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 |
| 01/10/2023 | Hoan Nguyen | 0.1 | Initial draft |

## Reviewers / Approval

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Version Approved | Position | Date |
| FirstName LastName | 1.0 | Udacity Reviewer  Enterprise Data Lake Architect |  |

## Key Contacts

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Role | Team | email |
| Hoan Nguyen | Data Architect | Medical Data Processing | nguyentranhoan.mr@gmail.com |

# Purpose

* Evaluate if building a Data Lake makes sense to solve the company’s challenges. Expected to provide a detailed Data Lake design and rationale behind your design decisions.
* The document contains a detailed technical design proposal for an enterprise data lake system. It includes descriptions of the architecture, explanations of how the proposed design can solve the company’s challenges. And document should clearly state any assumptions or potential risks to the design.
* The document is intended to showcase data architecture ability to design a technical solution for an enterprise data lake system and to demonstrate understanding of the business problem and recommendation for a solution.
* This document is created because CTO would like to build additional capabilities with the historical data that company has such as 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.
* The target audience is those who are in charge of business management and operation from Department managers, CTO to Department employees, Business development employees, …
* In scope: a design of data lake system architecture that can handle large volumes of data, integrating various data sources, providing a flexible and scalable solution that can meet computer’s needs, a detailed explanation of the proposed solution, including the technologies and tools used.
* Out of scope: implementation of the data architecture, data governance, machine learning.

# Requirements

Summary: to build a data lake which has

* 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

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 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

These requirements have been provided by the company via Company profile – Problem statement

# Data Lake Architecture design principles

Building a data lake can break down data silos and democratize data for value extraction. A central data repository empowers organizations to make data-driven decisions and innovate quickly.

Organizations want a cost-effective and elastic storage capacity to store disparate data sources that grow exponentially. They want to centrally govern and share vast amounts of data across different business units. Furthermore, they want to empower their employees and stakeholders to derive business insights with shorter time-to-value.

Considerations when designing a data lake:

* How do you collect, store, and analyze high-velocity data across various data types, including structured, unstructured, and semi-structured?
* How do you store and share petabytes of data on-demand globally and cost-effectively?
* How do you scale IT resources to support a high number of concurrent queries against your data and scale down automatically for cost savings?
* How do your users view, search, and run queries on multiple data repositories today?
* How do you derive future insights using historical data patterns and past scenarios?

**Scalability and Durability**

* Being a centralized data repository for an entire organization, a data lake must be scalable. This feature will help scale to any size of data while importing it in real-time.
* Durability is another essential aspect of a data lake where the core storage layer must be capable of providing consistent uptime while ensuring no loss or corruption of data.

**Support for Different Data**

* Among the major design considerations in a data lake is its capability to store unstructured, semi-structured and structured data. This flexibility enables organizations to transfer anything from raw, unprocessed data to fully aggregated analytical outcomes.

**Independent of Fixed Schema**

* Organizations must ensure their data lake allows the storage of all data that don’t conform to a design. Rather, only when data is read at the time of processing, should it be parsed and adapted into a schema, as necessary. This feature saves plenty of time (usually spent on defining a schema) for enterprises.

**Decoupling Storage From Compute**

* A research from Forrester estimates that 60 to 73 percent of data gathered by organizations is unused for business intelligence (BI) and [**analytics**](https://www.indiumsoftware.com/blog/structured-approach-to-data-preparation-for-advanced-analytics/). Therefore, a data lake architecture combining compute and storage spends on compute capacity that’s under-utilized. By decoupling storage from compute, data teams will effortlessly and economically scale storage to suit the proliferation of data sets.

**Security**

* Similar to any [**cloud-based deployment**](https://www.indiumsoftware.com/blog/6-main-benefits-of-having-a-cloud-data-warehouse-in-place/), security for a data lake is a priority. Broadly speaking, the three domains of security relevant to a data lake in the cloud are encryption, network-level security and access control.
* Encryption for stored data is essential, at least for those types of data that are not publicly available. Encryption in transit is another key consideration. Usually this is configured using built-in options for every service or through TLS/SSL with their associated certificates.
* Network-level security should be consistent with an organization’s overall security framework, though it plays a critical role in implementing a robust defense strategy by denying inappropriate access at the network level.
* Authentication and authorization are the key focus areas of access control.

**Metadata Storage**

* A data lake design must incorporate a metadata storage functionality to enable users to search and learn about the data sets in the lake.
* Some of the key principles to bear in mind to ensure metadata is created and maintained are enforcing a metadata requirement and automating the creation of metadata.

