Database Design 1 Project

- P1 group 32 -

Authors:
Alexander Sundquist (alex.sundquist@gmail.com)
Karl Pettersson (karl.pettersson.3370@student.uu.se)
Pontus Björklid (pontus.bjorklid@gmail.com)
Reuben Vas (reuben.vas.4578@student.uu.se
Vilmer S. Olin (ville.olin@gmail.com)

Assumptions

- You have to be a user to lay an order.
- Retail price with tax and retail price with discount is calculated from the retail price without tax * tax and discount respectively.
- Logo is not stored in the database, but is rather a part of the website.
- Breadcrumbs is not a part of the database, but we get it from the RELATED_TO relationship.
- Link is derived from a base and primary key.
- Every order must have an order ID.
- Email address is unique
- Product needs price

ER-diagram

First iteration

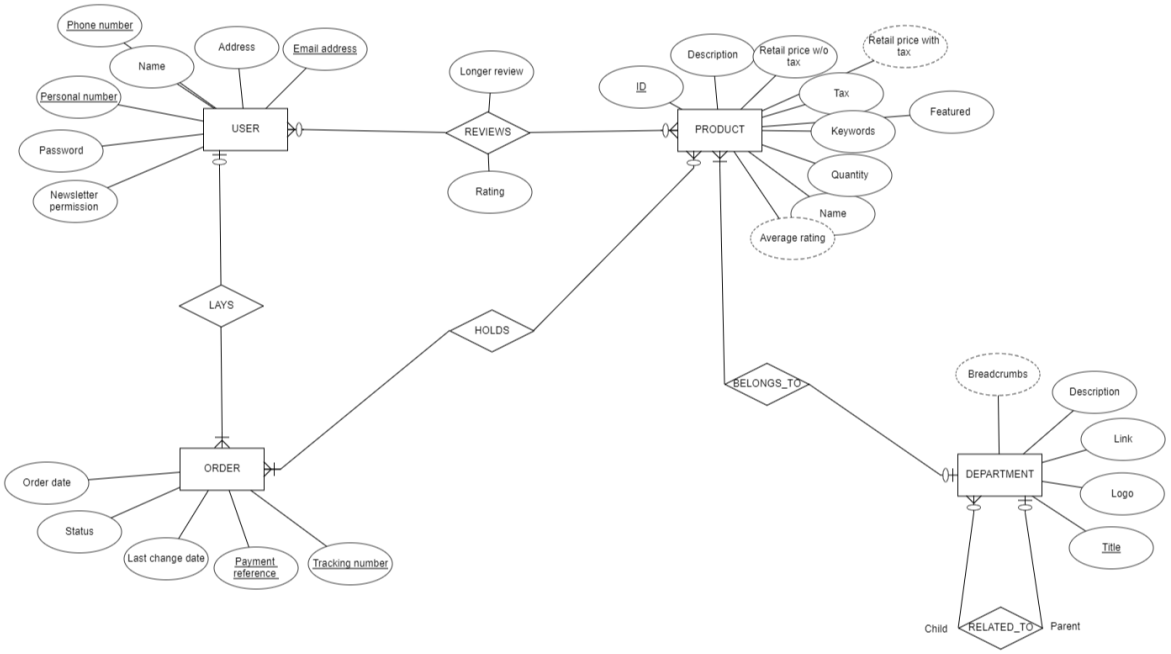

Figure 1: the first iteration ER-diagram

Second iteration

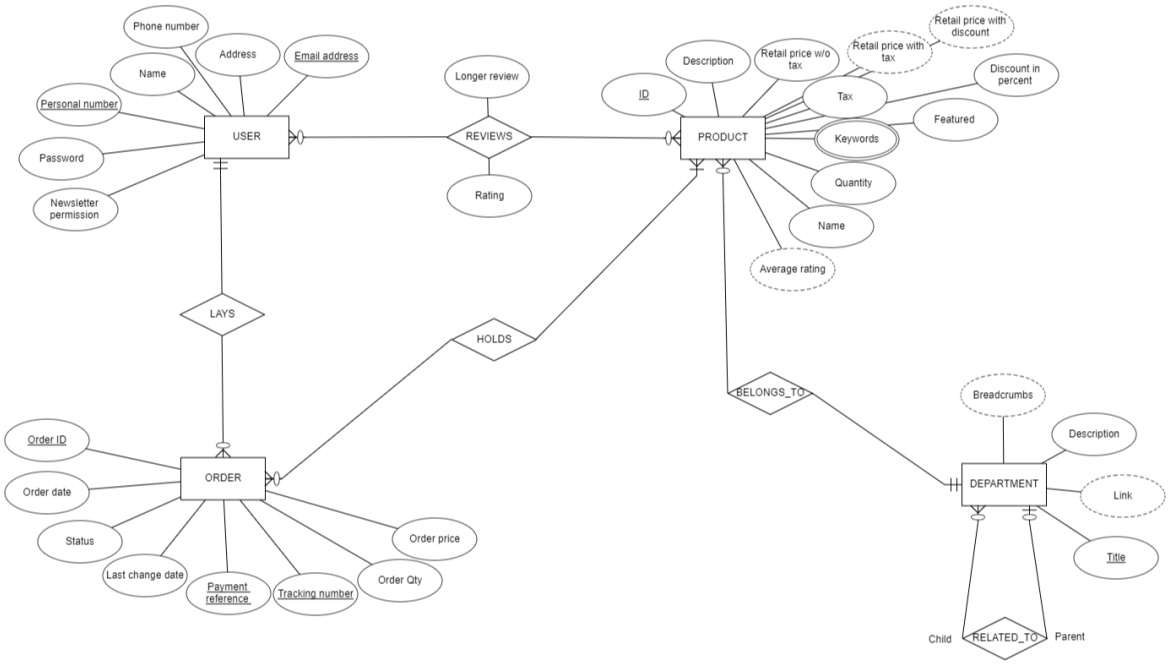

Figure 2: the second iteration ER-diagram

Third and final iteration

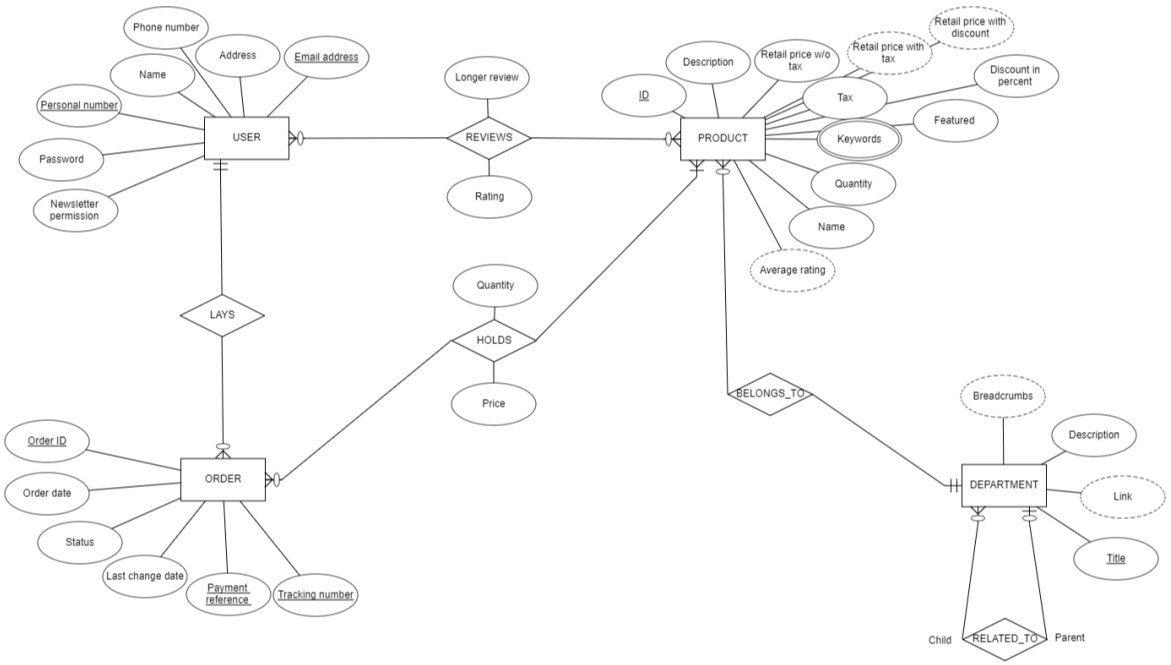

Figure 3: the third iteration ER-diagram

Normalization

Firstly, the ER-diagram was mapped to a relational model. Therefore, the schema had not been checked for any of the normal forms.

