Kmom04: Konceptuell modellering

DV1606 V21 Databasteknologier för webben

Författare: Nenad Cuturic, necu20

Innehållsförteckning

1 Konceptuell modell: BackStar online shopping

1.1 Beskrivning

Vi skall utveckla ett webbaserat system för onlineförsäljning av kaffe, te, kaffemuggar och liknande produkter. Produkterna skall presenteras med information om och bild av produkten.

Databasen som skall användas för lagring av information behöver hantera ett kundregister (kunder med kontaktdetaljer), ett produktregister (produkter med produktkod, namn, kort beskrivning, bild och pris) där varje produkt finns i en eller flera produktkategorier.

Databasen behöver innehålla ett lager där man ser hur många av varje produkt som finns i lagret och en notering om var produkten ligger i lagret (vilken hylla). En och samma produkt kan vara utspridd över olika hyllor i lagret.

När kunden beställer en produkt leder det till att det skapas en order som innehåller kundens detaljer tillsammans med vilka produkter som beställts och dess beställda antal.

Utifrån ordern skapas en plocklista som kan skickas till lagret för leverans. Plocklistan innehåller samma information som ordern, men med tillägget att varje produktrad mappas mot en lagerhylla så att lagerpersonalen kan se vilken hylla de kan hämta produkten på. Om produkten inte finns på lager skall en restnotering skapas med listan över saknade produkter. Plocklistan skall då innehålla enbart produkter på lager och samma gäller fakturan. När produkterna har kommit in på lagret betas restnoteringslistan av i tur och ordning, ny plocklista skapas och därefter ytterligare en faktura med restnoterade produkter.

När leveransen är packad så bifogas en faktura som har samma innehåll som ordern men nu med priset per produktrad och det summerade priset.

Det skall finnas en logg där man kan se viktiga händelser i systemet, vad hände, när hände det. Det kan till exempel vara när order/faktura skapades eller raderades.

1.2 Entitetslista

- Produkt
- Produktkategori
- Kund
- Lager
- Hylla
- Order
- Plocklista
- Restnotering
- Faktura
- Logg

1.3 Relationer

1. Alla produkter är kategoriserade enligt produktkategori.
2. Alla produkter är lagrade på Lagret.
3. Kunder skapar en order.
4. Varje order innehåller produkter.
5. Varje plocklista innehåller antingen samma produkter som order eller färre i fall produkten inte finns på lager. Dessutom innehåller plocklistan hyllplats för produkter från ordern.
6. Varje faktura innehåller produkter som finns på plocklistan.
7. Varje händelse (order, plocklista, restnotering, faktura) sparas i logg.
8. Varje order följs av att en plocklista skapas.
9. Order kan också följas av att restorder skapas i fall samtliga produkter i ordern inte finns på lager.
10. Plocklistan följs av att en faktura skapas.
11. När restnotering åtgärdats följs det av att en ny plocklista skapas baserad på restnoteringen.

Entiteter	**Produkt**	**Produktkategori**	**Kund**	**Hylla**	**Order**	**Plocklista**	**Restnotering**	**Faktura**	**Logg**
Produkt		är kategoriserad enligt		är lagrade på					
Produktkategori									
Kund					skapar/har				
Hylla									
Order	innehåller					följs av	kan följas av	har	leder till
Plocklista	innehåller			innehåller				följs av	leder till
Restnotering	innehåller					följs av			leder till
Faktura	innehåller		skickas till						leder till
Logg									

