Data Mapping Document-Independent Expenditures

Document Authorization

Document name	Data Mapping Document-Independent Expenditures.docx
Author(s)	Vanesa Lopez Garcia, Sabrina Mirtcheva, Daan Pelt, Moritz Steinbrecher, Hsiu-Chi Liu, Andrew Rizk, Pravat Ranjan Pasayat
Department	Group O-2-4, MBD-2018, IE HST
Last modified by	
Last modified date	25-Nov-2018
Authorization type	Baselined
Version	1.0

CONTENTS

1. Introduction

The purpose of this document is to illustrate the data integration process conducted between the independent expenditure dataset and the database schema previously created. Mapping acts as a glossary to show how data from the dataset is transformed from its original source form to the final target form and mapped/loaded in different tables. The scope of the document is described below.

2. Scope

The scope of the document is described below.

- **Data preparation or data transformation:** Raw dataset is transformed to a much cleaner and schema compatible form. Field specific transformations are performed on each column of the input dataset to extract a more coherent output dataset that will be later used during data loading process.

- **Data loading:** The output dataset generated after the data transformation step is used to load the necessary data into various dimension tables and the fact table.

- **Metadata approach:** This step describes how the metadata in dimensions and fact table are maintained.

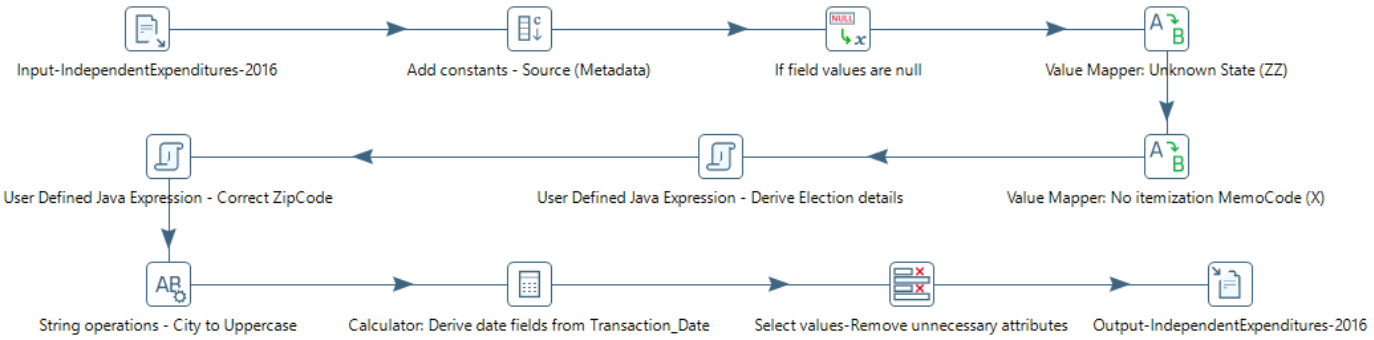
- **ETL execution:** This step describes the process to execute the data preparation and loading process using the tool PDI. During this process we will come across terminologies such as

    - **Transformation:** In a transformation, we execute a specific task such as either data preparation or data loading into a dimension.
    - **Job:** In a job, we perform one or more transformations.

- **Data mapping:** This step describes the overall mapping of dataset attributes into various tables of database schema.

3. Data preparation / Data transformation

Transformation name: TR_DATA_PREPARATION.ktr
Input: Source dataset (INPUT/IndependentExpenditures-2016.csv)
Output: Transformed dataset (OUTPUT/IndependentExpenditures-2016.csv)



As previously mentioned, the purpose of this transformation is to clean the raw independent expenditure dataset to use it for data loading process. All properties/attributes are treated from discrepancies. The implemented transformation is presented as follows.

1. Add constants – Source (Metadata): Add a source column as a reference to where the data originated. In our case, it's Independent-Expenditure2016.csv raw data file.

2. If field values are null: Null or empty values in different attributes are handled accordingly. To populate all the empty cell values, a value of 'UNK' (Annotation for unknown) is used for most properties with the exclusion of transaction data, transaction amount, and memo code fields. For transaction date (transaction_dt), empty values are replaced by a standard date format '0001-01-01' (Format: yyyy-mm-dd). For transaction amounts (transaction_amt), empty values were regarded as '0' donations. For memo code (memo_cd), a property that indicates whether an amount is included in the itemization total, empty values were regarded as 'Y' (Annotation for Yes) to indicate that the value is included in the itemization total.

