Übungsblatt 1 zu Kapitel 5

1) Löschen Sie die Relationen *Auftrag*, *Kunde* und *Personal*. Warum wird der Versuch des Löschens jeder dieser drei Relationen mit einer Fehlermeldung quittiert? Wie können wir dennoch diese Relationen löschen? Erzeugen Sie dann diese Relationen neu mit allen Fremdschlüsselbeziehungen!

   Hinweis: Zur Kontrolle betrachten Sie die SQL-Prozedur *bikeOracle.sql* oder *bikeMySQL.sql* auf dem Server. Lesen Sie diese Dateien mit einem Texteditor oder SQL Developer. Sie werden nur bekannte Befehle vorfinden! Hat das Verweigern des Löschens etwas mit Fremdschlüsseln zu tun?

   Beachten Sie, dass eine mit dem Befehl *Drop Table* gelöschte Tabelle nicht mit Rollback zurückgesetzt werden kann! Beachten Sie ferner, dass die Fremdschlüsselbedingung (nicht der Inhalt!) in der Relation *Auftragsposten* mit gelöscht wird. Auch diese Bedingung muss wieder hergestellt werden!

2) Erzeugen Sie eine Relation *Rechnung*. Diese Relation besitzt mindestens die Attribute *Rechnr*, *Datum*, *Kundnr*, *Gesamtbetrag*, *Bezahltdatum*, *Mahnung*. Setzen Sie hierzu geeignete Datentypen an. Geben Sie Primär- und Fremdschlüssel an. Achten Sie darauf, dass das Attribut *Mahnung* nur die Werte 'N' und 'J' annehmen darf. Übernehmen Sie mit Hilfe eines geeigneten *Insert*-Befehls alle relevanten Daten aus den Relationen *Auftrag* und *Auftragsposten*.

   Hinweis: Schreiben Sie die gesamte Aufgabe in eine einzige Datei namens *rechnung.sql*. Geizen Sie nicht mit Kommentaren und Ausgaben. Sie benötigen einen Create-Table-, einen Insert- und einen Commit-Befehl. Es wird empfohlen, die Auftragsnummer als Rechnungsnummer eins zu eins zu übernehmen. Wie wäre es noch mit einem Skript, das die Relation wieder entfernt?

3) Fügen Sie in der Relation *Auftragsposten* das Attribut *Einzelpreis* hinzu. Tragen Sie mit einem Update-Befehl alle Preise ein, indem diese aus existierenden Einträgen berechnet werden. In welcher Normalform befand sich diese Relation vor dem Einfügen des neuen Attributs, in welcher befindet sie sich jetzt nach dem Einfügen des Einzelpreises?

   Hinweis: Zusatzinformationen führen häufig zu Redundanz, so dass auch die Normalform kleiner wird. Ist das nun ein schlechtes Design oder ist dies akzeptabel?

4) Wir wollen die Integrität verbessern und die Attribute *GebDatum*, *Stand*, *Gehalt* und *Beurteilung* der Relation *Personal* auf zulässige Werte überprüfen lassen. Es ist sicher zu stellen, dass alle Mitarbeiter zwischen 1940 und 1998 geboren sind, entweder ledig (*led*), verheiratet (*verh*), geschieden (*ges*) oder verwitwet (*verw*) sind, das Gehalt zwischen 500 und 6000 Euro liegt und die Beurteilung entweder *Null* oder einen Wert zwischen 1 und 10 besitzt. Fügen Sie diese Bedingungen mittels eines *Alter-Table*-Befehls hinzu, wobei sicherzustellen ist, dass diese Bedingungen, falls gewünscht, auch wieder entfernt werden können.

   Hinweis: Sie können alle Bedingungen in einer Check-Bedingung zusammenfassen (ein einziger Alter-Table-Befehl). Sie können für jede Bedingung aber auch eine eigene Check-Bedingung verwenden (mehrere Alter-Table-Befehle). Verwenden Sie Constraint-Namen!

5) Die Relation *Auftragsposten* enthält aus Redundanzgründen nur den Gesamtpreis jedes einzelnen Auftragspostens. Schreiben Sie daher eine Sicht *VAuftragsposten*, die alle Daten der Relation *Auftragsposten* enthält und zusätzlich ein Attribut *Einzelpreis*.

   Hinweis: Warum ist diese Sicht nicht änderbar?

6) Die Relation *Teilestruktur* gibt einen guten Überblick über den Aufbau von komplexen Teilen. Allerdings stehen hier nur die Teilenummern. Schreiben Sie eine Sicht *VStruktur*, die der Relation *Teilestruktur* entspricht, wobei allerdings alle Teilenummern durch die entsprechenden Teilebezeichnungen zu ersetzen sind.

   Hinweis: Diese Aufgabe baut auf Aufgabe 14 aus Kapitel 4 auf.