Java Programming – JDBC - MS Access

Arthur Hoskey, Ph.D.
Farmingdale State College
Computer Systems Department

Today's Lecture

- MS Access Database
- Java Database Connectivity (JDBC)

Microsoft Access Database

- Microsoft Access is a relational DB.
- Part of Microsoft Office.
- Used for small desktop databases.
- Easy to use.

Microsoft Access – Create Database

- Create a blank database. The database file should be located in the NetBeans project directory for easiest access. Give the DB an appropriate name. We will use Persons for this example. The DB filename will have a .accdb extension.

Create a Database in Access

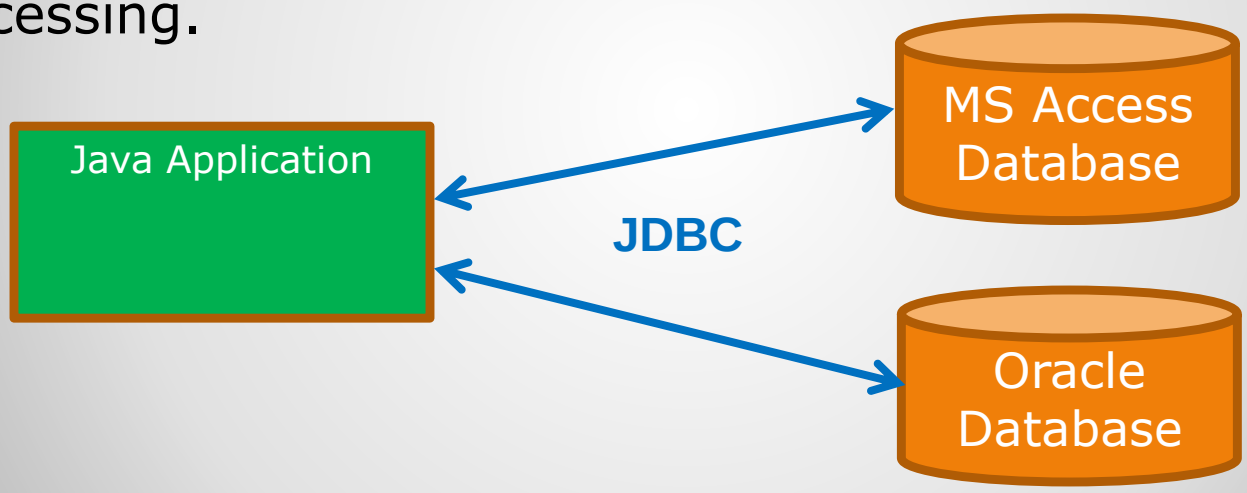
Create a Table and Fields

Microsoft Access – Create Table

- Go to Create|Table to add a table to the DB. You will see a new table appear in the main part of the screen (it will have a default name such as Table 1).
- The table's fields are listed across the top.
- You will see an ID field already in the table.
- Add Field. Press the "Click to Add" dropdown (second column in table). Choose the datatype for the field. Once you choose the datatype it will automatically let you edit the field name. You can try adding columns for first, last, and age.
- Once all fields are created you can add data directly into the table. Try putting a few rows of data in the table.
- Save. Go to File|Save. It will ask you to enter a name for the table. You can name it Persons for this example.

Java Database Connectivity (JDBC)

- JDBC defines a standard API to access a relational database from a Java application.
- You can access different databases using JDBC.
- The JDBC code you use in your application is basically the same no matter what type of relational database you are accessing.



Java Database Connectivity (JDBC)

- Add the following two dependencies to pom.xml (Maven file).
- Add as children of <dependencies> (add <dependencies> if you need to).

Dependency for creating MS Access files using Java code

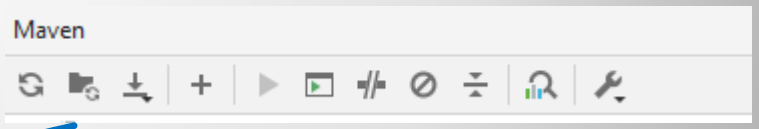
```
<dependency>
    <groupId>com.healthmarketscience.jackcess</groupId>
    <artifactId>jackcess</artifactId>
    <version>4.0.5</version>
</dependency>

<dependency>
  <groupId>net.sf.ucanaccess</groupId>
  <artifactId>ucanaccess</artifactId>
  <version>5.0.1</version>
</dependency>
```

Dependency for JDBC Driver for MS Access

Note: If Maven does not automatically download the dependencies, IntelliJ will not give you the option to choose "Import" when it does not recognize class names. Do the following:

Go to the Maven tab (on right). Press Reload All Maven Projects in the toolbar.



