CIRRUS

Project report submitted in fulfillment of the
requirement for the degree of
Bachelor of Technology

In
Computer Science and Engineering
By

SAGAR GARG (151345)

Under the supervision of
Mr. TEJESVI TADEPALLI (Solution delivery consultant, ZS ASSOCIATES.)

To

Department of Computer Science and Engineering and Information Technology
Jaypee University of Information Technology, Waknaghat,
Solan-173234, Himachal Pradesh

CERTIFICATE

Candidate's Declaration

This is to specify that the work which is being presented as the report entitled "CIRRUS" in partial fulfillment for the requirements for the award of the degree of **Bachelor of Technology** in Computer Science and Engineering/Information Technology submitted in the department of **Computer Science Engineering and Information Technology,** Jaypee University Of Information Technology, Waknaghat is an authentic record of my own work carried out in the period of 4th February,2019 to 17th May,2019 under the supervision of Mr. Tejesvi Tadepalli (Solution delivery consultant at ZS ASSOCIATES, Gurgaon).

The matter enclosed in the report has not been submitted for the award of any other degree or diploma.

Sagar Garg (151345)

This is to certify that the above statement made by the candidate is true to the best of my knowledge.

Mr. Tejesvi Tadepalli
Solution delivery consultant,
ZS associates.
Dated: 17th May, 2019

ACKNOWLEDGEMENT

I owe my profound gratitude to my project supervisor **Mr. Tejesvi Tadepalli,** who took keen interest and guided me through the development phase under the project **CIRRUS**, till the completion of my project by providing all the necessary information. This helped me thoroughly in carrying out fruitful research and sound technologies. I am thankful to him for all the support.

I express my sincere thanks to ‘Mr. Manish Gupta for allowing me to work on my project within premises of ZS associates, Gurgaon.

At the end I would like to express sincere thanks to all my colleagues and others who helped me during the course of this project work.

TABLE OF CONTENTS

List of Figures

ABSTRACT

Now-a-days, pharmaceutical industry is one of the fastest growing industries in the world. With increase in demand of pharmacy, it is beneficial for pharmaceutical organization to have a basic understanding of what they are how doing and how can their stats help them. We have made a system named C-SCAN where our pharmaceutical client can see the views, sales, growth and performance in their industry. Through this architecture, organizations would also be able to view what is going on in the pharma industry, they compare their growth with other organizations etc. In this fast-growing era, nobody has enough time to go and look out and analyze something in depth. Therefore, the information provided by C-SCAN in the form of reporting to our clients would make their life easier and help them to flourish in their domain.

Raw data is taken as input, and output are the formatted extracts with implied business rules.

We are using SQL, PostgreSQL, Python scripts, shell scripts, etc. in C-SCAN. The functionalities of the system can be updated as per the client specifications.

Chapter-1

INTRODUCTION

1.1 Introduction

ZS is a professional services firm that helps companies develops and delivers products for their customers. Our experts work side by side with clients, leveraging analytics, technology and strategy to create solutions that work in the real world. ZS focuses on doing the right things the right way and aims at getting it right the first time.

Pharma 101:

In US Pharma industry, patients do not get the drugs without proper prescription. Patients have payers, who pays the major amount of the drug. Payers are insurance companies; each patient has Its own payer and its own plan. In Pharmaceutical industry, doctors are considered as customers.

When a drug is manufactured, it requires an FDA approval. After going through many clinical trials, the drug gets approval from FDA and gets a unique NDC number. The drug goes through several phases.

Drug development and life cycle- Firstly, drug goes through a number of clinical trials; to ensure feasibility, safety, and efficacy. After the successful completion of trials drug gets the 'Patent'.

Life cycle: Launch, Growth, Maturation, Competition, Patent loss.

R_X: It is used to denote Prescription.

NR_X: New Prescription.

TR_X: NR_X + refills.

Patent approval: For a drug to get patent, it needs to go under some clinical trials.

- **Clinical Trial 0:** In this, drugs are tested on animals like rats, rabbits in different atmospheric conditions.
- **Clinical Trial 1:** Here, drugs are tested on humans like refuges, homeless people who opts for these trials for money.
- **Clinical Trial 2:** 'Placebo' effect comes into picture. Actual medicine and fake medicines are given to different people. If actual medicine cures more people, then it gets the patent. This trial ensures efficacy. It is also known as **'Double Blind'**, because even doctors don't know that fake medicines are given to the patients.
- **Clinical Trial 3:** In this, drugs are checked for side effects.

In the end, the final report of each 'Clinical Trial' is submitted to **'FDA'** which approves the drug for selling it.

On the day of approval, medicine gets a 'NDC' number from 'FDA'.

1.2 Problem Statement

It takes a huge amount for a manufacturer to manufacture a drug and get 'Patent'. Once the drug loses its Patent after 15 years it goes 'Generic' in the market.
To help the client earn the money back invested in manufacturing the drug and to gain maximum amount of profit from it in minimum time by gathering data for marketing and sales before the drug goes generic.

1.3 Objectives:

To provide solutions to our clients so that they can flourish in their domain with minimum of efforts and investment. To analyze the data for different clients coming from different vendors and to generate meaningful information after applying business rules and tools for better understanding.

1.4 Methodology:

We use Waterfall Model to built the project methodology.

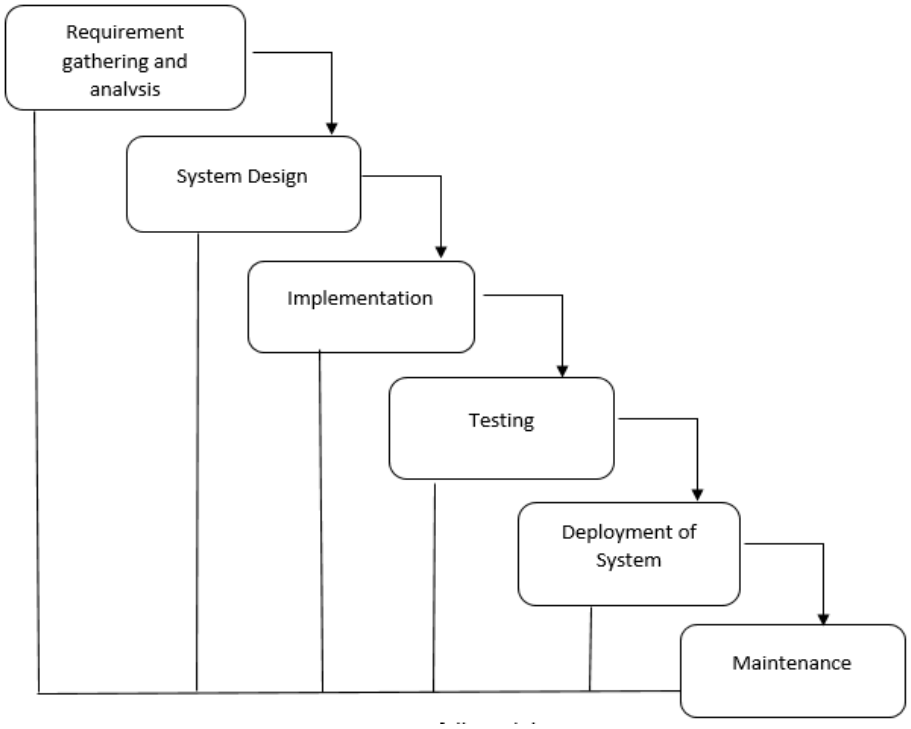


Fig 1.1 Methodology

Waterfall model is very easy to understand and use. In waterfall model, before moving to a new part of the project it must be ensured that the previous task is completed successfully.

Merits:

- This model works well for the projects where the requirements are well known and understood.
- In this type of model, different segments or tasks are processed and completed one at a time. Different phases do not intersect with each other.

Demerits:

- Not a healthy practice to use this model when the project is in development phase.
- Higher risk and ambiguity.

This model can be used when-

- Project is well defined and stable.
- Project requirements are well known in advance and not expected to get altered.

1.5 Organization

Chapter 1 provides a summary and introduction by identifying the objective of the project. The methodology used to build the project and the techniques used at various stages of development.

Chapter 2 The topics included to build and develop the system and understand the flow of process are included here.

Chapter 3 covers the all the system as well as software and hardware requirements along with the functional and non-functional requirements for better understanding.

Chapter 4 contains the Performance Analysis. Corresponding results with the screenshots are included in this chapter.

Chapter 5 includes the conclusion and future scope of this project's implementation for future improvements.

Chapter-2

LITERATURE SURVEY

2.1 DATA WAREHOUSE:

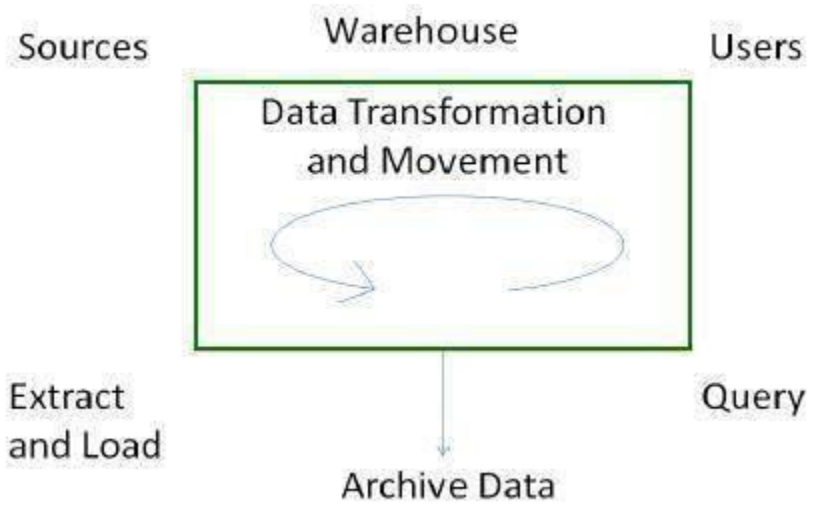
Data warehousing is a technology which sums up structured data from multiple sources that helps in comparing and analyzing the data for greater **business intelligence**.
Data warehouses are a bit different from standard operational databases in terms of design. The operational databases maintain strict accuracy of data in the moment by updating real-time data very rapidly. Data warehouses in contrast, are designed to give a long-range view of data over time.

BENEFITS:
Ability to analyze data from various sources and to overcome differences in storage schema using the ETL process.

Functions of Data Warehouse Tools and Utilities:

Below are the functions of data warehouse tools and technologies −

- **Data Extraction** − Collection of data from multiple heterogeneous sources.
- **Data Cleaning** – It involves discovering and correcting the errors in the data.
- **Data Transformation** – It converts the data to warehouse format from legacy format.
- **Data Loading** – This process involves summarizing, sorting, consolidating, checking integrity, and building partitions etc.
- **Refreshing** – It includes the updating of data into the warehouses.



.

Extract and Load Process:

Data extraction means taking the data from various source systems. Data load means extracting the data and loading it into the data warehouse.

Before processing the data into the warehouse, the information must be reconstructed gathered from the sources.

ETL-process:

ETL (Extract, Transform and Load) is a process responsible for pulling the data out of all the source systems and inserting it into a data warehouse. ETL involves the following tasks:

extracting the data from source, data from various source systems is consolidated into one data warehouse, which is ready for transformation and processing.

transforming the data includes:

- applying business rules (facts & dimensions),
- cleaning (mapping of NULL to 0 or 'Male' to 'M'),
- partitioning a single column into multiple columns.
- joining the data coming from several sources.
- applying any sort of data validation.

loading the data involves loading data into a data warehouse.

Business Intelligence(BI) - technology for gaining maximum amount of information from the available data for improving the business processes. It performs analysis and provides reasonable information that helps in making effective and high-quality business decisions.

2.2 Data Warehouse Schema Architecture:

SCHEMA:
Schema is defined as the logical description of the entire database. It consists of the names and description of each record along with the associated data-items and aggregates. Like a database, a data warehouse also requires maintaining of a schema. Relational model is used for database, whereas a data warehouse uses Star, Snow-flake, and Fact-Constellation schema.

There are many schema models used for data warehousing, among them commonly used are:

- **Star schema**
- **Snowflake schema**

Star schema:

Star schema architecture is the one of the simplest data warehouse schema. The center of the star consists of fact table whereas the edges of the star are dimension tables.

