Miniproject 1 - DIS

Lars Andersen, Mathias Winde Pedersen & Søren Skibsted Als

October 5, 2014

B

By looking at the data, different business processes have been considered. A central one is that of sale. Another process to model is the payment process, which is the pre-insertion of money.

We choose to focus on the sale business process.

For the granularity, the data already contains a timestamp for each sale, as of such we maintain that granularity.

That is timestamp without timezone, containing year, month, day, hour, minute and second. However, a weekday could be relevant as well, and when extracting the data, it could be interesting to include this information, same goes for season, holidays, and week number.

The granularity for a member is year, as we cannot go deeper there, without coming up with some fictive data.

The choices are reasonable for paying customers, which is reflected in the questions that can be asked.

For sales, it could be interesting to see if the sales increase during breaks, and the granularity here is fine with down to minutes, for seconds it is still fine as it does not require much effort to record this as well.

For member creation the year of creation is as close as we can get, thus this is also reasonable for the given data.

Questions that can be asked:

Does sales increase during break times (12:00 to 12:30)?

How has the total sales per year evolved?

Which product has been sold the most?

Which member spends the most money?

Which weekday, weeknumber, season, holiday have the largest revenue?

C

For SCDs products could be useful, as the price changes slowly over time, same goes for members, as they can be active or inactive.

For the time dimension, we skipped weekdays etc. as we would have to find a library of some sort to translate this. The dimensions have been specified in Figure 1. The schema can be seen in the code below and in Figure 2.

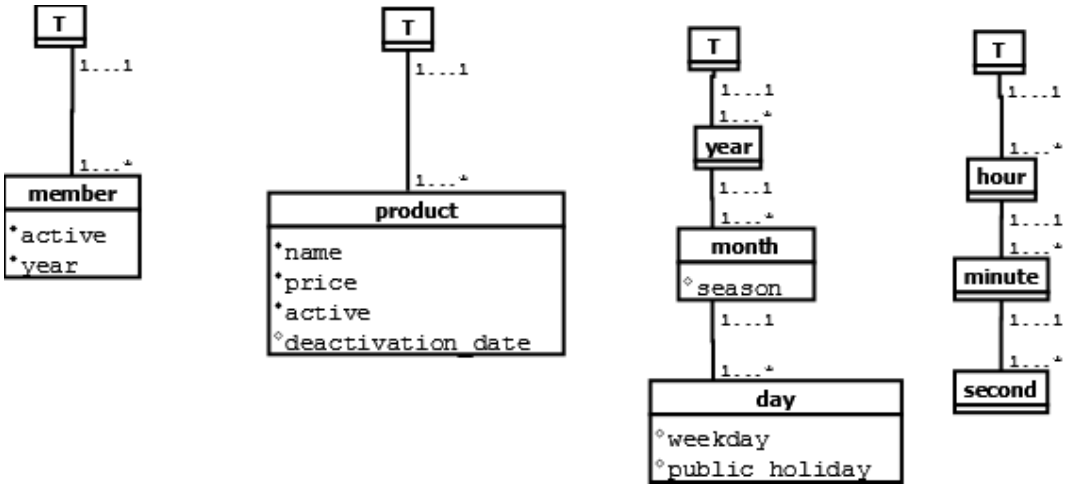

Figure 1: Dimensions, part C.

```
DROP TABLE sale;
DROP TABLE product;
DROP TABLE member;
DROP TABLE timeofday;
DROP TABLE date;

CREATE TABLE "date"
(
id serial PRIMARY KEY,
year int NOT NULL,
month int NOT NULL,
day int NOT NULL,
weekday int,
season int,
public_holiday int
);

CREATE TABLE timeofday
(
id serial PRIMARY KEY,
hour int NOT NULL,
minute int NOT NULL,
second int NOT NULL
);


CREATE TABLE member
(
surrogateid serial PRIMARY KEY,
memberid int NOT NULL,
active int NOT NULL,
```

```
year int NOT NULL,
version int NOT NULL DEFAULT 1,
changedate timestamp without time zone -- this is for future changes,
--fklub does not keep track of this
);

CREATE TABLE product
(
surrogateid serial PRIMARY KEY,
id int NOT NULL,
name varchar(100),
active int NOT NULL,
deactivation_date timestamp without time zone,
version int NOT NULL DEFAULT 1
);

CREATE TABLE sale
(
id serial PRIMARY KEY,
memberid int NOT NULL,
productid int NOT NULL,
dateid int NOT NULL,
timeofdayid int NOT NULL,
price int NOT NULL,
FOREIGN KEY(memberid) REFERENCES member(surrogateid),
FOREIGN KEY(productid) REFERENCES product(surrogateid),
FOREIGN KEY(dateid) REFERENCES "date"(id),
FOREIGN KEY(timeofdayid) REFERENCES timeofday(id)
);
```