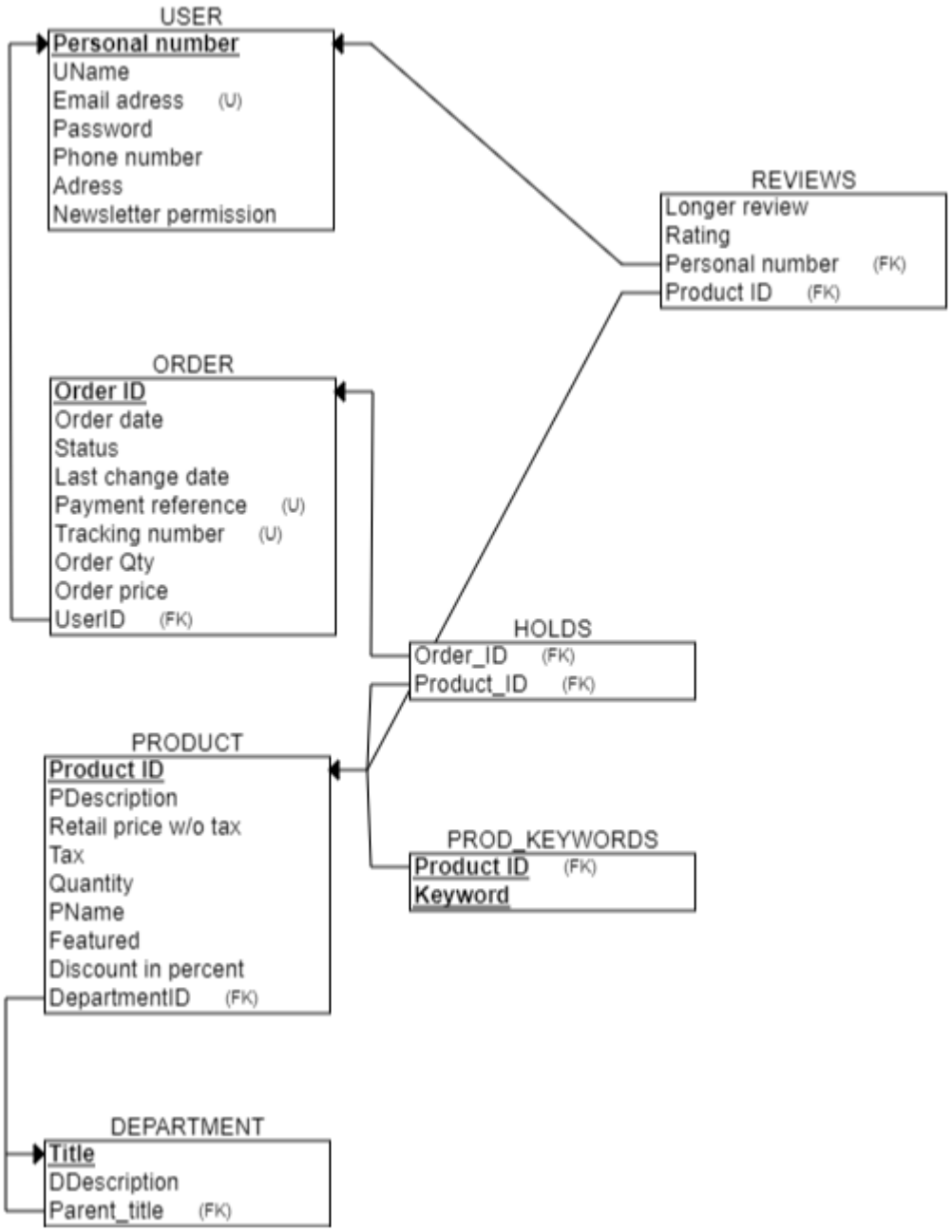

Figure 4: first relation model after mapping from the third iteration ER-diagram

After mapping the ER-diagram to a relational model, the schema was modified through the three normal forms. There were no major changes in the different normal forms, so we chose to only show the final schema on 3NF.

