Name: _____
Klasse: _____
Datum: _____
Schuljahr: 2022/23

Verwendete DB-Version:

```
+---------------------------------+-------------+
| Version MariaDB                 | Version gm3 |
+---------------------------------+-------------+
| 10.6.11-MariaDB-0ubuntu0.22.04.1 | 2022-08-12  |
+---------------------------------+-------------+
```

1 Aggregatsfunktionen

Die Buchhaltung möchte herausfinden, wie viel insgesamt an Gehalt ausbezahlt wird. Damit die Summe aller Gehälter berechnet werden kann, benötigt man eine **Aggregatsfunktion**. Dies sind Funktionen, die alle Werte einer Spalte zusammenfassen und auswerten. Für die Summe aller Gehälter benötigt man die Aggregatsfunktion `sum`. Werden Aggregatsfunktionen verwendet, so sollte auch ein aussagekräftiger Alias verwendet werden:

```
USE gm3;
SELECT sum(gehalt) as 'Summe aller Gehälter'
       FROM gehalt;
```

Kontrollergebnis

```
+----------------------+
| Summe aller Gehälter |
+----------------------+
| 79900.77             |
+----------------------+
```

In SQL sind die wichtigsten Aggregatsfunktionen:

`sum` Summiert alle Einträge in einer Spalte.

`min` Bestimmt den minimalen Wert einer Spalte.

`max` Bestimmt den maximalen Wert einer Spalte.

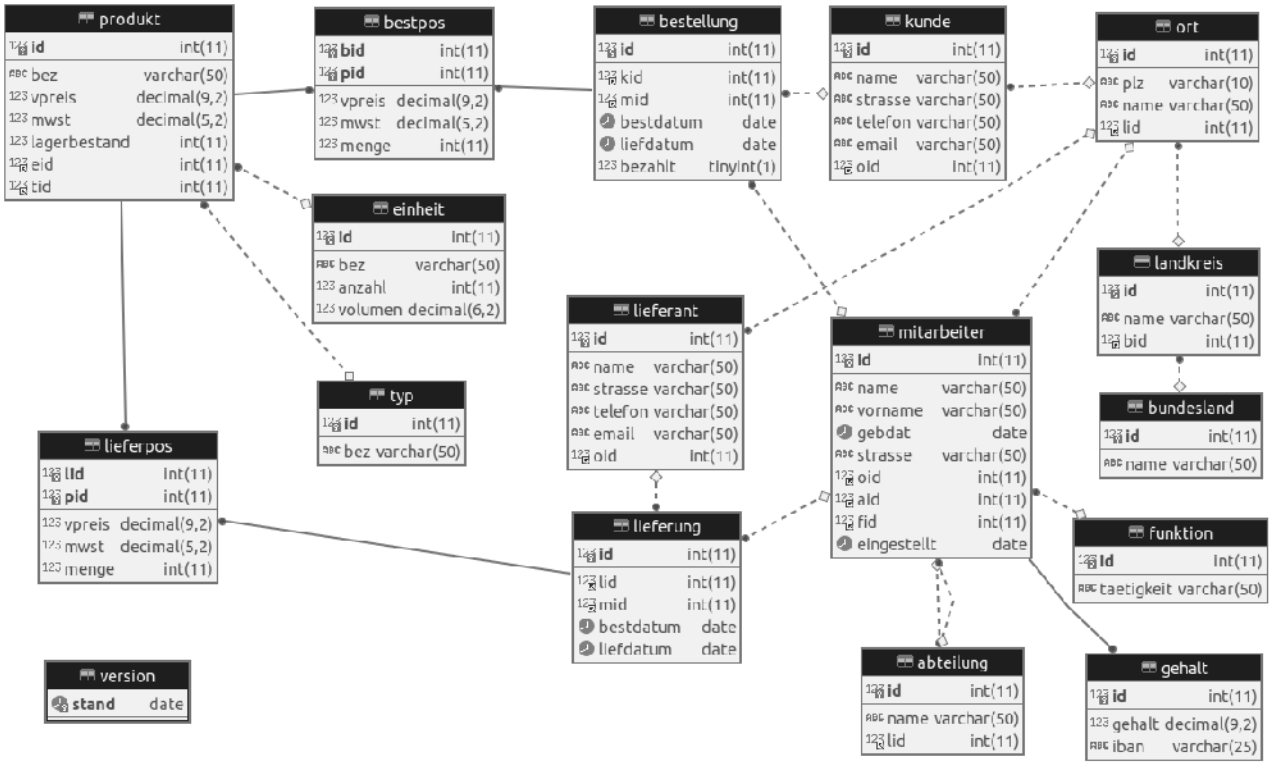
`avg` Bestimmt den Durchschnittswert einer Spalte.

`count` Bestimmt die Anzahl der Datensätze einer Spalte.

