Miniproject 1 - DIS

Lars Andersen, Mathias Winde Pedersen & Søren Skibsted Als

October 6, 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?

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.

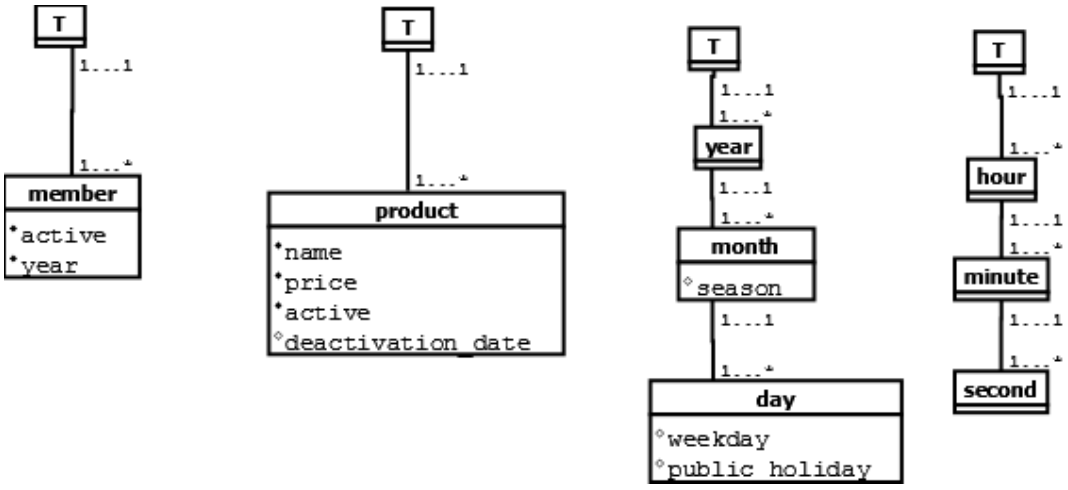

Figure 1: Dimensions, part C.

```
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)
);
```

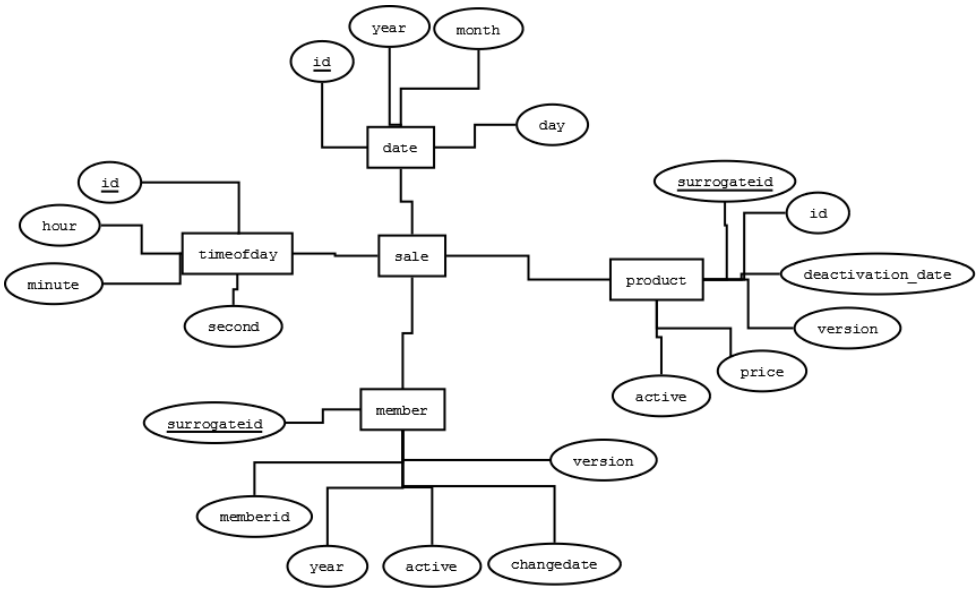

Figure 2: schema, part C.

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.

