Aufgabe 9

a)
In dem Programm wird ein JDBC-Driver verwendet.
IBM nutzt die Funktion .getConnection() aus der DriverManager Class, um eine Verbindung zu der gewünschten Datenbank herzustellen.
Außerdem nutzt das Programm die Funktion getMetaData(), um Informationen über die einzelnen Spalten in der Datenbank zu erhalten.
Für die SQL-Anweisung nutzt das Programm ein ResultSet, dessen Cursor vorwärts bewegt werden kann, um Zugriff auf die Daten aus den Zeilen der Datenbank zu erhalten.
Wenn die Verbindung zu der Datenbank nicht mehr benötigt wird, nutzt das Programm die Funktion .close(), um die Verbindung wieder zu schließen.

b)
Ein Problem der Anwendung ist, dass die Fehlermeldungen der Datenbank direkt an den Benutzer weitergegeben werden. Dadurch wird es einem potenziellen Angreifer erleichtert, herauszufinden, ob es bei dem Programm eine Schwachstelle gibt, sodass ein Angriff mittels SQL-Injection gelingen kann.
Denn wenn der Angreifer in das Eingabefeld ein ’ eingibt, kann er anhand der Fehlermeldung sehen, ob eine SQL-Injection möglich ist.
Die SQL-Injection ist auch schon die nächste Schwachstelle in diesem Programm.
Denn eine SQL Injection ist möglich, da z.B. in ResultSetTableModel.java in Zeile 229 in der Anweisung:

*rs = sqlFacade.executeQuery(“SELECT * from “ + tableName);*

kein Prepared Statement in Verbindung mit einem ? verwendet wird. Um sich gegen eine SQL-Injection abzusichern, sollte die Anweisung wie folgt aussehen:

```
PreparedStatement p = connection.preparedStatement(„“SELECT * from ?“)
p.setString(1, tableName);
rs = p.executeQuery();
```

Auch eine Überprüfung der Eingabe findet nicht statt, was die Gefahr einer SQL-Injection noch zusätzlich vergrößert.
Die nächste Schwachstelle in dem Programm betrifft das Updaten bzw. Löschen von Datensätzen. Denn laut IBM kann es hierbei in der Datenbank zu Duplikaten kommen. Bei einer Lösch- oder Update- Operation werden die entsprechenden Daten lediglich anhand mehrerer Values ermittelt, von denen jedoch keines einen eindeutigen und einzigartigen Schlüssel darstellt. Wenn also beim löschen oder updaten ein Datensatz angegeben wird, der zufällig

die gleichen Values hat, wie ein anderer, betrifft die Update- bzw. Lösch-Operation beide Datensätze, obwohl der User dies vielleicht garnicht wollte. Es wäre für dieses Operationen daher besser, Datensätze anhand eines einzigartigen Schlüssels zu identifizieren.