

## Lesson 01 Demo 04

### Writing CRUD Operations with PreparedStatement API

**Objective:** To implement PreparedStatement API with JDBC for writing CRUD operations for managing database records efficiently and securely

**Tool required:** Eclipse IDE

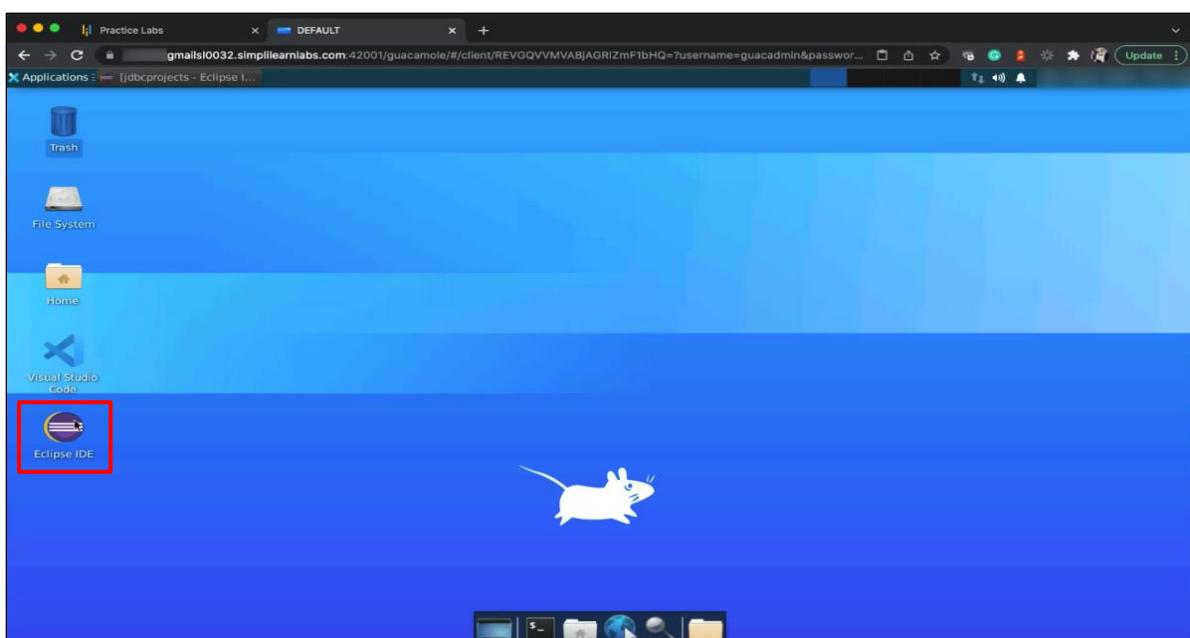
**Prerequisites:** Lesson 01 Demo 01

#### Steps to be followed:

1. Perform the Create operation
2. Perform the Update operation
3. Perform the Delete operation
4. Perform the getAllCustomer operation

#### Step 1: Perform the Create operation

##### 1.1 Open Eclipse IDE



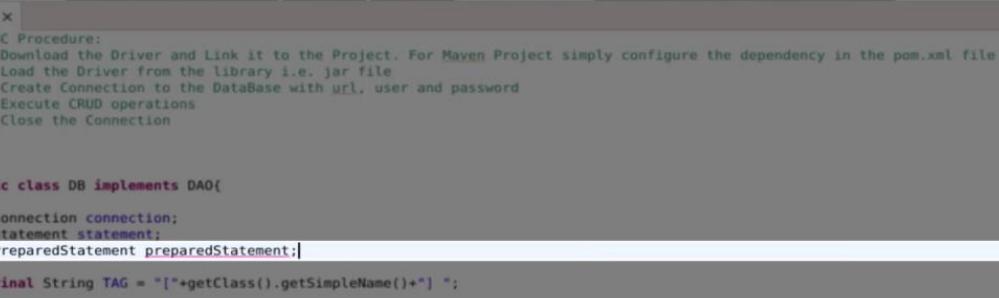
## 1.2 Open the **DB.java** file in the **CMS** project folder

The screenshot shows the Eclipse IDE interface with the following details:

- Title Bar:** Practice Labs - DEFAULT - jdbcprojects - CMS/src/...
- Toolbar:** File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, Help.
- Package Explorer:** Shows the project structure under CMS:
  - src/main/java
    - com.example.cms
    - com.example.cms.db
      - DB.java
      - DAO.java
    - com.example.cms.model
    - src/test/java
    - JRE System Library [JavaSE-1.8]
    - Maven Dependencies
    - src
    - target
    - pom.xml
  - JDBCConfiguration
- Editor:** DB.java (selected)

```
1 package com.example.cms;
2
3 import java.sql.Connection;
4
5 /**
6  * JDBC Procedure:
7  * 1. Download the Driver and Link it to the Project. For Maven Project simply configure the c
8  * 2. Load the Driver from the library i.e. jar file
9  * 3. Create Connection to the DataBase with url, user and password
10  * 4. Execute CRUD operations
11  * 5. Close the Connection
12  */
13
14 public class DB implements DAO{
15     Connection connection;
16     Statement statement;
17
18     final String TAG = "["+getClass().getSimpleName()+"] ";
19
20     public DB() {
21         try {
22             Class.forName("com.mysql.cj.jdbc.Driver");
23             System.out.println(TAG+"Driver Loaded");
24         } catch (Exception e) {
25             System.out.println("Exception Occurred: "+e);
26         }
27     }
28
29
30     public void createConnection() {
31         try {
32             Class.forName("com.mysql.cj.jdbc.Driver");
33             System.out.println(TAG+"Driver Loaded");
34         } catch (Exception e) {
35             System.out.println("Exception Occurred: "+e);
36         }
37     }
38
39     public void closeConnection() {
40         try {
41             Statement statement;
42             if(statement != null)
43                 statement.close();
44         } catch (Exception e) {
45             System.out.println("Exception Occurred: "+e);
46         }
47     }
48
49     public void createCustomer(Customer customer) {
50         try {
51             Statement statement;
52             if(statement != null)
53                 statement.executeUpdate("insert into customer values ('"+customer.getName()+"','"+customer.getAge()+"')");
54         } catch (Exception e) {
55             System.out.println("Exception Occurred: "+e);
56         }
57     }
58
59     public void updateCustomer(Customer customer) {
60         try {
61             Statement statement;
62             if(statement != null)
63                 statement.executeUpdate("update customer set age = "+customer.getAge()+" where name = '"+customer.getName()+"'");
64         } catch (Exception e) {
65             System.out.println("Exception Occurred: "+e);
66         }
67     }
68
69     public void deleteCustomer(int id) {
70         try {
71             Statement statement;
72             if(statement != null)
73                 statement.executeUpdate("delete from customer where id = "+id);
74         } catch (Exception e) {
75             System.out.println("Exception Occurred: "+e);
76         }
77     }
78
79     public List<Customer> getAllCustomers() {
80         try {
81             Statement statement;
82             if(statement != null)
83                 return statement.executeQuery("select * from customer").  
getList();
84         } catch (Exception e) {
85             System.out.println("Exception Occurred: "+e);
86         }
87         return null;
88     }
89 }
```
- Outline View:** Shows the class structure:
  - com.example.cms.db
  - DB
    - connection : Connection
    - statement : Statement
    - TAG : String
    - DB()
      - createConnection() : void
      - closeConnection() : void
      - createCustomer(Customer) : void
      - updateCustomer(Customer) : void
      - deleteCustomer(int) : void
      - getAllCustomers() : ArrayList<Customer>
- Bottom Status Bar:** Writable, Smart Insert, 1:1:0

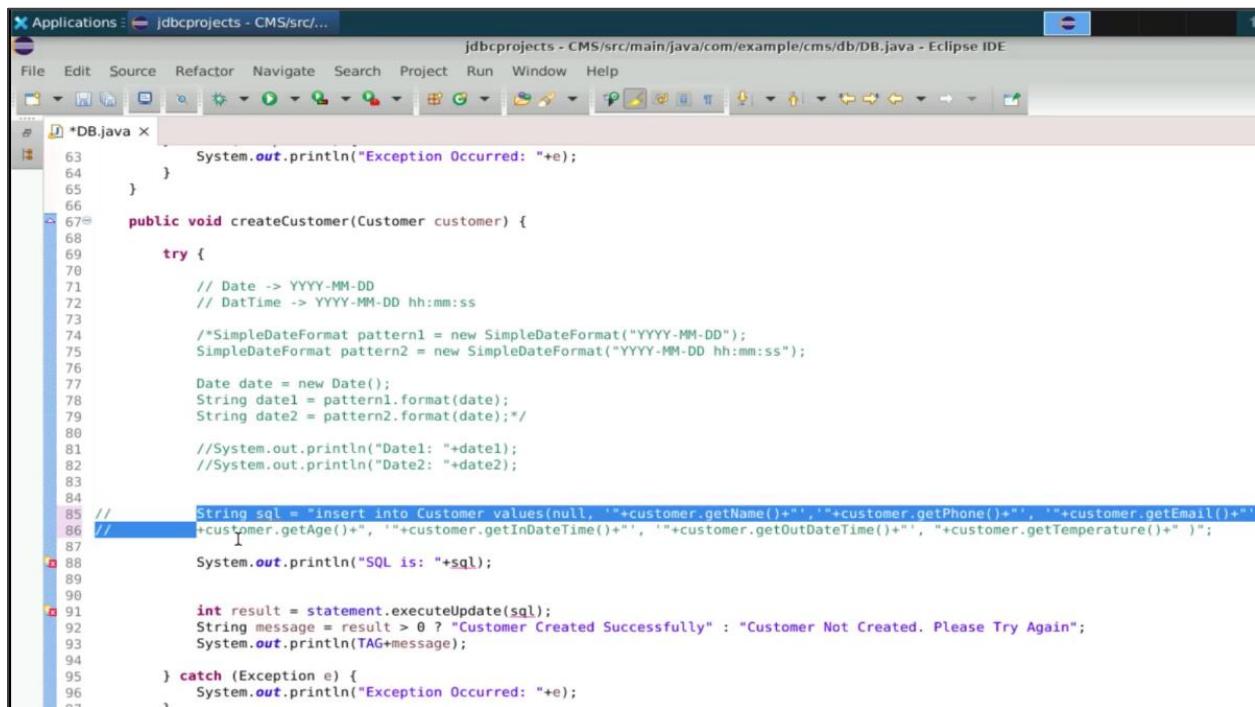
1.3 Write another API for **PreparedStatement** with a reference variable **preparedStatement**:



The screenshot shows the Eclipse IDE interface with the title bar "Applications - jdbcprojects - CMS/src/..." and "jdbcprojects - CMS/src/main/java/com/example/cms/db/DB.java - Eclipse IDE". The menu bar includes File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, and Help. The toolbar has various icons for file operations like Open, Save, Cut, Copy, Paste, Find, and Select. The left sidebar shows the package structure with a file named "DB.java". The main editor area contains the following Java code:

```
15 JDBC Procedure:  
16 1. Download the Driver and Link it to the Project. For Maven Project simply configure the dependency in the pom.xml file  
17 2. Load the Driver from the library i.e. jar file  
18 3. Create Connection to the DataBase with url, user and password  
19 4. EXECUTE CRUD operations  
20 5. Close the Connection  
21  
22 */  
23  
24 public class DB implements DAO{  
25  
26     Connection connection;  
27     Statement statement;  
28     PreparedStatement preparedStatement;  
29  
30     final String TAG = "["+getClass().getSimpleName()+"] ";  
31  
32     public DB() {  
33         try {  
34             Class.forName("com.mysql.cj.jdbc.Driver");  
35             System.out.println(TAG+"Driver Loaded");  
36         } catch (Exception e) {  
37             System.out.println("Exception Occurred: "+e);  
38         }  
39     }  
40  
41     public void createConnection() {  
42         try {  
43             String user = "john";  
44             String password = "john";  
45             String url = "jdbc:mysql://localhost/estore";  
46             connection = DriverManager.getConnection(url, user, password);  
47             statement = connection.createStatement();  
48         } catch (Exception e) {  
49             System.out.println("Exception Occurred: "+e);  
50         }  
51     }  
52  
53     public void insertData() {  
54         try {  
55             String query = "insert into product values(1, 'Laptop', 100000, 100);";  
56             statement.executeUpdate(query);  
57         } catch (Exception e) {  
58             System.out.println("Exception Occurred: "+e);  
59         }  
60     }  
61  
62     public void updateData() {  
63         try {  
64             String query = "update product set price = 150000 where id = 1";  
65             statement.executeUpdate(query);  
66         } catch (Exception e) {  
67             System.out.println("Exception Occurred: "+e);  
68         }  
69     }  
70  
71     public void deleteData() {  
72         try {  
73             String query = "delete from product where id = 1";  
74             statement.executeUpdate(query);  
75         } catch (Exception e) {  
76             System.out.println("Exception Occurred: "+e);  
77         }  
78     }  
79  
80     public void displayData() {  
81         try {  
82             String query = "select * from product";  
83             ResultSet rs = statement.executeQuery(query);  
84             while(rs.next()) {  
85                 System.out.println(rs.getString("name") + " " + rs.getInt("id") + " " + rs.getDouble("price"));  
86             }  
87         } catch (Exception e) {  
88             System.out.println("Exception Occurred: "+e);  
89         }  
90     }  
91 }  
92 }
```

