CDW_SAPP Database Operation Manual

Author: Aris Fernandez

Index

Directories and files

- Unzip aris_fernandez_de_case_study.zip into any CentOS directory.
- The structure of the contents of the zip file are as follows:

case_study	hive	branchtable.sql	
		creditcardtable.sql	
		customertable.sql	
		timetable.sql	
	not_optimized	oozie	non_optimal_coordinator.xml
			non_optimal_job_properties.properties
			non_optimal_workflow.xml
		sqoop	sqoop_imports.sh
	optimized	oozie	coordinator.xml
			job_properties.properties
			workflow.xml
		sqoop	sqoop_jobs.sh
	run	create_sqoop_jobs_optimal.sh	
		run_non_optimized_oozie.sh	
		run_optimized_oozie.sh	
	visualization	query1_hive.png	
		query1_powerbi.png	
		query2_hive.png	
		query2_python.png	
		visualization_hive_script.sql	
		query2.py	
		query1.pbix	
	db.sql		
	java_module_aris.zip		
	operation_manual.pdf		
	readme.txt		

- In the above diagram, file in each cell is located inside the directory on the cell to its immediate right. For example, ‘workflow.xml’ is located inside ‘oozie’ which is inside ‘optimized’ which is inside ‘case_study’.

JAVA Module

Setup:

- The java project is saved in a file named java_module_aris.zip
- To import the project into Eclipse, go to 'File>Import>General>Existing Projects into Workspace'. Here, click the 'select archive file' radio button and then browse to the folder created by unzipping the file above and click finish.
- A Java Project named Aris_Fernandez_Case_Study should now be among your eclipse projects.

Project Structure:

- The project consists of 5 packages inside the src folder. The packages are as follows:
  - dataAccessLayer
    - This package contains the classes and interfaces necessary for connecting to the database and acquiring data.
  - exceptions
    - This package contains a class with definitions for custom exceptions and also a class for implementing those exceptions (user input validation).
  - presentationLayer
    - This class contains classes for defining the on-screen presentation of the whole JAVA module. It contains the Main class that should be executed to run the whole module, as well as two other classes that can separately run the customer and transaction modules, if so is desired. These two classes have the suffix 'Main' to differentiate them.
  - resources
    - This class contains extra resources and utilities needed to run the whole module.

- transferObjects
    - This class defines the objects to be transferred to adhere to the DAO design patter. It consists of a Transaction class, a Customer class, as well as a class to represent several transactions at the same time.

- The project also contains an extra folder named 'Output' which contains the files created when information is saved to CSV files. Inside this folder there are two subfolders, one of which is named 'CustomersModule'. This folder contains four subfolders, each for one of the submodules of the customer's module. The Output folder also contains another folder named 'TransactionsModule' which contains three subfolders, one corresponding to each of its submodules.
- Two more files can also be found inside the project. One files is named 'DataBaseConfig.ini' which contains configuration properties for connecting to the MySQL database. The other file is named 'mysql-connector-java-8.0.15.jar' and this file is the JDBC connector used to connect to the database.

Big Data Module

- **Quick operation:**

  - For the ***non-optimized*** modules (**modules 2.2.1, 2.2.2, 2.2.3**), open a terminal and run the 'run_non_optimized_oozie.sh' script. The command to run the script is '***bash <path_to_case_study>/case_study/run/run_non_optimized_oozie.sh***'. Please replace '<path_to_case_study>' with the appropriate path in which you unzipped the files.
  - This script uploads all necessary files to HDFS and runs the appropriate oozie coordinator. It outputs to screen the corresponding job ID.
  - For the ***optimization*** module (**module 2.2.4**) first run the 'create_sqoop_jobs_optimal.sh' script with the command '***bash <path>/case_study/run/create_sqoop_jobs_optimal.sh***'. This script will prepare all the sqoop jobs needed for the optimized module. There is no need to open the meta-store since this script will open the meta-store before creating the jobs. This script removes the tables created during the un-optimized module, to prevent redundant data.
  - Next, run the script named 'run_optimized_oozie.sh' with '***bash <path>/case_study/run/run_optimized_oozie.sh***'. This file runs the sqoop jobs created in the stop above, and therefore that script must be run before this one.
  - The script uploads any needed file to HDFS before running the oozie coordinator. It then outputs to screen the corresponding oozie job ID. As before, there is no need to manually open the meta-store

- **Detailed Operation:**

2.2.1 Data Extraction and Transformation Module

- A script containing the sqoop import commands can be found at '<path>/case_study/not_optimized/sqoop/sqoop_imports.sh'
- This script can be run in a terminal to manually perform the sqoop imports. It is not necessary to do this since the oozie coordinator does it automatically bellow.

2.2.2 Data Loading Module

- Scripts containing the hive queries can be found in '<path>/case_study/hive/'
- This directory contains the following files: 'branchtable.sql', 'creditcardtable.sql', 'customertable.sql', 'timetable.sql'.

