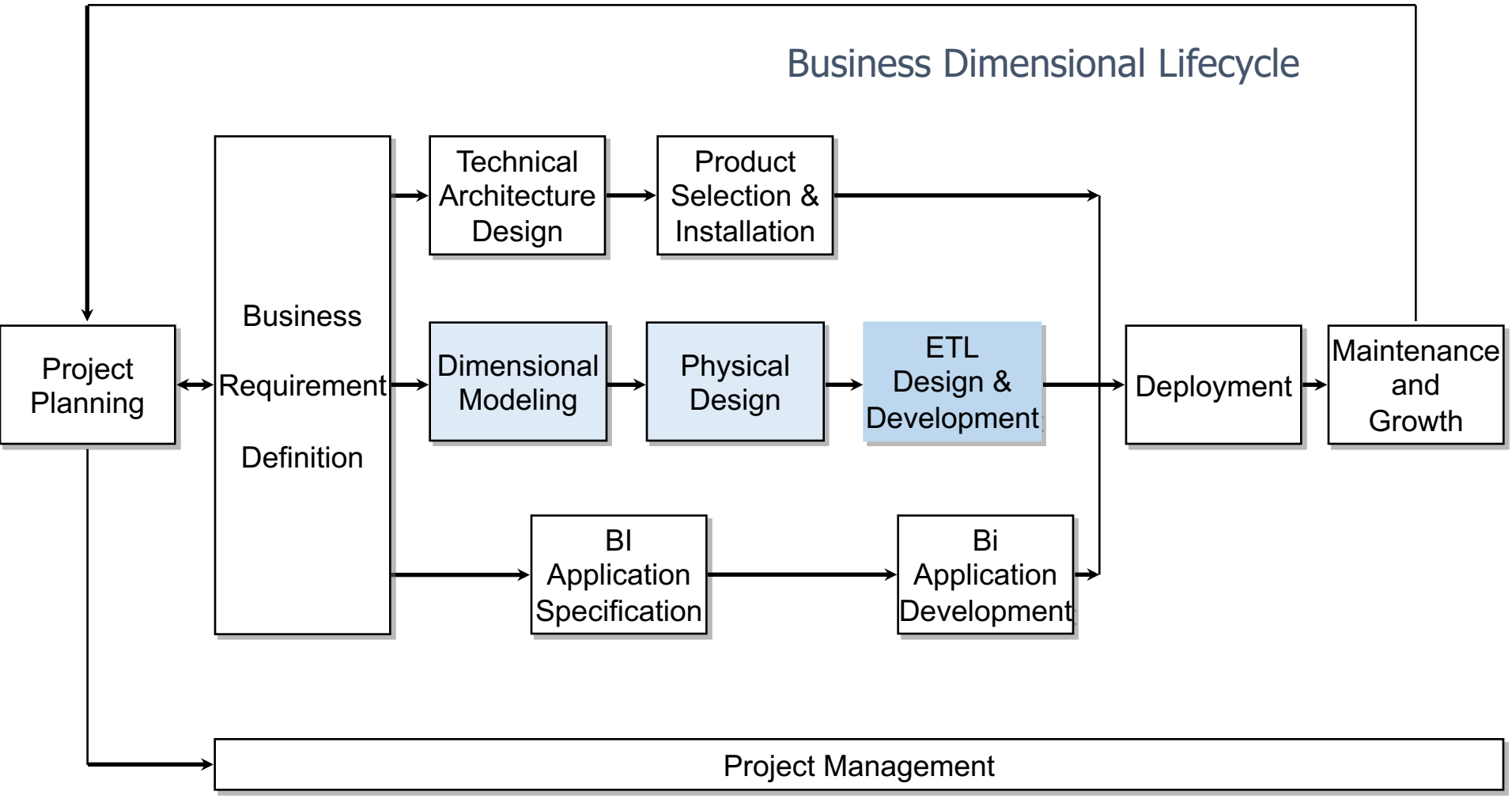
# CSI4142 Data Science

## Data staging

(Notes by Hl Viktor © Refer to Kimball et. al. Chapters 9 and 10, Han et.al. Chapter 3)

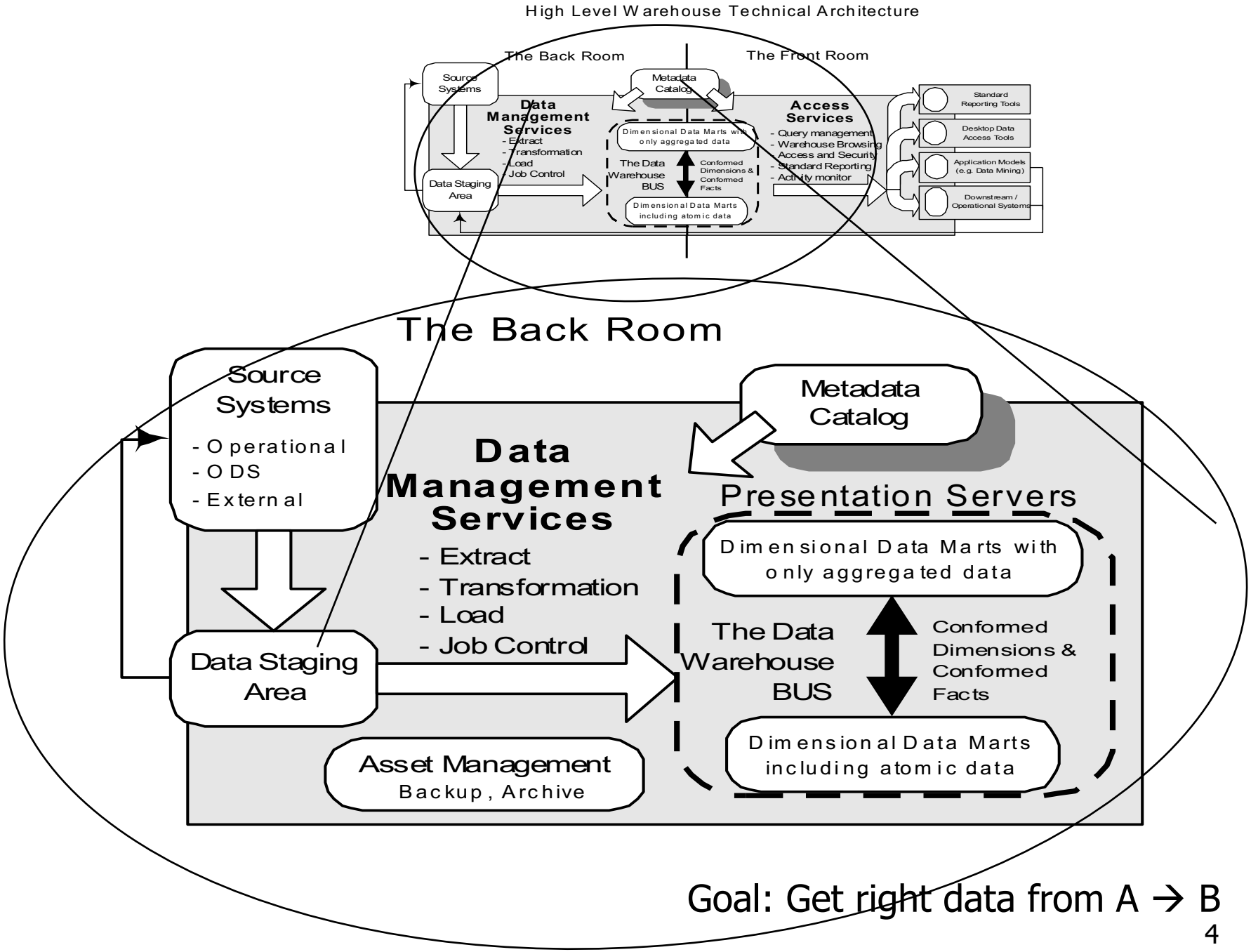
# Data Staging (ETL)



# The goal of data staging

Getting **the right data**

from A (Sources) to B (Data Marts)




Goal: Get right data from A → B

# So, what is the best way to do data staging? Considerations

- Round up the requirements
- Consider the Business Needs
- Study the Sources
- Look out for data limitations
- Decide on scripting languages
- Look at the staff skills
- Remember legacy licence(s) (!!!)

