|  |  |
| --- | --- |
| **Lab** | #2 |
| **Points:** | 200 |
| **Assignment Type:** | Group Submission |

# Lab #2: Data Ingestion, Processing through ETL Frameworks

## Description

In this lab, we will learn how to design and develop a basic Extract, Transform, and Load Framework. The Lab contains foundational elements for planning the ETL process. You will split up into the same groups designated for your final projects and a submission will occur as a package for each Group.

This part of the lab will require you to create a batch process using Windows Task Scheduler. The second part of the lab will force you to use a Python Script to execute stored procedures. The learning objective with this portion of the lab is to develop scripts and design a process to create and transform data from one source database to another.

### Create a simple batch job using batch process and task scheduler.

#### Overview

Using the source database, the OLTP **dav6100\_db\_2** database, we are going to create a batch process to generate a series of extracts for the dimension and fact tables to be consumed by the data warehouse database, **dav6100\_db\_2\_dw**. This set of extracted files will run on a task scheduled every day at 7:00 PM. The files will be placed in a folder to be consumed by the target database, **dav6100\_db\_2\_dw,** which is the data warehouse database. When the files are placed in the folder, a notification is sent to systematically notify consumers of those files that have been placed in that folder. For this part of the lab, we will not be consuming the files into the target database with a GET script, but we will be pushing the csv files to the folder; however, the figures below outline the entire end to end process.

A picture containing screenshot

Description automatically generated

A close up of a map

Description automatically generated

#### Part 1.1: Create a batch process to export dimension tables to folder– In Class (50 pts)

##### Resources

For this lab, you will need to have access to the following resources:

* *MySQL localhost database*
* *Python 3.7 and a Python IDE*
* *Microsoft Windows Task Scheduler[[1]](#footnote-1)*

The following are the files that are used in this lab:

|  |  |
| --- | --- |
| File name | Description |
| create-dav6100\_db\_2-database.txt | This is the DDL file to create the dav6100\_db\_2 OLTP database |
| dimension\_output\_example.sql | This file executes the sql script to generate csv files for the dimension tables |
| sql\_batch.bat | The sql batch file that will be referenced by the Task Scheduler |
| email\_example.py | The python script to send an email |
| py\_batch.bat | The python batch file that will be referenced by the Task Scheduler |

The following videos assist with executing this part of the lab:

|  |  |
| --- | --- |
| File name | Description |
| create-dav6100\_db\_2-database.txt | This is the DDL file to create the dav6100\_db\_2 OLTP database |
| dimension\_output\_example.sql | This file executes the sql script to generate csv files for the dimension tables |
| sql\_batch.bat | The sql batch file that will be referenced by the Task Scheduler |
| email\_example.py | The python script to send an email |
| py\_batch.bat | The python batch file that will be referenced by the Task Scheduler |

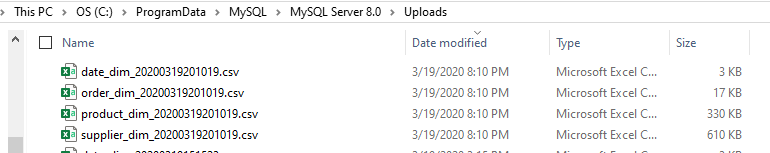
##### Steps to execute the Lab

1. **Create Database.** Use the DDL file uploaded to canvas to regenerate the database file dav6100\_db\_2. This will be the basis of the lab and will be used to generate the data extracts.
   1. Filename: create-dav6100\_db\_2-database.txt
2. **Create MySQL Script for update**. Write the database extract logic. This MySQL file will be used to generate the extracts for each dimension table and fact table in **dav\_6100\_db\_2\_dw.** 
   1. Filename: dimension\_output\_example.sql
   2. Specifications of the file are as follows:

|  |  |
| --- | --- |
| Parameter | Specification |
| Column Delimiter | || |
| Row Delimiter | \r\n |
| File Output location | C:\ProgramData\MySQL\MySQL Server 8.0\Uploads |
| File type | csv |

1. **Create Batch File for MySQL Extract Script.** Using the SQL script to generate the file extracts by referencing it in a .bat or batch file.
   1. Filename: sql\_batch.bat
2. **Test** **Batch File for MySQL Extract Script.** After you have written the batch script running it by using the Command Prompt or Terminal. You can also double click on the file to execute it. Be sure that the file was output to the correct location and populated with data. If not, you may need to alter the SQL script or the batch file.
3. **Create Emailer Script in Python.** Most batch processes are run while the user is not actively monitoring the server that executes the script. A script that sends an email to notify the appropriate parties that the script has executed is helpful to confirm that the process completed with or without errors and that it completed on time.[[2]](#footnote-2) In this case, we send an email that lets the user know that the batch process has executed using gmail. You will need to update the script with the appropriate email references and potentially install the library dependencies (e.g., SMTP)
   1. Filenames: email\_example.py, py\_batch.bat
4. **Test the Emailer Script in Python**. Like testing the batch and SQL files, you should execute the Python script to ensure that the emailer script is work as planned.
5. **Create Scheduler.** Next, using [Task Scheduler](https://www.dummies.com/computers/pcs/how-to-open-windows-task-scheduler/), you will run the sequence of scripts and have them run at 7 pm daily.
6. **Run Scheduler.** You can manually initiate the scheduler by clicking ‘Run’ and then view the results in the output folder.

