Lösungen der Übungsaufgaben von Kapitel 10

1. Legen Sie mit einem SQL - Befehl eine neue Tabelle PERSON_KURZ mit den Feldern

   Kurz_Id , Kurz_Name

   an. Machen Sie das so, dass Kurz_Id der Primärschlüssel wird und für Kurz_Name NULL-Werte nicht erlaubt sind. Kurz_Id sei vom Datentyp **int**, Kurz_Name vom Datentyp **varchar**(30). Überprüfen Sie, ob Ihre Setzungen richtig gemacht wurden.

```
CREATE TABLE PERSON_KURZ
(
    Kurz_Id Int    Primary Key,
    Name           varchar(30) not null
)
```

2. (Unbedingte Anzeigen)
   Schreiben Sie SELECT - Befehle für die folgenden Anzeigen:

   a) Zeige alle Orte an, die in der Tabelle PERSON vorkommen

```
SELECT    Ort
FROM      PERSON
```

b) Zeige alle Orte an, die in der Tabelle PERSON vorkommen, aber so, dass jeder Ort nur einmal angezeigt wird

```
SELECT  DISTINCT  Ort
FROM                    PERSON
```

3. (Restriktionen und Projektionen)

   Schreiben Sie SELECT - Befehle für die folgenden Anzeigen. Lassen Sie sich in jeder Aufgabe immer sowohl alle Attribute oder nur ausgewählte Attribute anzeigen:

   a) Zeige alle Personen an, die in Bonn wohnen

```
SELECT      *
FROM    PERSON
WHERE       Ort  =  'Bonn '
```

   b) Zeige alle Artikel, die teurer als 1000.- DM sind.

```
SELECT      *
FROM    ARTIKEL
WHERE       Preis  >  1000
```

   c) Zeige alle Artikel des Lieferanten mit der ***Id*** 6

```
SELECT      *
FROM    ARTIKEL
WHERE       LieferantId = 6
```

d) Zeige alle Artikel, deren ***Preis*** zwischen 100.- und 500.- liegt

```
SELECT      *
FROM    ARTIKEL
WHERE       Preis  >=  100  AND  Preis <=  500
```

e) Zeige alle Artikel der Lieferanten mit den ***Id***s 4,5,6

```
SELECT      *
FROM    ARTIKEL
WHERE       LieferantId  IN  (4 , 5 , 6)
```

f) Zeige alle Artikel, die nicht von den Lieferanten 4,5 oder 6 kommen

```
SELECT      *
FROM    ARTIKEL
WHERE       LieferantId  NOT  IN  (4 , 5 , 6)
```

g) Zeige alle Personen aus den Ländern Frankreich, Italien, Schweiz

```
SELECT      *
FROM    PERSON
WHERE       Land  =  'Frankreich'
OR          Land  =  'Italien'
OR        Land  =  'Schweiz'
```

h) Zeige alle Personen, die nicht in Frankreich, Italien oder der Schweiz wohnen

```
SELECT      *
FROM    PERSON
WHERE
(
       Land  <>  'Frankreich'
AND  Land  <>  'Italien'
AND  Land  <>  'Schweiz'
)
OR    Land  IS NULL
```

i) Zeige alle Artikel mit der ***ArtikelgruppeId*** 1, die teurer als 10.- sind

```
SELECT      *
FROM    ARTIKEL
WHERE       ArtikelgruppeId  =  1
AND         Preis            > 10
```

j) Zeige alle Personen, deren ***Name*** mit M anfängt

```
SELECT      *
FROM    PERSON
WHERE       Name LIKE  'M%'

(ACHTUNG: Nur In Access müssen Sie schreiben: Name LIKE  'M*' )
```

k) Zeige alle Personen, deren ***Name*** mit n , r oder s aufhört

```
SELECT      *
FROM    PERSON
WHERE       Name  LIKE  '%n'
OR              Name  LIKE  '%r'
OR              Name  LIKE  '%s'
```

(ACHTUNG: Nur In Access müssen Sie schreiben: Name LIKE '*n' usw.)

4. Schreiben Sie mit Hilfe von Unterabfragen die folgenden Anzeigen. Lösen Sie die Aufgaben sowohl mit dem IN - Operator als auch mit dem EXISTS-Operator

a) Zeige die ***Id***s aus der Tabelle KUNDE von allen Kunden, die etwas bestellt haben

IN	**EXISTS**
SELECT Id FROM KUNDE WHERE Id IN (SELECT KundeId FROM BESTELLUNGEN)	SELECT Id FROM KUNDE WHERE EXISTS (SELECT * FROM BESTELLUNGEN WHERE KundeId = KUNDE.Id)

b) Zeige die Namen aller Kunden, die etwas bestellt haben.

