Kaufhausdatenbank

Die relationale Datenbank eines Kaufhauses enthält folgende Tabellen:

Artikel

ArtNr	Bezeichnung	Verkaufspreis	Einkaufspreis
95	Kamm	1.25	0.80
97	Kamm	0.99	0.75
507	Seife	3.93	2.45
1056	Zwieback	1.20	0.90
1401	Räucherlachs	4.90	3.60
2045	Herrenhose	37.25	24.45
2046	Herrenhose	20.00	17.00
2340	Sommerkleid	94.60	71.50

Abteilung

Abteilungsname	Stockwerk	Abteilungsleiter
Lebensmittel	I	Josef Kunz
Lebensmittel	EG	Monika Stiehl
Textilien	II	Monika Stiehl

Bestand

Abteilungsname	ArtNr	Vorrat
Lebensmittel	1056	129
Lebensmittel	1401	200
Textilien	2045	14

```sql
CREATE TABLE Artikel (
  ArtNr INTEGER PRIMARY KEY NOT NULL,
  Bezeichnung VARCHAR(100) NOT NULL,
  Verkaufspreis FLOAT(2),
  Einkaufspreis FLOAT(2)
);

CREATE TABLE Abteilung (
  Abteilungsname VARCHAR(60) NOT NULL,
  Stockwerk VARCHAR(10) NOT NULL,
  Abteilungsleiter VARCHAR(100),
  PRIMARY KEY (Abteilungsname, Stockwerk)
);

CREATE TABLE Bestand (
  Abteilungsname VARCHAR(100) REFERENCES Abteilung(Abteilungsname),
  ArtNr INTEGER REFERENCES Artikel(ArtNr),
  Vorrat INTEGER,
  PRIMARY KEY (Abteilungsname, ArtNr)
);

INSERT INTO Artikel VALUES
  (95, 'Kamm', 1.25, 0.80),
  (97, 'Kamm', 0.99, 0.75),
  (507, 'Seife', 3.93, 2.45),
  (1056, 'Zwieback', 1.20, 0.90),
  (1401, 'Räucherlachs', 4.90, 3.60),
  (2045, 'Herrenhose', 37.25, 24.45),
  (2046, 'Herrenhose', 20.00, 17.00),
```

```
30    (2340, 'Sommerkleid', 94.60, 71.50);
31
32  INSERT INTO Abteilung VALUES
33    ('Lebensmittel', 'I', 'Josef Kunz'),
34    ('Lebensmittel', 'EG', 'Monika Stiehl'),
35    ('Textilien', 'II', 'Monika Stiehl');
36
37  INSERT INTO Bestand VALUES
38    ('Lebensmittel', 1056, 129)
39    ('Lebensmittel', 1401, 200)
40    ('Textilien', 2045, 14);
```

Formulieren Sie mit Hilfe von SQL folgende Anfragen:

(a) Gesucht sind alle Informationen über Herrenhose und Sommerkleid!

```sql
SELECT *
FROM Artikel
WHERE
  Bezeichnung = 'Herrenhose' OR
  Bezeichnung = 'Sommerkleid';
```

(b) Welche Artikelnummer hat der Zwieback?

```sql
SELECT ArtNr
FROM Artikel
WHERE
  Bezeichnung = 'Zwieback';
```

(c) Welche Waren (Artikelnummer und Verkaufspreis) werden für mehr als 25€ verkauft?

```sql
SELECT ArtNr, Verkaufspreis
FROM Artikel
WHERE Verkaufspreis > 25.00;
```

(d) Welche Artikel (Angabe der Bezeichnung) bietet das Kaufhaus an?

```sql
SELECT DISTINCT Bezeichnung
FROM Artikel;
```

(e) Gesucht sind die Artikelnummern aller Artikel mit Ausnahme der Artikelnummer 2046.

```sql
SELECT ArtNr
FROM Artikel
WHERE NOT (ArtNr = 2046);
```

