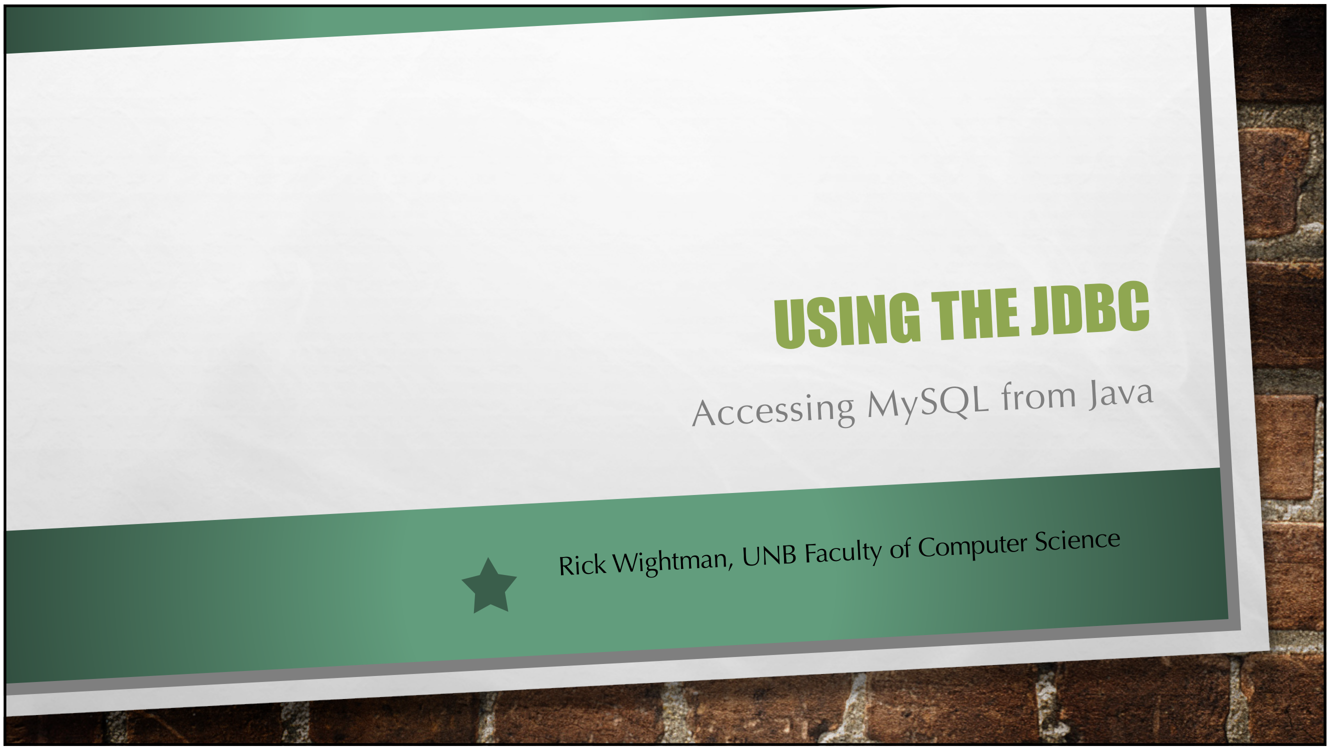
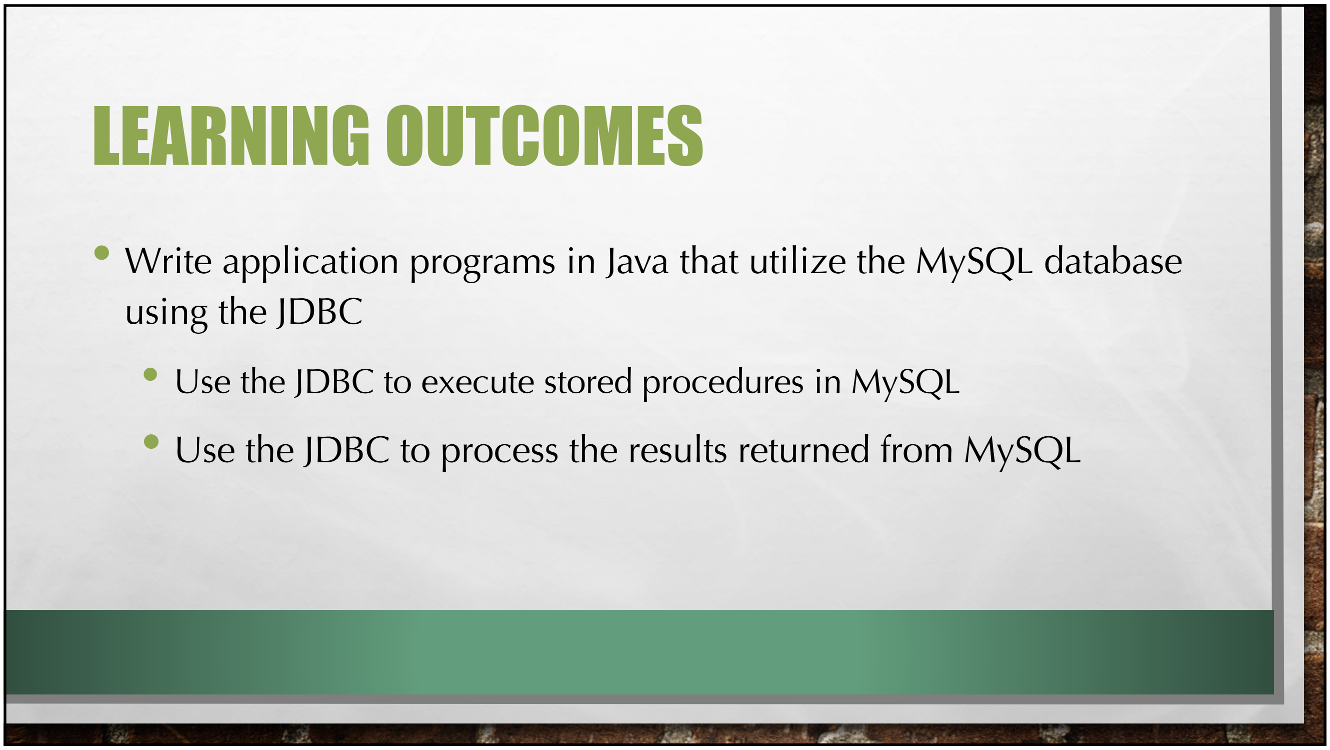