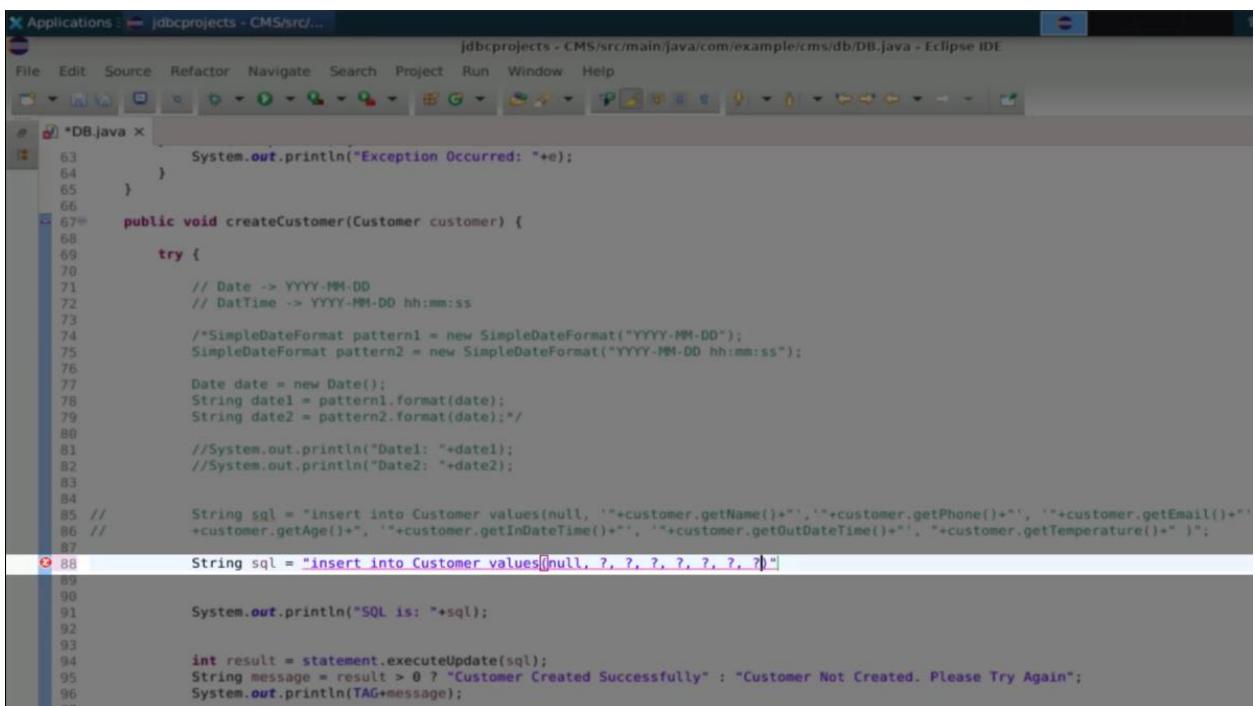
## 1.4 Comment out the previous SQL statement



```

  63
  64
  65
  66
  67  public void createCustomer(Customer customer) {
  68
  69      try {
  70          // Date -> YYYY-MM-DD
  71          // Datetime -> YYYY-MM-DD hh:mm:ss
  72
  73          /*SimpleDateFormat pattern1 = new SimpleDateFormat("YYYY-MM-DD");
  74          SimpleDateFormat pattern2 = new SimpleDateFormat("YYYY-MM-DD hh:mm:ss");
  75
  76          Date date = new Date();
  77          String date1 = pattern1.format(date);
  78          String date2 = pattern2.format(date);*/
  79
  80          //System.out.println("Date1: "+date1);
  81          //System.out.println("Date2: "+date2);
  82
  83
  84
  85      // String sql = "insert into Customer values(null, '"+customer.getName()+"','"+customer.getPhone()+"', '"+customer.getEmail()+""
  86      // "+customer.getAge()+"', '"+customer.getInDate()+"', '"+customer.getOutDate()+"', "+customer.getTemperature()+" )";
  87
  88      System.out.println("SQL is: "+sql);
  89
  90
  91      int result = statement.executeUpdate(sql);
  92      String message = result > 0 ? "Customer Created Successfully" : "Customer Not Created. Please Try Again";
  93      System.out.println(TAG+message);
  94
  95  } catch (Exception e) {
  96      System.out.println("Exception Occurred: "+e);
  97
  
```

## 1.5 Write the SQL statement for the **insert** operation in the **Customer** table



```

  63
  64
  65
  66
  67  public void createCustomer(Customer customer) {
  68
  69      try {
  70          // Date -> YYYY-MM-DD
  71          // Datetime -> YYYY-MM-DD hh:mm:ss
  72
  73          /*SimpleDateFormat pattern1 = new SimpleDateFormat("YYYY-MM-DD");
  74          SimpleDateFormat pattern2 = new SimpleDateFormat("YYYY-MM-DD hh:mm:ss");
  75
  76          Date date = new Date();
  77          String date1 = pattern1.format(date);
  78          String date2 = pattern2.format(date);*/
  79
  80          //System.out.println("Date1: "+date1);
  81          //System.out.println("Date2: "+date2);
  82
  83
  84
  85
  86
  87
  88      String sql = "insert into Customer values(null, ?, ?, ?, ?, ?, ?, ?)";
  89
  90
  91      System.out.println("SQL is: "+sql);
  92
  93
  94      int result = statement.executeUpdate(sql);
  95      String message = result > 0 ? "Customer Created Successfully" : "Customer Not Created. Please Try Again";
  96      System.out.println(TAG+message);
  
```

## 1.6 Initialize the preparedStatement

```

 63     System.out.println("Exception Occurred: "+e);
 64   }
 65 }
 66
 67 public void createCustomer(Customer customer) {
 68   try {
 69     // Date -> YYYY-MM-DD
 70     // DatTime -> YYYY-MM-DD hh:mm:ss
 71
 72     /*SimpleDateFormat pattern1 = new SimpleDateFormat("YYYY-MM-DD");
 73     SimpleDateFormat pattern2 = new SimpleDateFormat("YYYY-MM-DD hh:mm:ss");
 74
 75     Date date = new Date();
 76     String date1 = pattern1.format(date);
 77     String date2 = pattern2.format(date);*/
 78
 79     //System.out.println("Date1: "+date1);
 80     //System.out.println("Date2: "+date2);
 81
 82     String sql = "insert into Customer values(null, '"+customer.getName()+"','"+customer.getPhone()+"', '"+customer.getEmail()+""
 83     "+customer.getAge()+"', '"+customer.getInDateTime()+"', '"+customer.getOutDateTime()+"', "+customer.getTemperature()+" )";
 84
 85     String sql = "insert into Customer values(null, ?, ?, ?, ?, ?, ?, ?, ?, ?)";
 86     preparedStatement = connection.prepareStatement(sql);
 87
 88
 89     System.out.println("SQL is: "+sql);
 90
 91
 92
 93
 94     int result = statement.executeUpdate(sql);
 95     String message = result > 0 ? "Customer Created Successfully" : "Customer Not Created. Please Try Again";
 96     System.out.println(TAG+message);
 97
 98   } catch (Exception e) {
 99     System.out.println("Exception Occurred: "+e);
100   }
101
102 }
103
104 public void updateCustomer(Customer customer) {
105   try {
106
107     String sql = "update Customer set name = '"+customer.getName()+"', phone = '"+customer.getPhone()+"', email = '"+customer.getEmail()+"'
108     '+', birthDate = '"+customer.getBirthDate()+"', age = "+customer.getAge()+"', inDateTime = '"+customer.getInDateTime()+"', out
109     +' where cid = "+customer.getCid();
110
111     System.out.println("SQL is: "+sql);
112
113     int result = statement.executeUpdate(sql);
114     String message = result > 0 ? "Customer Updated Successfully" : "Customer Not Updated. Please Try Again";
115     System.out.println(TAG+message);
116
117   } catch (Exception e) {
118     System.out.println("Exception Occurred: "+e);
119   }
120 }
```

## 1.7 Cut (Ctrl+X) the selected line of code

```

 85 //      String sql = "insert into Customer values(null, '"+customer.getName()+"','"+customer.getPhone)+"', '"+customer.getEmail()+""
 86 //      "+customer.getAge()+"', '"+customer.getInDateTime)+"', '"+customer.getOutDateTime)+"', "+customer.getTemperature)+" )";
 87
 88 String sql = "insert into Customer values(null, ?, ?, ?, ?, ?, ?, ?, ?, ?)";
 89 preparedStatement = connection.prepareStatement(sql);
 90
 91 System.out.println("SQL is: "+sql);           I
 92
 93
 94     int result = statement.executeUpdate(sql);
 95     String message = result > 0 ? "Customer Created Successfully" : "Customer Not Created. Please Try Again";
 96     System.out.println(TAG+message);
 97
 98   } catch (Exception e) {
 99     System.out.println("Exception Occurred: "+e);
100   }
101
102 }
103
104 public void updateCustomer(Customer customer) {
105   try {
106
107     String sql = "update Customer set name = '"+customer.getName()+"', phone = '"+customer.getPhone()+"', email = '"+customer.getEmail()+"'
108     '+', birthDate = '"+customer.getBirthDate)+"', age = "+customer.getAge)+"', inDateTime = '"+customer.getInDateTime)+"', out
109     +' where cid = "+customer.getCid();
110
111     System.out.println("SQL is: "+sql);
112
113     int result = statement.executeUpdate(sql);
114     String message = result > 0 ? "Customer Updated Successfully" : "Customer Not Updated. Please Try Again";
115     System.out.println(TAG+message);
116
117   } catch (Exception e) {
118     System.out.println("Exception Occurred: "+e);
119   }
120 }
```

1.8 Paste the code above the preparedStatement object (line 89), and now pass the **String sql**

```

 85 // 
 86 // 
 87 
 88     String sql = "insert into Customer values(null, ?, ?, ?, ?, ?, ?, ?, ?)";
 89     System.out.println("SQL is: "+sql);
 90     preparedStatement = connection.prepareStatement(sql);
 91 
 92 
 93 
 94 
 95     int result = statement.executeUpdate(sql);
 96     String message = result > 0 ? "Customer Created Successfully" : "Customer Not Created. Please Try Again";
 97     System.out.println(TAG+message);
 98 
 99     } catch (Exception e) {
100         System.out.println("Exception Occurred: "+e);
101     }
102 }
103 
104 public void updateCustomer(Customer customer) {
105     try {
106 
107         String sql = "update Customer set name = '"+customer.getName()+"', phone = '"+customer.getPhone()+"', email = '"+customer.getEmail()+"',
108             birthDate = '"+customer.getBirthDate()+"', age = '"+customer.getAge()+"', inDateTime = '"+customer.getInDateTime()+"', out
109             +" where cid = '"+customer.getCid()+"';
110 
111         System.out.println("SQL is: "+sql);
112 
113         int result = statement.executeUpdate(sql);
114         String message = result > 0 ? "Customer Updated Successfully" : "Customer Not Updated. Please Try Again";
115         System.out.println(TAG+message);
116 
117     } catch (Exception e) {
118 
119     }
120 }