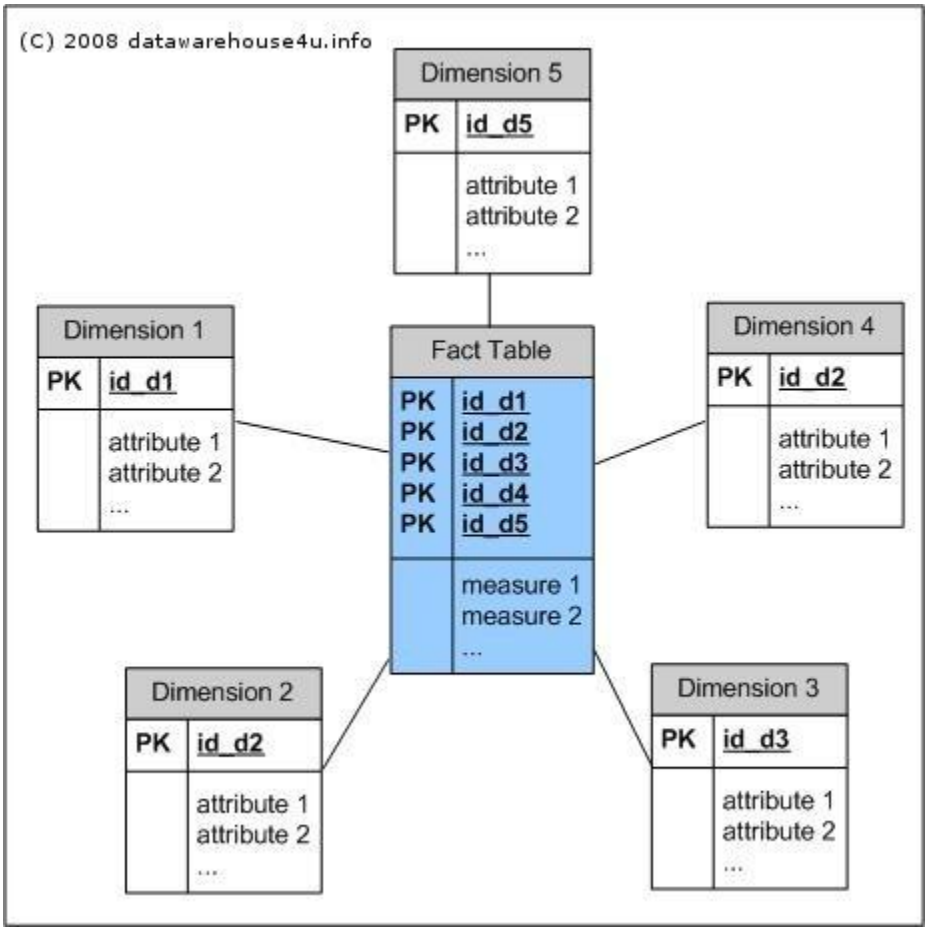


Fig 2.1 Star schema

- Every dimension in a star schema is devoted with 1-dimension table only.
- Dimension tables contains the entire set of attributes.
- Fact table is placed at the center. It includes the keys to each of five dimensions.

Snowflake Schema:

Snowflake schema is a variant of star schema. In this, the fact table is connected to multiple dimensions. Dimensions are aligned in a normalized form in multiple tables that are related to each other. The snowflake schema affects only the dimension tables and not the fact tables.

- In Snowflake schema some of the dimension tables are normalized.
- This normalization distributes the data into additional tables.

- Unlike Star schema, dimensions table in a snowflake schema are normalized. For instance, the item dimension table in star schema is normalized and is distributed into two-dimension tables, ‘item’ and ‘supplier’ table respectively.

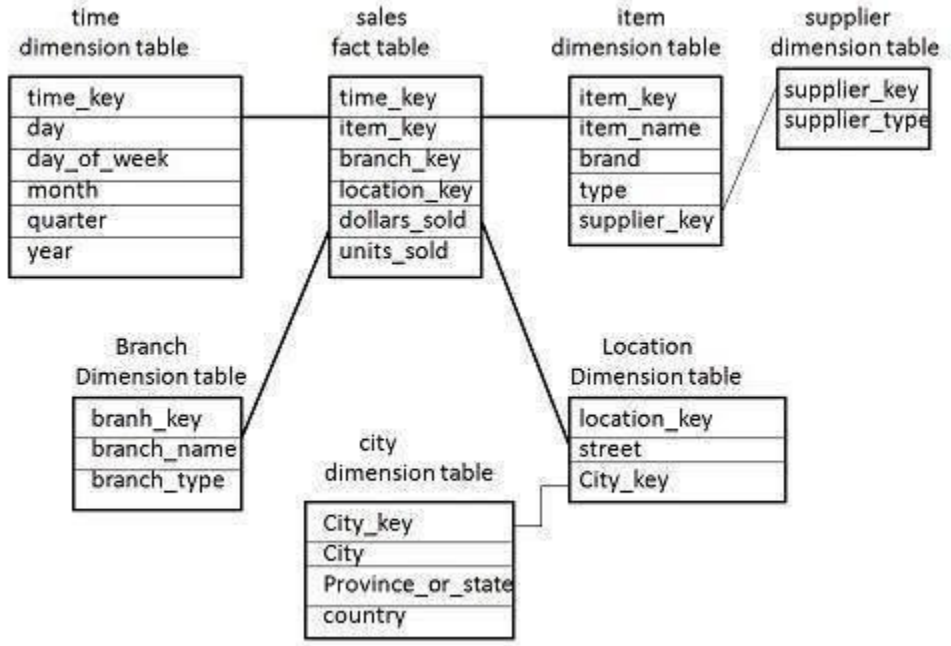


Fig 2.2 Snowflake schema

- Item dimension table contains the attributes item-key; item-name; type, brand, and supplier-key.
- The supplier-key is associated to supplier dimension table. The supplier dimension table further contains the attributes supplier-key and supplier-type.

Fact Constellation Schema:

Fact constellation schema can be constructed from each star schema by splitting the actual star schema into multiple star schemes. The fact constellation contains multiple fact tables that have many dimension tables. Fact constellation schema is collection of Star as well as Snowflake schema due to which it is also known as Galaxy schema.
Fact constellation schema has a more complicated design which is its main disadvantage.

The following diagram shows Fact constellation schema.

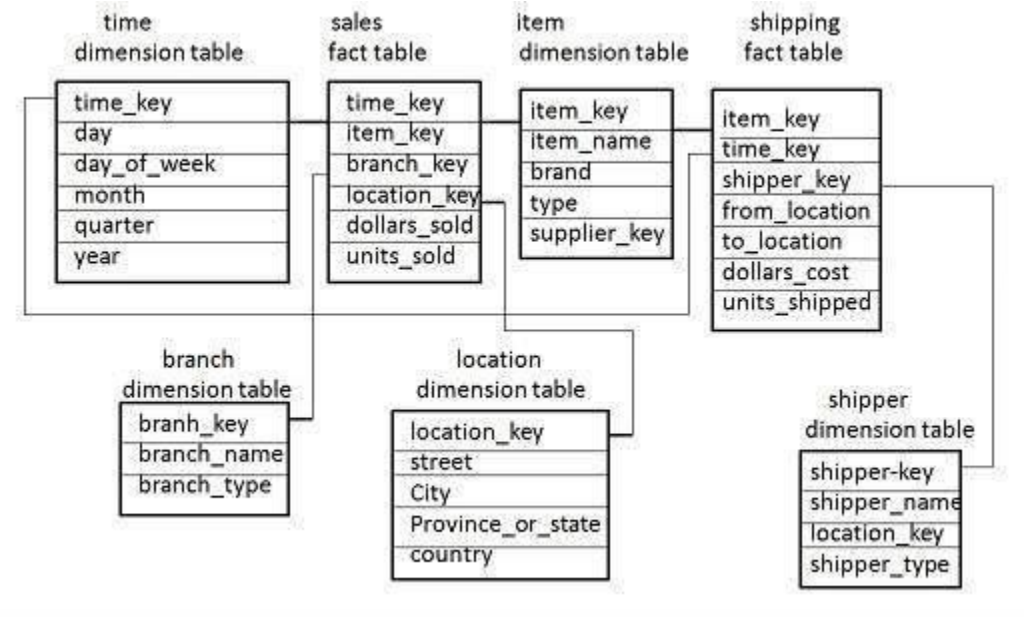


Fig 2.3 Fact constellation schema

- It is possible to share dimension tables between fact tables. For instance- item, time and location dimension tables are being shared between the shipping and sales fact table.

Fact Tables:
A fact table generally consists of two types of columns: foreign keys to dimension tables and measures to those that contain numeric facts. A fact table may have fact's data on detail or at an aggregated level.

Dimension Tables:
A dimension is a structure generally comprised of one or more hierarchies that differentiates data. If a dimension doesn't have a hierarchy or levels, then it is called **flat dimension or list**. Primary keys of every dimension tables are part of composite primary key of fact table. Dimensional values are well described by dimensional attributes. They are textual values. Dimension tables are usually smaller then fact table.

A fact table stores data about sales whereas a dimension table about geographic territory like market, state and cities.

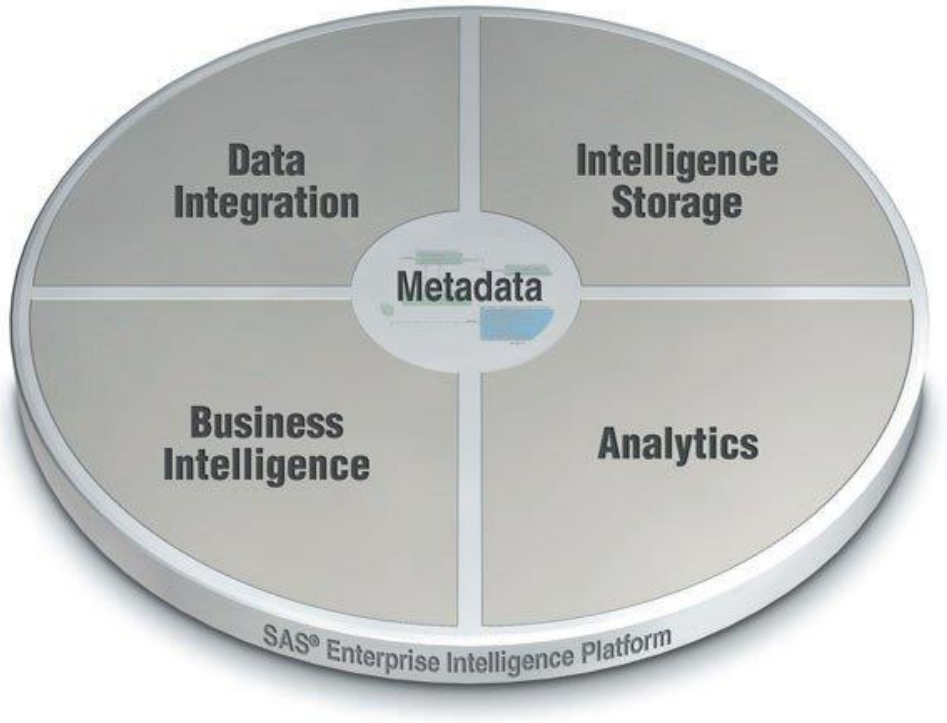


Fig 2.4 Metadata

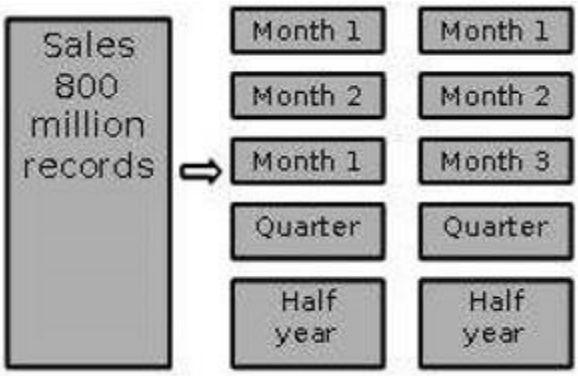
2.3 Partitioning:

Partitioning allows us to easily manage the data and enhances the performance of data. It helps in maintaining different requirements of the system. Partitioning of each fact table into multiple separate partitions eases the management of data warehouse optimizes the hardware performance.

HORIZONTAL Partition

Partition by Time into Different-sized Segments

Wherever aged data is accessed infrequently horizontal partitioning comes into picture. It is implemented as larger partition for inactive data and small partitions for current data.



VERTICAL Partition

Vertical partitioning, partitions the data vertically. The below graphic depicts vertical partitioning working.