Bei der Aggregatsfunktion `count` gilt dabei folgendes:

- Mit `SELECT count(name) FROM mitarbeiter` bestimmt man die Anzahl aller Mitarbeiternachnamen. `NULL`-Werte werden dabei **nicht** berücksichtigt.
- Mit `count(distinct name)` kann man die Anzahl aller **unterschiedlichen** Mitarbeiternachnamen bestimmen.
- Mit `count(*)` bestimmt man die Anzahl aller Datensätze (incl. eventueller kompletter `NULL`-Zeilen).

Tabellenmodell für den Getränkemarkt – gm3



Aufgaben zu Aggregatsfunktionen AUF-08-4-1

1. Bestimmen Sie das Maximum und Minimum von allen Gehältern.

Kontrollergebnis

```
+-----------------+-----------------+
| Maximales Gehalt | Minimales Gehalt |
+-----------------+-----------------+
| 10490.77         | 1100.00          |
+-----------------+-----------------+
```

2. Bestimmen Sie das durchschnittliche Gehalt aller Mitarbeiter. Runden Sie den Durchschnitt dabei auf zwei Nachkommastellen. (Hinweis: `ROUND`)

Kontrollergebnis

```
+--------------------+
| Durchschnittsgehalt |
+--------------------+
| 2496.90             |
+--------------------+
```

3. Wie viele unterschiedliche Mitarbeiter haben bisher Bestellungen betreut?

Kontrollergebnis

```
+---------------------------------+
| Anzahl unterschiedlicher Betreuer |
+---------------------------------+
| 12                              |
+---------------------------------+
```

4. Wie viele Mitarbeiter arbeiten durchschnittlich in einer Abteilung? Abteilungen ohne Mitarbeiter sollen für diese Rechnung nicht berücksichtigt werden.
Runden Sie dabei das Ergebnis auf eine Nachkommastelle.

Kontrollergebnis

```
+-----------------------------------------------+
| Durchschnittl. Anzahl Mitarbeiter pro Abteilung |
+-----------------------------------------------+
| 8.2                                           |
+-----------------------------------------------+
```

2 Gruppierung

Die Buchhaltung möchte die Summe aller Gehälter genauer aufgeschlüsselt haben, und zwar möchte sie eine Übersicht über die Summe der Gehälter, die pro Abteilung ausbezahlt werden:

Kontrollergebnis

```
+-----+-----------------+
| aid | Summe Gehaelter |
+-----+-----------------+
| 1   | 12298.00        |
| 2   | 13588.00        |
| 3   | 10703.00        |
| 4   | 22876.00        |
| 5   | 20435.77        |
+-----+-----------------+
```

Mit folgendem SQL-Befehl wurde schon mal eine Übersicht über die Mitarbeiter-ID, die dazugehörige Abteilungs-ID und das entsprechende Gehalt erstellt:

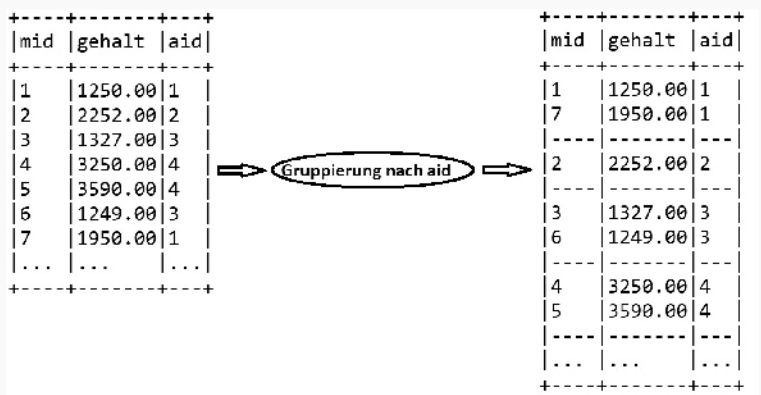
```
USE gm3;
SELECT m.id as 'mid', gehalt, aid
   FROM mitarbeiter m, gehalt g
   WHERE m.id=g.id
   LIMIT 7;
```

Kontrollergebnis (Auszug)

```
+-----+---------+-----+
| mid | gehalt  | aid |
+-----+---------+-----+
| 1   | 1250.00 | 1   |
| 2   | 2252.00 | 2   |
| 3   | 1327.00 | 3   |
| 4   | 3250.00 | 4   |
| 5   | 3590.00 | 4   |
| 6   | 1249.00 | 3   |
| 7   | 1950.00 | 1   |
+-----+---------+-----+
```

Um die gewünschte Aufschlüsselung nach der Abteilung zu bekommen, müssen alle Datensätze mit der selben `aid` zusammengefasst werden und anschließend alle Gehälter pro `aid` zusammengezählt werden. Das Zusammenfassen der Datensätze mit der selben `aid` nennt man **Gruppierung nach `aid`**:

