NATIONAL ENGINEERING CENTER
University of the Philippines
Diliman, Quezon City

7.0 Transformation and Loading Methodologies

Eugene Rex L. Jalao, Ph.D.

Associate Professor

Department Industrial Engineering and Operations Research

University of the Philippines Diliman

Module 2 of the Business Intelligence and Analytics Track of UP NEC and the UP Center of Business Intelligence

Outline for This Training

Outline for This Session

- Transformation Process
- Dimension Table Processing
- Dimension Update Types
- Fact Table Processing
- Loading Process
- Case Study

Data Transformation

- Extracted data is raw data and it cannot be applied to the data warehouse
- All the extracted data must be made usable in the data warehouse.

Quality of data

- Major effort within data transformation is the improvement of data quality.
- This includes filling in the missing values for attributes in the extracted data.
- Data quality is of paramount importance in the data warehouse because the effect of strategic decisions based on incorrect information can be devastating.

Basic Tasks in Data Transformation

- Selection
  - Select either whole records or parts of several records from the source systems.
- Splitting/joining –
  - Sometimes (uncommonly), you will be splitting the selected parts even further during data transformation.
  - Joining of parts selected from many source systems is more widespread in the data warehouse environment.

Basic Tasks in Data Transformation

- Conversion
  - It includes a large variety of rudimentary conversions of single fields for two primary reasons
    - one to standardize among the data extractions from disparate source systems,
    - To make the fields usable and understandable to the users.
- Summarization.
  - Sometimes it is not feasible to keep data at the lowest level of detail in the data warehouse.
  - It may be that none of the users ever need data at the lowest granularity for analysis or querying.

Basic Tasks in Data Transformation

- Enrichment
  - Rearrangement and simplification of individual fields to make them more useful for the data warehouse environment.
  - You may use one or more fields from the same input record to create a better view of the data for the data warehouse.
  - This principle is extended when one or more fields originate from multiple records, resulting in a single field for the data warehouse.

Major Transformation Types

- Format Revisions
- Decoding of Fields
- Calculated and Derived Values
- Splitting of Single Fields
- Merging of Information
- Character Set Conversion
- Conversion of Units of Measurements
- Date/Time Conversion
- Summarization
- Key Restructuring
- Deduplication

Outline for This Session

- Transformation Process
- **Dimension Table Processing**
- Dimension Update Types
- Fact Table Processing
- Loading Process
- Case Study

ETL Intermediate Dimension Tables

- Extract Tables (S, X)
- Master Tables (M)
- Change and Error Tables (C, E)
- Transform Tables (T)
- Insert Tables (I)
- Update Tables (U)
- Dimension Tables (D, H)

Dimension and Intermediate Table Naming Conventions

- S_Customers
- M_Customers
- X_Customers
- C_Customers
- E_Customers
- T_Customer
- I_Customer
- U_Customer
- D_Customer
- H_Customer_Audit_History

Full Extract Table (S)

- Allowable and non-allowable nulls accepted
- Source Logical Primary Key disabled
- Unique description and name constraints disabled
- Contains data directly extracted from source system
- Matched against master to detect changes
- Non-enforced constraints are enabled in change table

Extract Master Table (M)

- Used to detect changes if it is a complete rather than incremental extract
- Used to verify additions if it is an incremental extract

Incremental Extract Table (X)

- Contains data directly extracted from source system
- Allowable nulls changed to 'Missing Optional'
- Logical Primary Key enabled
- Unique description and name constraints enabled
- Non-allowable nulls either rejected or flagged
- As many constraints as possible should be applied here
  - Entity integrity – logical primary key
  - Missing required data – not null or blanks
- This is the place to catch as many errors as early as possible
  - Makes the rest of the processing cleaner
  - Minimizes work needed to be done

Change Table (C)

- Used to hold new and changed records from the match process in the case of full extracts
- Not used for incremental extracts

Transform Table (T)

- Holds new and changed dimension rows after transforms have been applied
- Has the structure of the dimension table where this data will eventually reside
- Up to this point all the intermediate tables had the structure of the source system files/tables

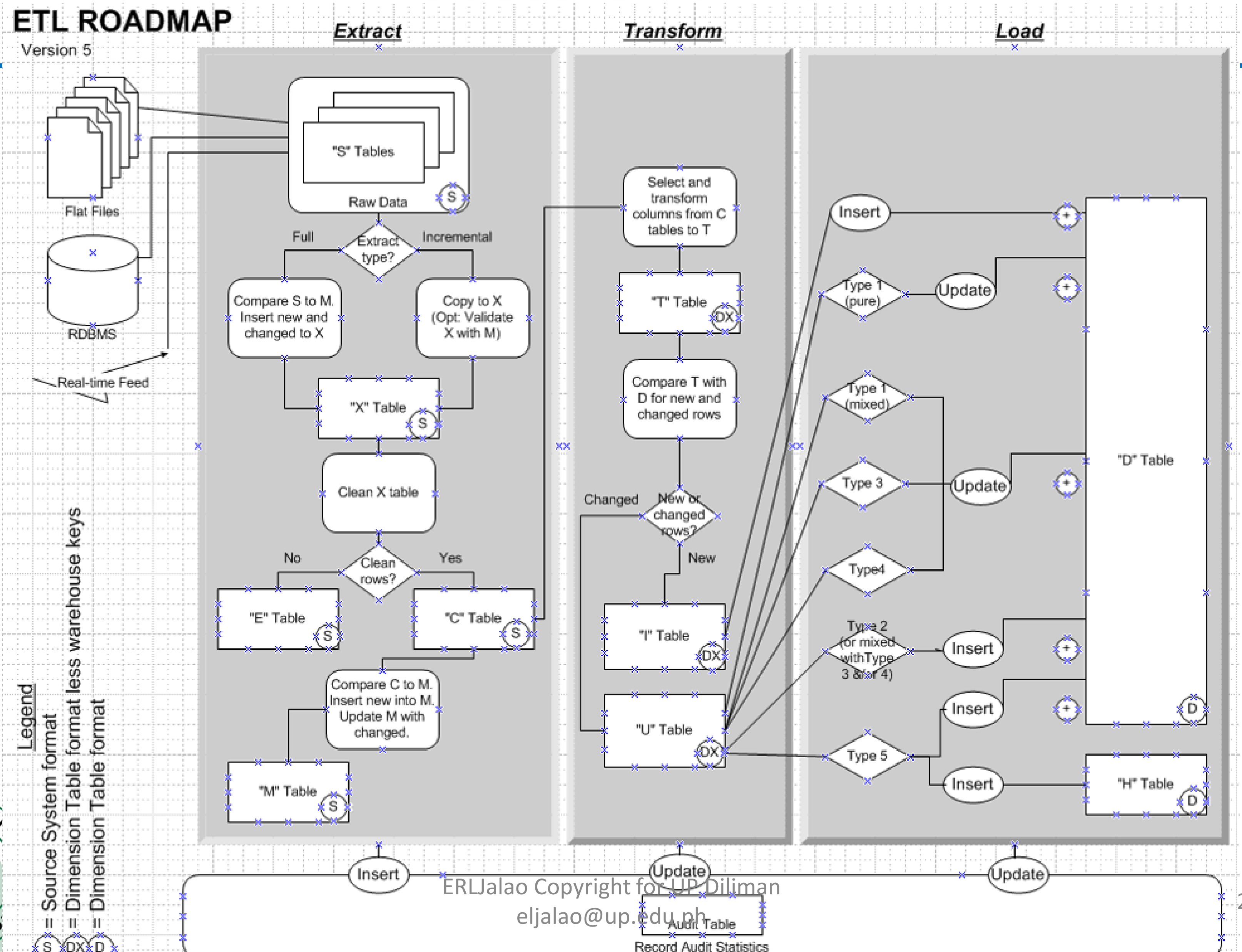
Transform Tables

- Insert Table (I)
  - Holds new dimension rows before being added to dimension
- Update table (U)
  - Holds changed dimension rows before being added to dimension

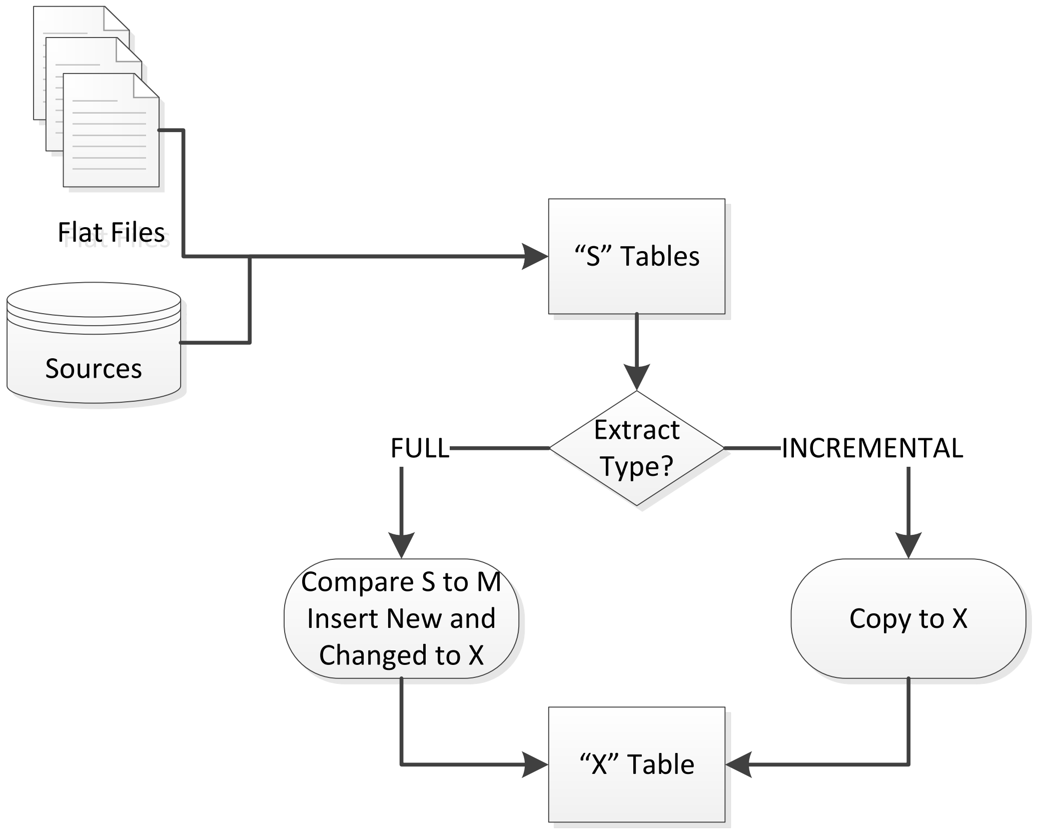
Dimension Audit History Table (H)

- Holds complete history of all detail and analytic attributes of a dimension
- Used if the dimension needs to be analyzed for auditing purposes
- Also known as a dimension history table
- Should only be used if need to use the dimension history table as a fact table
- Not joined to a fact table

