Business Intelligence and Datawarehousing

H2-A Visa Data Integration Process

MBD O-1
Team H

Introduction

After the initial step of designing a database model for H2-A visa application data we will be loading the data into an SQL database following this model. This document outlines the steps to be taken to load the historical data sets onto an SQL database and describes the underlying steps in the ETL process.

Instructions

To execute the ETL process designed follow the following instructions:

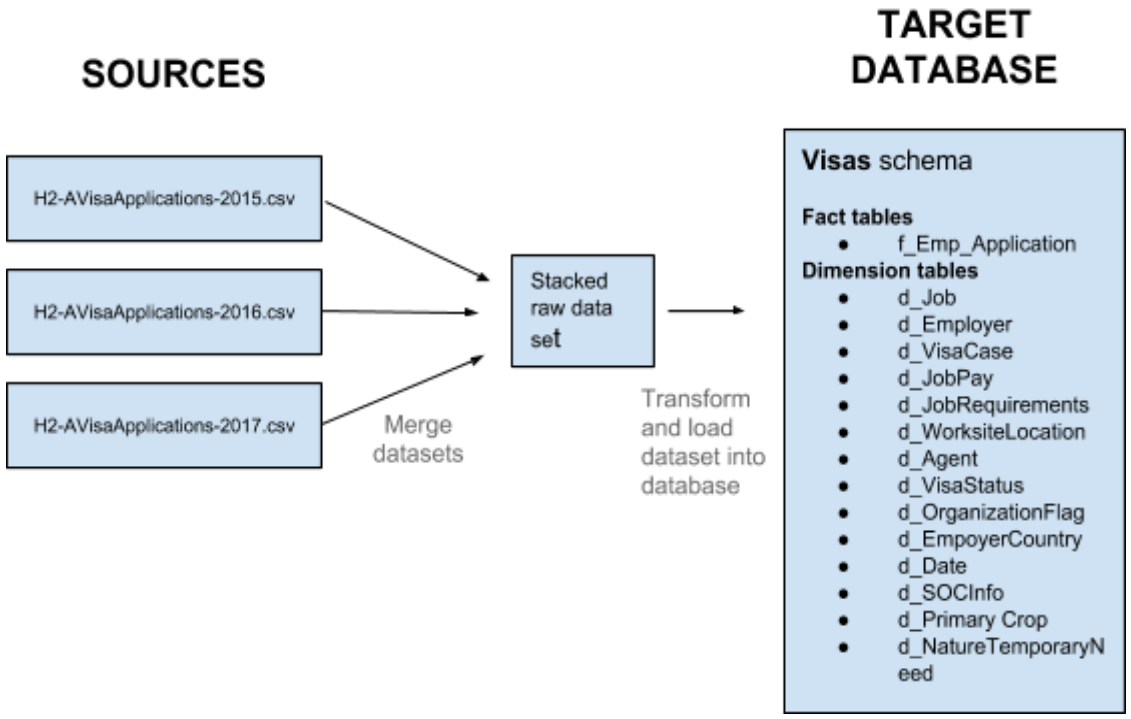
1) Open the database design model file **VisaModel.mwb** in **MySQLWorkbench** and **Forward Engineer** the model to create an SQL schema called **visas.**
2) Download the folder **H2AVisaTransformation**. This folder contains:
    a) All Pentaho transformations/jobs needed
    b) An empty **Output** folder
    c) An **Input** folder with original 3 CSV files for historical H2-A visa data for 2015, 2016, and 2017
        i) *Optional*: re-download and replace the 3 original raw CSV files in this folder
3) Open the job **JOB_VisaETL** in the **H2AVisaTransformation** folder in Pentaho**.**
    a) If necessary, update the connection setting named *'Visas'* in this Job and share it with all transformations linked from this Job by right clicking on it and clicking **Share**.
    -> Connection type: Generic Database
    -> Access: Native (JDBC)
    -> Connection:
    jdbc:mysql://localhost:3306/visas?useLegacyDatetimeCode=false&serverTimezone=UTC
    -> Driver Class Name: com.mysql.cj.jdbc.Driver
4) Run the job **JOB_VisaETL**
    a) This job will run all transformations needed to load the database.Total run time is around 12 minutes.