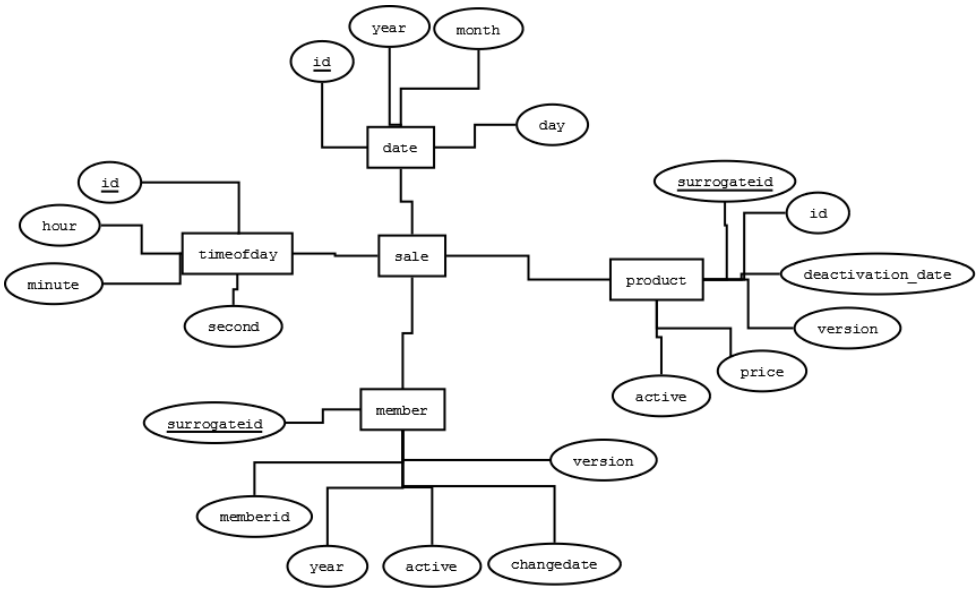

Figure 2: schema, part C.

D

We started this exercise with an overall plan, where we decided to load the data from the database, and then to transform this data and insert it into the dimension tables. After this, the fact table can be filled. We realised that the schema had to be changed, and as of such we moved the price from the dimension product, to the fact table, as price is what we measure. Additionally, we do not have incremental loads, but should be strived for in a larger and more permanent system.

In figure 3 a flow of how the dimensions are filled can be seen. As can be seen, members are added straightforward, as the old and new database map one to one, with just an additional surrogate key. However, a dirty fix is that we add a member with memberid -1, in order to handle an error that occurred when filling the fact table (more on this in the description of the fact table flow).

The filling of the product is straight forward too, but we do some datacleaning, as some product were active but had a deactivation date, this data got deactivated instead.

For the filling of the date and timeofday dimension, it gets a bit more complicated. What we do there is to extract the information (year, month, day, hour, minute, second). When this is done, we generate a primary key by concatenating the information. So for date the primary key consists of year month day, and then for timeofday it is hour minute second.

In figure 4, the flow for filling the fact table can be seen. The figure depicts how the various surrogatekeys gets looked up in the dimension tables, as well as the price which is looked up in the old product table. After the lookups, the same generation of surrogate keys as in the filling of date and time dimensions are used, to get the same keys. Lastly, the mentioned erroneous memberids are filtered, such that members with an invalid memberid(0) gets changed to -1.