USING THE JDBC

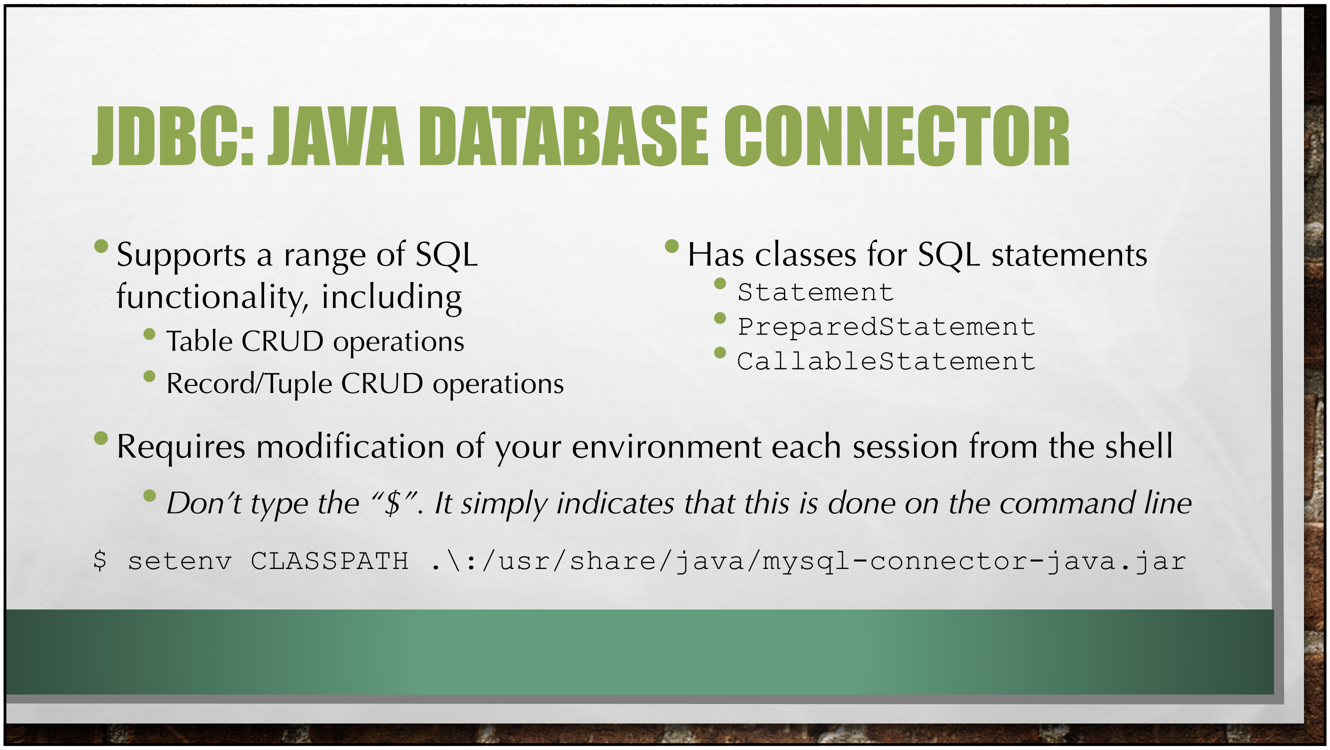
Accessing MySQL from Java

Rick Wightman, UNB Faculty of Computer Science



LEARNING OUTCOMES

- Write application programs in Java that utilize the MySQL database using the JDBC
  - Use the JDBC to execute stored procedures in MySQL
  - Use the JDBC to process the results returned from MySQL

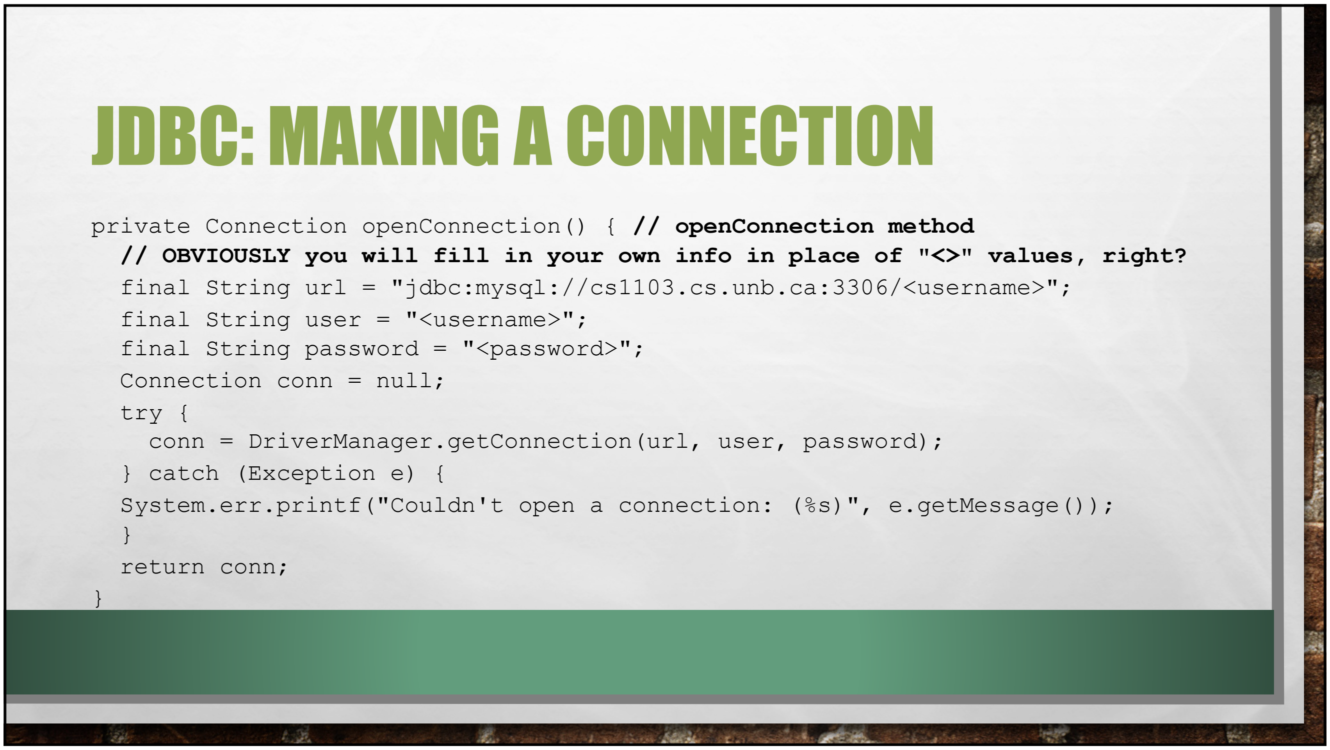


JDBC: JAVA DATABASE CONNECTOR

- Supports a range of SQL functionality, including
  - Table CRUD operations
  - Record/Tuple CRUD operations
- Has classes for SQL statements
  - `Statement`
  - `PreparedStatement`
  - `CallableStatement`
- Requires modification of your environment each session from the shell
  - *Don't type the "$". It simply indicates that this is done on the command line*

```
$ setenv CLASSPATH .\:/usr/share/java/mysql-connector-java.jar
```

