UNIT 6 TRENDS IN DATA WAREHOUSE

Structure

6.0 INTRODUCTION

In the earlier units, we have learned the conceptual aspects of a data warehouse including its types and their underlying architectures. You have also seen how new forms of databases evolved viz. Relational, Object-oriented, etc. which, in turn influenced the architectures of data warehouses to encompass the need for accommodating various forms of data. The earlier challenges of data warehousing were successfully addressed through research like innovative architectures, algorithms, modeling techniques, methodologies, etc..

Researchers have been trying to find better solutions for data warehousing and analytics for past few decades. The focus has been limited to relational technologies. Data management solutions began with databases (initially relational), then as the magnitude of data increased the world saw the advent of very-large-database systems (VLDBs). As data originated from various sources and consecutively possessed the directional attribute (via data-sharing and distributed storage, etc.) the research and technological focus evolved towards non-relational systems. Presently, we are in the phase of data–explosion called Big-Data and consequently there have been new innovations in this direction towards managing extremely huge magnitude of data without losing the essence of performance, scalability, reliability, security, etc..

In this unit we will discuss the trends in data warehousing namely data-lakes, complex data-marts, migration of data warehouses to cloud environment, real-time data warehouses, Hadoop software and how it supports data-warehouse and automated data warehouses.

6.1 OBJECTIVES

After going through this unit, you shall be able to:

- understand changing technological landscape related to data warehouse;
- describe data lakes, complex data marts and real time data warehousing;
- understand the concepts of big-data and Cloud-based data warehouse;
- describe how Hadoop supports the data warehouse design pattern;
- understand Data Warehouse automation and its necessity, and
- list examples that support the above data warehousing trends.

6.2 DATA WAREHOUSE – KEY CHALLENGES

When we attempt to understand a data warehouse from a technical perspective we comprehend that it is a database of considerable size ranging from several hundreds of gigabytes to several terabytes and in some cases, few petabytes too. The sheer enormity of database size, the complexity of analytical queries, data mining algorithms and the heterogeneous aspect of data loaded into a data warehouse makes the entire scenario incomprehensible and complicated. These results in significantly acute technological challenges for researchers while raising critical questions for which, solutions appear to be equally intricate or pragmatically irresolute.

It is evident that exhaustive research in data warehousing is going on in the areas namely, conceptual modeling of DWs and logical data models, data warehouse loading (data-refreshing), execution efficacy of OLAP queries and data mining algorithms, materialized views, data analysis techniques, metadata management, evolution management of DWs, stream-based, real-time and active data warehouses, and complex data warehousing (for example spatial, XML, object, multimedia).

Some of the research and technological challenges of utmost significance are dealt through research on topics like amalgamation of heterogeneous data sources, i.e., ETL, cleaning, source characterization, and data integration. Data source discovery, metadata management and standardization, handling data source evolution at an integration layer as well as data flow (ETL) performance optimization are topics which still lack sufficient attention in research. Inadequate research has been done on topics like: data management architectures for standard (table-like) data with unfinished solutions extent. Big data management has exposed many new challenges due to data-explosion, caused mainly by the diversity and velocity of data-generation. Solution development for the assimilation and storage of big data is imperative for

design and implementation of a fully functional big data architecture that can be used by analytics, recommendation systems, visualization systems, etc.

We have discussed the challenges faced by researchers pertinent to data warehousing issues like complications in utilizing semi-structured and unstructured data from various sources, etc. It is evident that the exponential growth in data, including the numerous attributes inherited had a profound impact on the architectural elements of data warehouse. There are yet many capabilities and extensions of functionalities that necessitate their integration into a state-of-the-art system. The additional functionalities if developed severally, are required to function seamlessly as an integral part of the primary system. Architectural impact of issues requires critical thought for implementing solutions that offer optimizing total cost of ownership (TCO) with substantial return on investment (ROI). Since researchers have already embarked upon the journey to build newer architectures to address the proliferation of data and its capture to produce meaningful results, we are likely to see new trends emerging in the form of solutions and challenges for the future.

Let us look at the key challenges faced by researchers which have been categorized based on their technical significance.

Computational techniques: A primary candidate for this research challenge is the computational scalability required with drastic increase in data volumes. Emerging technological trends like Cloud data warehouses (CDWs) may address this issue. Parallel and In-memory computational techniques are being developed and perfected.

Design: Efficacy of design has been of particular significance for researchers. Due to performance implications involved design techniques need to consider the following questions carefully:

a. How can design dependent latencies for cubes be minimized? For example, the computational costs of aggregated data in case of huge datasets may prove unreasonable.

b. How can the update and refresh times for data warehouses (specifically cubes) be improved upon? For example, there exist various techniques and strategies for accomplishing these tasks which tend to fail for unstructured/heterogeneous data.

Quality: The word has different connotations in different contexts and is understood accordingly. For example if the query is complicated the underlying cubes would be overwhelmed with number of *dimensions* and *measures* required to build the cubes, especially if the data is unstructured/heterogeneous in nature. This can be attributed to the skill level of the data warehouse user. On the other hand, if even a simple query cannot fetch data in a fair amount of time then the modeling techniques may be questioned. If the modeling is impeded by lack of functionality then the architecture or design of data warehouse software itself is questionable. Therefore, quality plays a significant role at every stage of the data warehousing process. Good governance and best practices may partially resolve the quality related issues but a holistic scrutiny would reveal any gaps if extant.

Size: The essence of practicality is eliminated when a data warehouse's computational capabilities are throttled. This is caused by the cumulative size of fact tables which tend to expand exponentially over huge datasets.

Operability: A data warehouse needs to provide seamless integration with external data sources for data capture and collection. Interoperability is also another challenge when communicating with various devices and software components. Data warehouses usually make use of Representational State Transfer Application Programming Interfaces (REST APIs) to overcome these issues but security risks increase with the use of such components.

In-memory representation: Distinct from the in-memory computation techniques mentioned above this issue deals with the efficacy of memory-based representation of the data warehouse and its components like OLAP cubes, datasets and result sets. For example, the increasing number of dimensions causes explosive increase in cardinal values for cells. Consequently, secondary and tertiary memory based solutions require critical analysis. SAP HANA which is a High-performance ANalytic Appliance multi-model database that stores data in its memory instead of keeping it on a disk. This results in data processing that is magnitudes faster than that of disk-based data systems, allowing for advanced, real-time analytics). SAP HANA is exemplar of such in-memory database implementation utilizing a cloud platform.

User Centric Interface: Like other tools mentioned above that are required for specific functions it is imperative that the interface allows the user to make optimal use of the data warehousing software. In order to extract maximum benefit (ROI) from the data warehouse implementation the user should not be exposed to the complexities of the system. This challenge has led to a trend of developing autonomous databases equipped with self-monitoring mechanisms and self-healing techniques. There is also a trend towards making as many data warehouse processes automated as possible with a goal of achieving zero maintenance. This is to reduce the maintenance burden on individuals and organizations. It also has other advantages like reduction in human error rates, saving time on maintenance tasks, etc. It also allows organizations to cut down on resources and maintain a lean environment. The user would only intervene if a business decision is involved. A user centric interface also reduces training expenditure while putting the controls at user's fingertips.

Pioneering Infrastructure foundation: Converged, Hyper-converged and Dynamic Infrastructures are deployed in data-centers. These are deployed either on-premises / onsite (private cloud) or on the Cloud (public cloud, e.g. Amazon Web Services (AWS), Google Cloud Platform GCP), Microsoft Azure (MSA), etc.) Or a combination of both Private and Public cloud also called as Hybrid cloud platforms. These platforms can provide an innovative foundation for medium to exceptionally large data warehouse implementations. Also availability of specialized computing infrastructure components like graphic processing units, Artificial Intelligence (AI) or Machine Learning (ML) based Processor chips are available (example NVIDIA's Tesla or Drive PX series processors, AMD's Fire Stream or Radeon Instinct processors). The data warehouse software also needs to improve its architecture and design constructs to derive the maximum benefit from such advanced infrastructures.

Complexity: For example building OLAP cubes over exceptionally large datasets attracts the penalty of rising complexity concerns that are usually unheard of in relational data warehouses. For example, there is an exponential increase in the number of *dimensions* in case of unstructured data in addition to several *measures* due to unstructured data.

Optimization and Innovations in Query Language: Traditional query languages like Multidimensional Expressions or MDX developed for OLAP databases was later wrapped in XML to be called mdXML to include XML data retrieval. With data undergoing more diverse forms and originating from equally varied sources the query languages require serious thought to improve their performance and capabilities to handle sufficiently large data from disparate sources while being able to build and populate cubes through query plan optimizations.

Data Visualization: Many organizations and institutions capture data from varied sources to assist in strategizing, business decisions, weather forecasts, predicting climate changes, knowledge visualization, cognitive maps, etc. In all these instances the sheer amount of data collected needs to be managed efficiently on various fronts to produce viable reports in order to visualize data. In healthcare systems that are dynamic in nature a patient's health parameters captured via various sensor devices need to be analyzed and evaluated in real-time. Other dynamic models like Climate

change prediction systems need to monitor data captured via Internet of Things (IoT) devices are used to predict weather/climate anomalies or impending catastrophes. These scenarios require state-of-the-art software and infrastructure components. The algorithms implemented via software and computed using the infrastructure has to make sense of the data and at times discover hidden meanings and therefore new knowledge. This requires specialized query tools for example GaphQL, etc. while at the same time the semi / unstructured aspect of data requires appropriate management software that is architecturally robust. There are other critical issues with AI / ML based data visualization like necessity for innovative algorithms, multi-parameter based optimizations, accuracy of results, etc.

In the next section, we will learn about the Data lakes.

6.3 DATA LAKES

A data lake is a central location that holds a large amount of data in its native, raw format. Compared to a hierarchical data warehouse, which stores data in files or folders, a data lake uses a flat architecture and object storage to store the data. Object storage stores data with metadata tags and a unique identifier, which makes it easier to locate and retrieve data across regions, and improves performance. By leveraging inexpensive object storage and open formats, data lakes enable many applications to take advantage of the data

.

Data lakes were developed in response to the limitations of data warehouses. While data warehouses provide businesses with highly performant and scalable analytics, they are expensive, proprietary and can't handle the modern use cases most companies are looking to address. Data lakes are often used to consolidate all of an organization's data in a single, central location, where it can be saved "as is," without the need to impose a schema (i.e. a formal structure for how the data is organized) up front like a data warehouse does. Data in all stages of the refinement process can be stored in a data lake: raw data can be ingested and stored right alongside an organization's structured, tabular data sources (like database tables), as well as intermediate data tables generated in the process of refining raw data. Unlike most databases and data warehouses, data lakes can process all data types — including unstructured and semi-structured data like images, video, audio and documents — which are critical for today's machine learning and advanced analytics use cases.

