

**Aufgabe 1: Die Drei-Ebenen-Schemaaarchitektur**

Skizzieren Sie die Drei-Ebenen-Schemaaarchitektur nach ANSI graphisch dar und erklären Sie die einzelnen Schichten.

- **Internes Schema**  
Lagt fest, in welcher Form die Daten der DB gespeichert werden (Speicherungsstruktur).
- **Konzeptuelles Schema**  
Auf Basis des Datenbanksmodells entwickelte logische Beschreibung des Realweltanschnitts.  
Abstraktion von der internen Darstellung, Anwendungszentrale Darstellung.
- **Externes Schema (Sichten)**  
Entspricht aus dem konzeptuellen Schema einem anwendungsspezifischen Ausschnitt. Ebene, auf der normalerweise Benutzer und Anwendungen arbeiten. Nichtrelevante Daten werden ggf. ausgeblendet bzw. anwendungsspezifische Darstellung möglich. Dient außerdem dem Datenschutz.

Es gibt jeweils ein internes und konzeptuelles Schema. Es kann aber mehrere Sichten geben.

Zudem wird auf die Abbildung von Folie 12 (Skript) bzw. die Abbildung 3.1 im Buch auf S. 152 verwiesen, die hier aus Platzgründen nicht angegeben ist.

**Aufgabe 2: Abarbeitung von SQL-Anfragen**

Siehe Angaben von Folie 422+423 bzw. im Buch Abbildung 7.7 auf S. 393.

**FROM:** Aufbau universelle Ausgangstabelle; diese Tabelle dient als Ausgangspunkt der Anfrage

**WHERE:** Selektion gewünschter Zeilen

**GROUP BY:** Teilt Eingabetabelle in Menge von Untertabellen gleicher Struktur auf. zu denselben Untertabellen gehören alle Zeilen, die in der in GROUP BY-Klausel genannten Spalten im Wert übereinstimmen

**HAVING:** Wählt aus den in GROUP BY-Klausel gebildeten Untertabellen diejenigen aus, die der in HAVING-Klausel angegebenen Bedingung genügen

**SELECT:** Bringt Ergebnismenge in die von Anwendung gewünschte Form (Projektion)

**ORDER BY:** Sortiert Eingabetabelle auf Basis der in dieser Klausel angegebenen Spaltennamen

**Aufgabe 3: Relationale Algebra**

Berechnen Sie die Ergebnismengen folgender Ausdrücke (überlassen Sie dabei nicht die unterschiedliche Anordnung der Attribute in den Relationen s und t):

a)  $\pi_2(t)$

b)  $r \bowtie s$

c)  $\sigma_{t_2 < t_1}(t) \bowtie s$

d)  $s - t$

a)

| K |
|---|
| 3 |
| 1 |
| 7 |

b)

| K | M | O | L |
|---|---|---|---|
| 1 | 7 | 7 | 3 |
| 7 | 4 | 7 | 6 |
| 7 | 4 | 7 | 3 |

c)

| K | M | O |
|---|---|---|
| 7 | 4 | 7 |

d)

| K | M | L |
|---|---|---|
| 1 | 4 | 3 |
| 7 | 4 | 6 |

2) Gegeben seien folgende Relationen:

Patient (Pat.Nr., Vorname, Nachname, Geburtsdatum, Diagnose)

Medikament (Med.Nr., Bezeichnung, Preis)

Verschreibung (Pat.Nr., Med.Nr., Menge, Datum)

a)  $\pi_{\text{Pat.Nr., Vorname, Nachname}}(\pi_{\text{Diagnose}} \text{ "Herzinfarkt" } \text{Patient})$

b)  $\pi_{\text{Med.Nr., Bezeichnung}}(\sigma_{\text{Diagnose} = \text{"Herzinfarkt"}}(\text{Patient} \bowtie \text{Medikament} \bowtie \text{Verschreibung} \bowtie \text{Patient}))$

c)  $\pi_{\text{Vorname, Nachname, Diagnose}}(\pi_{\text{Bezeichnung} = \text{"Aspirin"}}(\text{Menge} > 1)(\text{Patient} \bowtie \text{Verschreibung} \bowtie \text{Medikament}))$

- d) Transaktion: *Bestellung* (Patient) - (Patient)  $\rightarrow$  (Verschreibung)  $\rightarrow$  (Ausgabe Medikament))  
Alternativen möglich !!