3. Value Mapper: Unknown State (ZZ): A value mapper is created for the state column to redefine the unknown state values. As previously mentioned, empty state cells were tabulated with 'UNK'. In this step, 'ZZ' (unknow state) is replaced by 'UNK' to have a unified format.

4. Value Mapper: No itemization MemoCode (X): A value mapper is also created for memo code to replace the 'X' values with 'N' (Annotation for No). This indicates that the contribution is not included in the itemization total. So 'N' is used instead of 'X' for an easier understanding from a stakeholder's point of view.

5. User Defined Java Expression: Derive Election details: A user defined Java function is used to split the transaction PGI value into two new values. PGI has a unified format of 'EYYYY' where 'E' stands for election and 'YYYY' stands for election year. So the function was used to have two separate properties, one to specify the type of election and the other to specify the year of election. The functionality extends to using the global value 'UNK' if either election type or year were not found.

6. **User Defined Java Expression: Correct ZipCode:** A user defined Java function was used to correct the discrepancies in the USA zip codes. Correct zip codes formats generally length at 5 or 9 numbers. '0' and '00' were added to zip codes with 3 or 4 numbers, respectively, to unify the format. UNK was used to mark the empty cells.

7. **String Operations: City to Uppercase:** City values were converted to uppercase to avoid inconsistencies.

8. **Calculator: Derive date fields from Transaction_Date:** Transaction date (transaction_dt) was used to derive the day, month, and year of each contribution to create three different properties for loading into the date dimension.

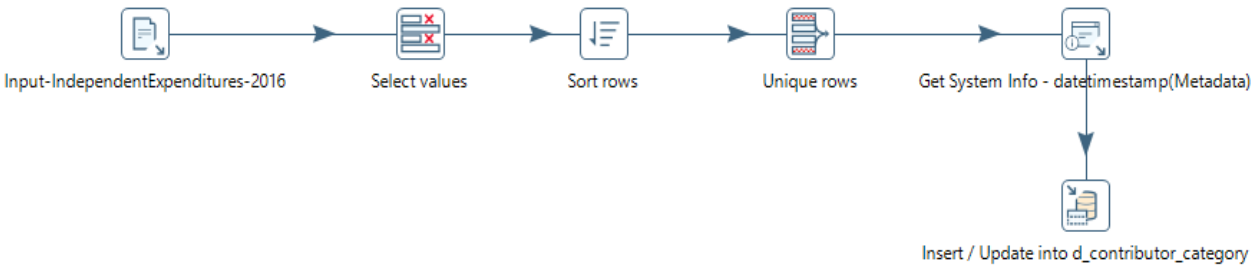
9. **Select values-Remove unnecessary attributes**: Lastly, unused attributes were removed to avoid complexity in the dataset. The removed attributes are file number (file_num), transaction ID (tran_id), and transaction PGI (transaction_pgi).

These transformations were executed, and the resulting output file is used as an input for data loading process.

4. Data loading

4.1 D_CONTRIBUTOR_CATEGORY table

Transformation name: TR_DIM_CONTRIBUTOR_CATEGORY.ktr
Input: Transformed dataset (OUTPUT/IndependentExpenditures-2016.csv)
Output: Insert into D_CONTRIBUTOR_CATEGORY



This transformation is used to load data into d_contributor_category table that holds data related to the type of the individual or group making the expenditure. For this dimension, the attribute entity_tp is extracted

and loaded into the table. This dimension is considered a child dimension for the D_CONTRIBUTOR table. The implemented transformation is presented as follows:

1. **Select values:** entity_tp and source attributes are extracted from the dataset.

2. **Sort rows:** Sort all rows in ascending order by entity type.