```

1.9 Substitute the data through the **getter** method, use the **setString()** method and index starting from 1 for each attribute of data, as shown below from line 92 to line 99:

```

 85 // 
 86 // 
 87 
 88     String sql = "insert into Customer values(null, ?, ?, ?, ?, ?, ?, ?, ?)";
 89     System.out.println("SQL is: "+sql);
 90 
 91     preparedStatement = connection.prepareStatement(sql);
 92     preparedStatement.setString(1, customer.getName());
 93     preparedStatement.setString(2, customer.getPhone());
 94     preparedStatement.setString(3, customer.getEmail());
 95     preparedStatement.setString(4, customer.getBirthDate());
 96     preparedStatement.setInt(5, customer.getAge());
 97     preparedStatement.setString(6, customer.getInDateTime());
 98     preparedStatement.setString(7, customer.getOutDateTime());
 99     preparedStatement.setFloat(8, customer.getTemperature());
100 
101     int result = statement.executeUpdate(sql);
102     String message = result > 0 ? "Customer Created Successfully" : "Customer Not Created. Please Try Again";
103     System.out.println(TAG+message);
104 
105     } catch (Exception e) {
106         System.out.println("Exception Occurred: "+e);
107     }
108 
109 }

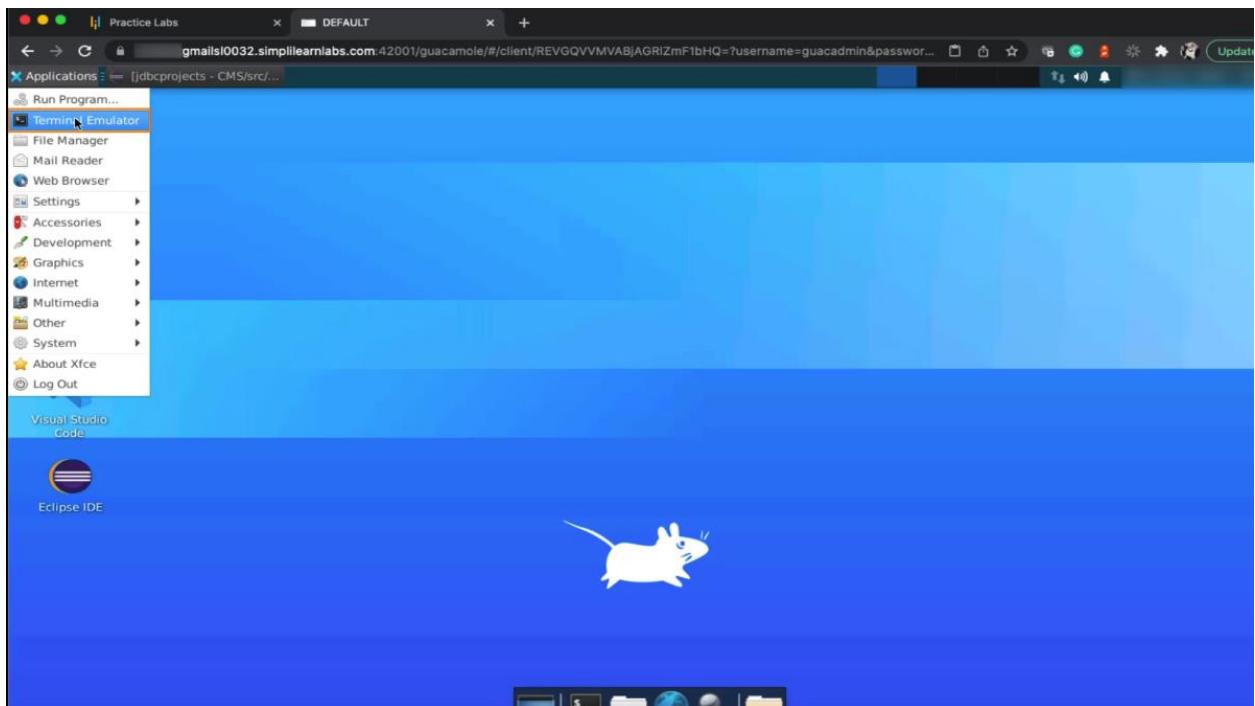
```