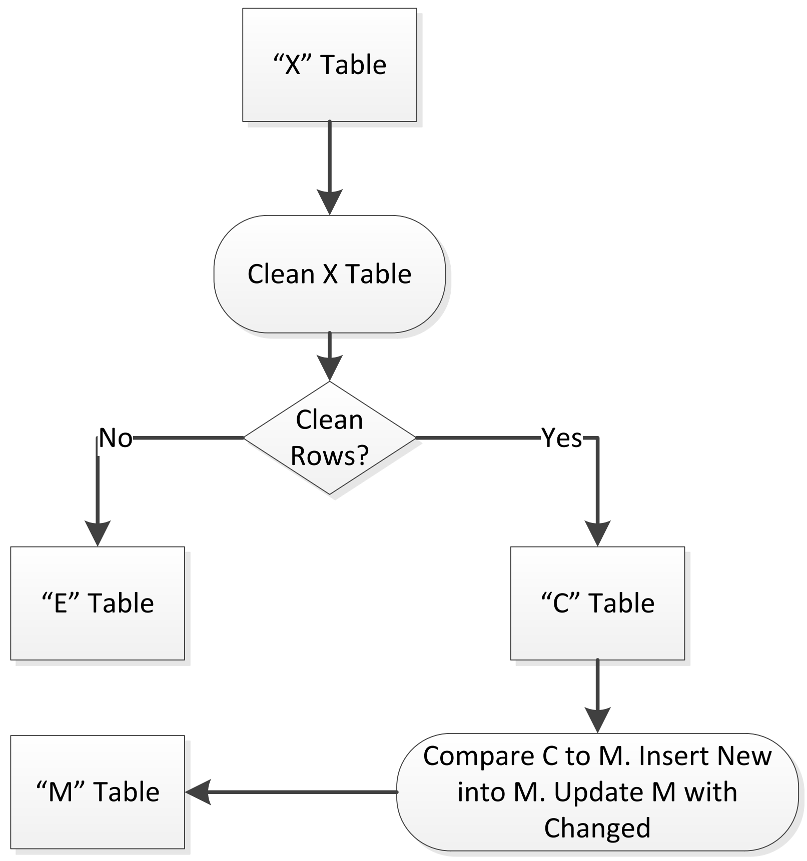
Process Overview



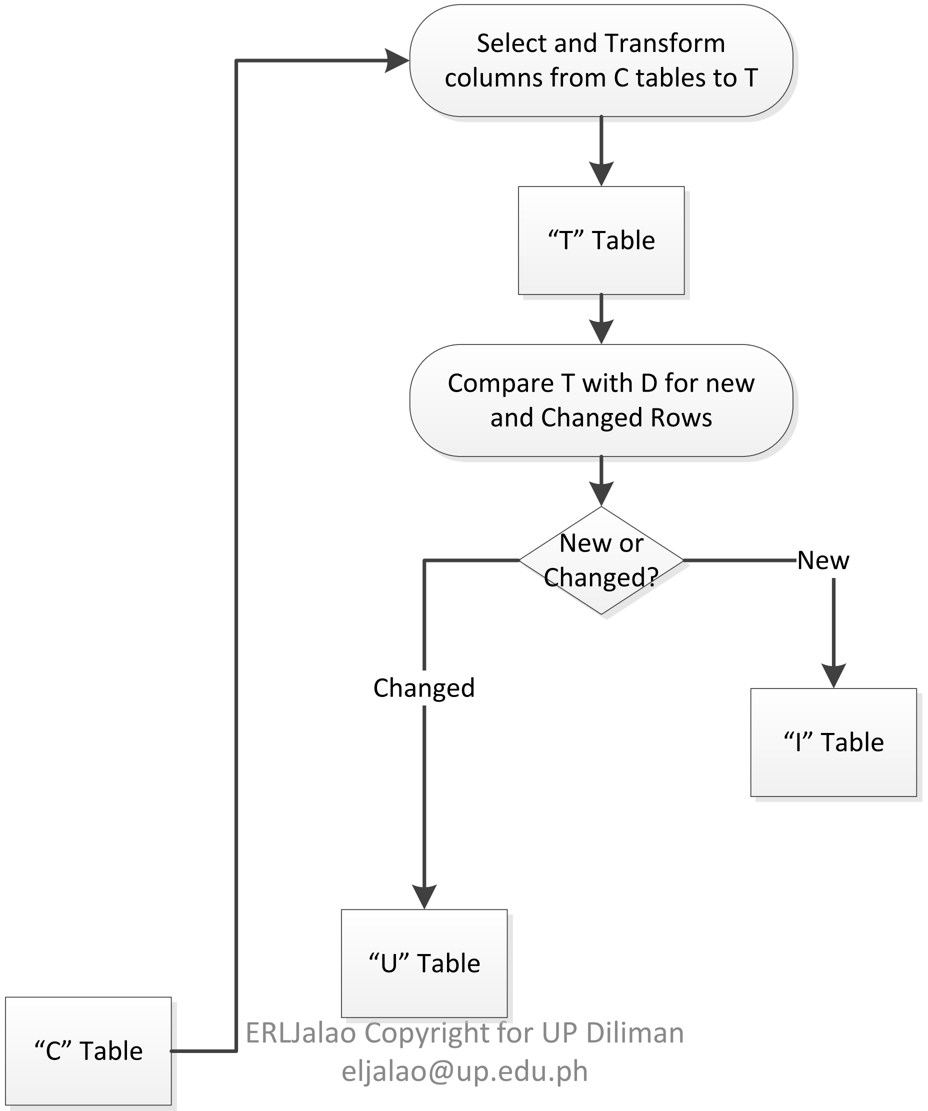
Step 1 – Source to Extract Table



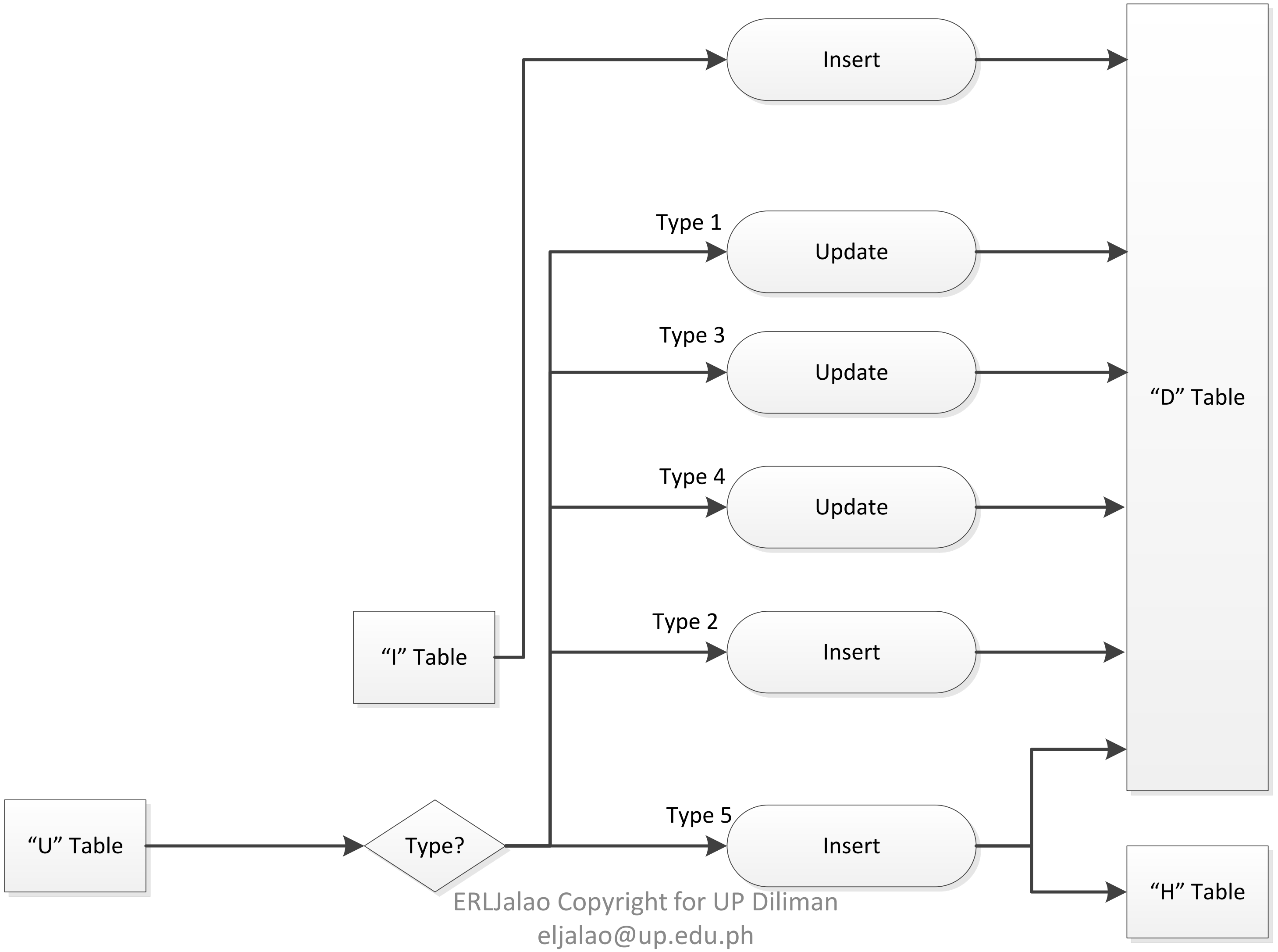
Step 2 – Find or Verify New & Changed



Step 3 – Insert into Respective Intermediate Table


Select and Transform columns from C tables to T
"T" Table
Compare T with D for new and Changed Rows
New or Changed?
New
Changed
"I" Table
"U" Table
"C" Table

Step 4 and 5


Insert
Type 1
Update
Type 3
Update
Type 4
Update
Type 2
Insert
Type 5
Insert
"D" Table
"I" Table
"U" Table
Type?
"H" Table

Full Extract Processing up to this point

- Load Source Table
- Minus (take set difference between) Source Table and Master Table to detect new and changed
- Insert clean rows into Change table, dirty rows into error table
- Transform columns from Change Tables to Transform tables
- Outer Join to dimension table to separate new and changed
- Insert new into Insert Table
- Insert changed into Update Table
- Insert into Data Warehouse

Incremental Extract Processing up to this point

- Load Extract Table
- Minus extract and master to ‘ensure’ only new and changed rows
- Insert clean rows into Change Table, dirty rows into error table
- Transform Columns from Change Tables to Transform Tables
- Outer Join to dimension table to separate new and changed
- Insert new into Insert Table
- Insert changed into Update Table

Outline for This Session

- Transformation Process
- Dimension Table Processing
- **Dimension Update Types**
- Fact Table Processing
- Loading Process
- Case Study

New Dimension Row Processing

- Apply transformations
- Generate new key
- Bulk load into dimension
- Insert into Dimension Audit History

Dimension Update Processing

- Apply transformations
- If a dimension table contains all type 1 columns then
- Simply – apply the updates
- In order
  - Use type 1 changed values
  - Handle type 3 changes
  - Handle type 4 changes
  - Handle type 2 changes

Dimension Attribute History

- Introduction to Dimension Attribute History
- Dimension Attribute History Type 1
- Dimension Attribute History Type 2
- Dimension Attribute History Type 4
- Dimension Attribute History Type 3
- Dimension Attribute History Type 5

Tracking Dimension Table Attribute History

- Five Considerations
- Five Methods
  - Ignore History
    - Overwrite
  - Track History
    - New Row
    - New Column
    - New Dimension
    - Dimension History Fact Table
- Many Combinations

Six Basic Considerations

- The reason the change occurred, is it a correction to a wrong value?
- Dimension table size
- The number of changes that need tracking
- The frequency of the changes
- Is the attribute analytical (not unique) or detail (unique)?
- Is the requirement analytical or audit based?