6.3.1 Need for a Data Lake

First and foremost, data lakes are open format, so users avoid lock-in to a proprietary system like a data warehouse, which has become increasingly important in modern data architectures. Data lakes are also highly durable and low cost, because of their ability to scale and leverage object storage. Additionally, advanced analytics and machine learning on unstructured data are some of the most strategic priorities for enterprises today. The unique ability to ingest raw data in a variety of formats (structured, unstructured, semi-structured) along with the other benefits mentioned, make a data lake the clear choice for data storage.

6.3.2 Data Warehouse Vs Data Lake

A data warehouse stores structured business data in its processed form. This approach requires fairly rigid schemas for well-understood types of data. While data warehouses are an important tool for enterprises to manage their important business data as a source for business intelligence, they don't work well with unstructured data.

Data lakes allow the storage of raw data, both relational, as well as non-relational that is intended to be used by data scientists and developers along with the business analysts.They take the data out of the silos and make it accessible to all business users promoting centralization of data.

The key differences are summarized in the Table1 given below:

Table 1: Data Warehouse Vs Data Lake

Attribute	**Data Warehouse**	**Data Lake**
Type of Data	Structured data from sources like transactional systems and operational databases.	Raw Data from varied sources like websites, mobile apps, IoT devices, social media channels etc.
Schema	Schema-on-write	Schema-on-read
Intended users	Primarily business analysts	Data scientists, developers and business analysts
Price/Performance	Fastest query results using higher cost storage	Query results getting faster using low-cost storage
Data Quality	Highly curated data that serves as the central version of the truth	Any data that may or may not be curated (ie. raw data)
Type of analytics	Business intelligence, visualization and batch reporting	Machine learning, predictive analytics, profiling and data discovery.
Agility	Fixed configuration, less agile	Highly agile, can be configured and reconfigured as per requirements.
Security	More secure storage	Higher accessibility makes ensuring security a challenge

6.3.2 Data Lake Maturity

A data lake is said to go through various stages of maturity during its life-cycle, considering that it is a fairly new design pattern. Bill (William) Inmon (the father of data warehouse) proposes data lake maturity process (or stages) as below.

According to ***Bill Inmon***, native or raw data is classified into three discreet categories viz. i) Analog data, ii) Application data and iii) Textual data. He states as below:

"In order to organize the different types of data into a structure that can be analyzed, it is necessary to create a high-level structure of data within the data lake. As data enters the lake it first enters the raw data pond. The purpose of the raw data pond is to serve as a holding cell. There is little or no analysis or other organized activity of the data while in the raw data pond.

Once it is time for analysis, the information in the raw data pond is sent to one of three different ponds based on the kind of data entailed. For example, analog, application and textual data all require a unique data pond." – ***Bill Inmon***

When the data has passed its useful life in the data pond it is moved from its respective data pond to an archival data pond as illustrated in Figure 1 below.

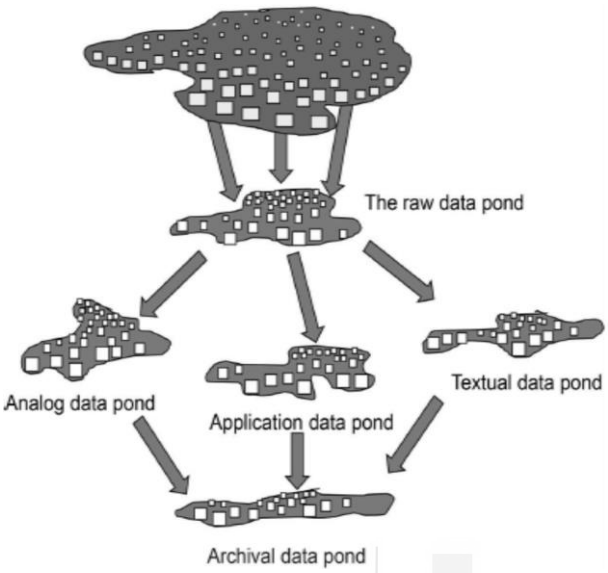

Figure 1: Data Lake Architecture by Bill Inmon

Alternatively ***Alex Gorelik*** introduces additional data lake stages, as part of data lake's maturity aspect, as below:

a. **Data Puddle**: A data mart built using big data technology with a sole purpose or as a single-project with data being loaded into it for the consumption of a single project or team. The data mart concept is well known and use of big data technology is mandated to reduce cost and improve performance.

b. **Data Pond**: Similar to that proposed by Bill Inmon, except in this case it is a collection of data puddles in the form of collocated data marts or similar to a poorly designed data warehouse with minimal transformation of source data. As such these data ponds restrict data usage solely to the projects that necessitate it.

c. **Data Lake**: It caters to business users in two significant aspects. Firstly as a self-support service through which business users are able to utilize the data and secondly, provisioning data to business even when not required by the project.

d. **Data Ocean**: Facilitate enterprise-wide availability of self-service data and data driven decision making, irrespective of the data location or its existence within the data lake.

The above components are shown in Figure 2 depicting data lake maturity process proposed by Alex Gorelik.

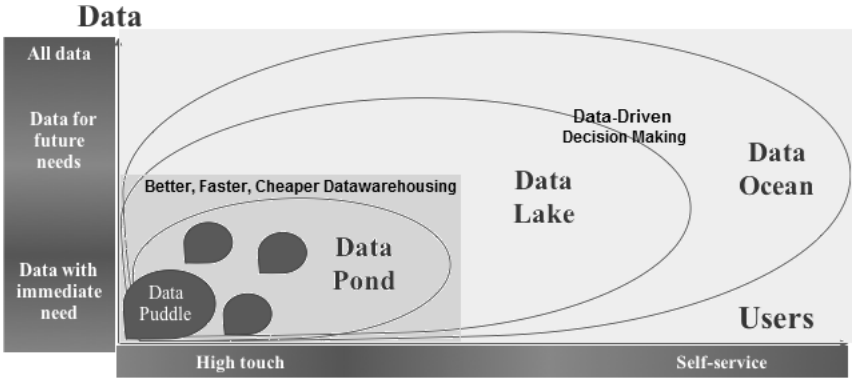

Figure 2: The Enterprise Big Data Lake by Alex Gorelik

6.3.3 Data Lake Architecture

The below given Figure 3 shows a standard proposal for an architecture of a data lake system. The system consists of four layers which are i) Ingestion Layer, ii) Storage Layer, iii)Transformation Layer, and iv) Interaction Layer.

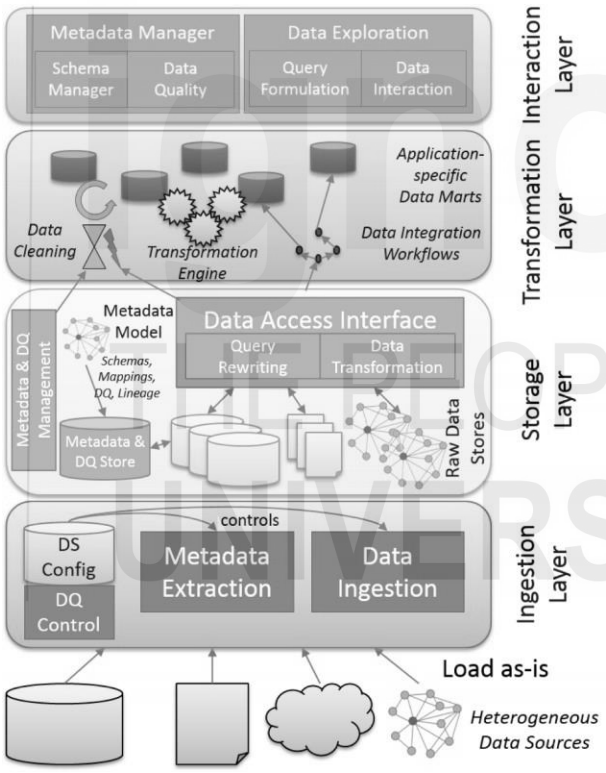

Figure 3: Layered Architecture of a Data Lake

i) Ingestion Layer

The Ingestion Layer is in charge of bringing data into the data lake system from various sources. One of the main features of the data lake concept is the ease with which any type of data can be ingested and loaded. Data lakes, on the other hand, have been repeatedly reported as requiring governance in order to avoid becoming data swamps. Administrators of data lakes are in charge of this critical topic.

The metadata extractor is the most key component of the ingestion layer. It should make it easier for data lake administrators to configure new data sources and make them accessible in the data lake. To accomplish this, the metadata extractor should extract as much metadata as possible from the data source (for example, schemas from

relational or XML sources) and store it in the data lake's metadata storage. The raw data, in addition to the meta- data, must be absorbed into the data lake. Since the raw data are kept in their original format, this is more like a "copy" operation, which is certainly less complex than an ETL (Extract-Transform-Load) process in data warehouses.

ii) Storage Layer

The *metadata repository* and the *raw data repositories* are the two key components of the storage layer. Since raw data repositories must be stored in their native format, data lake environments must support a variety of storage systems for relational, graph, XML, and JSON data. Hadoop appears to be a strong candidate for the storage layer's basic platform. To support data fidelity, however, additional components such as Tez or Falcon are needed.

iii) Transformation Layer

Data can be transformed from storage to user experiences using the transformation layer. It requires operations such as cleansing, format transformations, and so on.

iv) Interaction Layer

All of these functionalities that are needed to work with the data should be covered by the interaction layer. These functionalities should include visualization, annotation, selection and filtering of data, as well as basic analytical methods. It's also worth noting that more advanced analytics like machine learning and data mining should not be regarded as part of a data lake system's capabilities.

6.3.4 Data Swamp

A data swamp is a data pond that has expanded to the size of a data lake in the absence of self-service and governance facilities. At best, the data swamp is used like a data pond and at worst it is left unused. While various teams may frequently use areas of the lake for their projects the bulk of the data is dark/obfuscated, undocumented and therefore, unusable.

Data swamps can be accessed by those who are technologically proficient. This is accomplished by incising or chiseling small puddles for themselves and their teams. At the advent of data lakes many enterprises hastily bought Hadoop implementations and loaded them with petabytes of raw data, from various sources, without clarity on the manner of its utilization. In the absence of any systematic approaches to organize the data while loading it turned into an incomprehensible mess. Additionally, enterprises were prohibited by governance regulations against permitting data swamps' access to a wider audience without obfuscating sensitive data. Since the precise location of sensitive data was dubious, access was interdicted. Consequently, the data essentially remained unusable and unutilized.

☞ Check Your Progress 1

1) What are present challenges in data-management?

.....

.....

.....

2) Describe in your own words the logical data lake concept?

.....

.....

.....

3) What is your interpretation of a successful data lake and a data swamp?

