

UNIVERSITEIT TWENTE.

# Data Science [201400174]

Course year 2019/2020, Quarter 2A

DATE

February 4, 2020

EXCERPT

## Data Preparation and Visualization [DPV]

### TEACHERS

Maurice van Keulen  
Christin Seifert  
Mannes Poel  
Karin Groothuis-Oudshoorn  
Elena Mocanu  
Faiza Bukhsh  
Nicola Strisciuglio

### COURSE COORDINATOR

Christin Seifert  
Maurice van Keulen

### PROJECT OWNERS

Faiza Bukhsh  
Karin Groothuis-Oudshoorn  
Maurice van Keulen  
Elena Mocanu  
Mannes Poel  
Michel van Putten  
Mohsen Jafari Songhori  
Luc Wismans



# Data Preparation and Visualization [DPV]

## 1.1 Introduction

Data Warehousing, OLAP and Data Visualization are in essence technologies developed for Business Intelligence. They are, however, also effective for data science. The topic will teach (a) data warehousing techniques for extracting and transforming data (ETL), (b) modeling data for analytic purposes using the multidimensional modeling approach of OLAP, and (c) data visualisation techniques.

This topic comes in two flavours: you can do it in a *tool-based* fashion as well as in a *programming language-based fashion* (using the programming language **R**).

The topic uses open source or otherwise (temporarily) free to use tools to accomplish these tasks. The method and working of these tools are representative for what is used in practice, both in the fields of Business Intelligence as well as Data Science.

### 1.1.1 Global description of the practicum and project

The process of obtaining visualisations from raw data follows four steps:

1. *Determine the business questions.*
2. *Design a data warehouse,*  
i.e., design a database schema that can answer the business questions based on the multidimensional modelling method. You can in principle use any conceptual data modelling tool for this, but we do this by hand (i.e., with pen and paper). By setting up a database (we use PostgreSQL Server as DBMS) with this schema, you have obtained the *structure* of the data warehouse suitable for analysis ... but no data yet.
3. *Prepare data and fill the data warehouse,*  
i.e., extract relevant data from the raw sources, transform it, clean it, and store it in the database based on the principles of ETL. You could in principle do this with a self-written program and SQL statements, but we advocate a kind of visual programming for ETL similar to how many industrial tools work; the tool is called Pentaho ETL Community Edition also known as Kettle or Spoon. After this step, one has obtained a *filled* data warehouse.

#### 4. *Visualize the data,*

i.e., use a visualisation tool (we use **Tableau**) that connect to the data warehouse and presents the data in graphs that effectively answer the business questions.

The practicum assignments follow a different order: one first practices with steps 3 and 4, and then with step 2. The assignments of the two flavours tool-based and R-based are essentially the same, but you simply carry them out in a different manner. Therefore, either make the assignments in Section 1.3 *or* the ones in Section 1.4 for the respective flavour you have chosen. Information on the data sets and database which are used for both flavours, can be found in Section 1.2.

If you decide to apply this technology in a project, you are advised to follow the four steps for the given data set. You can distinguish yourself by formulating original business questions, by paying extra attention to visualisation or cleaning, or by constructing complex transformations that dig deeper in the data.

### 1.1.2 Study material and tools

- Data Warehousing Book - "Multidimensional Databases and Data Warehousing", Christian S. Jensen, Torben Bach Pedersen, and Christian Thomsen
- Data Warehouse: PostgreSQL Server (pre-installed on a remote server)
- Database administration: PhpPgAdmin (pre-installed on the same remote server; web-based user-interface)
- ETL / data transformation and cleaning:
  - Tool-based flavour: Pentaho Data Integrator (also called Kettle and Spoon)
  - R-based flavour: R and R-studio
- Data visualization: Tableau Public

### 1.1.3 Deliverables and obligatory items

Topic teachers: Maurice van Keulen (general and tool-based part) and Karin Groothuis-Oudhoorn (R-based part).

The practicum assignments specify what should be delivered. You are asked to include all those deliverables of all assignments of DPV into *one PDF file*. Submit the PDF-file to Canvas after you have completed all DPV assignments.

Tips:

- Whatever you have written with pen on paper, simply take a picture of it and include that.
- Kettle/Spoon can export an ETL workflow as .jpg
- For R, include the program code in the PDF
- Tableau can also export a dashboard visualisation as .jpg, but you may also make screenshots, of course.

## 1.2 Data set and database

The topic comes in two flavours: tool-based and R-based. The assignments of the two flavours tool-based and R-based are essentially the same, but you simply carry them out in a different manner. Both use the same data sets and store the cubes on the same database server. This section gives you information these.

For the practicum assignments, we use CSV-files (BI\_Raw\_Data.csv and SuperSales.zip) containing orders to a warehouse in the US. The files can be found in the Canvas site. The attributes in BI\_Raw\_Data.csv have the following meaning (the ones in SuperSales.zip are a straightforward extension):

attribute	description
Order_ID	Unique identifier for the order. Since one order can have more than one order line, i.e., one order can contain purchases for more than one product, there are multiple rows with the same Order_ID.
Order_Date_Year	Year of the order
Order_Date_Month	Month of the order
Order_Date_Day	Day of the order
Customer_Name	Name of the customer who placed the order
Customer_Country	The customer's country
Product_Name	The name of the product being ordered
Product_Category	The category of the product
Order_Price_Total	The total amount of the order
Product_Order_Unit_Price	The price of one unit of the product
Product_Order_Quantity	The number of product units ordered
Product_Order_Price_Total	The total price of the purchase of all units of that product, i.e., the multiplication of the previous two values

You will use a DBMS (database management system) to store and share your cubes. A DBMS is software that runs in the background and manages tables with data (the cubes you create are essentially also stored as tables with data). All other tools (including Kettle and Tableau) *connect* to this software to get access to and manipulate the data. A DBMS has already been pre-installed for you: PostgreSQL 10.10 running on server `bronto.ewi.utwente.nl`.

