# (Big) Data Engineering In Depth

## From Beginner to Professional

Mostafa Alaa Mohamed
Senior Big Data Engineer
MoustafaAlaa Moustafa Alaa @Moustafa_alaa22
mustafa.alaa.mohamed@gmail.com

[1]Big Data & Analytics Department, Epam Systems

The Definitive Guide to Big Data Engineering Tasks

# Table of Contents I

# Table of Contents II

# Table of Contents III

# Table of Contents IV

# Table of Contents V

# Table of Contents VI

# Table of Contents VII

# Table of Contents VIII

# Table of Contents IX

# Introduction To Data Management and Data Warehouse

# Chapter Objectives

- Be familiar with data management life-cycle.

# Chapter Objectives

- Be familiar with data management life-cycle.
- Understand the data abstraction and the data layer.

## Chapter Objectives

- Be familiar with data management life-cycle.
- Understand the data abstraction and the data layer.
- Motivation to DWH.

# Chapter Objectives

- Be familiar with data management life-cycle.
- Understand the data abstraction and the data layer.
- Motivation to DWH.
- What is the different types of DWH?

# Chapter Objectives

- Be familiar with data management life-cycle.
- Understand the data abstraction and the data layer.
- Motivation to DWH.
- What is the different types of DWH?
- Use cases for DWH and how it differ from the operational db.

# Chapter Objectives

- Be familiar with data management life-cycle.
- Understand the data abstraction and the data layer.
- Motivation to DWH.
- What is the different types of DWH?
- Use cases for DWH and how it differ from the operational db.
- Explain the data Encoding and Formats.

# Chapter Objectives

- Be familiar with data management life-cycle.
- Understand the data abstraction and the data layer.
- Motivation to DWH.
- What is the different types of DWH?
- Use cases for DWH and how it differ from the operational db.
- Explain the data Encoding and Formats.
- Show what is the challenges to build a DWH?

# Chapter Objectives

- Be familiar with data management life-cycle.
- Understand the data abstraction and the data layer.
- Motivation to DWH.
- What is the different types of DWH?
- Use cases for DWH and how it differ from the operational db.
- Explain the data Encoding and Formats.
- Show what is the challenges to build a DWH?
- What is the data modeling and its design?

# Data Management

# Data Management

- Data are a product.

# Data Management

- Data are a product.
- Data product has a life-cycle as following (simplified):

# Data Management

- Data are a product.
- Data product has a life-cycle as following (simplified):
  - **Question**, Idea, or service.

# Data Management

- Data are a product.
- Data product has a life-cycle as following (simplified):
    - **Question**, Idea, or service.
    - **Identifying** the source of information and the data type ex: (text, images, videos, audio, or sensors).

# Data Management

- Data are a product.
- Data product has a life-cycle as following (simplified):
    - **Question**, Idea, or service.
    - **Identifying** the source of information and the data type ex: (text, images, videos, audio, or sensors).
    - **Document** all details regarding the data including quality, security, efficiency, and access (consideration during the cycle).

# Data Management

- Data are a product.
- Data product has a life-cycle as following (simplified):
  - **Question**, Idea, or service.
  - **Identifying** the source of information and the data type ex: (text, images, videos, audio, or sensors).
  - **Document** all details regarding the data including quality, security, efficiency, and access (consideration during the cycle).
  - Delivery automation (Tools and Process) AKA **DevOps** cycle.

# Data Management

- Data are a product.
- Data product has a life-cycle as following (simplified):
  - **Question**, Idea, or service.
  - **Identifying** the source of information and the data type ex: (text, images, videos, audio, or sensors).
  - **Document** all details regarding the data including quality, security, efficiency, and access (consideration during the cycle).
  - Delivery automation (Tools and Process) AKA **DevOps** cycle.
  - **Extraction** Process (collection).

- Data are a product.
- Data product has a life-cycle as following (simplified):
  - **Question**, Idea, or service.
  - **Identifying** the source of information and the data type ex: (text, images, videos, audio, or sensors).
  - **Document** all details regarding the data including quality, security, efficiency, and access (consideration during the cycle).
  - Delivery automation (Tools and Process) AKA **DevOps** cycle.
  - **Extraction** Process (collection).
  - **Transformation** ex: (cleansing, Apply business logic, Organize).

# Data Management

- Data are a product.
- Data product has a life-cycle as following (simplified):
  - **Question**, Idea, or service.
  - **Identifying** the source of information and the data type ex: (text, images, videos, audio, or sensors).
  - **Document** all details regarding the data including quality, security, efficiency, and access (consideration during the cycle).
  - Delivery automation (Tools and Process) AKA **DevOps** cycle.
  - **Extraction** Process (collection).
  - **Transformation** ex: (cleansing, Apply business logic, Organize).
  - **Loading** or store the transformed data based on our usage or use case.

- Data are a product.
- Data product has a life-cycle as following (simplified):
  - **Question**, Idea, or service.
  - **Identifying** the source of information and the data type ex: (text, images, videos, audio, or sensors).
  - **Document** all details regarding the data including quality, security, efficiency, and access (consideration during the cycle).
  - Delivery automation (Tools and Process) AKA **DevOps** cycle.
  - **Extraction** Process (collection).
  - **Transformation** ex: (cleansing, Apply business logic, Organize).
  - **Loading** or store the transformed data based on our usage or use case.
  - Business Intelligence (**BI**) or data discovery (continues process).

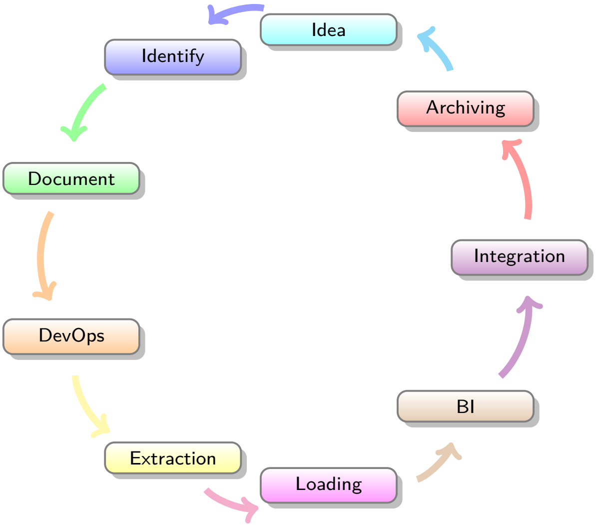
# Data Management

- Data are a product.
- Data product has a life-cycle as following (simplified):
  - **Question**, Idea, or service.
  - **Identifying** the source of information and the data type ex: (text, images, videos, audio, or sensors).
  - **Document** all details regarding the data including quality, security, efficiency, and access (consideration during the cycle).
  - Delivery automation (Tools and Process) AKA **DevOps** cycle.
  - **Extraction** Process (collection).
  - **Transformation** ex: (cleansing, Apply business logic, Organize).
  - **Loading** or store the transformed data based on our usage or use case.
  - Business Intelligence (**BI**) or data discovery (continues process).
  - **Integration** and publishing.

