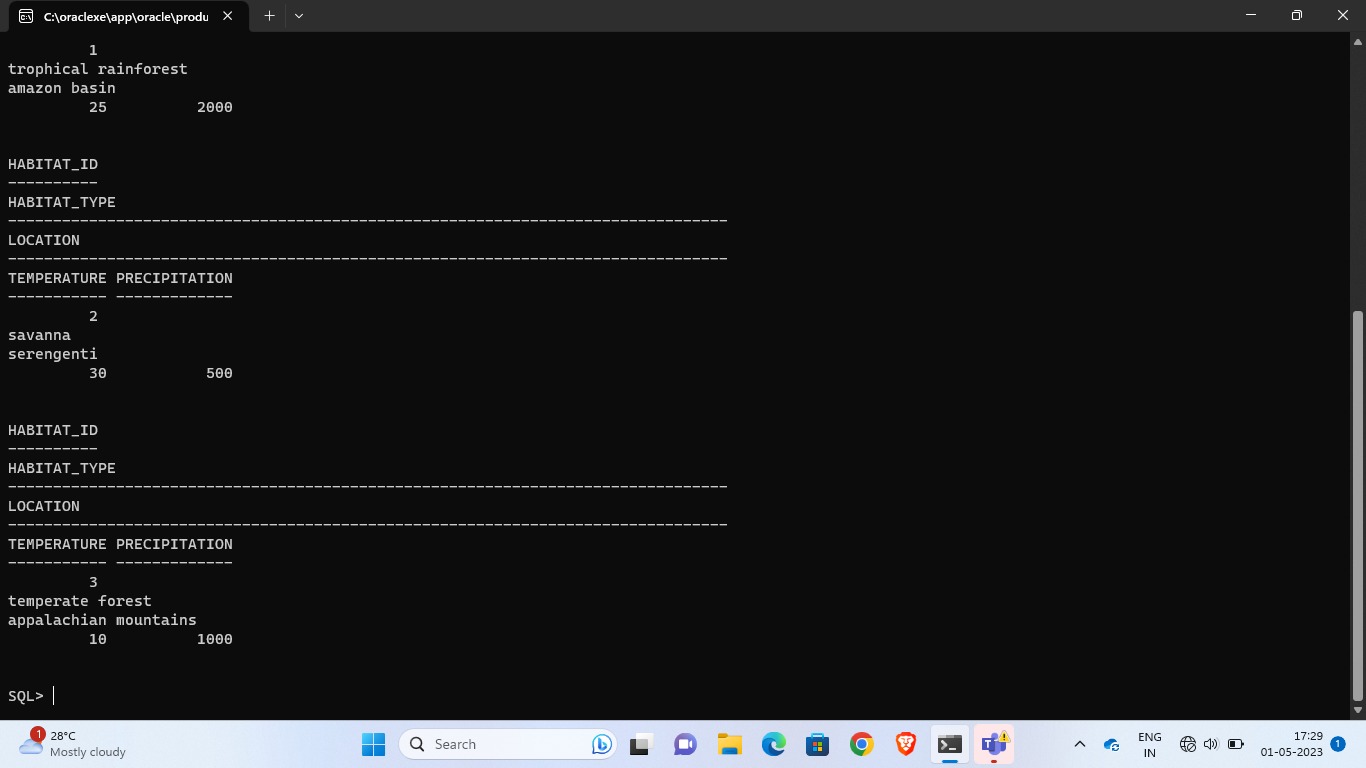
(2, 'Savanna', 'Serengenti', 30.0, 500.0),

(3, 'Temperate forest', 'Appalachian Mountains', 10.0, 1000.0),

(4, 'Arctic tundra', 'Alaska', -10.0, 250.0),

(5, 'Coral reef', 'Great Barrier Reef', 25.0, 2000.0);





INSERT INTO Species (species\_id, common\_name, scientific\_name, conservation\_status)

