Balaji Katakam

1. What is metadata? Distinguish technical, business, and process metadata. Give examples.

Solution:

In layman terms, Metadata can be defined as a description data about data in the warehouse. Metadata stores description about each type of variable used in the data warehouse. The data description that is used to represent the data is known as metadata. For an example, we can consider that the index table of a book works as a metadata for the description and contents of each part in the book including the page number. Metadata can also be called as data that leads us to detailed information about the data stored in a data warehouse.

With respect to data warehouse, I would be able to explain metadata in such a manner below.

- Metadata can be called as a road-map to a data warehouse.
- Metadata in a data warehouse helps us define the objects in a data warehouse.
- Metadata can be used as a directory. This directory helps us build the DSS (decision support system) to locate the contents of a data warehouse.

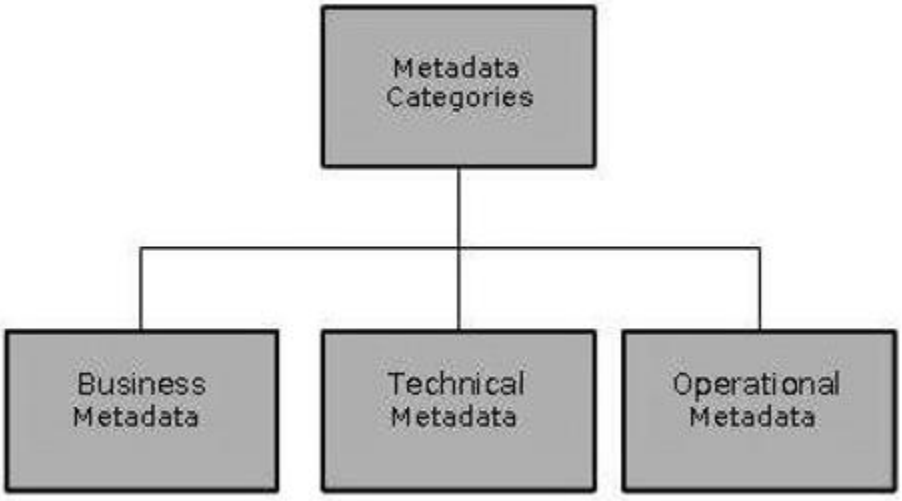
Data Warehouse Metadata are basically tiny short text-based descriptors stored in 1 or many metadata repositories that include.

a) Specific in detail Information related to the contents of the data warehouse, their location, their form of storage, the datatype, the usage and their structure
b) Information related to the processes that concurrently and continuously keep occurring in the data warehouse and the back-stage, especially concerning the refreshment of the warehouse with clean, up-to-date, semantically and structurally reconciled data
c) Every minute detail Information on the implicit semantics of data along with any other kind of data that helps the end-user exploit the information of the warehouse
d) Information on the infrastructure and physical characteristics of components and the sources of the data warehouse
e) Information regarding the security, authentication, authentication type, layers of security information, transmission information and usage statistics that can help the administrator, to optimize and improve the operational performance of the data warehouse to the best

Example of metadata:

Table of Contents

The various types of metadata in a data warehouse are:



Reference: https://www.tutorialspoint.com/dwh/dwh_metadata_concepts.htm

Business Metadata is defined as the business specific data i.e. it has the descriptive data regarding the ownership of information of each database, business specific definition of each data source and data mart, and the changing policies in the business.

Technical Metadata – This contains information regarding the various types of databases in system, their names, table and column names and their sizes, data types and allowed values. Technical metadata also includes information regarding the primary and foreign key attributes for the databases. The referential integrity is maintained using key attributes.