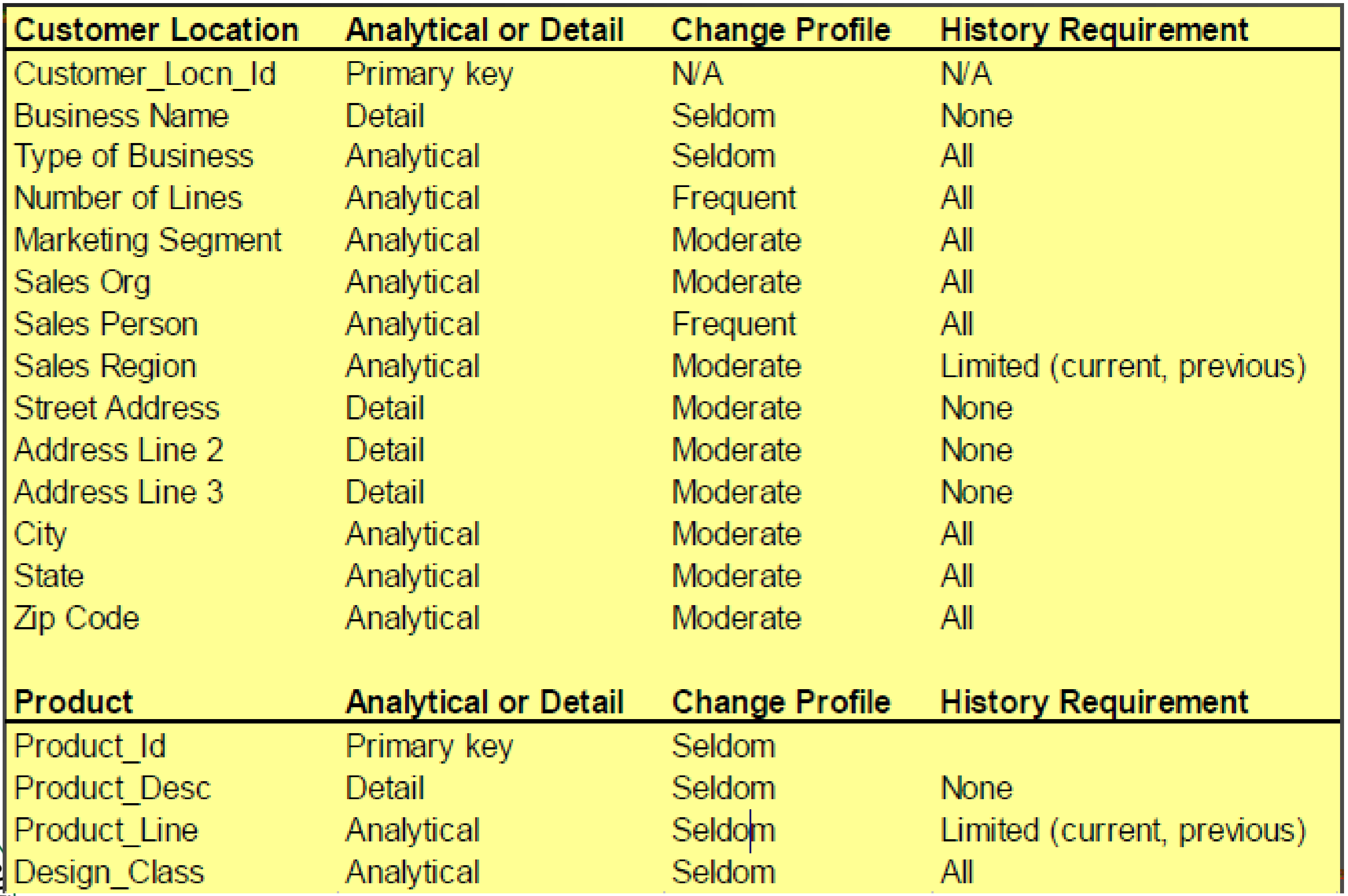
Dimension Attributes

- Change
  - Slowly
  - Quickly
  - In mass
- Lack of history can cause
  - Incorrect analyses
  - Erroneous decisions
- Tracking history should not affect
  - Analytical performance
  - Ease of use

Formal Methodology

- Based on Business Requirements
- Documented
- Signed off by Business Users
- Administered by Data Authority/Steward

Assess Requirements and Change Profile for Each Dimension Attribute



Customer Location	Analytical or Detail	Change Profile	History Requirement
Customer_Locn_Id	Primary key	N/A	N/A
Business Name	Detail	Seldom	None
Type of Business	Analytical	Seldom	All
Number of Lines	Analytical	Frequent	All
Marketing Segment	Analytical	Moderate	All
Sales Org	Analytical	Moderate	All
Sales Person	Analytical	Frequent	All
Sales Region	Analytical	Moderate	Limited (current, previous)
Street Address	Detail	Moderate	None
Address Line 2	Detail	Moderate	None
Address Line 3	Detail	Moderate	None
City	Analytical	Moderate	All
State	Analytical	Moderate	All
Zip Code	Analytical	Moderate	All

Product	Analytical or Detail	Change Profile	History Requirement
Product_Id	Primary key	Seldom	
Product_Desc	Detail	Seldom	None
Product_Line	Analytical	Seldom	Limited (current, previous)
Design_Class	Analytical	Seldom	All

Tracking Methods

- Ignore History
  - Type 1 - overwrite value
- Track History
  - Type 2 - create new row with updated values
  - Type 3 - use columns to store limited history
  - Type 4 - use analytical dimensions
  - Type 5 - use dimension history fact table

No History Required

- Type 1 - Overwrite old value with new value
- Always used for corrections
- When to use it?
  - If attribute history is not of analytical or historical importance
  - If the update is a correction

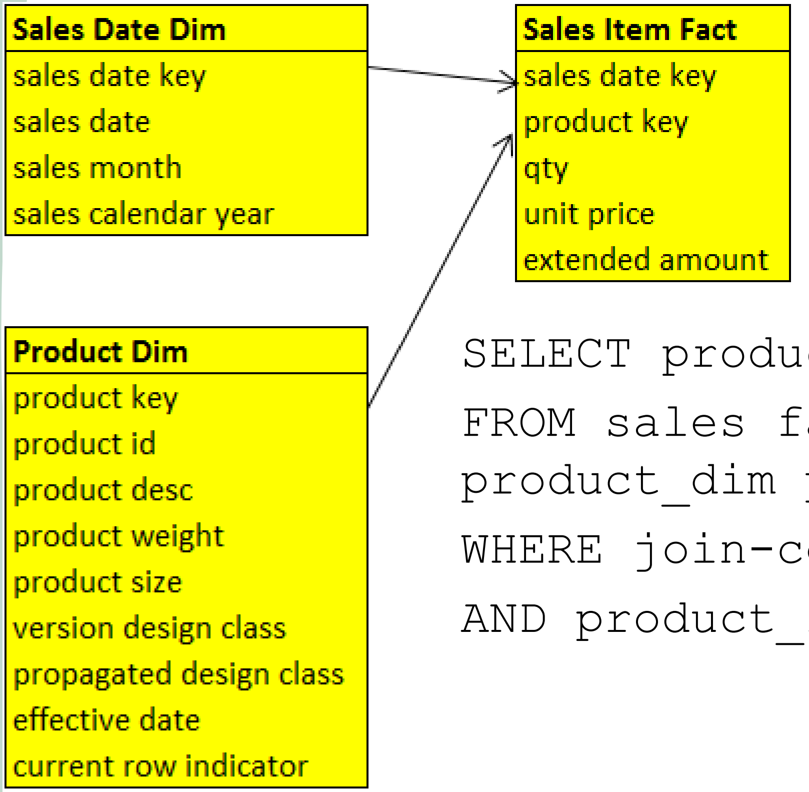
Type 1 - Example

- `UPDATE customer_dim SET street_address TO '2323 Jennings St.'`
- `UPDATE customer_dim SET phone_number TO '258-1913'`

All Changes Need To Be Tracked

- All changes are tracked
- Conditions:
  - The dimension is not too large
  - The changes are not too frequent
- Perfectly partitions history
- If one or more attributes are classified as Type 2, we classify the dimension as a Type 2 dimension.
- Does not mean that all attributes must be Type 2
- Add a new row to the dimension each time one or more tracked attributes change

Type 2 - Create Another Dimension Row with New Warehouse Key



Sales Fact sample data				
sales date key	product key	quantity	unit price	extended amount
23	10	25	5	125
73	1028	15	4	60

```
SELECT product_name, sum(extended_amount)
FROM sales fact sf, sales_date_dim sd,
product_dim pd
WHERE join-conditions
AND product_id = ‘1256238'
```