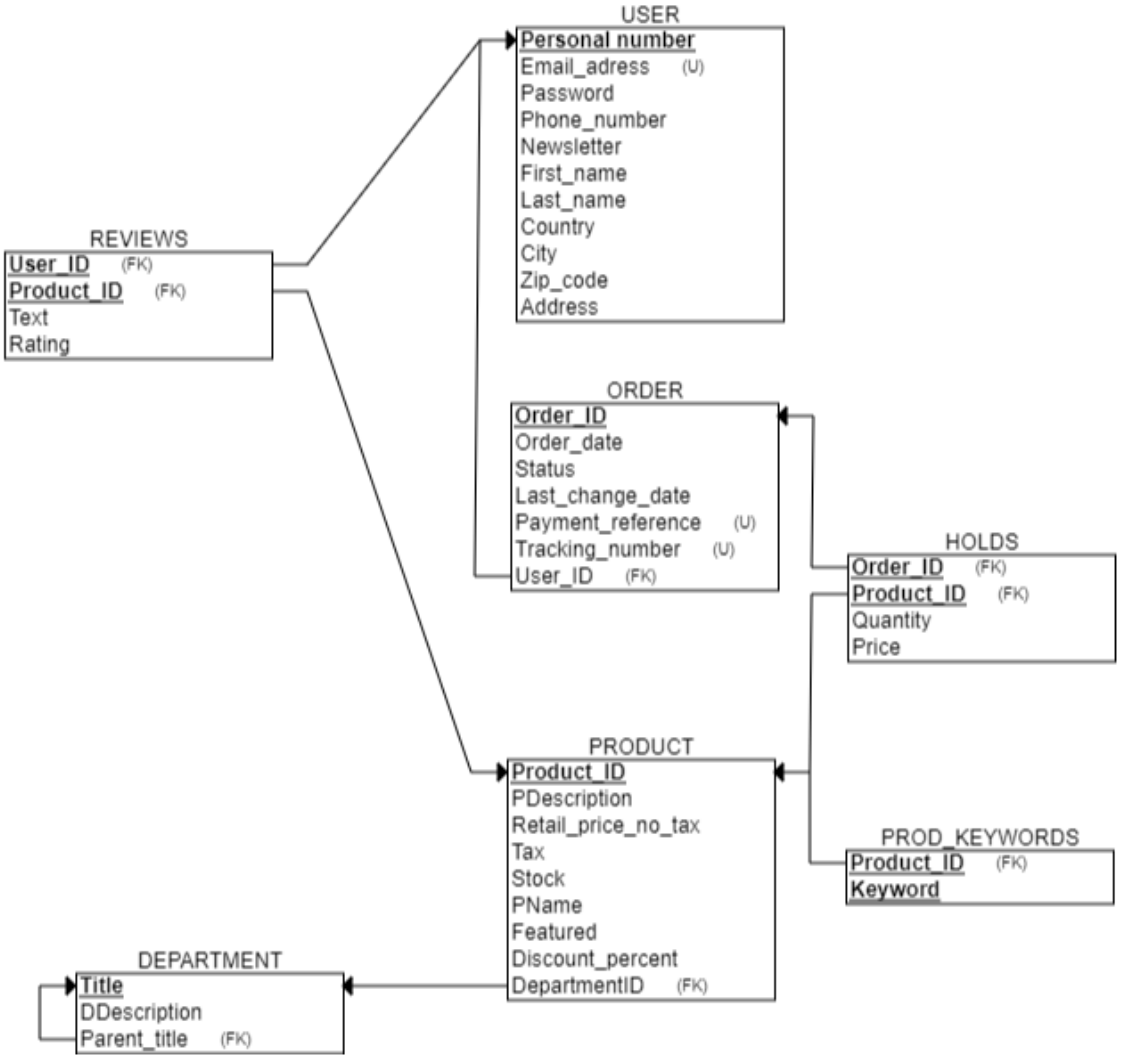

Figure 5: the schema on 3NF

Using SQL to generate and populate the tables

After normalizing the tables to 3NF the tables were generated and populated by using SQL.

The following code was used to generate the tables seen in figure 5. Some attribute names were changed from the figure.

Generating tables

```
CREATE TABLE USER(
      personal_number              CHAR(10)              NOT NULL,
      first_name                   VARCHAR(255)          NOT NULL,
      last_name                    VARCHAR(255)          NOT NULL,
      email_adress                 VARCHAR(255)          NOT NULL,
      pass_word                    VARCHAR(255)          NOT NULL,
```

```
      phone_number                 VARCHAR(20)            NOT NULL,
      country                      VARCHAR(255)           NOT NULL,
      city                         VARCHAR(255)           NOT NULL,
      zip_code                     VARCHAR(255)           NOT NULL,
      address                      VARCHAR(255)           NOT NULL,
      newsletter_permission        BOOL                   NOT NULL,
      PRIMARY KEY(personal_number),
      UNIQUE (email_adress)
      );

CREATE TABLE USER_ORDER(
      order_number                 INT(11)                NOT NULL,
      order_status                 SET("pending","confirmed", "shipped",
"delivered")                                             NOT NULL,
      last_change_date             DATE                   ,
      payment_reference            CHAR(12)               NOT NULL,
      tracking_number              VARCHAR(255)           ,
      user_id                      INT(11)                NOT NULL,
      PRIMARY KEY(order_number),
      FOREIGN KEY(user_id) REFERENCES USER(personal_number),
      UNIQUE(payment_reference, tracking_number)
);

CREATE TABLE DEPARTMENT(
      department_number      INT(11)                      NOT NULL,
      department_name        VARCHAR(255)                 NOT NULL,
      department_description VARCHAR(1511)                NOT NULL,
      parent_id              INT(11)                      ,
      PRIMARY KEY(department_number),
      FOREIGN KEY(parent_id) REFERENCES DEPARTMENT(department_number)
);


CREATE TABLE PRODUCT(
      product_number               CHAR(12)               NOT NULL,
      product_description          VARCHAR(1511)          ,
      retail_price_no_tax          FLOAT(64)              NOT NULL,
      tax_rate                     FLOAT(64)              NOT NULL,
      stock                        INT(255)               NOT NULL,
      product_name                 VARCHAR(255)           NOT NULL,
      featured                     BOOL                   NOT NULL,
      discount_percent             FLOAT(64)              NOT NULL,
      department_id                INT(11)                NOT NULL,
      PRIMARY KEY(product_number),
      FOREIGN KEY(department_id) REFERENCES
      DEPARTMENT(department_number)
);

CREATE TABLE REVIEWS(
      user_id                      INT(11)                NOT NULL,
      product_id                   INT(11)                NOT NULL,
```

```
      review_text                  VARCHAR(1511)          ,
      rating                       TINYINT                NOT NULL
                                   CHECK(
                                   rating > 0
                                   AND rating <=5),
      PRIMARY KEY(user_id, product_id),
      FOREIGN KEY(user_id) REFERENCES USER(personal_number),
      FOREIGN KEY(product_id) REFERENCES PRODUCT(product_number)
);

CREATE TABLE HOLDS(
      order_id                     INT(11)                NOT NULL,
      product_id                   INT(11)                NOT NULL,
      order_qty                    INT(255)               NOT NULL,
      order_price                  FLOAT(64)              NOT NULL,
      PRIMARY KEY(order_id, product_id),
      FOREIGN KEY(order_id) REFERENCES USER_ORDER(order_number),
      FOREIGN KEY(product_id) REFERENCES PRODUCT(product_number)
);

CREATE TABLE PRODUCT_KEYWORDS(
      product_id                   CHAR(12)               NOT NULL,
      keyword                      VARCHAR(255)           NOT NULL,
      PRIMARY KEY(product_id, keyword),
      FOREIGN KEY(product_id) REFERENCES PRODUCT(product_number)
);
```