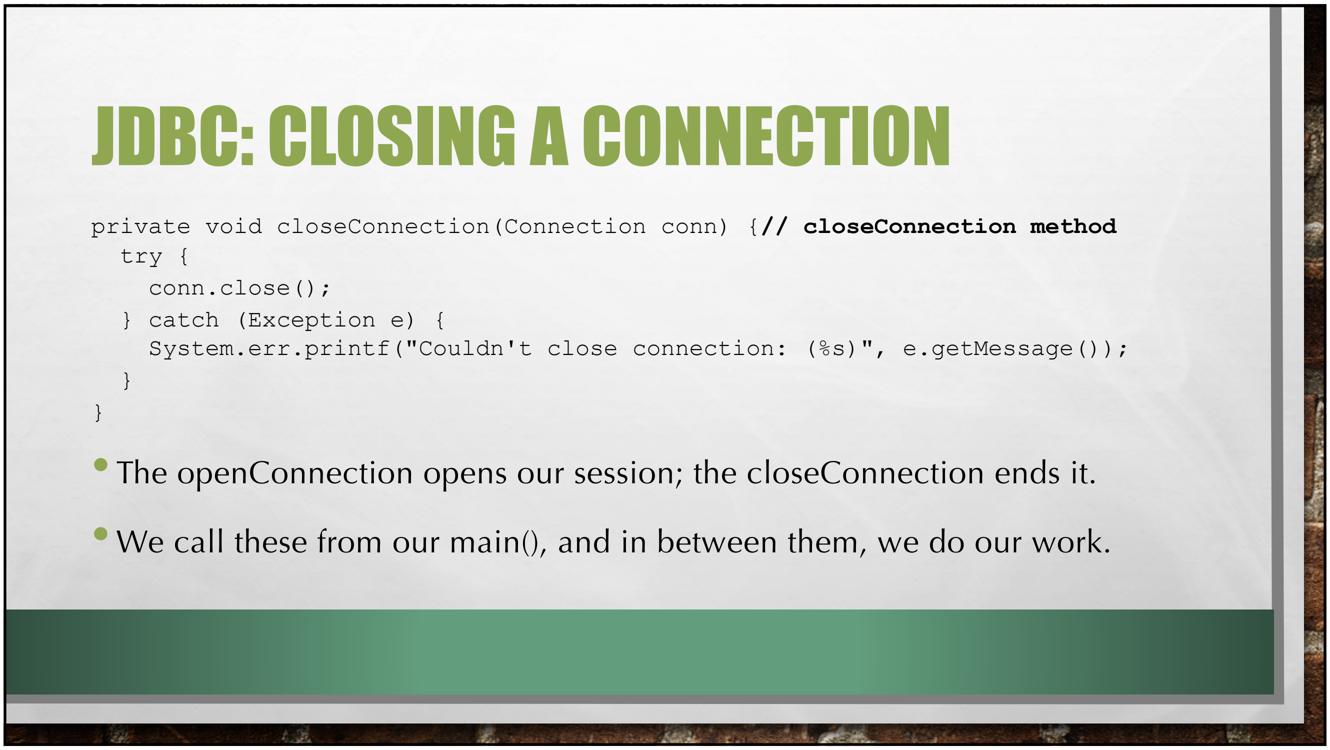
3



JDBC: MAKING A CONNECTION

```
private Connection openConnection() { // openConnection method
  // OBVIOUSLY you will fill in your own info in place of "<>" values, right?
  final String url = "jdbc:mysql://cs1103.cs.unb.ca:3306/<username>";
  final String user = "<username>";
  final String password = "<password>";
  Connection conn = null;
  try {
    conn = DriverManager.getConnection(url, user, password);
  } catch (Exception e) {
  System.err.printf("Couldn't open a connection: (%s)", e.getMessage());
  }
  return conn;
}
```

