Data Analysis and Integration Project

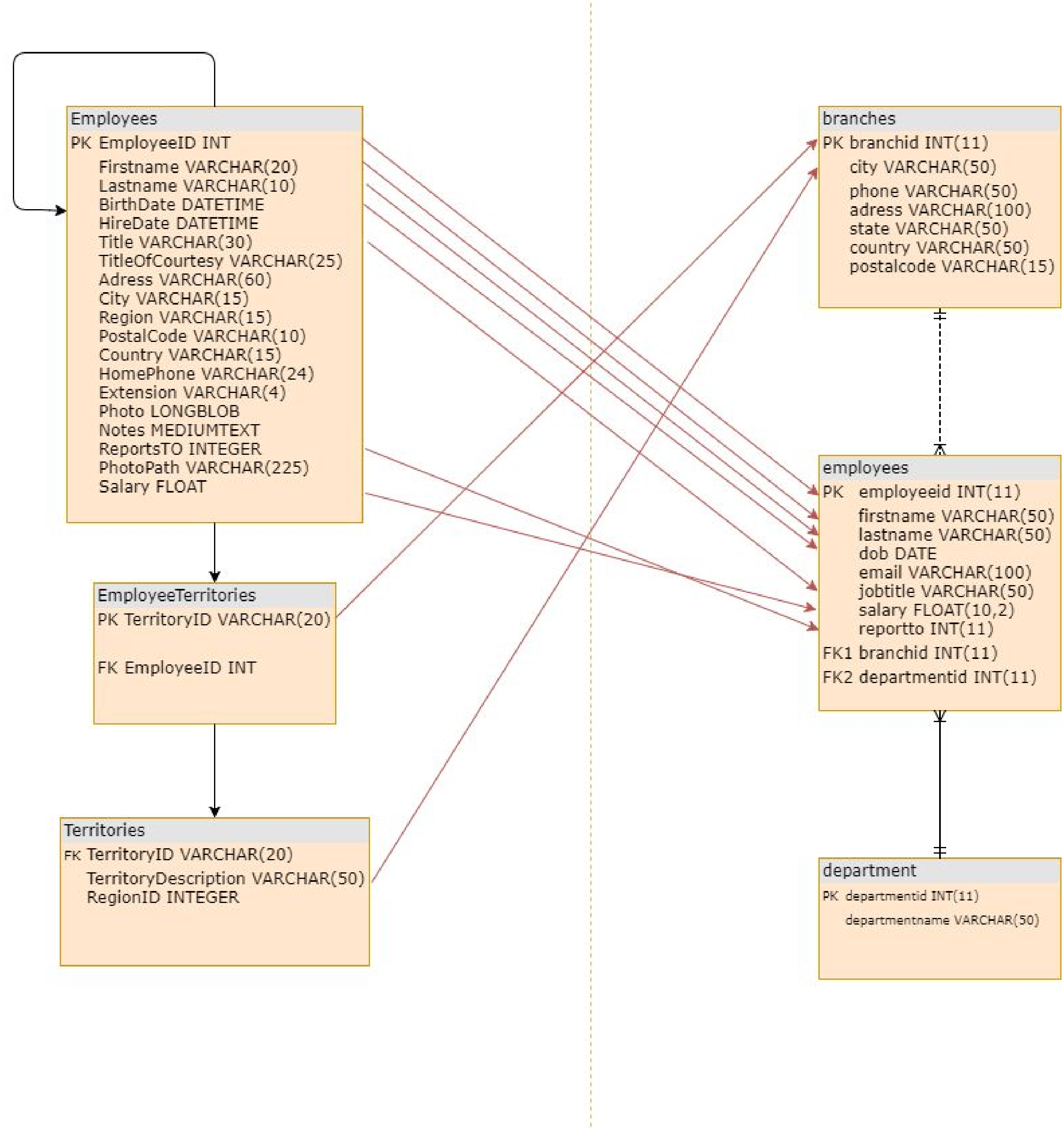
Group: 13

Name: Manuel Fernandes	**N:**76002
Name: Mário Reis	**N:**70969
Name: César Gonçalves	**N:** 78019

1.Schema Matching

Here we present a diagram with the correspondences between each table/column in the “Northwind” database and each table/column in the “company” database:

Northwind database **company Database**



2. Mediated Schema

We created the following common (mediated) schema between both databases:

Views	Description
All_Employees(EmployeeID,FirstName,LastName,BirthDate,ReportsTo,Title,Salary)	Returns the list of all employees from both databases
Works_In(EmployeeID,BranchID,City)	Returns the list where all the employees work from both databases

3. SQL Views

Based in the mediated schema, we created the schema mapping (i.e. views) to retrieve data from both databases at the same time.

To create the first view, All_Employees(employeeid, firstname, lastname, dob, reportto, jobtitle, salary, territoryid) to retrieve the list of all employees from both databases:

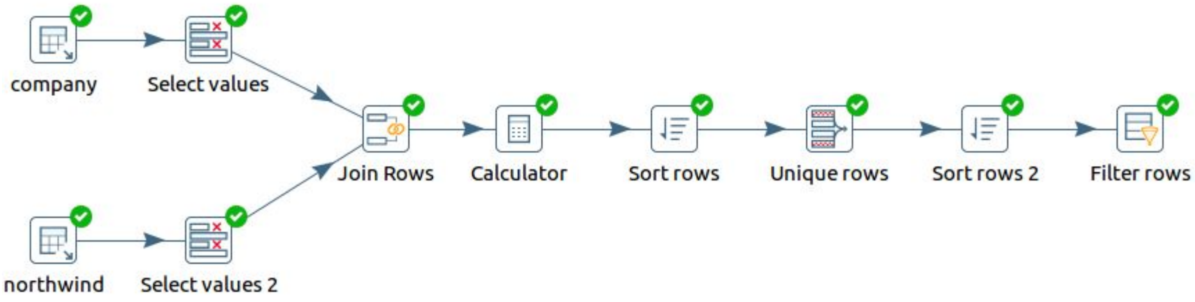
```
create or replace view
All_Employees(EmployeeID,FirstName,LastName,BirthDate,ReportsTo,Title,Salary) as
       (select a.EmployeeID, a.FirstName, a.LastName, a.BirthDate, a.ReportsTo,
a.Title, a.Salary from northwind.Employees as a)

union all

       (select b.employeeid, b.firstname, b.lastname, b.dob, b.reportto,
b.jobtitle, b.salary from company.employees as b)
order by EmployeeID;
```

To create the second view, Works_In(EmployeeID,BranchID,City) to retrieve the list where all the employees work from both databases:

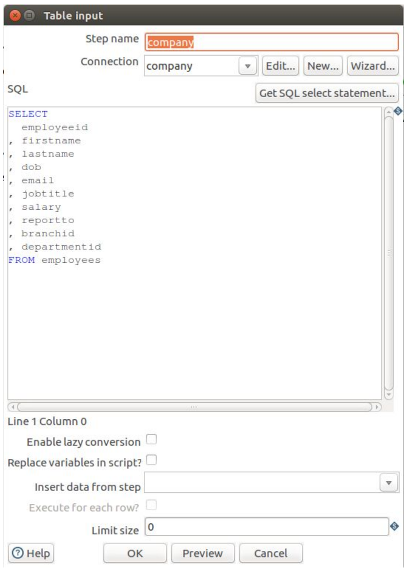
```
create or replace view Works_In(EmployeeID,BranchID,City) as
       (select a.EmployeeID, a.TerritoryID, b.TerritoryDescription from
northwind.EmployeeTerritories as a, northwind.Territories as b where a.TerritoryID
= b.TerritoryID)

union all

       (select d.employeeid, c.branchid, c.city from company.branches as c,
company.employees as d where c.branchid = d.branchid)
order by EmployeeID
```

4. Transformation in both databases

Here we present the transformation created to detect approximate duplicates between the job titles in both databases:



To start this transformation we created two input tables, one for the company database and the other one to the northwind database:



Step name: company

Connection: company

SQL

```
SELECT
  employeeid
, firstname
, lastname
, dob
, email
, jobtitle
, salary
, reportto
, branchid
, departmentid
FROM employees
```

Line 1 Column 0

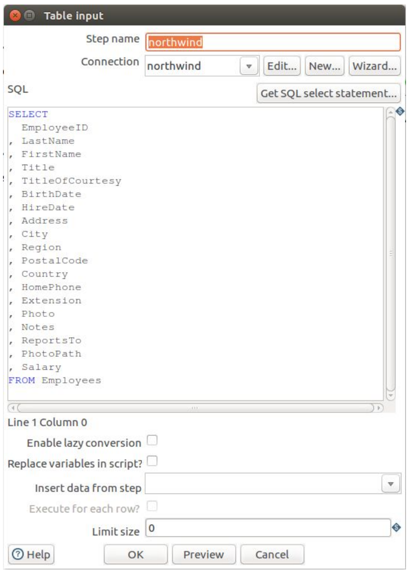
Enable lazy conversion □

Replace variables in script? □

Insert data from step

Execute for each row? □

Limit size: 0



Step name: northwind

Connection: northwind

SQL

```
SELECT
  EmployeeID
, LastName
, FirstName
, Title
, TitleOfCourtesy
, BirthDate
, HireDate
, Address
, City
, Region
, PostalCode
, Country
, HomePhone
, Extension
, Photo
, Notes
, ReportsTo
, PhotoPath
, Salary
FROM Employees
```

Line 1 Column 0

Enable lazy conversion □

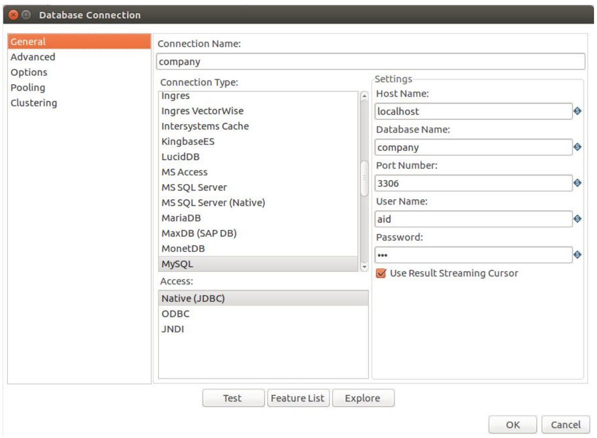
Replace variables in script? □

Insert data from step

Execute for each row? □

Limit size: 0

By clicking on “Edit…” we configured the database connection of each database:



Connection Name: company

Connection Type: MySQL

Access: Native (JDBC)

Settings

Host Name: localhost

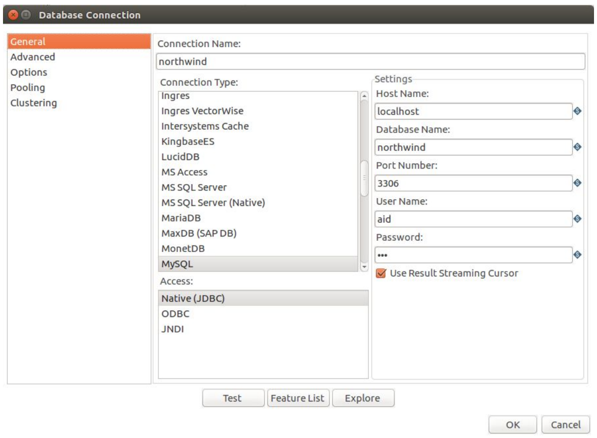
Database Name: company

Port Number: 3306

User Name: aid

Password: •••

☑ Use Result Streaming Cursor



Connection Name: northwind

Connection Type: MySQL

Access: Native (JDBC)

Settings

Host Name: localhost

Database Name: northwind

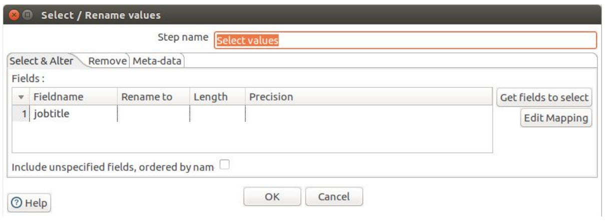
Port Number: 3306

User Name: aid

Password: •••

☑ Use Result Streaming Cursor

We selected the values “jobtitle” and ”title” of each table:

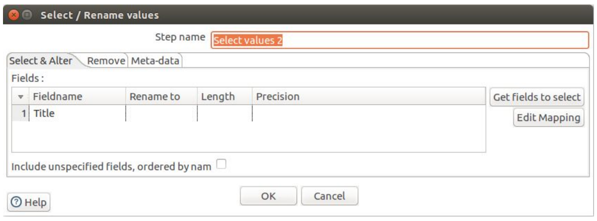


Step name: Select values

Fields :

