INFORMATION TECHNOLOGY UNIVERSITY

DATA SCIENCE LAB

# Dimensional Modeling

## CS 537- Big Data Analytics

Dr. Faisal Kamiran

# Types of Fact Tables

- Transaction fact tables
- Periodic snapshot fact tables
- Accumulating snapshot fact tables
- Factless fact tables

# Types of Fact Tables

| Fact Table Type | Usage |
|---|---|
| Transaction | Record facts (measurements) from transactions |
| Periodic snapshot | Track a given measurement at regular intervals |
| Accumulating snapshot | Track the progress of business processes through formally defined stages |
| Factless | Record Occurrence of a transaction that has no measurements |

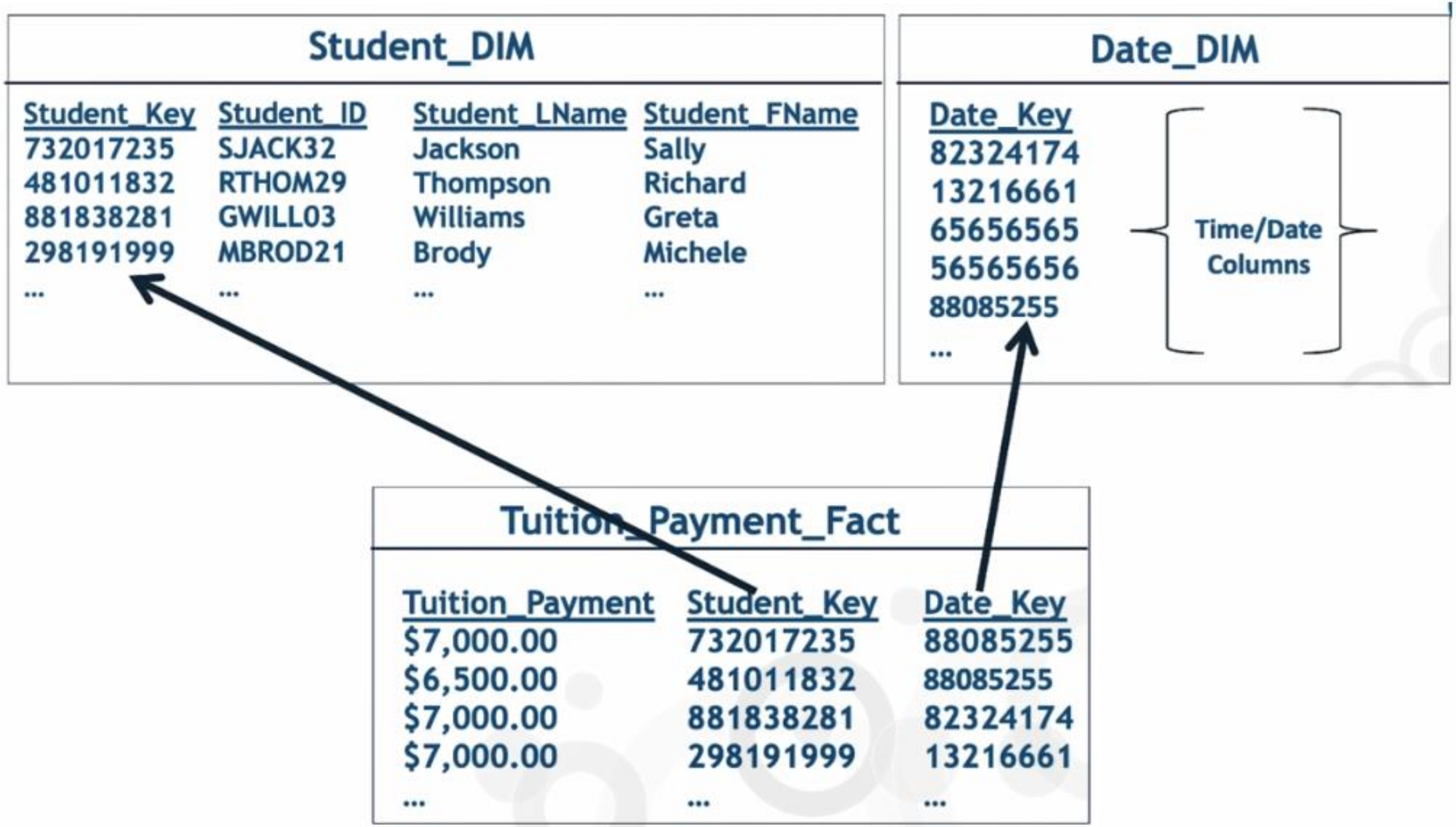
# Transaction Fact Table

- Each row in a transaction fact table corresponds to an event in space and time
- **Grain** is the individual transaction
- Literally: A table where we store our facts from transactions
- Each event is stored in the fact table only once
- Heart of dimensional models