Result:

product_name	sum
turbo xxx	**185**

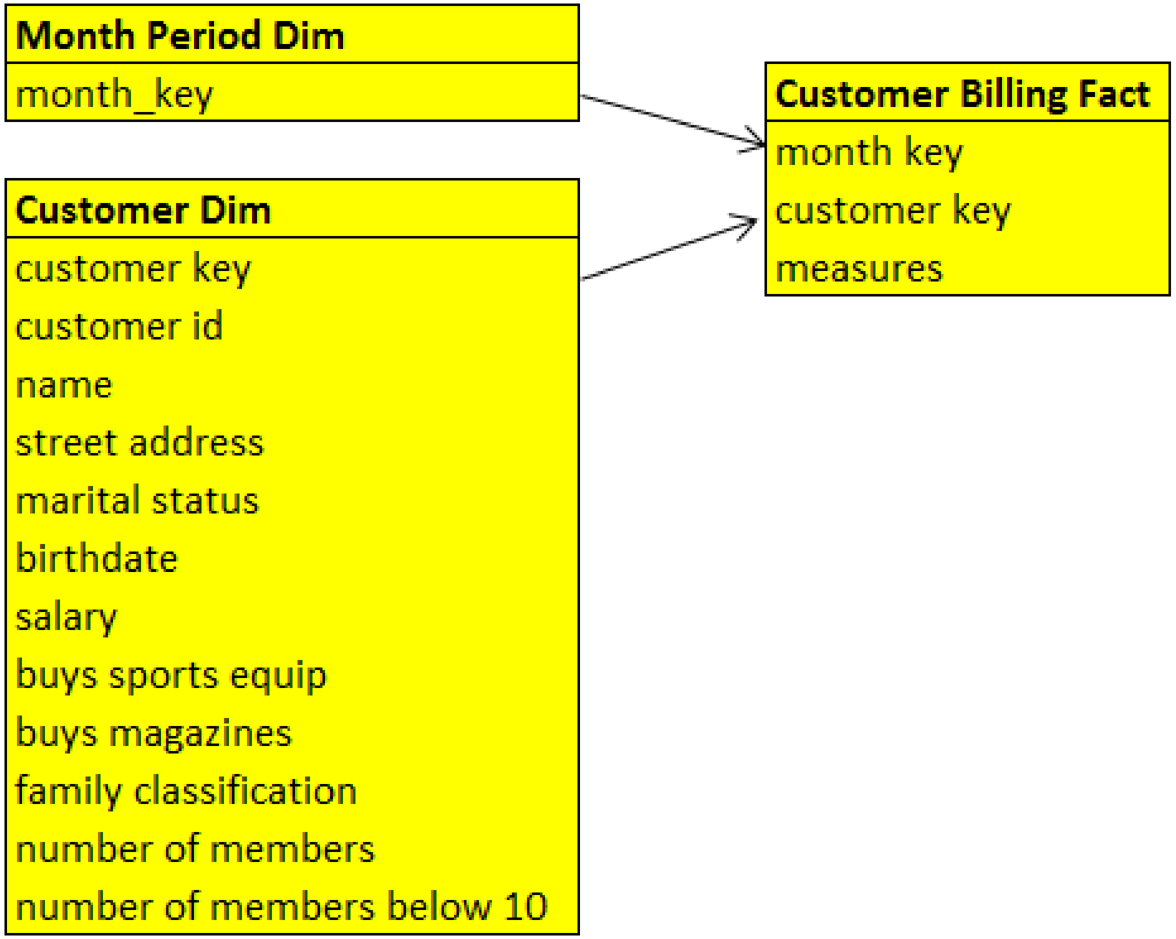
Product Dim Sample Data

product key	product id	product desc	product weight	product size	version design class	propagated design class	effective date	current row indicator
10	1256238	turbo xxx	20	6	A	M	12/1/1990	N
1028	1256238	turbo xxx	30	6	M	M	4/10/2000	Y

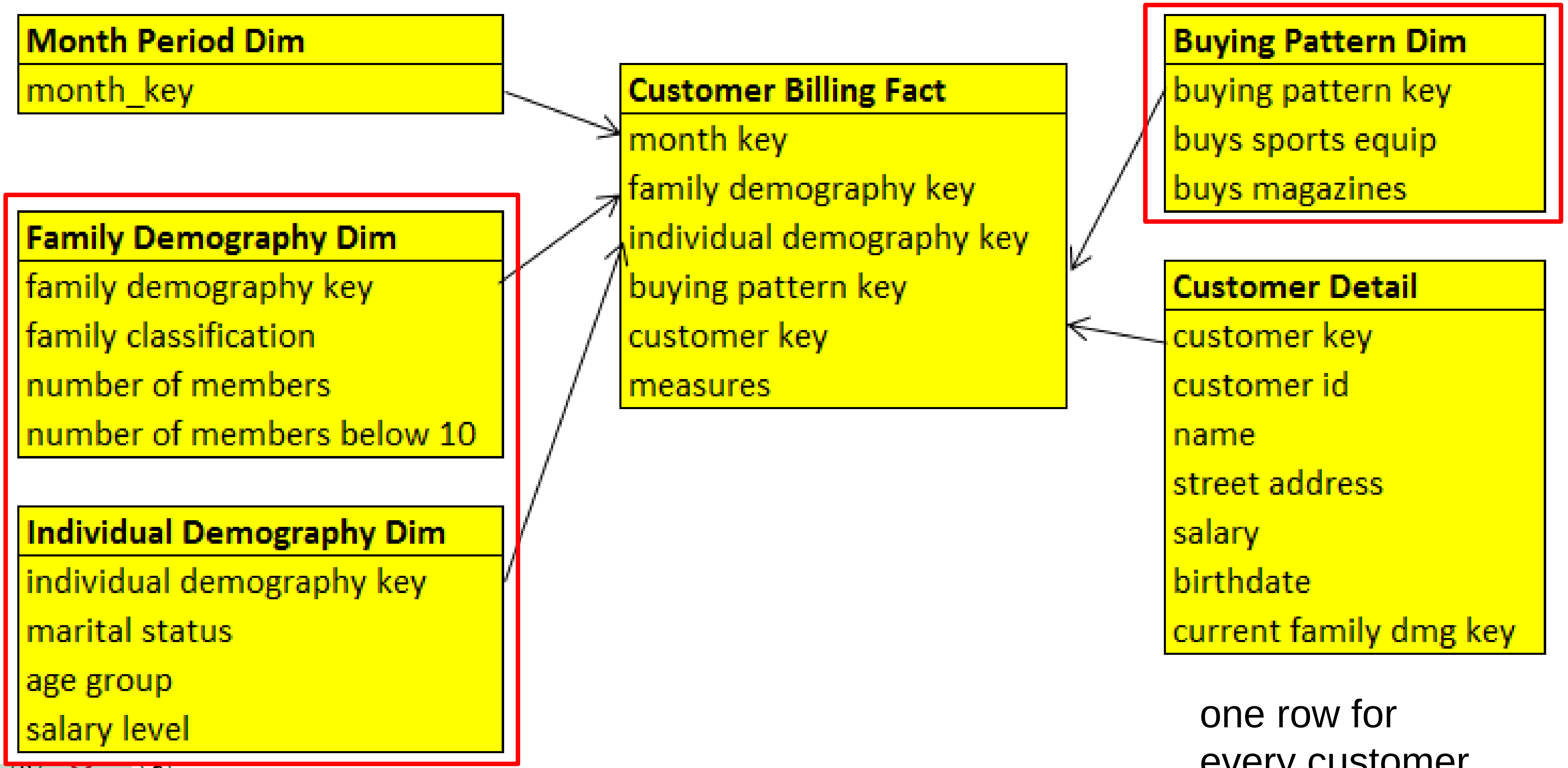
All Changes Need to be Tracked

- Frequent Changes
- Large table (upwards of 100,000 rows or more, rather than a smaller table)
- Use Type 4: Analytical Dimensions
- Other Names:
  - Profile dimensions
  - Mini-dimensions
  - Sub-dimensions

Type 4 Example
Customer Dimension



Type 4 - Create Analytical Dimensions



one row for
every customer

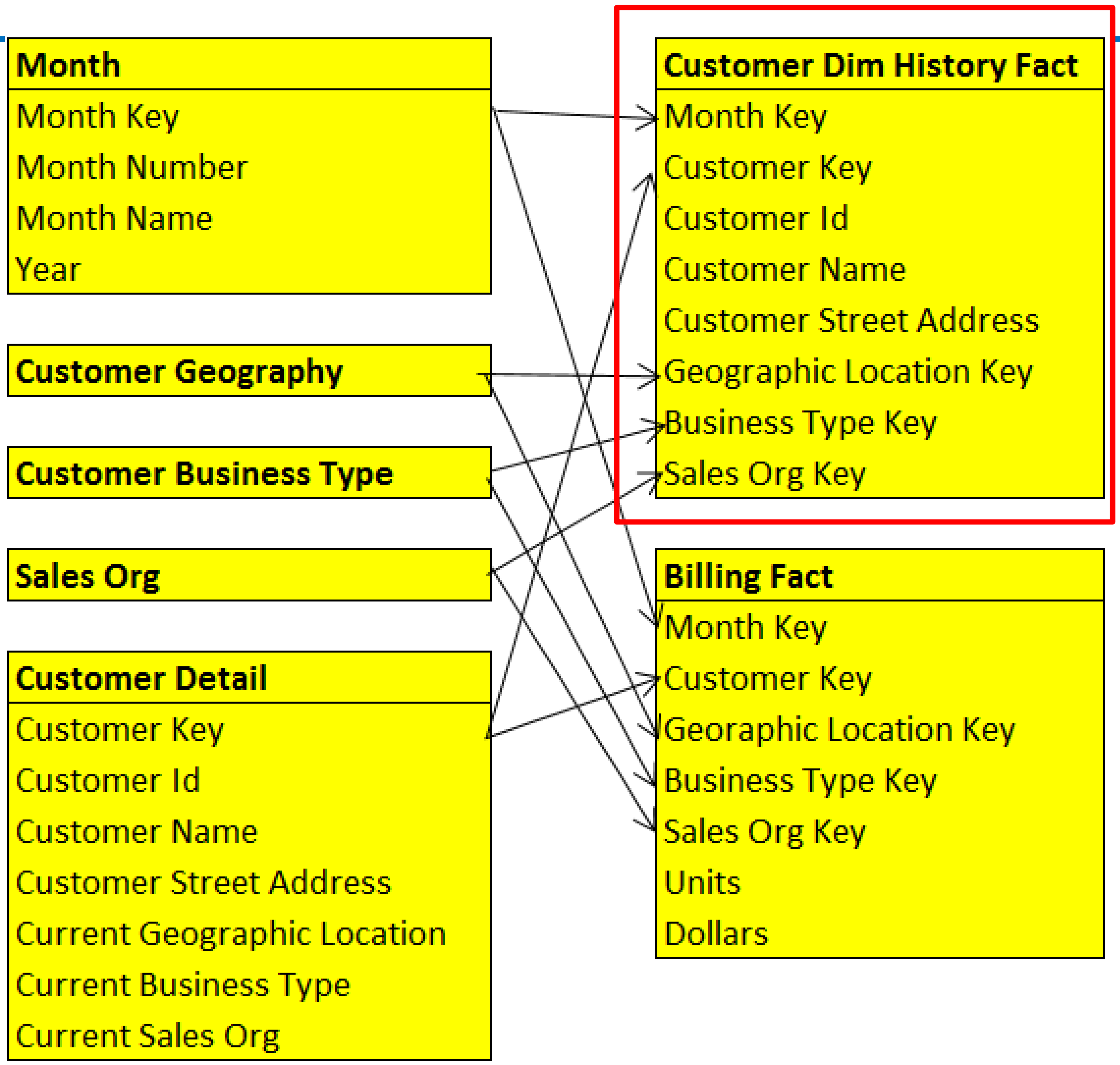
Automatic Fact Table History

- The fact table has keys to the customer attributes which were in effect when the fact table row was produced
- Dimension history fact table can be used to analyze dimension only history of all dimension attribute changes

Detail Attribute History and Dimension Change Tracking

- Need street address history
- Use Type 5 – Dimension History Fact Table
- It also supplements Type 4 Analytical Dimensions

