REAL TIME DATA WAREHOUSING (RT-DW) CHAIN

Configuration Manual

Abstract

Documentation on how to set up and configure the Real Time Data Warehousing (RT-DW) chain.

Ian Bassett

X06709711

MSc. Masters in Data Analytics

National College of Ireland

School of Computing

1. Introduction

The following configuration manual documents the implementation of the environment to conduct Real Time Data Warehousing through the RT-DW Chain. The layout of this manual begins with "Environment Specification" examining the environments in which the experiments were carried out. The next two sections, "Overview of Technology" and "Python Packages and Modules" inspect the tools that were used to produce the results shown in the report. The document then outlines how each environment is set up, starting with the VMWare Workstation Player with Ubuntu 14.04.LTS and Windows 10; and continues with the implementation of Apache Spark, SQL Server, Python and Salesforce.

The next area focuses on data preparation for Amazon reviews and Cosmos tweets Data sources. The development of the RT-DW chain is illustrated and how the approach from near time to real time was achieved.

2. Environment Specification

Two environments were used in the analysing big data performance:

2.1 Laptop Specification

Laptop: Dell Inspiron 15 3000 series

Windows Version: Windows 8.1

Processor: Intel(R) Core(TM) i5-4210U CPU @ 1.70GHz

Installed Memory (RAM): 8.00GB

System type: 64-bit operating system, x64 based processor

2.2 VMware Environments

2.2.1 Ubuntu Environment

Memory: 4096MB, 4GB

Processor: 1

Hard Disk (SCSI): 80GB

CD/DVD (SATA): ISO file, Ubuntu 14.04 LTS

Network Adapter: NAT

VM WorkStation: WorkStation 11.0 virtual machine

2.2.2 Windows Environment

Memory: 4096MB, 4GB

Processor: 1

Hard Disk (SCSI): 80GB

CD/DVD (SATA): ISO file, Windows 10

Network Adapter: NAT

VM Workstation: WorkStation 11.0 virtual machine

3. Overview of Technologies:

3.1 VMware Environments

3.1.1 Ubuntu 14.04 LTS

Summary:

Ubuntu is an open source Debian-based Linux operating system and distribution for personal computers, network servers and smart phones. It uses graphical shell, Unity as its Graphical User Interface (GUI).

Use:

- Used as the first environment to analyse big data performance between SQL Server and Hive on Spark.
- Macro Environment to implement the Real Time Data Warehousing (RT_DW) chain on a Linux operating System

3.1.2 Windows 10

Summary:

Windows is a Graphical User Interface (GUI) operating system by Microsoft. It replaces MS-DOS to eliminate the need of using command lines. Windows 10 introduces a new GUI design and introduces Cortana, a personal assistant to assist with user navigation.

Use:

- Used as the second environment to analyse big data performance between SQL Server and Hive on Spark.
- Macro Environment to implement the Real Time Data Warehousing (RT_DW) chain on a Windows operating System

3.2 Apache Spark 1.5.1

Summary:

Apache Spark is a general purpose cluster computing system with an in-memory data processing engine with a range of API's which allows data workers to efficiently execute graph processing with GraphX, MLlib for Machine Learning, Spark SQL for SQL, Spark Streaming with integration with Java, Scala, R, Python and Hive.

Currently it runs programs 100x times faster in memory and 10x from disk than Hadoop MapReduce. Spark can be run on Hadoop, Mesos and standalone or on the cloud with the ability to access diverse data sources including HDFS, HBase, S3 and Cassandra.

Use:

- Combined with Hive to measure Big Data performance between Hive on Spark and SQL Server.
- Micro Environment to implement the Real Time Data Warehousing (RT-DW) chain on a Linux operating System

3.3 Hive

Summary:

A component of Hortonworks Data Platform (HDF), Hive provides Hive Query Language (HQL) statements, similar to Structured Query Language (SQL) Statements, as an interface to leverage Data Warehouses for the Hadoop and Spark Platform.

Hive can be used through the command line interface, Java Database Connectivity (JDBC) and Open Source Database Connectivity (ODBC) applications to access and use the Hive JDBC/ODC drivers, also called the Hive Thrift Client. The Hive Thrift Client can communicate with applications written in C++, Python, Java, Ruby and PHP.

Use:

- Destination to store data for Benchmarking and Sentiment Analysis.
- Combined with Apache Spark to measure Big Data performance between Hive on Spark and SQL Server.
- Micro Environment to implement the Real Time Data Warehousing (RT-DW) chain on a Linux operating System

3.4 SQL Server

Summary:

SQL Server is a Relational Database Management System (RDBMS) designed for the enterprise environment that allows interaction through client machines. Data is stored and managed through ANSI SQL and T-SQL statements.

Use:

- Destination to store data for Benchmarking and Sentiment Analysis.
- To measure Big Data performance between Hive on Spark and SQL Server.
- Micro Environment to implement the Real Time Data Warehousing (RT-DW) chain on a Windows operating System

3.5 Python

Summary:

Python is an open source, object-oriented language that can be used on Windows and UNIX operating systems. Python is used for a variety of purposes and is heavily relied upon with Scientific Computing for Data Mining, Data Visualisation, and Database Integration among other functions.

The syntax is designed for universal understanding and readability; this is by Python's strict punctuation rules that remove curly braces from code. Python has been around for over twenty years and has a diverse range of libraries that can be utilized by its large supporting community of contributors.

Use:

- Functionality of the RT-DW chain in terms of benchmarking, cube analysis, sentiment analysis and Salesforce integration.
- Operator to start and direct the RT-DW chain process from beginning to end.
- Communicator between directories.

3.6 Salesforce

Summary:

Salesforce is a cloud computing social enterprise Software as a Service (SaaS) provider. Salesforce specialises in Customer Relational Management (CRM) systems which consists of a combination of cloud and application platforms such as, Marketing Cloud, Sales Cloud and Service Cloud.

Salesforce also provides Product as a Service (PaaS) products to create applications that can be integrated with Salesforce and Rypple, a social Human Resource (HR) performance management system.

Use:

- Gathers results from cube analysis and implements results into a Salesforce custom object.

3.7 Plotly

Summary:

Plotly is an online analytics and data visualisation tool that provides online graphing, stats and analytics with API libraries for Python, R, Perl, MathLab and Rest. It offers rich interactivity and web share ability unlike Ggplot, Matplotlib and MathLab. Visualisations are presented offline in html format, while online can be shared among colleagues and saved to the cloud.

Visualisations supported by Plotly are line and scatter plots, bar charts, bubble charts, 3D plots, streaming API among others.

Use:

- Presents results from the Time python module and visualises the difference in time between different data loads.

4. Python Packages and Modules:

4.1 Pypyodbc

Summary:

Associated with the python module Pyodbc, Pypyodbc is a 2.X and 3.X python module that allows the user to use to ODBC (Open Database Connectivity) to connect to almost any database from Windows, OS/X and Linux. Pypyodbc differs as a pure python module with no compilation and is cross platform with C Python, Iron Python among others.

Use:

- To query and connect to the SQL Server data warehouse.
- Executing SQL commands to bulk load data from csv files into SQL Server.

4.2 PyHive

Summary:

The PyHive module is an interface that connects Python to Hive and Presto. To interact with the Hive Data Warehouses, Python libraries *sasl, thrift* and *thrift-sasl* are required.

Use:

- To query and connect to the Hive on Spark data warehouse.
- Executing HQL commands to bulk load data from csv files into Hive.

4.3 Psutil

Summary:

Python system and process utilities (Psutil) is a cross platform library that monitors running system processes and retrieves information on system utilization in terms of CPU, memory, disk and network performance. Psutil is also used for profiling and the management of limiting and running processes. It is supported on Windows, Linux, OSX and Sun Solaris among others with 32-bit and 64-bit architecture with Python version from 2.6 to 3.5.

Use:

- To measure CPU, Disk and Memory performance when benchmarking between the Hive on Spark and SQL Server Data Warehouses.

4.4 Time

Summary:

This module is an umbrella of time-related functions. Time.clock () is the most suitable function for benchmarking Python and timing algorithms. It returns the current processor time as a floating point number on a UNIX operating system, where on Windows it returns wall-clock seconds.

Use:

- In terms of benchmarking, measures and documents time between different tables as they are loaded into different data warehouses.

4.5 Cubes

Summary:

The Cubes module is a light-weight Python Framework and OLAP HTTP Server. It specialises in aggregate browsing multi-dimensionally modelled data through slicing dimensions, drilldown hierarchies, retrieving dimension values or facts and provides Metadata for reporting applications.

Use:

- Access and query SQL Server and Hive on Spark data warehouses to perform drilldowns and calculate measures.
- The cube analysis results are collected and written into Salesforce.

4.6 CSV

Summary:

The cross separated value (CSV) package is used to implement classes to read and write tabular data into CSV format. Utilising the dictionary aspect of the package, DictReader and DictWriter create objects from the CSV data; and combined with the fieldname parameters; create dictionaries, with the key as the fieldname and the segmented CSV data as a value. This allows for data to be normalised correctly for benchmarking.

Use:

- To use Dict Reader to read CSV files into Python for Data manipulation and optimization.
- To use Dict Writer to write transformed dictionaries in Python to a CSV file, in preparation for benchmarking.

4.7 Simple Salesforce

Summary:

Simple Salesforce is a REST API client to Salesforce.com. It interacts with the API and returns an ordered dictionary of the API JSON response. This allows files to be benchmarked through a row by row import.

Use:

- To interact with the Salesforce platform and insert cube analysis from Python into a salesforce custom object.