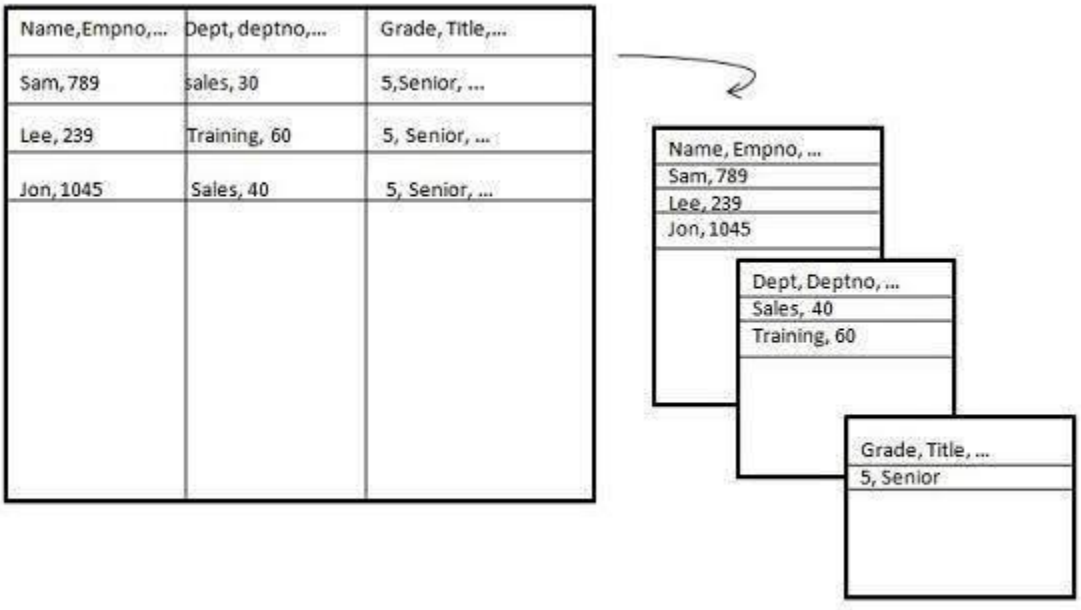


Fig 2.5 Vertical portioning

2.4 AWS SERVICES USED:

Amazon Redshift:

Amazon Redshift is a fully managed, fast and petabyte-scale data warehouse which makes it easier and cost-effective to analyze the entire data using the existing business intelligence tools. It allows us to start small without commitments, and scale to petabytes for very less cost of

traditional solutions. Users normally experience 3 times compression, thus significantly reducing the costs.

Features and Benefits:

Amazon Redshift uses columnar storage technology that improves I/O efficiency and provides multi parallel processing across the nodes due to which it allows us to deliver fast query performance. It has custom JDBC and ODBC drivers which allows us to use a broader range of familiar SQL clients. It also gives the option of standard PostgreSQL JDBC and ODBC drivers. Speed of the data load varies linearly with the cluster size, with integrations to Amazon S3, DynamoDB, Amazon Elastic MapReduce.

Amazon Redshift's data warehouse architecture allows us to automate the administrative tasks linked with configuring, provisioning, and monitoring a cloud data warehouse. Restores are very quick; we can start querying in few moments while the data is processed down in the background. Amazon S3 backups are incremental, continuous and automatic.

Fast

Optimized for Data Warehousing

It uses columnar storage and zone maps to reduce the amount of I/O needed to perform queries. It has a massively parallel processing (MPP) data warehouse architecture. SQL operations takes full advantage of all available resources. The hardware is designed for high data processing performance.

Cheap

We pay only for the resources that we use. It provides On-Demand pricing with no commitments.

Simple

Amazon Redshift configures the connections between the nodes, and secure the cluster.

Fully Managed

Amazon Redshift manages, and monitors the data warehouse. It monitors cluster health and takes regular backups.

Automated Backups:

Amazon Redshift continuously backs up new data on the cluster to Amazon S3 by using automated snapshot feature. Snapshots are incremental, continuous and automatic.

Secure:

Amazon Redshift provides firewall rules to control network access to the data warehouse cluster. We can run Amazon Redshift inside Amazon VPC to isolate the data warehouse cluster in a personally owned virtual network.

Amazon EC2:
Virtual Servers in Cloud computing. Acts like a bridge among all amazon services like RDS, Redshift

It is a web service. It allows us to configure compute capacity with minimal friction. It provides complete control of computing resources and runs on Amazon's computing environment. It reduces the time taken to boot new server instances to minutes, allowing quick scale capacity. It allows us to pay only for capacity that we actually use.

Python scripts and shell scripts run on EC2 processes everything.

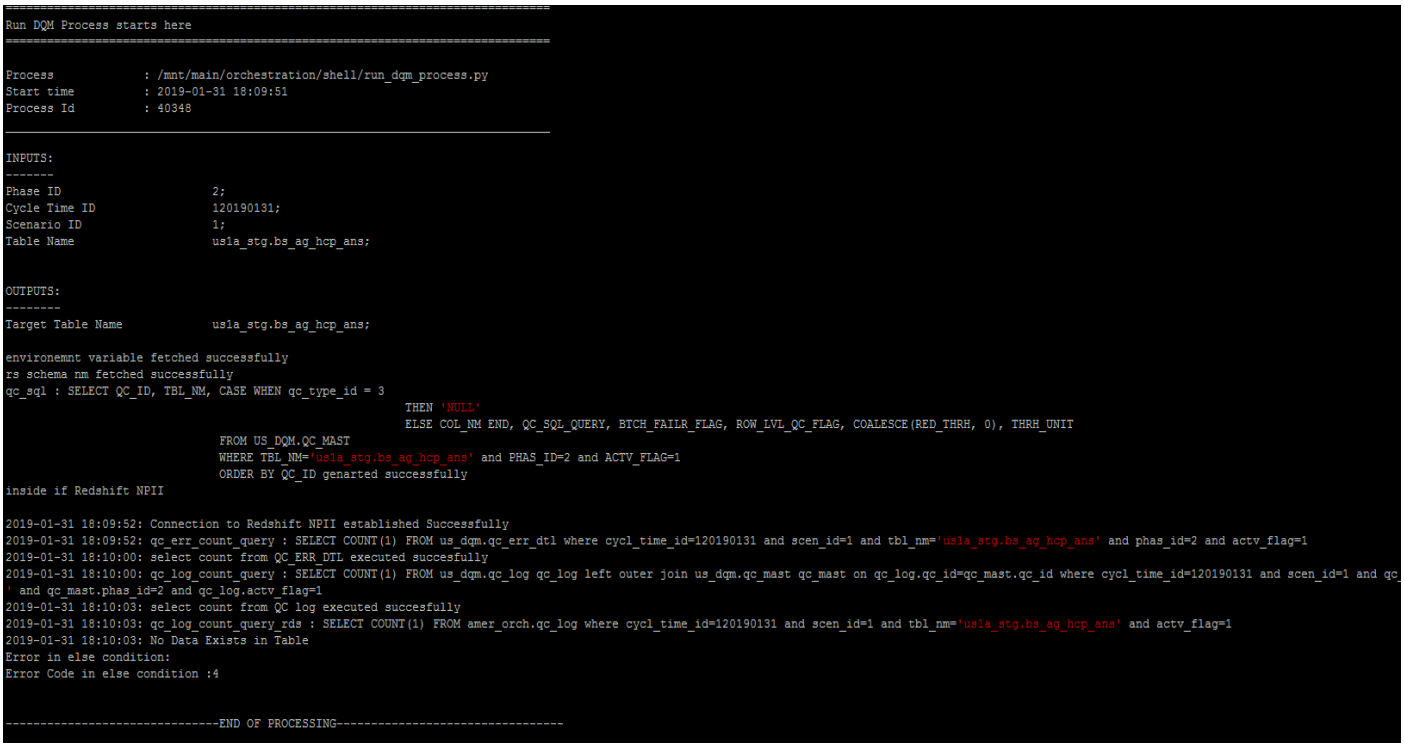


Fig 2.6 EC2 snapshot used in C-SCAN

Amazon Relational Database Service (RDS):

It is a managed Relational Database Service. RDS makes it easier to set up, and operate a relational database in cloud. It is cost-efficient and provides resizable capacity in managing time-consuming administration tasks, allows us to focus on our business.
It provides access to multiple database engines, like Amazon Aurora, MySQL, PostgreSQL, Oracle etc. The code, tools, and applications that we already use can be used with Amazon RDS.

Amazon Aurora:
Amazon Aurora is a MySQL and PostgreSQL compatible relational database built for the cloud. It combines the performance of high-end commercial databases with cost-effectiveness of open source databases.
It is up to 5X faster than standard MySQL databases and 3X faster than standard PostgreSQL databases. It provides the availability, security and reliability of commercial-grade databases at 1/10th of the cost. Aurora is fully managed by Amazon Relational Database Service which automates time-consuming tasks like hardware provisioning and backups.

Database Migration Service:
It migrates databases to AWS with minimal downtime.

It allows us to migrate databases to AWS securely and quickly. The source database remains completely operational during the migration, that minimizes downtime for applications that rely on the database.

It supports both homogenous migrations and heterogeneous migrations like Oracle to Oracle, and Oracle to Amazon Aurora. It allows us to stream the data to Redshift, Amazon S3, DynamoDB enabling easy analysis of data in petabyte-scale data warehouse.

AWS LAMBDA:
AWS Lambda allows us to run the code without managing servers and provisioning. We pay only for the compute time that we consume. Nothing is charged when the code is not running. With the help of Lambda, we can run the code for any application or backend service with no administration.
We just have to upload our code and Lambda looks after everything required to execute the code. We can automatically trigger our code from other AWS services.
It is server less, that is, we don't have to manage the server and is auto triggered. Lambda is used for file movement from one location to another.

Chapter-3

SYSTEM DEVELOPMENT

The essential priorities of the design are-

1) Functionality
2) Scalability
3) Maintainability
4) Security
5) Privacy
6) Reliability
7) Interfaces

3.1 Functional Requirements:

- Database must be centralized and well maintained.
- Employees should be able to create their Client account using employee ids. Employees can use the respective AWS services using their client employee id and passwords.
- Client ids for every employee is created by verifying their credentials. And they are provided the access for certain platforms and functionalities based on their roles and requirements.
- **Vendor access must be limited** -Vendors should be able to use the data by using valid credentials. But they must have limited data accessibility. They can't look into the C-SCAN buckets.
- **Admin access-** Not everyone has the access to make changes in the system layout or architecture model. Only certain employees are provided with such access. And only they can perform such operations.

3.2 Non-Functional Requirements:

- Back up of data stored in data servers.
- System should have enough security measures to restrict unauthorized user access.

- In case of any failure appropriate error message should be displayed.
- Data processing should be smooth and fast.
- PI and NPII data, that is, sensitive, confidential data and Non-sensitive data respectively.
- Commercial Data warehouse and Enterprise Data warehouse.

3.3 Software Requirements:

- AWS services
- Erwin
- Citrix- A client application which provides software accessibility to various software like SQL workbench, Aginity, Putty, EC2.
- Windows 7, Windows 8 or later

1.4 Hardware requirement:

- A segment of AWS server and data repository is owned by ZS's client, where the entire data is stored.
- 1.6 GHz or faster processor.
- Large GB of hard disk space.

Chapter-4

PERFORMANCE ANALYSIS

4.1 Project Development

4.1.1 Data Flow Diagram

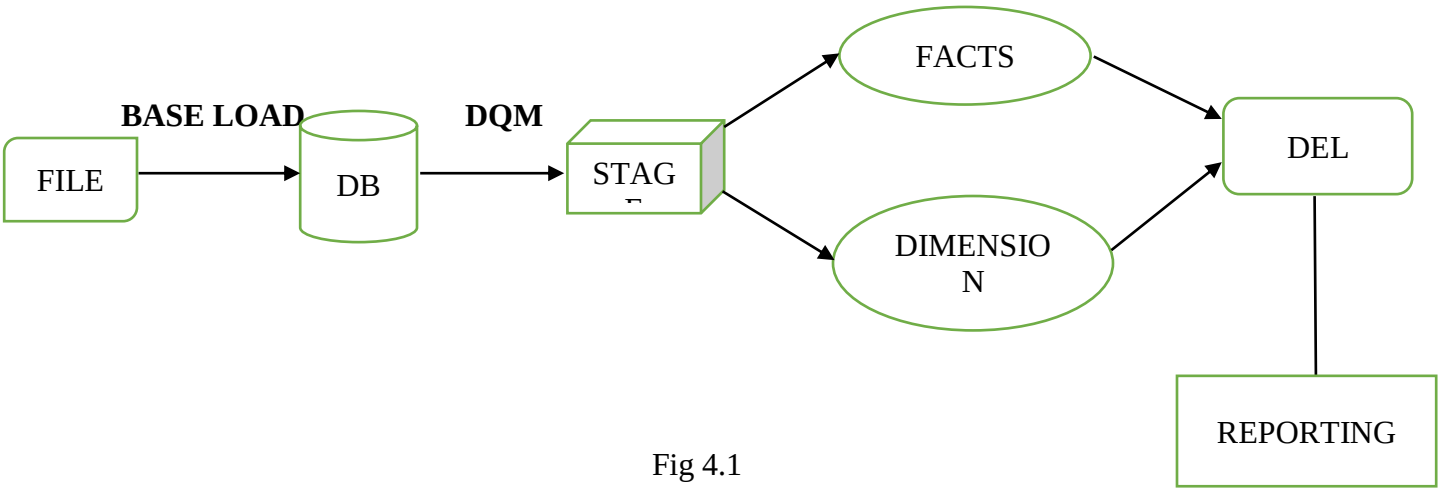


Fig 4.1

4.2 Functionalities

AWS services Processing:

- **S3:** It is used for file storage purpose. Vendors moves all files to S3 in different formats (like .txt, .csv).
- **Redshift:** OLAP database, it is the reporting tool of ZS.
- **RDS:** OLTP database, Process scheduling, ensures what process should occur when and after which process.
- **SNAPLOGIC:** ETL tool.

