Data Lake Architecture - A Comprehensive Design Document

Medical Data Processing Company

Tracker

Revision, Sign off Sheet and Key Contacts

Change Record

Date	Author	Version	Change Reference
06/04/2020	Francis Odo	0.1	Initial draft

Reviewers / Approval

Name	Version Approved	Position	Date
FirstName LastName	1.0	Technical Product Manager Enterprise Data Lake Architect	06/04/2020

Key Contacts

Name	Role	Team	email
Francis Odo	Data Architect	Medical Data Processing	

1. Purpose

This document presents a comprehensive design of a Data Lake repository system intended for Medical Data Processing Company in response to specific needs and requirements outlined in the problem statement and current growth challenges and problems facing the company. The intended purpose of this document is to clearly articulate a detailed architecture of a Data Lake solution and implementation strategies being presented for recommendation.

The target audience for this material are the high-level executives of the Medical Processing Company, IT Managers, Departmental Heads, Data Stewards and any stakeholders involved with operations and management of customer records. stakeholders.

Scope

This scope of this project includes :

a) Requirements and problem description from the Medical Data Processing Company's data network architecture and needs.
b) Data Lake architecture design principles
c) Assumptions
d) Data Lake Architecture for Medical Data Processing Company
e) Design consideration and rationale
f) Conclusion

Information about Tools or Product components applied in the architecture is outside the scope of this documentation. References will be provided as required where necessary.

2. Requirements

Problem Statement

Medical Data Processing Company continues to experience significant business growth, which is also transforming to various data volume expansion in every aspect of its operation. During the last few years, it has outgrown the capacity limit of a single SQL server system, which is leading to other problems and challenges for routine ETL, Backup FTP and Stored procedures custom scripts for data transformation, exports and transfers.

On top of the overall goal of achieving a modern data infrastructure, the Medical Data Processing Company would like to build additional capabilities with the historical data that company has, for use in building Machine Learning models, and near-real time

dashboards containing patient data for each facility without the need to move the data from one system to another.

Existing Technical Environment

- 1 Master SQL DB Server
- 1 Stage SQL DB Server
    - 64 core vCPU
    - 512 GB RAM
    - 12 TB disk space (70% full, ~8.4 TB)
    - 70+ ETL jobs running to manage over 100 tables
- 3 other smaller servers for Data Ingestion (FTP Server, data and API extract agents)
- Series of web and application servers (32 GB RAM Each, 16 core vCPU)

Current Data Volume

- Data coming from over 8K facilities
- 99% zip files size ranges from 20 KB to 1.5 MB
- Edge cases - some large zip files are as large as 40 MB
- Each zip files when unzipped will provide either CSV, TXT, XML records
- In case of XML zip files, each zip file can contain anywhere from 20-300 individual XML files, each XML file with one record
- Average zip files per day: 77,000
- Average data files per day: 15,000,000
- Average zip files per hour: 3500
- Average data files per hour: 700,000
- Data Volume Growth rate: 15-20% YoY

Business Requirements

- Improve uptime of overall system
- Reduce latency of SQL queries and reports
- System should be reliable and fault tolerant
- Architecture should scale as data volume and velocity increases
- Improve business agility and speed of innovation through automation and ability to experiment with new frameworks
- Embrace open source tools, avoid proprietary solutions which can lead to vendor lock-in
- Metadata driven design - a set of common scripts should be used to process different types of incoming data sets rather than building custom scripts to process each type of data source.
  Centrally store all of the enterprise data and enable easy access

Technical Requirements

- Ability to process incoming files on the fly (instead of nightly batch loads today)
- Separate the metadata, data and compute/processing layers
- Ability to keep unlimited historical data
- Ability to scale up processing speed with increase in data volume
- System should sustain small number of individual node failures without any downtime
- Ability to perform change data capture (CDC), UPSERT support on a certain number of tables
- Ability to drive multiple use cases from same dataset, without the need to move the data or extract the data
    - Ability to integrate with different ML frameworks such as TensorFlow
    - Ability to create dashboards using tools such as PowerBI, Tableau, or Microstrategy
    - Generate daily, weekly, nightly reports using scripts or SQL
- Ad-hoc data analytics, interactive querying capability using SQL

3. Data Lake Architecture design principles

In general, the main design principles and guidelines are based on Apache Spark eco-systems majorly due to its efficient in-memory real-time distributed processing capabilities, flexibility in allowing change in “source and targets” without retrofit, combined with cost-effectiveness.