Each group has their own database on this server. You can obtain a database for your group as follows (only one person of the group needs to do this to obtain a database for the whole group).

- Go to DAB: <http://bronto.ewi.utwente.nl/dab>
- If you do not have an account in DAB, create one with “Register here”. You need to use your student email address.  
**NB:** there is a known bug with the system that it sometimes produces an error if you click on “Register here”. The problem is that there is a “dab” missing in the URL: please change it from <http://bronto.ewi.utwente.nl/register> to <http://bronto.ewi.utwente.nl/dab/register>
- Sign in and choose the course “ds19202a” which stands for “Data Science 2019/2020 quartile 2A”.
- Fill in your group number and click on “Get credentials”.
- A database will be created for you and the system provides you with the credentials: a username (which is the same as the database name) and a password.
- For your convenience, your database already has three *schemas*: `ass2` and `ass3` for assignments 2 and 3, and `project` for use in the project. In this way, you can keep the tables for these apart.
- You can always return to DAB to look at your credentials again and to reset your database. **Warning:** you loose everything in the database when you reset it, i.e., for all schemas in the database.

The same server also runs a web-based database administration tool, called PhpPgAdmin. You can access it with this link: <http://bronto.ewi.utwente.nl/phpPgadmin>. If you login with the credentials you obtained from DAB, you will see your database and the three schemas (there is also a fourth; just ignore it). You can easily create, inspect and drop the tables in your database with this tool<sup>1</sup>

## 1.3 Description of the practical assignments - Tool-based

**NB:** The assignments in this section pertain to the flavour *Tool-based*. If you have experience with programming or are interested to start learning how to program, you can also decide to choose the *R-based flavour*. These are essentially the same assignments, but carried out with the programming language R. If you choose the R-based flavour, make the assignments of Section 1.4 instead.

<sup>1</sup>You can also use any other *PostgreSQL client* such as PgAdmin (host = `bronto.ewi.utwente.nl`; port = 5432).

Events				
dayofweek	year	nrofevents	totalparticipants	location
Friday	2016	2	15000	Euclideslaan
Friday	2016	1	5000	Domplein
Tuesday	2016	5	25000	Domplein
Wednesday	2016	5	30000	Domplein
Tuesday	2016	6	25000	Muntstraat
⋮	⋮	⋮	⋮	⋮

Figure 1.1: Example of a one-table cube containing data on events happening in the city of Utrecht.

### 1.3.1 Tools

The tools that you would need to install for the tool-based assignments are:

- Pentaho Data Integration / Kettle (ETL) community edition
- Tableau BI Visualiser

See Section 1.3.6 for solutions to common problems including installation problems.

#### Installation Pentaho Kettle

- Download Pentaho Data Integration / Kettle Community Edition (version 6.0 or higher) from <http://community.pentaho.com/> → “Data integration / Kettle”. **NB: Install the Community Edition, not a trial of the Business version!**
- Extract ZIP-file in any directory.
- Start “spoon.sh” (Mac/Unix) or “spoon.bat” (Windows)
- Create a new “Transformation” (“File” → “New” → “Transformation”).
- Create a new “Database Connection” (“File” → “New” → “Database Connection”).  
Connection type: PostgreSQL / Access: Native (JDBC) / Host Name: bronto.ewi.utwente.nl / Database Name: *same as user name* / Port Number: 5432 / User Name: *your user name* / Password: *your password*.  
Click on “Test” to see whether or not it is working properly.

The installation instructions above do not produce a working tool for everyone. Please see Section 1.3.6 for some common problems and how to solve them.

#### Installation Tableau

- Get an academic license for a year at <http://www.tableau.com/academic/students> or a trial version at <http://www.tableau.com/products/desktop> (version 9.2 or higher) and install it as instructed.
- Connect ... to a server ... PostgreSQL. You are likely to get an error message complaining that a driver is missing. Simply download the driver(s) from the specified place and install it. Then try again to connect.

### 1.3.2 Assignment 1: Facts and dimensions

*This is an on-paper assignment. You don't need any tools for it.* <sup>2</sup>



**1.1** See Figure 1.1 which contains data on events happening in the city of Utrecht. We illustrate the meaning of the data by explaining what the first row means: there were in total 2 events organised in Euclideslaan on a Friday in 2016 which both together drew 15000 participants.

- Which attributes are the facts; which attributes are the dimensions?
- Draw a conceptual star schema for this cube.

<sup>2</sup>Both examples are simplified versions of actual data from <http://data.overheid.nl>.