6.4 COMPLEX DATA

6.4.1 Introduction

We have already seen how big data has become pervasive across enterprises. We have also seen different ways in which data is generated through various sources like emails, files, IoT, Logs, etc. Enterprises. With the scale and speed of data adding to the type of data being generated, the term Complex data was coined to define all data that does not conform to standard data types like dates, currency, alphabets, numbers, etc. Complex data is still essential, as we have seen when dealing with raw data, in building data-insights and big data analytics. Complex data is processed as complex datasets and is used by pattern matching algorithms in Machine Learning models. Here we take a quick look of how the complex data trend has influenced the data-driven technologies and enterprises.

6.4.2 Complex data modeling

With data streams traversing many hubs and myriad technologies complex data modeling have become a standard practice across the industry. Complex data travels through data processing tools, ETLs, ERP software, data lakes and other paths. Complex data has its intrinsic traits like its own syntax, schema, technology, terminology and type. This diversity of traits complicates the work of data modelers.

Various models like statistical, polyglot, no payload, multi-level, etc. exist to store, process and analyze complex data. Some standard models are also available like Anchor models and Data Vault Models. Both of these models provide advantages like Scalability, Temporal data handling and are resilient to structural and content-based changes.

6.4.3 Complex data models

6.4.3.1 Anchor Model:

An example of the Anchor Model is shown in Figure 10a.

Anchor modeling has four elements to it:

Anchors
The anchors are used to model entities and events

Attributes
Attributes are used to model properties of anchors.

Ties
Ties are used to model the relationships between anchors.

Knots
Knots are used in the modeling of shared properties, such as states.

Attributes and ties can be versioned (termed as historization), when changes in the information the model need to be retained.

The different graphical symbols, representing the elements of the model, are similar to those used in entity-relationship modeling with a few extensions. An outline on a tie or attribute depicts versioning of changes.

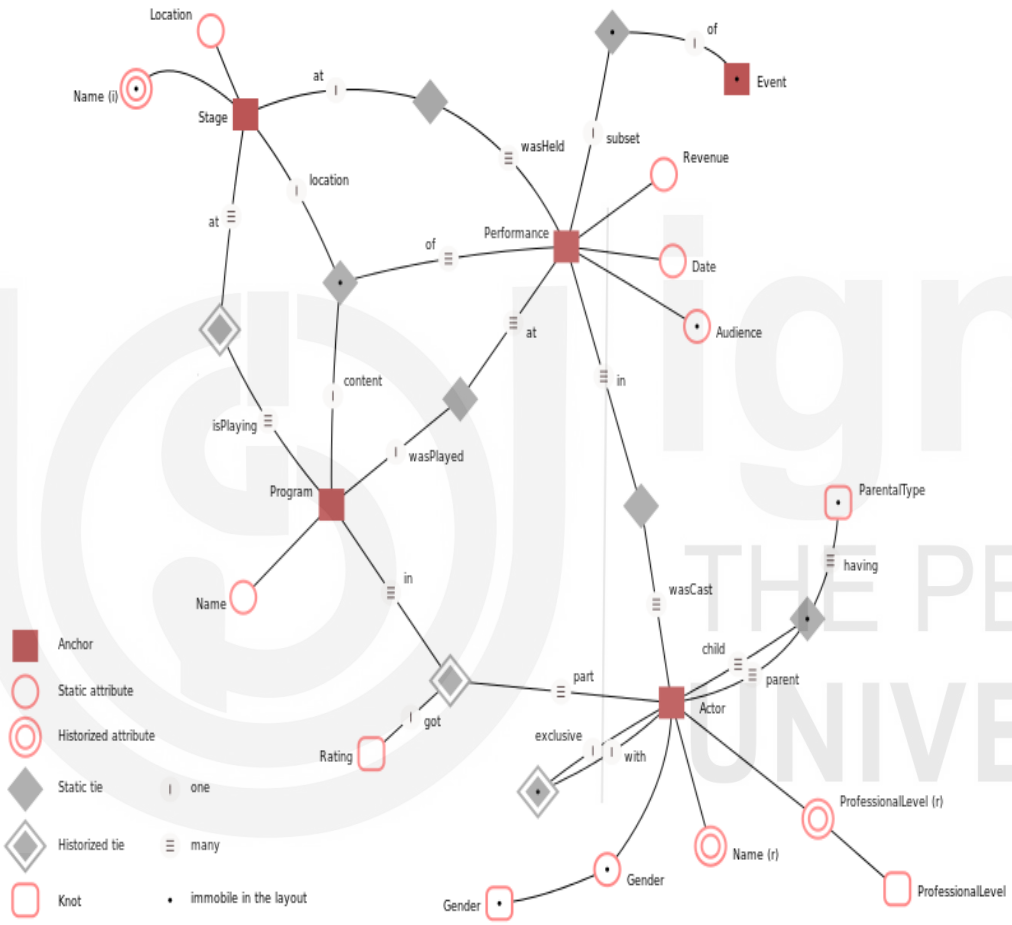

Figure-10a. An example of the Anchor Model

6.4.3.2 Data Vault Model

Data Vault is a database modeling technique offering long-term storage of historical data arriving from multiple operational systems. The model also allows coping with issues such as auditing, data traceability, data loading performance and resilience to change. Data traceability implies that every row of data in a data vault must be associated with a record source and date-of-load attributes. This information allows an auditor to trace values to their respective sources. Daniel (Dan) Linstedt developed the Data Vault model in 2000.

Figure 10b illustrates the Data vault model. The components of this model are briefly discussed below:

Hubs:

Hubs are comprised of a list of distinct business keys with low affinity to change and contain a surrogate key per Hub item. The Hub also contains metadata designating the source of the business key. Satellite tables; discussed below, store descriptive attributes for the data on the Hub (for example, a multiple languages description for the key). A Hub should have at least one satellite and at the least, should comprise of the following attributes:

Surrogate key: Connects the other structures to the table comprising the surrogate key
Business key: The driver attributes of the Hub structure may contain multiple fields.
Record source, Allows determining which system was the first to load a business key
Metadata (optional) Contains information on manual updates (user/time) and data extraction date.

Existence of multiple business keys is precluded in a Hub, except when two systems provide the same business key but having different meanings to resolve conflict.

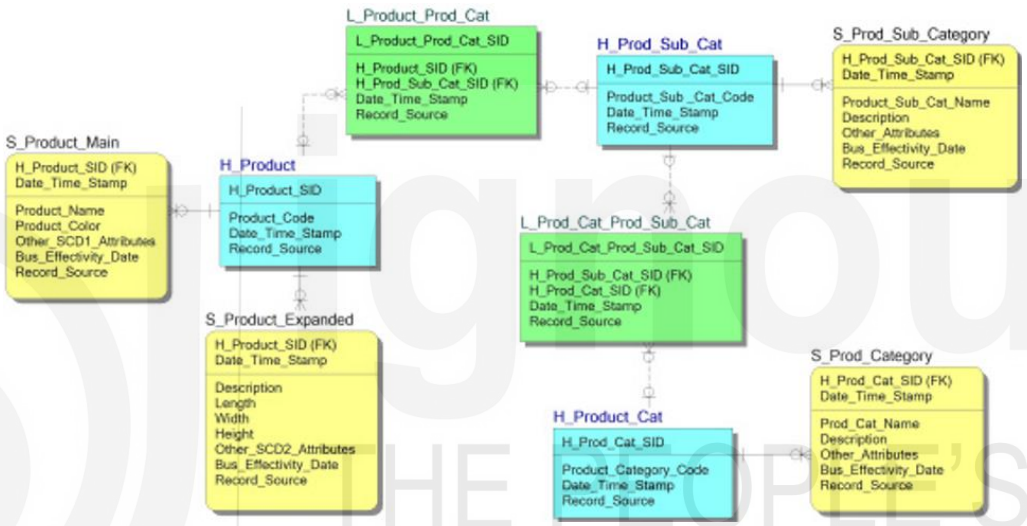

Figure-10b. An example of a simple Data Vault model with two hubs (blue), four satellites and a link (green)

Links

Link tables are models providing relations between business keys and are fundamentally many-to-many join tables, containing some metadata.

Links can connect to other links, to handle granularity variations. Using link references within another link is bad modeling practice, as it creates dependencies between links making parallel loading of links more challenging.

Sometimes links connect hubs to data that is not adequate for hub creation. This scenario happens when a non-real business key is associated with the link. This happens only when using the business key for another link or as key for attributes in a satellite is not possible. This technique has been called a 'peg-legged link' by Dan Linstedt on his (now defunct) forum.

Links also comprise of surrogate keys for linked hubs. The surrogate keys of the hubs, for the link and metadata describe the relationship source. As in hubs, the descriptive attributes for the relationship information gets stored in the satellite table.

Satellites

The hubs and link's temporal attributes and descriptive attributes stored in separate constructs called satellite tables or simply satellites. The satellites also comprise metadata connecting them to their parent link or hub, metadata designating the relationship source and attributes, also including a time-series with the attribute's start and end dates. While the links and hubs deliver the skeletal structure of the model, the satellites deliver the "meat or substance" of the model in the form of business processes contexts that are stored in links and hubs. Storage of these attributes is done relative to the timeline and content details and span from relatively simple (a linked satellite with only a timeline and a valid indicator) to relatively complex (all fields designating a client's entire profile).

Typically, the source system groups attributes in satellites. Rate of change for descriptive attributes like amount, size, color, speed or cost may vary. These attributes can be separated into various satellites based on the rate at which they change.

Metadata is contained within all the tables, designating the validation-date of the entry and system from which it originated, providing a holistic historical perception of the data as it is loaded into a data warehouse.

☞ Check Your Progress 2

1) Explain your understanding of complex data and the need for complex data modeling

.....

.....

.....

.....

2) Write a note on the Data Vault model.

.....

.....

.....

.....

6.5 CLOUD DATA WAREHOUSING

6.5.1 Introduction

The transmogrification of raw data into insights is a convoluted challenge and the solution is not simple. That challenge is made worse by the accelerated pace of environmental dynamics affecting the enterprises. The pace at which business is done is in fact accelerating. Things that we expected to be done in a week or overnight or an hour need to be done in real time. Innovations in various industries are taking place simultaneously or at an overwhelming pace.

The challenge for most enterprises is to discover opportunities to exploit this change rather than being overpowered by it. The answer, though superficially simple, lies in leveraging data and transform into a data-driven enterprise. This transformation should not be limited to a technological perspective, rather it needs to be assimilated and trusted in by people across the enterprise as the crux of organizational health and wellbeing.

Enterprises have to confront demands of space and location (capacity to store and utilize voluminous data. Further, new demands imposed by regulatory and compliance mandates, data velocity, new data sources, performance, scalability, QoS, etc. have compelled the enterprises to rethink their data-leveraging strategies.