High-Level ETL Plan

At a high-level this ETL process will extract information from historic raw datasets in CSV format on H2-A Visa applications and load an SQL database based on a previously decided database model design (**VisaModel.mwb).**

The sources for the historical data set are 3 separate CSV files containing visa application information for the years 2015, 2016, and 2017 respectively. Ultimately the goal is to load an SQL database schema **Visas** created based on the database design model with the relevant data from these files.

As described in the figure below, the



ETL Tool and Design

In order to execute the ETL process for the historical H2-A Visa datasets we will be using Pentaho Data Integration. A number of jobs and transformations will load the raw data sets and perform necessary transformations to correctly format the data and load the SQL database based on the database model previously decided on. The details on each transformation are described below in **Drill Down by Target Table** section.

Hierarchy of Tables

Based on the previously decided database design model there is a specific order in which the tables must be loaded onto the SQL database through the ETL process. A series of foreign keys link some SQL tables to others. Therefore, in order to load tables with any foreign key, the dependent dimension tables must be loaded first.

The overall order is therefore (1) load dimensional tables with no foreign keys, (2) load dimensional tables *with* foreign keys, and finally (3) load the fact table.

The list of tables to be loaded in each overall step is outlined below, color-coded to highlight direct dependencies between the tables.

1) Dimension tables with no foreign keys:
    a) d_CaseStatus
    b) d_Agent
    c) d_EmployerCountry
    d) d_Date
    e) d_SOCInfo
    f) d_PrimaryCrop
    g) d_JobRequirements
    h) d_JobPay
    i) d_NatureTemporaryNeed
    j) d_WorksiteLocation
    k) d_OrganizationFlag
2) Dimension tables with foreign keys:
    a) d_VisaCase
    b) d_Employer
    c) d_Job
3) Fact table:
    a) f_Emp_Application

Overall ETL Job

The overall ETL process is led by a main job **JOB_VisaETL**. This single job runs all transformations needed to extract the raw data, transform it, and load the database.

The first two steps of the job involve preparing and cleaning the data. During these two transformations a temporary CSV with a clean and complete data set will be saved in the working directory. A number of transformations are then executed to load the data into SQL tables. The order of these transformations aligns with the hierarchy previously described. Each transformation's specific ETL process is outlined in the **Drill Down** section below.

Drill Down by Transformation and Target Table

In this section each transformation in the ETL process is described. If the transformation loads a target table in the database this is highlighted in the **Target Table** section which includes the fields and respective data types for the table.

Data Preparation Transformation

TR_DataPreparation		
Extracting		3 Historical data sets for H2-A visa applications for the years 2015,2016, 2017, all in CSV format
Transforming	**1**	Input CSV datasets (*H2A-VisaApplications-2015.csv, H2A-VisaApplications-2016.csv, H2A-VisaApplications-2017.csv*)
	2	Add an Origin column to each year's data stream indicating from which year's CSV is the row (for error handling purposes)
	3	Add missing fields (primary_sub; trade_name_dba; agent_poc; worksite_county) to 2015 and 2016 data streams (since there are more fields in 2017)
	4	Delete field *serial_id* as it is not given and necessary in 2017 anymore
	5	Re-order and rename fields in 2015 & 2016 to match 2017 data stream
	6	Append data streams for 2015 and 2016
	7	Append previously appended data stream to 2017 stream
	8	Remove special characters from name_reqd_training, major, and employer_name fields which were splitting up rows
Loading		Export stacked dataset to a CSV file in the Output folder with name *2015_16_17_stacked_temp.csv*