Type 5 Dimension History Fact Table



Type 5 Summary

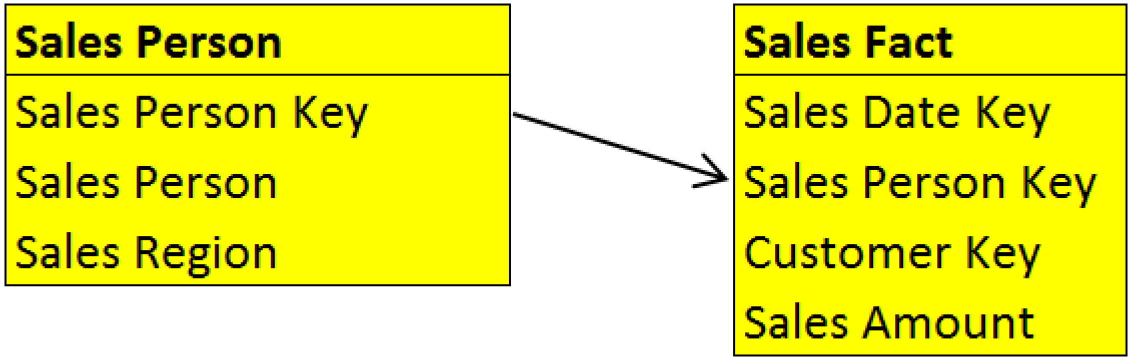
- Traps detail attribute history
- Should only be used to analyze dimension-only history
- Shouldn’t be used alone
- Should not be used to query transaction or event history - use other types for that

Only a Limited Number of Changes Need Tracked

- Current versus Last
- Current and Planned
- Type 3 Examples
  - Analysis on new alignment versus old alignment
  - Analysis on planned realignment
- Only a limited number of changes need tracked
  - Not elegant
  - But very effective and high performing

Example

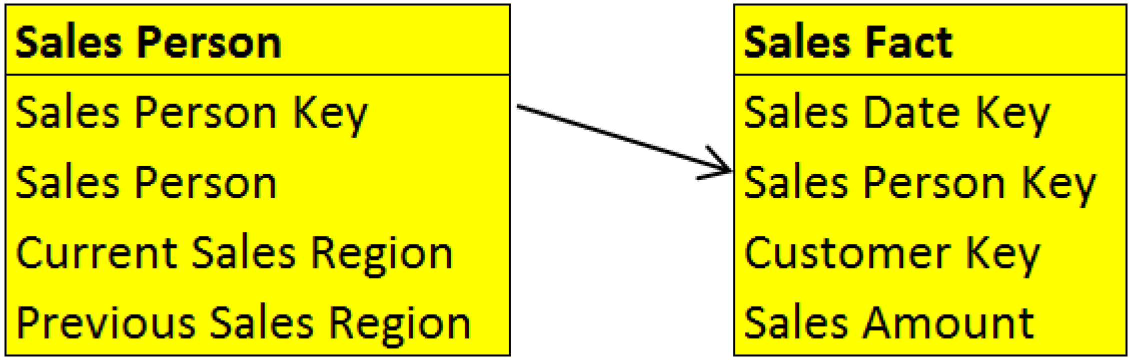
- Original Tables



Sales Person
Sales Person Key
Sales Person
Sales Region

Sales Fact
Sales Date Key
Sales Person Key
Customer Key
Sales Amount

- New Sales Person table with added column



Sales Person
Sales Person Key
Sales Person
Current Sales Region
Previous Sales Region

Sales Fact
Sales Date Key
Sales Person Key
Customer Key
Sales Amount

Sample Queries

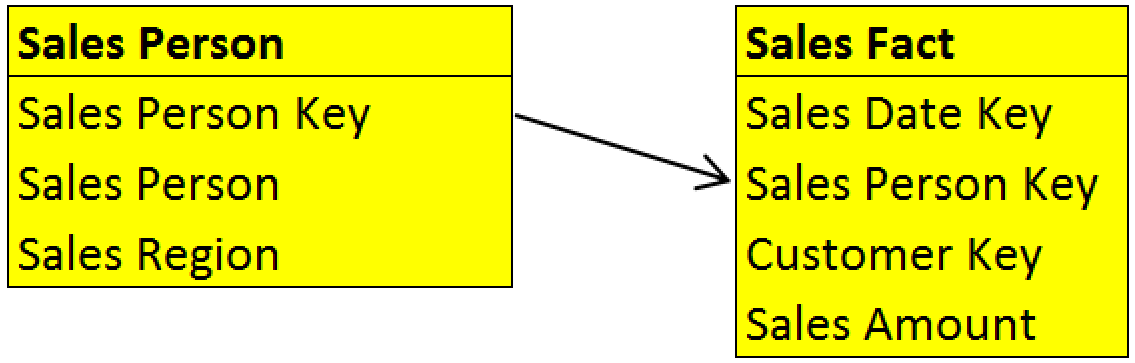
- Show me last month sales totals by new regions

  -
    ```
    SELECT current_sales_region,
    SUM(sales_amount) FROM sales_fact
    WHERE month = 'LAST'
    ```
- Show me last month sales totals by the old regions

  -
    ```
    SELECT previous_sales_region,
    SUM(sales_amount) FROM sales_fact
    WHERE month = 'LAST'
    ```

Example

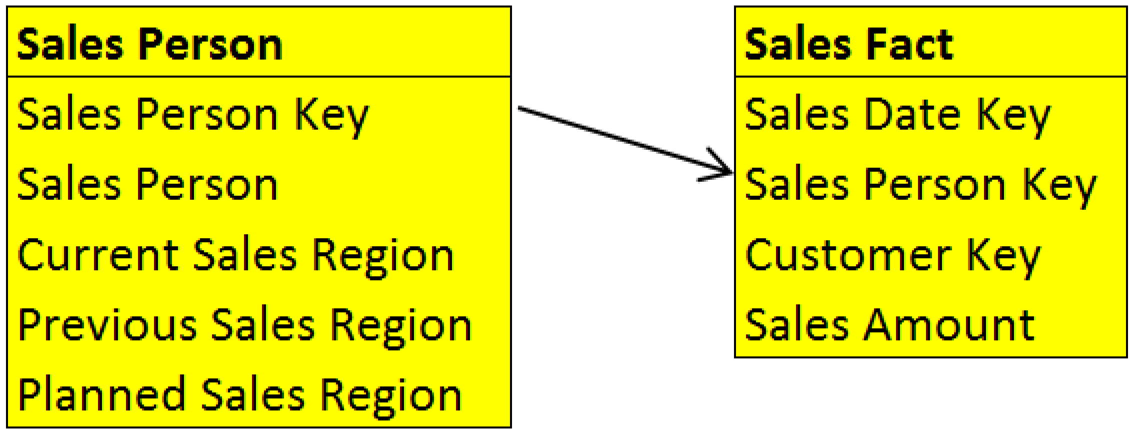
- Original Tables



Sales Person
Sales Person Key
Sales Person
Sales Region

Sales Fact
Sales Date Key
Sales Person Key
Customer Key
Sales Amount

- New Sales Person table with added column



Sales Person
Sales Person Key
Sales Person
Current Sales Region
Previous Sales Region
Planned Sales Region

Sales Fact
Sales Date Key
Sales Person Key
Customer Key
Sales Amount

Sample Queries

- Show me last month sales totals by the proposed regions
  - `SELECT planned_sales_region, SUM(sales_amount) FROM sales_fact WHERE month = ‘LAST’`

Choosing Dimension Attribute History Types

- Decision Matrix

History Requirement	History Strategy
None	Type 1
Limited	Type 3
All	Type 2,4,5

All	Analytical	small size - seldom, moderate or frequent chg	Type2
All		medium size - seldom and moderate chg	Type 2
All		large size - seldom and doesn't fit mini	Type 2
All		medium size - frequent chg	Type 4
All		large size - moderate and frequent chg	Type 4
All	Detail	all sizes	Type 5

Special Considerations

- Restating history
  - Essentially this means overriding history and replacing old values.
- Showing original or as of values
  - show both the original or ‘as of values’ and the new values

You Want History Restated

- Standard Type 2 attribute (small to moderate table, infrequent attribute changes)
- A product is now class B and you want the history restated
- Use Type 1 update instead of the standard Type 2

You Want History Restated …

- Standard Type 4 attribute (large dimension or frequent changes)
- Customer Sales Person
  - Sales Person Dim
  - Customer Detail Dim
  - Key must be updated in the fact table

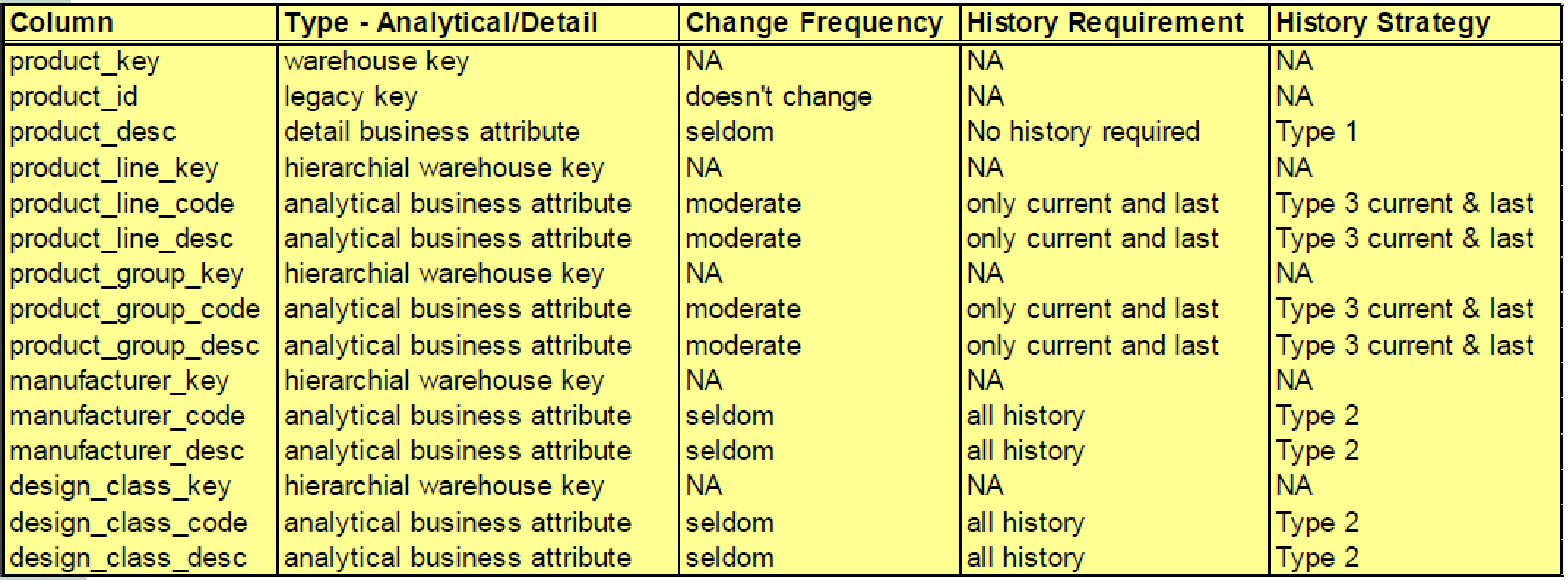
As of Values Required

- Who was the original sales person for the customer
  - Put attribute in Customer Detail Dimension
    - Original Sales Person Name