1.4 Enkelt ER-diagram med eniteter och relationer

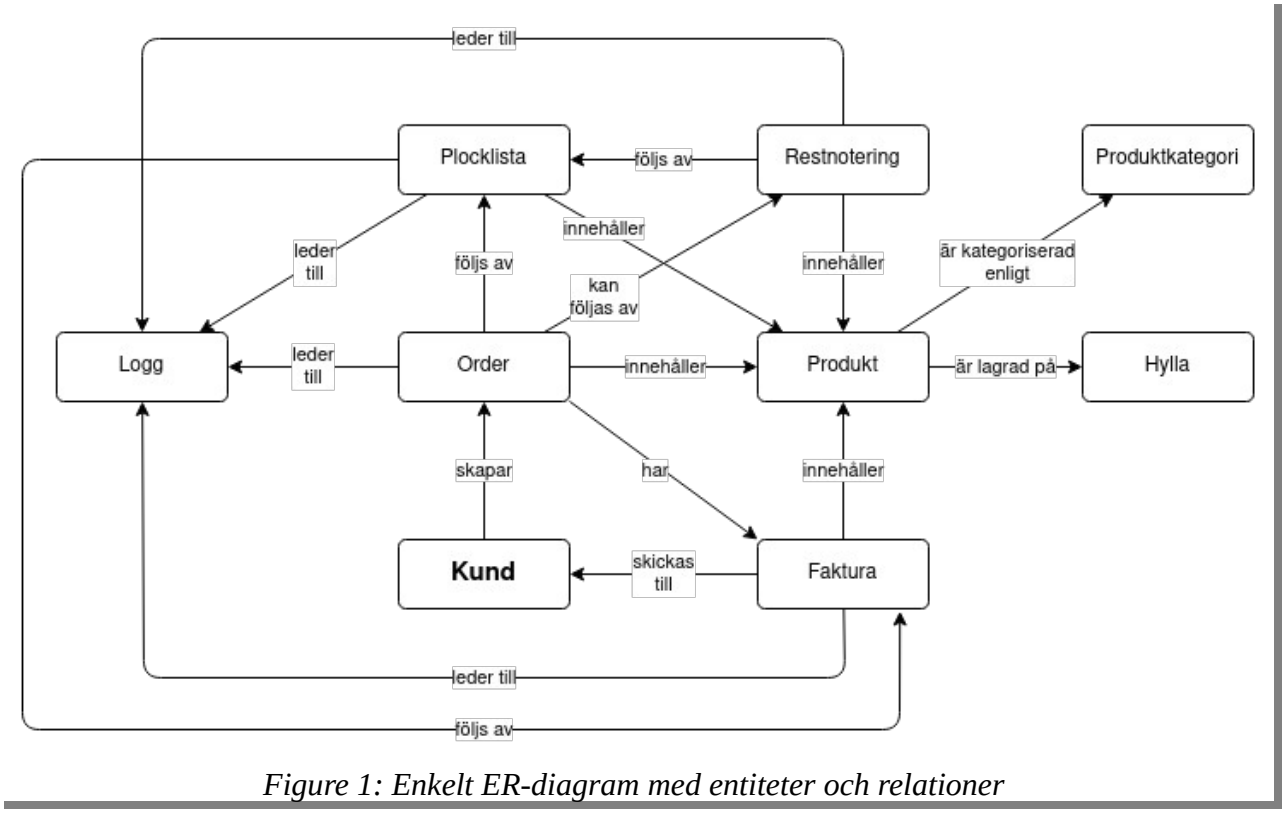

Figure 1: Enkelt ER-diagram med entiteter och relationer

1.5 Utökat ER-diagram med kardinalitet

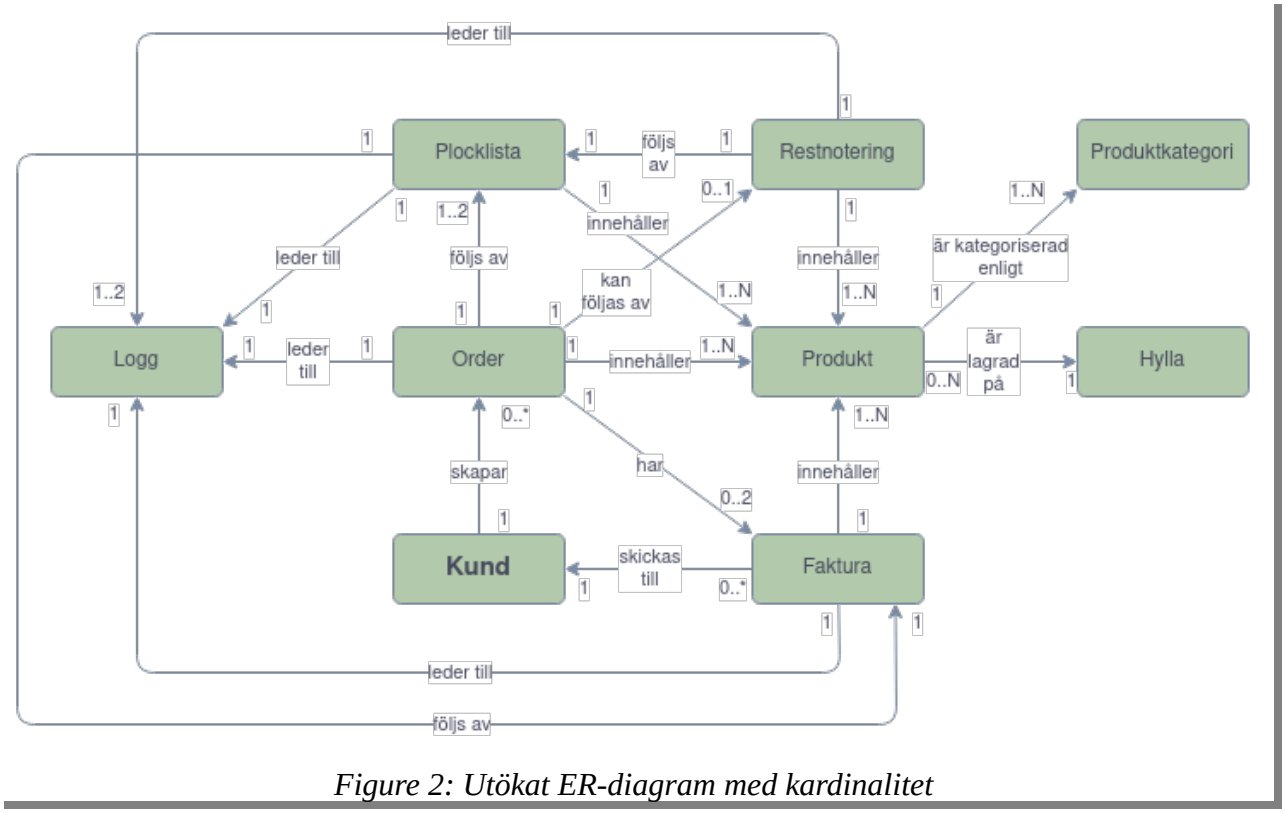

Figure 2: Utökat ER-diagram med kardinalitet

- En kund kan ha ingen eller fler ordrar men en order tillhör bara en kund
- En order kan ha 0 (innan varorna skickas), 1 alternativt 2 faktoror (beroende av om ordern blev uppded på grund av restnotering)
- En order följs av en plocklista men en eller två plocklistor kan tillhöra samma order (i fall en plocklista kommer efter restnoteringen)
- En order kan följas av en restnotering och en restnotering tillhör en order
- Restnotering följs av en plocklista, en plocklista kan vara assosiread till en restnotering
- En plocklista följs av en faktura, en fakturan kan vara assosiread till en plocklista
- En order, en plocklista, en restnotering och en faktura kan ha en till fler produkter
- En produkt kan tillhöra en till fler produktkategorier, en kategori kan innehålla 0 till fler produkter
- En hylla kan innehålla ingen eller fler produkter, en produkt kan finnas bara på en hylla
- En plocklista, en restnotering och en faktura leder till en händelse i logg
- En order leder till en till fler händelser i logg: när den skapas, när den uppdateras (kan ske fler gånger) och när den raderas

1.6 ER-diagram kompletterat med attribut och kandidatnycklar

- Produkt(id, namn, antal, kort beskrivning, bild)
- Produktkategori (id, namn, bild)
- Kund (id, namn, telefon, epost, fakturaadress, leveransadress, datum)
- Hylla (id, position)
- ProduktHylla (produkt_id, antal, hyll_id)
- Order (id, kund_id, datum, senast uppdaterat, status)
- OrderRad (order_id, produkt_id, antal)
- Plocklista (id, order_id, datum)
- PlocklistaRad (plocklista_id, produkt_id, hyll_id, antal)
- Restnotering (id, order_id, datum)
- RestnotRad (restnot_id, produkt_id, antal)
- Faktura (id, order_id, kund_id, summa)
- FakturaRad (faktura_id, produkt_id, antal, pris)
- Logg (id, timestamp, händelse)