# Transaction Fact Table

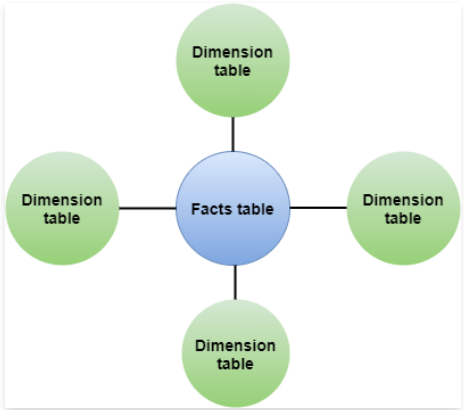
**Tuition_Payment_Fact**

| Tuition_Payment | Student_Key | Date_Key |
|---|---|---|
| $7,000.00 | 732017235 | 88085255 |
| $6,500.00 | 481011832 | 88085255 |
| $7,000.00 | 881838281 | 82324174 |
| $7,000.00 | 298191999 | 13216661 |
| ... | ... | ... |

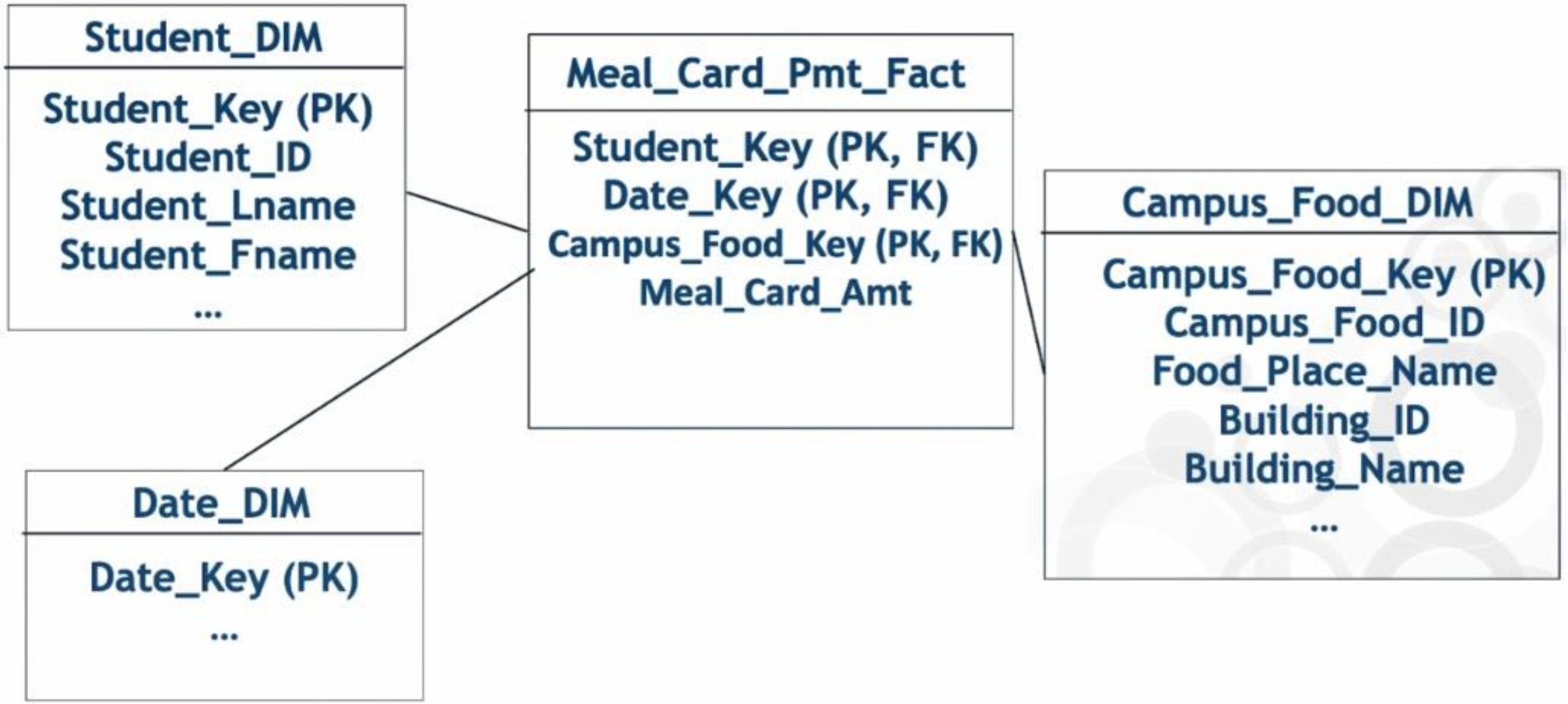

Student_DIM
Student_Key
732017235
481011832
881838281
298191999
...
Student_ID
SJACK32
RTHOM29
GWILL03
MBROD21
...
Student_LName
Jackson
Thompson
Williams
Brody
...
Student_FName
Sally
Richard
Greta
Michele
...
Date_DIM
Date_Key
82324174
13216661
65656565
56565656
88085255
...
Time/Date Columns
Tuition_Payment_Fact
Tuition_Payment
$7,000.00
$6,500.00
$7,000.00
$7,000.00
...
Student_Key
732017235
481011832
881838281
298191999
...
Date_Key
88085255
88085255
82324174
13216661
...

