-

DATABAS FÖR

RASTLÖS FÖRLAG

AV JENNIFER LEE

-

INNEHÅLL

KONCEPTUELL MODELL

1. BESKRIVNING

Vi ska utveckla ett webbaserat e-shop system för försäljning av förlagets produkter som i dagsläget består av böcker eller vinyler.

Produkterna är indelade i två huvudkategorier *böcker* och *vinyler*. Varje produkt kan tillhöra en kategori. Antingen är det en bok eller en vinyl. De ska kunna presenteras med bild, titel, författare, kort info, utgivningsår och pris. Varje produkt ska även lagra information såsom produktnamn (unikt), mått och vikt.

Kunder ska kunna lägga en order på produkter. Varje kund har ett unikt kundNr och kontaktuppgifter, faktureringsuppgifter. Varje order har ett unikt orderNr som innehåller produktnamn dess antal och pris.

Plocklistor skapas utifrån ordern där information om var produkterna i lagerregistret finns.

När ordern är redo för leverens skapas en faktura som är en kopia av ordern men som även inkluderar en summering av ordern samt moms, unikt faktureringsNr, faktureringsdatum och förfallodatum.

I längre loppet finns önskemål om:

- Räkna ut olika leverenskostnader utifrån sammanlagda mått och volym på ordern.
- Handla i shopen utan att vara registrerad

Men detta tas inte hänsyn till i denna databas.

2. DATABASENS ENTITETER

Utifrån databasens målbeskrivning "*Vi ska utveckla ett webbaserat e-shop system för försäljning av förlagets produkter som i dagsläget består av böcker och vinyler*."

Beskrivningen har många substantiv, produktID, kundID dessa är dock innehåll under respektive rubrik. En produkt kan vara en bok eller en vinyl. En kund har ett kundID. En faktura har en produkt och en kund. Vilket leder oss till 3.

- Produkt
- Produktkategori
- Kund
- Order
- Plocklista
- Faktura

3. RELATIONER

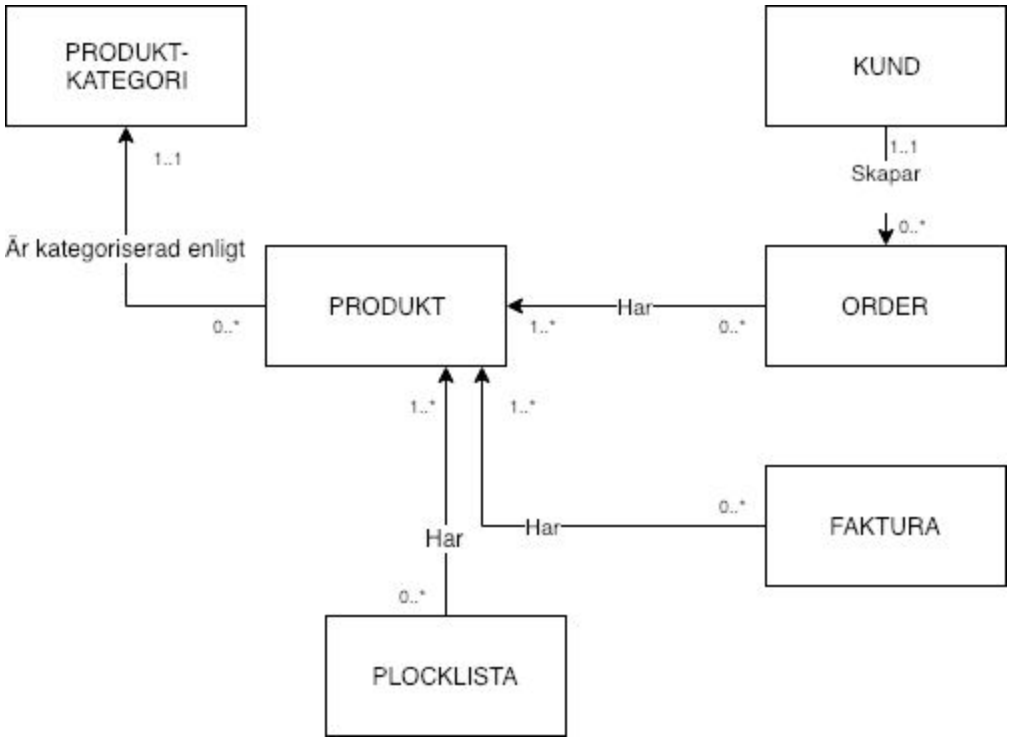
Verben som binder ihop databasens entiteter i punkt 2.