Institutes			Cities		
instID	institute	cityID	cityID	city	province
1	University of Twente	153	153	Enschede	Overijssel
2	Technical University Eindhoven	772	546	Leiden	Zuid-Holland
3	Leiden University	546	772	Eindhoven	Noord-Brabant
4	Maastricht University	935	935	Maastricht	Limburg
5	Transnational University Limburg	935	:	:	:
:	:	:	:	:	:

Registrations					
instID	study	Phase	year	sex	nrofregistrations
1	Technical Computer Science	Bachelor	2015	Female	13
1	Technical Computer Science	Bachelor	2015	Male	214
1	Technical Computer Science	Bachelor	2016	Female	22
1	Technical Computer Science	Bachelor	2016	Male	270
1	Computer Science	Master	2016	Female	23
1	Computer Science	Master	2016	Male	126
2	Technical Computer Science	Bachelor	2015	Female	19
2	Technical Computer Science	Bachelor	2015	Male	216
2	Technical Computer Science	Bachelor	2016	Female	30
2	Technical Computer Science	Bachelor	2016	Male	280
:	:	:	:	:	:

Figure 1.2: Example of a multi-table cube containing data on the numbers of registered students in The Netherlands

□

 **1.2** See Figure 1.2 which contains data on the numbers of registered students in The Netherlands. We illustrate the meaning of the data by explaining what the first row of table “Registrations” means: In 2015, there were 13 female students registered for the bachelor study “Technical Computer Science” at the University of Twente which is located in the city of Enschede in the province of Overijssel.

- Which attributes are the facts; which attributes are the dimensions?
- Draw a conceptual star schema for this cube.

□

**Deliverable:** Submit the answers to all DPV assignments as *one PDF file*. For this assignment, include your answers of course, which may also be a scan of handwritten answers.

### 1.3.3 Assignment 2: Re-create the demo from the lecture

Use the data spreadsheet **BI\_Raw\_Data.csv**.

A large warehouse located in the US has many customers for its different product from around the world. The warehouse manager Mr Jack Bezos has many large customers from over the world ordering different products from his warehouse. We follow the method of Section 1.1.1 where all steps are more or less already given (and demoed at the lecture).

#### Step 1: Business questions

Mr. Bezos wants to know answers for the following questions

- Who are his top-5 most valued customers?
- What are his top-5 most important products?

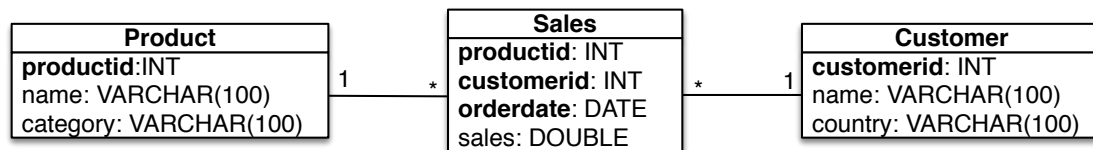


Figure 1.3: Starschema for Mr Bezos' warehouse. Primary key in **bold** (if more attributes are in bold, they are *together* a primary key).

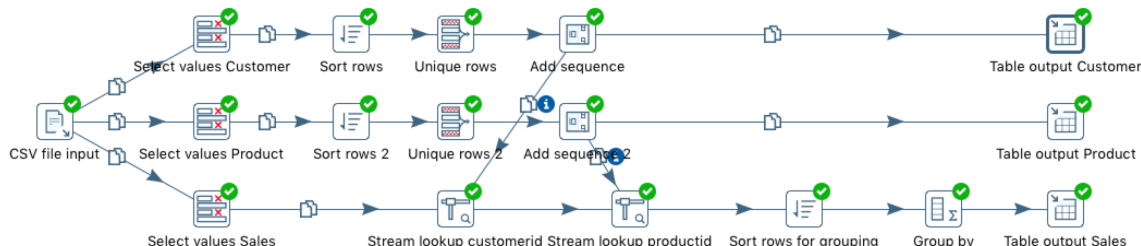


Figure 1.4: Kettle/Spoon ETL-flow for Mr Bezos' warehouse

### Step 2a: Multidimensional model

Notice that Mr. Bezos wants to know something about customers and products. These are *dimensions*. Time usually also is a dimension if you are interested in trends, so we also include the dimension 'orderdate'.

What is it he wants to know? It is 'most valued' and 'most important', but what does this mean concretely? How can we *measure* value and importance? Obviously in terms of money: the most valued customer is the one who bought for the most money, and the most important product is the one which was sold for the most money. Therefore, our *fact* is 'amount' (of money), i.e., 'sales'.

### Step 2b: Create tables in your database

We create a database schema for this multidimensional model given in Figure 1.3. There is already an empty *schema* in your PostgreSQL database with the name "ass2" intended to hold the tables for this assignment. We use the web-based database management tool PhpPgAdmin to create the tables and attributes in that schema. Open your browser at <http://bronto.ewi.utwente.nl/phpPgAdmin> to access the tool. Click on "PostgreSQL" under "Servers" to login and expand the panel on the left. You can create tables by navigating to "Schemas" → "ass2" → "Tables".

The database consists of three tables:<sup>3</sup>

- 'Customer' with attributes customerid:integer (primary key, NOT NULL), name:character varying(100), country:character varying(100)
- 'Product' with attributes productid: integer (primary key, NOT NULL), name:character varying(100), category:character varying(100)
- 'Sales' with attributes orderdate:date (primary key, NOT NULL), customerid:integer (primary key, NOT NULL), productid: integer (primary key, NOT NULL), sales:double precision

**Note:** The orderid is *not a key*! A customer can order more than one product in one order which results in several rows for one order. Furthermore, in its purest form, a cube does not store information on individual transactions or cases, but only aggregated data for each combination of dimension values. In the case of Mr Bezos, the cube does not store individual orders, but the sales (fact) for all combinations of the dimensions orderdate, customerid, and productid. In other words, the orderdate, customerid, and productid together are the primary key!

