Hinweis

Ich habe auf meinem eigenem Rechner MySQL 3.23.44 unter Windows 2000 installiert, um die nachfolgenden SQL-Befehle überprüfen zu können. Es ist möglich, dass in ANSI-SQL-92 unter Umständen effizientere Befehle als die verwendeten existieren. Diese Aussage gilt für alle bearbeiteten Aufgaben dieser (und der nächsten) Übungsblätter.

Aufgabe 5

Im folgenden gebe ich zwei SQL-Strings an: zuerst folgt der minimale, in der Theorie ausreichende String, danach der in MySQL tatsächlich verwendete. Bei letzterem ist zu beachten, dass ich den Tabellennamen jeweils „ueb2_“ voranstellte, um die Datenbank in späteren Übungen auch noch nutzen zu können.

a) *Fügen Sie der Tabelle* `r` *ein ganzzahliges Attribut* `b` *als zweites Attribut hinzu.*

```
ALTER TABLE r ADD b INT;
ALTER TABLE `dbs_uebung`.`ueb2_r` ADD `b` INT;
```

b) *Entfernen Sie das Attribut* `dummy` *aus der Tabelle* `s`.

```
ALTER TABLE s DROP dummy;
ALTER TABLE `ueb2_s` DROP `dummy‘;
```

c) *Sei das Attribut* `alt` *der Tabelle* `t` *vom Typ* `date`. *Benennen Sie das Attribut in* `neu` *um.*

```
ALTER TABLE t CHANGE alt neu DATE
ALTER TABLE `dbs_uebung`.`ueb2_t` CHANGE `alt` `neu` DATE
```

d) *Verändern Sie den Typen des Attributes* `zahl` *von* `int` *zu* `CHAR(20)`.

```
ALTER TABLE t CHANGE zahl zahl CHAR(20)
ALTER TABLE `dbs_uebung`.`ueb2_t` CHANGE `zahl` `zahl` CHAR(20)
```

e) *Wie muss die folgende Sicht angepasst werden, um das externe Schema beizubehalten ?*
Es sind drei Änderungen in der Datenbank vorgenommen worden, die für die Sicht Relevanz haben:
1. `s.dummy` wurde entfernt
2. `t.alt` wurde in `t.neu` umbenannt
3. `t.zahl` ist jetzt vom Datentyp `CHAR(20)` statt `INT`

Um dennoch die Sicht unverändert lassen zu können, wird 2. durch ein `AS` simuliert. Die letzte Bedingung des alten `WHERE` entfällt (1.). Nicht unproblematisch ist der geänderte Datentyp von `zahl` (3.), da er jetzt statt einer Zahl eine Zeichenkette ist und u.U. keine Berechnungen (`SUM` etc.) mehr darauf ausgeführt werden können.

```
CREATE VIEW test1 (a, c, e, f, alt, zahl) AS
         SELECT r.a, r.c, s.e, t.f, t.neu AS alt, t.zahl FROM r, s, t
         WHERE r.a = s.a AND s.f = t.f AND
         (r.c < (1.25*s.e) OR t.neu > „1973-04-27“);
```

Ich möchte noch anmerken, dass ursprünglich die `WHERE`-Bedingung mit `r1.a=s1.a` anfing, ich vermute, dass die Einsen Tippfehler waren und habe sie daher entfernt.

Aufgabe 6

a) $\pi_{a,b,c}\left(join\left(\sigma_{a='001'}(t), join(r,s)\right)\right)$

```
SELECT a,b,c FROM r NATURAL JOIN s NATURAL JOIN t
            WHERE s.a = „001“;
```

b) $\pi_{name,ort}\left(join_{Auftrag.KNr=Kunde.Nr}\left(\sigma_{Nr>200}(Auftrag) \cup \sigma_{Nr<100}(Auftrag)\right), Kunde\right)$

```
SELECT name,ort FROM Auftrag,Kunde WHERE
          Auftrag.KNr = Kunde.Nr AND
         (Auftrag.Nr<100 OR Auftrag.Nr>200);
```