# The Data staging steps

- A: Planning
  1. High level plan
  2. Choose a tool
  3. Detailed planning: dimension management, error handling
  4. Detailed planning by target table

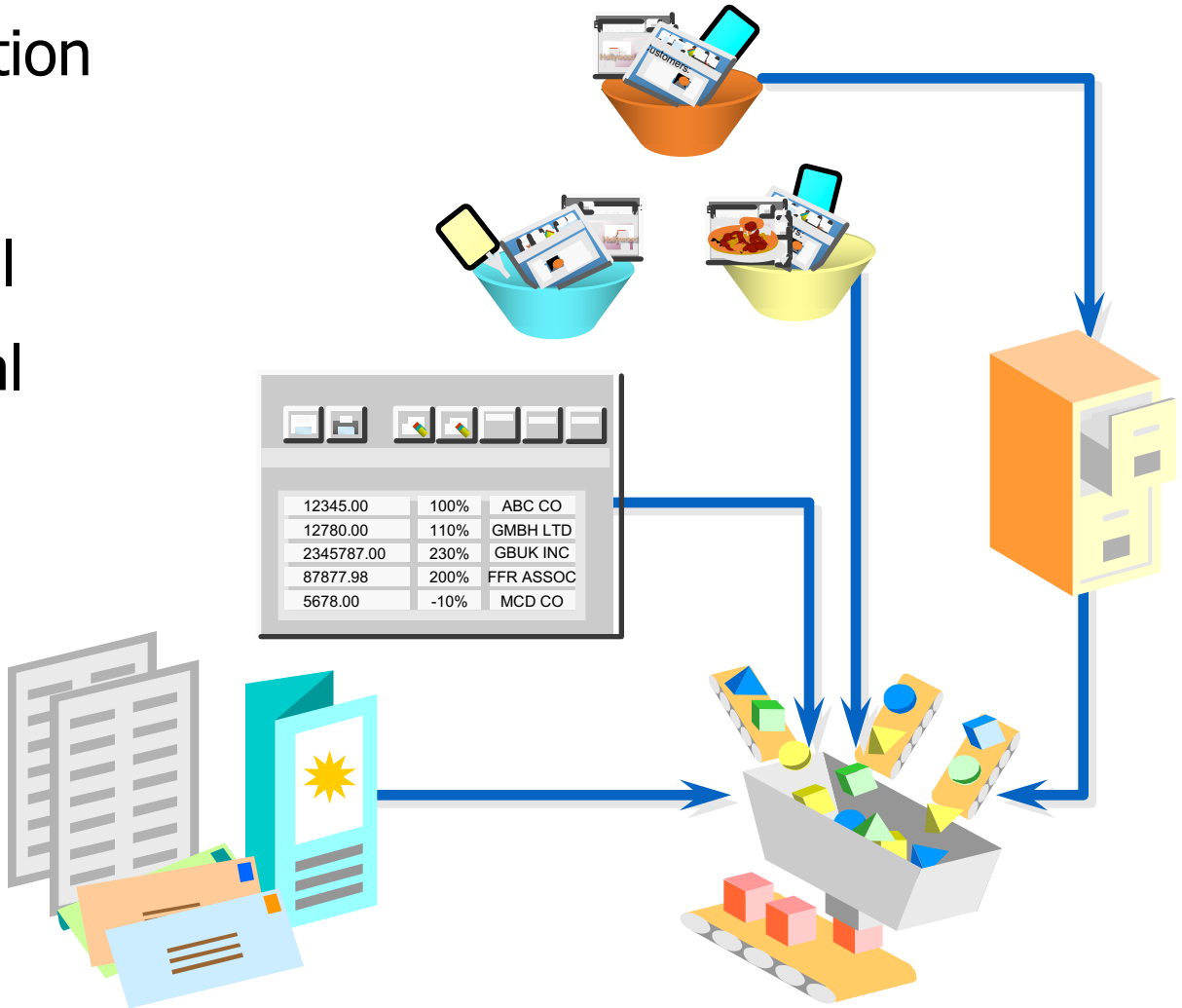
- B: Develop One-time Historic Load
- C: Develop Incremental Load

# Step A1: High-level Planning

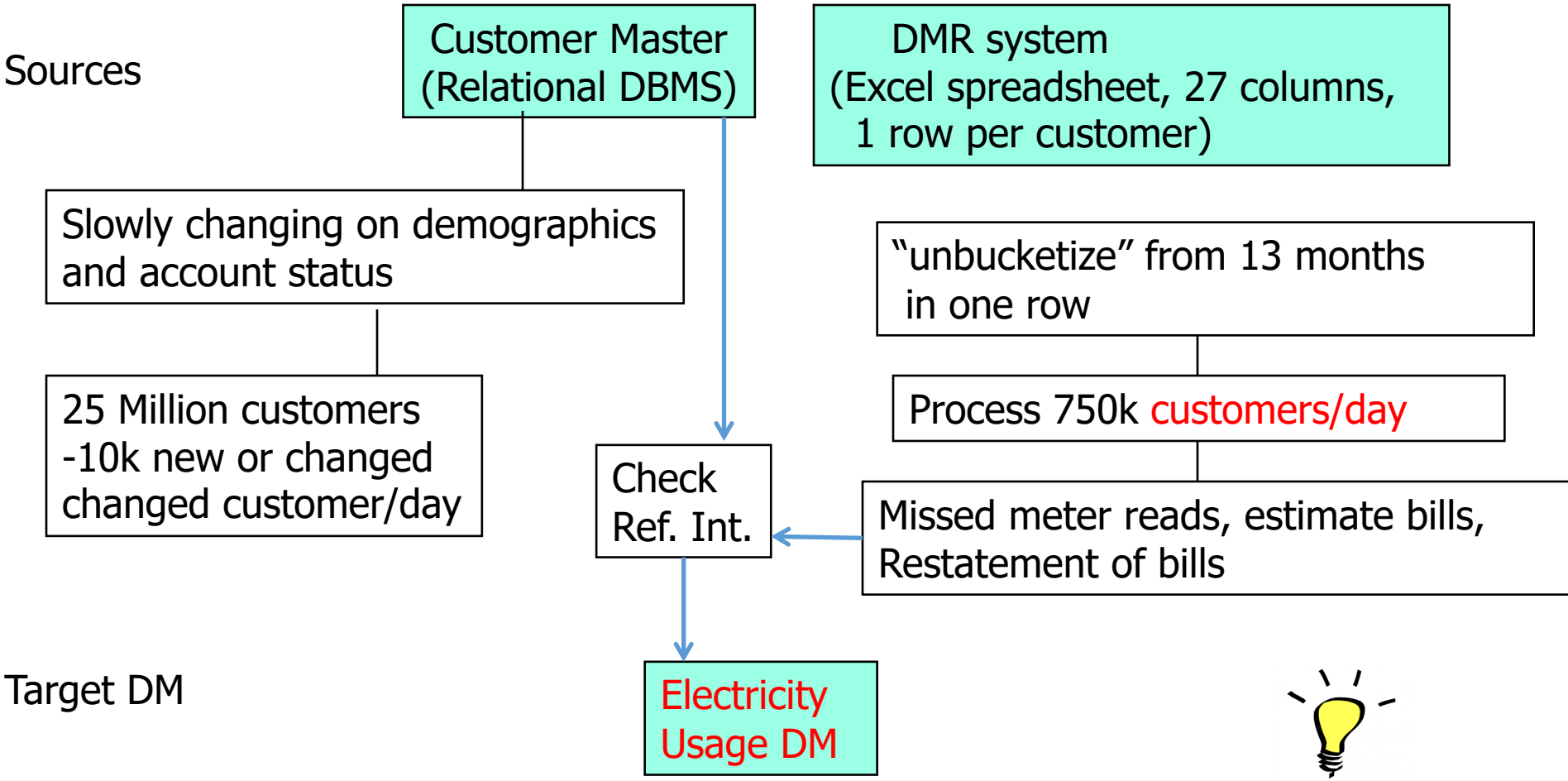
- Create a very high-level, **one-page schematic** of the source-to-target flow
- Identify starting and ending points
- Label known data sources
- Include placeholders for sources yet to be determined
- Label targets
- Include notes about known problems