C-SCAN moves the data from S3 to Redshift.

S3:

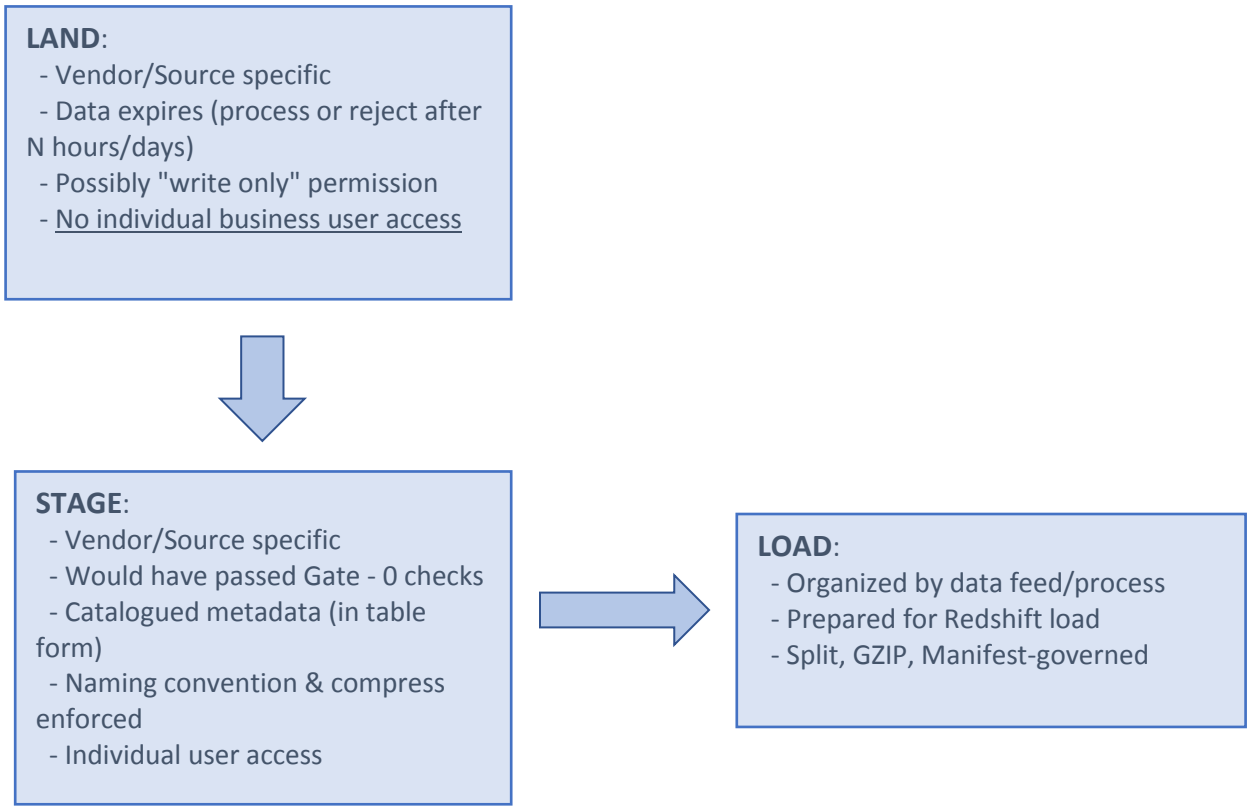


Fig 4.2 Data consumption flow diagram

S3 bucket used in C-SCAN-

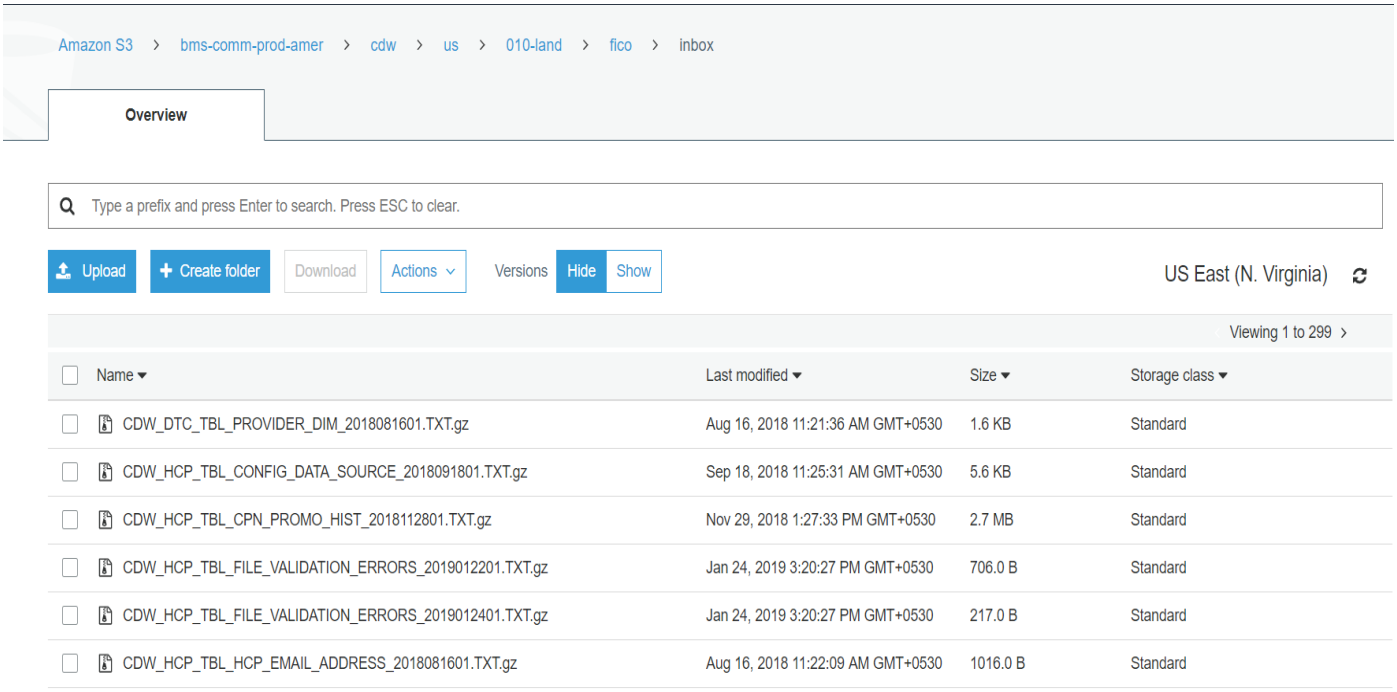


Fig 4.3 S3 bucket

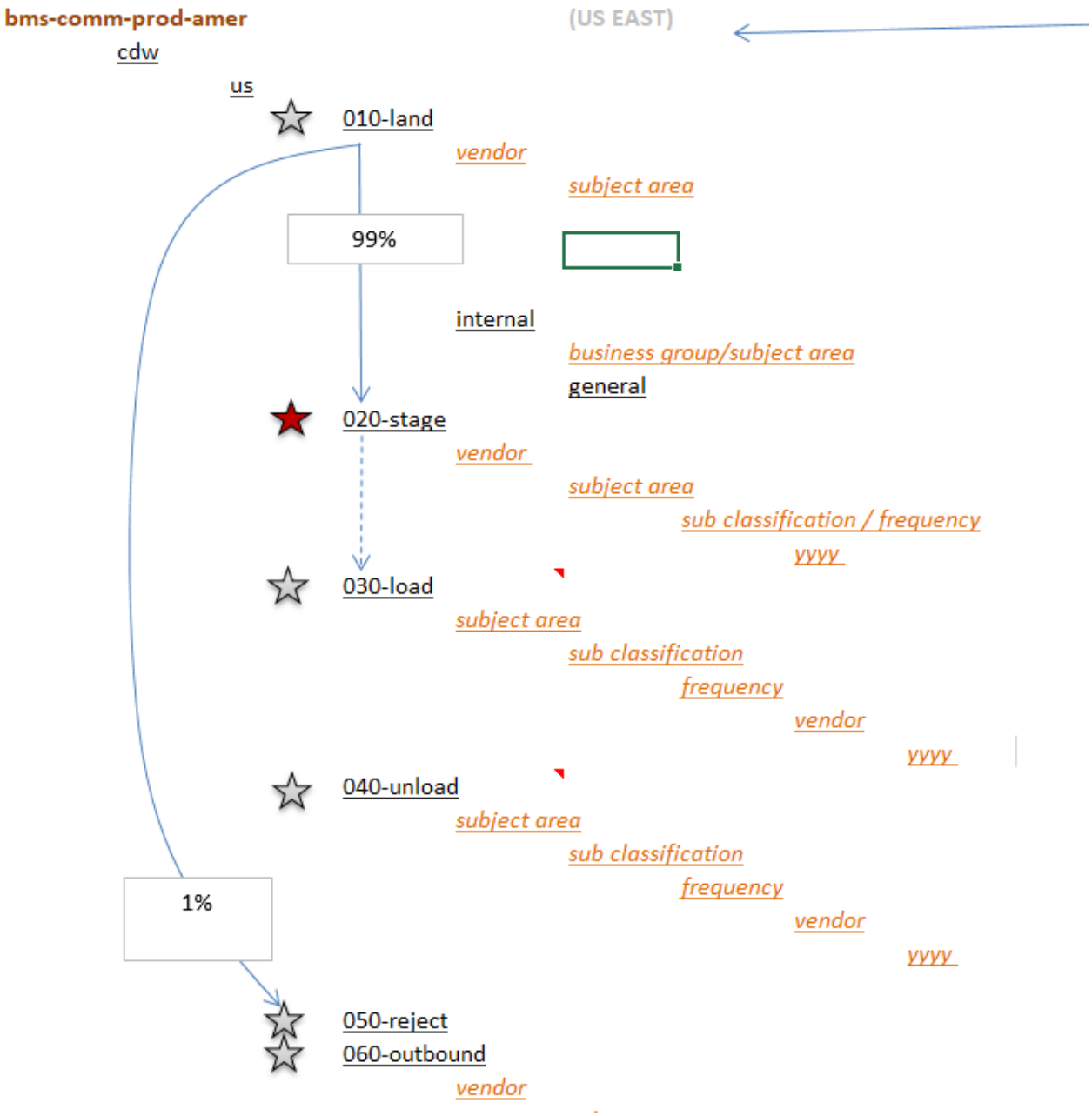


Fig 4.4 Data flow in S3.

FINDER AND DOWNLOADER:

Data enters C-SCAN by 3 ways:
S3, SFTP, and Database pull.
If the files are present at their respective S3 and SFTP locations, FINDER finds the file so that they can be consumed into C-SCAN tables.

Working of finder/downloader:

Step 1:

- Make entries in cntl_src_mast, that is, Entries in all the master tables.

Step 2:

- Make entries in cntl_mast_pri
- Based on priority finder picks the highest priority file first.

Step 3:

- Put the file on S3.

Step 4:

- Finder will start working.
- Finder goes to cntl_mast_pri table and picks subject area corresponding to the highest priority record.
- Finder goes to cntl_src_mast table and fetches all the files corresponding to that subject area.
- Finder goes to cntl_file_ftp_log and compares the picked file name against the present file names.
- If the file is identified as duplicate; finder marks it as a duplicate else, it is marked as a new file.

Step 5:

- Downloader will start working.
- If the file is present in sftp then it will be transferred to S3 land folder.
- File is transferred to stage for processing via a job. The file gets renamed("src_file_modf_name").
- DQM checks are applied.
- cntl_job_log will have data of batches, sub-batches and respective jobs.
- cntl_job_dep will have the job id and dependent job id. It also has the dtst id as file list. First of all, it will take the independent job id, execute it and then moves to the job_id dependent on it.
- After the completion of finder and downloader status of the file is updated in file log table.

Different file status and their meaning:

file_ftp_log		
0	File found	Finder
5	Duplicate File	Finder
10	New File	Finder
2	readme/trigger file is present	Finder
30	File in S3(all files are downloaded)	Downloader
40	Ready for processing	Job
50	L1 automation #directly base gets loaded & we process thereafter	Job

Downloader makes the status of the files '30' after finding them at S3.

At S3, the raw files are present at LAND, which is also known as vendor bucket. After that the file moves to STAGE folder through a LAND to STAGE job.

For further processing we move the tables from LAND to STAGE and rename the actual file name to a modified name.