IN	EXISTS

SELECT Name FROM PERSON
WHERE Id IN
(
  SELECT KundeId
  FROM BESTELLUNGEN
)
	SELECT Name FROM PERSON
	WHERE EXISTS
	(
	  SELECT *
	  FROM BESTELLUNGEN
	  WHERE KundeId = PERSON.Id
	)

c) Zeige die Namen aller Kunden, die Bestellungen von mehr als 50 Stück eines Artikels aufgegeben haben.

IN	EXISTS

SELECT Name FROM PERSON
WHERE Id IN
(
  SELECT KundeId
  FROM BESTELLUNGEN
  WHERE Menge > 50
)
	SELECT Name FROM PERSON
	WHERE EXISTS
	(
	  SELECT *
	  FROM BESTELLUNGEN
	  WHERE KundeId = PERSON.Id
	  AND Menge > 50
	)

d) Zeige die Namen aller Kunden, die den Artikel mit der ***Bezeichnung*** '*Hilfsmotoren*' bestellt haben

IN	**EXISTS**
SELECT Name FROM PERSON WHERE Id IN (SELECT KundeId FROM BESTELLUNGEN WHERE ArtikelId IN (SELECT Id FROM ARTIKEL WHERE Bezeichnung = 'Hilfsmotoren'))	SELECT Name FROM PERSON WHERE EXISTS (SELECT * FROM BESTELLUNGEN WHERE KundeId = PERSON.Id AND EXISTS (SELECT * FROM ARTIKEL WHERE ARTIKEL.Id = ArtikelId AND Bezeichnung = 'Hilfsmotoren'))

5. (Aggregatfunktionen)

a) Zeigen Sie die größte Bestellmenge in der Tabelle Bestellungen

```
SELECT MAX(Menge) FROM BESTELLUNGEN
```

b) Zeigen Sie alle Attribute der Sätze in der Tabelle Bestellungen, bei denen die ***Bestellmenge*** gleich der kleinsten überhaupt vorkommenden Bestellmenge ist

```
SELECT  *   FROM  BESTELLUNGEN
WHERE  Menge =        (
                            SELECT MIN(Menge)  FROM  BESTELLUNGEN
                      )
```

c) Zeigen Sie ***Id*** und Bezeichnung der Artikel an, deren ***Bestellmenge*** in der Tabelle Bestellungen maximal ist

```
SELECT    Id, Bezeichnung
FROM      ARTIKEL
WHERE     Id  IN
(
      SELECT     ArtikelId
      FROM       BESTELLUNGEN
       WHERE     Menge  =
       (
            SELECT     MAX(Menge)
            FROM       BESTELLUNGEN
       )
)
```

d) Finde Sie für jeden Kunden die Bestellung mit der geringsten Menge. Lösen Sie diese Aufgabe zunächst so, dass Sie nur die Attribute aus der Tabelle Bestellungen, also ***Id***, ***KundeId***, ***ArtikelId***, ***Menge*** und ***Datum*** sehen

```
SELECT  B1.*
FROM    BESTELLUNGEN  AS  B1
WHERE  B1.Menge  =
(
      SELECT   MIN(B2.Menge)
      FROM     BESTELLUNGEN  AS  B2
      WHERE    B2.KundeId  =  B1.KundeId
)
```

e) Wie Aufgabe 5d) - aber jetzt sollen auch noch Kundenname und Vorname aus der Tabelle PERSON mit angezeigt werden (Ihr erster **Join** in SQL)

```
SELECT  B1.*, Name, Vorname
FROM    BESTELLUNGEN  AS  B1,  PERSON
WHERE  B1.KundeId  =  PERSON.Id
AND  B1.Menge  =
(
      SELECT   MIN(B2.Menge)
      FROM     BESTELLUNGEN  AS  B2
      WHERE    B2.KundeId  =  B1.KundeId
  )
```

6. (Aggregatfunktionen)

a) Finden Sie die Anzahl der Bestellungen des Artikels mit der ***ArtikelId*** *18*

```
SELECT  COUNT(*)
FROM    BESTELLUNGEN
WHERE   ArtikelId  =  18
```

(Es muss die Zahl 2 herauskommen)

b) Finden Sie die Anzahl der Bestellungen des Artikels *Banane*

```
SELECT  COUNT(*)
FROM    ARTIKEL  A, BESTELLUNGEN  B
WHERE   A.Id = B.ArtikelId
AND     A.Bezeichnung = 'Banane'
```