Example of Type Selection

Column	Type - Analytical/Detail	Frequency of Change	History Requirement
product_key	warehouse key	NA	NA
product_id	legacy key	doesn't change	NA
product_desc	detail business attribute	seldom	No history required
product_line_key	hierarchial warehouse key	NA	NA
product_line_code	analytical business attribute	moderate	only current and last
product_line_desc	analytical business attribute	moderate	only current and last
product_group_key	hierarchial warehouse key	NA	NA
product_group_code	analytical business attribute	moderate	only current and last
product_group_desc	analytical business attribute	moderate	only current and last
manufacturer_key	hierarchial warehouse key	NA	NA
manufacturer_code	analytical business attribute	seldom	all history
manufacturer_desc	analytical business attribute	seldom	all history
design_class_key	hierarchial warehouse key	NA	NA
design_class_code	analytical business attribute	seldom	all history
design_class_desc	analytical business attribute	seldom	all history

Decide on Method



Column	Type - Analytical/Detail	Change Frequency	History Requirement	History Strategy
product_key	warehouse key	NA	NA	NA
product_id	legacy key	doesn't change	NA	NA
product_desc	detail business attribute	seldom	No history required	Type 1
product_line_key	hierarchial warehouse key	NA	NA	NA
product_line_code	analytical business attribute	moderate	only current and last	Type 3 current & last
product_line_desc	analytical business attribute	moderate	only current and last	Type 3 current & last
product_group_key	hierarchial warehouse key	NA	NA	NA
product_group_code	analytical business attribute	moderate	only current and last	Type 3 current & last
product_group_desc	analytical business attribute	moderate	only current and last	Type 3 current & last
manufacturer_key	hierarchial warehouse key	NA	NA	NA
manufacturer_code	analytical business attribute	seldom	all history	Type 2
manufacturer_desc	analytical business attribute	seldom	all history	Type 2
design_class_key	hierarchial warehouse key	NA	NA	NA
design_class_code	analytical business attribute	seldom	all history	Type 2
design_class_desc	analytical business attribute	seldom	all history	Type 2

Outline for This Session

- Transformation Process
- Dimension Table Processing
- Dimension Update Types
- **Fact Table Processing**
- Loading Process
- Case Stufy

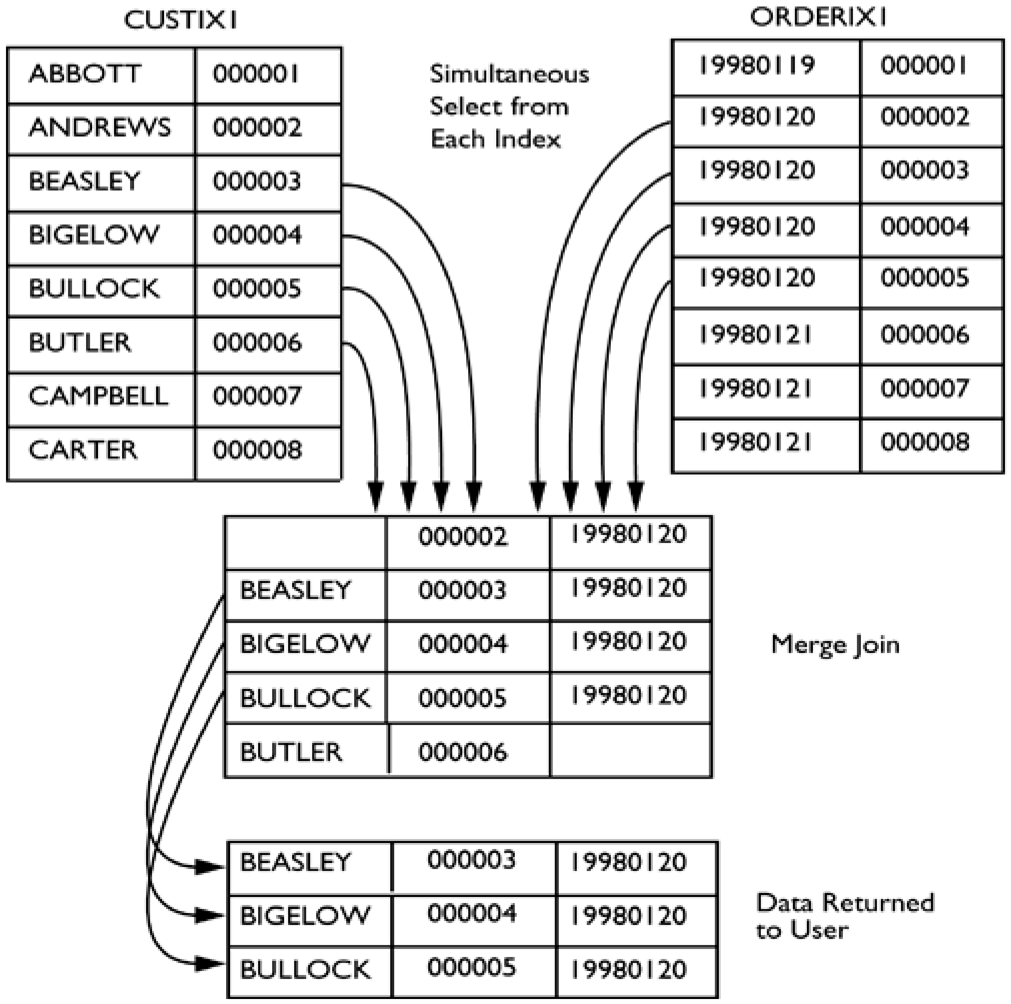
Fact Table and Transaction Dimension Processing

- Volume makes efficiency, performance, and scalability even more important than with dimension tables
- Relatively straight forward process except for sophisticated calculations
- Several methods
- Pick the best one for your situation

Dimension Key Lookup Methods

- Use operational keys to look up the dimension key for the each respective dimension
- Sort merge
  - Create a lookup table or file with the operational key and the assigned warehouse key sorted in operational key order
- No No No
  - For each fact table row select dimkey from dim where d.optkey = f.optkey (Cursor Based)

Dimension Key Lookup Methods



CUSTIX1

ABBOTT	000001
ANDREWS	000002
BEASLEY	000003
BIGELOW	000004
BULLOCK	000005
BUTLER	000006
CAMPBELL	000007
CARTER	000008

ORDERIX1

19980119	000001
19980120	000002
19980120	000003
19980120	000004
19980120	000005
19980121	000006
19980121	000007
19980121	000008

	000002	19980120
BEASLEY	000003	19980120
BIGELOW	000004	19980120
BULLOCK	000005	19980120
BUTLER	000006	

BEASLEY	000003	19980120
BIGELOW	000004	19980120
BULLOCK	000005	19980120

http://etutorials.org/Misc/advanced+dba+certification+guide+and+reference/Chapter+6.+The+DB2+Optimizer/Joining+in+DB2+UDB/

Divide and Conquer

- For extremely large volumes
- Parallelize by breaking up input into multiple files
- Parallelize by assigning serial key and break into multiple paths
  - Key + measures
  - Key + dimkey1
  - -----------------
  - Key + dimkeyn
  - Merge results

Mix Methods

- Parallelize input by multiple files
- Sort by largest dimension
- Do hash lookups for smaller dimensions

Lookup Failures in Fact Table Loads

- Choice 1: Reject fact table rows containing invalid product ids until they have been added to the dimension table
- Choice 2: Add row to fact table with default product key signifying invalid product id
  - Write error row
- Choice 3: Add new dimension record with new key for new product id with all other attributes set to unknown (use with caution)

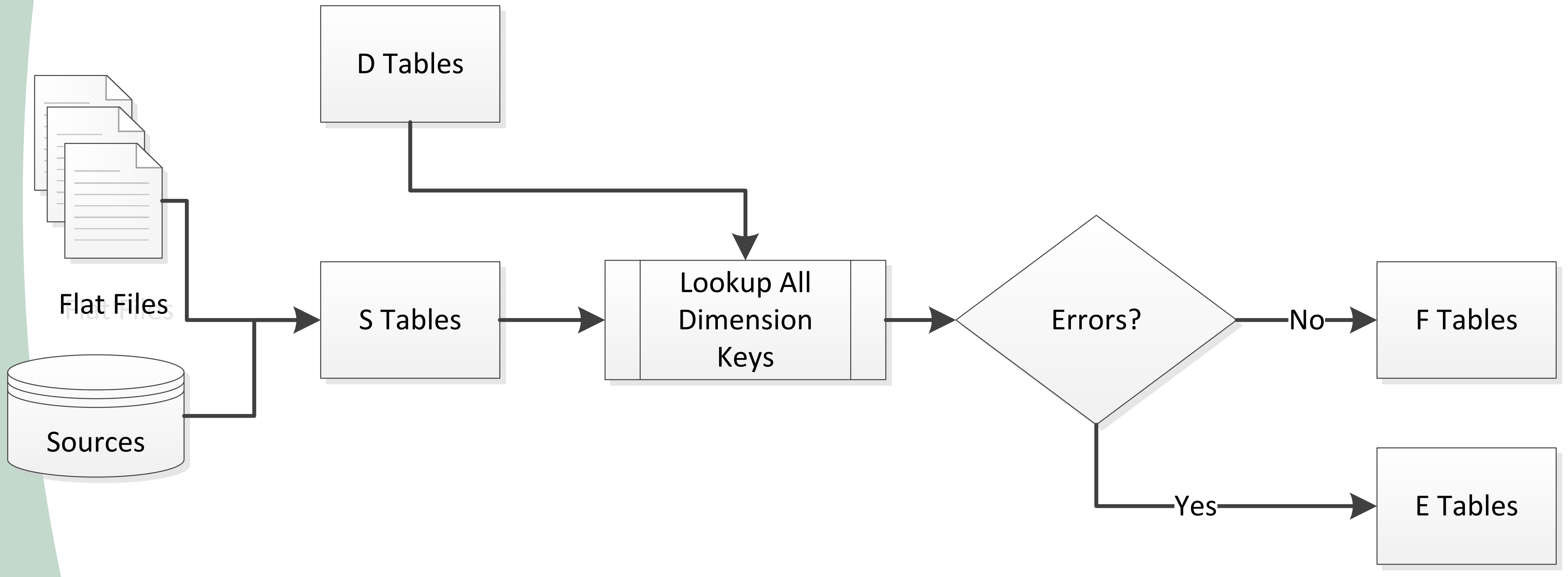
Processing the Error Rows

- Fix the source problem
  - Usually valid code not added yet
  - Or erroneous code entered at data entry
- Add a new row to the dimension after the correct code and data are available
- If you used a key for an invalid row, do you go back and update the fact table after the dimension is updated?

