Project 5 Task 1: Explanation

**When you are done,** write a brief explanation of how/where (file path and line numbers as seen on Github, and snapshots of usage in code) you use the JDBC connection pooling and prepared statements in your code. You should submit this report to Github.

I was able to enable JDBC connection pooling by:

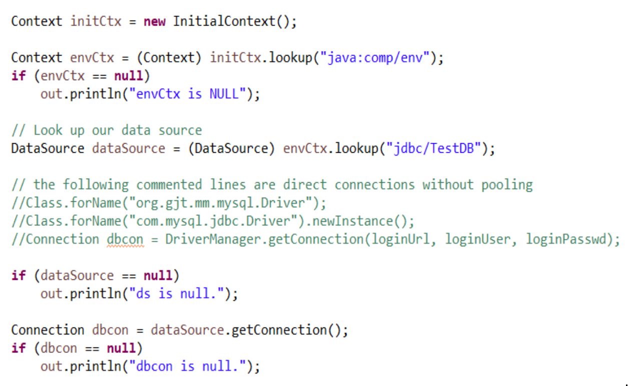
1) Creating a initial Context Object) 'Context' initCtx = new InitialContext();

2) Lookup connection in the connection Pool) `Context envCtx = (Context)initCtx.lookup("java:comp/env");` - a Context will look at all the pool connections.

3) Look up our Data Source) DataSource dataSource = (DataSource) envCtx.lookup("jdbc/TestDB"); - This statement will create a dataSource to connect to our mysql database.

Below are two tables where it explains how and where I used JDBC connection pooling and prepared statements in my code.

JDBC Connection Pooling Explanation: - I only included 1 explanation and 1 snapshot because they are all basically a repetition of the same task and will look/have the same purpose.

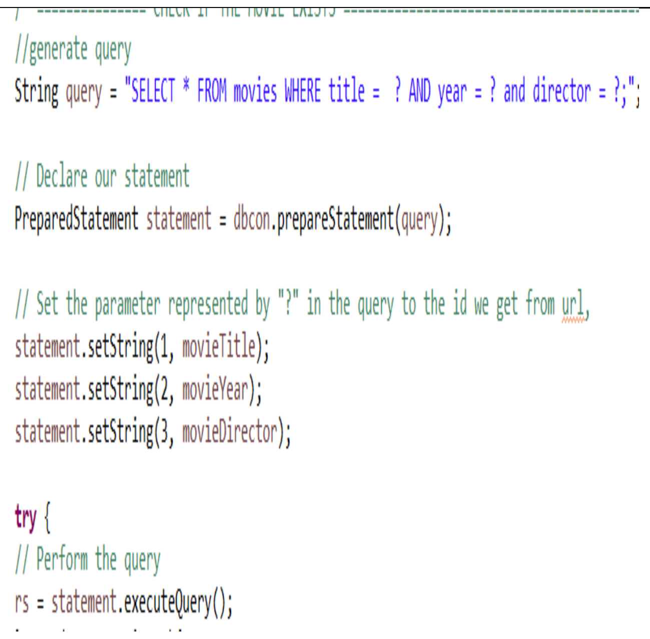
| File Path: | Line Number | How | Snapshot of Usage |
|---|---|---|---|
| /AddMovieServlet.java | 52-59 | 1.Created Context Object 2.used Context object to do a lookup in the connection pool. 3.used DataSource to get connection and execute queries. |  |
| /AndroidLoginServlet.java | 56-76 | | |
| /CheckoutServlet..java | 62-84 | | |
| /EmployeeLoginServlet.java | 47-67 | | |
| /GenreSearchServlet.java | 49-67 | | |
| /InsertStarServlet.java | 52-59 | | |
| /LoginServlet.java | 46-64 | | |

