

MODUL PRAKTIKUM

IS 545 – DATA WAREHOUSE
PROGRAM SARJANA S1 SISTEM INFORMASI
FAKULTAS TEKNIK DAN INFORMATIKA



**PROGRAM STUDI SISTEM INFORMASI
FAKULTAS TEKNIK DAN INFORMATIKA
UNIVERSITAS MULTIMEDIA NUSANTARA**

Gedung B Lantai 5, Kampus UMN
Jl. Scientia Boulevard, Gading Serpong, Tangerang, Banten-15811 Indonesia
Telp: +62-21.5422.0808 (ext. 1803), email: ict.lab@umn.ac.id, web: umn.ac.id

DAFTAR ISI

DAFTAR ISI	2
CAPAIAN PEMBELAJARAN PRAKTIKUM.....	3
MODUL 1 PENTAHU DATA INTEGRATION	4
MODUL 2 (PDI TRANSFORMATION: READING AND WRITING).....	9
MODUL 3 (PDI LOADING & LOOKING PUZZLES DIMENSION)	13
MODUL 4 (PDI: HIERARCHICAL RELATIONSHIP).....	19
MODUL 5 (LOADING DIMENSIONS WITH DATA).....	25
MODUL 6 (MONDRIAN OLAP CUBE).....	33
MODUL 7 (INTRODUCING PARTITIONING and CLUSTERING)	38
MODUL 8 (OPTIMIZING DATA FLOWS)	43
MODUL 9 (ETL/ EXTRACTION, TRANSFORMATION, AND LOADING)	48
MODUL 10 (DATA WAREHOUSE DESIGN: SAKILA DVD RENTAL)	54
MODUL 11 (ETL DEVELOPMENT LIFECYCLE).....	56
MODUL 12 (HANDLING DIMENSION TABLES)	58
MODUL 13 (LOADING FACT TABLES)	60
MODUL 14 (PRD: FUNDAMENTAL OF REPORTING).....	62

Capaian pembelajaran praktikum mata kuliah Data Warehouse, adalah mahasiswa memiliki kemampuan dan penguasaan teknik-teknik pembangunan serta pemeliharaan Data Warehouse menggunakan software Pentaho Data Integration, dengan beberapa kompetensi berikut ini:

CPMK01 : menguasai konsep Data Warehouse meliputi model dimensional, pengoperasian OLAP dan arsitektur Data Warehouse – C2;

CPMK02 : memahami dan mampu mengimplementasikan konsep desain Data Warehouse yang sesuai dengan tujuan bisnis/ organisasi – C3;

CPMK03 : menguasai teknik-teknik explorasi Data Warehouse meliputi standar bahasa MDX (MultiDimensional eXpressions) untuk multidimensi database –C5;

CPMK04 : mampu merancang desain fisik Data Warehouse yang dapat memberikan kinerja waktu respons untuk kueri ad-hoc kompleks yang ideal – C3;

CPMK05 : menguasai konseptual desain dan implementasi proses ETL yang sesuai dengan kebutuhan bisnis/ organisasi – C4,

CPMK06: mampu merencanakan pembangunan dan eksploitasi Data Warehouse dan prinsip pemodelan dimensi untuk mendukung proses pengambilan keputusan bisnis/ organisasi – C6.

MODUL 1

PENTAHO DATA INTEGRATION



DESKRIPSI TEMA

Pengenalan dan penggunaan tools Integrasi Data Pentaho, mencakup instruksi untuk instalasi PDI (Pentaho Data Integration) dan pengenalan penggunaan codingless programming menggunakan desain grafis dari Spoon.

CAPAIAN PEMBELAJARAN MINGGUAN (SUB-CAPAIAN PEMBELAJARAN)

CMPK01-1 (C2)

Mahasiswa mampu memahami gambaran secara umum terhadap penggunaan alat bantu/ tools dilingkungan Data Warehouse untuk selanjutnya melakukan penerapan transformasi data untuk kebutuhan multidimensi database dan OLAP.

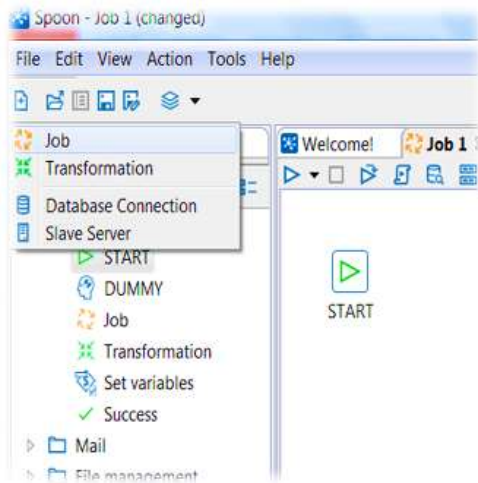
1. Installing PDI & Launching the PDI Graphical Designer – Spoon
2. Introducing transformations
3. Creating a Hello World! Transformation
4. Previewing and running a Transformation


PENUNJANG PRAKTIKUM

1. Windows Operating System
2. Java Runtime Environment (JRE) & Java Development Kit (JDK) 8.0 version (*installed*)
3. Pentaho Data Integration tools (PDI) release 7.1 atau 8.0.

LANGKAH-LANGKAH PRAKTIKUM

1. **Installing PDI & Launching the PDI Graphical Designer – Spoon**
 - a. Firstly, make sure your Windows Environment Variable named JAVA_HOME to point to the JRE, JAVA_HOME =: C:\Program Files\Java\jre1.8.0_144
 - b. Download page at [Pentaho from Hitachi Vantara - Browse /Data Integration at SourceForge.net](#)
 - c. Unzip the downloaded file in a folder of your choice, as, for example d:/IS545/PDI
 - d. Now that you've installed PDI, you're ready to start working with the data. PDI has a desktop designer and before start your spoon please execute the "set-pentaho-env.bat" by double-click.
 - e. Start Spoon, run "Spoon.bat" from within the PDI installed directory.
 - f. The main window shows up. The Welcome! window appears with some useful links provided.
 - g. Spoon is the tool with which you create, preview, and run transformations. The following screenshot shows you the basic work areas: Main Menu, Main Toolbar, Steps Tree, Transformation Toolbar, and Canvas (Work Area).

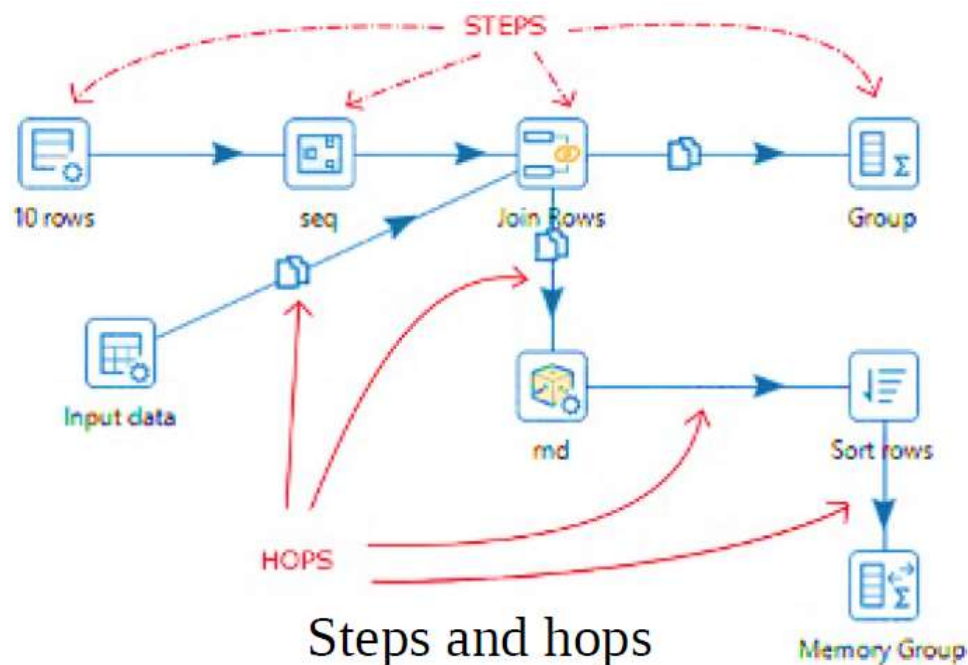


- **Job**, is an Kettle component to manage control flow of task (flow control)
- Create Job:
 - File | New | Job or press CTRL+ALT+N
 - Every Job must be starting by step of **Start**
- Job running by sequence and having configuration of Job Scheduling
- Jo icon in Spoon is 

- **Step**, is an Kettle module/component to do certain of task and specific in Job/Trasnformation
- Total Step in Kettle more than 100 standard Step (default Steps) and segmented into two categories:
 - **Job Steps** : Step running by sequence and more focused into organizing flow whole of process of ETL task
 - **Transformation Steps**: Step running by parallel and emphazise onto I/O data
- Reference of how to use various steps available on PDI/Kettle user guide by these url:
 - <http://wiki.pentaho.com/display/EAI/Pentaho+Data+Integration+Job+Entries> (job steps)
 - <http://wiki.pentaho.com/display/EAI/Pentaho+Data+Integration+Steps> (Transformation steps)
- We also could develop new step by plugin mechanism using of Java programming

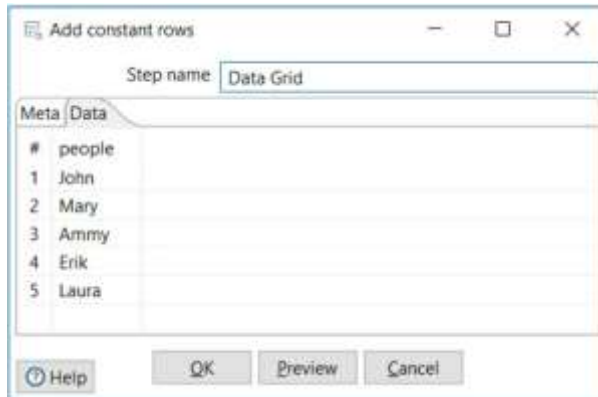
2. Introducing transformations

- A Transformation is an entity made of steps linked by hops. These steps and hops build paths through which data flows: the data enters or is created in a step, the step applies some kind of Transformation to it, and finally, the data leaves that step.
- Therefore, it's said that a Transformation is data flow oriented. Graphically, steps are represented with small boxes, while hops are represented by directional arrows, as depicted in the following sample:



3. Creating a Hello World! Transformation

- Open Spoon. From the main menu and navigate to File | New | Transformation.
- On the left of the screen, under the Design tab, you'll see a tree of Steps. Expand the Input branch by double-clicking on it.
- Then, left-click on the Data Grid icon and without releasing the button, drag and drop the selected icon to the main canvas.
- Double-click on the Data Grid step you just put on the canvas, and fill the Meta tab and fill the grid with some names, as in the following screenshot. Then click on OK to close the window:

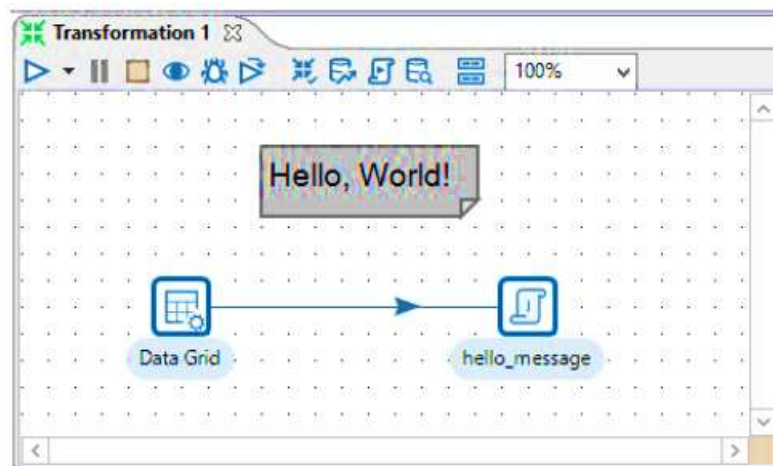


- Put the mouse cursor over the Data Grid step and wait until a tiny toolbar shows up succeeding the Data Grid icon, as shown:



Mouseover assistance toolbar

- Click on the output connector (the icon highlighted in the preceding image) and drag it towards the User Defined Java Expression (UDJE) step. A greyed hop is displayed.
- When the mouse cursor is over the UDJE step, release the button. A link—a hop from now on is created from the Data Grid step to the UDJE step.
- Double-click the UDJE icon and fill the grid as shown. Then close the window.
- Done! We have a draft for our first Transformation. A Data Grid with the names of a list of people, and a script step that builds the **hello_message**.

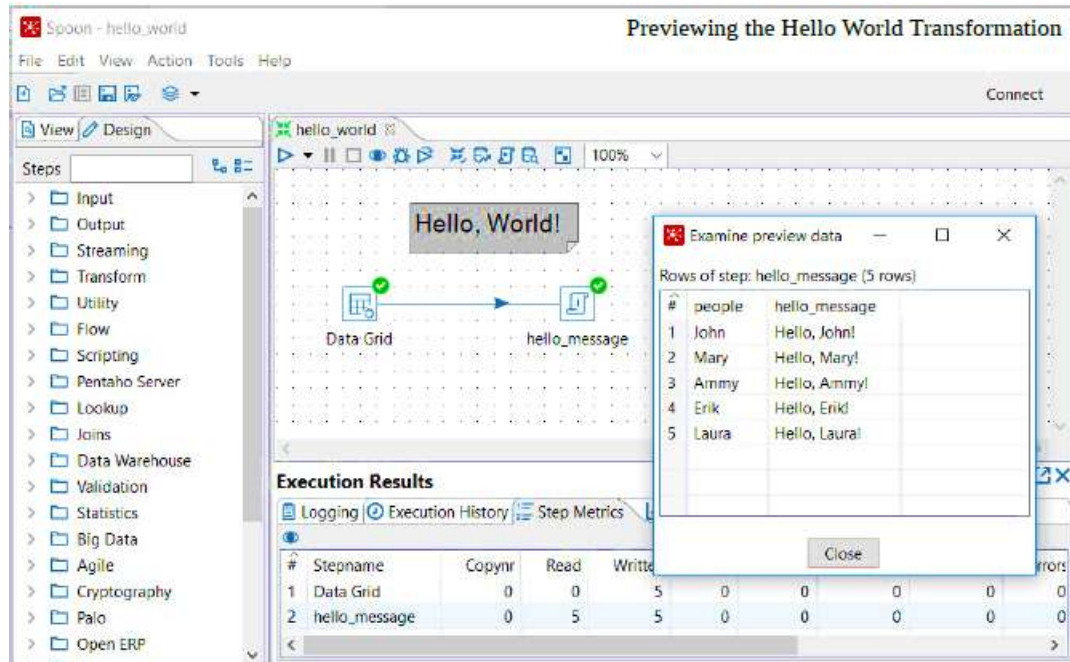



Hello World Transformation

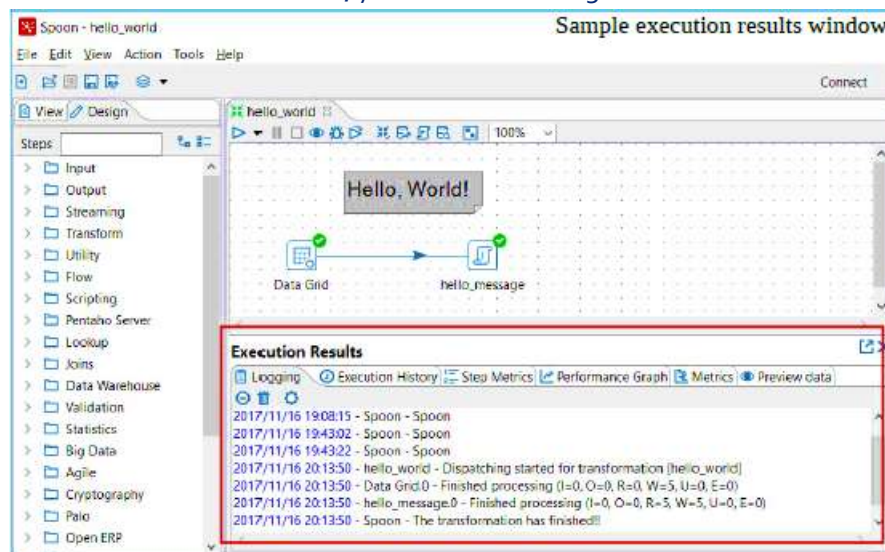
4. Previewing and running a Transformation

In our Transformation, we will preview the output of the User Defined Java Expression step:

- Select the User Defined Java Expression step by left-clicking on it.
- Click on the Preview icon in the bar menu preceding in the main canvas: Preview icon in the Transformation toolbar
- The Transformation debug dialog window will appear. Click on the Quick Launch button.
- A window will appear to preview the data generated by the Transformation, as shown in the following screenshot:

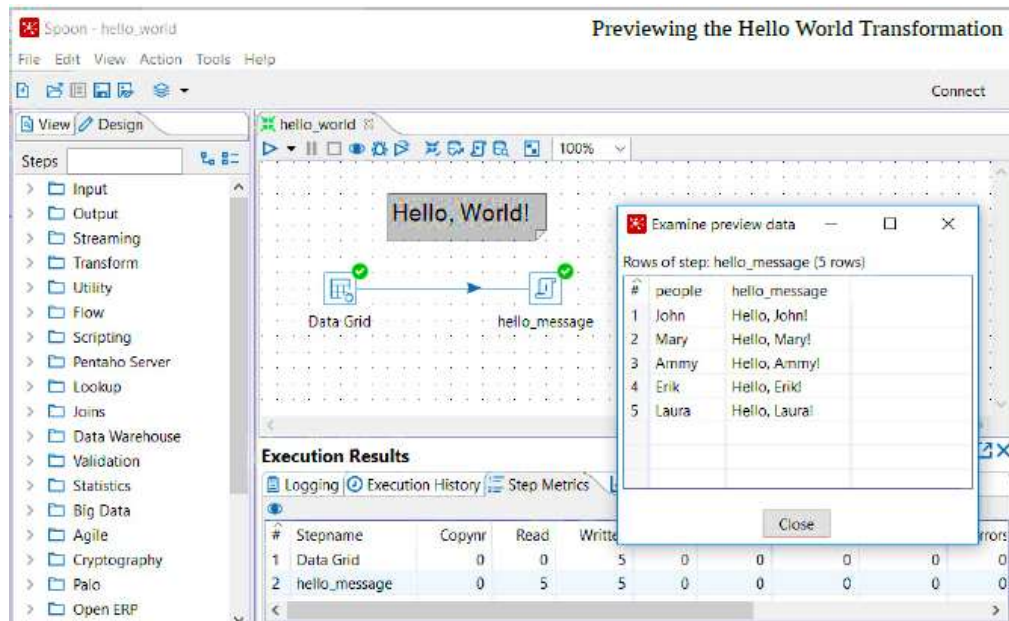


- Close the preview window.
- Once we have the Transformation ready, we can run it: Click on the Run icon:  in the Transformation toolbar. A window named Run Options appears. Click on Run.
- At the bottom of the screen, you should see a log with the result of the execution:

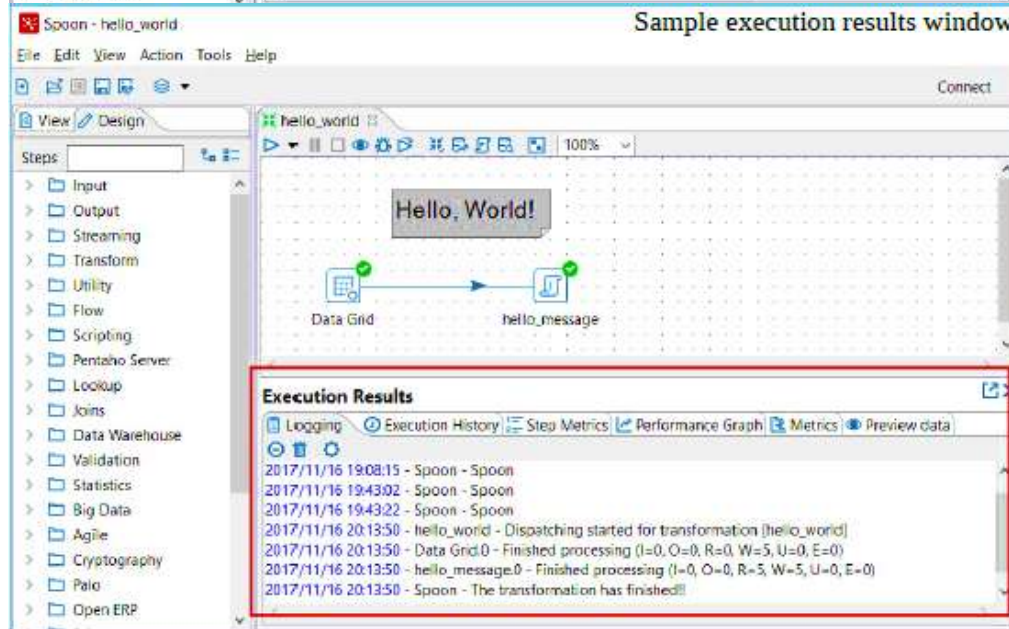


Save your transformation to "Tugas IS545 Lab Pentaho Data Integration yourname-NIM.ktr".

