DATA INTEGRATION DESIGN

INSURANCE COMPANY

DECEMBER 3, 2019

Group D, MBD-2

Mohammed Al Yousef

Jad Bechara

Patricia Carcamo Acosta

Ting-Lun Fan

Blanca Fernandez-Cuesta

Lenny Tahon

Contents

1. Data Integration Strategy

1.1 Origin

Data analyzed was obtained from the CoIL 2000 Challenge and contains information on customers of an insurance company and consists of 86 variables of product usage and socio-demographic data derived from zip codes. The dataset contains 5,822 rows meant to represent clients of the insurance company. The purpose of the original dataset in the challenge was to build a prediction model for clients who were more likely to purchase Caravan Insurance. This data used in this process is the file “ticdata2000.txt Training data. (1M)” available at the following link http://kdd.ics.uci.edu/databases/tic/tic.html.

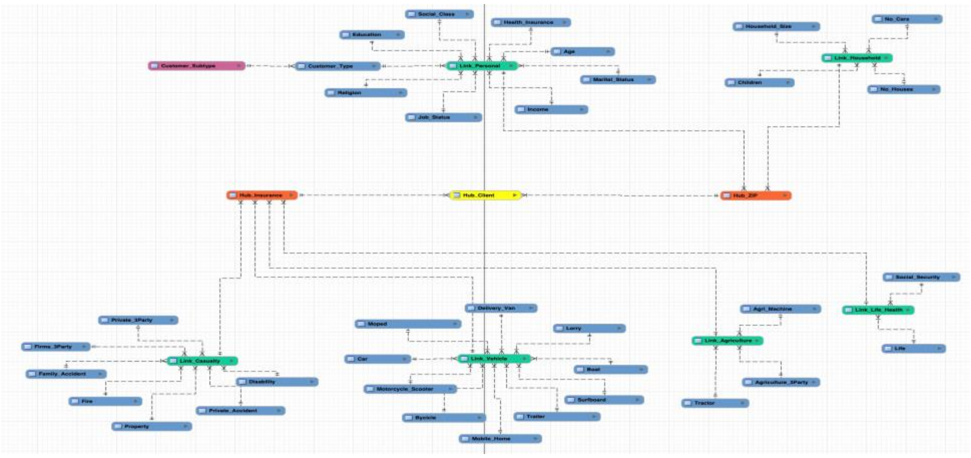
The data warehouse designed intends to provide the insurance company cross-functionally by any of its business lines namely Casualty, Life and Health, Vehicle and Agriculture. The data integration process is capable of transforming the data in its raw state and serving it to the data vault where it can then be used to obtain business insights.

1.2 Transformation

The original data was analyzed against the data definition table to understand the value factors each variable represent. Pentaho Data Integration Version 8.3 was used to transform the data in two steps: first, mapping the values of the variables to their definition; then, serving this data to the tables in the data vault. As in the original data warehouse design, it made sense to group these tasks in the same way our data structure is organized, the socio-demographic tables and the insurance tables. These two steps are organized in two jobs and each job contains multiple transformations which will be outlined in the next section.

1.3 Destination

Once the original data set is transformed (Job1) the goal is to deposit this information into the data vault hosted in MySQL Server. The data vault contains 3 hub tables, 6 link tables, and 35 satellite tables. These tables are structured by relationship in socio-demographic tables, and insurance tables.



1.4 Quality Issues

During the ETL Process, some quality issues were uncovered with regard to consistency of table structure. The updated sql file is included in the deliverables. The following changes were made to the previous data warehouse design in order to successfully complete data integration:

- No_Houses Table, change No_Houses_MAANTHUI type from VARCHAR to INT
- Add USER to Children table
- Update Age Table, Age_Range_MGEMLEEF type from INT to VARCHAR
- Update Moped Table, rename field TIMEDATE to TIMESTAMP
- Update Car Table, change Car_Number_APERSAUT from VARCHAR to INT
- Update Customer_Subtype table, fields marked as UQ (unique) should be NN (not null)

2 Data Mapping

Once the original data set is transformed, we are able to map the values to the tables in our data warehouse design. The approach we applied to data mapping involved consistency and following a sequential order. First, data feeds the satellite tables and the variables which will be used as foreign keys are created. Then, the foreign keys can be moved up to the link and hub tables. We created transformations for each group of tables and its link, and the final transformation includes the data going to the link and hub tables. We divided this task into 7 transformations and then created the job to run them all. In our process, these are TR3 – TR9 and Job2. We are able to verify the accurate mapping of values by running a query in MySQL.

