*Top of Form*

*user: ialam Md.Idris Alam - 1138671*

*:*

1. *Which of the following statements loads the JDBC-ODBC driver?* 
   1. *Class.loadClass(sun.jdbc.odbc.JdbcOdbcDriver)*
   2. ***Class.forName("sun.jdbc.odbc.JdbcOdbcDriver")***
   3. *Class.forName(sun.jdbc.odbc.JdbcOdbcDriver)*
   4. *Class.loadClass("sun.jdbc.odbc.JdbcOdbcDriver")*

1. *Which of the following statements are true about CallableStatement ?* 
   1. *CallableStatement is more efficient than PreparedStatement.*
   2. ***CallableStatement is a subinterface of PreparedStatement***
   3. ***CallableStatement is for executing predefined functions and procedures.***
   4. *CallableStatement is for SQL query statements only. You cannot create a CallableStatement for SQL update statements.*

1. *You run a SELECT statement, and multiple duplicates of values are retrieved. What keyword can you use to retrieve only the non duplicate data?* 
   1. *INDIVIDUAL*
   2. *SEPARATE*
   3. *DUBPLICATE*
   4. ***DISTINCT***

1. *What information may be obtained from a ResultSetMetaData object?* 
   1. *database URL and product name*
   2. *JDBC driver name and version*
   3. ***number of columns in the result set***
   4. *number of rows in the result set*

1. *What information may be obtained from a DatabaseMetaData object?* 
   1. ***JDBC driver name and version***
   2. *maximum number of connections to the database*
   3. ***database URL and product name***
   4. *maximum table name length and maximum number of columns in a table*

1. *In a relational data model, \_\_\_\_\_\_\_\_\_ imposes constraints on the data.* 
   1. *Structure*
   2. *Language*
   3. ***Integrity***
   4. *SQL*

1. *What happens if you call the method close() on a ResultSet object?* 
   1. ***the database and JDBC resources are released***
   2. *you will get a SQLException, because only Statement objects can close ResultSets*
   3. *the ResultSet, together with the Statement which created it and the Connection from which the Statement was retrieved, will be closed and release all database and JDBC resources*
   4. *the method close() does not exist for a ResultSet. Only Connections can be closed*

1. *Which of the following statements are true about PreparedStatement?* 
   1. ***PreparedStatement is a subinterface of Statement***
   2. ***PreparedStatement is efficient for repeated executions.***
   3. ***The parameters in a prepared statement are denoted using the ? sign.***
   4. *PreparedStatement is for SQL query statements only. You cannot create a PreparedStatement for SQL update statements.*

1. *Are ResultSets updateable?* 
   1. *No, ResultSets are never updateable. You must explicitly execute DML statements (i.e.insert, delete and update) to change the data in the underlying database*
   2. *Yes, but only if you call the method openCursor() on the ResultSet, and if the driver and database support this option*
   3. ***Yes, but only if you indicate a concurrency strategy when executing the statement, and if the driver and database support this option***
   4. *Yes, but only if the ResultSet is an object of class UpdateableResultSet, and if the driver and database support this option*

1. *What statements are correct about batched insert and updates?* 
   1. ***To do a batched update/insert, you call addBatch(String statement) on a Statement object for each statement you want to execute in the batch***
   2. *To create a batch of insert and update statements, you create an object of type Batch,and call the method addStatement(String statement) for each statement you want to execute in the batch*
   3. *None of the above*
   4. *Batch insert and updates are only possible when making use of parameterized queries.*

1. *What is the return value from  
   stmt.executeUpdate("insert into T values (100, 'Smith')")* 
   1. ***an int value indicating how many rows are effected from the invocation***
   2. *void*
   3. *an object that contains the status of the execution*
   4. *a value indicating whether the SQL statement has been executed successfully*

1. *What is the function of Class.forName?* 
   1. *Both are false*
   2. *In establishing a connection is to have the appropriate driver connect to the DBMS*
   3. *Both are true*
   4. ***It is used to create an instance of a driver and register it with the DriverManager***