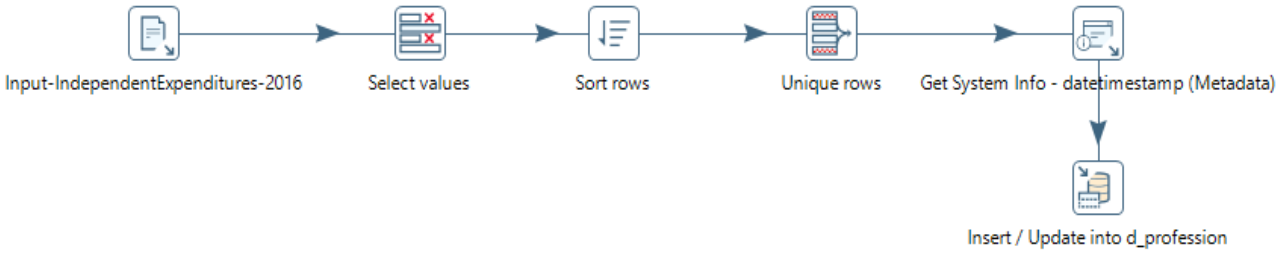
3. **Unique rows:** Get the unique/distinct entity_tp.

4. **Get System Info- datetimestamp(Metadata):** System time info is extracted to add a time stamp for when the data is loaded in the table.

5. **Insert / Update into d_contributor_category:** Insert the variable entity type into the dimension d_contributor_category. First check if the specific entity type received from the dataset is already present in the d_contributor_category table. If not present, then insert the entity type in the d_contributor_category table, else do nothing.

4.2 D_PROFESSION table

Transformation name: TR_DIM_PROFESSION.ktr
Input: Transformed dataset (OUTPUT/IndependentExpenditures-2016.csv)
Output: Insert into D_PROFESSION



This transformation is used to identify and extract the employer and occupation of the contributor and then inserted into the d_profession dimension table. This dimension is considered a child dimension for the d_contributor dimension. The implemented transformation is presented as follows:

1. **Select values:** All the profession related attributes were selected such employer and occupation along with the source attribute.

2. **Sort rows:** Rows are sorted in an ascending order based of occupation and employer attributes.

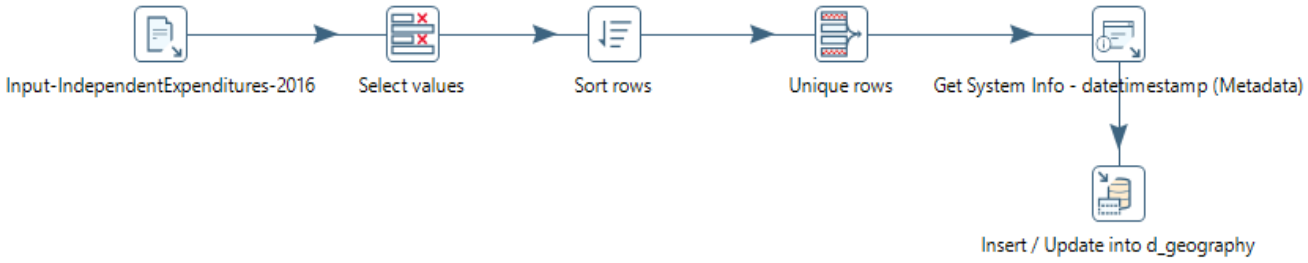
3. **Unique rows:** Unique combinations of employer and occupation are identified.

4. **Get System Info- datetimestamp(Metadata):** System time info is extracted to add a time stamp for when the data is loaded in the table.

5. **Insert / Update into d_professon:** Insert the variable occupation and employer into the dimension d_profession. First check if the specific combination of occupation and employer received from the dataset is already present in the d_profession table. If not present, then insert that combination of occuparion and employer in the d_profession table, else do nothing.

4.3 D_ GEOGRAPHY

Transformation name: TR_DIM_GEOGRAPHY.ktr
Input: Transformed dataset (OUTPUT/IndependentExpenditures-2016.csv)
Output: Insert into D_GEOGRAPHY



The geography dimension defines the geographic location of the contributor. For this dimension, geography related attributes such as zip code, city, and state are identified from the prepared data file and inserted into the geography dimension; d_geography table. The implemented transformation is presented as follows:

1. **Select values:** First the zip code, city, state, and source are extracted from the dataset.

2. **Sort rows:** The extracted attributes are then sorted ascendingly based on the three attributes zip code, city, and state.

