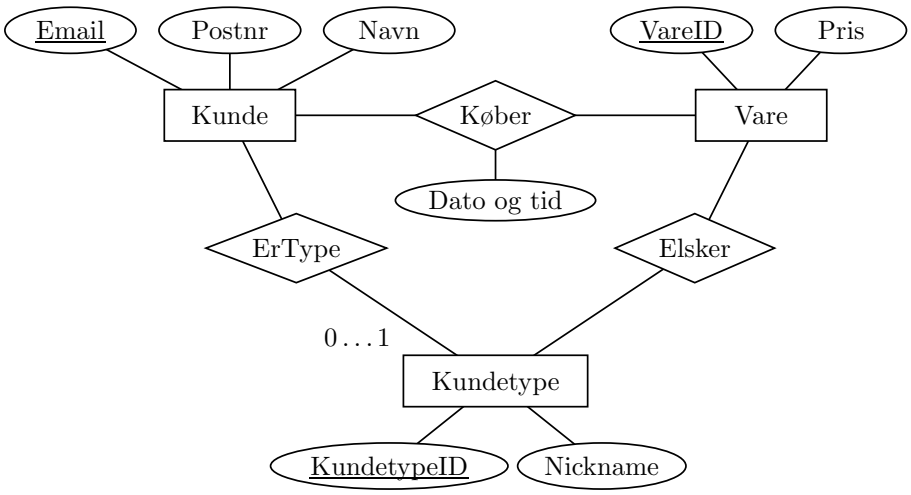
Her er et uddrag af et ER-diagram for en butikskædes database:



Database-designeren har valgt at have tre entities med de angivne attributter. Keys for entities er angivet med understregning. Vedkommende har også valgt, at en instans af Kunde kan være med i (være i relationship til) højst én Kundetype. Bortset fra dette er alle relationships mange-til-mange (dvs. uden cardinality constraints).

Disse entities og relationships kan oversættes til flg. relationer i den relationelle model:

Kunde(Email: string, Postnr: integer, Navn: string)
Vare(VareID: integer, Pris: float)
Kundetype(KundetypeID: integer, Nickname: string)
Køber(Email: string, VareID: integer, DatoOgTid: timestamp)
ErType(Email: string, KundetypeID: integer)
Elsker(VareID: integer, KundetypeID: integer)

Bemærk, at en relation (dvs. en tabel) R, som repræsenterer et relationship, ofte får en key, der består af begge keys fra de to entities, som dette rela-

tionship forbinder. Disse to keys vil normalt hver blive erklæret som foreign keys i R.

Hvis man vil modellere den angivne cardinality constraint (ikke pensum), kan man f.eks. gøre det ved at justere på key i relationen ErType:

> ErType(Email: string, KundetypeID: integer)

I SQL kan man oprette tilsvarende schema'er via f.eks. disse kommandoer:

> CREATE TABLE Kunde (Email CHAR(50) PRIMARY KEY, Postnr INTEGER, Navn CHAR(50));
>
> CREATE TABLE Vare (VareID INTEGER PRIMARY KEY, Pris REAL);
>
> CREATE TABLE Kundetype (KundetypeID INTEGER PRIMARY KEY, Nickname: CHAR(40));
>
> CREATE TABLE Køber (Email CHAR(50) REFERENCES Kunde, VareID INTEGER REFERENCES Vare, DatoOgTid TIMESTAMP, PRIMARY KEY (Email, VareID, DatoOgTid));
> [Vi vælger, at en køber kan købe samme vare flere gange, sætter derfor key lig alle attributter.]
>
> CREATE TABLE ErType (Email CHAR(50) PRIMARY KEY REFERENCES Kunde, KundetypeID INTEGER REFERENCES Kundetype);
>
> CREATE TABLE Elsker (VareID INTEGER REFERENCES Vare, KundetypeID INTEGER REFERENCES Kundetype, PRIMARY KEY (VareID, Kundetype));

En søgning efter "email på alle kunder, som bor i postnummer 5000" kan beskrives således i SQL:

```
SELECT Email
FROM Kunde
WHERE Postnr = 5000;
```

En søgning (query) efter "Vare-ID for alle varer, som er købt af kunder af type 4 efter 1. januar 2023" kan beskrives således i SQL:

```
SELECT V.VareID AS Vare-ID
FROM Vare V, Køber KV, Kunde K, ErType ET
WHERE V.VareID = KV.VareID AND
        K.Email = KV.Email AND
        K.Email = ET.Email AND
        ET.KundetypeID = 4 AND
        KV.DatoOgTid >= '2023-01-01';
```