4

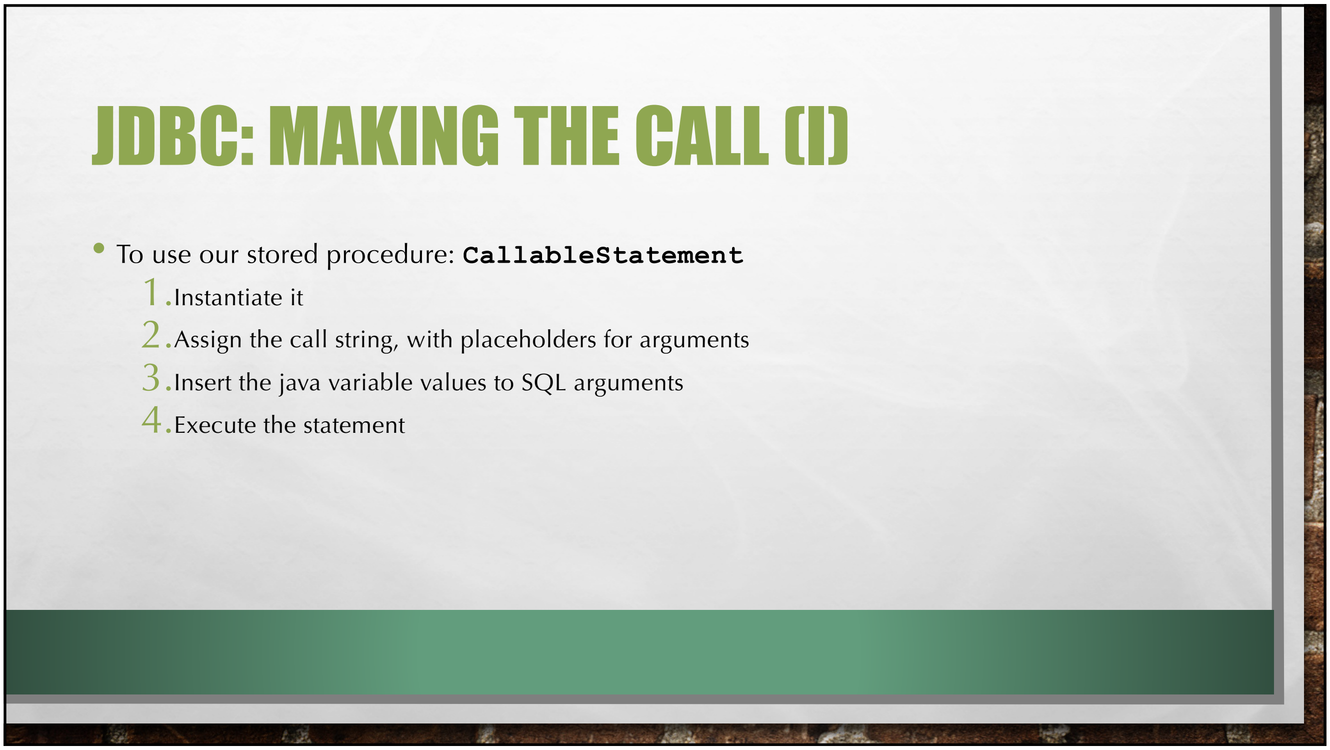


JDBC: CLOSING A CONNECTION

```
private void closeConnection(Connection conn) {// closeConnection method
  try {
    conn.close();
  } catch (Exception e) {
    System.err.printf("Couldn't close connection: (%s)", e.getMessage());
  }
}
```

- The openConnection opens our session; the closeConnection ends it.
- We call these from our main(), and in between them, we do our work.