**Expected output:**



Once you have completed this portion of the lab save your output files in a zipped packaged and send them to [brandon.chiazza@yu.edu](mailto:brandon.chiazza@yu.edu). You should send one per group.

Your file name should look like: <<Group\_Name>>\_Lab\_2\_Submission\_1\_1

#### Part 1.2 Use stored procedures and Python to update database, execute stored procedure on call – With your group (150 pts)

Next, you will need to use the Python files to execute a stored procedure using a Type 1 Dimension added to your DW database. The script will use the OLTP database, **dav6100\_db\_2**, as the source data to update dimension and fact tables using stored procedures in the OLAP database, **dav6100\_db\_2\_dw**. Before running the fact table stored procedure, you must run the dimension stored procedure. The Python script references the scripts to be executed and can be executed on demand as in the example above. Additionally, the emailer code could be added to the stored procedure functions and it could be entirely self-contained.

Before running the steps below, you will need to be sure that you have created the database for the datawarehouse using the **dav6100\_db\_2\_dw** schema.

| File name | Description |
| --- | --- |
| create-dav6100\_db\_2\_dw-database.txt | This is the DDL file to create the dav6100\_db\_2\_dw OLAP database |
| updateDimensionProc.txt | This file creates the Dimension tables using a stored procedure |
| updateFactProc.txt | This file creates the Fact tables using a stored procedure |
| runStoredProcedures.py | The python script connects to the database and runs the stored procedures |

You will take the following steps:

1. **Design the process.** Use a basic ETL Step Function diagram to draw out your process.
2. **Create stored procedures for updating dimensions.** With the dav6100\_db\_2\_dw, you can create stored procedures for populating the dimensions. Again, you must have created the schema for the OLAP database.
   1. Filename: updateDimensionProc.txt
3. **Create stored procedures for updating facts.** With the dav6100\_db\_2\_dw, you can create stored procedures for populating the dimensions. Again, you must have created the schema for the OLAP database.
   1. Filename: updateFactProc.txt
4. **Execute and test the stored procedures**. You can execute and test the stored procedures within MySQL itself.
5. **Create the Python script.** Using the mysql connector import library in Python, you can create connections to your MySQL instance and execute stored scripts or procedures. We will do this with the Python stored here.
   1. Filename: runStoredProcedures.py
6. **Execute the Python Script with the stored procedure.** Once you have completed the script, you can run and execute the Python script and note that the stored procedures successfully updated the dav6100\_db\_\_dw database.
7. **Edit the Stored Procedures to create a type 2 dimension as its output.** Create a set of columns on the dimensional stored procedure to provide an order status, effective date, and end date column. This will apply a Type 2 dimension to the schema.
   1. **Add three new fields.** ALTER table statements to update the dim\_orders dimension to include three new fields: order\_status (varchar(200)), eff\_date (datetime), end\_date (datatime)
   2. **Save the DDL file.** After you have updated the schema, save the file.

Once you have completed this portion of the lab save your output files in a zipped packaged and send them to [brandon.chiazza@yu.edu](mailto:brandon.chiazza@yu.edu). You should send one per group.

Your file name should look like: <<Group\_Name>>\_Lab\_2\_Submission\_1\_2.

## Grading Rubric

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Criteria | Rating | | | | Points |
|  | Exemplary (**50** pts) | Proficient (**40** pts) | Developing (**30** pts) | Not Attempted (0 pts) |  |
| Part 1.1 | Files were provided with no issues and scripts were updated to reflect relevant code updates (local to personal computer usage) | Files were provided with little or no issues and scripts were updated to reflect relevant code updates (local to personal computer usage) | Files were provided with several issues or only some of the files were provided scripts were updated to reflect relevant code updates (local to personal computer usage) | No Files were provided or was not present | **50 pts** |
|  | Exemplary (150 pts) | Proficient (120 pts) | Developing (90 pts) | Not Attempted (0 pts) |  |
| Part 1.2 | Files were provided with no issues and scripts were updated to reflect relevant code updates (local to personal computer usage) | Files were provided with little or no issues and scripts were updated to reflect relevant code updates (local to personal computer usage) | Files were provided with several issues or only some of the files were provided scripts were updated to reflect relevant code updates (local to personal computer usage) | No Files were provided or was not present | **150 pts** |

1. If you are using Mac or Linux, it is recommended that you install parallel desktops to use Windows for this lab. The same tasks can be scheduled using Linux; however, this Lab will refer to Windows-based tasks and executions. [↑](#footnote-ref-1)
2. Note in this example we use gmail. Gmail requires that you use a gmail App password. You can generate one by following the instructions [here](https://support.google.com/accounts/answer/185833?hl=en). [↑](#footnote-ref-2)