Enterprises have realized that a data warehouse implementation is no more a silver bullet for their problems considering the environmental dynamics and other issues discussed above. Therefore, we have recently witnessed the trend of extant data warehouses migrating to cloud while new ones are adopting cloud as their data warehouse implementation platform. This forethought is a consequence of realizing issues of capacity planning, infrastructure costs, usage policies, governance, flexibility of utilization based expenditure and many more.

6.5.2 Reasons for Migrating to the cloud

On-premises data warehouse implementations are reducing and consequently its business is diminishing too. Vendors like Google BigQuery, AWS Redshift, Snowflake, Azure Synapse Analytics, etc. are popular ones for cloud hosted data warehouses. These vendors offer near-instantaneous installations with storage, computing and network resources with increased performance and scalability benefits. The supporting software and platform are always maintained up to date. Most enterprises or a new data lake tend to choose a cloud-hosted data repository.

Enterprises are migrating to cloud data warehouses because they get instant access to all the infrastructure resources necessary to scale that solution. They also derive a pay for use benefit as they pay only for additional memory, compute, storage, networking or other resources they might need to scale. Best-of-breed applications can also be availed through SaaS to their cloud-implementations, whereby their entire business intelligence and analytics stack can be provided as a service. They can scale up or scale down as required with minimal installation or maintenance charges. The trend towards cloud-based data warehousing is clearly evident and that is where this market is progressing.

To sum up, enterprises are adopting the cloud migration path due to the following advantages offered by a cloud platform:

i. Improved performance
ii. Flexibility of cost through flexibility of cloud resources utilization
iii. Migration of existing products to the cloud environment
iv. Derive benefits that are inherent to a cloud platform

6.5.3 Challenges of cloud data warehouses

Although cloud-based data warehouses are an essential component of the future of enterprise-data, there are additional tasks to be performed and therefore, challenges involved. The significant challenges of hosting data warehouses in a cloud environment are as below:

i. Data extraction, transformation and subsequent loading
ii. Context and User based Data access
iii. Management of heterogeneous data velocity
iv. Ensuring instant availability of new data sources
v. Ensuring data quality
vi. Management of sensitive data and compliance to regulatory mandates
vii. Interoperability with tools and infrastructure external to cloud environment
viii. Communicating with legacy systems that could not be migrated to cloud for technical and organizational reasons

ix. Data governance and obfuscation of sensitive data
x. Automation of data offload for sophisticated analytics and ML

6.5.4 Building a successful cloud data warehouse

Building a successful CDW entails overcoming the challenges mentioned in the previous section. Since cloud data warehouses deal with enterprise data at scale, eliminating or reducing the impediments in its path to derive maximum business value forms a crucial element in building a successful CDW. To ensuring a successful CDW implementation the CDW has to go through the following stages.

Stage 1 - Formulating an approach for data curation at scale and data integration

Firstly, enterprise resource scalability is imperative with increasing volumes of data that results from business growth. A precise and pertinent data curation and integration approach is a deciding factor in an enterprise's ability to succeed as a data-driven enterprise. This success in turn is based on an enterprise's ability to leverage data analytics based insights ahead of its competition.

In order to carefully plan the approach to data integration the following considerations are essential:

i. Preparedness for enterprise changes, besides technological changes, or the ability to prepare for such scenarios.
ii. Enterprise readiness to adopt new technologies including unknown ones developed in the future
iii. Future objectives of your enterprise that would have a critical impact on your data integration architecture already in place
iv. Enterprise expectations from this approach in terms of adding measurable and tangible value to its business growth

The above considerations entail due diligence in solution implementations as below: Transformation into a data-driven enterprise involves changes on the technological front but also at various other levels. These changes affect both the technical and business stakeholders alike. To list a significant few, the changes involve enterprise risks, costs, dependencies, ROI, benefit-amendments, assumptions, and cultural issues. For large enterprises, this necessitates the presence of change management professionals, who educate, train and influence stakeholders mindsets to the imminent changes. Enterprises planning to undergo a data-driven transformation may hire change-management agencies or professionals based on the scope and dynamics of their businesses growth.

Architecture of systems employing data integration technologies should be flexible enough to accommodate new tools in the future with minimal impact on the architecture and consequently the system services. Conceptualization of future-proof architectures is essential to make a system capable to adapt to yet unknown technologies that might be developed in the future.

Enterprises may have plans to expand their data-leveraging capabilities, for example moving their Enterprise Resource Planning (ERP) data i.e. data from ERP modules like Customer Relationship Management (CRM) / Supply Chain Management (SCM) / Human Resource Management (HRM), etc. to the cloud data warehouse. However, the actual objectives of the enterprise might involve efficient master data or workflow management to emerge as a leader in customer services landscape. Such objectives need to be stated upfront in order to formulate the best approach that caters effectively to present and future business requirements.

Wisdom dictates that expenditure on the latest and greatest technological tools might not be adequate for business growth if they cannot cater to the specific functions of an enterprise. Also, availability of great feature-sets in a tool does not necessarily guarantee business growth if only a few of them can be truly exploited to an enterprise's advantage. Tools that offer customization or flexibility in terms of specific functionality needs, choice of features, etc., in a data integration platform are available. A clear understanding of the customizations involved allows technology decision makers to understand the implementation components required to collate data and ensure its availability across the enterprise. It also helps the enterprise understand TCO of the data integration platform, which helps to optimize and improvise the approach to extract maximum ROI. Further, the approach also enables a precise understanding of the business benefits expected from a data integration platform and subsequently, their measurability and tangibility.

Stage 2 – Leveraging Data Integration platform for on-demand data provisioning

It is imperative for an enterprise to develop faster decision making capabilities. However, the vast volumes and heterogeneous characteristics of data make it extremely difficult to do so if it is not able to scale its infrastructure and tool-sets to match the rate of data explosion (i.e. rate of increase of data velocity, diversity and volume). Data-explosion also triggers the need to assimilate and utilize data at equivalently faster rates i.e. before its business worth diminishes. The fresh the data is, the most up-to-date the data visualizations, which give an almost instantaneous view of the most recent business landscape. These representations allow an enterprise to make informed decisions to strategize, improvise, adapt and steer their business on the path to success.

To extract maximum value from data, the right data should be available to right people, and at the right time. After the enterprise has soured huge volumes of data it may develop the capability to store. This data needs to be transformed based on business requirements so that it is available to various people with specific expertise related to business areas. For example, data engineers should have access to raw data, analysts should have access to curate and organized data (e.g. in relational format), and other business functions like human resources, finance, manufacturing etc. should have access to their respective data sets. Sensitive data would also require access controls and obfuscation for cross functional requirements. All this advanced work will necessitate time and resources, creating latencies at different levels. The latencies at different levels like data extraction, data storage, organization and management, access control, access to desensitized data, transformation, loading, etc. Also, if the data cannot be visualized at scale through the use of appropriate tools the business decisions get delayed. It is easy for us to understand now; that in order to reduce the machine level latencies a meticulously planned data integration approach is imperative.

An approach to a fully optimized data integration platform to supplement business data needs of business will allow data-sharing across the enterprise, irrespective of functional domains and thereby, enhancing data visualizations through combined insights. Data provisioning puts data at the fingertips of stakeholders with very little dependence on technical personnel thus allowing for self-service. As a result of self-service capabilities operational latencies are drastically reduced while at the same time increasing innovative data insights. As ideas and representations are shared across functional domains, they can be improved upon through collective inputs and critical thinking. A thoughtfully planned approach should consider the advantages within the boundaries of data governance.

Stage 3 – Ensuring high quality of heterogeneous data across CDWs

As already mentioned, in Stage 1 above, utilizing a vast collection of the best tools does not guarantee best results if the tools cannot be customized to the business requirements of an enterprise. The technical dependencies cannot be resolved in case of some tools that offer deep customizations and a host of features (of which, all might not be applicable). Conversely, tools that are easy to implement may have limited capabilities and therefore cannot be classified as enterprise applications and their implementation, despite, is likely to cause more operational latencies than if not implemented at all. Therefore, it is necessary to scrupulously evaluate and scrutinize various tools, for your data platform so that they can meet the current and future needs of the business. Ensuring high quality of data availability, ubiquitously across your CDWs, is now possible through cloud-aware ETL tools which people with basic technical knowledge can easily use. The cloud based ETL tools are compliant to security and regulatory requirements and employ state-of-the art AI/ML techniques making the data reliable and accessible. As discussed in Stage 2, data sharing is necessary for timely decision making through mitigation of operational latencies. Implementing the right set of tools across different functional domains can expedite analytics and decision support capabilities.

☞ Check Your Progress 3

1) What are the reasons for migrating to a Cloud data warehouse and what are the challenges involved for a CDW?

.....

.....

.....

.....

6.6 REAL TIME DATA WAREHOUSING (RTDWg)

6.6.1 Introduction

What follows on from the discussion on various stages of implementing a successful cloud data warehouse, in the above section, is the most essential and significant attribute of a data-driven enterprise, which is timely availability of data across the enterprise. Timely or instantaneous data availability combined with appropriate access has many connotations, based on performance, context, relevance, perspective, etc. The rationale behind this expectation is that in a data-driven enterprise, data undergoes various stages before it can be available to the management in the form of meaningful representations. The representations or data visualization reports need to be based on trustworthy and most current data to be of any worth to the business. To make this possible the underlying processes – of sourcing the data, transforming it and making it available to various functional domains across the enterprise – need to happen as optimally as possible within the shortest possible time.

A birds-eye view or a look at the bigger picture makes it evident that the data platform that drives the enterprise should essentially work on-time across every stage of the journey of data. If we look at the detailed level, we realize that every element expects data to move immediately or in real-time. The laws of science do not make the real-time concept possible; therefore, real-time terminology is sometimes altered to near-real-time. Enterprises set performance thresholds for their real-time interpretation, and if data warehousing capability falls within these thresholds, it is termed a real-time data warehouse (RTDW). The proliferation of new data or updates to actual data across a data warehouse immediately has become synonymous with real-time data warehousing and is the favored definition in the industry.

.

CDWs are an excellent candidate for enterprises wanting to implement real-time data warehousing because of their intrinsic feature to scale resources e.g. computational, memory-related, storage, networking, etc. Despite the best resources available, processes like data transformation, and data loading into a dimensional model are distinctively complex. Also, taking into account the massive amounts of data involved, the complexity gets further compounded. Therefore, the processes for updating new data coming in from different sources require specialized handling to make the latest data available in real-time.