1. *What statements are correct about JDBC transactions* 
   1. *A transaction is a set of successfully executed statements in the database*
   2. *All of the above*
   3. *A transaction is finished when commit() or rollback() is called on the Transaction object*
   4. *A* ***transaction is finished when commit() or rollback() is called on the Connection object***

1. *In a relational data model, \_\_\_\_\_\_\_\_\_ defines the representation of the data.* 
   1. *Language*
   2. ***Structu****re*
   3. *Integrity*
   4. *SQL*

1. *Where is com.mysql.jdbc.Driver located?* 
   1. *in a JAR file ojdbc14.ja*
   2. *in a JAR file classes12.jar*
   3. ***in a JAR file mysqljdbc.jar***
   4. *in the standard Java library bundled with JDK*

1. *In your program you want to use the JDBC-ODBC Bridge driver. What code do you use?* 
   1. ***Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");***
   2. *Class.callName("sun.jdbc.odbc.JdbcOdbcDriver");*
   3. *Class.Name.init("sun.jdbc.odbc.JdbcOdbcDriver");*
   4. *Class.callfunc("JdbcOdbcDriver");*

1. *Suppose a callable statement is created as follows:  
     
   CallableStatement callableStatement = connection.prepareCall(  
   "{call sampleProcedure(?, ?, ?)}");  
     
   Assume that the first parameter is an IN parameter with value John. To set this parameter value, use* 
   1. ***callableStatement.setString(1, "John");***
   2. *callableStatement.setString(0, "John");*
   3. *callableStatement.setString(1, 'John');*
   4. *callableStatement.setString(0, 'John');*

1. *Which of the following are interfaces?* 
   1. ***ResultSet***
   2. *DriverManager*
   3. ***Connection***
   4. ***Statement***

1. *Which packages contain the JDBC classes?* 
   1. *java.jdbc and javax.jdbc*
   2. ***java.sql and javax.sql***
   3. *java.rdb and javax.rdb*
   4. *java.jdbc and java.jdbc.sql*

1. *If a prepared statement preparedStatement is a SQL SELECT statement, you execute the statement using \_\_\_\_\_\_\_\_\_.* 
   1. *preparedStatement.query();*
   2. *preparedStatement.execute();*
   3. *preparedStatement.executeUpdate();*
   4. ***preparedStatement.executeQuery();***

1. *Suppose a prepared statement is created as follows:  
     
   Statement preparedStatement = connection.prepareStatement  
   ("insert into Student (firstName, mi, lastName) " +  
   "values (?, ?, ?)");  
     
   To set a value John to the first parameter, use* 
   1. *preparedStatement.setString(0, 'John');*
   2. *preparedStatement.setString(0, "John");*
   3. *preparedStatement.setString(1, 'John');*
   4. ***preparedStatement.setString(1, "John****");*

1. *SQL \_\_\_\_\_\_\_\_ statements may change the contents of a database.* 
   1. ***SELECT***
   2. *UPDATE*
   3. *DELETE*
   4. *INSERT*

1. *Database meta data are retrieved through \_\_\_\_\_\_\_\_\_\_\_\_.   
   explanation:  
   Explanation: getMetaData() on a Connection object returns a DatabaseMetaData object.* 
   1. *a Statement object*
   2. *a PreparedStatement object*
   3. *a ResultSet Object*
   4. ***a Connection object***

1. *Suppose that your program accesses MySQL or Oracle database. Which of the following statements are true?* 
   1. ***If the driver for MySQL and Oracle are not in the classpath, the program will have a runtime error, indicating that the driver class cannot be loaded.***
   2. *If the driver for MySQL and Oracle are not in the classpath, the program will have a syntax error.*
   3. ***If the database is not available, the program will have a runtime error, when attempting to create a Connection object.***
   4. *If the database is not available, the program will have a syntax error.*