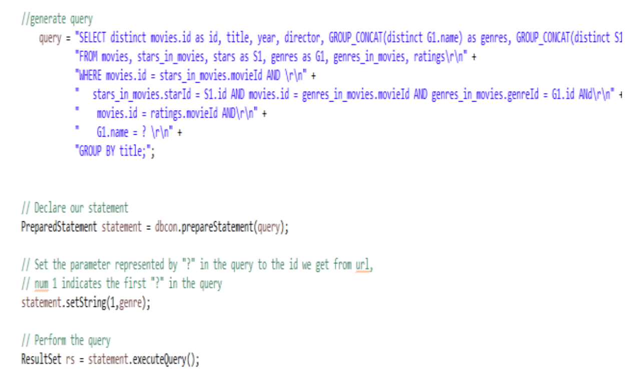
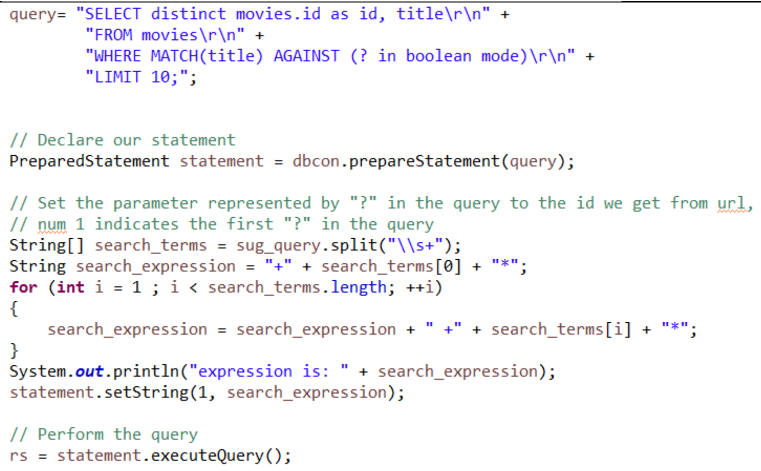
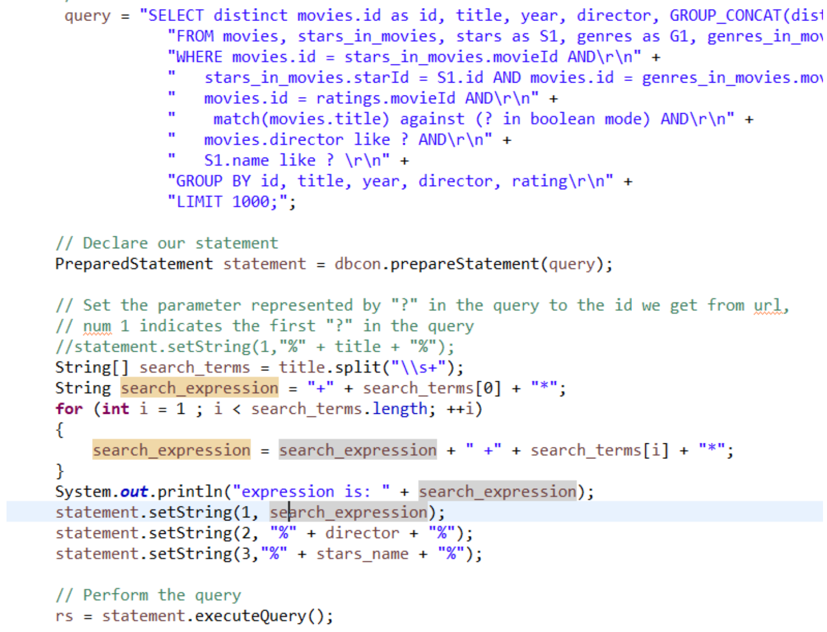
```
Context initCtx = new InitialContext();

Context envCtx = (Context) initCtx.lookup("java:comp/env");
if (envCtx == null)
    out.println("envCtx is NULL");

// Look up our data source
DataSource dataSource = (DataSource) envCtx.lookup("jdbc/TestDB");

// the following commented lines are direct connections without pooling
//Class.forName("org.gjt.mm.mysql.Driver");
//Class.forName("com.mysql.jdbc.Driver").newInstance();
//Connection dbcon = DriverManager.getConnection(loginUrl, loginUser, loginPasswd);

if (dataSource == null)
    out.println("ds is null.");

Connection dbcon = dataSource.getConnection();
if (dbcon == null)
    out.println("dbcon is null.");
```

| | | | |
|---|---|---|---|
| /MovieSuggestion.java | 43-57 | | |
| /Results.java | 44-57 | | |
| /SearchServlet.java | 43-67 | | |
| /SingleMovieServlet.java | 40-58 | | |
| /SingleStarServlet.java | 40-55 | | |
| /TitleSearchServlet.java | 49-67 | | |

Prepared Statement Explanation:

1) I stored the query in a string variable. Any place that requires user input will be replaced with a '?'
2) I created a 'PreparedStatement' object and initialized it a Conection Object where I use my string query as a argument. The Connection object will connect my servlet to my mysql instance.
3) I set the value of any '?' in my PreparedStatement with the statement 'statement.set*'.
4) I execute the query and store the result into a 'ResultSet' object.