HASIL/ LUARAN



Sample execution results window



REFERENSI

-Utama-

1. María Carina Roldán. 2017. Learning Pentaho Data Integration 8 CE Third Edition. Packt Publishing.
2. Matt Casters, Roland Bouman, Jos van Dongen. 2010. Pentaho® Kettle Solutions: Building Open Source ETL Solutions with Pentaho Data Integration Wiley Publishing, Inc.
3. Pentaho Community website [Data Integration \(hitachivantara.com\)](http://hitachivantara.com)

-Pendukung-

4. Informasi/ Pengetahuan tambahan lainnya yang bisa didapatkan dari url/website tertentu.

MODUL 2

(PDI TRANSFORMATION: READING AND WRITING)



DESKRIPSI TEMA

Mahasiswa mampu menggunakan Pentaho Data Integration (PDI) untuk melakukan transformasi membaca data source dan membuat keluaran tabel database dengan kemampuan Metadata Injection

CAPAIAN PEMBELAJARAN MINGGUAN (SUB-CAPAIAN PEMBELAJARAN)

CMPK02-1 (C3)

Mampu merealisasikan beberapa kegiatan pada pengolahan data yang dilakukan termasuk pengujian transformasi dalam rangka merepresentasikan data model hubungan entitas dan model relasional pada perencanaan desain logis terhadap Data Warehouse yang direncanakan.

1. Create MySQL DB and Additional Software Development Installation
2. Examine Preview Sales Fact Northwind DB
3. Writing the Output

PENUNJANG PRAKTIKUM

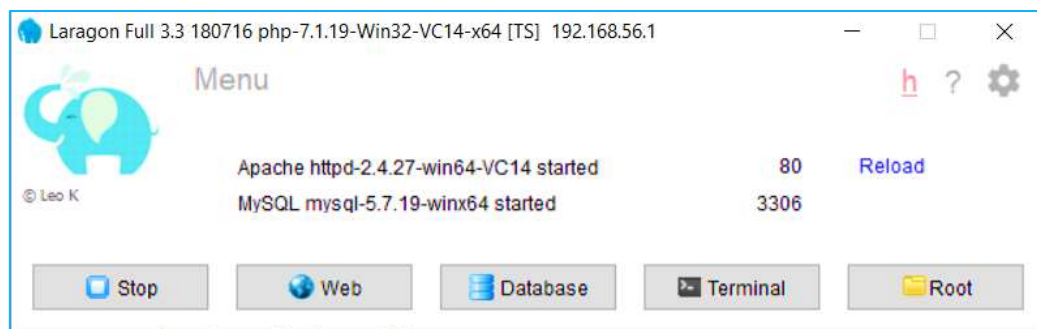
1. Windows Operating System
2. Java Runtime Environment (JRE) & Java Development Kit (JDK) 8.0 version (*installed*)
3. **Laragon** isolated dev environment on Windows
4. **SQLyog** management utility for MySQL databases front-end
5. Pentaho Data Integration tools (**PDI**) release 7.1 atau 8.0.

LANGKAH-LANGKAH PRAKTIKUM

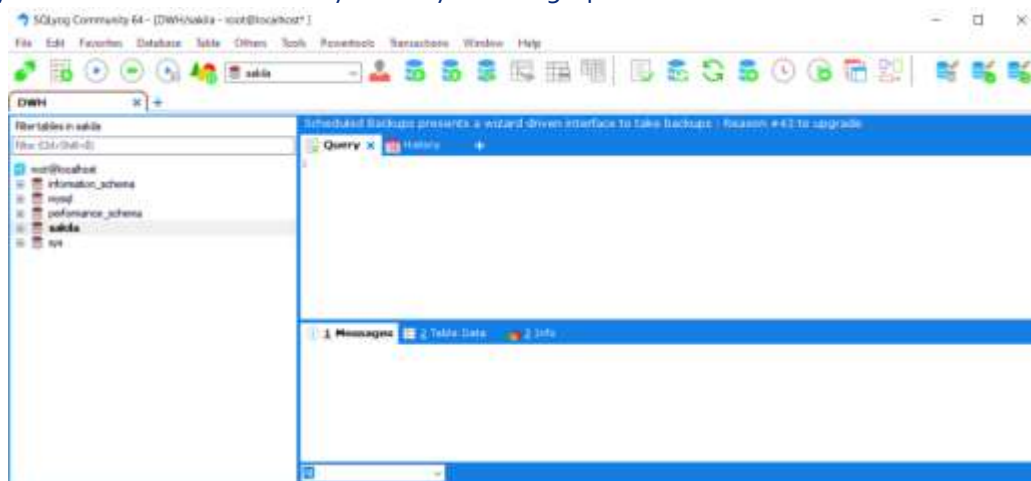
1. Create MySQL DB and Additional Software Development Installation

In this practicum, we will build and explore the NorthWind relational MySQL database, for this purpose it is necessary to prepare additional tools due to managing the MySQL database using Laragon and SQLyog:

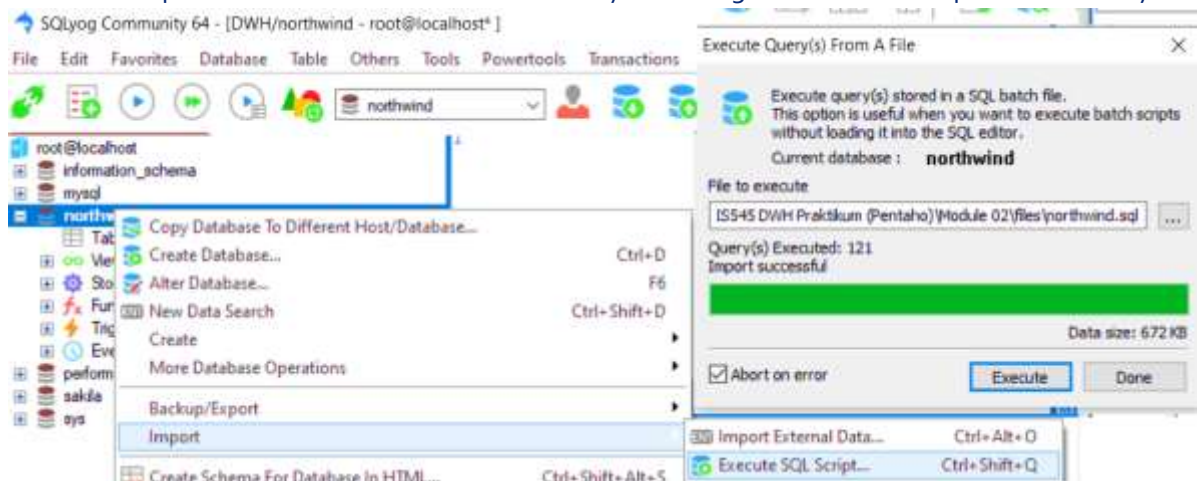
- a. **Laragon**: A great fast and easy way to create an isolated dev environment on Windows. Includes Mysql, PHP Memcached, Redis, Apache, and awesome for working with your Laravel projects. We need this Laragon to setup MySQL Virtual server and manage our MySQL Databases.
 - i. Download the source code from <https://sourceforge.net/projects/laragon/files/releases/4.0/laragon-full.exe/download> and
 - ii. follow the instructions on [Installation | Laragon - portable, isolated, fast & powerful universal development environment for PHP, Node.js, Python, Java, Go, Ruby.](#)
 - iii. **Start your MySQL Virtual** server as shown on below screenshot:



- b. **SQLyog** is a powerful and feature-packed Management Utility designed to run as a front-end for MySQL databases, easy to install and download the source code from <http://fs2.download82.com/software/bbd8ff9dba17080c0c121804efbd61d5/sqllyog-community-edition/SQLyog-12.4.3-0.x64Community.exe>. We need this SQLyog to make easy manage of our MySQL relational databases by the way of GUI /graphical user interface.



- c. Create **northwind** MySQL database by right-click-root@localhost)
d. Create and upload some tables of northwind DB by executing the northwind.sql till successfully.

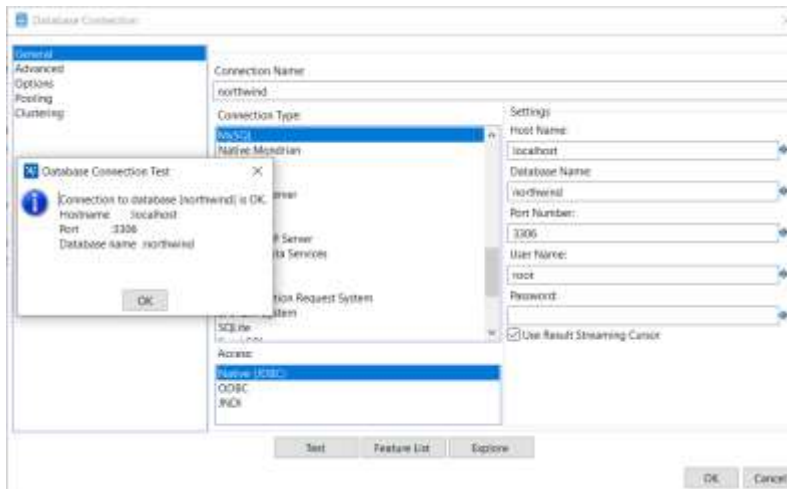


- e. Now, you had 13 tables in your northwind DB and ready to be explore by PDI.

2. Examine Preview Sales Fact Northwind DB

Now that you have completed the building of Northwind database and continue to do preview your sales fact by PDI transformation task:

- Forstly copy file **"mysql-connector-java-5.1.44-bin"** to your **PDI lib folder** (i.e. "D:\Document\UMN\Lab\ Pentaho\ data-integration\lib") then restarted your PDI.
- Open your new transformation (File – New -Trasformation), then drag the **table input** step to the canvas, named by "Extraction depuis table".
- Create new connection named by "nothwind" and test your connection to northwind MySQL database:



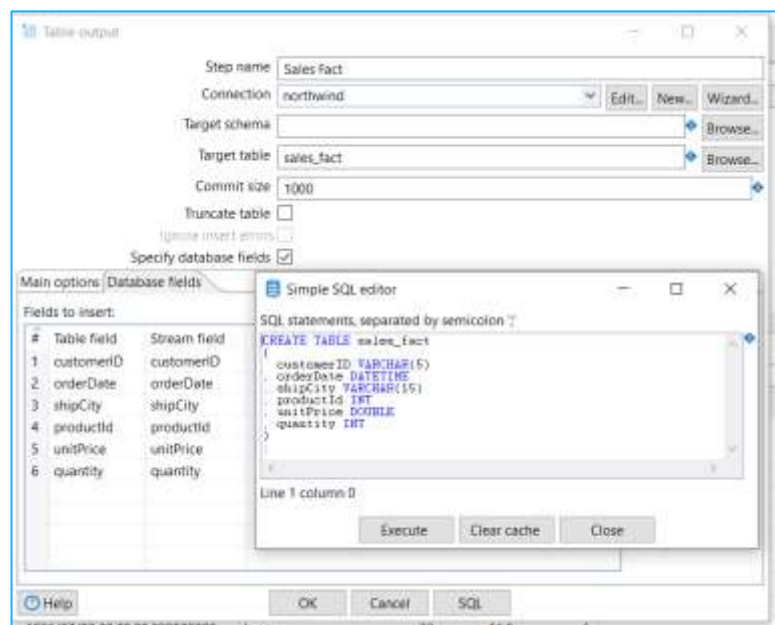
- d. Type the SQL statement to preview sales fact by order ID:

```
SELECT o.customerID, o.orderDate, o.shipCity, od.productId, unitPrice, quantity FROM orders o join order_details od on o.orderID = od.orderID
```
- e. Run your transformation and see the preview result at Execution result-preview data.
 Save your transformation to "Tugas IS545 Lab PDI Read & Write (A) yourname-NIM.ktr"

3. Writing the Output

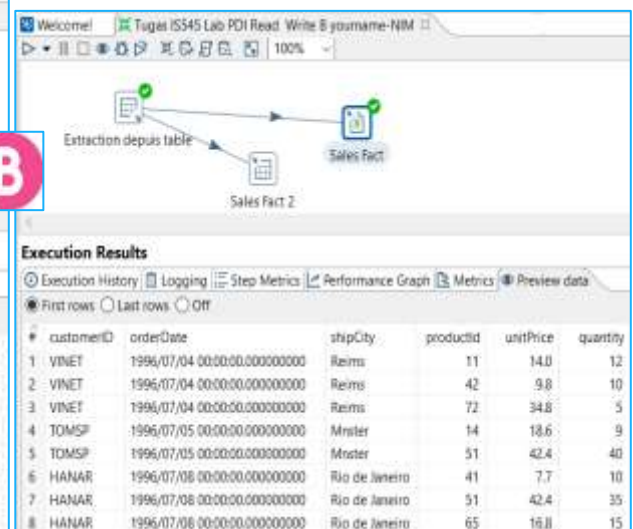
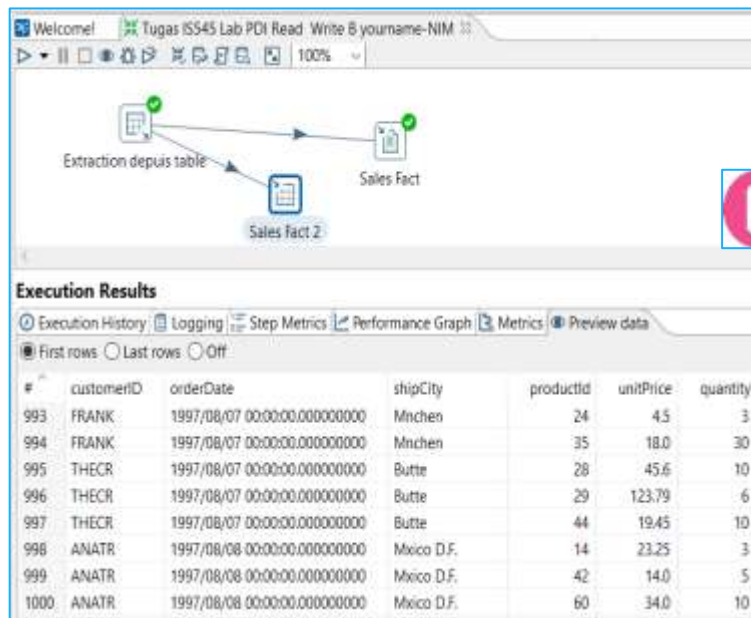
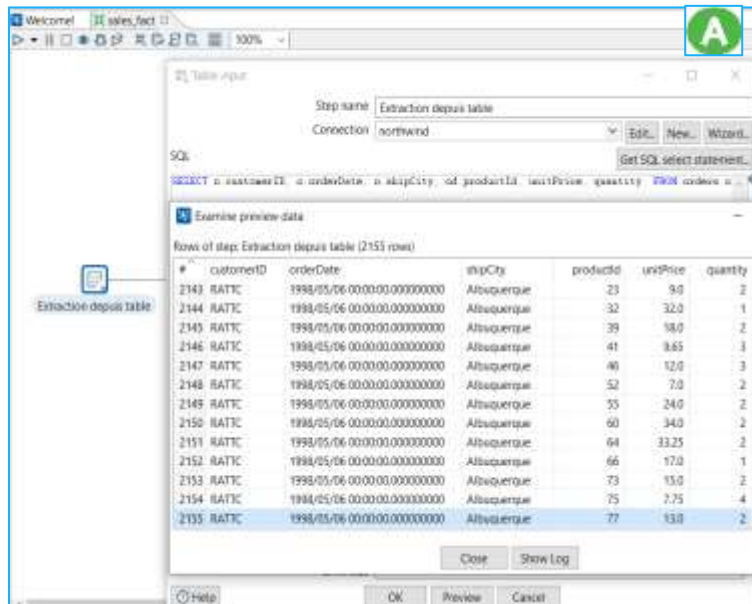
Next using of your "Tugas IS545 Lab PDI Read & Write (A) yourname-NIM.ktr" is to write the results of the previous query output to the spreadsheet file and create a new salesfact table:

- a. From your "Tugas IS545 Lab PDI Read & Write (A) yourname-NIM.ktr" transformation, drag the **microsoft excel output** step to canvas and named into "Sales Fact".
- b. Fill the "Filename" by **sales_fact.xls** on your chosen folder and goto "Fields" tab to "Get Fileds", here the PDI will fill your fields form by automatically.
- c. Drag the **table output** step to your canvas and named as "Sales Fact-2" and filled the connection to your northwind MySQL database.
- d. Named your output table (**target tale**) as **sales_fact** and ticked the "specify database fields", this function automatically will fill the "database fields form".
- e. Click below "SQL button" then execute the script by clicking the execute button to create your output table on northwind database.



- f. Run your transformation and see the preview result at Execution result-preview data for each output.
- Save your transformation to "Tugas IS545 Lab PDI Read & Write (B) yourname-NIM.ktr"

HASIL/ LUARAN



REFERENSI

-Utama-

1. María Carina Roldán. 2017. Learning Pentaho Data Integration 8 CE Third Edition. Packt Publishing.
2. Matt Casters, Roland Bouman, Jos van Dongen. 2010. Pentaho® Kettle Solutions: Building Open Source ETL Solutions with Pentaho Data Integration Wiley Publishing, Inc.
3. Pentaho Community website [Data Integration \(hitachivantara.com\)](http://hitachivantara.com)

-Pendukung-

4. Informasi/ Pengetahuan tambahan lainnya yang bisa didapatkan dari url/website tertentu.

MODUL 3

(PDI LOADING & LOOKING PUZZLES DIMENSION)

DESKRIPSI TEMA

Mahasiswa mampu menginterpretasikan konsep Dimensi dalam bentuk sebuah entitas yang menggambarkan kegiatan bisnis/ organisasi, memberikan contoh penggunaan dimensi dan tabel dimensi dengan beberapa penjelasannya mengenai pemahaman mahasiswa terhadap :

- ▶ Dasar-dasar dimensi
- ▶ Memuat dimensi waktu
- ▶ Memuat dimensi yang berubah perlahan (SCD/ Slowly Changing Dimensions)
- ▶ Memuat jenis dimensi lain

Mahasiswa mampu memanfaatkan alat bantu SQLyog untuk melakukan tranformasi data penjualan JigSaw, melaporkan order pembelian, laporan pembelian vs ketersediaan barang pesanan, memberikan alternatif ketersediaan barang/stock yang ada serta melakukan evaluasi perubahan pada melalui dimensi waktu perubahan.

CAPAIAN PEMBELAJARAN MINGGUAN (SUB-CAPAIAN PEMBELAJARAN)

CMPK03-1 (C3)

Memahami konsep dan langkah-langkah proses PDI melalui pembuatan job/ pekerjaan pada implementasi Data Warehouse yang sesuai dengan kebutuhan memanipulasi data dan kubus data secara interaktif.

1. SQLyog tools utilizing: Creating Database and Data Warehouse
2. Looking up stock for ordered products
3. Looking up if there are stock for ordered products by filter row
4. Looking for alternative puzzles, same theme as the out-of-stock puzzles
5. Loading the Puzzles dimension

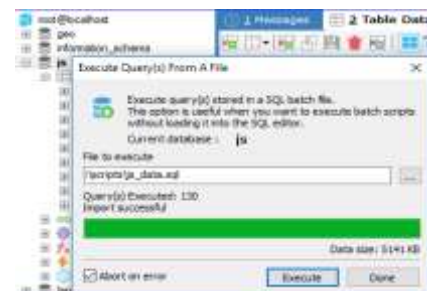
PENUNJANG PRAKTIKUM

1. Windows Operating System
2. Java Runtime Environment (JRE) & Java Development Kit (JDK) 8.0 version (*installed*)
3. **Laragon** isolated dev environment on Windows (*installed*)
4. **SQLyog** management utility for MySQL databases front-end (*installed*)
5. Pentaho Data Integration tools (PDI) release 7.1 atau 8.0.