(f) Gib die Artikelnummern und die Verkaufspreise aller Herrenhosen aus, die für höchstens 25€ verkauft werden! Der Spaltenname für die Verkaufspreise soll in der Ergebnistabelle „Sonderangebot“ heißen.

```sql
SELECT ArtNr, Verkaufspreis AS Sonderangebot
FROM Artikel
WHERE Bezeichnung = 'Herrenhose' AND Verkaufspreis <= 25;
```

(g) Gib Artikelnummer und Verkaufspreis aller Artikel aus, die im Einkauf zwischen 80 Cent und 5€ kosten.

```sql
SELECT ArtNr, Verkaufspreis
FROM Artikel
WHERE Einkaufspreis BETWEEN 0.80 AND 5.00;
```

Teilaufgabe 2

(a) Geben Sie die SQL-Befehle an, mit der die Tabellenschemata von Artikel und Bestand erzeugt werden können. Wählen Sie dabei geeignete Domänen.

```sql
CREATE TABLE Artikel (
  ArtNr INTEGER PRIMARY KEY NOT NULL,
  Bezeichnung VARCHAR(100) NOT NULL,
  Verkaufspreis FLOAT(2),
  Einkaufspreis FLOAT(2)
);

CREATE TABLE Bestand (
  Abteilungsname VARCHAR(100) REFERENCES Abteilung(Abteilungsname),
  ArtNr INTEGER REFERENCES Artikel(ArtNr),
  Vorrat INTEGER,
  PRIMARY KEY (Abteilungsname, ArtNr)
);
```

(b) Es treten nun nacheinander die folgenden Änderungen auf. Aktualisieren Sie den Tabellenbestand mit den entsprechenden SQL-Befehlen:

(i) Ein Sommerkleid mit der Artikelnummer 2341, dem Einkaufspreis 70€ und dem Verkaufspreis 90,75€ wird in das Artikelsortiment aufgenommen.

```sql
INSERT INTO Artikel
VALUES (2341, 'Sommerkleid', 90.75, 70.00);
```

(ii) Der Artikel mit der Nummer 2341 wird wieder aus dem Sortiment genommen, da er den Qualitätsstandards nicht entsprochen hat.

```sql
DELETE FROM Artikel WHERE ArtNr = 2341;
```

(iii) Eine Bürste mit der Artikelnummer 2 wird in das Sortiment aufgenommen. Einkaufs- bzw. Verkaufspreis sind noch nicht festgelegt.

```sql
INSERT INTO Artikel (ArtNr, Bezeichnung)
VALUES (2, 'Bürste');
```

(iv) Eine Damenhose (Verkaufspreis 89€, Einkaufspreis: 60,50€) wird neu in das Sortiment aufgenommen. Eine Artikelnummer wurde noch nicht festgelegt.

`ArtNr` ist der Primärschlüssel der Tabelle `Artikel`. Bei Eingabe eines neuen Datensatzes müssen mindestens die Werte aller Attribute, die zum Primärschlüssel gehören, angegeben werden. Da aber im Fall der Damenhose die Artikelnummer noch nicht festgelegt ist, ist eine Eingabe der Damenhose-Daten in die Tabelle Artikel nicht möglich. Hinweis: Denken Sie also immer daran, dass bei Einfügen von Datensätzen der Primärschlüssel keine NULL-Werte enthalten darf!

(v) Die Herrenhosen werden aus dem Sortiment genommen und deshalb aus der Tabelle Artikel gelöscht.

```
DELETE FROM Bestand WHERE ArtNr = 2045;
DELETE FROM Artikel WHERE Bezeichnung = 'Herrenhose';
```

(vi) Die neue Abteilungsleiterin der Lebensmittelabteilung heißt Elvira Sommer.

```
UPDATE Abteilung
SET Abteilungsleiter = 'Elvira Sommer'
WHERE Abteilungsnahme = 'Lebensmittel';
```

(vii) Die Abteilung Feinkost hat einen Bestand von 150 Räucherlachspackungen.