	Fieldname	Rename to	Length	Precision
1	jobtitle			

Include unspecified fields, ordered by nam □



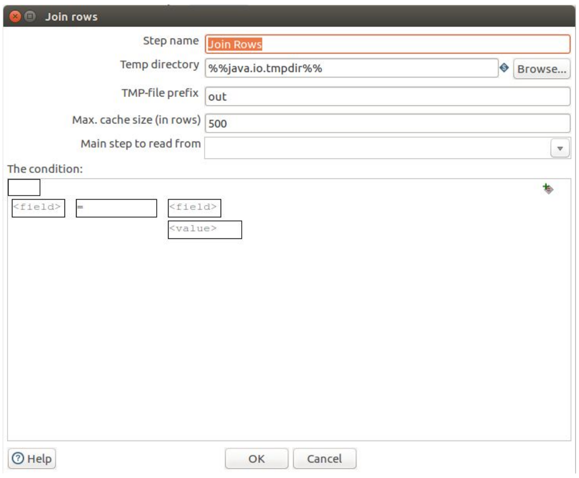
Step name: Select values 2

Fields :

	Fieldname	Rename to	Length	Precision
1	Title			

Include unspecified fields, ordered by nam □

Than, we joined the rows selected:



Step name: Join Rows

Temp directory: %%java.io.tmpdir%%

TMP-file prefix: out

Max. cache size (in rows): 500

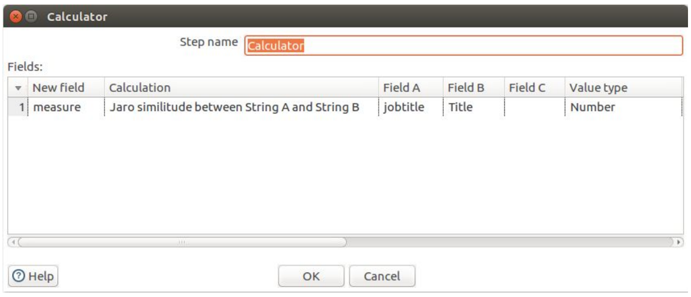
Main step to read from:

The condition:

<field> = <field>

<value>

Afterwards, we created a new field to measure the similarity between the “jobtitle” and the “Title”. We used the Jaro similarity measure, after some testing with other measures:

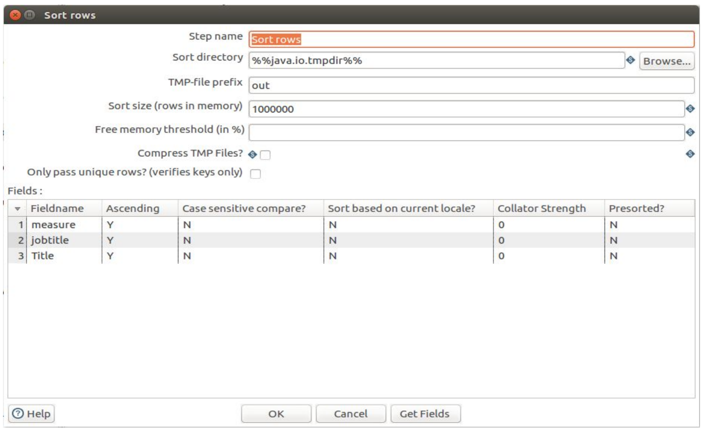


Step name: Calculator

Fields:

	New field	Calculation	Field A	Field B	Field C	Value type
1	measure	Jaro similitude between String A and String B	jobtitle	Title		Number

We sorted rows on ascending by “measure”, than by “jobtitle” and finally by “Title” :



Step name: Sort rows

Sort directory: %%java.io.tmpdir%%

TMP-file prefix: out

Sort size (rows in memory): 1000000

Free memory threshold (in %):

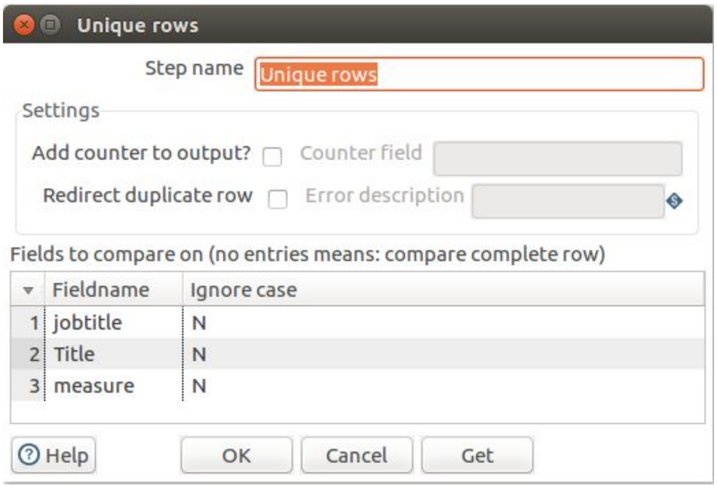
Compress TMP Files? □

Only pass unique rows? (verifies keys only) □

Fields :

	Fieldname	Ascending	Case sensitive compare?	Sort based on current locale?	Collator Strength	Presorted?
1	measure	Y	N	N	0	N
2	jobtitle	Y	N	N	0	N
3	Title	Y	N	N	0	N

With Unique rows, we removed duplicate values:



Step name: Unique rows

Settings

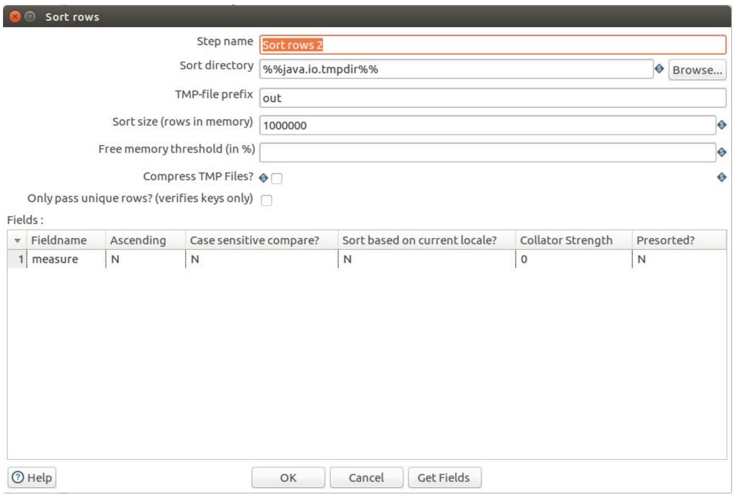
Add counter to output? □ Counter field

Redirect duplicate row □ Error description

Fields to compare on (no entries means: compare complete row)

	Fieldname	Ignore case
1	jobtitle	N
2	Title	N
3	measure	N

We sorted the “measure” row on descending to be easier to identify which are the potential duplicates with the highest measure:



Step name: Sort rows 2

Sort directory: %%java.io.tmpdir%%

TMP-file prefix: out

Sort size (rows in memory): 1000000

Free memory threshold (in %):

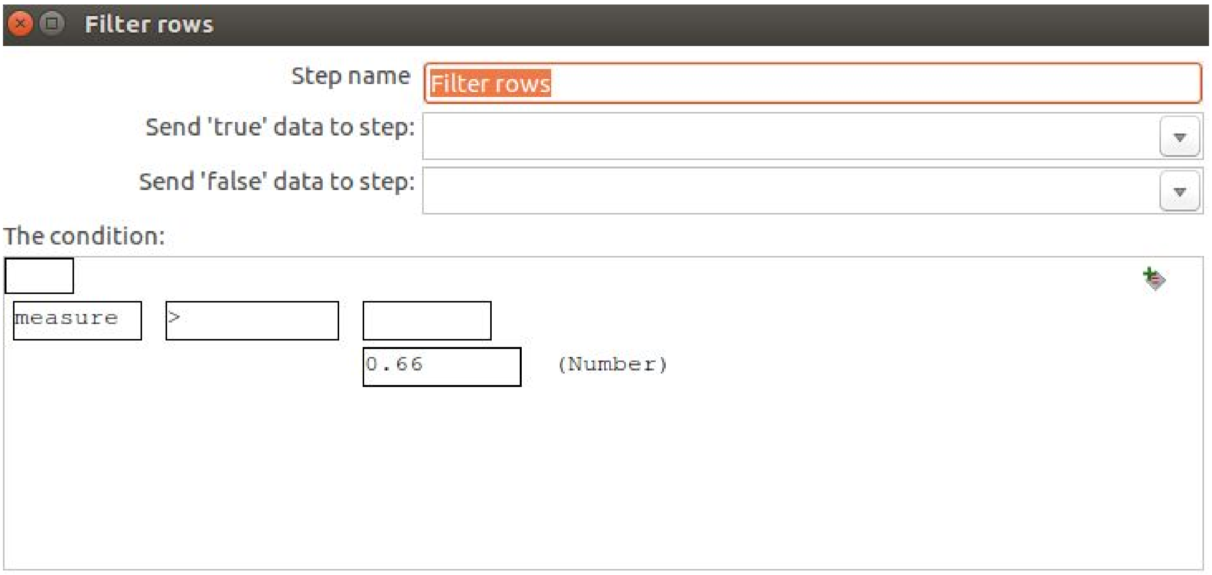
Compress TMP Files? □

Only pass unique rows? (verifies keys only) □

Fields :

	Fieldname	Ascending	Case sensitive compare?	Sort based on current locale?	Collator Strength	Presorted?
1	measure	N	N	N	0	N

Finally we apply a acceptable threshold with Filter Rows:



Step name: Filter rows

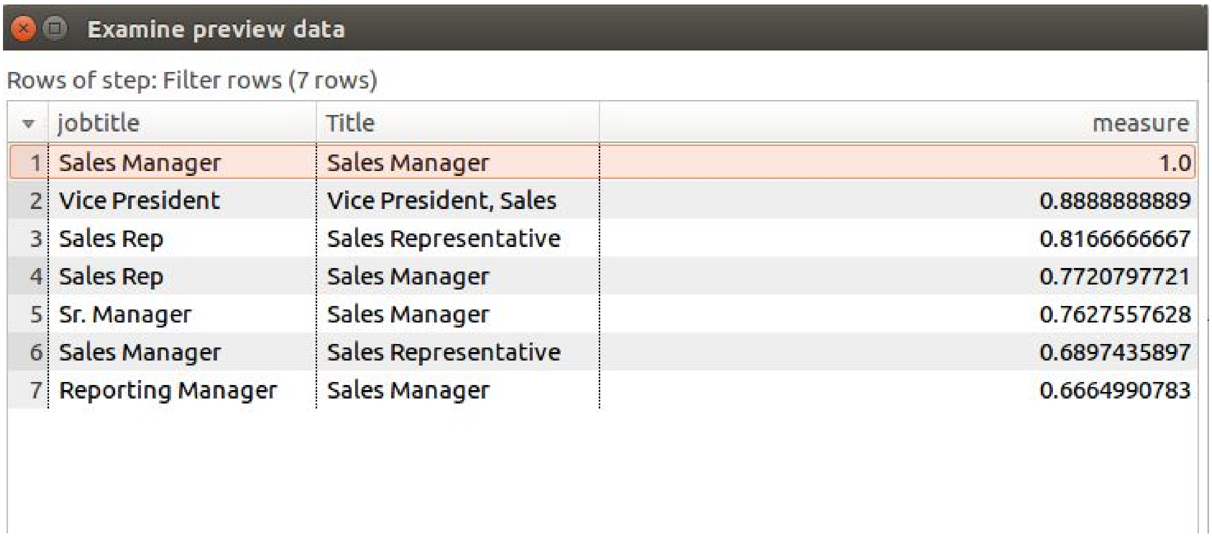
Send 'true' data to step:

Send 'false' data to step:

The condition:

measure > 0.66 (Number)

Finally, we can preview the output of this transformation which is a list of pairs of potential duplicates:



Rows of step: Filter rows (7 rows)

	jobtitle	Title	measure
1	Sales Manager	Sales Manager	1.0
2	Vice President	Vice President, Sales	0.8888888889
3	Sales Rep	Sales Representative	0.8166666667
4	Sales Rep	Sales Manager	0.7720797721
5	Sr. Manager	Sales Manager	0.7627557628
6	Sales Manager	Sales Representative	0.6897435897
7	Reporting Manager	Sales Manager	0.6664990783