Once the files move to STAGE folder, BASE load comes into process which loads the data into the C-SCAN tables through another job. Once the data is refreshed into BASE tables, DQM's are applied. The rejected data moves into a separate set of tables called the base_hist tables and the corrected data moves into the staging tables for further processing.

After that MDM and business tools are applied to integrate the data into Level-2 tables, business rules are applied according to the client's needs, so that they can get the desired data.
Views of tables are created to provide access to multiple teams for analysis.

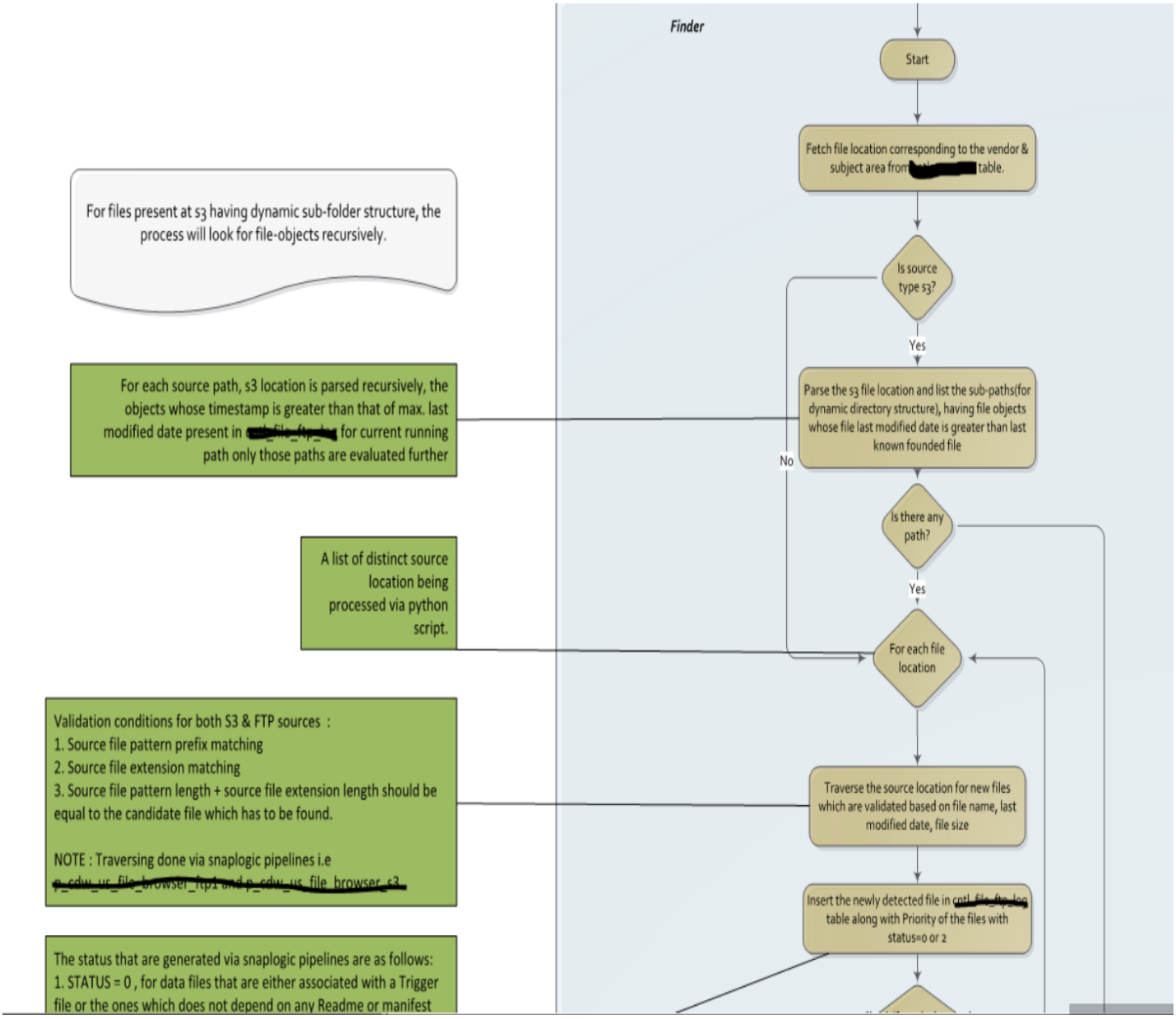


Fig 4.5(a) Finder flow chart-1

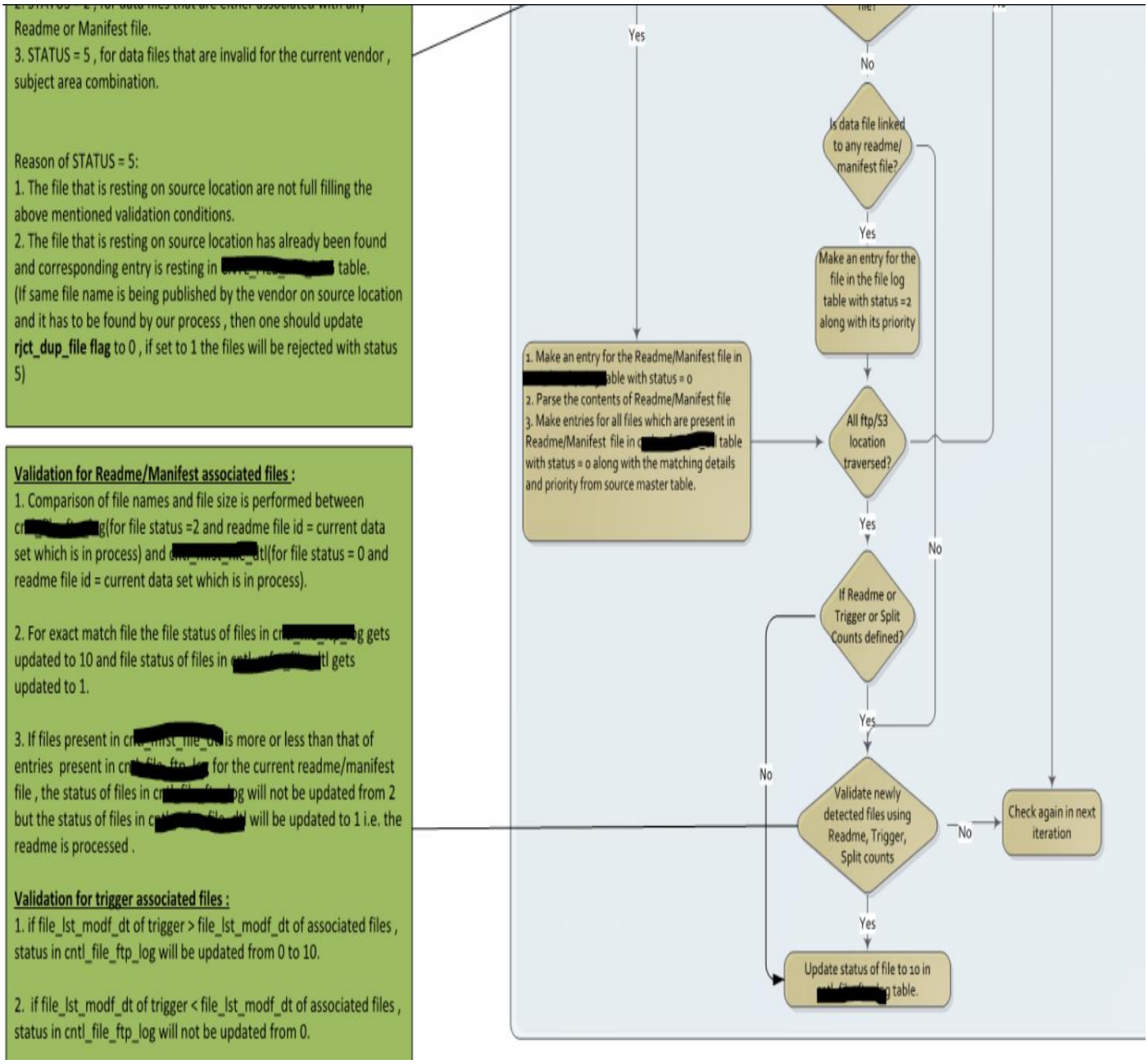


Fig 4.5(b) Finder flow chart-2.

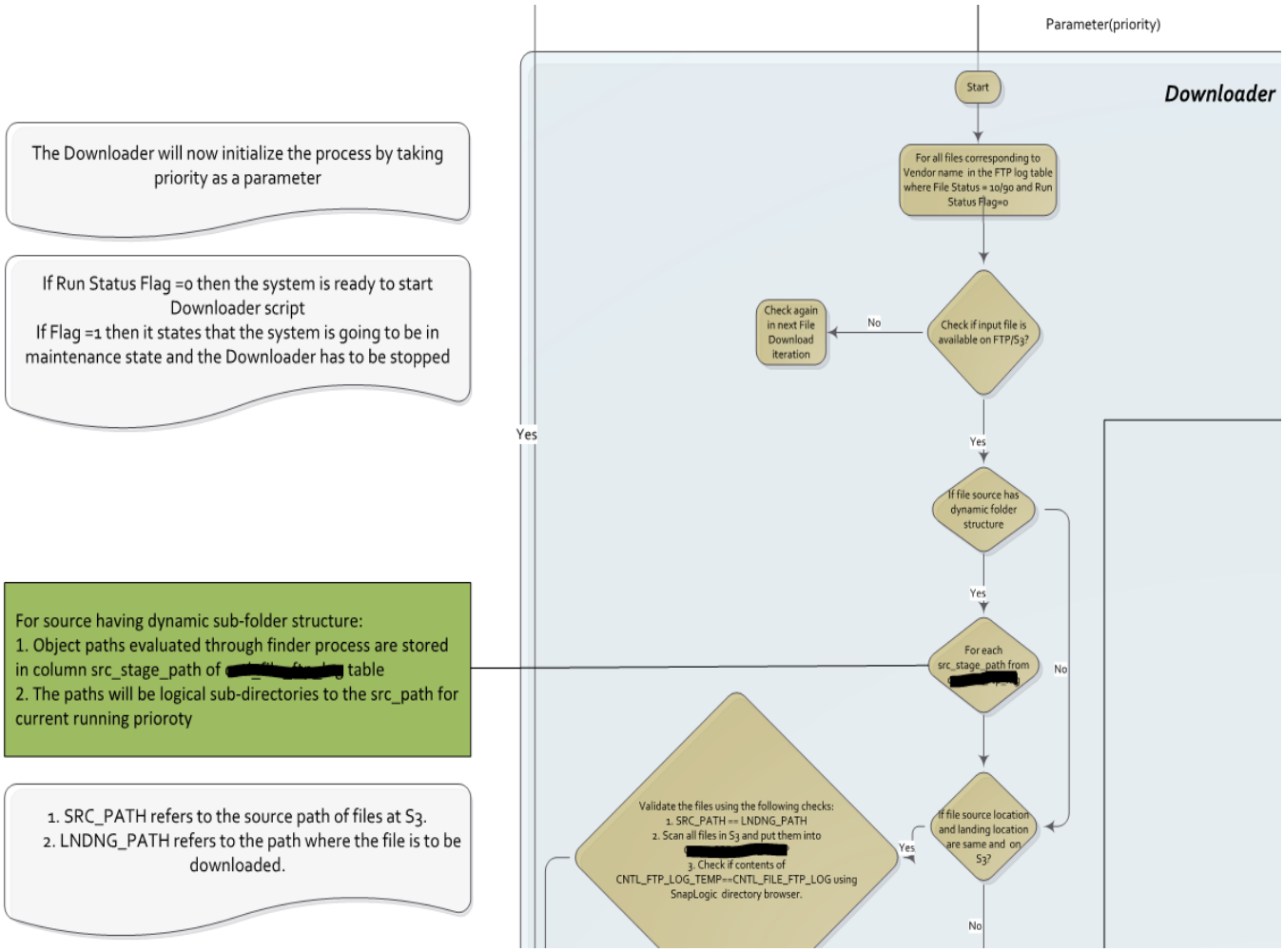


Fig 4.6(a) Downloader flow chart-1.