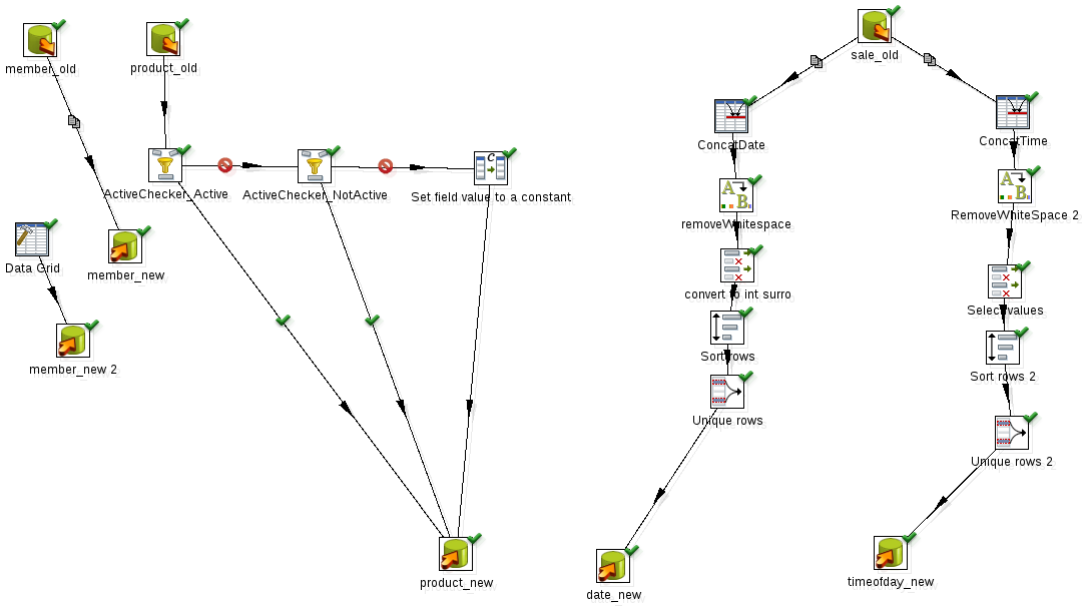

Figure 3: Flow for filling of dimensions

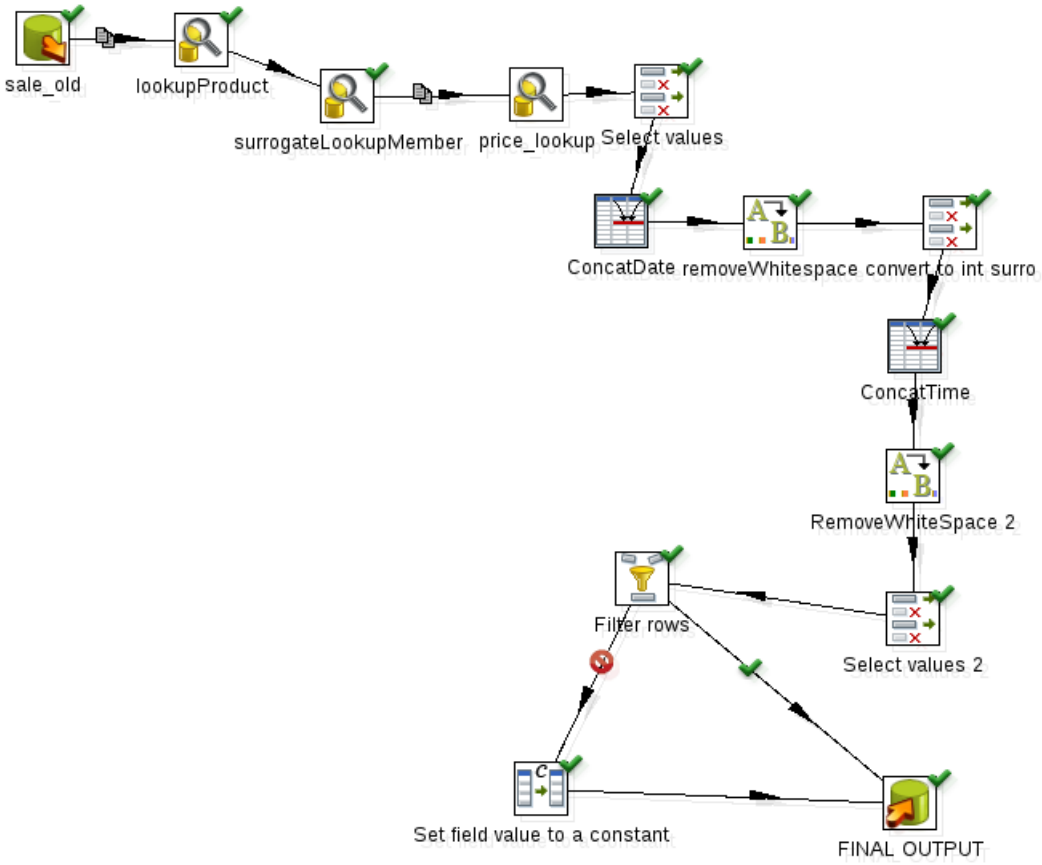

Figure 4: Flow for filling the fact table