# Identification of Source Systems

- Production
- Archive
- Internal
- External

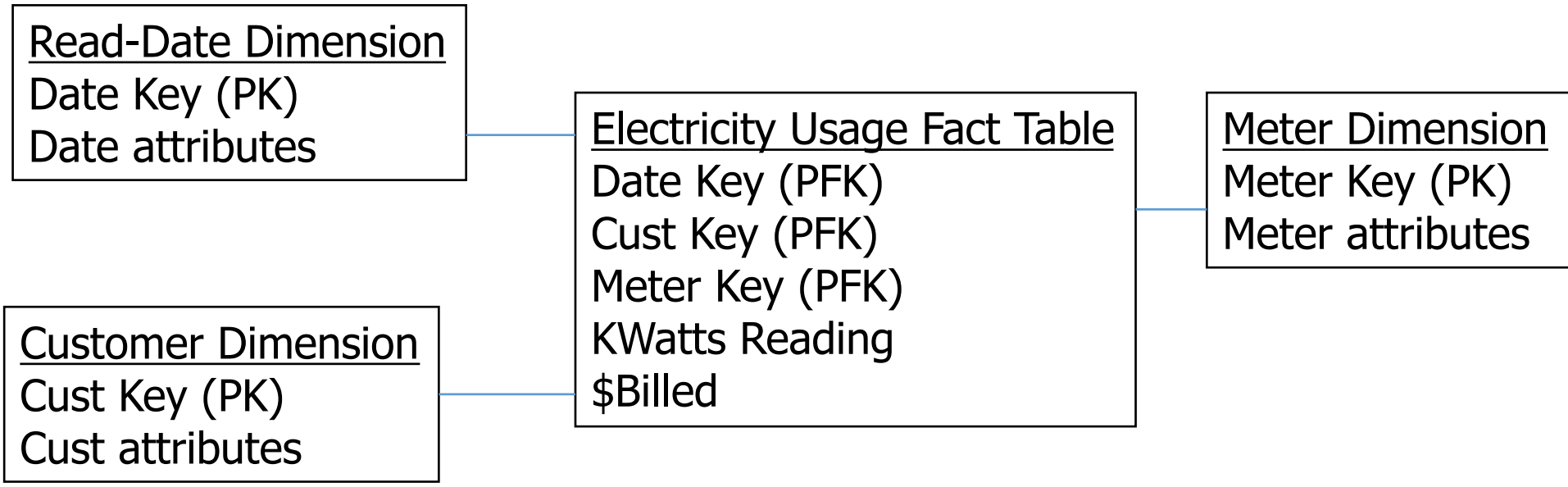


# Basic high-level data staging plan schematic: An electricity usage example



# Electricity usage star schema

- We would get the details about the meter types, etc. from the suppliers
- Customer would include demographic data based on postal code (e.g. from Stats Canada)



# Source data

| CustomerID | kWh | Bill | KWh | Bill | Kwh | Bill | … | Kwh | Bill |
|---|---|---|---|---|---|---|---|---|---|
| AWD1001 | 389 | $54.59 | 750 | $88.77 | 500 | $76.00 | | 600 | $74.01 |
| DWAS4522 | 900 | $203.44 | 750 | $127.61 | 400 | $70.23 | | 700 | $101.23 |
| … | | | | | | | | | |

- Readings data from Jan 2016 to Jan 2017 (Excel): 27 columns
- RDMS records for Customers

| Customer-ID | Name | City | Meter-type | other attributes |
|---|---|---|---|---|
| AWD1001 | Jane | Calgary | LCD100 | |
| DWAS4522 | Joe | Ottawa | RCD203 | |

# Un-bucketise…

| CustomerID | kWh | Bill | Month |
|---|---|---|---|
| AWD1001 | 389 | $54.59 | Jan-16 |
| AWD1001 | 750 | $88.77 | Feb-16 |
| AWD1001 | 500 | $76.00 | Mar-16 |
| … | … | … | … |
| AWD1001 | 600 | $74.01 | Jan-17 |
| DWAS4522 | 900 | $203.44 | Jan-16 |
| DWAS4522 | 750 | $127.61 | Feb-16 |
| DWAS4522 | 400 | $70.23 | Mar-16 |
| … | … | … | … |
| DWAS4522 | 700 | $101.23 | Jan-17 |

| Customer-ID | Name | City | Meter-type | other attributes |
|---|---|---|---|---|
| AWD1001 | Jane | Calgary | LCD100 | |
| DWAS4522 | Joe | Ottawa | RCD203 | |

# Step A2: Choose data staging tools

- Do it yourself, writing scripts (e.g. json)
- Use a data staging tool
  - All major data warehouse vendors offer one
  - Drawbacks: black box, learning curve

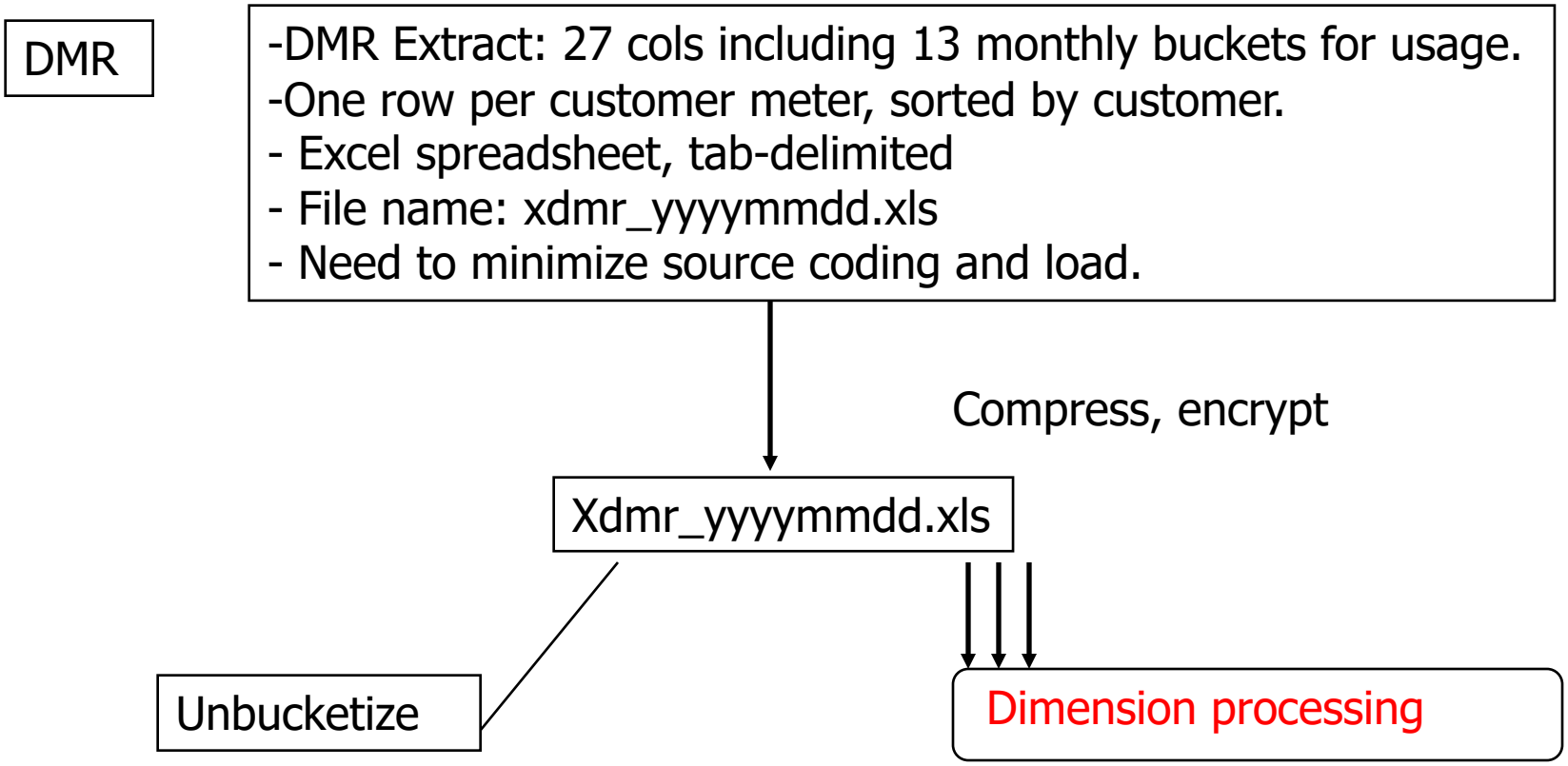
# Step A3: General planning

- Extraction from multiple sources (timing, information fusion)
- Archiving (when?)
- Data quality management
- Change management (when? how?)

