Automation Testing

Selenium with JDBC

A Day in the Life of an Automation Test Engineer

Joel is an Automation Engineer who wants to check the database performance of his most recent project.

Automation Test Engineers typically interact with various databases to retrieve data, and use the data for processing, or to add assertions to test cases. Joel's task is to fetch the data and use the data to check the performance of the database.

To know more about it, let us go through the lesson.

Learning Objectives

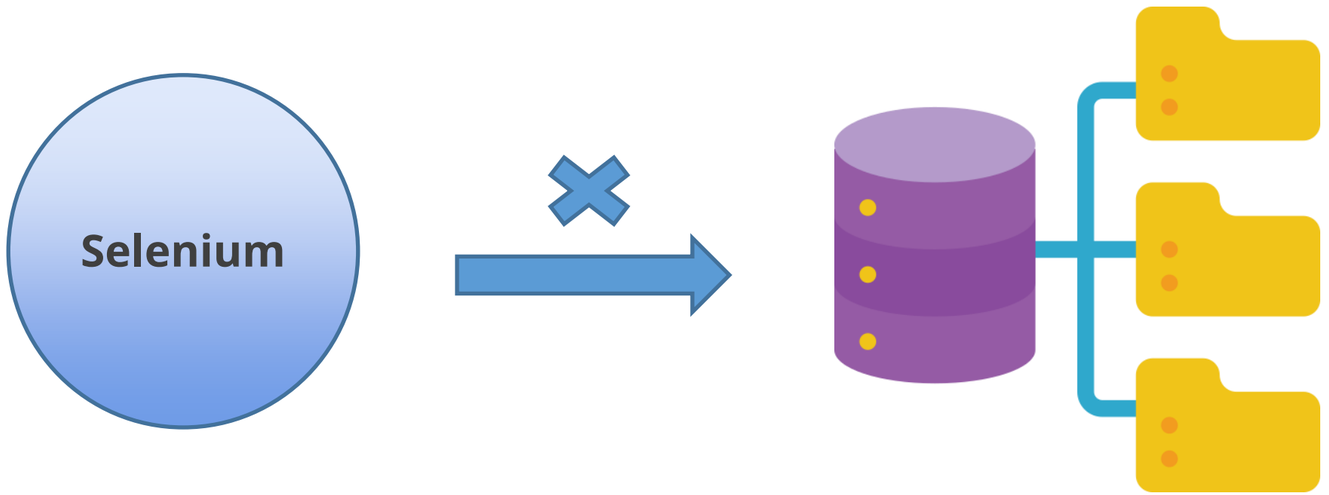
By the end of this lesson, you will be able to :

- Analyze the Selenium with JDBC
- Define the importance of JDBC
- Discuss the applications of JDBC

Why Selenium With JDBC?

Why Selenium With JDBC?

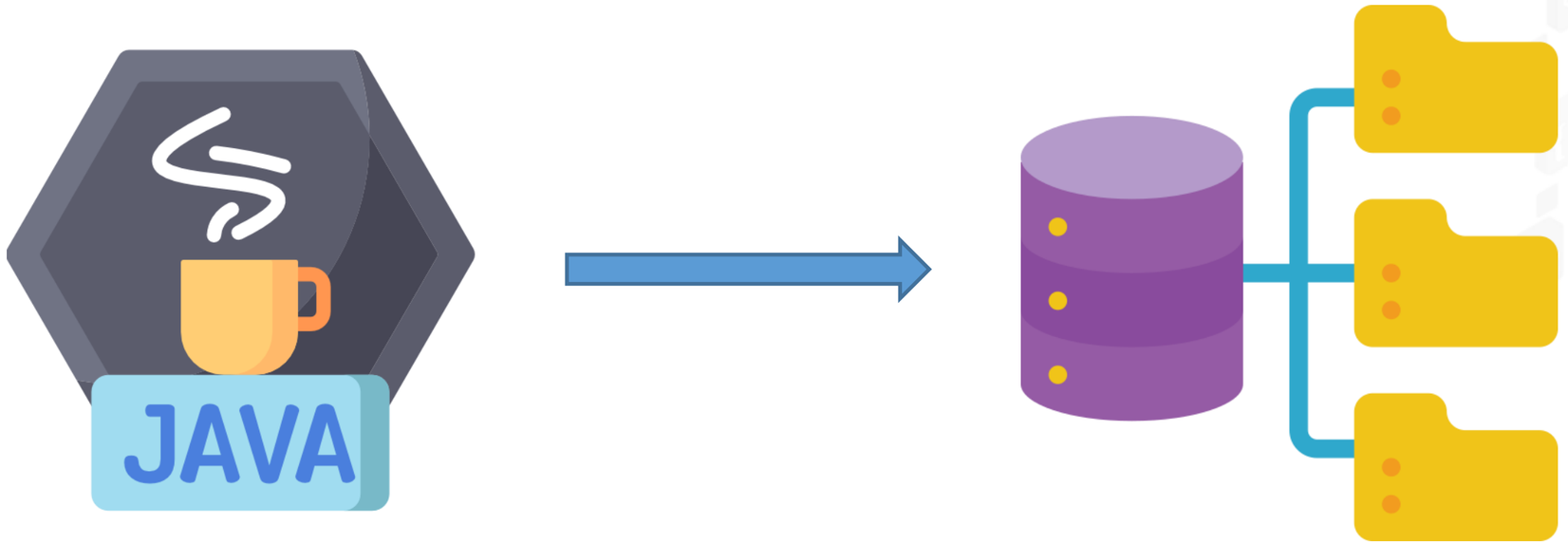
During web automation testing, the user often interacts with databases to retrieve data for use during the call flows or to add assertions.