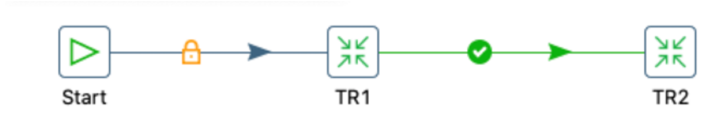
Each data field is mapped to its corresponding table using the Table Output Transformation. In Pentaho, we assigned the target table and specified the database fields we were mapping. This mapping included the value of the fields in that particular table and the metadata. An important step in the mapping process is that each step needs to return an auto-generated key for that particular table. In this way, we create the foreign key to the link tables. The link table is mapped once all the tables that are connected to it have been mapped and their foreign keys are available in the process. The link table also needs to return an auto-generated key in order to feed the hub table.

A crucial factor for us in the data mapping process was the naming of the variables. We were careful to be consistent in the transformation of the raw data to name each of the variables the corresponding name to the variable in the table output. This allowed us to assign the correct stream field to the table field where this value was assigned. This also allowed us to more easily check our data warehouse for issues with variable types because we could see the output of the transformed data to ensure it would be accepted once the output command was made.

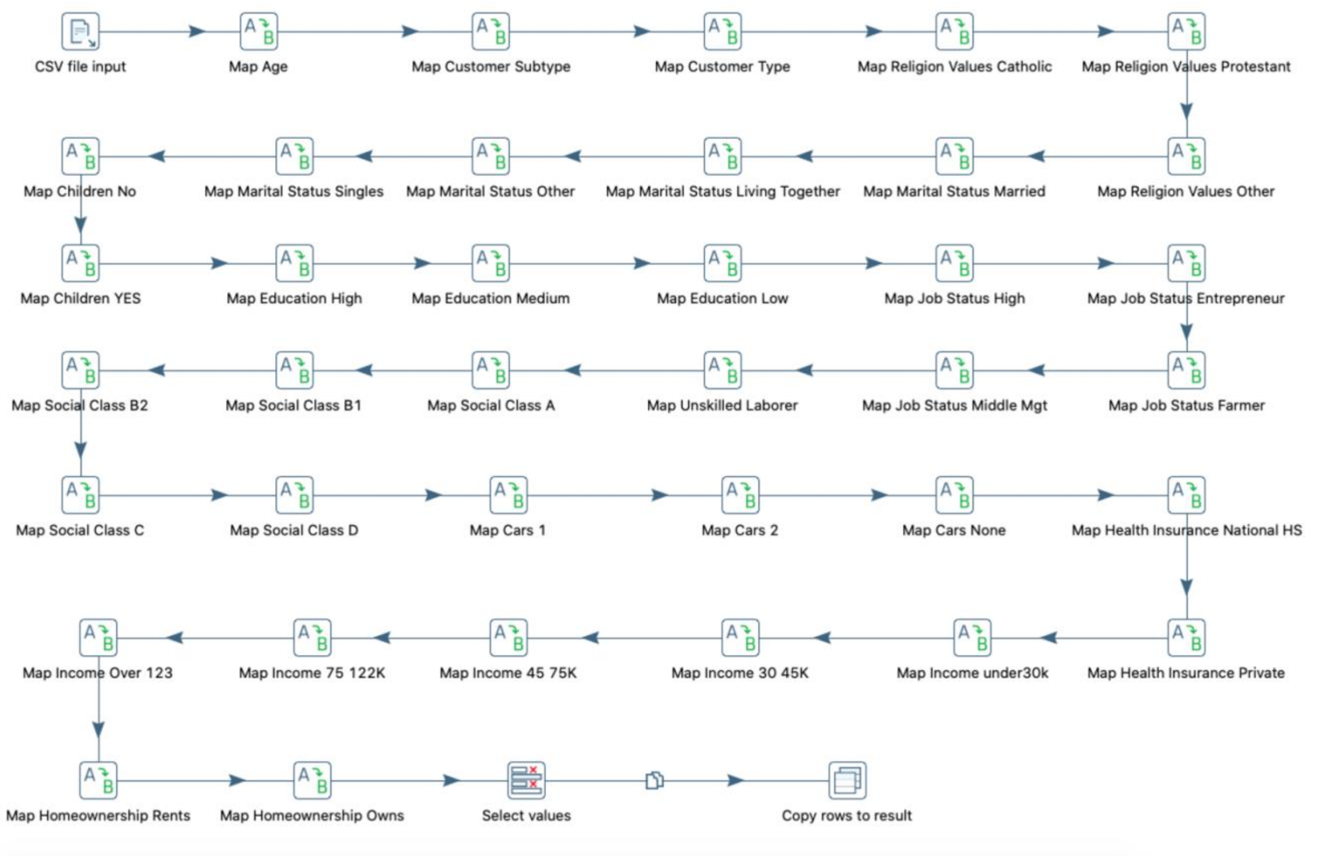
3 ETL Process

3.1 Job 1

Job1 is assigned the task to retrieve the original dataset from the input folder in our directory and transform it to the dataset that will be moved into our data structure. Job 1 contains two transformations, TR1 maps the values of the variables in the socio-demographic tables and TR2 maps the values in the insurance tables.

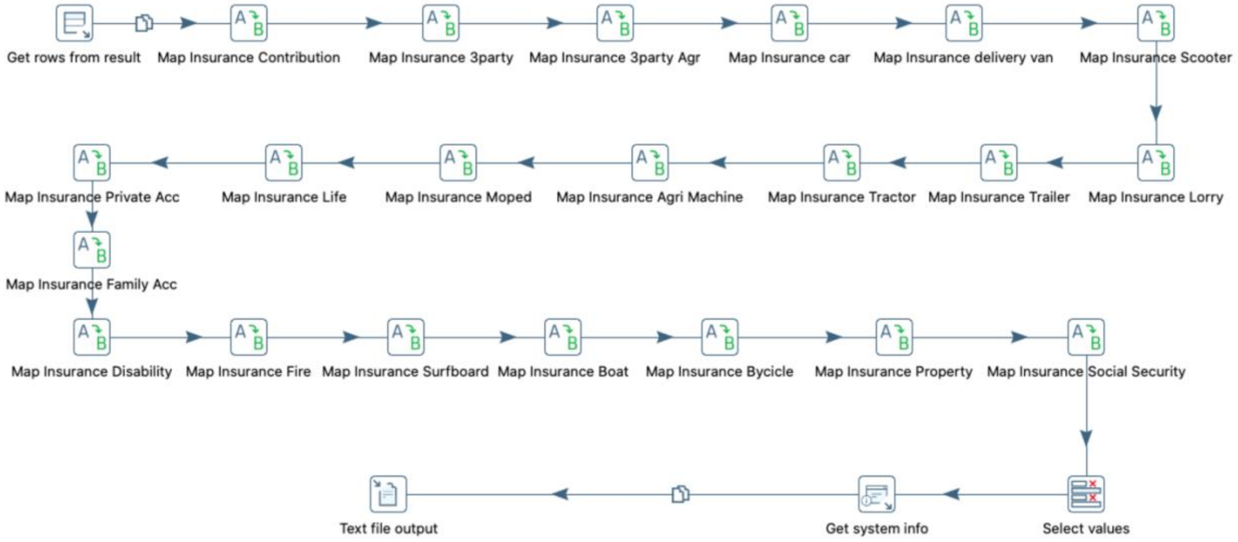


3.2.1 Transformation 1 (TR1)



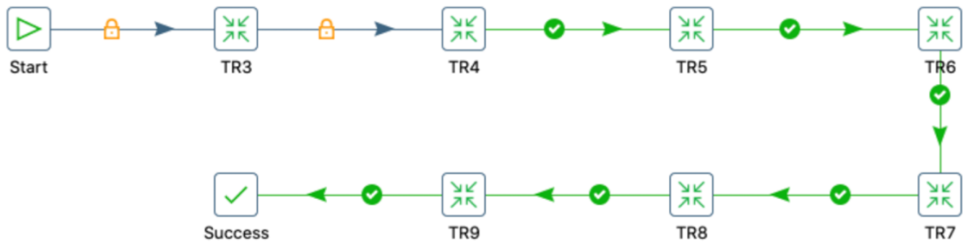
The original variables transformed are integers and our transformation changes these integers into string values according to the definition table. This is only possible by creating a new column in the original dataset with this new fieldname assignment. In this step, we are careful to match the new fieldname to the name that will be needed in the output stage. The select values transformation allows us to then remove the fieldnames that are no longer needed. The result from TR1 is copied and received by TR2.