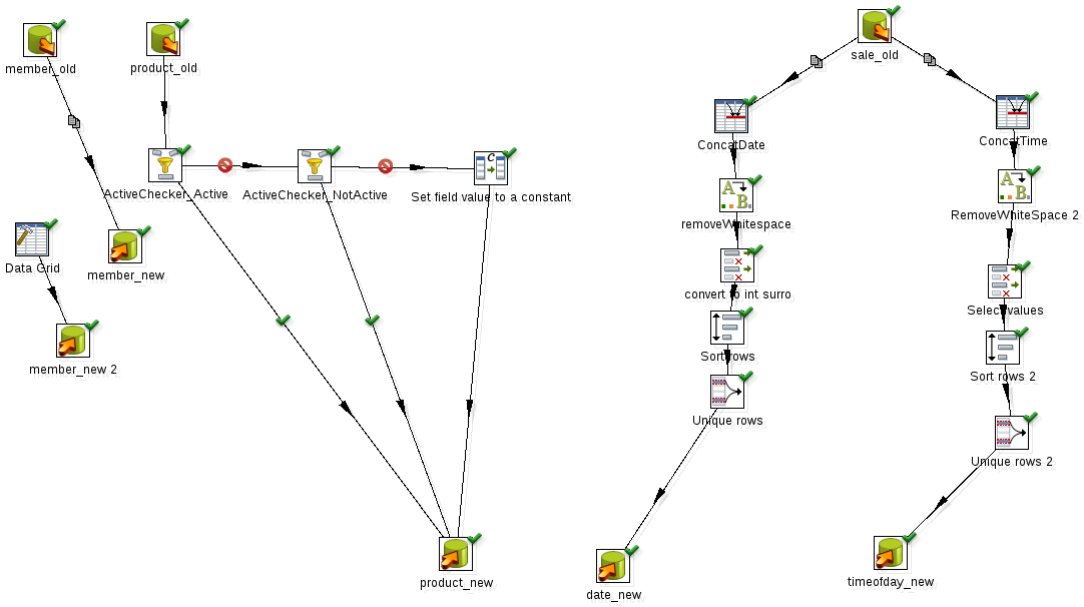

Figure 3: Flow for filling of dimensions

E

For this part, the work consists of writing the xml file specifying the cube. This can be seen in Figure 5 The xml file consists of the definitions of the dimensions first, whereafter the cube is defined, referencing the defined dimensions. For date and time, a default TimeDimension type is used, as this can specify the wanted hierachy.

F

Do note for this section that the price is counted in danish "oere" so you have to divide by 100 to get the sale in DKK. In section B we wanted to see if sales increase during break times, and it can be seen that Figure 6 confirms this.

Next we examine the total sales per year, seen in Figure 7. The sudden drop in 2008 is probably because the whole year has not been logged there yet. As can be seen, the sales have increased almost for each year, with a slight drop in 2003, 2005, and 2007, but this is not by a lot, as the overall tendency is an increase in sales.

We also take a look at which product has been sold the most, see Figure 8. As can be seen, soda has been sold the most, with beer on a second place.

This rounds up some interesting analysis that can be performed, and shows that the model works.