## 1.10 Comment on the calling of the `executeUpdate()` function, as it will remain similar to the above lines

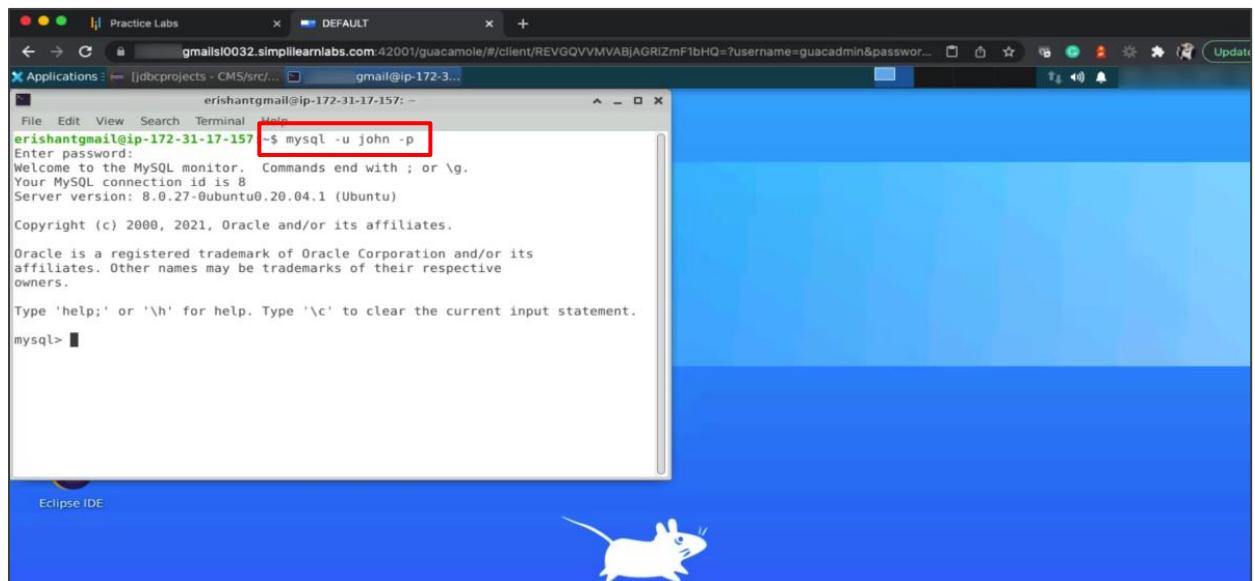
```

85 // 
86 // 
87 
88 String sql = "insert into Customer values(null, '"+customer.getName()+"', '"+customer.getPhone()+"', '"+customer.getEmail()+"', '"+customer.getBirthDate()+"'"+customer.getAge()+"', '"+customer.getInDateTime()+"', '"+customer.getOutDateTime()+"', '"+customer.getTemperature()+"')"; 
89 System.out.println("SQL is "+sql); 
90 
91 PreparedStatement = connection.prepareStatement(sql); 
92 PreparedStatement.setString(1, customer.getName()); 
93 PreparedStatement.setString(2, customer.getPhone()); 
94 PreparedStatement.setString(3, customer.getEmail()); 
95 PreparedStatement.setInt(4, customer.getBirthDate()); 
96 PreparedStatement.setInt(5, customer.getAge()); 
97 PreparedStatement.setString(6, customer.getInDateTime()); 
98 PreparedStatement.setString(7, customer.getOutDateTime()); 
99 PreparedStatement.setFloat(8, customer.getTemperature()); 
100 
101 //int result = statement.executeUpdate(); 
102 String message = result > 0 ? "Customer Created Successfully" : "Customer Not Created. Please Try Again"; 
103 System.out.println(TAG+message); 
104 
105 } catch (Exception e) { 
106     System.out.println("Exception Occurred: "+e); 
107 } 
108 } 
109 } 
110 } 
111 } 
112 } 
113 public void updateCustomer(Customer customer) { 
114     try { 
115         String sql = "update Customer set name = '"+customer.getName()+"', phone = '"+customer.getPhone()+"', email = '"+customer.getEmail()+"', birthDate = '"+customer.getBirthDate()+"', age = "+customer.getAge()+"', inDateTime = '"+customer.getInDateTime()+"', outDateTime = '"+customer.getOutDateTime()+"' where cid = "+customer.getCid(); 
116         System.out.println("SQL is: "+sql); 
117         int result = statement.executeUpdate(sql); 
118         String message = result > 0 ? "Customer Updated Successfully" : "Customer Not Updated. Please Try Again"; 
119         System.out.println(TAG+message); 
120     } catch (Exception e) { 
121         System.out.println("Exception Occurred: "+e); 
122     } 
123 } 
124 } 
125 } 
126 } 
127 } 
128 } 
129 } 
130 } 
131 } 
132 } 
133 } 
134 } 
135 } 
136 } 
137 } 
138 } 
139 } 
140 } 
141 } 
142 } 
143 } 
144 } 
145 } 
146 } 
147 } 
148 } 
149 } 
150 } 
151 } 
152 } 
153 } 
154 } 
155 } 
156 } 
157 } 
158 } 
159 } 
160 } 
161 } 
162 } 
163 } 
164 } 
165 } 
166 } 
167 } 
168 } 
169 } 
170 } 
171 } 
172 } 
173 } 
174 } 
175 } 
176 } 
177 } 
178 } 
179 } 
180 } 
181 } 
182 } 
183 } 
184 } 
185 } 
186 } 
187 } 
188 } 
189 } 
190 } 
191 } 
192 } 
193 } 
194 } 
195 } 
196 } 
197 } 
198 } 
199 } 
200 } 
201 } 
202 } 
203 } 
204 } 
205 } 
206 } 
207 } 
208 } 
209 } 
210 } 
211 } 
212 } 
213 } 
214 } 
215 } 
216 } 
217 } 
218 } 
219 } 
220 } 
221 } 
222 } 
223 } 
224 } 
225 } 
226 } 
227 } 
228 } 
229 } 
230 } 
231 } 
232 } 
233 } 
234 } 
235 } 
236 } 
237 } 
238 } 
239 } 
240 } 
241 } 
242 } 
243 } 
244 } 
245 } 
246 } 
247 } 
248 } 
249 } 
250 } 
251 } 
252 } 
253 } 
254 } 
255 } 
256 } 
257 } 
258 } 
259 } 
260 } 
261 } 
262 } 
263 } 
264 } 
265 } 
266 } 
267 } 
268 } 
269 } 
270 } 
271 } 
272 } 
273 } 
274 } 
275 } 
276 } 
277 } 
278 } 
279 } 
280 } 
281 } 
282 } 
283 } 
284 } 
285 } 
286 } 
287 } 
288 } 
289 } 
290 } 
291 } 
292 } 
293 } 
294 } 
295 } 
296 } 
297 } 
298 } 
299 } 
300 } 
301 } 
302 } 
303 } 
304 } 
305 } 
306 } 
307 } 
308 } 
309 } 
310 } 
311 } 
312 } 
313 } 
314 } 
315 } 
316 } 
317 } 
318 } 
319 } 
320 } 
321 } 
322 } 
323 } 
324 } 
325 } 
326 } 
327 } 
328 } 
329 } 
330 } 
331 } 
332 } 
333 } 
334 } 
335 } 
336 } 
337 } 
338 } 
339 } 
340 } 
341 } 
342 } 
343 } 
344 } 
345 } 
346 } 
347 } 
348 } 
349 } 
350 } 
351 } 
352 } 
353 } 
354 } 
355 } 
356 } 
357 } 
358 } 
359 } 
360 } 
361 } 
362 } 
363 } 
364 } 
365 } 
366 } 
367 } 
368 } 
369 } 
370 } 
371 } 
372 } 
373 } 
374 } 
375 } 
376 } 
377 } 
378 } 
379 } 
380 } 
381 } 
382 } 
383 } 
384 } 
385 } 
386 } 
387 } 
388 } 
389 } 
390 } 
391 } 
392 } 
393 } 
394 } 
395 } 
396 } 
397 } 
398 } 
399 } 
400 } 
401 } 
402 } 
403 } 
404 } 
405 } 
406 } 
407 } 
408 } 
409 } 
410 } 
411 } 
412 } 
413 } 
414 } 
415 } 
416 } 
417 } 
418 } 
419 } 
420 } 
421 } 
422 } 
423 } 
424 } 
425 } 
426 } 
427 } 
428 } 
429 } 
430 } 
431 } 
432 } 
433 } 
434 } 
435 } 
436 } 
437 } 
438 } 
439 } 
440 } 
441 } 
442 } 
443 } 
444 } 
445 } 
446 } 
447 } 
448 } 
449 } 
450 } 
451 } 
452 } 
453 } 
454 } 
455 } 
456 } 
457 } 
458 } 
459 } 
460 } 
461 } 
462 } 
463 } 
464 } 
465 } 
466 } 
467 } 
468 } 
469 } 
470 } 
471 } 
472 } 
473 } 
474 } 
475 } 
476 } 
477 } 
478 } 
479 } 
480 } 
481 } 
482 } 
483 } 
484 } 
485 } 
486 } 
487 } 
488 } 
489 } 
490 } 
491 } 
492 } 
493 } 
494 } 
495 } 
496 } 
497 } 
498 } 
499 } 
500 } 
501 } 
502 } 
503 } 
504 } 
505 } 
506 } 
507 } 
508 } 
509 } 
510 } 
511 } 
512 } 
513 } 
514 } 
515 } 
516 } 
517 } 
518 } 
519 } 
520 } 
521 } 
522 } 
523 } 
524 } 
525 } 
526 } 
527 } 
528 } 
529 } 
530 } 
531 } 
532 } 
533 } 
534 } 
535 } 
536 } 
537 } 
538 } 
539 } 
540 } 
541 } 
542 } 
543 } 
544 } 
545 } 
546 } 
547 } 
548 } 
549 } 
550 } 
551 } 
552 } 
553 } 
554 } 
555 } 
556 } 
557 } 
558 } 
559 } 
560 } 
561 } 
562 } 
563 } 
564 } 
565 } 
566 } 
567 } 
568 } 
569 } 
570 } 
571 } 
572 } 
573 } 
574 } 
575 } 
576 } 
577 } 
578 } 
579 } 
580 } 
581 } 
582 } 
583 } 
584 } 
585 } 
586 } 
587 } 
588 } 
589 } 
590 } 
591 } 
592 } 
593 } 
594 } 
595 } 
596 } 
597 } 
598 } 
599 } 
600 } 
601 } 
602 } 
603 } 
604 } 
605 } 
606 } 
607 } 
608 } 
609 } 
610 } 
611 } 
612 } 
613 } 
614 } 
615 } 
616 } 
617 } 
618 } 
619 } 
620 } 
621 } 
622 } 
623 } 
624 } 
625 } 
626 } 
627 } 
628 } 
629 } 
630 } 
631 } 
632 } 
633 } 
634 } 
635 } 
636 } 
637 } 
638 } 
639 } 
640 } 
641 } 
642 } 
643 } 
644 } 
645 } 
646 } 
647 } 
648 } 
649 } 
650 } 
651 } 
652 } 
653 } 
654 } 
655 } 
656 } 
657 } 
658 } 
659 } 
660 } 
661 } 
662 } 
663 } 
664 } 
665 } 
666 } 
667 } 
668 } 
669 } 
670 } 
671 } 
672 } 
673 } 
674 } 
675 } 
676 } 
677 } 
678 } 
679 } 
680 } 
681 } 
682 } 
683 } 
684 } 
685 } 
686 } 
687 } 
688 } 
689 } 
690 } 
691 } 
692 } 
693 } 
694 } 
695 } 
696 } 
697 } 
698 } 
699 } 
700 } 
701 } 
702 } 
703 } 
704 } 
705 } 
706 } 
707 } 
708 } 
709 } 
710 } 
711 } 
712 } 
713 } 
714 } 
715 } 
716 } 
717 } 
718 } 
719 } 
720 } 
721 } 
722 } 
723 } 
724 } 
725 } 
726 } 
727 } 
728 } 
729 } 
730 } 
731 } 
732 } 
733 } 
734 } 
735 } 
736 } 
737 } 
738 } 
739 } 
740 } 
741 } 
742 } 
743 } 
744 } 
745 } 
746 } 
747 } 
748 } 
749 } 
750 } 
751 } 
752 } 
753 } 
754 } 
755 } 
756 } 
757 } 
758 } 
759 } 
760 } 
761 } 
762 } 
763 } 
764 } 
765 } 
766 } 
767 } 
768 } 
769 } 
770 } 
771 } 
772 } 
773 } 
774 } 
775 } 
776 } 
777 } 
778 } 
779 } 
780 } 
781 } 
782 } 
783 } 
784 } 
785 } 
786 } 
787 } 
788 } 
789 } 
790 } 
791 } 
792 } 
793 } 
794 } 
795 } 
796 } 
797 } 
798 } 
799 } 
800 } 
801 } 
802 } 
803 } 
804 } 
805 } 
806 } 
807 } 
808 } 
809 } 
810 } 
811 } 
812 } 
813 } 
814 } 
815 } 
816 } 
817 } 
818 } 
819 } 
820 } 
821 } 
822 } 
823 } 
824 } 
825 } 
826 } 
827 } 
828 } 
829 } 
830 } 
831 } 
832 } 
833 } 
834 } 
835 } 
836 } 
837 } 
838 } 
839 } 
840 } 
841 } 
842 } 
843 } 
844 } 
845 } 
846 } 
847 } 
848 } 
849 } 
850 } 
851 } 
852 } 
853 } 
854 } 
855 } 
856 } 
857 } 
858 } 
859 } 
860 } 
861 } 
862 } 
863 } 
864 } 
865 } 
866 } 
867 } 
868 } 
869 } 
870 } 
871 } 
872 } 
873 } 
874 } 
875 } 
876 } 
877 } 
878 } 
879 } 
880 } 
881 } 
882 } 
883 } 
884 } 
885 } 
886 } 
887 } 
888 } 
889 } 
890 } 
891 } 
892 } 
893 } 
894 } 
895 } 
896 } 
897 } 
898 } 
899 } 
900 } 
901 } 
902 } 
903 } 
904 } 
905 } 
906 } 
907 } 
908 } 
909 } 
910 } 
911 } 
912 } 
913 } 
914 } 
915 } 
916 } 
917 } 
918 } 
919 } 
920 } 
921 } 
922 } 
923 } 
924 } 
925 } 
926 } 
927 } 
928 } 
929 } 
930 } 
931 } 
932 } 
933 } 
934 } 
935 } 
936 } 
937 } 
938 } 
939 } 
940 } 
941 } 
942 } 
943 } 
944 } 
945 } 
946 } 
947 } 
948 } 
949 } 
950 } 
951 } 
952 } 
953 } 
954 } 
955 } 
956 } 
957 } 
958 } 
959 } 
960 } 
961 } 
962 } 
963 } 
964 } 
965 } 
966 } 
967 } 
968 } 
969 } 
970 } 
971 } 
972 } 
973 } 
974 } 
975 } 
976 } 
977 } 
978 } 
979 } 
980 } 
981 } 
982 } 
983 } 
984 } 
985 } 
986 } 
987 } 
988 } 
989 } 
990 } 
991 } 
992 } 
993 } 
994 } 
995 } 
996 } 
997 } 
998 } 
999 } 
1000 } 
1001 } 
1002 } 
1003 } 
1004 } 
1005 } 
1006 } 
1007 } 
1008 } 
1009 } 