3.2.2 Transformation 2 (TR2)



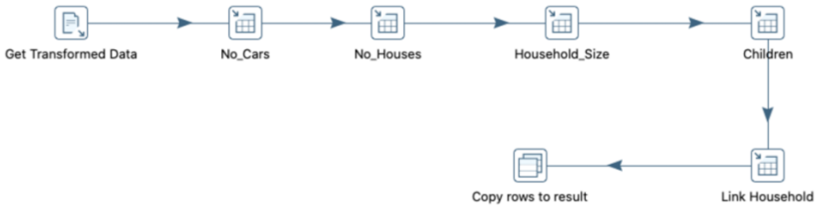
In transformation 2, we get the rows from the previous transformation and the values for Insurance contribution variables are mapped according to the values in the data definition table. Like in TR1, new fieldnames are created with string type to accommodate this transformation. The select values transformation allows us to remove the fieldnames that are no longer needed, and to name the fieldnames for the columns whose numeric value did not need to be changed. The output of this transformation is a csv file named "TransformedData" we save in our current directory's output folder.

3.2 Job 2

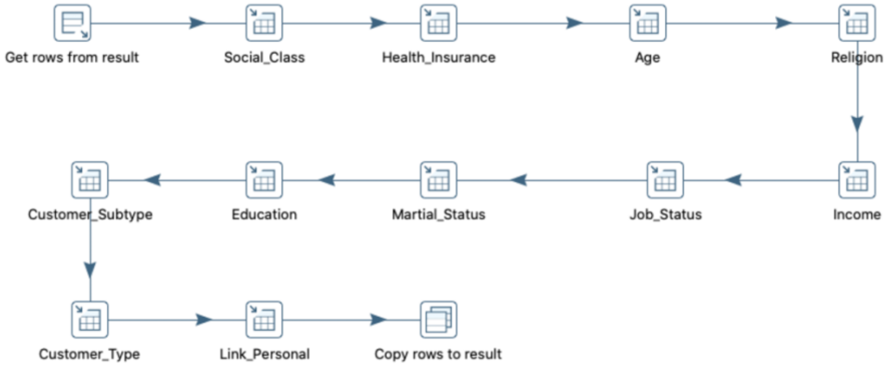


Job 2 executes the transformations that output the data result from Job1 to the assigned tables in our data vault. This job contains 7 transformations TR3 and TR4 output data to the socio demographic tables, and TR5 to TR8 output data to the insurance tables. TR9 outputs data to the link and hub tables. The output of one transformation is passed on to the following transformation. In each table output transformation, the variables are assigned to the corresponding table variables, foreign keys are generated, and metadata is stored.

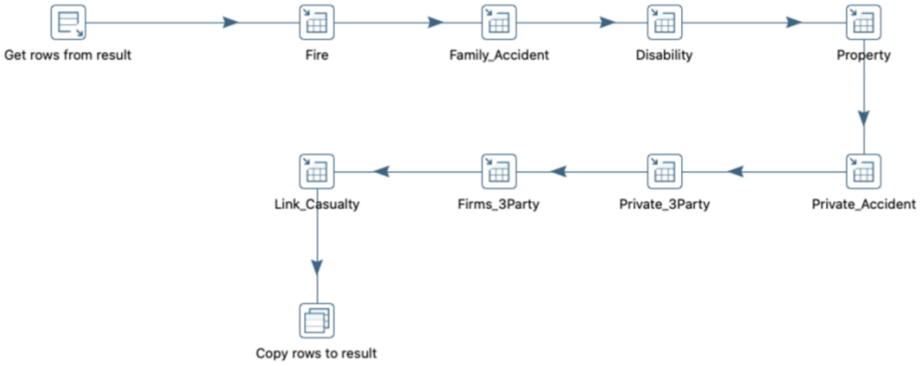
3.2.1 Transformation 3 (TR3)



