Aufgabe 2: SQL

Gegeben sind folgende Relationen aus einem Kundenverwaltungssystem:

```
Kunde : {[ ID, Vorname, Nachname, PLZ ]}
Produkt : {[ GTIN, Bezeichnung, Bruttopreis, MWStSatz) ]}
Kauf : {[ ID[Kunde], GTIN[Produkt], Datum, Menge ]}
```

```sql
CREATE TABLE Kunde(
        ID INTEGER PRIMARY KEY,
        Vorname VARCHAR(30),
        Nachname VARCHAR(30),
        PLZ INTEGER
);

CREATE TABLE Produkt(
        GTIN INTEGER PRIMARY KEY,
        Bezeichnung VARCHAR(40),
        Bruttopreis NUMERIC(7,2),
        MWStSatz INTEGER
);

CREATE TABLE Kauf(
  ID INTEGER REFERENCES Kunde(ID),
  GTIN INTEGER REFERENCES Produkt(GTIN),
  Datum DATE,
  Menge INTEGER,
  PRIMARY KEY (ID, GTIN, Datum)
);

INSERT INTO Kunde VALUES
  (1, 'Max', 'Mustermann',91052),
  (2, 'Erika', 'Musterfrau',91052),
  (3, 'Max', 'Meyer',91058),
  (4, 'Hans', 'Schmidt',91054),
  (5, 'Eva', 'Müller',91056),
  (6, 'Hanna', 'Winter',20251),
  (7, 'Bert', 'Sommer',20251),
  (8, 'Jakob', 'Sommer',20251);

INSERT INTO Produkt VALUES
  (123, 'Buch', 12.30,19),
  (124, 'Kaffee', 4.30,7),
  (125, 'Pullover', 36.40,19),
  (113, 'Heft', 2.30,19),
  (023, 'Honig', 3.20,7),
  (155, 'T-Shirt', 19.30,19),
  (189, 'Nudeln', 1.30,7),
  (004, 'Sonnenbrille', 40.60,19),
  (324, 'Hammer', 22.80,19),
  (112, 'Topf', 50.20,19),
  (453, 'Klopapier', 3.30,7),
  (765, 'Duschgel', 1.89,19),
  (889, 'Deko', 5.89,19);

INSERT INTO Kauf VALUES
  (1, 123, '2019-04-11', 1),
  (1, 124, '2019-04-11', 21),
  (1, 125, '2019-04-11', 1),
  (1, 765, '2019-04-11', 4),
  (1, 453, '2019-04-11', 1),
  (1, 324, '2019-04-11', 3),
```

```
55    (1, 113, '2019-04-11', 2),
56    (1, 023, '2019-04-11', 1),
57    (1, 189, '2019-04-11', 1),
58    (1, 112, '2019-04-11', 7),
59    (1, 155, '2019-04-11', 7),
60    (1, 004, '2019-05-11', 6),
61    (7, 112, '2019-04-11', 7),
62    (5, 112, '2019-04-11', 7),
63    (8, 112, '2019-06-23', 5),
64    (8, 112, '2019-04-12', 3),
65    (2, 112, '2019-04-23', 1),
66    (2, 112, '2019-08-11', 8),
67    (4, 112, '2019-10-10', 2),
68    (2, 453, '2019-09-24', 4),
69    (4, 004, '2019-07-30', 9);
```

Verwenden Sie im Folgenden nur Standard-SQL und keine produktspezifischen Erweiterungen. Sie dürfen bei Bedarf Views anlegen. Geben Sie einen Datensatz, also eine Entity, nicht mehrfach aus.

(a) Schreiben Sie eine SQL-Anweisung, die die Tabelle *„Kauf“* anlegt. Gehen Sie davon aus, dass die Tabellen *„Kunde“* und *„Produkt“* bereits existieren.

```sql
CREATE TABLE IF NOT EXISTS Kauf (
  ID INTEGER REFERENCES Kunde(ID),
  GTIN INTEGER REFERENCES Produkt(GTIN),
  Datum DATE,
  Menge INTEGER,
  PRIMARY KEY (ID, GTIN, Datum)
);
```

(b) Schreiben Sie eine SQL-Anweisung, die *Vorname* und *Nachname* aller *Kunden* mit der *Postleitzahl 20251* ausgibt, absteigend sortiert nach *Nachname* und bei gleichen *Nachnamen*, absteigend nach *Vorname*.

```sql
SELECT Vorname, Nachname
FROM Kunde
WHERE PLZ = 20251
ORDER BY Nachname DESC, Vorname DESC;
```