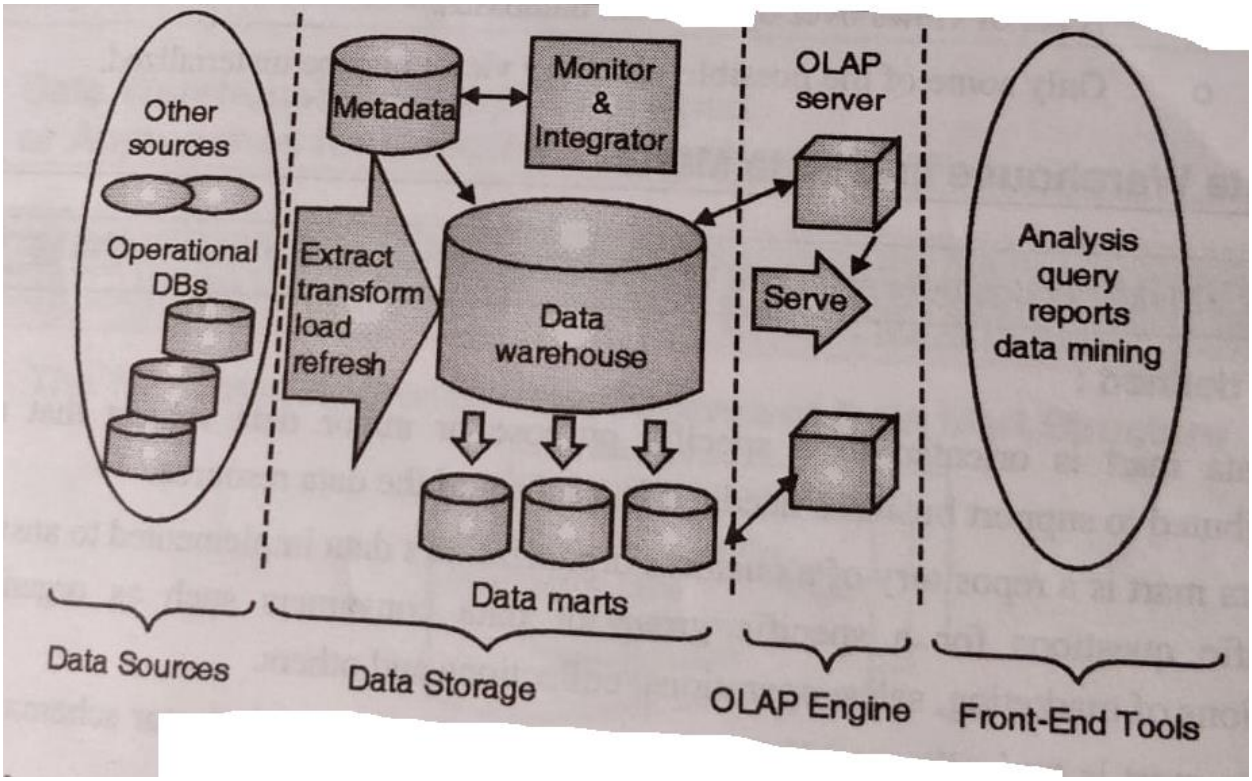
Process Metadata − It includes the information related to the various processes carried out throughout the data warehouse. It is an integration mechanism of the various data sources, basically a collection of data that stores that outputs of the operational data stores.

Business Metadata	Technical Metadata	Process Metadata
Business Metadata stores business definitions of the data.	Technical Metadata represents the ETL process.	Process Metadata are subject oriented, integrated, time current, volatile collection of data that supports systems activities and outputs of the operational data stores.
It describes the ownership of data and related information.	It describes data mapping and transformation from source systems to data warehouse.	They are dynamic in nature.
High level definitions of all fields in the data warehouse and details on cubes, aggregates and DataMart are possessed by 'Business Metadata'.	It is usually more complex than business metadata and it involves multiple dependencies at-times.	It is consistent with all the changes occurring in the system. For E.g. Addition, update and deletion of transactional data at any given time.
E.g. Following information needs to be provided to describe business metadata: • DW Table Name • DW Column Name • Business Name • Field Type	E.g. It can be structured in following ways: • Source Database • Target Database • Source Tables • Source Columns • Target Table • Target Column • Transformation	E.g. • Current Flag Indicator • Load Date • Load Cycle Identifier • Update Date • Operational System(s) Identifier

2. What is the value of architecture in data warehousing?

The main essence of a data warehouse is to integrate data related to various departments from various types of systems into a single database so that this data can be used to help the firm in bringing out predictive analytics, high level data summarization to get a better understanding of the firm's current performance and compare it with the past.