# Data Management

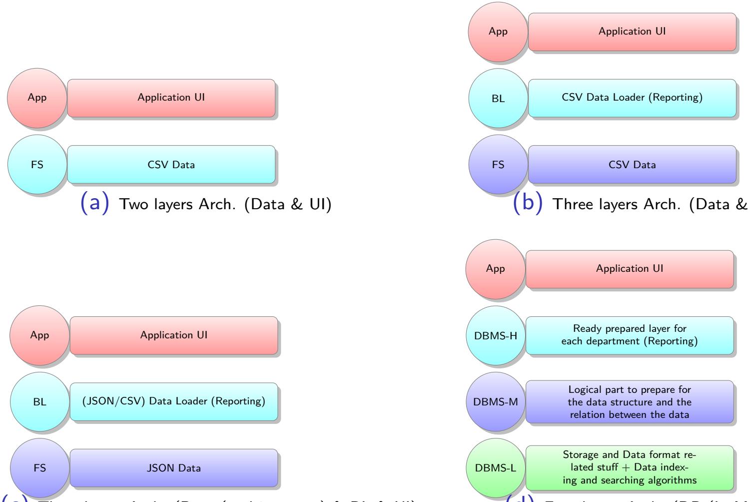
- Data are a product.
- Data product has a life-cycle as following (simplified):
  - **Question**, Idea, or service.
  - **Identifying** the source of information and the data type ex: (text, images, videos, audio, or sensors).
  - **Document** all details regarding the data including quality, security, efficiency, and access (consideration during the cycle).
  - Delivery automation (Tools and Process) AKA **DevOps** cycle.
  - **Extraction** Process (collection).
  - **Transformation** ex: (cleansing, Apply business logic, Organize).
  - **Loading** or store the transformed data based on our usage or use case.
  - Business Intelligence (**BI**) or data discovery (continues process).
  - **Integration** and publishing.
  - Data retention or **archiving** process ex: (Hot or Cold storage).

# Data Management Life-Cycle



# Data Abstraction

# Motivation to Data Layers (Use Case)




(a) Two layers Arch. (Data & UI)

(b) Three layers Arch. (Data & BL & UI)

(c) Three layers Arch. (Data (multi-sources) & BL & UI)

(d) Four layers Arch. (DB (L, M, H) & UI)

Figure: Data Abstraction Journey

# Motivation to Data Layers (Solution Thinking)

- How can we think about a data solution or challenges in the data products?

# Motivation to Data Layers (Solution Thinking)

- How can we think about a data solution or challenges in the data products?
  - Requirements analysis.

# Motivation to Data Layers (Solution Thinking)

- How can we think about a data solution or challenges in the data products?
  - Requirements analysis.
  - Identify the problem (challenges).

# Motivation to Data Layers (Solution Thinking)

- How can we think about a data solution or challenges in the data products?
  - Requirements analysis.
  - Identify the problem (challenges).
  - Think about how to overcome the challenges.

# Motivation to Data Layers (Solution Thinking)

- How can we think about a data solution or challenges in the data products?
  - Requirements analysis.
  - Identify the problem (challenges).
  - Think about how to overcome the challenges.
  - Ask your self the following questions:

# Motivation to Data Layers (Solution Thinking)

- How can we think about a data solution or challenges in the data products?
  - Requirements analysis.
  - Identify the problem (challenges).
  - Think about how to overcome the challenges.
  - Ask your self the following questions:
    - Can we solve the problem using the current data structure with adding new features?

# Motivation to Data Layers (Solution Thinking)

- How can we think about a data solution or challenges in the data products?
  - Requirements analysis.
  - Identify the problem (challenges).
  - Think about how to overcome the challenges.
  - Ask your self the following questions:
    - Can we solve the problem using the current data structure with adding new features?
    - What if we enhance/change the data structure or modeling?

# Motivation to Data Layers (Solution Thinking)

- How can we think about a data solution or challenges in the data products?
  - Requirements analysis.
  - Identify the problem (challenges).
  - Think about how to overcome the challenges.
  - Ask your self the following questions:
    - Can we solve the problem using the current data structure with adding new features?
    - What if we enhance/change the data structure or modeling?
    - Could it help if we change the backend engine (ex: DBMS system)?

# Motivation to Data Layers (Solution Thinking)

- How can we think about a data solution or challenges in the data products?
  - Requirements analysis.
  - Identify the problem (challenges).
  - Think about how to overcome the challenges.
  - Ask your self the following questions:
    - Can we solve the problem using the current data structure with adding new features?
    - What if we enhance/change the data structure or modeling?
    - Could it help if we change the backend engine (ex: DBMS system)?
- To answer these questions you need to understand the **data layers**.

# Data Layers (Abstraction)

- Any data product (database) contains multi-layers.

# Data Layers (Abstraction)

- Any data product (database) contains multi-layers.
- Each layer responsible for different tasks and operations.

# Data Layers (Abstraction)

- Any data product (database) contains multi-layers.
- Each layer responsible for different tasks and operations.
- Each layers interacts with (hardware or software or mixed).

# Data Layers (Abstraction)

- Any data product (database) contains multi-layers.
- Each layer responsible for different tasks and operations.
- Each layers interacts with (hardware or software or mixed).
- To eliminate the complexity for data interactions not all internal processes are shared or available for the user.

# Data Layers (Abstraction)

- Any data product (database) contains multi-layers.
- Each layer responsible for different tasks and operations.
- Each layers interacts with (hardware or software or mixed).
- To eliminate the complexity for data interactions not all internal processes are shared or available for the user.
- The developer for each layer hide internal irrelevant details from developer (users).

# Data Layers (Abstraction)

- Any data product (database) contains multi-layers.
- Each layer responsible for different tasks and operations.
- Each layers interacts with (hardware or software or mixed).
- To eliminate the complexity for data interactions not all internal processes are shared or available for the user.
- The developer for each layer hide internal irrelevant details from developer (users).
- The process of **hiding** irrelevant details from developer (user) is called data **abstraction**.

# Data Layers (Abstraction)

## Definition

**Data Abstraction and Data Independence**: DBMS comprise of complex data-structures. In order to make the system efficient in terms of retrieval of data, and reduce complexity in terms of usability of users, developers use abstraction i.e. hide irrelevant details from the users. This approach simplifies database design.

- There are 3 levels of data abstraction.

## Definition

**Data Abstraction and Data Independence**: DBMS comprise of complex data-structures. In order to make the system efficient in terms of retrieval of data, and reduce complexity in terms of usability of users, developers use abstraction i.e. hide irrelevant details from the users. This approach simplifies database design.

- There are 3 levels of data abstraction.
  - Physical Level

# Data Layers (Abstraction)

## Definition

**Data Abstraction and Data Independence**: DBMS comprise of complex data-structures. In order to make the system efficient in terms of retrieval of data, and reduce complexity in terms of usability of users, developers use abstraction i.e. hide irrelevant details from the users. This approach simplifies database design.

- There are 3 levels of data abstraction.
  - Physical Level
  - Logical/ Conceptual Level.

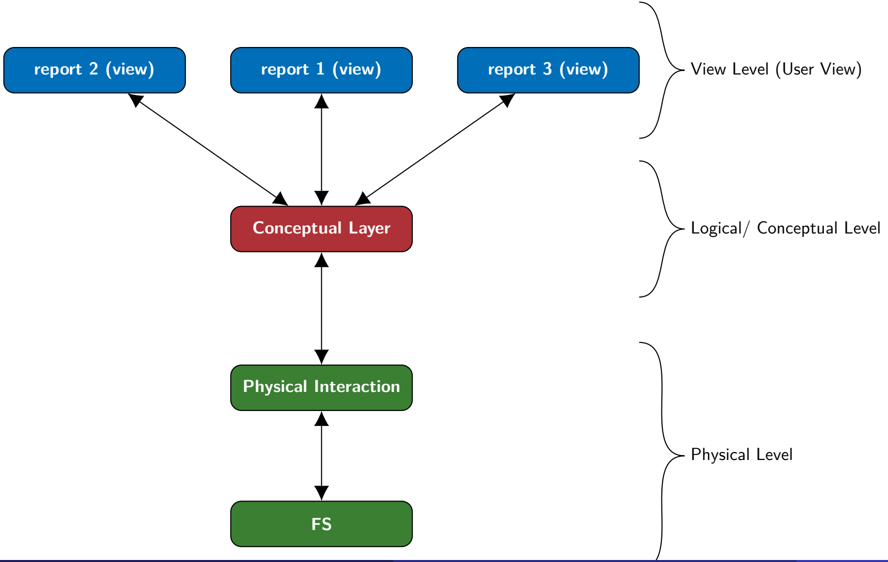
# Data Layers (Abstraction)