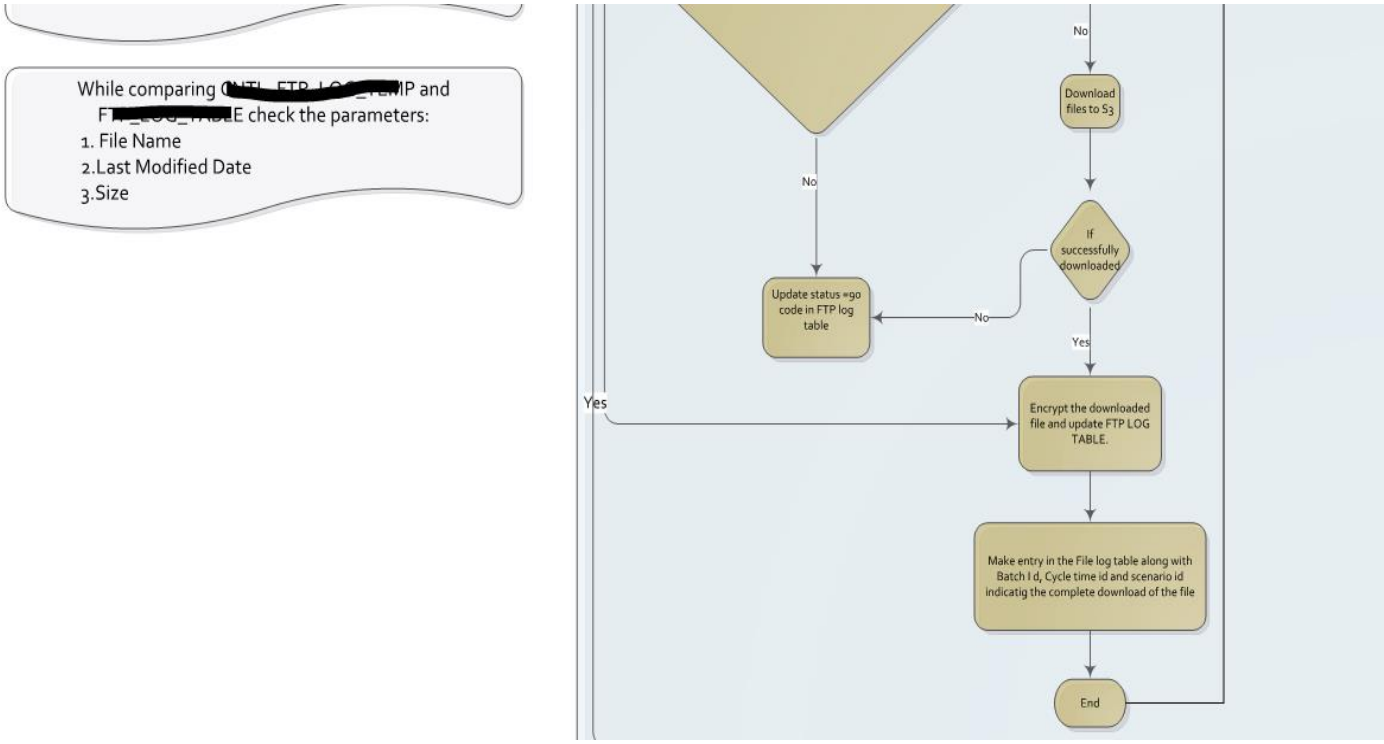


Fig 4.6(b) Downloader flow chart-2.

Finder/Downloader follows ‘Round Robin’ and ‘Queue Management’ approach.

Vendor Extracts:

Vendor extracts are exported data from the commercial data warehouse database in various formats. These are customized as per business requirement. They are based over C-SCAN Level 2 and Level 3 data. The users for these extracts are mostly teams who are specialized in a specific market and have a
Sources for which file movement doesn’t occur through ‘Lambda’, extracts are generated for those who are not using C-SCAN and uses our output as input.
In vendor extracts we pick the data from the database, makes a file of it and puts it on S3/SFTP according to the user’s accessibility.

Facts & Dimensions:
Facts contains transactional data, i.e, sales etc. Therefore, it is fast changing as sales changes on a daily basis.
Facts contains foreign keys whereas ‘Dimensions’ contain real time objects or entity.
Dimensions are slowly changing as compared to facts.

SLOWLY CHANGING DIMENSIONS:

Slowly Changing Dimensions (SCD) – These are dimensions which changes at a very slow pace over time, instead of changing regularly.

Some common SCD's used are:

- **Type 0** - Passive method
- **Type 1** - Overwriting the previous value
- **Type 2** - Creating an additional record
- **Type 3** - Adding a new column

- **SCD-0**:
  These are one-time tables; no change occurs once they gets created.
  No specific actions are performed. Some dimensions remains the same as they were first time inserted, others might be overwritten.

  Eg)-

  old value

Pres_id	Zip
AC	1001

  New value

Pres_id	Zip
AC	1002

  Modified value

Pres_id	Zip
AC	1001

- **SCD-1:**
  These are known as 'No History' dimensions, values gets directly updated. New dimensional values are overwritten on the previous dimensions. These are easy to maintain. They are used for data where changes are caused by processing corrections

Eg)

old value

Pres_id	**Zip**
AC	1001

New value

Pres_id	**Zip**
AC	1002

Modified value

Pres_id	Zip
AC	1002

- **SCD-2**:

These are the unlimited history dimensions.
Entire history of dimensions are kept in the database. Attribute changes can be identified by adding a new row with a new surrogate key to the dimension table. The columns used here to identify the history record are 'effective date' and 'current flag indicator'. There can be only one record with current indicator set to 'Y', i.e active record.

For 'effective-date' columns, that is, start date and end date, the end date for recent record is normally set to the value 9999-12-31.

Before change:

Cust_ID	**Cust_Name**	**Cust_ZIP**	**Strt_Date**	**End_Date**	**Crnt_Flag**
1	Cust_1	1001	22-07-2018	31-12-9999	Y

After change:

Cust_ID	**Cust_Name**	**Cust_ZIP**	**Strt_Date**	**End_Date**	**Crnt_Flag**
1	Cust_1	1001	22-07-2018	17-05-2019	N
2	Cust_1	1002	17-05-2019	31-12-9999	Y

Eg)

Pres_id	**Zip**	**Strt_dt**	**End_dt**	**Actv_flag**
AC	1001	1-JAN	10-MAR	N
AC	1001	10-MAR	-----	Y

- **SCD-3**:

These are the limited history dimensions. In this only the current value (recent) and the previous(old) value of the dimension is kept in the database. The recent value is loaded into 'current' column and the previous value is stored in 'previous' column. This is the least commonly used technique.

Before change:

Cust_ID	Cust_Name	Crnt_ZIP	Pre_ZIP
1	Cust_1	1001	1001

After change:

Cust_ID	Cust_Name	Crnt_ZIP	Pre_Type
1	Cust_1	1002	1001

Type 4 – Uses historical table. In this, a historical table is used to keep track of all dimension's attribute historical changes for each of the dimension. The 'main' dimension table keeps only the current data.

Current table:

Cust_ID	Cust_Name	Cust_ZIP
1	Cust_1	1001

Historical table:

Cust_ID	Cust_Name	Cust_ZIP	Strt_Date	End_Date
1	Cust_1	1003	01-01-2010	21-07-2010
1	Cust_1	1002	22-07-2010	17-05-2012
1	Cust_1	1001	18-05-2012	31-12-9999

Type 6 – Type-6 includes the approaches of types 1, 2 and 3 (1+2+3=6).

Customer_ID	Customer_Name	Current_ZIP	Historical_Type	Start_Date	End_Date	Current_Flag
1	Cust_1	1001	1002	01-01-2018	21-07-2018	N
2	Cust_1	1001	1003	21-07-2018	17-05-2019	N
3	Cust_1	1001	1001	17-05-2019	31-12-9999	Y

4.3 Testing Analysis:

DQM (DATA QUALITY MANAGEMENT):

Overview of DQM:

Common/Base Checks applied to all files

Standard quality control checks that data provider must perform to quality check the input data feeds every month.

Common/Base Checks:
•Transport Checks: Validate that number of records and control totals match up with the control file
File Layout Checks: E.g., columns are in correct order, correct delimiter, new line sequence (\n) record length.
•File Inventory Checks: Verify that all required files are part of deliverables.
•File names match agreed upon naming conventions.
Each file should have the "month_id" value = Current Data month.

Note: All of the above checks are expected to be applied to all data files and are considered to be high priority quality checks.

QC Type	Definition	Example(s)	Variables for which this check	
			Categ-orical?	Conti-nuous
Length check	Validate that the length of each value for a given field is as	Length of Customer ID is 7	☐	
Data Type check	Validate that the data type for each value for a given field is as expected	Customer ID has a format of 3 characters and 4 numerics (3a4n)	☐	☐
NOT NULL check	Validate that the field that is defined as "NOT NULL" has no missing	All records in the Customer ID column	☐	☐
Domain Check	Validate that the value of a given field lies in a pre-defined domain (i,e. a known list of values)	Rx_type can be in (N,T,B) only or First Name can not be (aaa,	☐	
Range check	Validate that the value falls in a	All sales fall within $0	☐	☐
Referential integrity check	Validate that values of interest exist in a master file or master table	Customer ID “jlf1234” exists in the customer	☐	
Transport check (with control total file)	Validate that the number of records and control totals match up with the control total file	# records in the sales file is 1,000 and sales add up to $1 MM	☐	☐
Uniqueness check	Validate that there are no duplicates for the given	Sales data is unique by customer and	☐	
Comparison Check (Control charts) (Variation in # Records, Adds, Drops)	Validate that the # records or unique count of values fall within the control limits	# Unique Customer IDs must fall within the control limits	☐	

Fig 4.7(a) DQM checks.

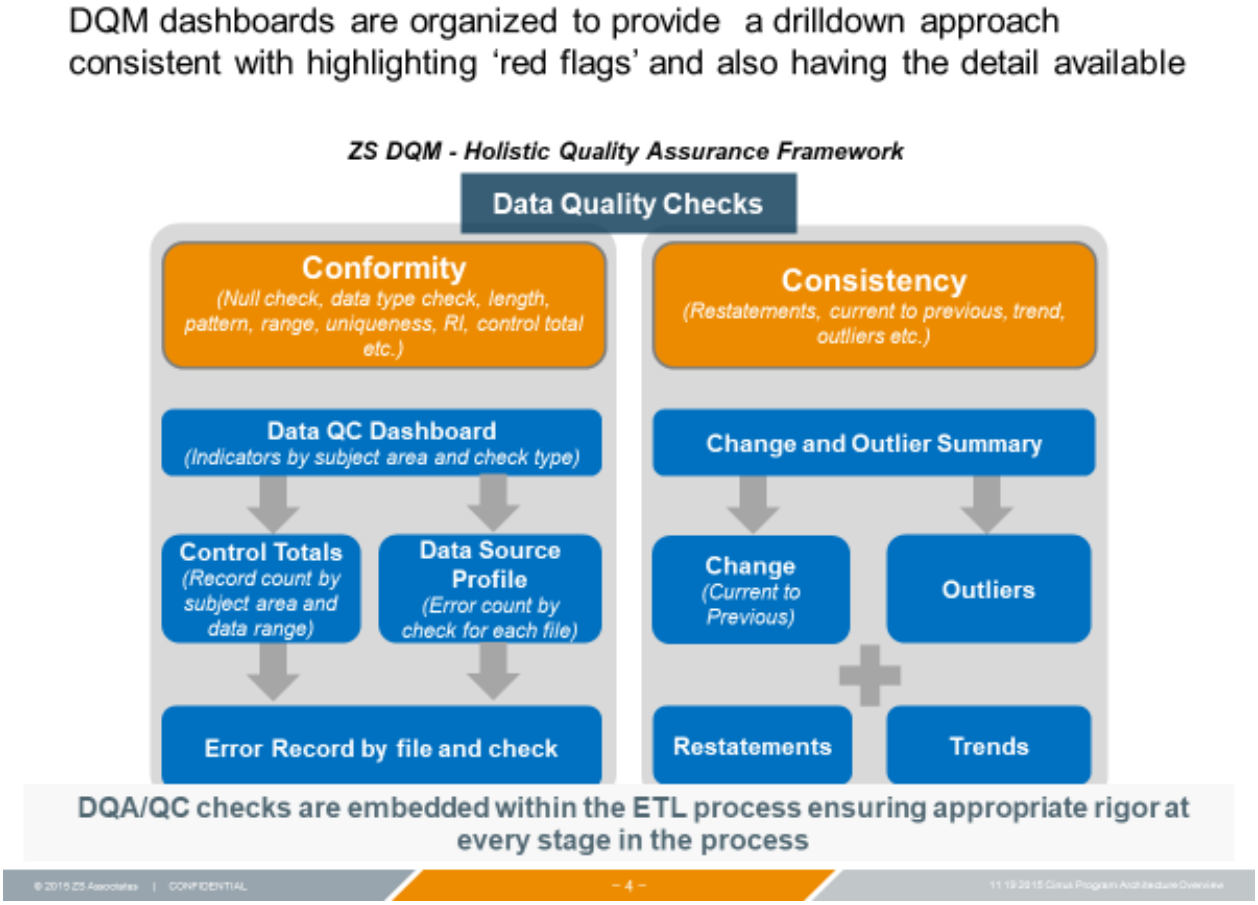


Fig 4.7(b) DQM checks.

Consistency checks are very imported for critical data like the **sales** of any drug. Restatements and outlier checks are implied for such data and are very crucial for roll up data.