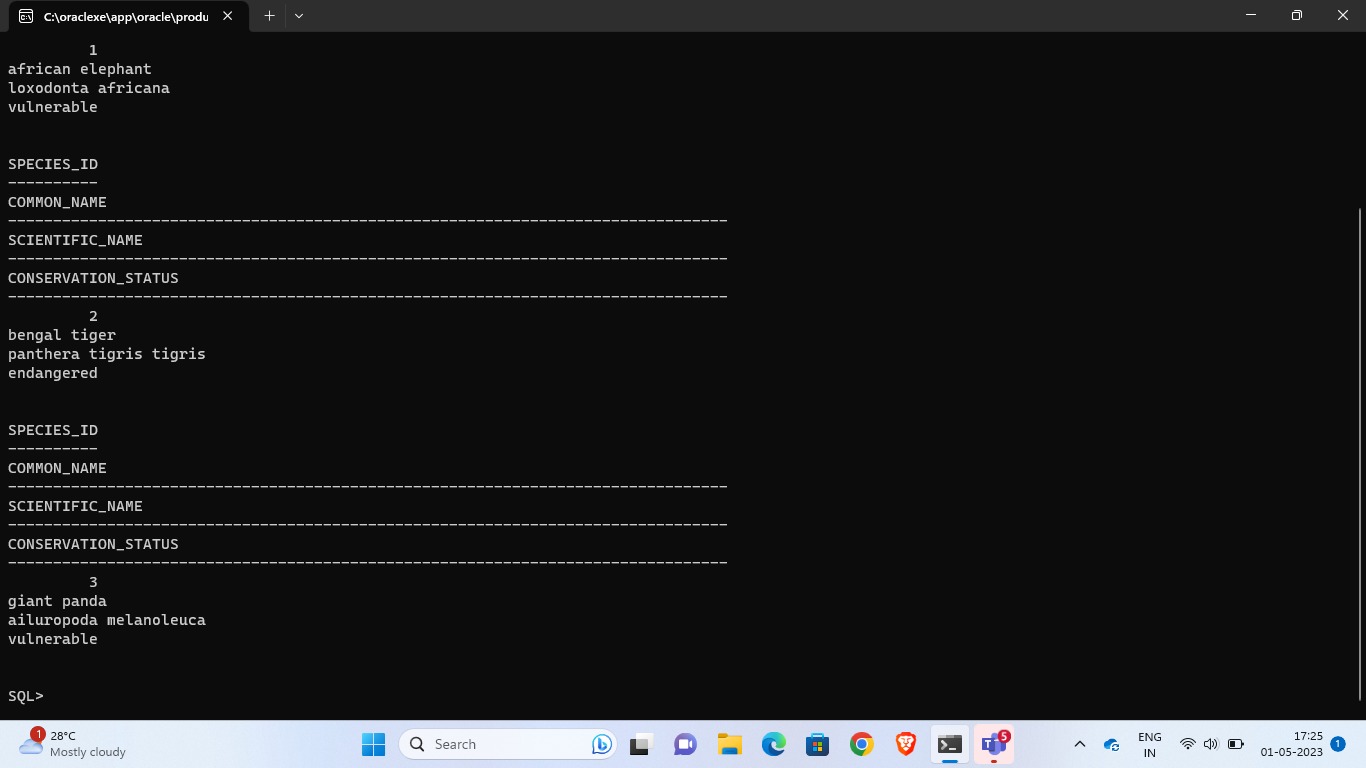
VALUES (1, 'African elephant', 'Loxodonta africana', 'Vulnerable'),

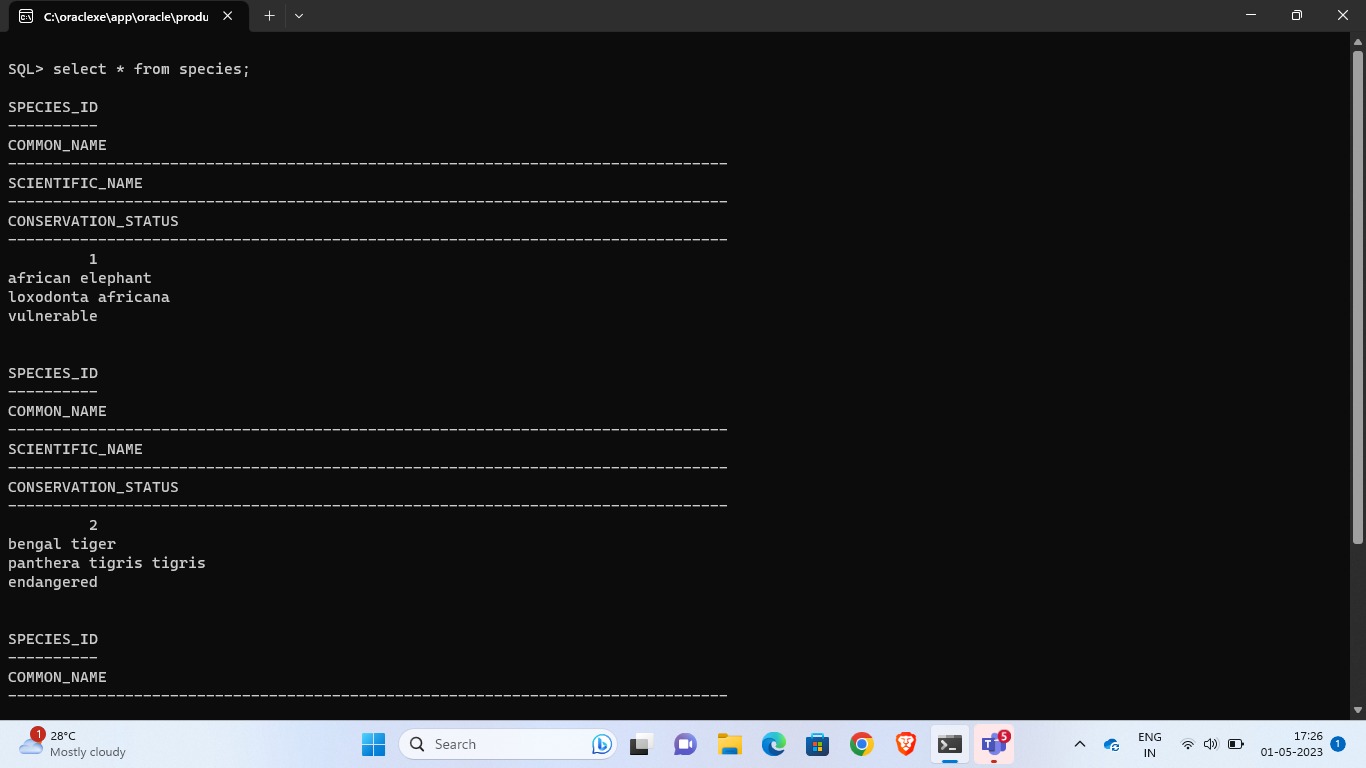
(2, 'Bengal tiger', 'Panthera tigris tigris', 'Endangered'),