1. *What is correct about DDL statements (create, grant,...)?* 
   1. ***DDL statements are treated as normal sql statements, and are executed by calling the execute() method on a Statement (or a sub interface thereof) object***
   2. *Support for DDL statements will be a feature of a future release of JDBC*
   3. *DDL statements can not be executed by making use of JDBC, you should use the native database tools for this.*
   4. *To execute DDL statements, you have to install additional support files*

1. *To execute a SELECT statement "select \* from Address" on a Statement object stmt, use* 
   1. *stmt.executeUpdate("select \* from Address");*
   2. *stmt.query("select \* from Address");*
   3. *stmt.execute("select \* from Address");*
   4. ***stmt.executeQuery("select \* from Address");***

1. *What is, in terms of JDBC, a DataSource?* 
   1. *A DataSource is the Java representation of a physical data source*
   2. *A DataSource is the basic service for managing a set of JDBC drivers*
   3. *A DataSource is a registry point for JNDI-services*
   4. ***A DataSource is a factory of connections to a physical data source***

1. *How do you know in your Java program that a SQL warning is generated as a result of executing a SQL statement in the database?* 
   1. ***You must invoke the getWarnings() method on the Statement object (or a sub interface thereof)***
   2. *You must query the ResultSet object about possible warnings generated by the database*
   3. *You must catch the checked SQLException which is thrown by the method which executes the statement*
   4. *You must catch the unchecked SQLWarningException which is thrown by the method which executes the statement*

1. *To connect to a local MySQL database named test, use* 
   1. *Connection connection = DriverManager.connect("jdbc:mysql://localhost/test");*
   2. *Connection connection = DriverManager.getConnection(jdbc:mysql://localhost/test);*
   3. *Connection connection = DriverManager.getConnection("mysql:jdbc://localhost/test");*
   4. ***Connection connection = DriverManager.getConnection("jdbc:mysql://localhost/test");***

1. *Result set meta data are retrieved through \_\_\_\_\_\_\_\_\_\_\_\_.* 
   1. *a PreparedStatement object*
   2. ***a ResultSet Object***
   3. *a Statement object*
   4. *a Connection object*

1. *To create a statement on a Connection object conn, use* 
   1. *Statement statement = Connection.createStatement();*
   2. *Statement statement = connection.create();*
   3. ***Statement statement = conn.createStatement();***
   4. *Statement statement = conn.statement();*

1. *How can you execute a stored procedure in the database?* 
   1. ***Call method execute() on a CallableStatement object***
   2. *Call method executeProcedure() on a Statement object*
   3. *Call method run() on a ProcedureCommand object*
   4. *Call method execute() on a StoredProcedure object*

1. *How can you retrieve information from a ResultSet?* 
   1. *By invoking the method get (..., String type) on the ResultSet, where type is the database type*
   2. *By invoking the method get (..., Type type) on the ResultSet, where Type is an object which represents a database type*
   3. ***By invoking the special getter methods on the ResultSet: getString (...), get Boolean (...),getClob (...),...***
   4. *By invoking the method getValue (...), and cast the result to the desired java type*

1. *Invoking Class.forName method may throw \_\_\_\_\_\_\_\_\_\_\_.* 
   1. *SQLException*
   2. *IOException*
   3. ***ClassNotFoundException***
   4. *RuntimeException*

1. *Which statements about JDBC is true?* 
   1. ***JDBC stands for Java DataBase Connectivity***
   2. *None of the above*
   3. *JDBC is an API to bridge the object-relational mismatch between OO programs and relational databases*
   4. *JDBC is an API to connect to relational, object- and XML data sources*

1. *Suppose a callable statement is created as follows:  
     
   CallableStatement callableStatement = connection.prepareCall(  
   "{call sampleProcedure(?, ?, ?)}");  
     
   Assume that the second parameter is an OUT parameter with value John. To register this parameter, use* 
   1. *none of them*
   2. *callableStatement.registerOutParameter(0, java.sql.Types.STRING);*
   3. ***callableStatement.registerOutParameter(2, java.sql.Types.STRING);***
   4. *callableStatement.registerOutParameter(1, java.sql.Types.STRING);*