# Periodic Snapshot Fact Table

- Record aggregated measurements of transactions over a period
- Period may be a day, week or month etc.
- Grain is the period
- Transactions are recorded regularly at each cycle of the defined period



# Example: Recording Student Meal Card Payments

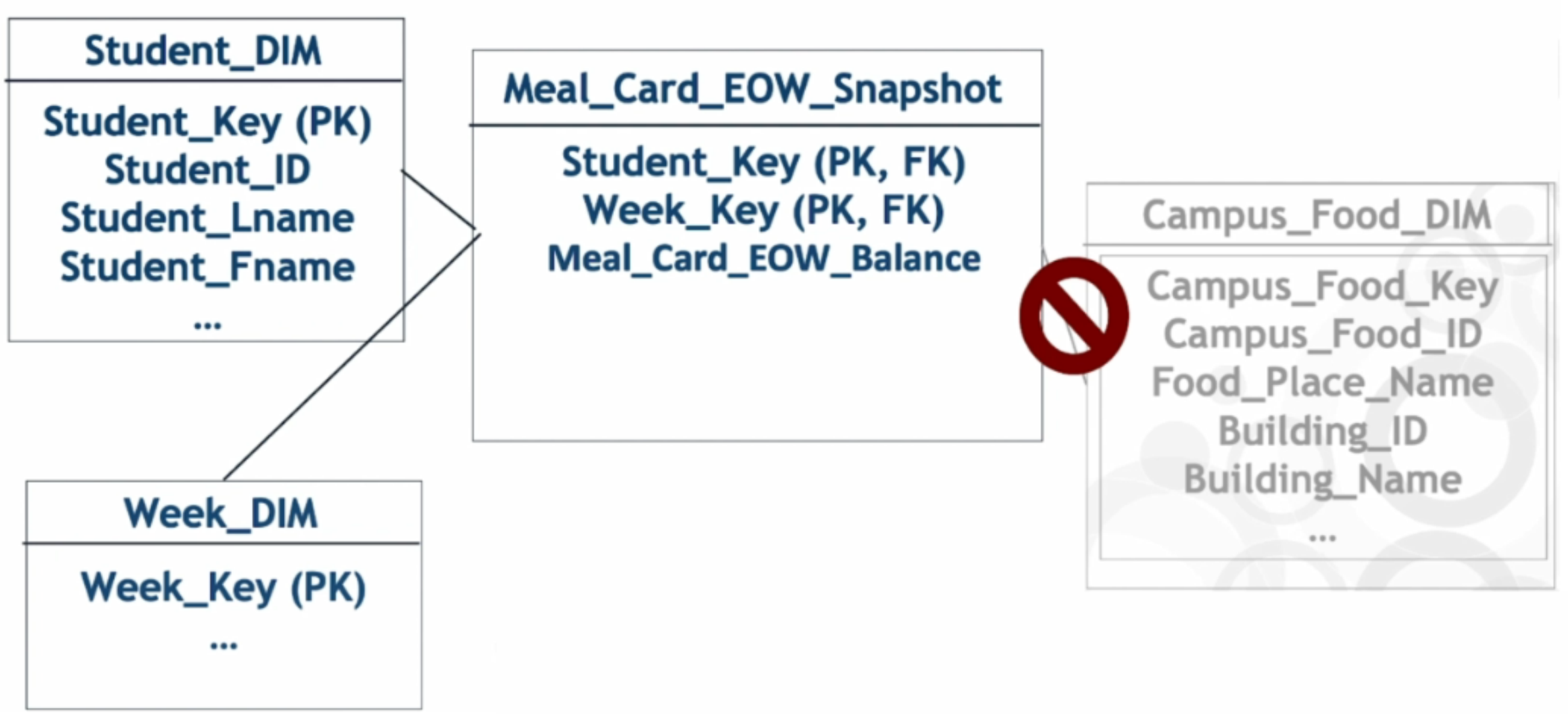


# Periodic Snapshot Fact Table

**Objective**

- Track and analyze **end-of-week** meal account balances throughout the semester
- For certain types of business questions, periodic snapshot fact tables are more easily structured and provide more direct access
- This can be done with the regular transaction grained fact table, but the analytic queries will be complex and compute intensive
- We will create a periodic snapshot fact table, which will summarize the transactions occurring each week