1010 } 
1011 } 
1012 } 
1013 } 
1014 } 
1015 } 
1016 } 
1017 } 
1018 } 
1019 } 
1020 } 
1021 } 
1022 } 
1023 } 
1024 } 
1025 } 
1026 } 
1027 } 
1028 } 
1029 } 
1030 } 
1031 } 
1032 } 
1033 } 
1034 } 
1035 } 
1036 } 
1037 } 
1038 } 
1039 } 
1040 } 
1041 } 
1042 } 
1043 } 
1044 } 
1045 } 
1046 } 
1047 } 
1048 } 
1049 } 
1050 } 
1051 } 
1052 } 
1053 } 
1054 } 
1055 } 
1056 } 
1057 } 
1058 } 
1059 } 
1060 } 
1061 } 
1062 } 
1063 } 
1064 } 
1065 } 
1066 } 
1067 } 
1068 } 
1069 } 
1070 } 
1071 } 
1072 } 
1073 } 
1074 } 
1075 } 
1076 } 
1077 } 
1078 } 
1079 } 
1080 } 
1081 } 
1082 } 
1083 } 
1084 } 
1085 } 
1086 } 
1087 } 
1088 } 
1089 } 
1090 } 
1091 } 
1092 } 
1093 } 
1094 } 
1095 } 
1096 } 
1097 } 
1098 } 
1099 } 
1100 } 
1101 } 
1102 } 
1103 } 
1104 } 
1105 } 
1106 } 
1107 } 
1108 } 
1109 } 
1110 } 
1111 } 
1112 } 
1113 } 
1114 } 
1115 } 
1116 } 
1117 } 
1118 } 
1119 } 
1120 } 
1121 } 
1122 } 
1123 } 
1124 } 
1125 } 
1126 } 
1127 } 
1128 } 
1129 } 
1130 } 
1131 } 
1132 } 
1133 } 
1134 } 
1135 } 
1136 } 
1137 } 
1138 } 
1139 } 
1140 } 
1141 } 
1142 } 
1143 } 
1144 } 
1145 } 
1146 } 
1147 } 
1148 } 
1149 } 
1150 } 
1151 } 
1152 } 
1153 } 
1154 } 
1155 } 
1156 } 
1157 } 
1158 } 
1159 } 
1160 } 
1161 } 
1162 } 
1163 } 
1164 } 
1165 } 
1166 } 
1167 } 
1168 } 
1169 } 
1170 } 
1171 } 
1172 } 
1173 } 
1174 } 
1175 } 
1176 } 
1177 } 
1178 } 
1179 } 
1180 } 
1181 } 
1182 } 
1183 } 
1184 } 
1185 } 
1186 } 
1187 } 
1188 } 
1189 } 
1190 } 
1191 } 
1192 } 
1193 } 
1194 } 
1195 } 
1196 } 
1197 } 
1198 } 
1199 } 
1200 } 
1201 } 
1202 } 
1203 } 
1204 } 
1205 } 
1206 } 
1207 } 
1208 } 
1209 } 
1210 } 
1211 } 
1212 } 
1213 } 
1214 } 
1215 } 
1216 } 
1217 } 
1218 } 
1219 } 
1220 } 
1221 } 
1222 } 
1223 } 
1224 } 
1225 } 
1226 } 
1227 } 
1228 } 
1229 } 
1230 } 
1231 } 
1232 } 
1233 } 
1234 } 
1235 } 
1236 } 
1237 } 
1238 } 
1239 } 
1240 } 
1241 } 
1242 } 
1243 } 
1244 } 
1245 } 
1246 } 
1247 } 
1248 } 
1249 } 
1250 } 
1251 } 
1252 } 
1253 } 
1254 } 
1255 } 
1256 } 
1257 } 
1258 } 
1259 } 
1260 } 
1261 } 
1262 } 
1263 } 
1264 } 
1265 } 
1266 } 
1267 } 
1268 } 
1269 } 
1270 } 
1271 } 
1272 } 
1273 } 
1274 } 
1275 } 
1276 } 
1277 } 
1278 } 
1279 } 
1280 } 
1281 } 
1282 } 
1283 } 
1284 } 
1285 } 
1286 } 
1287 } 
1288 } 
1289 } 
1290 } 
1291 } 
1292 } 
1293 } 
1294 } 
1295 } 
1296 } 
1297 } 
1298 } 
1299 } 
1300 } 
1301 } 
1302 } 
1303 } 
1304 } 
1305 } 
1306 } 
1307 } 
1308 } 
1309 } 
1310 } 
1311 } 
1312 } 
1313 } 
1314 } 
1315 } 
1316 } 
1317 } 
1318 } 
1319 } 
1320 } 
1321 } 
1322 } 
1323 } 
1324 } 
1325 } 
1326 } 
1327 } 
1328 } 
1329 } 
1330 } 
1331 } 
1332 } 
1333 } 
1334 } 
1335 } 
1336 } 
1337 } 
1338 } 
1339 } 
1340 } 
1341 } 
1342 } 
1343 } 
1344 } 
1345 } 
1346 } 
1347 } 
1348 } 
1349 } 
1350 } 
1351 } 
1352 } 
1353 } 
1354 } 
1355 } 
1356 } 
1357 } 
1358 } 
1359 } 
1360 } 
1361 } 
1362 } 
1363 } 
1364 } 
1365 } 
1366 } 
1367 } 
1368 } 
1369 } 
1370 } 
1371 } 
1372 } 
1373 } 
1374 } 
1375 } 
1376 } 
1377 } 
1378 } 
1379 } 
1380 } 
1381 } 
1382 } 
1383 } 
1384 } 
1385 } 
1386 } 
1387 } 
1388 } 
1389 } 
1390 } 
1391 } 
1392 } 
1393 } 
1394 } 
1395 } 
1396 } 
1397 } 
1398 } 
1399 } 
1400 } 
1401 } 
1402 } 
1403 } 
1404 } 
1405 } 
1406 } 
1407 } 
1408 } 
1409 } 
1410 } 
1411 } 
1412 } 
1413 } 
1414 } 
1415 } 
1416 } 
1417 } 
1418 } 
1419 } 
1420 } 
1421 } 
1422 } 
1423 } 
1424 } 
1425 } 
1426 } 
1427 } 
1428 } 
1429 } 
1430 } 
1431 } 
1432 } 
1433 } 
1434 } 
1435 } 
1436 } 
1437 } 
1438 } 
1439 } 
1440 } 
1441 } 
1442 } 
1443 } 
1444 } 
1445 } 
1446 } 
1447 } 
1448 } 
1449 } 
1450 } 
1451 } 
1452 } 
1453 } 
1454 } 
1455 } 
1456 } 
1457 } 
1458 } 
1459 } 
1460 } 
1461 } 
1462 } 
1463 } 
1464 } 
1465 } 
1466 } 
1467 } 
1468 } 
1469 } 
1470 } 
1471 } 
1472 } 
1473 } 
1474 } 
1475 } 
1476 } 
1477 } 
1478 } 
1479 } 
1480 } 
1481 } 
1482 } 
1483 } 
1484 } 
1485 } 
1486 } 
1487 } 
1488 } 
1489 } 
1490 } 
1491 } 
1492 } 
1493 } 
1494 } 
1495 } 
1496 } 
1497 } 
1498 } 
1499 } 
1500 } 
1501 } 
1502 } 
1503 } 
1504 } 
1505 } 
1506 } 
1507 } 
1508 } 
1509 } 
1510 } 
1511 } 
1512 } 
1513 } 
1514 } 
1515 } 
1516 } 
1517 } 
1518 } 
1519 } 
1520 } 
1521 } 
1522 } 
1523 } 
1524 } 
1525 } 
1526 } 
1527 } 
1528 } 
1529 } 
1530 } 
1531 } 
1532 } 
1533 } 
1534 } 
1535 } 
1536 } 
1537 } 
1538 } 
1539 } 
1540 } 
1541 } 
1542 } 
1543 } 
1544 } 
1545 } 
1546 } 
1547 } 
1548 } 
1549 } 
1550 } 
1551 } 
1552 } 
1553 } 
1554 } 
1555 } 
1556 } 
1557 } 
1558 } 
1559 } 
1560 } 
1561 } 
1562 } 
1563 } 
1564 } 
1565 } 
1566 } 
1567 } 
1568 } 
1569 } 
1570 } 
1571 } 
1572 } 
1573 } 
1574 } 
1575 } 
1576 } 
1577 } 
1578 } 
1579 } 
1580 } 
1581 } 
1582 } 
1583 } 
1584 } 
1585 } 
1586 } 
1587 } 
1588 } 
1589 } 
1590 } 
1591 } 
1592 } 
1593 } 
1594 } 
1595 } 
1596 } 
1597 } 
1598 } 
1599 } 
1600 } 
1601 } 
1602 } 
1603 } 
1604 } 
1605 } 
1606 } 
1607 } 
1608 } 
1609 } 
1610 } 
1611 } 
1612 } 
1613 } 
1614 } 
1615 } 
1616 } 
1617 } 
1618 } 
1619 } 
1620 } 
1621 } 
1622 } 
1623 } 
1624 } 
1625 } 
1626 } 
1627 } 
1628 } 
1629 } 
1630 } 
1631 } 
1632 } 
1633 } 
1634 } 
1635 } 
1636 } 
1637 } 
1638 } 
1639 } 
1640 } 
1641 } 
1642 } 
1643 } 
1644 } 
1645 } 
1646 } 
1647 } 
1648 } 
1649 } 
1650 } 
1651 } 
1652 } 
1653 } 
1654 } 
1655 } 
1656 } 
1657 } 
1658 } 
1659 } 
1660 } 
1661 } 
1662 } 
1663 } 
1664 } 
1665 } 
1666 } 
1667 } 
1668 } 
1669 } 
1670 } 
1671 } 
1672 } 
1673 } 
1674 } 
1675 } 
1676 } 
1677 } 
1678 } 
1679 } 
1680 } 
1681 } 
1682 } 
1683 } 
1684 } 
1685 } 
1686 } 
1687 } 
1688 } 
1689 } 
1690 } 
1691 } 
1692 } 
1693 } 
1694 } 
1695 } 
1696 } 
1697 } 
1698 } 
1699 } 
1700 } 
1701 } 
1702 } 
1703 } 
1704 } 
1705 } 
1706 } 
1707 } 
1708 } 
1709 } 
1710 } 
1711 } 
1712 } 
1713 } 
1714 } 
1715 } 
1716 } 
1717 } 
1718 } 
1719 } 
1720 } 
1721 } 
1722 } 
1723 } 
1724 } 
1725 } 
1726 } 
1727 } 
1728 } 
1729 } 
1730 } 
1731 } 
1732 } 
1733 } 
1734 } 
1735 } 
1736 } 
1737 } 
1738 } 
1739 } 
1740 } 
1741 } 
1742 } 
1743 } 
1744 } 
1745 } 
1746 } 
1747 } 
1748 } 
1749 } 
1750 } 
1751 } 
1752 } 
1753 } 
1754 } 
1755 } 
1756 } 
1757 } 
1758 } 
1759 } 
1760 } 
1761 } 
1762 } 
1763 } 
1764 } 
1765 } 
1766 } 
1767 } 
1768 } 
1769 } 
1770 } 
1771 } 
1772 } 
1773 } 
1774 } 
1775 } 
1776 } 
1777 } 
1778 } 
1779 } 
1780 } 
1781 } 
1782 } 
1783 } 
1784 } 
1785 } 
1786 } 
1787 } 
1788 } 
1789 } 
1790 } 
1791 } 
1792 } 
1793 } 
1794 } 
1795 } 
1796 } 
1797 } 
1798 } 
1799 } 
1800 } 
1801 } 
1802 } 
1803 } 
1804 } 
1805 } 
1806 } 
1807 } 
1808 } 
1809 } 
1810 } 
1811 } 
1812 } 
1813 } 
1814 } 
1815 } 
1816 } 
1817 } 
1818 } 
1819 } 
1820 } 
1821 } 
1822 } 
1823 } 
1824 } 
1825 } 
1826 } 
1827 } 
1828 } 
1829 } 
1830 } 
1831 } 
1832 } 
1833 } 
1834 } 
1835 } 
1836 } 
1837 } 
1838 } 
1839 } 
1840 } 
1841 } 
1842 } 
1843 } 
1844 } 
1845 } 
1846 } 
1847 } 
1848 } 
1849 } 
1850 } 
1851 } 
1852 } 
1853 } 
1854 } 
1855 } 
1856 } 
1857 } 
1858 } 
1859 } 
1860 } 
1861 } 
1862 } 
1863 } 
1864 } 
1865 } 
1866 } 
1867 } 
1868 } 
1869 } 
1870 } 
1871 } 
1872 } 
1873 } 
1874 } 
1875 } 
1876 } 
1877 } 
1878 } 
1879 } 
1880 } 
1881 } 
1882 } 
1883 } 
1884 } 
1885 } 
1886 } 
1887 } 
1888 } 
1889 } 
1890 } 
1891 } 
1892 } 
1893 } 
1894 } 
1895 } 
1896 } 
1897 } 
1898 } 
1899 } 
1900 } 
1901 } 
1902 } 
1903 } 
1904 } 
1905 } 
1906 } 
1907 } 
1908 } 
1909 } 
1910 } 
1911 } 
1912 } 
1913 } 
1914 } 
1915 } 
1916 } 
1917 } 
1918 } 
1919 } 
1920 } 
1921 } 
1922 } 
1923 } 
1924 } 
1925 } 
1926 } 
1927 } 
1928 } 
1929 } 
1930 } 
1931 } 
1932 } 
1933 } 
1934 } 
1935 } 
1936 } 
1937 } 
1938 } 
1939 } 
1940 } 
1941 } 
1942 } 
1943 } 
1944 } 
1945 } 
1946 } 
1947 } 
1948 } 
1949 } 
1950 } 
1951 } 
1952 } 
1953 } 
1954 } 
1955 } 
1956 } 
1957 } 
1958 } 
1959 } 
1960 } 
1961 } 
1962 } 
1963 } 
1964 } 
1965 } 
1966 } 
1967 } 
1968 } 
1969 } 
1970 } 
1971 } 
1972 } 
1973 } 
1974 } 
1975 } 
1976 } 
1977 } 
1978 } 
1979 } 
1980 } 
1981 } 
1982 } 
1983 } 
1984 } 
1985 } 
1986 } 
1987 } 
1988 } 
1989 } 
1990 } 
1991 } 
1992 } 
1993 } 
1994 } 
1995 } 
1996 } 
1997 } 
1998 } 
1999 } 
2000 } 
2001 } 
2002 } 
2003 } 
2004 } 
2005 } 
2006 } 
2007 } 
2008 } 
2009 } 
2010 } 
2011 } 
2012 } 
2013 } 
2014 } 
2015 } 
2016 } 
2017 } 
2018 } 
2019 } 
2020 } 
2021 } 
2022 } 
2023 } 
2024 } 
2025 } 
2026 } 
2027 } 
2028 } 
2029 } 
2030 } 
2031 } 
2032 } 
2033 } 
2034 } 
2035 } 
2036 } 
2037 } 
2038 } 
2039 } 
2040 } 
2041 } 
2042 } 
2043 } 
2044 } 
2045 } 
2046 } 
2047 } 
2048 } 
2049 } 
2050 } 
2051 } 
2052 } 
2053 } 
2054 } 
205
```

