**Data Modelling – Big Data**

Standards / Best Practices

Version 0.2

**Datametica.jpg**

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| 10/01/2015 | 0.1 |  | Manish Kumar  Niraj Kumar |
| 10/02/2015 | 0.2 | Document Formatting changes | Parashuram |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table of Contents

1. Overview 5

2. Enterprise Data Hub 5

4. Data Architecture 6

5. Layered Data Architecture 7

5.1 Incoming Layer 9

5.1.1 Incoming Layer - Data Standards 9

5.1.1.1 File Format 9

5.1.1.2 Compression 10

5.1.1.3 Naming Conventions 10

5.1.1.4 Metadata 15

5.1.1.5 Access 16

5.1.1.6 Table Type 16

5.1.1.7 Serialization 16

5.1.1.8 Partitioning 17

5.1.1.9 Small files 18

5.1.1.10 Compressed Data Load 18

5.2 Gold Layer 18

5.2.1 Gold Layer - Data Standards 19

5.2.1.1 File Format 19

5.2.1.2 Compression 19

5.2.1.3 Partitioning 19

5.2.1.4 Naming Conventions 19

5.2.1.5 Metadata 23

5.2.1.6 Access 24

5.2.1.7 Data Format 24

5.2.1.8 Data Types 25

5.2.1.9 Best Practices 26

5.3 Smith Layer 27

5.3.1 Smith Layer - Data Standards 27

5.3.1.1 File Format and Compressions 27

5.3.1.2 Partitioning Scheme 27

5.3.1.3 Metadata 27

5.3.1.4 Naming Conventions 28

5.3.1.5 Access 29

5.4 Outgoing Layer 30

5.4.1 File Format and Compressions 30

5.4.2 Naming Conventions 30

5.4.3 Metadata Management 30

5.4.4 Partitioning Scheme 31

5.5 Work Layer 31

5.5.1 File Format and Compressions 31

5.6 Archive Layer 31

5.7 Backup Layer 33

5.8 Security Layer 33

5.8.1 Encryption 33

5.8.2 Access 33

5.9 Directory Structure 34

# Overview

Any enterprise will have many operational systems which are critical for managing the day to day operations of business. When it comes to analysis the data and taking effective business decisions there are many challenges faced by enterprises, some of the challenges have been listed below

**Data Complexity - ETL Mesh**

* Fragmented data flow using traditional ETL tools requires more effort
* Same data being represented into multiple reports
* Report gets data from multiple data sources – Significant ETL effort

**Scalability & Compute Limitation**

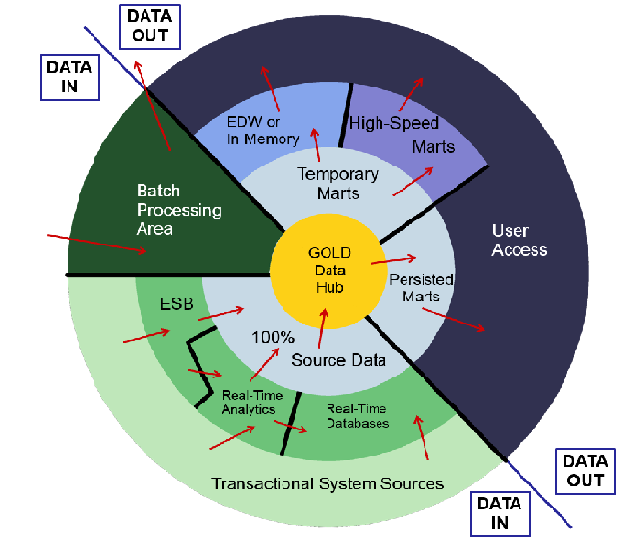
* Legacy Enterprise Data Warehouses have severe capacity and compute scalability issues.

**Data Silos and limitation of current information system architecture**

* Enterprise data is scattered
* Difficult to run predictive analytics to derive value hidden in data - Disparate systems
* No solution exists to process unstructured dataset or Real-time data feeds
* Self Service to data requires significant reoccurring ETL and modeling effort

In order to drive value and new services from data, enterprises have to change the way the data is handled. The Enterprise Data Lake / Enterprise Data Hub (EDH) using Big Data ecosystem is the future state solution for enterprises to ensure that the business users get the data in time to take effective business decisions and drive profitable growth.

# Enterprise Data Hub

1. EDH is a unified solution to store all data, for as long as desired or required, in its original fidelity; integrated with existing infrastructure and tools.

It provides flexibility to run a variety of enterprise workloads—including batch processing, interactive SQL, enterprise search, and advanced analytics.

It provides robust security, governance, data protection, and management that enterprises require. Leveraging the “Source-Once and Re-Use” approach, drives efficiency, reduces data silos reduces latency and time to value, massively improves analytics and discovery, greatly reduces costs

This document will provide an overview of data architecture approach specifically for Big Data and will also provide details related to various data operations, patterns, solutions, tools/techniques and best practices to ensure a standard way of implementing the big data solutions.

Big Data is not just about velocity, volume, veracity and variety. It is about how you identify the right information from data that is growing exponentially, and use it to add business value. Hadoop provides a low cost, but dependable solution to tackle data management problems. However, don’t expect great results by just getting your Big Data project underway. It requires a focused, analytical, use-case driven approach that organizations need to be seriously committed to. That’s when you can transform your big data into what it is meant to give you – smart information that directly translates into efficiencies, returns and growth!

To achieve business efficiencies, returns and growth, data architecture is a critical piece, as it sets data standards for all its data systems as a vision or a model of the eventual interactions between those data systems. Data Architecture describes how data is processed, stored, and utilized in an information system. It provides criteria for data processing operations so as to make it possible to design data flows and also control the flow of data in the system.

# Data Architecture

A well-defined Data Architecture provides the ability to meet data volume, latency, quality, volatility, variety, auditability, traceability, and security and integration requirements. It serves as a blueprint and guide for current and future data projects.