5. Creation of the data warehouse tables

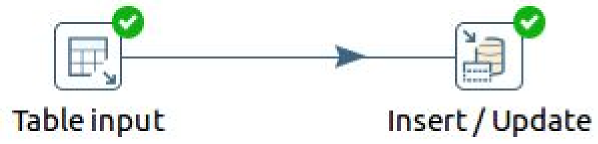
Here we present the SQL instructions needed to create the data warehouse tables:

```
DROP DATABASE IF EXISTS northwind_dw;
CREATE DATABASE northwind_dw;

USE northwind_dw;

CREATE TABLE dim_customer (
       CUSTOMERID VARCHAR(5),
       COMPANYNAME VARCHAR(255),
       CITY VARCHAR(255),
       COUNTRY VARCHAR(255),
       PRIMARY KEY (CUSTOMERID)
);

CREATE TABLE dim_product (
       PRODUCT_CODE INT,
       PRODUCTID INT,
       PRODUCTNAME VARCHAR(255),
       CATEGORYNAME INT,
       DATE_FROM DATETIME,
       DATE_TO DATETIME,
       PRIMARY KEY (PRODUCT_CODE)
);

CREATE TABLE dim_supplier (
       SUPPLIERID INT,
       COMPANYNAME VARCHAR(255),
       CITY VARCHAR(255),
       COUNTRY VARCHAR(255),
       PRIMARY KEY (SUPPLIERID)
);

CREATE TABLE dim_time (
       TIME_ID DATETIME,
       YEAR_ID INT,
       MONTH_ID INT,
       MONTH_NAME VARCHAR(255),
       DAY_ID INT,
       PRIMARY KEY (TIME_ID)
);

CREATE TABLE fact_order (
       ORDERID INT,
       PRODUCT_ID INT,
       QUANTITY INT,
       SALES INT,
       CUSTOMERID VARCHAR(5),
       PRODUCT_CODE INT,
       SUPPLIERID INT,
       TIME_ID DATETIME,
       PRIMARY KEY (ORDERID,PRODUCT_ID),
       FOREIGN KEY (CUSTOMERID) REFERENCES dim_customer (CUSTOMERID),
       FOREIGN KEY (PRODUCT_CODE) REFERENCES dim_product (PRODUCT_CODE),
       FOREIGN KEY (SUPPLIERID) REFERENCES dim_supplier (SUPPLIERID),
       FOREIGN KEY (TIME_ID) REFERENCES dim_time (TIME_ID)
);
```

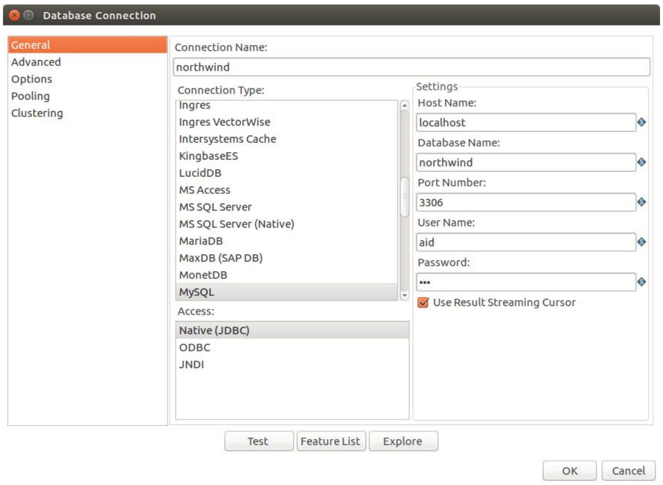
6. Implementation of the ETL process in PDI

Dim_customer:

Transformation to link the table dim_customer and the corresponding tables in northwind_db.



We started by including the table input which we linked to our northwind database. From here, we selected CustomerID, CompanyName, City and Country, from the Customers table.



Connection Name: northwind

Connection Type: MySQL

Access: Native (JDBC)

Settings

Host Name: localhost

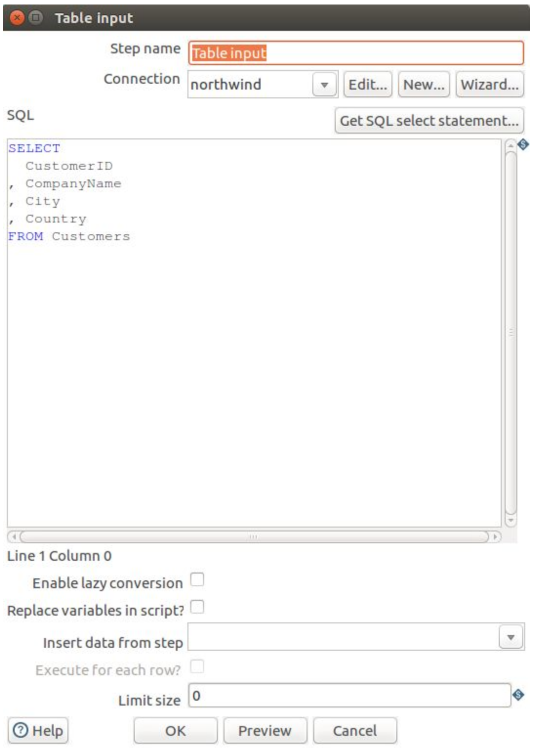
Database Name: northwind

Port Number: 3306

User Name: aid

Password: •••

☑ Use Result Streaming Cursor



Step name: Table input

Connection: northwind

SQL

```
SELECT
  CustomerID
, CompanyName
, City
, Country
FROM Customers
```

Line 1 Column 0

Enable lazy conversion □

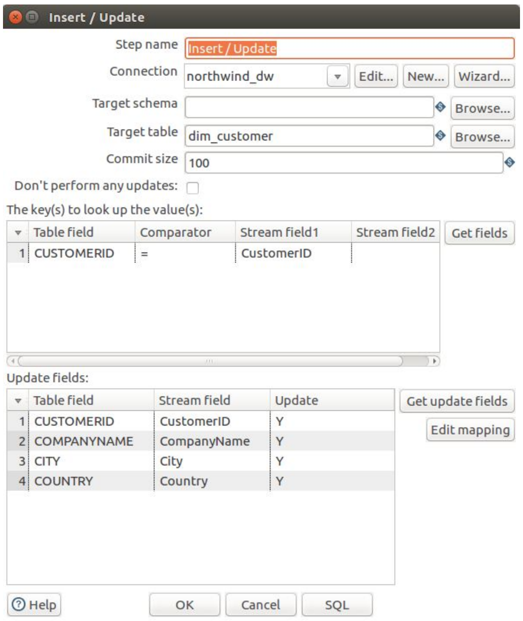
Replace variables in script? □

Insert data from step

Execute for each row? □

Limit size 0

Then, we added Insert/Update in each we linked the fields from our northwind_db to our northwind_dw.



Step name: Insert / Update

Connection: northwind_dw

Target schema:

Target table: dim_customer

Commit size: 100

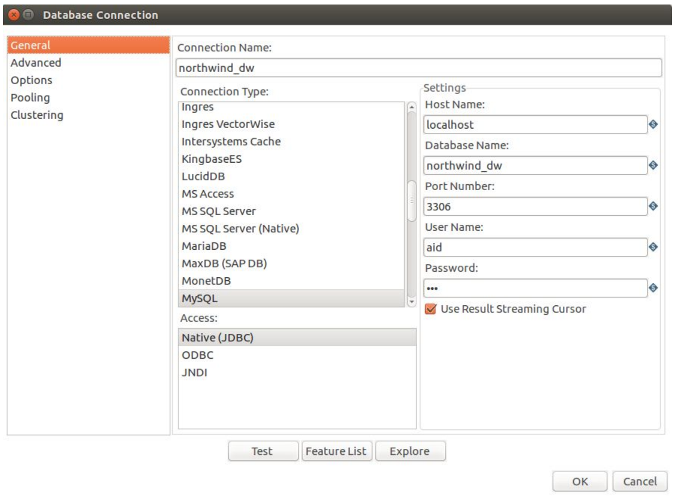
Don't perform any updates: □

The key(s) to look up the value(s):

#	Table field	Comparator	Stream field1	Stream field2
1	CUSTOMERID	=	CustomerID	

Update fields:

#	Table field	Stream field	Update
1	CUSTOMERID	CustomerID	Y
2	COMPANYNAME	CompanyName	Y
3	CITY	City	Y
4	COUNTRY	Country	Y



Connection Name: northwind_dw

Connection Type: MySQL

Access: Native (JDBC)

Settings

Host Name: localhost

Database Name: northwind_dw

Port Number: 3306

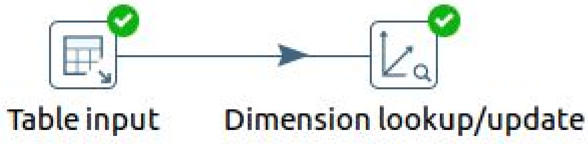
User Name: aid

Password: •••

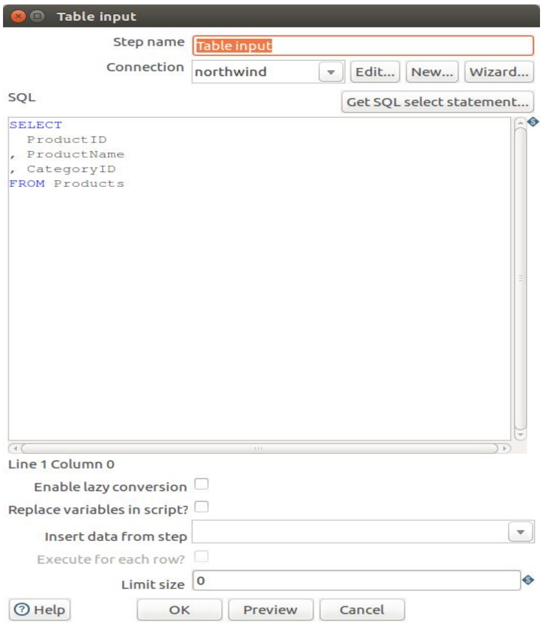
☑ Use Result Streaming Cursor

Dim_product:

Transformation to link the table dim_product and the corresponding tables in northwind_db and adding a slowly-changing dimension.



Firstly, we used table input to get the information from our northwind database and select the fields: ProductID, ProductName and CategoryID from the Products table.



Step name: Table input

Connection: northwind

SQL

```
SELECT
  ProductID
, ProductName
, CategoryID
FROM Products
```

Line 1 Column 0

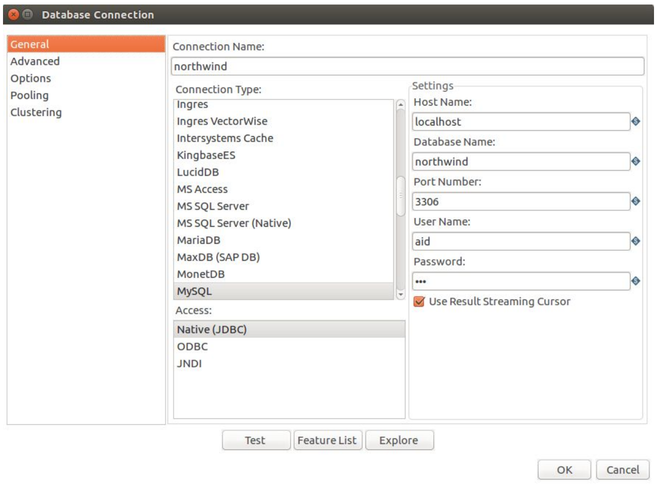
Enable lazy conversion □

Replace variables in script? □

Insert data from step

Execute for each row? □

Limit size 0



Connection Name: northwind

Connection Type: MySQL

Access: Native (JDBC)

Settings

Host Name: localhost

Database Name: northwind

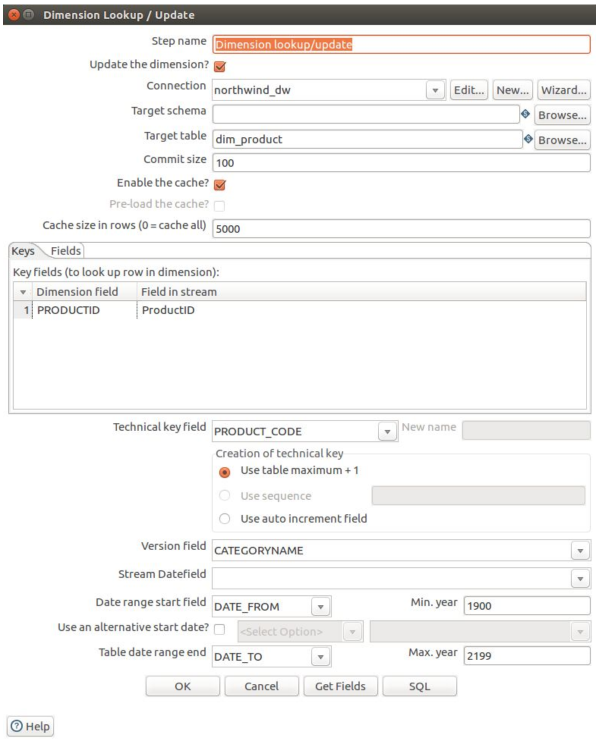
Port Number: 3306

User Name: aid

Password: •••

☑ Use Result Streaming Cursor

Then, we match our table key, Product key with our data warehouse one. Afterwards, we match the other fields, we select our technical field which we created, PRODUCT_CODE, we select our slowly-changing dimension, CATEGORY_NAME and finally, we match our dates and link everything with our northwind_dw.