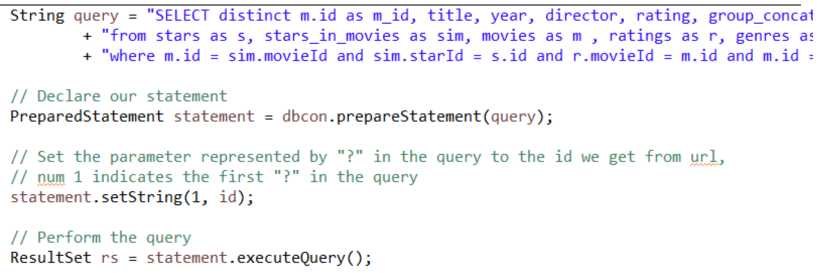
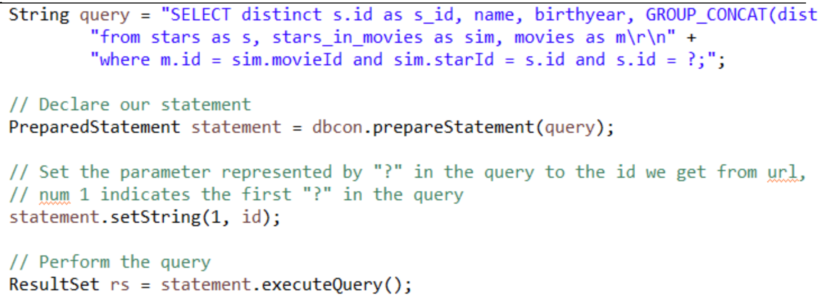
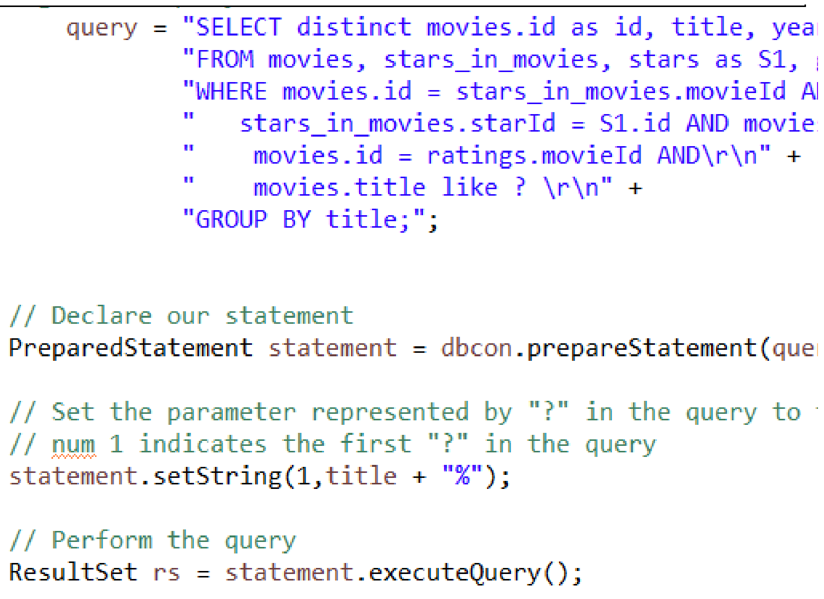
| File Path: | Line Number | How | Snapshot of Usage |
|---|---|---|---|
| /AddMovieServlet.java | 102-107 | I used a prepare statement to execute a query to add a movie to the database. I used a prepare statement to improve performance and increase security for prevention against mysql injection attacks. |  `//generate query` `String query = "SELECT * FROM movies WHERE title =  ? AND year = ? and director = ?;";` `// Declare our statement` `PreparedStatement statement = dbcon.prepareStatement(query);` `// Set the parameter represented by "?" in the query to the id we get from url,` `statement.setString(1, movieTitle);` `statement.setString(2, movieYear);` `statement.setString(3, movieDirector);` `try {` `// Perform the query` `rs = statement.executeQuery();` |