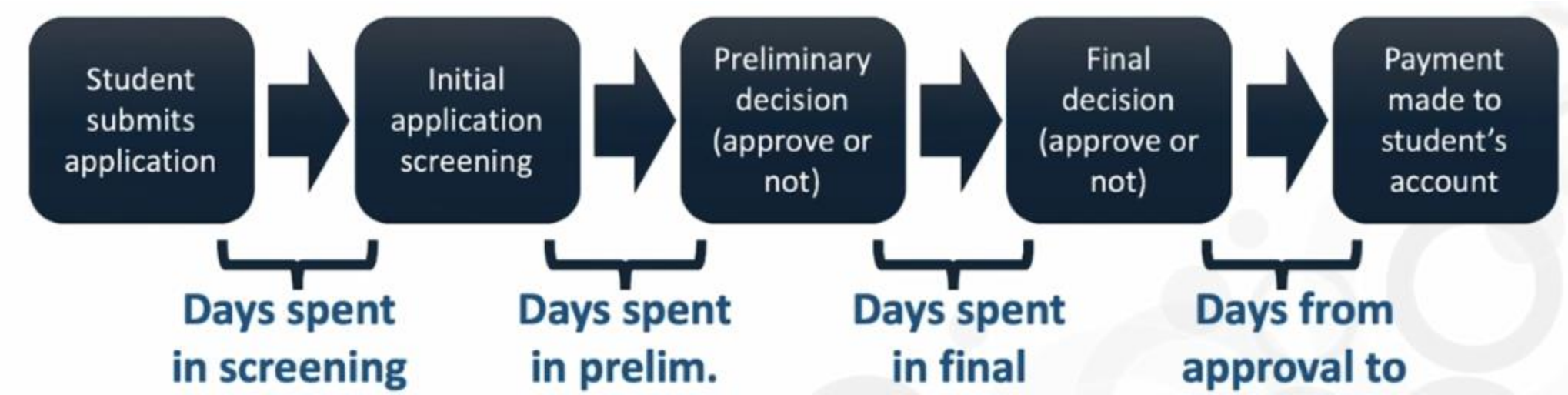
# Recording Student End of Week Meal Card Payments



# Accumulating Snapshot Fact Table

- Used to track the progress of a business process through formally defined stages
- Measure elapsed time spent in each phase
- Include both the completed and in-progress phases
- Can also track other measures (in addition to time) as process proceeds
- Introduces concept of relationships from a fact table back to a single dimension table