# Step A4: Detailed planning by table

- Drill down by target table, graphically sketching any complex data restructuring or transformations
- Identify attribute hierarchies (normalize the source)
- Graphically illustrate the surrogate-key generation process
- Develop a preliminary job sequencing

# Step A4: First draft of historic load schematic for the fact table



# Dimension Processing….

- Customer
- Meter
- Read-Date

| Cust-key | Name | City | other attributes |
|---|---|---|---|
| 100 | Jane | Calgary | |
| 101 | Joe | Ottawa | |

| Meter-key | Meter-type | other attributes |
|---|---|---|
| 400 | LCD100 | |
| 401 | RCD203 | |

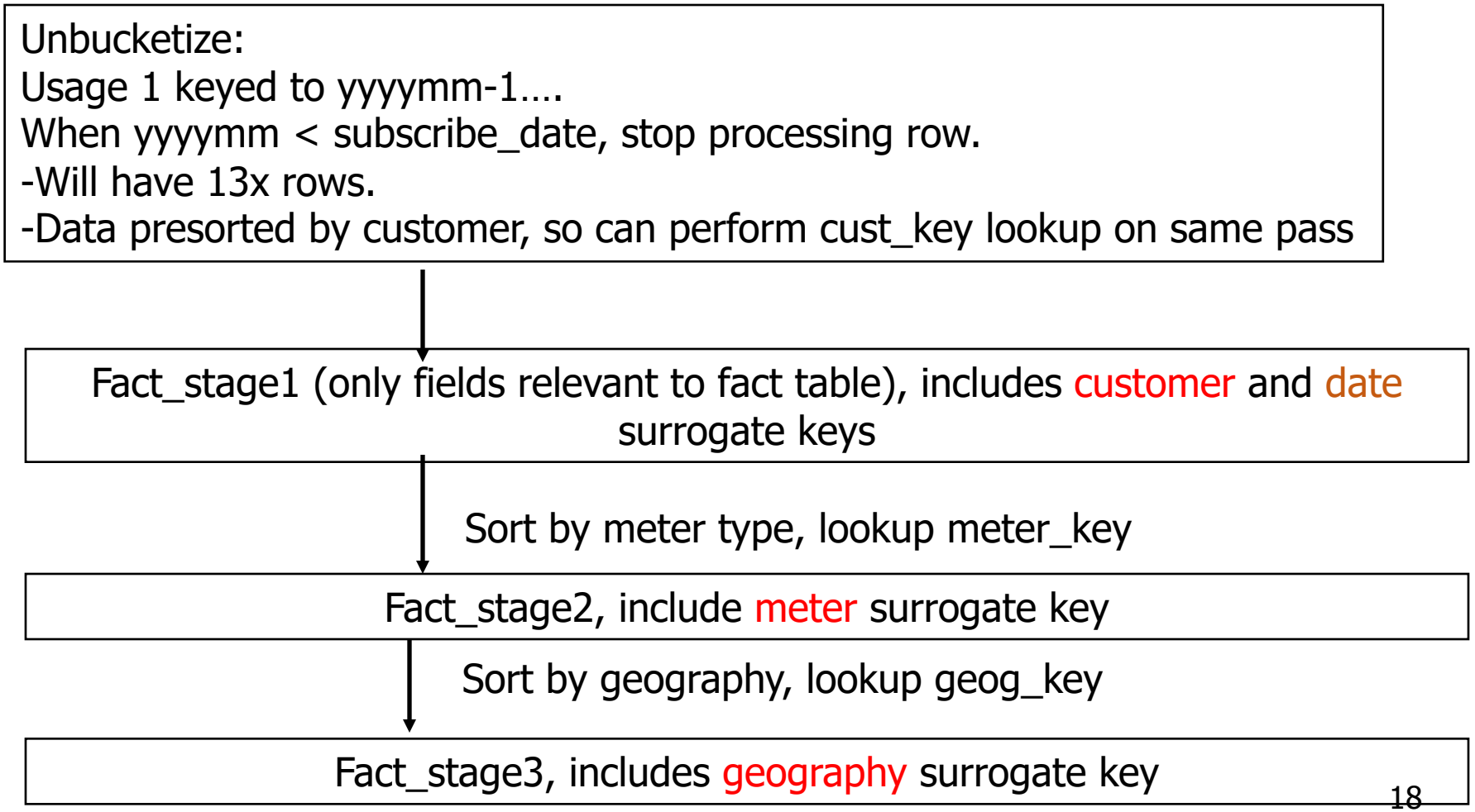
Remember:-

- Denormalised
- Avoid snowflakes

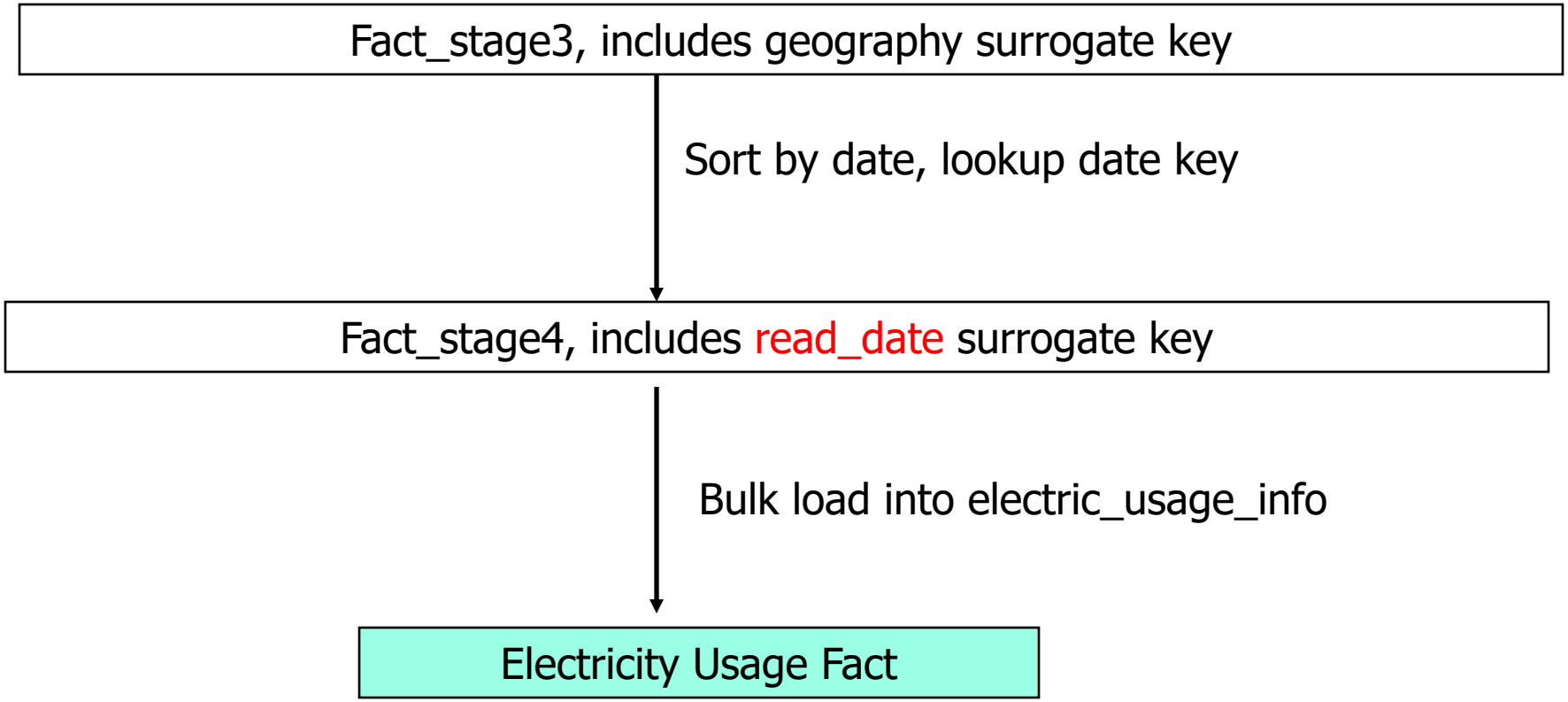
| CustomerID | kWh | Bill | Month |
|---|---|---|---|
| AWD1001 | 389 | $54.59 | Jan-16 |
| AWD1001 | 750 | $88.77 | Feb-16 |
| AWD1001 | 500 | $76.00 | Mar-16 |
| … | … | … | … |
| AWD1001 | 600 | $74.01 | Jan-17 |
| DWAS4522 | 900 | $203.44 | Jan-16 |
| DWAS4522 | 750 | $127.61 | Feb-16 |
| DWAS4522 | 400 | $70.23 | Mar-16 |
| … | … | … | … |
| DWAS4522 | 700 | $101.23 | Jan-17 |