Data Cleaning Transformation

TR_DataCleaning		
Extracting		Load the previously outputted file 2015_16_17_stacked_temp.csv with the data on all 3 years stacked
Transforming	**1**	Input stacked temporary CSV
	2	Reformat metadata on necessary columns
	3	Replace any NULL values in the ‘experience required’ field to FALSE
	4	Replace any NULL values in the ‘training required’ field to FALSE
	5	Replace empty pay rate values to average value (given our model restricts this field to be NOT NULL since is is a mandatory visa application field and there are only 7 total rows affected)
	6	Extract the dates from the DATETIME fields with regex operations (*cert_begin_date, cert_end_date, job_start_date, job_end_date, decision_date, case_received_date*)
	7	Trim fields for 6 date columns
	8	Set correct format for 6 date columns
	9	Replace missing date values with a default date of ‘01/01’1900’
	10	Merge the 2 employer address fields and the 2 employer telephone fields
	11	Replace any NULL values in the ‘SuperviseOtherEmp’ field to FALSE
	12	Replace any NULL values in the ‘FullTime’ field to FALSE
	13	Replace any NULL values in the ‘BasicNumberHours’ field by the average value
	14	Replace any NULL values in soc_code swa_name, basic_unit_of_pay , employer_name , agent_attorney_name , soc_title, hourly_work_schedule_am , hourly_work_schedule_pm, job_title, organization_flag , nature_of_temporary_need, employer_country, primary_crop by empty string
	15	Replace any NULL values in the NumberWorkersReq NumberWorkersCert field by 0
Loading		Export stacked dataset to a CSV file in the Output folder with name 2015_16_17_stacked.csv

Case Status Transformation

TR_CaseStatus		
Extracting		Load the previously outputted CSV 2015_16_17_stacked.csv with the data on all 3 years stacked and data cleansing done in previous transformation.
Transforming	**1**	Select the desired column (CaseStatus) from input file
	2	Sort rows for posterior duplicate removal
	3	Remove duplicates with Unique rows step
	4	Add sequence ID as identifier
Loading		Load data in d_CaseStatus table in visas Schema

Target Table Details	
d_CaseStatus	
Field name	**Data type**
id_CaseStatus	INT
CaseStatus	VARCHAR

Agent Transformation

TR_Agent		
Extracting		Load the previously outputted CSV 2015_16_17_stacked.csv with the data on all 3 years stacked and data cleansing done in previous transformation.
Transforming	**1**	Select the desired columns related with Agent Dimension from input file.
	2	Sort rows for posterior duplicate removal
	3	Remove duplicates with Unique rows step
	4	Add sequence ID as identifier
Loading		Load data in d_Agent table in visas Schema

Target Table Details	
d_Agent	
Field name	**Data type**
id_Agent	INT
AgentAttorneyName	CHAR
AgentAttorneyCity	CHAR
LawFirmName	CHAR
AgentAttorneyState	CHAR

Visa Case Transformation

TR_VisaCase		
Extracting		Load the previously outputted CSV 2015_16_17_stacked.csv with the data on all 3 years stacked and data cleansing done in previous transformation.
Transforming	**1**	Select the desired columns related with VisaCase Dimension from input file.
	2	Sort rows for posterior duplicate removal
	3	Remove duplicates with Unique rows step
	4	Look for the CaseStatus and Agent foreign keys
	5	Add sequence ID as identifier
Loading		Load data in d_VisaCase table in visas Schema

Target Table Details	
d_VisaCase	
Field name	**Data type**
idf_VisaCase	INT
id_CaseStatus	INT
id_Agent	INT
CaseNumber	VARCHAR
SWA_name	CHAR

Organization Flag Transformation

TR_OrganizationFlag		
Extracting		Load the previously outputted CSV 2015_16_17_stacked.csv with the data on all 3 years stacked and data cleansing done in previous transformation.
Transforming	**1**	Select the desired columns related with OrganizationFlag Dimension from input file.
	2	Sort rows for posterior duplicate removal
	3	Remove duplicates with Unique rows step
	4	Add sequence ID as identifier
Loading		Load data in d_OrganizationFlag table in visas Schema

Target Table Details	
d_OrganizationFlag	
Field name	**Data type**
id_OrganizationFlag	INT
OrganizationFlag	VARCHAR

Employer Country Transformation

TR_EmployerCountry		
Extracting		Load the previously outputted CSV 2015_16_17_stacked.csv with the data on all 3 years stacked and data cleansing done in previous transformation.
Transforming	**1**	Select the desired columns related with EmployerCountry Dimension from input file.
	2	Sort rows for posterior duplicate removal
	3	Remove duplicates with Unique rows step
	4	Add sequence ID as identifier
Loading		Load data in d_EmployerCountry table in visas Schema