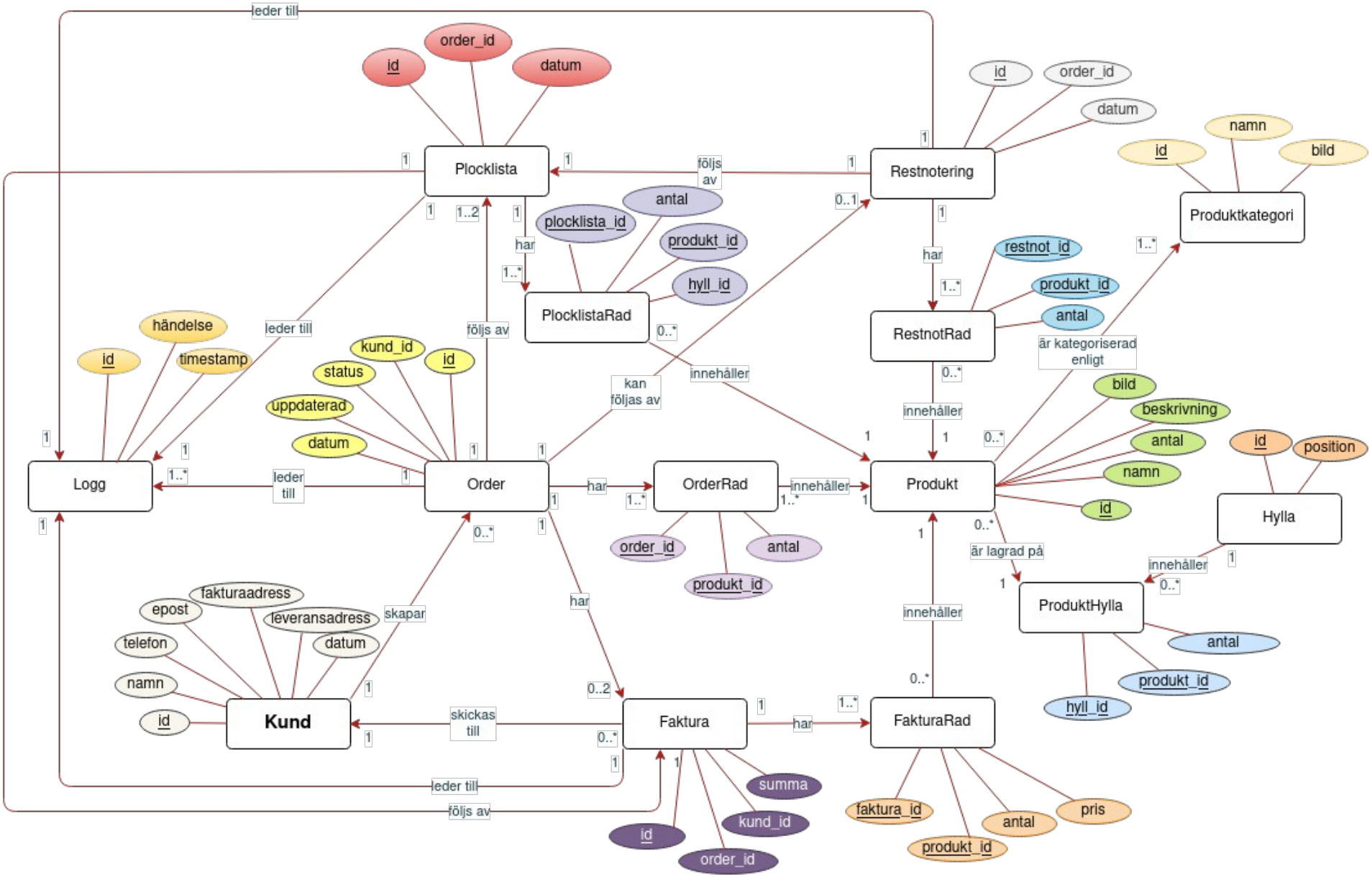

Figure 3: ER-diagram med attribut och kandidatnycklar

2 Logisk modell

2.1 ER-diagrammet enligt relationsmodellen

- Produkt(id, namn, antal, kort beskrivning, bild)
- Produktkategori (id, namn, bild)
- Kattegorisering (produkt_id, kategori_id)
- Kund (id, namn, telefon, epost, fakturaadress, leveransadress, datum)
- Hylla (id, position)
- ProduktHylla (produkt_id, hyll_id, antal)
- Order (id, kund_id, datum, senast uppdaterat, status)
- OrderRad (order_id, produkt_id, antal)
- Plocklista (id, order_id, datum)
- PlocklistaRad (plocklista_id, produkt_id, hyll_id, antal)
- Restnotering (id, order_id, datum)
- RestnotRad (restnot_id, produkt_id, antal)
- Faktura (id, order_id, kund_id, summa)
- FakturaRad (faktura_id, produkt_id, antal, pris)
- Logg (id, timestamp, händelse)

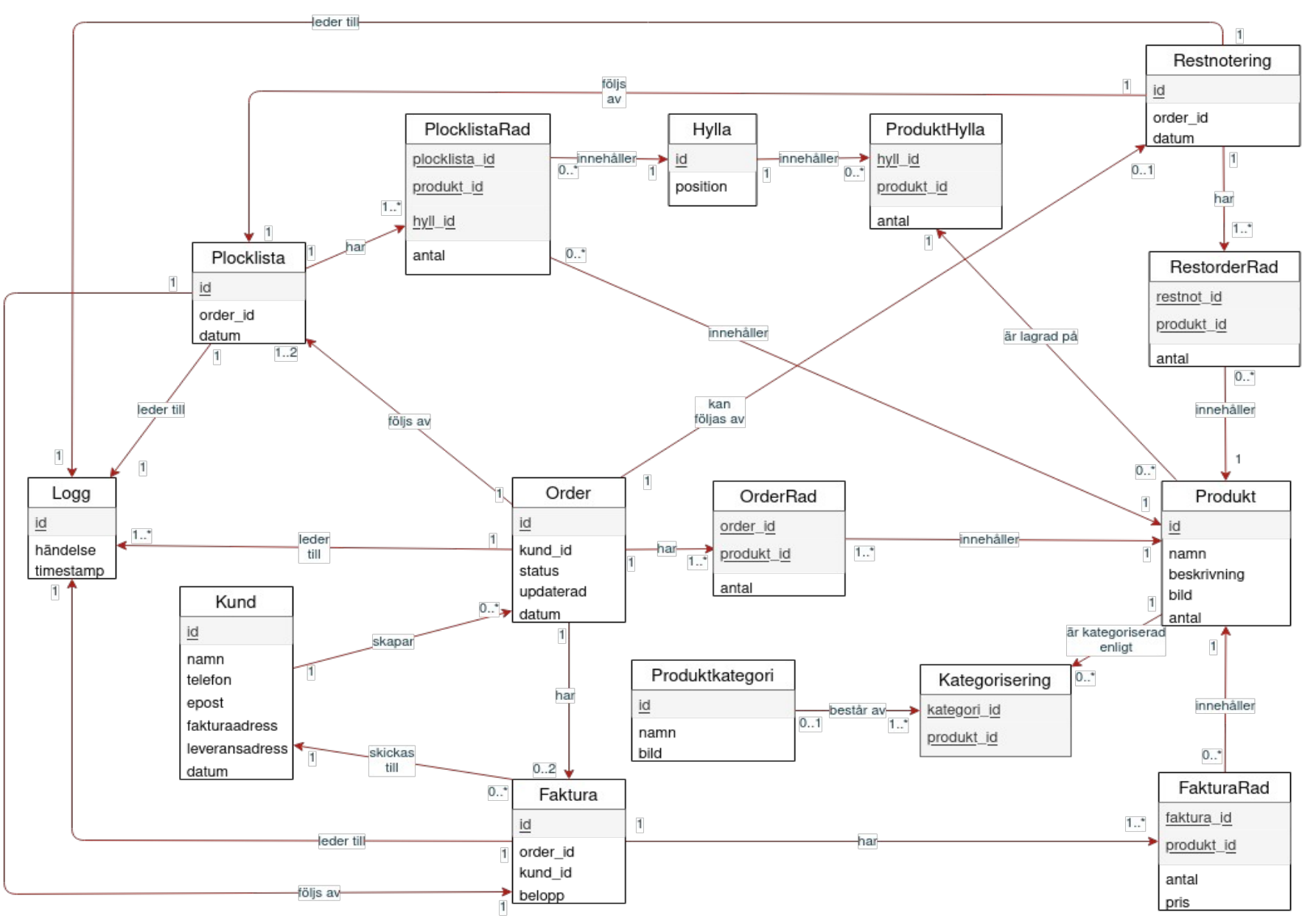

Figure 4: ER-diagram med attribut och kandidatnycklar.