Die Attribute ArtNr und Abteilungsname der Tabelle Bestand sind Fremdschlüssel. Ein neuer Datensatz darf in die Tabelle nur eingefügt werden, wenn die Fremdschlüsselwerte in den entsprechenden (Primärschlüssel-)Attribute der referenzierten Tabelle auch existieren. Die Abteilung Feinkost, genauer gesagt den Abteilungsnamen 'Feinkost' gibt es in Abteilung aber noch nicht.

i. Lösungsmöglichkeit: Die Aktualisierung kann nicht durchgeführt werden.
ii. Lösungsmöglichkeit: Die entsprechende Abteilung Feinkost wird – natürlich in „Absprache“ mit der Kaufhausleitung – eingeführt und ein dementsprechender Datensatz in Abteilung eingefügt.

```
INSERT INTO Abteilung (Abteilungsname) VALUES ('Feinkost');
INSERT INTO Bestand VALUES ('Feinkost', 1401, 150);
```

(c) Formulieren Sie folgende Anfragen in SQL:

(i) Gesucht sind Artikelnummer und Vorrat aller Artikel aus der Textil-Abteilung.

```
SELECT ArtNr, Vorrat FROM Bestand WHERE Abteilungsname =
↪ 'Textilien';
```

(ii) Gesucht sind alle Informationen über die Abteilungen, die im zweiten Stock platziert sind oder von Frau Stiehl geleitet werden.

```sql
SELECT * FROM Abteilung
WHERE Stockwerk = 'II' OR Abteilungsleiter = 'Monika Stiehl';
```

(d) Formulieren Sie folgende SQL-Anfragen umgangssprachlich:

(i) SQL-Anfrage:

```sql
SELECT DISTINCT Abteilungsleiter
FROM Abteilung
WHERE NOT (Abteilungsname = 'Kosmetik');
```

> Gesucht sind die Namen aller Abteilungsleiter mit Ausnahme der Kosmetik-Abteilung. Duplikate sollen eliminiert werden.

(ii) SQL-Anfrage:

```sql
SELECT ArtNr
FROM Bestand
WHERE Abteilungsname = "Lebensmittel" AND Vorrat <= 100;
```

> Gesucht sind die Nummern der Artikel, von denen in der Lebensmittelabteilung maximal 100 vorrätig sind.

(e) Interpretieren Sie nun die obigen Tabellen als Repräsentationen der drei Relationen Artikel, Abteilung und Bestand. Bestimmen Sie die Ergebnisrelationen folgender relationaler Ausdrücke:

(i) $\pi_{ArtNr,Bezeichnung}(Artikel)$

95	Kamm
97	Kamm
507	Seife
1056	Zwieback
1401	Räucherlachs
2045	Herrenhose
2046	Herrenhose
2340	Sommerkleid

(ii) $\pi_{Abteilungsname}(Bestand)$

Lebensmittel
Textilien

(iii) $\sigma_{((Vorrat<100 \wedge ArtNr>1500) \vee ArtNr<1100)}(Bestand)$

Lebensmittel	1056	129
Textilien	2045	14

(iv) $\sigma_{((Vorrat<100 \wedge (ArtNr>1500 \vee ArtNr<1100)}(Bestand)$

Textilien 2045 14

(v) $\pi_{ArtNr}(\sigma_{Bezeichnung=Herrenhose}(Artikel))$

2045
2046

(vi) $\pi_{Abteilungsname}(Abteilung) - \pi_{Abteilungsname}(Bestand)$

Kosmetik

(vii) $\pi_{Bezeichnung,Einkaufspreis}(\sigma_{Einkaufspreis<2.50}(Artikel)) \cup \pi_{Bezeichnung,Einkaufspreis}(\sigma_{Einkaufsreis>20.00}(Artikel))$

Die letzten Zeile ist nicht in der Musterlösung dabei. Ich glaube aber es müsste so stimmen.