Lookup Failure Consequences

- Incomplete data until resolved
- Requires fact table update to correct
- Requires dimension update when corrected (can be very dangerous)

Source to Fact Table Processing



Publish Data Quality

- Total Sales figures for 9/5/2009 contain:
  - 2% invalid product codes
  - 0.5% invalid salesperson codes
  - 5% invalid geography codes
- Northwest regional sales for 9/5/2009 contain: 0.5% invalid product codes
- 2% of Product Sales figures for 9/5/2009 are due to invalid products

Outline for This Session

- Transformation Process
- Dimension Table Processing
- Dimension Update Types
- Fact Table Processing
- **Loading Process**
- Case Study

Data Loading

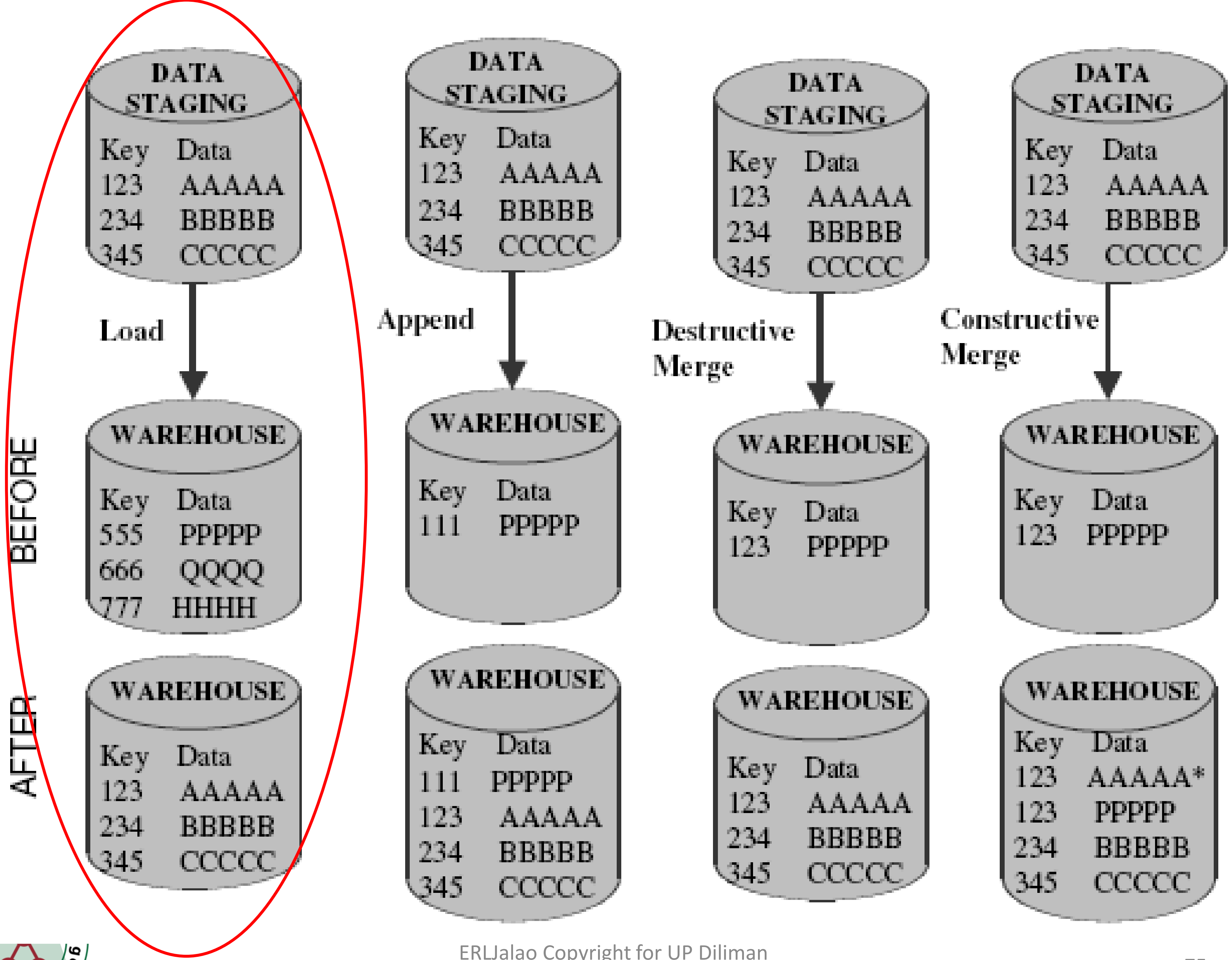
- Data loading takes the prepared data, applies it to the data warehouse, and stores it in the database
- Terminology:
  - Initial Load — populating all the data warehouse tables for the very first time
  - Incremental Load — applying ongoing changes as necessary in a periodic manner
  - Full Refresh — completely erasing the contents of one or more tables and reloading with fresh data (initial load is a refresh of all the tables)

Applying Data: Techniques and Processes

- Load
- Append
- Destructive Merge
- Constructive Merge

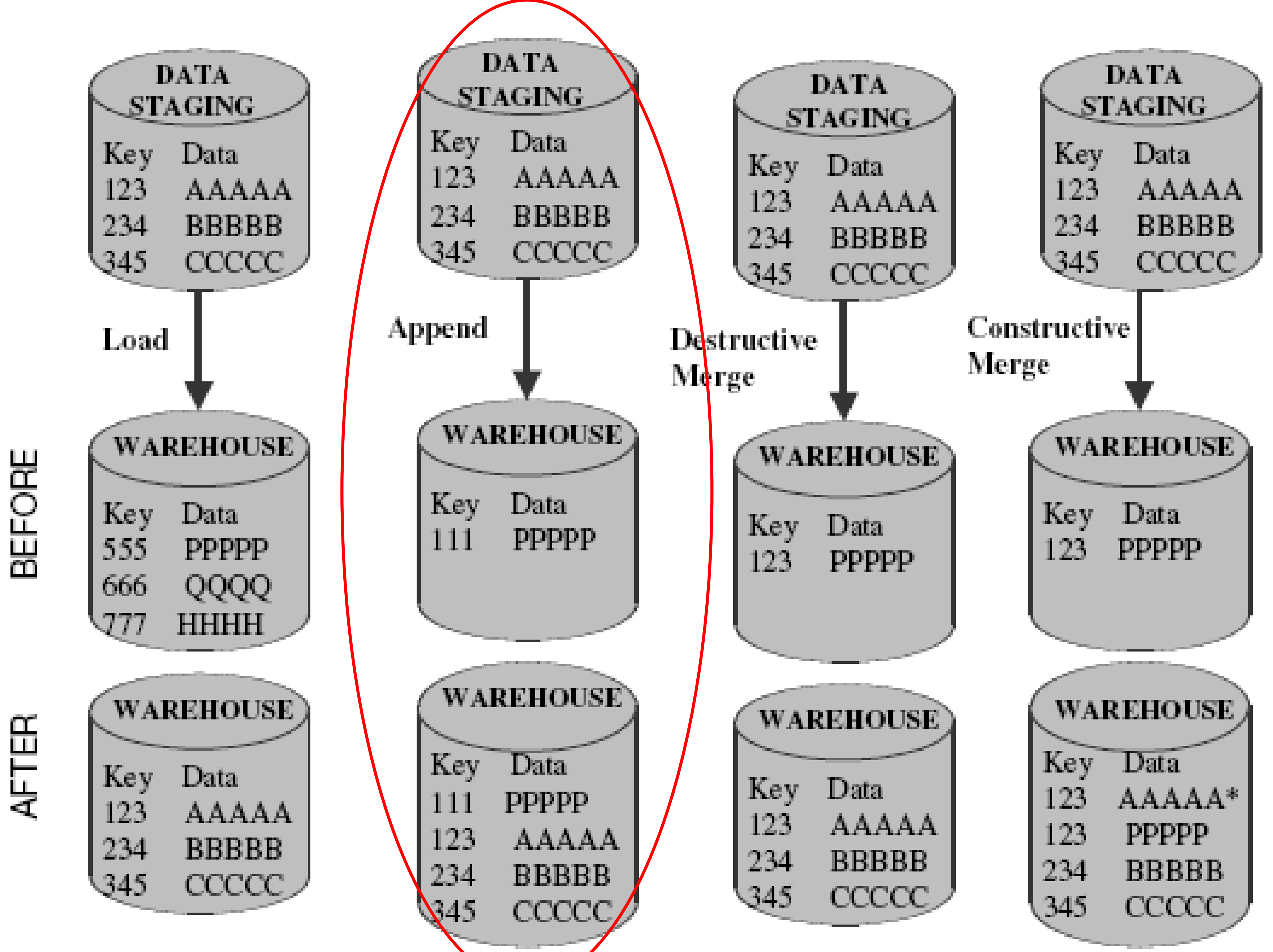
Load

- 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.


DATA STAGING
Key Data
123 AAAAA
234 BBBBB
345 CCCCC
Load
Append
Destructive Merge
Constructive Merge
BEFORE
WAREHOUSE
Key Data
555 PPPPP
666 QQQQ
777 HHHH
Key Data
111 PPPPP
Key Data
123 PPPPP
AFTER
Key Data
123 AAAAA
234 BBBBB
345 CCCCC
Key Data
111 PPPPP
123 AAAAA
234 BBBBB
345 CCCCC
Key Data
123 AAAAA*
123 PPPPP
234 BBBBB
345 CCCCC