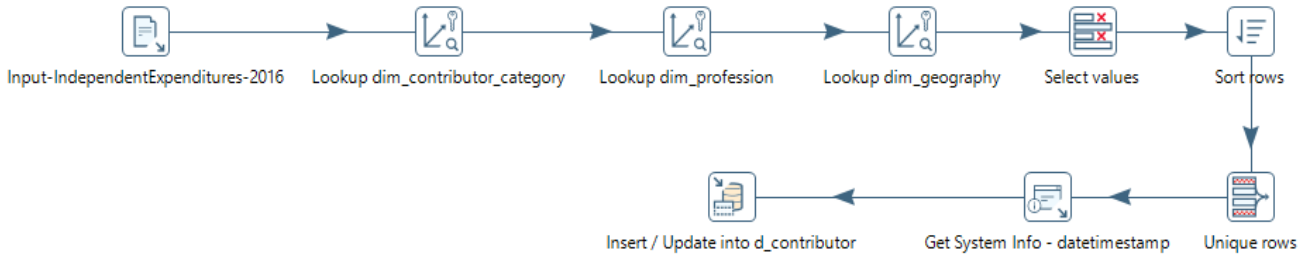
3. **Unique rows:** Unique combinations of the three attributes were identified and selected from the output file.

4. **Get System Info- datetimestamp(Metadata):** System time info is extracted to add a time stamp for when the data is loaded in the table.

5. **Insert / Update into d_grography:** Insert the variable state, city and zipcode into the dimension d_geography. First check if the specific combination of state, city and zipcode received from the dataset is already present in the d_geography table. If not present, then insert that combination of state, city and zipcode in the d_geography table, else do nothing.

4.4 D_ CONTRIBUTOR

Transformation name: TR_DIM_CONTRIBUTOR.ktr
Input: Transformed dataset (OUTPUT/IndependentExpenditures-2016.csv)
Output: Insert into D_CONTRIBUTOR



The purpose of this transformation is to collect relevant information regarding the contributor of a transaction and load it into the d_contributor table. The variables that hold contributor's information are name, fec_id_name, id_geography, id_profession and id_contributor_category. Furthermore, this dimension is the "parent dimension" of three previously mentioned "child dimensions". The implemented transformation is presented as follows:

1. **Lookup dim_contributor_category:** Find out the primary key for the entity_type value of the contributor in d_contributor_category table. This primary key is present as a foreign key in d_contributor table.

2. **Lookup dim_profession:** Find out the primary key for the occupation and employer value of the contributor in d_profession table. This primary key is present as a foreign key in d_contributor table.

3. **Lookup dim_geography:** Find out the primary key for the state, city and zipcode value of the contributor in d_geography table. This primary key is present as a foreign key in d_contributor

table.

4. **Select values:** Select name, fec_id_name, id_geography, id_profession, id_contributor_category and source from the dataset.

5. **Sort rows:** The extracted attributes are then sorted ascendingly based on name, fec_id_name, id_geography, id_profession and id_contributor_category attributes.

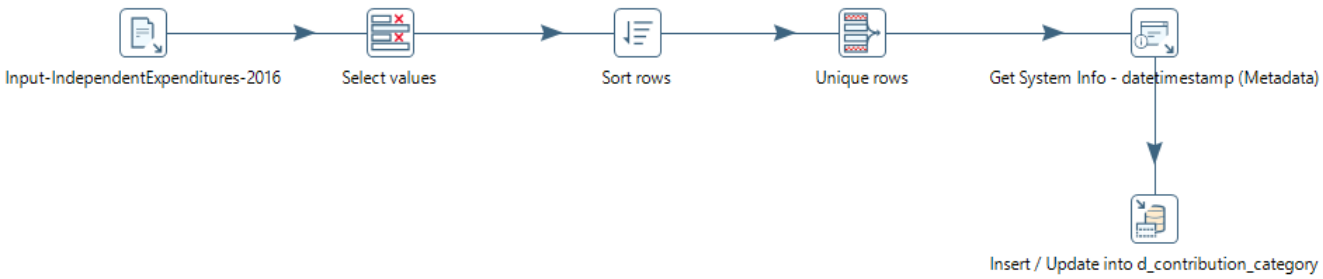
6. **Unique rows:** Unique combination of name, fec_id_name, id_geography, id_profession and id_contributor_category attributes are identified.