|  |  |  |  |
|---|---|---|---|
| /GenreSearchServlet.java | 85-102 |  |  |
| /MovieSuggestion.java | 84-105 |  |  |
| /SearchServlet.java | 141-158 |  |  |

```
//generate query
    query = "SELECT distinct movies.id as id, title, year, director, GROUP_CONCAT(distinct G1.name) as genres, GROUP_CONCAT(distinct S1
            "FROM movies, stars_in_movies, stars as S1, genres as G1, genres_in_movies, ratings\r\n" +
            "WHERE movies.id = stars_in_movies.movieId AND \r\n" +
            "   stars_in_movies.starId = S1.id AND movies.id = genres_in_movies.movieId AND genres_in_movies.genreId = G1.id ANd\r\n" +
            "    movies.id = ratings.movieId AND\r\n" +
            "    G1.name = ? \r\n" +
            "GROUP BY title;";


// Declare our statement
PreparedStatement statement = dbcon.prepareStatement(query);

// Set the parameter represented by "?" in the query to the id we get from url,
// num 1 indicates the first "?" in the query
statement.setString(1,genre);

// Perform the query
ResultSet rs = statement.executeQuery();
```

```
query= "SELECT distinct movies.id as id, title\r\n" +
        "FROM movies\r\n" +
        "WHERE MATCH(title) AGAINST (? in boolean mode)\r\n" +
        "LIMIT 10;";


// Declare our statement
PreparedStatement statement = dbcon.prepareStatement(query);

// Set the parameter represented by "?" in the query to the id we get from url,
// num 1 indicates the first "?" in the query
String[] search_terms = sug_query.split("\\s+");
String search_expression = "+" + search_terms[0] + "*";
for (int i = 1 ; i < search_terms.length; ++i)
{
    search_expression = search_expression + " +" + search_terms[i] + "*";
}
System.out.println("expression is: " + search_expression);
statement.setString(1, search_expression);

// Perform the query
rs = statement.executeQuery();
```

```
    query = "SELECT distinct movies.id as id, title, year, director, GROUP_CONCAT(dist
               "FROM movies, stars_in_movies, stars as S1, genres as G1, genres_in_mov
               "WHERE movies.id = stars_in_movies.movieId AND\r\n" +
               "   stars_in_movies.starId = S1.id AND movies.id = genres_in_movies.mov
               "   movies.id = ratings.movieId AND\r\n" +
               "    match(movies.title) against (? in boolean mode) AND\r\n" +
               "   movies.director like ? AND\r\n" +
               "   S1.name like ? \r\n" +
               "GROUP BY id, title, year, director, rating\r\n" +
               "LIMIT 1000;";

// Declare our statement
PreparedStatement statement = dbcon.prepareStatement(query);

// Set the parameter represented by "?" in the query to the id we get from url,
// num 1 indicates the first "?" in the query
//statement.setString(1,"%" + title + "%");
String[] search_terms = title.split("\\s+");
String search_expression = "+" + search_terms[0] + "*";
for (int i = 1 ; i < search_terms.length; ++i)
{
    search_expression = search_expression + " +" + search_terms[i] + "*";
}
System.out.println("expression is: " + search_expression);
statement.setString(1, search_expression);
statement.setString(2, "%" + director + "%");
statement.setString(3,"%" + stars_name + "%");

// Perform the query
rs = statement.executeQuery();
```