# Example: Student Financial Aid Application Process

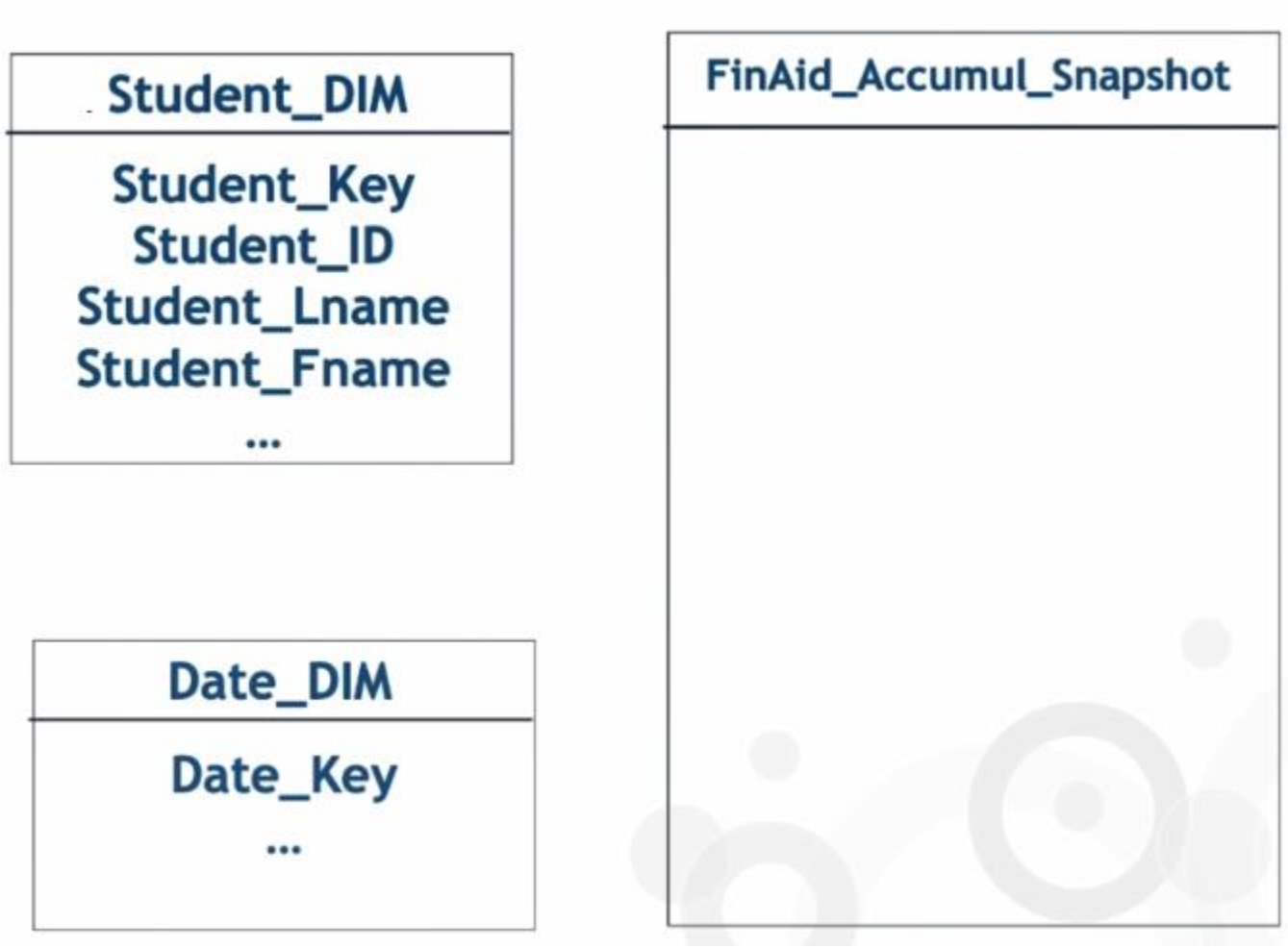


# Example: Student Financial Aid Application Process

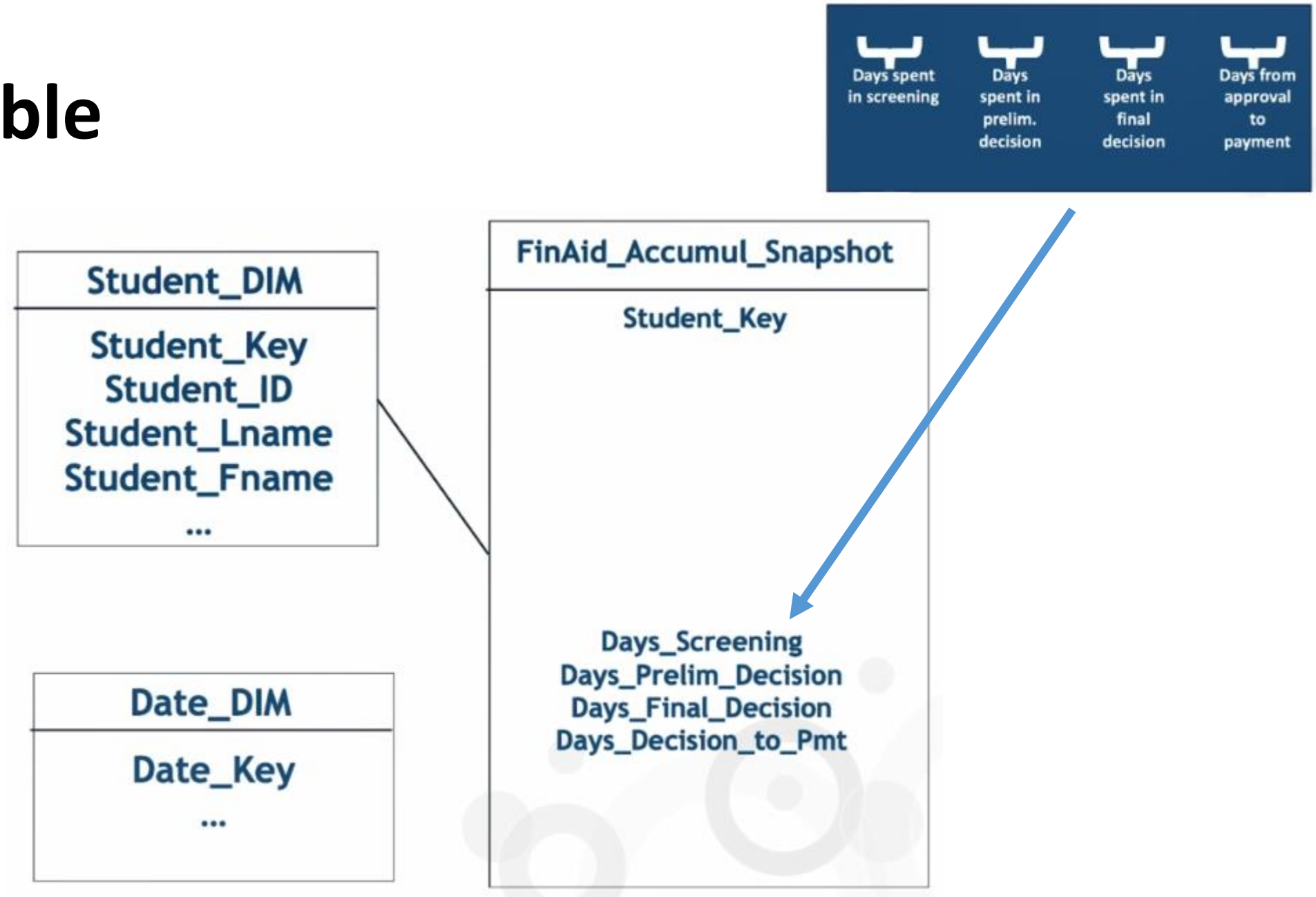
Dimensions for this process

- Student
- Time (Specific day a process begins)