Target Table Details	
d_EmployerCountry	
Field name	**Data type**
id_EmployerCountry	INT
EmployerCountry	CHAR

Employer Transformation

TR_Employer		
Extracting		Load the previously outputted CSV 2015_16_17_stacked.csv with the data on all 3 years stacked and data cleansing done in previous transformation.
Transforming	**1**	In the initial step, we are selecting the desired columns related with Employer Dimension from input file.
	2	Look for the Employer Country foreign key
	3	Sort rows for posterior duplicate removal
	4	Remove duplicates with Unique rows step
	5	Add sequence ID as identifier
Loading		Load data in d_Employer table in visas Schema

Target Table Details	
d_Employer	
Field name	**Data type**
id_Employer	INT
id_EmployerCountry	INT
EmployerName	CHAR
EmployerAddress	VARCHAR
EmployerState	CHAR
EmployerPostalCode	VARCHAR
EmployerPhone	VARCHAR
PrimarySub	CHAR
NAICS_Code	VARCHAR
TradeNameDBA	VARCHAR
AgentIsPOCEmployer	Boolean (TINYINT)
EmployerCity	CHAR

Date Transformation

TR_Date		
Extracting		Load the previously outputted CSV 2015_16_17_stacked.csv with the data on all 3 years stacked and data cleansing done in previous transformation.
Transforming	**1**	In the initial step we are selecting each of the Dates available separately
	2	Sort each Date field for posterior duplicate removal
	3	Remove duplicates with Unique rows step
	4	Appending each one of the Dates consecutively until they are all stacked
	5	Sort the stacked list for posterior removal of duplicates
	6	Remove duplicates from the stacked list with Unique rows step
	7	Add sequence ID as identifier
Loading		Load data in d_Date table in visas Schema

Target Table Details	
d_Date	
Field name	**Data type**
id_Date	INT
Date	Date

Job Requirements Transformation

TR_JobRequirements		
Extracting		Load the previously outputted CSV 2015_16_17_stacked.csv with the data on all 3 years stacked and data cleansing done in previous transformation.
Transforming	**1**	Select the desired columns related with JobRequirements Dimension from input file.
	2	Sort rows for posterior duplicate removal
	3	Remove duplicates with Unique rows step
	4	Add sequence ID as identifier
Loading		Load data in d_JobRequirements table in visas Schema

Target Table Details	
d_JobRequirements	
Field name	**Data type**
idd_JobRequirements	INT
TrainingReq	Boolean (TINYINT)
NumMonthTraining	Boolean (TINYINT)
EmpExperienceReq	Boolean (TINYINT)
EmpExperienceNumMonths	Boolean (TINYINT)
NameReqTraining	VARCHAR

Nature Temporary Need Transformation

TR_NatureTemporaryNeed		
Extracting		Load the previously outputted CSV 2015_16_17_stacked.csv with the data on all 3 years stacked and data cleansing done in previous transformation.
Transforming	**1**	Select the desired columns related with NatureTemporaryNeed Dimension from input file.
	2	Sort rows for posterior duplicate removal
	3	Remove duplicates with Unique rows step
	4	Add sequence ID as identifier
Loading		Load data in d_NatureTemporaryNeed table in visas Schema

Target Table Details	
d_NatureTemporaryNeed	
Field name	**Data type**
id_NatureTemporaryNeed	INT
NatureTemporaryNeed	CHAR

Primary Crop Transformation

TR_PrimaryCrop		
Extracting		Load the previously outputted CSV 2015_16_17_stacked.csv with the data on all 3 years stacked and data cleansing done in previous transformation.
Transforming	**1**	Select the desired columns related with PrimaryCrop Dimension from input file.
	2	Sort rows for posterior duplicate removal
	3	Remove duplicates with Unique rows step
	4	Add sequence ID as identifier
Loading		Load data in d_PrimaryCrop table in visas Schema