4.8 Shutil

Summary:

The python Module Shutil offers high level operations in the moving and removal of files. It also allows users to copy and delete directories.

Use:

- Shutil is used to move files in the RT-DW chain from the beginning through to the end after the file has been processed through a python script.

4.9 OS

Summary:

The Operating System (OS) module is the baseline that provides dependent functionality on a range of operating systems. This allows operating systems to be leveraged in gathering information and working with processes. OS specialises in Process Parameters, File Object Creation, File Descriptor Operations, Process Management, Miscellaneous System Information and Miscellaneous Functions.

Use:

- On Windows, the chain starter activates Python Scripts based on OS calls.
- OS.listdir returns a list of entities in the directory from the path name; this combined with a For Loop allows all files to be accessed in Python.

4.10 Urllib

Summary:

Urllib module is used to fetch data across the World Wide Web. Using *Urllib.urlopen ()* users gather and open network objects based on URL, files and opened sockets to a server to utilise resources.

Use:

Used to connect a CSV file to http://text-processing.com/api/sentiment/. This carries out sentiment analysis on Amazon Reviews and Cosmos Tweets to determine positive and negative opinion.

4.11 SubProcess

Summary:

SubProcess allows the creation of new processes, gathers return codes and connects to pipes in terms on input/output and errors. Using the call function, it allows commands from other python scripts to be executed.

Use:

- On Linux, the chain starter activates Python Scripts based on SubProcess calls.

4.12 Pandas

Summary:

Pandas is an open source, BSC library that provides Data Analysis tools for Python. In terms of data preparation, Pandas offers the ability to combine multiple data frames for Denormalization through inner, outer, left and right joins. The *Numpy* module is required to use Pandas.

Use:

- In preparing to denormalise the Amazon datasets, Pandas are used to create data frames based on matching values between reviews and Metadata. Data is then saved in CSV format.

4.13 Tempfile

Summary:

Supported on all platforms, Tempfile generates temporary files and directories. Using the Named Temporary file () function allows users to have secure temporary files that appear on the file system and can be retrieved from the named attribute. Files being re-opened are dependent on operating systems. They can be re-opened on Unix/Linux but not on Windows NT or Later. In the Named Temporary File function, if delete is set to true; the file will be deleted after the python file is closed.

Use:

- On Ubuntu, creates temp files when normalising data in Python.

4.14 Ast

Summary:

The Abstract Syntax Trees (Ast) Module assists Python in processing trees with regards to Python abstract syntax grammar. Using ast.literal_eval () ast evaluates nodes, Unicode and Latin-1 encoded strings to determine whether nodes and strings are one of Python's literal structures. Being either string, numbers, tuples, Booleans, dicts, lists or none.

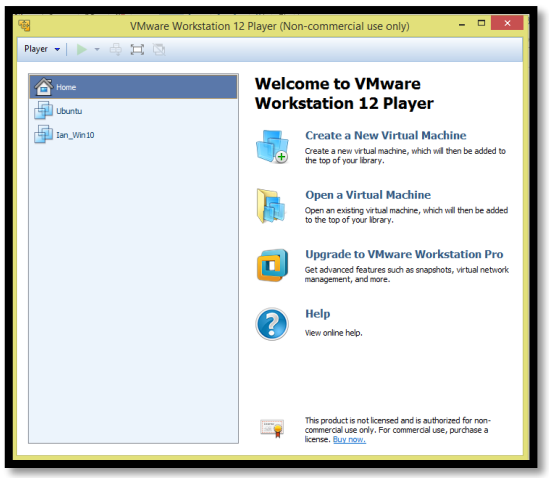
Use:

- Evaluates Amazon Meta data to convert a string into a dictionary. This leads to the conversion into CSV data.

5. VMware Workstation Player:

5.1 VMware installation

1. Open the web browser and go to the VMware downloads page https://my.vmware.com/web/vmware/downloads.
2. Under desktop & end-user computing, look for VMware workstation player and select *download product*.
3. Select the latest version and select the VMware workstation player for *Windows 64-bit operating systems* and click *download*. The VMware workstation player will automatically download.
4. Execute the setup file to be directed to the VMware workstation setup wizard. Navigate the wizard accepting the license agreement, declare installation location, define user experience settings and establish shortcuts. Click *finish* or *upgrade*, depending on whether you have a VMware workstation already installed. A reboot may also be required for the changes to take effect.
5. Sign up an email to VMware to get access to the workstation for free. Installation is complete.

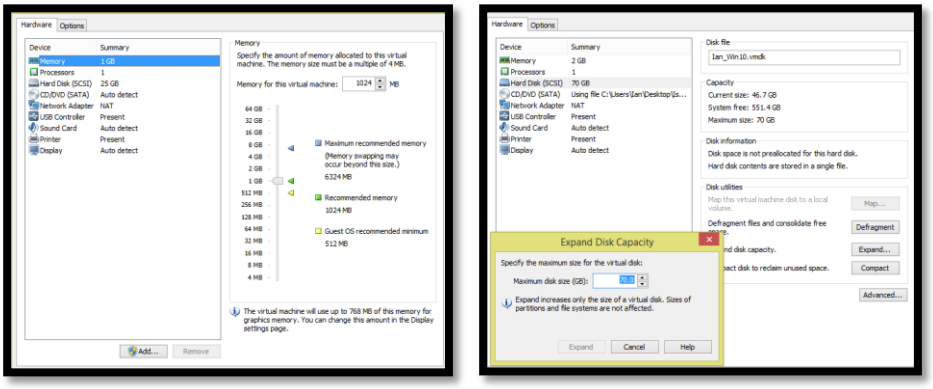


5.1.1 Interface of the VMware Workstation after installation.

5.2 Ubuntu Installation

1. Open the web browser, open the URL and download *Ubuntu version 14.04 LTS recommended 64 bit version* at: http://www.ubuntu.com/download/desktop
2. Click *download*. The Ubuntu version 14.04 LTS ISO file will begin to download. Download speed will vary depending on download performance.
3. Open the VMware workstation, right click on home and select *'Create a new VM.'*

4. The Virtual Machine Wizard will request where to install the virtual machine from. Browse installer disk image file (ISO) and choose the Ubuntu ISO file.
5. Easy install will ask for the virtual machine names, username and password confirmation.
6. Name the Virtual Machine and where it is located on the C:\\ Drive.
7. When specifying the disk capacity, VMware will recommend how much disk capacity should be used. Add an extra 20 GB to the recommended amount and store to virtual disk as a single file to improve performance with very large disks.
8. Easy Install will direct the user to download VMware tools to assist in maintaining the performance of the virtual machine. Download the updates.
9. Easy Install will process the files with Ubuntu being installed and operating on the VMware workstation
10. Return to the VMware workstation main menu, right click on the virtual machine and click *settings*.
11. In Hardware select the green icon on recommended memory to ensure the virtual machine will work and prevent the laptop from crashing.
12. (Optional) To expand virtual drive, click on *Hard Disk (SCSI)* and on the right side under disk utilities select *expand* to adjust the disk size in GB.

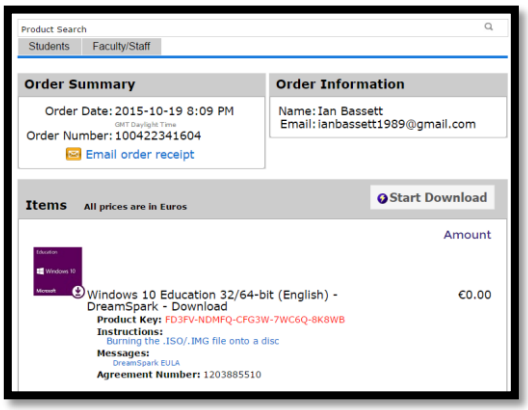


5.2.1 on the left, adjusting memory to the recommended settings. On the right, expanding the virtual disk.

5.3 Windows Installation

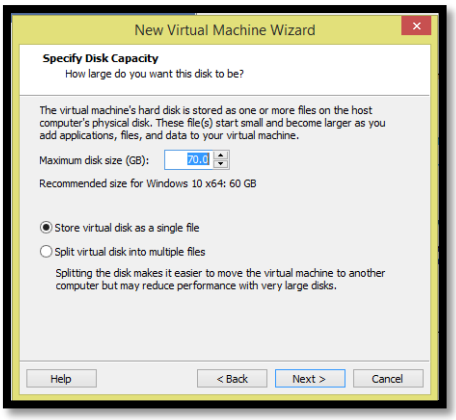
1. Access the URL, search for college intuition National College of Ireland and click *'Visit the Webstore'* to access the webstore at:
   https://www.dreamspark.com/Institution/Access.aspx
2. By clicking *search*, the web browser will direct to the National College of Ireland's webstore. To access enter campus credentials.

3. Entering the site, a number of products are available to download. Select *Windows 10* to get more information.
4. Under Windows EDU, select *'Windows 10 Education 32-64-bit (English) – DreamSpark'*, choose 64-bit version and click *'add to cart'*.
5. This will direct to the shopping cart, with the option to continue shopping or direct to checkout.
6. Click *"Checkout"* and a Microsoft Dream Spark direct subscription agreement, read and click *"I accept."*
7. An overview of the products being checked are shown with contact information. This contact information will be used to send a receipt through an email to the user on what was purchased. Click *'proceed with order'*.
8. In the order summary page in the right hand corner click *'Start Download'*. Download speed will vary depending on download performance.