The architecture plays a major role in the success of a data warehouse since it's very difficult to create a data warehouse with minimum experience into the live environment and in depth understanding of the various databases and their working. Once the architecture is figured out in a well-defined format the rest of the project is covered safe and has the right direction of implementation. Since only 20% of the data warehouse projects are successful. The planning, layout, structure and architecture of a data warehouse play a major role in success of a data warehouse for a firm.



Reference: Data Warehousing and Mining by Pallavi Halarnkar and Arti Deshpande

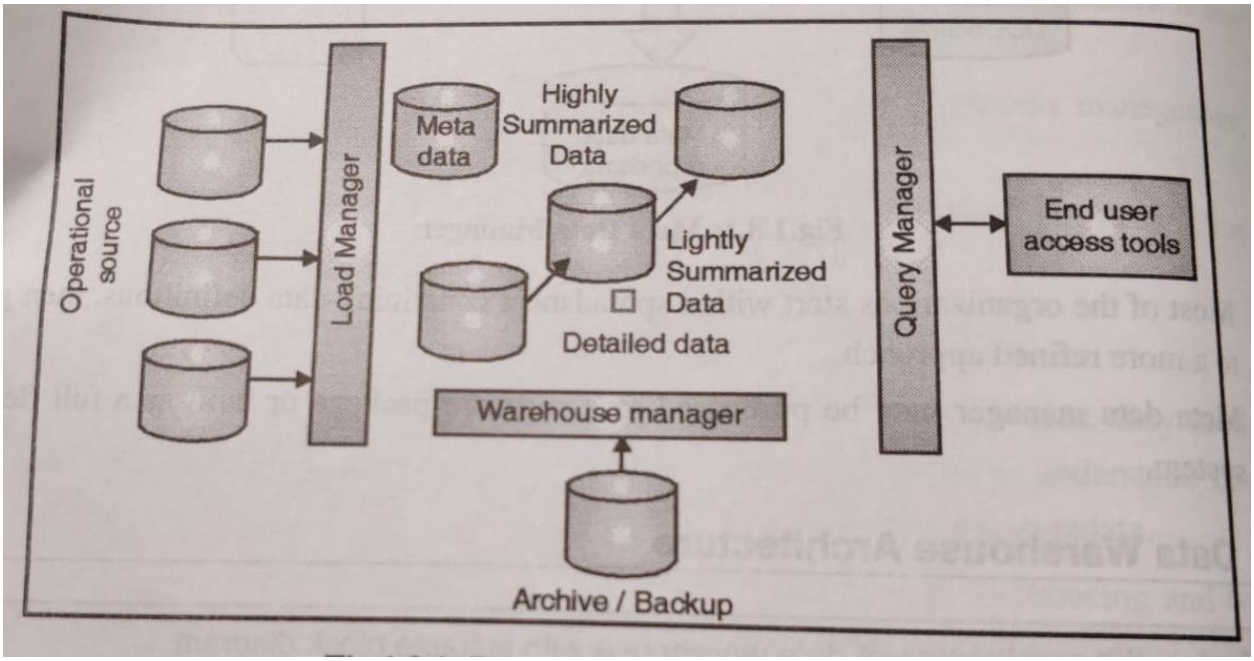
There are 3 tiers in a data warehouse architecture.

1. Bottom Tier (Data Sources and Data Storage)

- This part contains the various types of databases in the server and the applications.
- These various types of file system of databases can be hierarchical, flat or legacy systems
- These databases are extracted into the data warehouses by using API's (basically also called a gateways)
- The most commonly used gateways are JDBC, ODBC, OLE-DB to get connected to these databases.

2. Middle Tier
    - This tier contains the online analytical processing engine. This can either be MOLAP / ROLAP. Depending on the requirements of the company.
    - This connects the data warehouse with the front-end tools.

3. Top Tier
    - This tier consists of the front-end tools that can provide the Analysis, Queries, Reports and data mining.
    - The top tier is used mainly by the C-Suite executives such as CEO, CIO, CTO and the CFO. They can get a quick view of the company's current status just by having a quick glance over the graphs and projection provided by the front-end tools.
    - The various types of tools that can be used in this layer are Tableau, QlikView, Spotfire and Alteryx. They are one of the major leaders in the market currently to be used as the Business Intelligence and Reporting tools in the fortune 500 companies.