7. **Get System Info- datetimestamp(Metadata):** System time info is extracted to add a time stamp for when the data is loaded in the table.

8. **Insert / Update into d_contributor:** Insert the variable name, fec_id_name, id_geography, id_profession and id_contributor_category into the dimension d_contributor. First check if the specific combination of name, fec_id_name, id_geography, id_profession and id_contributor_category received from the dataset is already present in the d_contributor table. If not present, then insert that combination of name, fec_id_name, id_geography, id_profession and id_contributor_category in the d_contributor table, else do nothing.

4.5 D_CONTRIBUTION_CATEGORY table

Transformation name: TR_DIM_CONTRIBUTION_CATEGORY.ktr
Input: Transformed dataset (OUTPUT/IndependentExpenditures-2016.csv)
Output: Insert into D_CONTRIBUTION_CATEGORY



For this transformation the purpose is to identify the different contribution categories from the dataset in order to load them into the target table d_contribution_category. After this transformation, the stakeholder

can easily search for the different contribution types of each transaction. The implemented transformation is presented as follows:

1. **Select values:** transaction_tp and source attributes are extracted from the dataset.

2. **Sort rows:** Sort all rows in ascending order by transaction_tp.

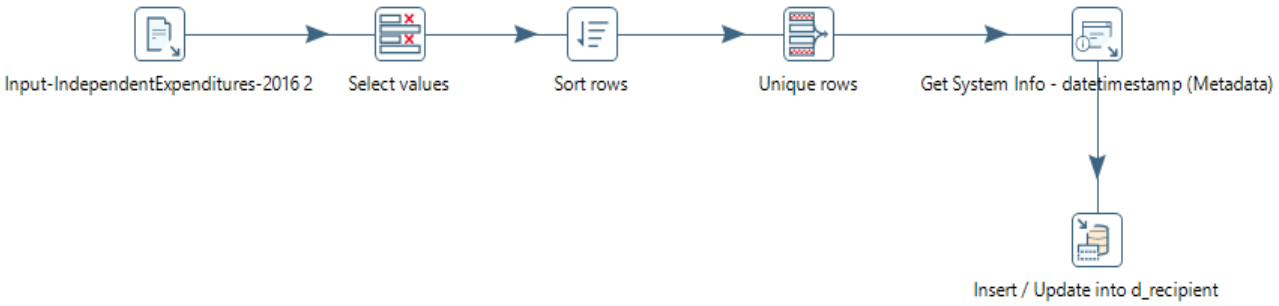
3. **Unique rows:** Get the unique/distinct transaction_tp.

4. **Get System Info- datetimestamp(Metadata):** System time info is extracted to add a time stamp for when the data is loaded in the table.

5. **Insert / Update into d_contribution_category:** Insert the variable transaction_tp into the dimension d_contribution_category. First check if the specific transaction_tp received from the dataset is already present in the d_contribution_category table. If not present, then insert the entity type in the d_contribution_category table, else do nothing.

4.6 D_RECIPIENT table

Transformation name: TR_DIM_RECIPIENT.ktr
Input: Transformed dataset (OUTPUT/IndependentExpenditures-2016.csv)
Output: Insert into D_RECIPIENT



For this transformation the purpose is to identify the different committee ID's and candidate ID's present in the dataset in order to load them into the target table: d_recipient. The implemented transformation is presented as follows:

1. **Select values:** cmte_id, cand_id and source attributes are extracted from the dataset.

2. **Sort rows:** Sort all rows in ascending order by cmte_id and cand_id.

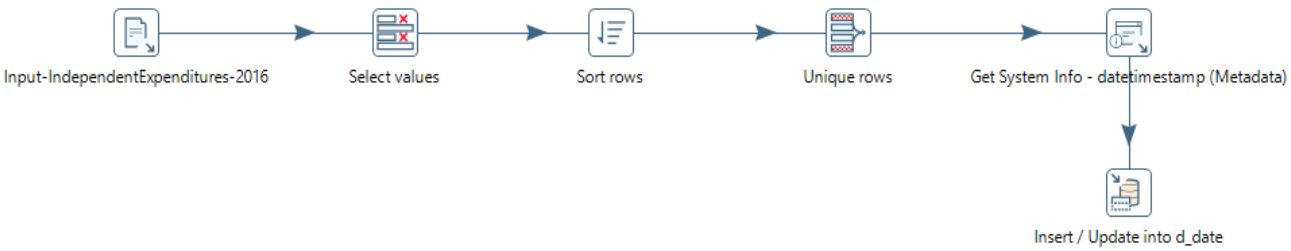
3. **Unique rows:** Get the unique/distinct combination of cmte_id and cand_id.