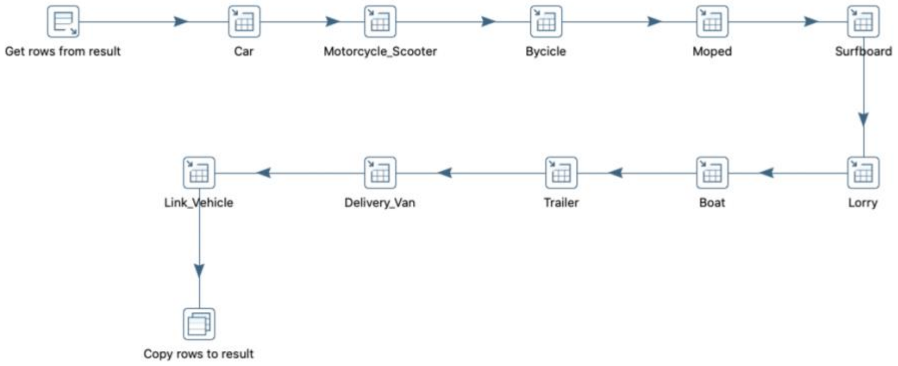
3.2.2 Transformation 4 (TR4) - Personal



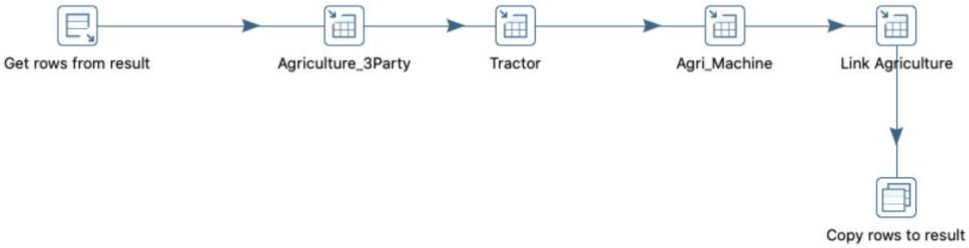
3.2.3 Transformation 5 (TR5) - Casualty



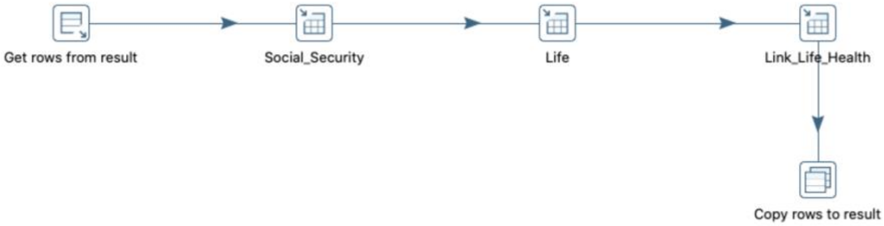
3.2.4 Transformation 6 (TR6) - Vehicle



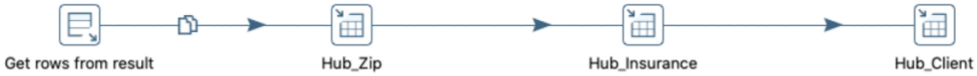
3.2.5 Transformation 7 (TR7) - Agriculture



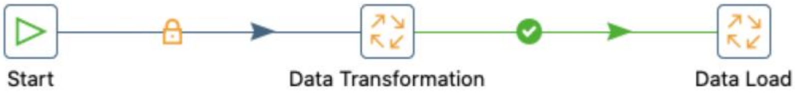
3.2.6 Transformation 8 (TR8) - Life-Health



3.2.7 Transformation 9 (TR9) – Link and Hub Tables



3.3 Job 3



Job 3 executes the entire ETL process from start to finish, transforming raw data and outputting it to the pertaining tables in MySQL Server. Data Transformation is Job1, Data Load is Job2. Running this Job is the only step needed perform the entire process.