First button is for Reload

Link for dependency:
https://search.maven.org/artifact/net.sf.ucanaccess/ucanaccess

Maven MS Access Dependencies

Update module-info.java

- Java Module. Higher level grouping compared to packages.
- module-info.java contains module information (what other modules it requires to run, which packages within this module are allowed to be used by other modules).
- module-info.java is located in the src/Java directory.
- If your project has a module-info.java file, then you must add some requires statements.
- An IntelliJ JavaFX project is automatically setup as a module, so you will need to modify the module-info.java file in this case.

```
module <your package name will be here> {
   // Other requires are here

   requires java.sql;
   requires com.healthmarketscience.jackcess;

   // Other code here
}
```

Module Dependency
This module requires the java.sql module (need this to use JDBC)

Module Dependency
This module requires the com.healthmarketscience.jackcess module (need this to use Jackcess)

- Use the following code to create a DB in code.
- It wil only create the DB file if it does not already exist.

```
String dbFilePath = ".//Persons.accdb";
String databaseURL = "jdbc:ucanaccess://" + dbFilePath;

File dbFile = new File(dbFilePath);

if (!dbFile.exists()) {
    try (Database db =
            DatabaseBuilder.create(Database.FileFormat.V2010, new File(dbFilePath))) {
        System.out.println("The database file has been created.");
    } catch (IOException ioe) {
        ioe.printStackTrace(System.err);
    }
}
```

Do not create if the file already exists

Call DatabaseBuilder.create to create the db file

Java – Create a Database using Code

Java – Create a JDBC Connection

- Use the following code to open a JDBC connection to your MS Access DB:
- JDBC code requires imports from SQL lib. Here is the connection import: import java.sql.Connection;

```
String databaseURL = "";
Connection conn = null;

try {

    databaseURL = "jdbc:ucanaccess://.//Persons.accdb";
    conn = DriverManager.getConnection(databaseURL);

} catch (SQLException ex) {
    Logger.getLogger(Main.class.getName()).log(Level.SEVERE, null, ex);
}
```

Persons.accdb is the database filename (assumes it is stored in the project directory)

Open connection to the DB

- Use the following code to create a table in a DB using code.

```
try {
   String dbFilePath = ".//Persons.accdb";
   String databaseURL = "jdbc:ucanaccess://" + dbFilePath;

   Connection conn = DriverManager.getConnection(databaseURL);

   String sql;
   sql = "CREATE TABLE Persons (First nvarchar(255), Last nvarchar(255), Age INT)";

   Statement createTableStatement = conn.createStatement();
   createTableStatement.execute(sql);

   conn.commit();
} catch (SQLException sqlException) {
   sqlException.printStackTrace();
}
```

Get connection to db

Put SQL create in a string

Create a Statement instance and run the SQL create

Java – Create a Table in a Database using Code

- Use the following code to drop a table from a DB using code.

```
try {
   String dbFilePath = ".//Persons.accdb";
   String databaseURL = "jdbc:ucanaccess://" + dbFilePath;
   Connection conn = DriverManager.getConnection(databaseURL);

   String sql;
   sql = "DROP TABLE Persons";

   Statement createTableStatement = conn.createStatement();
   createTableStatement.execute(sql);

   conn.commit();
} catch (SQLException sqlException) {
   sqlException.printStackTrace();
}
```

Get connection to db

Put SQL drop in a string

Create a Statement instance and run the SQL drop

Java – Drop a Table from a Database using Code

- The following code queries the MS Access DB using an already opened JDBC connection (assumes that a table named Persons exists in the DB):

```
try {
    String tableName = "Persons";
    Statement stmt = conn.createStatement();
    ResultSet result = stmt.executeQuery("select * from " + tableName);

    while (result.next()) {
        int id = result.getInt("ID");
        String first = result.getString("First");
        String last = result.getString("Last");
        int age = result.getInt("Age");

        System.out.printf("%d %s %s %d\n", id, first, last, age);
    }
} catch (SQLException except) {
   except.printStackTrace();
}
```

ResultSet contains all rows of data from the DB

Loop through the ResultSet

Get data for each column and store in variables

Java – Query the DB using JDBC

Java –JDBC Insert Using a PreparedStatement

- The PreparedStatement class makes it easy to put values into a statement.

The three ? characters will be filled with data by the PreparedStatement class

```
String sql = "INSERT INTO Persons (First, Last, Age) VALUES (?, ?, ?)";

PreparedStatement preparedStatement = null;
try {
    preparedStatement = conn.prepareStatement(sql);
    preparedStatement.setString(1, first);
    preparedStatement.setString(2, last);
    preparedStatement.setInt(3, age);
    preparedStatement.executeUpdate();
} catch (SQLException e) {
    throw new RuntimeException(e);
}
```

Assumes that conn is a valid connection to the database

Replace the first ? In the SQL string with data from the first variable

Replace the third ? In the SQL string with data from the age variable. It uses setInt since age is an int.

- You can delete all rows from a table in a database.

Create SQL string to delete all records from the table

```
String sql = "DELETE FROM Persons";
PreparedStatement preparedStatement = null;

try {
    preparedStatement = conn.prepareStatement(sql);

    int rowsDeleted = preparedStatement.executeUpdate();
} catch (SQLException e) {
    throw new RuntimeException(e);
}
```

Assumes that conn is a valid connection to the database

Run the delete statement. It will return the number of rows that were deleted.

Java –JDBC Delete All Table Data

- You can delete rows that match certain values from a table in a database.

```
String first = "Mateo";
String last = "Lopez";

String sql = "DELETE FROM Persons WHERE first=? AND last=? ";
PreparedStatement preparedStatement = null;

try {
    preparedStatement = conn.prepareStatement(sql);
    preparedStatement.setString(1, first);
    preparedStatement.setString(2, last);

    int rowsDeleted = preparedStatement.executeUpdate();
} catch (SQLException e) {
    throw new RuntimeException(e);
}
```

Deleting records that have "Mateo" in first and "Lopez" in last

Deletes records that match the given first and last names

Assumes that conn is a valid connection to the database

Run the delete statement. It will return the number of rows that were deleted.

Java –JDBC Delete Single Item

- End of Slides

End of Slides