Gruppierung



Nach diesem Gruppieren können dann für jeden Abteilungsblock die Gehälter zusammengezählt werden. In SQL wird für das Gruppieren das Schlüsselwort **GROUP BY** verwendet, dieses steht nach der `WHERE`-Klausel:

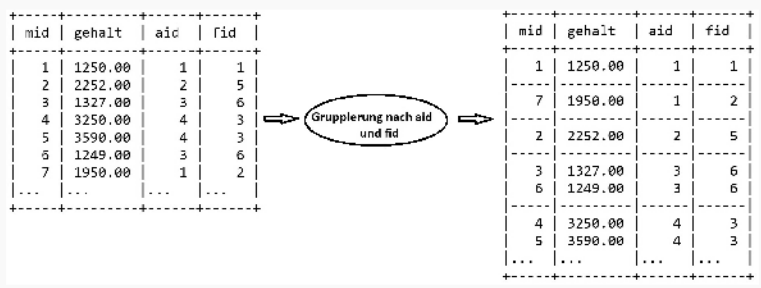
```
USE gm3;
SELECT aid, sum(gehalt) as 'Summe Gehaelter'
   FROM mitarbeiter m, gehalt g
   WHERE m.id = g.id
   GROUP BY aid;
```

Kontrollergebnis

```
+-----+-----------------+
| aid | Summe Gehaelter |
+-----+-----------------+
| 1   | 12298.00        |
| 2   | 13588.00        |
| 3   | 10703.00        |
| 4   | 22876.00        |
| 5   | 20435.77        |
+-----+-----------------+
```

Möchte man zusätzlich die Summe aller Gehälter pro Abteilung für jede Funktion erfassen, so muss nach den zwei Attributen `aid` und `fid` gruppiert werden:

Gruppierung



In SQL lautet der entsprechende Befehl folgendermaßen:

```
USE gm3;
SELECT aid, fid, sum(gehalt) as 'Summe Gehaelter'
   FROM mitarbeiter m, gehalt g
   WHERE m.id=g.id
   GROUP BY aid, fid
   LIMIT 7;
```

Kontrollergebnis (Auszug)

```
+-----+-----+-----------------+
| aid | fid | Summe Gehaelter |
+-----+-----+-----------------+
| 1   | 1   | 1250.00         |
| 1   | 2   | 1950.00         |
| 1   | 4   | 1100.00         |
| 1   | 5   | 4228.00         |
| 1   | 6   | 3770.00         |
| 2   | 1   | 2000.00         |
| 2   | 3   | 3366.00         |
+-----+-----+-----------------+
```

Möchte man beispielsweise beim obigen Ergebnis noch die Mitarbeiter-ID's mit anzeigen, so führt folgender SQL-Befehl unter MySQL oder MariaDB zu folgendem Ergebnis:

```
USE gm3;
SELECT m.id as 'mid', aid, sum(gehalt) as 'Summe Gehaelter'
   FROM mitarbeiter m, gehalt g
   WHERE m.id = g.id
   GROUP BY aid;
```

Kontrollergebnis

```
+-----+-----+-----------------+
| mid | aid | Summe Gehaelter |
+-----+-----+-----------------+
| 1   | 1   | 12298.00        |
| 2   | 2   | 13588.00        |
| 3   | 3   | 10703.00        |
| 4   | 4   | 22876.00        |
| 11  | 5   | 20435.77        |
+-----+-----+-----------------+
```

Je nach SQL-Dialekt und Einstellung beim Server kann man sehen, dass die Mitarbeiter-ID's 5 und 6 nicht im Ergebnis für Abteilung 4 bzw. 3 auftauchen. Andere SQL-Dialekte, wie beispielsweise MS SQL sind da um einiges strikter und werfen eine Fehlermeldung:
`Column mid is invalid in the select list because it is not contained in either an aggregate function or the GROUP BY clause`
Wird in einer SQL-Abfrage der Befehl `GROUP BY` verwendet, so dürfen bei MS SQL in der `SELECT`-Zeile, neben Attributen in einer Aggregatsfunktion, nur noch Attribute aus dem `GROUP BY`- Befehl verwendet werden.

Aufgaben zu Gruppierung AUF-08-4-2

1. Bestimmen Sie für jeden Mitarbeiter (Id), der schon einmal eine Bestellung betreut haben, die Anzahl der Bestellungen, die er jeweils betreut hat. Sortieren Sie dabei das Ergebnis nach dieser Anzahl absteigend.

Kontrollergebnis (Auszug)

```
+-----+------------------------------+
| mid | Anzahl betreuter Bestellungen |
+-----+------------------------------+
| 24  | 16                           |
| 11  | 15                           |
| 6   | 13                           |
| 9   | 13                           |
+-----+------------------------------+
```

2. Bestimmen Sie für jeden Mitarbeiter (Id, Name, Vorname) die Anzahl der Lieferungen, die er jeweils betreut hat. Sortieren Sie dabei das Ergebnis nach dieser Anzahl aufsteigend. Bei gleicher Anzahl soll nach der Mitarbeiter-ID aufsteigend sortiert werden.

