Data Warehousing and Data Mining

ETL-2

Alaa Khalaf Hamoud

Contents

Contents

Contents

4.1.1 Selection

- Takes place at the **beginning** of data transformation (usually **forms part** of the **extraction function itself**).
- You select either **whole records** or **parts of several records** from the source systems.
- In some cases, the **composition** **of the source structure** may not be **amenable for selection of the necessary parts during** data extraction.
  - It is **prudent** to extract the whole record and then do the **selection** as part of the transformation function.

4.1.2 Conversion

- This is an **all-inclusive task**.
- It includes **a large variety of rudimentary conversions** of single fields for two primary reasons:
  1. To **standardize among the data extractions** from disparate source systems,
  2. To **make the fields usable and understandable** to the users.

4.1.3 Summarization

- **Sometimes**, it is not **feasible** to keep data at the **lowest level of detail** in your DW.
- May be none of DW users need data at the **lowest granularity** for analysis or querying.
  - For example, for a grocery chain, sales data at the lowest level of detail for every transaction at the checkout may not be needed. Storing sales by product by store by day in the DW may be quite adequate.
  - Here, data transformation function includes **summarization** of daily sales by product and by store.

4.1.4 Enrichment

- This task is the **rearrangement** and **simplification** of **individual fields** to make them more useful for DW.

- You may use **one or more fields** from the same input record to create **a better view** of the data for the DW.

4.1.5 Format Revisions

- **Revisions** include **changes to the data types and lengths** of individual fields.
  - **Product package types** may be **indicated** by **codes** and **names** in which the fields are **numeric** and **text data types**.
  - Again, the **lengths of the package types** may **vary** among the different source systems. It is wise to **standardize** and **change the data type** to **text** to provide values meaningful to the users.

4.1.6 Decoding of Fields

- When you have the same data items described by a plethora of field values.
  - Gender → M, F or 1,2
  - Department→ IS, CS

- You need to **decode** all such **cryptic codes** and change these into values that make **sense to the users**..

4.1.7 Calculated and Derived Values

- **Depending on DW users and their needs.**

- Calculated and derived **column produce new data** to **enrich query and data analysis**.
  - Quarter, Month, Week Day← Date
  - Amount in Dinar← Amount in Dollar/1200IQ

- You should use **the tools** that **derive** the new data or column.

4.1.8 Splitting of Single Fields

- Earlier legacy systems stored names and addresses of customers and employees in **large text fields**.
  - Address (Basrah-Ashar-Street10).
  - Ali Sami Ali.
- You need to **store individual components** of names and addresses in separate fields in your DW for **two reasons**:
  - Improve the operating **performance** by **indexing** on **individual components**.
  - Your users may need to **perform analysis** by using **individual components** such as city, state, and zip code.

4.1.9 Merging of Information

- This is **not the opposite** of splitting of single fields.
- Merging Info**. does not** literally mean the **merging of several fields to create a single field of data**.
  - For example, information about a product may come from different data sources. The product code and description may come from one data source. The relevant package types may be found in another data source. The cost data may be from yet another source. In this case, merging of information denotes the combination of the product code, description, package types, and cost into a single entity.

4.1.10 Character set conversion

- Refers to conversion of character sets to an **agreed standard character set** for textual data in the DW.
  - If you have **mainframe legacy systems** as source systems, the source data from these systems will be in **EBCDIC characters**.
  - If PC-based architecture is the choice for your DW, then you must convert the mainframe **EBCDIC format to the ASCII format**.

4.1.11 Conv. of Units of Measurements

- Many companies today have **global branches**.
- Measurements in many European countries are in **metric units.**
- If your company has overseas operations, you may have to convert the metrics so that the numbers are all in **one standard unit of measurement**.

4.1.12 Date/Time Conversion

- This type relates to **representation of date and time in standard formats**.
  - For example, the American and the British date formats may be standardized to an international format.
  - The date of October 11, 2008 is written as 10/11/2008 in the U.S. format and as 11/10/2008 in the British format. This date may be standardized to be written as 11 OCT 2008.