5.3.1 Windows 10 product details from the National College of Ireland Webstore.

9. Open the VMware workstation, right click on home and select *'Create a new VM.*'
10. The Virtual Machine Wizard will request where to install the virtual machine from. Select the Windows 10 ISO file.
11. Choose the guest operating system as Microsoft Windows and selecting *version 10 x64*.
12. Select the Virtual Machine Name and where it will be located on the machine.
13. When specifying the disk capacity, VMware will recommend how much disk capacity should be used. Add an extra 20 GB to the recommended amount and store to virtual disk as a single file to improve performance with very large disks.



5.3.2 Specifying the Virtual Machine Size at 70.0 GB and storing the virtual machine as a single file.

14. Click next and receive and overview on the virtual machine that will be created. Click *Finish.*
15. In Hardware select the green icon on recommended memory to ensure the virtual machine will work and prevent the laptop from crashing.
16. (Optional) To expand virtual drive, click on *Hard Disk (SCSI)* and on the right side under "Disk Utilities" select *expand* to adjust the disk size in GB.
17. Start the Windows 10 Virtual Machine and check for updates. Once the operating system is updated, select the language to install, the time and currency format; and keyboard or input method.

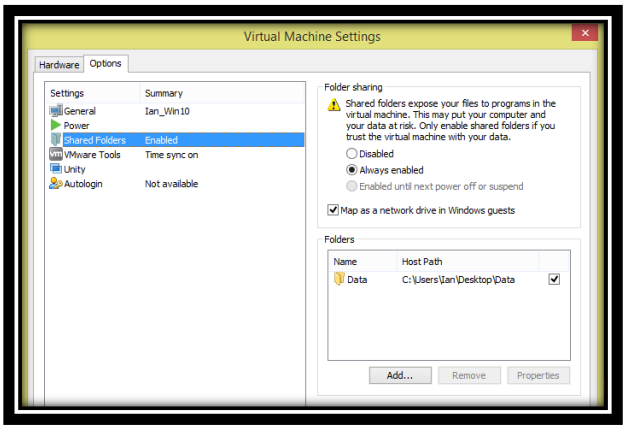
5.3.3 Windows 10 setup screen asking the user for their preferences before installation.

18. Enter the product key from Dream Spark and accept the license agreement to continue the installation. The wizard will process the files with Windows 10 being installed and operating on the VMware workstation.

5.4 Shared Folders

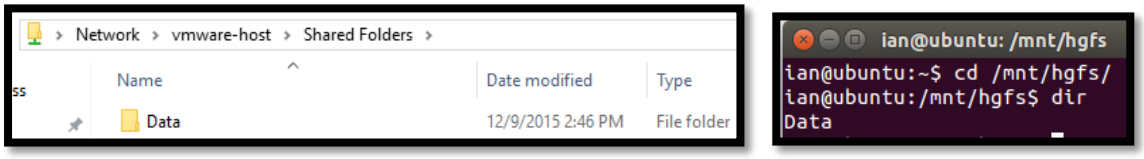
1. To connect host and guest operating systems folders together, go to a workstation and right click on *settings*.

2. In settings switch from *hardware* to *options* and select *shared folders*. The Operating system will present different options to connect the folders.



5.4.1 The virtual machine settings to enable shared folders.

3. Under folder sharing, select *always enabled* to allow communication between both operating systems. To successfully share on a windows platform tick 'Map as a Network Drive in Windows Guests'
4. To add a folder click *add*; this will launch a shared folder wizard. The wizard is comprised of two stages. The first stage will ask for the host path to the folder and the name of this path. The second stage will specify the shared folder attributes and permissions, asking the user if they would like to enable the folder to be shared and whether it is read-only.
5. To determine if folders are shared successfully, access the Windows 10 file explorer and click on *network* and click on *vmware-host*. The share folders directory will contain all folders shared between operating systems. On a Linux operating system, open up a terminal and change the directory to */mnt/hgfs*. Type *dir* to list all the shared folders between the operating systems.
6. In case shared folders do not appear on a Linux operating system, the reasoning behind this is that an old version of VMware tools with non-functioning vmhgfs module kernel is installed. To solve this error, type into the terminal *sudo vmware-config-tools.pl -d --clobber-kernel-modules=vmhgfs*. This will run the configuration with clobber kernel modules setting, overwriting the existing vmhgfs module



5.4.2 'Data' Folder connected to Windows (left) and Ubuntu (right)

6. Apache Spark

6.1 Download and Extraction

1. In the VMware workstation, open the web browser and download Spark 1.5.1 release. http://spark.apache.org/downloads.html
2. Choose the recommended mirror site to download.
3. The Archive Manger will open the tar.gz folder. Extract spark.1.5.1 to the home directory.
4. To build Spark.1.5.1 with Maven, two requirements must be met. Ubuntu must have Java +7 and Maven 3.3.3

6.2 Java installation and Java Home

1. Access the terminal and add a repository for java with *sudo apt-add-repository ppa:webup8team/java*.
2. Once complete, update sources with *sudo apt-get update.*
3. To install Java7, type into the command terminal *sudo apt-get install oracle-java7-installer*
4. To confirm java installation, type into the command *java –version.*

```
ian1989@ubuntu:~$ java -version
java version "1.7.0_80"
Java(TM) SE Runtime Environment (build 1.7.0_80-b15)
Java HotSpot(TM) 64-Bit Server VM (build 24.80-b11, mixed mode)
```

6.2.1 Java successfully installed on Ubuntu.

5. All applications that use java function correctly through Java Home. To set up Java home, open a new terminal and access the root through *sudo su*.
6. To discover where java is located on the Ubuntu machine, type *which java*.
7. To set the Java home globally edit bash.bashrc file. Type *gedit .bashrc*
8. Add the following lines at the end of the bashrc in figure 5.1.2
9. Restart the terminal and type *echo $JAVA_HOME* to get the new path.

```
JAVA_HOME=/usr/lib/jvm/java-7-oracle
export JAVA_HOME
PATH=$PATH:$JAVA_HOME
export PATH
```

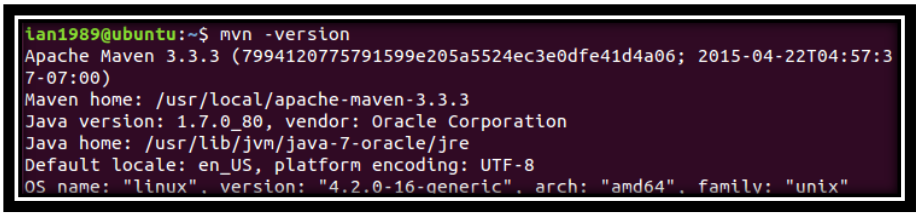
6.2.2 Declaring the Java Home

6.3 Maven Installation

1. To download Maven 3.3.3 binary from repository, use the terminal and type:

   wget http://mirrors.sonic.net/apache/maven/maven-3/3.3.3/binaries/apache-maven-3.3.3-bin.tar.gz

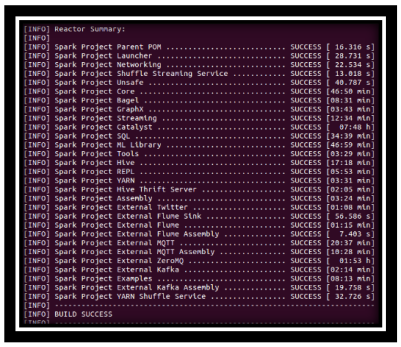
2. To Unzip the tar folder, type *tar -zxf apache-maven-3.3.3-bin.tar.gz*
3. Move the application directory to usr/local. *sudo cp -R apache-maven-3.3.3 /usr/local*
4. To make a soft link in usr/bin for universal access to mvn, type: *sudo ln -s /usr/local/apache-maven-3.3.3/bin/mvn /usr/bin/mvn*
5. Type *mvn –version* to check if Maven installed successfully. If the command fails to execute, the reason is that the directory path to Java Home is not set correctly. Adjust the Java Home directory path to execute successfully.



6.3.1 Maven successfully installed on Ubuntu.

6.4 Building Apache Spark

1. To build Apache Spark with hive integration, change to the spark directory with *cd spark-1.5.1*.
2. To ensure a successful build, configure Maven to use more memory by setting MAVEN_OPTS. Type: *export MAVEN_OPTS="-Xmx2g -XX:MaxPermSize=512M -XX:ReservedCodeCacheSize=512m"*
3. To build and enable hive integration with JDBC server and CLI in Spark SQL, In the terminal, access the spark directory and build with: *mvn -Pyarn -Phadoop-2.4 -Dhadoop.version=2.4.0 -Phive -Phive-thriftserver -DskipTests clean package.* Build time will be determined by memory, disk and CPU performance.



6.4.1 Apache Spark successful build

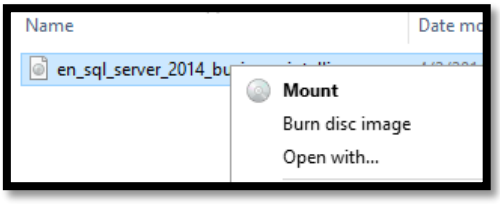
7. SQL Server

7.1 Download SQL Server from DreamSpark

1. Access the VMware Workstation Player, log into Windows 10 and access the web browser.
2. Navigate to the URL, search for college intuition National College of Ireland and click 'Visit the Webstore' to access the webstore at: https://www.dreamspark.com/Institution/Access.aspx
3. By clicking search, the web browser will direct to the National College of Ireland's webstore. To access, enter "Campus Credentials."
4. Search for SQL Server 2014 Business Intelligence 32/64-bit (English) DreamSpark Download and select it for download.
5. This will direct to the shopping cart, with the option to continue shopping or direct to checkout.
6. Click Checkout and a Microsoft Dream Spark direct subscription agreement, read and click "I accept."
7. An overview of the products being checked out are shown with contact information. This contact information will be used to send a receipt through an email to the user on what was purchased. Click 'proceed with order'.
8. In the order summary page in the right hand corner, click 'Start Download'. Download speed will vary depending on download performance.