	Produkt	**Produktkat**	**Kund**	**Order**	**Plocklista**	**Faktura**
Produkt		Kategoriseras enligt				
Produktkat						
Kund				Skapar		

				en		
Order	har					
Plocklista	har					
Faktura	har					

- Alla produkt kategoriseras efter produktkategorisering
- Varje kund skapar en order
- Varje order innehåller produkt
- Varje plocklista innehåller produkt
- Varje faktura innehåller produkt

4. ER-DIAGRAM



5. ER-DIAGRAM MED KARDINALITET

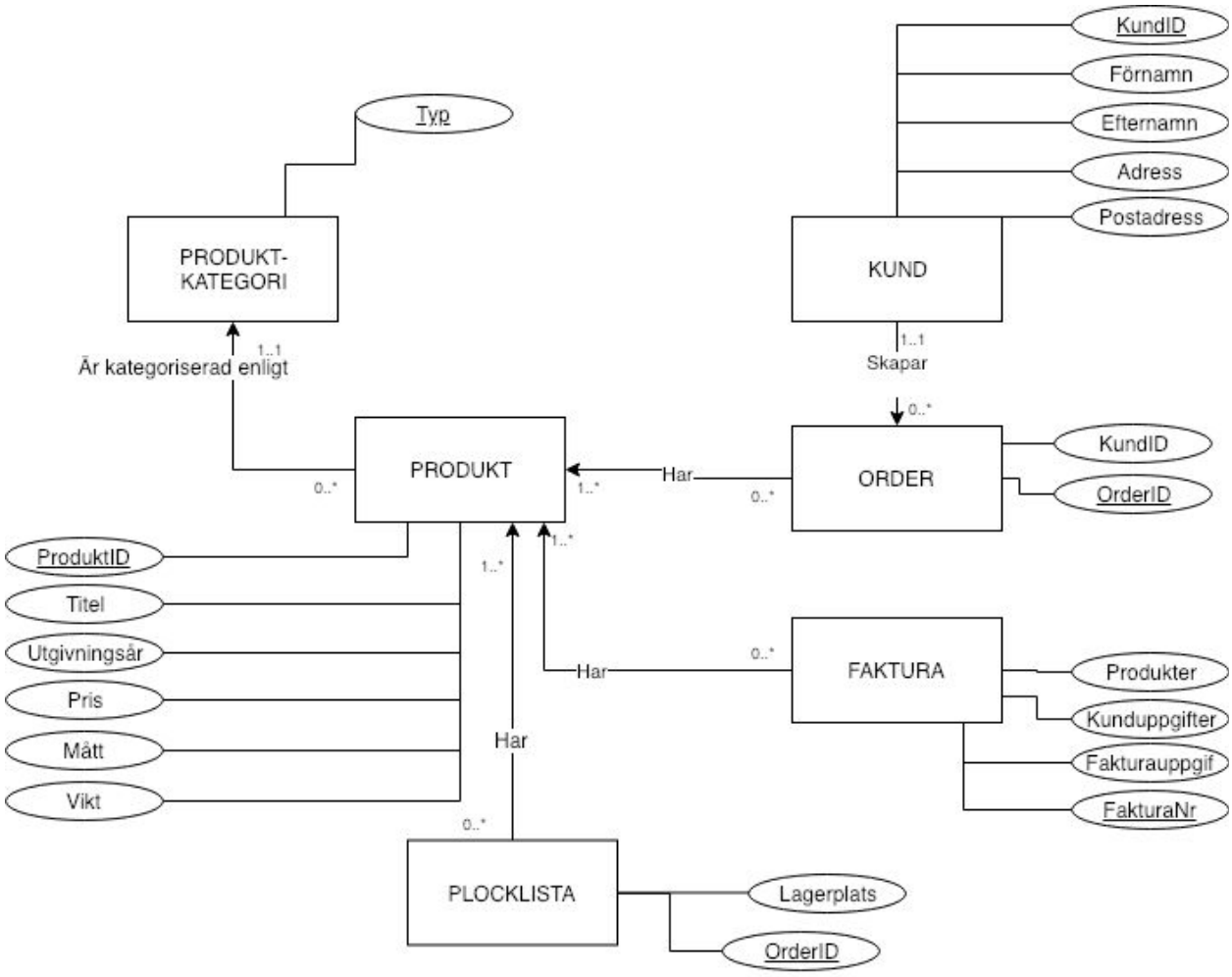
Hur entiteterna förhåller sig till varandra. Ses i bilden ovan och förklaras i text här.