# Assumptions

Building data lake design is an iterative process that anchors on your business objectives and desired outcomes, ongoing measurement against those objectives, and adapting to lessons learned to support new objectives and priorities:

* Anchor on business objectives, not technology
* Start small with a focused objective, and then learn and grow
* Ensure that the data lake can deliver business-ready data
* Design from the start for data protection and data security
* Build a data topology in support of the specialized needs of the users, devices, and APIs instead of hardcoding to technology
* Leverage the innovation of multi-cloud, automation, and management
* Don't forget the people

Questions while designing the architecture is how to:

* Set up storage in a scale manner.
* Move data.
* Cleanse, prep, and catalog data.
* Configure and enforce security and compliance policies.
* Make data available for analytics.

Missing in the problem statement

* On-premises or cloud only
* With technologies preferred: open source or paid services.
* Budget to build, run and maintain data lake.

# Data Lake Architecture for Medical Data Processing Company

A screenshot of a diagram

Description automatically generated

# Design Considerations and Rationale

## Ingestion Layer

The purpose of the Ingestion Layer of the Data Lake Architecture is to ingest Raw Data into the Data Lake. There is no data modification in this layer.

The layer can ingest Raw Data in real-time or in batches, which is in turn organized into a logical folder structure. The Ingestion Layer can pull data from different external sources like social media platforms, wearable devices, IoT devices, and Data Streaming devices.

The good thing with this layer is that it can quickly ingest any type of data including:

* Video streams from security cameras.
* Real-time data from health monitoring devices.
* All kinds of telemetry data.
* Photographs, videos, and geolocation data from mobile devices.

Data will be planned to ingest batching and streaming, all format accepted, via agent direct connect, APIs, open sources, ...

Tools would be used:

* Apache Flume, Apache Nifi, Apache Kafka: Open source, huge support community
* AWS Lambda, AWS Glue: AWS native services
* GCP cloud function, GCP Dataproc: GCP native services
* Azure Data factory, Azure Databricks, Azure Data Share: Azure native services

The ingestion layer design scale would be considered. Multi-threading archived large flies could be applied.

## Storage Layer

* A data lake provides for interrogation based upon any number of details contained within the acquired data. Data lake storage allows us to store most any type and size of data and subsequently search for something and not be sure of what your search will find nor exactly what the data’s format will be.
* The system handles 20% YoY Data Growth rate by applying data lifecycle, data retention.
* Plan to handle back-up and recovery is to create a replication storage if the master storage.
* Metadata describes the name, data type, size, and nature of data. In addition to the data files in the data lake, you will notice metadata files at a folder level that corresponds to the data files. Metadata in a data lake is written in a machine-readable format that is described by the Common Data Model (CDM) standard. When you install the Export to Data Lake feature and select data to add to the data lake, the system writes metadata files in addition to the data. Metadata will be managed in a separated repository.
* All file formats will be accepted to load to data lake to get rid of data silos and to enrich all the data.
* Securing your data lake begins with implementing very fine-grained controls that allow authorized users to view, access, process, and modify particular assets, and ensure that unauthorized users are blocked from taking any actions that would compromise data confidentiality and security. A complicating factor is that access roles may evolve over various stages of a data asset’s processing and lifecycle.
  + Data will be encrypted all the way.
  + Apply access policy to the data.
  + Usage audit data lake

## Processing Layer

* The layer that implements the business logic and analytical applications consume the data.
* Data will be processed in two main manners:

+ Batching

+ Streaming

* Data version could be enabled to manage data change.
* Make it easy to integrate services, enable to query ad-hoc.
* Tools could be involved for processing:

+ AWS: Glue, lambda, EMR, …

+ GCP: data fusion, cloud function, dataproc, …

+ Azure: data factory, functions, databricks, …

+ Open source: Hadoop, apache spark, Apache Flink, data streamset, …

* The proposed architecture scale with respect to processing:

+ For cloud based: we can choose either auto scale or on-demand scale provided cloud provider

+ For on-premises: Vertical and/or horizontal scale could be applied.

## Serving Layer

* The serving layer is a layer in a big data architecture that is responsible for serving data to end users or applications. It is the layer that provides low latency access to data for real time applications. The serving layer typically consists of a distributed database or key-value store that can handle high volumes of read and write requests.
* Mainly syndicated and curated data would be used for analysis. Plus, raw data could be used if needed for Machine learning and so on.
* Since data is available all connected services can be using the data. Additionally, all allowed 3rd party services integrated can access and use the data.

# 8. Conclusion

The architecture of the proposed data lake:

* For storage: as the company has quite a lot amount of historical data and at the same time data grow in size in the pretty fast pace, Data lake can handle it well by all ingesting techniques, apply data life cycle, data retention to optimize data storage. Making sure the all data could be accessed in an easy way.
* For data processing: as requested we want to make the most out of the data to optimize all facilities, there’s a number of tools proposed to achieve the goals.
* For serving layer: once data available in the data lake, tools are listed that could be used to serve all your needs.
* For security and data governance: since we gather all data to the data lake, we will apply all the ways to protect and keep track of data by implementing security and data governance solution as listed above.

# 9. References