<sup>3</sup> Although usually a good thing, we do not create any foreign key declarations, because that will hinder you in re-running your data transformations.





Figure 1.5: Alternative Kettle/Spoon ETL-flow for Mr. Bezos' warehouse

### Step 3: ETL — Prepare data and fill the database

We use the Pentaho Data Integration tool called Kettle or Spoon to *extract* the data from the .csv file, to *transform* it into three output tables (one fact table and two dimension tables), and to *load* it into the database we just created.

Create a Kettle ETL flow similar to the one from Figure 1.4 and execute it. If something goes wrong, you can inspect the Tab “Logging” for error messages. Use the Tab “Preview data” to inspect the intermediary results of each step.

Some advice on ETL construction method

- *Small do-test steps*  
Do: Add only one or two small bits, then execute and verify the result, before continuing.  
Do not: Add many steps and then don't know where the mistake is when you receive an error.
- *Read the error message carefully*  
It may contain a lot of gibberish you don't understand, but part of it may provide clues to what is wrong.
- *GIYF: Google Is Your Friend*  
You may think Googling is not academic, but the internet is full of information on what may have caused certain errors and what you can do to fix them.
- *Verify*  
Check your table contents with phpPhAdmin (Browse) to verify that the data sent by the “Table output” really arrived properly in the tables of your database.

Kettle supports more powerful components than the ones of Figure 1.5, that can perform frequently used patterns of operations in one go. To get a better idea of such components, also create a Kettle ETL flow similar to one in Figure 1.5. It does exactly the same thing, but it uses the special “Combination Lookup / Update”, instead of more basic components such as “Sort rows”, “Unique rows”, “Get Value from Sequence”, and “Stream Value Lookup”. Try it out as well and inspect the intermediary results of each step using “Preview data”.

### Step 4: Visualize

Now that we have our data ready in a form suitable for analysis, it is time to address the two business problems raised in the beginning. We create a dashboard with metrics to measure the required KPIs. We use the visualization tool Tableau for this.

Connect it to the PostgreSQL database. For this you have to choose PostgreSQL and then you get a pop-up menu where you can select the server ('bronto.ewi.utwente.nl'), the database ('name of your database'), username and password. If all is well, you should see the three tables you created. Drag them to the top area starting with the fact table “Sales”, then the other two. Notice that Tableau automatically recognizes how the relationships between the tables are: a benefit from a database schema conforming to a multidimensional model. Click “Update now” to read the data into Tableau (or use “Update automatically”).

To start our dashboard, click on “Sheet 1”, and drag Customer.Name in Y-axes and Sales.Sales in X-axes. Then click “Sort name ascending by sales”. You now have a visualization that answers the first business question about the most valued customer. Make another one for product answering the question about most important product.

**Deliverable:** Include in your PDF file the following

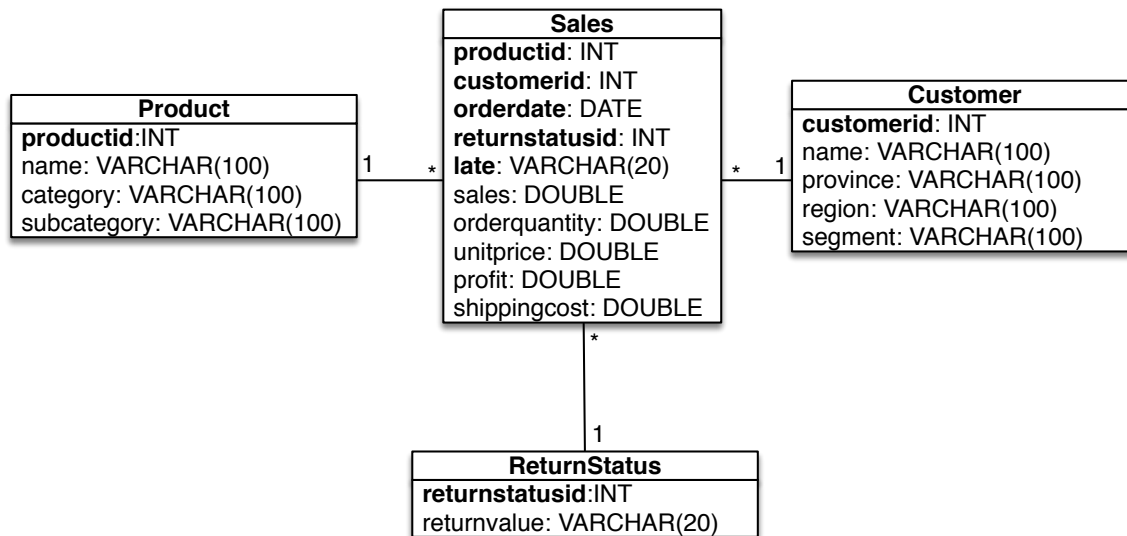


Figure 1.6: Starschema for SuperSales

- Include a screenshot of your Spoon/Kettle transformation
- Export the dashboard visualizations as JPG and also include those as pictures in the PDF.