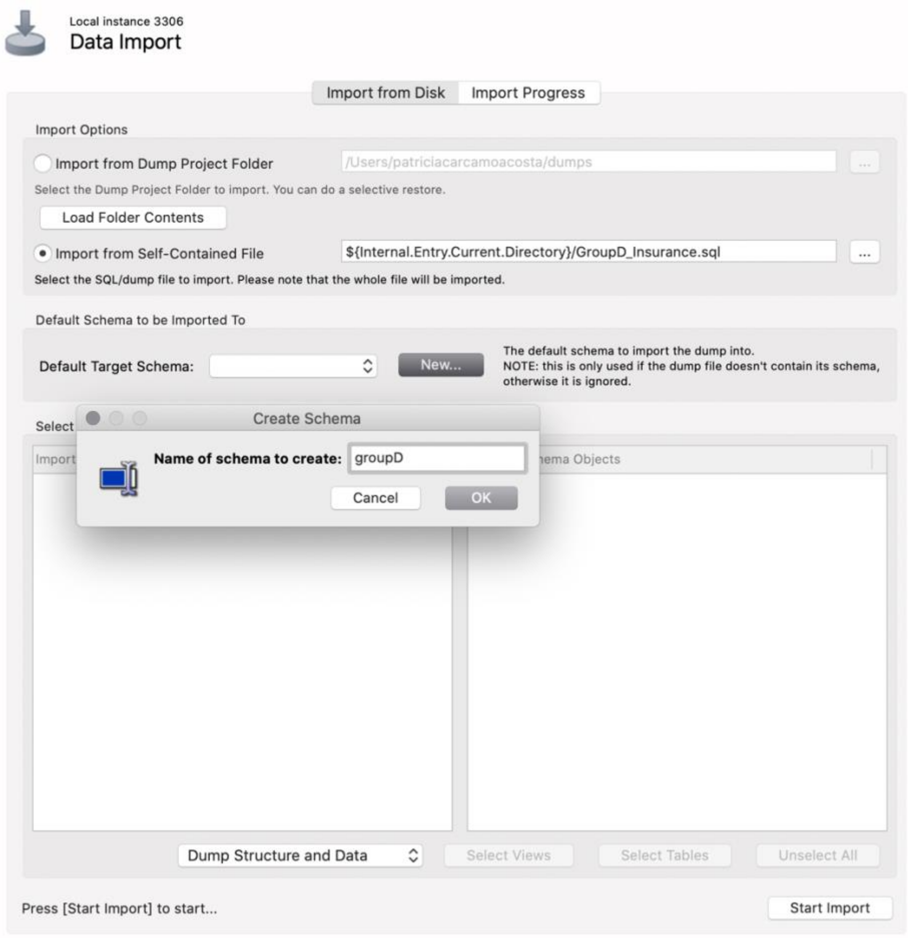
4 ETL Process Instructions and Special Considerations

4.1 Database Load

The changes outlined in section 1.4 Quality Issues were made to our original database design. The updated database is included in the deliverables of this project. The SQL script is named “GroupD_Insurance.sql”.

Load this to MySQL Workbench following these steps:

1. Connect to local instance localhost, port 3306. Username: root. Password: iembd2019 (or the password you changed it to)
2. In the Server tab, select Data Import. Here we will locate the file and create a new schema where the database will be created.
3. To locate the file: Select Import from Self-Contained File and locate the file “**GroupD_Insurance.sql**”.
4. To create a new schema: Under Default Schema to be Imported To, click New button. Then, name the schema **groupD**. Finally click Start Import.



Local instance 3306
Data Import

Import Options

○ Import from Dump Project Folder: /Users/patriciacarcamoacosta/dumps

Select the Dump Project Folder to import. You can do a selective restore.

● Import from Self-Contained File: ${Internal.Entry.Current.Directory}/GroupD_Insurance.sql

Select the SQL/dump file to import. Please note that the whole file will be imported.

Default Schema to be Imported To

Default Target Schema:

The default schema to import the dump into.
NOTE: this is only used if the dump file doesn't contain its schema, otherwise it is ignored.