Populating tables

Department

```
insert into DEPARTMENT
values(1,   "Homepage", "This is the homepage of the website", );

insert into DEPARTMENT
values(102, "Sandals",  "This department contains cool wear for cool
feet",      321);

insert into DEPARTMENT
values(111, "Tank",-"tops     This department contains "grymma
tank-tops"",     123);

insert into DEPARTMENT
values(123, "Clothes",  "This department contains "najs kläder"",     1);

insert into DEPARTMENT
values(242, "Boots",    "This department contains dope boots for winter
and autumn",     321);
```

```
insert into DEPARTMENT
values(251, "Pants",   "This department contains all kinds of pants",
      123);

insert into DEPARTMENT
values(253, "Underwear",     "This department contains "trosor" och
"kalsonger""     123);

insert into DEPARTMENT
values(321, "Shoes",   "This department contains "feta skor"", 1);

insert into DEPARTMENT
values(444, "Pants",   "This department contains everything ranging
from "fula byxor" to "hängslebyxor"",   123);

insert into DEPARTMENT
values(634, "Spirits", "This department contains spirits",     666);

insert into DEPARTMENT
values(635, "Beers",   "This department contains beers", 666);

insert into DEPARTMENT
values(651, "Snus",    "This department contains snus",  666);

insert into DEPARTMENT
values(666, "NSFW",    "This department contains tobacco and alcohol",
      1);

insert into DEPARTMENT
values(696, "Sneakers", "This department contains cool sneakers",
321);
```

Holds

```
insert into HOLDS
values(181001,   33,   1,    1099);

insert into HOLDS
values(181001,   34,   1,    128.738);

insert into HOLDS
values(181001,   35,   2,    79.8976);

insert into HOLDS
values(181001,   1101, 8,    395.88);

insert into HOLDS
values(181002,   37,   19,   4939.03);

insert into HOLDS
values(181002,   143,  1,    699);
```

Product

```
insert into PRODUCT
values(‘1101’,"Home grown bottle small company beer", 32.99, 1.5,
1500, "Ångans Bryggeri Stark 05l",0,0,635);

insert into PRODUCT
values(‘143’,    "Classic dad sandals", 559.2,1.25,50,"Classic Casual
Beach Sandals",0,0,102);

insert into PRODUCT
values(‘156’,    "Not too overused and very nice colored
\"kalsonger\"", 25,1.25,     200,  "Thrifted White Briefs",1,0.2,253);

insert into PRODUCT
values(‘237’,    "Chonky boots for edgy young adults",   1599.2,
1.25, 79,"Chonky Black Leather Boots",0,0.4,242);

insert into PRODUCT
values(‘32’,"Blue jeans with loose fit",479.2,1.25,103,"Loose Fit Blue
Jeans",0,0,444);

insert into PRODUCT
values(‘33’,"Clean white sneakers",879.2,     1.25, 58,"Abibos Jan
Smoth",0,0,696);

insert into PRODUCT
values(‘34’,"Wife beater",102.99,1.25,150,"Classical wife
beater",0,0,111);

insert into PRODUCT
values(‘35’,"Really good snus",24.968,1.6,300,"Lundgrens",1    ,0,651);

insert into PRODUCT
values(‘36’,"Great Whiskey; Stored in burgundy barrells for way too
long",372.468,1.6,1,"Talubardine Premium Collection",1,0,634);

insert into PRODUCT
values(‘37’,"Strongest Vodka in the world",162.468,1.6,27,"Extra
Absoult Vodka 44%",0,0,634);

insert into PRODUCT
values(‘38’,"Way too expencive sneakers",8000,1.25,2,"Air Jordan 1 -
Chicago Blue",1,0,321);

insert into PRODUCT
values(‘452’,    "Signature calm rangel pants",1455.15,1.25,54,"Calm
Rangel Pool Cut",1,0,251);
```

Product keywords

```
insert into PRODUCT_KEYWORDS
values ('1101', 'Beer');

insert into PRODUCT_KEYWORDS
values ('1101', 'Local');

insert into PRODUCT_KEYWORDS
values ('1101', 'Organic');

insert into PRODUCT_KEYWORDS
values ('143','Aero-dynamic');

insert into PRODUCT_KEYWORDS
values ('143','Comfy');

insert into PRODUCT_KEYWORDS
values ('156', 'Briefs');

insert into PRODUCT_KEYWORDS
values ('237', 'Edgy');

insert into PRODUCT_KEYWORDS
values ('237', 'Warm');

insert into PRODUCT_KEYWORDS
values ('32', 'Trendy');

insert into PRODUCT_KEYWORDS
values ('33', 'Trendy');

insert into PRODUCT_KEYWORDS
values ('34', 'Classical look');

insert into PRODUCT_KEYWORDS
values ('35', 'Fresh taste');

insert into PRODUCT_KEYWORDS
values ('35', 'Local');

insert into PRODUCT_KEYWORDS
values ('36', 'Smokey');

insert into PRODUCT_KEYWORDS
values ('37', 'Cleansing');

insert into PRODUCT_KEYWORDS
values ('37', 'Refreshing');

insert into PRODUCT_KEYWORDS
```

```
values ('37', 'Strong');

insert into PRODUCT_KEYWORDS
values ('38', 'Classic');

insert into PRODUCT_KEYWORDS
values ('38', 'Comfy');

insert into PRODUCT_KEYWORDS
values ('38', 'Trendy');

insert into PRODUCT_KEYWORDS
values ('452', 'Comfy');

insert into PRODUCT_KEYWORDS
values ('452', 'Trendy');
```