Balaji Katakam

Reference: Data Warehousing and Mining by Pallavi Halarnkar and Arti Deshpande

This is the procedure involved in extracting the data from the data source till the End user access tools.

The data in a data warehouse comes from the databases of the operational systems and the legacy systems and files. This data is extracted from these systems using various types of API's or system calls, and the data us stored in the data staging area where all the data is cleaned, ordered, sorted, transformed to prepare the data to be pushed into the data warehouse.

Once the data warehouse is setup, we can utilize a OLAP engine to help integrate a layer to help query the data, which will further be connected to the End-User access tools which provide the final end user applications such as data visualization, mining, important graphs and statistics.

3. Why do we need a dimensional model? Why isn't ER sufficient?

Let me start this off by defining what each one of them does and define them out.

Dimensional Model

The definition of a Dimensional model is given as:
A dimensional model is a data structure specially designed for the most optimal use of a Data warehouse. This concept of Dimensional Modelling was developed and first introduced by Ralph Kimball and comprises of "fact" and "dimension" tables.
A Dimensional model is created to introduce functions such as read, summarize, analyze numeric information like values, counts, etc. in a data warehouse.

ER Model

The ER (Entity Relationship) Model is defined as
The relational models are optimized for addition, updating and deletion of data in a real-time Online Transaction System.

The drawbacks of an ER model are:

- There is Loss of information: Some parts of the whole information remain hidden in the ER model diagram

- The ER model represents specific and limited relationship in its model, whereas in comparison with other data models like relational model etc. they have better relationship representations.
- Data manipulation cannot be represented in an ER model.
- ER models are very popular across the market for creating and designing high level design for a database

- There is no specific nomenclature or industry standard or rules for notation. This make is difficult to communicate but any layman can create an ER diagram with a basic understanding of it.

The advantages of having a dimensional model

- Bring together data from many different sources and create a single, consistent user view.
- Support the ad hoc queries that arise from real business questions.
- Maximize flexibility and scalability.
- Optimize the end-user experience.

Balaji Katakam

A comparison of Dimensional Model and ER Model

Dimensional Model	ER Model
Support Ad-hoc querying for business analyst and complex analysis	Support for OLTP and ODS (Operational Data Store)
Entities are linked through a series of joins	Entities are linked through a series of joins
Simplify the view of the data model. You can rotate the data cube to see different views of the data	The data model has only one dimension
It is asymmetric	It is symmetric. All tables look the same.
Permits redundancy	Removes the redundancy in data
It is extensible to accommodate unexpected new data elements and new design decisions. The application is not changed	If the model is modified, the applications are modified.
It is robust. The dimensional model design can be done independent of expected query patterns.	It is variable in structure and very vulnerable to changes in the user's querying habits.
The model is easy and understandable	The model for enterprise is very hard for people to visualize and keep in their heads.
The model really models a business. It is a body of standard approaches for handling common modeling situations in the business world.	The model does not really model a business. It models the micro relationships among data elements.

The reason behind ER Model not being suitable for Data warehouse is

- It's not easy to understand an ER model by any normal or end user. There are no specific industry standards to create an ER model. ER Model can be a good way for database specific modelling.
- One cannot navigate through an ER Model to go back and forth in the model.
- ER Model's do not have a User Interface/ User experience which make it difficult for end users to use it.
- Complex queries cannot be executed on an ER Model. They can be used to execute simple queries.
- ER Models are highly normalized tables and relational tables. They have high performance for retrieval and changes.

4. **How do we track and change history?**

Balaji Katakam

In a data warehouse there are a few changes that occur in the system. These changes are very few and have varied impacting and priority level of the information in the data warehouse. On an Average there are less than 12-15 changes per year in a customer's data. These changes have 3 different types. They should either be tracked or should not be tracked. Even if these changes must be tracked then these changes need to track to what level/extent/number of layers.

A few examples that I can suggest would be:

1. A change in customer email address.
   A change in email address is not a very crucial information, so we wouldn't maintain a history of his past email id.
2. A change in customer address
   Change in customer address is important so that we can collect more information about him. This information can be stored for up to most recent 3 changes in history

There are basically 3 types of changes in the data warehouse. These 3 types of changes are for slowly changing dimensions.