In particular, data architecture describes:

* How data is persistently stored?
* How components and processes reference and manipulate this data?
* How external/legacy systems access the data?
* Interfaces to data managed by external/legacy systems
* Implementation of common data operation

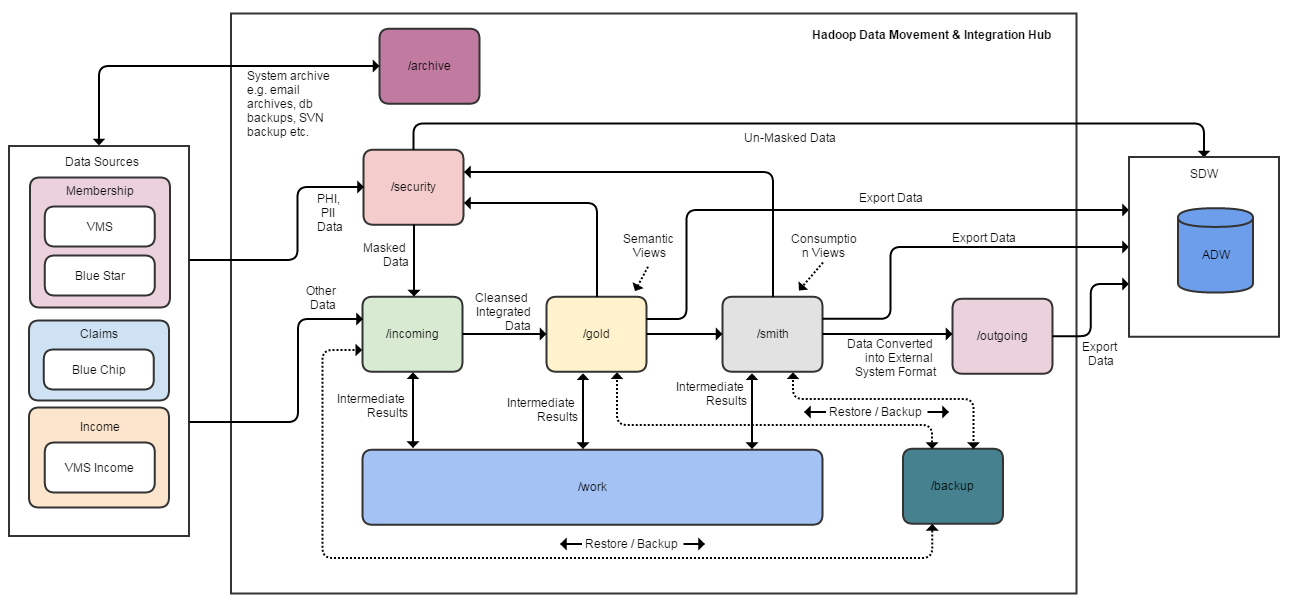
Consistent, reliable, scalable and reusable. These are all hallmarks of a data architecture approach that supports a growing enterprise. Having a solid data architecture plan in place will allow you to leverage and realize the full value of your data. In the following sections we will explore the way to build a consistent, reliable, scalable and reusable Enterprise Data Lake/Hub.

# Layered Data Architecture

In any data lake/hub, a data life-cycle aims at converting raw data to a more transformed/aggregated or de-normalized structure. To achieve so, it involves complex processing steps, security, user access/authentication/authorization, fail-over recovery and data load to variety of downstream systems. In the age of big data, managing such data life-cycles becomes increasingly challenging, hence the need for a correctly design and implemented data lake/Hub architecture. .

The key to any modern data architecture is the layered organization of data on a Hadoop distributed file system. Irrespective of kind of Hadoop-augmented big data application, you can decompose a data lake/Hub into logical groupings of components called layers. Layers help to differentiate between the different kinds of tasks performed in the lake, making it easier to create a design that supports simpler management. In the Datametica layered architecture, each data layer is defined with specific purpose keeping in mind data life-cycle phases. Apart from simplifying data life-cycle complexities, a layered architecture is advantageous in many other ways.

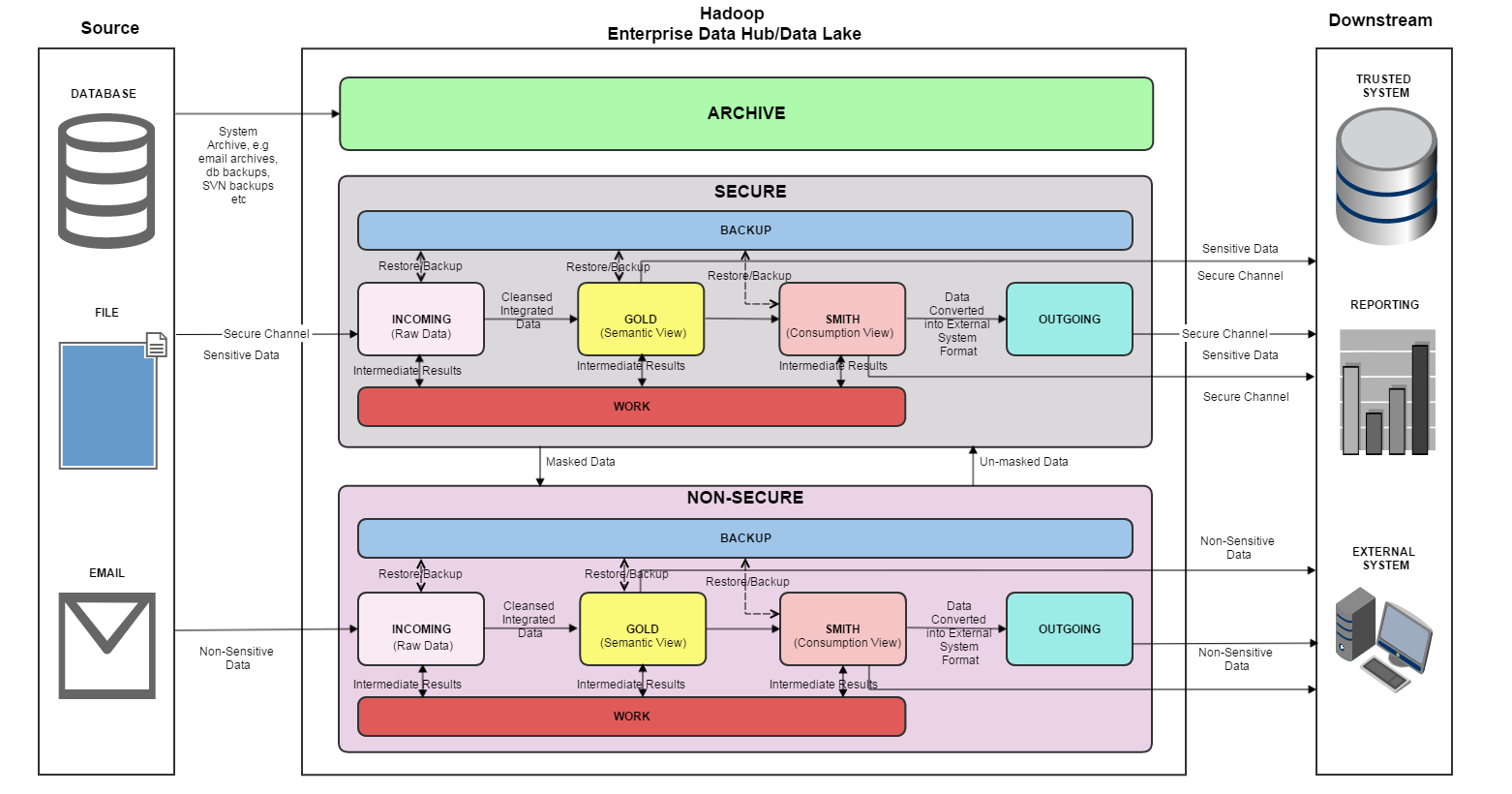
1. Layered structure makes it easier to share data between teams working with the same data sets and drives data source re-use.
2. It also allows for enforcing access and quota controls to reduce the likelihood of accidental deletion or corruption or sensitive information hiding.
3. Better change management: Each layer represents a designated area in which data are changed as per to each layer purpose.
4. Better security, layers can have specific security and encryption patterns.
5. Better failure recovery as backups can be restored layer wise.
6. Better control over schema, file formats, compressions etc. You can choose from different options available as they are better suited for each layer operations.