Reviews

```
insert into REVIEWS
values(1234,36, "i give to neighbor, very bad", 2);

insert into REVIEWS
values(1234,37, "hate", 0);

insert into REVIEWS
values(4046353,35, "i need more instructions to handle this", 0);

insert into REVIEWS
values(109015825, 32, "Too loose for me", 1);

insert into REVIEWS
values(109015825,35, "not good", 2);

insert into REVIEWS
values(109015825,36, "Gave this to 14 year old son - he like very
much", 5);

insert into REVIEWS
values(109015825,143,"love these", 4);

insert into REVIEWS
values(2147483647,34, "not as expected but still ok", 0);

insert into REVIEWS
values(2147483647,35, "Very good \"snus\"", 5);

insert into REVIEWS
values(2147483647,36, "i give to neighbor, very bad", 2);
```

```
insert into REVIEWS
values(2147483647,37, "i'm very sad", 1);

insert into REVIEWS
values(2147483647,156, "not so comfy", 1);

insert into REVIEWS
values(2147483647,237, "good", 3);

insert into REVIEWS
values(2147483647,452, "nice", 1);
```

User

```
insert into USER
values(“0001135789”,”Vilmer S”, “Olin”, “vilmer.s.olin@sqlmail.net”,
“andiwillalwaysloveyou”,”0734448564”,”Sweden”,”Uppsala”,”75327”,”Botvid
sgatan 14B”,0);

insert into USER
values(“0004046353”,”Pontus”,”Björklid”,”pontus@bjorklid.se”,”egandomän
haha”,“0732634926”,”Sweden”,”Uppsala”,”75314”,“Sturegatan 9”,0);

insert into USER
values(“0109015825”,”Alex”,”Sunkan”,”alex.sunk@gmail.com”,”sunkosaurus”
,”0734868585”,”Sweden”,”Uppsala”,”75413”,”Nånstansrunttriangeln”,1);

insert into USER
values(“1010119217”,”Lille”,”Putt”,”lpakaputt@sqlmail.net”,”lösen544”,”
08635472”,”Sweden”,”Uppsala”,”43724”,”Plusplan 20”,1);

insert into USER
values(“6404252988”, “Torsten”, “Tass”, “torsten.tass@sqlmail.net”,
“pontus123”,”0728548371”,”Sweden”,”Uppsala”,”43722”,”Torpängen 89”,0);

insert into USER
values(“6703057665”, “Rupin”, “Baas”, “rupin.baas@mailers.fi”,
“thisispassword”, “0745838822”, “Sweden”, “Uppsala”, “43720”,
“Gatuvägen 1”, 1);

insert into USER
values(“6909176542”, “Karl”, “XVI Gustaf”, “knugen@kungligaslottet.se”,
“needettrorjaginte”, “08888820”, “Sweden”, “Stockholm”, “11101”,
“Kungliga Slottet”, 1);

insert into USER
values(“9712245432”, “Rhudy”, “Vaes”, “rhudy.vaes@sqlmail.net”,
“hasdgfauwq2018”. “0701997575”, “Sweden”, “Mamlö”, “14454”, “Centrum
9”, 1);

insert into USER
```

```
values(“9810147653”, “Roopyn”, “Whas”, “iamroopyn@sqlmail.net”,
“username”, “0741234562”, “Sweden”, “Åkersberga”, “65237”, “Gräsgatan
19”, 1);

insert into USER
values(“9910142142”, “Råjpin”, “Vass”, “rajpin.vass@hotmail.com”,
“iloverojpin”, “0705678921”, “Sweden”, “Uppsala”, “75314”,
“Nykterhetsvännernasriksförbundsgatan”, 1);

insert into USER
values(“9910142144”, “Rajbin”, “Vass”,  “rajbin.vass@hottermail.com”,
“roybon123”, “0705634523”, “Sweden, “Uppsala”, “75346”, “Carolina
Rediviva”, 0);
```

User order

```
insert into USER_ORDER
values(181001, “delivered”,NULL, “975654321”,”zb6570-1033”, 109015825);

insert into USER_ORDER
values(181002, “shipped”,NULL, “975726538”, “bg6353-1034”,4046353);

insert into USER_ORDER
values(181003,”pending”,NULL, “975621534”,”uh1683-1093”, 2147483647);

insert into USER_ORDER
values(191001,”pending”,”2022-12-12”,”986356284”,
“sw4683-1239”,2147483647);
```