Step name: Dimension lookup/update
Update the dimension? ☑
Connection: northwind_dw
Target schema:
Target table: dim_product
Commit size: 100
Enable the cache? ☑
Pre-load the cache? □
Cache size in rows (0 = cache all): 5000

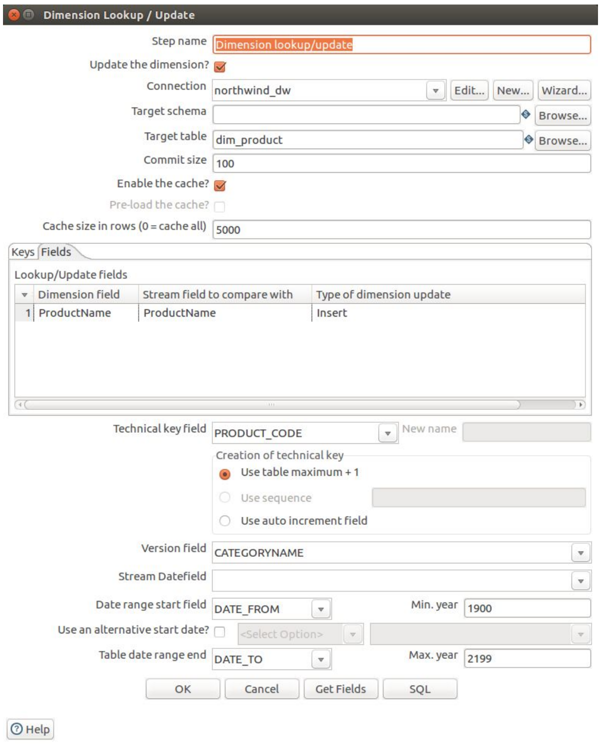
Key fields (to look up row in dimension):

	Dimension field	Field in stream
1	PRODUCTID	ProductID

Technical key field: PRODUCT_CODE New name:

Creation of technical key
● Use table maximum + 1
○ Use sequence
○ Use auto increment field

Version field: CATEGORYNAME
Stream Datefield:
Date range start field: DATE_FROM Min. year: 1900
Use an alternative start date? □ <Select Option>
Table date range end: DATE_TO Max. year: 2199



Step name: Dimension lookup/update
Update the dimension? ☑
Connection: northwind_dw
Target schema:
Target table: dim_product
Commit size: 100
Enable the cache? ☑
Pre-load the cache? □
Cache size in rows (0 = cache all): 5000

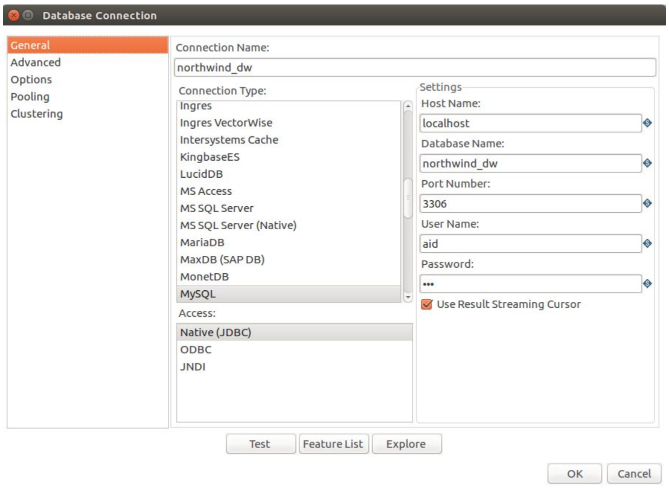
Lookup/Update fields

	Dimension field	Stream field to compare with	Type of dimension update
1	ProductName	ProductName	Insert

Technical key field: PRODUCT_CODE New name:

Creation of technical key
● Use table maximum + 1
○ Use sequence
○ Use auto increment field

Version field: CATEGORYNAME
Stream Datefield:
Date range start field: DATE_FROM Min. year: 1900
Use an alternative start date? □ <Select Option>
Table date range end: DATE_TO Max. year: 2199

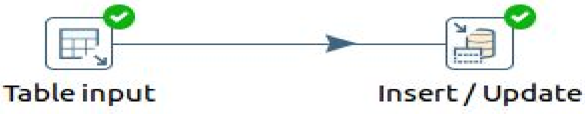


Connection Name: northwind_dw
Connection Type: MySQL
Access: Native (JDBC)

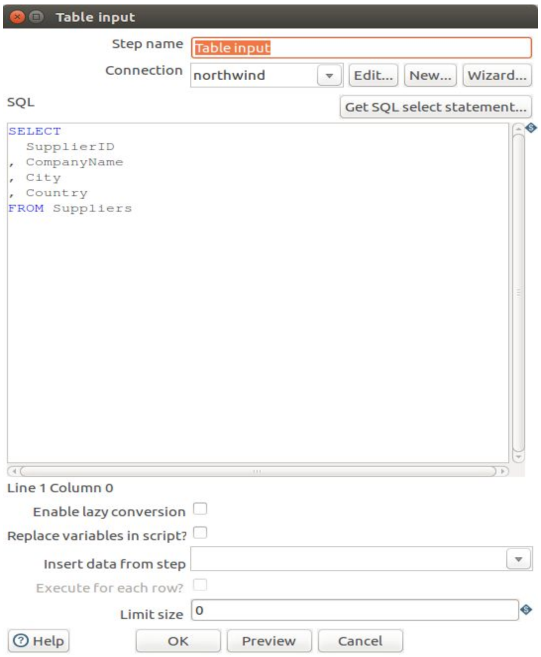
Settings
Host Name: localhost
Database Name: northwind_dw
Port Number: 3306
User Name: aid
Password: •••
☑ Use Result Streaming Cursor

Dim_supplier:

Transformation to link the table dim_supplier and the corresponding tables in northwind_db.



We started by including the table input which we linked to our northwind database. From here we selected SupplierID, CompanyName, City and Country, from the Suppliers table.



Step name: Table input

Connection: northwind

SQL

```
SELECT
  SupplierID
, CompanyName
, City
, Country
FROM Suppliers
```

Line 1 Column 0

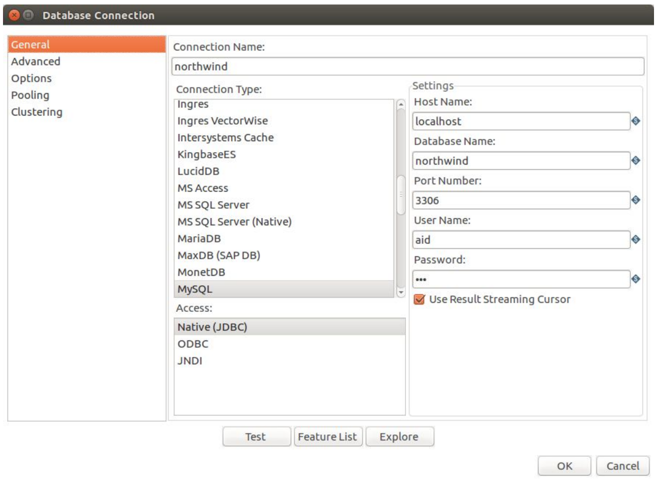
Enable lazy conversion □

Replace variables in script? □

Insert data from step

Execute for each row? □

Limit size 0



Connection Name: northwind

Connection Type: Ingres, Ingres VectorWise, Intersystems Cache, KingbaseES, LucidDB, MS Access, MS SQL Server, MS SQL Server (Native), MariaDB, MaxDB (SAP DB), MonetDB, MySQL

Access: Native (JDBC), ODBC, JNDI

Settings

Host Name: localhost

Database Name: northwind

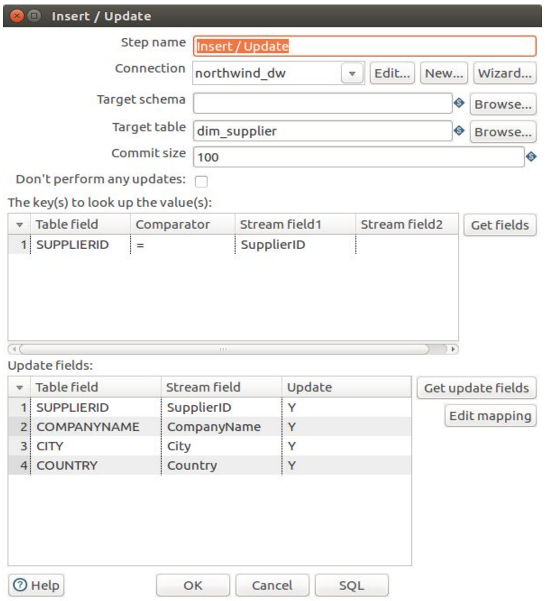
Port Number: 3306

User Name: aid

Password: •••

☑ Use Result Streaming Cursor

Then, we added Insert/Update in each we linked the fields from our northwind_db to our northwind_dw.



Step name: Insert / Update

Connection: northwind_dw

Target schema

Target table: dim_supplier

Commit size: 100

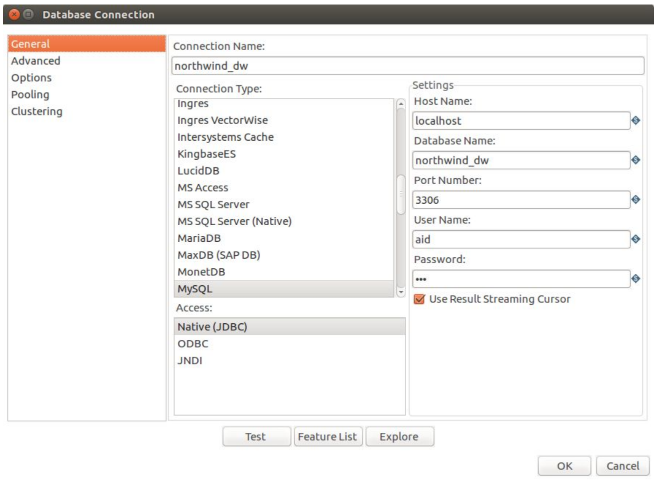
Don't perform any updates: □

The key(s) to look up the value(s):

	Table field	Comparator	Stream field1	Stream field2
1	SUPPLIERID	=	SupplierID	

Update fields:

	Table field	Stream field	Update
1	SUPPLIERID	SupplierID	Y
2	COMPANYNAME	CompanyName	Y
3	CITY	City	Y
4	COUNTRY	Country	Y



Connection Name: northwind_dw

Connection Type: Ingres, Ingres VectorWise, Intersystems Cache, KingbaseES, LucidDB, MS Access, MS SQL Server, MS SQL Server (Native), MariaDB, MaxDB (SAP DB), MonetDB, MySQL

Access: Native (JDBC), ODBC, JNDI

Settings

Host Name: localhost

Database Name: northwind_dw

Port Number: 3306

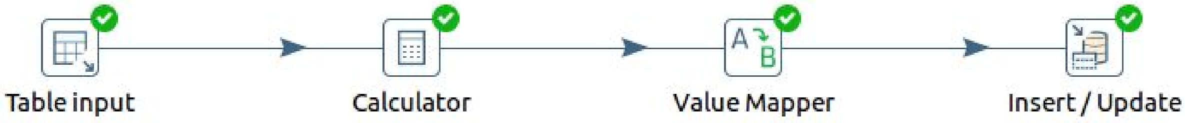
User Name: aid

Password: •••

☑ Use Result Streaming Cursor

Dim_time:

Transformation to link the table dim_time and the corresponding tables in northwind_db.



First, we linked our northwind database with Pentaho, then we selected the field OrderDate from the Orders table.