Open Architecture - The design architecture of Data Lake is an open architecture with no vendor lock-in or restriction to a particular tool. With that principle of openness in mind;

- Data is stored in Avro and Parquet open standard formats that are accessible by most tools applicable to Data Lake. No proprietary format of any type for storing data.
- Amazon S3, the object storage type, is used in the architecture. This will allow the data to be available at all times regardless of the tool being used to access the data. Another advantage is cost savings compared to storing data in Data Warehouse, which tends to be more costly.

Ensure High Performance - Performing data analytics on stored data is resource intensive. Among the steps taken to ensure high performance are;

- Stored data are packaged with the Metadata necessary to understand the structure of the data for fast and efficient retrieval.
- Apache Parquet and ORC file formats are supported for use of columnar. With columnars formats, you can query only the columns needed and avoid scanning redundant data.

Accommodate user needs and skillset - A Data Lake is intended to store different types of data from various sources in a central repository for use by any department of the

organization for various purposes, including yet to be discovered. Following that principle;

- Creating an efficient data pipeline will ensure any of the data consumers have access to the data using the tools they currently have in operation and familiar with, without relying on manual efforts by data providers for every new request.
- The Data Lake architecture offers the capability to store multiple copies of the data for different use cases and consumers. Automated ETL pipelines ingest raw data and perform relevant transformations per use case.

4. Assumptions

1. The implementation of the Data Lake will reside in the cloud environment to reduce the complexity of hardware and software requirements and configuration, as well as overall cost.
2. The cloud vendor is Amazon AWS
3. Hardware platform is Amazon AWS EC2
4. Storage is Amazon AWS S3
5. Operating System is Linux
6. Application Software Platform for processing is Apache Spark

Key questions to which answers are needed

1. Medical Data Processing Company's budget for the new architecture
2. Cost analysis for the new Data Lake design and implementation
3. Training for data stewards and other personnels
4. Testing and roll-out plan
5. Medical records specific regulatory policies and guidelines related

Risks

a) There is the risk of Data Lake becoming a dumping ground or congested with data that the organization may not know what to do with.

   Organizations should work on providing easy access to enhance meaningful analytics by data consumers.