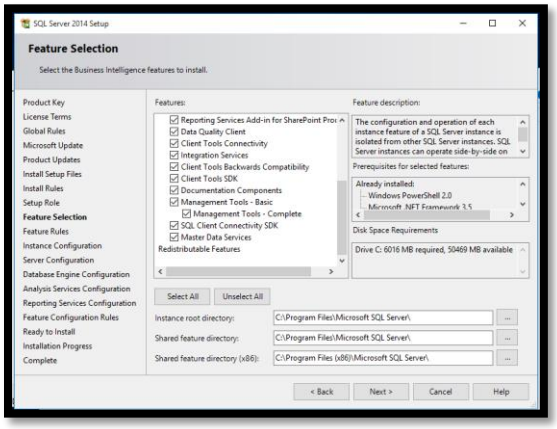
7.2 SQL Server 2014 Installation

1. Install the .net framework 3.5 SP 1: http://www.microsoft.com/en-us/download/details.aspx?id=22
2. Access SQL Server folder that has been downloaded from DreamSpark and right click on the SQL Server ISO file to mount.



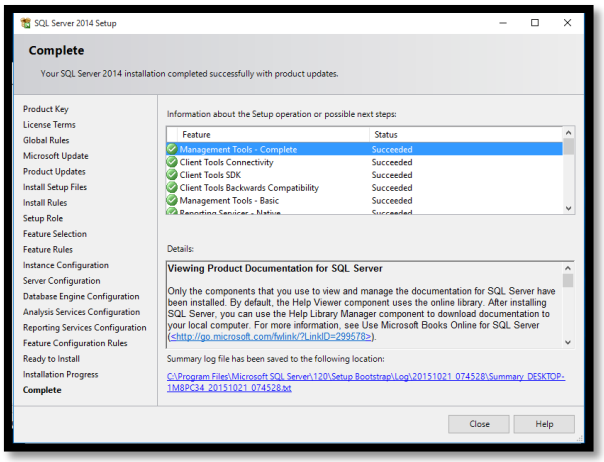
7.2.1 Mount the SQL Server ISO file in Windows 10 Virtual Machine

3. This directs to the SQL Server Installation Center, Click *New SQL Server stand-alone installation or add features to an existing installation*.
4. Navigate through the setup wizard, entering the product key to install setup files.
5. In Features Selection, Click *Select Install* and *next >.*



7.2.2. Enabling all features in SQL Server.

6. In Instance Configuration, select Default Instance and click *next >.*
7. In Server Configuration, accept the recommended settings and click *next >*.
8. In Database Engine Configuration, Under Server Configuration set the authentication mode to *windows authentication mode*. When specifying SQL Server Administrator, select *Add Current User* and click *next >*.
9. In Analysis Service Configuration, under Server Configuration set the server mode as *Multidimensional and Data Mining Mode*. To specify what users have administrative permissions to analysis services, click *Add Current User*. Continue by pressing *next >*.
10. In Reporting Services Configuration, under Reporting Services Native Mode, select *install and configure*. Under Reporting Services SharePoint Integrated Mode select *install only*. Click *next >*.
11. In Ready to install, review what will be installed and click *install* to begin Installation.



7.2.3 SQL Server successful Installation.

8. Python

8.1 Installation

8.1.1 Windows:

1. Open the web browser and access the following URL: https://www.python.org/downloads/
2. Under a specific release, select *Python 2.7.10*.
3. Under Files, select *Windows x86-64 MSI installer*.
4. Execute *Windows x86-64 MSI installer.exe,* this will direct towards Python's Setup Wizard.
5. When asked to set up Python for all users, select *Install for all users* and select *next >*.
6. Under Select Destination Directory, choose *C:\Python27* for the files. Press *next >*.
7. Under Customize Python 2.7.10, accept default settings and click *next >.* Python wizard will begin and be installed.

8.1.1 Selecting a destination directory in Python

8.1.2 Linux:

1. Python is installed while building Apache Spark-1.5.1

8.2 Python Package and Module Import

The Python Package Index is a repository of software for the Python programming language. Packages are directories of python modules, while modules are source files containing classes, functions and global variables. To install modules and packages, pip is used.

8.2.1 Windows:

1. Pip is supplied with python version 2.7.10, to install packages and modules open the *control panel*.
2. Change the directory based on where Python is located; in this case, *cd C:\Python27*.

3. In Python's Directory cd directory to scripts, type *cd scripts*.
4. To start pip, insert into the command terminal *pip2.7.exe*.
5. To install a package or module, type "pip install" (package or module) example: *pip install pyhive*. The module or package will be collected and installed into Python's libraries.

8.2.2 Linux:

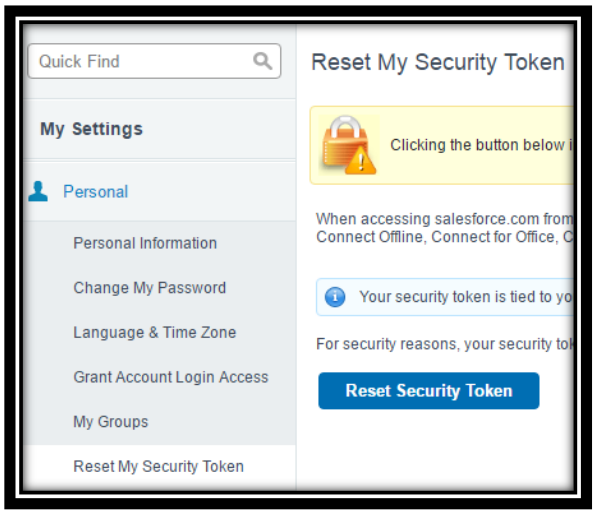
1. To install Pip on Linux open the terminal and type *sudo apt-get install python-pip*.
2. To install a python package or module type: *pip install pyhive*.

9. Salesforce Environment:

After gaining results from the Data Warehouse Cube Analysis, results will be entered into Salesforce Custom Object. This will be a storage for results and shows compatibility of the Data Warehousing Chain into other aspects of Data Analytics.

9.1 Salesforce Setup:

1. Set up a Salesforce developers account by clicking on the following link and fill out the form. https://developer.salesforce.com/signup
2. To access Salesforce developer edition, sign in here. https://login.salesforce.com/
3. To get the security token to implement into the Salesforce Python Script, go to your *name* on the top right corner and click *My Settings*.
4. On the left hand side go to *Personal* and *Click Reset my Security Token*
5. This will open the Security Token section of *Personal*, Select *Reset Security Token*.
6. Go to the email address that was created with Salesforce to gather the security token.



9.1.1 Resetting the Security Token in Salesforce

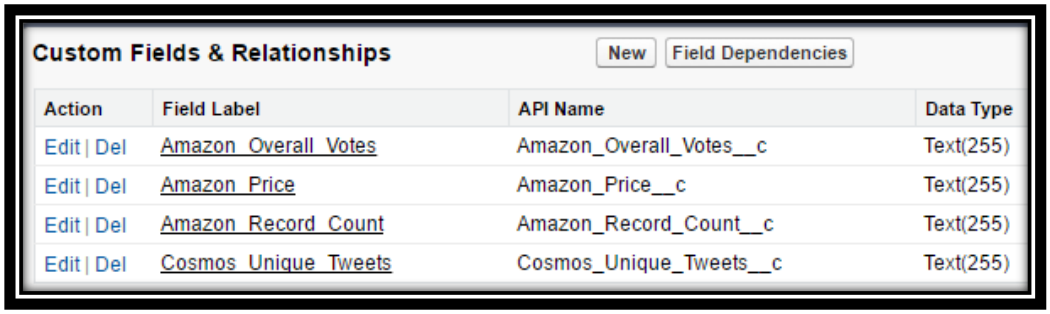
9.2 Custom Object Setup

1. To create a new object, go to the top right corner of Salesforce and click *setup*.
2. On the *setup* page, navigate to the left hand side of Salesforce and under *Build*, select *Create* and *Objects*.
3. In Custom *Objects*, select *New Custom Object*.
4. Under *Custom Object Information*, Insert a name for *Label*; this will create a label for tabs, page layouts and reports. *Object Name* should automatically populate with the same name as *label*, this allows the object to be referenced by the API.

5. Under *Enter Record Name Label and Format*, the *record name* is a mandatory field that appears on page layouts, search results, related lists, lookups and key lists. Insert a name for *record Name* and select *Data Type* as *Auto Number*. Write the *Display Format* as A-{0000} and set starting number as 0.
6. In *Object Classification*, tick all boxes to allow Bulk and Streaming API access and the ability to share.
7. In *Deployment Status*, select *Deployed* and tick the *allow search* box in *Search Status*. Click *Save*.