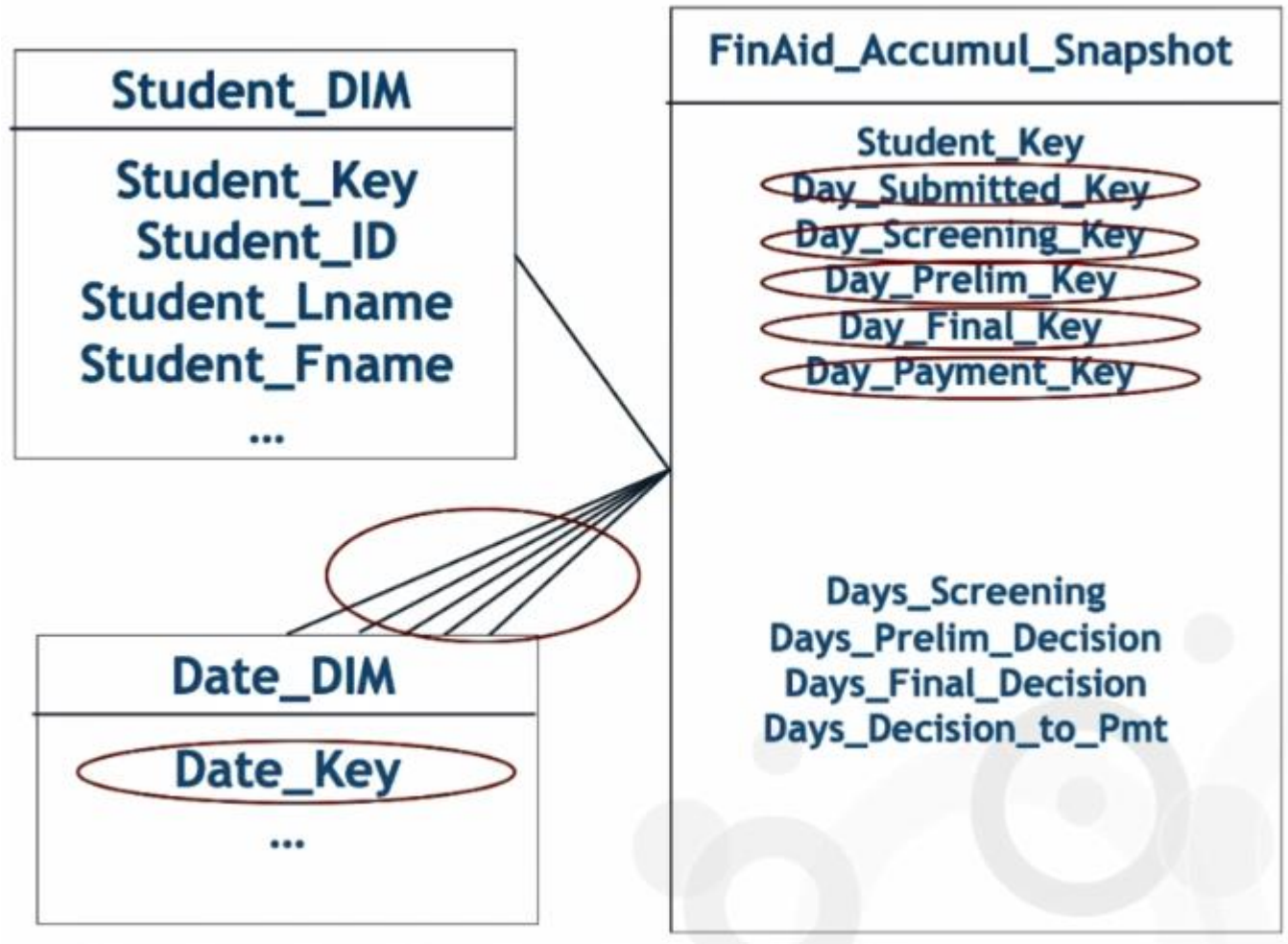
# Example: Fact and dimension tables



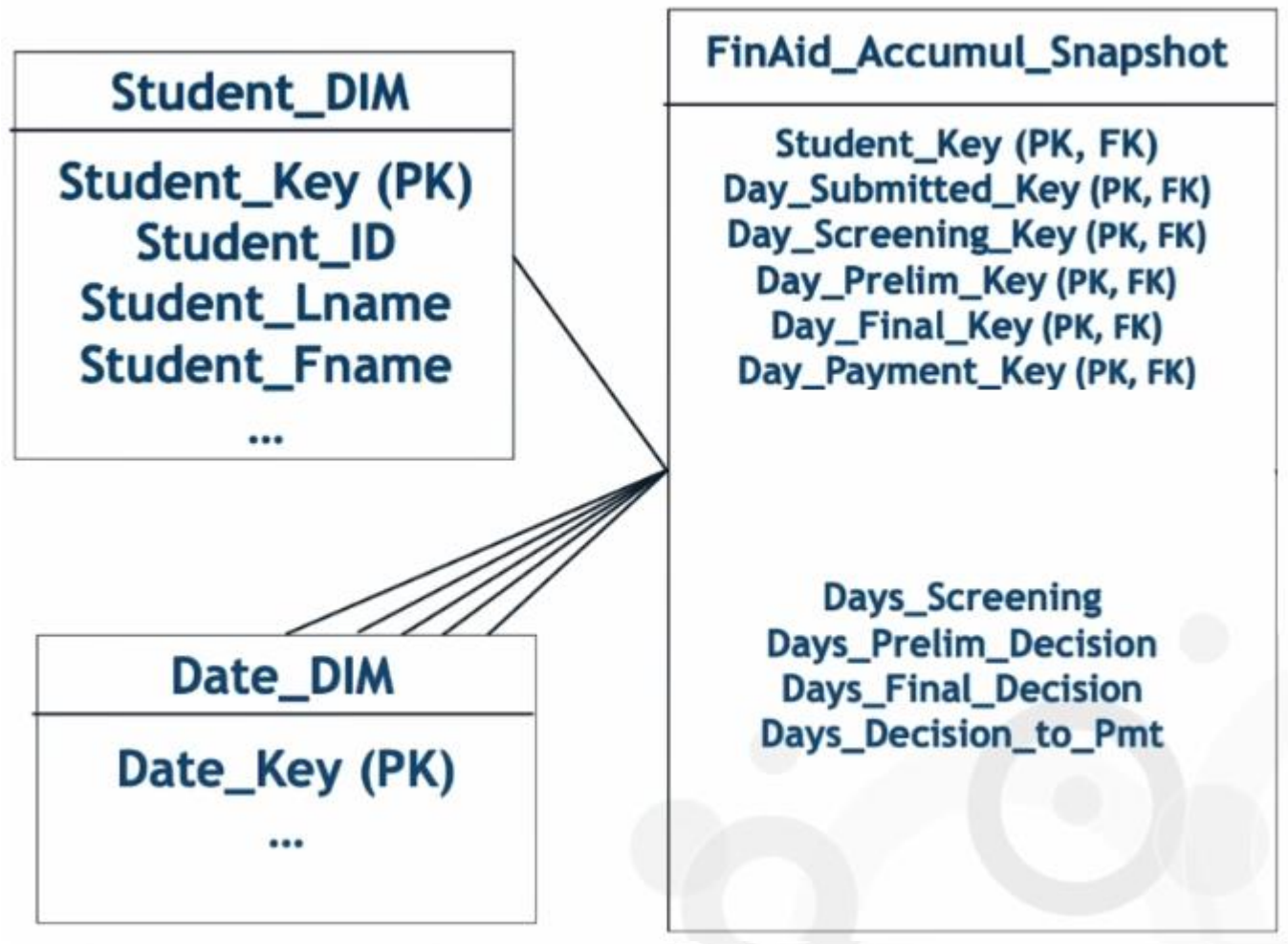
# Fact Table


Days spent in screening
Days spent in prelim. decision
Days spent in final decision
Days from approval to payment
Student_DIM
Student_Key
Student_ID
Student_Lname
Student_Fname
...
FinAid_Accumul_Snapshot
Student_Key
Days_Screening
Days_Prelim_Decision
Days_Final_Decision
Days_Decision_to_Pmt
Date_DIM
Date_Key
...

# Multiple links to data dimension



# Accumulating Snapshot Fact Table



# Factless Fact Table

Record the *occurrence* of events which have no numerical results associated with them.