Kontrollergebnis (Auszug)

```
+----+--------+---------+-----------------------------+
| id | name   | vorname  | Anzahl betreuter Lieferungen |
+----+--------+---------+-----------------------------+
| 38 | Huber  | Sepp     | 0                           |
| 6  | Wolff  | Bettina  | 1                           |
| 9  | Schulz | Wilfried | 1                           |
| 13 | Beck   | Rafaela  | 1                           |
+----+--------+---------+-----------------------------+
```

3. Bestimmen Sie für jedes Produkt (Id, Bezeichnung) in welchen Mengen es insgesamt schon bestellt wurde. Sortieren Sie dabei das Ergebnis nach der Produkt-ID aufsteigend.

Kontrollergebnis (Auszug)

```
+----+-------------------------------------+----------------+
| id | bez                                 | Bestellte Menge |
+----+-------------------------------------+----------------+
| 1  | Binding Export                      | 113            |
| 2  | Dachsenfranz Kellerbier Bügelflasche | 53             |
| 3  | Eichbaum Export                     | 21             |
| 4  | Heidelberger Export                 | 70             |
| 5  | Kurpfalz Bräu Kellerbier            | 114            |
| 6  | Welde Export                        | 0              |
+----+-------------------------------------+----------------+
```

4. Ermitteln Sie für jede Bestellung (Id, Bestelldatum) den gesamten zu zahlenden Rechnungsbetrag (Netto). Dieser setzt sich aus dem Produkt von Verkaufspreis und Menge jedes einzelnen Produkts zusammen. Sortieren Sie dabei das Ergebnis nach diesem Rechnungsbetrag absteigend.

Kontrollergebnis (Auszug)

```
+-----+------------+-----------------+
| id  | bestdatum  | Rechnungsbetrag |
+-----+------------+-----------------+
| 3   | 2021-07-22 | 9385.28         |
| 68  | 2021-10-09 | 8404.62         |
| 26  | 2021-08-21 | 7790.29         |
| 132 | 2021-11-04 | 7503.58         |
+-----+------------+-----------------+
```

5. Bestimmen Sie für jeden Mitarbeiter, wie oft er einen bestimmten Lieferanten betreut hat (Mitarbeiter, die keinen Lieferanten betreuen, sollen nicht berücksicht werden). Sortieren Sie dabei das Ergebnis nach der Mitarbeiter-ID aufsteigend. Bei gleicher Mitarbeiter-ID soll nach dem Namen des Lieferanten aufsteigend sortiert werden.

Kontrollergebnis (Auszug)

```
+----+--------+---------+--------------------------------+--------+
| id | name   | vorname | name                           | Anzahl |
+----+--------+---------+--------------------------------+--------+
| 1  | Lorenz | Sophia  | Adelholzener Alpenquellen GmbH | 1      |
| 1  | Lorenz | Sophia  | Augustiner Brauerei GmbH       | 1      |
| 1  | Lorenz | Sophia  | Winzer Franke KG               | 3      |
| 2  | Ritter | Tatjana | Adelholzener Alpenquellen GmbH | 2      |
+----+--------+---------+--------------------------------+--------+
```

6. Optional: für Experten

Ermitteln Sie für jeden Lieferanten (Id, Name), wie viel diesem pro Monat im Jahr 2021 aufgrund der gelieferten Produkte zu zahlen war. Dieser Betrag setzt sich ebenfalls aus dem Produkt von Menge und Preis jedes Produkts zusammen. Für den Monat soll dabei der entsprechende deutsche Name angezeigt werden. Sortieren Sie dabei das Ergebnis nach dem Monat aufsteigend. Bei gleichem Monat soll nach der Lieferanten-Id aufsteigend sortiert werden.

Kontrollergebnis (Auszug)

```
+----+--------------------------+---------+---------------+
| id | name                     | Monat   | Einkaufspreis |
+----+--------------------------+---------+---------------+
| 2  | Red Bull AG              | Januar  | 2628.28       |
| 3  | Coca Cola Deutschland AG | Januar  | 1702.18       |
| 4  | Winzer Franke KG         | Januar  | 2994.92       |
| 1  | Augustiner Brauerei GmbH | Februar | 3607.15       |
| 2  | Red Bull AG              | Februar | 1999.28       |
| 3  | Coca Cola Deutschland AG | Februar | 3261.96       |
+----+--------------------------+---------+---------------+
```

3 Schlüsselwort HAVING

Die Buchhaltung möchte nur die Abteilungen, in denen insgesamt mindestens 20.000 € Gehalt an die jeweiligen Mitarbeiter ausbezahlt werden.
Man könnte nun versuchen diese Bedingung mithilfe des `sum`-Befehls in der `WHERE`-Klausel zu formulieren:

```
SELECT aid, sum(gehalt) as 'Summe Gehaelter'
   FROM mitarbeiter m INNER JOIN gehalt g ON m.id=g.id
   WHERE sum(gehalt)>=20000
   GROUP BY aid;
```