#### Aufgabe 4: SQL

Gegeben seien folgende Relationen:

Produkt (produktNr, produktName, preis, hersteller)  
Verkäufer (verkäuferNr, verkäuferName, gehalt, abteilung)  
Kunde (kundenNr, kundenName, stadt)  
Rechnung (rechnungNr, kundenNr, verkäuferNr, produktNr, anzahl, datum)

- 1) Geben Sie die folgenden SQL-Anweisungen an:

- a) Erzeugen Sie die Tabellen Produkt und Rechnung. Wählen Sie geeignete Datentypen und berücksichtigen Sie die Primär- und Fremdschlüsselattribute.

```
CREATE TABLE Produkt (  
    produktNr INTEGER PRIMARY KEY,  
    produktName VARCHAR(20),  
    preis DECIMAL,  
    hersteller VARCHAR(20) );  
  
CREATE TABLE Rechnung (  
    rechnungNr INTEGER,  
    kundenNr INTEGER,  
    verkäuferNr INTEGER,  
    produktNr INTEGER,  
    anzahl INTEGER,  
    datum DATE,  
    CONSTRAINT RechnungPK  
    PRIMARY KEY (rechnungNr, kundenNr),  
    CONSTRAINT produktFK  
    FOREIGN KEY (produktNr) REFERENCES Produkt  
    CONSTRAINT kundenFK  
    FOREIGN KEY (kundenNr) REFERENCES Kunde,  
    CONSTRAINT verkäuferFK  
    FOREIGN KEY (verkäuferNr) REFERENCES Verkäufer );
```

- b) Fügen Sie ein Mobiltelefon von Siemens mit dem Namen SL65 in die Produkt-Relation ein. Es soll 240 Euro kosten und die Produktnummer 7 haben.

```
INSERT INTO Produkt VALUES(7, 'SL65', 240.00, 'Siemens');
```

- 2) Formulieren Sie die folgenden beiden SQL-Anfragen unter Verwendung von Unterabfragen und geeigneten Mengen-orientierten Predikaten:

- a) Geben Sie die Kundennummer aller Kunden aus, die ausschließlich Produkte von Siemens bestellt (bzw. in Rechnung gestellt bekommen) haben.

```
SELECT kundenNr  
FROM Rechnung AS Rechnung1  
WHERE 'Siemens' = ALL (SELECT hersteller  
FROM Rechnung NATURAL JOIN Produkt  
WHERE Rechnung1.kundenNr = Rechnung.kundenNr)
```

```
SELECT kundenNr  
FROM Rechnung  
WHERE kundenNr NOT IN (SELECT kundenNr  
FROM Rechnung NATURAL JOIN Produkt)
```

```
WHERE hersteller <> 'Siemens')
```

b) Geben Sie das (die) billigste Produkt(e) vom Hersteller „Siemens“ an.

```
SELECT *
FROM Produkt
WHERE preis < ALL (SELECT preis
                  FROM Produkt
                  WHERE hersteller='Siemens');
```

3) Formulieren Sie die folgenden SQL-Anfragen unter Verwendung von geeigneten Aggregatfunktionen.

a) Geben Sie das minimale und das maximale Gehalt aller Verkäufer der Elektroabteilung an.

```
SELECT MIN(gehalt) AS minGehaltElektro,
       MAX(gehalt) AS maxGehaltElektro
FROM Verkäufer
WHERE abteilung = 'Elektroabteilung';
```

b) Geben Sie für jede Abteilung die Anzahl der dort angestellten Verkäufer an.

```
SELECT abteilung, COUNT(verkaeuferNr) AS anzahlVerkaeufer
FROM Verkäufer
GROUP BY abteilung;
```

c) Geben Sie für alle Verkäufer der Elektroabteilung, die weniger als 100 unterschiedliche Kunden bedient haben, ihre Verkäufernummern, Verkäufernamen und die Anzahl der von ihnen bedienten Kunden an.

```
SELECT verkaeuferNr, verkaeuferName,
       COUNT(DISTINCT kundenNr) AS kundenAnzahl
FROM Verkäufer NATURAL JOIN Rechnung
WHERE abteilung = 'Elektroabteilung'
GROUP BY verkaeuferNr
HAVING COUNT(DISTINCT kundenNr) < 100;
```