4. **Get System Info- datetimestamp(Metadata):** System time info is extracted to add a time stamp for when the data is loaded in the table.

5. **Insert / Update into d_recipient:** Insert the variable cmte_id and cand_id into the dimension d_recipient. First check if the specific combination of cmte_id and cand_id received from the dataset is already present in the d_recipient table. If not present, then insert the combination of cmte_id and cand_id in the d_recipient table, else do nothing.

4.7 D_DATE table

Transformation name: TR_DIM_DATE.ktr
Input: Transformed dataset (OUTPUT/IndependentExpenditures-2016.csv)
Output: Insert into D_DATE



This purpose of this transformation is to extract the date at which the transaction is made by a contributor. The attributes month, day, and year are previously extracted from the date during data transformation process. All these date related fields are then loaded into the dimension d_date. The implemented transformation is presented as follows:

1. **Select values:** year, month, day, source, and transaction_dt attributes are extracted from the dataset.

2. **Sort rows:** Rows were sorted based on the transaction date in an ascending order.

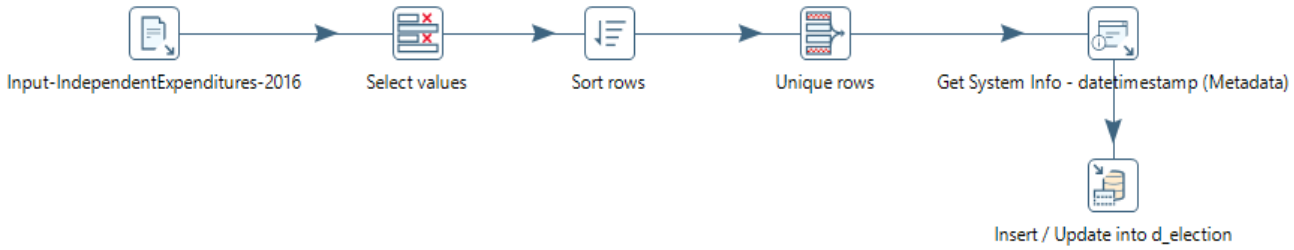
3. **Unique rows:** Get the unique/distinct transaction dates.

4. **Get System Info- datetimestamp(Metadata):** System time info is extracted to add a time stamp for when the data is loaded in the table.

5. **Insert / Update into d_date:** Insert the variable year, month, day and transaction date into the dimension d_date. First check if the specific transaction date received from the dataset is already present in the d_date table. If not present, then insert the year, month, day and transaction date in the d_date table, else do nothing.

4.8 D_ELECTION table

Transformation name: TR_DIM_ELECTION.ktr
Input: Transformed dataset (OUTPUT/IndependentExpenditures-2016.csv)
Output: Insert into D_ELECTION



The purpose of this transformation is to identify the election type and election year (originally extracted from the PGI property), extract them, and then inserted into the dimension d_election. The implemented transformation is presented as follows:

1. **Select values:** election type and election year attributes are extracted from the dataset.

2. **Sort rows:** Rows were sorted based on the election type and election year in an ascending order.

3. **Unique rows:** Get the unique/distinct combinations of election type and election year.

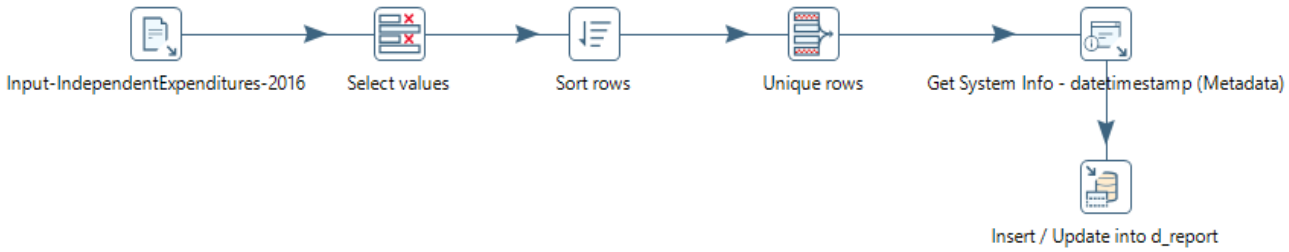
4. **Get System Info- datetimestamp(Metadata):** System time info is extracted to add a time stamp for when the data is loaded in the table.