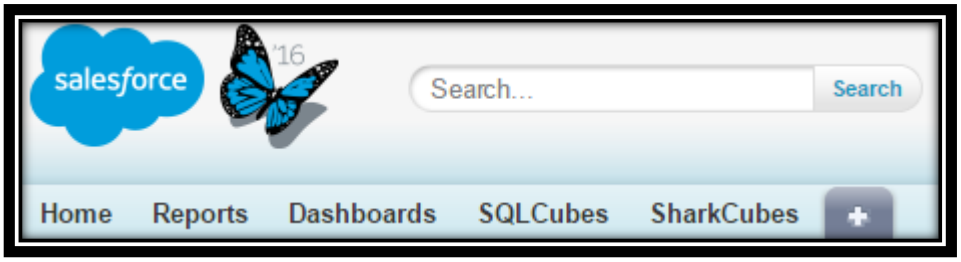
9.2.1 Salesforce Custom Object Created

8. The next step involves creating custom fieldnames to input data. In the custom object, scroll down to *Custom Field and Relationships* and select *new*.
9. In *datatype*, choose text and press *next*.
10. In *Enter the details*, insert the *Field Label*, *Length* of the text field and *Field Name*. Click *next*.
11. In *Establish field-level security*, accept the default settings and click *next*.
12. In *add to page layouts*, choose the default selection and click *save*. Repeat the process to create new custom fields.



9.2.2 Creation of Custom Fields in Salesforce Custom Object.

13. To see the results being loaded into Salesforce, a tab needs to be created. To create a new tab, go to the top right corner of Salesforce and click *setup*.
14. On the *setup* page, navigate to the left hand side of Salesforce and under *Build*, select *Create* and *Tabs*.
15. In *Custom Object Tabs* select *new*. In *Object*, choose an existing custom object and pick a *tab style*. Press *next*.
16. In *add to profiles* and *add to custom apps*, accept the default settings and press *next* and save.



9.2.3 Custom Tabs from Custom Objects are automatically displayed in Salesforce.

10. Data Preparation

10.1 Amazon

The Amazon Reviews dataset is supplied by the Stanford Network Analysis Project (SNAP). The data set includes 142.8 million reviews over a span of eighteen years, including user information, product ratings and plain text reviews. The data is presented in JSON format with raw reviews consisting of 20GB with metadata at 1.9GB.

The dataset is prepared for benchmarking by slicing data into segments and demoralising with the metadata and calculations through Python.

10.1.1 Acquiring the Dataset

1. Access the SNAP Stanford page to access the dataset: *http://snap.stanford.edu/data/web-Amazon.html*
2. Under *Note*, click for the new and improved Amazon dataset; this corrects duplication issues and offers a more complete data/metadata set. The URL is: http://jmcauley.ucsd.edu/data/amazon/
3. Once clicking on a dataset link, the user is informed that, in order to gain access to the dataset, they need to contact Julian McAuley at *julian.mcauley@gmail.com* to obtain a link.
4. Access the link Julian McAuley has supplied. When selecting data, download data and metadata based on per-category files. The reason behind this is based on two reasons. The first is by selecting categories, downloads are faster; and secondly the combined raw dataset is unordered and would require longer processing time to demoralise the data, where per category is ordered and is more efficient in processing.

10.1.2 Slicing a JSON Extract

Two Python scripts were created for slicing Json files. The reasoning behind this is that though Reviews and Meta files appear to be in JSON format, Meta is not a JSON file. This is due to the Meta data having single quotes where JSON require double quotes for its strings.

Python Modules and Packages used: Json, ast, csv, Tkinter, tkFileDialog, tkSimpleDialog and from itertools islice.

Approach:

1. *Tkinter* and *tkFileDialog* open the directory to select a JSON file for slicing.
2. After the JSON file is open, *Islice from itertools* will slice the first 'N' amount of lines. The datatype for these lines are strings; they need to be converted into a list of dictionaries to be loaded correctly into CSV.

3. A new CSV file is created by combining a directory path with the name of the file and its then transformed into CSV and opened.
4. The CSV *DictWriter* will write the list of dictionaries to a CSV file. Fieldnames will specify what data is to be taken from each dictionary and the line terminator is based on *'/n'*.
5. The sliced data is delivered into a For Loop where the review Json files are turned into dictionaries with *json.loads ()* from the *Json* module, while Meta Json files are turned into dictionaries with *ast.literal_eval ()* from the *ast* module.
6. As the dictionaries are processed through the For Loop, each dictionary is written into the CSV file.

For more information: Access the *A_Amazon_Preparation* folder on the disc and select *Amazon_Data_Coversion* and *Amazon_Data_Conversion_Meta*.

10.1.3 Denormalization, Analysis and Increasing the Volume of the Data.

This Python script has three functions:

- To load multiple tables into a data warehouse at the same time. Denormalization is carried out to ensure loading from one giant file instead of separate files. The approach is to create an inner join between matching data between reviews and Meta Json Extracts.
- Analysis is carried out on the reviews to discover insights before importation, to expand on the data warehouse and to carry out analysis for data mining preparation.
- To stress test benchmarking, data volume needs to increase to determine performance.

Python Modules and Packages used: Tkinter, tkFileDialog, CSV, Pandas and OS.

Approach:

1. *Tkinter* and *tkFileDialog* access the review and metadata files that will be used in the Python Script.
2. *Pandas* reads the excel files into Python and merges the data files together, creating a data frame from an inner join based on the 'asin' field. Finally Pandas saves the data frame to a CSV file, which is created from manipulation of a split string.
3. This *CSV* file is a temp file and read into the python script by the CSV module. Analysis is carried out on the review to determine:
    - The character length of the review
    - Text content of the review: short, moderate, long.
    - Breaking date down to day, month and year.
    - Determine if a review is popular or unpopular.

4. A new CSV file is created by combining the Pandas data frame and analysis by updating the dictionary and writing through DictWriter.
5. To duplicate records, run the write row command again in the For Loop.

For more information: Access the *A_Amazon_Preparation* folder on the disc and select *Amazon_Data_Coversion_Join_Master.*

10.2 Cosmos

The Collaborative Online Social Media Observatory (COSMOS) is an Economic and Social Research Council (ESRC) strategy that brings a diverse background of scientists to analyse multiple dimensions in terms of social media. The COSMOS software platform enables user to access data by removing barriers in acquiring and analysing data. Twitter data from the COSMOS application was extracted for testing.

10.2.1 Acquiring the COSMOS Application

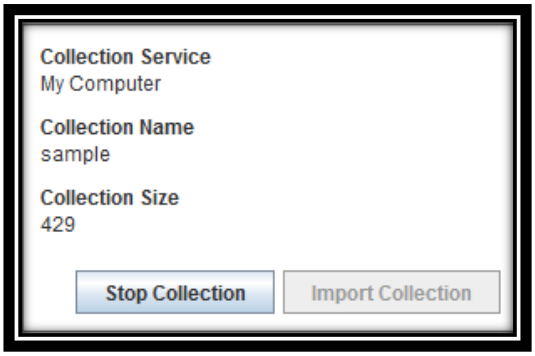
1. Access *http://www.cosmosproject.net/*, select the header *Download COSMOS* and complete the form. A member of the team will be in contact to supply you a link to the platform.
2. Download the ZIP file and extract the file.

10.2.2 Data Extraction: COSMOS Streaming API

1. In the COSMOS folder, select *COSMOS-DESKTOP-WINDOWS*, this will open the COSMOS application.
2. On the top left corner, click *Data* and select *Start New Twitter Collection*. This will present the Twitter Collection wizard.

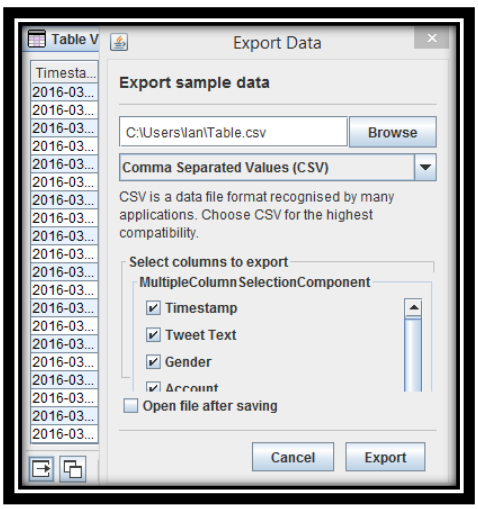
10.2.2.1 COSMOS Application: Starting a new twitter collection

3. Under Collection Service, select *My Computer* and *Authorise*. This will open a page requesting a Twitter account to authorise the COSMOS streaming API.
4. Enter a registered Twitter username and password to receive a pin number. Insert the pin number to complete authorization.
5. The required parameter to start collecting tweets is to assign a name to *Collection Name*. Select *Collection Type* as *Sample* and click *Start Collection*.



10.2.2.2 COSMOS Twitter collection

6. As seen in 9.2.2.2, tweets are being collected. When the dataset size has been reached, click *Stop Collection*, followed by *Import Collection,* followed by *Import*.
7. Next, double click on the data collection, and query without any parameters. This will bring the data into a spreadsheet format in the workspace for extraction.
8. On the bottom left corner of the View, there is an option to export the data to a CSV format. On the CSV format wizard, select all columns and press *Export*.



10.2.2.3 Exporting the Twitter data collection into CSV format.

9.3.3 COSMOS Data Preparation:

This Python Script will expand on the social media data acquired from COSMOS in order to develop the COSMOS data warehouse and carry out further insight from what the original data has supplied. Its structure is similar to *Amazon_Data_Coversion_Join_Master* with the exception that Pandas and merging data frames are not necessary. Another exception is to ensure that the time stamp produces numeric figures and not *'########'.*

Python Modules and Packages used: Tkinter, tkFileDialog, CSV and OS.

Approach:

1. *Tkinter* and *tkFileDialog* access the COSMOS Social Media file that will be used in the Python Script.
2. Using the *CSV* module, the file is read into python through DictReader and analysis is carried out discovering insights with regard to:
    - Character length of the tweet.
    - Text content of the tweet: short, moderate, long.
    - RT, to determine if the tweet is original or a re-tweet.
    - Breaking date down to day, month and year.
3. With the analysis complete, the dictionary is updated and expanded. Using the file name from a destination path split, a new CSV file is created for benchmarking through DictWriter.
4. Similar to Amazon Reviews, run the write row command twice in the For Loop to duplicate values.