### 1.3.4 Assignment 3: Do it yourself

Use the spreadsheets in **SuperSales.zip**

We follow the method of Section 1.1.1 again, but now only steps 1 and 2 are given; the assignment is to do steps 3 and 4 yourself.

#### Step 1: Business questions

For this new warehouse in Canada, Mr. Bezos wants to know answers for the following questions

- Which products/product categories made the most loss?
- Which products/product categories were shipped really late (more than 2 days)?
- Which products/product categories were returned the most?

#### Step 2: Multidimensional model and database structure

Re-create the star schema of Figure 1.6 for Mr. Bezos to answer his business questions. We require that

- The dimension “ReturnStatus” has a separate table containing two rows representing the only two values “Returned” and “NotReturned”.
- The dimension “Late” is an inlined dimension representing the only two values “Late” and “NotLate”.

#### Step 3: ETL — Prepare data and fill the database

The main goal of this assignment is that you do this step yourself.

**Note:** There is an important complication in this case. Notice that the star schema models a proper cube in the pure sense of multidimensional modeling. For each combination of dimension values, there should be exactly one row in the fact table. Suppose that the same customer orders the same product twice on the same

day and both products are not late and not returned. In other words, all dimensions are the same for these two orders. Therefore, we need to combine the facts of these two orders, so that we end up with exactly one row for this particular combination of dimension values.

As you know from assignment 2, the suggested way of accomplishing this is by using the Kettle component “Group by”. Note that ‘combine fact’ may mean different things for different facts: sales, profit, order quantity and shipping cost should be summed, but unit price should not.

**Tips:** There is more than one CSV-file for which the information needs to be combined. A handy component to accomplish such a combination is “MergeJoin”. The rows of two streams are joined if they have the same value(s) for one or more key attributes (e.g., “OrderID”). Find out yourself what ‘join type’ means and which one you need here.

For the ‘Late’ dimension you need to do some calculations. Handy components for doing calculations, i.e., adding a new attribute with the result of a calculation, are “Calculator” and “Formula”.

#### Step 4: Visualization

Realize a dashboard that answers these business questions.

**Deliverable:** Include in your PDF file the following

- Include a picture of your multi-dimensional model
- Include a screenshot of your Spoon/Kettle transformation
- Include pictures of your dashboard visualization(s)

### 1.3.5 Assignment 4: multidimensional modeling. Case “Mobile app beta tester service”

*This is an on-paper assignment. You don’t need any tools for it.*

The company *Pear* sells smartphones. For downloading and updating mobile applications they have the “Pear app store”. Pear management has a new business idea: a beta tester service for game developers. The purpose of beta testing is mainly for getting feedback on game play: do you become bored too quickly, is it challenging enough, etc. The purpose is not so much finding bugs. Good gamers can offer themselves as beta tester and developers can hire them. Developers pay Pear for a subscription plus a fee for hiring a beta tester of which a small percentage goes to Pear.

A subscription includes support for developers in finding the “best” beta testers for their particular application. Pear needs to realize a data warehouse for this purpose with all gaming data of the beta testers. They have in their databases timestamps of start of game app, switch to other app, and relevant events (turn, next level, completion of game, etc.) as well as scores of completed games, and data on the apps: name, solitary/multiplayer, name of developer, a fine-grained category, etc.

(you may assume more data to be available if it is reasonable a company as Pear would have it; make these assumptions explicit)

A beta tester receives a small amount of money for joining in exchange for allowing Pear to disclosing their data to developers. A second purpose of the data warehouse is that during a beta test, Pear provides the developer with information on how much time the beta tester has devoted to beta testing the game.

- (a) For being able to determine the right fact(s) for the star schema an important question is “What is *good*”, i.e., “What makes a particular gamer a good beta tester for a particular kind of game?” More concretely, how can you determine a score that quantifies this ‘goodness’ or ‘suitability’, which obviously needs to be calculated from the available data.  
Propose a formula for “goodness score” and explain why a high value is an indicator for a good beta tester.
- (b) What is the business question, or what are the business questions in this case? Formulate them as accurately as possible.

- (c) Give a star schema. Explain your design by describing the most important design choices and considerations.

**Deliverable:** Include in your PDF file the answers to the questions. Also include a picture of the star schema. It may be a scan or handwritten answers.

### 1.3.6 Trouble shooting

**Running “spoon.sh” or “spoon.bat” doesn’t work** Run “spoonDebug” instead. It first asks a few questions: answer Y, Y, and N. Then it will show an error message.

- If the error message refers to not enough memory, then edit the “spoon.sh” or “spoon.bat” and change the “-Xmx2048m” into “-Xmx1024m”.
- If the error message refers to “No path found” or “unsupported major.minor number”, then Java is not installed or a too old version of Java is installed. First download and install the latest version of “Java SE Runtime Environment” from Oracle’s website (for Windows choose x64 and exe; for Mac OSX choose dmg). If you still get a “No path found” on Windows then Pentaho is unable to find your Java installation. Do the following
  1. Right-click This pc, select properties
  2. Click Advanced system settings
  3. Click Environment Variables
  4. Click the upper New button
  5. Fill in the variable name with “JAVA\_HOME”
  6. Click “Browse Directory”
  7. Go to Program files and find the java “jre...” directory, select this and click ok
  8. Click ok and launch spoon.bat

**Null pointer exception when trying to draw a line between two components** This is a known bug of Pentaho Kettle that sporadically, but consistently occurs. Restarting and doing it again doesn’t work. A workaround that does seem to work, however, is drawing the line (called ‘hop’ in Kettle) the other way around, so from the target to the source.