| | | | |
|---|---|---|---|
| /SingleMovieServlet | 61-73 | | <br>`String query = "SELECT distinct m.id as m_id, title, year, director, rating, group_concat`<br>`        + "from stars as s, stars_in_movies as sim, movies as m , ratings as r, genres as`<br>`        + "where m.id = sim.movieId and sim.starId = s.id and r.movieId = m.id and m.id =`<br><br>`// Declare our statement`<br>`PreparedStatement statement = dbcon.prepareStatement(query);`<br><br>`// Set the parameter represented by "?" in the query to the id we get from url,`<br>`// num 1 indicates the first "?" in the query`<br>`statement.setString(1, id);`<br><br>`// Perform the query`<br>`ResultSet rs = statement.executeQuery();` |
| /SingleStarServlet.java | 66-73 | | <br>`String query = "SELECT distinct s.id as s_id, name, birthyear, GROUP_CONCAT(dist`<br>`        "from stars as s, stars_in_movies as sim, movies as m\r\n" +`<br>`        "where m.id = sim.movieId and sim.starId = s.id and s.id = ?;";`<br><br>`// Declare our statement`<br>`PreparedStatement statement = dbcon.prepareStatement(query);`<br><br>`// Set the parameter represented by "?" in the query to the id we get from url,`<br>`// num 1 indicates the first "?" in the query`<br>`statement.setString(1, id);`<br><br>`// Perform the query`<br>`ResultSet rs = statement.executeQuery();` |
| /TitleSearchServlet.java | 85-102 | | <br>`    query = "SELECT distinct movies.id as id, title, yea`<br>`            "FROM movies, stars_in_movies, stars as S1,`<br>`            "WHERE movies.id = stars_in_movies.movieId A`<br>`            "    stars_in_movies.starId = S1.id AND movie`<br>`            "     movies.id = ratings.movieId AND\r\n" +`<br>`            "     movies.title like ? \r\n" +`<br>`            "GROUP BY title;";`<br><br>`// Declare our statement`<br>`PreparedStatement statement = dbcon.prepareStatement(que`<br><br>`// Set the parameter represented by "?" in the query to`<br>`// num 1 indicates the first "?" in the query`<br>`statement.setString(1,title + "%");`<br><br>`// Perform the query`<br>`ResultSet rs = statement.executeQuery();` |
| | | | |
| | | | |
| | | | |

## Task 2

- Address of AWS and Google instances

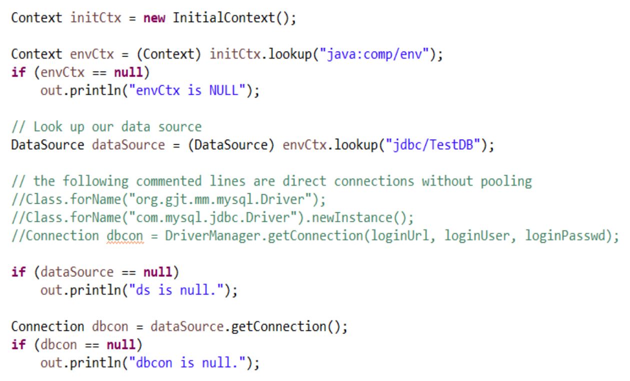
| Google ip | AWS 1 ip | AWS 2 IP | AWS 3 IP | Project url |
|---|---|---|---|---|
| 104.198.102.203 | 13.57.250.62 | 54.183.189.126 | 54.183.219.83 | http://IP:port/project1/ |

- Have you verified that they are accessible? Does Fablix site get opened both on Google's 80 port and AWS' 8080 port?

  The Fabflix website successfully loads on both the Google 80 port and AWS 8080 port. When accessing the Google 80 port, the user is sent to either AWS 2 or AWS 3 on port 8080 with Apache load balancing.