**

**Fig: Generic Layered Architecture**

The layers in an enterprise data hub/data lake are as follows

|  |  |
| --- | --- |
| **Layer Directory** | **Purpose** |
| **/incoming** | Consists of all the raw data, full history and full detailed records similar to the source system. No user is allowed to query this data directly. |
| **/gold** | It contains table with semantic view of the data that is created using the raw data from the incoming layer. This layer is the single version of truth and contains data in de-normalized format. |
| **/smith** | It provides a layer for conforming, aggregations and the application of business rules. Smith data can contain modified, joined or subsets of data primarily sourced through the Gold layer. Smith data can be sourced from Incoming, Work, and/or Gold Layer data. In some cases this Layer can work as Temporary data mart as well. |
| **/outgoing** | It contains the data from gold or smith layer but in the format that needs to be sent to external target system for consumption. |
| **/work** | It is a layer used to store all the intermediate transformation outputs. |
| **/security** | All the sensitive data lands into this layer. Then batch jobs run on this data to mask it and then the output is stored into non-secure layer. Only very privileged users should be allowed to query this layer. Also this layer should be used to unmask data in case it needs to be made available to the downstream target systems. |
| **/archive** | It is used for storing backups for all other enterprise systems. For example, database backups, emails etc. |
| **/backup** | This should be physically on separate Hadoop cluster. Daily, Weekly, Monthly snapshots of the HDFS should be stored into this cluster so that they can be retrieved in case of primary cluster data loss. |



**Fig: (Security) Use Case driven Layered Architecture**

# Incoming Layer

This is the landing layer in Hadoop. Data in incoming layer is untouched version or raw form of data. It is a near replica of source system. It stores data as it arrives from source systems. Having data in its raw form means it will always be possible to perform new processing and analytics with the data as requirements change.It also maintains the history of raw data as per defined SLA.

The next section will provide more details on the data standards that need to be applied to the incoming layer, these standards will be around the file formats, compression techniques, access, table type, partitioning and serialization.

# Incoming Layer - Data Standards

# File Format

Incoming layer is the layer that will have its dependency on external data sources for ingesting data. Keeping that in mind for incoming layer there are some specific file format properties that are suitable.

* The format should be able to support splittable compression as data will be used in processing jobs such as Map Reduce. Splittable formats enable Hadoop to split files into chunks for processing, which is critical to efficient parallel processing.
* It should support serialization in a language agnostic manner as source system can be implemented in any language.
* Better support to handle file corruption. In case of failure, read should not stop the entire read, instead it should continue reading good records. Handling corrupted records should be taken separately. But it should not stop other data processing.

Avro is most suited format for incoming layer for text data types. In case of binary data like images, sequence files are more suitable. However, if splittable unit of binary data is larger than Hadoop storage blocks size then you may consider putting the data in its own file without using any Hadoop container format.

# Compression

For Incoming layer, snappy compression is preferred as data would be used for further processing. We do not want compression to slow down the data processing job specially in case of real time/micro batch processing. Snappy doesn’t offer the best compression sizes but it does provide a good trade-off between speed and size. Processing performance with Snappy is significantly better than other compression formats. Snappy is intended to be used with a Hadoop container format like Avro, since it’s not inherently splittable. It is recommended to run periodic jobs to compress the data in incoming layer.

# Naming Conventions

* Database Name: Database name should be “incoming”
* Table Names: All incoming table name use following naming convention  
   <subject\_area>\_<source\_system>\_<qualified\_name>\_<suffix>
* Table name suffix can be:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | Audit Tables | \_AUD | | Link Tables | \_LINK | | Transaction Tables | \_TXN | | Validation Tables | \_TBL | | Cross Reference Tables | \_XREF | | Fact Table | \_FCT | | Dimension Table | \_DIM | | Views | \_VW | | Work | \_WRK | | History | \_HIST | | Current | \_CURR | |