*** Roll-up data:** If data is from week 1-10 and a new record is added, say week 11, then the new data available will be of week 2-11. This type of data is said to be roll up data.

* **Outlier check:** Say we have sales for a product as follows:

Month	JAN	FEB	MAR	APRIL
Sales	$500	$450	$570	$100

If we look at the trend, we are observing a drop or gain of $50-100. But for April we are seeing a drop of $400 which is unexpected. In such cases, DQM gets failed and this known as outlier check

There are various data quality checks that ensures quality management.
For example- NULL check; data length check; data type check etc.

In C-SCAN, if there is DQM failure we can identify it by looking into the qc_mast table and identifying the qc_id of the rejected record. We can check for the type of failure also by looking at the qc_type_id, for example- NULL check; data length check.
We can get the incorrected table name and the corresponding column name by looking into the qc_mast table.

There are three types of threshold indicators in C-SCAN that gives insights into the DQM failure, namely,

- **Red Threshold**
- **Yellow threshold**
- **Criticality**

Red threshold:

if the red threshold is active, i.e, 1 then it rejects the whole file. The entire process stops, and no data enters the tables. If the red threshold is 0 then it moves to the yellow threshold.
Eg) column is NULL, which is not supposed to be NULL, then the entire file is rejected.

Red Threshold	Yellow threshold	Criticality
1	0	0

Yellow Threshold:

If yellow threshold flag is active, i.e, 1 then it checks the criticality.

1. If criticality is also 1 then it rejects that record and sends it to the vendor, bad records drops, and process goes ahead.

Red Threshold	Yellow threshold	Criticality
0	1	1

2. If criticality is 0 then it loads the data in stage without dropping the record and file goes ahead.

Red Threshold	Yellow threshold	Criticality
0	1	0

Phase_id:

It allows us to know where exactly has the DQM failed. That is, at base level, stage level or at Level-2.

NOTE:
Records that are rejected by yellow threshold and criticality are sent to the base_hist tables.

C-SCAN consists of various databases, namely,

- **REDSHIFT**: Data-warehouse, user based
- **RDS-ETL:** RDS aurora
- **RDS-META:** Orchestration, PostgreSQL

RDS is row-wise storage whereas Redshift is columnar storage. Columnar storage is better than row-wise storage because there are less number of I/0. Therefore, less data enters the RAM.

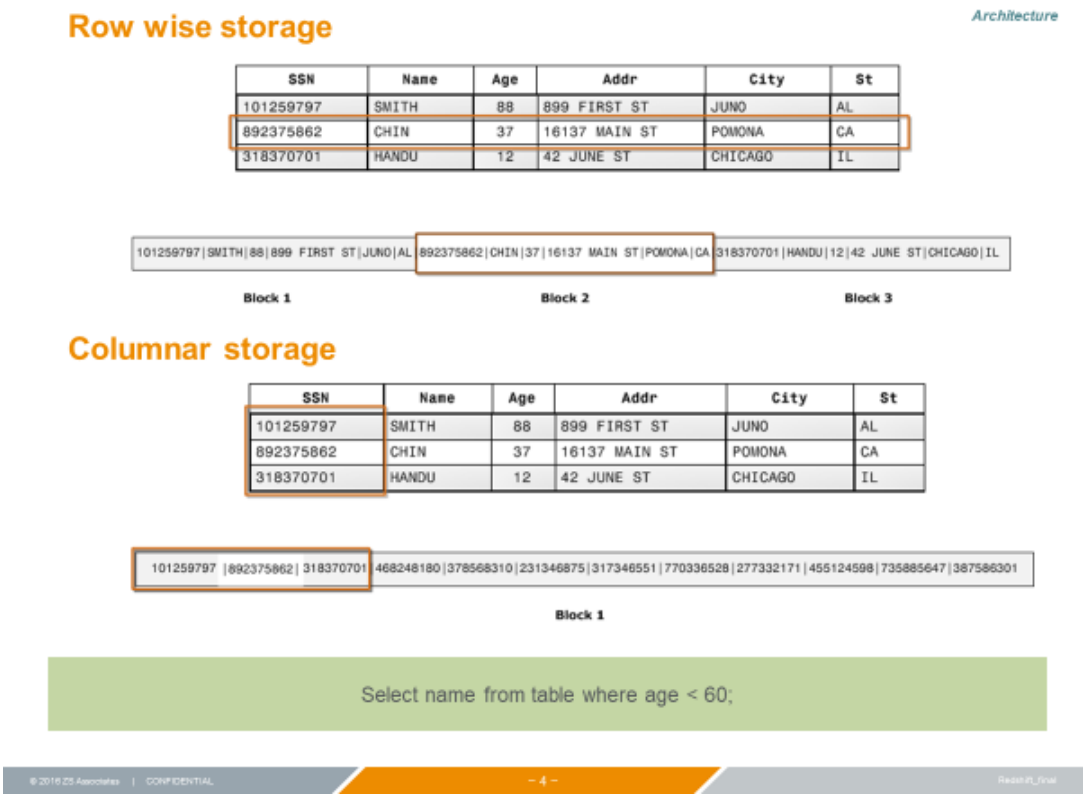


Fig 4.8 Redshift Block storage.

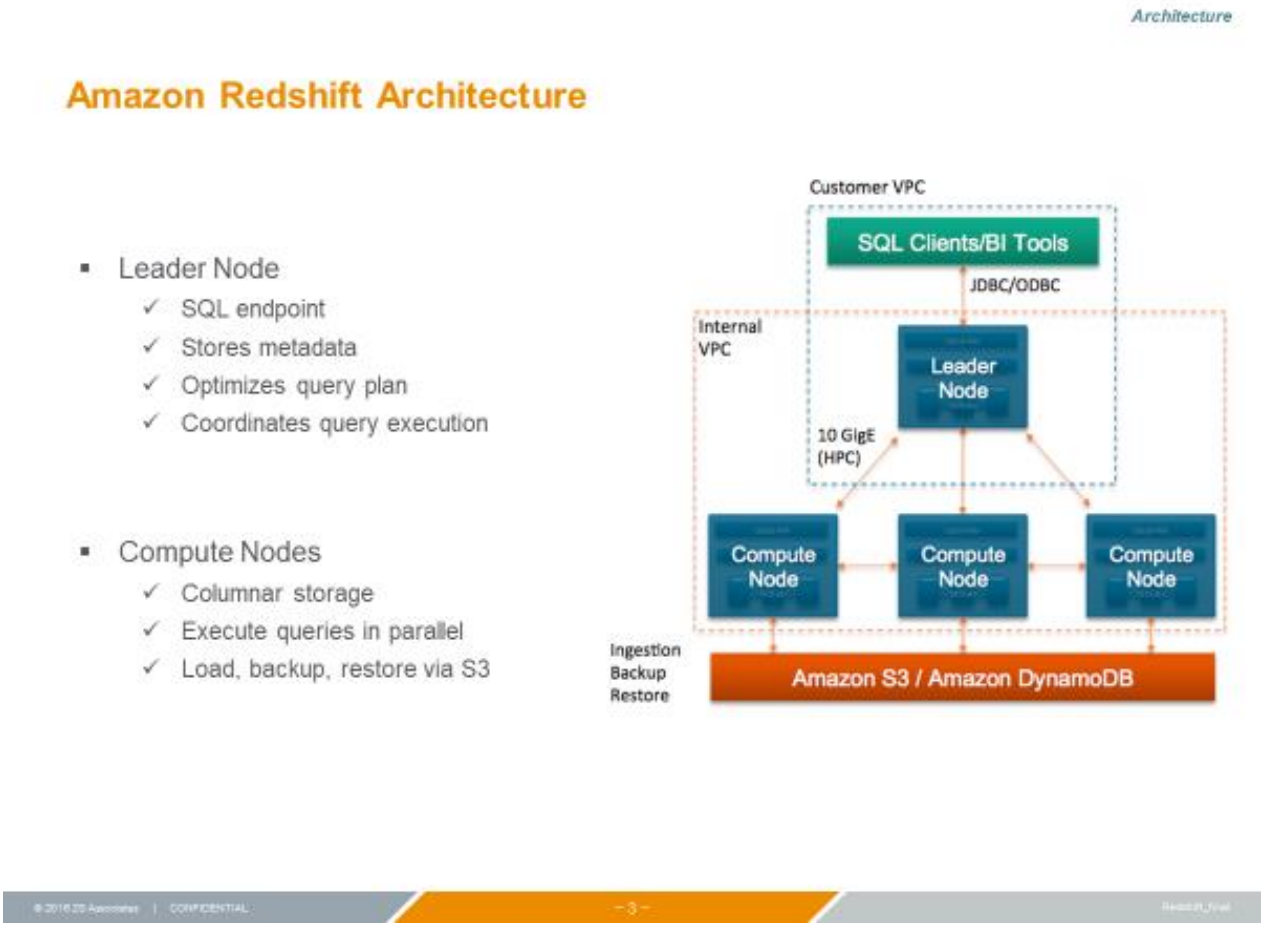


Fig 4.9 Redshift architecture

Data is always stored in Compute Nodes(CN) and is computed there only. CN works independently of each other. Leader nodes(LN) aggregates the computed data stored in the compute nodes and reverts it to the client. This framework is much like the Master-slave framework. Multi parallel processing takes place in Redshift.

C-SCAN uses two types of Redshift clusters:

- Data storage(DS) – for NPII cluster
- Data compute(DC) – for PII cluster

DC is faster than DS as it is compute optimized.

RULE ENGINE:

Snap-logic Pipeline:

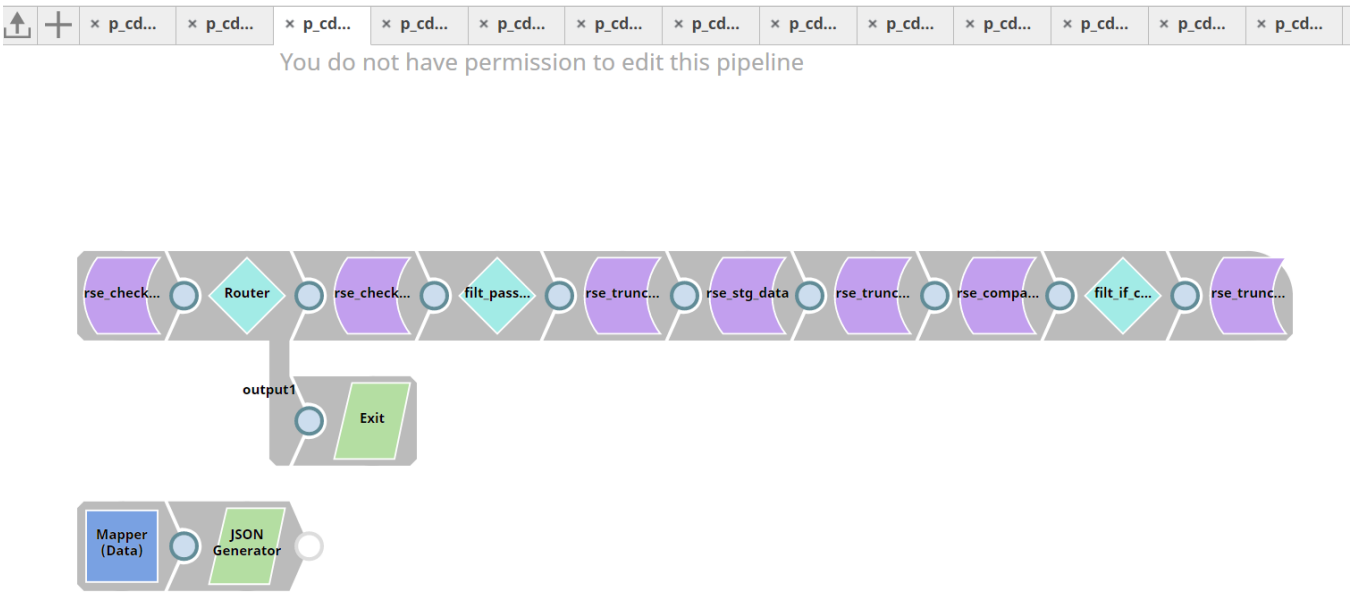


Fig 4.10 Snap-logic pipeline snap

Entire processing is held here. All the pipelines run at the back end through the rule engine.
We have three main tables in the rule engine –

1 "**cntl_rule_type"**
2 **"cntl_rule_param"**
3 **"cntl_etl_sql"**
The whole description of the pipelines and the rule engine is mentioned in "**cntl_rule_type".**
We can get the rule number for any source, for any job from here.

"cntl_rule_param" defines the sequence of jobs. It ensures what job should be executed at what time. Then it moves to **"cntl_etl_sql"** where 'pre-SQL' checks are applied.

Majority of the rule engine runs on Redshift, whereas all log tables and orchestration run on RDS.

Data Consumption flow:
The data is consumed through a number of levels. They are-

When a file is loaded into C-SCAN, the data goes through certain levels.