Relation	**Typ**	
Produkt - Produktkategori	1:1	
Produkt - Plocklista	0:M	
Produkt - faktura	0:M	

Produkt - order	0:M	
Kund - order	0:M	

- En produkt kan bara tillhöra en produktkategori (se målet), en produktkategori kan innehålla 0 till massor av produkter.
- Varje produkt kan ligga i 0 eller massor av plocklistor, plocklistan existerar endast om det finns produkter i den.
- En produkt existerar även om de inte finns i någon faktura och kan finnas i flera fakturor, men fakturan existerar endast om det finns en produkt.
- Produkter kan ligga i 0 till flera ordrar men ordrar finns bara om det ligger en produkt i dem.
- Varje kund kan skapa 0 till massor av ordrar, en order kan bara vara kopplad till en kund.

6. ATTRIBUT OCH KANDIDATNYCKLAR



- Produkt (ProduktID, Titel, Utgivningsår, Pris, Mått, Vikt)
- Produktkategori(Typ)
- Kund(KundID, Förnamn, Efternamn, Adress, Postadress)
- Order (KundID, OrderID, Produkter)
- Faktura (FakturaNR, Fakturauppgifter, Kunduppgifter, Produkter)

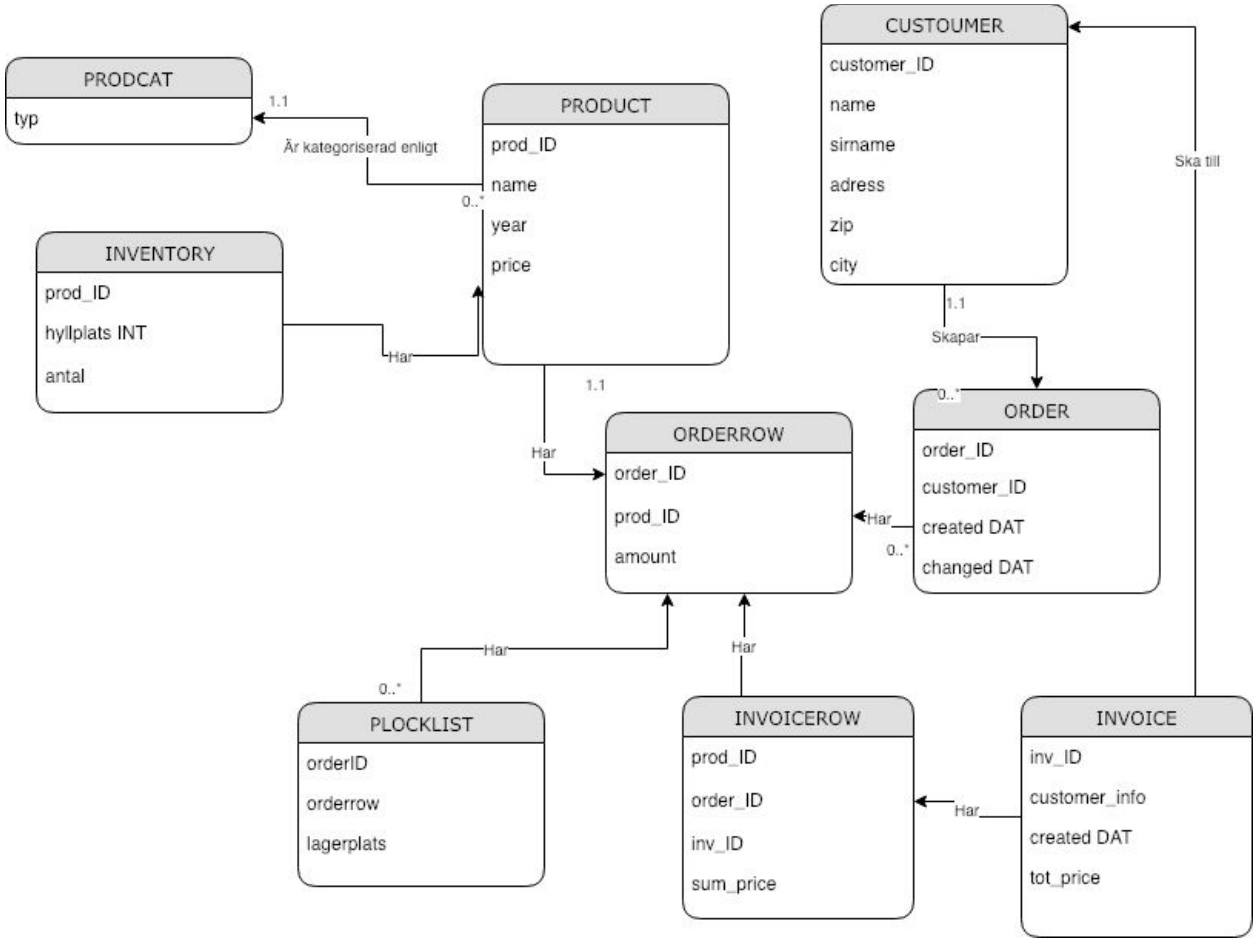
LOGISK MODELL