We already know that the primary component of data warehousing i.e. the ETL process takes care of data movement from its source to the data warehouse storage and is extremely resource intensive. The mandates of real-time data warehousing require drastic alterations to the way ETL process takes place in addition to other components of the data warehouse. Researchers have come up with some unconventional approaches e.g. novel ETL architectures, update methods – cached, partial, separate synchronized DWHs, etc., to ensure data availability in real-time. Figure 11 is a conceptual representation of real-time big data analytics which gives us a glimpse of what it means to implement a real-time data warehouse.

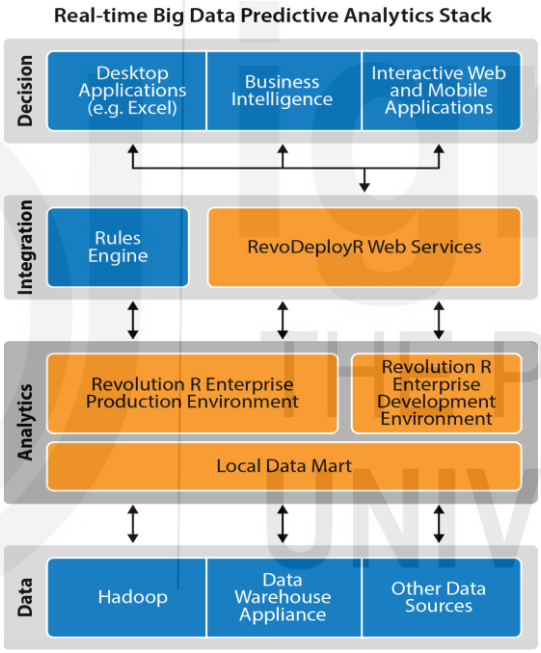

Figure-11. From David Smith's presentation "Real-Time Big Data Analytics: From Deployment to Production"

Hadoop is a popular choice at the data-layer as it offers many tools for ingesting and transforming data in real-time. It also allows for integration of other tools like Spark for replacing the built-in MapReduce tool. For example, the Spark engine that powers the Shark (a RTDW) data warehouse gives a 10-100 times faster performance than MapReduce. The Analytics and Decision layers are bound together by the Integration Layer and this composite collection of layers is fed through various components of the data layer.

6.6.2 Real-Time Data Warehouse architecture

The Integration Layer binds together the Analytics and Decision layers and this composite collection of layers is fed through various components of the data layer. We can see that these changes expose atypical constructs that are essential for the way a) data sources are added/updated, b) data is propagated across the warehouse and c) data is loaded into various repositories for heterogeneous data. We discuss a few

architectures that were proposed over the past decade and have greatly influenced some of the popular data warehousing components today.

The proposed construct comprises of following component blocks:

i) Data production systems hosted on *data sources* and responsible for populating the data warehouse. This block comprises of a source flow regulator (SFlowR) that periodically (based on preset or customized policy) transmits data to the data warehouse after the data has been identified for pertinent changes.

ii) The *data processing area* (DPA) responsible for data-quality and transformation. This block contains the data flow regulator (DFlowR) used for detecting the transmission-ready source.

iii) After cleaning, an ETL workflow collects the records from the transmitting data source and transforms them to data warehouse format. The DPA also has other myriad responsibilities like ensuring QoS, checkpoint generation, caching data into reservoirs if the data warehouse pipelines are full, etc. Post-processing, the data transmission to the data warehouse, is orchestrated by a workflow regulator (WFlowR) component.

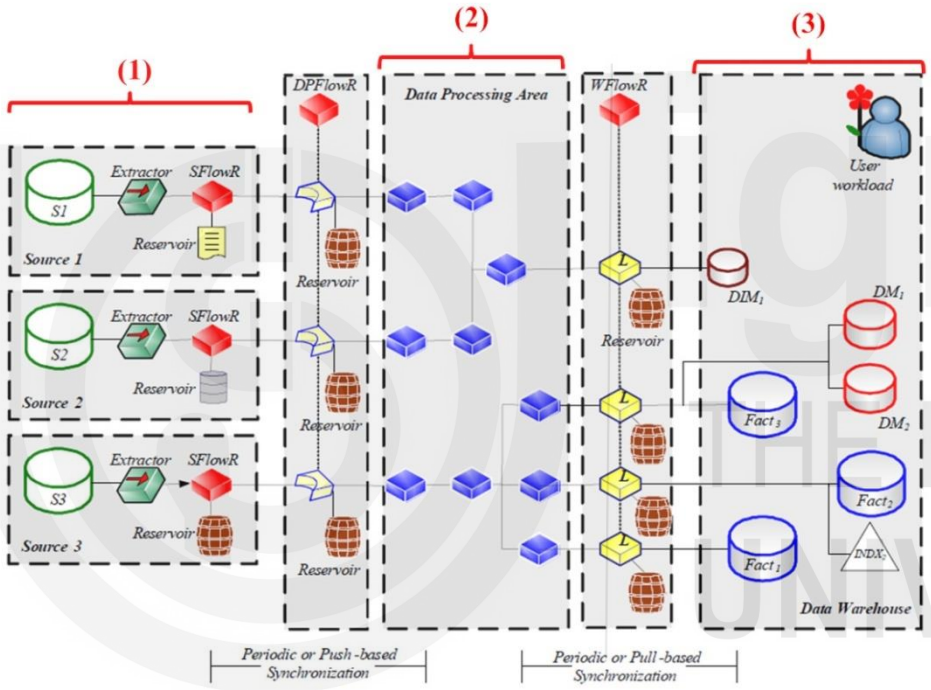

Figure-12. Near-Real-Time data warehouse architecture.
(courtesy - Vassiliadis, P., Simitsis, A.: Near Real Time ETL)

iv) The *data warehouse* itself. The data warehouse is comprised of dimensions, fact tables, materialized views and indexes (primarily bitmaps and B+ trees) which makes it a complex repository. Data propagation through this block depends on the flow regulation components and the real-time aspect is ensured through the successful performance of these components.

An alternative architectural construct is proposed by Obali and his team is shown in Figure-13.

The architectural components are: a) Metadata b) Web Service (WS) client, c) Web Service provider, d) ETL e) Real-time Partition, f) Real-time data integration and g) Data warehouse.

Change Data Capture (CDC) designating the modified data is initiated by the WS client and sent to the WS provider through a related web service. CDC involves use of various techniques for identifying changes to data like: direct extraction of source data if it is new, timestamp-based, application aided, file comparison based and trigger

criteria based data capture. Data is populated in the real-time partition through a WS provider web service based on the data transmitted by WS client. The received data is separated into data and metadata. A structure query language (SQL) is generated, by a web service through SQL generator, using metadata and inserting data into data warehouse log tables. The generated SQL is executed to populate the data warehouse data repository (database).

The data orchestration between the data warehouse and real-time partition is handled by the real-time data integration component which acts as glue between the two. When there is a user-request for historical data the real-time data integration component generates a query that is sent to the data warehouse. In case of a hybrid data request (both historical and recent) a query is generated which integrates the data from the real-time partition and the data warehouse.

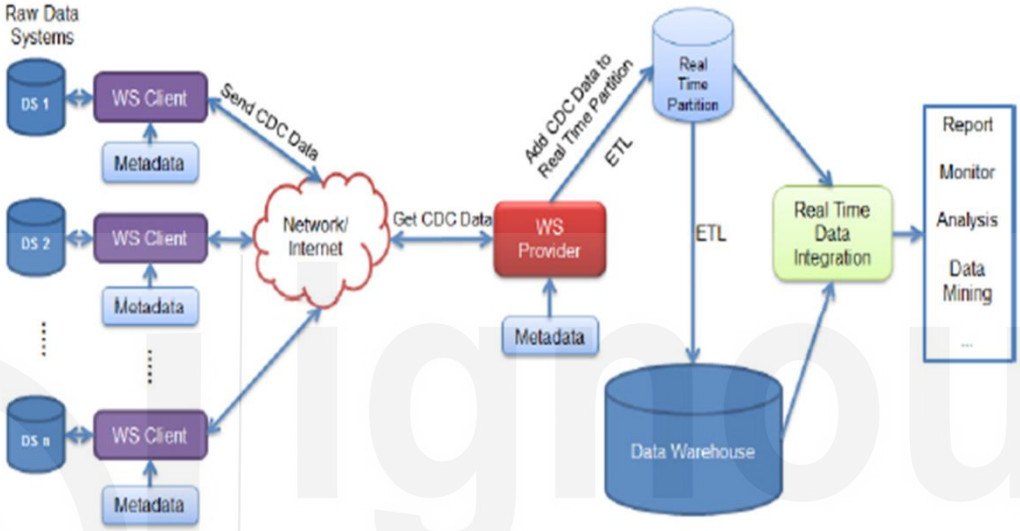

Figure-13. Real-time data warehouse architecture based on web serices (courtesy: Obali, M., Dursun, B., Erdem, Z., Grr, A.K.: A real time data warehouse approach for data processing.)

Yet another architecture by Ferreira and his team can be seen in Figure-14.

The architectural components involve: Data brought into an Input Stream Buffer component in a packaged format from the data sources component. The packaging is meant to alleviate the burden of identifying the data for changes. A transform ETL component brings cleaned data to a Dynamic DW (DDW) component from where the historical data is moved to the Static DW component. A Merger component, used for querying data from Static and Dynamic data warehouse (M-SDDW) handles the query generation directed towards the appropriate data repository for relevant user-request.

Based on the integration policies the data from either of the independent repositories (D-DW or S-DW) can be integrated just in time. Also data completeness can be achieved, for example through foreign-key resolutions or existence of related transactional information.

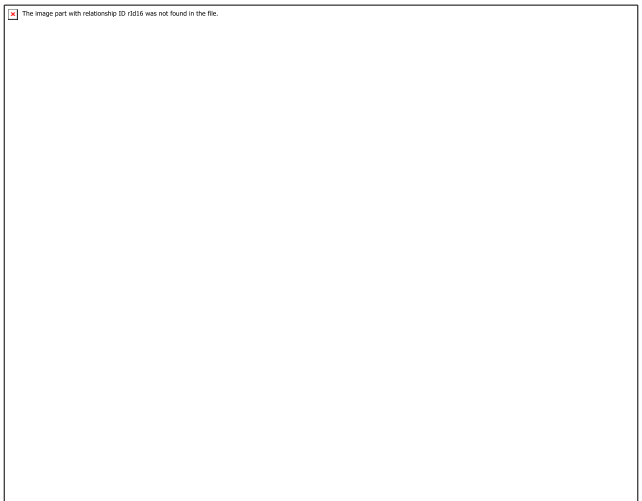

Figure-14. Real-time data warehouse architecture model proposed by Ferreira et al (courtesy: Ferreira, N., Martins, P., Furtado, P.: Near real-time with traditional data ware-house architectures)

6.6.3 Real-time Data Warehouse Architecture Tradeoffs

It is understood from the above architectures, few changes required across the entire data warehouse ecosystem to implement a real time data warehouse. Consequently, there are a few architectural tradeoffs that users of RTDWs need to understand.