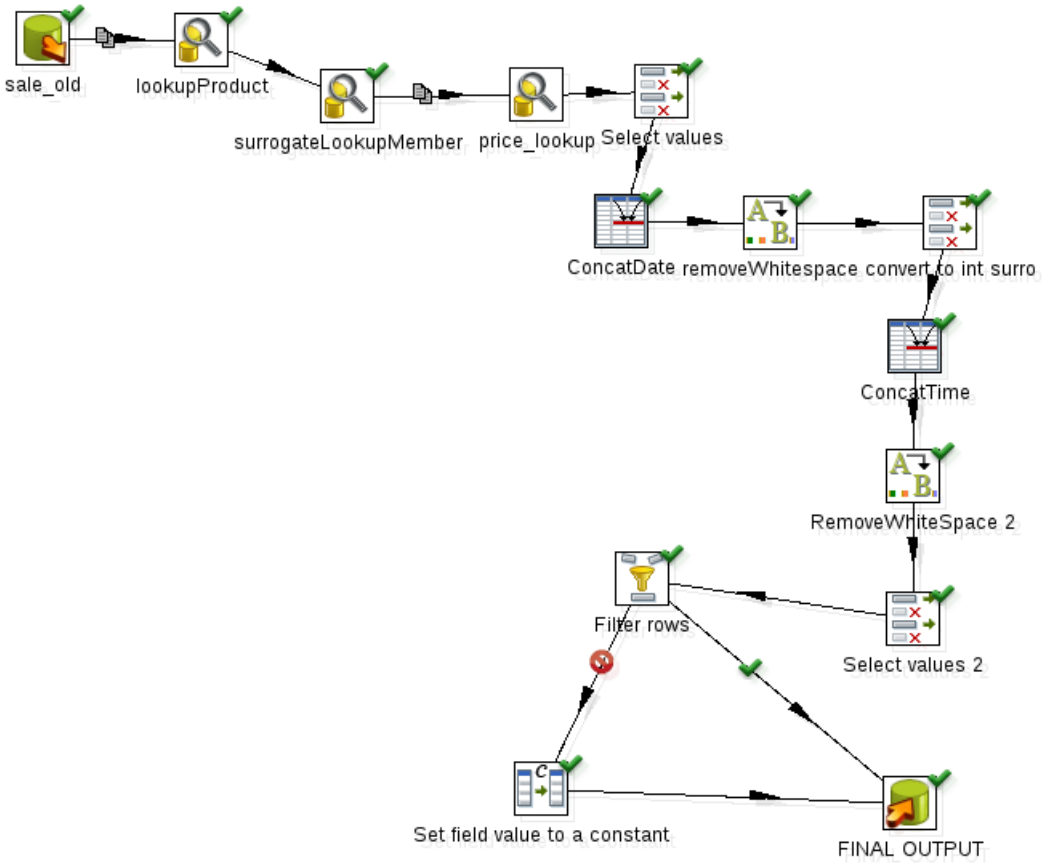

Figure 4: Flow for filling the fact table

```
<?xml version="1.0"?>
<Schema name="FKlubDW">
 <Dimension name="member">
   <Hierarchy hasAll="true" allMemberName="All Members"
                primaryKey="surrogateid">
     <Table name="member"/>
     <level name="memberid" column="memberid" uniqueMembers="false"
                type="Numeric"/>
     <Property name="active" column="active" type="Numeric"
                dependsOnLevelValue="true"/><--property example-->
   </Hierarchy></Dimension>

  <Dimension name="product" type="StandardDimension">
   <Hierarchy hasAll="true" allMemberName="All Products"
                primaryKey="surrogateid">
     <Table name="product"/>
     <level name="productid" column="id" type="Numeric"/>
   </Hierarchy></Dimension>

 <Dimension name="date" type="TimeDimension">
  <Hierarchy hasAll="true" allMemberName="All Periods" primaryKey="id">
    <Table name="date"/>
    <Level name="Year" column="year" uniqueMembers="true"
                levelType="TimeYears" type="Numeric"/>
    <Level name="Month" column="month" uniqueMembers="false"
                ordinalColumn="month" levelType="TimeMonths" type="Numeric"/>
    <Level name="Day" column="day" uniqueMembers="false"
                ordinalColumn="day" levelType="TimeDays" type="Numeric"/>
  </Hierarchy></Dimension>

  <Dimension name="time" type="TimeDimension">
  <Hierarchy hasAll="true" allMemberName="All Times"
        primaryKey="id">
    <Table name="timeofday"/>
    <Level name="hour" column="hour" uniqueMembers="true"
                levelType="TimeHours" type="Numeric"/>
    <Level name="minute" column="minute" uniqueMembers="false"
                ordinalColumn="minute" levelType="TimeMinutes"
                        type="Numeric"/>
    <Level name="second" column="second" uniqueMembers="false"
                ordinalColumn="second" levelType="TimeSeconds"
                        type="Numeric"/>
  </Hierarchy></Dimension>

 <Cube name="Sales">
   <Table name="sale" />
   <DimensionUsage name="member" source="member" foreignKey="memberid" />
   <DimensionUsage name="product" source="product" foreignKey="productid"/>
   <DimensionUsage name="date" source="date" foreignKey="dateid"/>
   <DimensionUsage name="time" source="time" foreignKey="timeofdayid"/>
   <Measure name="price" column="price" aggregator="sum"
        formatString="Standard"/>
 </Cube>
</Schema>
```

Figure 5: Definition of Cube

hour	price
0	1,209,000
1	1,133,800
2	689,550
3	206,800
4	128,175
5	86,400
6	134,975
7	994,050
8	9,635,525
9	14,906,350
10	18,870,375
11	17,244,700
12	28,263,000
13	20,420,575
14	17,754,925
15	14,919,275
16	9,990,850
17	5,664,975

Figure 6: Sales increase during break times

Year	price
1996	188,200
1997	2,359,375
1998	13,696,025
1999	15,180,275
2000	15,763,800
2001	16,402,075
2002	19,679,050
2003	18,017,550
2004	19,876,325
2005	17,882,675
2006	20,225,450
2007	19,932,050
2008	61,650

Figure 7: Sales per year

name	price
½L Vand excl. pant	66,970,550
Øl	20,272,250
½L Matilde cacao	19,476,750
Kaffe/Choko(1 kop)	11,578,400
Juice	8,738,675
F-Julefrokost	5,233,500
Kandidat betaling	5,085,000
½L drikkecultura	4,445,650
150ml Cultura	4,244,050
Guldøl & Juleøl	4,054,100
Juleøl/Påskeøl	3,793,900
Hyttetur	3,445,300
Cocio	3,409,475
½L Letmælk	3,040,250
½L Vand incl. pant	2,269,200
Kandidatfest	2,147,000
150ml Yoghurt	1,572,425
Ale No. 16	1,101,400

Figure 8: Sales per product