Jedoch führt die Ausführung dieses Befehls zu folgender Fehlermeldung:

```
ERROR 1111 (HY000): Invalid use of group function
```

Dies liegt an der internen Auswertungsreihenfolge eines SQL-Statements. Zuerst werden die `FROM`- und `WHERE`-Klauseln ausgewertet und erst danach wird die `GROUP BY`-Klausel ausgeführt. Jedoch kann die Bedingung `sum(gehalt)>=20000` erst nach der Gruppierung ausgewertet werden.
Für Bedingungen, die nach der Gruppierung ausgewertet werden sollen, benötigt man ein neues Schlüsselwort `HAVING`. Diese Klausel steht nach der `GROUP BY`-Klausel:

```
USE gm3;
SELECT aid, sum(gehalt) as 'Summe Gehaelter'
   FROM mitarbeiter m INNER JOIN gehalt g ON m.id = g.id
   GROUP BY aid
   HAVING sum(gehalt)>=20000;
```

Kontrollergebnis

```
+-----+-----------------+
| aid | Summe Gehaelter |
+-----+-----------------+
| 4   | 22876.00        |
| 5   | 20435.77        |
+-----+-----------------+
```

Notiz:

Die schrittweise Auswertungsreihenfolge eines SQL-Befehls wird an folgendem Beispiel illustriert:

```
SELECT A, sum(D)
   FROM ...
   Where...
   GROUP BY A, B
   HAVING sum(D)<10 AND max(C)=4;
```

1. Schritt: From/Where

```
+----+----+----+----+
| A  | B  | C  | D  |
+----+----+----+----+
|  1 |  2 |  3 |  4 |
|  1 |  2 |  4 |  5 |
|  2 |  3 |  3 |  4 |
|  3 |  3 |  4 |  5 |
|  3 |  3 |  6 |  7 |
+----+----+----+----+
```

2. Schritt: Gruppenbildung

```
+----+----+----+----+
| A  | B  | C  | D  |
+----+----+----+----+
|  1 |  2 |  3 |  4 |
|  1 |  2 |  4 |  5 |
|----|----|----|----|
|  2 |  3 |  3 |  4 |
|----|----|----|----|
|  3 |  3 |  4 |  5 |
|  3 |  3 |  6 |  7 |
+----+----+----+----+
```

3. Schritt: Interne Aggregation

```
+----+----+-------+-------+
| A  | B  | sum(D)| max(C)|
+----+----+-------+-------+
|  1 |  2 |     9 |     4 |
|  2 |  3 |     4 |     3 |
|  3 |  3 |    12 |     6 |
+----+----+-------+-------+
```

4. Schritt: having (Selektion)

```
+----+----+-------+-------+
| A  | B  | sum(D)| max(C)|
+----+----+-------+-------+
|  1 |  2 |     9 |     4 |
+----+----+-------+-------+
```

5. Schritt: Projektion

```
+----+-------+
| A  | sum(D)|
+----+-------+
|  1 |     9 |
+----+-------+
```

Notiz:

Aufgaben zu HAVING AUF-08-4-3

1. Geben Sie für jede Abteilung (Id) die jeweilige ausbezahlte Summe an Gehalt an. Dabei sollen nur Abteilungen im Ergebnis aufgelistet werden, in denen mindestens sechs Mitarbeiter arbeiten.

Kontrollergebnis

```
+-----+-----------------+
| aid | Summe Gehaelter |
+-----+-----------------+
| 1   | 12298.00        |
| 2   | 13588.00        |
| 3   | 10703.00        |
| 4   | 22876.00        |
+-----+-----------------+
```

2. Bestimmen Sie für alle Produkte (Id, Bezeichnung) die Anzahl der verkauften Menge. Dabei sollen nur diejenigen Produkte angezeigt werden, von denen mindestens 200 Einheiten verkauft worden sind. Sortieren Sie das Ergebnis nach der verkauften Menge aufsteigend, dann nach der Bezeichnung.

Kontrollergebnis (Auszug)

```
+-----+----------------------------+-----------------+
| id  | bez                        | Verkaufte Menge |
+-----+----------------------------+-----------------+
| 112 | Coca-Cola Zero             | 202             |
| 184 | Ensinger Gourmet Bio Still | 202             |
| 123 | Odenwald Quelle Classic    | 214             |
+-----+----------------------------+-----------------+
```

3. Ermitteln Sie alle Produkte (Id, Bezeichnung), die von mehr als zwei unterschiedlichen Lieferanten geliefert werden. Sortieren Sie dabei nach der Produkt-Id aufsteigend.