Create Schema

Name of schema to create: groupD

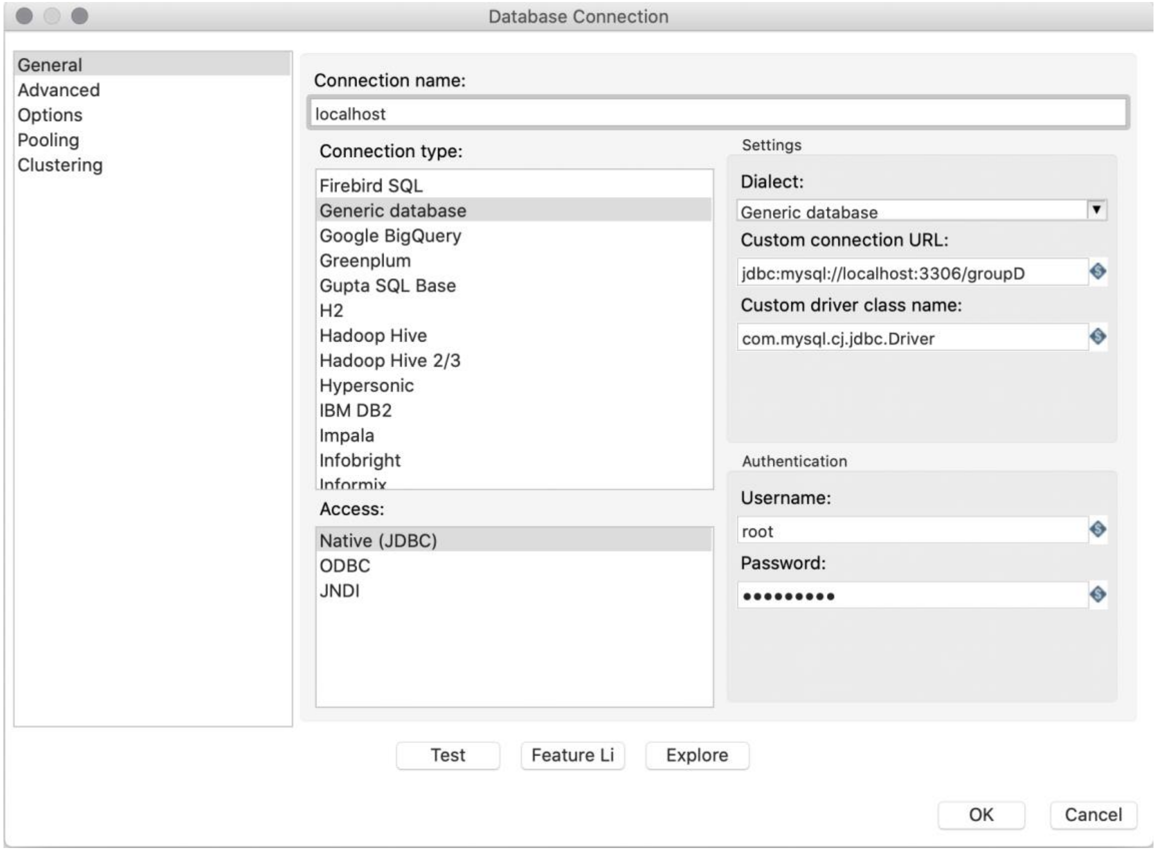
Dump Structure and Data

Press [Start Import] to start...

4.2 Database Connection

In Pentaho Data Integration, go to view Database Connections, the following screen opens up.

7. Connection name: **localhost**
8. Connection Type: **Generic database**
9. Custom Connection URL: **jdbc:mysql://localhost:3306/groupD**
10. Custom driver class name**: com.mysql.cj.jdbc.Driver**
11. Username: **root**
12. Password: **iembd2019** (or a different password if you changed it)
13. Click **OK** button



Database Connection

Connection name: localhost

Connection type: Generic database

Access: Native (JDBC)