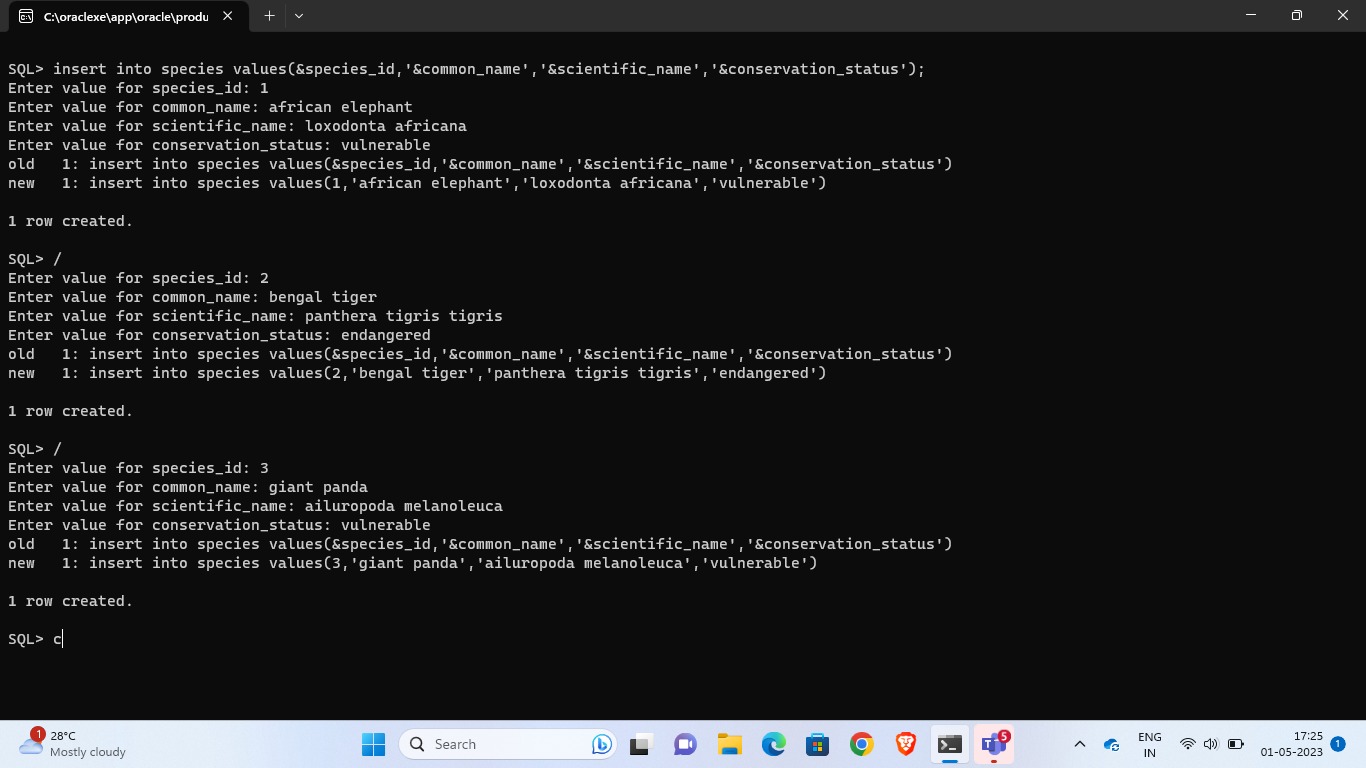
(3, 'Giant panda', 'Ailuropoda melanoleuca', 'Vulnerable'),

(4, 'Blue whale', 'Balaenoptera musculus', 'Endangered'),

(5, 'Snow leopard', 'Panthera uncia', 'Vulnerable');







INSERT INTO LocationSighting (sighting\_id, species\_id, location, date\_time, observer\_name)