| Customer-ID | Name | City | Meter-type | other attributes |
|---|---|---|---|---|
| AWD1001 | Jane | Calgary | LCD100 | |
| DWAS4522 | Joe | Ottawa | RCD203 | |

# Step A4: First draft of historic load schematic for the fact table (e.g.)



# Planning Step A4: First draft of historic load schematic for the fact table (cont)



# The Data staging steps

- A: Planning
  1. High level plan
  2. Choose a tool
  3. Detailed planning: dimension management, error handling
  4. Detailed planning by target table
- B: Develop One-time Historic Load
  1. Populate dimension tables
  2. Populate fact table (and create data mart)
  3. Consider data preprocessing for analytics
- C: Develop Incremental Load

# B: Develop one-time historic load

1. Build and test the historic dimension table loads
2. Build and test the historic fact table loads, including surrogate key lookup and substitution

# Step B1: Populate dimension tables

- Static dimension extract
- Creating and moving the result set
  - Data compression
  - Data encryption
- Static dimension transformation
- Simple data transformations
- Surrogate key assignment
- Combining from separate sources
- Validating one-to-one and one-to-many relationships

# Surrogate key assignment

- Use integer “autonumbers”, increasing by 1
- Maintain a table with the production_key → surrogate_key matches

| SKU | Product | Brand | Supplier |
|---|---|---|---|
| 12319319 | Milk | Natrel | Saputo |
| 12319336 | Milk | Quebon | Saputo |
| 12319353 | Milk | Grand Pre | Lactantia |
| 12319370 | Cream | Quebon | Saputo |
| 12319387 | Cream | Natrel | Saputo |
| 12319404 | Brie | Yellow | Metro |
| 12319421 | Brie | French | Metro |
| 12319438 | Cheddar | Trappe | Fromage |
| 12319455 | Gouda | Trappe | Fromage |

| SKU | Product-key |
|---|---|
| 12319319 | |
| 12319336 | |
| 12319353 | |
| 12319370 | |
| 12319387 | |
| 12319404 | |
| 12319421 | |
| 12319438 | |
| 12319455 | |

| Product--key | Product | Brand | Supplier |
|---|---|---|---|
| | Milk | Natrel | Saputo |
| | Milk | Quebon | Saputo |
| | Milk | Grand Pre | Lactantia |
| | Cream | Quebon | Saputo |
| | Cream | Natrel | Saputo |
| | Brie | Yellow | Metro |
| | Brie | French | Metro |
| | Cheddar | Trappe | Fromage |
| | Gouda | Trappe | Fromage |

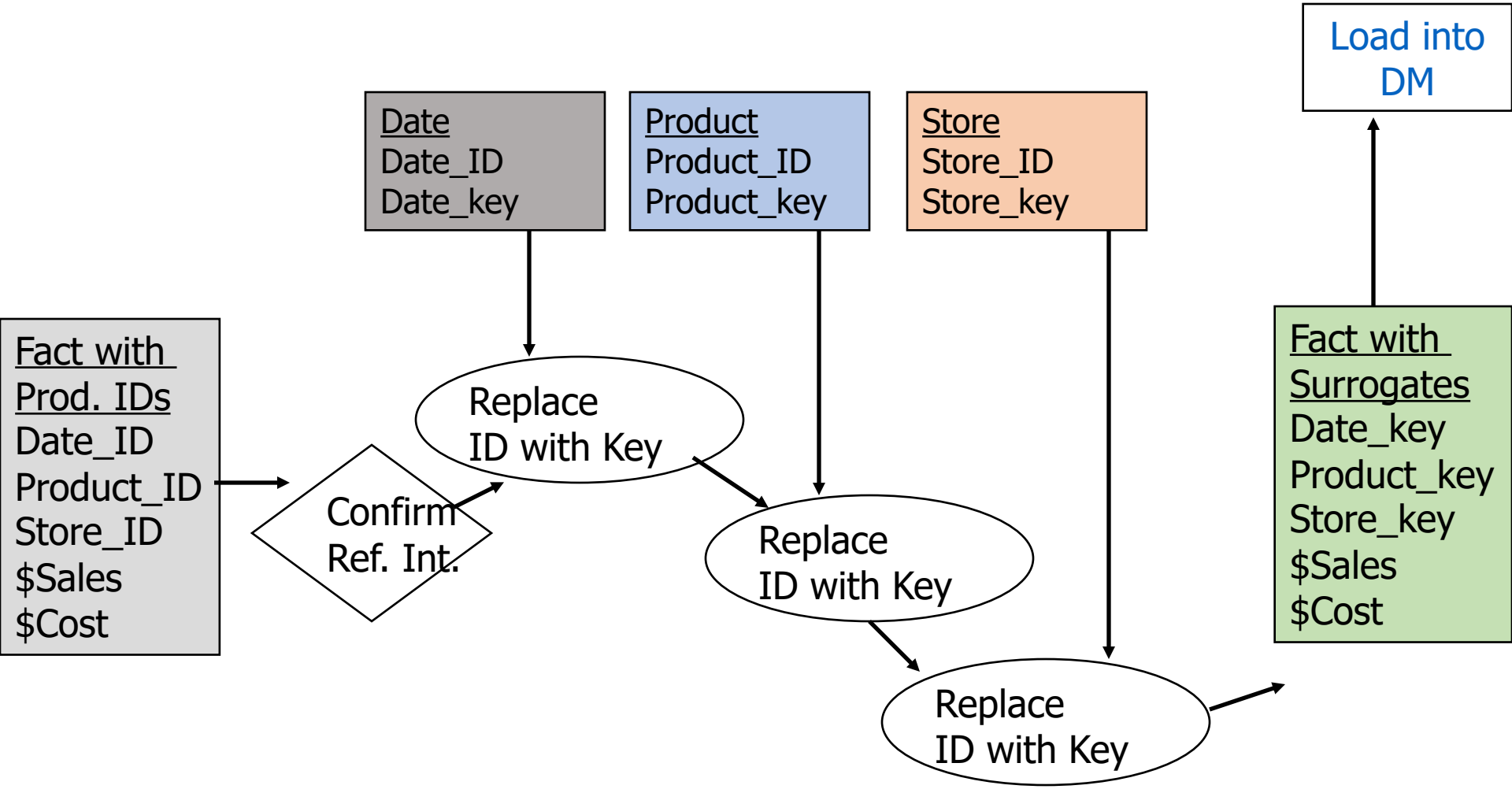
# Step B1:
# Populate a simple dimension table (cont).

- Load
  - Bulk loader
  - Turn off logging
  - Pre-sort the file
  - Transform with caution
  - Aggregations
  - Use the bulk loader to perform “within-database” inserts
- Index management
  - Drop and re-index
  - Keep indexes in place

# Step B3:
# Historic load of Atomic-level DM

- Fact table processing
  - Fact table surrogate key lookup
- Ensure Referential Integrity!!!

