Datenbanken 1 Praktikum 6

Aufgabe 1 – JDBC

Es sollen aus der Datenbank alle männlichen Personen ausgelesen werden, die bisher ein oder mehrere Artikel im Warenkorb haben. Es sollen die Kundennummer, der Nachname und der Vorname auf der Konsole ausgegeben werden.

a) Wie lautet die zugehörende JDBC-Abfrage als Statement und als PreparedStatement?

SELECT-Abfrage:

SELECT DISTINCT Kundennummer, Nachname, Vorname FROM Kunde Join Warenkorb USING (IdKunde) WHERE Anzahl > 0 AND Anrede = 'Herr'

Statement:

```
String sqlString = "SELECT DISTINCT Kundennummer, Nachname, Vorname FROM Kunde Join
Warenkorb USING (Kundennummer) WHERE Anzahl > 0 AND Anrede = 'Herr'";
Statement stmt = con.createStatement();
ResultSet rs = stmt.executeQuery(sqlString);
while(rs.next())
{
        String kundennummer = Integer.toString(rs.getInt(1));
        String name = rs.getString(2);
        String vorname = rs.getString(3);
        System.out.println(kundennummer+ ": " + name + ", " + vorname);
}
```

PreparedStatement:

```
PreparedStatement ps = con.prepareStatement("SELECT DISTINCT IdKunde, Nachname, Vorname
FROM Kunde Join Warenkorb USING (kundennummer) WHERE Anzahl > 0 AND Anrede = ?");
ps.setString(1, "Herr");
ResultSet rs = ps.executeQuery();
while(rs.next())
{
        String kundennummer = Integer.toString(rs.getInt(1));
        String name = rs.getString(2);
        String vorname = rs.getString(3);
        System.out.println(kundenID + ": " + name + ", " + vorname);

}
```

b) Wie wird das ResultSet ausgewertet? Müssen NULL-Werte behandelt werden?

Das Ergebnis einer SELECT Anfrage wird zeilenweise in ein ResultSet Objekt abgelegt.
Da man hier keinen richtigen Index wie z.B. bei einem Array hat, greift man auf die einzelnen Zeilen per Cursor zu. In der API ist eine Liste der Methoden zu finden, die mit einem ResultSet angewendet werden können. In unserer Aufgabe a) benutzen wir z.B. die Methode next() um von der ersten Zeile im ResultSet zur letzten Zeile durchzugehen. Diese Methode gibt einen boolean zurück und lässt sich daher mit einer while-Schleife abfragen, die so lange läuft, bis keine Zeilen mehr vorhanden sind und die Methode next() false zurückgibt und den Schleifendurchlauf beendet. Falls der Wert einer Spalte ein SQL-NULL-Wert ist, muss man beachten, dass in bestimmten Fällen vom Treiber Defaultwerte ausgegeben werden und es so nicht immer zu erkennen ist, ob das Ergebnis stimmt.
Bei getString() wird im Falle einer SQL-NULL auch tatsächlich null zurückgegeben, aber bei

Zahlenwerten bekommt man stattdessen eine 0 und bei getBoolean() standardmäßig ein false. Wenn man wissen möchte, ob eine solche Spalte einen SQL-NULL-Wert liefert, sollte man die Methode wasNull() benutzen:

```
String s = rs.getString( column );
if ( rs.wasNull() )
System.out.println( "SQL-NULL" );
```

(Näheres zu finden in [Ullenboom: Java ist auch eine Insel] Abschnitt 23.6.5)

c) Schreiben Sie ein Java-Programm und testen Sie es aus. *Wichtig hierbei ist der Kern, d.h. JDBC und SQL Aspekte! Bitte keine Fenster und Ähnliches programmieren.*

```
import java.sql.*;

public class Datenbankanbindung {
	Connection conn;
	String url = "jdbc:oracle:thin:@172.22.112.100:1521:fbpool";

	// Anbindung an die Datenbank kann im Konstruktor erfolgen
	public Datenbankanbindung() {
		try {
			Class.forName("oracle.jdbc.driver.OracleDriver");
			// ggf. url, login und password anpassen
			conn = DriverManager.getConnection(url,"C##FBPOOLNN", "oracle");
		} catch (SQLException e) {
			System.out.println("*** SQLException:\n" + e);
			e.printStackTrace();
		} catch (Exception e) {
			System.out.println("*** Exception:\n" + e);
			e.printStackTrace();
		}
	}
	public static void main(String[] args) throws SQLException {
		Datenbankanbindung d = new Datenbankanbindung();
		// Ausgabe
		System.out.println("Ausgabe einfachesStatement()");
		d.einfachesStatement();
		System.out.println("-----------------------------------------------");
		System.out.println("Ausgabe preparedStatement()");
		d.preparedStatement();
	}

	// Methode, um ein einfaches Statement auszuführen
	public void einfachesStatement() {
		// Unsere Datenbankanfrage als String
		String sqlString = "SELECT DISTINCT kundennummer, Nachname, Vorname FROM Kunde
Join Warenkorb USING (kundennummer) WHERE Anzahl > 0 AND Anrede = 'Herr'";
		// Exception abfangen, die von createStatement() erzeugt werden könnte
		try {
			// einfaches Statement erzeugen
			Statement stmt = conn.createStatement();
			// ResultSet erzeugen und Statement (unsere Anfrage) ausführen
			ResultSet rs = stmt.executeQuery(sqlString);
```

```
                    // ResultSet auslesen und ausgeben (Die Zahlen repräsentieren die
                    // Spalten im resultSet)
                    while (rs.next()) {
                        String kundennummer = Integer.toString(rs.getInt(1));
                        String name = rs.getString(2);
                        String vorname = rs.getString(3);
                        System.out.println(kundennummer + " : " + name + " " + vorname);
                    }
                } catch (SQLException se) {
                    System.out.println("Fehler in erster Methode");
                }
        }

        // Methode, um ein preparedStatement auszuführen
        public void preparedStatement() {
            // Exception abfangen, die von prepareStatement() erzeugt werden könnte
            try {
                // PreparedStatement erzeugen
                PreparedStatement ps = conn.prepareStatement("SELECT DISTINCT
kundennummer, Nachname, Vorname FROM Kunde Join Warenkorb USING (kundennummer) WHERE
Anzahl > 0 AND Anrede = ?");
                // Parameter werden übergeben
                ps.setString(1, "Herr");
                // ResultSet erzeugen und Statement (unsere Anfrage) ausführen
                ResultSet rs = ps.executeQuery();
                // ResultSet auslesen und ausgeben
                while (rs.next()) {
                    String kundennummer = Integer.toString(rs.getInt(1));
                    String name = rs.getString(2);
                    String vorname = rs.getString(3);
                    System.out.println(kundenId + " : " + name + " " + vorname);
                }
            } catch (SQLException se) {
                System.out.println("Fehler in zweiter Methode");
            }
        }
}
```