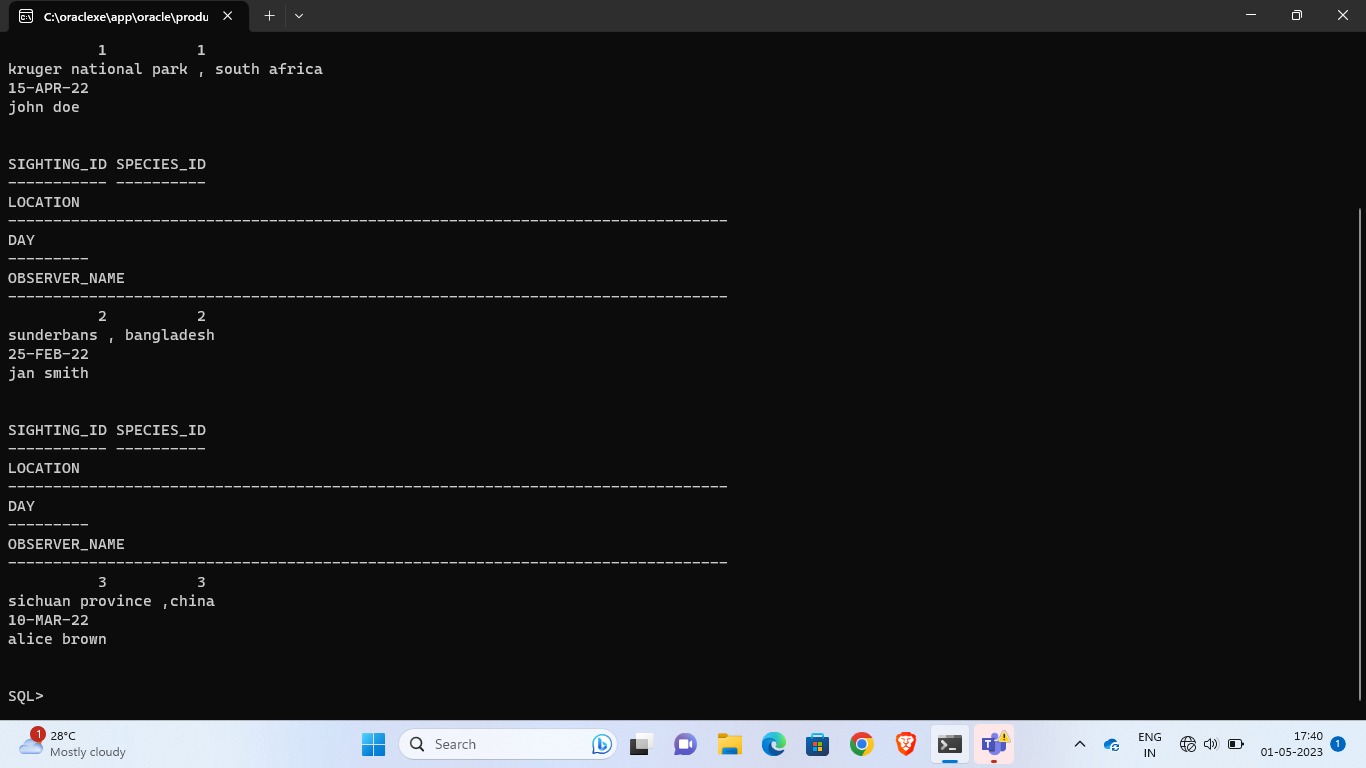
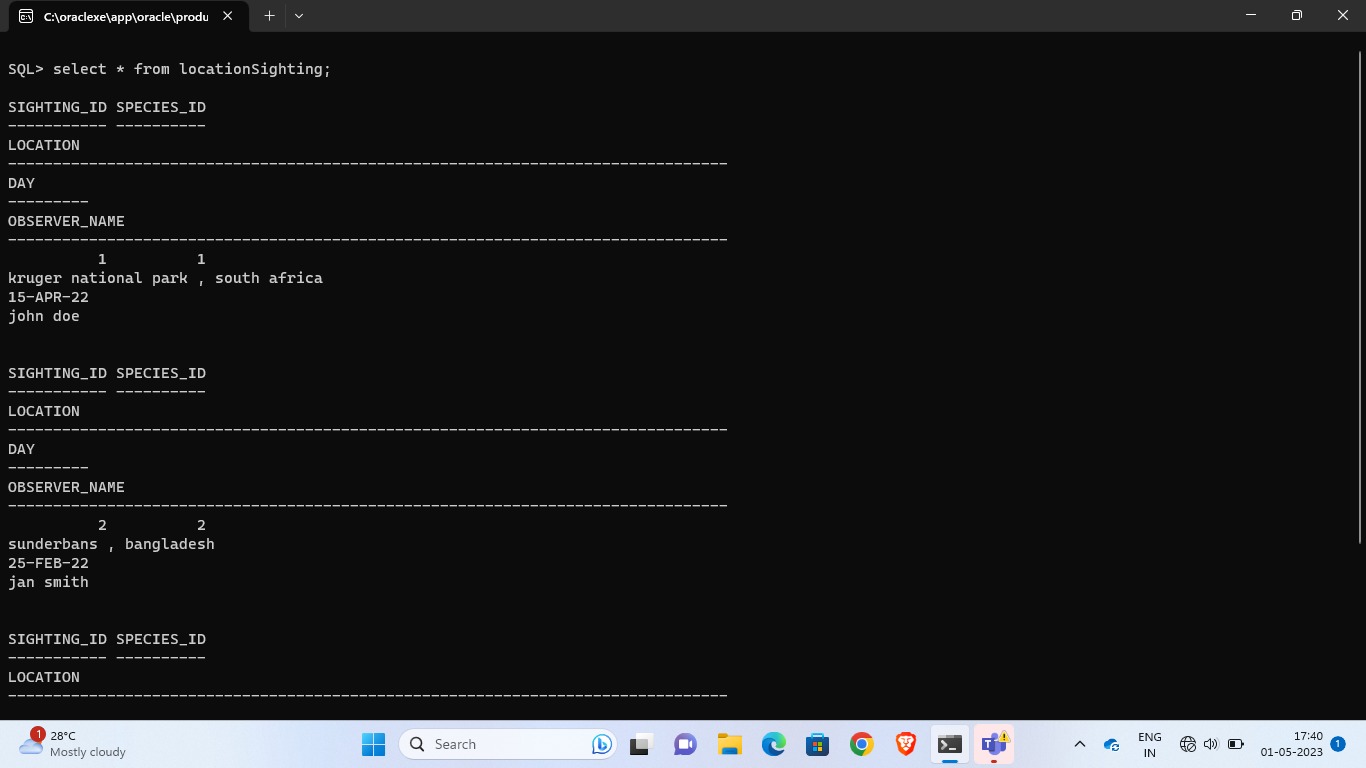
VALUES (1, 1, 'Kruger National Park, South Africa', '2022-04-15 10:30:00', 'John Doe'),