2.2 ER-diagrammet med primära och främmande nycklar

- Produkt(id, namn, antal, kort beskrivning, bild)
- Produktkategori (id, namn, bild)
- Kattegorisering (#produkt_id, #kategori_id)
- Kund (id, namn, telefon, epost, fakturaadress, leveransadress, datum)
- Hylla (id, position)
- ProduktHylla (#produkt_id, #hyll_id, antal)
- Order (id, #kund_id, datum, senast uppdaterat, status)
- OrderRad (#order_id, #produkt_id, antal)
- Plocklista (id, #order_id, datum)
- PlocklistaRad (#plocklista_id, #produkt_id, #hyll_id, antal)
- Restnotering (id, #order_id, datum)
- RestnotRad (#restnot_id, #produkt_id, antal)
- Faktura (id, #order_id, #kund_id, summa)
- FakturaRad (#faktura_id, #produkt_id, antal, pris)
- Logg (id, timestamp, händelse)

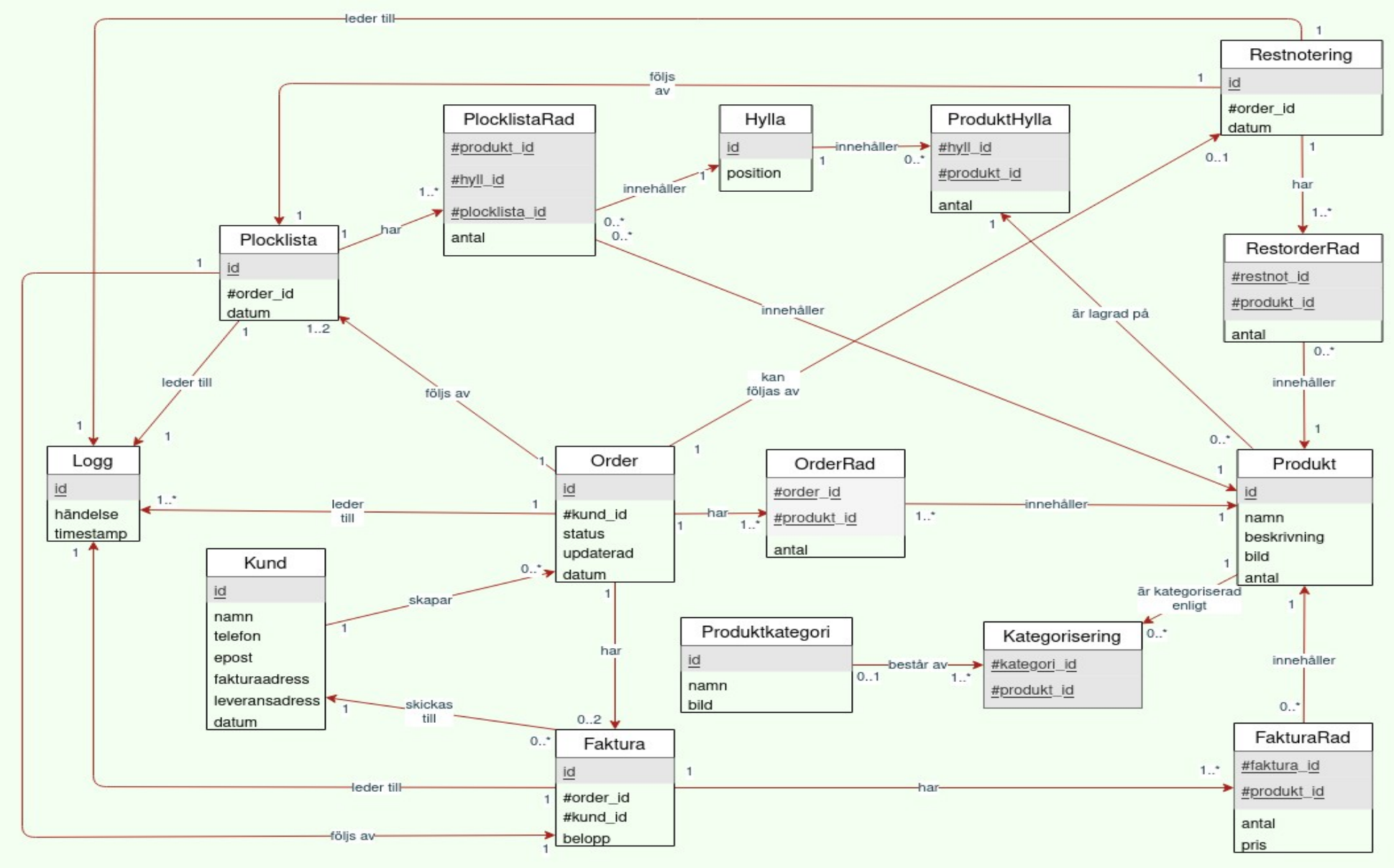

Figure 5: ER-diagramm med primära och främmande nycklar samt kompletterande attribut.