**The attribute names from the database are not correctly displayed in the dialogue window for the Table Output component** This is also a known bug that sometimes happens. Kettle doesn’t seem to have loaded the table and attribute names from the database. The “Table Output” component then displays the attribute names from the ‘flow’ instead. If this occurs, you simply type them in yourself (exactly). You do not need to select the name from the drop menu.

**Error “Duplicate entry ‘some number’ for key ‘PRIMARY’ ”** This means that Kettle is trying to insert a row in the database with a value for the primary key that already exists. This typically occurs in two different situations:

- Your primary key is actually not a primary key, because it is not unique. For example, the orderid is not unique, because a customer may buy several products in the same order, which will produce two rows with the same orderid. In this case, change your schema such that another attribute or combination of attributes is the primary key (in our example, the combination orderid and productid).
- If you run Kettle a second time, the data of the previous run is still in the database. This second run will try to re-insert the same tuples, which the DBMS refuses because it will produce different rows with the same value for the primary key. You could empty the database every time, but there is an easier solution. The ‘table output’ component of Kettle has an option “Truncate table”. Set it, and it will empty (truncate) the table each time the flow is run.

**Error “Field ‘a’ doesn’t have a default value”** Every attribute of a table that have “NOT NULL” switched on should always have a value for each row. But a DBMS also has a facility to define a ‘default value’ that is filled in every time you try to store a NULL in the attribute.

The error usually occurs when you forget to specify the attribute ‘a’ in the Table Output component. Forgetting to specify it means that Kettle is not storing a value for that attribute, hence it stores NULL, hence the error message. Go to the “Database fields” tab of the Table Output component and make sure all attributes are specified.

**“binary string” vs. “normal string”** Sometimes you may encounter an error message complaining about “data type error” where you also see the mention of “String<binary-string>”. You must know that there are two types of string in Spoon: binary string and normal string. If you import data from a CSV-file using “CSV file input” component, then the string attributes are actually typed as binary string. Certain data type conversions or filters require string attributes to be normal string. So, if you see such an error message for a certain attribute, you can try to first convert the attribute to a normal string. You can use the “Select values” component for this: it has a “Meta-data” tab where you can set “Binary to Normal?” to “Y”.

**Conversion errors** Sometimes you may see “Conversion error: null” as value in a column when you inspect an intermediary result. When reading data using the “CSV file input” component, Spoon will attempt to postpone the conversion of values from string to other data types. So, if there is a column with a date in it, is not really used, the data will never be converted to a date data type. But, as soon as Spoon needs to compare values or has to perform an action that requires the values to be non-string data values, then it can no longer postpone and will convert the data values. This is called “Lazy conversion”.

If the conversion fails for some reason, then the error will occur at the step that required the conversion, while actually the problem is with “CSV file input”. There are also some known bugs that occur due to lazy conversion. Advice is: switch off the lazy conversion in this component.

## 1.4 Description of the practical assignments - R-based

### 1.4.1 Tools

The tools that you would need to install for the topic assignments are:

- R and Rstudio
- Tableau BI Visualiser

#### Installation R and RStudio

- Install the latest version of R from <https://cran.r-project.org>.
- Install the latest version of RStudio (desktop version, free license) from <http://www.rstudio.com/>.
- Open RStudio
- Go to File > new file > R script: you will see the file in the upper left part of Rstudio. In this file you can type syntax and run in later. Do not forget to save it once and a while.
- Go to the right lower pane and click on ‘Install’. Now you can install the libraries DBI, RPostgreSQL, readr, dplyr and lubridate.
- After installing these packages you need to activate them by e.g. type the following piece of R script, select it all and click on ‘Run’:

```
library(DBI)
library(RPostgreSQL)
library(readr)
library(dplyr)
library(lubridate)
```

- Instead of installing packages via the menu as described above you could have also used (e.g.): `install.packages("dplyr")`.
- By typing `help(<name>)` with the name of the function you want to get the helpfile from you will see the documentation of that function in the lower right corner of R-studio for the tab Help.
- It is advised to work in RStudio with projects. So make first a new directory on your laptop with the name 'datascience' (off course you can choose another name) and a subdirectory with the name 'data'. Then in RStudio goto File > New Project and select Existing directory and navigate to the just made directory 'datascience'. If you now open a new script file it will be automatically saved in that directory. Moreover, if you put the datafiles (e.g. BI\_Raw\_Data.csv) in the data subdirectory you need only to specify 'data/filename' when you need to import some data.
- For more background on R we refer to the book "R for Datascience", H. Wickham (see <http://r4ds.had.co.nz/>).

**Installation Tableau** Please follow the instructions for "Installation Tableau" from the 'Tool-based' section (see Section 1.3.1).

## 1.4.2 Assignment 1: Facts and dimensions

This assignment is the same for Tool-based and R-based. See Section 1.3.2 for the actual assignment.

## 1.4.3 Assignment 2: Re-create the demo from the lecture

Use the data spreadsheet **BI\_Raw\_Data.csv**.

A large warehouse located in the US has many customers for its different product from around the world. The warehouse manager Mr Jack Bezos has many large customers from over the world ordering different products from his warehouse. We follow the method of Section 1.1.1 where all steps are more or less already given (and demoed at the lecture).