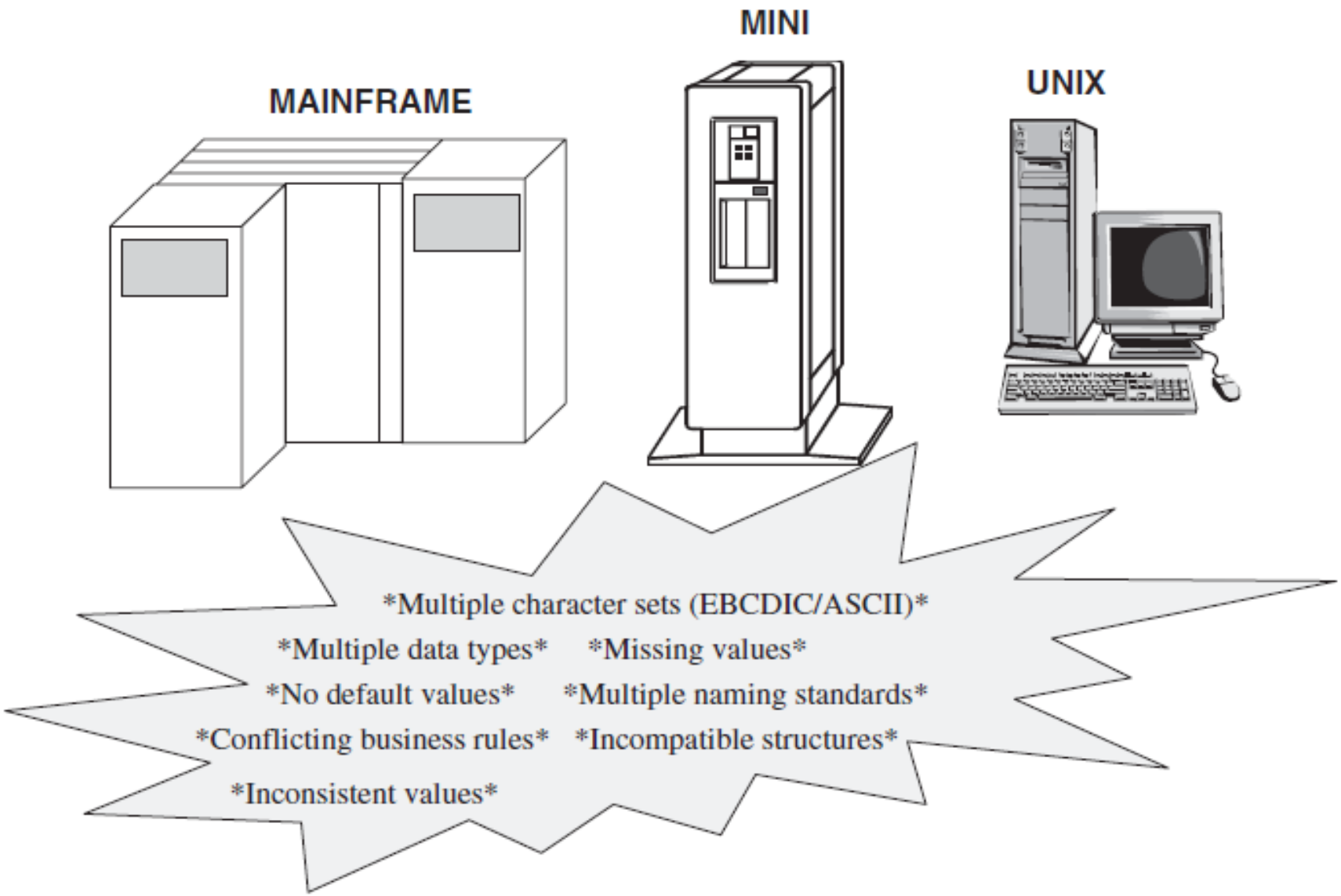
4.1.13 Key Restructuring

- If the product is moved to another DW, the DW part of the product key will have to be changed.
  - Solution is **surrogate key**.
- When choosing keys for your DW database tables, avoid such keys with **built-in meanings**.
- The **key restructuring is** transform such keys into **generic keys** generated by the system itself.

4.1.14 Deduplication

- In many companies, the customer files have several records for the same customer.
  - Mostly, the duplicates are the result of creating **additional records by mistake**.
- In your DW, you want to keep a **single** record for one customer and link all the **duplicates** in the source systems to this single record.
  - This process is called **deduplication** of the customer file.
  - Employee files and, sometimes, product master files have this kind of duplication problem.