For more information: Access the *B_COSMOS_Preparation* folder on the Disc and select *COSMOS_Preparation.py.*

11. Real Time Data Warehousing (RT-DW) Chain:

The development of the RT-DW Chain is broken down into four areas: The first part is the development of the directories, which the files will pass through until completion. The second step is to develop the Data Warehouse and Cube schema. Construction of the Python scripts for the chain is next, with a goal for each directory. Finally the "Chain Activator" is placed to activate the chain into real-time.

11.1 Chain Environment

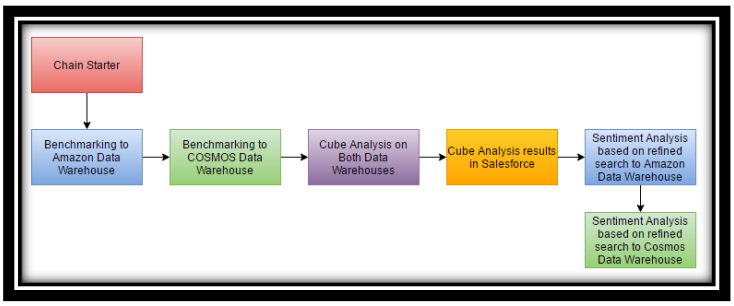
When developing python scripts for the RT-DW chain, the ideal is to map out what needs to be achieved. In this case the goal is to:

- Benchmark data into the data warehouses.
- Gather a cube analysis on both Amazon and COSMOS data warehouses.
- Place the cube analysis results into a Salesforce Custom object.
- Conduct a sentiment analysis on the CSV file by looking for a certain criteria and deploying the results in a data warehouse table.

The goal in this case is four objectives. This goal is the main function of the chain and will be used with different data sources, to process data and results. Support structures will need to be placed to ensure operation. Four support requirements are:

- Loading – Loading is the directory where all the python scripts will be stored for the chain to locate and activate. This will ensure the file will go through the chain and the objectives are achieved.
- Cubes – A data warehouse may have multiple cubes, the cubes directory is the location for all cube schemas in JSON format. This will allow transfer to new cubes easier by changing the name in the cube script.
- Temp Processor - The Temp Processor directory is used for operating systems that cannot work with Python's Temp file. The files in the Temp Processor directory allow for python scripts to access and manipulate data before benchmarking and analysis.
- Log (Optional) – Log allows users to record results in a text file and place the results in its directory. This is optional as results can be saved to a data warehouse table.

The final aspect of the Data Warehousing Chain is the vital Chain Starter that will automate the process by running servers and setting the file count to enable the execution of Python scripts.



11.1.1 Data Warehouse Chain Process

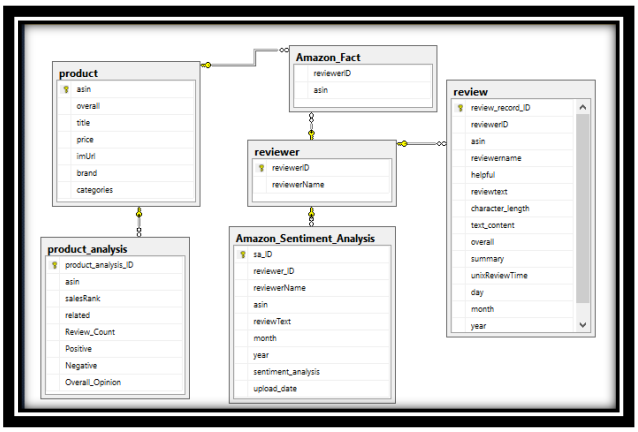
11.2 Data Warehouse Development

Two data warehouses and testing Tools:

Amazon Data Warehouse:

Tables: Eight

Table	**Function**	**Number of Columns**	**Columns**
Amazon Fact	Connects the Product and Reviewer table to establish snowflake schema.	2	reviewerID, asin
Reviewer	Primary Key table that connects to Amazon Fact and Reviews	2	reviewerID, reviewerName
Review	Table that records all reviews by reviewer.	14	review record ID, reviewerID, asin, reviewername, helpful, reviewtext, character_length, text_content, overall, summary, unixReviewTime, day, month, year
Product	Primary Key Table that connects to Amazon Fact and Product Analysis.	7	asin, overall, title, price, imUrl, brand, categories
Product Analysis	Analysis of products based per review.	8	product analysis ID, asin, salesRank, related, Review_Count, Positive, Negative, Overall_Opinion
Amazon Sentiment Analysis	Sentiment Analysis table on Amazon Reviews.	7	SA ID, reviewer ID, reviewerName, asin, reviewtext, month, year, sentiment analysis, Upload Date
Support Tables:			
Temp Reviewer	Support table to filter duplicate data for the Reviewer Table	2	reviewerID, reviewerName
Temp Product:	Support table to filter duplicate data for the Product Table	7	asin, overall, title, price, imUrl, brand, categories

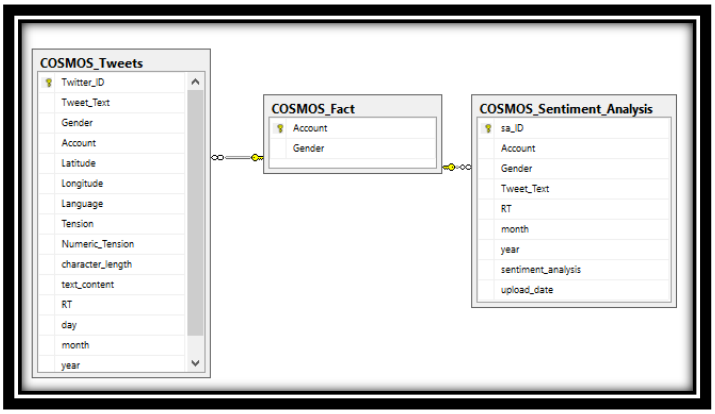


11.2.1 Amazon Data Warehouse

COSMOS Data Warehouse:

Tables: Four

Table	**Function**	**Number of Columns**	**Columns**
COSMOS Fact	Primary Key table that connect to COSMOS Tweets	2	Account, Gender
COSMOS Tweets	Table that lists twitter analysis from python processing and COSMOS application.	14	Tweet Text, Gender, Account, Latitude, Longitude, Language, Tension, Numeric Tension, character_length, text_content, RT, day, month, year
COSMOS Sentiment Analysis	Sentiment Analysis table on COSMOS Tweets.	9	SA ID, Account, Gender, Tweet Text, RT, month, year, sentiment analysis, upload date
Support Tables:			
Temp COSMOS Fact	Support table that removes duplicates from COSMOS FACT.	2	Account, Gender



11.2.2 COSMOS Data Warehouse

Testing Tools:

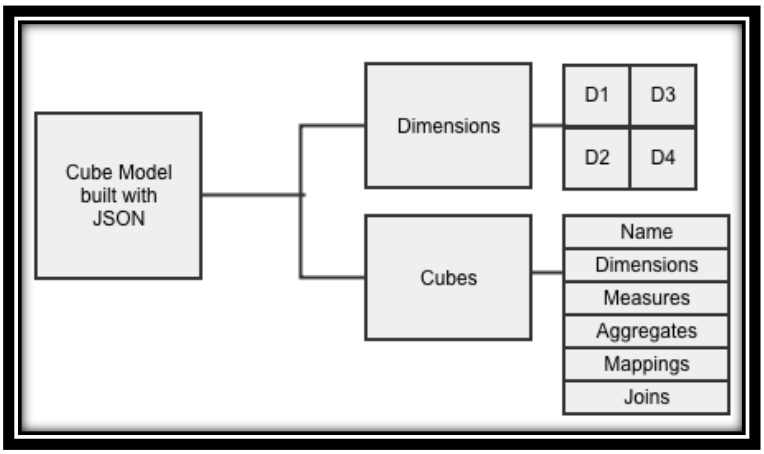
Tables: Five

Table	**Function**	**Number of Columns**	**Columns**
Time Log	Measures the time performance during benchmarking	14	Active Processes , duplicate, Amazon file , temp reviewer table load time , reviewer table load time, temp product table load time , product table load time , product analysis table load , fact table load , review table load , cosmos file , temp cosmos fact load , cosmos fact load , cosmos tweets load ,
CPU Performance Benchmarking	Measures CPU performance during benchmarking	16	Active Processes , duplicate, amazon file , pre amazon cpu , post temp reviewer cpu , post reviewer cpu , post temp product cpu , post product cpu , post product analysis cpu , post fact cpu , post review cpu , cosmos file , pre cosmos cpu , post temp cosmos fact cpu , post cosmos fact cpu , post cosmos tweets cpu
Memory Performance Benchmarking	Measures Memory performance during benchmarking	16	Active Processes , duplicate , amazon file , pre amazon memory , post temp reviewer memory , post reviewer memory , post temp product memory , post product memory , post product analysis memory , post fact memory , post review memory, cosmos file , pre cosmos memory , post temp cosmos fact memory , post cosmos fact memory , post cosmos tweets memory
Disk Performance Benchmarking	Measures Disk performance during benchmarking	16	Active Processes , duplicate , amazon file , pre amazon disk , post temp reviewer disk , post reviewer disk , post temp product disk , post product disk , post product analysis disk , post fact disk , post review disk , cosmos file , pre cosmos disk , post temp cosmos fact disk , post cosmos fact disk , post cosmos tweets disk
data loss performance benchmarking	Measures Data Loss performance during benchmarking	44	Active Processes , duplicate, Amazon file , Pre Temp Reviewer Count , Post Temp Reviewer Count , Temp Reviewer File Count , Temp Reviewer Records Processed , Pre Reviewer Count, Post Reviewer Count , Reviewer File Count , Reviewer Records Processed , Pre Temp Product Count , Post Temp Product Count , Temp Product File Count , Temp Product Records Processed , Pre Product Count, Post Product Count , Product File Count, Product Records Processed , Pre PA Count ,Post PA Count , PA file Count , PA Records Processed , Pre Amazon Fact Count , Post Amazon Fact Count ,Amazon Fact File Count ,