### Step 1: Business questions

Mr. Bezos wants to know answers for the following questions

- Who are his top-5 most valued customers?
- What are his top-5 most important products?

### Step 2a: Multidimensional model

Notice that Mr. Bezos wants to know something about customers and products. These are *dimensions*. Time usually also is a dimension if you are interested in trends, so we also include the dimension 'orderdate'.

What is it he wants to know? It is 'most valued' and 'most important', but what does this mean concretely? How can we *measure* value and importance? Obviously in terms of money: the most valued customer is the one who bought for the most money, and the most important product is the one which was sold for the most money. Therefore, our *fact* is 'amount' (of money), i.e., 'sales'.

### Step 2b: Create tables in your database

We create a database schema for this multidimensional model given in Figure 1.3. There is already an empty *schema* in your PostgreSQL database with the name "ass2" intended to hold the tables for this assignment. We use the web-based database management tool PhpPgAdmin to create the tables and attributes in that schema. Open your browser at <http://bronto.ewi.utwente.nl/phpPgadmin> to access the tool. Click

on “PostgreSQL” under “Servers” to login and expand the panel on the left. You can create tables by navigating to “Schemas” → “ass2” → “Tables”.

The database consists of three tables:<sup>4</sup>

- ‘Customer’ with attributes customerid:integer (primary key, NOT NULL), name:character varying(100), country:character varying(100)
- ‘Product’ with attributes productid: integer (primary key, NOT NULL), name:character varying(100), category:character varying(100)
- ‘Sales’ with attributes orderdate:date (primary key, NOT NULL), customerid:integer (primary key, NOT NULL), productid: integer (primary key, NOT NULL), sales:double precision

**Note:** The orderid is *not a key*! A customer can order more than one product in one order which results in several rows for one order. Furthermore, in its purest form, a cube does not store information on individual transactions or cases, but only aggregated data for each combination of dimension values. In the case of Mr Bezos, the cube does not store individual orders, but the sales (fact) for all combinations of the dimensions orderdate, customer, and product. In other words, the orderdate, customerid, and productid together are the primary key!

### Step 3: ETL — Prepare data and fill the database

We use R/RStudio to extract the data from the .csv file, to transform it into three tables: one facts table ‘sales’ and two dimension tables ‘product’ and ‘customer’, and load it into the database. Open first a new syntax file to put your R code in. A useful package to work with tables is `dplyr`. With the package `readr` you can import data into R, e.g command separated files. So start with installing and loading both packages in R/RStudio.

```
library(readr)
library(dplyr)
```

Then, we load the data by using the command “`read_delim`” and put it into the object “data0”:

```
data0 <- read_delim(file = "data/BI_Raw_data.csv",
                    delim = ";", col_names = TRUE, col_types = NULL)

head(data0)
```

Use the function ‘`head`’ to inspect the first five rows of the imported data. You can neglect the fact that in the product or customer names some special characters are not imported correctly.

In Figure 1.5 you can see the ETL flow for preparing the data from the tool-based part. For example, for the product table you first need to select the columns from the data0 object that are attributes of products (with function `select()`), i.e. the name (`Product_Name`) and the category (`Product_Category`) then rename the column `Product_Name` to `name` and the column `Product_Category` to `category` with function `rename()`. In the original .csv file each row is a sales transaction so the products and category columns contain a lot of duplicates. In the products table you want to have only one row for each product/category combination and a productid, the primary key. So for each name and category combination you need to delete the duplicate rows, this can be done by first grouping with function `group_by()` the rows based on name and category and then use the function `distinct()`. Finally, you have to ungroup the data with function `ungroup()` and attach a new column with the primary key with variable name `productid` to the table with the function `mutate()` with `row_number()`.

An example of all steps to make the product table is as follows (with the pipe operator):

```
# Step 1: make Product table 'product':
product <- data0 %>%
  select(Product_Name, Product_Category) %>%
  rename(name = Product_Name, category = Product_Category) %>%
```

<sup>4</sup>Although usually a good thing, we do not create any foreign key declarations, because that will hinder you in re-running your data transformations.

```

arrange(name, category) %>%
group_by(name, category) %>%
distinct() %>%
ungroup() %>%
mutate(productid = row_number())

```

Check that you have now created a table `product` that contains the required columns and 77 rows. Similarly, you can make the customer table. For the sales table you first need to select the columns from the original data that you need and then join the product and customer table to add the `productid` and `customerid` to the sales table and finally you need to drop columns that are redundant in the sales table. In this way you can join the sales and product table (assuming that you have already created the sales table):

```

sales <- sales %>%
  full_join(product, by = c("Product_Name" = "name",
    "Product_Category" = "category")) %>%
  select( -Product_Name, -Product_Category)

```

Finally you have to fill the database. From R you can connect to the PostgreSQL database server with the functions `dbDriver`, `dbConnect` as follows (first install and load the packages `DBI`, `RPostgreSQL` (for schema `ass2`):

```

drv <- dbDriver("PostgreSQL")
con <- dbConnect(drv, port = 5432, host = "bronto.ewi.utwente.nl",
  dbname = "<name db>", user = "<username>", password = "<passwd>",
  options="-c search_path=ass2")
dbWriteTable(con, "product", value = product, overwrite = T, row.names = F)
dbWriteTable(con, "customer", value = customer, overwrite = T, row.names = F)
dbWriteTable(con, "sales", value = sales, overwrite = T, row.names = F)

```

Check now on the website <http://bronto.ewi.utwente.nl/phppgadmin/> that you have indeed filled the three tables in your database.