# Surrogate key pipeline for Store Example



# Transformations for Analytics and Data Mining

- Flag normal, abnormal, out of bounds, or impossible facts
- Recognize random or noise values from context and mask out
- Apply a uniform treatment to null values
- Flag fact records with changed status
- Classify an individual record by one of its aggregates
- Add computed fields as inputs or targets
- Map continuous values into ranges
- Normalize values between 0 and 1
- Convert from textual to numeric or numeral category
- Emphasize the unusual case abnormally to drive recognition

# Steps to transform the data
(Chapter 3 of Han et. al.)

1. Data cleaning
2. Data integration and transformation
3. Data reduction

Design decision: done during data staging or by user applications (or at both ends)

- Depends on domain, organization culture, end user needs and skills

# Why clean the data?

- Incomplete; e.g. age missing
- Noisy; e.g. age = 130 (human)
- Inconsistent; e.g. province = “BC” and postal code = “K1N”

Others:

- Redundant duplicates (referential integrity: “John Smith”)
- Incorrect formats (inches versus meters)
- Etc.

# Data Cleaning

- Importance
  - “Data cleaning is the number one problem in data science”—DCI survey
- Data cleaning tasks
  - Fill in missing values
  - Identify outliers and smooth out noisy data
  - Correct inconsistent data
  - Resolve redundancy caused by data integration

# Missing values

| Product--key | Product | Brand | Supplier |
|---|---|---|---|
| 100 | Milk | Natrel | Saputo |
| 101 | Milk | | |
| 102 | Milk | Grand Pre | Lactantia |
| 103 | Cream | Quebon | Saputo |
| 104 | ? | Natrel | Saputo |
| 105 | Brie | Yellow | Metro |
| 106 | ? | ? | ? |
| 107 | Cheddar | Trappe | Fromage |
| 108 | Gouda | Trappe | Fromage |

- Ignore
- Fill manual
- Use default value (e.g. unknown)
- Use mean value (e.g. average income of all clients)
- Use mean value of class or grouping (e.g. average income of all clients from Orleans suburb in 30-35 age group)
- Use most probable value (e.g. use a decision tree to predict age of a client)

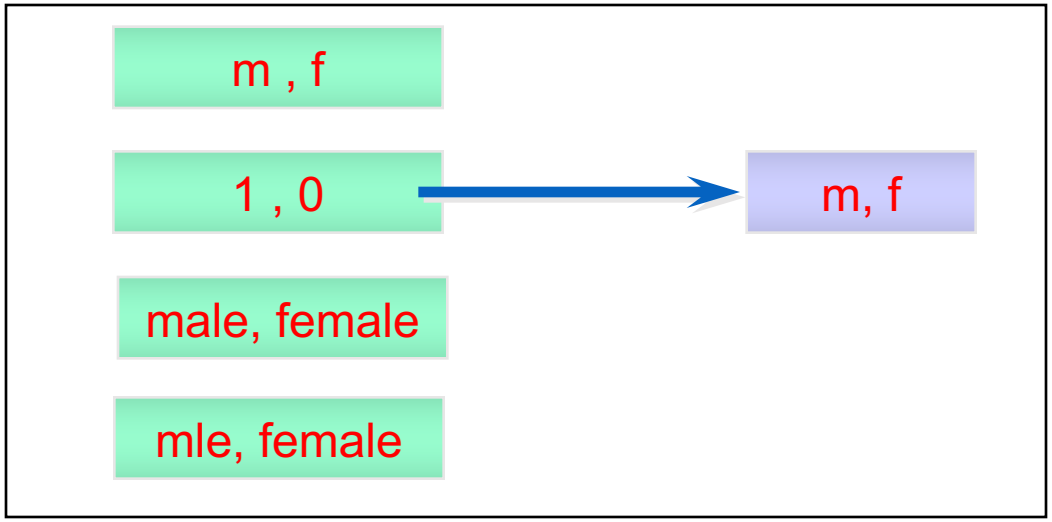
- May introduce BIAS into data
- May not be correct!

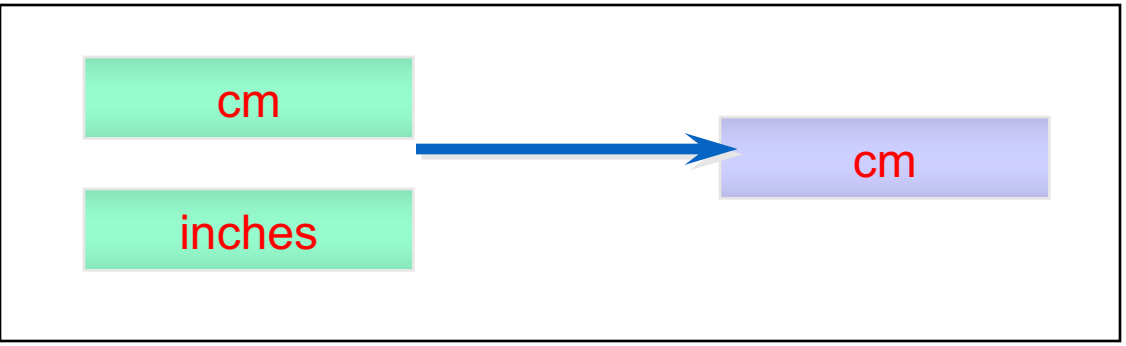
# Data Cleaning: Transform data

- Eliminate anomalies:
  - No unique key
  - Data naming, coding anomalies
  - Data meaning anomalies
  - Spelling and text inconsistencies

| CUSNUM | NAME | ADDRESS |
|---|---|---|
| 90328575 | Oracle Corp | 100 NE 1st Street, Tampa |
| 90328575 | Oracle | 100 NE. First St., Tampa |
| 90238475 | Oracle Services | 100 North East 1st St., FLA |
| 90233479 | Oracle Limited | 100 N.E. 1st St. |
| 90233489 | Oracle Computing | 15 Main Road, Ft. Lauderdale |
| 90234889 | Oracle Corp. UK | 15 Main Road, Ft. Lauderdale, FLA |
| 90345672 | Oracle Corp UK Ltd | 181 North Street, Key West, FLA |

# Data Cleaning: Multiple standards



# Data Cleaning: Noisy Data

- Noise: **random error or variance in a measured variable**
- Incorrect attribute values may due to
  - faulty data collection instruments
  - data entry problems
  - data transmission problems
  - inconsistency in naming convention

| SKU | Product | Brand | Supplier | Price |
|---|---|---|---|---|
| 12319319 | Milk | Natrel | Saputo | $5.99 |
| 12319336 | Milk | Quebon | Saputo | $5.49 |
| 12319353 | Milk | Grand Pre | Lactantia | $4.99 |
| 12319370 | Cream | Quebon | Saputo | $3.49 |
| 12319387 | Cream | Natrel | Saputo | $449.00 |
| 12319404 | Brie | Yellow | Metro | $7.99 |
| 12319421 | Brie | French | Metro | $6.49 |
| 12319438 | Cheddar | Trappe | Fromage | $6.99 |
| 12319455 | Gouda | Trappe | Fromage | $0.19 |