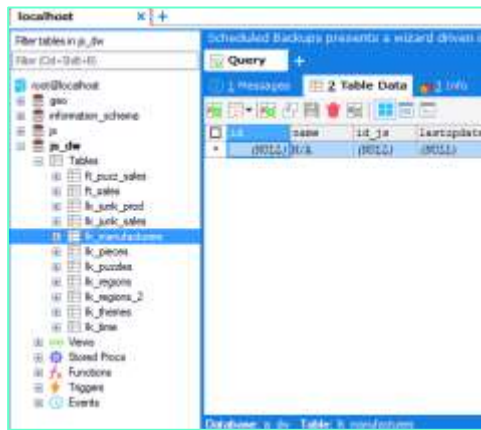
LANGKAH-LANGKAH PRAKTIKUM

1. SQLyog tools utilizing: Creating Database and Data Warehouse

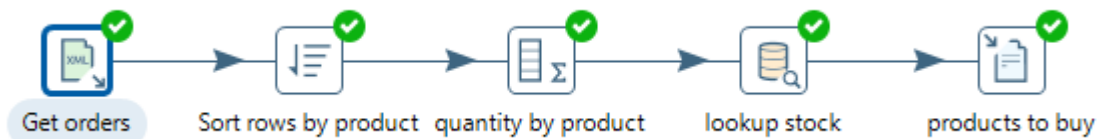
- a. By SQLyog, create js database by executing js.sql script and load the data by js_data.sql



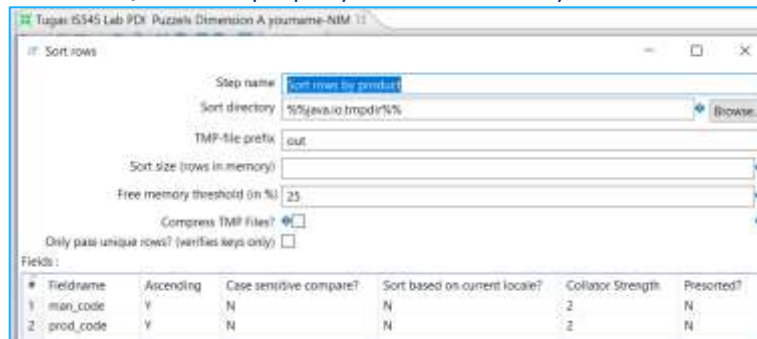
- b. Again, by SQLyog, create js_dw database as data warehouse by execute the js_dw.sql script.



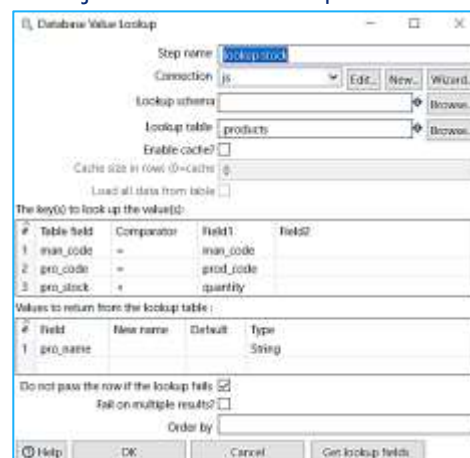
2. Looking up stock for ordered products



- a. Firstly, you must load the ordered data from the orders.xml file, drag the “get data from xml” step and filled in the selected file by “\${Internal.Transformation.Filename. Directory}/orders.xml” then “get fields” in the fields tab to get all fields automatically.
- b. To create the dimension, look like properly make it sorted by manufactures and product:



- c. Due to no info of how many product ordered, the make it counted by grouped the items into manufactures and product in “group field” of **Group By** step and fill the “Aggregates” by Name=Quantity; Subject=ordernumber; Type=Number (N).
- d. Then matched those input to the js database to lookup stock data by **Database Value Lookup** step:

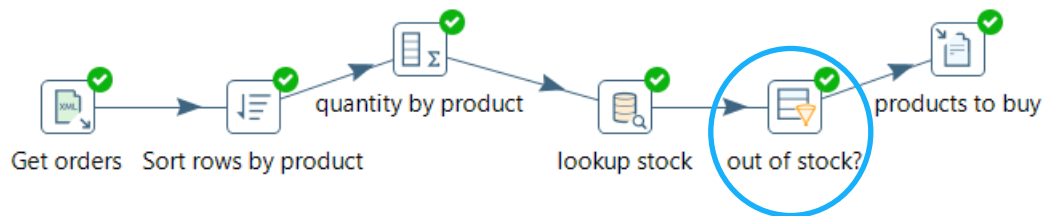


The look up condition was “product stock < quantity”, means searching for the out of stock product to buy and reported the list of product ordered to output text file.

- e. Finally, send the output of looking up to the text file using of **Text file output** step and fill in the filename by “\${Internal.Transformation. Filename.Directory}/pdi_files/output//products_to_buy”. Please make sure you had fields inside by pressing the “get fields” on fields tab. Save your transformation into “Tugas IS545 Lab PDI Puzzels Dimension (A) yourname-NIM.ktr”.

3. Looking up if there are stock for ordered products by filter row

This transformation was same as the previous Database Lookup, to looking up the out of stock product to ordered. The difference is that the search process uses **Filter Row** step and displaying the output in the form of available stock conditions (not the ordered product).



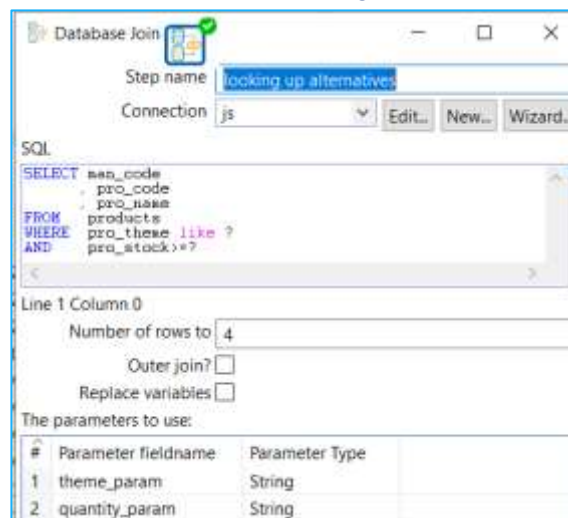
- a. Do the same step of 2.a) until 2.d) and throw away the condition of “pro_stock < quantity”. As a replacement drag the **Filter rows** step and filled in the condition by “pro_stock < quantity”.
- b. Finally, send the output of looking up to the text file using of **Text file output** step and fill in the filename by “\${Internal.Transformation. Filename.Directory}/pdi_files/output//products_to_buy”. Please make sure you had fields inside by pressing the “get fields” on fields tab.

Save your transformation into “Tugas IS545 Lab PDI Puzzels Dimension (B) yourname-NIM.ktr”.

4. Looking for alternative puzzles, same theme as the out-of-stock puzzles

On this practice you will learn how to Concatenate the same product ordered when grouped it into manufactures and product. Then giving some advices to other product when the product requested was out of stock.

- a. Do the same step of 2.a) until 2.d)
- b. Add the **Select/Rename values** step to the canvas, rename fields of “quantity to quantity_param” and “pro_theme to theme_param” in the ‘select&alter’ tab.
- c. Add the **Database Join** step to the canvas named it by” looking up alternatives” as follows:



Save your transformation into "Tugas IS545 Lab PDI Puzzels Dimension (C) yourname-NIM.ktr" and run it to get the result.

5. Loading the Puzzles dimension

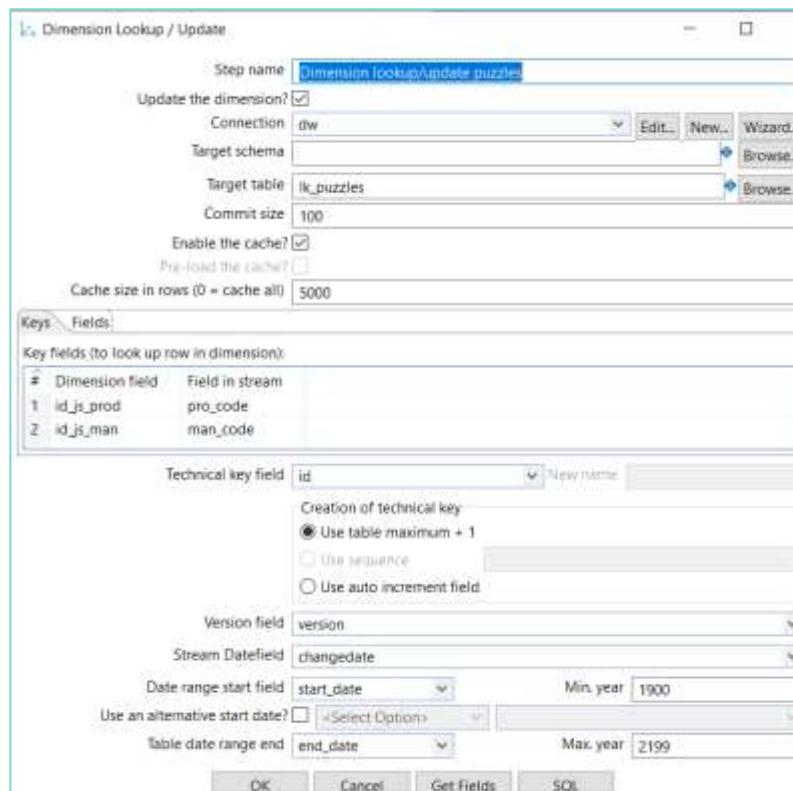
Now we will do keep the history of Puzzles product changes with the Dimension lookup/update step and inserting them into **lk_puzzles table** in js_dw database (previously the lk_puzzles table was empty).

- Open Spoon. From the main menu and navigate to File | New | Transformation.
- On the left of the screen, under the Design tab, you'll see a tree of Steps. Expand the Input branch by double-clicking on it and drag the **Table input** step to your canvas as follows:



You could press the "preview" button to see the query result.

- To keep your history changes of the puzzles product, you need the Add constant values and filled in the "Name=changedate"; "Type=Date"; "Format=dd/MM/yyyy"; "Value=01/01/2021"; "empty string=N".
- Finally, insert the query results into lk_puzzles table by fulfilled the component of **Dimension Lookup/Update** step as follows :



Save your transformation into "Tugas IS545 Lab PDI Puzzels Dimension (D) yourname-NIM.ktr"
and run it to get the result

HASIL/ LUARAN

Execution Results

Execution History | Metrics | Performance Graph | Metrics

First rows | Last rows | products to buy

#	man_code	prod_code	quantity	pro_name
1	EDU	ED13_93	1	Times Square
2	RAV	RVZ50031	2	Disney World Map
3	RAV	RVZ50106	1	Star Wars Clone Wars

> UMN > DWH > 2020 > Week#8 > pdi_files > output

Name: products_to_buy | Date modified: 15/03/2020 22:21 | Type: Text Document

products_to_buy - Notepad

```
File Edit Format View Help
man_code;prod_code;pro_name;quantity
EDU;ED13_93;Times Square;1
RAV;RVZ50031;Disney World Map;2
RAV;RVZ50106;Star Wars Clone Wars;1
```

A

Get orders | Sort rows by product | quantity by product | lookup stock | out of stock? | products to buy

Execution Results

Execution History | Logging | Step Metrics | Performance Graph | Metrics | Preview

First rows | Last rows | Off

#	man_code	prod_code	quantity	pro_name	pro_stock
1	EDU	ED13_93	1	Times Square	0
2	RAV	RVZ50031	2	Disney World Map	1
3	RAV	RVZ50106	1	Star Wars Clone Wars	0

products_to_buy - Notepad

```
File Edit Format View Help
man_code;prod_code;quantity;pro_name;pro_stock
EDU;ED13_93;1;Times Square;0
RAV;RVZ50031;2;Disney World Map;1
RAV;RVZ50106;1;Star Wars Clone Wars;0
```

B

Tugas IS545 Lab PDI Puzzels Dimension C yourname-NIM

Get orders | Sort rows by product | quantity by product | Select values | lookup stock | looking up alternatives

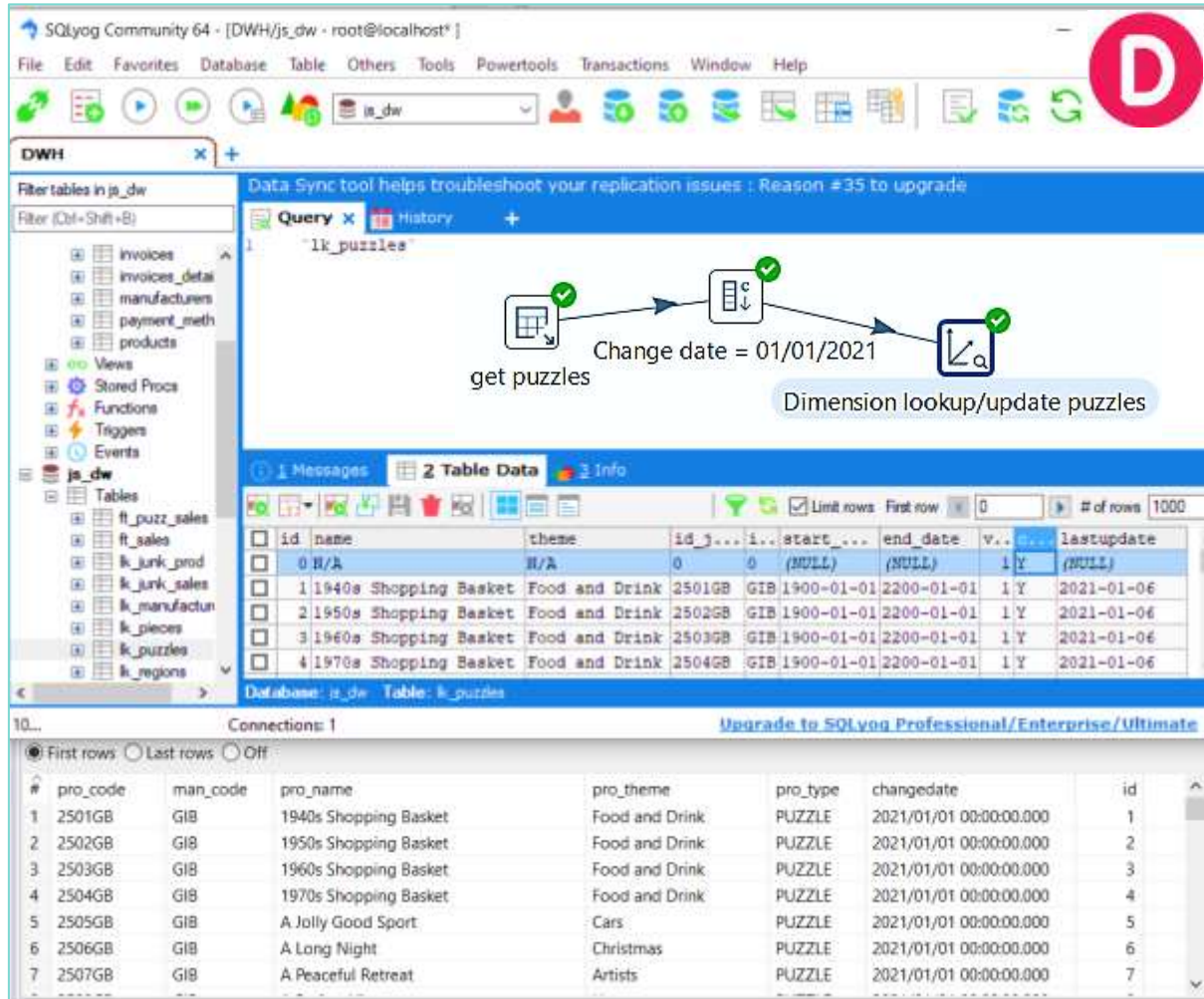
Execution Results

Execution History | Logging | Step Metrics | Performance Graph | Metrics | Preview data

First rows | Last rows | Off

#	customers	quantity_param	theme_param	pro_name	man_code	pro_code	pro_name_1
1	8411	1	Famous Landmarks	Times Square	GIB	2523GB	Beautiful Britain The Cotswolds
2	8411	1	Famous Landmarks	Times Square	GIB	2524GB	Beautiful Britain The Lake District
3	8411	1	Famous Landmarks	Times Square	GIB	2525GB	Beautiful Britain The West Country
4	8411	1	Famous Landmarks	Times Square	GIB	2604GB	London - Looking North
5	8328, 59	2	Disney	Disney World Map	CLE	CMT1016	Cars
6	8328, 59	2	Disney	Disney World Map	CLE	CMT1021	Disney Princesses - Make Up
7	8328, 59	2	Disney	Disney World Map	CLE	CMT1022	Disney Villains
8	8328, 59	2	Disney	Disney World Map	CLE	CMT1059	Princesses
9	6392	1	Hollywood	Star Wars Clone Wars	RAV	RVZ50058	Indiana Jones
10	6392	1	Hollywood	Star Wars Clone Wars	RAV	RVZ50105	Star Wars
11	6392	1	Hollywood	Star Wars Clone Wars	TDC	TDC1CLT	Lost Puzzle #1 - The Hatch
12	6392	1	Hollywood	Star Wars Clone Wars	TDC	TDC3MEN	Lost Puzzle #3 - The Numbers

C



SQLyog Community 64 - [DWH/js_dw - root@localhost*]

File Edit Favorites Database Table Others Tools Powertools Transactions Window Help

DWH

Filter tables in js_dw

Filter (Ctrl+Shift+B)

invoiced
invoices_detail
manufacturers
payment_meth
products
Views
Stored Procs
Functions
Triggers
Events
js_dw
Tables
ft_puzz_sales
ft_sales
lk_junk_prod
lk_junk_sales
lk_manufactun
lk_pieces
lk_puzzles
lk_regions

Data Sync tool helps troubleshoot your replication issues : Reason #35 to upgrade

Query x History

1 "lk_puzzles"

get puzzles

Change date = 01/01/2021

Dimension lookup/update puzzles

2 Table Data

Limit rows First row 0 # of rows 1000

id	name	theme	id_3...	i...	start...	end_date	v...	lastupdate
0	N/A	N/A	0	0	(NULL)	(NULL)	1 Y	(NULL)
1	1940s Shopping Basket	Food and Drink	2501GB	GIB	1900-01-01	2200-01-01	1 Y	2021-01-06
2	1950s Shopping Basket	Food and Drink	2502GB	GIB	1900-01-01	2200-01-01	1 Y	2021-01-06
3	1960s Shopping Basket	Food and Drink	2503GB	GIB	1900-01-01	2200-01-01	1 Y	2021-01-06
4	1970s Shopping Basket	Food and Drink	2504GB	GIB	1900-01-01	2200-01-01	1 Y	2021-01-06

Database: js_dw Table: lk_puzzles

10... Connections: 1 Upgrade to SQLyog Professional/Enterprise/Ultimate

First rows Last rows Off

#	pro_code	man_code	pro_name	pro_theme	pro_type	changedate	id
1	2501GB	GIB	1940s Shopping Basket	Food and Drink	PUZZLE	2021/01/01 00:00:00.000	1
2	2502GB	GIB	1950s Shopping Basket	Food and Drink	PUZZLE	2021/01/01 00:00:00.000	2
3	2503GB	GIB	1960s Shopping Basket	Food and Drink	PUZZLE	2021/01/01 00:00:00.000	3
4	2504GB	GIB	1970s Shopping Basket	Food and Drink	PUZZLE	2021/01/01 00:00:00.000	4
5	2505GB	GIB	A Jolly Good Sport	Cars	PUZZLE	2021/01/01 00:00:00.000	5
6	2506GB	GIB	A Long Night	Christmas	PUZZLE	2021/01/01 00:00:00.000	6
7	2507GB	GIB	A Peaceful Retreat	Artists	PUZZLE	2021/01/01 00:00:00.000	7

REFERENSI

-Utama-

1. María Carina Roldán. 2017. Learning Pentaho Data Integration 8 CE Third Edition. Packt Publishing.
2. Matt Casters, Roland Bouman, Jos van Dongen. 2010. Pentaho® Kettle Solutions: Building Open Source ETL Solutions with Pentaho Data Integration Wiley Publishing, Inc.
3. Pentaho Community website [Data Integration \(hitachivantara.com\)](https://hitachivantara.com)

-Pendukung-

4. Informasi/ Pengetahuan tambahan lainnya yang bisa didapatkan dari url/website tertentu.

MODUL 4

(PDI: HIERARCHICAL RELATIONSHIP)



DESKRIPSI TEMA

Tabel induk-anak (parent-child table) adalah tabel yang di dalamnya terdapat hubungan referensi mandiri. dengan kata lain terdapat hubungan hierarki di antara baris-barisnya. Contoh dari praktikum kali ini adalah tabel karyawan Steel Wheels, yang salah satu kolomnya berisi referensi karyawan dalam bentuk peranan hierarki dijabarkan sebagai berikut:

- ▶ Seorang perwakilan penjualan melapor kepada manajer penjualan
- ▶ Seorang manajer penjualan melapor kepada wakil presiden
- ▶ Seorang wakil presiden melapor kepada presiden
- ▶ Presiden adalah tingkat tertinggi dalam hierarki. Hanya ada satu karyawan dengan peran ini.