(c) Schreiben Sie eine SQL-Anweisung, die zu jedem Einkauf mit mehr als 10 unterschiedlichen Produkten den *Nachnamen* des *Kunden* und den *Bruttogesamtpreis* des Einkaufs ausgibt. Ein Einkauf ist definiert als Menge aller Produkte, die ein bestimmter Kunde an einem bestimmten Datum kauft.

```sql
SELECT Nachname, SUM(Bruttopreis * Menge)
FROM Kunde k, Produkt p, Kauf x
WHERE k.ID = x.ID AND p.GTIN = x.GTIN
GROUP BY Datum, Nachname, k.ID
HAVING COUNT (*) > 10;
```

(d) Schreiben Sie eine SQL-Anweisung, die die *GTINs* aller Produkte ausgibt, die an mindestens einen in der Datenbank enthaltenen PLZ-Bereich noch nie verkauft worden sind. Als in der Datenbank enthaltener PLZ-Bereich gelten alle in der Tabelle *„Kunde“* enthaltenen PLZs. Ein Produkt gilt als an einen PLZ-Bereich verkauft, sobald es von mindestens einem Kunden

aus diesem PLZ-Bereich gekauft wurde. Produkte, die bisher noch gar nicht verkauft worden sind, müssen nicht berücksichtigt werden.

```
SELECT PLZ
FROM Kunde k
GROUP BY PLZ
HAVING NOT EXISTS (
  SELECT DISTINCT Kauf.GTIN
  FROM Kauf, Kunde
  WHERE Kauf.ID = Kunde.ID AND Kunde.PLZ = k.PLZ
);
```

in Mysql gibt es kein EXCEPT

```
SELECT GTIN FROM
(SELECT GTIN, PLZ
FROM Kunde, Produkt)
EXCEPT
(SELECT DISTINCT x.GTIN, k.PLZ
FROM Kunde k, Kauf x
WHERE k.ID = x.ID)
```

```
WITH tmp AS (
  SELECT x.GTIN, k.PLZ
  FROM Kunde k, Kauf x
  WHERE x.ID = k.ID
  GROUP BY x.GTIN, k.PLZ
)

SELECT GTIN
FROM tmp
WHERE EXISTS
(SELECT Kunde.PLZ
FROM Kunde LEFT OUTER JOIN tmp
ON Kunde.PLZ = tmp.PLZ
WHERE tmp.PLZ IS NULL);
```

oder eleganter

```
SELECT DISTINCT GTIN
FROM
(SELECT GTIN, PLZ
FROM Kunde, Produkt)
EXCEPT
(SELECT x.GTIN, k.PLZ
FROM Kunde k, Kauf x
WHERE x.ID = k.ID
GROUP BY x.GTIN, k.PLZ)
```

Statt WITH koennen auch VIEWs erstellt werden ! Eine Konstruktion mit NOT IN sollte auch moeglich sein

(e) Schreiben Sie eine SQL-Anweisung, die die Top-Ten der am meisten verkauften Produkte ausgibt. Ausgegeben werden sollen der Rang (1 bis 10) und die Bezeichnung des Produkts. Gehen Sie davon aus, dass es keine zwei Produkte mit gleicher Verkaufszahl gibt und verwenden Sie keine produktspezifischen Anweisungen wie beispielsweise `ROWNUM`, `TOP` oder `LIMIT`.

```sql
WITH Gesamtverkauf AS
(SELECT k.GTIN, Bezeichnung, SUM (Menge) AS Gesamtmenge)
FROM Produkt p, Kauf k
WHERE p.GTIN = k.GTIN
GROUP BY k.GTIN, Bezeichnung)

SELECT g1.Bezeichnung, COUNT (*) AS Rang
FROM Gesamtverkauf g1, Gesamtverkauf g2
WHERE g1.Gesamtmenge <= g2.Gesamtmenge
GROUP BY g1.GTIN, g1.Bezeichnung
HAVING COUNT (*) <=10
ORDER BY Rang;
```

(f) Schreiben Sie eine SQL-Anweisung, die alle Produkte löscht, die noch nie gekauft wurden.

```sql
DELETE FROM Produkt
WHERE GTIN NOT IN
(
  SELECT DISTINCT GTIN
  FROM Kauf
);
```