Type 1 Change: Correction of errors

- These changes are changes based on correction of errors in the OLTP/Source systems.
- For Example: Change in Customer Name/Gender/Birth Date.
- These changes don't need to have a history. The company is not concerned with a change in gender/Name/Date of Birth of a customer.
- These changes are directly overwritten over the current data.
- There is no history preserved in the data warehouse for these types of information.

It's very easy to implement this. It directly overwrites the specific cell that contains old data and overwrites it with the new data. The old values are not preserved, and the key values remain unaffected.

Type 2 Change: Preservation History

The general principles for this type of change are:

- They usually relate to true changes in the source systems.
- There is a need to preserve history of this detail in the data warehouse since history of this variable is important for the company.
- Each and Every Change of the attribute must be preserved and tracked.

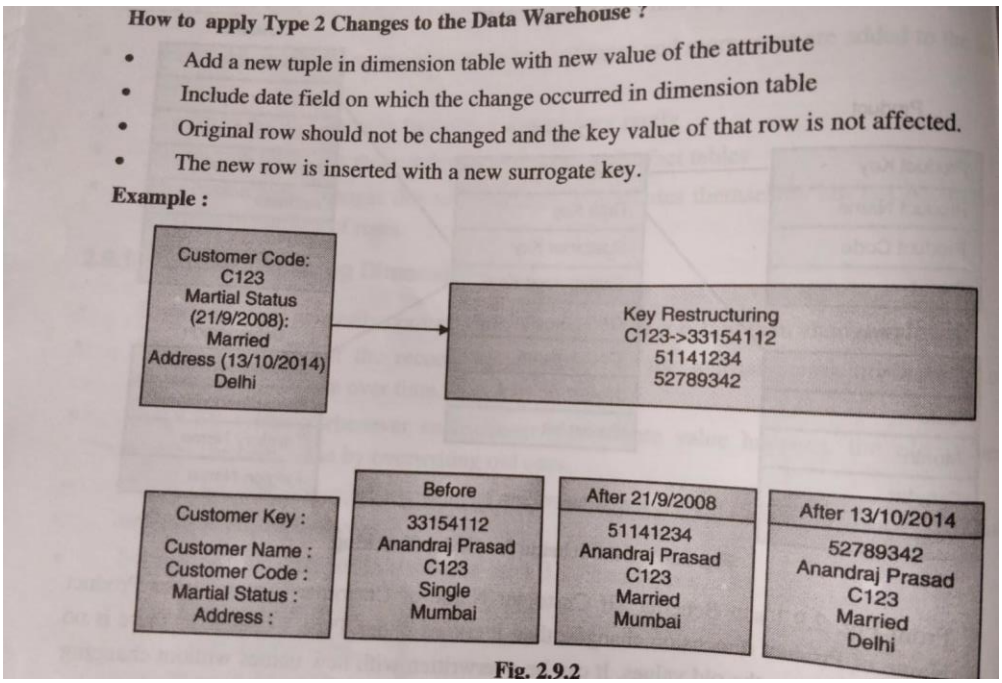


Fig. 2.9.2

Type 3 Change: Tentative Soft Revisions

The general principles for type 3 changes are:

- They usually relate to “Soft” or tentative changes in the source systems
- There is need to keep track of history with old and new values of the changes involved in the attribute
- These changes are used to compare performance across the transition of every change in the system
- This provides the ability to track forward and backward
-

How to apply Type 3 Changes to the data warehouse:

- No new dimension row is needed
- The existing queries will seamlessly switch to the current value
- Any queries that need to use the old value must be revised accordingly
- The technique works best for one soft change at a time
- If there is a succession of changes, more sophisticated techniques must be advised

5. **What does grain mean and why is it important?**

Grain means the highest level of detail that can be made available to a given fact table and its schema. It is generally given by the number of records per key within the table. The Grain in a database table has several important features that make it a key feature/ element while building/creating fact tables or Schemas.

When you identify the grain, you specify exactly what a fact table record contains. The grain conveys the level of detail that is associated with the fact table measurements. When you identify the grain, you also decide on the level of detail you want to make available in the dimensional model. If more detail is included, the level of granularity is lower. If less detail is included, the level of granularity is higher.
Each table (either fact or dimension) contains some level of detail that is associated with it. The grain of the dimensional model is the finest level of detail that is implied when the fact and dimension tables are joined. For example, the granularity of a dimensional model that consists of the dimensions Date, Store and Product is 'product sold in store by day'.