CAPAIAN PEMBELAJARAN MINGGUAN (SUB-CAPAIAN PEMBELAJARAN)

CMPK03-2 (C4)

Memahami konsep pembacaan dan penulisan beberapa format file dalam rangka penyiapan data untuk pemodelan MultiDimensi dan Aplikasi OLAP untuk pemodelan hirarki dalam rangka eksplorasi data.

1. Load all employees
2. The president is the highest level in the hierarchy
3. Designing and running Employees Loading jobs
 - ▶ A vice-president reports to the president
 - ▶ A sales manager reports to a vice-president
 - ▶ A sales representative reports to a sales manager

PENUNJANG PRAKTIKUM

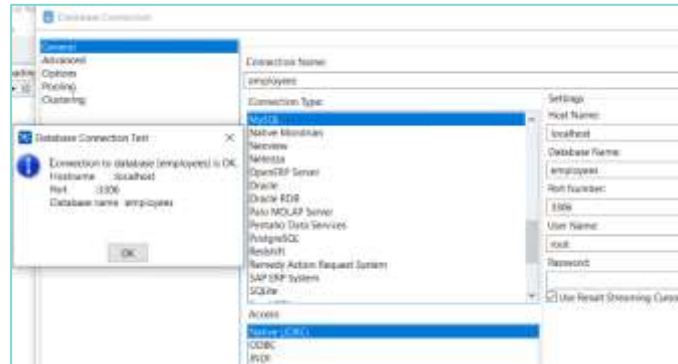
1. Windows Operating System
2. Java Runtime Environment (JRE) & Java Development Kit (JDK) 8.0 version (*installed*)
3. **Laragon** isolated dev environment on Windows (*installed*)
4. **SQLyog** management utility for MySQL databases front-end (*installed*)
5. Pentaho Data Integration tools (**PDI**) release 7.1 atau 8.0.

LANGKAH-LANGKAH PRAKTIKUM

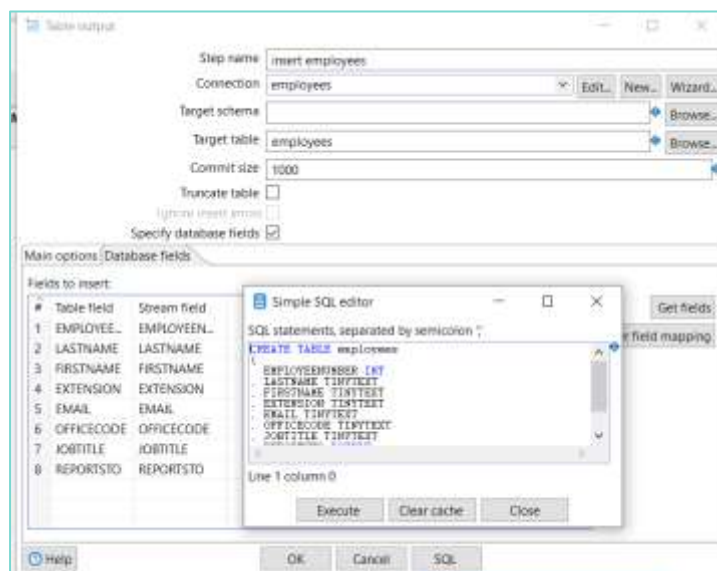
Firstly, by SQLyog, create the empty **employees** database.

1. **Load all employees to employees table in employees database**
 - a. Create new transformation to load the rest of the employees. Define a named parameter named LEVEL that will represent the role of the employees being loaded.
 - b. Use a **Text file input** step to read the file of employees (employees.txt).
 - c. Use a **Get Variables** step to add the variable LEVEL as a new field named level by "string" type.
 - d. Use a **Join rows** step to merge data from text file input as employees data and get the level of employees from get variable step. Filled in "TNP-file prefix=out" and the condition "<field> = <field> <value>"

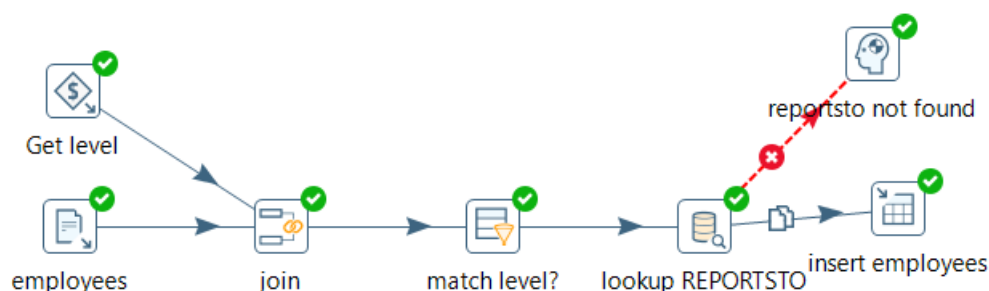
- e. Add a **Filter rows** step to filter the employees to load based on their role. In order to do that, enter the following condition: "JOBTITLE REGEXP level".
- f. Add a **Database value lookup** step to find out the employee number of the employee who is one above in the hierarchy: In the upper grid add a row with the condition EMAIL = REP_TO. Use the lower grid to get the field EMPLOYEENUMBER and rename it to REPORTSTO. Please make sure you had created your database connection into employees database:



- g. Add a **Table Output** step by condition :
 - i. Use Dummy step to get end process when employee not found
 - ii. and use it (**table output**) to insert the records in the table employees.
 Please make sure you had created the employees table in employees database:



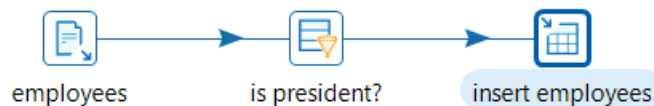
- h. Your final transformation looks like this:



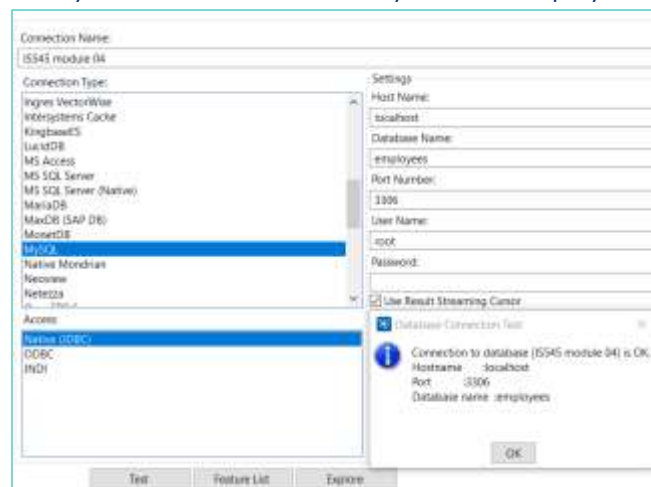
- i. Save your job into "Tugas IS545 Lab Hierarchical Relationship (A) yourname-NIM.ktr" and press F9 to run it, you will got these empty result! Why ?
- j. The answer is, you need to give them *level occupation* as the value for the **LEVEL** parameter.

2. **The president is the highest level in the hierarchy**, there is a single employee with this role.

- a. Create a transformation (named "**Tugas IS545 Lab Hierarchical Relationship (B) yourname-NIM.ktr**") that inserts the record for the president who is the first in the hierarchy, and doesn't report to anyone.
- b. The transformation should read the input file (**employees.txt**) , filter the record with **JOBTITLE=President**, and insert the data into the employee's table.



Please do make sure you had connection already into the employees database:



- c. This transformation the only one to inserting the president "1002 | **Murphy** | **NULL** | **President**".
- d. Save your transformation and run it later.

Finally create a job (**KJB**) to put all together.

What is KJB ?

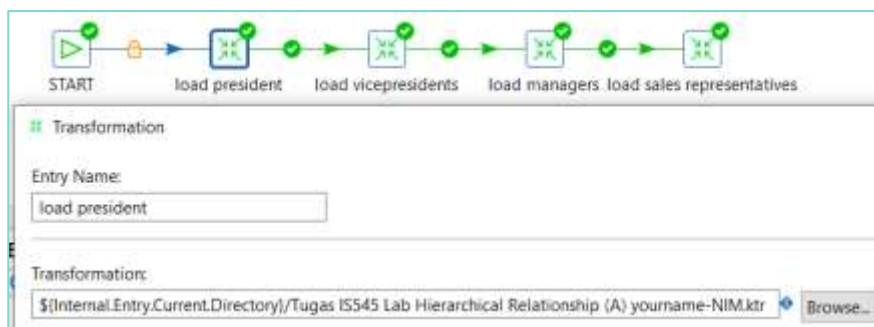
KJB file is a Kettle Job File. Kettle (Kettle Extraction Transformation Transport Load Environment) is a scalable and extensible open source ETL and data integration tool that lets you extract data from databases, flat and XML files, web services, ERP systems, and OLAP cubes.

3. Designing and running Employees Loading jobs

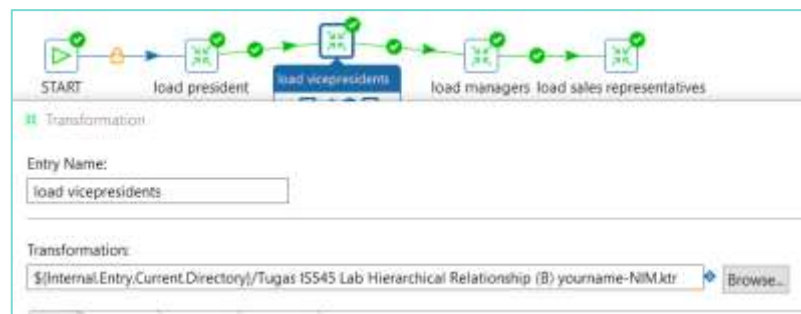
A Job is an entity made of Job entries linked by hops. These entries and hops build paths, telling the engine the sequence of individual tasks to accomplish. Therefore, we say that a Job is task oriented. Graphically, Job entries are represented with small boxes, whereas hops are represented with directional arrows, as depicted in the following sample:



1. Open Spoon. Navigate to File | New | Job or press Ctrl + Alt + N. A new Job will be created.
2. To the left of the screen, there is a tree with Job entries. Expand the General category of Job entries.
3. Drag to the work area a START entry, and four Transformation job entries. Link all of them in a row.
4. Use the second transformation "Tugas IS545 Lab Hierarchical Relationship (B) yourname-NIM.ktr" entry to execute the transformation that loads the president.



5. A vice-president reports to the president
Double-click the first Transformation "Tugas IS545 Lab Hierarchical Relationship (A) yourname-NIM.ktr" entry and configure it to run the transformation that loads the other employees.

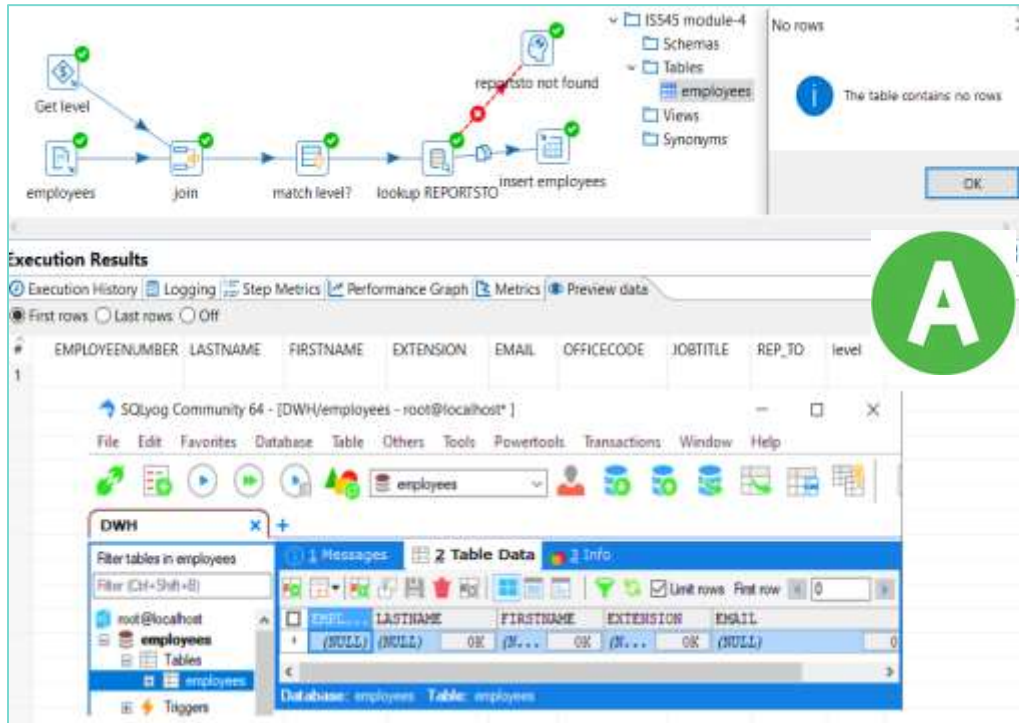


to run the transformation that loads the other employees. Under the Parameters tab, add a parameter named LEVEL with value VP.*.

6. A sales manager reports to a vice-president.
Double-click the first Transformation "Tugas IS545 Lab Hierarchical Relationship (A) yourname-NIM.ktr" entry and configure it to run the transformation that loads the other employees. Under the Parameters tab, add a parameter named LEVEL with value .*Manager.*.
7. A sales representative reports to a sales manager
Double-click the first Transformation "Tugas IS545 Lab Hierarchical Relationship (A) yourname-NIM.ktr" entry and configure it to run the transformation that loads the other employees. Under the Parameters tab, add a parameter named LEVEL with value Sales Rep.*.

Save your job into "Tugas IS545 Lab Hierarchical Relationship (C) yourname-NIM.kjb" and press **F9** to run it

HASIL/ LUARAN

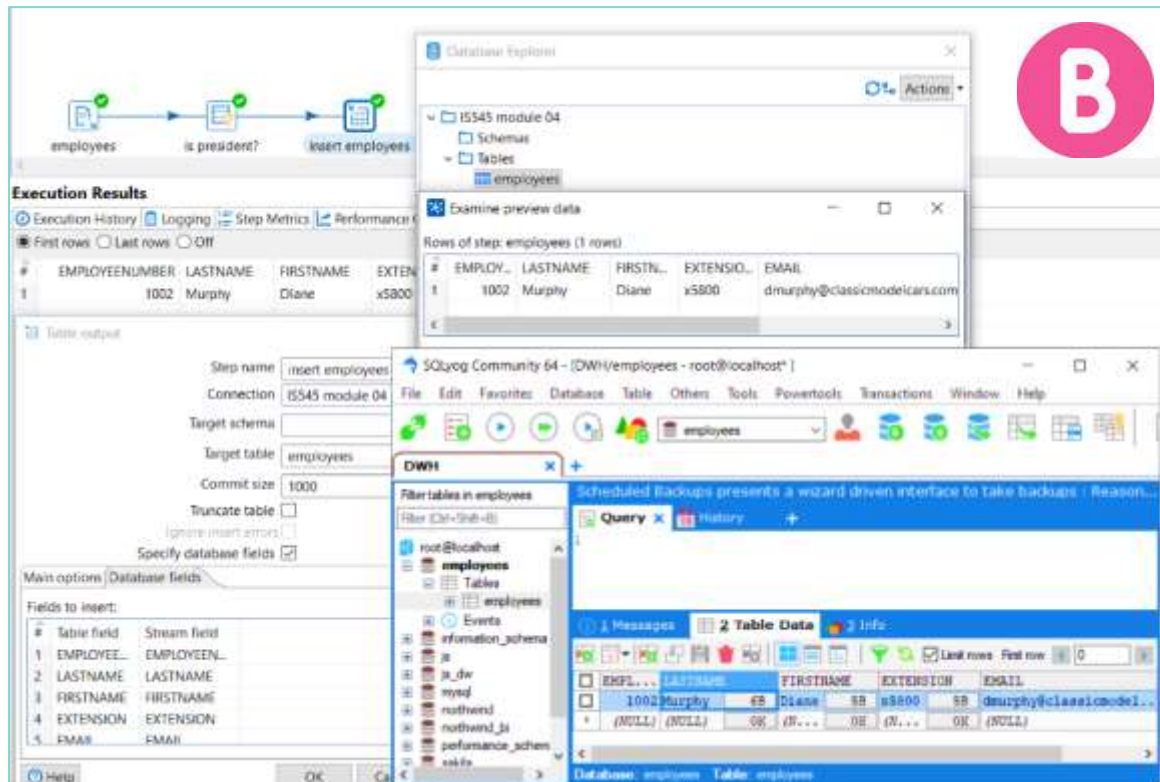


Execution Results

#	EMPLOYEENUMBER	LASTNAME	FIRSTNAME	EXTENSION	EMAIL	OFFICECODE	JOBTITLE	REP_TO	level
1									

SQLyog Community 64 - [DWH/employees - root@localhost*]

Database: employees Table: employees



Execution Results

#	EMPLOYEE...	LASTNAME	FIRSTNAME	EXTENSION	EMAIL
1	1002	Murphy	Diane	x5800	dmurphy@classicmodelcars.com

SQLyog Community 64 - [DWH/employees - root@localhost*]

Database: employees Table: employees

Database Explorer

Tugas IS545 Lab Hierarchical Relations

Tugas IS545 Lab Hierarchical Relations

100%

START → load president → load vicepresidents → load managers → load sales representatives

Examine preview data

Rows of step: employees (23 rows)

#	EMPLO...	LASTNAME	FIRSTNAME	EXTENSION	EMAIL	OFFI...	JOBTITLE
1	1002	Murphy	Diane	x5800	dmurphy@classicmodelcars.com	1	President
2	1056	Patterson	Mary	x4611	mpatterso@classicmodelcars.com	1	VP Sales
3	1076	Firrelli	Jeff	x9273	jfirrelli@classicmodelcars.com	1	VP Marketing
4	1088	Patterson	William	x4871	wpatterson@classicmodelcars.com	6	Sales Manager
5	1102	Bondur	Gerard	x5408	athompson@classicmodelcars.com	4	Sale Manager
6	1143	Bow	Anthony	x5428	bhoward@classicmodelcars.com	1	Sales Manager
7	1165	Jennings	Leslie	x3291	ljennings@classicmodelcars.com	1	Sales Rep
8	1166	Thompson	Leslie	x4065	lthompson@classicmodelcars.com	1	Sales Rep
9	1188	Firrelli	Julie	x2173	jfirrelli@classicmodelcars.com	2	Sales Rep
10	1216	Patterson	Steve	x4334	spatterso@classicmodelcars.com	2	Sales Rep
11	1286	Tseng	Foon Yue	x2248	ftseng@classicmodelcars.com	3	Sales Rep
12	1323	Vanauf	George	x4102	gvanauf@classicmodelcars.com	3	Sales Rep
13	1337	Bondur	Loui	x6493	lbondur@classicmodelcars.com	4	Sales Rep
14	1370	Hernandez	Gerard	x2028	ghernande@classicmodelcars.com	4	Sales Rep
15	1401	Castillo	Pamela	x2759	pcastillo@classicmodelcars.com	4	Sales Rep
16	1501	Bott	Larry	x2311	lbott@classicmodelcars.com	7	Sales Rep
17	1504	Jones	Barry	x102	bjones@classicmodelcars.com	7	Sales Rep
18	1611	Fixter	Andy	x101	afixter@classicmodelcars.com	6	Sales Rep
19	1612	Marsh	Peter	x102	pmarsh@classicmodelcars.com	6	Sales Rep

REFERENSI

-Utama-

1. Adrián Sergio Pulvirenti, María Carina Roldán. 2011. Pentaho Data Integration 4 Cookbook. Packt Publishing.
2. Matt Casters, Roland Bouman, Jos van Dongen. 2010. Pentaho® Kettle Solutions: Building Open Source ETL Solutions with Pentaho Data Integration Wiley Publishing, Inc.
3. Pentaho Community website

-Pendukung-

4. Informasi/ Pengetahuan tambahan lainnya yang bisa didapatkan dari url/website tertentu.

MODUL 5

(LOADING DIMENSIONS WITH DATA)

DESKRIPSI TEMA

Merepresentasikan bentuk, manipulasi serta pengelolaan tabel dimensi data. Meliputi penanganan performa pemutakhiran data, catatan perubahan data yang dilakukan yang sesuai dengan atribut kunci dalam entitas melalui penerapan teknik SCD/ Slowly Changing Dimensions.