Utökande av den konceptuella modellen till en logisk modell.
Vi tar bland annat bort N:M-sambandstyp.

7. ENLIGT RELATIONSMODELLEN

Uppbrytning av den logiska modellen med kopplingstabeller.
En del justeringar har skett sen den konceptuella modellen.

- Customer (customer_ID) skapar en Order (order_ID)
- Order innehåller en eller flera OrderRow (prod_ID, amount)
- OrderRow innehåller ett antal av en viss Product.
- Invoice (inv_ID) en eller flera InvoiceRow (inv_ID) som är en kopia av OrderRow men som har en summering av priset.


PRODCAT
typ
1.1
Är kategoriserad enligt
PRODUCT
prod_ID
name
year
price
0..*
CUSTOUMER
customer_ID
name
sirname
adress
zip
city
Ska till
INVENTORY
prod_ID
hyllplats INT
antal
Har
1.1
1.1
Skapar
0..*
ORDER
order_ID
customer_ID
created DAT
changed DAT
Har
ORDERROW
order_ID
prod_ID
amount
Har
0..*
Har
Har
0..*
PLOCKLIST
orderID
orderrow
lagerplats
INVOICEROW
prod_ID
order_ID
inv_ID
sum_price
Har
INVOICE
inv_ID
customer_info
created DAT
tot_price

8. KEYS

Orangea och understrukna är primära nycklar och främmande nycklar är markerade med en #.