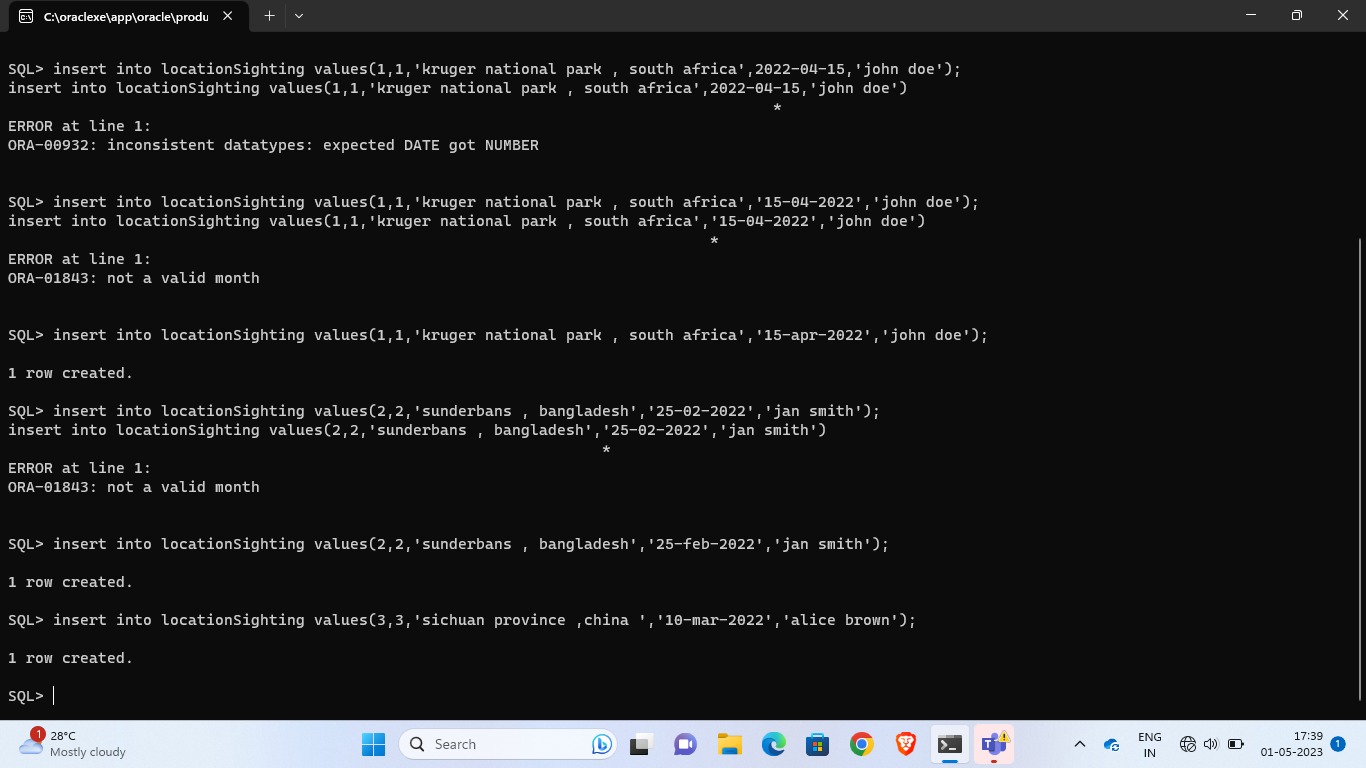
(2, 2, 'Sundarbans, Bangladesh', '2022-02-25 15:45:00', 'Jane Smith'),