Kontrollergebnis (Auszug)

```
+-----+----------------------------+
| id  | bez                        |
+-----+----------------------------+
| 46  | Paulaner Weißbier Hefe hell |
| 88  | Fanta                      |
| 101 | Coca-Cola                  |
+-----+----------------------------+
```

4. Ermitteln Sie alle Kunden (Id, Name), die im Oktober 2021 mehr als fünf Bestellungen aufgegeben haben.

Kontrollergebnis

```
+----+-----------------------------+-------------+
| id | name                        | Bestellungen |
+----+-----------------------------+-------------+
| 3  | Dorfwirt                    | 6           |
| 5  | Pizzeria da Mario           | 7           |
| 9  | Kräuterrestaurant Liebstöckl | 6           |
| 11 | Villa im Tal                | 6           |
+----+-----------------------------+-------------+
```

5. Bestimmen Sie für alle Mitarbeiter (Id, Nachname, Anzahl betreuter Lieferungen) die Anzahl betreuter Lieferungen. Dabei sollen im Ergebnis nur Mitarbeiter angezeigt werden, die weniger als drei Lieferungen betreut haben. Sortieren Sie das Ergebnis nach der der Anzahl betreuter Lieferungen aufsteigend. Bei gleicher Anzahl soll nach der Mitarbeiter-ID aufsteigend sortiert werden.

Kontrollergebnis (Auszug)

```
+----+--------+-----------------------------+
| id | name   | Anzahl betreuter Lieferungen |
+----+--------+-----------------------------+
| 38 | Huber  | 0                           |
| 6  | Wolff  | 1                           |
| 9  | Schulz | 1                           |
+----+--------+-----------------------------+
```

6. Bestimmen Sie alle Produkte (Id, Bezeichnung, Summe Kunden), die (eventuell mehrmals) nur von **einem** Kunden gekauft wurden. Sortieren Sie dabei das Ergebnis nach der Produkt-Id aufsteigend.

Kontrollergebnis (Auszug)

```
+----+--------------------------------+-------------+
| id | bez                            | Summe Kunden |
+----+--------------------------------+-------------+
| 3  | Eichbaum Export                | 1           |
| 4  | Heidelberger Export            | 1           |
| 48 | Paulaner Weißbier Hefe dunkel  | 1           |
+----+--------------------------------+-------------+
```

7. Optional: für Experten

Gesucht sind alle Lieferanten (Id), zusammen mit den (aufs Jahr gerechnet) durchschnittlichen Lieferkosten pro Monat (gerundet auf zwei Nachkommastellen), die jeden Monat im Jahr 2021 Produkte geliefert haben.

Kontrollergebnis

```
+----+-----------------------------------+
| id | Durchschnittliche Kosten pro Monat |
+----+-----------------------------------+
| 3  | 2098.91                           |
+----+-----------------------------------+
```

4 Subselect

Die Marketingabteilung möchte für Analysezwecke die Namen und Verkaufspreise derjenigen Produkte, die (im Vergleich zu allen Produkten) überdurchschnittlich viel kosten.

Ein erster Ansatz könnte dabei wie folgt aussehen:

```
SELECT bez, vpreis
   FROM Produkt
   WHERE vpreis > avg(vpreis)
```

Dieser Befehl liefert jedoch die Fehlermeldung:

```
ERROR 1111 (HY000): Invalid use of group function
```

Dies liegt daran, dass bei der Überprüfung jeder einzelne Datensatz durchgegangen wird und die `avg`-Funktion den Preis des aktuellen Datensatzes verwendet. Nur wenn die Aggregatsfunktion im `SELECT`-Statement steht, berechnet `avg` den Durchschnitt aus allen Preisen. Damit also alle Produkte mit einem überdurchschnittlichen Preis herausgefunden werden können, ist ein so genanntes **`Subselect`** notwendig:

```
USE gm3;
SELECT bez, vpreis
   FROM produkt
   WHERE vpreis > (
      SELECT avg(vpreis) FROM produkt
   )
   ORDER BY 2 asc, 1
   LIMIT 4;
```

Kontrollergebnis

```
+------------------+--------+
| bez              | vpreis |
+------------------+--------+
| Coca-Cola        | 13.25  |
| Coca-Cola light  | 13.25  |
| Coca-Cola Zero   | 13.25  |
| Fanta Orange     | 13.25  |
+------------------+--------+
```

Zum Vergleich der Durchschnittspreis:

```
USE gm3;
SELECT avg(vpreis) as 'Durchschnittspreis'
   FROM produkt;
```

Kontrollergebnis

```
+--------------------+
| Durchschnittspreis |
+--------------------+
| 13.120037          |
+--------------------+
```

Zu beachten ist, ob ein Subselect in der `Where`- Klausel nur **einen** (wie im obigen Beispiel) oder **mehrere** Werte zurückgeben kann. Wird nur ein Wert zurückgegeben, so kann dieser Wert in der Hauptabfrage mit `<`, `>`, `=`, `<>`, `...` verglichen werden.
Liefert ein Subselect mehrere Werte zurück, so können diese Operatoren **nicht mehr** verwendet werden, da es sonst zur folgenden Fehlermeldung kommt:

```
ERROR 1242 (21000): Subquery returns more than 1 row
```