In Selenium, there is no built-in functionality for interacting with databases.

JDBC

The Java Database Connectivity allows users to execute SQL statements using a SQL-API interface.



It connects the Java programming language to a wide range of databases.

JDBC

The JDBC API provides the following classes and interfaces:

- Driver Manager
- Connection
- Result Set
- Driver
- Statement
- SQL Exception

JDBC Importance and Applications

Why JDBC?

The JDBC API allows Java programs to handle databases and perform the following operations:

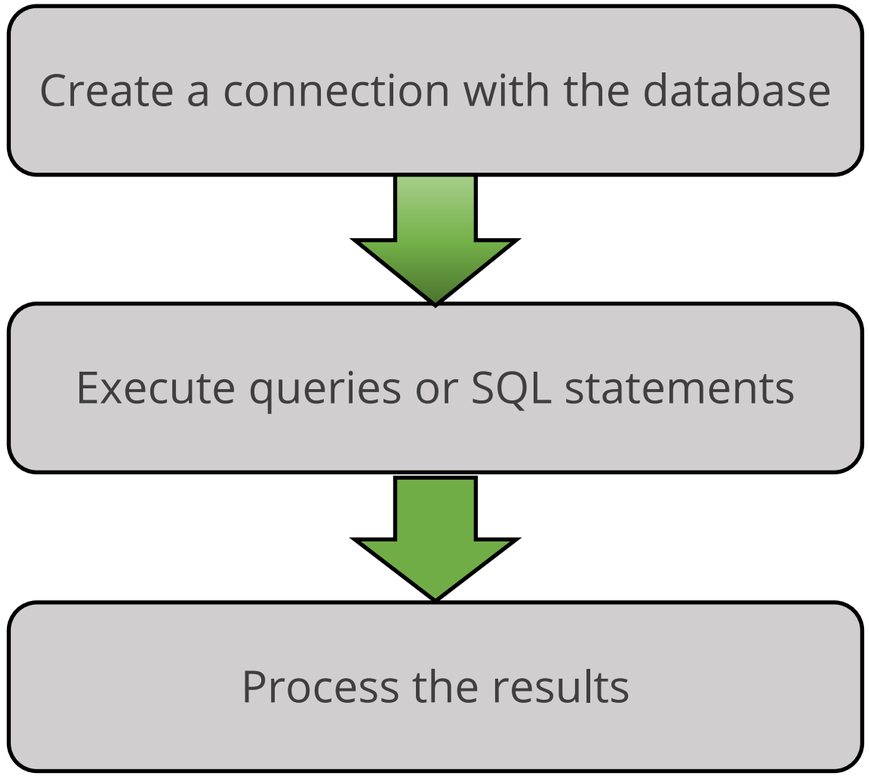
01 Connect to the database

02 Execute select queries or update statements

03 Retrieve the database result

Java Database Connectivity

To test the database with Selenium, the user should follow the following steps:



Database Connection

To connect to the database, the syntax is as follows:

Example:

```
Connection connection = DriverManager.getConnection(URL,
“userid”, “password” )
```

- “userid” is the username to access the database.
- “password” is the password for the above user.
- “URL” is of format jdbc:<dbtype>://<ipaddress>:<portnumber>/db_name
- <dbtype> is the driver for the database you are trying to connect. To connect to the Oracle database this value will be “Oracle”.

Database Connection

Connecting to the database with the name "emp" in the MYSQL URL would look like this:

Example:

```
jdbc:mysql://localhost:3036/emp
```

SQL Queries to the Database

Once the connection has been established, the next step is to create the query.

Example:

```
Statement statement = con.createStatement("select *  from
employee;");
```

SQL Queries to the Database

Once the connection has been established, the next step is to execute the query.

Example:

```
Result = statement.executeQuery();
```

Process the Results

The Result Set object holds the results of the executed query.

There are many advanced methods available in Java for processing results.

Process the Results

The following are some of the most used methods:

01 beforeFirst()

02 afterLast()

03 first()

04 last()

05 absolute(intRow)

06 relative(intRow)

Process the Results

The following are some of the most used methods:

07 previous()

08 next()

09 getRow()

10 moveToInsertRow()

11 moveToCurrentRow()

Key Takeaways

- All web applications store data in some form of database, like MySQL or SQL Server.
- Data can be retrieved from databases using the JDBC library.
- The JDBC interface allows users to execute SQL statements using a SQL-API interface.
- To process the results, the following methods used are the next(), previous(), last(), and many other.

Thank You