The principles of ETL system are presumed to remain unchanged on the journey to a RTDW. Also responsibilities towards data integration, backup, recovery, archival, regulatory compliance, security, quality, governance, etc. are to be maintained. The trade-offs that ensure on the RTDW journey is briefly explained as below:

6.6.3.1 Batch file replacement

Use of message queues and transaction log files are recommended by Ralph Kimball, replacing batch files. From a batch file's perspective, the data is complete with all keys resolved. Data from message queues and transaction logs is raw data that does not fall within the purview of business rules or corrective processes and is instantaneous.

6.6.3.2 Restrict scrutiny of data quality

Here the scrutiny is applied to columnar data. As data requests start to become more demanding, some data quality will need to be compromised by decreasing mandates on data quality. While this is done, users will need to be made aware of data unreliability and may require compensatory measures in the form of corresponding batch oriented ETL pipeline that periodically replaces real-time data with complete and accurate data.

6.6.3.3 Publish Facts with Dimensions

Old records of dimensions should be allowed to contain facts that are received early. For example, transaction details for a customer might be available even before a customer or item code is generated. We notice that the facts are available prior to dimensional data and in such scenarios if data for dimensions cannot be determined (due to real-time imperatives) an older version of the dimension must be used or a generic template could be used, otherwise. The dimension data should be posted into the hot (real time) partition as and when revised data is received or through the batch update scheduled for the end of day. Nevertheless, users should be informed that dimensions may contain transient data disparate from the fact data.

6.6.3.4 Preclude Data-Staging

Enterprise information integration (EII) systems stream data directly to the user's interface (screens) from the production sources, thereby precluding its transfer to ETL pipeline's permanent repository. Such circumstances, if extant in your enterprise, should be discussed with the senior management to appropriately assign responsibility for backup, recovery, archival, security, etc. of the uncommitted data.

6.6.3.5 Real-time partitions for Data Visualization

The demands for real-time data have increased with the decreasing time intervals of data transition between DW/Business Intelligence (BI) teams and production transaction processing. A real-time partition design as an extension of traditional data warehouse is a possible solution for resolving the increasing demands for data in real time. This is similar to the architectural construct illustrated in Figure-14 above. The real-time partition needs to conform to the following requisites:

- Contain complete set of activities since the last update of the static data warehouse
- Connect precisely as a true physical fact-table partition to the static DWs grain and content
- Allow continuous trickling of data through simple indexes. The ideal possibility of real-time partition being completely indexed may not be viable in case of relational databases where indexes are logically disparate to the partitioning structure.
- Mapping real-time partition in memory to support high performance queries even without extant indexes.

6.6.3.6 Real-time partition for Transactions

Considering that the granularity of a fact table within the static DW is at transactional level it will contain a single row per transaction. The dimensional structure of real-time partition would be similar to the SDW and may not have any indexes as it requires continuous data loading. The real-time partition would also lack time series as it is supposed to contain only day's data.

For a comparatively large enterprise in the retail business with approximately 10 million daily transactions, the fact table would be sizable in volume. Assuming that the width of each transaction grain is 44 bytes (eight dimensions, three facts and four byte columns) you would have 440 MB of data daily. That would mean, yearly, about 160 GB of raw data in the fact table with heavy indexing and aggregations. However, the daily slice of 440 MB of real-time data should be pinned in memory. The real-time partition may have affinity for fast-loading and demonstrate high speed query performance simultaneously.

6.6.3.7 Real-time partition based on periodic snapshots

If we consider the same example as above but with a monthly granularity then the real-time partition would constitute the current active rolling month. Considering a customer-base of 15 million customers, a 24 month time series would accumulate 360 million rows in its fact table that would again be massively indexed with supporting aggregations. The real-time partition is a snapshot of the currently progressing month that is continuously updated as the month advances. Amendment of fully summative facts and semi-summative balances would be done on availability of respective data. The fact table superset of customer types across the enterprise would be quite narrow with say four facts and five dimensions giving us a real-time partition of 600MB allowing the pinning of the real-time partition in memory. With the rest of the monthly data arriving on the last day of the month, the real-time partition could be merged as the latest month onto the comparatively stable fact table. Then the process could be iterated for the subsequent month by flushing the real-time partition.

☞ Check Your Progress 4

1) How is a real-time data warehouse different from a traditional data warehouse?

.....

.....

.....

.....

2) Why is CDW a good candidate for real-time data warehousing

.....

.....

.....

.....

6.7 DATA WAREHOUSING AND HADOOP

6.7.1 Introduction

The title of this section may sound confusing at first but as we explore the Hadoop system in detail, we should be able to gain more clarity as to why data warehousing and Hadoop necessitates a combined discussion. Although the data warehousing and Hadoop trend evolved sometime back and has become a mature topic today it is still a continuing trend and therefore, requires an overview to comprehend the concepts involved.

Before we dive into the specifics of the Hadoop system we need to go through a few preliminary concepts and understand them so that we can discuss this topic of data warehousing and Hadoop without any confusion or doubts. We have already studied Data warehouse in the previous chapters above. In this section we will understand the definition and features of Hadoop. We will also compare and contrast Hadoop with a Data warehouse system. We will see how Hadoop can supplement and support a DW. Finally, we will take a look at the advantages and challenges involved in Hadoop supported DW.

6.7.2 What is Hadoop?

Hadoop is an open-source java-oriented framework comprising two fundamental components: i) Storage and ii) Processing and was developed, especially, to deal with massive volumes of data in a distributed computing ecosystem. The storage component of Hadoop is called the Hadoop Distributed File System (HDFS) while the processing component is termed as MapReduce.

Essentially, Hadoop, today, comprises of the following modules:

1. The Hadoop Distributed File System

   The HDFS allows data storage with easy accessibility. The data storage resides in an environment guaranteeing *highly reliability* and *highly availability.*

2. MapReduce

   MapReduce is principally a combination of two processes. Mapping and Reduction. Data is read from the data sources and converted to a format conducive to data analysts. This is the mapping process. The data is then subjected to mathematical operations like aggregations, grouping, encoding

ranges, etc. also called as reductions and hence the terminology - *reduce*. In the Hadoop framework, it is the MapReduce programming model implementation for processing data on a large scale also termed as *data processing at scale*.

3. Hadoop Common

   Hadoop common is a collection of libraries and tools required by other Hadoop modules.

4. YARN

   YARN is a platform that orchestrates and manages clustered computing resources and uses them for scheduling user-applications.

5. Hadoop Ozone

   Ozone is an object store (a data storage architecture that manages data as objects, through a combination of a globally unique identifier, metadata and the data itself)

6.7.3 Hadoop Architecture

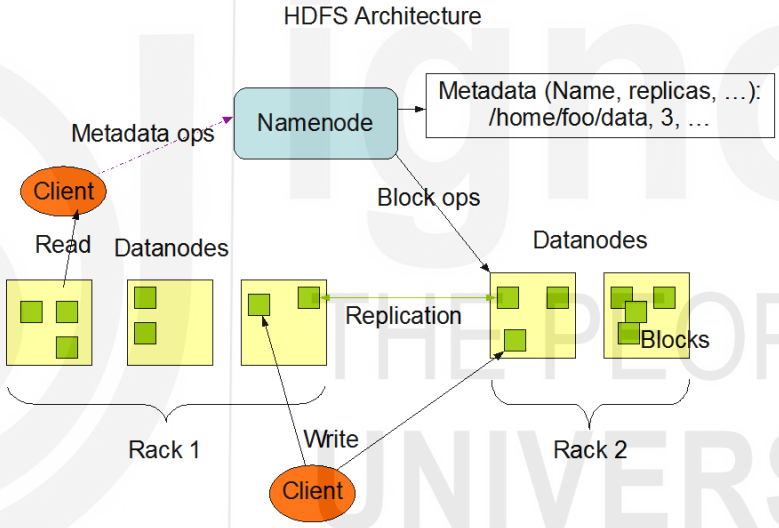

Figure-15. HDFS Architecture for Hadoop 3.3.0
(from Hadoop documentation available at hadoop.apache.org)

The HDFS architecture as shown in Figure-15 implements a master/slave model and has the following components:

NameNode

It is a single node and functions as the master server managing the file-system namespace and regulating client access to files. It handles operations like opening, closing and renaming of files/directories. It also determines the mapping of blocks to DataNodes. NameNode is the arbitrator and repository for all HDFS metadata and its sole existence greatly simplifies the system architecture. The system design is such that user data never flows through the NameNode.

DataNodes

There are multiple DataNodes, usually one per node in the cluster. HDFS allows user data to be stored in files. Files are, internally, split into one or more blocks which are stored in a set of DataNodes and they handle read and write requests from the clients.

Based on instruction from the NameNode, the DataNodes perform block creation, deletion and replication.

NameNodes and DataNodes are software designed to run on commodity machines. Any machine that supports Java can run the NameNode or DataNode as HDFS is built using the Java language. HDFS inherits all the advantages of Java based deployment like portability to a variety of machines.

The File System Namespace

HDFS supports hierarchical file organization and allows creation of directories with files within the directories. It supports file/directory creation, deletion, renaming or moving files/directories within the hierarchy. Additionally, it supports user quotas and access permissions including transparent encryption and snapshots.

Data Replication

HDFS reliably stores very large files across machines in a large cluster. Each file is stored as a sequence of blocks that are replicated for fault tolerance. With the exception of the last block, all blocks are uniform in size. An application can configure the number of replicas of a file. Replication factor is also configurable before and after file creation. Files in HDFS are write-once (except for appends and truncates) and strictly have one writer at any given time. The NameNode makes all decisions pertinent to block replication and periodically receives a Blockreport and a Heartbeat from every DataNode in the cluster.

Hadoop is often used to mean both base modules and sub-modules and also the environment. It may also refer to collection of supplementary software that can be integrated with or function as extensions of Hadoop framework. Some prevalent software are Apache Flume, Apache HBase, Apache Hive, Cloudera Impala, Apache Oozie, Apache Phoenix, Apache Pig, Apache Spark, Apache Sqoop, Apache Storm and Apache ZooKeeper.

6.7.4 Comparison of Hadoop and Data warehouse

Now that we had an overview of Hadoop let us find out its relation with a data warehouse by comparing the two. Data warehouse and Hadoop can be compared on the basis of various characteristics common to both as follows:

6.7.4.1Data

Data sources can be added or removed in both DWs and Hadoop in addition to allowing heterogeneous data sources. A DW allows consumption of structured data which is transformed to its schema format. Hadoop allows for consumption of both structured and semi-structured data.