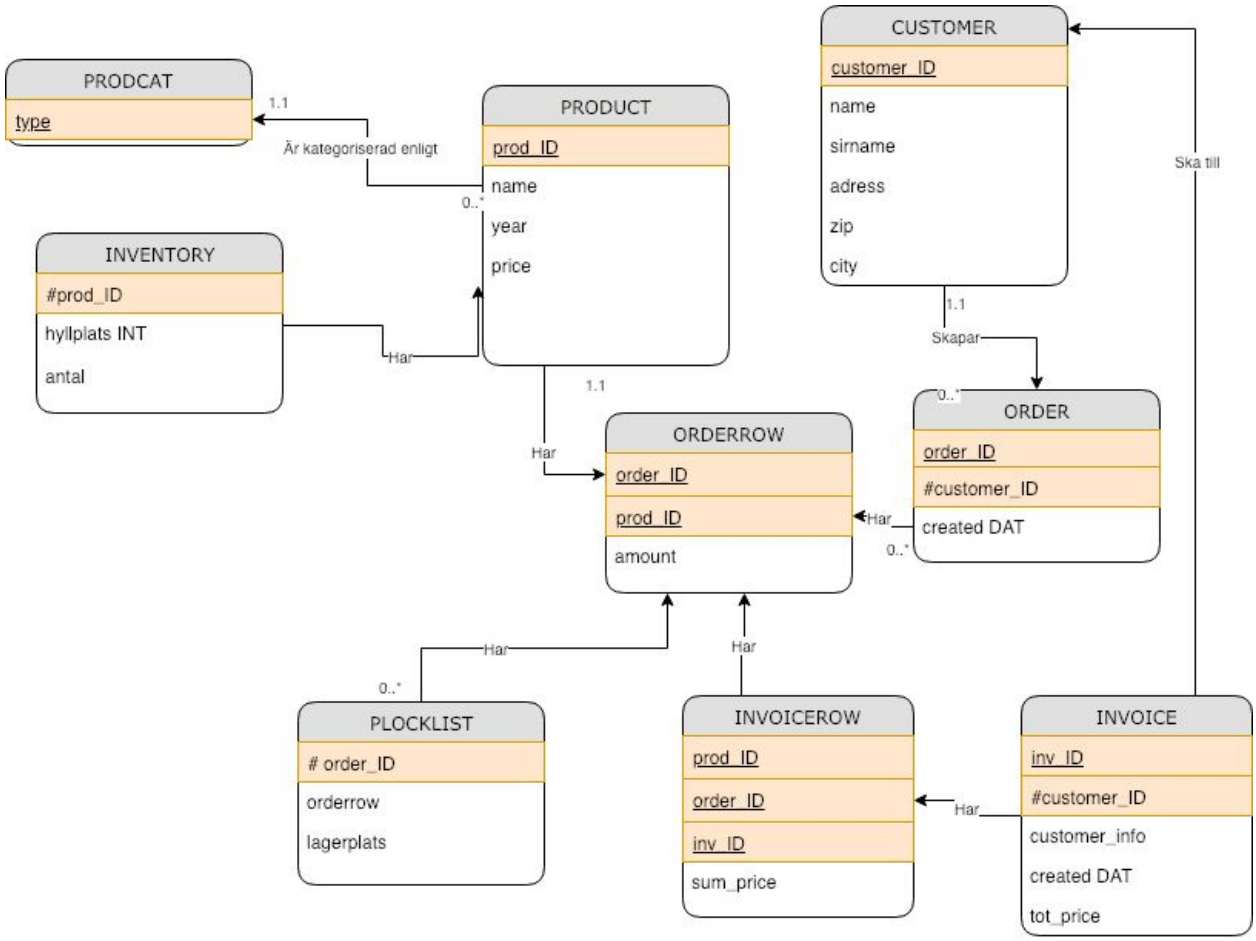
Customer (customer_ID) , skapar en Order. Order (order_ID) är också beroende av customer (#customer_ID). Varje order innehåller en Orderrow (order_ID, prod_ID).

Alla customers ska också få en Invoice (inv_ID, #customer_ID) och varje Invoice innehåller en eller flera InvoiceRow(inv_ID, order_ID, prod_ID).

Plocklist (#ord_ID) är beroende av det finns en order.

Inventory (#prod_ID) hämtar också info från produkterna.

ProduktionsCategory (type).



9. FYSISK MODELL

Jag börjar med att skapa alla mina tabeller och koppla ihop nycklarna.
När jag jobbade med denna insåg jag att att jag missat ett par nycklar från tidigare skiss.
OrderRow tänkte jag bara skulle ha främmande nycklar kopplade till order och produkt men insåg att jag kanske vill kunna lägga dessa i en viss ordning eller kolla på en specifik rad i ordern och då behövde jag även ett id för denna tabell.

För att skriva själva koden har jag ändå utgått från den logiska modellen och parallellkollat mot Mos video "Föreläsning 2017-04-12: webapp + oophp kmom03".
Vilket också fick mig att inse att jag behövde vissa tabeller som ej är angivna tidigare t.ex. Prod2Cat.

Ordningen är lite utifrån ett flödelseschema.
Vad ska jag göra, jo jag ska göra en databas för en e-shop.
Börjar med Producter som ska kategoriseras, och som sedan ska till lagret.
Därefter har jag min kund, kunden skapar en order som innehåller orderrader och genererar plocklista.
Till sist ska jag ha en faktura som ska ges till kunden.

Allra sist gick jag till toppen av min kod för att skriva in mina DROP IF EXISTS av samtliga tabeller.

Jag testkörde själva koden och fick då ändra ordningen på mina drop.

Koden finns på sista sidan.

10. API

Databasen ska stödja följande

1. Lägga till en ny Produkt.
2. Redigera och radera befintlig Produkt.
3. Visa information samtliga Produkter.
4. Visa information efter Produktkategori.
5. Skapa, redigera och radera en Kund.
6. Skapa, redigera och radera en Order.
7. Skapa och radera en Plocklista.
8. Skapa, redigera och radera en Faktura.