To get the info from the tables in the database you can also do that using R (assuming you have created the tables `customer`, `sales` and `product`):

```

dbListTables(con)
str(dbReadTable(con, "customer"))
str(dbReadTable(con, "sales"))
str(dbReadTable(con, "product"))

```

or if a table is in schema `ass2`:

```

dbGetQuery(con,
  "SELECT table_name FROM information_schema.tables
  WHERE table_schema='ass2'" ) ## to get the tables from schema ass2
str(dbReadTable(con, c("ass2", "sales")))

```

#### Step 4: Visualize

Now that we have our data ready in a form suitable for analysis, it is time to address the two business problems raised in the beginning. We create a dashboard with metrics to measure the required KPIs. We use the visualization tool Tableau for this.

Connect it to the PostgreSQL database. For this you have to choose PostgreSQL and then you get a pop-up menu where you can select the server (`'bronto.ewi.utwente.nl'`), the database (`'name of your database'`), username and password. If all is well, you should see the three tables you created. Drag them to the top area starting with the fact table “Sales”, then the other two. Notice that Tableau automatically recognizes how the relationships between the tables are: a benefit from a database schema conforming to a multidimensional model. Click “Update now” to read the data into Tableau (or use “Update automatically”).



To start our dashboard, click on “Sheet 1”, and drag Customer.Name in Y-axis and Sales.Sales in X-axis. Then click “Sort name ascending by sales”. You now have a visualization that answers the first business question about the most valued customer. Make another one for product answering the question about most important product.

**Deliverables:** Include in your PDF file the following

- Include the R source code
- Export the dashboard visualizations as JPG and also include those as pictures in the PDF.
- Include the output of the following R-code:

```
dbGetQuery(con,
            "SELECT table_name FROM information_schema.tables
            WHERE table_schema='ass2'" ## to get the tables from schema ass2
str(dbReadTable(con, c("ass2", "<table_name>")))
```

**NB:** Repeat the ‘str’ line for each table you have and fill in the table name where it says “<table\_name>”. In the ‘dbGetQuery’ line, you need not fill in anything, so write “table\_name” literally there.

### 1.4.4 Assignment 3: Do it yourself

Use the spreadsheets in **SuperSales.zip**

We follow the method of Section 1.1.1 again, but now only steps 1 and 2 are given; the assignment is to do steps 3 and 4 yourself.

#### Step 1: Business questions

For this new warehouse in Canada, Mr. Bezos wants to know answers for the following questions

- Which products/product categories made the most loss?
- Which products/product categories were shipped really late (more than 2 days)?
- Which products/product categories were returned the most?

#### Step 2: Multidimensional model and database structure

Re-create the star schema of Figure 1.6 for Mr. Bezos to answer his business questions. We require that

- The dimension “ReturnStatus” has a separate table containing two rows representing the only two values “Returned” and “NotReturned”.
- The dimension “Late” is an inlined dimension representing the only two values “Late” and “NotLate”.

#### Step 3: ETL — Prepare data and fill the database

The main goal of this assignment is that you do this step yourself.

**Note:** There is an important complication in this case. Notice that the star schema models a proper cube in the pure sense of multidimensional modeling. For each combination of dimension values, there should be exactly one row in the fact table. Suppose that the same customer orders the same product twice on the same day and both products are not late and not returned. In other words, all dimensions are the same for these two orders. Therefore, we need to combine the facts of these two orders, so that we end up with exactly one row for this particular combination of dimension values.

As you know from assignment 2, the suggested way of accomplishing this is by using the function `group_by()` and then `summarise()`. Note that combining facts may mean different things for different facts: sales, profit, orderquantity and shippingcost should be summed, but unitprice should not.

**Tips:**

- The dates in the CSV file are just characters, but to be able to calculate e.g. the number of days a product is too late you have to transform them into real dates. The function `dmy()` takes a string and transforms it into the right class. When you have obtained the `orderdate` and `shipdate` in the appropriate format you can calculate the number of days with: `interval(orderdate, shipdate)/ddays()`. Next you can use the function `if_else` to make a column with two values "Late" and "NotLate".
- There is more than one CSV-file for which the information needs to be combined. A handy function to accomplish such a combination is `full_join()`. The rows of two streams are joined if they have the same value(s) for one or more key attributes (e.g., "orderid").

#### Step 4: Visualization

Realize a dashboard that answers these business questions.

**Deliverable:** Include in your PDF file the following

- Include a picture of your multi-dimensional model / database schema
- Include the R source code
- Export the dashboard visualizations as JPG and also include those as pictures in the PDF.
- Include the output of the following R-code:

```
dbGetQuery(con,
            "SELECT table_name FROM information_schema.tables
            WHERE table_schema='ass3'" ) ## to get the tables from schema ass3
str(dbReadTable(con, c("ass3", "<table_name>")))
```

**NB:** Repeat the 'str' line for each table you have and fill in the table name where it says "<table\_name>". In the 'dbGetQuery' line, you need not fill in anything, so write "table\_name" literally there.

#### 1.4.5 Assignment 4: multidimensional modeling. Case "Mobile app beta tester service"

This assignment is the same for Tool-based and R-based. See Section 1.3.5 for the actual assignment.