## 1.12 Open the terminal

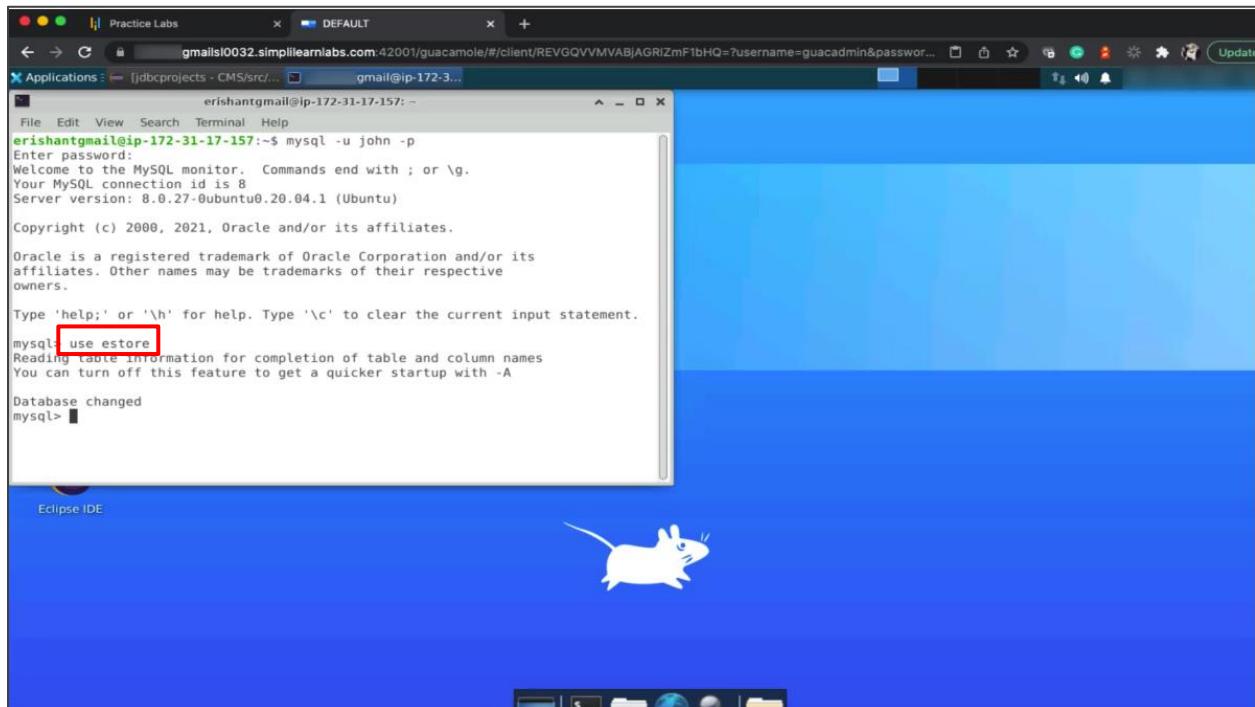


## 1.13 Log in into SQL using the command mysql -u john -p and the password is john.

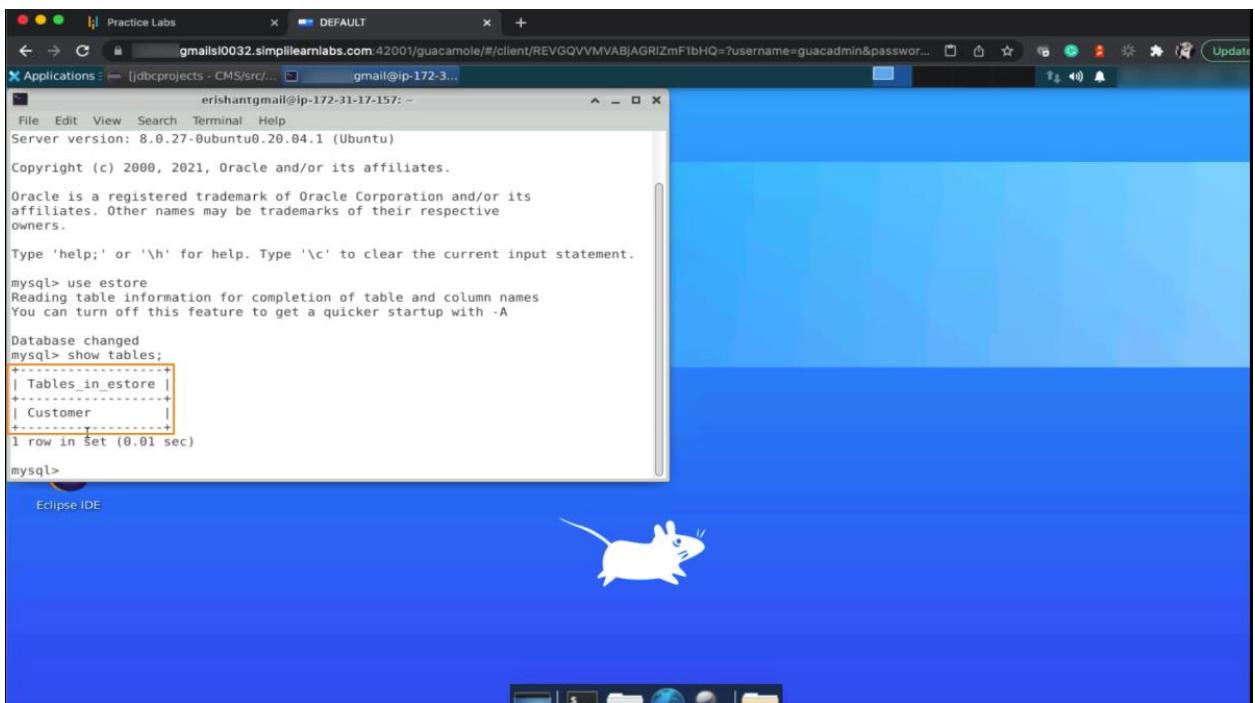


**Note:** A user named **john** has already been created for the database.

### 1.14 Run command **use estore;** to change the database



### 1.15 Run the **show tables;** command to check available tables in the **estore** database



### 1.16 Run the **describe Customer;** command to see table details

```

+-----+
| Tables_in_estore |
+-----+
| Customer        |
+-----+
1 row in set (0.01 sec)

mysql> describe Customer;
+-----+-----+-----+-----+-----+
| Field | Type   | Null | Key | Default | Extra       |
+-----+-----+-----+-----+-----+
| cid   | int    | NO   | PRI | NULL    | auto_increment |
| name  | varchar(256)| YES  |     | NULL    |              |
| phone | varchar(28) | YES  |     | NULL    |              |
| email | varchar(256)| YES  |     | NULL    |              |
| birthDate | date | YES  |     | NULL    |              |
| age   | int    | YES  |     | NULL    |              |
| inDateTime | datetime | YES  |     | NULL    |              |
| outDateTime | datetime | YES  |     | NULL    |              |
| temperature | float | YES  |     | NULL    |              |
+-----+-----+-----+-----+-----+
9 rows in set (0.01 sec)

mysql>

```

### 1.17 Run the select command as **select \* from Customer;** to see data from the table

```

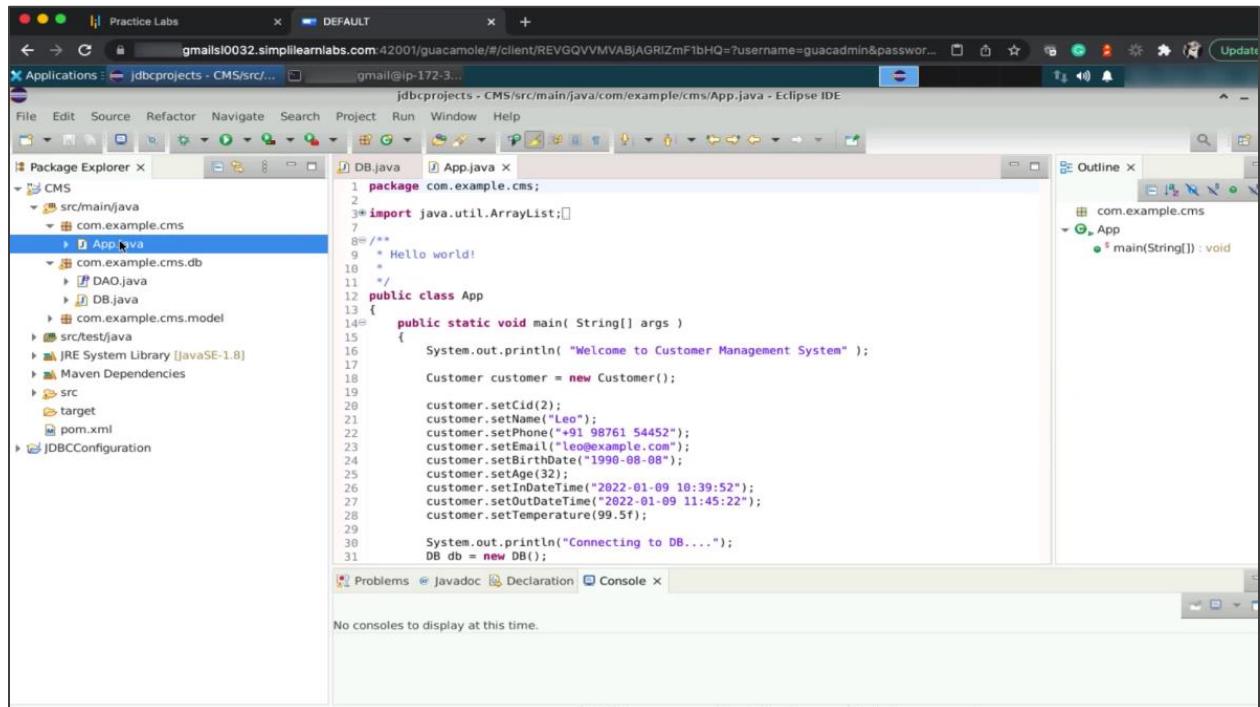
+-----+
| birthDate | date | YES |     | NULL    |           |
| age       | int  | YES |     | NULL    |           |
| inDateTime | datetime | YES |     | NULL    |           |
| outDateTime | datetime | YES |     | NULL    |           |
| temperature | float | YES |     | NULL    |           |
+-----+
9 rows in set (0.01 sec)

mysql> select * from Customer;
+-----+-----+-----+-----+-----+
| cid | name      | phone      | email      | birthDate | a |
| age | inDateTime | outDateTime | temperature |          | a |
+-----+-----+-----+-----+-----+
| 2  | John Watson | +91 98761 22222 | john.watson@example.com | 1990-08-08 |
| 32 | 2022-01-08 10:39:52 | 2022-01-08 11:45:22 | 98.5 |          |
| 3  | Leo         | +91 98761 54452 | leo@example.com | 1990-08-08 |
| 32 | 2022-01-09 10:39:52 | 2022-01-09 11:45:22 | 99.5 |          |
+-----+-----+-----+-----+-----+
2 rows in set (0.00 sec)

mysql>

```

### 1.18 Go back to Eclipse and open the App.java file to use the insert query

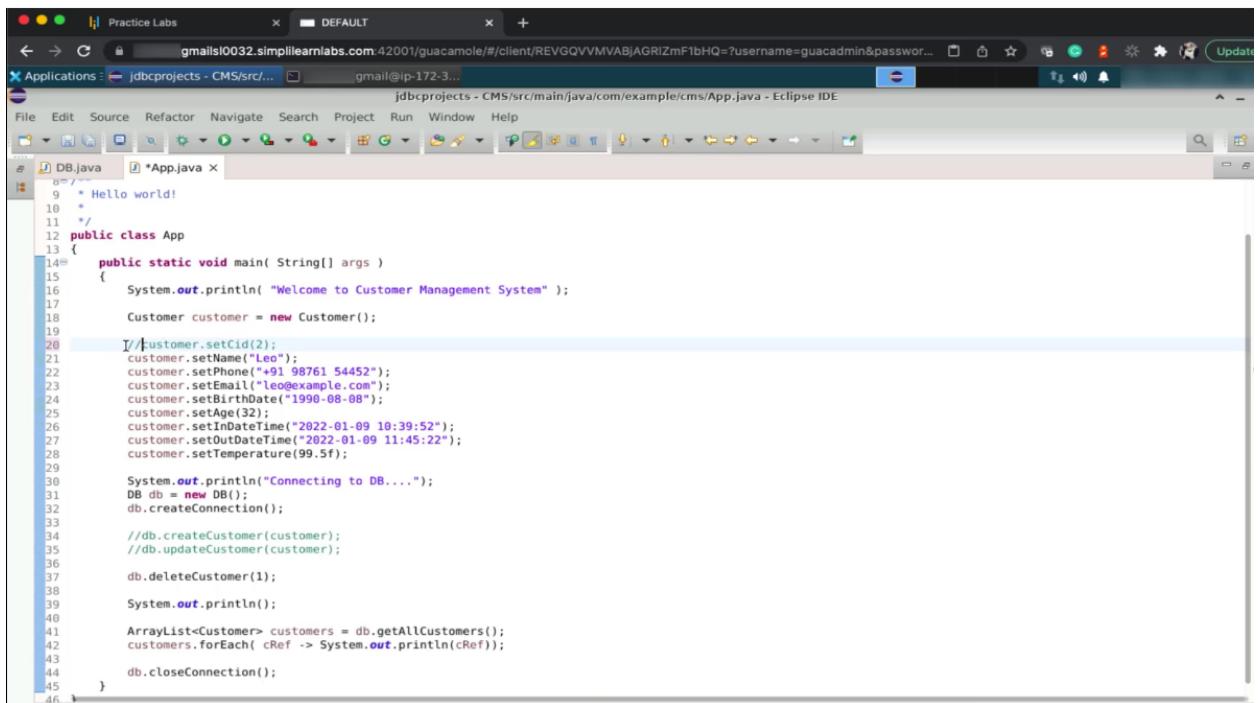


```

package com.example.cms;
import java.util.ArrayList;
/*
 * Hello world!
 */
public class App
{
    public static void main( String[] args )
    {
        System.out.println( "Welcome to Customer Management System" );
        Customer customer = new Customer();
        customer.setCid(2);
        customer.setName("Leo");
        customer.setPhone("+91 98761 54452");
        customer.setEmail("leo@example.com");
        customer.setBirthDate("1990-08-08");
        customer.setAge(32);
        customer.setInDateTime("2022-01-09 10:39:52");
        customer.setOutDateTime("2022-01-09 11:45:22");
        customer.setTemperature(99.5f);
        System.out.println("Connecting to DB....");
        DB db = new DB();
    }
}