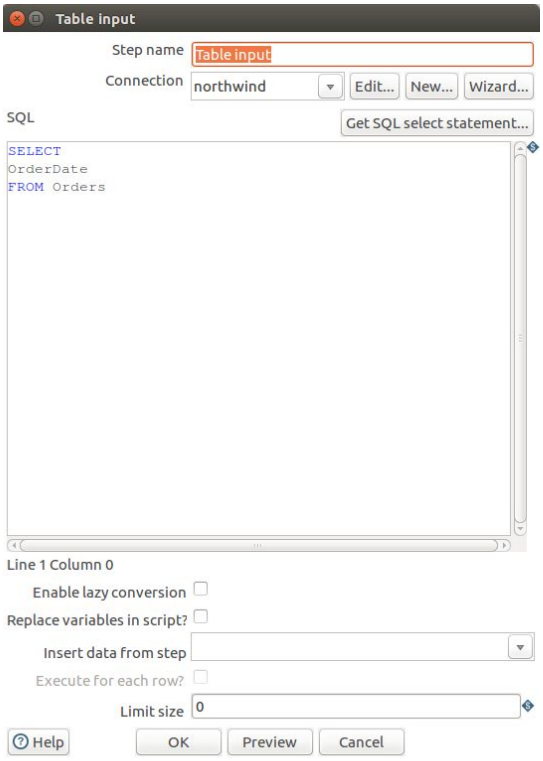


Table input

Step name: Table input

Connection: northwind

SQL

```
SELECT
OrderDate
FROM Orders
```

Line 1 Column 0

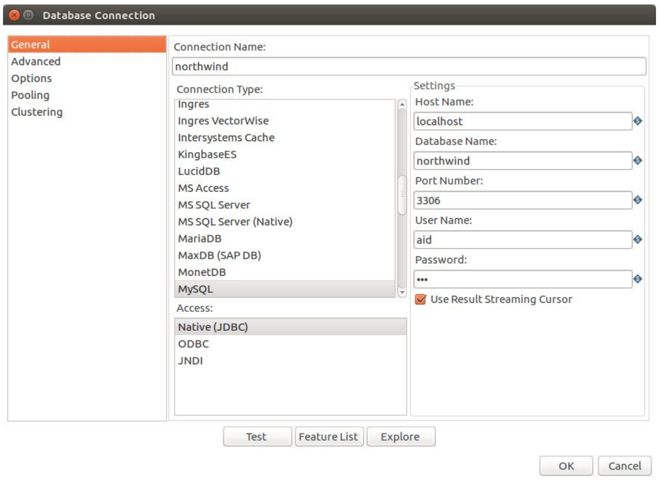
Enable lazy conversion □

Replace variables in script? □

Insert data from step:

Execute for each row? □

Limit size: 0



Database Connection

Connection Name: northwind

Connection Type: MySQL

Access: Native (JDBC)

Settings

Host Name: localhost

Database Name: northwind

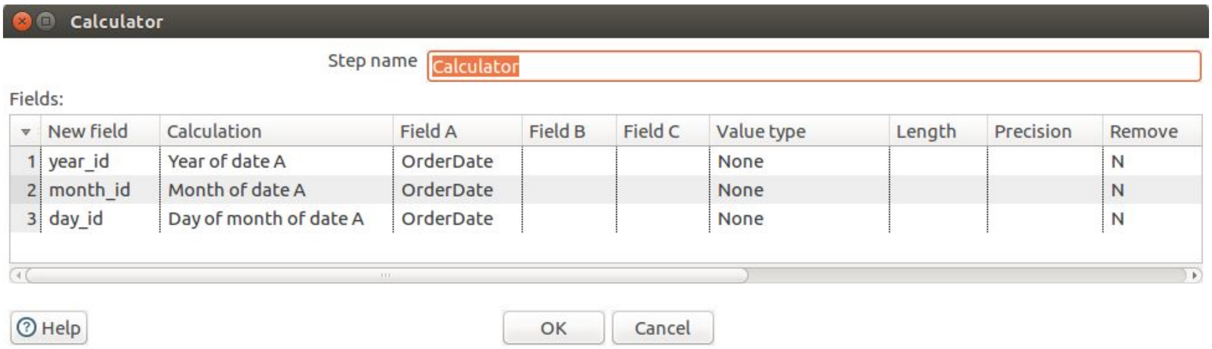
Port Number: 3306

User Name: aid

Password: •••

☑ Use Result Streaming Cursor

Afterwards, we transformed our OrderDate into 3 fields, year, month and day, using the calculator.



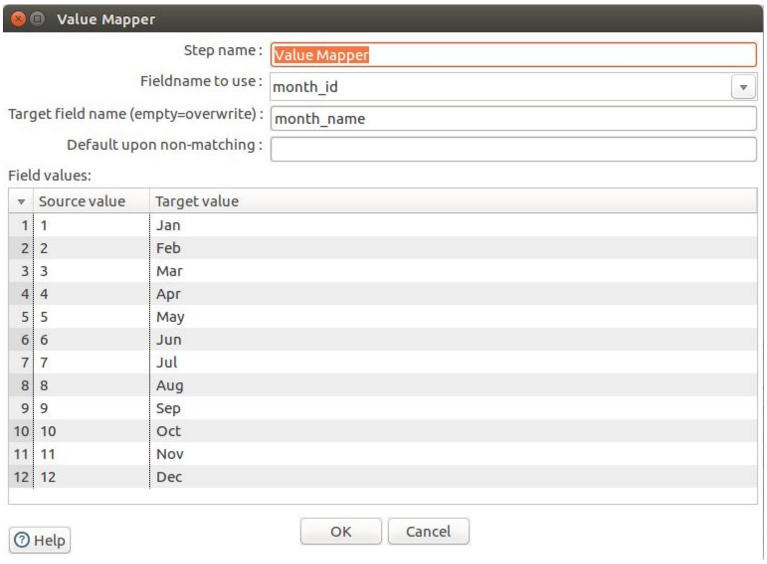
Calculator

Step name: Calculator

Fields:

	New field	Calculation	Field A	Field B	Field C	Value type	Length	Precision	Remove
1	year_id	Year of date A	OrderDate			None			N
2	month_id	Month of date A	OrderDate			None			N
3	day_id	Day of month of date A	OrderDate			None			N

Then, we matched the month number with the respective month name, using value mapper.



Value Mapper

Step name : Value Mapper

Fieldname to use : month_id

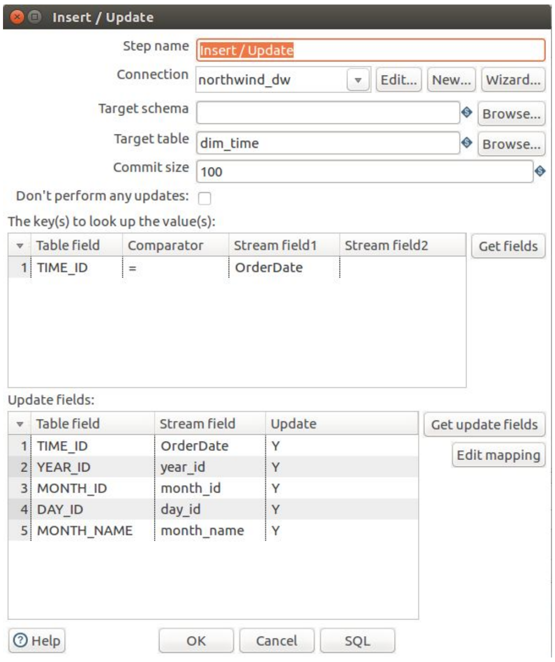
Target field name (empty=overwrite) : month_name

Default upon non-matching :

Field values:

	Source value	Target value
1	1	Jan
2	2	Feb
3	3	Mar
4	4	Apr
5	5	May
6	6	Jun
7	7	Jul
8	8	Aug
9	9	Sep
10	10	Oct
11	11	Nov
12	12	Dec

Finally, we linked our new fields with the table dim_time in northwidn_dw.



Insert / Update

Step name: Insert / Update

Connection: northwind_dw

Target schema:

Target table: dim_time

Commit size: 100

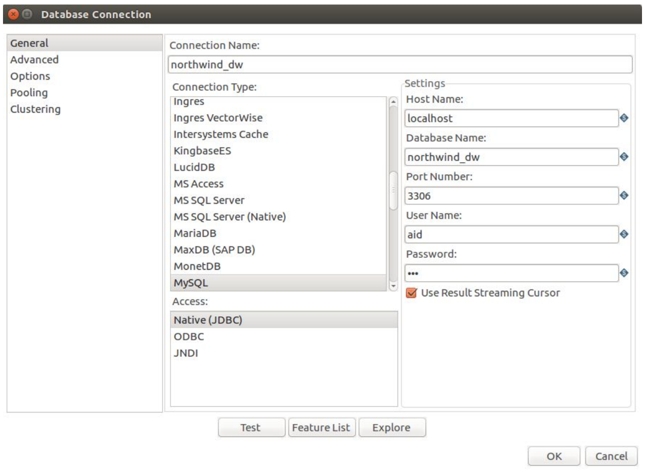
Don't perform any updates: □

The key(s) to look up the value(s):

	Table field	Comparator	Stream field1	Stream field2
1	TIME_ID	=	OrderDate	

Update fields:

	Table field	Stream field	Update
1	TIME_ID	OrderDate	Y
2	YEAR_ID	year_id	Y
3	MONTH_ID	month_id	Y
4	DAY_ID	day_id	Y
5	MONTH_NAME	month_name	Y



Database Connection

Connection Name: northwind_dw

Connection Type: MySQL

Access: Native (JDBC)

Settings

Host Name: localhost

Database Name: northwind_dw

Port Number: 3306

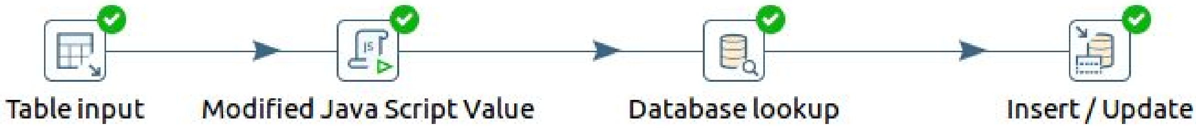
User Name: aid

Password: •••

☑ Use Result Streaming Cursor

fact_order:

Transformation to link the table fact_order and the corresponding tables in northwind_db.



We started by using table input, where we linked our northwind database and extracted all the fields from Orders, OrderDetails and Suppliers.

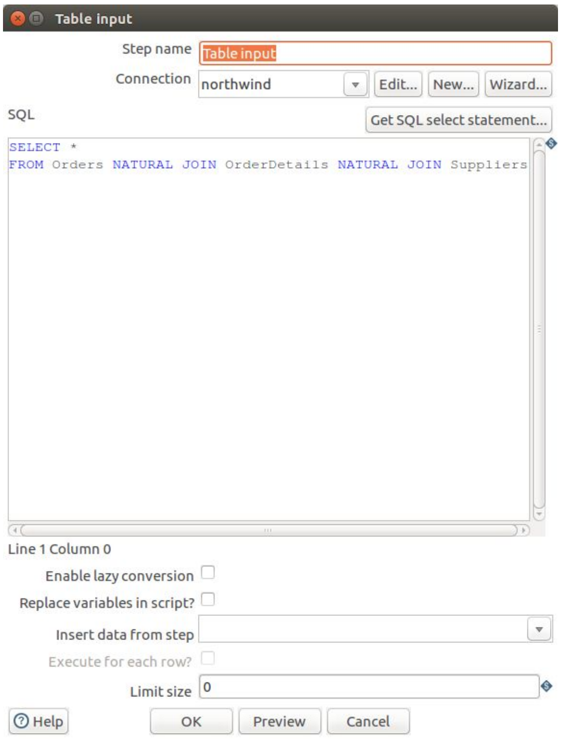


Table input

Step name: Table input

Connection: northwind

SQL

```
SELECT *
FROM Orders NATURAL JOIN OrderDetails NATURAL JOIN Suppliers
```

Line 1 Column 0

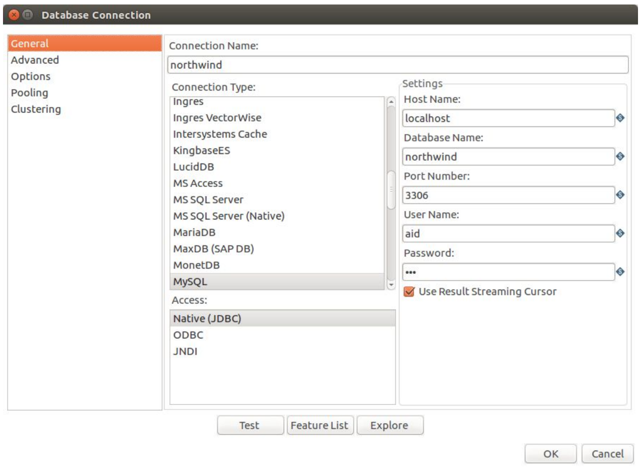
Enable lazy conversion □

Replace variables in script? □

Insert data from step:

Execute for each row? □

Limit size: 0



Database Connection

Connection Name: northwind

Connection Type: Ingres, Ingres VectorWise, Intersystems Cache, KingbaseES, LucidDB, MS Access, MS SQL Server, MS SQL Server (Native), MariaDB, MaxDB (SAP DB), MonetDB, MySQL (selected)

Access: Native (JDBC) (selected), ODBC, JNDI

Settings

Host Name: localhost

Database Name: northwind

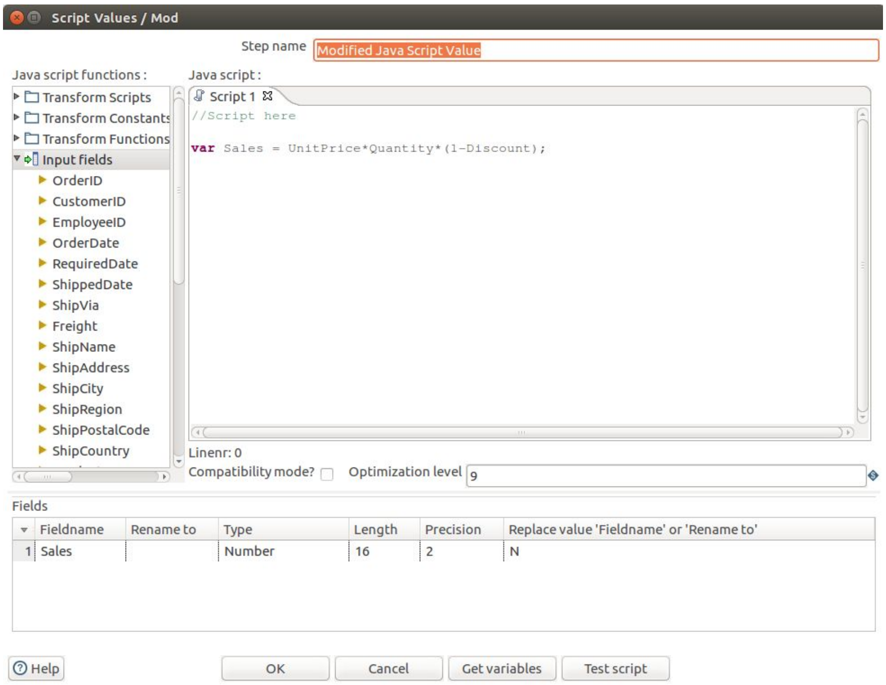
Port Number: 3306

User Name: aid

Password: •••

☑ Use Result Streaming Cursor

Then, we used a Modified Java Script Value to implement the Sales formula, from the UnitPrice, Quantity and Discount, as asked.



Script Values / Mod

Step name: Modified Java Script Value

Java script functions:

Java script:

Script 1

```
//Script here

var Sales = UnitPrice*Quantity*(1-Discount);
```

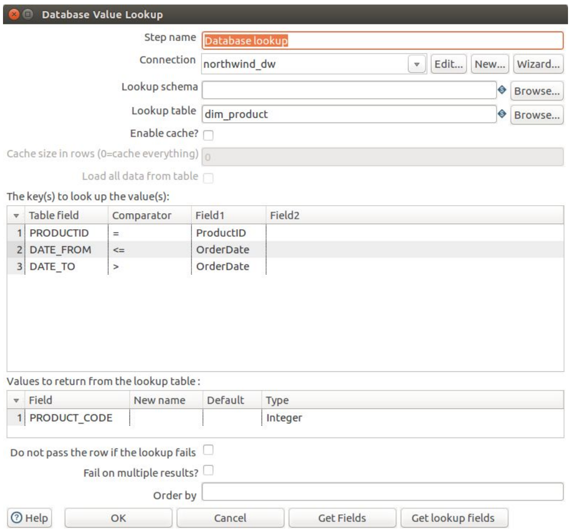
Linenr: 0

Compatibility mode? □ Optimization level: 9

Fields

	Fieldname	Rename to	Type	Length	Precision	Replace value 'Fieldname' or 'Rename to'
1	Sales		Number	16	2	N

Afterwards, we used Database Lookup to link the fields from the dims we had created, the slowly-changing dimension.



Database Value Lookup

Step name	Database lookup
Connection	northwind_dw
Lookup schema	
Lookup table	dim_product
Enable cache?	□
Cache size in rows (0=cache everything)	0
Load all data from table	□

The key(s) to look up the value(s):

	Table field	Comparator	Field1	Field2
1	PRODUCTID	=	ProductID	
2	DATE_FROM	<=	OrderDate	
3	DATE_TO	>	OrderDate	

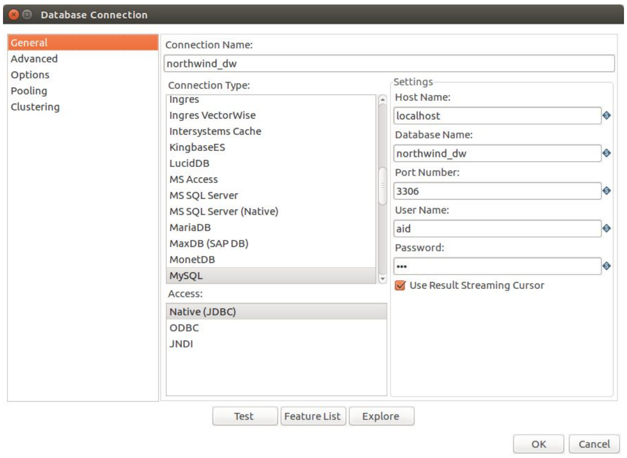
Values to return from the lookup table:

	Field	New name	Default	Type
1	PRODUCT_CODE			Integer

Do not pass the row if the lookup fails □

Fail on multiple results? □

Order by



Database Connection

General
Advanced
Options
Pooling
Clustering

Connection Name: northwind_dw

Connection Type: Ingres, Ingres VectorWise, Intersystems Cache, KingbaseES, LucidDB, MS Access, MS SQL Server, MS SQL Server (Native), MariaDB, MaxDB (SAP DB), MonetDB, MySQL (selected)

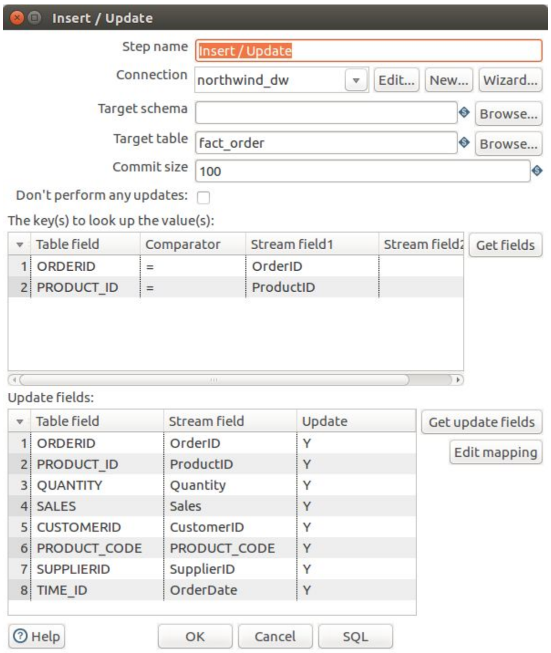
Access: Native (JDBC) (selected), ODBC, JNDI

Settings

Host Name:	localhost
Database Name:	northwind_dw
Port Number:	3306
User Name:	aid
Password:	•••

☑ Use Result Streaming Cursor

To finish, we used insert/update to link all the fields with our northwind data warehouse.



Insert / Update

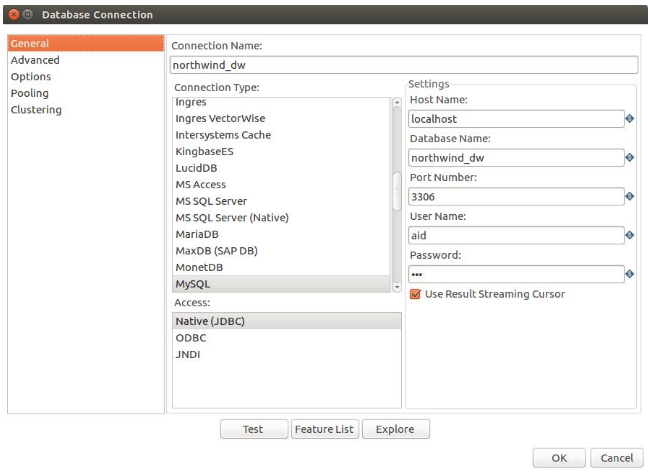
Step name	Insert / Update
Connection	northwind_dw
Target schema	
Target table	fact_order
Commit size	100
Don't perform any updates:	□

The key(s) to look up the value(s):

	Table field	Comparator	Stream field1	Stream field2
1	ORDERID	=	OrderID	
2	PRODUCT_ID	=	ProductID	

Update fields:

	Table field	Stream field	Update
1	ORDERID	OrderID	Y
2	PRODUCT_ID	ProductID	Y
3	QUANTITY	Quantity	Y
4	SALES	Sales	Y
5	CUSTOMERID	CustomerID	Y
6	PRODUCT_CODE	PRODUCT_CODE	Y
7	SUPPLIERID	SupplierID	Y
8	TIME_ID	OrderDate	Y



Database Connection

General
Advanced
Options
Pooling
Clustering

Connection Name: northwind_dw

Connection Type: Ingres, Ingres VectorWise, Intersystems Cache, KingbaseES, LucidDB, MS Access, MS SQL Server, MS SQL Server (Native), MariaDB, MaxDB (SAP DB), MonetDB, MySQL (selected)

Access: Native (JDBC) (selected), ODBC, JNDI

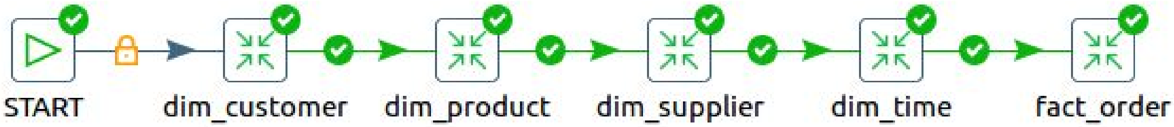
Settings

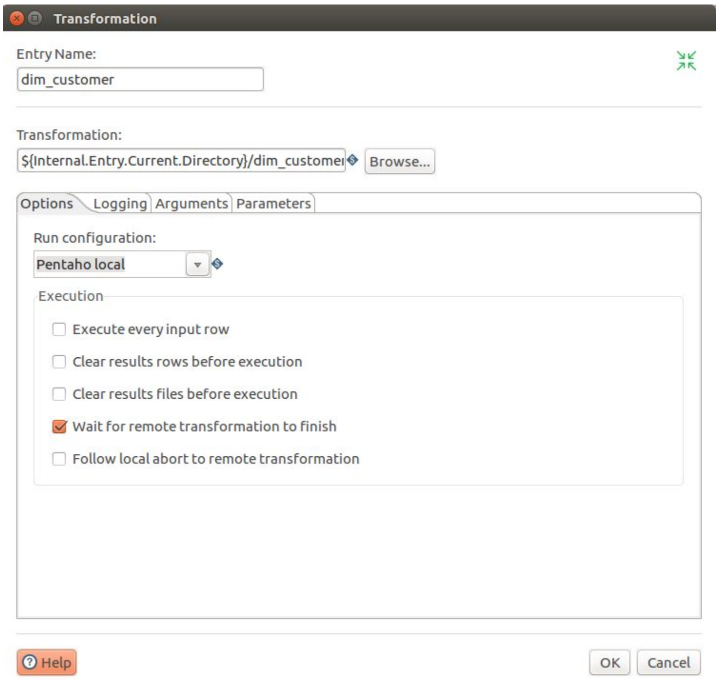
Host Name:	localhost
Database Name:	northwind_dw
Port Number:	3306
User Name:	aid
Password:	•••

☑ Use Result Streaming Cursor

Load Job:

Job to launch all the ETL processes we have just created.