4.2 Data Integration and Consolidation



4.2 Data Integration and Consolidation

- A **challenge** is the pulling together of all the **source data from many disparate**, dissimilar source systems.
- **Data Integration problems** occur when many data sources from different platforms used to implement DW.
- Sources do not conform the same set of **business rules** (**naming conventions** and varied **standards** for **data representation**)
- Two problems affect the data integration (**Entity Identification and Multiple Sources**).

4.2.1 Entity Identification Problem

- You have to design **complex algorithms** to **match records** from all the **source files** and form groups of matching records.

- **No matching algorithm** can **completely** determine the groups.

- If the matching criteria are **too tight**, then **some records will escape the groups**.

4.2.2 Multiple Sources Problem

- **Less common** and **less complex** than the entity identification problem.

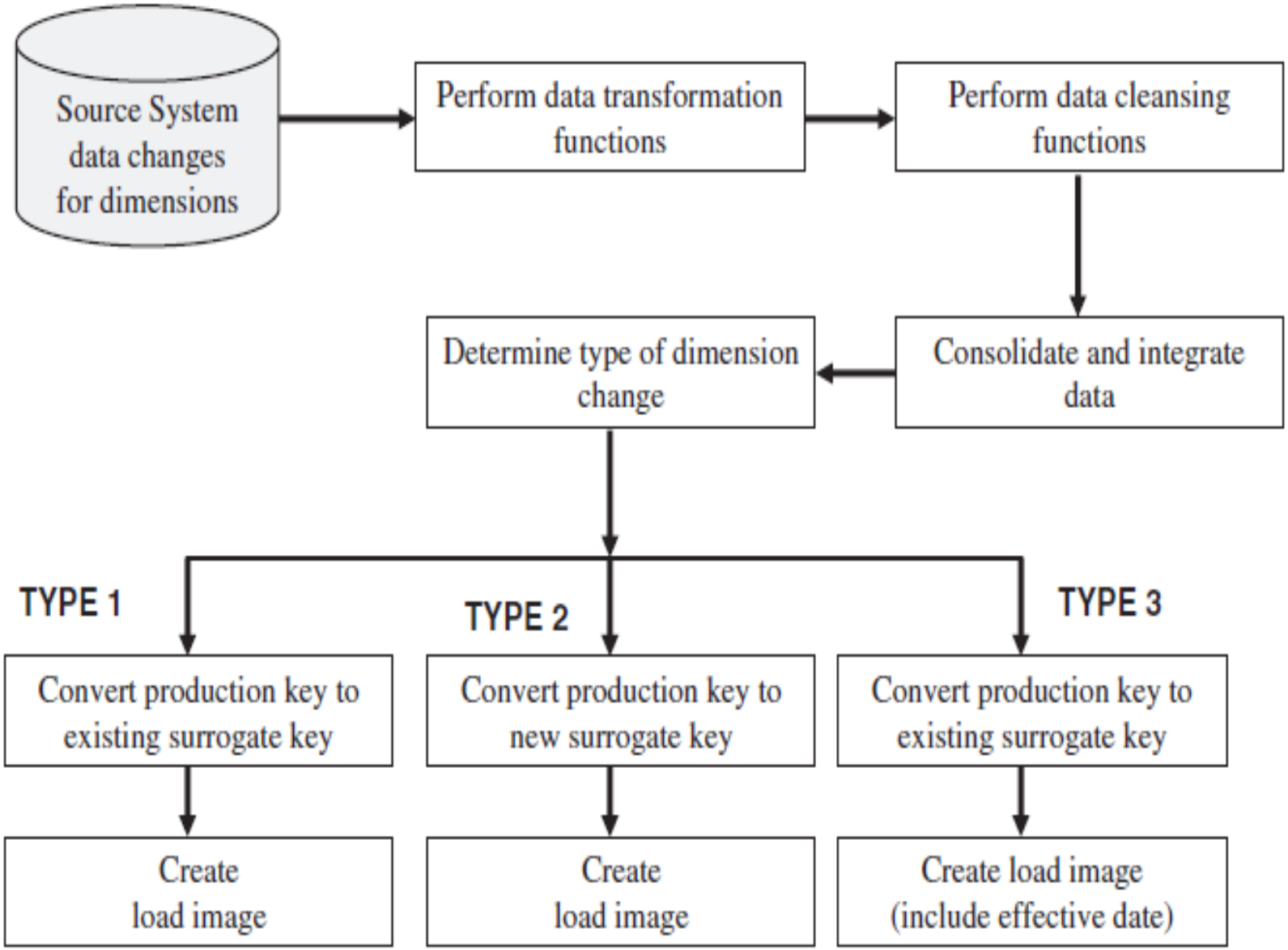
- This problem results from a **single data element having more than one source**.