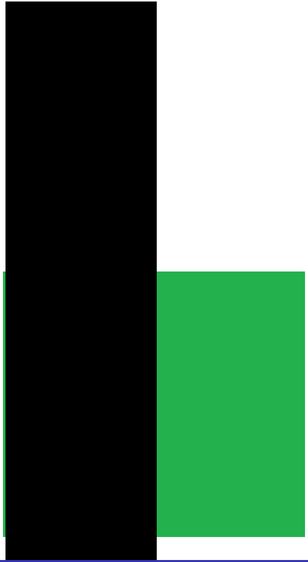
## Definition

**Data Abstraction and Data Independence**: DBMS comprise of complex data-structures. In order to make the system efficient in terms of retrieval of data, and reduce complexity in terms of usability of users, developers use abstraction i.e. hide irrelevant details from the users. This approach simplifies database design.

- There are 3 levels of data abstraction.
    - Physical Level
    - Logical/ Conceptual Level.
    - View Level.

# Data Layers (Abstraction)



- **Physical level (Internal)**:

# Physical level

- **Physical level (Internal)**:
  - Lowest level.

# Physical level

- **Physical level (Internal)**:
  - Lowest level.
  - Describes **how** data is stored.

- **Physical level (Internal)**:
  - Lowest level.
  - Describes **how** data is stored.
  - Describes the data structure.

# Physical level

- **Physical level (Internal)**:
  - Lowest level.
  - Describes **how** data is stored.
  - Describes the data structure.
  - It allows you to modify the lowest level (Physical part) without any change in the logical schema. These change could be

- **Physical level (Internal)**:
  - Lowest level.
  - Describes **how** data is stored.
  - Describes the data structure.
  - It allows you to modify the lowest level (Physical part) without any change in the logical schema. These change could be
    - Using a new storage device.

# Physical level

- **Physical level (Internal)**:
  - Lowest level.
  - Describes **how** data is stored.
  - Describes the data structure.
  - It allows you to modify the lowest level (Physical part) without any change in the logical schema. These change could be
    - Using a new storage device.
    - Change the structure of the data used for storage.

# Physical level

- **Physical level (Internal)**:
  - Lowest level.
  - Describes **how** data is stored.
  - Describes the data structure.
  - It allows you to modify the lowest level (Physical part) without any change in the logical schema. These change could be
    - Using a new storage device.
    - Change the structure of the data used for storage.
    - Change the files type or used different storage strcuture.

# Physical level

- **Physical level (Internal)**:
  - Lowest level.
  - Describes **how** data is stored.
  - Describes the data structure.
  - It allows you to modify the lowest level (Physical part) without any change in the logical schema. These change could be
    - Using a new storage device.
    - Change the structure of the data used for storage.
    - Change the files type or used different storage strcuture.
    - Changing the access method.

# Physical level

- **Physical level (Internal)**:
  - Lowest level.
  - Describes **how** data is stored.
  - Describes the data structure.
  - It allows you to modify the lowest level (Physical part) without any change in the logical schema. These change could be
    - Using a new storage device.
    - Change the structure of the data used for storage.
    - Change the files type or used different storage strcuture.
    - Changing the access method.
    - Modifying indexes.

- **Physical level (Internal)**:
    - Lowest level.
    - Describes **how** data is stored.
    - Describes the data structure.
    - It allows you to modify the lowest level (Physical part) without any change in the logical schema. These change could be
        - Using a new storage device.
        - Change the structure of the data used for storage.
        - Change the files type or used different storage strcuture.
        - Changing the access method.
        - Modifying indexes.
        - Change the compression algorithm or hashing technique.

## Example

- Database contains product information.

# Physical level

## Example

- Database contains product information.
- Physical layer describes

# Physical level

## Example

- Database contains product information.
- Physical layer describes
  - Storage mechanism and the blocks (bytes, gigabytes, terabytes etc.).

# Physical level

## Example

- Database contains product information.
- Physical layer describes
  - Storage mechanism and the blocks (bytes, gigabytes, terabytes etc.).
  - The amount of memory used.

# Physical level

## Example

- Database contains product information.
- Physical layer describes
  - Storage mechanism and the blocks (bytes, gigabytes, terabytes etc.).
  - The amount of memory used.
  - Usually this layer abstracted from the programmers.

# Logical level

- **Logical level (Conceptual)**:

# Logical level

- **Logical level (Conceptual)**:
  - Intermediate level.

# Logical level

- **Logical level (Conceptual)**:
  - Intermediate level.
  - Describes **what** data is stored.

# Logical level

- **Logical level (Conceptual)**:
  - Intermediate level.
  - Describes **what** data is stored.
  - Describes what is the relationship between the stored data.

# Logical level

- **Logical level (Conceptual)**:
  - Intermediate level.
  - Describes **what** data is stored.
  - Describes what is the relationship between the stored data.
  - It allows you to change the logical view without altering the external view, API, or programs. These change could be

# Logical level

- **Logical level (Conceptual)**:
  - Intermediate level.
  - Describes **what** data is stored.
  - Describes what is the relationship between the stored data.
  - It allows you to change the logical view without altering the external view, API, or programs. These change could be
    - Add new table.

# Logical level

- **Logical level (Conceptual)**:
  - Intermediate level.
  - Describes **what** data is stored.
  - Describes what is the relationship between the stored data.
  - It allows you to change the logical view without altering the external view, API, or programs. These change could be
    - Add new table.
    - Change the records merge or delete without affecting the running applications.

- **Logical level (Conceptual)**:
  - Intermediate level.
  - Describes **what** data is stored.
  - Describes what is the relationship between the stored data.
  - It allows you to change the logical view without altering the external view, API, or programs. These change could be
    - Add new table.
    - Change the records merge or delete without affecting the running applications.
    - Change attribute (Add,delete) to existing table.

# Logical level

## Example

- Database contains product information.

# Logical level

## Example

- Database contains product information.
- Logical Layer describes

# Logical level

## Example

- Database contains product information.
- Logical Layer describes
  - The product fields and their data types.

## Example

- Database contains product information.
- Logical Layer describes
  - The product fields and their data types.
  - How this product interact with other entities in the database.

# Logical level

## Example

- Database contains product information.
- Logical Layer describes
  - The product fields and their data types.
  - How this product interact with other entities in the database.
  - The programmers design this level based on the business knowledge and the requirements.

# View level

- **View level (External)**:

# View level

- **View level (External)**:
  - Highest level.

- **View level (External)**:
  - Highest level.
  - **View** of the data stored?

# View level

- **View level (External)**:
  - Highest level.
  - **View** of the data stored?
  - Designed for category of users needs.

- **View level (External)**:
  - Highest level.
  - **View** of the data stored?
  - Designed for category of users needs.
  - It is the final interface for the user.

# View level

- **View level (External)**:
  - Highest level.
  - **[View]** of the data stored?
  - Designed for category of users needs.
  - It is the final interface for the user.
  - It could be extended or hidden based on user's role.

- **View level (External)**:
  - Highest level.
  - **View** of the data stored?
  - Designed for category of users needs.
  - It is the final interface for the user.
  - It could be extended or hidden based on user's role.
  - Not all the views is extended to all users and there is an authentication based on the category.