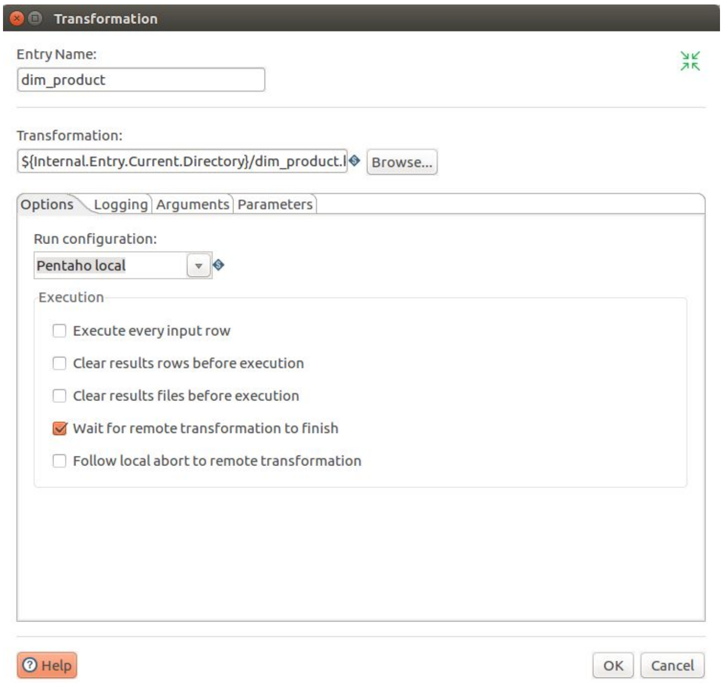
Entry Name: dim_customer

Transformation: ${Internal.Entry.Current.Directory}/dim_customer

Run configuration: Pentaho local

Execution

- ☐ Execute every input row
- ☐ Clear results rows before execution
- ☐ Clear results files before execution
- ☑ Wait for remote transformation to finish
- ☐ Follow local abort to remote transformation



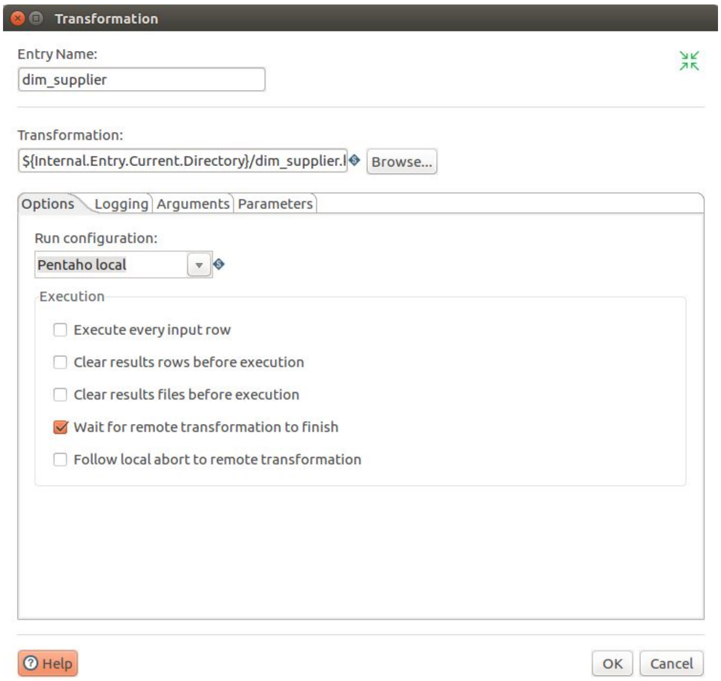
Entry Name: dim_product

Transformation: ${Internal.Entry.Current.Directory}/dim_product.k

Run configuration: Pentaho local

Execution

- ☐ Execute every input row
- ☐ Clear results rows before execution
- ☐ Clear results files before execution
- ☑ Wait for remote transformation to finish
- ☐ Follow local abort to remote transformation



Entry Name: dim_supplier

Transformation: ${Internal.Entry.Current.Directory}/dim_supplier.k

Run configuration: Pentaho local

Execution

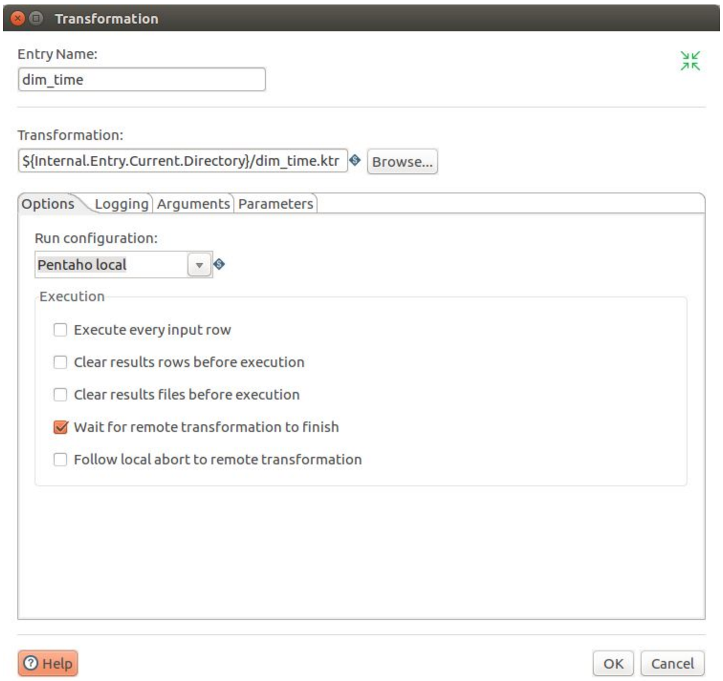
□ Execute every input row

□ Clear results rows before execution

□ Clear results files before execution

☑ Wait for remote transformation to finish

□ Follow local abort to remote transformation



Entry Name: dim_time

Transformation: ${Internal.Entry.Current.Directory}/dim_time.ktr

Run configuration: Pentaho local

Execution

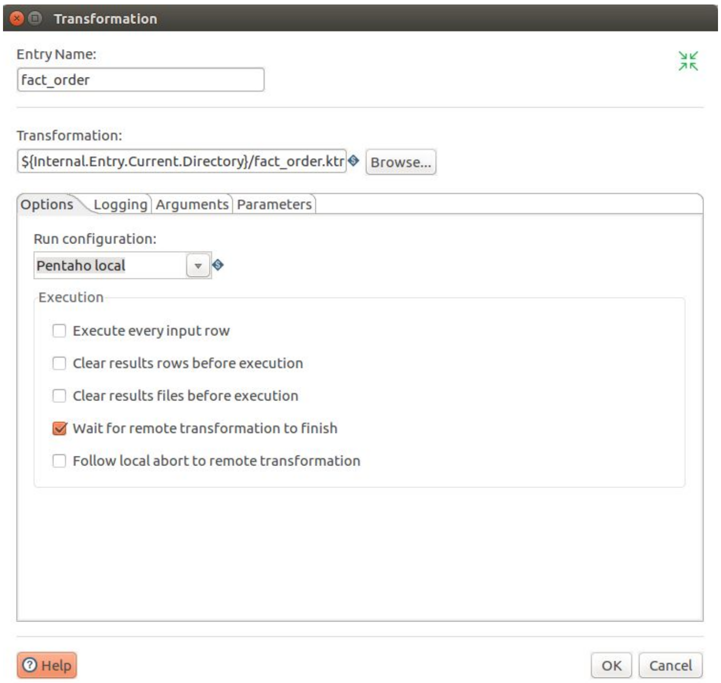
□ Execute every input row

□ Clear results rows before execution

□ Clear results files before execution

☑ Wait for remote transformation to finish

□ Follow local abort to remote transformation



Entry Name: fact_order

Transformation: ${Internal.Entry.Current.Directory}/fact_order.ktr

Run configuration: Pentaho local

Execution

- □ Execute every input row
- □ Clear results rows before execution
- □ Clear results files before execution
- ☑ Wait for remote transformation to finish
- □ Follow local abort to remote transformation

7. XML Cube Definition

Here we present the XML code of the Cube Definition created with PSW:

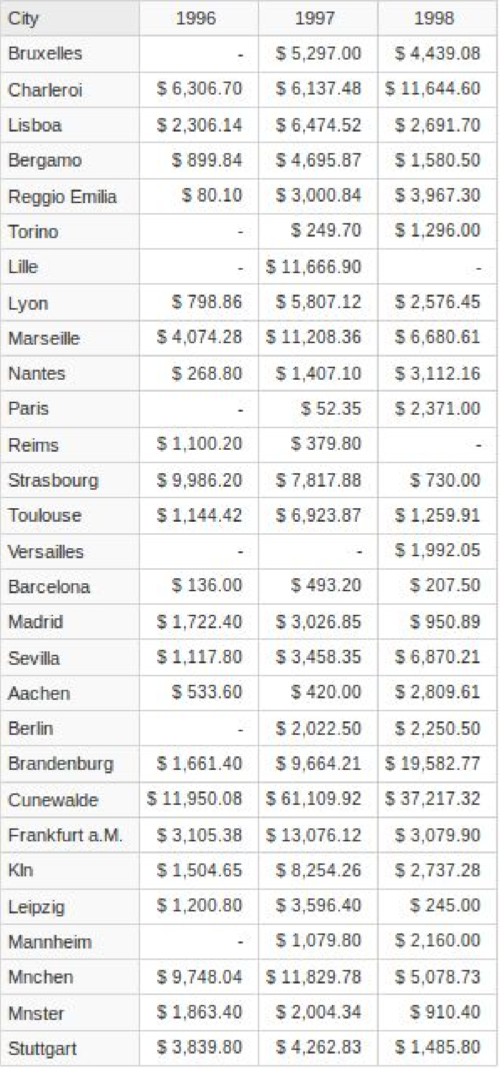
```
<Schema name="northwind_dw">
  <Cube name="Orders" visible="true" cache="true" enabled="true">
    <Table name="fact_order">
    </Table>
    <Dimension type="StandardDimension" visible="true" foreignKey="CUSTOMERID"
name="Customer">
      <Hierarchy name="Customer Hierarchy" visible="true" hasAll="true"
allMemberName="All Customers" primaryKey="CUSTOMERID">
        <Table name="dim_customer">
        </Table>
        <Level name="Country" visible="true" column="COUNTRY" type="String"
uniqueMembers="false" levelType="Regular">
        </Level>
        <Level name="City" visible="true" column="CITY" type="String"
uniqueMembers="false" levelType="Regular">
        </Level>
        <Level name="Company Name" visible="true" column="COMPANYNAME"
type="String" uniqueMembers="false" levelType="Regular">
        </Level>
      </Hierarchy>
    </Dimension>
    <Dimension type="StandardDimension" visible="true" foreignKey="PRODUCT_CODE"
name="Product">
      <Hierarchy name="Product Hierarchy" visible="true" hasAll="true"
allMemberName="All Products" primaryKey="PRODUCT_CODE">
        <Table name="dim_product">
        </Table>
        <Level name="Product ID" visible="true" column="PRODUCTID" type="Integer"
uniqueMembers="false" levelType="Regular">
        </Level>
        <Level name="Product Name" visible="true" column="PRODUCTNAME"
type="String" uniqueMembers="false" levelType="Regular">
        </Level>
        <Level name="Category Name" visible="true" column="CATEGORYNAME"
type="Integer" uniqueMembers="false" levelType="Regular">
        </Level>
      </Hierarchy>
    </Dimension>
    <Dimension type="TimeDimension" visible="true" foreignKey="TIME_ID"
name="Time">
      <Hierarchy name="Time Hierarchy" visible="true" hasAll="true"
allMemberName="All Years" primaryKey="TIME_ID">
        <Table name="dim_time">
        </Table>
        <Level name="Year" visible="true" column="YEAR_ID" type="Integer"
uniqueMembers="false" levelType="TimeYears">
        </Level>
        <Level name="Month ID" visible="true" column="MONTH_ID" type="Integer"
uniqueMembers="false" levelType="TimeMonths">
        </Level>
        <Level name="Month Name" visible="true" column="MONTH_NAME" type="String"
uniqueMembers="false" levelType="TimeMonths">
        </Level>
        <Level name="Day" visible="true" column="DAY_ID" type="Integer"
uniqueMembers="false" levelType="TimeDays">
        </Level>
      </Hierarchy>
    </Dimension>
    <Dimension type="StandardDimension" visible="true" foreignKey="SUPPLIERID"
name="Supplier">
      <Hierarchy name="Supplier Hierarchy" visible="true" hasAll="true"
allMemberName="All Suppliers" primaryKey="SUPPLIERID">
        <Table name="dim_supplier">
        </Table>
        <Level name="Country" visible="true" column="COUNTRY" type="String"
uniqueMembers="false" levelType="Regular">
        </Level>
        <Level name="City" visible="true" column="CITY" type="String"
uniqueMembers="false" levelType="Regular">
        </Level>
        <Level name="Company Name" visible="true" column="COMPANYNAME"
type="String" uniqueMembers="false" levelType="Regular">
        </Level>
      </Hierarchy>
    </Dimension>
    <Measure name="Sales" column="SALES" datatype="Numeric" formatString="$
#,###.00" aggregator="sum" visible="true">
    </Measure>
    <Measure name="Quantity" column="QUANTITY" datatype="Integer"
formatString="#,###" aggregator="sum" visible="true">
    </Measure>
  </Cube>
</Schema>
```