SQL KODEN

```
-- drop all tables if exists
DROP TABLE IF EXISTS Prod2Cat;
DROP TABLE IF EXISTS ProductCategory;
DROP TABLE IF EXISTS Inventory;
DROP TABLE IF EXISTS OrderRow;
DROP TABLE IF EXISTS Plocklist;


DROP TABLE IF EXISTS InvoiceRow;
DROP TABLE IF EXISTS Invoice;


DROP TABLE IF EXISTS ProdOrder;


DROP TABLE IF EXISTS Customer;
DROP TABLE IF EXISTS Product;


-- PRODUCT(id)
--
```

```
CREATE TABLE Product
(
    id INT NOT NULL AUTO_INCREMENT,
  title CHAR(100),
  info VARCHAR(300),
  price FLOAT,

  PRIMARY KEY (id)
);


-- PRODUCTCATEGORY(prod_type)
--
CREATE TABLE ProductCategory
(
    id INT NOT NULL AUTO_INCREMENT,
  cat CHAR(50),

  PRIMARY KEY (id)
);

-- PROD2CAT(id, #prod_id, #cat_id)
--
CREATE TABLE Prod2Cat
(
    id INT NOT NULL AUTO_INCREMENT,
  prod_id INT,
  cat_id INT,

  PRIMARY KEY (id),
  FOREIGN KEY (prod_id) REFERENCES Product(id),
  FOREIGN KEY (cat_id) REFERENCES ProductCategory(id)
);

-- INVENTORY (id, #prod_type)
--
CREATE TABLE Inventory
(
    id INT NOT NULL AUTO_INCREMENT,
  prod_id INT,
  shelf INT,
  items INT,

  PRIMARY KEY (id),
  FOREIGN KEY (prod_id) REFERENCES Product(id)
);

-- CUSTOMER (id)
--
CREATE TABLE Customer
```

```
(
    id INT AUTO_INCREMENT,
  mail VARCHAR(50),
  fname CHAR(100),
  sirname CHAR(100),
  adress VARCHAR(300),
  zip INT(6),
  city CHAR(30),

  PRIMARY KEY (id)
);

-- PRODORDER (id, #customer_ID)
-- could not use ORder... stupid

CREATE TABLE ProdOrder
(
    id INT NOT NULL AUTO_INCREMENT,
  customer_id INT,
  created DATETIME DEFAULT CURRENT_TIMESTAMP,
  updated DATETIME DEFAULT NULL ON UPDATE CURRENT_TIMESTAMP,
  deleted DATETIME DEFAULT NULL,
  delivery DATETIME DEFAULT NULL,

  PRIMARY KEY (id),
  FOREIGN KEY (customer_id) REFERENCES Customer(id)
);

-- ORDERROW(id, #order_ID, #prod_ID)
--
CREATE TABLE OrderRow
(
    id INT AUTO_INCREMENT,
  order_id INT,
  prod_id INT,
  amount INT,

  PRIMARY KEY(id),
  FOREIGN KEY (order_id) REFERENCES ProdOrder(id),
  FOREIGN KEY (prod_id) REFERENCES Product(id)
);

-- PLOCKLIST(id, #order_id, #prod_id)
-- varje plocklista innehåller flera orderrows - som är kopplade till själva ProdOrder.

CREATE TABLE PlockList
(
    id INT AUTO_INCREMENT,
  order_id INT,
  prod_id INT,
```

```
    items INT,

    PRIMARY KEY (id),
    FOREIGN KEY (order_id) REFERENCES ProdOrder(id),
    FOREIGN KEY (prod_id) REFERENCES Product(id)
);

-- INVOICE (inv_ID, #customer_id)
--
CREATE TABLE Invoice
(
        id INT NOT NULL AUTO_INCREMENT,
    customer_id INT,
    created DATETIME DEFAULT CURRENT_TIMESTAMP,

    PRIMARY KEY (id),
    FOREIGN KEY (customer_id) REFERENCES Customer(id)
);

-- INVOICEROW (id, #prod_id, #order_id, #inv_id)
CREATE TABLE InvoiceRow
(
        id INT AUTO_INCREMENT,
    prod_id INT,
    order_id INT,
    inv_id INT,
    total FLOAT,

    PRIMARY KEY (id),
    FOREIGN KEY (prod_id) REFERENCES Product(id),
    FOREIGN KEY (order_id) REFERENCES ProdOrder(id),
    FOREIGN KEY (inv_id) REFERENCES Invoice(id)
);
```