# View level

## Example

- Database contains product information.

# View level

## Example

- Database contains product information.
- It could be designed to show the sales of product in specific region.

# View level

## Example

- Database contains product information.
- It could be designed to show the sales of product in specific region.
- We might hide information about some products based on the teams or users.

Let's answer our previous the question, How can we solve data challenges?

# Data solution thinking (Summary)

- Let's split the problem based on the data layers.

# Data solution thinking (Summary)

- Let's split the problem based on the data layers.
  - View layer

# Data solution thinking (Summary)

- Let's split the problem based on the data layers.
  - View layer
    - When we need to add/remove/create new reports it is usually view layer.

# Data solution thinking (Summary)

- Let's split the problem based on the data layers.
  - View layer
    - When we need to add/remove/create new reports it is usually view layer.
    - We don't need to change the logical or physical layer to support the view layer.

# Data solution thinking (Summary)

- Let's split the problem based on the data layers.

# Data solution thinking (Summary)

- Let's split the problem based on the data layers.
  - Logical Layer

# Data solution thinking (Summary)

- Let's split the problem based on the data layers.
  - Logical Layer
    - When you have missing sources into your logical layer and you need to add this source and its structure.

# Data solution thinking (Summary)

- Let's split the problem based on the data layers.
  - Logical Layer
    - When you have missing sources into your logical layer and you need to add this source and its structure.
    - There is a performance issue in the existing reports and you need to change in the model. For example, reduce the join by creating new join table (*materialized view*).

# Data solution thinking (Summary)

- Let's split the problem based on the data layers.
  - Logical Layer
    - When you have missing sources into your logical layer and you need to add this source and its structure.
    - There is a performance issue in the existing reports and you need to change in the model. For example, reduce the join by creating new join table (*materialized view*).
    - Update the data type or the existing relation which could help to fix some data or performance issues.

# Data solution thinking (Summary)

- Let's split the problem based on the data layers.

# Data solution thinking (Summary)

- Let's split the problem based on the data layers.
    - Physical Layer

# Data solution thinking (Summary)

- Let's split the problem based on the data layers.
  - Physical Layer
    - When our problem is hardly or impossible to be fix by obtimize the query (view)/ logical layer it is time for physical change.

# Data solution thinking (Summary)

- Let's split the problem based on the data layers.
  - Physical Layer
    - When our problem is hardly or impossible to be fix by obtimize the query (view)/ logical layer it is time for physical change.
    - If we need to change your storage/compression/structure/access technique.

# Data solution thinking (Summary)

- Let's split the problem based on the data layers.
  - Physical Layer
    - When our problem is hardly or impossible to be fix by obtimize the query (view)/ logical layer it is time for physical change.
    - If we need to change your storage/compression/structure/access technique.
    - If we need to change the data orientation structure from row to column or key-value storage, It is time to change the physical layer.

# Introduction to DWH

# Motivation to Data Warehouse (DWH)

# Motivation to Data Warehouse (DWH)

- Data could be a product for some companies.

# Motivation to Data Warehouse (DWH)

- Data could be a product for some companies.
- It could be decision support for other products or businesses.

# Motivation to Data Warehouse (DWH)

- Data could be a product for some companies.
- It could be decision support for other products or businesses.
- Reporting the results after passing the data life-cycle will be from storage (Database).

# Motivation to Data Warehouse (DWH)

- Data could be a product for some companies.
- It could be decision support for other products or businesses.
- Reporting the results after passing the data life-cycle will be from storage (Database).
- There are some challenges facing the people who work on data management backend:

# Motivation to Data Warehouse (DWH)

- Data could be a product for some companies.
- It could be decision support for other products or businesses.
- Reporting the results after passing the data life-cycle will be from storage (Database).
- There are some challenges facing the people who work on data management backend:
  - Performance.

- Data could be a product for some companies.
- It could be decision support for other products or businesses.
- Reporting the results after passing the data life-cycle will be from storage (Database).
- There are some challenges facing the people who work on data management backend:
  - Performance.
  - Integration.

# Motivation to Data Warehouse (DWH)

- Data could be a product for some companies.
- It could be decision support for other products or businesses.
- Reporting the results after passing the data life-cycle will be from storage (Database).
- There are some challenges facing the people who work on data management backend:
  - Performance.
  - Integration.
  - Applying analytical functions.

# Motivation to Data Warehouse (DWH)

- Data could be a product for some companies.
- It could be decision support for other products or businesses.
- Reporting the results after passing the data life-cycle will be from storage (Database).
- There are some challenges facing the people who work on data management backend:
  - Performance.
  - Integration.
  - Applying analytical functions.
- Vendors who are working to solve the above challenges creating their own product of DWH and their ultimate work is to optimize the above points.

# Motivation to Data Warehouse (DWH)

**Definition (What is Data Warehousing?)**

A DWH is defined as a technique for collecting and managing data from varied sources to **provide meaningful business insights**. It is a blend of technologies and components which aids the strategic use of data.

The real concept was given by Inmon Bill. He was considered as a father of the DWH. He had written about a variety of topics for building, usage, and maintenance of the warehouse & the Corporate Information Factory

# Motivation to Data Warehouse (DWH)

- The DWH is not a product but an environment.

# Motivation to Data Warehouse (DWH)

- The DWH is not a product but an environment.
- It is a process of transforming data into information and make it available to users in a **timely manner** to make a difference.

# Motivation to Data Warehouse (DWH)

- The DWH is not a product but an environment.
- It is a process of transforming data into information and make it available to users in a **timely manner** to make a difference.
- It is an architectural construct of an information system which provides users with current and historical decision support information which is difficult to access or present in the traditional operational data store.

# Motivation to Data Warehouse (DWH)

- The DWH is not a product but an environment.
- It is a process of transforming data into information and make it available to users in a **timely manner** to make a difference.
- It is an architectural construct of an information system which provides users with current and historical decision support information which is difficult to access or present in the traditional operational data store.
- The DWH is the core of the BI system which is built for data analysis and reporting.

# Motivation to Data Warehouse

Data warehouse system is also known by the following names:

- Decision Support System (DSS).
- Business Intelligence Solution.
- Executive Information System.
- Management Information System.
- Analytic Application.
- Data Warehouse.

# Differences Between DWH and Operational DB

# DWH vs Operational databases

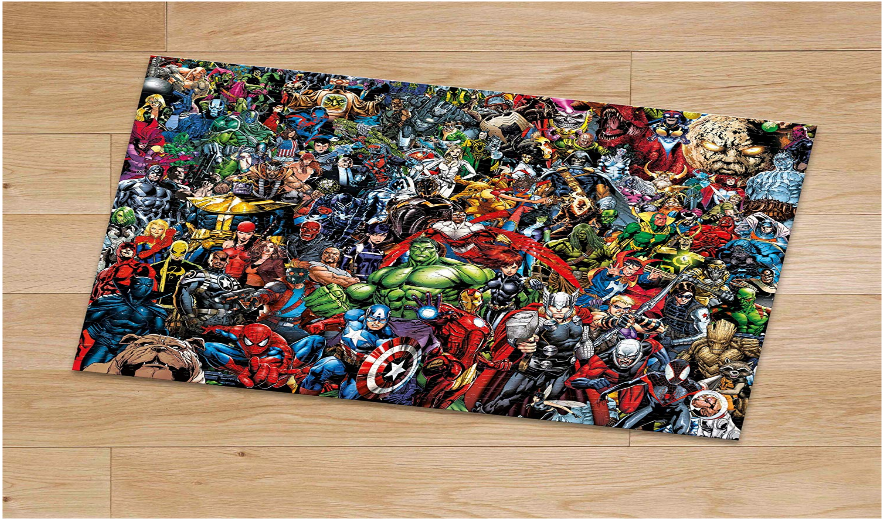
| Metric | Transactions DB | DWH |
|---|---|---|
| Volume | GB/TB | TB/PB |
| Historical | Short-term | Long-Term |
| rows | <1000M | 1000M> |
| Orientation | Product | Subject or multi products |
| Business Units | Product team | Multi organizational units |
| Normalization | Normalized | Not required (De-normalized in many use cases) |
| Data Model | Relational | Star Schema or Multi-dim |
| Intelligence | Reporting | Advanced reporting and Machine Learning |
| Use cases | Online transactions & operations | Centeralized storage (360°) |