By declaring the grain, you can visualize the dimensionality of a fact table/schema very precisely, and you can therefore confidently examine your data sources, deciding whether a dimension can be attached to this data. The discipline of insisting on the grain declaration at the beginning of a dimensional design keeps you from making the mistake of combining entities that don't coexist in real data sources. Every fact table design must be rooted in the realities of available physical data sources.

All grain definitions should start at the lowest, most atomic grain and should describe the physical process that collects the data. Once the grain of the fact table is established with such clarity, the next steps of the design process can proceed smoothly. Continuing with our retail example, we can immediately include or exclude possible dimensions from the logical design of the retail fact table. Benefiting from the very atomic definition, we can propose many dimensions. The power of keeping to the grain arises from the clarity that the grain definition supplies to the design process. Keeping to the grain means building physical fact tables around each atomic measurement event. These tables are the least difficult to implement and they provide the most durable and flexible foundation for fact table processing.

The grain declaration lets us think creatively about adding dimensions to a fact table design that may not obviously be present in the source data. In retail sales data, marketplace causal factors like promotions and competitive effects may be very important to understanding the data, but this information may not be present in a literal sense in the data extract. The most important result of declaring the grain of the fact table is anchoring the discussion of the dimensions. Declaring the grain means saying exactly what a fact table record represents. Remember that a fact table record captures a measurement. Declaring the grain lets you be equally clear about the measured numeric facts. Simply put, the facts must be true to the grain.

References:
Textbook: Ralph Kimball
Data warehousing and Mining by Arti Deshpande and Pallavi Halarnkar

Balaji Katakam

https://www.guru99.com/dimensional-model-data-warehouse.html
https://pctechnicalpro.blogspot.com/2017/04/advantages-disadvantages-er-model-dbms.html
http://www.sys-seminar.com/EE/Files/dimensional_data_modeling.pdf
https://www.tutorialspoint.com/dwh/dwh_metadata_concepts.htm
https://tdwi.org/articles/2006/05/09/the-importance-of-metadata-for-etl.aspx
https://www.oreilly.com/library/view/data-warehousing-fundamentals/9780471412540/9780471412540_metadata_in_the_data_warehouse.html
https://en.wikipedia.org/wiki/Extract,_transform,_load
https://www.tutorialspoint.com/dwh/dwh_metadata_concepts.htm
https://www.researchgate.net/publication/27313315_Does_data_warehouse_end-user_metadata_add_value
https://en.wikipedia.org/wiki/Metadata_repository
https://www.guru99.com/data-warehouse-architecture.html
https://www.tutorialspoint.com/dwh/dwh_architecture.htm
https://panoply.io/data-warehouse-guide/data-warehouse-architecture-traditional-vs-cloud/
http://datawarehouse4u.info/
https://www.tutorialspoint.com/dwh/dwh_olap.htm
https://en.wikipedia.org/wiki/Dimensional_modeling
https://www.thoughtspot.com/fact-and-dimension/dimensional-data-modeling-4-simple-steps
https://www.guru99.com/dimensional-model-data-warehouse.html
https://en.wikipedia.org/wiki/Entity%E2%80%93relationship_model
https://www.guru99.com/er-modeling.html
https://www.tutorialspoint.com/dbms/er_model_basic_concepts.htm
https://www.1keydata.com/datawarehousing/slowly-changing-dimensions-type-1.html
https://www.nuwavesolutions.com/slowly-changing-dimensions/
http://datawarehouse4u.info/SCD-Slowly-Changing-Dimensions.html
http://www.datamartist.com/data-granularity-avoid-going-against-the-grain
https://www.ibm.com/support/knowledgecenter/en/SS9UM9_9.1.1/com.ibm.datatools.dimensional.ui.doc/topics/c_dm_design_cycle_2_idgrain.html
http://etltesters.blogspot.com/2014/05/what-is-granularity.html
http://www.kimballgroup.com/2007/07/keep-to-the-grain-in-dimensional-modeling/
http://www.kimballgroup.com/2003/03/declaring-the-grain/