			Amazon Fact Records Processed , Pre Review Count , Post Review Count , Review File Count , Review Records Processed , cosmos file , Pre Temp Cosmos Fact Count , Post Temp Cosmos Fact Count ,Temp Cosmos Fact File Count , Temp Cosmos Fact Records Processed ,Pre Cosmos Fact Count , Post Cosmos Fact Count , Cosmos Fact File Count , Cosmos Fact Records Processed , Pre Cosmos Tweets Count, Post Cosmos Tweets Count ,Cosmos Tweets File Count , Cosmos Tweets Records Processed

11.3 Building a Cube Model with JSON.

To conduct cube analysis in the RT-DW chain, a Cube Model must be imported into the cube script to perform aggregate browsing, slicing and drilldowns on the data warehouses. This section discusses the implementation of a JSON cube.



11.3.1 The structure of the cube model being built in JSON.

When developing a cube model, two sub-sections are required to create an operational cube. These sub-sections are building dimensions and the cube. Once built and loaded into the workspace, use a small sample dataset to test for accurate results.

Components of the cube Model:

Dimensions:

- **Dimensions**: List of created dimensions to use for the data warehouse. Recommended to be implemented but is not necessary for dimensionless cubes.

Cube:

- **Name:** Name is a required unique identifier for the cube that will be imported into Python. When developing the cube, the name will need to be an exact match to the fact table.
- **Dimensions:** List of Dimensions that will be imported into the cube.
- **Measures:** List of cube measures of the data warehouse.
- **Aggregate:** List of aggregate measures. Example Min, Max, Sum,
- **Mapping:** Mapping of logical attributes to physical attributes.
- **Joins:** Specification of real table joins.

11.4 Chain Functionality with Python.

11.4.1 Benchmarking

To load data into multiple tables, two scripts are created per data warehouse. Within one main function, there are sub-functions, which represent tables and testing parameters. The benchmarking scripts take denormalized files and normalises data using temp files, bulk loading and support tables.

Python Modules and Packages used: CSV, Tempfile, pyhive, Pypyodbc, OS shutil, Psutil, time

Approach:

1. After importing the selected packages, declare the destination paths. These paths will read the directories on where the data is located for processing, where to move the data once processed and with regards to Windows, where the temp processors are for normalisation.
2. In the main function, connect to the data warehouse server. Use *pypyodbc* for SQL server and *Pyhive* for Hive.
3. Using the *CSV* and *OS* module, denormalized files are opened into python from a For Loop reading a directory. Once in python the denormalized file is transferred through each table that is represented as a function.
4. To normalise data, temp files are created to extract the required data and to batch load data into the table. Hive on Spark uses *Tempfile* while SQL Server needs to use created CSV files due to a limitation of Windows.
5. Data is batch loaded into the tables but to batch process primary key tables, foreign keys constraints are disabled before, and re-enabled after loading. To batch load primary key

tables, support tables are required. Batch load all data to the support table and distinct data within that table to populate the primary key table. In Hive this will lead to accurate cube analysis where on Windows this will remove Python's 'line to line' loading limitation and transfer to batch processing.

6. After the file is loaded into the data warehouse, the file is then moved to the destination path using *shutil*.
7. To measure performance parameters, data is recorded pre and post loading. In terms of timing, time is recorded by subtracting pre table time with post table time using the time module. Data loss is saved by querying the data warehouse and file counts. Finally Memory, CPU and Disk are recorded through snapshots from the *Psutil* module. Each calculation is assigned to a global variable and loaded into their respective tables.

For more information: Access the *Chain_Enviroments* folder on the Disc and select either environment and in *Loading* chose either *Amazon_Loading_SQL_Final2.py* or *Amazon_Loading_Hive3.py.*

11.4.2 Cube Analysis – Salesforce Integration

With data loaded into the data warehouses, the next phase is to get a cube analysis from the Cube and Salesforce Block. The script is broken down into three functions. The Main, Amazon and Cosmos. In the Main Function, Amazon and Cosmos functions are called where both these functions interact with their respective data warehouse.

Python Modules and Packages used: Cubes, CSV, Simple Salesforce and date time.

Approach:

1. To access a data warehouse for cube analysis, use the Python module *Cubes* and configure with the server.
2. Next, Import a JSON created Cube Model from the directory path.
3. Declare the Cube to the workspace browser and aggregate the browser, the name of the cube should be named after the fact table in the JSON script.
4. To get specific measures, use *result.summary()* and to perform drilldowns on dimension use *browser.aggregate()*
5. Assign the measures to variables for Salesforce integration. (Optional) With regards drilldowns, dimensions can be written to a csv file using *CSV*, Data Warehouse or Salesforce using the Bulk API.

6. With all data warehouses measures recorded, Use *Simple Salesforce* and access the API with a username, password and Security Token.
7. To populate Salesforce, assign the cube variable to their salesforce counterparts. When using Custom objects, API names end with *'__c'* and the Primary ID number in salesforce API is always declared *'Name'* (*Note: Primary ID number is not needed if it is declared as 'Auto ID'*.)
8. (Optional) Use Date time to upload the exact time cube analysis is uploaded to Salesforce with *Datetime.datetime.now()*

For more information: Access the *Chain_Enviroments* folder on the Disc and select either environment and in *Loading* chose either *Cube_SQL* or *Cube_Hive* script.

11.4.3 Data Mining – Sentiment Analysis

With the cube analysis recorded, the final stage of the chain is to conduct data mining. In this implementation a sentiment analysis is recording Amazon Reviews and Cosmos tweets to determine if they are positive or negative. There are three functions in this python Script: Main, Amazon and Cosmos. Similarly with Cube Analysis and Salesforce, the main function contains the Amazon and Cosmos sub-functions. Files in both the Amazon and Cosmos Sentiment Analysis directories are processed and loaded into the data warehouses.

Python Modules and Packages used: CSV, pyhive, pypyodbc, Urllib, Tempfile, Shutil, OS

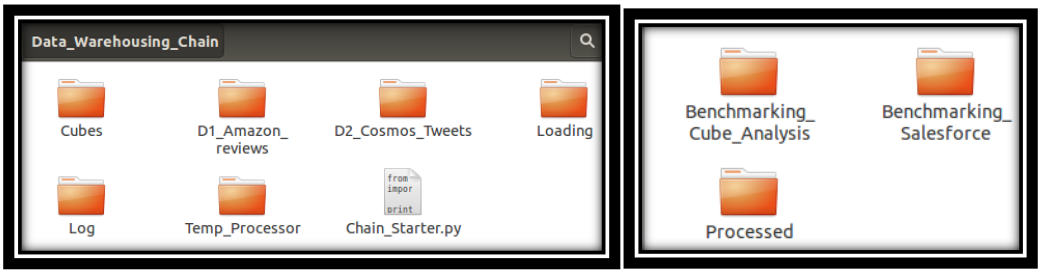
Approach:

1. First declare the directory paths that contain the CSV files. If using Windows, declare directory paths for temp processors.
2. In the Main function, using *pyhive* or *pypyodbc*, connect the server and configure the warehouse.
3. Using *OS*, the directory will open each *CSV* file and will filter the For Loop based on a specific criteria. The reviews and tweets are converted to lowercase letters and then through *Urllib,* a sentiment analysis is carried out across the internet to determine positive and negative strings. This data is saved to normalised data files using *Tempfile* on Linux or Windows created files.
4. The normalised data is then bulk loaded into each data warehouse.
5. Once the file is processed, *shutil* moves the document into the next directory.

For more information: Access the *Chain_Enviroments* folder on the Disc and select either environment and in *Loading* chose either *Amazon_Review_Sentiment_Analysis* or *Data_Warehouse_Sentiment_Analysis* script.

11.5 Chain Activation:

Currently the scripts in the *loading* directory need to be executed manually to produce results. To automate the scripts in a row, one after another a master python script must be created. In Windows, OS calls from the OS module will activate the scripts where the SubProcess module will enable the scripts to run on Linux. Control Parameters can be placed using 'If and Else' statements to control file count, where *'#'* can deactivate scripts. To start up a server, the master script can navigate through the directories to activate it. Place the *Chain Starter* python with the directories.

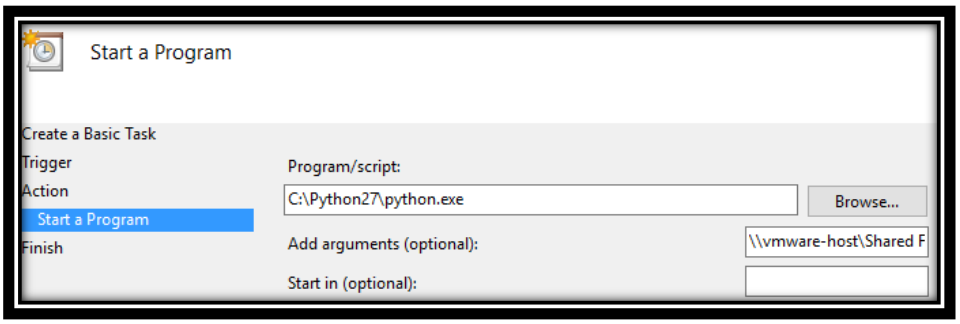


11.5.0.1 (Left) The Data Warehousing Chain, (Right) D1_Amazon_Reviews data source chain

Once the Chain Starter is activated through the terminal, command prompt or clicking on the python script, the chain is activated. To remove manual input Windows Task Scheduler and Linux Crontab are needed.