# Transnational DB Use cases

# Transnational DB Use cases

# DWH Use cases

# DWH Use cases



# DWH Use cases

# Types of DWH

Types of Data Warehouse

**Enterprise Data Warehouse (EDWH)** It provides decision support service across the enterprise. It offers a unified approach for organizing and representing data (DWH Model). It offers data classifications according to the subject with privileges policy.

**Operational Data Store (ODS):** is a central database that provides an up-to-date (real-time) data from multiple transnational systems for operational reporting into a single DWH.

**Data Mart:** A data mart is a subset of the data warehouse. It specially designed for a particular line of business, such as sales, finance, sales or finance. In an independent data mart, data can collect directly from sources.

# DWH vs ODS vs Data Mart

| Metric | DWH | ODS | Data Mart |
|---|---|---|---|
| Latency | Day -1 | Real-time | Day -1 |
| Data level | Transnational | Transnational | Summary |
| Historical | Long-term | Snapshot | Aggregated Long-Term |
| Size | TB/PB | GB | GB/TB |
| Orientation | Multi sources | Multi sources | Product |
| Business Units | Multi organizational units | Product team | Business team |

# Use Cases of Operational DB vs DWH

# Use case (Operational DB)

- A telecommunication company named **XTec**.

# Use case (Operational DB)

- A telecommunication company named **XTec**.
- They have lots of systems. One of this systems is a CRM system as example of operational DB.

# Use case (Operational DB)

- A telecommunication company named **XTec**.
- They have lots of systems. One of this systems is a CRM system as example of operational DB.
  - The CRM system handles the customer activities with the company including (sales, change in customer plans, and other activities).

# Use case (Operational DB)

- A telecommunication company named **XTec**.
- They have lots of systems. One of this systems is a CRM system as example of operational DB.
  - The CRM system handles the customer activities with the company including (sales, change in customer plans, and other activities).
  - This system has a backend database (MySQL).

# Use case (Operational DB)

- A telecommunication company named **XTec**.
- They have lots of systems. One of this systems is a CRM system as example of operational DB.
  - The CRM system handles the customer activities with the company including (sales, change in customer plans, and other activities).
  - This system has a backend database (MySQL).
  - CRM team can report their sales and customer activities from their database.

## Use case (Operational DB)

- A telecommunication company named **XTec**.
- They have lots of systems. One of this systems is a CRM system as example of operational DB.
  - The CRM system handles the customer activities with the company including (sales, change in customer plans, and other activities).
  - This system has a backend database (MySQL).
  - CRM team can report their sales and customer activities from their database.
  - Product owner can take a decision based on their system backend reports.

# Use case (DWH)

- What is the need for DWH?

# Use case (DWH)

- What is the need for DWH?
  - This company has other systems for example: billing, charging, signaling.

# Use case (DWH)

- What is the need for DWH?
  - This company has other systems for example: billing, charging, signaling.
  - They need to report information related to the CRM, billing, and signaling source systems in one report.

# Use case (DWH)

- What is the need for DWH?
  - This company has other systems for example: billing, charging, signaling.
  - They need to report information related to the CRM, billing, and signaling source systems in one report.
  - So, they need to ingest (transfer) the data from the source systems to one single database.

# Use case (DWH)

- What is the need for DWH?
  - This company has other systems for example: billing, charging, signaling.
  - They need to report information related to the CRM, billing, and signaling source systems in one report.
  - So, they need to ingest (transfer) the data from the source systems to one single database.
  - The decision from the DHW is a **global and strategical decision.**

# Use case (DWH)

- What is the need for DWH?
  - This company has other systems for example: billing, charging, signaling.
  - They need to report information related to the CRM, billing, and signaling source systems in one report.
  - So, they need to ingest (transfer) the data from the source systems to one single database.
  - The decision from the DHW is a **global and strategical decision.**
  - If the company needs to build a machine learning model which needs data from different sources. They need to load the data from a centralized database rather than read each source alone.

# Use case (DWH)

The Full picture required a DWH. However, we still need the other operational databases for product development perspective.

# Use case (ODS)

- Why do we need the ODS?

# Use case (ODS)

- Why do we need the ODS?
- How does it fit in our system?

# Use case (ODS)

**XTec** has a call center system which handles the customer inquiries. This system requires the some data related to usage, customer information, billing details to be calculated and accumulated in **real-time** to be able to give the customer the right answer for his inquires.

# Use case (ODS)

- So, What is the challenge for this system?

# Use case (ODS)

- So, What is the challenge for this system?
  - It needs specific information from different source systems.

# Use case (ODS)

- So, What is the challenge for this system?
  - It needs specific information from different source systems.
  - It requires to track the source system database changes or update in real-time.

# Use case (ODS)

- So, What is the challenge for this system?
  - It needs specific information from different source systems.
  - It requires to track the source system database changes or update in real-time.
  - It's functionality is based on the aggregate data not the transactions for example (It needs the total outgoing calls till time or it needs the total charging amounts from prepaid or the available limits from billing if it is postpaid).

# Use case (ODS)

- ODS is based on change data capture (CDC). This approach used to determine the data change and apply action based on this change.

# Use case (ODS)

- ODS is based on change data capture (CDC). This approach used to determine the data change and apply action based on this change.
- ODS uses the real-time aggregations to support the online systems from different source systems.

# DWH Characteristics

# DWH Characteristics

- The characteristics of DWH:
  - Integrated: *DWH is an integrated environment which allows us to integrate different source systems. Data are modeled (organized) into a unified manner.*
  - Time-Variant: *Data modeled (organized) based on time periods (hourly, daily, weekly, monthly, quarterly, yearly, etc.)*
  - Subject-oriented: *DWH main target is to support business needs for the whole organization including (decision makers, departments, and specific user requirements).*
  - Non-Volatile: *It refers to the data will not erased or deleted (It could be archived and retrieved when needed). Data can be accumulated daily the new snapshots (refreshed at based on the source system interval. For example, It could be updated daily, weekly, and monthly).*

# Hot vs Cold Storage

# Hot vs Cold Storage

SOME DETAILS HERE

# DWH Architecture

# DWH Architecture Layers

- DWH Architecture contains the following layers:
    - Source system layer.

# DWH Architecture Layers

- DWH Architecture contains the following layers:
  - Source system layer.
  - Extraction layer.

# DWH Architecture Layers

- DWH Architecture contains the following layers:
  - Source system layer.
  - Extraction layer.
  - Staging Area.

# DWH Architecture Layers

- DWH Architecture contains the following layers:
  - Source system layer.
  - Extraction layer.
  - Staging Area.
  - Data Modeling.

# DWH Architecture Layers

- DWH Architecture contains the following layers:
  - Source system layer.
  - Extraction layer.
  - Staging Area.
  - Data Modeling.
  - ETL layer.

