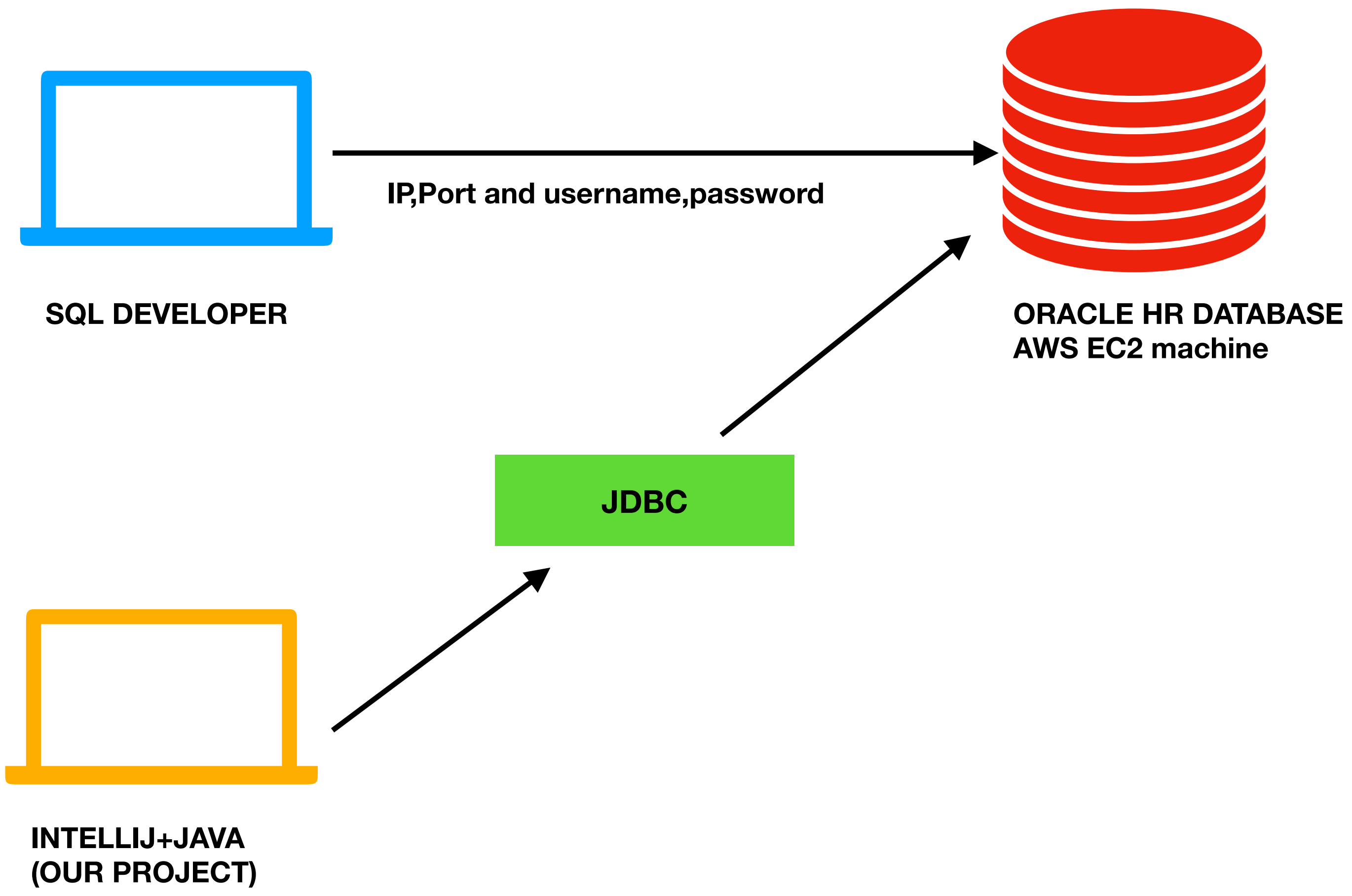


JAVA— —Selenium—Browser
JAVA— —Apache Poi—Excel Files
JAVA — — JDBC — Databases

JDBC is like middle man, translator

Ready classes written in java, works for all database we just need driver for our type of database, code will not change.