Target Table Details	
d_PrimaryCrop	
Field name	**Data type**
id_PrimaryCrop	INT
PrimaryCrop	CHAR

Job Pay Transformation

TR_JobPay		
Extracting		Load the previously outputted CSV 2015_16_17_stacked.csv with the data on all 3 years stacked and data cleansing done in previous transformation.
Transforming	**1**	Select the desired columns related with JobPay Dimension from input file.
	2	Sort rows for posterior duplicate removal
	3	Remove duplicates with Unique rows step
	4	Add sequence ID as identifier
Loading		Load data in d_JobPay table in visas Schema

Target Table Details	
d_JobPay	
Field name	**Data type**
id_JobPay	INT
BasicUnitPay	CHAR
OvertimeRateFrom	DECIMAL
OvertimeRateTo	DECIMAL
BasicRatePay	DECIMAL

Worksite Location Transformation

TR_WorksiteLocation		
Extracting		Load the previously outputted CSV 2015_16_17_stacked.csv with the data on all 3 years stacked and data cleansing done in previous transformation.
Transforming	**1**	Select the desired columns related with WorksiteLocation Dimension from input file.
	2	Sort rows for posterior duplicate removal
	3	Remove duplicates with Unique rows step
	4	Add sequence ID as identifier
Loading		Load data in d_WorksiteLocation table in visas Schema

Target Table Details	
d_WorksiteLocation	
Field name	**Data type**
id_WorksiteLocation	INT
WorksiteCity	VARCHAR
WorksiteState	VARCHAR
WorksitePostalCode	VARCHAR
OtherWorksiteLocation	Boolean (TINYINT)

SOC Info Transformation

TR_SOCInfo		
Extracting		Load the previously outputted CSV 2015_16_17_stacked.csv with the data on all 3 years stacked and data cleansing done in previous transformation.
Transforming	**1**	Select the desired columns related with SOCInfo Dimension from input file.
	2	Sort rows for posterior duplicate removal
	3	Remove duplicates with Unique rows step
	4	Add sequence ID as identifier
Loading		Load data in d_SOCInfo table in visas Schema

Target Table Details	
d_SOCInfo	
Field name	**Data type**
id_SOCCode	INT
Title	VARCHAR
SOCCode	VARCHAR

Job Transformation

TR_Job		
Extracting		Load the previously outputted CSV 2015_16_17_stacked.csv with the data on all 3 years stacked and data cleansing done in previous transformation.
Transforming	**1**	Select the desired columns related with Job Dimension from input file.
	2	Look for the foreign keys related to the Job Dimension
	3	Add sequence ID as identifier
Loading		Load data in d_Job table in visas Schema

Target Table Details	
d_Job	
Field name	**Data type**
id_Job	INT
JobTitle	CHAR
id_PrimaryCrop	INT
id_NatureTemporaryNeed	INT
BasicNumberHours	INT
HourlyWorkScheduleAM	VARCHAR
HourlyWorkSchedulePM	VARCHAR
id_JobPay	INT
SuperviseOtherEmp	BOOLEAN
SuperviseHowMany	INT
id_WorksiteLocationl	INT
JobStartDate	INT
JobEndDate	INT
id_SOCCode	INT
FullTime	BOOLEAN
id_JobRequirementsID	INT
SWA_JobIDNumber	VARCHAR

Fact Table Transformation

TR_FactVisaApplication		
Extracting		Input the stacked csv with all the data related to the Fact Table
Transforming	**1**	Select the desired columns related with f_emp_application fact table from input file.
	2	Look for the foreign keys related to the f_emp_application
	3	Add sequence ID as identifier
Loading		Load data in f_emp_application table in visas Schema

Target Table Details	
f_emp_application	
Field name	**Data type**
idf_Emp_Application_No	INT
id_Employer	INT
idf_VisaCase	INT
id_OrganizationFlag	INT
Cert_Beg_Date	INT
Cert_End_Date	INT
Cert_Received_Date	INT
Dec_Date	INT
id_JobID	INT
NumberWorkersReq	INT
NumberWorkersCert	INT
Visa_Type	VARCHAR