CAPAIAN PEMBELAJARAN MINGGUAN (SUB-CAPAIAN PEMBELAJARAN)

CMPK03-3 (C3)

Mahasiswa memahami konsep desain logis Data Warehous dan mampu menerjemahkan konseptual skema multidimensi ke skema logis, serta dapat menjelaskan bagaimana operasional OLAP dapat diimplementasikan dan penanganan perubahan dimensi data melalui teknik-teknik SCD/ Slowly Changing Dimensions.

1. Loading a time dimension
2. Loading Type 1 SCD/ Slowly Changing Dimensions
3. Loading Type 2 SCD/ Slowly Changing Dimensions

PENUNJANG PRAKTIKUM

1. Windows Operating System
2. Java Runtime Environment (JRE) & Java Development Kit (JDK) 8.0 version (*installed*)
3. **Laragon** isolated dev environment on Windows (*installed*)
4. **SQLyog** management utility for MySQL databases front-end (*installed*)
5. Pentaho Data Integration tools (**PDI**) release 7.1 atau 8.0.

LANGKAH-LANGKAH PRAKTIKUM

- ▶ Understanding dimensions technical details, Dimensions should have a special record for the unavailable data. This implies that besides one record for every member of the dimension, you should have a record with the key equal to zero, and N/A or unknown, or something that represents invalid data for all the descriptive attributes.
- ▶ These practices continuing the module-3 " PDI Loading & Looking Puzzles Dimension" managing Jigsaw data warehouse (js_dw database).

1. Loading a time dimension

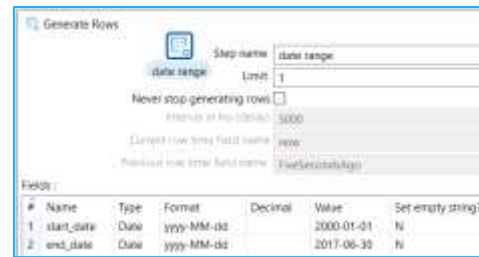
- ▶ The time dimension is a special dimension used to describe a business in terms of when things happened. Time dimensions are meant to answer questions related to time, for example, do I sell more on Mondays or on Fridays? Am I selling more this quarter than the same quarter last year?

- ▶ Let's start with the creation of the Transformation. The objective is to generate a dataset with all dates in between a given range of dates and keep it into lk_time_simple at js data warehouse.

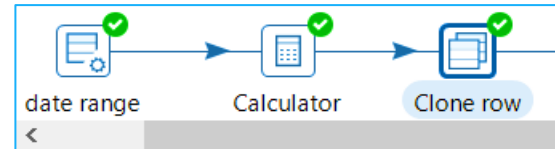
- a. Firstly create a connection into js_dw database were previously created on module-3 " PDI Loading & Looking Puzzles Dimension":



- b. Create a new Transformation. From the Input group of steps, drag to the canvas the Generate Rows step, and configure it as shown:



- c. From the Transform category of steps, add the Calculator step, and create a hop that goes from the Generate Rows step to this one.
- d. Double-click on the Calculator step and add the field named diff_dates as the difference between end_date and start_date. That is, configure it exactly the same way as you did in the previous section.
- e. Now add the Clone row step. You will find it inside the Utility group of steps.
- f. Create a hop from the Calculator step towards this new step. Edit the Clone row step.
- g. Select the Nr clone in field? option to enable the Nr Clone field textbox. In this textbox, type diff_dates.
- h. Now select the Add clone num to output? option to enable the Clone num field textbox. In this textbox, type delta. Run a preview. You should see the following:




Execution Results

Execution History | Logging | Step Metrics

First rows | Last rows | Off

#	start_date	end_date	diff_dates	delta
1	2000-01-01	2020-12-31	7670	0
2	2000-01-01	2020-12-31	7670	1
3	2000-01-01	2020-12-31	7670	2
4	2000-01-01	2020-12-31	7670	3
5	2000-01-01	2020-12-31	7670	4
6	2000-01-01	2020-12-31	7670	5

- i. Add another Calculator step, and create a hop from the Clone row step to this one.
- j. Edit the new step, and add the field named a_single_date. As Calculation, select Date A + B Days. As Field A, select start_date and as Field B, select delta.
- k. Finally, as a Value type, select Date. For the rest of the columns, leave the default values. Run a final preview. You should see this:



Calculator

Step name: date attributes

Fields:

#	New field	Calculation	Field A	Field B	Field C	Value type	Le...	Pre...	Remove	Conversion mask
1	a_single_date	Date A + B Days	start_date	delta		Date			N	yyyyMMdd
2	year	Year of date A	a_single_date			Integer			N	
3	month	Month of date A	a_single_date			Integer			N	
4	day	Day of month of date A	a_single_date			Integer			N	
5	week_day	Day of week of date A	a_single_date			Integer			N	

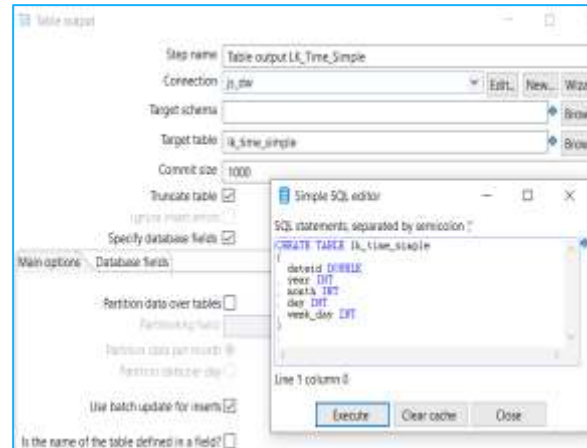
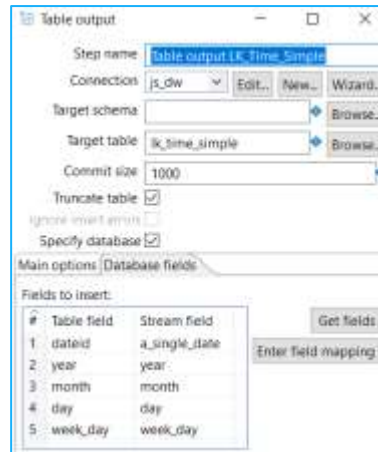
Execution Results

Execution History | Logging | Step Metrics | Performance Graph | Metrics | Preview data

First rows | Last rows | Off

#	start_date	end_date	diff_dates	delta	a_single_date	year	month	day	week_day
1	2000-01-01	2020-12-31	7670	0	20000101	2000	1	1	7
2	2000-01-01	2020-12-31	7670	1	20000102	2000	1	2	1
3	2000-01-01	2020-12-31	7670	2	20000103	2000	1	3	2
4	2000-01-01	2020-12-31	7670	3	20000104	2000	1	4	3
5	2000-01-01	2020-12-31	7670	4	20000105	2000	1	5	4
6	2000-01-01	2020-12-31	7670	5	20000106	2000	1	6	5
7	2000-01-01	2020-12-31	7670	6	20000107	2000	1	7	6
8	2000-01-01	2020-12-31	7670	7	20000108	2000	1	8	7

- l. Add two Select values steps. Use the first to change the metadata of the date from Date to String, using yyyyMMdd as the mask. Use the second step to change the metadata from String to Number using # as the mask.
- m. Finally, at the end of the stream, add a Table output step. We will use it to insert the data into the time lookup table. As a table, type lk_time_simple, click on Truncate table, click on Specify database fields and **execute the SQL to create the table output of lk_time_simple**, and fill the Database fields grid as follows:

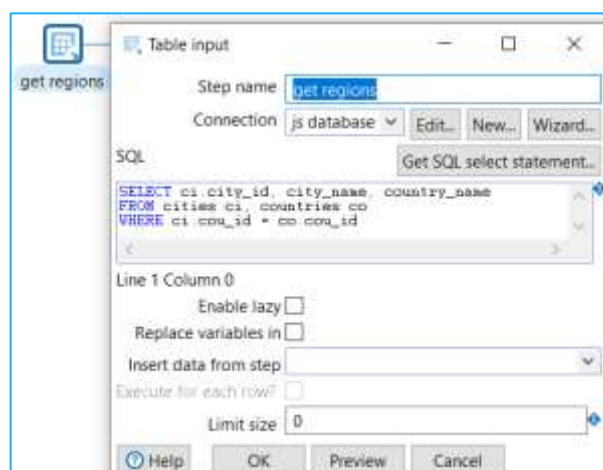


- n. Save your job into "Tugas IS545 Lab Loading Dimensions (A) yourname-NIM.ktr" and run the Transformation.

With the Database explorer or an external tool, run a SELECT on the lookup table. You should see the table populated with the dates and their attributes. This example showed a very simple version of the dimension. Depending on your business, you may need many more fields describing a date. You shouldn't have any problems in calculating new fields in this Transformation and adapting the dimension table to store them.

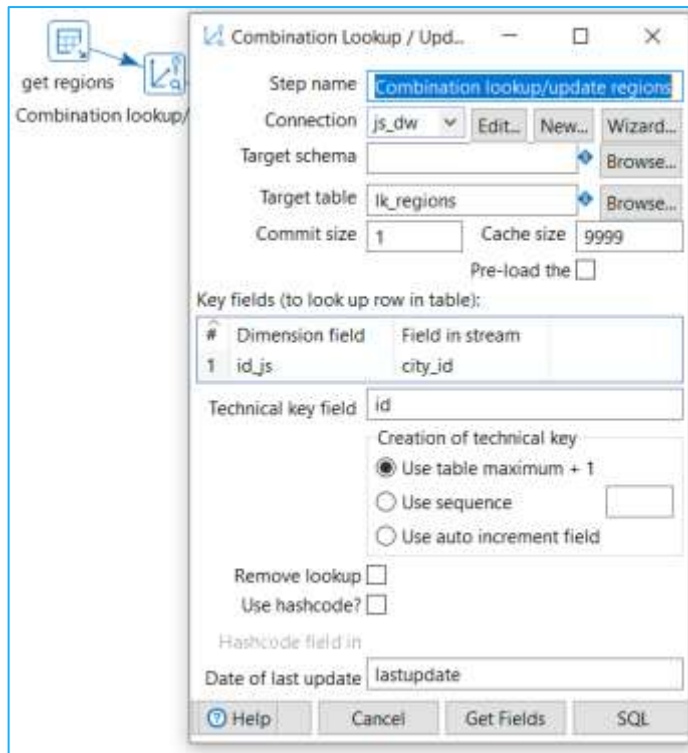
2. Loading Type 1 SCD/ Slowly Changing Dimensions

- ▶ We will explain the loading process of a Type 1 SCD by example. We will load the region dimension, which contains geographical information—cities and countries.
 - ▶ The source of our dimension will be the cities and countries tables, and the dimension table to be loaded is lk_regions. First of all, we have to gather the region information from the source:
- a. Launch Spoon and create a new Transformation. Drag a Table input step to the work area and double-click on it. As Connection, select your connection to **js database**.
 - b. In the SQL area, type the following query and click on the OK button.

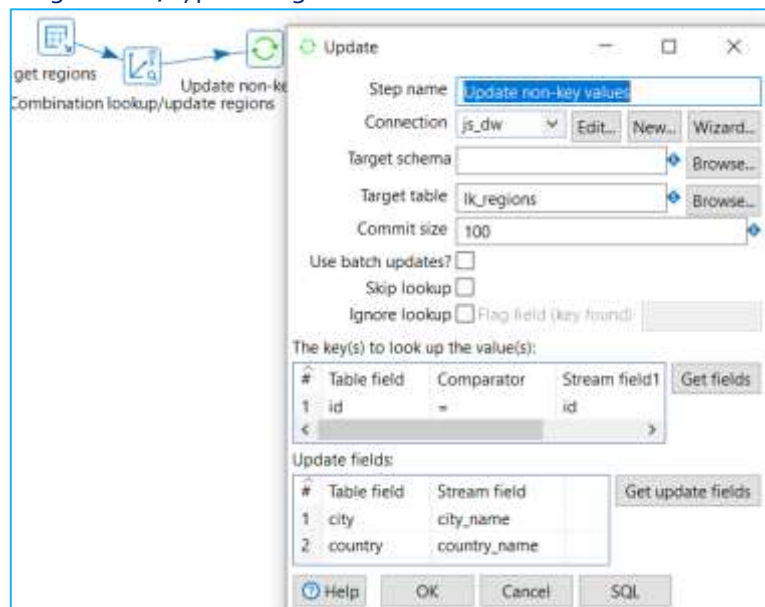


Now that you have all the data to populate the dimension, do the following:

- c. After the Table input step, add a Combination lookup/update step. You will find it in the Data Warehouse category of steps.
- d. Double-click on the step; as Connection, select dw. As Target table, browse and select lk_regions or simply type it. Set Commit size to 1. Fill the grid, as shown in the following screenshot:



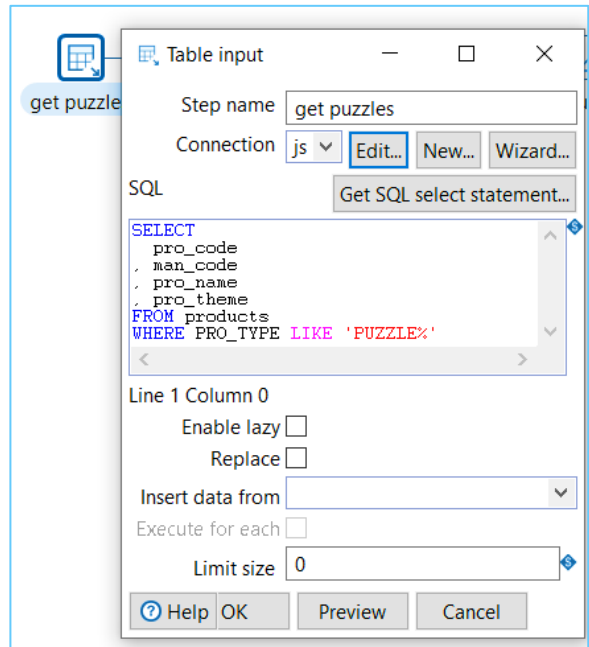
- e. As Technical key field, type id, and as Date of last update field (optional), type lastupdate.
- f. Close the window. After the Combination lookup/update step, add an Update step and doubleclick on it.
- g. As Connection, select js_dw and as Target table, type lk_regions.
- h. Fill the upper grid by adding the condition id = id. The id attribute to the left is the table's id, while id to the right is the stream id.
- i. Fill the lower grid. Add one row with the city and city_name values. Add a second row with the country and country_name values. This will update the table columns city and country with the values city_name and country_name coming in the stream.

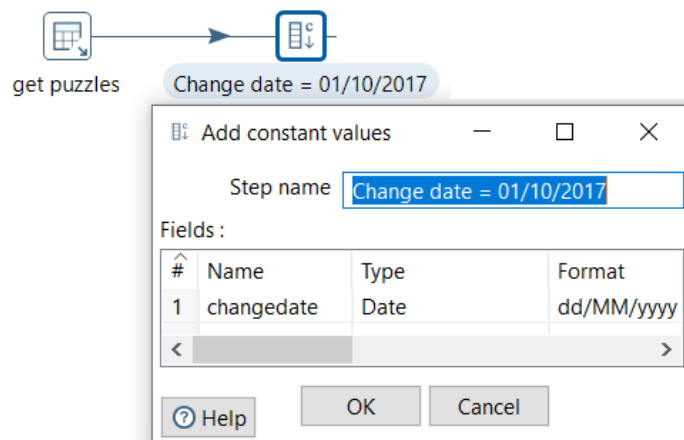


- j. Save your job into "Tugas IS545 Lab Loading Dimensions (B) yourname-NIM.ktr" and run it.

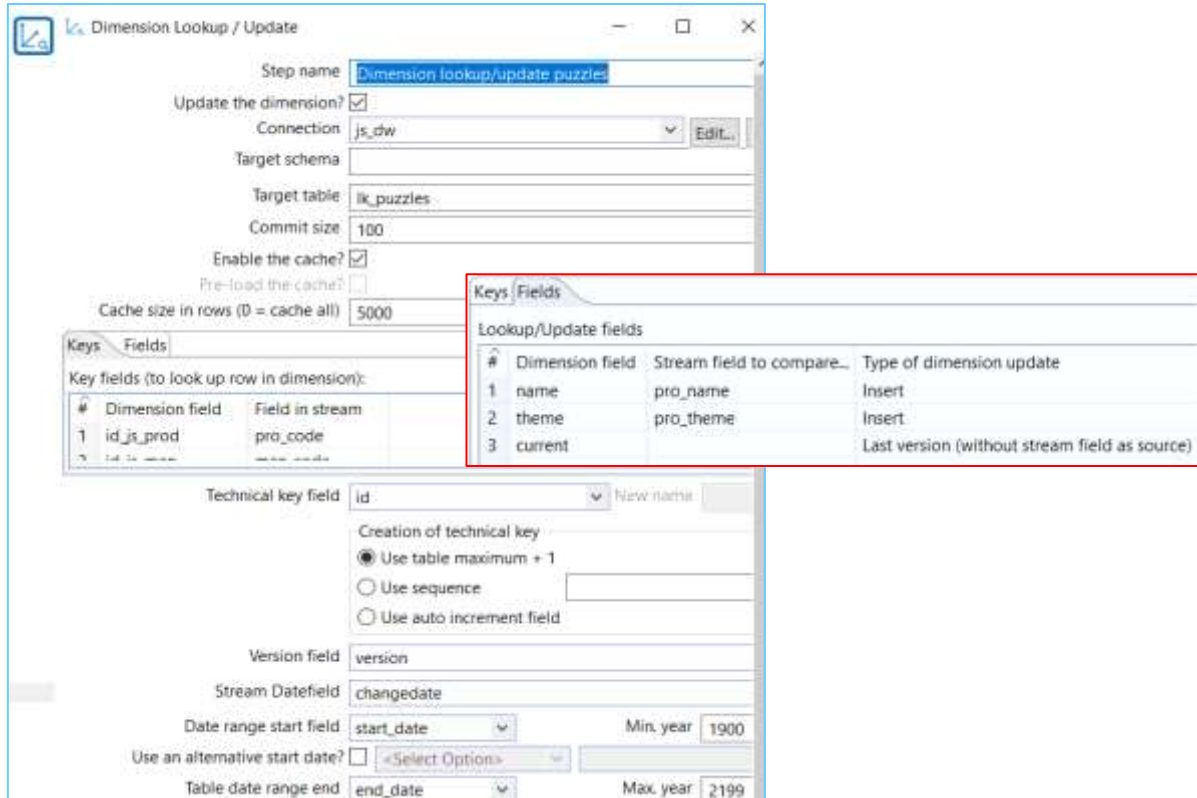
- ❖ You could see from the results; this behavior implies that the Combination lookup/update step merely inserts the row for each city, whereas the Update step is the one that adds the city and country information.
- ❖ Note that the dimension table lk_regions has a column named region that you didn't update because you don't have data for this column. The column is filled with a default value set in the DDL definition of the table.
- ❖ A commit size equal to 1 certainly degrades the performance when loading dimensions of high cardinality. If that's the case, you have an alternative way to load Type I SCD.
- ❖ The REGION dimension is a typical Type I SCD dimension. If some description changes, it makes no sense to keep the old values. The new values simply overwrite the old ones.

3. Loading Type 2 SCD/ Slowly Changing Dimensions

- ▶ Sometimes you would like to keep a history of the changes. This is when Type II Slowly Changing Dimensions come into play.
- ▶ A Type II SCD keeps the whole history of the data of your dimension. Some typical examples of attributes for which you would like to keep history are sales territories that change over time, categories of products that are reclassified from time to time, promotions that you apply to products, which are valid in a given range of dates.
- ▶ Now you will learn how to load a dimension that keeps a history. We will load the PUZZLE dimension for the first time. This is how we proceed:
 - Create a new Transformation.
 - Drag a Table input step to the work area and double-click on it. 3. As Connection, select js. In the SQL area, type the following query:
 
 - Click on OK.
 - Add an Add constants step and create a hop from the Table input step toward it.
 - Use the step to add a Date field named changedate. As format, type dd/MM/yyyy and as value, type 01/10/2017. After the Add constants step, add a Dimension lookup/update step. You will find it in the Data Warehouse category of steps.