Users of Raw data are typically absent in a DW scenario with analysts requiring preprocessed data. Hadoop enables the use of raw data by data engineers and data scientists enabling innovative knowledge discoveries for the enterprise. In case of a DW this advantage is lost during the data transformation process.

Hadoop works on the principle of schema-on-read while a DW works on the principle of schema-on-write. A schema-on-read allows adaptability to the heterogeneous nature of data and therefore, its accommodation for business utilization. Conversely fitting data to the dimensional structure in a DW imposes restrictions on which data can be accommodated.

Hadoop allows distributed data processing at large scale data by virtue of its file system capabilities and architecture supplemented by integrated or add-on toolsets which support the Hadoop ecosystem. A traditional data warehouse architecture would require substantial modifications to its architecture to support a similar set of tools in addition to changing its data repository.

6.7.4.2Performance

Hadoop tools like Spark, Impala, etc. allow for near-real-time implementations for data propagation across the system allowing data-visualizations on the most current datasets. Ingested data is brought to the analytics stage many times faster in Hadoop when compared to a DW.

6.7.4.3Governance

Governance in Hadoop is relatively easier due to its highly configurable architecture. Also Hadoop's architectural constructs of data storage, consumption of data in a specific manner (due to its schema-on-read feature) and availability of tools with individual system-wide functionality make governance across the ecosystem easy to manage.

6.7.4.4Security

Hadoop implements security at block level in addition to offering transparent encryption. Desensitizing data becomes easy due to massive storage and computing capacity of Hadoop. DWs cannot maintain desensitized datasets for longer periods. Also the granularity and methods of security differ vastly when compared to that provided by Hadoop

6.7.4.5Flexibility

Hadoop is intrinsically flexible in scaling its compute, storage, memory and network resources. This flexibility lends agility to the entire system and data becomes pervasive across the enterprise. Traditional DWs lack this capacity and would involve a lot of cost and efforts to incorporate such flexibility into their architectures.

6.7.5 Conceptual architecture of Hadoop Data Warehouse

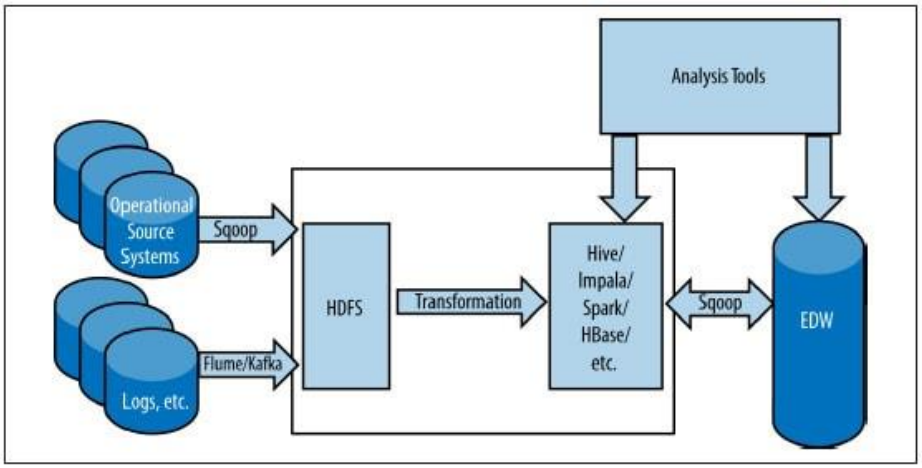

Figure-16. Conceptual Hadoop Data warehouse architecture
(courtesy: Vithal S, article titled "Hadoop Data warehouse Design Considerations" on dwgeek.com)

A Hadoop Data warehouse can be implemented using Apache Hive or Cloudera Impala. Impala offers better performance than Hive due to its massively parallel processing (MPP) capabilities.

The architecture in Figure16 shows a Hadoop supplemented and supported DW. In this scenario, the DW inherits benefits of the Hadoop system augmenting the ETL processes that are at the core of a DW.

On the left side of the figure data sources are shown which could include other heterogeneous sources like sensor data, logs and other structured and semi-structured data. The components of DW in the context of Hadoop are explained as follows:

6.7.5.1Data Extraction

Tools like Apache Sqoop can be utilized for data extraction from OLTP database sources. Sqoop can also be used to export data to OLTP databases after data processing is completed within the Hadoop system.

6.7.5.2Data Transformation

Data is ingested into the Hadoop system precluding data staging. This ingested data is transformed and loaded into target data repositories within assigned directories. Tools like Spark, Cloudera Impala, Apache Hive, Apache Pig or MapReduce (the integral processing component of Hadoop) can be used for data transformations.

6.7.5.3 Data Loading

In the context of Hadoop data loading implies data hosting within Hadoop through Apache Hive or Cloudera Impala. Data can be exported to external DWs for ancillary analysis

6.7.5.4 Data Analysis / Visualization

As mentioned above data can be exported for assisting analysis through external DWs or data can be sources directly from the Hadoop system by BI, data analytics and visualization tools. Apache Sqoop can be used for the exporting data assisted through Python or Shell scripts.

6.7.6 Advantages of building a Hadoop Data Warehouse

Hadoop alleviates traditional data warehouse design challenges as follows:

- Heterogeneous data processing at scale
- Improving ETL processing throughput
- Good candidate for near-real-time data warehousing
- Efficient workload management
- Hadoop offers architectural and data level flexibility through scalable and configurable components
- The TCO of the DW system can be reduced through elastic utilization and customizations thereby, increasing the ROI

6.7.7 Challenges of building a Hadoop Data Warehouse

The challenges of a Hadoop based Data warehouse are as follows:

- Permeating the benefits of patterns in raw data to the structured components of a DW
- Data propagation latencies may arise in certain cases where large amounts of data are made available disproportionate to the data being consumed
- Use of Hadoop as an archival repository for historical data may introduce costs for import/export of data if historical data is required.

6.8 DATA WAREHOUSE AUTOMATION

.

6.8.1 Introduction

Conventional data warehouses or for that matter even some implementations of cloud data warehouses perform repetitive tasks manually with the excuse of business dynamics and constantly changing reporting parameters. This has an adverse effect on the productivity, costs and overall quality expected from the data warehousing platform.

The latest trend is towards data warehouse automation (DWA) which deals with accelerating and automating the data warehouse life-cycle (from conceptualization to implementation). Some enterprises may technically perceive it as automation beginning from analysis of source systems right up to testing and documentation processes.

6.8.2 DWA Maturity

DWA involves the use of sophisticated tools and architectural models to automate planning, design, integration and implementation of DWs through their life-cycle reducing repetitive and time consuming tasks like source analysis, ETL scripts deployment, etc. The automation process requires careful thought for evaluating and selecting the right tools that can drastically improve savings on costs and time at a fraction of TCO of data warehouses.

The functionalities of automation software have matured with explosive growth of data and the race for data-driven insights to leverage businesses. Data explosion has led to the necessity for large-scale data storage and high performance integration platforms. Further, the diversity and velocity of data from various sources (like IoT, social media, etc.) require data processing at an equivalent scale.

6.8.2.1Maturity Stages of DWA

6.8.2.1.1 DW Architectures and Database management systems (DBMSs)

The earliest data warehouses came into existence with the advent of disk storage that required storing large volumes of data in the form of databases. These requirements stimulated the ideas for relational data and dimension based data marts. Subsequently, SQL enabled Relational DBMSs and proprietary ETL software became available.

6.8.2.1.2 Standardization of DW Architecture

DW Architectures were being formalized and normalized to include automation of source data analysis and components for ingestion and transformation of

heterogeneous data. This would enable the utilization of any data source without worrying about data formats and any specialized treatment for the same.

6.8.2.1.3 Challenges of Enterprise data warehouse and data automation imperative

The pace of changes in the industry posed myriad challenges for enterprises like inefficient metadata management, expensive development and protracted development cycles. Many systems did not integrate efficiently with the underlying components like database management systems, other repositories within the system, etc.

The current DWA software are aligned with business requirements and technological innovations

6.8.2.1.4 Data warehouse automation tools

A DWA tool provides a seamless experience, free from the hassles of coding, for integration and transition of diverse data from its source to a DW and other related components. The tool automates deployment of ETL scripts, batch-processing of data and demonstrates various functions like:

- Processes for high performance ETL and reliable ELT based integration of data
- Modeling of data at source
- Normalized, De-normalized and data structures with multiple dimensions
- Seamless integration with various data sources

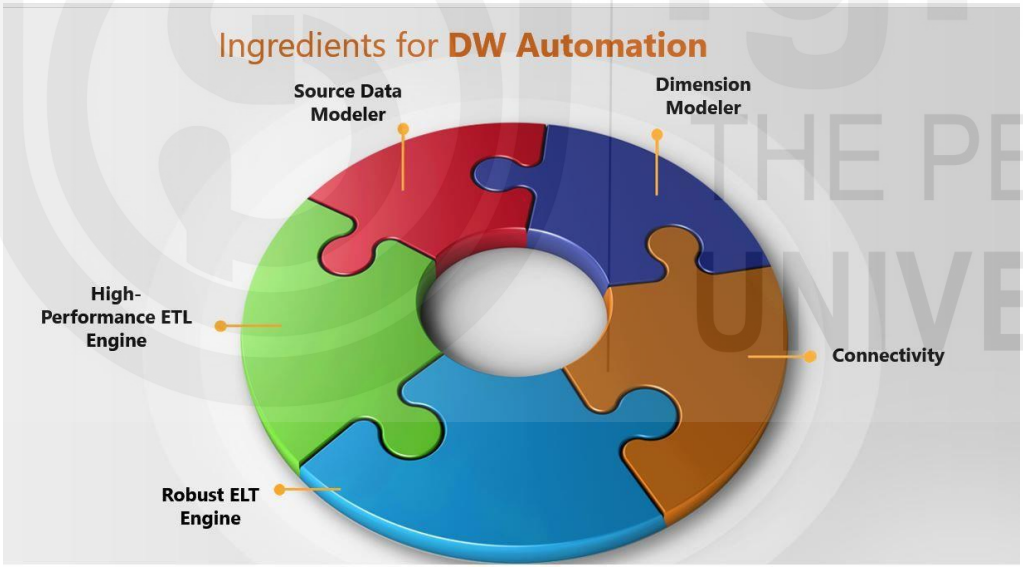

Figure-17. Essential Components of DWA tools

6.8.3 Assessment of DWA preparedness

Enterprises need to evaluate their level of preparedness based on the following needs:

6.8.3.1 DW architecture:

Assess whether your DW architecture is based on a distinctive set of components or whether it uses best-of-breed components and best practices. A precise understanding of the various components that you have implemented and their interactions is of

prime importance to understand the changes required and their impact on your enterprise in terms of cost and efforts.