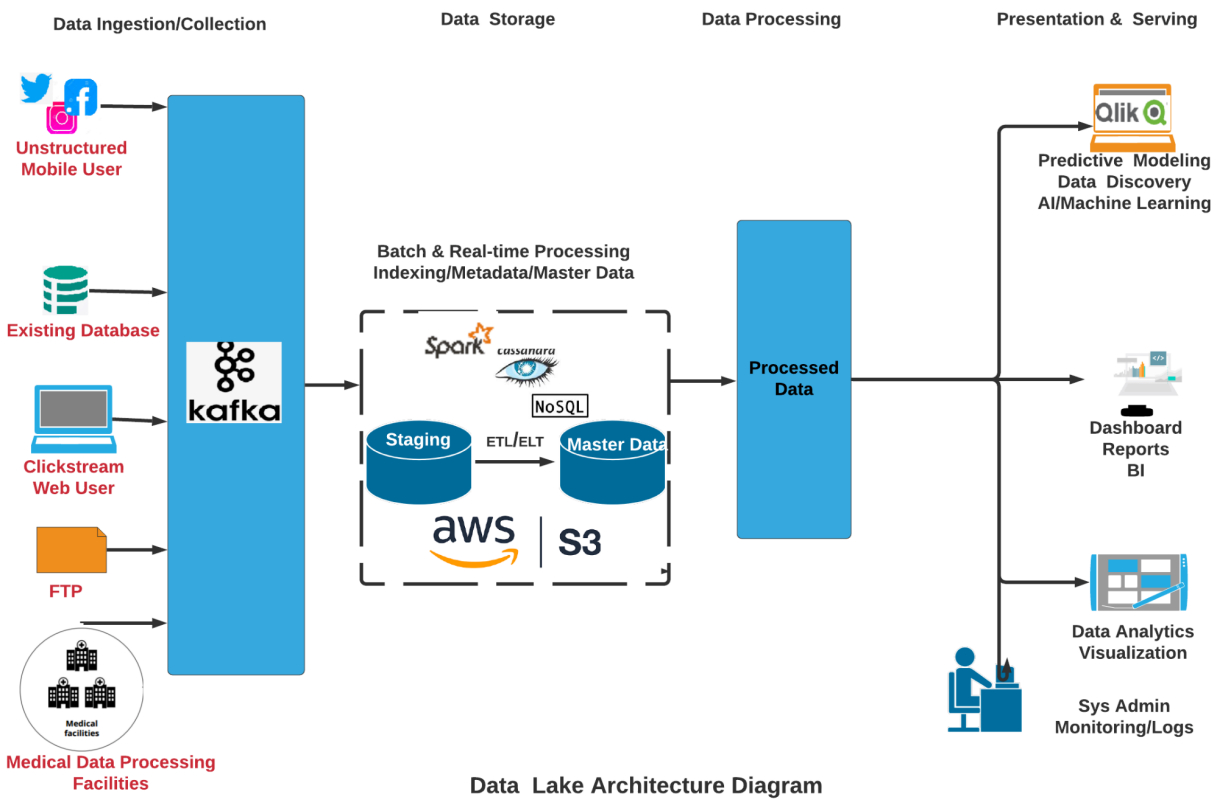
b) Overwhelmingly required knowledge or technical know-how at the early stage of migration tends to slow the progress.

   Organizations should plan ahead and invest in training to allow users such as Data Analysts, Data Scientists and Business Intelligence personnels to gain the necessary and adequate knowledge of tools provided.

c) Tendencies to lose track of data information management or mismanaged information tracking.

Enforce strict guidelines and rule policies consistently for Metadata Management as files are being created and stored in the work-flow process.

5. Data Lake Architecture for Medical Data Processing Company



Data Lake Architecture Diagram

6. Design Considerations and Rationale

a. Ingestion Layer

Collection and integration of different types of data from various sources into a central data repository is what is known as ingestion, an entry point into the Data Lake system. The central data repository in this case is the Data Lake. Data can be manually ingested by coding ETL/ELT for every source and interface of incoming data or preferably, using a data ingestion tool such as Qlik. Data can be of various kinds, typically, Transactional Data from RDBMS, Event Logs, Web Server Logs, NoSQL data, Social Media data, Sensor & IoT data, and Third-Party data.

Qlik - Qlik provides techniques for easy migration of data from SQL Database server, ETL pipeline to Apache Spark environment, and real-time message streaming of multi-sourced data to Kafka message brokers.

Qlik manages ingestion scalability with its replication techniques and services as part of processing.

Apache Kafka - Apache Kafka is a community distributed event streaming platform capable of handling trillions of events a day. Initially conceived as a messaging queue, Kafka is based on an abstraction of a distributed commit log.

Some other tools considered for ingestion in the architecture, but not chosen include:

WaveFront - A high-performance streaming analytics platform that supports observability for metrics, counters, histograms, and traces/spans. Wavefront is unique because it scales to very high data ingestion rates and query loads.

- Wavefront offers very rich and effective features. However, there are overlaps with Apache Sparks platform which is an open-source.