# DWH Architecture Layers

- DWH Architecture contains the following layers:
  - Source system layer.
  - Extraction layer.
  - Staging Area.
  - Data Modeling.
  - ETL layer.
  - Storage layer.

# DWH Architecture Layers

- DWH Architecture contains the following layers:
    - Source system layer.
    - Extraction layer.
    - Staging Area.
    - Data Modeling.
    - ETL layer.
    - Storage layer.
    - Logical layer.

# DWH Architecture Layers

- DWH Architecture contains the following layers:
  - Source system layer.
  - Extraction layer.
  - Staging Area.
  - Data Modeling.
  - ETL layer.
  - Storage layer.
  - Logical layer.
  - Reporting (UI) layer.

# DWH Architecture Layers

- DWH Architecture contains the following layers:
    - Source system layer.
    - Extraction layer.
    - Staging Area.
    - Data Modeling.
    - ETL layer.
    - Storage layer.
    - Logical layer.
    - Reporting (UI) layer.
    - Metadata layer.

# DWH Architecture Layers

- DWH Architecture contains the following layers:
  - Source system layer.
  - Extraction layer.
  - Staging Area.
  - Data Modeling.
  - ETL layer.
  - Storage layer.
  - Logical layer.
  - Reporting (UI) layer.
  - Metadata layer.
  - System operations layer.

# DWH Architecture Overview

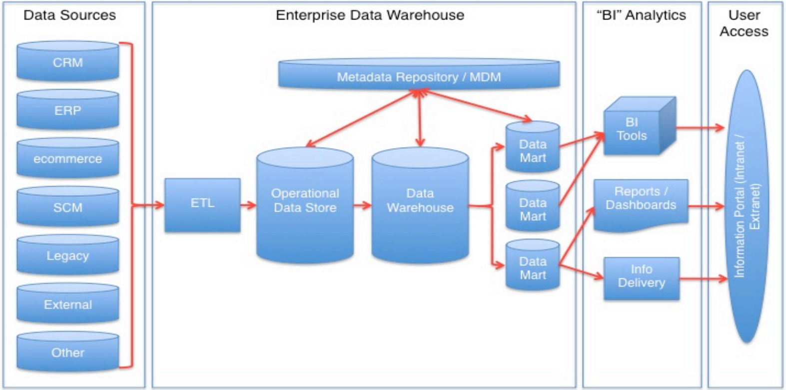



Figure: taken from

# Source System Integration Process

# Source System Integration Process

- In some companies they hire or dedicate a team for this part (business analyst, system analyst, data analyst, or demand team).

# Source System Integration Process

- In some companies they hire or dedicate a team for this part (business analyst, system analyst, data analyst, or demand team).
- Before we start, ALL communications from start till the end should be documented into any format.

# Source System Integration Process

- In some companies they hire or dedicate a team for this part (business analyst, system analyst, data analyst, or demand team).
- Before we start, ALL communications from start till the end should be documented into any format.
  - Conflounce page, Word, or Excel sheet.

# Source System Integration Process

- In some companies they hire or dedicate a team for this part (business analyst, system analyst, data analyst, or demand team).
- Before we start, ALL communications from start till the end should be documented into any format.
  - Conflounce page, Word, or Excel sheet.
  - Make the discussion online and put comments to make the history available always.

# Source System Integration Process

- In some companies they hire or dedicate a team for this part (business analyst, system analyst, data analyst, or demand team).
- Before we start, ALL communications from start till the end should be documented into any format.
  - Conflounce page, Word, or Excel sheet.
  - Make the discussion online and put comments to make the history available always.
  - All tasks should be clear what is the expected output for example (analysis means to document data structure, format, column names, etc..).

# Source System Integration Process

- Requirements gathering.

# Source System Integration Process

- Requirements gathering.
- Identify the stakeholders (Data owner(s)).

# Source System Integration Process

- Requirements gathering.
- Identify the stakeholders (Data owner(s)).
- Data Analysis includes but not only (format, latency, and column definitions).

# Source System Integration Process

- Requirements gathering.
- Identify the stakeholders (Data owner(s)).
- Data Analysis includes but not only (format, latency, and column definitions).
- Connectivity analysis and security (assessment).

# Source System Integration Process

- Requirements gathering.
- Identify the stakeholders (Data owner(s)).
- Data Analysis includes but not only (format, latency, and column definitions).
- Connectivity analysis and security (assessment).
- Technical discussion about the best way to ingest the data.

# Source System Integration Process

- Requirements gathering.
- Identify the stakeholders (Data owner(s)).
- Data Analysis includes but not only (format, latency, and column definitions).
- Connectivity analysis and security (assessment).
- Technical discussion about the best way to ingest the data.
- Data Ingestion method and format.

# Source System Integration Process

- Requirements gathering.
- Identify the stakeholders (Data owner(s)).
- Data Analysis includes but not only (format, latency, and column definitions).
- Connectivity analysis and security (assessment).
- Technical discussion about the best way to ingest the data.
- Data Ingestion method and format.
- Sign or confirmation for every point between the stakeholders.

# Source System Integration Process

- Requirements gathering.
- Identify the stakeholders (Data owner(s)).
- Data Analysis includes but not only (format, latency, and column definitions).
- Connectivity analysis and security (assessment).
- Technical discussion about the best way to ingest the data.
- Data Ingestion method and format.
- Sign or confirmation for every point between the stakeholders.
- This layer deliver a data analysis (Source system interface ) document.

# Extraction Layer

# Extraction Layer

- In some companies they hire or dedicate a team for this part (extraction or ingestion team) but in other companies it is part of the data engineering team.

# Extraction Layer

- In some companies they hire or dedicate a team for this part (extraction or ingestion team) but in other companies it is part of the data engineering team.
- This layer take the output analysis and decisions from the previous layer (source system analysis) and implement the extraction (quality from the previous team output highly affect this team).

# Extraction Layer

- In some companies they hire or dedicate a team for this part (extraction or ingestion team) but in other companies it is part of the data engineering team.
- This layer take the output analysis and decisions from the previous layer (source system analysis) and implement the extraction (quality from the previous team output highly affect this team).
- There are many consideration this team need to take care about or deal with but we can summarize it in the following:

- In some companies they hire or dedicate a team for this part (extraction or ingestion team) but in other companies it is part of the data engineering team.
- This layer take the output analysis and decisions from the previous layer (source system analysis) and implement the extraction (quality from the previous team output highly affect this team).
- There are many consideration this team need to take care about or deal with but we can summarize it in the following:
  - Data latency will affect the tool and the methodology (stream or batch).

# Extraction Layer

- In some companies they hire or dedicate a team for this part (extraction or ingestion team) but in other companies it is part of the data engineering team.
- This layer take the output analysis and decisions from the previous layer (source system analysis) and implement the extraction (quality from the previous team output highly affect this team).
- There are many consideration this team need to take care about or deal with but we can summarize it in the following:
  - Data latency will affect the tool and the methodology (stream or batch).
  - Data extraction method (push or pull).

# Extraction Layer

- In some companies they hire or dedicate a team for this part (extraction or ingestion team) but in other companies it is part of the data engineering team.
- This layer take the output analysis and decisions from the previous layer (source system analysis) and implement the extraction (quality from the previous team output highly affect this team).
- There are many consideration this team need to take care about or deal with but we can summarize it in the following:
  - Data latency will affect the tool and the methodology (stream or batch).
  - Data extraction method (push or pull).
  - Data size and format compared with the available resources for this project.