(3, 3, 'Sichuan Province, China', '2022-03-10 08:00:00', 'Alice Brown'),

(4, 4, 'Gulf of Maine, USA', '2022-01-20 12:15:00', 'Bob Green'),

(5, 5, 'Himalayas, Nepal', '2022-05-01));





***Implementation:***

**import** javax.swing.\*;

**import** java.awt.\*;

**import** java.awt.event.\*;

**import** java.sql.\*;

**public** **class** MainFrame **extends** JFrame **implements** ActionListener {

**private** JPanel mainPanel;

**private** JComboBox<String> tablesComboBox;

**private** JButton insertButton;

**private** JButton updateButton;

**private** JButton deleteButton;

**private** JButton viewButton;

**private** JTextArea viewTextArea;

**private** Connection conn;

**private** String[] tableNames = {"animalspecies", "features", "symptoms"};

**private** String jdbcURL = "jdbc:oracle:thin:@localhost:1521/xe";

**private** String username = "ani";

**private** String password = "123";

**public** MainFrame() {

**try** {

Class.*forName*("oracle.jdbc.driver.OracleDriver");

} **catch** (Exception q) {

q.printStackTrace();

}

setTitle("Animal species");

setDefaultCloseOperation(JFrame.***EXIT\_ON\_CLOSE***);

setLayout(**new** BorderLayout());

mainPanel = **new** JPanel();

mainPanel.setLayout(**new** FlowLayout());

add(mainPanel, BorderLayout.***NORTH***);

JLabel selectTableLabel = **new** JLabel("Select Table:");

mainPanel.add(selectTableLabel);

tablesComboBox = **new** JComboBox<>(tableNames);

mainPanel.add(tablesComboBox);

insertButton = **new** JButton("INSERT");

insertButton.addActionListener(**this**);

mainPanel.add(insertButton);

updateButton = **new** JButton("UPDATE");

updateButton.addActionListener(**this**);

mainPanel.add(updateButton);

deleteButton = **new** JButton("DELETE");

deleteButton.addActionListener(**this**);

mainPanel.add(deleteButton);

viewButton = **new** JButton("VIEW");

viewButton.addActionListener(**this**);

mainPanel.add(viewButton);

viewTextArea = **new** JTextArea(10, 40);

viewTextArea.setEditable(**false**);

JScrollPane scrollPane = **new** JScrollPane(viewTextArea);

add(scrollPane, BorderLayout.***CENTER***);

pack();

setVisible(**true**);

}

@Override

**public** **void** actionPerformed(ActionEvent ae) {

String tableName = (String) tablesComboBox.getSelectedItem();

**if** (ae.getSource() == insertButton) {

insertValues(tableName);

} **else** **if** (ae.getSource() == updateButton) {

updateSelectedRows(tableName);

} **else** **if** (ae.getSource() == deleteButton) {

deleteSelectedRows(tableName);

} **else** **if** (ae.getSource() == viewButton) {

displayTable(tableName);

}

}

**private** **void** insertValues(String tableName) {

JTextField idField = **new** JTextField(10);

JTextField nameField = **new** JTextField(20);

JTextField descriptionField = **new** JTextField(50);

JPanel inputPanel = **new** JPanel(**new** GridLayout(3, 2));

inputPanel.add(**new** JLabel("ID:"));

inputPanel.add(idField);

inputPanel.add(**new** JLabel("Name:"));

inputPanel.add(nameField);

inputPanel.add(**new** JLabel("Description:"));

inputPanel.add(descriptionField);

**int** result = JOptionPane.*showConfirmDialog*(**this**, inputPanel, "Insert values for " + tableName,

JOptionPane.***OK\_CANCEL\_OPTION***, JOptionPane.***PLAIN\_MESSAGE***);

**if** (result == JOptionPane.***OK\_OPTION***) {

**try** {

Connection con = DriverManager.*getConnection*(jdbcURL, username, password);

String query = "INSERT INTO " + tableName + " VALUES (?, ?, ?)";

PreparedStatement stmt = con.prepareStatement(query);

stmt.setInt(1, Integer.*parseInt*(idField.getText()));

stmt.setString(2, nameField.getText());

stmt.setString(3, descriptionField.getText());