Stich - Stitch is an enterprise ETL SaaS solution seamlessly flowing data from multiple data sources to a data warehouse. It's fast, it's cost-efficient and widely used in the enterprise domain.

- Stitch is a great product for ingesting data. However, the model of business and scaling was a major factor in my decision not to use Stitch.

Equalum - An enterprise-grade real-time data ingestion tool that offers an end-to-end solution for collecting, transforming, manipulating, and synchronizing data. It is capable of rapidly accelerating past traditional change data capture (CDC) and ETL tools, with its real-time or batch processing.

- Equalum is a proven stable product for data network architecture. However, the decision not to select this product was based on overlapping features.

b. Storage Layer

The storage layer in this architecture is the Amazon AWS S3 Bucket cloud facility. With AWS S3 Bucket, storage can easily be configured for use as required at a reasonably cheaper cost compared to on-premise implementation. The cloud storage environment will accommodate the need and requirements for large size storage in the Petabyte range, satisfying the requirement for 20% YoY data growth projected.

Backup/Restore and Recovery

Amazon AWS S3 is equipped with Backup/Restore and Recovery service, which can be easily activated and configured. Cloudera is another suitable tool for this particular purpose. Regardless of which of the tools is implemented, guiding policies should be activated to address security. Such rules and policies are meant to control access and create restrictions with respect to READ and WRITE privileges and maintaining data integrity.

Backup and Recovery operation is needed for Data Lake upgrade, archiving content for later reference and roll-over content from test to production. Routine Backup for a Data Lake includes the Backup of Metadata, but may or may not include the actual data.

Metadata Management

Metadata represents the information about the data, which can be of the type descriptive, administrative or structural. A well designed Metadata facilitates search and retrieval of data from a repository .It is constantly being updated as new data is ingested and archived. Strategies for backup will usually outline when Backup should be performed, as well as, how frequent and who's responsible.
Format for Metadata varies widely depending on domain, regulatory environment and the level of detail being applied in the information contained in the Metadata. Documenting information on archived data can be very difficult. However, with the appropriate tool and format it can be accomplished successfully. Among common formats are Standard Text, EML, DDI(Data Documentation Initiative), MIBBI(Minimum Information for Biological and Biomedical Investigations) and Dublin Core.
Standard Text is the recommended format for simplicity and easy search enhancement.

Security

Security implementation for the Data Lake is a shared responsibility model between the Medical Data Processing Company and the service provider. Ideally, the Medical Data Processing Company is responsible for User Access, Data and Applications, while the service provider is responsible for Operating System, Network Traffic, Hypervisor, Infrastructure and Physical. There is a long list of areas to cover. At the minimum, security strategy must address the following features:

- Cloud Access Security Brokers (CASB)
- Business analytics/intelligence
- Logs
- IP restrictions
- API gateways

The reason for this is to ensure user/network access control and protect the infrastructure.

c. Processing Layer

Ingested data arrives raw in its original format. Processing of ingested data happens in stages depending on formats and the intended use. In many cases, the use is not yet determined. However, there are specific minimum transformations that have to be performed such as Null-Value handling, Data-Type correction and Missing Value imputation.

Data Processing Tools

Data processing requires tools with specific technologies for achieving the type of processing needed. Apache Spark is an ideal Big Data processing engine for this type of data operational environment. Apache Spark Framework RDD is for transformation and action, Spark SQL(Hive) for manipulating DataFrame/Datasets, Spark Streaming(Kafka) for real-time transactional data, Spark MLib and GraphX. Typical processing use-cases are Batch processing, Realtime processing and Change Data Capture(CDC) or Ongoing Replication.

Batch Processing - Collects data and processes all the data in bulk at a later time

Realtime Processing - Continuous input of data being processed in near real-time

Change Data Capture - Captures changes made in a database and ensures replication to Data Warehouse

SQL query is an essential feature of a functional database infrastructure. Apache Spark open source offers Hive for SQL queries. This is much needed for performing Joins, merges, selects and other defined operations that are not included in stored procedures, but are necessary for data wrangling and manipulation.