- f. Double-click on the Dimension lookup/update step. As Connection, select js_dw database. As Target table, type lk_puzzles.
- g. Fill the Key fields, Fill the lower section of the configuration window.
- h. Select the Fields tab and fill it in with the following information:



- i. Close the setting window. Save the Transformation into "Tugas IS545 Lab Loading Dimensions (C) yourname-NIM.ktr" and press **F9** to run it .
 - j. Explore the js_dw database and do a preview of the lk_puzzles table. You should see that a special record was automatically inserted for unavailable data. In order to insert.
- ❖ This is how it works. The Dimension lookup/update step looks in the dimension table for a record that matches the information you put in the Keys grid of the setting window. Also, the period from start_date to end_date of the record must contain the value of the datefield stream, which in our sample Transformation is 01/10/2017:
 - If the lookup fails, it inserts a new record,
 - If a record is found, the step inserts or updates records depending on how you configured the step.
 - ❖ As this was the first time we run the Transformation and the table was empty, the lookup failed in all cases. As a result, the step inserted all the products.

date_range Tugas IS545 Lab Loading Dimensions A yourname-NIM

date range → Calculator → Clone row → date attributes → date to string → date to number → Table output LK_Time_1

Execution Results

Execution History | Logging | Step Metrics | Performance Graph | Metrics | Preview data

First rows | Last rows | Off

#	start_date	end_date	diff_dates	delta	a_single_date	year	month	day	week_day
1	2000-01-01	2020-12-31	7670	0	20000101	2000	1	1	7
2	2000-01-01	2020-12-31	7670	1	20000102	2000	1	2	1
3	2000-01-01	2020-12-31	7670	2	20000103	2000	1	3	2
4	2000-01-01	2020-12-31	7670	3	20000104	2000	1	4	3
5	2000-01-01	2020-12-31	7670	4	20000105	2000	1	5	4
6	2000-01-01	2020-12-31	7670	5	20000106	2000	1	6	5
7	2000-01-01	2020-12-31	7670	6	20000107	2000	1	7	6
8	2000-01-01	2020-12-31	7670	7	20000108	2000	1	8	7

Tugas IS545 Lab Loading Dimensions B yourname-NIM

get regions → Update → Update non-k → Database Explorer

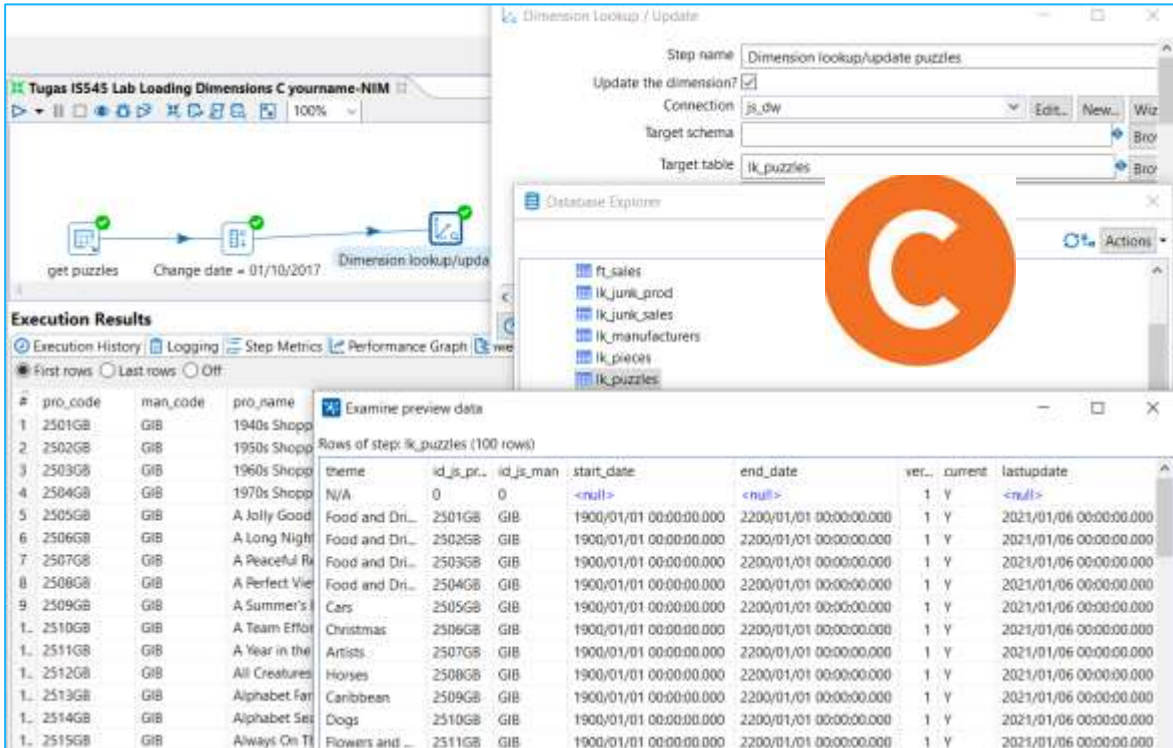
Execution Results

Execution History | Logging | Step Metrics | Performance Graph | Metrics | Preview data

First rows | Last rows | Off

Rows of step: lk_regions (100 rows)

#	id	city	country	reg...	id_js	lastupdate
1	1	Kabul	Afghanistan	N/A	1001	2021/01/10 00:00:00.000
2	2	Tirana	Albania	N/A	2001	2021/01/10 00:00:00.000
3	3	Alger	Algeria	N/A	3001	2021/01/10 00:00:00.000
4	4	Andorra ...	Andorra	N/A	4001	2021/01/10 00:00:00.000
5	5	Luanda	Angola	N/A	5001	2021/01/10 00:00:00.000
6	6	Saint Joh...	Antigua and ...	N/A	6001	2021/01/10 00:00:00.000
7	7	Buenos ...	Argentina	N/A	7001	2021/01/10 00:00:00.000
8	8	Córdoba	Argentina	N/A	7002	2021/01/10 00:00:00.000
9	9	La Plata	Argentina	N/A	7003	2021/01/10 00:00:00.000
1..	10	Rosario	Argentina	N/A	7004	2021/01/10 00:00:00.000
1..	11	Yerevan	Armenia	N/A	8001	2021/01/10 00:00:00.000
1..	12	Oranjest...	Aruba	N/A	9001	2021/01/10 00:00:00.000
1..	13	Canberra	Australia	N/A	10001	2021/01/10 00:00:00.000
1..	14	Wien	Austria	N/A	11001	2021/01/10 00:00:00.000
1..	15	Baku	Azerbaijan	N/A	12001	2021/01/10 00:00:00.000
1..	16	Nassau	Bahamas	N/A	13001	2021/01/10 00:00:00.000
1..	17	al-Mana...	Bahrain	N/A	14001	2021/01/10 00:00:00.000
1..	18	Dhaka	Bangladesh	N/A	15001	2021/01/10 00:00:00.000
1..	19	Bridgeto...	Barbados	N/A	16001	2021/01/10 00:00:00.000
2..	20	Minsk	Belarus	N/A	17001	2021/01/10 00:00:00.000



The screenshot displays the Pentaho Data Integration (Kettle) interface. The main window shows a workflow with three steps: 'get puzzles', 'Change date = 01/10/2017', and 'Dimension lookup/update'. The 'Dimension lookup/update' step is selected, and its configuration is shown in the right-hand pane. The configuration includes the step name 'Dimension lookup/update puzzles', a checked 'Update the dimension?' box, a connection to 'js_dw', a target schema of 'ik_puzzles', and a target table of 'ik_puzzles'. Below the configuration pane, the 'Database Explorer' shows a list of tables: 'ft_sales', 'ik_jurnal_prod', 'ik_jurnal_sales', 'ik_manufacturers', 'ik_pieces', and 'ik_puzzles'. The 'Execution Results' pane at the bottom shows the 'First rows' of the 'ik_puzzles' table. The table has columns: 'pro_code', 'man_code', 'pro_name', 'theme', 'id_js_pr...', 'id_js_man', 'start_date', 'end_date', 'ver...', 'current', and 'lastupdate'. The data is organized into 10 rows, each representing a different product category and its associated sales data.

#	pro_code	man_code	pro_name	theme	id_js_pr...	id_js_man	start_date	end_date	ver...	current	lastupdate
1.	2501GB	GIB	1940s Shopp	N/A	0	0	<null>	<null>	1	Y	<null>
2.	2502GB	GIB	1950s Shopp	N/A	0	0	<null>	<null>	1	Y	<null>
3.	2503GB	GIB	1960s Shopp	N/A	0	0	<null>	<null>	1	Y	<null>
4.	2504GB	GIB	1970s Shopp	N/A	0	0	<null>	<null>	1	Y	<null>
5.	2505GB	GIB	A Jolly Good	Food and Dri...	2501GB	GIB	1900/01/01 00:00:00.000	2200/01/01 00:00:00.000	1	Y	2021/01/06 00:00:00.000
6.	2506GB	GIB	A Long Night	Food and Dri...	2502GB	GIB	1900/01/01 00:00:00.000	2200/01/01 00:00:00.000	1	Y	2021/01/06 00:00:00.000
7.	2507GB	GIB	A Peaceful Rk	Food and Dri...	2503GB	GIB	1900/01/01 00:00:00.000	2200/01/01 00:00:00.000	1	Y	2021/01/06 00:00:00.000
8.	2508GB	GIB	A Perfect Vie	Food and Dri...	2504GB	GIB	1900/01/01 00:00:00.000	2200/01/01 00:00:00.000	1	Y	2021/01/06 00:00:00.000
9.	2509GB	GIB	A Summer's I	Cars	2505GB	GIB	1900/01/01 00:00:00.000	2200/01/01 00:00:00.000	1	Y	2021/01/06 00:00:00.000
1.	2510GB	GIB	A Team Effor	Christmas	2506GB	GIB	1900/01/01 00:00:00.000	2200/01/01 00:00:00.000	1	Y	2021/01/06 00:00:00.000
1.	2511GB	GIB	A Year in the	Artists	2507GB	GIB	1900/01/01 00:00:00.000	2200/01/01 00:00:00.000	1	Y	2021/01/06 00:00:00.000
1.	2512GB	GIB	All Creatures	Horses	2508GB	GIB	1900/01/01 00:00:00.000	2200/01/01 00:00:00.000	1	Y	2021/01/06 00:00:00.000
1.	2513GB	GIB	Alphabet Far	Caribbean	2509GB	GIB	1900/01/01 00:00:00.000	2200/01/01 00:00:00.000	1	Y	2021/01/06 00:00:00.000
1.	2514GB	GIB	Alphabet Ses	Dogs	2510GB	GIB	1900/01/01 00:00:00.000	2200/01/01 00:00:00.000	1	Y	2021/01/06 00:00:00.000
1.	2515GB	GIB	Always On Ti	Flowers and ...	2511GB	GIB	1900/01/01 00:00:00.000	2200/01/01 00:00:00.000	1	Y	2021/01/06 00:00:00.000

REFERENSI

-Utama-

1. María Carina Roldán. 2017. Learning Pentaho Data Integration 8 CE Third Edition. Packt Publishing.
2. Matt Casters, Roland Bouman, Jos van Dongen. 2010. Pentaho® Kettle Solutions: Building Open Source ETL Solutions with Pentaho Data Integration Wiley Publishing, Inc.
3. Pentaho Community website

-Pendukung-

4. Informasi/ Pengetahuan tambahan lainnya yang bisa didapatkan dari url/website tertentu.

MODUL 6

(MONDRIAN OLAP CUBE)



DESKRIPSI TEMA

Mahasiswa dapat menginterpretasikan Aplikasi Kubus OLAP yang terukur dengan Mondrian yang merupakan OLAP Cube Server open source menggunakan contoh Foodmart MySQL database, serta merepresentasikannya dalam online platform menggunakan Apache Tomcat JSP.

Mondrian merupakan salah satu komponen utama dari platform Intelijen Bisnis Pentaho (PBI/ Pentaho Business Intelligence), berfungsi sebagai konektor JDBC untuk OLAP, menghasilkan kueri SQL ke database dan memproses data hasil.

CAPAIAN PEMBELAJARAN MINGGUAN (SUB-CAPAIAN PEMBELAJARAN)

CMPK03-4 (C5)

Mahasiswa dapat menjelaskan pemahaman penggunaan Bahasa MDX /Multidimensional Expressions dalam rangka mendesain data cube/ kubus data dan kebutuhan kueri untuk mengekstraksi data dari kubus data.

1. Apache Tomcat Installation and Setup
2. Create MySQL Foodmart Database Sample
3. Apache Mondrian Web Application Setup
4. Apache Mondrian Web Application Deployment

PENUNJANG PRAKTIKUM

1. Windows Operating System
2. Java Runtime Environment (JRE) & Java Development Kit (JDK) 8.0 version (*installed*)
3. **Laragon** isolated dev environment on Windows (*installed*)
4. **SQLyog** management utility for MySQL databases front-end (*installed*)
5. Pentaho Data Integration tools (**PDI**) release 7.1 atau 8.0.
6. JSP Apache Tomcat 7.0.91
7. MDX Apache Mondrian-3.5.0.

LANGKAH-LANGKAH PRAKTIKUM

1. JSP Apache Tomcat Installation and Setup

Apache Tomcat is an open-source implementation of the Java Servlet, JavaServer Pages, Java Expression Language and WebSocket technologies. Tomcat provides a "pure Java" HTTP web server environment in which Java code can run.

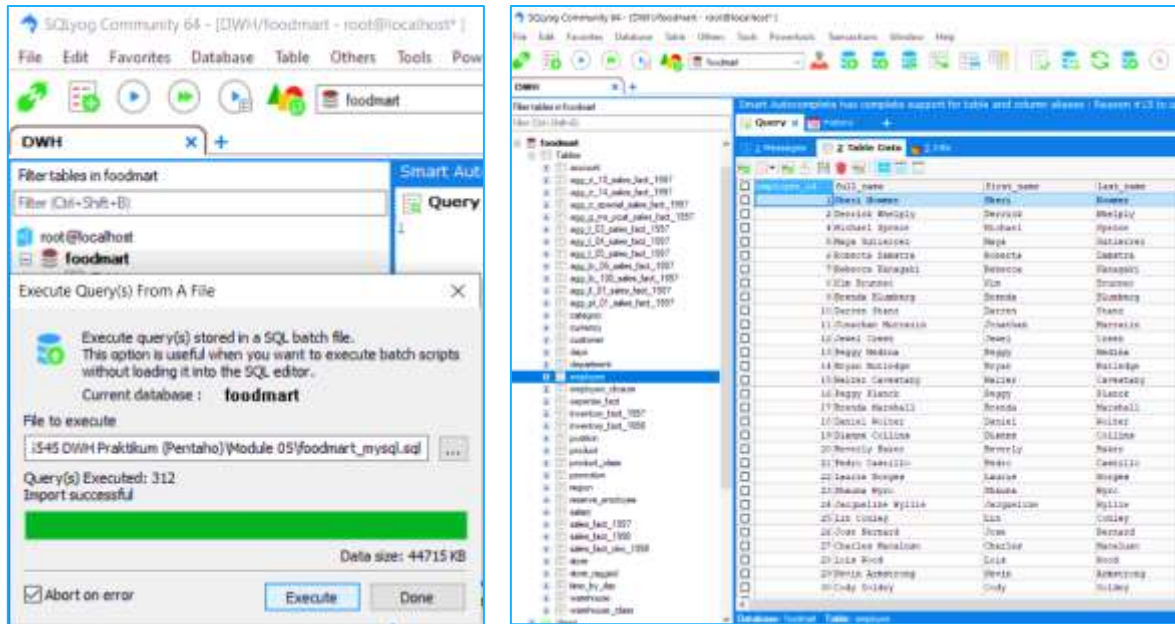
- a. Download **apache-tomcat-7.0.91-windows-x64.zip** at <http://jakarta.apache.org/site/binindex.cgi>
- b. Save and extract to your desktop on location of C:\Program Files\apache-tomcat-7.0.91
- c. In the Tomcat folder, open the bin folder then Click the startup.bat icon.
- d. You should see a black and white Java command window and you should not see any obvious java error messages.
- e. Open your browser and point to <http://localhost:8080> and you will have the Tomcat welcome page.

- f. Save your <http://localhost:8080> screenshot into "Tugas IS545 Lab Mondrian OLAP Cube (A) yourname-NIM.jpg".

2. Create MySQL Foodmart Database Sample

Mondrian provided Foodmart database sample in MS Access format. But we have provided a full generated MySQL script populate Foodmart so you use it from **IS545 foodmart_mysql.tar**.

- Extract the script file IS545 foodmart_mysql.tar using 7zip compression utility.
- Using of SQLyog, create Foodmart database and execute the foodmart_mysql.sql to data upload.

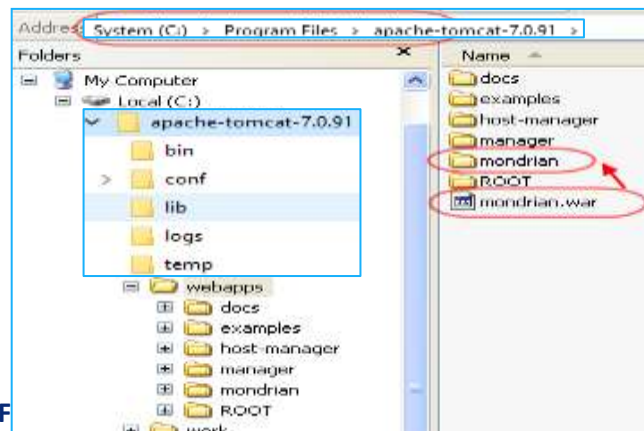


- Save the screenshot of your Foodmart database and contents into "Tugas IS545 Lab Mondrian OLAP Cube (B) yourname-NIM.jpg"

3. Apache Mondrian Web Application Setup

Apache Mondrian as an open-source olap server written in pure Java, Mondrian is part of the Pentaho Open Source BI Suite. Pentaho aims to deliver the best possible user experience by integrating Mondrian with other open-source components such as Kettle, Pentaho Reporting, and Weka.