**int** rowsInserted = stmt.executeUpdate();

**if** (rowsInserted > 0) {

JOptionPane.*showMessageDialog*(**this**, "A row is inserted into " + tableName);

}

stmt.close();

con.close();

} **catch** (Exception ex) {

ex.printStackTrace();

JOptionPane.*showMessageDialog*(**this**, "Could not perform insert: " + ex.getMessage());

}

}

}

**private** **void** updateSelectedRows(String tableName) {

String selectedRows = viewTextArea.getSelectedText();

**if** (selectedRows == **null** || selectedRows.isEmpty()) {

JOptionPane.*showMessageDialog*(**this**, "No rows selected for update.");

**return**;

}

String[] rows = selectedRows.split("\n");

JTextField columnField = **new** JTextField(20);

JTextField valueField = **new** JTextField(50);

JPanel inputPanel = **new** JPanel(**new** GridLayout(2, 2));

inputPanel.add(**new** JLabel("Column Name:"));

inputPanel.add(columnField);

inputPanel.add(**new** JLabel("New Value:"));

inputPanel.add(valueField);

**int** result = JOptionPane.*showConfirmDialog*(**this**, inputPanel, "Update selected rows in " + tableName,

JOptionPane.***OK\_CANCEL\_OPTION***, JOptionPane.***PLAIN\_MESSAGE***);

**if** (result == JOptionPane.***OK\_OPTION***) {

**try** {

Connection con = DriverManager.*getConnection*(jdbcURL, username, password);

String columnName = columnField.getText();

String newValue = valueField.getText();

**for** (String row : rows) {

String[] rowData = row.split("\t");

**int** id = Integer.*parseInt*(rowData[0]);

String query = "UPDATE " + tableName + " SET " + columnName + " = ? WHERE id = ?";

PreparedStatement stmt = con.prepareStatement(query);

stmt.setString(1, newValue);

stmt.setInt(2, id);

stmt.executeUpdate();

stmt.close();

}

JOptionPane.*showMessageDialog*(**this**, "Selected rows updated in " + tableName);

con.close();

} **catch** (Exception ex) {

JOptionPane.*showMessageDialog*(**this**, "Could not perform update: " + ex.getMessage());

}

}

}

**private** **void** deleteSelectedRows(String tableName) {

String selectedRows = viewTextArea.getSelectedText();

**if** (selectedRows == **null** || selectedRows.isEmpty()) {

JOptionPane.*showMessageDialog*(**this**, "No rows selected for delete.");

**return**;

}

String[] rows = selectedRows.split("\n");

**int** result = JOptionPane.*showConfirmDialog*(**this**, "Delete selected rows from " + tableName + "?",

"Confirmation", JOptionPane.***YES\_NO\_OPTION***, JOptionPane.***QUESTION\_MESSAGE***);

**if** (result == JOptionPane.***YES\_OPTION***) {

**try** {

Connection con = DriverManager.*getConnection*(jdbcURL, username, password);

**for** (String row : rows) {

String[] rowData = row.split("\t");

**int** id = Integer.*parseInt*(rowData[0]);

String query = "DELETE FROM " + tableName + " WHERE id = ?";

PreparedStatement stmt = con.prepareStatement(query);

stmt.setInt(1, id);

stmt.executeUpdate();

stmt.close();

}

JOptionPane.*showMessageDialog*(**this**, "Selected rows deleted from " + tableName);

con.close();

} **catch** (Exception ex) {

JOptionPane.*showMessageDialog*(**this**, "Could not perform delete: " + ex.getMessage());

}

}

}

**private** **void** displayTable(String tableName) {

viewTextArea.setText("");

**try** {

Connection con = DriverManager.*getConnection*(jdbcURL, username, password);

Statement stmt = con.createStatement();

String query = "SELECT \* FROM " + tableName;

ResultSet rs = stmt.executeQuery(query);

ResultSetMetaData rsmd = rs.getMetaData();

**int** columnCount = rsmd.getColumnCount();

// Display column names

StringBuilder header = **new** StringBuilder();

**for** (**int** i = 1; i <= columnCount; i++) {

header.append(rsmd.getColumnName(i)).append("\t");

}

viewTextArea.append(header.toString() + "\n");

// Display tablerows

**while** (rs.next()) {

StringBuilder row = **new** StringBuilder();

**for** (**int** i = 1; i <= columnCount; i++) {

row.append(rs.getString(i)).append("\t");

}

viewTextArea.append(row.toString() + "\n");

}

stmt.close();

con.close();

} **catch** (Exception ex) {

JOptionPane.*showMessageDialog*(**this**, "Could not display the table: " + ex.getMessage());