3 Fysisk modell

3.1 ER-model: iteration 2

Under arbetetet med mysql blev det en del förändringar:

1. Order är ett reserverat ord i SQL så tabellen fick heta Beställning (bestallning).
2. hyll_id blev hyllla_id för att kunna följa namnkonventionen
3. Samtliga id fick tabellnamn som prefix
4. Samtliga tabellnamn skrevs om till gemener
5. Namn har fått mer unika namn som produktnamn mm.
6. Namn i kund-tabellen delades i förnamn och efternamn

Den färdiga databasen som ER-diagram är presenterad i Figure 6 automatiskt genererat från MySQL Workbench. Steget innan var att skriva setup.sql och dll.sql (finns i appendix) och sedan baserat på det autogenerera ER-diagram och snygga till den.

3.2 Funktionslistan

1. Lägg till, uppdatera och ta bort en kund
2. Lägg till, uppdatera och ta bort en produkt
3. Lägg till, uppdatera och ta bort en produktkategori
4. Lägg till och ta bort en produkt i/från en produktkategori
5. Skapa, uppdatera och tabort en order
6. Skapa en plocklista
7. Skapa och uppdatera en restorder
8. Skapa en faktura
9. Skapa och ta bort en hylla
10. Uppdatera lagerstatus för en produkt
11. Skapa en logghändelse