- Download the Mondrian package from url of <https://sourceforge.net/projects/mondrian/files/mondrian/mondrian-3.5.0/mondrian-3.5.0.zip/download>.
- Copy the mondrian.war file from mondrian-3.5.0.zip\mondrian-3.5.0\lib to the webapps folder of Tomcat when the Tomcat server is running.
- After a while the server will extract / deploy the war package, your folder structure will look like the following screen:

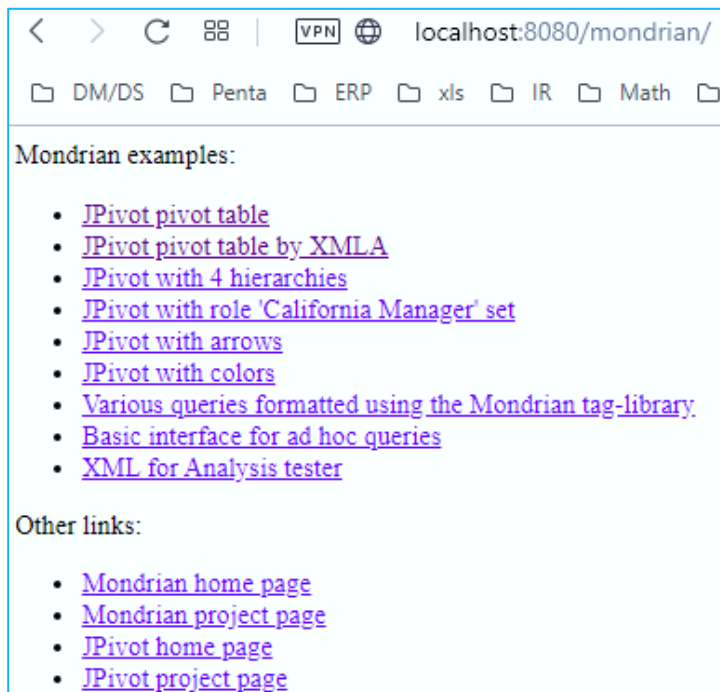


- ▶ Open the queries folder from the Mondrian application (C: \ apache-tomcat-6.0.16 \ webapps \ mondrian \ WEB-INF \ queries)
- ▶ Edit the mondrian.jsp file - which contains the MDX configuration and query tags for Mondrian - and search for the line with the following contents:
 - ▶ Change the line of


```
<jp:mondrianQuery id="query01" jdbcDriver="sun.jdbc.odbc.JdbcOdbcDriver"
jdbcUrl="jdbc:odbc:MondrianFoodMart" catalogUri="/WEB-INF/queries/FoodMart.xml">
```

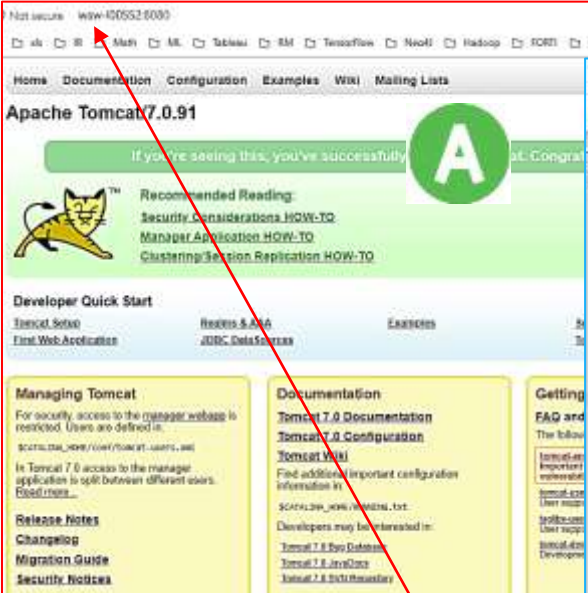
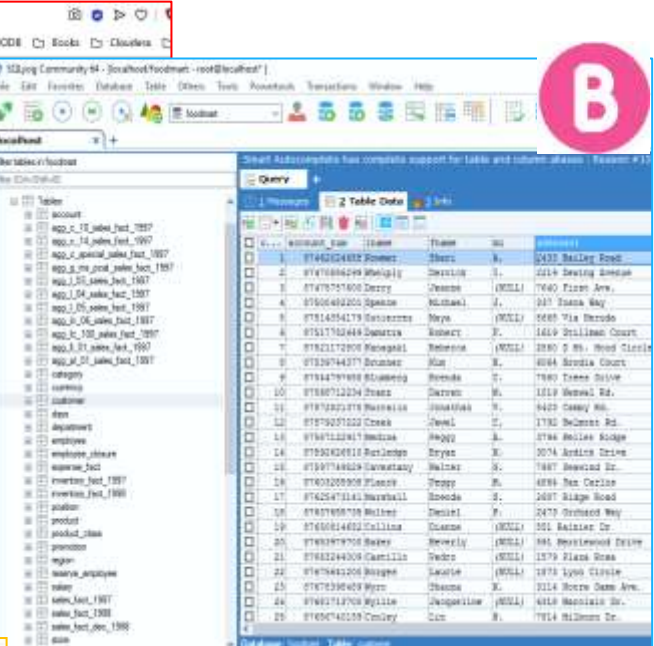
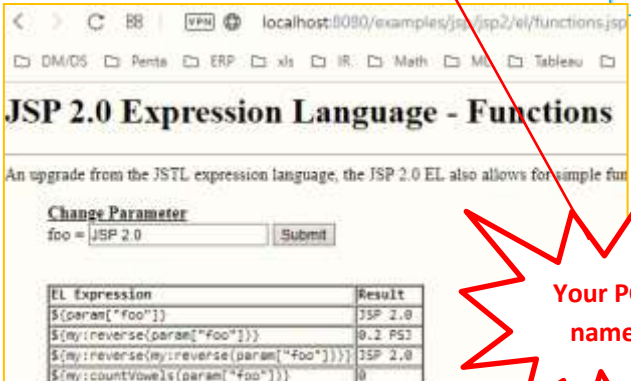
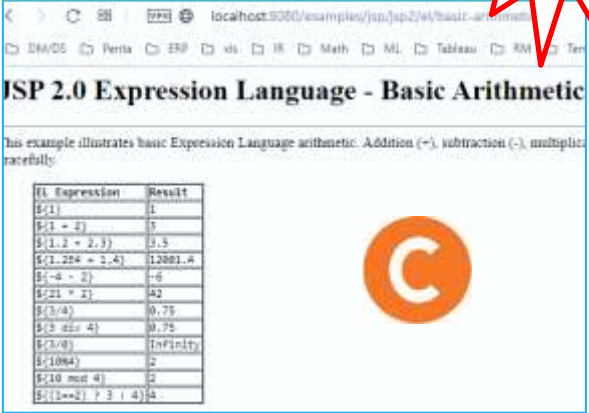

 To be


```
<jp:mondrianQuery
id="query01"
jdbcDriver="com.mysql.jdbc.Driver"
jdbcUrl="jdbc:mysql://localhost/foodmart?user=root&password="
catalogUri="/WEB-INF/queries/FoodMart.xml">
```
 - ▶ Do the same for the following jsp files:
 - i. fourheir.jsp
 - ii. colors.jsp
 - iii. arrows.js
- ▶ Restarted your Apache Tomcat.
- ▶ Now we are ready to look at our multidimensional table. In our browser, reopen the address http: // localhost: 8080 / mondrian.
- ▶ Select the first example that appears, which is "JPivot pivot table". This will retrieve the mondrian.jsp configuration file that we previously edited.



- d. Deploy some Multidimensional and Graphical reports from Foodmart database and save the screenshot of your Foodmart database and contents into "Tugas IS545 Lab Mondrian OLAP Cube (D) yourname-NIM.jpg"

HASIL/ LUARAN

Promotion Media	Product	Unit Sales	Store Cost	Store Sales
All Media	Baked Goods	7,870	6,564.09	16,455.43
	Baking Goods	20,245	15,370.61	38,670.41
	Breakfast Foods	3,317	2,756.80	6,941.46
	Canned Foods	19,026	15,894.53	39,774.34
	Canned Products	1,812	1,317.13	3,314.52
	Dairy	12,885	12,228.85	30,508.85
	Deli	12,037	10,108.87	25,318.93
	Eggs	4,132	3,684.90	9,200.76
	Frozen Foods	26,655	22,030.66	55,207.50
	Meat	1,714	1,465.42	3,669.89
	Produce	37,792	32,831.33	82,248.42
	Seafood	1,764	1,520.70	3,809.14
	Snack Foods	30,545	26,963.34	67,609.82
	Snacks	6,884	5,827.58	14,550.05
	Starchy Foods	5,262	4,705.91	11,756.07

EL Expression	Result
$\$(1)$	1
$\$(1 + 2)$	3
$\$(1.2 + 2.3)$	3.5
$\$(1.254 + 2.4)$	3.654
$\$(-4 - 2)$	-6
$\$(21 * 2)$	42
$\$(3/4)$	0.75
$\$(3 div 4)$	0.75
$\$(3/0)$	Infinity
$\$(10/4)$	2.5
$\$(10 mod 4)$	2
$\$(1==2) ? 3 : 4)$	4

Year	Unit Sales	Store Cost	Store Sales
1997	~15,000	~10,000	~25,000
1998	~20,000	~12,000	~32,000
1999	~25,000	~15,000	~40,000
2000	~30,000	~18,000	~48,000
2001	~35,000	~20,000	~55,000
2002	~40,000	~22,000	~62,000
2003	~45,000	~24,000	~69,000
2004	~50,000	~26,000	~76,000
2005	~55,000	~28,000	~83,000
2006	~60,000	~30,000	~90,000
2007	~65,000	~32,000	~97,000
2008	~70,000	~34,000	~104,000
2009	~75,000	~36,000	~111,000
2010	~80,000	~38,000	~118,000
2011	~85,000	~40,000	~125,000
2012	~90,000	~42,000	~132,000
2013	~95,000	~44,000	~139,000
2014	~100,000	~46,000	~146,000
2015	~105,000	~48,000	~153,000
2016	~110,000	~50,000	~160,000
2017	~115,000	~52,000	~167,000
2018	~120,000	~54,000	~174,000
2019	~125,000	~56,000	~181,000
2020	~130,000	~58,000	~188,000

REFERENSI

-Utama-

1. Matt Casters, Roland Bouman, Jos van Dongen. 2010. Pentaho® Kettle Solutions: Building Open Source ETL Solutions with Pentaho Data Integration Wiley Publishing, Inc.
2. Pentaho Community website

MODUL 7

(INTRODUCING PARTITIONING and CLUSTERING)



DESKRIPSI TEMA

- ▶ Pembelajaran penggunaan metode pembagian /partitioning standar yang tersedia di PDI menggunakan pemilihan bidang pemartisian, dan PDI membagi nilainya dengan jumlah partisi yang telah ditentukan sebelumnya.
- ▶ Sebagai contoh, mahasiswa dapat membuat partisi skema dengan tiga partisi dan memilih Severity sebagai bidang partisi. Kemudian, PDI akan menghitung sisa nilai pembagian sebanyak tiga bidang yang merupakan jumlah partisi. PDI akan melakukan perhitungan pada checksum yang dibuat dari nilai Severity.

CAPAIAN PEMBELAJARAN MINGGUAN (SUB-CAPAIAN PEMBELAJARAN)

CMPKo4-1 (C4)

Mahasiswa dapat memahami teknik dan strategi partisi pada proses desain fisik, yang bertujuan untuk meningkatkan kinerja dan pengelolaan Data Warehouse, fokus pada implementasi relasional data cubes, dengan tiga teknik umum untuk meningkatkan kinerja dalam sistem Data Warehouse: materialized views, indexing, dan partitioning.

"Splitting a stream into two or more streams based on a condition"

1. Data streams search and Keeping_just_true_rows
2. Searching by Splitting a stream into two or more streams based on a condition
3. Optimize process by Avoiding the Use of Nested Filter Rows Steps
4. Overcoming the Difficulties of Complex Conditions: Splitting by java expression

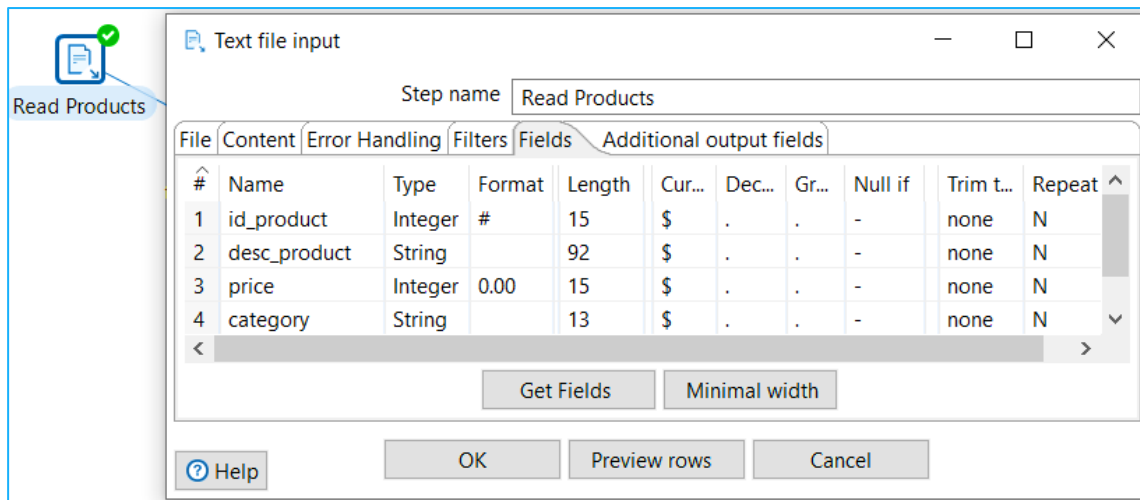
PENUNJANG PRAKTIKUM

1. Windows Operating System
2. Java Runtime Environment (JRE) & Java Development Kit (JDK) 8.0 version (*installed*)
3. Laragon isolated dev environment on Windows (*installed*)
4. SQLyog management utility for MySQL databases front-end (*installed*)
5. Pentaho Data Integration tools (PDI) release 7.1 atau 8.0.

LANGKAH-LANGKAH PRAKTIKUM

1. Data streams search and Keeping_just_true_rows

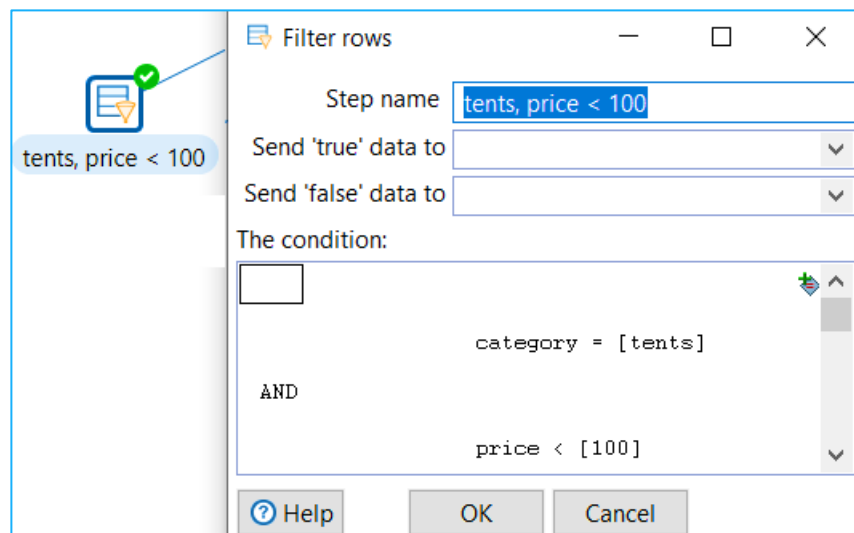
- ▶ Learn to use the Filter rows step in order to split a single stream into different smaller streams.
 - ▶ See alternative and more efficient ways for doing the same thing in different scenarios.
 - ▶ Let's assume that you have a set of outdoor products in a text file, and you want to differentiate tents from other kinds of products, and also create a subclassification of the tents depending on their prices.
- a. Create a transformation.
 - b. Drag a Text file input step into the canvas and fill in the File tab to read the file named outdoorProducts.txt. If you are using the sample text file, type, as the Separator.
 - c. Under the Fields tab, use the Get Fields button to populate the grid. Adjust the entries so that the grid looks like the one shown in the following screenshot:



d. Now, let's add the steps to manage the flow of the rows. To do this, drag two Filter rows steps from the Flow category. Also, drag three Dummy steps that will represent the three resulting streams.

e. Create the hops, when you create the hops, make sure that you choose the options according to the image: Result is TRUE for creating a hop with a green icon, and Result is FALSE for creating a hop with a red icon in it.

f. Double-click on the first Filter rows step and complete the condition, as shown in the following screenshot:



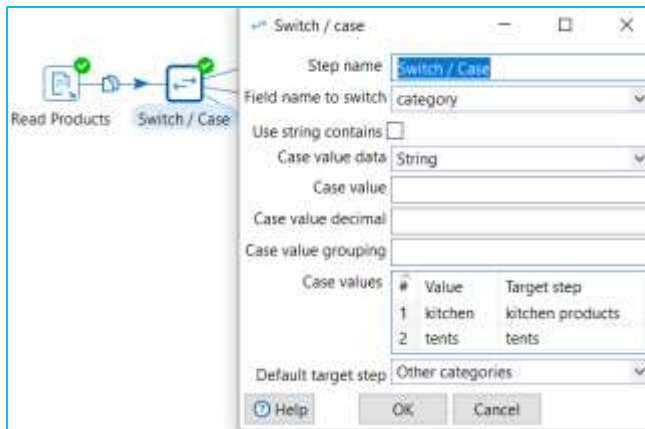
g. Double-click on the second Filter rows step and complete the condition with price < 100.

h. You have just split the original dataset into three groups. You can verify it by previewing each Dummy step. The first one has products whose category is not tents; the second one, the tents under 100 US\$; and the last group, the expensive tents, those whose price is over 100 US\$.

Save the Transformation into "Tugas IS545 Lab Partitioning and Clustering (A) yourname-NIM.ktr" and press F9 to run it.

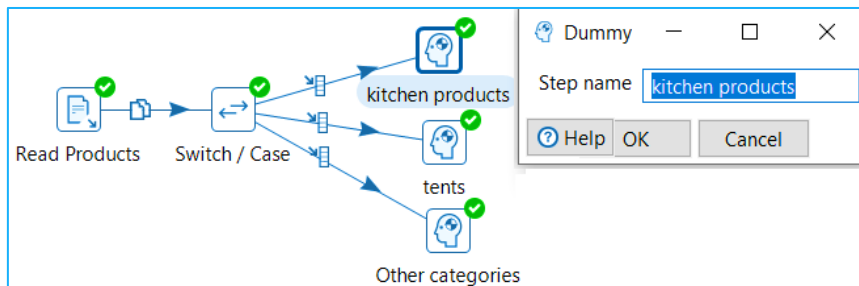
2. Searching by Splitting a stream into two or more streams based on a condition

- ▶ Let us assume that you have to send the rows to different steps depending on the category.
- ▶ The best way to do this is with the Switch / Case step. This way you avoid adding one Filter rows step for each category
- a. In this step, you have to select the field to be used for comparing. You do it in the Field name to switch listbox. In the Case values grid, you set the Value-Target step pairs. The following screenshot shows how to fill in the grid for our particular problem:



b. The following are some considerations about this step:

- ▶ You can have multiple values directed to the same target step.
- ▶ You can leave the value column blank to specify a target step for empty values.
- ▶ You have a listbox named Default target step to specify the target step for rows that do not match any of the case values.
- ▶ You can only compare with an equal operator.



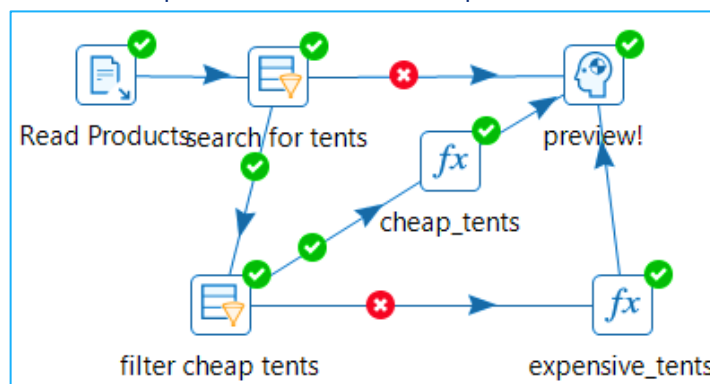
Save the Transformation into "Tugas IS545 Lab Partitioning and Clustering (B) yourname-NIM.ktr" and press F9 to run it.

3. Optimize process by Avoiding the Use of Nested Filter Rows Steps

- ▶ Suppose that you have to split the streams simply to set some fields and then join the streams again.
- ▶ For example, assume that you want to change the category as follows:

Condition	New category
Category equal to tents and price below 100	cheap_tents
Category equal to tents and price above or equal to 100	expensive_tents
Category different from tents	Keep the old category

a. Doing this with a simple nested Filter rows steps leads to a transformation, such as the following:

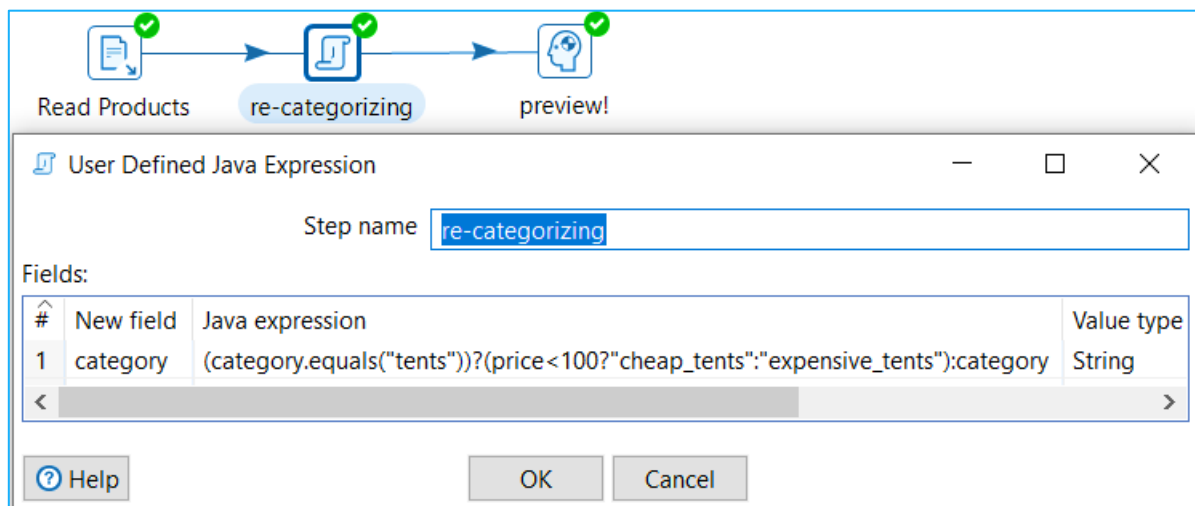


Save the Transformation into "Tugas IS545 Lab Partitioning and Clustering (C) yourname-NIM.ktr" and press F9 to run to get the results.