}

}

**public** **static** **void** main(String[] args) {

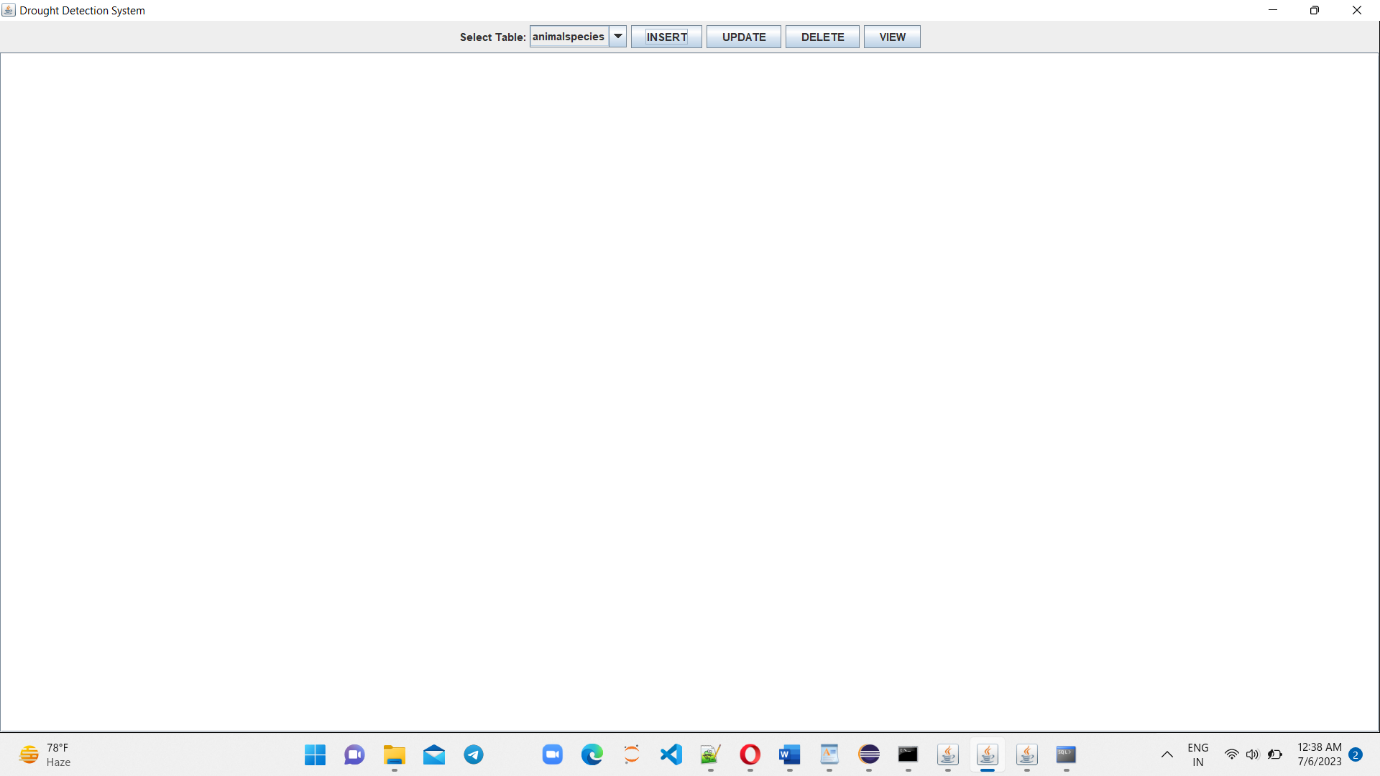
SwingUtilities.*invokeLater*(() -> **new** MainFrame());

}

}

* ***IMPLEMENTATION :***

**Welcome page :**



INSERT PAGE:

A screenshot of a computer

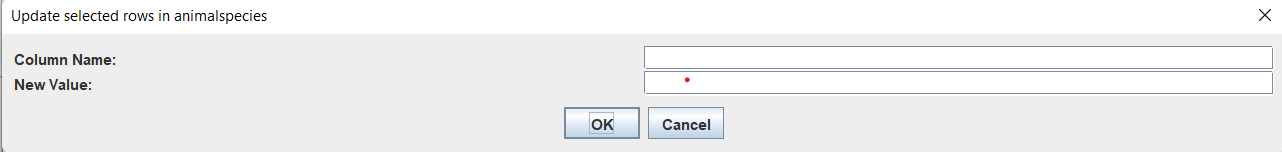
Description automatically generated

DELETE PAGE

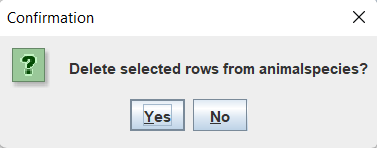
A screenshot of a computer

Description automatically generated

***UPDATE PAGE***



***SUCCESS MESSAGE***

******

***FUTURE WORK :***

In the future we aim to add better search features using many other parameters and add other attributes in the projects table too.

**REFERENCES :**

<https://docs.oracle.com/javase/7/docs/api/>

<https://www.geeksforgeeks.org/establishing-jdbc-connection-in-java/>

<https://www.javatpoint.com/java-awt>