MySQL Workbench’s Reverse Engineer

When executing MySQL Workbench’s Reverse Engineer function the following diagram was attained:

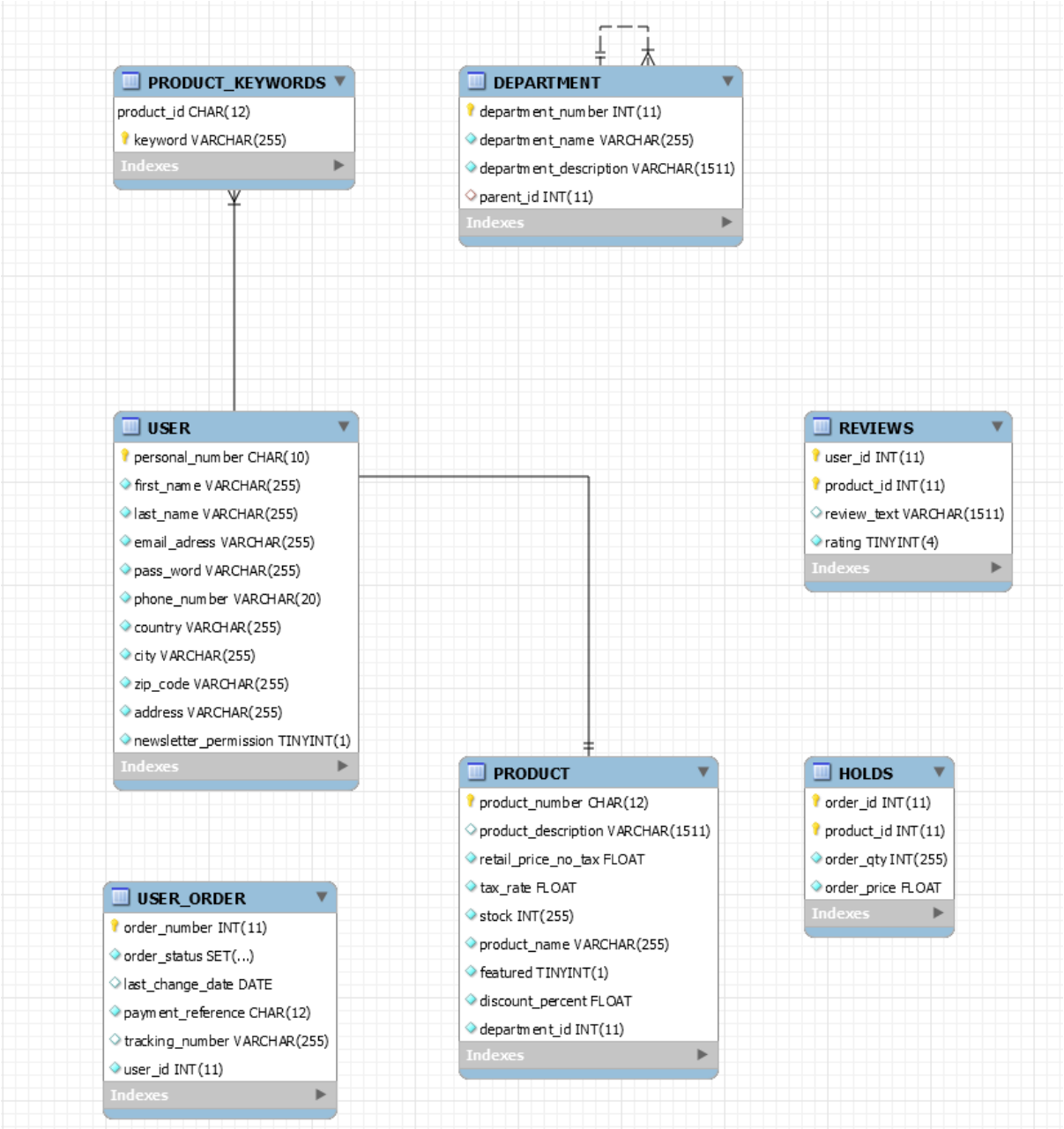

Figure 6: the diagram obtained through MySQL's Reverse Engineer function.

SQL queries

The query scripts for query 1-4 are unfortunately on another computer that we can't reach right now. We will complete this part of the hand-in tomorrow, 3/10, and send in the complete version.