- In some companies they hire or dedicate a team for this part (extraction or ingestion team) but in other companies it is part of the data engineering team.
- This layer take the output analysis and decisions from the previous layer (source system analysis) and implement the extraction (quality from the previous team output highly affect this team).
- There are many consideration this team need to take care about or deal with but we can summarize it in the following:
  - Data latency will affect the tool and the methodology (stream or batch).
  - Data extraction method (push or pull).
  - Data size and format compared with the available resources for this project.
- This layer output is a minimal data cleansing (no transformation) into the staging/landing layer.

# Staging Layer

## Staging Layer

- This layer handled by the same team who own the storage part in most of the organizations.

# Staging Layer

- This layer handled by the same team who own the storage part in most of the organizations.
- Segregation of this layer if it uses different storage type or multi-teams access this layer for a different purpose (ex: Kafka) *Kafka is not a storage layer but it could be landing layer*.

# Staging Layer

- This layer handled by the same team who own the storage part in most of the organizations.
- Segregation of this layer if it uses different storage type or multi-teams access this layer for a different purpose (ex: Kafka) *Kafka is not a storage layer but it could be landing layer*.
- All the ETL layers are working on top of this layer.

# Staging Layer

- This layer handled by the same team who own the storage part in most of the organizations.
- Segregation of this layer if it uses different storage type or multi-teams access this layer for a different purpose (ex: Kafka) *Kafka is not a storage layer but it could be landing layer*.
- All the ETL layers are working on top of this layer.
- The decision of the storage type is based on the use case and the data.

# Data Modeling

# Data Modeling Objective

- Understanding the data modeling and its roles.

# Data Modeling Objective

- Understanding the data modeling and its roles.
- Be aware about its importance.

# Data Modeling Objective

- Understanding the data modeling and its roles.
- Be aware about its importance.
- Explore different types of data modeling.

# Data Modeling Objective

- Understanding the data modeling and its roles.
- Be aware about its importance.
- Explore different types of data modeling.
- We will not go in details about how to design in this part (we will explain it later and in the appendix).

# What is data model?

Data model is

- An abstract model that organizes elements of data.

# What is data model?

Data model is

- An abstract model that organizes elements of data.
- It describes the objects, entities and data structure properties, semantic, and constraint.

# What is data model?

Data model is

- An abstract model that organizes elements of data.
- It describes the objects, entities and data structure properties, semantic, and constraint.
- It formalizes the relationship between entities.

# What is data model?

Data model is

- An abstract model that organizes elements of data.
- It describes the objects, entities and data structure properties, semantic, and constraint.
- It formalizes the relationship between entities.
- It describes how application (report) API data manipulation.

# What is data model?

Data model is

- An abstract model that organizes elements of data.
- It describes the objects, entities and data structure properties, semantic, and constraint.
- It formalizes the relationship between entities.
- It describes how application (report) API data manipulation.
- It describes the conceptual design of a business or an application with its flow, logic, semantic information (rules), and how things are done.

# What is data model?

Data model is

- An abstract model that organizes elements of data.
- It describes the objects, entities and data structure properties, semantic, and constraint.
- It formalizes the relationship between entities.
- It describes how application (report) API data manipulation.
- It describes the conceptual design of a business or an application with its flow, logic, semantic information (rules), and how things are done.
- It refers to a set of concepts used in defining such as entities, attributes, relations, or tables.

# What is data model?

Data model is not

- a science.

Data model is

# What is data model?

Data model is not

- a science.
- a static design for each organization.

Data model is

# What is data model?

Data model is not

- a science.
- a static design for each organization.
- a type of database.

Data model is

# What is data model?

Data model is not

- a science.
- a static design for each organization.
- a type of database.
- a new invention which needs to be done for each project.

Data model is

# What is data model?

Data model is not

- a science.
- a static design for each organization.
- a type of database.
- a new invention which needs to be done for each project.

Data model is

- a general concepts which lead to build full architecture.

# What is data model?

Data model is not

- a science.
- a static design for each organization.
- a type of database.
- a new invention which needs to be done for each project.

Data model is

- a general concepts which lead to build full architecture.
- an engineering design practices.

# What is data model?

Data model is not

- a science.
- a static design for each organization.
- a type of database.
- a new invention which needs to be done for each project.

Data model is

- a general concepts which lead to build full architecture.
- an engineering design practices.
- different based on the use case and the database type.

# What is data model?

Data model is not

- a science.
- a static design for each organization.
- a type of database.
- a new invention which needs to be done for each project.

Data model is

- a general concepts which lead to build full architecture.
- an engineering design practices.
- different based on the use case and the database type.
- customizable and we can utilize some of ready built architecture.

# What is data model?

Data model is not

- a science.
- a static design for each organization.
- a type of database.
- a new invention which needs to be done for each project.

Data model is

- a general concepts which lead to build full architecture.
- an engineering design practices.
- different based on the use case and the database type.
- customizable and we can utilize some of ready built architecture.
- affecting the information reporting

# What is data model?

Data model is

- The initial part before starting integration with any new source system.

# What is data model?

Data model is

- The initial part before starting integration with any new source system.
- It the connection layer between the business requirements and the technical design.

# What is data model?

Data model is

- The initial part before starting integration with any new source system.
- It the connection layer between the business requirements and the technical design.
- It is also the translation between logical and physical layer.

# What is data model?

Data model is

- The initial part before starting integration with any new source system.
- It the connection layer between the business requirements and the technical design.
- It is also the translation between logical and physical layer.
- It is unified across the all systems and has the same patterns and practices.

# What is data model?

Data model is

- The initial part before starting integration with any new source system.
- It the connection layer between the business requirements and the technical design.
- It is also the translation between logical and physical layer.
- It is unified across the all systems and has the same patterns and practices.
- It could be engage with any source systems integration from early stages.

# What is data model?

Data model is

- The initial part before starting integration with any new source system.
- It the connection layer between the business requirements and the technical design.
- It is also the translation between logical and physical layer.
- It is unified across the all systems and has the same patterns and practices.
- It could be engage with any source systems integration from early stages.
- This stage output is data model design document or mapping sheet.

# Why does data models are important?

- Data models are currently affecting software design.
- It decides how engineers will think about the problem they are solving.

# Data Model Design vs Implementation

REVIEW THIS EXAMPLE

- You need to build a home. So, how do we design this home?

# Data Model Design vs Implementation

REVIEW THIS EXAMPLE

- You need to build a home. So, how do we design this home?
    - Determine if the home is one level or multi-level and decide man bedrooms and bathrooms for each floor. (User needs)

# Data Model Design vs Implementation

## REVIEW THIS EXAMPLE

- You need to build a home. So, how do we design this home?
  - Determine if the home is one level or multi-level and decide man bedrooms and bathrooms for each floor. (User needs)
  - Hire an architect to put the architecture in more detailed way for example, the size for each room, the distribution of the wireds, where the plumbing fixtures will be placed, etc. (Architecture phase)

## REVIEW THIS EXAMPLE

- You need to build a home. So, how do we design this home?
    - Determine if the home is one level or multi-level and decide man bedrooms and bathrooms for each floor. (User needs)
    - Hire an architect to put the architecture in more detailed way for example, the size for each room, the distribution of the wireds, where the plumbing fixtures will be placed, etc. (Architecture phase)
    - Decide the decorations, colors for each room, carpets, etc.

# Data Model Design vs Implementation

## REVIEW THIS EXAMPLE

- You need to build a home. So, how do we design this home?
  - Determine if the home is one level or multi-level and decide man bedrooms and bathrooms for each floor. (User needs)
  - Hire an architect to put the architecture in more detailed way for example, the size for each room, the distribution of the wireds, where the plumbing fixtures will be placed, etc. (Architecture phase)
  - Decide the decorations, colors for each room, carpets, etc.
- What do we do for the implementation?