* Column Names: All incoming column names use following naming conventions  
  <source\_system\_column\_name>\_<qualified\_name>\_<suffix>
* Column Abbreviations: Some of the columns that can abbreviated in qualified\_names are:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | Columns | Abbreviation | | ADDRESS | ADDR | | ADMINISTRATION | ADMIN | | ALTERNATE | ALT | | AMOUNT | AMT | | AMERICAN | USA | | APPLICATION | APPL | | AUTHORITY | AUTH | | AVERAGE | AVG | | BREASTHEIGHT | BH | | BUSINESS | BUS | | CANADIAN | CDN | | CATEGORY | CAT | | CLASSIFICATION | CLASS | | CLIENT | CLI | | COLLECTION | CLCTN | | COLUMN | COL | | COMMENT | CMT | | COMMISSION | COMM | | COMMITTEE | CTTE | | COMPANY | CO | | CONDITION | CONDTN | | CONTROL | CTL | | CONVERSION | CNV | | COORDINATE | COORD | | CORPORATION | CORP | | CORRECTION | CRCTN | | COUNT | CNT | | CREDIT | CR | | DATE | DT | | DAY | DY | | DESCRIPTION | DESC | | DESTINATION | DEST | | DEPARTMENT | DEPT | | DETAIL | DTL | | DEVELOPMENT | DEV | | DIAMETER | DIAM | | DIAMETER AT BREAST HEIGHT | DBH | | DIAMETER INSIDE BARK DIB | DIB | | DIAMETER OUTSIDE BARK DOB | DOB | | DISTURBANCE DISTRB | DISTRB | | DOUBLE BARK THICKNESS DBT | DBT | | DISTRICT (Forest District) DIST | DIST | | DIVISION DIV | DIV | | DOCUMENT DOC | DOC | | EFFECTIVE | EFF | | ELEMENT | ELMNT | | ERROR | ERR | | ESTIMATE | EST | | EXECUTIVE | EXEC | | EXPIRY | EXP | | FACTOR | FCTR | | FEDERAL | FED | | GROUP | GRP | | HARVEST | HARV | | HECTARES | HA | | HEIGHT | HGHT | | HOUR | HR | | IDENTIFICATION | ID | | INDEX | INDX | | INDICATOR | IND | | INITIAL | INIT | | INVENTORY | INV | | JURISDICTION | JURIS | | LATITUDE | LAT | | LENGTH | LEN | | LETTER | LTR | | LICENCE | LIC | | LOAD | LD | | LOCATION | LOCN | | LONGITUDE | LONG | | MANAGEMENT | MGT | | MAXIMUM | MAX | | METERS CUBED | M3 | | METERS SQUARED | M2 | | METHOD | MTHD | | MINIMUM | MIN | | MINUTE | MN | | MONTH | MO | | NAME | NM | | NUMBER | NO | | ORGANIZATION | ORG | | PAYMENT | PAY | | PERCENT | PCT | | PERMIT | PRMT | | PIECE | PCE | | POSITION | POS | | PREVIOUS | PREV | | PRIMARY | PRI | | PRODUCT | PROD | | PROJECT | PROJ | | QUANTITY | QTY | | RECEIVED | RECV | | REFERRED | REF | | REGION (Forest Region) | REG | | REGISTRATION | REGN | | RESPONSE CENTRE | RC | | REQUEST | RQST | | REQUIRED | REQ | | REQUIREMENT | RQMT | | RETURN | RET | | REVENUE | REV | | SCHEDULE | SCHED | | SCREEN | SCR | | SEARCH | SRCH | | SECONDARY | SEC | | SECTION | SECT | | SEQUENCE | SEQ | | SERVICE | SRVC | | SILVICULTURE | SILV | | SOURCE | SRCE | | SPECIES | SPP | | STATEMENT | STMT | | STATUS | STS | | STATUTORY | STAT | | STATISTICS | STATS | | TENURE | TENR | | TEXT | TXT | | TIMBER | TMBR | | TIMBER SUPPLY AREA | TSA | | TIMBER SUPPLY BLOCK | TSB | | TIME | TM | | TIMESTAMP | TS | | TITLE | TTL | | TOTAL | TOT | | TRANSACTION | TXN | | TREATMENT | TRTMT | | TYPE | TYP | | USERID | UID | | VALUE | VAL | | VERSION | VER | | VISITATION | VISIT | | VOLUME | VOL | | WITHDRAWAL | WD | | WEIGHT | WGT | | XREF | XF | | YEAR | YR | | YEAR-TO-DATE | YTD | |

# Metadata

Following is the list of business metadata that needs to be captured in incoming layer:

* Subject Area: Source subject area like membership.
* Business Rules: Any specific rules that have been applied over the data or that needs to be applied later on in the data lifecycle.
* Source processes: Any specific processes/division/function that data may have been sourced from.
* Destination: Any target system that data is going to be used for.
* Data Type: PHI/PII/HIPAA compliant sensitive data.

Following is the list of operations metadata that needs to be captured in incoming layer:

* Job ID/Batch ID: The Job or process that is associated with the data.
* Script Name: Any specific script name that associated with the data.
* Script Type: Script type that is associated with the data like Hive SQLs, Map Reduce, Pig, Shell etc.
* Log Locations/Files: Any specific log details that is associated with the data.
* Operations Type: What type of operations that data is associated with like append only, historical updates etc.
* User group and permissions: What kind of user or groups has access to data?
* Source metadata: Any specific metadata that is received from source.
* Number of records: Number of records that are added/modified in a job.

Following is the list of technical metadata that needs to be captured in incoming layer:

* Table/Schema definition: Underlying table/schema definition that is associated with data
* Database Details: What kind of database data is associated with?
* File Formats: What is the underlying file format associated with data? Like Avro, RCFile etc
* Column definitions: What kind of column data is associated with?
* Source Data checksums: Any checksums that is received from source.
* Actual Data checksums: Actual checksums associated with data.
* Min/Max: Any minimum and maximum value across table data.
* Unique records: Number of unique records
* Total number of records: Total number of records in a table.

# Access

|  |  |  |
| --- | --- | --- |
| Role | Read | Write |
| Business Users |  |  |
| Operation Support/Engineering/Ingestion Team |  |  |
| Data Scientists |  |  |

# Table Type

In Hive/HCatalog supported Hadoop Data Lake it is advisable to have all tables in Incoming layer as external tables. External Tables are the combination of Hive table definitions and HDFS managed folders and files. The table definition exists independent from the data, so that even if the table is dropped, the HDFS folders and files remain in their original state. Using external tables establishes the fact that incoming data life cycle is not in entirely in control of HIVE.

There are external processes that can actually control the data. In Incoming layer there are external processes that would contributing to data lifecycle. One is the data load scripts that would load data from external sources. Another can be the archival process that would archive the data as per SLAs and remove them from incoming directory.

# Serialization

Serialization is core to a distributed processing system such as Hadoop. It allows data to be converted into a format that can be efficiently stored as well as transferred across a network connection. Avro is a language-neutral data serialization system that is preferred for Incoming layer. Avro data files are compressible and splittable. Avro supports schema evolution that is, the schema used to read a file does not need to match the schema used to write the file. This makes it possible to add new fields to a schema as requirements change.

# Partitioning

Partitioning addresses key issues in supporting very large tables by letting you decompose them into smaller partitions thereby facilitating better management of influencing factors such as data loading, aging, and archival. Following are the recommended strategy for data partitioning in incoming layer.

* **Default/Zeroth Partition**

This would be landing zone for any data load from source. This setup ensures that Hadoop jobs do not pick files for processing unless and until all files are copied from source. Once all files are copied then these files are moved to actual partition.

* **Data load: Full/Historical batch load**

For full load it is recommended to have day, month and year partitions

* **Data load: Intra-day batch load**

If target system is designed only for intra-day batch data load then it is advisable to partition data by day, month, week, quarter and year. If you have **multiple sources** from which data is loaded into the lake, then it is advisable to have an additional partition on top of time partitions mentioned above.

* **Data load: Micro-Batch, real-time**

In this case it is advisable to partition data by hour, day, month, week, quarter and year.

A good strategy for time-range partitioning, for example, is to determine the **approximate size of your data accumulation over different granularities of time**, and start with the granularity that results in “modest” growth in the number of partitions over time, while each partition contains files at least on the order of the file system block size or multiples thereof. This balancing keeps the partitions large, which optimizes throughput for the general case query. Another solution is to use two levels of partitions along different dimensions.