- A straightforward solution is to **assign a higher priority to one** of the **two sources** and **pick up the other sources**.

4.3 Transformation for Dimension Attributs

- Three ways to handle the three types of **slowly changing dimensions**.
  - **Type 1 changes** are **corrections of errors**. These changes are applied to the DW **without any need to preserve history**.
  - **Type 2 changes** **preserve** the **history** in the DW.
  - **Type 3 changes** are tentative changes where your users need the ability to analyze the metrics in both ways with the changes and without the changes.

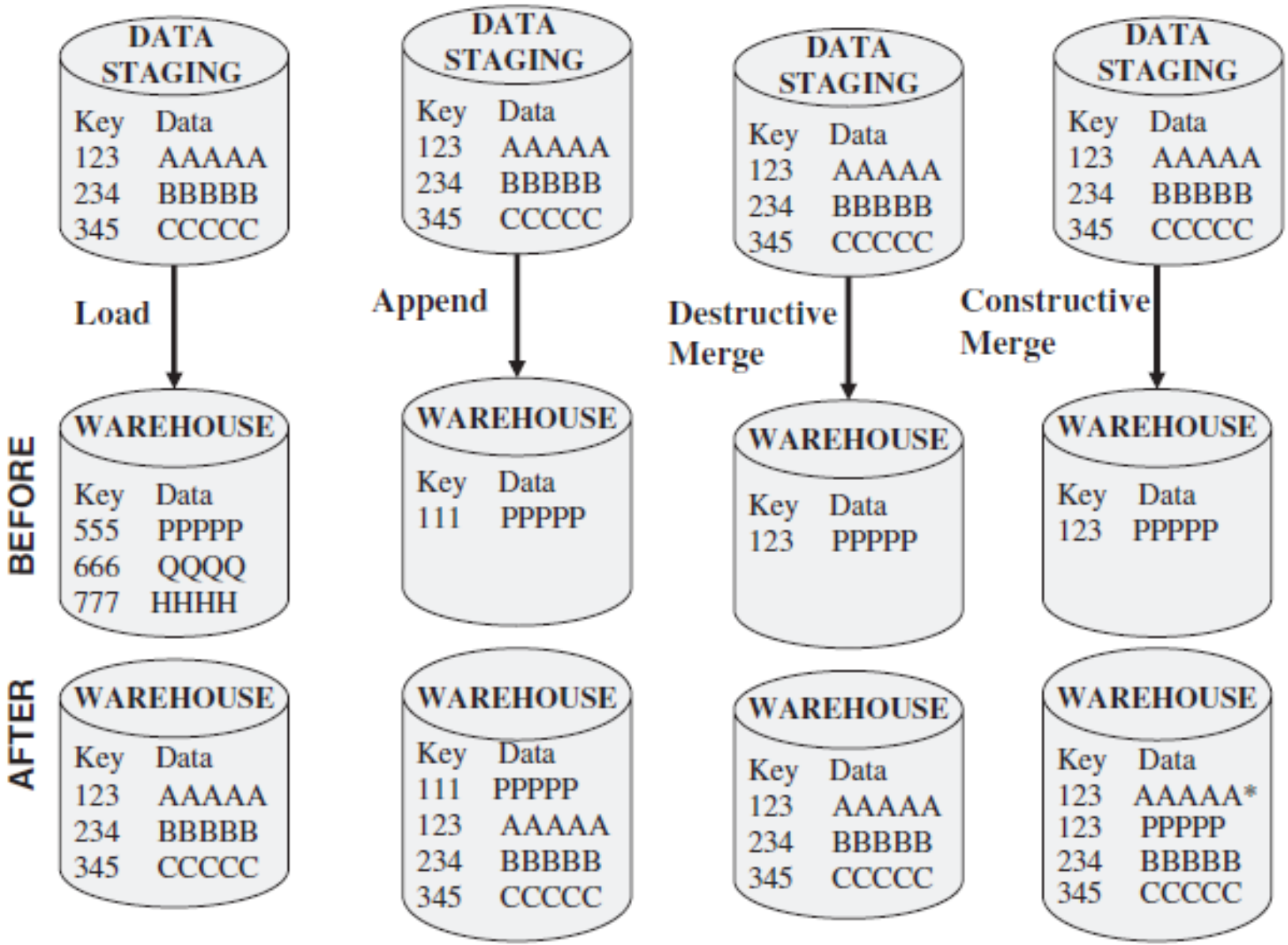
4.3 Transformation for Dimension Attributs