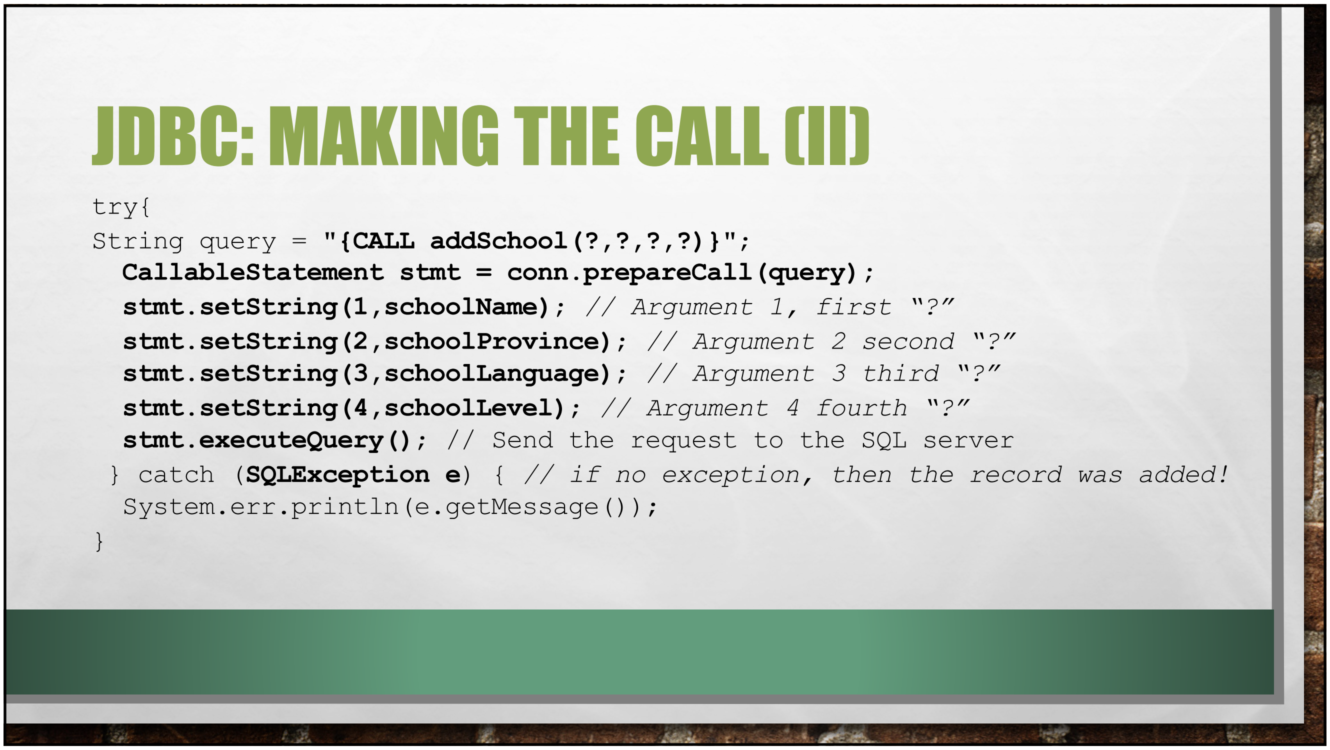
5



JDBC: MAKING THE CALL (I)

- To use our stored procedure: **`CallableStatement`**
  1. Instantiate it
  2. Assign the call string, with placeholders for arguments
  3. Insert the java variable values to SQL arguments
  4. Execute the statement