4. Overcoming the Difficulties of Complex Conditions: Splitting by java expression

You can do Tugas IS545 Lab Partitioning and Clustering (C) yourname-NIM.ktr with the same thing in a simpler way:

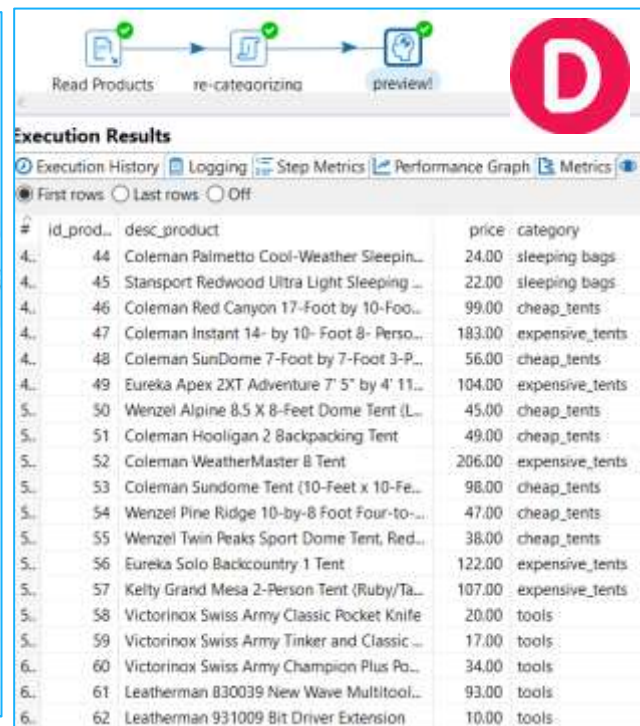
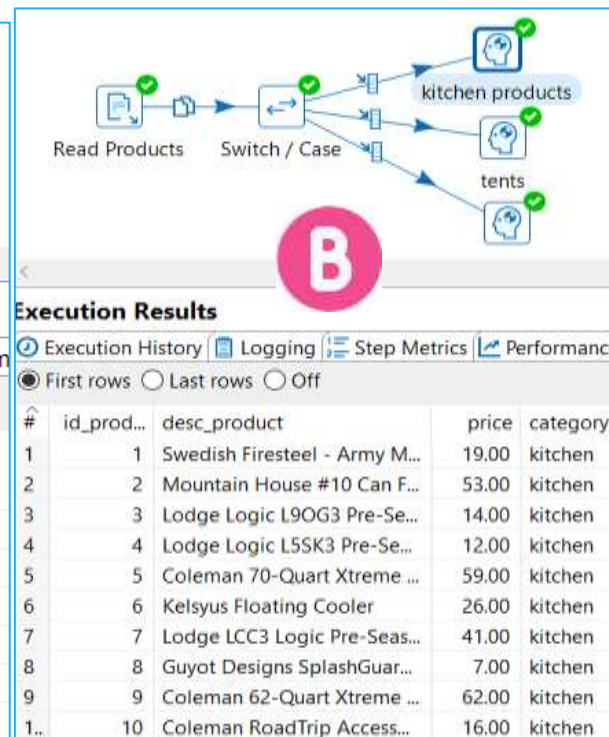
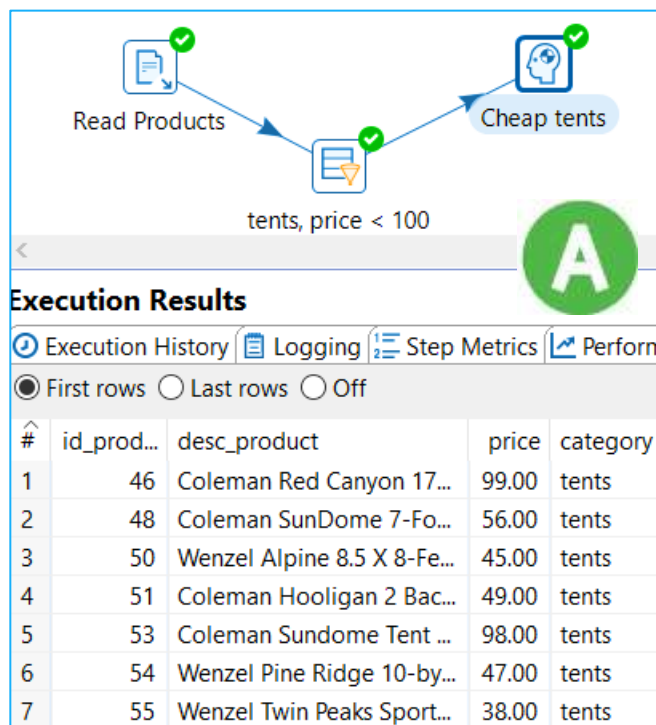
- ▶ Replace all the steps, but fill the Text file input with a User Defined Java Expression step located in the Scripting category
- a. Drag and use a User Defined Java Expression step located in the Scripting category.
- b. In the setting window of this step, add a row in order to replace the value of the category field: as New field and Replace value type category.
- c. As Value type select String. As Java expression, type the following:



- d. The preceding expression uses the Java ternary operator?
- e. If you're not familiar with the syntax, think of it as shorthand for the if-then-else statement.
- f. For example, the inner expression `price<100?"cheap_tents":`
 - ▶ `"expensive_tents"` means if (`price<100`) then return `"cheap_tents"`
 - ▶ else return `"expensive_tents"`

Save the Transformation into "Tugas IS545 Lab Partitioning and Clustering (D) yourname-NIM.ktr" and press F9 to run it.

HASIL/ LUARAN



REFERENSI

-Utama-

1. María Carina Roldán. 2017. Learning Pentaho Data Integration 8 CE Third Edition. Packt Publishing.
2. Matt Casters, Roland Bouman, Jos van Dongen. 2010. Pentaho® Kettle Solutions: Building Open Source ETL Solutions with Pentaho Data Integration Wiley Publishing, Inc.
3. Pentaho Community website

MODUL 8

(OPTIMIZING DATA FLOWS)



DESKRIPSI TEMA

Penerapan peningkatan kinerja dan pengelolaan Data Warehouse, melalui optimalisasi aliran data menggunakan teknik menggabungkan baris dari dua aliran data dengan struktur/ format data yang sama atau berbeda yang. Dalam kasus ini, aliran berasal dari sumber yang berbeda dan tidak selalu memiliki struktur yang sama, akibatnya, menggabungkan aliran data tidak semudah hanya membuat langkah penggabungan/ join aliran data dengan bebas.

CAPAIAN PEMBELAJARAN MINGGUAN (SUB-CAPAIAN PEMBELAJARAN)

CMPK04-2 (C4)

Mahasiswa dapat memahami teknik dan strategi partisi pada proses desain fisik, yang bertujuan untuk meningkatkan kinerja dan pengelolaan Data Warehouse, melalui mekanisme pengindeksan alternatif digunakan seperti: bitmap dan join indexes. Terakhir, partisi atau fragmentasi yang membagi konten relasi menjadi beberapa file, biasanya berdasarkan kisaran nilai dari suatu atribut.

“Unstructured Data: Merging rows of two streams with the same or different structures”

1. Merging rows of two streams the different structures
2. Comparing two streams and generating differences

PENUNJANG PRAKTIKUM

1. Windows Operating System
2. Java Runtime Environment (JRE) & Java Development Kit (JDK) 8.0 version (*installed*)
3. **Laragon** isolated dev environment on Windows (*installed*)
4. **SQLyog** management utility for MySQL databases front-end (*installed*)
5. Pentaho Data Integration tools (**PDI**) release 7.1 atau 8.0

LANGKAH-LANGKAH PRAKTIKUM

1. Merging rows of two streams the different structures

- ▶ The Issues can quickly arise if row formats and column orders are mixed between streams. This recipe gives you the tips to make it easier.
- ▶ Suppose that you received data about roller coasters from two different sources. The data in one of those sources looks like the following:

roller_coaster|speed|park|location|country|YearTop Thrill

Dragster|120 mph|Cedar Point|Sandusky, Ohio||2003Dodonpa|106.8
mph|Fuji-Q Highland|FujiYoshida-shi|Japan|2001Steel Dragon 2000|95
mph|Nagashima Spa Land|Mie|Japan|2000Millennium Force|93 mph|Cedar
Point|Sandusky, Ohio||2000Intimidator 305|90 mph|Kings
Dominion|Doswell, Virginia||2010Titan|85 mph|Six Flags Over
Texas|Arlington, Texas||2001Furious Baco|84
mph|PortAventura|Spain||2007...

- ▶ The other source data looks like the following:
`attraction|park_name|top_speed|trains_qt|ride_timeExpedition`
`Everest|Disney's Animal Kingdom|50 mph|6 - 34 passenger|Goofy's`
`Barnstormer|Disney's Magic Kingdom|25 mph|2 - 16`
`passenger|Gwazi|Busch Gardens Tampa|50 mph|4 - 24 passenger|2`
`minutes, 30 secondsJourney To Atlantis|SeaWorld Orlando||8 passenger`
`boats|Kraken|SeaWorld Orlando|65 mph|3 - 32 passenger|2 minutes, 2`
`seconds...`
- ▶ When you need to merge the rows of two streams into a single stream, you have to do all you can to make the metadata of the streams alike. That's what you did in this recipe. In the first stream, you added the fields you needed that were absent. You also selected and reordered the desired fields to resemble the second stream. After that, you changed the metadata of the top_speed field in the second stream. You converted the field from Integer to Number, which was the type of the analogous field in the first stream.
- ▶ We want to merge those rows into a single dataset with the following columns:
 - attraction
 - park_name
 - speed
 - trains_qt
 - ride_time
 - a. Download the files [roller_coasters_I.txt](#) and [roller_coasters_II.txt](#) from the UMN e-Learning.
 - b. Create a transformation and drag two Text file input steps into the canvas.
 - c. Use one of the steps to read the file [roller_coasters_I.txt](#). Set the data types as follows: The speed as a Number with Format #0.### mph, and the rest of the fields as String. Do a preview to make sure that you are reading the file properly.
 - d. Drag the cursor over the step and press Space to see the output fields:

#	Fieldname	Type	L...	P...	Step origin	Stor...	Mask	D...	G...	Trim	Comments
1	roller_coaster	String	-	-	roller_coasters_I	normal				none	
2	speed	Number	-	-	roller_coasters_I	normal	#0.### mph			none	
3	park	String	-	-	roller_coasters_I	normal				none	
4	location	String	-	-	roller_coasters_I	normal				none	
5	country	String	-	-	roller_coasters_I	normal				none	
6	year	String	-	-	roller_coasters_I	normal				none	

- e. Use the other step to read the file [roller_coasters_II.txt](#). Set the data type of top_speed to Integer and the rest of the fields to String. Do a preview to make sure that you are reading the file properly.
- f. Drag the cursor over the step and press Space to see the output fields:

#	Fieldname	Type	L...	P...	Step origin	Storage	Mask	D...	G...	Trim	Comments
1	attraction	String	-	-	roller_coasters_II	normal				none	
2	park_name	String	-	-	roller_coasters_II	normal				none	
3	top_speed	Integer	-	0	roller_coasters_II	normal				none	
4	trains_qt	String	-	-	roller_coasters_II	normal				none	
5	ride_time	String	-	-	roller_coasters_II	normal				none	

As you can see, the outputs of the streams are different. You have to insert the necessary steps to make them alike. That's the purpose of the next steps.

- g. After the first Text file input step add an Add constants step. Use it to add two fields of typeString. Name the fields trains_qt and ride_time and as Value, type Not available.

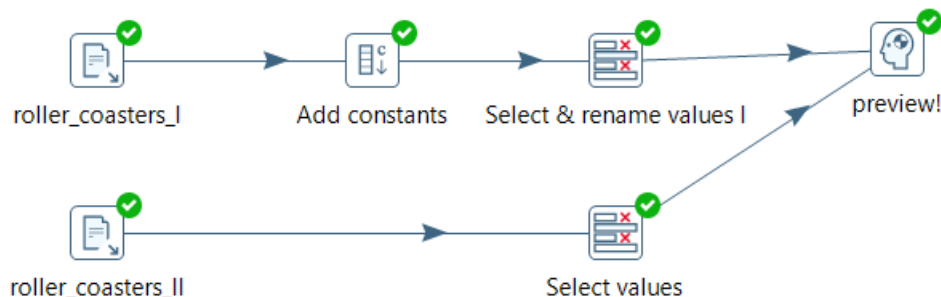
- h. After it, add a Select values step. Fill in the Select & Alter tab as shown in the following screenshot:

#	Fieldname	Rename to	Length	Precision
1	Roller_Coaster	attraction		
2	park	park_name		
3	Speed			
4	trains_gt			
5	ride_time			

- i. After the second Text file input step, add another Select values step. Select the Meta-data tab and fill it in as shown in the following screenshot:

#	Fieldname	Rename to	Type	Length	Precision	Bina...	Format
1	top_speed	speed	Number			N	#0.### mph

- j. Repeat the procedure to see the output fields of the streams: Drag the cursor over the last step of each stream and press Space. Now, both streams should have the same layout.
- k. Finally, join the streams with a Dummy step as depicted in the following diagram:



- l. Do a preview on the Dummy step.
- m. Save the Transformation into "Tugas IS45 Lab Optimizing Data Flows (A) yourname-NIM.ktr" and press Fg to run it. You will see something similar to the result.

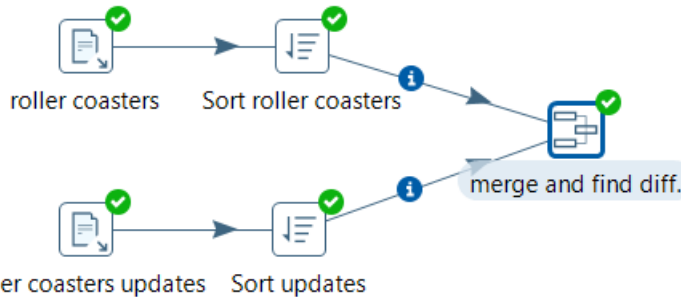
2. Comparing two streams and generating differences

- Suppose that you have two streams with the same structure and want to find out the differences in the data and you have a file with information about the fastest roller coasters around the world.
- Now, you get an updated file and want to find out the differences between the files: There can be new roller coasters in the list; maybe some roller coasters are no longer among the fastest. Besides, you were told that in the old file, there were some errors about the location, country, and year information, so you are also interested in knowing if some of these have changed.

- a. Both files have the same structure and look like the following:

```
Roller_Coaster|Speed|park|location|country|Year
Kingda Ka|128 mph|Six Flags Great Adventure|Jackson, New
Jersey||2005
Top Thrill Dragster|120 mph|Cedar Point|Sandusky, Ohio||2003
Dodonpa|106.8 mph|Fuji-Q Highland|FujiYoshida-shi|Japan|2001
Steel Dragon 2000|95 mph|Nagashima Spa Land|Mie|Japan|2000
Millennium Force|93 mph|Cedar Point|Sandusky, Ohio||2000
...
```

- b. Create a transformation. Drag a Text file input step into the canvas and use it to read the filetop_roller_coasters.txt. As a separator, type |.
- c. Do a preview to make sure that you are reading the file as expected.
- d. Add a Sort rows step to sort the rows by roller_coaster and park.
- e. Repeat the steps 2.a to 2.d to read the file named top_roller_coasters_updates.txt and sort the rows also by roller_coaster and park.
- f. From the Join category, add a Merge Rows (diff) step and use it to join both streams as depicted in the following diagram:



- g. Double-click on the step you just added. In the Reference rows origin: select the name of the step coming from the stream that reads the top_roller_coasters.txt file.
- h. In the Compare rows origin: select the name of the step coming from the stream that reads thetop_roller_coasters_updates.txt file.
- i. As Flag fieldname, type flag.
- j. Fill the Keys to match: and Values to compare: grids as shown in the following screenshot:

Keys to match :		Values to compare :	
#. ▲	Key field	#. ▲	Value field
1	roller_coaster	1	location
2	park	2	country
		3	year

- k. Close the window. Save the Transformation into "Tugas IS545 Lab Optimizing Data Flows (B) yourname-NIM.ktr" and press F9 to run it.
- l. The Merge Rows (diff) step is used for comparing two streams and finding out the differences between them. The output of the step is a single stream.
- m. The output stream contains a new field that acts as a flag indicating the kind of difference found as below explanation:

Result of the comparison	Flag	Example
The key was only found in the reference stream	new	Formula Rossa roller coaster
The key was only found in the compared stream	deleted	Colossos roller coaster
The key was found in both streams and the fields typed in the Value to compare grid are equal	identical	Millennium Force roller coaster. The location (Sandusky,Ohio), country (empty), andyear (2000) were the same in both streams.
The key was found in both streams but at least one of the fields typed	changed	Furious Baco roller coaster. The location changed from Spain toSalou and the Country changed from empty to Spain.

Execution Results

☒ Execution History
 ☐ Logging
 ☐ Step Metrics
 ☐ Performance Graph
 ☐ Metrics
 ☐ Preview data

☐ First rows
 ☒ Last rows
 ☐ Off

#	attraction	park_name	speed	trains_qt	ride_time
4	Journey To Atlantis	SeaWorld Orlando	<null>	8 passenger boats	<null>
5	Kraken	SeaWorld Orlando	65 mph	3 - 32 passenger	2 minutes, 2 seconds
6	Kumba	Busch Gardens Tampa	60 mph	4 - 32 passenger	2 minutes, 54 seconds
7	Manta	SeaWorld Orlando	56 mph	3 - 32 passenger	2 minutes, 35 seconds
8	Montu	Busch Gardens Tampa	60 mph	3 - 32 passenger	3 minutes
9	Shamu Express	SeaWorld Orlando	28 mph	1 - 28 passenger	<null>
1..	Sheikra	Busch Gardens Tampa	70 mph	5 - 24 passenger	3 minutes
1..	Top Thrill Dragster	Cedar Point	120 mph	<null>	<null>
1..	Dodonpa	Fuji-Q Highland	106.8 mph	<null>	<null>
1..	Steel Dragon 2000	Nagashima Spa Land	95 mph	<null>	<null>
1..	Millennium Force	Cedar Point	93 mph	<null>	<null>
1..	Intimidator 305	Kings Dominion	90 mph	<null>	<null>
1..	Titan	Six Flags Over Texas	85 mph	<null>	<null>

Execution Results

☒ Execution History
 ☐ Logging
 ☐ Step Metrics
 ☐ Performance Graph
 ☐ Metrics
 ☐ Preview data

☒ First rows
 ☐ Last rows
 ☐ Off

#	roller_coaster	speed	park	location	country	year	flag
1	Colossos	74.6 mph	Heide Park	Soltau	Germany	2001	deleted
2	Dodonpa	106 mph	Fuji-Q Highland	FujiYoshida-shi	Japan	2001	identical
3	Extreme Rusher	84 mph	Happy Valley	Beijing	China	<n...	new
4	Formula Rossa	149 mph	Ferrari World	Dubai	United Arab ...	2010	new
5	Fujiyama	81 mph	Fuji-Q Highland	FujiYoshida-shi	Japan	1996	deleted
6	Furious Baco	84 mph	Port Aventura	Salou	Spain	2007	changed
7	Goliath	85 mph	Six Flags Magic Mountain	Valencia, California	<null>	2000	identical
8	Intimidator 305	90 mph	Kings Dominion	Doswell, Virginia	<null>	2010	identical
9	Kingda Ka	128 mph	Six Flags Great Adventure	Jackson, New Jers...	<null>	2005	identical
1..	Millennium Force	93 mph	Cedar Point	Sandusky, Ohio	<null>	2000	identical
1..	Phantom's Revenge	85 mph	Kennywood	West Mifflin, Penn...	<null>	2001	identical
1..	Son of Beast	78.3 mph	Kings Island	Cincinnati, Ohio	<null>	2000	deleted
1..	Steel Dragon 2000	95 mph	Nagashima Spa Land	Mie	Japan	2000	identical
1..	Thunder Dolphin	81 mph	Tokyo Dome City	Tokyo	Japan	2003	deleted
1..	Titan	85 mph	Six Flags Over Texas	Arlington, Texas	<null>	2001	identical
1..	Top Thrill Dragster	120 mph	Cedar Point	Sandusky, Ohio	<null>	2003	identical
1..	Xcelerator	82 mph	Knott's Berry Farm	Buena Park, Califo...	<null>	2002	deleted

REFERENSI

-Utama-

1. María Carina Roldán, Adrián Sergio Pulvirenti. 2011. Pentaho Data Integration 4 Cookbook. Packt Publishing.
2. Matt Casters, Roland Bouman, Jos van Dongen. 2010. Pentaho® Kettle Solutions: Building Open Source ETL Solutions with Pentaho Data Integration Wiley Publishing, Inc.
3. Pentaho Community website

-Pendukung-

4. Informasi/ Pengetahuan tambahan lainnya yang bisa didapatkan dari url/website tertentu.