Bezeichnung	Einkaufspreis
Kamm	0.80
Kamm	0.75
Seife	2.45
Zwieback	0.90
Herrenhose	24.45
Sommerkleid	71.50

Teilaufgabe 2

(a) Formulieren Sie nachfolgende Anfragen in SQL mit Hilfe von Joins!

- Wie viele Packungen Zwieback sind noch vorrätig?

Hinweis: In obigem Lösungsansatz wird berücksichtigt, dass ein Artikel, hier der Zwieback, in mehreren Abteilungen verkauft werden kann. Geht man davon aus, dass Zwieback nur in einer Abteilung verkauft wird, kann man die Aggregatfunktion SUM weglassen.

```
SELECT SUM(b.Vorrat)
FROM Bestand b, Artikel a
WHERE b.ArtNr = a.ArtNr AND a.Bezeichnung = 'Zwieback';
```

- In welchem Stockwerk wird Räucherlachs verkauft?

```
SELECT Abteilung.Stockwerk
FROM Artikel, Abteilung, Bestand
WHERE Artikel.ArtNr = Bestand.ArtNr AND
Bestand.Abteilungsname = Abteilung.Abteilungsname AND
Artikel.Bezeichnung = 'Räucherlachs';
```

(b) Formulieren Sie folgende Anfragen an die Kaufhaus-Datenbank unter Verwendung von geschachtelten SELECT-Anweisungen!

- Gib die Bezeichnungen und die Artikelnummern aller Artikel aus, die nicht mehr als der Artikel mit der Artikelnummer 1401 kosten!

Hinweis: Durch Hinzufügen der Bedingung NOT(ArtNr=1401) wird der Artikel mit der Nummer 1401 in der Ergebnistabelle nicht aufgeführt

```
SELECT Bezeichnung, ArtNr AS Artikelnummer
FROM Artikel
WHERE Verkaufspreis <= (
  SELECT Verkaufspreis FROM Artikel WHERE ArtNr = 1401
);
```

- Gesucht sind Bezeichnung und Verkaufspreis aller Artikel, die in der Textilienabteilung verkauft werden!

```
SELECT Bezeichnung, Verkaufspreis
FROM Artikel WHERE ArtNr in (
  SELECT ArtNr FROM Bestand WHERE Abteilungsname = 'Textilien'
);
```

- Welche Produkte (Angabe der Bezeichnung) werden im Erdgeschoss verkauft?

```
SELECT DISTINCT Bezeichnung
FROM Artikel
WHERE ArtNr in (
  SELECT ArtNr
  FROM Bestand
  WHERE Abteilungsname IN (
    SELECT Abteilungsname
    FROM Abteilung
    WHERE Stockwerk = 'EG'
  )
);
```

- Gib die Namen aller Abteilungsleiter aus, in deren Abteilungen von jedem Artikel weniger als 100 Exemplare vorrätig sind!

```
SELECT DISTINCT Abteilungsleiter
FROM Abteilung
WHERE NOT EXISTS (
  SELECT *
  FROM Bestand
  WHERE (Abteilung.Abteilungsname =
  Bestand.Abteilungsname) AND Vorrat >= 100
);
```

(c) Lösen Sie die Aufgabe 1b) Punkt 1 ohne Verwendung einer geschachtelten SQL Anfrage! (Gib die Bezeichnungen und die Artikelnummern aller Artikel aus, die nicht mehr als der Artikel mit der Artikelnummer 1401 kosten!)

```
SELECT a.Bezeichnung, a.ArtNr as Artikelnummer
FROM Artikel a, Artikel b
WHERE
  a.Verkaufspreis <= b.Verkaufspreis AND
  b.ArtNr = 1401;
```

(d) Formulieren Sie nachfolgende Anfragen mit Mengenoperatoren!