```

### 1.19 Comment out Cid as it is an auto-increment



```

/*
 * Hello world!
 */
public class App
{
    public static void main( String[] args )
    {
        System.out.println( "Welcome to Customer Management System" );
        Customer customer = new Customer();
        customer.setCid(2);
        customer.setName("Leo");
        customer.setPhone("+91 98761 54452");
        customer.setEmail("leo@example.com");
        customer.setBirthDate("1990-08-08");
        customer.setAge(32);
        customer.setInDateTime("2022-01-09 10:39:52");
        customer.setOutDateTime("2022-01-09 11:45:22");
        customer.setTemperature(99.5f);
        System.out.println("Connecting to DB....");
        DB db = new DB();
        db.createConnection();
        db.createCustomer(customer);
        db.updateCustomer(customer);
        db.deleteCustomer(1);
        System.out.println();
        ArrayList<Customer> customers = db.getAllCustomers();
        customers.forEach(cRef -> System.out.println(cRef));
        db.closeConnection();
    }
}

```

## 1.20 Change the details in the customer fields to create a new customer dataset

```
16     System.out.println("Welcome to Customer Management System ");
17
18     Customer customer = new Customer();
19
20     //customer.setCid(2);
21     customer.setName("George");
22     customer.setPhone("+91 98761 54452");
23     customer.setEmail("leorge@example.com");
24     customer.setBirthDate("1990-08-08");
25     customer.setAge(32);
26     customer.setInDateTime("2022-01-09 10:39:52");
27     customer.setOutDateTime("2022-01-09 11:45:22");
28     customer.setTemperature(99.5f);
29
30     System.out.println("Connecting to DB...");
31     DB db = new DB();
32     db.createConnection();
```

```
19
20     //customer.setCid(2);
21     customer.setName("George");
22     customer.setPhone("+91 99009 54452");
23     customer.setEmail("leorge@example.com");
24     customer.setBirthDate("1990-08-08");
25     customer.setAge(32);
26     customer.setInDateTime("2022-01-09 10:39:52");
27     customer.setOutDateTime("2022-01-09 11:45:22");
28     customer.setTemperature(99.5f);
29
30     System.out.println("Connecting to DB...");
31     DB db = new DB();
32     db.createConnection();
```

```
19
20     //customer.setCid(2);
21     customer.setName("George");
22     customer.setPhone("+91 99009 88008");
23     customer.setEmail("george@example.com");
24     customer.setBirthDate("1986-08-08");
25     customer.setAge(32);
26     customer.setInDateTime("2022-01-09 10:39:52");
27     customer.setOutDateTime("2022-01-09 11:45:22");
28     customer.setTemperature(99.5f);
29
30     System.out.println("Connecting to DB...");
31     DB db = new DB();
32     db.createConnection();
```

```
19
20     //customer.setCid(2);
21     customer.setName("George");
22     customer.setPhone("+91 99009 88008");
23     customer.setEmail("george@example.com");
24     customer.setBirthDate("1986-04-04");
25     customer.setAge(32);
26     customer.setInDateTime("2022-01-09 10:39:52");
27     customer.setOutDateTime("2022-01-09 11:45:22");
28     customer.setTemperature(99.5f);
29
30     System.out.println("Connecting to DB...");
31     DB db = new DB();
32     db.createConnection();
```

```
19
20     //customer.setCid(2);
21     customer.setName("George");
22     customer.setPhone("+91 99009 88008");
23     customer.setEmail("george@example.com");
24     customer.setBirthDate("1986-04-04");
25     customer.setAge(32);
26     customer.setInDateTime("2022-01-12 10:39:52");
27     customer.setOutDateTime("2022-01-12 11:45:22");
28     customer.setTemperature(99.5f);
29
30     System.out.println("Connecting to DB...");
31     DB db = new DB();
32     db.createConnection();
```

## 1.21 Run the code, and the output can be seen in the console as **Customer Created Successfully**

```

DB.java
9 * Hello world!
10 */
11 */
12 public class App
13 {
14     public static void main( String[] args )
15     {
16         System.out.println( "Welcome to Customer Management System" );
17
18         Customer customer = new Customer();
19
20         //customer.setId(2);
21         customer.setName("George");
22         customer.setPhone("+91 99009 88888");
23         customer.setEmail("george@example.com");
24         customer.setBirthDate("1980-04-04");
25         customer.setAge(32);
26         customer.setInDateTime("2022-01-12 10:39:52");
27         customer.setOutDateTime("2022-01-12 11:45:22");
28         customer.setTemperature(99.5f);
29
30         System.out.println("Connecting to DB....");
31         DB db = new DB();
32         db.createConnection();
33
34         db.createCustomer(customer);
35         //db.updateCustomer(customer);
36
37         //db.deleteCustomer(l);
38
39         //System.out.println();
40
41         //ArrayList<Customer> customers = db.getAllCustomers();
42         //customers.forEach( cRef -> System.out.println(cRef));
43
44         db.closeConnection();
45     }
46

```

Console Output:

```

terminated> App [Java Application] /usr/eclipse/plugins/org.eclipse.jdt.core/openjdk.hotspot
Welcome to Customer Management System
Connecting to DB...
[DB] Driver Loaded
[DB] Connection Created
SQL Is: insert into Customer values(null, ?, ?, ?, ?, ?, ?, ?, ?)
[DB] Customer Created Successfully
[DB] Connection Closed. Close Status: true

```

## 1.22 Go back to the terminal and run the **select** command

```

File Edit View Search Terminal Help
mysql> select * from Customer;
+-----+-----+-----+-----+-----+-----+
| birthDate | date | YES | NULL | NULL |
+-----+-----+-----+-----+-----+
9 rows in set (0.01 sec)

mysql> select * from Customer;
+-----+-----+-----+-----+-----+-----+
| cid | name | phone | email | birthDate | a
ge | inDateTime | outDateTime | temperature |
+-----+-----+-----+-----+-----+-----+
| 2 | John Watson | +91 98761 22222 | john.watson@example.com | 1990-08-08 |
| 32 | 2022-01-08 10:39:52 | 2022-01-08 11:45:22 | 98.5 |
| 3 | Leo | +91 98761 54542 | leo@example.com | 1990-08-08 |
| 32 | 2022-01-09 10:39:52 | 2022-01-09 11:45:22 | 99.5 |
+-----+-----+-----+-----+-----+
2 rows in set (0.00 sec)

mysql> select * from Customer;
db.createCustomer(customer);
//db.updateCustomer(customer);

```

A record inserted can now be seen in the output as:

```

Applications - jdbcprojects - CMS/src... gmail@ip-172-3...
File Edit Source Refactor Navigate Search Project Run Window Help
erishant@gmail.com:ip-172-3-157: ~
DB.java
mysql> select * from Customer;
+----+-----+-----+-----+-----+-----+
| id | name | phone | email | birthDate | temperature |
+----+-----+-----+-----+-----+-----+
| 3 | Leo | +91 98761 54452 | leo@example.com | 1990-08-08 | 99.5 |
| 32 | 2022-01-09 10:39:52 | 2022-01-09 11:45:22 | | 99.5 |
+----+-----+-----+-----+-----+-----+
2 rows in set (0.00 sec)

mysql> insert into Customer values(null, ?, ?, ?, ?, ?, ?, ?);
Query OK, 1 row affected (0.00 sec)

mysql> select * from Customer;
+----+-----+-----+-----+-----+-----+
| id | name | phone | email | birthDate | temperature |
+----+-----+-----+-----+-----+-----+
| 2 | John Watson | +91 98761 22222 | john.watson@example.com | 1990-08-08 | 98.5 |
| 3 | Leo | +91 98761 54452 | leo@example.com | 1990-08-08 | 99.5 |
| 32 | 2022-01-09 10:39:52 | 2022-01-09 11:45:22 | | 99.5 |
| 4 | George | +91 99009 88088 | george@example.com | 1986-04-04 | 98.5 |
| 32 | 2022-01-12 10:39:52 | 2022-01-12 11:45:22 | | 99.5 |
+----+-----+-----+-----+-----+-----+
3 rows in set (0.00 sec)

mysql>
db.createCustomer(customer);
//db.updateCustomer(customer);
//db.deleteCustomer(1);
//System.out.println();
//ArrayList<Customer> customers = db.getAllCustomers();
//customers.forEach( cRef -> System.out.println(cRef));
db.closeConnection();
}
}

```

## Step 2: Perform the Update operation

2.1 Open Eclipse. Go to the **DB.java** file and find the **updateCustomer()** method. Comment the SQL string and substitute it with the wildcard characters.

```

Practice Labs x DEFAULT x +
gmails003.simplilearnlabs.com:42001/guacamole/#/client/REVGQVVMVABJAGRIZmF1bHQ=?username=guacadmin&password=guacadmin
Applications - jdbcprojects - CMS/src... gmail@ip-172-3...
File Edit Source Refactor Navigate Search Project Run Window Help
DB.java x App.java
103 //int result = statement.executeUpdate(sql);
104 int result = preparedStatement.executeUpdate();
105 String message = result > 0 ? "Customer Created Successfully" : "Customer Not Created. Please Try Again";
106 System.out.println(TAG+message);
107 }
108 } catch (Exception e) {
109 System.out.println("Exception Occurred: "+e);
110 }
111 }
112 }
113 }
114 public void updateCustomer(Customer customer) {
115 try {
116 // String sql = "Update Customer set name = '"+customer.getName()+"', phone = '"+customer.getPhone()+"', email = '"+customer.getEmail()+"",
117 // "", birthDate = '"+customer.getBirthDate()+"', age = "+customer.getAge()+"', inDateTime = '"+customer.getInDateTime()+"', outDateTime = '"+customer.getOutTime()+"',
118 // "+ where cid = "+customer.getId();
119 // String sql = "update Customer set name = '"+customer.getName()+"', phone = '"+customer.getPhone()+"', email = '"+customer.getEmail()+"",
120 // "+", birthDate = "+customer.getBirthDate()+"", age = "+customer.getAge()+"", inDateTime = "+customer.getInDateTime()+"", outDateTime = "+customer.getOutTime()+"",
121 // "+ where cid = "+customer.getId();
122 String sql = "update Customer set name = "+customer.getName()+", phone = "+customer.getPhone()+", email = "+customer.getEmail()+",
123 // "+", birthDate = "+customer.getBirthDate()+"", age = "+customer.getAge()+"", inDateTime = "+customer.getInDateTime()+"", outDateTime = "+customer.getOutTime()+"",
124 // "+ where cid = "+customer.getId();
125 System.out.println("SQL is: "+sql);
126 int result = statement.executeUpdate(sql);
127 String message = result > 0 ? "Customer Updated Successfully" : "Customer Not Updated. Please Try Again";
128 System.out.println(TAG+message);
129 }
130 } catch (Exception e) {
131 System.out.println("Exception Occurred: "+e);
132 }
133 }
134 }
135 }
136 }

```

## 2.2 Write the lines of code from 124 to 141 in which we are creating the **preparedStatement** for updating the customer details

```

108     }
109     catch (Exception e) {
110         System.out.println("Exception Occurred: "+e);
111     }
112 }
113
114 public void updateCustomer(Customer customer) {
115     try {
116
117         String sql = "update Customer set name = '"+customer.getName()+"', phone = '"+customer.getPhone()+"', email = '"+customer.getEmail()+""
118         //      , birthDate = '"+customer.getBirthDate()+"', age = '"+customer.getAge()+"', inDateTime = '"+customer.getInDateTime()+"', outDateTime = '"+customer.getOutDateTime()+"'
119         //      where cid = "+customer.getCid();
120
121         String sql = "update Customer set name = ?, phone = ?, email = ?, birthDate = ?, age = ?, inDateTime = ?, outDateTime = ?, temperature = ? where cid = ?";
122
123         System.out.println("SQL is: "+sql);
124
125         preparedStatement = connection.prepareStatement(sql);
126
127         preparedStatement.setString(1, customer.getName());
128         preparedStatement.setString(2, customer.getPhone());
129         preparedStatement.setString(3, customer.getEmail());
130         preparedStatement.setString(4, customer.getBirthDate());
131         preparedStatement.setInt(5, customer.getAge());
132         preparedStatement.setString(6, customer.getInDateTime());
133         preparedStatement.setString(7, customer.getOutDateTime());
134         preparedStatement.setFloat(8, customer.getTemperature());
135
136         preparedStatement.setInt(9, customer.getCid());
137
138         //int result = statement.executeUpdate(sql);
139         int result = preparedStatement.executeUpdate();
140         String message = result > 0 ? "Customer Updated Successfully" : "Customer Not Updated. Please Try Again";
141         System.out.println(TAG+message);
142
143     } catch (Exception e) {
144         System.out.println("Exception Occurred: "+e);
145     }