5. **Insert / Update into d_election:** Insert the variable election type and election year into the dimension d_election. First check if the specific combination of election type and election year

received from the dataset is already present in the d_election table. If not present, then insert the election type and election year in the d_election table, else do nothing.

4.9 D_REPORT table

Transformation name: TR_DIM_REPORT.ktr
Input: Transformed dataset (OUTPUT/IndependentExpenditures-2016.csv)
Output: Insert into D_REPORT



The purpose of this transformation is to extract the report related attributes such as amndt_ind and rpt_tp and then load into d_report. The implemented transformation is presented as follows:

1. **Select values:** amndt_ind and rpt_tp attributes are extracted from the dataset.

2. **Sort rows:** Rows were sorted based on the amndt_ind and rpt_tp in an ascending order.

3. **Unique rows:** Get the unique/distinct combinations of amndt_ind and rpt_tp.

4. **Get System Info- datetimestamp(Metadata):** System time info is extracted to add a time stamp for when the data is loaded in the table.

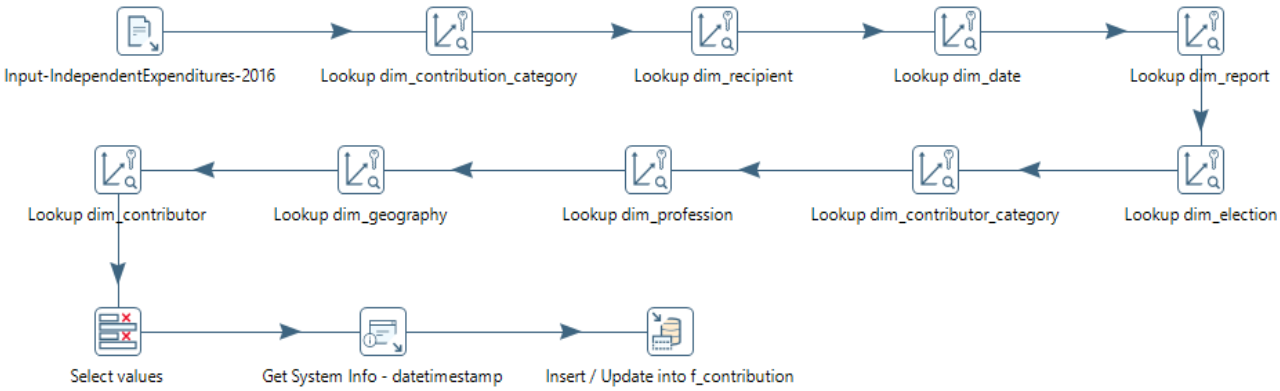
5. **Insert / Update into d_election:** Insert the variable amndt_ind and rpt_tp into the dimension d_report. First check if the specific combination of amndt_ind and rpt_tp received from the dataset is already present in the d_report table. If not present, then insert the amndt_ind and rpt_tp in the d_report table, else do nothing.

4.10 F_ CONTRIBUTION

Transformation name: TR_FACT.ktr

Input: Transformed dataset (OUTPUT/IndependentExpenditures-2016.csv)

Output: Insert into F_CONTRIBUTION



The purpose of this transformation is to collect relevant information regarding the contribution/expenditure and load it into the f_contribution table. Furthermore, this dimension is the "parent dimension" of all previously mentioned "child dimensions". The implemented transformation is presented as follows:

1. **Lookup dim_contribution_category:** Find out the primary key for the transaction_type value of the expenditure in d_contribution_category table. This primary key is present as a foreign key in f_contribution table.

2. **Lookup dim_recipient:** Find out the primary key for the committee_id and candidate_id value of the expenditure in d_recipient table. This primary key is present as a foreign key in f_contribution table.