2.2.3 Process Automation Module

- In HDFS, create a folder named 'case_study' inside the root directory. Inside this folder, create a new folder named 'not_optimized', and inside this one create another folder named 'oozie'. The final structure should be '/case_study/not_optimized/oozie'
- Upload the contents of '<path>/case_study/not_optimized/oozie' in Linux to the path created above (***hdfs dfs -put -f <path>/case_study/not_optimized/oozie /case_study/not_optimized***)
- The step above should have uploaded 'non_optimal_coordinator.xml' and 'non_optimal_workflow.xml' to the specified folder in HDFS.
- The hive scripts needed to load data into hive can be found in the folder '<path>/case_study/hive'. They must be uploaded to '/case_study/hive' in HDFS. To do this, use the command (***hdfs dfs -put -f <path>/case_study/hive /case_study***)
- Finally, run the following command: '***oozie job -oozie http://localhost:11000/oozie -config <path>/case_study/not_optimized/oozie/non_optimal_job_properties.proper***

ties -run’. This will prepare the oozie coordinator. Please remember to change ‘<path>’ accordingly.

- **2.2.4 Process Optimization Module**
  - This part is similar to the non-optimized module above. The only difference is the source and destination of oozie files. Files from ‘<path>/case_study/optimized/oozie’ must be uploaded to ‘/case_study/optimized/oozie’ in HDFS. (***hdfs dfs -put -f <path>/case_study/optimized/oozie /case_study/optimized***)
  - The previous command should have uploaded the files named ‘workflow.xml’ and ‘coordinator.xml’.
  - This module uses the same hive scripts as the previous module with the differentiation between modules made with different input parameters in oozie, therefore they must be uploaded to the same place. If the hive files were uploaded for the non-optimized modules, there is no need to upload them again.
  - One important difference between this optimized module and the above module is that it uses sqoop jobs in order to import only new data. Because of this, before submitting the oozie job, the sqoop jobs must be first created.
  - The sqoop job creation script is located in ‘<path>/case_study/optimized/sqoop’. This file can be used to create the sqoop jobs one by one by copying and pasting each job in a command line terminal or the file can also be run as a bash script with the command ‘***bash <path>/case_study/optimized/sqoop/sqoop_jobs.sh’***.
  - Regardless of which option is used to create the sqoop jobs, please be advised that before creating the jobs, the sqoop meta store must be running. This can be done by running the command ‘***sqoop-metastore***’, but this command must be run using a different command line since it binds the standard input until the server is shut down. To shut down the meta-store, if needed, please do it with the command ‘***sqoop-metastore –shutdown’***

- To complete the process, type the command '***oozie job -oozie http://localhost:11000/oozie -config <path>/case_study/optimized/oozie/ job_properties.properties -run***' on a Linux terminal. Make sure that the sqoop meta store is open while oozie runs, since the sqoop jobs depend of the meta-store.

2.2.5 Data Visualization

- Pictures for the visualization module can be found inside the directory '<path>/case_study/visualization'.
- The folder contains four images. Two of the images were created using the hive visualization tools. These images have the '_hive' suffix attached to the filename. Hive provides very rough visualization tools, thus the other two images included in the folder were created using other tools. One image was made using Matplotlib in Python (made with the script 'query2.py', please see the miscellaneous section of this manual) and the other was created using MS PowerBI Desktop (made with the powerBI file named 'query1.pbix'). Each image has a corresponding suffix.

Troubleshooting

- Script to run optimized module fails: Sometimes when running the script named 'run_optimized_oozie.sh' right after running the script to create the sqoop jobs the running script fails. This problem can usually be resolved by rerunning the script. This is usually caused by the script failing to open meta store.
- Oozie sqoop nodes fail due to missing JAR file: The sqoop jobs running inside oozie workflows need a jar file with definitions on how to handle JSON files. If this error happens please upload the file named 'java-json.jar' into HDFS. This file must be uploaded to the path 'user/oozie/share/lib/lib_XXX/sqoop' inside HDFS. The 'XXX' part of the 'lib_XXX' folder is a system dependent number. While setting up this project the value of XXX was '20161025075203'.
- Oozie sqoop node fail with launcher error: The MySQL queries run by sqoop depend on an updated database different from the default database in the HDP virtual machine. Place make sure that the proper database is installed in MySQL. The database can be installed from the file named 'db.sql' included with the project.
- The java module fails to connect to the database: Same as above, the java module depends on an updated MySQL database. Please install the appropriate database.

Miscellaneous

To recreate the visualization of query number 2 using python, it is first necessary to have the libraries numpy, pandas, and matplotlib installed. It is also necessary to have a Hive ODBC driver in Windows set up with the name ‘hive’. Lastly, the pyodbc python library is also needed in order to allow python to connect to the ODBC driver.