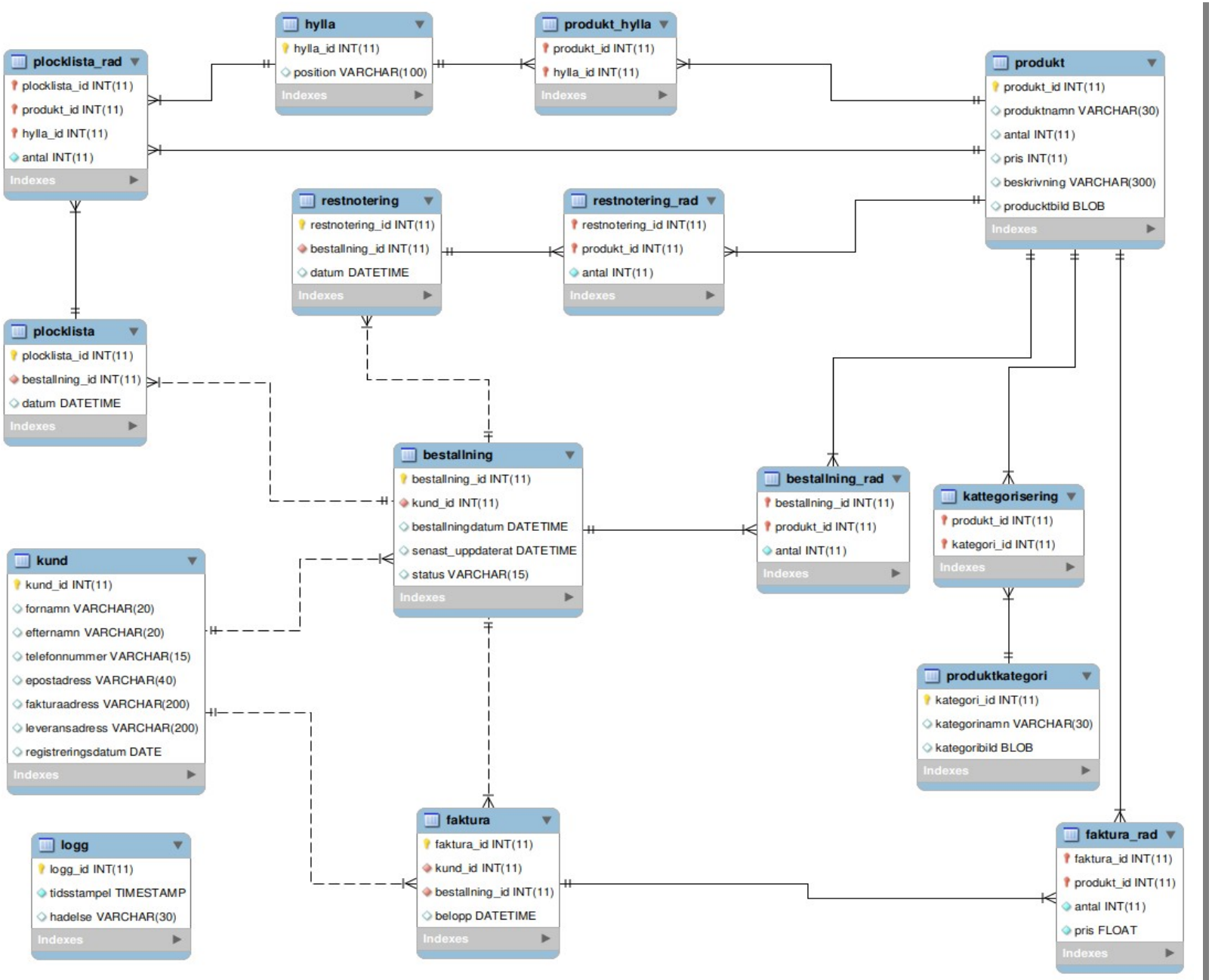

Figure 6: ER-diagram iteration 2

4 Appendix

4.1 ddl.sql

```
--
-- file: ddl.sql
-- Skapa tabeller för eshop.
-- By necu20 for course databas.
-- 2021-02-016
--

SET NAMES 'utf8';
USE eshop;

DROP TABLE IF EXISTS logg;
DROP TABLE IF EXISTS kattegorisering;
DROP TABLE IF EXISTS produkt_hylla;
DROP TABLE IF EXISTS bestallning_rad;
DROP TABLE IF EXISTS plocklista_rad;
DROP TABLE IF EXISTS plocklista;
DROP TABLE IF EXISTS restnotering_rad;
DROP TABLE IF EXISTS restnotering;
DROP TABLE IF EXISTS faktura_rad;
DROP TABLE IF EXISTS faktura;
DROP TABLE IF EXISTS bestallning;

DROP TABLE IF EXISTS produkt;
DROP TABLE IF EXISTS produktkategori;
DROP TABLE IF EXISTS kund;
DROP TABLE IF EXISTS hylla;


-- produkt(id, namn, antal, kort beskrivning, bild)
DROP TABLE IF EXISTS produkt;
CREATE TABLE produkt
(
```

```
    `produkt_id` INT AUTO_INCREMENT NOT NULL,
    `produktnamn` VARCHAR(30),
    `antal` INT,
    `pris` INT,
    `beskrivning` VARCHAR(300),
    `producktbild` blob,

    PRIMARY KEY (produkt_id)
)
ENGINE=INNODB
CHARSET utf8
COLLATE utf8_swedish_ci
;

-- produktkategori (id, namn, bild)
DROP TABLE IF EXISTS produktkategori;
CREATE TABLE produktkategori
(
    `kategori_id` INT AUTO_INCREMENT NOT NULL,
    `kategorinamn` VARCHAR(30),
    `kategoribild` blob,

    PRIMARY KEY (kategori_id)
)
ENGINE=INNODB
CHARSET utf8
COLLATE utf8_swedish_ci
;

-- kattegorisering (#produkt_id, #kategori_id)
DROP TABLE IF EXISTS kattegorisering;
CREATE TABLE kattegorisering
(
    `produkt_id`  INT NOT NULL,
    `kategori_id` INT NOT NULL,
```

```
    PRIMARY KEY (produkt_id, kategori_id),
    FOREIGN KEY (produkt_id) REFERENCES produkt(produkt_id),
    FOREIGN KEY (kategori_id) REFERENCES produktkategori(kategori_id)
)
ENGINE=INNODB
CHARSET utf8
COLLATE utf8_swedish_ci
;

-- kund (id, namn, telefon, epost, fakturaadress, leveransadress, datum)
DROP TABLE IF EXISTS kund;
CREATE TABLE kund
(
    `kund_id`            INT AUTO_INCREMENT NOT NULL,
    `fornamn`            VARCHAR(20),
    `efternamn`          VARCHAR(20),
    `telefonnummer`      VARCHAR(15),
    `epostadress`        VARCHAR(40),
    `fakturaadress`      VARCHAR(200),
    `leveransadress`     VARCHAR(200),
    `registreringsdatum` DATE,

    PRIMARY KEY (kund_id)
)
ENGINE=INNODB
CHARSET utf8
COLLATE utf8_swedish_ci
;

-- hylla (id, position)
DROP TABLE IF EXISTS hylla;
CREATE TABLE hylla
(
    `hylla_id`   INT AUTO_INCREMENT NOT NULL,
    `position`   VARCHAR(100),
```

```
    PRIMARY KEY (hylla_id)
)
ENGINE=INNODB
CHARSET utf8
COLLATE utf8_swedish_ci
;

-- produkthylla (#produkt_id, #hyll_id, antal)
DROP TABLE IF EXISTS produkt_hylla;
CREATE TABLE produkt_hylla
(
    `produkt_id`    INT NOT NULL,
    `hylla_id`      INT NOT NULL,

    PRIMARY KEY (produkt_id, hylla_id),
    FOREIGN KEY (produkt_id) REFERENCES produkt(produkt_id),
    FOREIGN KEY (hylla_id) REFERENCES hylla(hylla_id)
)
ENGINE=INNODB
CHARSET utf8
COLLATE utf8_swedish_ci
;

-- order (id, #kund_id, datum, senast uppdaterat, status)
DROP TABLE IF EXISTS bestallning;
CREATE TABLE bestallning
(
    `bestallning_id`           INT AUTO_INCREMENT NOT NULL,
    `kund_id`            INT NOT NULL,
    `bestallningdatum`         DATETIME,
    `senast_uppdaterat`  DATETIME,
    `status`             VARCHAR(15),

    PRIMARY KEY (bestallning_id),
    FOREIGN KEY (kund_id) REFERENCES kund(kund_id)
)
```

```
ENGINE=INNODB
CHARSET utf8
COLLATE utf8_swedish_ci
;