3. **Lookup dim_date:** Find out the primary key for the transaction_date value of the expenditure in d_date table. This primary key is present as a foreign key in f_contribution table.

4. **Lookup dim_report:** Find out the primary key for the amendment_id and report_type value of the expenditure in d_report table. This primary key is present as a foreign key in f_contribution table.

5. **Lookup dim_election:** Find out the primary key for the election_type and election_year value of the expenditure in d_election table. This primary key is present as a foreign key in f_contribution table.

6. **Lookup dim_contributor_category:** Find out the primary key for the entity_type value of the contributor in d_contributor_category table. This primary key is present as a foreign key in d_contributor table.

7. Lookup dim_profession: Find out the primary key for the occupation and employer value of the contributor in d_profession table. This primary key is present as a foreign key in d_contributor table.

8. Lookup dim_geography: Find out the primary key for the state, city and zipcode value of the contributor in d_geography table. This primary key is present as a foreign key in d_contributor table.

9. Lookup dim_contributor: Find out the primary key for the name, fec_id_name, id_geography, id_profession, id_contributor_category value of the expenditure in d_contributor table. This primary key is present as a foreign key in f_contribution table.

10. Select values: Select transaction_amt, image_num, memo_cd, memo_text, id_contribution_category, id_recipient, id_date, id_report, id_election, id_contributor, sub_id and source from the dataset.

11. Get System Info- datetimestamp(Metadata): System time info is extracted to add a time stamp for when the data is loaded in the table.

12. Insert / Update into d_contribution: First check if the specific sub_id received from the dataset is already present in the d_contributor table. If not present, then insert transaction_amt, image_num, memo_cd, memo_text, id_contribution_category, id_recipient, id_date, id_report, id_election, id_contributor, sub_id and source in the d_contributor table, else do nothing.

5. Metadata approach

Metadata are said to be the data of data i.e. information that describes the business data. We have included below metadata in all our dimensions and fact table. During the ETL processes, these metadata attributes are filled with relevant information.

1. Source: This metadata describes the point of origin of the data. During the data preparation step (section 3), an extra column is added for the source attribute and filled with a constant value 'IndependentExpenditures-2016.csv'. This value is loaded into all the dimensions and fact table.

2. Datestamp: This metadata describes the time when the data is loaded (inserted/updated) into a specific dimension or fact table. This variable is created in each transformation related to data loading (section 5). The datestamp is extracted from the system and loaded into various

dimensions and fact table.

6. ETL execution

ETL is executed using various jobs. Below job is executed in order to transform the data first and then load the transformed data into the dimension and fact table.

1. Execute the job **JB_PARENT_INDEPENDENT_EXPENDITURE.kjb**
   This the parent job that initiates below mentioned child jobs sequentially to complete the whole ETL process.

   a. Execute the job **JB_DATA_PREPARATION.kjb**
      This job executes TR_DATA_PREPARATION in the background.

   b. Execute the job **JB_LOAD_SECONDARY_DIM_DATA.kjb**
      This job executes all the data loading transformations for the secondary dimensions (child of child). Secondary dimension transformations such as TR_DIM_CONTRIBUTOR_CATEGORY, TR_DIM_PROFESSION and TR_DIM_GEOGRAPHY are executed sequentially by this job.

   c. Execute the job **JB_LOAD_PRIMARY_DIM_DATA.kjb**
      This job executes all the data loading transformations for the primary dimensions (child of fact). Primary dimension transformations such as TR_DIM_CONTRIBUTOR, TR_DIM_CONTRIBUTION_CATEGORY, TR_DIM_RECIPIENT and TR_DIM_DATE, TR_REPORT and TR_ELECTION are executed sequentially by this job.

   d. Execute the job **JB_LOAD_FACT_DATA.kjb**
      This job executes the data loading transformation for the fact table. Fact table transformations is TR_FACT.

7. Data mapping:

Please find the attached data mapping sheet (Data Mapping. xlsx)

Sheet-1: Mapping-Data Preparation: Mapping and transformation of attributes from Input raw dataset to the Output transformed dataset.

Sheet-2: Mapping-Data Loading: Mapping of attributes from transformed dataset to database schema i.e. to

fact and various dimensions tables.

Data Mapping.xlsx

(This excel document is submitted separately – in case you encounter issue opening the attachment)