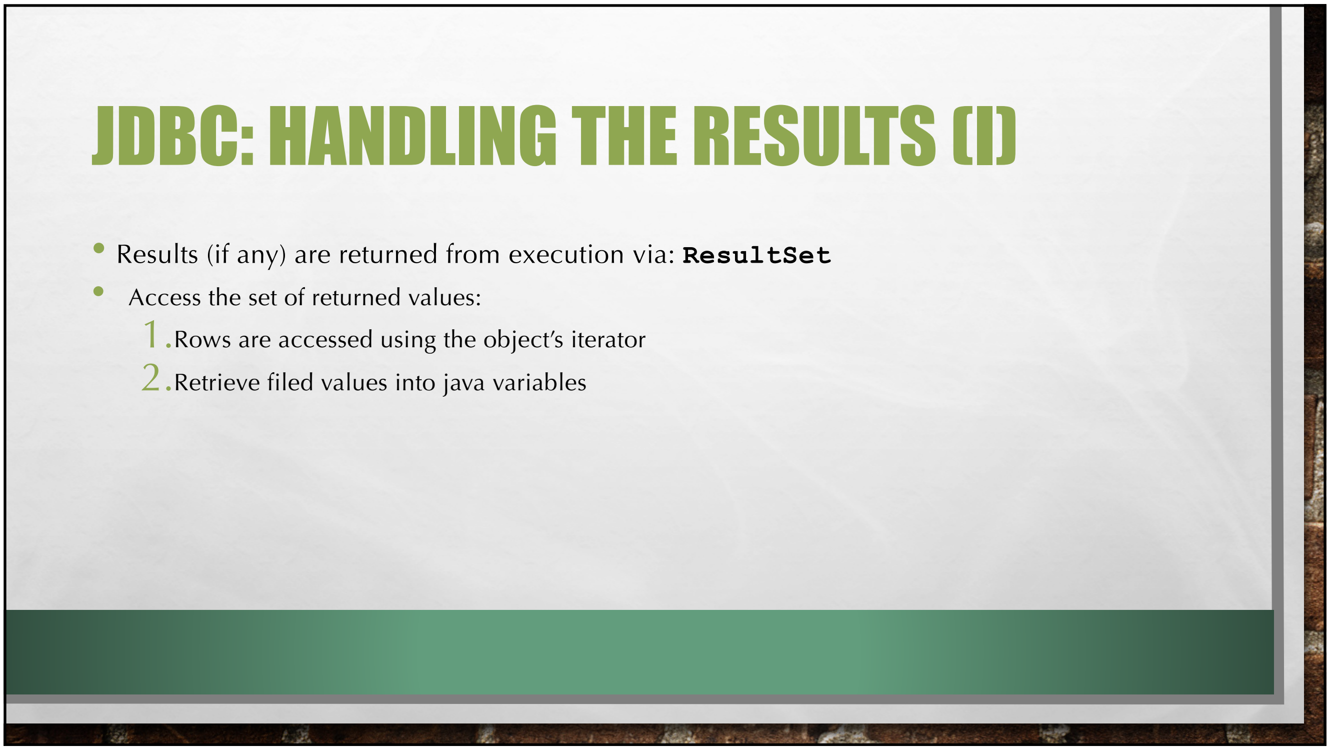
6



JDBC: MAKING THE CALL (II)

```
try{
String query = "{CALL addSchool(?,?,?,?)}";
  CallableStatement stmt = conn.prepareCall(query);
  stmt.setString(1,schoolName); // Argument 1, first “?”
  stmt.setString(2,schoolProvince); // Argument 2 second “?”
  stmt.setString(3,schoolLanguage); // Argument 3 third “?”
  stmt.setString(4,schoolLevel); // Argument 4 fourth “?”