(Es muss die Zahl 3 herauskommen)

c) Finden Sie die Anzahl der Bestellungen eines Artikels der Artikelgruppe *Kleinteile*

```
SELECT  COUNT(*)
FROM    ARTIKEL A, ARTIKELGRUPPE AG, BESTELLUNGEN B
WHERE   A.ArtikelgruppeId = AG.Id
AND     A.Id = B.ArtikelId
AND     AG.Bezeichnung = 'Kleinteile'
```

(Es muss die Zahl 5 herauskommen)

d) Finden Sie für jeden Kunden die Anzahl der verschiedenen ***ArtikelId***s, die er bestellt hat. Ausgegeben werden sollen: ***Name*** und ***Vorname*** des Kunden und die Anzahl der verschiedenen ***ArtikelId***s, die er bestellt hat - absteigend nach der ***ArtikelId*** - Anzahl sortiert

```
SELECT    Name, Vorname, COUNT(DISTINCT ArtikelId) AS Anzahl
FROM      PERSON P, BESTELLUNGEN B
WHERE     P.Id = B.KundeId
GROUP BY  Name, Vorname
ORDER BY  3 DESC
```

Hinweis: In Access dürfen Sie kein DISTINCT hinter Count schreiben

e) Finden Sie für jeden Kunden die Gesamtmenge aller Artikel, die er bestellt hat. Ausgegeben werden sollen: Name und Vorname des Kunden und die Gesamtmenge aller Artikel, die er bestellt hat - absteigend nach der Menge sortiert

```
SELECT    Name, Vorname, SUM(Menge) AS Gesamtmenge
FROM      PERSON P, BESTELLUNGEN B
WHERE     P.Id = B.KundeId
GROUP BY  Name, Vorname
ORDER BY  3 DESC
```

f) Finden Sie für jeden Kunden die Gesamtbetrag, den er an Sie bezahlt hat bzw. zu bezahlen hat. Ausgegeben werden soll: Name und Vorname des Kunden und der Gesamtbetrag - absteigend nach dem Gesamtbetrag sortiert

```
SELECT    Name, Vorname, SUM(Menge*Preis) AS Gesamtbetrag
FROM      PERSON P, BESTELLUNGEN B, ARTIKEL A
WHERE     P.Id = B.KundeId
AND       A.Id = B.ArtikelId
GROUP BY Name, Vorname
ORDER BY 3 DESC
```

g) Finden Sie für jeden Lieferanten die Anzahl der verschiedenen Artikel, die er liefert. Ausgegeben werden sollen: Name und Vorname des Lieferanten und die Anzahl der Artikel, die er liefert - absteigend nach der Artikelanzahl sortiert

```
SELECT    Name, Vorname, COUNT(A.Id) AS Anzahl
FROM      PERSON P, ARTIKEL A
WHERE     P.Id = A.LieferantId
GROUP BY Name, Vorname
ORDER BY 3 DESC
```

7. (Joins)

a) Ermitteln Sie die Daten aus der Tabelle ARTIKEL von allen Artikeln, die der Kunde Arnold Hau bestellt hat

```
SELECT  A.*
FROM    ARTIKEL  A,  BESTELLUNGEN  B,   PERSON  P
WHERE   A.Id        =   B.ArtikelId
AND     B.KundeId   =   P.Id
AND     P.Name      =   'Hau'
AND     P.Vorname   =   'Arnold'
```

b) Ermitteln Sie die Namen und Vorname aller Kunden, für die keine Bestellungen vorliegen

```
SELECT      Name, Vorname
FROM        PERSON P, KUNDE K
WHERE       P.Id    =   K.Id
AND         P.Id    NOT  IN
(
        SELECT      KundeId
        FROM        BESTELLUNGEN
)
```

c) Finden Sie für jeden Kunden die Gesamtbetrag, den er an Sie bezahlt hat bzw. zu bezahlen hat. Ausgegeben werden soll Kundenname und Gesamtbetrag - aufsteigend nach dem Gesamtbetrag sortiert, aber nur für die Kunden, bei denen dieser Gesamtbetrag größer als 1000.- ist

```
SELECT    Name, SUM(Menge*Preis) AS Gesamtbetrag
FROM      PERSON P, BESTELLUNGEN B, ARTIKEL A
WHERE     P.Id = B.KundeId
AND       B.ArtikelId = A.Id
GROUP BY  Name
HAVING    SUM(Menge*Preis) > 1000
ORDER BY  2
```

d) Ermitteln Sie den durchschnittlichen Geldwert aller Bestellungen

```
SELECT    AVG(Preis*Menge) AS Durchschnitt
FROM      BESTELLUNGEN B, ARTIKEL A
WHERE     B.ArtikelId = A.Id
```