5. Data Loading

- The **whole process of moving data into the DW repository** is referred to in several ways (**applying the data, loading the data, and refreshing the data**).
- During the loads, the **DW** has to be **offline**.
- You need to find a **window of time** when the **loads** may be **scheduled** without **affecting your DW users**.
- Data may be applied in the following **four different modes (Load, Append, Destructive merge, Constructive merge).**

5.1 Load Modes



5.1.1 Load

- If the **target table** to be loaded **already exists** and **data exists** in the **table**, the load process **wipes out the existing data** and **applies the data** from the incoming file.

- If the **table** is already **empty** before loading, the load process **simply applies the data** from the incoming file.

5.1.2 Append

- If **data already exists** in the table, the append process **unconditionally adds the incoming data, preserving the existing data** in the target table.
- When an incoming record is a duplicate of an already existing record ,you may **define how to handle an incoming duplicate**.
- The incoming record may be allowed to be added as a **duplicate** or may be **rejected** during the append process.

5.1.3 Destructive Merge

- Merge In this mode, you apply the incoming data to the target data.
- If the **primary key** of an **incoming record matches** with the **key of an existing record**, **update** the **matching** target record.
- If the **incoming record** is a **new record without** a **match** with **any existing record**, **add** the incoming record to the target table.

5.1.4 Constructive Merge

- This mode is slightly **different** from the **destructive merge**.
- If the **primary key** of **an incoming record matches** with the **key of an existing record**, **leave** the **existing record**, **add** the **incoming record**, and **mark the added record** as superseding the old record.

5.2 Load Stages

- Load stages refer to **what you intend to do** with data once it's loaded to DW.
- **Load stages include:**
  - **Initial Load**
  - **Incremental Load**
  - **Full Refresh**

5.2.1 Initial Load

- **Bring** the **data** from **staging area** into **DW**.
- **Takes time** and **memory intensive** due to **size** of data.
- Must **divide the load process**, **schedule** the **loads** and **use a specific mode**.

5.2.2 Incremental Load

- **Detect and load** the **ongoing changes** from the source systems.
- **Changes** to the source systems are always **tied to specific times**, irrespective of whether or not they are based on **explicit time stamps** in the source systems (need method).
- If the time stamp is also part of the primary key or if the time stamp is included in the comparison between the incoming and the existing records, then **constructive merge** may be used to preserve the periodic nature of the changes.

5.2.3 Full Refresh

- Involves **periodically rewriting** **the entire DW**.
  - Sometimes, you may also do **partial refreshes** to **rewrite** only **specific** tables.
- **Partial refreshes** are **rare** because every **dimension** table is intricately **tied** to the **fact table**.
- In the case of full refreshes, **data exists in the target** tables before incoming data is applied.
- The existing data must be **erased** before applying the **incoming data**.
  - Just as in the case of the **initial load**, the **load** and **append modes** are applicable to **full refresh**.

5.3 Update VS Refresh

- **Update** application of **incremental changes** in the data sources.
- **Refresh** **complete reload** **at specified intervals.**
- **Refresh** is a **much simpler** option than **update**.
- To use the **update** option, you have to **devise** the **proper strategy** to extract the **changes** from each data source.
- Then you have to determine the **best strategy** to apply the changes to the DW.

5.3 Update VS Refresh

- The refresh option simply involves the **periodic replacement** of complete DW tables.
- But **refresh** jobs can take a **long time to run**.
- If you have to run refresh jobs every day, you may have to keep the DW down for **unacceptably long times**.
- The case **worsens** if your **database has large tables**.

5.4 Dimension and Fact Table Load

- The key of the fact table is the **concatenation** of the **keys of the dimension tables**.
  - Therefore, for this reason, **dimension records are loaded first**.
- You have to create the **concatenated key** for the **fact table record** from the **keys** of the **corresponding dimension records**.
- Perform fact table **surrogate key look-up**.

End of ETL-2