* **Delayed buckets for one-day reconciliation of delayed data loads.**

The next two section we have listed best practices related to handling small files and compressed files in the incoming layer

# Small files

Since Data immutability is one of core principles of Incoming layer, any incremental load would append to existing data set. In Hadoop based Data Lake, it’s possible to “append” data by adding it as new files or partitions. When appending to directories with additional files, it may result in numerous small files. HDFS is optimized for large files. If the requirements call for a two-minute append process that ends up producing lots of small files, then a periodic process to combine smaller files will be required to get the benefits from larger files.

# Compressed Data Load

Incoming layer will have 2 directories. One folder will contain the data which can be directly processed or can be queried upon. Other folder will have data which cannot be directly processed e.g. zip folders. So if we have zip file in this folder it will be unzipped in work layer and files will be moved into incoming layer where data can be queried.

# Gold Layer

The Gold Layer is the “single-version of the truth” for the enterprise data set. Gold Layer will usually have de-normalized, enriched and qualified data set. If you are planning to build the enterprise data lake / data hub it is recommended to build source similar data set as the single version of truth. In the initial stage many enterprises will not have clear requirements on building the de-normalized data set, this source similar layer will act as the basic foundation for building any further de-normalized data sets.

The data will be “lightly standardized” (see data formatting standard) and the data in the Gold layer will be properly named and defined in Hive and will use External tables.

Data in the Gold layer should be designed with an eye for consumption at the back-end. Data should be de-normalized when this makes sense. For example, if a request and response transactions are consumed into Hadoop, these records should be combined so that the users of this data won’t need to combine the data. Another combining strategy would be to include dimensional, or reference data in the file to avoid joins at the back end.

Caution needs to be taken in this approach. Very large files (like RIM data) could be made much larger if all dimensional codes are translated out. We also need to consider data availability and source data changes. Data from disparate sources may not be available at the same time. This could delay the availability of data. Also, source data change management is a consideration. As source data files change, this could have a negative impact upon combined data.

The next section will provide more details on the data standards that need to be applied to the gold layer, these standards will be around the file formats, access, table type, and data type.

# Gold Layer - Data Standards

# File Format

Gold layer contains data that is single version of truth of Enterprise data. As many enterprise level applications and users will be accessing the Gold layer, it is one of the layers where query processing would be maximum hence it is Important to ensure that Gold layer file formats be optimized for query performance. On the other hand, gold layer would most of the times contain high volumes of data therefore the file format to store large volume of data should be storage efficient.

It is recommended to have ORC file format for HDP and Parquet for CDH. It should use default compression codec available with the respective file formats.

# Compression

Snappy would be preferred compression codec for Gold layer as this is split table and gives better performance and compression ratio. It is by default supported by Hadoop.

# Partitioning

Preferred partitioning is date based partitions since most of the analytical queries in Gold layer would be aggregated across dates. In cases where there are many updates/queries based on unique leys like member id or SSN other options of Hive bucketing, hash based partitioning is preferred.

# Naming Conventions

* **Database Name:** Database name should be “Gold”

Table Names: All incoming table name use following naming convention  
 <subject\_area>\_<sub\_suject\_area>\_<qualified\_name>\_<suffix>  
  
**Table Naming Standards**

* All table names are abbreviated based on standard abbreviations.
* Table names shall be unique within a model.
* Table names shall not be greater than 30 characters.
* Physical table comments should be restricted to 250 characters. The logical entity description can contain more than 250 characters. If the table description requires more than 250 characters, enter the text in the logical model where the 250 character limit does not exist.
* Table definition must begin with the table name (e.g. Telephone contains all the specific address of a telephone within a zone and prefix for our members, providers, etc.).
* All textual definitions must be in non-technical business terminology.
* Table name must not use letter abbreviations that don’t exist in the NSM naming file. (e.g. AA,BB)
* Table name must not begin with numbers (e.g. 5, 10, 15, etc.).
* Work tables should be prefixed with WRK-. Standard naming conventions are applied, and meta-data is documented whenever possible.
* Cross Reference tables should be prefixed with XREF. Standard naming conventions are applied, and meta-data is documented whenever possible. (ADW, TDW, and Future repositories )
* Cross Reference tables should be suffixed with XREF. Reference tables should be suffixed with REF. Standard naming conventions are applied, and meta-data is documented whenever possible. (ODS and EDW Daily)
* Cross Reference tables should be suffixed with REFR. Standard naming conventions are applied, and meta-data is documented whenever possible. (EDW Monthly)
* Reference tables should be suffixed with REF. Standard naming conventions are applied, and meta-data is documented whenever possible. Should be code tables.(All Repositories going forward)
* All history tables should be suffixed with HIST.
* Table partitioning designations should be done if information is available.

**Column Naming Standards**

* All column names are abbreviated based on standard abbreviations.
* All column names must end with a valid class word.
* Column name must not begin with numbers (e.g. 5, 10, 15, etc.).
* Column name must not use letter abbreviations that don’t exist in the NSM naming file. (E.g. AA, BB).
* All standard columns in a table must be ordered alphabetically.
* All like named columns or role-named columns must share the same metadata characteristics in the model (e.g. GNDR\_CD must be defined the same in all tables).
* Physical column comments cannot exceed 250 characters. Logical comments are not restricted to this limitation. If a comment exceeds 250 characters, ensure the comment is originally entered in the logical model so the logical model text is not truncated by ERwin.
* Column definition must begin with the column name (e.g. Telephone Number Identifies the specific address of a telephone within a zone and prefix).
* Standard WIP Column Color should be removed in the subsequent release.
* If the foreign key is a code and there is no applicable parent value, use the appropriate code value transformation rules defined in the Special Rules for Codes bullet below.
* If the foreign key is a warehouse surrogate key (DECIMAL 18) and there is no applicable parent value, use 0 or a NOF row as required. Note that a zero row need not exist in the parent.
* Although we provide the DDL for all foreign keys, they are currently disabled in the database. They are checked by the ETL application, and provide performance improvements via the optimizer (move to ddl template document)

**View naming standards**

|  |  |  |
| --- | --- | --- |
| Type of View | Description | Standards |
| Restricted views | Views that display fewer than the full set of columns contained in the original view | suffixed with "\_VW" |
| Join views | Views based on a join or union of two or more tables | suffixed with "\_VJ" |
| Application view | The view is application specific, the view should be prefixed with the application name |  |