- **L-0 (LEVEL 0)**
- **L-1 (LEVEL 1)**
- **L-2 (LEVEL 2)**
- **L-3 (LEVEL 3)**
- **VENDOR EXTRACTS**
- **DEL(REPORTING)**

To ensure entire data is loaded into the database we define larger data types for huge data. At staging, we change the generalized data type into actual data types.
Eg) varchar (20) at land and varchar (10) at stage.

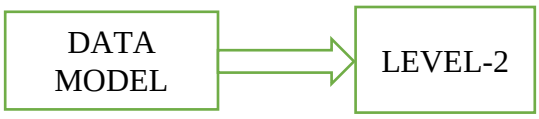
L-0 (LEVEL 0): Here C-SCAN acts as a mediator. The data is not consumed in C-SCAN, but it is moved from one location to another, that is, from one source to another.
L-1 (LEVEL 1): Data processing takes place at this level, where data is refreshed into C-SCAN tables through 'BASE' and 'STAGE' loads.

- **BASE** contains the actual raw data received from different vendors as it is.
  BASE is truncate data, after base send data to stage, base data truncates.
- **STAGE** contains the processed and arranged data, i.e, refined data after DQM checks.

The raw is loaded into the BASE tables as it is without any modifications. After the base load gets completed the data moves to the STAGE after passing certain 'Quality Checks'. For the records which gets failed in the data process, they move to base history table which contains all the rejected records.

If a file contains 100 records out of which 10 gets failed due to DQM failure, then those 10 records move to base-hist tables which contains all the rejected records. And the remaining 90 records moves to the STAGE.

L-2 (LEVEL 2): is the level at which the actual data warehouse exists. Certain necessary 'Business rules' are applied at this level to make the data meaningful for the users as well as for C-SCAN itself. After the rules are applied successfully the data flows to the 'Facts' and 'Dimension' tables. 'MASTER DATA MANAGEMENT' comes into picture.
Every staging table has its corresponding fact and dimension table.



Data model are of two types:

- **LDM (LOGICAL DATA MODEL)**
- **PDM (PHYSICAL DATA MODEL)**

LOGICAL DATA MODEL contains the table structure with logical name.

PHYSICAL DATA MODEL contains the actual column names and data types.

STTM: It stands for Source to target mapping. It tells us how to put data from stage to Level-2.

NOTE: Level-2 tables are not truncate tables whereas level-3 tables are.

L-3 (LEVEL 3): It is based on micro-strategy, it pulls data from C-SCAN. The data after going through C-SCAN tables at different levels and passing the DQM checks enters the L-3 tables. Further, additional Business rules and decisions are applied at this level according to the user requirements and specification. The data is modified according to what the client or users wants; what they want to see from the given set of data for further analysis.

Vendor extracts are generated from both L-2 and L-3 tables. They help the users to do analysis at their end and generate meaningful information.

DEL Reporting- Reporting tables are according to the customer demand. Once the data is refreshed completely in C-SCAN it is reported back to the clients and users.

RESULT

After having multiple meetings with the client and the vendor, we came to an agreement about how the files will be delivered to C-SCAN. We applied all the necessary DQM checks and other quality checks to avoid discrepancies. Data types and the filed capacities of the files were decided to ensure smooth data processing. Tables were created in C-SCAN for the same so that the data in the files gets refreshed in them which can be used for analysis and other business requirements.

Data Source	Dataset Name	Column Name	Data Type	Size	QC Type	Qc Description	Criticality	Yello...
Healthstar	Product Details	Record Type	CHAR	2	Null Check	Field should not be null	0	1
Healthstar	Product Details	HealthSTAR Internal ID	INTEGER		Null Check	Field should not be null	0	1
Healthstar	Product Details	Product Name	CHAR	200	Null Check	Field should not be null	0	1
Healthstar	Product Details	Product ID Primary	INTEGER		Null Check	Field should not be null	0	1
Healthstar	Product Details	Effective Start Date	DATE		Null Check	Field should not be null	0	1
Healthstar	Product Details	Effective End Date	DATE		Null Check	Field should not be null	0	1
Healthstar	Product Details	Status	BOOLEAN		Null Check	Field should not be null	0	1
Healthstar	Product Details	Record Type	CHAR	2	Length Check	Field should not exceed length specified	0	0
Healthstar	Product Details	Product Name	CHAR	200	Length Check	Field should not exceed length specified	0	0
Healthstar	Product Details	HealthSTAR Internal ID	INTEGER		Type Check	Check if data type is having Integer Data	0	0
Healthstar	Product Details	Product ID Primary	INTEGER		Type Check	Check if data type is having Integer Data	0	0
Healthstar	Product Details	Effective Start Date	DATE		Type Check	Check if data type is having Date Type Data	0	0
Healthstar	Product Details	Effective End Date	DATE		Type Check	Check if data type is having Date Type Data	0	0
Healthstar	Product Details	Status	BOOLEAN		Type Check	Check if data type is having Boolean Data	0	0

Fig 5.1 DQM list for Health-Star

File Name	Description/Purpose
Control	Control file sent with each file set, indicating the row count for each file in the file set
Product Details	All Brands/Products (from BMS Master data)
Field Organization	Field Sales Force Organization Nodes and Hierarchy (from BMS Master data)
Field Roster	Field Sales Roster Members (from BMS Master data)
Roster Assignment	Assignment of Field Sales Members to Organizations, by Role (from BMS Master data)
Product Alignment	Alignment of Field Sales Organizations to Products (from BMS Master data)
Customer Alignment	Alignment of Field Sales Organizations to Customers/Targets (from BMS Master data)
Home Office User	Home Office Users enabled in HCP Connect, by Role
Activity Type	Master List of Activity Types
Activity Format	Master List of Activity Formats
Activity Delivery Type	Master List of Activity Delivery Types
Need Details	Need Assessment details as entered in HCP Connect
Initiative Type	Master List of Initiative Types
Initiative Details	Initiative details in entered in HCP Connect
Initiative Brand	List of Brands defined for each Initiative
Initiative Topic	List of Topics defined for each Initiative
Initiative Audience	Audience Member(s) allowable for this Initiative
Initiative Speaker Types	Allowable Speaker Types for this Initiative
Topic Details	Master list of Topics in HCP Connect
Topic Product	Topic-to-Product relationships
Speaker Type	Master list of Speaker Types
Speaker Details	Speaker details as stored/managed in HCP Connect
Speaker Contract	Speaker Contract details as stored/managed in HCP Connect
Speaker Contract Product	Speaker Contract -to-Product relationships as stored/managed in HCP Connect
Speaker Training	Speaker Training records in HCP Connect (Speaker-to-Topic)
Speaker Cancel Fees	Speaker Cancellation Fees by Contract and Meeting Format Type
Activity Details	Activity details as stored/managed in HCP Connect
Activity Cost Type	Master List of all Cost/Expense Types
Activity Speaker	Details for Speaker(s) for an Activity as stored/managed in HCP Connect
Activity Host	Details for Host(s) for an Activity as stored/managed in HCP Connect
Activity Costs	Details for all Activity Costs as stored/managed in HCP Connect
Activity Check	Details for all Activity Checks as stored/managed in HCP Connect
Activity Attendee	Details for all Activity Invitees/Attendees as stored/managed in HCP Connect
Activity Survey	Activity Survey details as captured in HCP Connect

Fig 5.2 Files received from Health star with description

File Name: bms_hs_product_details_yyyymmdd.txt

Field Name	Description	Relationship	Field Type	Required?	Max Length	Sample Value
Record Type	Identifies the file type for this record (see Overview)		CHAR	Yes	2	10
HealthSTAR Internal ID	HealthSTAR ID for this record	Primary Key	INTEGER	Yes	10	3
Product Name	BMS Product Name (from CPM)		CHAR	Yes	200	REGIMEN (OPDIVO + YERVOY)
Product ID Primary	BMS CPM Product ID		INTEGER	Yes	10	2008928
Product ID Secondary	BMS GPDV Product ID		INTEGER		10	9000000052
Effective Start Date	Effective Date range for this record		DATE (MM/DD/YYYY)	Yes		9/30/2015
Effective End Date	Effective Date range for this record		DATE (MM/DD/YYYY)	Yes		12/31/9999
Status	Status of this record (Active, Inactive, etc.)		BOOLEAN	Yes		1

Fig 5.3 Product detail file structure

File Name: bms_hs_customer_alignment_yyyymmdd.txt

Field Name	Description	Relationship	Field Type	Required?	Max Length	Sample Value
Record Type	Identifies the file type for this record (see Overview)		CHAR	Yes	2	24
HealthSTAR Internal ID	HealthSTAR ID for this record	Primary Key	INTEGER	Yes	10	226
Org Node ID	HCP Connect Org Node ID	Foreign Key	INTEGER	Yes	10	367120
Customer ID	BMS Customer Master ID (CMEH BP ID)		CHAR	Yes	20	5179425
Customer Type	Customer Type (Individual vs. Org)		CHAR		20	Individual
Customer Last/Org Name	Customer name		CHAR		100	Jason
Customer First Name	Customer name		CHAR		50	Dazley
Effective Start Date	Effective Date range for this record		DATE (MM/DD/YYYY)	Yes		10/6/2014
Effective End Date	Effective Date range for this record		DATE (MM/DD/YYYY)	Yes		1/1/9999
Status	Status of this record (Active, Inactive, etc.)		BOOLEAN	Yes		1

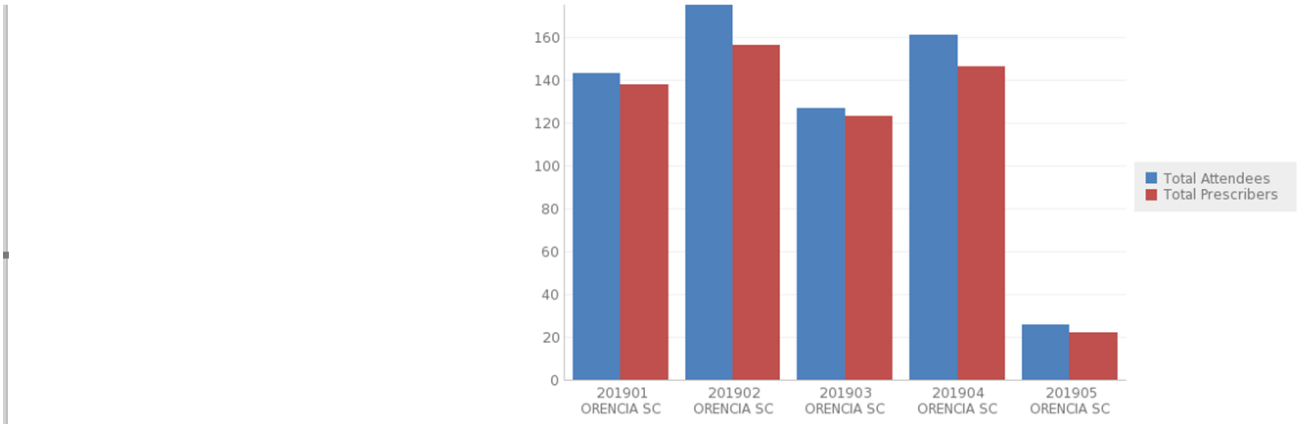
Fig 5.4 Customer alignment file structure

After the data processing is completed till level 2, data is integrated to Level -3 through five tables-

- Event: Data about the event
- Attendee: Data about the attendees like the number of attendees etc.
- Speaker: Data about the speaker, that is, doctor who talks about his/her drug.
- Expense: Data about the expense of the event.
- Speaker Contract: Data about the contract signed by the doctor.

Data from these five tables is generated as extracts, hence five extract files are generated for Health-star.

In the end, all these extracts provides meaningful information that impact the Business in terms of sales of the product, event granularity etc.



Year Month - YYYYMM	Brand	Total Attendees	Total Prescribers
201901	ORENCIA SC	143	138
201902	ORENCIA SC	178	156
201903	ORENCIA SC	127	123
201904	ORENCIA SC	161	146
201905	ORENCIA SC	26	22
Total		635	585

Fig 5.5 DEL Reporting

Chapter-5

CONCLUSION

5.1 Conclusion

It is concluded that the system works well, provides solutions and satisfies the clients need. The system, architecture and the processing are tested very well for various use cases and errors are properly debugged to avoid process failures. The main aim of the project is fulfilled successfully as it reports back to the users or clients to extract meaningful information from the data for further analysis and making strategies.

5.2 Future Scope:

Further advanced business rules can be applied to gain as much information from the existing data. DQM checks can be applied and modified according to the user's demand. Further enhancements can be made in case data layout is expected to get changed in near future to avoid data failure and data mismatch.

References:

- www.zspace.com
- www.amazonaws.cn
- www.datawarehouse4u.info
- http://zsservices.com/
- ZS internal documents