There are other options when it comes to tools for the Data Lake architecture, many of which provide overlapping features. The rationale in this case is to reduce the complexity of managing so many tools and avoid cost overruns. Apache Spark Open Source offers most of what is needed to operate a fully functional Data Lake satisfactorily and exceeds the requirements of Medical Data Processing Company. Other tools could be applied as needed for future planning.
Among other platforms that serves scalable distributed Big Data processing purposes are:

Apache Hadoop - Open-source software utilities that facilitate resource clusters to solve problems involving massive amounts of data and computation. It includes a framework for distributed storage and processing of Big Data with its MapReduce data processing techniques.

- The Apache Hadoop platform was not chosen due to the fact its data processing methodology and techniques are not as fast compared to Apache Spark, which has in-memory processing architecture.

Google BigQuery - A fully-managed, high-performance serverless data warehouse that enables scalable analysis over petabytes of data. Big Query is modeled as Platform as a Service(PaaS) with built-in AI and Machine Learning capabilities.

- The Google BigQuery was not chosen because Apache Spark is an open-source with more flexibility and extended contribution from developers.

Ceph - Ceph is an open source distributed storage system designed to provide Object, Block and file storage with guaranteed performance and reliability without single point of failure through replication and scalability to the exabyte level.

- Ceph offers Big Data storage solution techniques similar to that of Apache Spark and Amazon S3. Apache Sparks offers better advantage through flexibility and scalability, together with YARN resource management, and was considered a suitable fit for the Data Lake architecture.

Resource Management

Another critical aspect of the Apache Spark Framework is hardware and software resource management capability. The containerized architecture allows flexible resource configuration and management, which provides for easy scaling. YARN, the resource entity within Apache Spark, offers the option of selecting the number of nodes needed for a server instance or multiple server instances. Given that the Data Lake Architecture will run on AWS Cloud EC2, one of the key advantages is the on-demand auto-config for GPU/CPU resources. It simply means that the system monitors performance and usage, and can increase resources as needed(Elastic Configuration).

d. Serving Layer

The serving layer consists of cleaned, organized and ready-to-use data. It is the most transformed or refined data layer in the system. Only processed data resides at this layer of the architecture. At this point the data is somewhat purified for analytics or AI/Machine Learning modeling, dashboard or story-telling. Data can be used for various decision-support systems by Data Scientists, Data Analysts, Business Intelligence personnel or anyone with applicable need. We should note that any data here can be traced to its original source.

8. Conclusion

Design and implementation of the Data Lake architecture is one major step in the data migration process. We recommend Design Verification and Tests to follow, and lastly the actual move of data to the cloud environment. This usually requires starting with the ones with least impact on the business to the most complex.

This Data Lake architecture reveals the detail of every aspect of the network with applicable tools based on the needs and requirements identified. There will likely be other needs that may arise or circumstances that were unforeseen in the near future. However, the foundation of the architecture was planned to accommodate the needs of the modern day data repository with provisions for the existing database and Data Warehouse.

What is next?

1. Proof of concept, test and verification. Collaboration with potential vendors.

2. A written migration strategy showing a detailed plan.
3. Roadmap for the migration and timeframe

9. **References**

https://data.research.cornell.edu/content/writing-metadata

https://docs.wavefront.com/wavefront_data_ingestion.html

https://www.stitchdata.com/docs/developers

Metadata Best Practices. DataONE. http://www.dataone.org/best-practices/metadata

https://www.mcafee.com/enterprise/en-us/security-awareness/cloud/cloud-security-best-practices.html

https://www.predictiveanalyticstoday.com/data-ingestion-tools/

https://www.qlik.com/us/data-ingestion/data-ingestion-tool

https://www.upsolver.com/blog/four-principles-data-lake-architecture

https://stackshare.io/stackups/google-bigquery-vs-spark

https://www.equalum.io/product