# Data Cleaning: Noise

Idea: Smooth out the noise from the data

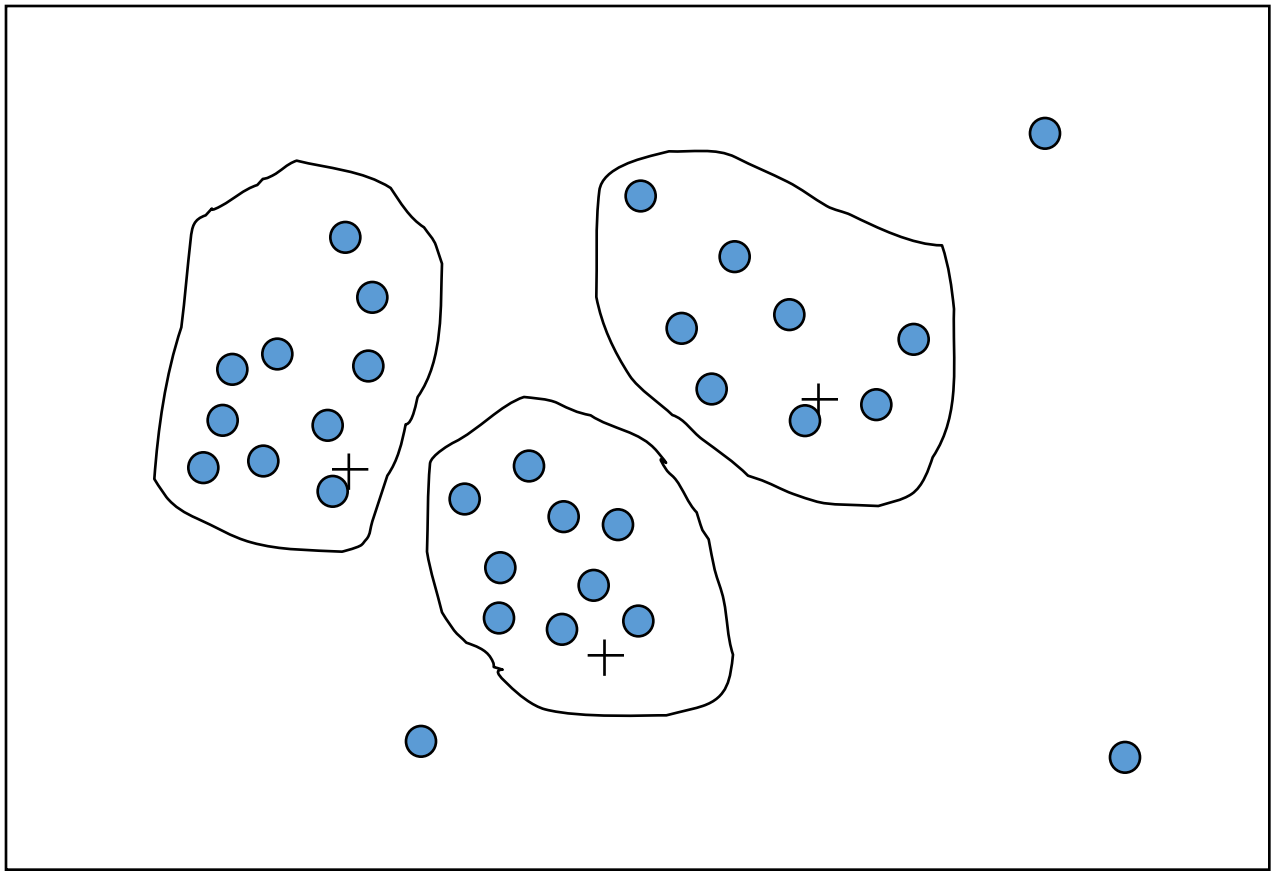
- Binning: place data in buckets or bins of neighbors
  - first sort data and partition into (equal-frequency) bins
  - then one can smooth by bin means, smooth by bin median, smooth by bin boundaries, etc.
- Regression: fit the data to a function using linear or multiple linear regression (more later)
- Clustering: useful for finding outliers (more later)
- Should always involve human inspection

# Binning Methods for Data Smoothing

- Sorted data for price (in dollars): 4, 8, 9, 15, 21, 21, 24, 25, 26, 28, 29, 34

* Partition into equal-frequency (equi-depth) bins:
  - Bin 1: 4, 8, 9, 15
  - Bin 2: 21, 21, 24, 25
  - Bin 3: 26, 28, 29, 34
* Smoothing by bin means:
  - Bin 1: 9, 9, 9, 9
  - Bin 2: 23, 23, 23, 23
  - Bin 3: 29, 29, 29, 29
* Smoothing by bin boundaries:
  - Bin 1: 4, 4, 4, 15
  - Bin 2: 21, 21, 25, 25
  - Bin 3: 26, 26, 26, 34

# Cluster Analysis: See the outliers



# Steps to transform the data
(more later)

1. Data cleaning
2. Data integration and transformation (later)
3. Data reduction (later)

# So, you have designed your data mart, loaded the historic data….

What is next, in terms of data staging?

# The Next Step…

C: Incremental table staging

# Step C1: Determining whether dimension records have been changed

Today's complete set
of dimensional records
Customer_id
Customer_type
Name
Address
Service_begin_date
….

Yesterday's complete set
of dimensional records
Customer_id
Customer_type
Name
Address
Service_begin_date
….

Compare difference &
Apply policy decisions

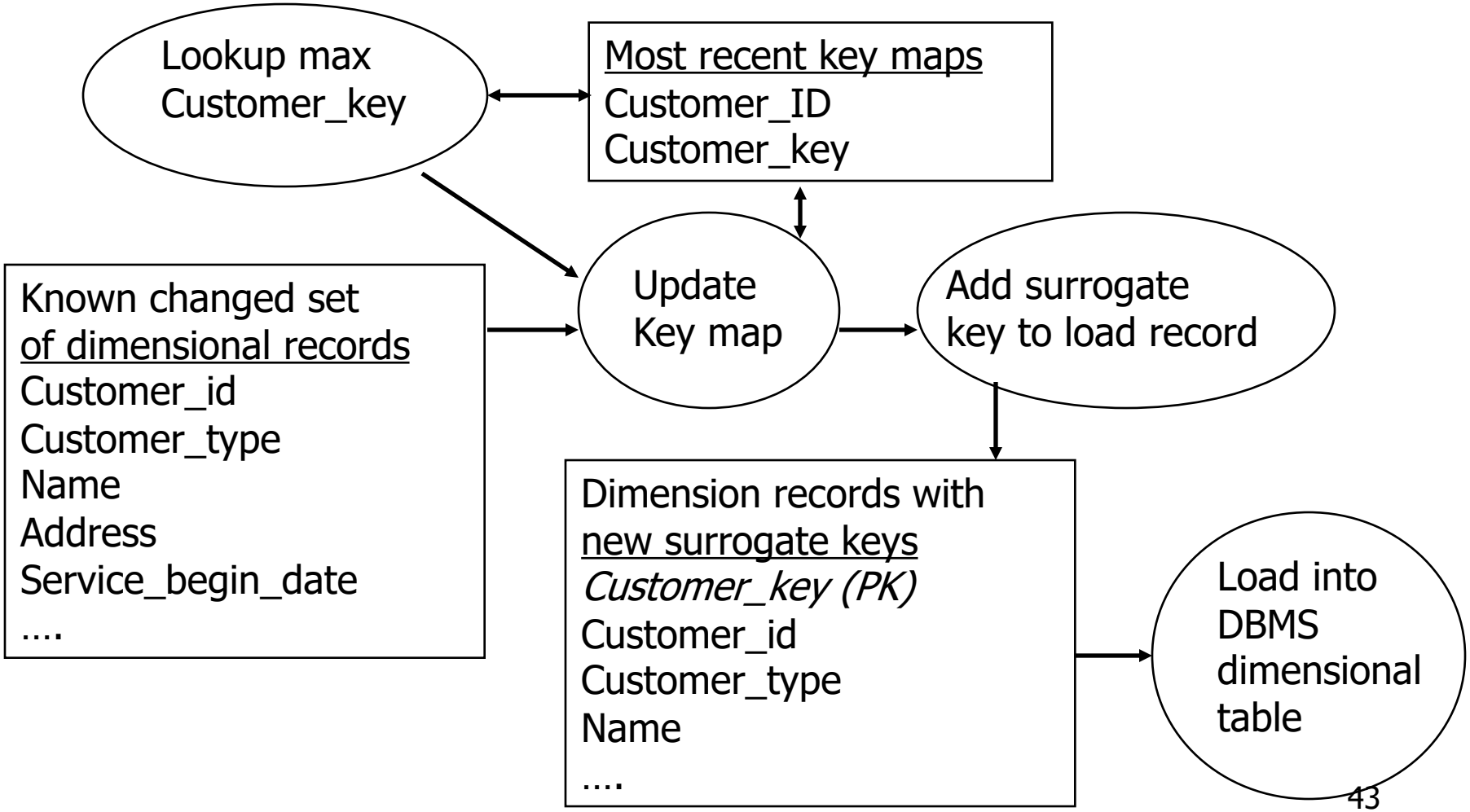
Known changed set
of dimensional records
Customer_id
Customer_type
Name
Address
Service_begin_date
….