### Aufgabe 3: Synchronisationsverfahren

1) Gegeben sei das folgende Schedule:

R1(x) R4(y) W3(x) R2(x) R3(x) W1(x) R4(x) W3(x) W4(y) W2(y) W2(x) (serialisierbar)

a) Erstellen Sie für diese Schedule den Abhängigkeitsgraphen.

b) Ist sie serialisierbar (begründen Sie Ihre Antwort)?  
Wenn ja, geben Sie die äquivalente serielle Schedule an.  
(Beachten Sie auch Aufgabe 4.2 weiter unten!)

a)



b) serialisierbar (IP)  $\Rightarrow$  T5 T4 T2 T1 T3 oder T4 T3 T2 T1 T5

W3(x) R4(y) R4(x) W4(y) W2(y) R2(x) R1(x) W1(x) R3(x) W3(x) W3(x)

a) T1 ist wirkungslos, immer noch serialisierbar :-> T4 T3 T2 T3 oder T3 T4 T2 T3

W3(x) R4(y) R4(z) W4(y) W2(y) R2(x) R3(x) W3(x) W3(x)

2) Gegeben sei das folgende Schedule:

R1(x) R4(y) W3(x) R2(x) R3(x) W1(x) R4(x) W3(x) W4(y) W2(y) W3(x)

Welche (um die Sperrern erweitert) Ergebnisschedule würde vom Zweiphasensperrenprotokoll Psycholining erzeugen? (Geben Sie die vollständige Schedule an!)

W3(x) R4(y) R4(x) W4(y) W2(y) R2(x) R1(x) W1(x) R3(x) W3(x) W3(x)

Benötigte Sperrern:

|    |       |       |       |  |
|----|-------|-------|-------|--|
| T1 | x1(x) |       | x1(y) |  |
| T2 | x2(x) | x2(y) |       |  |
| T3 | x3(x) |       | x3(x) |  |
| T4 |       | x4(y) | x4(x) |  |
| T5 | x5(x) |       |       |  |

R1(x) R4(y) W3(x) R2(x) R3(x) W1(x) R4(x) W3(x) W4(y) W2(y) W3(x)

|        | x                       | y              | z                       | Stück                   |
|--------|-------------------------|----------------|-------------------------|-------------------------|
| R1(x)  | x1(x)                   |                | x1(x)<br>R1(x)<br>u1(x) |                         |
| R4(y)  |                         | x4(y)<br>R4(y) | x4(x)                   |                         |
| W3(x)  |                         |                |                         | W3(x)                   |
| R2(x)  |                         |                |                         | W3(x)<br>R2(x)          |
| R3(x)  |                         |                |                         | W3(x)<br>R3(x)<br>R3(x) |
| W1(x)  | W1(x)<br>u1(x)          |                |                         | W3(x)<br>R3(x)<br>R3(x) |
| >W3(x) | x2(x)<br>W3(x)<br>u2(x) |                |                         | R2(x)<br>R3(x)          |
| >R2(x) |                         |                |                         | R3(x)<br>R2(x)          |
| >R3(x) |                         |                |                         | R2(x)<br>R3(x)          |
| R4(x)  |                         |                | R4(x)<br>u4(x)          | R2(x)<br>R3(x)          |
| >R2(x) |                         |                |                         | R3(x)<br>R3(x)          |
| >R3(x) | x2(x)                   |                | x2(x)<br>R3(x)          | R2(x)                   |
| W2(x)  | W2(x)<br>u2(x)          |                |                         | R2(x)                   |
| >R2(x) |                         |                |                         | R2(x)                   |
| W4(y)  |                         | W4(y)<br>u4(y) |                         | R2(x)                   |
| >R2(x) | x2(x)<br>R2(x)<br>u2(x) |                |                         |                         |
| W2(y)  |                         | W2(y)<br>u2(y) |                         |                         |
| W3(x)  |                         |                | W3(x)<br>u3(x)          |                         |