Für solche Subselects kann beispielsweise der **IN** Operator verwendet werden:

```
USE gm3;
SELECT id
   FROM lieferant
   WHERE id NOT IN (
      SELECT l.lid
         FROM lieferung l INNER JOIN lieferpos lpos ON l.id=lpos.lid
                          INNER JOIN produkt p ON p.id=pid
                          INNER JOIN typ t ON t.id=p.tid
         WHERE t.bez like '%wein%'
   );
```

Kontrollergebnis

```
+----+
| id |
+----+
| 1  |
+----+
```

Ergebnis des Subselects

```
+-----+
| lid |
+-----+
| 5   |
| 2   |
| 4   |
| 3   |
+-----+
```

1. Beschreiben Sie in Worten, welches Ergebnis das Subselect liefert und was dadurch mit dem gesamten SQL-Statement ermittelt wird.

Diese Art von Subselects für Bedingungen können auch in der `Having`–Klausel verwendet werden, falls die entsprechende Bedingung erst nach der Gruppierung ausgewertet werden soll.

Hinweis: Subselects stellen in der Regel einen erheblichen Aufwand für die Datenbank dar. Überlegen Sie sich daher immer, ob ein Subselect wirklich notwendig ist oder das Subselect durch einen perfomanteren `JOIN` ersetzt werden kann.

Notiz:

Optional für Experten:
Neben Subselects in der `WHERE`-Klausel, können diese auch in der `SELECT` bzw. `From`-Klausel verwendet werden.
Ein SQL-Statement mit Subselects in der `SELECT`-Klausel könnte beispielsweise folgendermaßen aussehen:

```
USE gm3;
SELECT m.id,
      (SELECT count(*) FROM mitarbeiter mi
         INNER JOIN bestellung b ON mi.id=b.mid WHERE m.id=mi.id) as 'X1',
      (SELECT count(*) FROM mitarbeiter mi
         INNER JOIN lieferung l  ON mi.id=l.mid WHERE m.id=mi.id) as 'X2'
   FROM mitarbeiter m
   ORDER by 1
   LIMIT 4;
```

Kontrollergebnis (Auszug)

```
+----+----+----+
| id | X1 | X2 |
+----+----+----+
| 1  | 11 | 5  |
| 2  | 8  | 6  |
| 3  | 10 | 4  |
| 4  | 0  | 4  |
+----+----+----+
```

2. Was beschreibt dabei (Spaltenüberschriften):

X1

X2

Informieren Sie sich, was ein Subselect in der `SELECT`-Klausel bewirkt und worauf ist dabei zu achten?

Notiz:

Optional für Experten:
Eine SQL-Abfrage mit einem Subselect in der `FROM`-Klausel könnte beispielsweise folgendermaßen aussehen:

```
USE gm3;
SELECT m.id, m.name
   FROM mitarbeiter m LEFT JOIN
      (SELECT mid FROM lieferung
         WHERE month(bestdatum)=6 AND year(bestdatum)=2021
      ) x ON m.id=x.mid
   WHERE x.mid IS NULL
   LIMIT 4;
```

Kontrollergebnis (Auszug)

```
+----+---------+
| id | name    |
+----+---------+
| 1  | Lorenz  |
| 2  | Ritter  |
| 4  | Richter |
| 5  | Wieland |
+----+---------+
```

3. Informieren Sie sich über Subselects in der `From`-Klausel und beschreiben Sie das Ergebnis obiger SQL-Abfrage in eigenen Worten und was ist dabei zu beachten?

Notiz:

Aufgaben zu Subselect AUF-08-4-4

1. Bestimmen Sie Id und Nachname derjenigen Mitarbeiter, die, bezogen auf das Gehalt aller Mitarbeiter, ein überdurchschnittliches Gehalt bekommen. Sortieren Sie dabei das Ergebnis nach dem Nachnamen aufsteigend.
Hinweis: Formulieren Sie ein analoges Subselect (wie im vorherigem Beispiel) in der `WHERE`-Klausel.

Kontrollergebnis (Auszug)

```
+----+----------+
| id | name     |
+----+----------+
| 8  | Hagen    |
| 19 | Hoffmann |
| 15 | Humpe    |
| 31 | Kaiser   |
+----+----------+
```

2. Bestimmen Sie die Id und Bezeichnung der billigsten Produkte.
Hinweis: Achten Sie darauf, dass es mehrere Produkte mit dem minimalen Preis geben kann.
Hinweis: Bestimmen Sie den minimalen Preis in einem Subselect und verwenden Sie dieses in der `WHERE`-Klausel.