**List of Class words**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Logical Name | EDH Abbreviation | Data Type | Definition | Example |
| Address | ADDR | VARCHAR(N) | Designation of a place where something is located or something is to be delivered | Home Address |
| Amount | AMT | Number(P, S) | A measure or sum of some quantity in a given unit of measurement | Sales Amount |
| Area | ARA | NUMBER | Numeric value representing the amount of surface area of something. | Square Feet Area |
| Blob | BLB |  | Picture, sound files, etc. | JPG, MP3 |
| Code | CD | CHAR(N) | One or more letters or numbers that have a specific meaning or interpretation | Store Format Code |
| Count | CNT | Number | The number of items which are of the same type | Total Net Transaction Count |
| Date | DT | Date | The point in time, started in terms of year, month and day. The format is YYYYMMDD | Email SignUp Date |
| DateTime | TS | Timestamp | Point in time at which something happens, existed, or is to happen. Time part is significant | Create DateTime |
| Description | DESC | VARCHAR(N) | Textual information that gives identifying information | Email Format Description |
| Identifier | ID | NUMBER | An identification number for a person or thing | Customer Id |
| Image | IMG |  |  | SKU Image |
| Indicator | IND | CHAR(1) | Either Y or N | Active Indicator |
| Month | MTH | CHAR(N) | Alphabetic or numeric value representing a specific month of a year | Plan Month |
| Name | NM | VARCHAR(N) | A word or words by which an entity is conventionally designated or distinguished from others. | Customer Name |
| Number | NBR | NUMBER | Alphanumeric field which identifies something. | Location Number |
| Percent | PCT | Number(P, S) | A special kind of RATE which is usually bounded by 1 and 100, and is calculated by dividing a partial amount by a total amount | Sales To Plan Amount Percent |
| Quantity | QTY |  |  |  |
| Range | RNG | VARCHAR(N) | The length from lowest to the highest value numbers in a set | Age Range |
| Rate | RT | Number | A ratio or proportion that is derived from or applied to another number | Customer Satisfaction Rate |
| Text | TXT | VARCHAR(N) | Unformatted alphanumeric descriptive information | Additional Text |
| Time | TM |  |  |  |
| Title |  | VARCHAR(N) | Title is similar to NAME but NAME is preferred for things that do not have multiple instances | Customer Title |
| Year | YR | Char(4) | Numeric value representing a given year | Plan Year |
| Factor | FACTR |  | A discrete number that can be used in an arithmetic expression |  |
| Period | PRD |  | A partial date usually consisting of only the century, year and month | |

# Metadata

Following is the list of business metadata that needs to be captured in Gold layer:

* Subject Area: Source subject area like membership.
* Business Rules: Any specific rules that have been applied over the data or that needs to be applied later on in the data lifecycle.
* Consumption Processes: Any enterprise system/process that is querying the data.
* Data Type: PHI/PII/HIPAA compliant sensitive data or masked data or non-sensitive data.
* Source Systems: This would be list of sources from which data is collected from.

Following is the list of operations metadata that needs to be captured in Gold layer:

* Job ID/Batch ID: The Job or process that is associated with the data.
* Script Name: Any specific script name that associated with the data.
* Script Type: Script type that is associated with the data like Hive SQLs, Map Reduce, Pig, Shell etc.
* Log Locations/Files: Any specific log details that is associated with the data.

Following is the list of technical metadata that needs to be captured in Gold layer:

* Table/Schema definition: Underlying table/schema definition that is associated with data
* Database Details: What kind of database data is associated with?
* File Formats: What is the underlying file format associated with data? Like Avro, RCFile etc ..
* Column definitions: What kind of column data is associated with?
* Min/Max: Any minimum and maximum value across table data.
* Unique records: Number of unique records
* Total number of records: Total number of records in a table.
* Total number of records category wise/department wise.
* Business Effective Start and End date.
* System Dependencies: Any enterprise dependent system that is going to be affected by change in data.
* Table Type: Data is associated with what kind of table like fact/dimension/aggregated/de-normalized/normalized etc.
* Incoming Source Table Names: This would be list of incoming layer table from which data is collected from.

# Access

|  |  |  |
| --- | --- | --- |
| Role | Read | Write |
| Business Users |  |  |
| Operation Support/Engineering Team |  |  |
| Data Scientists |  |  |
| Data Analyst |  |  |
| Business Analyst |  |  |
| Reporting Application IDs |  |  |

# Data Format

The following is a list of data formatting standards that should be followed for data in the Gold layer. These transformations may occur in a source extraction program, between Incoming to Gold, or in the Hadoop Work layer, but they are required prior to landing in the Gold layer.

* The format of date fields (Class word of ‘dt’) should be standardized to ‘YYYY-MM-DD’ for consistent processing.
* The format for time fields (Class word of ‘tm’) should be standardized to ‘HH:MM:SS’. Hours should be in military (24 hr format) time.
* The format for time stamp fields (Class word of ‘ts’) should be standardized to ‘YYYY-MM-DD HH:MM: SS.MSM’, where ‘MSM’ is milliseconds. Milliseconds are optional. Include if they are available from the source.
* A load\_ts (load timestamp) field needs to be added to record layouts that do not contain a dependable extraction/transaction timestamp. This is a house keeping field. This facilitate in tracking when the data was loaded. The load\_ts field should be added to the beginning of the Gold file layout.
* If a file contains signs for numeric fields the following should be adhered to. Plus signs should be removed. Negative signs should be added to the beginning of the field.
* Fields that contains multiple logical fields should be stored as is. If the data content can be deciphered, the deciphered, separated data should also be retained in separate fields if the transformation logic is understood. The deciphering should be done to facilitate easier data selection and joining. Retaining a copy of the mixed field is done to mimic the source system.
* Trim off any preceding or trailing spaces within a field.
* Format amount fields to eliminate commas.
* For computational fields, leading zeros should be removed. If the value is less than one, then a leading zero should be retained. For example a source data value of 000.59 should be formatted as 0.59
* Fields that contains default dates should retain the defaults from the source if they have meaning. For example, an expiration date may contain a value of ‘2999-12-31’. While this date may reflect a true expiration date, it is used as a default date for products that have not expired. This data should not be changed.
* If it is understood that a field contains (default) data that represents null values, the values should be changed to contain an empty string. For example, some IMS programs contain ‘0000-00-00’ for date fields or ‘000000’ in other fields, which means there is no data. The zeros take up space and can hinder analysis. The zero data should be changed to contain an empty string. Care should be taken to understand the data.
* For common corporate reference data (character or numeric) if it is understood that a field should contain 3 characters, the field should be consistently formatted to contain 3 characters. For example, Division\_nbr is a 3 position number. Leading zeros are expected for division ‘009’, however a value of ‘0009’ can cause data selection problems. The leading zero should be trimmed for consistency. Care should be taken to understand the data.
* Use ^A(\001) as the field delimiter, ^B(\002) for collection items, and ^C(\003) for map keys. Do not put a delimiter before the first column and/or after the last column. New line input is the row delimiter. For deeper levels not specified here (004 and beyond), follow the default Hive delimiter standard.
* Light to heavy transformations
* Conformed dimensions must be present and used
* Facts must conform to dimensions
* Transformation of the gold layer are allowed to accommodate specific analytics or BI need – they have to be derived from the conforming dimensions and facts.