Aufgabe 7

Die angegebenen SQL-Strings sind so formuliert, wie sie auch in MySQL funktionieren und nutzen daher nicht immer den vollen ANSI-SQL-92-Standard aus. Grau hinterlegte Zeilen dienen dazu, dass in der Beispieldatenbank Tabellen erzeugt werden, die die Ergebnismenge beinhalten.

a) *Zeige alle Aufträge (Nummer, Bezeichnung, Status), die Artikel enthalten, bei denen die Gewinnspanne über 20% des EK-Preises liegt. Anmerkung: Gewinnspanne = VK-Preis – EK-Preis*

```
INSERT INTO aufgabe7a
SELECT DISTINCT auftrag.* FROM auftrag NATURAL JOIN auftragspositionen
                                       NATURAL JOIN artikel
       WHERE vk_preis > 1.2*ek_preis;
```

AuftragNr	Bezeichnung	Status
100	Reparatur	ok
110	Wartung	ok
120	Einbau	i.A.
130	Wartung	Fehler
140	Reparatur	erl.

b) Zeige den Lagerbestand (Artikelnummer, Bezeichnung, Anzahl) von den Artikeln mit der geringsten Anzahl pro Lager.

```
INSERT INTO aufgabe7b
SELECT lagerort.artikelnr, bezeichnung, MIN(anzahl)
         FROM artikel, lagerort
         GROUP BY Lager;
```

ArtikelNr	Bezeichnung	Anzahl
890	Anschlusskasten	2
890	Anschlusskasten	3

c) *Zeige den Lagerbestand (Artikelnummer, Bezeichnung, Anzahl) von Artikeln mit weniger als 20 Stück insgesamt vorrätig an.*

```
INSERT INTO aufgabe7c
SELECT lagerort.artikelnr, bezeichnung, SUM(anzahl)
         FROM lagerort NATURAL JOIN artikel
         GROUP BY lagerort.artikelnr
         HAVING SUM(anzahl)<20;
```

ArtikelNr	Bezeichnung	Anzahl
500	Komplett Standard	9
501	Komplett Deluxe	16
890	Anschlusskasten	5

d) *Zeige die (Einkaufs-) Kosten und die (Verkaufs-) Erlöse sowie die Gewinnspanne aller Aufträge an. Dabei soll pro Auftrag (Nummer, Bezeichnung) nur eine Zeile in der Ergebnis-Tabelle erscheinen.*

```
INSERT INTO aufgabe7d
SELECT auftrag.auftragnr, auftrag.bezeichnung,
       SUM(auftragspositionen.anzahl*ek_preis) AS einkaufskosten,
       SUM(auftragspositionen.anzahl*vk_preis) AS verkaufserloese,
      SUM(auftragspositionen.anzahl*(vk_preis-ek_preis)) AS gewinnspanne
       FROM auftrag NATURAL JOIN auftragspositionen
                    NATURAL JOIN artikel
       GROUP BY auftrag.auftragnr;
```

AuftragNr	Bezeichnung	Einkaufskosten	Verkaufserloese	Gewinnspanne
100	Reparatur	274,80	339,40	64,60
110	Wartung	161,10	203,00	41,90
120	Einbau	350,00	499,90	149,90
130	Wartung	350,00	499,90	149,90
140	Reparatur	460,00	699,90	239,90
150	Neubau	265,00	299,90	34,90

e) *Lösche alle Kunden, die über keinen Auftrag verfügen, und gebe anschließend die Tabelle* `Kunden` *aus.*

```
INSERT INTO aufgabe7e
DELETE FROM kunde WHERE kunde.kundennr NOT IN
            (SELECT kundennr FROM akrel);
```

KundenNr	Name	Ort	Strasse
200	Müller	Berlin	Torstr 140
201	Meier	Potsdam	Zeppelinstr. 87b
202	Schmidt	Potsdam	Am Luftschiffhafen 1
204	Müller	Werder	Dorfweg 17
303	Toll GmbH	Berlin	Prenzlauer Allee 147