# Handling change

Slowly changing dimensions

- Type 0: No change (e.g. Date-of-Birth)
- Type 1: Overwrite
- Type 2: Add new row
- Type 3: Keep history (add new attribute)
- Type 4: Add history table/dimension

# Step C1: Handling changed dimensions



# Handling Change: Type 1 overwrite

- Often caused by data capturing errors

| Cust-key | Name | Age | City | Marital-Status |
|---|---|---|---|---|
| 122 | Ann | 20 | Ottawa | Single |

| Cust-key | Name | Age | City | Marital-Status |
|---|---|---|---|---|
| 122 | Anne | 20 | Ottawa | Single |

# Handling Change: Type 2a

- Add new row

| Cust-key | Name | Age | City | Marital-Status |
|---|---|---|---|---|
| 122 | ANN | 20 | Ottawa | Single |

- Suppose we currently have 2345 cust-keys in our mart

| Cust-key | Name | Age | City | Marital-Status |
|---|---|---|---|---|
| 2346 | ANN | 20 | Montreal | Single |

- From today, Ann is linked to the FACT using cust-key 2346

# Handling Change: Type 2b

- Add new row

| Cust-key | Name | Age | City | Marital-Status |
|---|---|---|---|---|
| 122 | ANN | 20 | Ottawa | Single |

- Suppose we currently have 2345 cust-keys in our mart

| Cust-key | Name | Age | City | Marital-Status | Current? |
|---|---|---|---|---|---|
| 122 | ANN | 20 | Ottawa | Single | No |
| 2346 | ANN | 20 | Montreal | Single | Yes |

- From today, Ann is linked to the FACT using cust-key 2346

# Handling Change: Type 2c

- Add new row

| Cust-key | Name | Age | City | Marital-Status |
|---|---|---|---|---|
| 122 | ANN | 20 | Ottawa | Single |

- Suppose we currently have 2345 cust-keys in our mart

| Cust-key | Name | Age | City | Marital-Status | Effective-date |
|---|---|---|---|---|---|
| 122 | ANN | 20 | Ottawa | Single | 13/2/2002 |
| 2346 | ANN | 20 | Montreal | Single | 1/1/2018 |

- From today, Ann's record is linked to the FACT with cust-key 2346

# Handling Change: Type 3

- A new attribute is used to keep history

| Cust-key | Name | Age | City | Marital-Status |
|---|---|---|---|---|
| 122 | ANN | 20 | Ottawa | Single |

| Cust-key | Name | Age | City | Old-Marital-Status | Effective date | New-Marital-Status |
|---|---|---|---|---|---|---|
| 122 | ANN | 20 | Ottawa | Single | 14/02/2018 | Married |

# Handling Change: Type 4

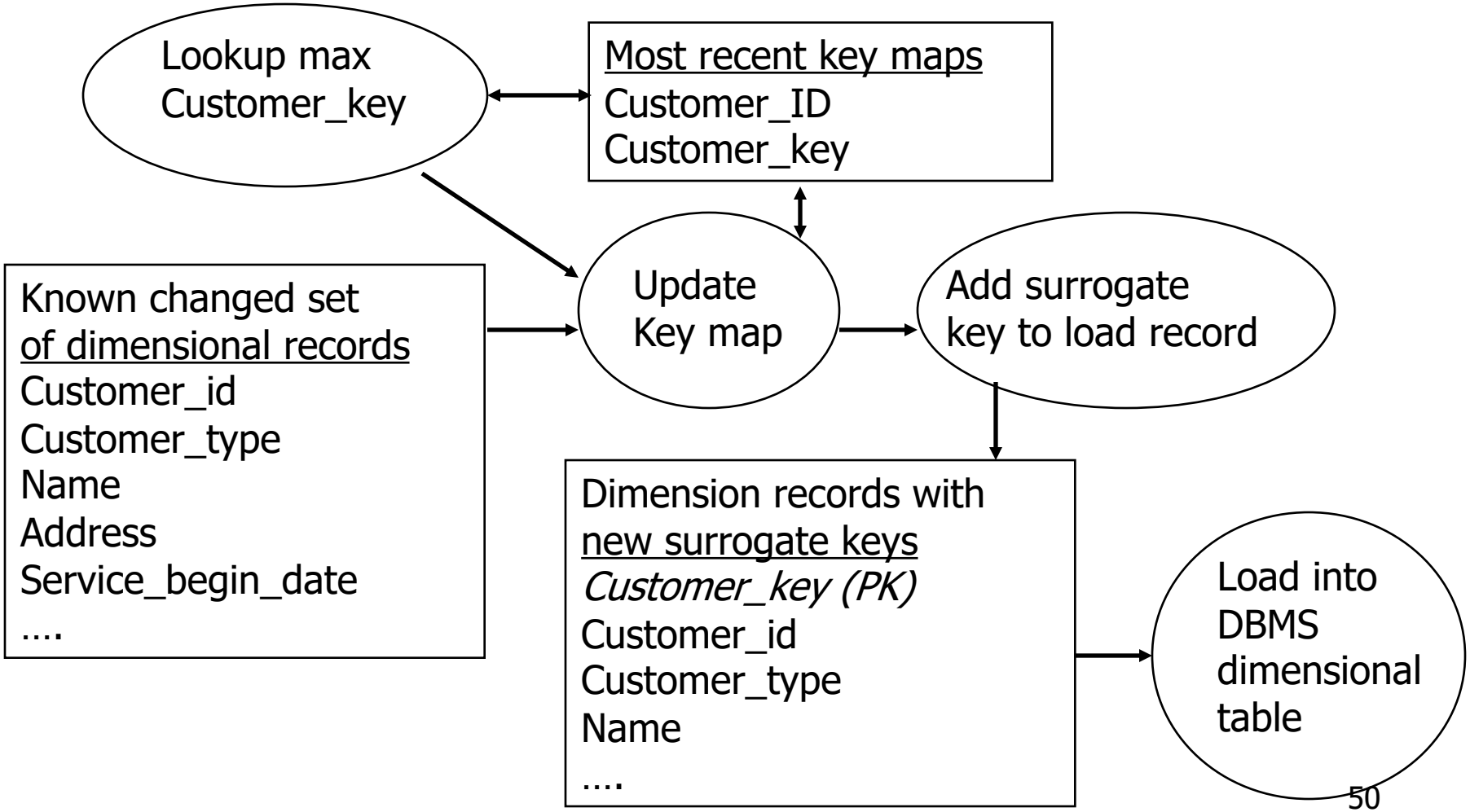
- Add another, new, separate "history" dimension
- Customer dimension has current data:

| Cust-key | Name | Age | City | Marital-Status |
|---|---|---|---|---|
| 2347 | ANN | 20 | Montreal | Married |

- "Customer-History" dimension keeps history:

| Cust-key | Name | Age | City | Marital-Status | Effective-date |
|---|---|---|---|---|---|
| 122 | ANN | 20 | Ottawa | Single | 13/2/2002 |
| 2346 | ANN | 20 | Montreal | Single | 1/1/2018 |
| 2347 | ANN | 20 | Montreal | Married | 14/2/2018 |

# Step C1: Handling changed dimensions



# Step C2: Incremental Fact Table Staging

- Incremental fact table extracts
  - New transactions
  - Updated transactions (correcting info)
  - Database logs
  - Replication
- Incremental fact table load
- Speeding up the load cycle
  - More frequent loading
  - Partitioned files and indexes
  - Parallel processing

# Step D:
# Aggregate Table and OLAP Loads

- Build aggregates
- Maintain aggregates
- Prepare OLAP loads (if any)
  - Cube-like structure based on dimensional model
  - MOLAP engines build own optimized aggregates
    - Oracle Essbase
    - Microsoft Analysis Services
    - DB2 UDB OLAP

# The last data staging step: Automation


- Typical operational functions
  - Job definition: flow and dependency
  - Job scheduling: time and event based
  - Monitoring
  - Logging
  - Exception handling
  - Error handling
  - Notification
- Determine job control approach
- Record extract metadata
- Record operations metadata
- Ensure data quality
- Set up archiving in the data staging area
- Develop disk space management procedures

# Summary…

- Designing and building the data mart
  - Dimensional modeling
  - Aggregates and Indexes
  - Data staging

??

# Next…

Data  Analytics