e) Geben Sie die Daten aller Bestellungen aus, deren Geldwert größer als der durchschnittliche Geldwert aller Bestellungen ist (natürlich ohne die explizite Wertangabe aus der vorigen Aufgabe zu benutzen). Ausgegeben werden sollen die Daten der Tabelle BESTELLUNGEN dazu der jeweilige Geldwert - aufsteigend nach Geldwert sortiert

```
SELECT      B1.*, Menge*Preis  AS  Bestellwert
FROM        BESTELLUNGEN   B1, ARTIKEL  A1
WHERE       B1.ArtikelId  =   A1.Id
AND         Menge * Preis  >
(
        SELECT      AVG(Preis*Menge)
        FROM        BESTELLUNGEN  B2,  ARTIKEL  A2
        WHERE       B2.ArtikelId  =   A2.Id
)
ORDER BY  6
```

f) Zeigen Sie jede Bestellung an und – falls die Bestellung bereits erledigt wurde – dazu noch das Commitdatum. Sortieren Sie die Anzeige absteigend nach dem Commitdatum.

```
SELECT      B.*,  EB.Commitdatum
FROM        BESTELLUNGEN  B
                    LEFT OUTER JOIN
                                ERLEDIGTEBESTELLUNGEN  EB
            ON B.Id  =  EB.Id
ORDER BY  Commitdatum       DESC
```

g) Zeigen Sie jeden Kunden mit seinem Namen und seiner Kundennr an und – falls dieser Kunde auch noch etwas bestellt hat – die zugehörigen Bestelldaten.

```
SELECT   P.Name, K.Kundennr, B.*
FROM     (
                  PERSON P  INNER JOIN  KUNDE K
                  ON P.Id = K.Id
         )
              LEFT OUTER JOIN BESTELLUNGEN B
              ON K.Id = B.KundeId
ORDER BY  P.Name
```

h) Zeigen Sie jeden Kunden mit seinem Namen und seiner Kundennr an und die Anzahl seiner Bestellungen. Es sollen auch die Kunden mit keiner Bestellung angezeigt werden. Sortieren Sie abwärts nach der Anzahl der Bestellungen.

```
SELECT   P.Name, K.Kundennr, COUNT(B.Id)
FROM     (
                  PERSON P  INNER JOIN  KUNDE K
                  ON P.Id = K.Id
         )
              LEFT OUTER JOIN BESTELLUNGEN B
              ON K.Id = B.KundeId
GROUP BY  P.Name, K.Kundennr
ORDER BY  3 DESC
```

8. (Insert und referentielle Integrität)

   Sie wollen in die Tabelle BESTELLUNGEN (***Id***, ***KundeId***, ***ArtikelId***, ***Menge***, ***Bestelldatum***) den Satz (*99* , *14* , *15* , *100*) einfügen. Geben Sie hierfür den SQL - Befehle an, der gleichzeitig die referentielle Integrität bezüglich der Tabellen KUNDE und ARTIKEL mit überprüft.

```
INSERT   INTO   BESTELLUNGEN
(
        Id, KundeId, ArtikelId, Menge
)
SELECT      99, K.Id, A.Id, 100
FROM        KUNDE K, ARTIKEL A
WHERE       K.Id = 14
AND         A.Id = 15
```

9. (Update)

   a) Erhöhen Sie den ***Preis*** aller Artikel, deren Bestell***menge*** in der Tabelle BESTELLUNGEN überdurchschnittlich hoch ist, um 12%

```
UPDATE      ARTIKEL
SET         Preis = Preis*1.12
WHERE       Id   IN
(
    SELECT      ArtikelId
    FROM        BESTELLUNGEN B1
    WHERE       B1.Menge >
    (
        SELECT      AVG(Menge)
        FROM        BESTELLUNGEN B2
    )
)
```

b) Erniedrigen Sie die Preise aller Artikel um 5%, für die keine Bestellungen vorliegen

```
UPDATE      ARTIKEL
SET         Preis = Preis * 0.95
WHERE       Id  NOT IN
(
    SELECT      ArtikelId
    FROM        BESTELLUNGEN
)
```

10. (Delete und referentielle Integrität)

Sie wollen den Lieferanten mit der ***Id*** 3 löschen. Geben Sie ein einziges SQL-Statement an, mit dem dieser Lieferant genau dann gelöscht wird, wenn er nicht in der Artikel-Tabelle als Lieferant eines Artikels geführt wird.

```
DELETE FROM   LIEFERANT
WHERE         Id = 3
AND           Id  NOT IN
(
    SELECT      LieferantId
    FROM        ARTIKEL
)
```

http://www.springer.com/978-3-8351-0163-0

Datenbanken
Theorie, Entwurf und Programmierung relationaler Datenbanken
Schubert, M.
2007, XII, 344 S. Mit Online-Extras., Softcover
ISBN: 978-3-8351-0163-0