- Welcome text for the homepage

```
SELECT department_description
FROM DEPARTMENT
WHERE department_name = 'Homepage';
```

- List of the top level departments with fields needed for the homepage

```
SELECT department_name, department_description
FROM DEPARTMENT
WHERE parent_id = '1';
```

- List of the featured products with fields needed for the homepage

```
SELECT product_name, product_description, stock,
retail_price_no_tax,tax_rate,discount_percent, department_id
FROM PRODUCT
WHERE featured = '1';
```

- Given a product, list all keyword-related products

```
select product_name
from PRODUCT
where product_number in
      (select product_id
    from PRODUCT_KEYWORDS
    where keyword in
            (select keyword
        from PRODUCT_KEYWORDS
        where product_id = "38"))
```

- Given an department, list of all its products (title, short description, current retail price) with their average rating

```
# byt ut 634 i department_id i where mot det department man vill få upp
produkterna för

SELECT P.product_number ,P.product_name, P.product_description,
P.retail_price_no_tax*P.tax_rate as retail_price, avg(R.rating) as
average_rating
FROM ht22_1_project_group_32.PRODUCT as P,
ht22_1_project_group_32.REVIEWS as R
where department_id=634 and P.product_number=R.product_id
group by P.product_number
```

- List of all products on sale sorted by the discount percentage (starting with the biggest discount)

```
SELECT * FROM ht22_1_project_group_32.PRODUCT where discount_percent>0
order by discount_percent desc;
```

SQL to create indices

The following code was used to create the index NewsletterIndex for the USER table on the attribute newsletter_permission.

```
CREATE INDEX newsletter_index
ON USER (newsletter_permission);
```

We thought it would make sense to make an index of the newsletter permission, as the attribute takes the boolean values of 0 and 1, which makes it easy to sort. The query that would benefit from this index would be the SELECT command as searching for the users who want to receive newsletters would be much faster.

In figure 7 and 8 the outputs of the EXPLAIN statement, before and after implementing the newletter_index, is shown.

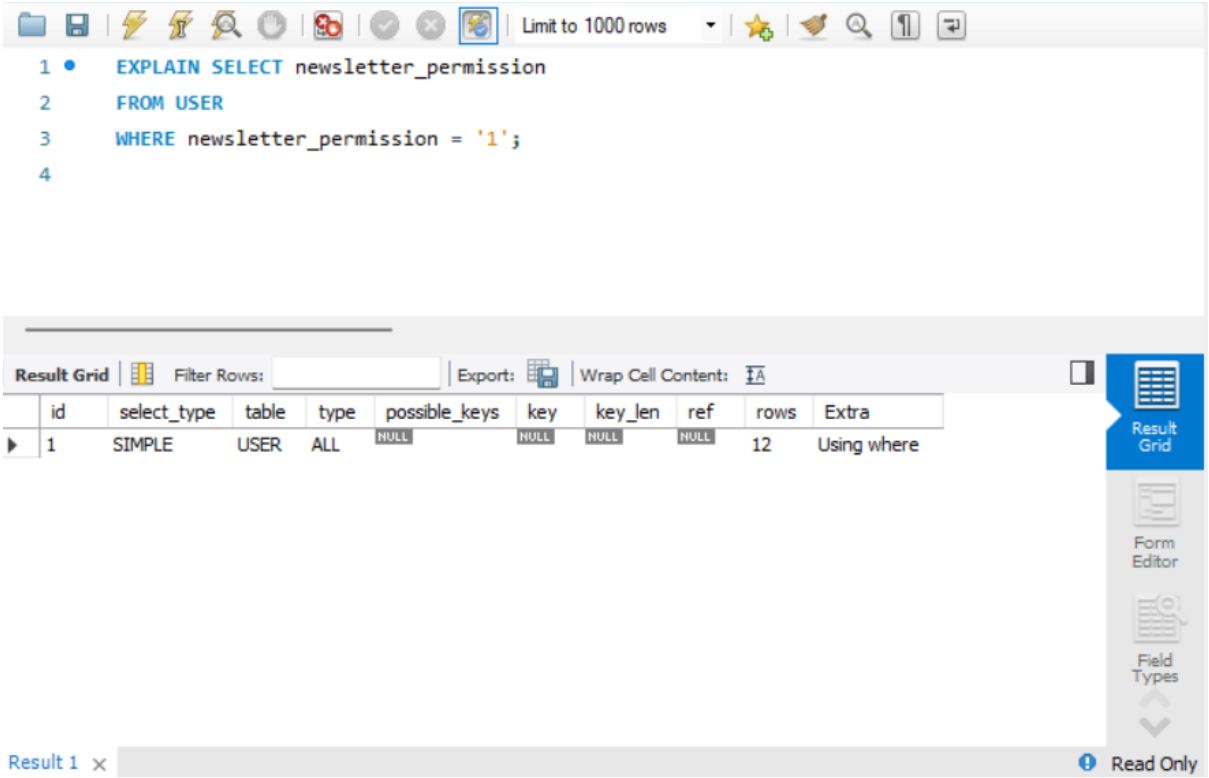

Figure 7: explain output before implementing the index

```
EXPLAIN SELECT newsletter_permission
FROM USER
WHERE newsletter_permission = '1';
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

id	select_type	table	type	possible_keys	key	key_len	ref	rows	Extra
1	SIMPLE	USER	ref	newsletter_index	newsletter_index	1	const	7	Using index

Result Grid | Form Editor | Field Types

Result 2 | Read Only

Figure 8: explain output after implementing the index

Source code

Regarding Milestone 4, the source code of part H can be found in the directory called "project". The code is runned initially from ./DBApp.py with the business logic in ./controller.py where each database functionality mentioned in the assignment has its own file or module in the same directory.