6.8.3.2 Business requirements:

Assess how business requirements are handled within your organization and the methodologies practiced (waterfall, prototype, agile, etc.) for identifying and understanding them.

6.8.3.3 Policies and Procedures:

Evaluate the structure of organizational processes, compliance requirements, business continuity policies and plans (disaster and recovery management, security, governance etc.), and others. Assess whether the operations are resource and time intensive, vague/detailed, complex or fragile in nature.

6.8.3.4 Data ecosystem maintenance:

Scrutinize whether the maintenance of your data infrastructure is organized, complicated or heavily resource dependent

6.8.3.5 Process and Requirements Persistence:

Evaluate the frequency with which your enterprise changes its processes and requirements or whether it persists for a reasonable amount of time before carefully incorporating changes.

6.8.3.6 Quality and accessibility:

Assess how frequently data is accessed. Also, find out how the delivery of analytical reports is expected whether it is within reasonable time or on an immediate basis. This will let you understand the expectations about data quality, QoS and quality of deliverables.

6.8.4 Advantages of DW automation:

A few significant advantages of DW automation are as follows:

- End-to-end data pipeline acceleration
- Automation of data capture and streaming
- Automation of data management and integration
- Optimization of data propagation paths
- Ensure automatic data flows in their entirety
- Automatic set up of target-oriented data models
- Transformation of data lakes into DWs.

☞ Check Your Progress 5

1) How would DWA be advantageous in the case of an enterprise like Amazon?

.....

.....

.....

.....

6.9 SUMMARY

In this section we followed the trends in data warehousing. We noticed that the trends originated based on industrial implementations of research based solutions offered to challenges of previous generation data warehousing challenges. Therefore, we concluded that those challenges which we face presently and are resolved through research in that area will become trends of the future.

Within the topic of Data Warehousing Trends we had an overview of the following sub-topics:

- Data Lakes

  We saw how the concept of data lakes originated with data explosion, termed as Big Data now. We discussed various data lake architectures and learned the best practices for implementing a successful data lake so that it does not turn into a Data Swamp. We also discussed Self-service, a very important component and a design pattern which forms an integral part of any Big Data based implementation.

- Complex Data Marts

  In the sub-topic of complex data marts, we studied how complex data marts came into existence. We also discussed its design process compared to a conventional data mart.

- Cloud Data Warehousing

  CDW is a comparatively newer trend and with the advent of Big Data the paradigm of DW modeling has changed. We have all the advantages of the cloud to support a cloud hosted data warehouse. We had an overview of the challenges and best practices for implementing a successful CDW

- Real Time Data Warehousing

  Real-time data warehousing is a very popular and closely followed trend by major enterprises. In the race for data dominance and biggest market share enterprises want to be able to take decisions based on the most recent data. Instantaneous data accessibility and availability has become the need of the hour with enterprises investing a lot of money and resources to make it a reality. We reviewed a few major proposed architectures that have already been put into practice today. Finally, we discussed the architectural tradeoffs or comprises that are required for deriving the benefits of real-time data availability.

- Data Warehousing and Hadoop

  Hadoop was developed to counter the challenges of big data that comes in unstructured or semi-structured forms. The obvious performance and scalability benefits came to the attention of enterprises with large data warehouse implementation. These enterprises thought of utilizing Hadoops toolset to build Hadoop based data warehouses. We studied about the conceptual architectures for the same and discussed the advantages and challenges involved.

- Data Warehouse Automation

  Under this sub-topic we had an overview of how DWA has matured over the years and what benefits could be had from automating almost the entire component set of the data warehouse. Best practices were studied including the challenges involved and the benefits possible post DWA.

This unit has touched upon some very important and critical topics that are either already established or would become the mainstay of the future. Students are

requested to study these trends in detail by referring to texts and documentation mentioned below and through their own curious efforts.

6.10 SOLUTIONS/ANSWERS

Check your Progress 1

1. Many challenges have come to the fore-front with increasing volumes and speed of data. Data has increased due to various technological advances resulting in increase in the number of sources and diversity.

   Researchers have tried to address myriad challenges subsequent to data explosion and the significant ones being worked upon by maximum researchers are as below:

- Conceptual modeling of DWs and logical data models,
- Data warehouse loading (data-refreshing),
- execution efficacy of OLAP queries and data mining algorithms,
- materialized views,
- data analysis techniques,
- metadata management,
- evolution management of DWs,
- stream-based, real-time and active data warehouses, and
- complex data warehousing (for example spatial, XML, object, multimedia)

2. The concept of logical data lakes originated through the implementation of multiple data lakes in a distributed environment. Logical data lakes form an abstraction layer over the implementation of a data lake or data lakes and allow their use without exposing the complexities of location, data sources, target systems, etc. and allow enterprises to perceive data access as an operation on a single entity that spans across multiple locations and brings data from various repositories together. Draw the architectural representation and explain it in your own words. Elaborate on the components and advantages of the concept.

3. A successful data lake is well cataloged both before and after data ingestion so that data traceability to its source is maintained. Data is well organized and is available through self-service for people across the enterprise. People are aware of data contained within a data lake and utilize it based on their specific needs. Other standards or best practices are adhered to all times like data governance, data quality maintenance, metadata generation, master data management, etc.

Check your progress 2

1. Complex data is data that has no definitive structure or form for example sensor data. Complex data also has high velocity and volumes and does not conform to any standard rules of a schema, technology or function. It is also an attribute of Big data. Complex data forms a major portion of the raw data that is ingested into a data lake and is considered to be of significance for data insights by data engineers and data scientists.

   As the race for leveraging businesses based on data-driven insights intensifies, enterprises want to derive the maximum business value hidden within the complex data. In order to do this complex data needs to be processed and converted to usable form. As conventional models are not adequately equipped to

handle complex data that may be unstructured or semi-structured new models are required to efficiently process complex data and make it useful for data analysis.

2. Please see the information related to Data Vault on *wikipedia.org* and frame your answer accordingly. Draw the example illustration and briefly explain the various components of the Data Vault model.

Check your progress 3

1. Enterprises migrate to the cloud due to the following benefits offered by a CDW implementation:

    a. Improved performance

    The availability of cloud-based database management software and tools supplemented by sophisticated infrastructure allow performance that is many times faster than on-premises DWs. The ability to easily scale resources based on processing requirements and latencies are drastically reduced compared to alternative implementations of the DW.

    b. Flexibility of cost through flexibility of cloud resources utilization

    Cloud implementations allow elastic use of resources on a pay-as-you-use basis which lends flexibility in terms of costs and resource utilization. Conversely, in on-premises based implementations the stakeholders are stuck with the resources and require additional costs to scale their infrastructure resources.

    c. Migration of existing products to the cloud environment

    Most enterprises may have previously migrated their ERP systems or other software to the cloud or may be thinking of implementing other projects on the cloud. This makes it all the more imperative to have a CDW to reduce data propagation latencies and integration with the systems already on the cloud.

    d. Derive benefits that are inherent to a cloud platform

    A cloud platform offers many benefits based on the myriad services it offers like Software as a service (SaaS), Platform as a service (PaaS), Infrastructure as a service (IaaS), etc. These services can be exploited to derive maximum benefits. For example by exploiting SaaS we can have best of breed software (like ETL tools, database management software, etc.) to support and supplement our Data warehouse implementation.

Check your progress 4

1. Compare the real-time and traditional DWs based on their design, function, performance, data quality and data visualization expectations

    For example you could elaborate on the below:
    Design components for a traditional DW involve batch-based ETL, data quality, data source scheduling, data completeness is guaranteed through the data processing and data integration components.

For a Real-time DW ETL is not batch-based and uses various other approaches to load new data and update existing data, data quality needs to be compromised as data is consumed immediately as it arrives and waiting for it to be complete in all respects (like foreign key constraints, missing values, etc.) is not permitted. Data processing practices need to be modified to process partial information and later ensure completeness through incremental updates, data integration may require partitioning to hold incomplete information. Data staging needs to be abandoned to eliminate time lost in scrutinizing data.

2. CDW offers various benefits like performance, resource scalability, etc. which make it easy to implement the design changes required for Real-time DW. CDW implementation supports sophisticated tools for data storage, processing and manipulation. CDW also makes data integration easy through its toolset allowing for better data quality and data propagation across the system

Check your progress 5

1. End-to-end data pipeline acceleration would increase the flow of data through its system modules (Inventory, order processing, bill-generation, delivery tracking and order fulfillment and customer feedback

Automation of data capture and streaming would benefit Amazon to collect significant amount of data at various stages from when a customer buys a product to order fulfillment and feedback

Automation of data management and integration would aid Amazon to avoid data duplication, reduce the risks and challenges of data import / export to from other systems, generate data value for business and increase QoS.

Optimization of data propagation paths will ensure data availability with minimum effort and maximum benefit

Amazon can ensure automatic data flows in their entirety i.e. ensure that all the details required for a transaction are completely captured and the data is automatically delivered to the systems requesting it for analysis.

Automatic set up of target-oriented data models would enable Amazon to ensure that data reaches its intended audience through the automatic selection of data source, data processing and integration tools and finally the analytics systems.

As Amazon operates on a very large scale spanning various continents the amount of data generated would be humongous and contained in data lakes. Transformation of data lakes into DWs would enable Amazon to ensure regulatory compliances. It would also allow data localization in the case of countries requiring it for political and other reasons. Through all this the DWs can still be integrated and enjoy the benefits of a data lake ecosystem.

6.11 FURTHER READINGS

1. Robert Wrembel, Alberto Abelló and Il-Yeol Song, *DOLAP data warehouse research over two decades: Trends and challenges*. 2019, Information Systems, pp. 44-47, Elsevier (*https://doi.org/10.1016/j.is.2019.06.004*)

2. Bill Inmon, *'Data Lake Architecture: Designing the Data Lake and Avoiding the Garbage Dump'*, 2016, ISBN- 9781634621205, Technics Publications (https://www.technicspub.com/)
3. Alex Gorelik, *'The Enterprise Big Data Lake: Delivering the Promise of Big Data and Data Science'*, 2019, ISBN- 9781491931554, O'Reilly Media Inc. (http://oreilly.com/catalog/errata.csp?isbn=9781491931554)
4. Vassiliadis, P., Simitsis, A.: Near Real Time ETL, pp. 1-31. Springer US, Boston, MA (2009), http://dx.doi.org/10.1007/978-0-387-87431-9_2
5. Obali, M., Dursun, B., Erdem, Z., Grr, A .K.: *A real time data warehouse approach for data processing. In: Signal Processing and Communications Applications Conference* SIU, 2013 21st. pp. 1-4 (2013)
6. Kimball, Ralph, Ross Margy.: *'The Data Warehouse Toolkit: The Definitive Guide to Dimensional Modeling'*, Third Edition, 2013, John Wiley & Sons Inc.