- Gibt es registrierte Artikel, die noch nicht im Bestand aufgeführt sind?

```sql
SELECT ArtNr FROM Artikel
EXCEPT
SELECT ArtNr FROM Bestand;
```

- Welche Artikel (Artikelnummer) sind registriert und bereits im Bestand aufgeführt?

```sql
SELECT ArtNr FROM Artikel
INTERSECT
SELECT ArtNr FROM Bestand;
```

- Welche Artikel (Bezeichnung und Artikelnummer) sind bereits registriert und im Bestand aufgeführt?

```sql
SELECT Bezeichnung, ArtNr FROM Artikel WHERE ArtNr IN (
  SELECT ArtNr FROM Artikel
  INTERSECT
  SELECT ArtNr FROM Bestand
);
```

(e) Formulieren Sie folgende Anfragen in SQL:

- Welche Artikel mit dem Anfangsbuchstaben “S” gibt es?

```sql
SELECT Bezeichnung FROM Artikel WHERE Bezeichnung LIKE 'S%';
```

- Welche Artikel haben an der 3. Stelle ein “i”?

```sql
SELECT Bezeichnung FROM Artikel WHERE Bezeichnung LIKE '__i%';
```

- Heißt der Artikel “Zwieback” oder “Zweiback”?

```sql
SELECT Bezeichnung FROM Artikel WHERE Bezeichnung LIKE
↪ 'Zw__back';
```

Teilaufgabe 4

(a) Welche Artikel (Artikelnummer, Abteilungsname) werden in den Abteilungen angeboten? Die Ausgabe soll absteigend nach der Artikelnummer sortiert werden. Bei gleicher Artikelnummer sollen die betroffenen Abteilungen alphabetisch aufgelistet werden.

```sql
SELECT ArtNr, Abteilungsname
FROM Bestand
ORDER BY ArtNr DESC, Abteilungsname;
```

(b) Wie viele verschiedene Waren werden in der Lebensmittelabteilung verkauft?

```sql
SELECT COUNT(*)
FROM Bestand
WHERE Abteilungsname = 'Lebensmittel';
```

(c) Wie viele verschiedene Waren werden in den einzelnen Abteilungen verkauft?

```sql
SELECT Abteilungsname, COUNT(*)
FROM Bestand
GROUP BY Abteilungsname;
```

(d) Wie viel kostet der billigste, wie viel der teuerste Artikel?

```sql
SELECT MIN(Verkaufspreis), MAX(Verkaufspreis)
FROM Artikel;
```

(e) Gib die Namen aller Abteilungen aus, deren Gesamtvorrat an Artikel kleiner als 100 ist!

```sql
SELECT Abteilungsname
FROM Bestand
GROUP BY Abteilungsname
HAVING COUNT(Vorrat) < 100;
```

(f) Gesucht sind Bezeichnung und Verkaufspreis aller in der Datenbank gespeicherten Artikel. Die Ausgabe soll alphabetisch aufgelistet werden. Bei gleicher Bezeichnung sollen die teureren Artikel zuerst aufgelistet werden.

```sql
SELECT Bezeichnung, Verkaufspreis
FROM Artikel
ORDER BY Bezeichnung, Verkaufspreis DESC;
```

(g) Gib für alle Artikel, von denen (unabhängig von der Abteilung) noch mindestens 130 Exemplare vorrätig sind, die Artikelnummer und den aktuellen Vorrat aus!

```sql
SELECT ArtNr, SUM(Vorrat)
FROM Bestand
GROUP BY ArtNr
HAVING SUM(Vorrat) >= 130;
```

Teilaufgabe 5

(a) Sicht `view1`: Gesucht sind alle Informationen zu Artikeln, an denen das Kaufhaus mehr als 35% verdient.

```sql
CREATE VIEW view1 AS
SELECT *
FROM Artikel
WHERE Verkaufspreis > 1.35 * Einkaufspreis;
```

(b) Sicht `view2`: Gesucht sind alle Informationen zu Artikeln, an denen das Kaufhaus mehr als 35% verdient und die für höchstens 50 € verkauft werden.

```sql
CREATE VIEW view2 AS
SELECT *
FROM view1
WHERE Verkaufspreis <= 50;
```