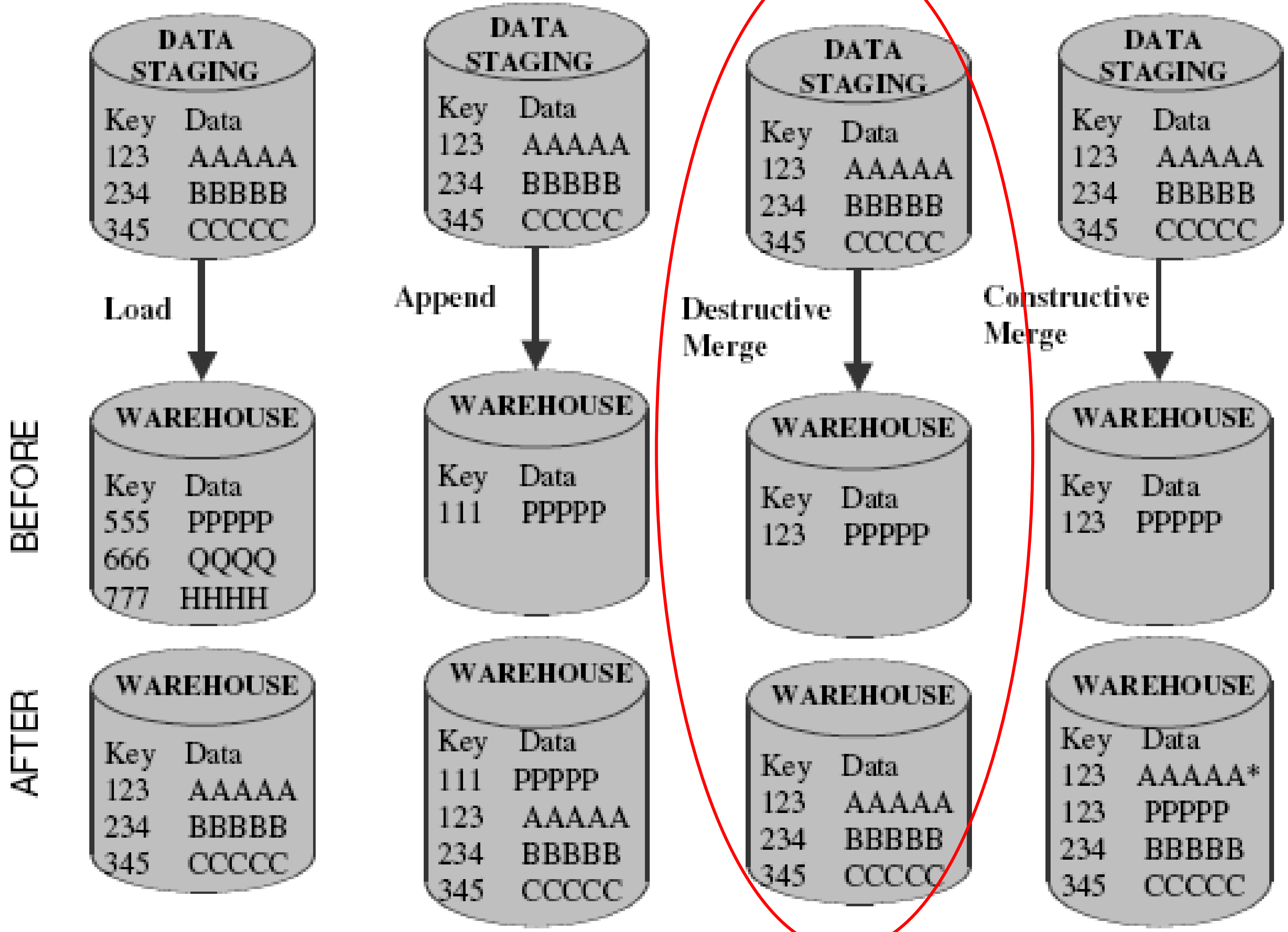
Append

- Extension of the load.
- 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.
  - In the other option, the incoming duplicate record may be rejected during the append process.


DATA STAGING
Key Data
123 AAAAA
234 BBBBB
345 CCCCC
Load
Append
Destructive Merge
Constructive Merge
BEFORE
WAREHOUSE
Key Data
555 PPPPP
666 QQQQ
777 HHHH
Key Data
111 PPPPP
Key Data
123 PPPPP
AFTER
WAREHOUSE
Key Data
123 AAAAA
234 BBBBB
345 CCCCC
Key Data
111 PPPPP
123 AAAAA
234 BBBBB
345 CCCCC
Key Data
123 AAAAA
234 BBBBB
345 CCCCC
Key Data
123 AAAAA*
123 PPPPP
234 BBBBB
345 CCCCC

Destructive Merge

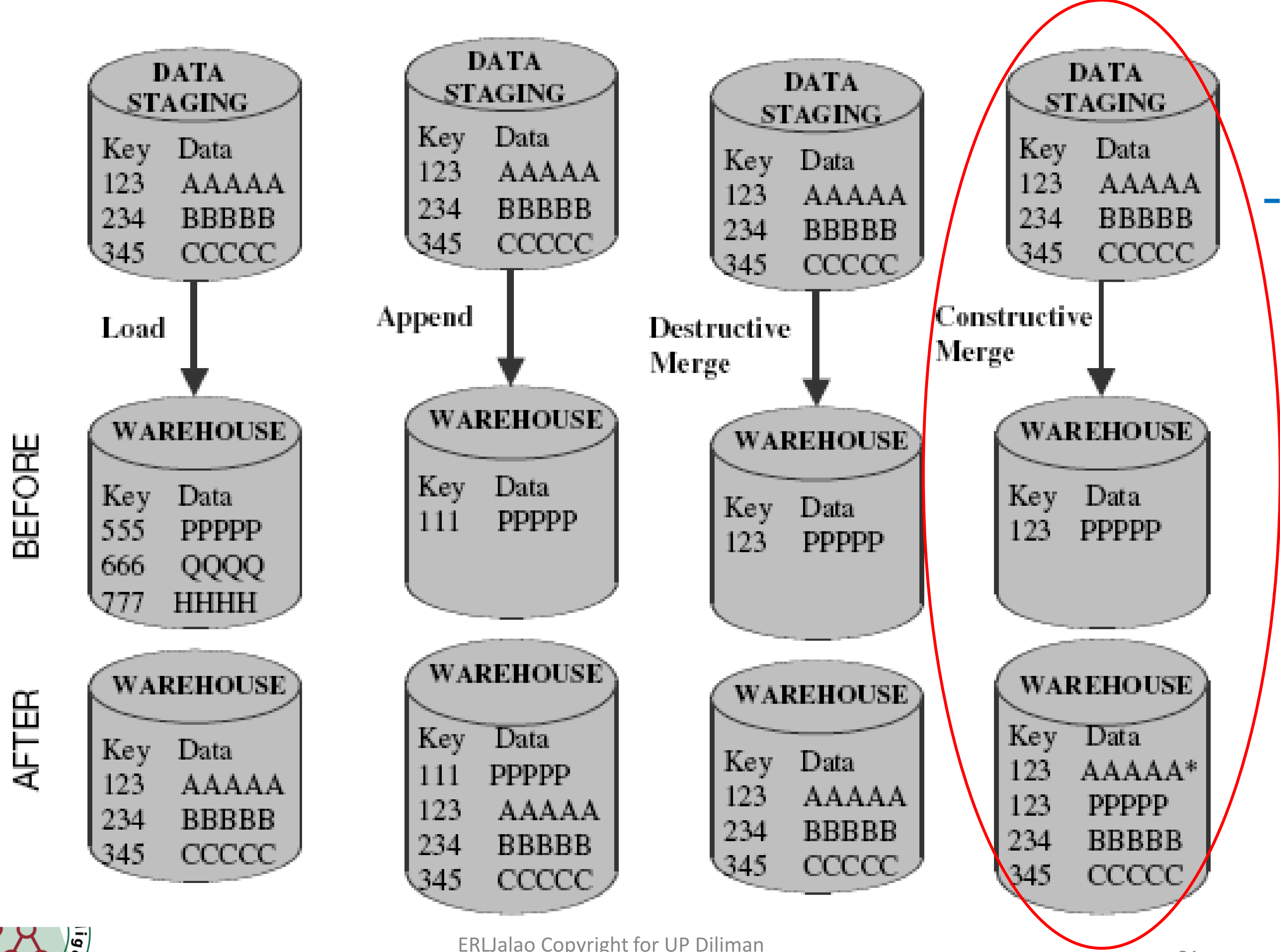
- Applies incoming data to the target data.
- If the primary key of an incoming record matches with the key of an existing record, update the matching target record.
- If the incoming record is a new record without a match with any existing record, add the incoming record to the target table.



BEFORE
AFTER
DATA STAGING
Key Data
123 AAAAA
234 BBBBB
345 CCCCC
Load
WAREHOUSE
Key Data
555 PPPPP
666 QQQQ
777 HHHH
WAREHOUSE
Key Data
123 AAAAA
234 BBBBB
345 CCCCC
DATA STAGING
Key Data
123 AAAAA
234 BBBBB
345 CCCCC
Append
WAREHOUSE
Key Data
111 PPPPP
WAREHOUSE
Key Data
111 PPPPP
123 AAAAA
234 BBBBB
345 CCCCC
DATA STAGING
Key Data
123 AAAAA
234 BBBBB
345 CCCCC
Destructive Merge
WAREHOUSE
Key Data
123 PPPPP
WAREHOUSE
Key Data
123 AAAAA
234 BBBBB
345 CCCCC
DATA STAGING
Key Data
123 AAAAA
234 BBBBB
345 CCCCC
Constructive Merge
WAREHOUSE
Key Data
123 PPPPP
WAREHOUSE
Key Data
123 AAAAA*
123 PPPPP
234 BBBBB
345 CCCCC

Constructive Merge

- 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.


BEFORE
AFTER
DATA STAGING
Key Data
123 AAAAA
234 BBBBB
345 CCCCC
Load
WAREHOUSE
Key Data
555 PPPPP
666 QQQQ
777 HHHH
WAREHOUSE
Key Data
123 AAAAA
234 BBBBB
345 CCCCC
DATA STAGING
Key Data
123 AAAAA
234 BBBBB
345 CCCCC
Append
WAREHOUSE
Key Data
111 PPPPP
WAREHOUSE
Key Data
111 PPPPP
123 AAAAA
234 BBBBB
345 CCCCC
DATA STAGING
Key Data
123 AAAAA
234 BBBBB
345 CCCCC
Destructive Merge
WAREHOUSE
Key Data
123 PPPPP
WAREHOUSE
Key Data
123 AAAAA
234 BBBBB
345 CCCCC
DATA STAGING
Key Data
123 AAAAA
234 BBBBB
345 CCCCC
Constructive Merge
WAREHOUSE
Key Data
123 PPPPP
WAREHOUSE
Key Data
123 AAAAA*
123 PPPPP
234 BBBBB
345 CCCCC

Outline for This Session

- Transformation Process
- Dimension Table Processing
- Dimension Update Types
- Fact Table Processing
- Loading Process
- **Case Study**

Case Study 4

- ETL Case Walkthrough using R

NATIONAL ENGINEERING CENTER
University of the Philippines
Diliman, Quezon City

Center for Business Intelligence

Overview of Training Modules of UP NEC

https://www.facebook.com/upnecanalytics

http://www.upnec.com/

6 Modules

- Analyst Level
  1. Introduction to Business Intelligence
  2. Data Warehousing
  3. Data Mining
- Professional Level
  4. Optimization
  5. Forecasting and Time Series Analysis
  6. R for BI

Analyst Level Certifications

	Data Mining Analyst	Data Warehousing Analyst	Business Intelligence Analyst
Introduction to BI	√	√	√
Data Warehousing		√	√
Data Mining	√		√
Exam	√	√	√

Certification Exam

- Coverage:
  - BI Analyst: All Three Modules
  - Data Mining Analyst: Module 1 and Module 3
  - Data Warehousing Analyst: Module 1 and Module 2
- Type
  - Multiple Choice, Concept Based Questions (From Notes)
  - 3 Hours for BI Analyst
  - 2 Hours for Data Mining and Data Warehousing Analyst
- Offered Monthly c/o Ms. Rea

Professional Level Certifications

	Data Mining Professional	Business Intelligence Professional
Introduction to BI	√	√
Data Warehousing		√
Data Mining	√	√
Optimization		√
Time Series and Forecasting	√	√
Advanced Data Mining	√	√
Exam	√	√

References

- Kimball, Ralph, Margy Ross, Warren Thornthwaite, Joy Mundy, and Bob Becker, *The Data Warehouse Life Cycle Toolkit, Second Edition*, Wiley, 2008, ISBN 978-0-470-14977-5
- Schmitz, Michael D. UCI Irvine Data Warehousing Notes (2014), High Performance Data Warehousing
- Simon, Alan. CIS 391 PPT Slides
- Jeltema ,Bernie, UCI Irvine Data Warehousing Notes (2014), Strategic Frameworks, Inc.