11.5.1 Chain Starter (Windows Task Scheduler)

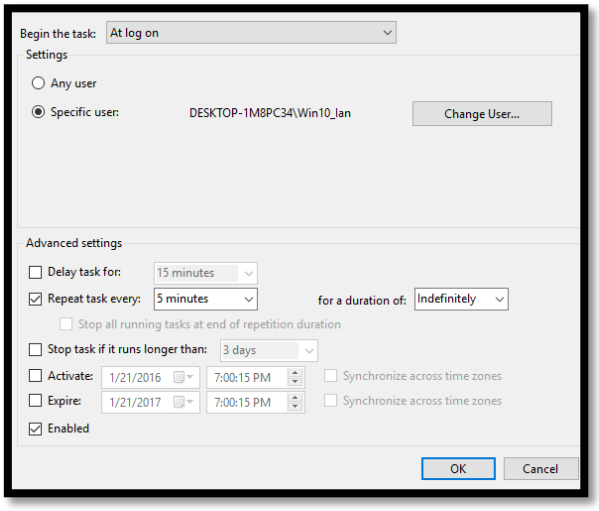
1. Go to *Control Panel* and search *Schedule Tasks* under *Administrative Tools*. An alternate approach is search with *Task Scheduler*.
2. On the right side bar named *Actions*, select *Create Basic Task*.
3. In the wizard, insert *name* and *description*.
4. Under *Trigger*, select *When I Log on*.
5. Under *Action*, select *start a program*.
6. Under *Start a program*, *Program/Script* will be *Python.exe* path and the argument is the location of the file.
7. At *Finish*, click *Finish*.



11.3.1.1 Chain Starter Python Script gets activated at log on.

Currently Task Scheduler activates the chain starter Python Script when the user logs in and stops after one execute. To repeat for N amount of minutes, the task must be altered.

8. On the left hand side of *Task Scheduler* under *Task Scheduler (local)*, select *Task Scheduler Library* to see all tasks.
9. Select the created task and right click *Properties*.
10. At the *general* tab under *security options*, select *run with highest privileges* and configure for the operating system, in this case *Windows 10*.
11. At the *Triggers* tab, click *edit* and under *Advanced Settings* choose five minutes for *repeat task every:*. Assign indefinitely to *for a duration of:* and tick *Enabled*.



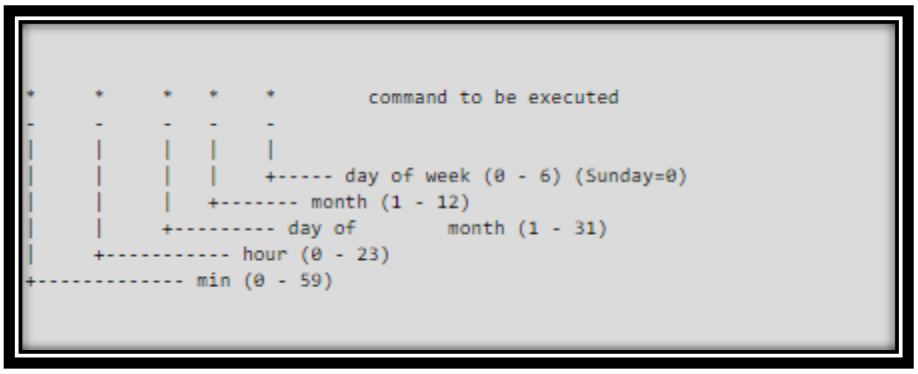
10.3.1.2 Editing a task to repeat every five minutes.

12. At the *Settings* tab, ensure that it allows the task to run on demand. Enable to run task as soon as possible after a scheduled task is missed and by restarting it at one minute intervals with three attempts. The first five of six boxes on this wizard should be ticked.
13. After completion, restart Windows for changes to take effect.

11.5.2 Chain Activation: Chain Starter (Linux Crontab)

1. Access the terminal and type *crontab –e*
2. To run the chainstarter.py program every two minutes, type into the terminal:

   */2 * * * * /usr/bin/python /home/ian/Desktop/Data_Warehousing_Chain/Chain_Starter.py*



11.4.2.1 Crontab Syntax to repeat function us/ (Supplied by http://www.adminschoice.com/)

3. Press *'CTRL + X'* to save modifications.

12. Appendix

The appendix is located on the CD-ROM attached to the configuration manual. If reading this online, the appendix is located under Ancillary Artefacts. The contents of the CD-ROM/Ancillary Artefacts are the tools to develop a sandbox that allows the reader to test the RT-DW chain on their computer. Please ensure when running experiments that you change the Python directory paths to the machine directory paths.

12.1 CD-ROM/ Ancillary Artefacts Contents:

Section A: A_Amazon_Preperation

Section A contains one folder consisting of three python scripts to prepare the Amazon Reviews Dataset:

1. Amazon_Data_Conversion.py (Data Conversion of Amazon Reviews JSON files)

2. Amazon_Data_Conversion_Meta.py (Data Conversion of Amazon Meta JSON files)

3. Amazon_Data_Conversion_Join_master.py (Python Script that Denormalises Reviews, Meta and Analysis Data.)

Section B: B_COSMOS_Perperation

Section B contains one folder consisting of one python script to prepare the COSMOS Dataset:

1. COSMOS_Preperation.py (Python Script that Denormalises Cosmos and Analysis Data)

Section C: C_Chain_Enviroments

Section C contains two folders on where the experiments were carried out.

1. SQL_Data_Warehousing_Chain

Folders:

1A: Cubes (Cubes Directory)

1A.1 Amazon_retro2 (Cube Model for Amazon Data Warehouse)

1A.2 cosmos (Cube Model for Cosmos Data Warehouse)

1B: D1_Amazon_reviews (Amazon Data Directory)

1B.1: Benchmarking_Cube_Analysis (Benchmarking)

1B.2: Benchmarking_Salesforce (Cube Analysis - Salesforce, Sentiment Analysis)

1B.3: Processed (Completed Files)

1C: D2_Cosmos_Tweets (Cosmos Data Directory)

1C.1: Benchmarking_Cube_Analysis (Benchmarking)

1C.2: Benchmarking_Salesforce (Cube Analysis - Salesforce, Sentiment Analysis)

1C.3: Processed (Completed Files)

1D: Loading (Python Scripts Directory)

1D.1: Amazon_Loading_SQL_Final2.py (Amazon Benchmarking Script for SQL Server)

1D.2: COSMOS_Loading_SQL.py (COSMOS Benchmarking Script for SQL Server)

1D.3: Cube_SQL.py (Cube Analysis for SQL Server Data Warehouses)

1D.4: Data_Warehouse_Sentiment_Analysis (Sentiment Analysis for Amazon Reviews and Cosmos_Tweets)

1E: Temp_Processor (Location for temp files)

1F: Chain_Starter.py (Windows Task Scheduler activates script for real time.)

2. Hive_on_Spark_Data_Warehousing_Chain

Folders:

2A: Cubes (Cubes Directory)

2A.1 Amazon5 (Cube Model for Amazon Data Warehouse)

2A.2 cosmos (Cube Model for Cosmos Data Warehouse)

2B: D1_Amazon_reviews (Amazon Data Directory)

2B.1: Benchmarking_Cube_Analysis (Benchmarking)

2B.2: Benchmarking_Salesforce (Cube Analysis - Salesforce, Sentiment Analysis)

2B.3: Processed (Completed Files)

2C: D2_Cosmos_Tweets (Cosmos Data Directory)

2C.1: Benchmarking_Cube_Analysis (Benchmarking)

2C.2: Benchmarking_Salesforce (Cube Analysis - Salesforce, Sentiment Analysis)

2C.3: Processed (Completed Files)

2D: Loading (Python Scripts Directory)

2D.1: Amazon_Loading_Hive3.py (Amazon Benchmarking Script for Hive on Spark)

2D.2: COSMOS_Loading_Hive2.py (COSMOS Benchmarking Script for Hive on Spark)

2D.3: Cube_Hive.py (Cube Analysis for Hive Data Warehouses)

2D.4: Amazon_Review_Sentiment_Analysis (Sentiment Analysis for Amazon Reviews)

2D.5: Amazon_Loading_Hive_Alternative.py(Alternative Amazon Benchmarking Script for Hive on Spark)

2E: Temp_Processor (Location for temp files)

2F: Chain_Starter.py (Crontab activates script for real time.)

Section D: D_Documents

Section D contains two folders of documentaion to read and assit in development on any machine.

1. Deliverables:

Documents:

1A. Report:

1B: Configuration Manual:

2. Warehouse Scripts:

Documents:

2A: Hive_on_Spark_Script

2B: SQL_Server_Script

Section E: E_Sample_Data.

Section E contains one folder with two sample datasets for testing.

1. Amazon_Reviews_2500 (Consists of 2500 records for the Amazon Data Warehouse)

2. Cosmos_Tweets_2500 (Consists of 2500 records for the Cosmos Data Warehouse)