1. *How can you execute DML statements (i.e. insert, delete, update) in the database?* 
   1. *By making use of the execute(...) statement of the DataModificationStatement object*
   2. ***By invoking the execute(...) or executeUpdate(...) method of a normal Statement object or a sub-interface object thereof***
   3. *By making use of the InsertStatement, DeleteStatement or UpdateStatement classes*
   4. *By invoking the executeInsert(...), executeDelete(...) or executeUpdate(...) methods of the DataModificationStatement object*

1. *A database URL for a MySQL database named test on host panda.armstrong.edu is \_\_\_\_\_\_\_\_.* 
   1. *jdbc.mysql://panda.armstrong.edu/test*
   2. *jdbc:mysql:/panda.armstrong.edu/test*
   3. ***jdbc:mysql://panda.armstrong.edu/test***
   4. *jdbc.mysql.//panda.armstrong.edu/test*

1. *How can you start a database transaction in the database?* 
   1. ***By setting the autoCommit property of the Connection to false, and execute a statement in the database***
   2. *By asking a Transaction object to your Connection, and calling the method begin() on it*
   3. *By asking a Transaction object to your Connection, and setting the autoCommit property of the Transaction to false*
   4. *By calling the method beginTransaction() on the Connection object*

1. *Which type of Statements can execute parameterized queries?* 
   1. *All kinds of Statements (i.e. which implement a sub interface of Statement)*
   2. *ParameterizedStatement*
   3. *ParameterizedStatement and CallableStatement*
   4. ***PreparedStatement***

1. *Analyze the following code:  
   ResultSet resultSet = statement.executeQuery  
   ("select firstName, mi, lastName from Student where lastName "  
   + " = 'Smith'");  
   System.out.println(resultSet.getString(1))* 
   1. *resultSet.getString(1) returns the mi field in the result set.*
   2. *If the SQL SELECT statement returns no result, resultSet is null.*
   3. ***resultSet.getString(1) returns the firstName field in the result set.***
   4. ***The program will have a runtime error, because the cursor in resultSet does not point to a row. You must use resultSet.next() to move the cursor to the first row in the result set. Subsequently, resultSet.next() moves the cursor to the next row in the result set.***

1. *What happens if you call deleteRow() on a ResultSet object?* 
   1. ***The row you are positioned on is deleted from the ResultSet and from the database***
   2. *The row you are positioned on is deleted from the ResultSet, but not from the database*
   3. *You will get a compile error: the method does not exist because you can not delete rows from a ResultSet*
   4. *The result depends on whether the property synchonizeWithDataSource is set to true or false*

1. *What is the meaning of ResultSet.TYPE\_SCROLL\_INSENSITIVE* 
   1. *The meaning depends on the type of data source, and the type and version of the driver you use with this data source*
   2. *This means that the ResultSet is insensitive to scrolling*
   3. ***This means that the ResultSet is sensitive to scrolling, but insensitive to changes made by others***
   4. *This means that the Resultset is sensitive to scrolling, but insensitive to updates, i.e. not updateable*

1. *Which of the following statements are true?* 
   1. ***You may load multiple JDBC drivers in a program.***
   2. ***You can send queries and update statements through a Statement object.***
   3. *You may create multiple statements from one connection.*
   4. ***You may create multiple connections to a database.***

1. *Which one of the following will not get the data from the first column of ResultSet rs, returned from executing the following SQL statement: SELECT name, rank, serialNo FROM employee.?* 
   1. *rs.getString(1)*
   2. *none of them*
   3. *rs.getString(“name”)*
   4. ***rs.getString(2)***

1. *A database URL for an access database source test is \_\_\_\_\_\_\_\_.* 
   1. *sun.jdbc:odbc:test*
   2. ***jdbc:odbc:test***
   3. *jdbcodbc:test*
   4. *test*

*Bottom of Form*