JAVA — Selenium — ChromeDriver— Chrome
JAVA — Selenium — FirefoxDriver— Firefox

JAVA—JDBC — OracleDriver—Oracle DB
JAVA —JDBC — Postgresql Driver - Postgresql

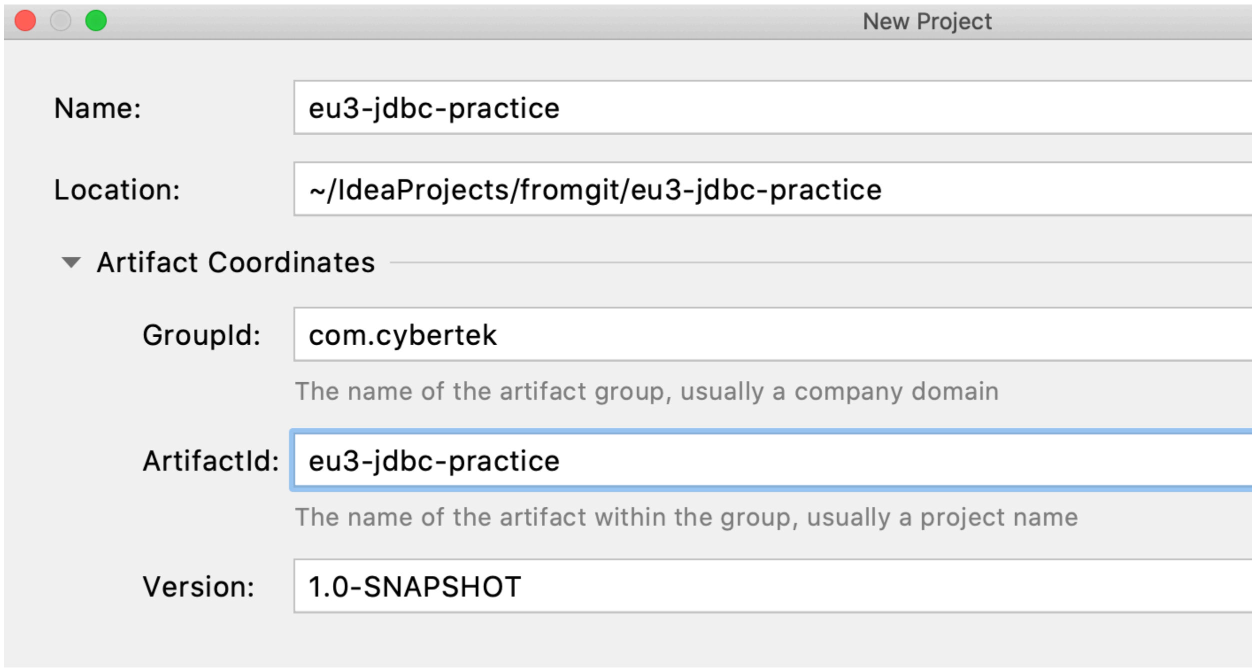
Using JDBC, we can programmatically access databases and read/update/create/delete data.

JDBC libraries already come with java standard edition.
For every database, there is a Driver that enables programmatic access to database. Those drivers are developed by database manufacturers.

Using JDBC will help with code reusability, as code written for one database will work for another type of database, and all we need to do will change the driver/credentials to database.

==

Intellij new maven project
1.File>New>Project — chose maven



Name: eu3-jdbc-practice

Location: ~/IdeaProjects/fromgit/eu3-jdbc-practice

Artifact Coordinates

GroupId: com.cybertek
The name of the artifact group, usually a company domain

ArtifactId: eu3-jdbc-practice
The name of the artifact within the group, usually a project name

Version: 1.0-SNAPSHOT

2.Click Finish

3.Set your java to 8

```
<properties>
    <maven.compiler.source>1.8</maven.compiler.source>
    <maven.compiler.target>1.8</maven.compiler.target>
</properties>
```

4.Add following dependencies

```
<dependencies>
   <dependency>
      <groupId>oracle</groupId>
      <artifactId>ojdbc6</artifactId>
      <version>11.2.0.3</version>
   </dependency>
</dependencies>

<repositories>
   <repository>
      <id>oracle</id>
      <url>http://www.datanucleus.org/downloads/maven2/</url>
   </repository>
</repositories>
```

5.Create a package under tests/java named: jdbctests

6.New java class: Main

==

ResultSet Methods

next()—> move to next row

previous()—> move to previous row

beforeFirst()—> goes before the first row

afterLast()—>goes after last row

getRow()—> get the current row number

last()—> moves to last row

absolute() —> goes specific row

ResultSet.*TYPE_SCROLL_INSENSITIVE*

—> allow us to navigate up and down in query result.

ResultSet.*CONCUR_READ_ONLY —>*

Read only, don't update the database

==

Column name —> rsMetadata.getColumnName(i)

Column value —> resultSet.getObject(i)

I —> column index

Number of rows —>while(resultSet.next()){}

Number of Column —> rsMetadata.getColumnCount();

DBUtils Methods

DBUtils.createConnection();—> create connection to db that you put information inside the method.

DBUtils.destroy()—> closes the connection

DBUtils.getQueryResultMap(String query) —> return list of maps, useful when you are getting multiple rows of result.

DBUtils.getRowMap(String query) —> returns maps of string object, useful when we have only one result.