  stmt.executeQuery(); // Send the request to the SQL server
 } catch (SQLException e) { // if no exception, then the record was added!
  System.err.println(e.getMessage());
}
```

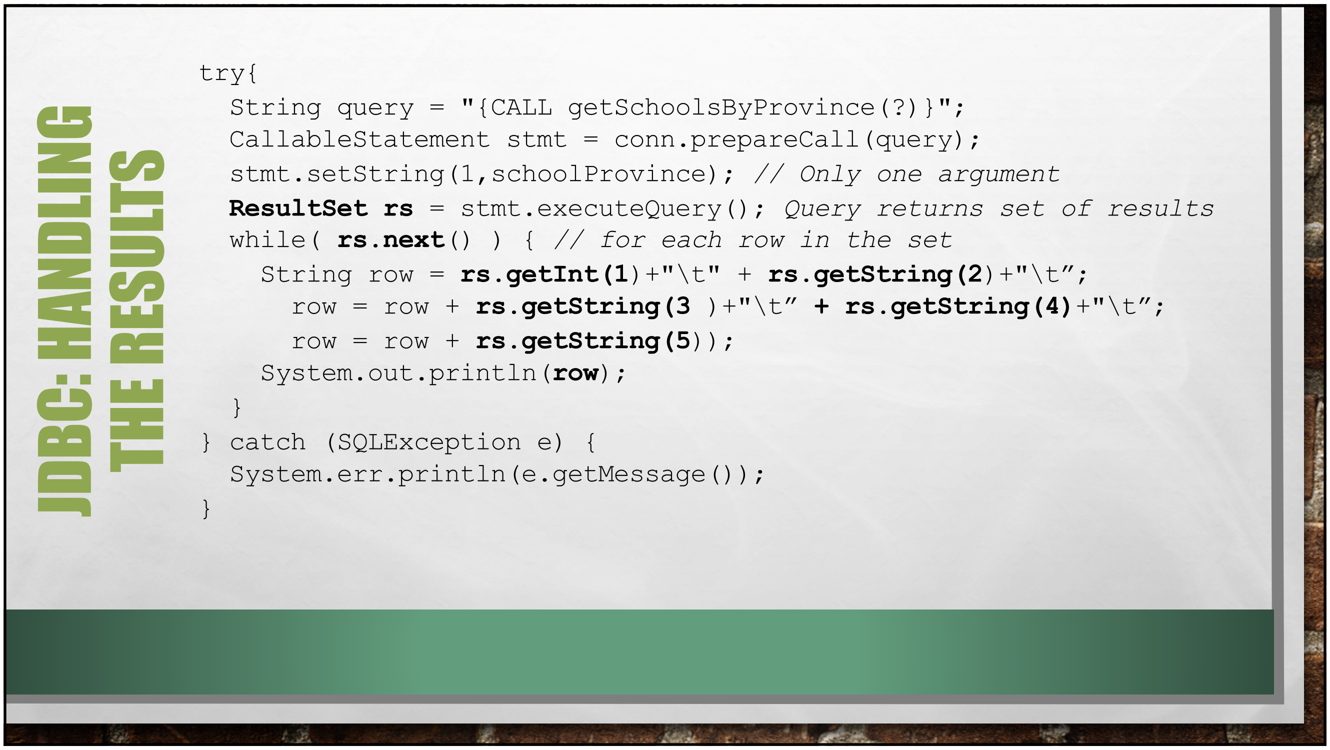
7



JDBC: HANDLING THE RESULTS (I)

- Results (if any) are returned from execution via: **ResultSet**
- Access the set of returned values:
  1. Rows are accessed using the object’s iterator
  2. Retrieve filed values into java variables