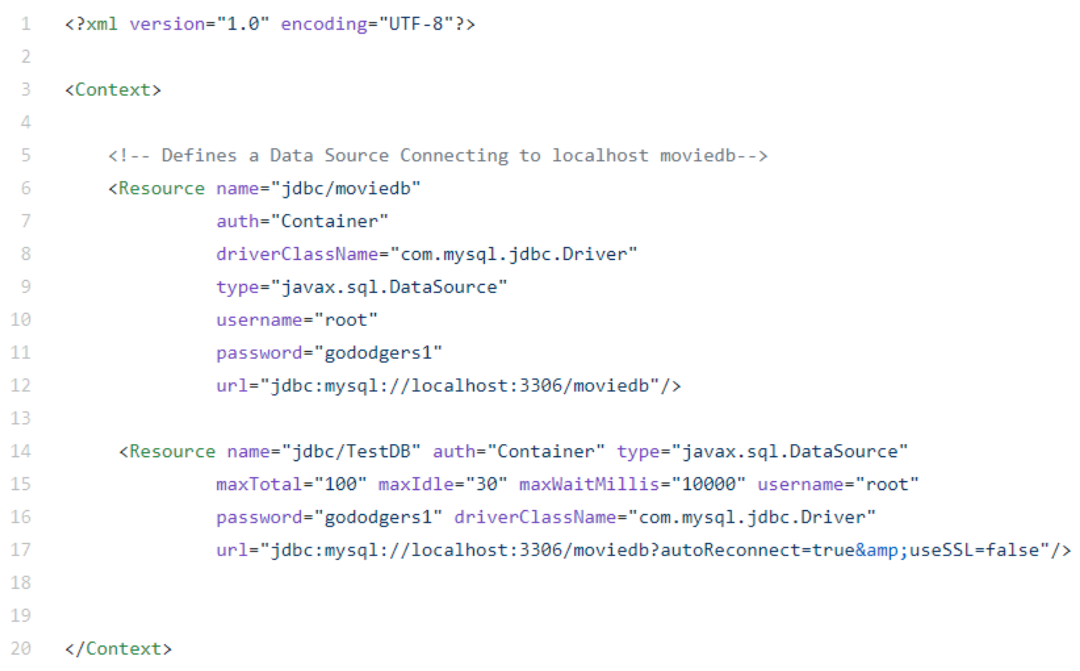
- Explain how connection pooling works with two backend SQL (in your code)?

  Since the master and slave SQL databases are both synchronized, whenever a new connection is requested, the connection pool cache is checked before establishing a new connection to either the master or slave database. This was done by using a context object to lookup a connection in the connection pool cache. (lines 52-59)



```
Context initCtx = new InitialContext();

Context envCtx = (Context) initCtx.lookup("java:comp/env");
if (envCtx == null)
    out.println("envCtx is NULL");

// Look up our data source
DataSource dataSource = (DataSource) envCtx.lookup("jdbc/TestDB");

// the following commented lines are direct connections without pooling
//Class.forName("org.gjt.mm.mysql.Driver");
//Class.forName("com.mysql.jdbc.Driver").newInstance();
//Connection dbcon = DriverManager.getConnection(loginUrl, loginUser, loginPasswd);

if (dataSource == null)
    out.println("ds is null.");

Connection dbcon = dataSource.getConnection();
if (dbcon == null)
    out.println("dbcon is null.");
```

- How read/write requests were routed?
  The architecture was setup so that the master database could be both read and written to, while the slave could only be read from. This was done by their respective tomcat instance (2 and 3). In the Fabflix project, the context.xml file was modified (lines 14-17) to allow a connection through a testDB which would route read/write requests through the master database, which is synchronized with the slave. This synchronization was done on each of the instances by connecting the master and slave MySQL database with a replication user and master log file with a given starting position.



```
<?xml version="1.0" encoding="UTF-8"?>

<Context>

    <!-- Defines a Data Source Connecting to localhost moviedb-->
    <Resource name="jdbc/moviedb"
              auth="Container"
              driverClassName="com.mysql.jdbc.Driver"
              type="javax.sql.DataSource"
              username="root"
              password="gododgers1"
              url="jdbc:mysql://localhost:3306/moviedb"/>

     <Resource name="jdbc/TestDB" auth="Container" type="javax.sql.DataSource"
              maxTotal="100" maxIdle="30" maxWaitMillis="10000" username="root"
              password="gododgers1" driverClassName="com.mysql.jdbc.Driver"
              url="jdbc:mysql://localhost:3306/moviedb?autoReconnect=true&amp;useSSL=false"/>


</Context>
```

**TASK 3**

The C script to calculate the average times is in:

cs122b-projects/project1/calcAverage.c

The log files are in: (same dir as this report)

***I did not log the cases removing prepared statements or pooling.***

cs122b-projects/scaled_1thread
cs122b-projects/scaled_10thread
cs122b-projects/single_1thread
cs122b-projects/ single_10thread
cs122b-projects/single_https10thread

The HTML file is in:

cs122b-projects/jmeter_report.html (same dir as this report)

WAR and readme is in:

cs122b-projects/project1.war
cs122b-projects/README.md