## REVIEW THIS EXAMPLE

- You need to build a home. So, how do we design this home?
  - Determine if the home is one level or multi-level and decide man bedrooms and bathrooms for each floor. (User needs)
  - Hire an architect to put the architecture in more detailed way for example, the size for each room, the distribution of the wireds, where the plumbing fixtures will be placed, etc. (Architecture phase)
  - Decide the decorations, colors for each room, carpets, etc.
- What do we do for the implementation?
  - Hire a contractor to build (implement the design) the home.

# Data Model Design vs Implementation

## REVIEW THIS EXAMPLE

- You need to build a home. So, how do we design this home?
  - Determine if the home is one level or multi-level and decide man bedrooms and bathrooms for each floor. (User needs)
  - Hire an architect to put the architecture in more detailed way for example, the size for each room, the distribution of the wireds, where the plumbing fixtures will be placed, etc. (Architecture phase)
  - Decide the decorations, colors for each room, carpets, etc.
- What do we do for the implementation?
  - Hire a contractor to build (implement the design) the home.
  - This phase will implement the design but it also include some detail related to the actual way to build the tools and the material. (Physical Design)

# Data Model Design Principle

Decide what is the limitation of this part what is in and what is out to be part of the appendix

- facts, start schema, dimensional modeling techniques.
- Fact Tables and Dimension Tables.
- Multidimensional Model(Star, Snowflake, and Galaxy Schema).
- Support Roll Up, Drill Down, and Pivot Analysis
- Time Phased / Temporal Data
- Operational Logical and Physical Data Models
- Normalization and Denormalization
- Model Granularity : Level of Detail

# ETL Process

# What is ETL?

- The ETL (Extraction, Transformation, Loading) is main core function for any data engineering (DWH) team.

# What is ETL?

- The ETL (Extraction, Transformation, Loading) is main core function for any data engineering (DWH) team.
- This team takes the delivered output from the previous stage (data modeling) and start to implement the mapping.

# What is ETL?

- The ETL (Extraction, Transformation, Loading) is main core function for any data engineering (DWH) team.
- This team takes the delivered output from the previous stage (data modeling) and start to implement the mapping.
- The implementation of the ETL preferred to be unified across the team members and the organization unless there is a special case of license of capacity.

# ETL Characteristics

- Successful ETL design have the following characteristics:

# ETL Characteristics

- Successful ETL design have the following characteristics:
  - ☑ Maintainable.

# ETL Characteristics

- Successful ETL design have the following characteristics:
  - ☑ Maintainable.
  - ☑ Reusable.

# ETL Characteristics

- Successful ETL design have the following characteristics:
  - ☑ Maintainable.
  - ☑ Reusable.
  - ☑ Well-Performed.

# ETL Characteristics

- Successful ETL design have the following characteristics:
  - ☑ Maintainable.
  - ☑ Reusable.
  - ☑ Well-Performed.
  - ☑ Reliable.

# ETL Characteristics

- Successful ETL design have the following characteristics:
  - ☑ Maintainable.
  - ☑ Reusable.
  - ☑ Well-Performed.
  - ☑ Reliable.
  - ☑ Resilient.

# ETL Characteristics

- Successful ETL design have the following characteristics:
  - ☑ Maintainable.
  - ☑ Reusable.
  - ☑ Well-Performed.
  - ☑ Reliable.
  - ☑ Resilient.
  - ☑ Secure.

# ETL Best Practice

- To implement the previous characteristics you need to have the following:

# ETL Best Practice

- To implement the previous characteristics you need to have the following:
  - ☑ Logging.

# ETL Best Practice

- To implement the previous characteristics you need to have the following:
  - ☑ Logging.
  - ☑ Auditing.

- To implement the previous characteristics you need to have the following:
  - ☑ Logging.
  - ☑ Auditing.
  - ☑ Data Lineage.

# ETL Best Practice

- To implement the previous characteristics you need to have the following:
  - ☑ Logging.
  - ☑ Auditing.
  - ☑ Data Lineage.
  - ☑ Modularity.

# ETL Best Practice

- To implement the previous characteristics you need to have the following:
  - ☑ Logging.
  - ☑ Auditing.
  - ☑ Data Lineage.
  - ☑ Modularity.
  - ☑ Atomicity.

# ETL Best Practice

- To implement the previous characteristics you need to have the following:
  - ☑ Logging.
  - ☑ Auditing.
  - ☑ Data Lineage.
  - ☑ Modularity.
  - ☑ Atomicity.
  - ☑ Error Handling.

# ETL Best Practice

- To implement the previous characteristics you need to have the following:
  - ☑ Logging.
  - ☑ Auditing.
  - ☑ Data Lineage.
  - ☑ Modularity.
  - ☑ Atomicity.
  - ☑ Error Handling.
  - ☑ Managing Bad Data (Rejection Handling).

# ETL Logging

- Logging

# ETL Logging

- Logging
  - Logging.

# ETL Logging

- Logging
  - Logging.
  - Logging.

# ETL Logging

- Logging
    - Logging.
    - Logging.
    - Logging.

# ETL Auditing

- Logging

# ETL Auditing

- Logging
  - Logging.

# ETL Auditing

- Logging
  - Logging.
  - Logging.

# ETL Auditing

- Logging
  - Logging.
  - Logging.
  - Logging.

# ETL Data Lineage

- Logging

# ETL Data Lineage

- Logging
  - Logging.

# ETL Data Lineage

- Logging
  - Logging.
  - Logging.

# ETL Data Lineage

- Logging
  - Logging.
  - Logging.
  - Logging.

# ETL Modularity

- Logging

# ETL Modularity

- Logging
    - Logging.

# ETL Modularity

- Logging
    - Logging.
    - Logging.

# ETL Modularity

- Logging
    - Logging.
    - Logging.
    - Logging.

# ETL Atomicity

- Logging

# ETL Atomicity

- Logging
  - Logging.

# ETL Atomicity

- Logging
  - Logging.
  - Logging.

# ETL Atomicity

- Logging
  - Logging.
  - Logging.
  - Logging.

# ETL Error Handling

- Logging

# ETL Error Handling

- Logging
    - Logging.

# ETL Error Handling

- Logging
  - Logging.
  - Logging.

## ETL Error Handling

- Logging
  - Logging.
  - Logging.
  - Logging.

# ETL Rejection Handling

- Logging

# ETL Rejection Handling

- Logging
    - Logging.

# ETL Rejection Handling

- Logging
  - Logging.
  - Logging.

# ETL Rejection Handling

- Logging
    - Logging.
    - Logging.
    - Logging.

# ETL vs ELT When? Why?

# Storage layer

# Storage layer

# Logical layer

# Logical layer

# Reporting (UI) layer

# Reporting (UI) layer

# Metadata layer

# Metadata layer

# System operations layer

# System operations layer

# DWH Architecture Overview

There are mainly three types of Datawarehouse Architectures: -

- Single-tier architecture.
- Two-tier architecture.
- Three-tier architecture.

# File Formats

# File Formats

- Any Big Data solution working based distributed systems.

# File Formats

- Any Big Data solution working based distributed systems.
- What is distributed systems in brief?

# Data Encoding and Formats

# Data Encoding and Formats

- Any Big Data solution working based distributed systems.

# Data Encoding and Formats

- Any Big Data solution working based distributed systems.
- What is distributed systems in brief?

# Data Compression Technique

# Data Compression Technique

- Any Big Data solution working based distributed systems.

# Data Compression Technique

- Any Big Data solution working based distributed systems.
- What is distributed systems in brief?

# Data Archiving and Retention

# Data Archiving and Retention

- some details about hot vs cold storage,

# DWH On Cloud

# Further Readings and Assignment