Settings

Dialect: Generic database

Custom connection URL: jdbc:mysql://localhost:3306/groupD

Custom driver class name: com.mysql.cj.jdbc.Driver

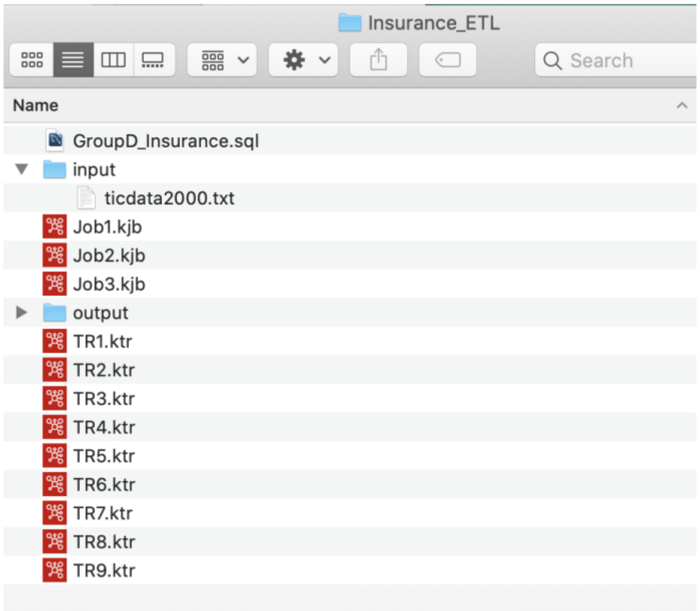
Authentication

Username: root

Password: •••••••••

4.3 Setting Current Directory

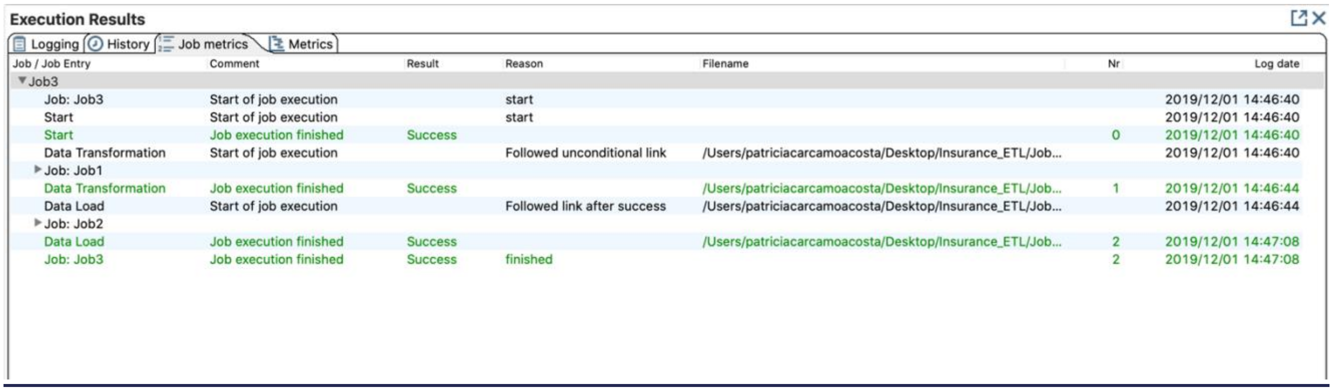
Create a directory for our project in your Desktop and name it **Insurance_ETL** Create sub-directories named **input** and **output.** The original dataset **ticdata2000.txt** should be saved in the **input** sub-directory. Save the jobs and transformations to this project directory.



GroupD_Insurance.sql
input
ticdata2000.txt
Job1.kjb
Job2.kjb
Job3.kjb
output
TR1.ktr
TR2.ktr
TR3.ktr
TR4.ktr
TR5.ktr
TR6.ktr
TR7.ktr
TR8.ktr
TR9.ktr

4.4 Run-Time

Job 3 executes the entire ETL process from start to finish, transforming raw data and outputting it to the pertaining tables in MySQL Server. The run time for this entire process is 28 seconds.



Job / Job Entry	Comment	Result	Reason	Filename	Nr	Log date
▼Job3						
Job: Job3	Start of job execution		start			2019/12/01 14:46:40
Start	Start of job execution		start			2019/12/01 14:46:40
Start	Job execution finished	Success			0	2019/12/01 14:46:40
Data Transformation	Start of job execution		Followed unconditional link	/Users/patriciacarcamoacosta/Desktop/Insurance_ETL/Job...		2019/12/01 14:46:40
▸Job: Job1						
Data Transformation	Job execution finished	Success		/Users/patriciacarcamoacosta/Desktop/Insurance_ETL/Job...	1	2019/12/01 14:46:44
Data Load	Start of job execution		Followed link after success	/Users/patriciacarcamoacosta/Desktop/Insurance_ETL/Job...		2019/12/01 14:46:44
▸Job: Job2						
Data Load	Job execution finished	Success		/Users/patriciacarcamoacosta/Desktop/Insurance_ETL/Job...	2	2019/12/01 14:47:08
Job: Job3	Job execution finished	Success	finished		2	2019/12/01 14:47:08