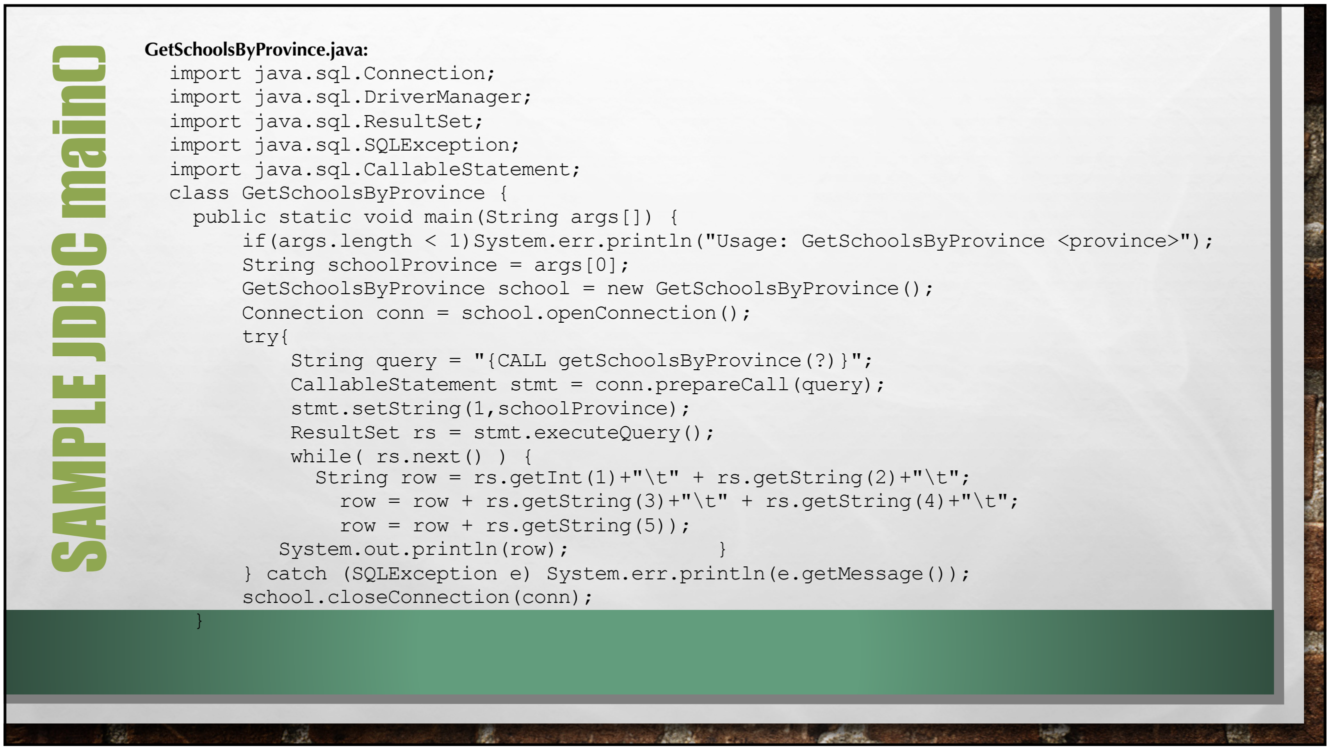
8



JDBC: HANDLING THE RESULTS

```
try{
  String query = "{CALL getSchoolsByProvince(?)}";
  CallableStatement stmt = conn.prepareCall(query);
  stmt.setString(1,schoolProvince); // Only one argument
  ResultSet rs = stmt.executeQuery(); Query returns set of results
  while( rs.next() ) { // for each row in the set
    String row = rs.getInt(1)+"\t" + rs.getString(2)+"\t”;
      row = row + rs.getString(3 )+"\t” + rs.getString(4)+"\t”;
      row = row + rs.getString(5));
    System.out.println(row);
  }
} catch (SQLException e) {
  System.err.println(e.getMessage());
}
```

9



SAMPLE JDBC main()

GetSchoolsByProvince.java:

```
  import java.sql.Connection;
  import java.sql.DriverManager;
  import java.sql.ResultSet;
  import java.sql.SQLException;
  import java.sql.CallableStatement;
  class GetSchoolsByProvince {
    public static void main(String args[]) {
        if(args.length < 1)System.err.println("Usage: GetSchoolsByProvince <province>");
        String schoolProvince = args[0];
        GetSchoolsByProvince school = new GetSchoolsByProvince();
        Connection conn = school.openConnection();
        try{
            String query = "{CALL getSchoolsByProvince(?)}";
            CallableStatement stmt = conn.prepareCall(query);
            stmt.setString(1,schoolProvince);
            ResultSet rs = stmt.executeQuery();
            while( rs.next() ) {
              String row = rs.getInt(1)+"\t" + rs.getString(2)+"\t";
                row = row + rs.getString(3)+"\t" + rs.getString(4)+"\t";
                row = row + rs.getString(5));
           System.out.println(row);            }
        } catch (SQLException e) System.err.println(e.getMessage());
        school.closeConnection(conn);
    }
```

10