# Data Types

The following is a list of data type standards that should be followed for data stored in the Gold and Smith layers. The following data types should be used if the source is a relational source (RDBMS) such as Teradata, Oracle, SQL Server, etc. If the data source is a file or unstructured source where formatting cannot be guaranteed, then it is more appropriate to define all incoming fields as STRINGs. Assigning appropriate data types will improve system performance along with enhancing the display of data within end user reporting tools.

* Attribute fields should also be defined as STRING.
* Quantity fields which are additive should be defined as an Integer type (TINYINT, SMALLINT, INT, BIGINT).
* Amount fields which are additive should be defined as DOUBLE.

**Note:** It is important that any fields being defined as integer, float, or double be validated to have the appropriate data types in those fields. Invalid data types can result in incorrect results and parsing issues leading to possible program errors. Additionally, it is important to have the correct integer definition. If, for example, you define a field as SMALLINT and the value coming from the source is greater than 32,767 then this field will be returned as NULL by Hadoop. Thus if you were aggregating values from your source system and Hadoop and compared the values, they would not match.

# Best Practices

* For fact tables, on incoming data add auditing column, clean the data and take data into gold layer. Once the de-normalize structure is finalized then put it into gold layer and decommission previous tables.
* For dimension tables directly put de-normalized table.
* Enterprise wise used tables should be in gold layer. It is okay if they have aggregated data. Gold layer will serve as single point of truth at any given time.
* Data Archiving from this layer will be business driven.

# Smith Layer

The name of this layer comes from blacksmith, gunsmith or barrel smith. A smith was a forger, a molder or developer of something. The intent of this to provide a layer for conforming, aggregations and the application of business rules. Smith data can contain modified, joined or subsets of data primarily sourced through the Gold layer. The lifecycle of data in the Smith layer is similar to data in the Gold layer. Smith data can be sourced from Incoming, Work, and/or Gold Layer data. A Hive data structure shall be defined. If gold standards are not matched, then data can be put in this layer. This Layer can work as Temporary data mart as well. Over the period of time if Smith tables are used enterprise wise then those can be moved to Gold layers.

The next section will provide more details on the data standards that need to be applied to the smith layer, these standards will be around the file formats, access, table type, and data type.

# Smith Layer - Data Standards

# File Format and Compressions

Smith layer would follow same guidelines as Gold Layer.

# Partitioning Scheme

Smith layer should follow same partitioning scheme as Gold.

# Metadata

Following is the list of business metadata that needs to be captured in Smith layer:

* Subject Area: Source subject area like membership.
* Business Rules: Any specific rules that have been applied over the data or that needs to be applied later on in the data lifecycle.
* Consumption Processes: Any enterprise system/process that is querying the data.
* Data Type: PHI/PII/HIPAA compliant sensitive data or masked data or non-sensitive data.
* Source Systems: This would be list of sources from which data is collected from.

Following is the list of operations metadata that needs to be captured in Smith layer:

* Job ID/Batch ID: The Job or process that is associated with the data.
* Script Name: Any specific script name that associated with the data.
* Script Type: Script type that is associated with the data like Hive SQLs, Map Reduce, Pig, Shell etc.
* Log Locations/Files: Any specific log details that is associated with the data.

Following is the list of technical metadata that needs to be captured in Smith layer:

* Table/Schema definition: Underlying table/schema definition that is associated with data
* Database Details: What kind of database data is associated with?
* File Formats: What is the underlying file format associated with data? Like Avro, RCFile etc ..
* Column definitions: What kind of column data is associated with?
* Min/Max: Any minimum and maximum value across table data.
* Unique records: Number of unique records
* Total number of records: Total number of records in a table.
* Total number of records category wise/department wise.
* Business Effective Start and End date.
* System Dependencies: Any enterprise dependent system that is going to be affected by change in data.
* Table Type: Data is associated with what kind of table like fact/dimension/aggregated/de-normalized/normalized/ad-hoc etc.
* Gold Source Table Names: This would be list of Gold layer table from which data is collected from.
* Destination Table names: Any destination table names data is going to be loaded.

# Naming Conventions

* Database Name: Database name should be “Smith”
* Table Names: All incoming table name use following naming convention  
   <subject\_area>\_<sub\_suject\_area>\_<target\_system>\_<qualified\_name>\_<suffix>
* Rest all conventions would be same as Gold layer.

# Access

Access would be given in this layer on need basis and would be completely driven by business requirements. However, some of the most commonly used access roles are in the table below.

|  |  |  |
| --- | --- | --- |
| Role | Read | Write |
| Operation Support/Engineering Team |  |  |
| Data Scientists |  |  |
| Data Analyst |  |  |
| Business Analyst |  |  |
| Reporting Application IDs |  |  |

# Outgoing Layer

This layer is the storage location for files going to downstream applications. One file may feed multiple downstream applications.

# File Format and Compressions

This would be completely governed by downstream system requirements. Avro is preferred choice with snappy compressions.

# Naming Conventions

# Metadata Management

Following is the list of business metadata that needs to be captured in outgoing layer:

* Subject Area: Source subject area like membership.
* Business Rules: Any specific rules that have been applied over the data or that needs to be applied later on in the data lifecycle.
* Consumption Processes: Any enterprise system/process that is querying the data.
* Data Type: PHI/PII/HIPAA compliant sensitive data or masked data or non-sensitive data.
* Source Systems: This would be list of sources from which data is collected from.

Following is the list of operations metadata that needs to be captured in outgoing layer:

* Job ID/Batch ID: The Job or process that is associated with the data.
* Script Name: Any specific script name that associated with the data.
* Script Type: Script type that is associated with the data like Hive SQLs, Map Reduce, Pig, Shell etc.
* Log Locations/Files: Any specific log details that is associated with the data.