Kontrollergebnis

```
+-----+--------------------------+
| id  | bez                      |
+-----+--------------------------+
| 121 | Mineralquelle Q4 Classic |
| 141 | Mineralquelle Q4 Medium  |
+-----+--------------------------+
```

3. Welche Kunden (Id, Name) haben im Monat Oktober im Jahr 2021 keine Bestellung aufgegeben?
Hinweis: Bestimmen Sie in einem Subselect alle Kunden, die im entsprechenden Monat eine Bestellung aufgegeben haben. Verwenden Sie dieses Subselect mit `NOT IN` in der `where`-Klausel.

Kontrollergebnis

```
+----+--------------------------------+
| id | name                           |
+----+--------------------------------+
| 22 | Restaurant im Schloss Biebrich |
| 23 | Königlicher Hirschgarten       |
+----+--------------------------------+
```

4. Ermitteln Sie diejenigen Lieferungen (Id und Lieferdatum), die bisher am meisten Produkte beinhaltet haben. Geben Sie ebenfalls diese Anzahl mit aus.
Hinweis: Achten Sie darauf, dass es mehrere solche Lieferungen geben kann.
Bestimmen Sie in einem Subselect die Anzahl der Produkte pro Lieferposition und bestimmen Sie die maximale Anzahl mithilfe des Tricks `ORDER BY ... LIMIT 1`. Verwenden Sie dieses Subselect in der `HAVING`-Klausel.

Kontrollergebnis

```
+----+------------+-----------------+
| id | liefdatum  | Anzahl Produkte |
+----+------------+-----------------+
| 40 | 2021-07-08 | 6               |
| 50 | 2021-08-17 | 6               |
+----+------------+-----------------+
```

5. Von welchen Lieferanten (Id, Name) hat der Mitarbeiter mit der ID 3 noch nie Lieferungen betreut?
Hinweis: Bestimmen Sie in einem Subselect alle Lieferanten, die der entsprechende Mitarbeiter betreut hat. Verwenden Sie dieses Subselect mit `NOT IN` in der `WHERE`-Klausel.

Kontrollergebnis

```
+----+--------------------------+
| id | name                     |
+----+--------------------------+
| 2  | Red Bull AG              |
| 3  | Coca Cola Deutschland AG |
+----+--------------------------+
```

6. Ermitteln Sie alle Kunden (Id, Name), die bisher ausschließlich Bier-Produkte gekauft haben.
Hinweis: Bestimmen Sie in einem Subselect alle Kunden, die bisher Produkte gekauft haben, bei denen die Produkttyp-Bezeichnung nicht „Bier“ enthält. Verwenden Sie dieses Subselect mit `NOT IN` in der `WHERE`-Klausel.

Kontrollergebnis

```
+----+--------------------------+
| id | name                     |
+----+--------------------------+
| 23 | Königlicher Hirschgarten |
+----+--------------------------+
```

7. Welche Mitarbeiter (Id, Name) haben schon von allen Lieferanten Lieferungen betreut?
Hinweis: Bestimmen Sie in einem Subselect die Anzahl aller Lieferanten und verwenden Sie das Ergebnis dieses Subselects in der `HAVING`-Klausel.

Kontrollergebnis

```
+----+------+
| id | name |
+----+------+
| 12 | Kamp |
+----+------+
```

8. Optional: für Experten

In dieser Aufgabe soll für jeden Monat (in deutscher Schreibweise) im Jahr 2021, in dem Produkte verkauft wurden, das Produkt (Id, Bezeichnung) ermittelt werden, das am meisten verkauft wurde. Das Ergebnis soll dabei nach dem Monat aufsteigend sortiert sein.

Kontrollergebnis

```
+-----------+-----+----------------------------------------+-------+
| Monat     | pid | bez                                    | menge |
+-----------+-----+----------------------------------------+-------+
| Juni      | 119 | Gerolsteiner Classic                   | 48    |
| Juli      | 123 | Odenwald Quelle Classic                | 90    |
| August    | 221 | Odenwald-Quelle Apfel-Johannisbeer Pet | 116   |
| September | 31  | Bitburger Pils Stubbi                  | 91    |
| Oktober   | 220 | Odenwald-Quelle Apfel-Kirsch Pet       | 205   |
| November  | 53  | Schöfferhofer Weizenbier Hefe dunkel   | 87    |
+-----------+-----+----------------------------------------+-------+
```

8a. Vorgehensweise:

Erstellen Sie zur besseren Übersichtlichkeit erst eine entsprechende View, in der für jeden Monat die Menge des am meist verkauften Produkts erfasst wird. Erfassen Sie in dieser View den Monat als Zahl.

Kontrollergebnis

```
+-------+-------+
| monat | menge |
+-------+-------+
| 6     | 48    |
| 7     | 90    |
| 8     | 116   |
| 9     | 91    |
| 10    | 205   |
| 11    | 87    |
+-------+-------+
```

8b. Verwenden Sie anschließend diese View in einer zusätzlichen SQL-Abfrage, um das gewünschte Ergebnis incl. Produkt-ID zu bekommen.

Notiz: