64 JDBC Interview Questions and Answers - Freshers, Experienced

Next Page »

Dear Readers, Welcome to **JDBC Interview questions** with answers and explanation. These 64 solved **JDBC questions** will help you prepare for technical interviews and online selection tests conducted during campus placement for freshers and job interviews for professionals.

After reading these **tricky JDBC questions**, you can easily attempt the objective type and multiple choice type questions on JDBC.

What are the types of statements in JDBC?

JDBC API has 3 Interfaces and their key features are as follows:

- **Statement**: It is used to run simple SQL statements like select and update. Statement interfaces use for general-purpose access to your database. It is useful when you are using static SQL statements at runtime. The Statement interface cannot accept parameters.

- **PreparedStatement**: A SQL statement is pre-compiled and stored in a PreparedStatement object. It is used to run Pre compiled SQL. This object can then be used to efficiently execute this statement multiple times. The object of Prepared Statement class can be created using Connection.prepareStatement() method. This extends Statement interface.

- **CallableStatement**: This interface is used to execute the stored procedures. This extends Prepared Statement interface.

The object of Callable Statement class can be created using Connection.prepareCall() method.

What causes No suitable driver error?

"No suitable driver" occurs during a call to the DriverManager.getConnection method, may be for any of the following reasons:

- Due to failing to load the appropriate JDBC drivers before calling the getConnection method.
- It might be specifying an invalid JDBC URL, one that is not recognized by JDBC driver.
- This error can occur if one or more the shared libraries needed by the bridge cannot be loaded.

What does setAutoCommit do?

setAutoCommit() invoke the commit state query to the database. To perform batch updation we use the setAutoCommit() which enables us to execute more than one statement together, which in result minimizes the database call and sends all statement in one batch.

setAutoCommit() allows us to commit the transaction commit state manually the default values of the setAutoCommit() is true.

Why Prepared Statements are faster?

Prepared execution is faster than direct execution for statements executed more than three or four times because the statement is compiled only once. Prepared statements and JDBC driver are linked with each other. We can bind drivers with columns by triggering the query into the database. When we execute Connection.prepareStatement(), all the columns bindings take place, in order to reduce the time.

What restrictions are placed on method overriding?

The restriction on method overloading is the signature of the method.

- The signature is the number, type, and order of the arguments passed to a method.
- Overridden methods must have the same name, argument list, and return type.
- Any method which has the same name cannot have the same signature.
- They can have the same return types in the same scope.
- The compiler uses the signature to detect which overloaded method to refer when an overloaded method is called.
- If two methods have the same name and signature the compiler will throw a runtime error.

What are types of JDBC drivers?

There are four types of drivers defined by JDBC as follows:

- **JDBC/ODBC**: These require an ODBC (Open Database Connectivity) driver for the database to be installed. It is used for local connection.

- **Native API (partly-Java driver)**: This type of driver uses a database API to interact with the database. It also provides no host redirection.

- **Network Protocol Driver**: It makes use of a middle-tier between the calling program and the database. The client driver communicates with the net server using a database-independent protocol and the net server translates this protocol into database calls.

- **Native Protocol Drive**: This has a same configuration as a type 3 driver but uses a wire protocol specific to a particular vendor and hence can access only that vendor's database.

Is it possible to connect to multiple databases simultaneously? Using single statement can

one update or extract data from multiple databases?

Yes, it is possible but it depends upon the capabilities of the specific driver implementation, we can connect to multiple databases at the same time. We perform the following steps:

- Minimum one driver will be used to handle the commits transaction for multiple connections.
- To update and extract data from the different database we use single statement. For this we need special middleware to deal with multiple databases in a single statement or to effectively treat them as one database.

What are the differences between setMaxRows(int) and SetFetchSize(int)?

The difference between setFetchSize and setMaxRow are:

- setFetchSize(int) defines the number of rows that will be read from the database when the ResultSet needs more rows whereas setMaxRows(int) method of the ResultSet specifies how many rows a ResultSet can contain at a time.

- In setFetchSize(int), method in the java.sql.Statement interface will set the 'default' value for all the ResultSet derived from that Statement whereas in setMaxRow(int) default value is 0, i.e. all rows will be included in the ResultSet.

- The setMaxRows affects the client side JDBC object while the setFetchSize affects how the database returns the ResultSet data.

How can I manage special characters when I execute an INSERT query?

The special characters meaning in SQL can be preceded with a special escape character in strings, e.g. "\". In order to specify the escape character used to quote these characters, include the following **syntax** on the end of the query:

```
{escape 'escape-character'}
```

For example, the query

```
SELECT NAME FROM IDENTIFIERS WHERE ID LIKE '\_%'
{escape '\'}
finds identifier names that begin with an underscore.
```

What is the benefit of having JdbcRowSet implementation? Why do we need a JdbcRowSet like wrapper around ResultSet?

The JdbcRowSet implementation is a wrapper around a ResultSet object has following advantages over ResultSet:

- It makes possible to use the ResultSet object as a JavaBeans component.
- A JdbcRowSet can be used as a JavaBeans component, thus it can be created and configured at design time and executed at run time.
- It can be used to make a ResultSet object scrollable and updatable. All RowSet objects are by default scrollable and updatable.

Explain basic steps in writing a Java program using JDBC.

JDBC makes the interaction with RDBMS simple and intuitive. When a Java application needs to access database :

- Load the RDBMS specific JDBC driver because this driver actually communicates with the database.
- Open the connection to database, for sending SQL statements and get results back.
- Create JDBC Statement object containing SQL query.
- Execute statement which returns result set. ResultSet contains the tuples of database table as a result of SQL query.

- Process the result set.
- Close the connection.

I have the choice of manipulating database data using a byte[] or a java.sql.Blob. Which has best performance?

We use java.sql.Blob, because of following reason:

- It does not extract any data from the database until we trigger a query to the database.
- We use byte[] for inserting data in the database when data is not upload in the database till yet.
- java.sql.Blob is used when extraction of the data is performed.

What are DML and DDL?

Data Manipulation Language (DDL) this portion of the SQL standard is concerned with manipulating the data in a database as opposed to the structure of a database. The DML deals with the SELECT, INSERT, DELETE, UPDATE, COMMIT and ROLLBACK.

Data Definition Language (DDL) this portion of the SQL standard is concerned with the creation, deletion and modification of database objects like tables, indexes and views. The core verbs for DDL are CREATE, ALTER and DROP. While most DBMS engines allow DDL to be used dynamically, it is often not supported in transactions.

How can you load the drivers?

It is very simple and involves just one line of code to load the driver or drivers we want to use.

For example, We want to use the JDBC-ODBC Bridge driver, the following code will load it:

```
Class.forName(”sun.jdbc.odbc.JdbcOdbcDriver”);
```

Driver documentation will give you the class name to use. For instance, if the class name is jdbc.DriverHELLO, you would load the driver with the following line of code:

```
Class.forName(”jdbc.DriverHELLO”);
```

How do I insert an image file (or other raw data) into a database?

All raw data types should be read and uploaded to the database as an array of bytes, byte[].

- Originating from a binary file.
- Read all data from the file using a FileInputStream.
- Create a byte array from the read data.
- Use method setBytes(int index, byte[] data); of java.sql.PreparedStatement to upload the data.

Discuss the significances of JDBC.

The significances of JDBC are given below:

- JDBC stands for Java Database Connectivity.

- Java Database Connectivity (JDBC) is a standard Java API .

- Its purpose is to interact with the relational databases in Java.

- JDBC is having a set of classes & interfaces which can be used from any Java application.

- By using the Database Specific JDBC drivers, it interacts with a database without the applications of RDBMS.

Name the new features added in JDBC 4.0.

The major features introduced in JDBC 4.0 are :

- Auto-loading by JDBC driver class.

- Enhanced Connection management

- RowId SQL enabled.

- DataSet implemented by SQL by using Annotations

- Enhancements of SQL exception handling

- Supporting SQL XML files.

How do Java applications access the database using JDBC?

Java applications access the database using JDBC by :

- Communicating with the database for Loading the RDBMS specific JDBC driver

- Opening the connection with database

- Sending the SQL statements and get the results back.

- Creating JDBC Statement object which contains SQL query.

- Executing statement to return the Resultset(s) containing the tuples of database table which is a result of SQL query.

- Processing the result set.

- Closing the connection.

Briefly tell about the JDBC Architecture.

The JDBC Architecture consists of two layers:

1.The JDBC API
2.The JDBC Driver API

- The JDBC API provides the application-JDBC Manager connection.

- The JDBC Driver API supports the JDBC Manager-to-Driver Connection.

- The JDBC API interacts with a driver manager, database-specific driver for providing transparent connectivity for the heterogeneous databases.

- The JDBC driver manager authenticates that the correct driver has been used to access each data source.

- The driver manager supports multiple concurrent drivers connected to the multiple heterogeneous databases.

Explain the life cycle of JDBC.

The life cycle for a servlet comprises of the following phases:

- **DriverManager**: for managing a list of database drivers.

- **Driver** : for communicating with the database.

- **Connection** : for interfacing with all the methods for connecting a database.

- **Statement** : for encapsulating an SQL statement for passing to the database which had been parsed, compiled, planned and executed.

- **ResultSet**: for representing a set of rows retrieved for the query execution.

Describe how the JDBC application works.

A JDBC application may be divided into two layers:

- Driver layer

- Application layer

- The Driver layer consists of DriverManager class & the JDBC drivers.

- The Application layer begins after putting a request to the DriverManager for the connection.

- An appropriate driver is chosen and used for establishing the connection.

- This connection is linked to the application layer.

- The application needs the connection for creating the Statement kind of objects by which the results are obtained.

How a database driver can be loaded with JDBC 4.0 / Java 6?

- By providing the JAR file, the driver must be properly configured.

- The JAR file is placed in the classpath.

- It is not necessary to explicitly load the JDBC drivers by using the code like Class.forName() to register in the JDBC driver.

- The DriverManager class looks after this, via locating a suitable driver at the time when the DriverManager.getConnection() method is called.

- This feature provides backward-compatibility, so no change is needed in the existing JDBC code.

What does the JDBC Driver interface do?

- The JDBC Driver interface provides vendor-specific customized implementations of the abstract classes.

- It is provided normally by the JDBC API.

- For each vendor the driver provides implementations of the java.sql.Connection, PreparedStatement, Driver, Statement, ResultSet and CallableStatement.

What is represented by the connection object?

- The connection object represents the communication context.

- All the communication with the database is executed via the connection objects only.

- Connection objects are used as the main linking elements.

What is a Statement?

- The Statement acts just like a vehicle via which SQL commands are sent.

- By the connection objects, we create the Statement kind of objects.

```
Statement stmt = conn.createStatement();
```

- This method returns the object, which implements the Statement interface.

Define PreparedStatement.

- A PreparedStatement is an SQL statement which is precompiled by the database.

- By precompilation, the PreparedStatement improves the performance of the SQL commands that are executed multiple times (given that the database supports PreparedStatement).

- After compilation, PreparedStatement may be customized before every execution by the alteration of predefined SQL parameters.

Code:

```
PreparedStatement pstmt = conn.prepareStatement("UPDATE
data= ? WHERE vl = ?");
pstmt.setBigDecimal(1, 1200.00);
pstmt.setInt(2, 192);
```

Differentiate between a Statement and a PreparedStatement.

- A standard Statement is used for creating a Java representation for a literal SQL statement and for executing it on the database.

- A PreparedStatement is a precompiled Statement.

- A Statement has to verify its metadata in the database every time.

- But, the PreparedStatement has to verify its metadata in the database only once.

- If we execute the SQL statement, it will go to the STATEMENT.

- But, if we want to execute a single SQL statement for the multiple number of times, it'll go to the PreparedStatement.

What is the function of setAutoCommit?

- When a connection is created, it is in auto-commit mode.

- This means that each individual SQL statement is to be treated as a single transaction.

- The setAutoCommit will be automatically committed just after getting executed.

- The way by which two or more statements are clubbed into a transaction to disable the auto-commit mode is :

```
con.setAutoCommit (false);
```

- Once auto-commit mode is disabled, no SQL statements will be committed until we call the method ‘commit’ explicitly.

Code :

```
con.setAutoCommit(false);
PreparedStatement updateSales = con.prepareStatement(
"UPDATE COFFEE SALES = ? WHERE COF_NAME LIKE ?");
updateSales.setInt(1, 50); updateSales.setString(2, "Colombian");
updateSales.executeUpdate();
PreparedStatement updateTotal =
con.prepareStatement("UPDATE COFFEES SET TOTAL = TOTAL
+ ? WHERE COF_NAME LIKE ?");
updateTotal.setInt(1, 50);
updateTotal.setString(2, "Colombian");
updateTotal.executeUpdate();
con.commit();
con.setAutoCommit(true);
```

How do we call a stored procedure from JDBC?

- The foremost step is to create a CallableStatement object.

- With the Statement and PreparedStatement object, it is done with an open Connection object.

- A CallableStatement object contains a call to a stored procedure.

Code:

```
CallableStatement cs = con.prepareCall("{call SHOW_Sales}");
ResultSet rs = cs.executeQuery();
```

What is SQLWarning and discuss the procedure of retrieving warnings?

- SQLWarning objects, a subclass of SQLException is responsible for the database access warnings.

- Warnings will not stop the execution of an specific application, as exceptions do.

- It simply alerts the user that something did not happen as planned.

- A warning may be reported on the Connection object, the Statement object (including PreparedStatement and CallableStatement objects) or on the ResultSet object.

- Each of these classes has a getWarnings method, which you must invoke in order to see the first warning reported on the calling object:

Code :

```
SQLWarning warning = stmt.getWarnings();
if (warning != null)
{
  System.out.println("n---Warning---n");
  while (warning != null)
  {
     System.out.println("Message: " + warning.getMessage());
     System.out.println("SQLState: " + warning.getSQLState());
     System.out.println("Vendor error code: ");
     System.out.println(warning.getErrorCode());
     System.out.println("");
     warning = warning.getNextWarning();
  }
}
```

How can we move the cursor in a scrollable result set?

- The new features added in the JDBC 2.0 API are able to move a resultset’s cursor backward & forward also.

- There are some methods that let you direct the cursor to a particular row and checking the position of the cursor.

Code :

```
Statement stmt =
con.createStatement(ResultSet.TYPE_SCROLL_SENSITIVE,
ResultSet.CONCUR_READ_ONLY);
ResultSet srs = stmt.executeQuery(”SELECT COF_NAME, Sale
_COFFEE”);
```

- Three constants can be added to the ResultSet API for indicating the kind of the ResultSet object. The constants are:

a. TYPE_FORWARD_ONLY
b. TYPE_SCROLL_INSENSITIVE
c. TYPE_SCROLL_SENSITIVE.

- The ResultSet constants for specifying whether a resultset is read-only or updatable are:

a. CONCUR_READ_ONLY
b. CONCUR_UPDATABLE.

How do we load the drivers?

- To Load the driver or drivers we need to use a very simple one line of code.

- If we want to use the JDBC/ODBC Bridge driver, the following code will load it:

```
Class.forName(”sun.jdbc.odbc.JdbcOdbcDriver”);
```

- The driver documentation gives the class name to use.

- **For example**, if the class name is jdbc.DriverXYZ, we can load the driver using the below line of code:

Code:

```
Class.forName(”jdbc.DriverXYZ”);
```

What Class.forName does, while loading the drivers?

- It is used for creating an instance of a driver
- It is used for registering with the DriverManager.
- When we have loaded a driver, it connects with the DBMS.

How can you make a connection?

- To establish a connection we need to have an appropriate driver, connected to the DBMS.

- The below line of code illustrates the idea:

Code:

```
String url = “jdbc:odbc: rima”;
Connection con = DriverManager.getConnection(url, “rima”, “J8");
```

What are the factors that the JDBC driver performance depends upon?

The JDBC driver performance depends upon:

- The driver code quality
- The driver code size
- The database server & its load capability
- The network topology
- The number of times the request is being translated to a different API.

How do I find whether a parameter exists in the request object?

- **The following code implies it**

```
boolean hasFo = !(request.getParameter("fo") == null ||
request.getParameter("fo").equals(""));
```

```
Or

boolean hasParameter =
request.getParameterMap().contains(theParameter);
```

Expalin the method of calling a stored procedure from JDBC.

- PL/SQL stored procedures are called from the JDBC programs by creating the prepareCall() in the Connection object.

- A call to the method prepareCall() carries out variable bind parameters as input parameters, output variables & makes an object instance of the CallableStatement class.

- **The following line of code implies this** :

```
Callable Statement stproc_ stmt = conn.prepareCall("{call
procname(?,?,?)}");
```

where conn is the instance of the Connection class.

Name the types of JDBC drivers.

The four types of drivers defined by JDBC are:

Type 1: JDBC/ODBC - This requires an ODBC (Open Database Connectivity) driver for the databases to be installed. This type of drivers work by converting the submitted queries into an equivalent ODBC queries and forwarding them via native API which invokes directly to the ODBC driver. It provides host less redirection capability too.

Type 2: Native type API (partly-Java driver) - This type of driver uses a vendor-specific driver or database API for interacting with the database. An example of such an API is Oracle OCI (Oracle Call Interface).

Type 3: Open Net Protocol - This is vendor non-specific and works by forwarding database requests using a net server component. The net server accesses the database. The client driver connects with the server using a database-indifferent protocol and the server translates this protocol into database calls.

Type 4: Proprietary Protocol-Net(pure Java driver) -This is same as per configuration as type 3 driver while it uses a wire protocol directed towards a particular vendor and so it can access only that vendor's database.

Explain how to Make Updates to the Updatable ResultSets.

- The JDBC 2.0 API can update rows in a ResultSet using the methods in the Java rather than using a SQL command.

- But before doing that, we create a ResultSet object which is updatable.

- For doing this, we give the ResultSet CONCUR_UPDATABLE in the createStatement method.

Code:

```
Connection con =
DriverManager.getConnection("jdbc:mySubprotocol:mySubName");
Statement stmt =
con.createStatement(ResultSet.TYPE_SCROLL_SENSITIVE,
ResultSet.CONCUR_UPDATABLE);
ResultSet uprs = stmt.executeQuery("SELECT COF_NAME,
PRICE ");
```

What are the utilities of the CallableStatement?

- CallableStatement is mainly used in the JDBC applications.

- CallableStatement is used to invoke stored procedures.

- This is mainly used in functions.

Differentiate between TYPE_SCROLL_INSENSITIVE and TYPE_SCROLL_SENSITIVE.

- We will get a scrollable ResultSet object if we specify either one of the ResultSet constants.

- The difference between the two depends on, whether a resultset is showing fv changes or not.

- This difference depends on certain methods which are called to detect changes or not.

- The resultset TYPE_SCROLL_INSENSITIVE does not show the change to it but the ResultSet srs = TYPE_SCROLL_SENSITIVE will show the change.

The following code explains the difference :

```
Statement stmt =
con.createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,
ResultSet.CONCUR_READ_ONLY);

ResultSet srs = stmt.executeQuery("SELECT COF_NAME, PRICE
FROM COFFEES");
srs.afterLast();
while (srs.previous())
{
  String name = srs.getString("COF_NAME");
  float price 1= srs.getFloat("PRICE");
  System.out.println(name + " " + price1);
}
```

How are JDBC Statements used?

- A Statement is an object, by which we send an SQL statement to the DBMS.

- We create a Statement object and then execute it.

- For a SELECT Statement, the method used is executeQuery.

- The Statement that creates or modifies a table is executeUpdate.

- For creating a Statement object an instance of an active connection is required.

- In the following code, we use our Connection object con to create the Statement object

```
Statement stmt = con.createStatement();
```

How can we retrieve data from the ResultSet?

- JDBC returns back the results in a ResultSet object.

- So we have to declare an instance of the class ResultSet to keep the results.

- The following **code** declares the ResultSet object rs.

```
ResultSet rs = stmt.executeQuery (”SELECT COF_NAME, PRICE
FROM COFFEES”);
String s = rs.getString (”NAME”);
```

- The method getString calls ResultSet object rs, So getString() retrieves the value stored in the column NAME in the row Ssrs.

What are the different types of Statements? How we can you use PreparedStatement.

The different types of Statements are

- Regular statement (uses createStatement method)
- PreparedStatement (uses prepareStatement method)
- CallableStatement (uses prepareCall)

PreparedStatement :

- PreparedStatement is derived from the class Statement.

- To execute a Statement object several times a PreparedStatement object is required.

- The PreparedStatement object contains not only a compiled SQL statement, but also a precompiled SQL statement too.

Code:

```
PreparedStatement updateSales
=con.prepareStatement("UPDATE account = ? WHERE
CON_NAME ?");
```

Explain the two tier and three tier architecture of JDBC.

Two tier and three tier are structures supported by JDBC API info accessing database.
The Java applet can directly interact with the database with help of JDBC driver which has the ability to communicate with data base management system which is being accessed under the two tier model. Under three tier model the middle tier gets a role to play. Commands are forwarded to SQL statements via the middle tier. Then the SQL statement is processes by the database and sent to user via the middle tier again.

When do we use execute method in Java JDBC?

When a statement has a possibility to return more than one result set object then execute method is put to use. It is also used when the statement returns more than one update count or a couple of result objects which update count. These situations occur rarely when store procedures are being executed or unknown sql string is executed dynamically. For instance, a stored procedure might be executed with the use of callable statement object and in result the procedure might update, after which it selects and follows the same pattern. In case, of stored procedure the value to be returned will be known.

What does JDBC do?

JDBC can be used to perform the following functions:

1. **Establishing connection with a database:** It establishes the connection to the database that is used inside JAVA only and it allows the drivers to be defined such that it can be accessed from anywhere. The JDBC drivers are used for connectivity with other databases as well using the different methodology and functions.

2. **Sending of SQL statements:** This allow sending of queries and request to get the data from the database. The SQL statement provides a way to allow user in a convenient manner to produce the statements.

3. **Processing of results:** this allow easy processing of the result that is processed after getting the request from the client side.

What are the higher level APIs under development on top of JDBC currently?

There are two higher level APIs that fall under development on top of JDBC:

1. SQL which act as an embedder for Java. This is planned by at least one vendor. SQL is implemented by DBMS which is a language specially designed for operation with database. It is required by JDBC that statements of SQL are passing to methods of java. The programmer is allowed to mix SQL statements with Java by the embedded SQL pre-processor.

2. Relational database tables are directly mapped to Java classes. There are plans declared for this by Java soft and some other companies for its implementation. Under this form of mapping every row of the table is converted into instance for the class and the columns are converted intro attributes for every instance.

Compare JDBC and ODBC and how is JDBC required in this context.

To draw a comparison between jdbc and odbc lets us consider the following points:

1. ODBC is inappropriate for directly using it from Java as it uses C interface.
The calling from java to c faces a number of drawbacks under security concern and various other reasons.

2. Translating of ODBC C API into a java API is not desirable. This ODBC acts as a bridge between different databases and the connectivity. This provides better connectivity to the databases that are not compatible with the JDBC drivers and database.

3. Odbc is difficult to learn as it mixes advanced features making it complex.

4. Jdbc thus is required for enabling pure java solution.

What is Java soft framework?

There are 3 products components of JDBC under Java soft framework. The component allows the driver to manage properly and be responsible for the connection and communication between the java applications. It includes the right JDBC driver and isolates it from the rest to perform the functions.

- **JDBC driver manager:** it manages all the drivers that are present in the Java directory and it allows the driver to be present at the time it is required.

JDBC driver test suite: this consists of the test suite cases that are required to test a particular driver for its functionality of being accepted by the community and implemented for the use.

JDBC-ODBC Bridge: this is the bridge driver that connects other than JDBC drivers. This allows the databases to be connected like MS Access with Java database.

What are the different types of drivers under JDBC?

There are four types of JDBC drivers which are as follows:

1. **JDBC-ODBC bridge plus ODBC driver**: It is responsible for providing access to JDBC via ODBC.

2. **Native-API partly-Java driver**: It is responsible for calling into client calls on API for difference system bases.

3. **JDBC-Net pure Java driver**: It is responsible for translating JDBC calls into DBMS net independent protocol which is further translated into DBMS protocol by a server.

4. **Native-protocol pure Java driver**: It is responsible for converting JDBC calls to network protocol which is used directly by DBMS.

What are different parts of a URL?

A URL consists of three parts:

1. **Protocol for accessing the information** - Examples for this are file transfer protocol, hyper text transfer protocol etc. Note that a colon is placed after a protocol.

```
ftp://javasoft.com/docs/JDK-1_apidocs.zip
http://java.sun.com/products/JDK/1.1
```

2. **Information about the host** - This gives the information related to the host which tells us where the resource resides. It begins with a double slash or single slash depending on whether it is an Internet application or not and always ends with a single slash.

3. **Path of access**: In the following example, products and JDK

are directories, and 1.0.2 is a file. This URL gives the location of the Java Developer’s Kit, version 1.0.2:

```
http://java.sun.com/products/JDK/1.0.2
```

What are the three parts of a JDBC URL?

A JDBC URL consists of three parts, which are separated by colons:

1. **Protocol:** It is always JDBC.

2. **Subprotocol:** It is the mechanism of driver or database connectivity which might be supported by 1 or more drivers. Example of a sub-protocol name is ODBC, to access a database through a JDBC-ODBC bridge, one might use a URL such as the following:

```
jdbc:odbc:sid
```

In this example, the sub-protocol is ODBC, and the sub-name sid is a local ODBC data source.

3. **Sub-name:** It is the way by which database is identified. It can vary and have sub name depending on the protocol.

What are the types of JDBC drivers that exist?

There are four types of JDBC drivers that exist and these drivers are used for relational database servers. The drivers are:

- **Type 1:** is used for the native code driver that is used for the ODBC driver and allows the connection can be made to different databases.

- **Type 2:** is a vendor specific database type that is used for the library on the client side. The script is written that allow the communication to be done over the network.

- **Type 3:** uses pure-java driver that allows the server-side middleware to be connected to the database and communicate with the Java components that are used in the application.

- **Type 4:** these are the drivers that are used for the native protocol database.

What are the classes and methods used for sending SQL statements to database?

There are 3 classes provided to send SQL statements to database and to each class relates a method. The classes and methods for creating these are as follows:

1. **Statement**: It is used for sending simple SQL statements and is created by method create statement.

2. **PreparedStatement**: It is used for SQL statements which take one or more parameters as input arguments and is created by PreparedStatement.

3. **CallableStatement**: It is used for executing SQL stored procedures and is created by method prepare call.

State the different connection methods used for creating different types of sql.

Following is the list of different types of SQL statements and which connection method is appropriate for creating them:
The create Statement method is used for the following

1. Simple SQL statements (no parameters). prepareStatement method is used for

2. SQL statements with one or more IN parameters.

3. Simple SQL statements that are executed frequently.
The prepare Call method is used for the following
1. call to stored procedures

In which ways is driver class is registered with drive manager?

The driver class is automatically registered with drive manager in two ways when loaded which are as follows:

1. Calling method class forName. This loads the driver class explicitly. As there is no dependency on external setup this way is recommended for loading driver.
The following code is used for loading class acme.db.Driver:
Class.forName("acme.db.Driver");

2. Adding driver to the java.lang.System property jdbc.drivers. It is a list of driver classnames, separated by colons. The DriverManager class loads this. When the DriverManager class is intialized, it looks for the system property jdbc.drivers, and if the user has entered one or more drivers, the DriverManager class attempts to load them. The loads using the system properties can be done on the startup using the following command:

```
jdbc.drivers=foo.bah.Driver:wombat.sql.Driver:bad.test.ourDriver;
```

Give an example of code used for setting up connection with a driver.

Following is an example of code used normally for setting up connection with driver such as a JDBC-ODBC bridge driver:

```
Class.forName("sun.jdbc.odbc.JdbcOdbcDriver"); //loads the driver
String url = "jdbc:odbc:sid";
DriverManager.getConnection(url, "userID", "passwd");
```

Explain creation of statement object with connection method create method with help of an example.

Statement object is created with connection method create statement in the following way :

```
Conctn con = DriverManager.getConnection(url, "sid", "");
Statement stmt = con.createStatement();
```

Sql statement sent to database is supplied as an argument to 1 of the methods for execution of a statement object :

```
ResultSet rs = stmt.executeQuery(“SELECT a, b, c FROM Table2);
```

When does the result for an sql statement return null?

This happens when result is not a ResultSet for an sql statement and the method getResultSet will return null. This implies that the result is an update count or that there are no more results.

The way to find out what the null really means in this case is by calling the method getUpdateCount, which will return an integer. This integer shall be the number of rows affected by the calling statement or -1 indicating either that the result is a result set or that there are no results.

When method getResultSet has already returned null, it means that the result is not a
ResultSet object, and then a return value of -1 has to mean that there are no more results. In simple words, there are no results (or no more results) when the following is true:

```
((stmt.getResultSet() == null) && (stmt.getUpdateCount() == -1))
```

Give a way to check that all result sets have bin accessed and update counts are generated by execute method.

This can be demonstrated by the following way to be sure that one has accessed all the result sets and update counts generated by a call to the method execute:

```
stmt.execute(queryStringWithUnknownResults);
while(true)
{
    int rowCount = stmt.getUpdateCount();
    if(rowCount > 0)
    {
        System.out.println("Rows changed = " + count);
        stmt.getMoreResults();
        continue;
    }
    if(rowCount = 0)
    {
        System.out.println(" No rows changed or statement was DDL
command");
        stmt.getMoreResults();
        continue;
    }
        ResultSet rs = stmt.getResultSet;
        if(rs != null)
        {
            while(rs.next())
            {
                stmt.getMoreResults();
                continue;
            }
            break;
        }
}
```

What is an escape syntax?

Escape syntax is used for signaling the driver that the code within it should be handled differently. The escape clause is demarcated by curly braces and a key word:
{keyword . . . parameters . . . }
Keyword is used for indicating the kind of escape clause, as shown below.

```
Escape for LIKE escape characters
fn for scalar functions
d, t, and ts for date and time literals
call or ? = call for stored procedures
oj for outer joins
```

Give an example for execution of SQL statement.

The following code can be taken to understand the execution of a SQL statement

```
java.sql.Statement stmt = conn.createStatement();
ResltSet r = stmt.executeQuery("SELECT a, b, c FROM Table1");
while (r.next())
{
  int i = r.getInt("a");
  String s = r.getString("b");
  float f = r.getFloat("c");
  System.out.println("ROW = " + i + " " + s + " " + f);
}
```

Name different methods for getting streams.

JDBC API consists of three methods for getting streams, each with a different return value:

1. **getBinaryStream** returns a stream that simply provides the raw bytes from the database without any conversion.

2. **getAsciiStream** returns a stream that provides one-byte ASCII characters.

3. **getUnicodeStream** returns a stream that provides two-byte Unicode characters.

Next Page »

Java packages - JDBC classes, interfaces, Java.SQL and Javax.SQL

Java packages - java.sql: java.sql is an API to access and process the data stored in a database, typically a relational database using the java...

How to open a database connection using JDBC

How to open a database connection using JDBC - Opening a database connection: The database connection should be established after registering the drivers. The getConnection is used to open a database connection. The following code snippet illustrates this:...

How to send SQL statements to databases for execution?

JDBC : How to send SQL statements to databases for execution? - The following illustrates with examples, to send SQL statements to databases for execution:...

Post your comment

Discussion Board

correction in required in content

http://careerride.com/Interview-Questions-Java-JDBC.aspx

correction is required here------>

What are the types of statements in JDBC?
JDBC API has 3 Interfaces and their key features of these are as follows:
Statement: which is used to run simple SQL statements like select and update. The object of ****Statement class**** can be created using Connection.createStatement() method.

here while giving the demonstration of "java.sql.Statement" might be mistakenly you have written Statement Class ......while it is interface..
So i humbly request you to plz modify this web page content so that other readers may not misguide.
dipanshu 12-21-2012

Related Content

- Core Java - Part 1
- Core Java - Part 2
- Core Java for Freshers
- Core Java for Experienced
- Java Classes
- Java Variables
- Overloading & Overriding
- Abstract classes & interfaces
- Java Data Types
- Java Arrays

Java Exception

Java Events

Java Virtual Machine

Java Input and Output

Java URL Connections

Java Sockets

JNDI

Applets

Java Abstract Window Toolkit

Java Drawing AWT components

JDBC - Part 1

JDBC - Part 2

Java Threading

Java multithreading

Javascript - Part 1

Javascript - Part 2

Java Design Patterns

Java Localization

Core Java Tutorial