-- bestallningrad (#bestallning_id, #produkt_id, antal)
DROP TABLE IF EXISTS bestallning_rad;
CREATE TABLE bestallning_rad
(
    `bestallning_id`      INT NOT NULL,
    `produkt_id`    INT NOT NULL,
    `antal`         INT NOT NULL,

    PRIMARY KEY (bestallning_id, produkt_id),
    FOREIGN KEY (bestallning_id) REFERENCES bestallning(bestallning_id),
    FOREIGN KEY (produkt_id) REFERENCES produkt(produkt_id)
)
ENGINE=INNODB
CHARSET utf8
COLLATE utf8_swedish_ci
;

-- plocklista (id, #bestallning_id, datum)
DROP TABLE IF EXISTS plocklista;
CREATE TABLE plocklista
(
    `plocklista_id` INT AUTO_INCREMENT NOT NULL,
    `bestallning_id`      INT NOT NULL,
    `datum`         DATETIME,

    PRIMARY KEY (plocklista_id),
    FOREIGN KEY (bestallning_id) REFERENCES bestallning(bestallning_id)
)
ENGINE=INNODB
CHARSET utf8
COLLATE utf8_swedish_ci
```

```
;

-- plocklistarad (#plocklista_id, #produkt_id, #hyll_id, antal)
DROP TABLE IF EXISTS plocklista_rad;
CREATE TABLE plocklista_rad
(
    `plocklista_id` INT NOT NULL,
    `produkt_id`    INT NOT NULL,
    `hylla_id`      INT NOT NULL,
    `antal`         INT NOT NULL,

    PRIMARY KEY (plocklista_id, produkt_id, hylla_id),
    FOREIGN KEY (plocklista_id) REFERENCES plocklista(plocklista_id),
    FOREIGN KEY (produkt_id) REFERENCES produkt(produkt_id),
    FOREIGN KEY (hylla_id) REFERENCES hylla(hylla_id)
)
ENGINE=INNODB
CHARSET utf8
COLLATE utf8_swedish_ci
;

-- restnotering (id, #bestallning_id, datum)
DROP TABLE IF EXISTS restnotering;
CREATE TABLE restnotering
(
    `restnotering_id` INT AUTO_INCREMENT NOT NULL,
    `bestallning_id`        INT NOT NULL,
    `datum`           DATETIME,

    PRIMARY KEY (restnotering_id),
    FOREIGN KEY (bestallning_id) REFERENCES bestallning(bestallning_id)
)
ENGINE=INNODB
CHARSET utf8
COLLATE utf8_swedish_ci
;
```

```
-- restnotrad (#restnot_id, #produkt_id, antal)
DROP TABLE IF EXISTS restnotering_rad;
CREATE TABLE restnotering_rad
(
    `restnotering_id`   INT NOT NULL,
    `produkt_id`   INT NOT NULL,
    `antal`        INT NOT NULL,

    PRIMARY KEY (restnotering_id, produkt_id),
    FOREIGN KEY (restnotering_id) REFERENCES restnotering(restnotering_id),
    FOREIGN KEY (produkt_id) REFERENCES produkt(produkt_id)
)
ENGINE=INNODB
CHARSET utf8
COLLATE utf8_swedish_ci
;

-- faktura (id, #bestallning_id, #kund_id, summa)
DROP TABLE IF EXISTS faktura;
CREATE TABLE faktura
(
    `faktura_id`    INT AUTO_INCREMENT NOT NULL,
    `kund_id`       INT NOT NULL,
    `bestallning_id`      INT NOT NULL,
    `belopp`        DATETIME,

    PRIMARY KEY (faktura_id),
    FOREIGN KEY (kund_id) REFERENCES kund(kund_id),
    FOREIGN KEY (bestallning_id) REFERENCES bestallning(bestallning_id)
)
ENGINE=INNODB
CHARSET utf8
COLLATE utf8_swedish_ci
;
```

```
-- fakturarad (#faktura_id, #produkt_id, antal, pris)
DROP TABLE IF EXISTS faktura_rad;
CREATE TABLE faktura_rad
(
    `faktura_id`   INT NOT NULL,
    `produkt_id`   INT NOT NULL,
    `antal`        INT NOT NULL,
    `pris`         FLOAT NOT NULL,

    PRIMARY KEY (faktura_id, produkt_id),
    FOREIGN KEY (faktura_id) REFERENCES faktura(faktura_id),
    FOREIGN KEY (produkt_id) REFERENCES produkt(produkt_id)
)
ENGINE=INNODB
CHARSET utf8
COLLATE utf8_swedish_ci
;

-- logg (id, timestamp, händelse)
DROP TABLE IF EXISTS logg;
CREATE TABLE logg
(
    `logg_id`       INT AUTO_INCREMENT NOT NULL,
    `tidsstampel`   TIMESTAMP,
    `hadelse`       VARCHAR(30),

    PRIMARY KEY (logg_id)
)
ENGINE=INNODB
CHARSET utf8
COLLATE utf8_swedish_ci
;
```