8. MDX Query

To demonstrate some of the capabilities of the MDX language we decided to create a query that show the sales over the years in each column of the cities of Belgium, Portugal, Italy, France, Spain and Germany in the rows. Here is displayed the MDX query:

```
SELECT Time.Year.Members ON COLUMNS,
GENERATE({Customer.Country.Belgium, Customer.Country.Portugal,
Customer.Country.Italy, Customer.Country.France, Customer.Country.Spain,
Customer.Country.Germany},
DESCENDANTS(Customer.CurrentMember, Customer.City)) ON ROWS
From Orders
Where Measure.Sales
```

The results of the MDX query in Saiku are the following:



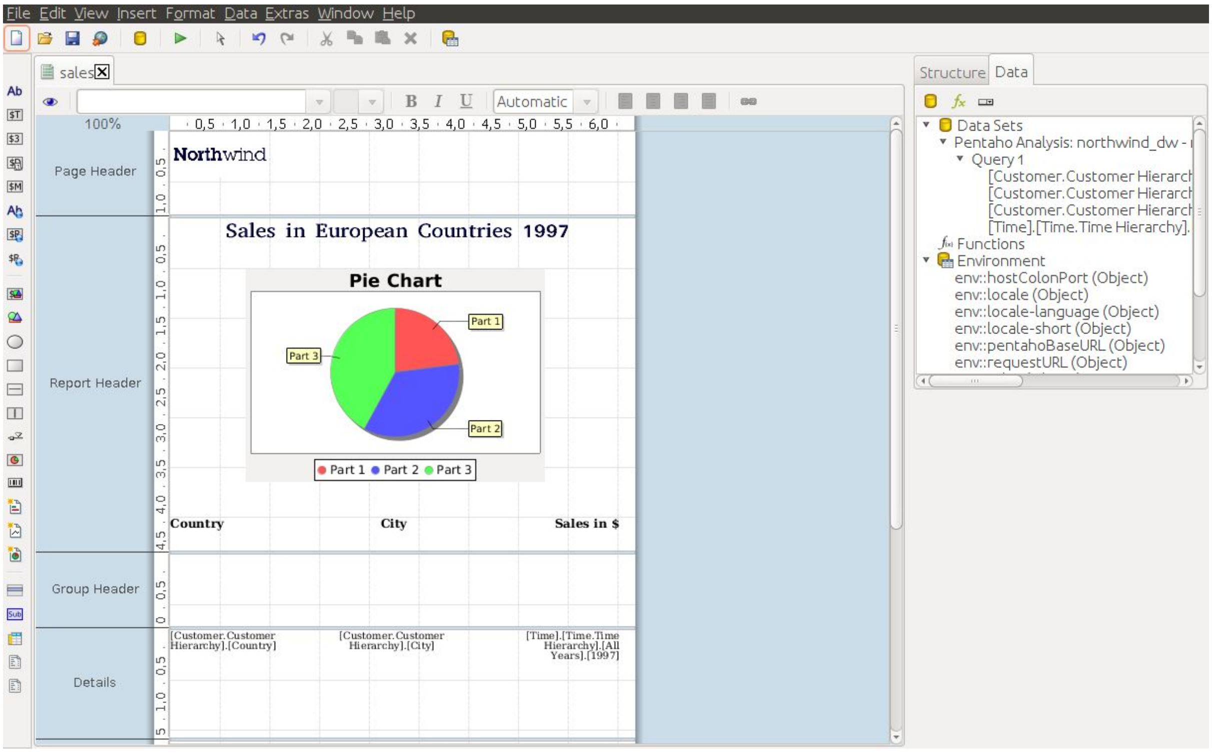
City	1996	1997	1998
Bruxelles	-	$ 5,297.00	$ 4,439.08
Charleroi	$ 6,306.70	$ 6,137.48	$ 11,644.60
Lisboa	$ 2,306.14	$ 6,474.52	$ 2,691.70
Bergamo	$ 899.84	$ 4,695.87	$ 1,580.50
Reggio Emilia	$ 80.10	$ 3,000.84	$ 3,967.30
Torino	-	$ 249.70	$ 1,296.00
Lille	-	$ 11,666.90	-
Lyon	$ 798.86	$ 5,807.12	$ 2,576.45
Marseille	$ 4,074.28	$ 11,208.36	$ 6,680.61
Nantes	$ 268.80	$ 1,407.10	$ 3,112.16
Paris	-	$ 52.35	$ 2,371.00
Reims	$ 1,100.20	$ 379.80	-
Strasbourg	$ 9,986.20	$ 7,817.88	$ 730.00
Toulouse	$ 1,144.42	$ 6,923.87	$ 1,259.91
Versailles	-	-	$ 1,992.05
Barcelona	$ 136.00	$ 493.20	$ 207.50
Madrid	$ 1,722.40	$ 3,026.85	$ 950.89
Sevilla	$ 1,117.80	$ 3,458.35	$ 6,870.21
Aachen	$ 533.60	$ 420.00	$ 2,809.61
Berlin	-	$ 2,022.50	$ 2,250.50
Brandenburg	$ 1,661.40	$ 9,664.21	$ 19,582.77
Cunewalde	$ 11,950.08	$ 61,109.92	$ 37,217.32
Frankfurt a.M.	$ 3,105.38	$ 13,076.12	$ 3,079.90
Kln	$ 1,504.65	$ 8,254.26	$ 2,737.28
Leipzig	$ 1,200.80	$ 3,596.40	$ 245.00
Mannheim	-	$ 1,079.80	$ 2,160.00
Mnchen	$ 9,748.04	$ 11,829.78	$ 5,078.73
Mnster	$ 1,863.40	$ 2,004.34	$ 910.40
Stuttgart	$ 3,839.80	$ 4,262.83	$ 1,485.80

9. Report created with PRD

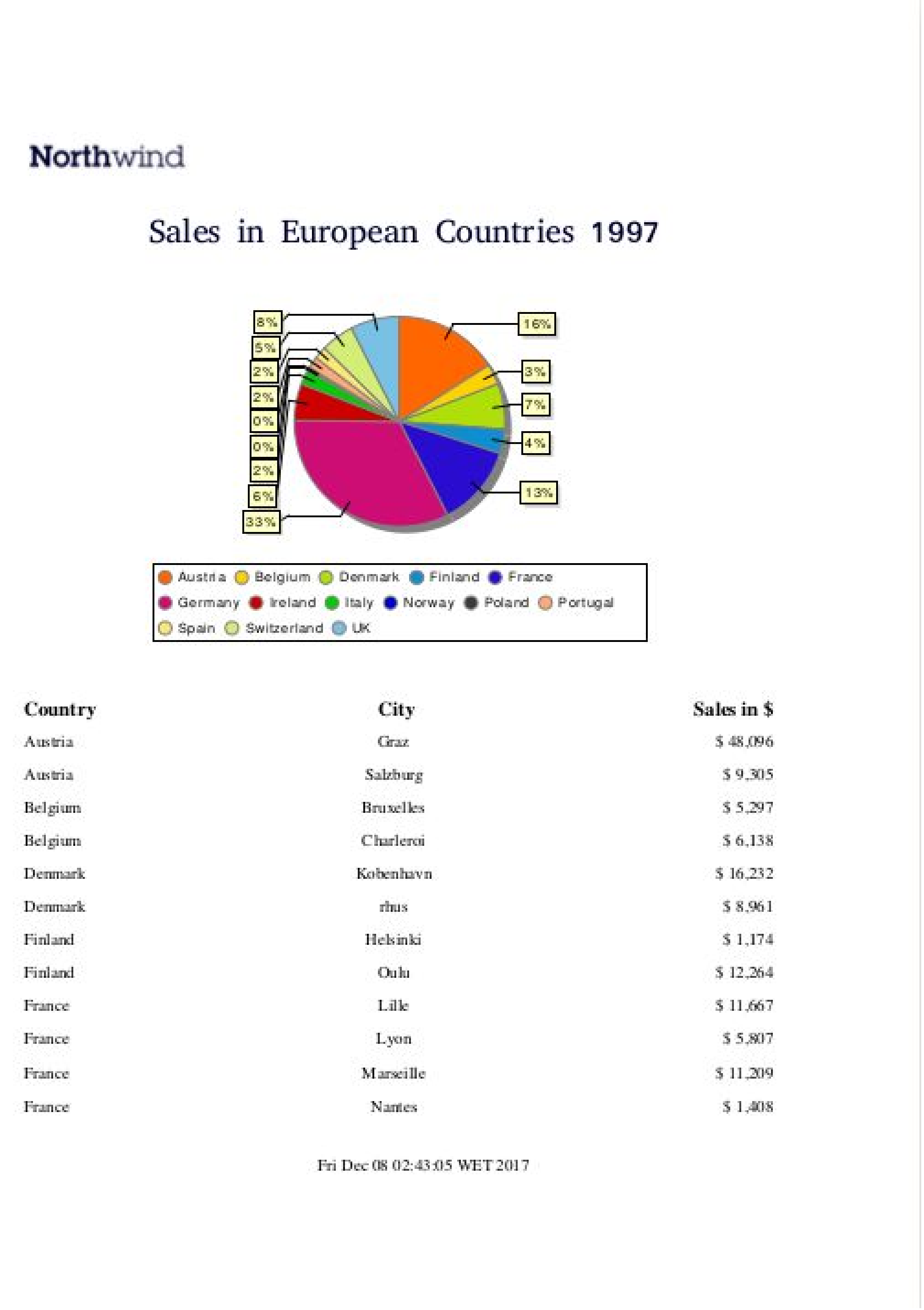
To elaborate a custom report we used Pentaho Report Designer. We decided to create a query that show the sales in European Countries in 1997. Here is displayed the MDX query:

```
SELECT Time.[1997] ON COLUMNS,
GENERATE({Customer.Country.Austria,Customer.Country.Belgium,Customer.Country.Denmar
k,Customer.Country.Finland,Customer.Country.France,Customer.Country.Germany,Custome
r.Country.Ireland,Customer.Country.Italy,Customer.Country.Norway,Customer.Country.P
oland,Customer.Country.Portugal,Customer.Country.Spain,Customer.Country.Sweden,Cust
omer.Country.Switzerland,Customer.Country.UK},
DESCENDANTS(Customer.CurrentMember,
Customer.City)) ON ROWS
FROM Orders
WHERE Measures.Sales
```

Design Mode in PRD:


File Edit View Insert Format Data Extras Window Help
sales
Automatic
100%
Structure
Data
Page Header
Report Header
Group Header
Details
Northwind
Sales in European Countries 1997
Pie Chart
Part 1
Part 2
Part 3
Country
City
Sales in $
[Customer.Customer Hierarchy].[Country]
[Customer.Customer Hierarchy].[City]
[Time].[Time.Time Hierarchy].[All Years].[1997]
Data Sets
Pentaho Analysis: northwind_dw -
Query 1
[Customer.Customer Hierarch
[Customer.Customer Hierarch
[Customer.Customer Hierarch
[Time].[Time.Time Hierarchy].
Functions
Environment
env::hostColonPort (Object)
env::locale (Object)
env::locale-language (Object)
env::locale-short (Object)
env::pentahoBaseURL (Object)
env::requestURL (Object)

PreView Mode of the first page of the report in PRD:



Northwind

Sales in European Countries 1997

● Austria ● Belgium ● Denmark ● Finland ● France
● Germany ● Ireland ● Italy ● Norway ● Poland ● Portugal
○ Spain ● Switzerland ● UK

Country	City	Sales in $
Austria	Graz	$ 48,096
Austria	Salzburg	$ 9,305
Belgium	Bruxelles	$ 5,297
Belgium	Charleroi	$ 6,138
Denmark	Kobenhavn	$ 16,232
Denmark	rhus	$ 8,961
Finland	Helsinki	$ 1,174
Finland	Oulu	$ 12,264
France	Lille	$ 11,667
France	Lyon	$ 5,807
France	Marseille	$ 11,209
France	Nantes	$ 1,408

Fri Dec 08 02:43:05 WET 2017