```

## 2.3 Go to the **App.java** file, uncomment the **updateCustomer** call, and comment on the **createCustomer** call. Make a few changes in customer details to update.

```

8 /**
9  * Hello world!
10 */
11
12 public class App
13 {
14     public static void main( String[] args )
15     {
16         System.out.println( "Welcome to Customer Management System" );
17
18         Customer customer = new Customer();
19
20         customer.setCid(3);
21         customer.setName("George G");
22         customer.setPhone("+91 99009 11111");
23         customer.setEmail("george.g@example.com");
24         customer.setBirthDate("1986-04-04");
25         customer.setAge(35);
26         customer.setInDateTime("2022-01-12 09:00:00");
27         customer.setOutDateTime("2022-01-12 10:30:00");
28         customer.setTemperature(98.8f);
29
30         System.out.println("Connecting to DB....");
31         DB db = new DB();
32         db.createConnection();
33

```

2.4 Run the code with changes. You can see the updated customer details.

```

 1 // DB.java
 2 /**
 3  * Hello world!
 4  */
 5
 6 public class App
 7 {
 8     public static void main(String[] args)
 9     {
10         System.out.println("Welcome to Customer Management System");
11
12         Customer customer = new Customer();
13
14         customer.setId(3);
15         customer.setName("George G");
16         customer.setPhone("+91 99009 1111");
17         customer.setEmail("george.g@example.com");
18         customer.setBirthDate("1986-04-04");
19         customer.setAge(35);
20         customer.setInDateTime("2022-01-12 09:00:00");
21         customer.setOutDateTime("2022-01-12 10:30:00");
22         customer.setTemperature(98.8f);
23
24         System.out.println("Connecting to DB...");
25         DB db = new DB();
26         db.createConnection();
27
28         //db.createCustomer(customer);
29         db.updateCustomer(customer);
30
31         //db.deleteCustomer(1);
32
33         //System.out.println();
34     }
35 }

```

<terminated> App [Java Application] /usr/eclipse/plugins/org.eclipse.jdt.core/openjdk.hotspot.Welcome to Customer Management System  
Connecting to DB...  
[DB] Driver Loaded  
[DB] Connection Created  
SQL is: update Customer set name = ?, phone = ?, email = ?, birthDate = ?  
[DB] Customer Updated Successfully  
[DB] Connection Closed. Close Status: true

2.5 Open the terminal and run the **select** command. You can see a customer with **id = 3** has been updated.

```

 1 // DB.java
 2 /**
 3  * Hello world!
 4  */
 5
 6 public class App
 7 {
 8     public static void main(String[] args)
 9     {
10         System.out.println("Connecting to DB...");
11         DB db = new DB();
12         db.createConnection();
13
14         //db.createCustomer(customer);
15         db.updateCustomer(customer);
16
17         //db.deleteCustomer(1);
18
19         //System.out.println();
20
21         //ArrayList<Customer> customers = db.getAllCustomers();
22         //customers.forEach(cRef -> System.out.println(cRef));
23
24         db.closeConnection();
25     }
26 }

```

erishant@gmail@ip-172-31-17-157: ~

cid	name	phone	email	inDateTime	outDateTime	temperature	birthDate	age
32	John Watson	+91 98761 22222	john.watson@example.com	2022-01-08 10:39:52	2022-01-08 11:45:22	98.5	1990-08-08	35
32	Leo	+91 98761 54542	leo@example.com	2022-01-09 10:39:52	2022-01-09 11:45:22	99.5	1990-08-08	35
32	George G	+91 99009 11111	george.g@example.com	2022-01-12 09:00:00	2022-01-12 10:30:00	98.8	1986-04-04	34
32	George G	+91 99009 88088	george@example.com	2022-01-12 10:39:52	2022-01-12 11:45:22	99.5	1986-04-04	35

3 rows in set (0.00 sec)

mysql> select \* from Customer;

cid	name	phone	email	inDateTime	outDateTime	temperature	birthDate	age
32	John Watson	+91 98761 22222	john.watson@example.com	2022-01-08 10:39:52	2022-01-08 11:45:22	98.5	1990-08-08	35
32	Leo	+91 98761 54542	leo@example.com	2022-01-09 10:39:52	2022-01-09 11:45:22	99.5	1990-08-08	35
32	George G	+91 99009 11111	george.g@example.com	2022-01-12 09:00:00	2022-01-12 10:30:00	98.8	1986-04-04	34
32	George G	+91 99009 88088	george@example.com	2022-01-12 10:39:52	2022-01-12 11:45:22	99.5	1986-04-04	35

3 rows in set (0.00 sec)

<terminated> App [Java Application] /usr/eclipse/plugins/org.eclipse.jdt.core/openjdk.hotspot.Welcome to Customer Management System  
Connecting to DB...  
[DB] Driver Loaded  
[DB] Connection Created  
SQL is: update Customer set name = ?, phone = ?, email = ?, birthDate = ?  
[DB] Customer Updated Successfully  
[DB] Connection Closed. Close Status: true

### **Step 3: Perform the Delete operation**

3.1 Use the **delete** operation, use the **executeUpdate()** function and print a message for the successful deletion of the record (line 151 to line 160)

The screenshot shows the Eclipse IDE interface with the title bar "Applications - jdbcprojects - CMS/src...". The menu bar includes File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, Help. The toolbar has various icons for file operations. The left sidebar shows two open files: "DB.java" and "App.java". The code editor displays Java code for database operations using JDBC. A red box highlights a section of the code in the "DB.java" file, specifically the deleteCustomer method. This highlighted code is as follows:

```
//String sql = "delete from Customer where cid = "+cid;
String sql = "delete from Customer where cid = ?";
PreparedStatement = connection.prepareStatement(sql);
PreparedStatement.setInt(1, cid);
//int result = statement.executeUpdate(sql);
int result = PreparedStatement.executeUpdate();

String message = result > 0 ? "Customer Deleted Successfully" : "Customer Not Deleted. Please Try Again";
System.out.println(TAG+message);
```

3.2 Call the **deleteCustomer()** method, save the code, and then run it in the **App.java** file. In the output, as you can see the customer record is deleted, and it prints **Customer Deleted Successfully.**

```
16     System.out.println( "Welcome to Customer Management System" );
17
18     Customer customer = new Customer();
19
20     customer.setId(4);
21     customer.setName("George G");
22     customer.setPhone("+91 99009 11111");
23     customer.setEmail("george.g@example.com");
24     customer.setBirthDate("1986-04-04");
25     customer.setAge(35);
26     customer.setInDateTime("2022-01-12 09:00:00");
27     customer.setOutDateTime("2022-01-12 10:30:00");
28     customer.setTemperature(98.8f);
29
30     System.out.println("Connecting to DB....");
31     DB db = new DB();
32     db.createConnection();
33
34     //db.createCustomer(customer);
35     //db.updateCustomer(customer);
36
37     db.deleteCustomer(3);
38
39     //System.out.println();
40
41     //ArrayList<Customer> customers = db.getAllCustomers();
42     //customers.forEach( cRef -> System.out.println(cRef));
43
44     db.closeConnection();
45 }
```

The screenshot shows the Eclipse IDE interface with the title bar "jdbcprojects - CMS/src/main/java/com/example/cms/App.java - Eclipse IDE". The left pane displays the Java code for App.java, which creates a Customer object and prints a welcome message. The right pane shows the "Console" tab with the following output:

```
<terminated> App [Java Application] /usr/eclipse/plugins/org.eclipse.just.openjdk.hotspot.jdk11/bin/java -jar /tmp/jetty-0_0_0_0-8080-CMS-0.0.1-SNAPSHOT.war
Welcome to Customer Management System
Connecting to DB...
[DB] Driver Loaded
[DB] Connection Created
[DB] Customer Deleted Successfully
[DB] Connection Closed. Close Status: true
```

## Step 4: Perform the getAllCustomers operation

### 4.1 Comment out the ResultSet in the DB.java file

The screenshot shows the Eclipse IDE interface with the title bar "Applications - jdbcprojects - CMS/src/main/java/com/example/cms/db/DB.java - Eclipse IDE". The left pane displays the Java code for DB.java, specifically the getAllCustomers() method. The line of code that retrieves the ResultSet from the executeQuery method is commented out with a double slash (//).

```
public ArrayList<Customer> getAllCustomers() {
    ArrayList<Customer> customers = new ArrayList<Customer>();
    try {
        String sql = "select * from Customer";
        //ResultSet set = statement.executeQuery(sql);
        while(set.next()) {
            Customer customer = new Customer();
            customer.setCid(set.getInt("cid"));
            customer.setName(set.getString(2));
            customer.setPhone(set.getString(3));
            customer.setEmail(set.getString(4));
            customer.setBirthDate(set.getString(5));
            customer.setAge(set.getInt(6));
            customer.setInDateTime(set.getString(7));
            customer.setOutDateTime(set.getString(8));
            customer.setTemperature(set.getFloat(9));
            customers.add(customer);
        }
    } catch (Exception e) {
        System.out.println("Exception Occurred: "+e);
    }
    return customers;
}
```

#### 4.2 Create the `PreparedStatement` and `ResultSet` object to call the `executeQuery()` function

```

171 public ArrayList<Customer> getAllCustomers() {
172     ArrayList<Customer> customers = new ArrayList<Customer>();
173 
174     try {
175         String sql = "select * from Customer";
176         preparedStatement = connection.prepareStatement(sql);
177         //ResultSet set = statement.executeQuery(sql);
178 
179         ResultSet set = preparedStatement.executeQuery();
180 
181         while(set.next()) {
182             Customer customer = new Customer();
183 
184             customer.setCustomerId(set.getInt("cid"));
185             customer.setName(set.getString(2));
186             customer.setPhone(set.getString(3));
187             customer.setEmail(set.getString(4));
188             customer.setBirthDate(set.getString(5));
189             customer.setAge(set.getInt(6));
190             customer.setInDate(set.getString(7));
191             customer.setOutDate(set.getString(8));
192             customer.setTemperature(set.getFloat(9));
193 
194             customers.add(customer);
195         }
196     } catch (Exception e) {
197         System.out.println("Exception Occurred: "+e);
198     }
199 
200     return customers;
201 }
202 
203 
```

#### 4.3 Go to the `App.java` file. Uncomment the `ArrayList` and add a `forEach` loop to get all customer details.

```

8 * Hello world!
9 *
10 */
11 
12 public class App
13 {
14     public static void main( String[] args )
15     {
16         System.out.println( "Welcome to Customer Management System" );
17 
18         Customer customer = new Customer();
19 
20         customer.setCustomerId(4);
21         customer.setName("George G");
22         customer.setPhone("+91 99009 11111");
23         customer.setEmail("george.g@example.com");
24         customer.setBirthDate("1986-04-04");
25         customer.setAge(35);
26         customer.setInDate("2022-01-12 09:00:00");
27         customer.setOutDate("2022-01-12 10:30:00");
28         customer.setTemperature(98.8f);
29 
30         System.out.println("Connecting to DB....");
31         DB db = new DB();
32         db.createConnection();
33 
34         //db.createCustomer(customer);
35         //db.updateCustomer(customer);
36 
37         //db.deleteCustomer(3);
38 
39         //System.out.println();
40 
41         ArrayList<Customer> customers = db.getAllCustomers();
42         customers.forEach( cRef -> System.out.println(cRef));
43 
44         db.closeConnection();
45     }
46 }
```

4.4 Run the code, and you will see all customer details as output on the console

By following these steps, you have successfully implemented **PreparedStatement** API with JDBC for writing CRUD operations for managing database records efficiently and securely.