Following is the list of technical metadata that needs to be captured in outgoing layer:

* Table/Schema definition: Underlying table/schema definition that is associated with data
* Database Details: What kind of database data is associated with?
* File Formats: What is the underlying file format associated with data? Like Avro, RCFile etc ..
* Column definitions: What kind of column data is associated with?
* Min/Max: Any minimum and maximum value across table data.
* Unique records: Number of unique records
* Total number of records: Total number of records in a table.
* Total number of records category wise/department wise.
* Business Effective Start and End date.
* System Dependencies: Any enterprise dependent system that is going to be affected by change in data.
* Table Type: Data is associated with what kind of table like fact/dimension/aggregated/de-normalized/normalized/ad-hoc etc.
* Gold/Smith Source Table Names: This would be list of Gold/Smith layer table from which data is collected from.
* Destination Table names: Any destination table names data is going to be loaded
* Target File System type: This could be XMLs, JSON, CSV etc..

# Partitioning Scheme

Most of the cases there are no specific partitioning scheme in outgoing layer as data is not persisted for long here.

# Work Layer

This is a working area for data transformation. This is a temporary work area. Data is specific to a job or process, so other jobs or facilities should not be accessing this data. In this layer, data can be Hive exposed. The Hive definitions do not need to follow field naming standards. Jobs that create work layer files and tables should delete these files and tables after processing. Production data that has not been changed after 7 days, will be systematically deleted. For the QA environment, data will be deleted after 30 days. Administrators can setup some policies to clean the data from this layer time to time.

# File Format and Compressions

Since work layer would be primarily used for processing, it is recommended to have ORC or Parquet file format with snappy compression.

# Archive Layer

This layer is reserved for external data that is being stored on HDFS for online, long term retention. This will be a non-active directory and will be used only for backup/recovery purpose. No one will have access to this directory. Archive layer data differs from other data in that its primary purpose is to be available for reloading to the source system. Because of the reloading requirement, the lifecycle of the data is greatly simplified. The file and data structure are not modified from the source system. Data is loaded directly into the Archive layer, bypassing Incoming and Work. A Hive schema will not be defined; however, documentation of the DDL or data layout is required.

Data backup will depend on source system policy. In case on complete backup being taken again existing data will be deleted. Data transfer will be done via NFS gateway. Recommended replication factor is 2. Recommended compression codec is bzip2.

# Backup Layer

This layer has to be part of physically separate cluster. Backup of primary Hadoop cluster shall be taken time to time to this separate cluster. Users might delete the data accidentally or multiple system crash may happen. In those scenarios, primary cluster can be recovered from Backup. Data from incoming, gold, smith and outgoing layers present in both secure and non-secure layers must be backed up. Other layers can be based on organizational requirements data backup can be taken as those layers can be formed again from incoming, archive or outgoing layer data.

# Security Layer

All the data that is termed sensitive as per the laws and regulations, is stored into this layer first. This will be the landing zone for any sensitive data. Sensitive data would be stored in its raw form first. Any processing that involves sensitive data has to be done in this layer first. Layers (incoming, work, gold, smith, outgoing) in security and non-security layer will be same. It’s just that security layer will be encrypted. After that if needed, data can be moved to non-secure layers but only after masking it. Batch jobs can be scheduled to run on sensitive data to mask it and then the output is stored into incoming layer, gold, and smith layer. Also this layers should be used to unmask data in case it needs to be made available to the downstream target systems.

# Encryption

HDFS implements transparent data encryption that transparently encrypts and decrypts data. Applications have to make no changes in their code. Data can only be encrypted and decrypted by the authorized client. HDFS never stores or has access to unencrypted data or unencrypted data encryption keys. If someone will try to copy files from TDE to other zone it file will not be readable. Security layer would be encrypted using TDE(Transparent Data Encryption).

# Access

Data can only be visible to power users that have secure privileges.

# Directory Structure

* Names in the directory path shall be in lower case.
* Names in the directory path should be spelled out except for well-known company abbreviated names.
* Use underscore “\_” to connect the names if there is more than one word in the directory path.
* For the Incoming data layer, the directory structure is: /incoming/<Functional Source Group Name>/<Functional Source Name>/<AdditionalQualifiers>/initial\_load or ongoing/<Additional Qualifiers>/<Run Date>/<FileName>/
  + Example: /incoming/mkt/mkt\_\*\*\*/initial\_load/mkt\_sales\_transactions/mkt\_slstrn\_2014.txt.04

/incoming/mkt/ongoing/20101227

* For ongoing files, the Run Date needs to be either a level in the directory structure or as part of the File\_Name.
* For the Work data layer, the standard directory name is: /work/<Functional Source Group Name>/<Functional Source Name>/
  + Example: /work/mkt/esb/
* For the Gold data layer, there are two types of formats. The first is for non-partitioned tables, the second is for partitioned tables.
* The standard directory structure for non-partitioned data is:

/gold/<Subject Area>/<Sub-Subject Area>/<Data sub-table name>/<Additional Qualifiers>/”current” or a File\_Name

* Example: /gold/inventory/store\_stock\_status/current
* Example: /gold/time/fiscal\_weeks/fiscal\_weeks.txt
* The standard directory structure for partitioned data is:

/gold/<Subject Area>/<Sub Subject Area>/<Data sub-table name>/<Additional Qualifiers>/<Partition key>/<File\_Name>

* Example: /gold/transaction/pos/mkt/mktsales/transaction\_year\_nbr=2014/part-00089
* The Smith data layer can contain two types of data. Data which is enterprise in scope/use but aggregated between several gold layer tables. It can also contain data which is specific for a team or department.
* The directory format for enterprise data should follow the gold layer format with /smith/ as the high level qualifier.
  + Example: /smith/item/hierarchy/current
* Datasets which are specific to a team or department should follow the following format: /smith/<Team or Department Name>/<Sub-Subject Area>/<Additional qualifiers>/<File\_Name>
  + Example: /smith/dp/dataengine/sales\_cost\_current
* For the Outgoing data layer the standard directory name is: /outgoing/<Functional Target>/<Run Date>/<Sub-Subject Area>/<File\_Name>

**Legend:**

<Functional Source Group Name>: The source system name: pos, cdw, star, sprs, etc.

<Functional Source Name>: This is a more specific source group name that qualifies the data. For example, sywr is a Functional Source Group Name, under this group name, there is epsilon or esb.

<Additional Qualifiers>: Additional qualifiers can be at various levels of a directory structure. Depending upon the source or target, additional levels of detail may be needed to define the structure.

<Run Date>: This is the date when the file lands in the cluster access layer. The format is YYYYMMDD

<File\_Name>: A file name. May contain a sequential number or date component.

<Subject Area>: Data subject area such as customer, geographic, item, transaction, inventory, time, etc.

<Sub-Subject Area>: A further qualifier of subject area. For example, within inventory we have store\_stock\_status or cdfc.

<Partition key>: The format should be the name of the partition key = partition value. For example ‘transaction\_year\_nbr = 2014’.

<Program>: Name of the program that created the data.