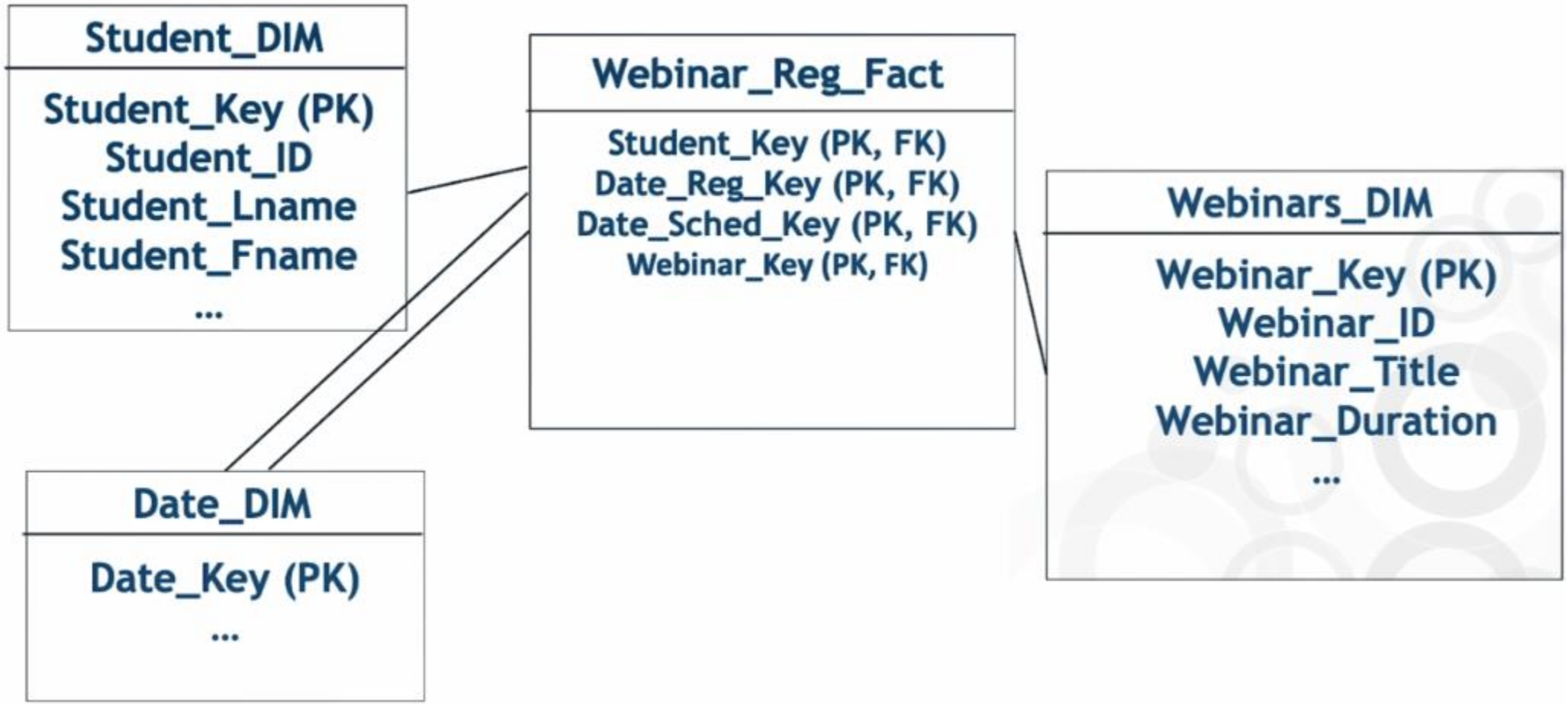
**Example**

"Students can register for online webinar throughout the semester"

We want to track:

- Which students register
- Which webinar they register for
- Date of registration
- Scheduled date of webinar

# Factless Fact Table



# Essential Rules of Dimensional Modeling

**(Recommended by Kimball)**

- Load detailed atomic data into dimensional structures
- Structure dimensional models around business processes.
- Ensure that every fact table has an associated date dimension table.
- Ensure that all facts in a single fact table are at the same grain or level of detail.

# Essential Rules of Dimensional Modeling

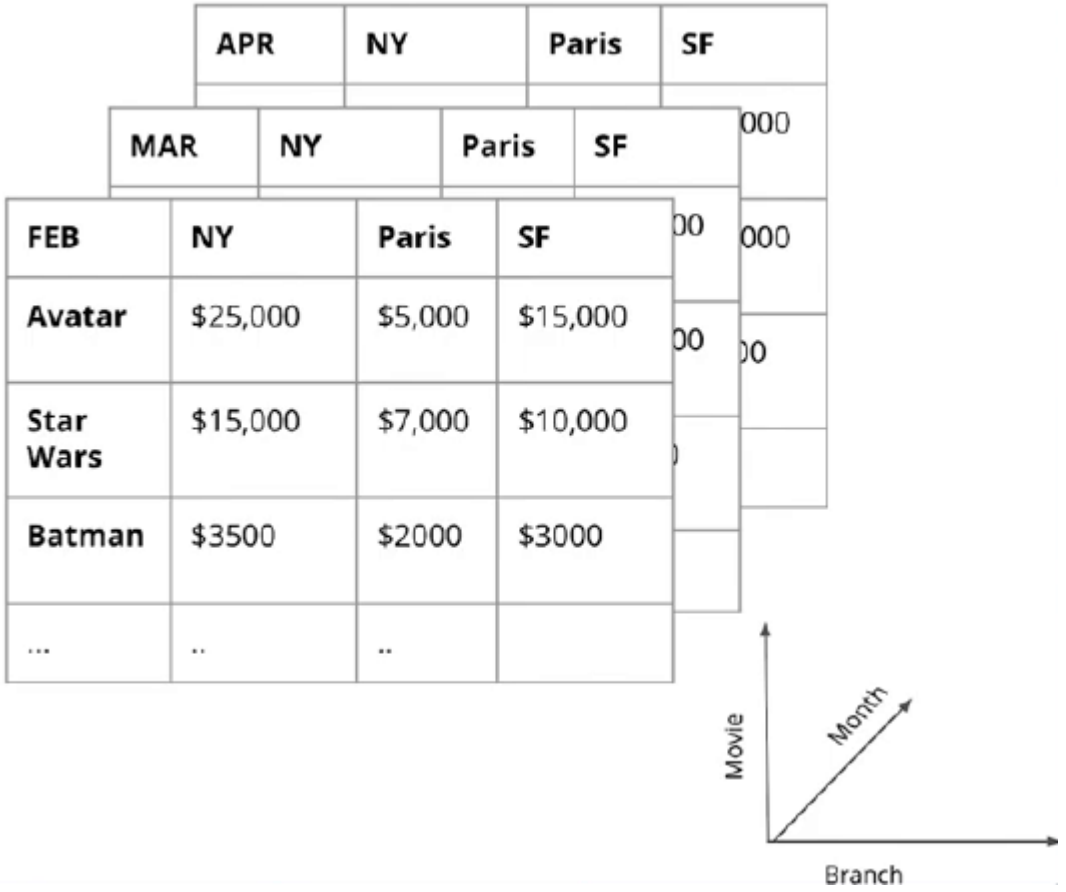
**(Recommended by Kimball)**

- Resolve many-to-many relationships in fact tables.
- Resolve many-to-one relationships in dimension tables.
- Make certain that dimension tables use a surrogate key.
- Create conformed dimensions to integrate data across the enterprise.
- Continuously balance requirements and realities to deliver a DW solution that is accepted by business users and that supports their decision-making.

# OLAP Cubes

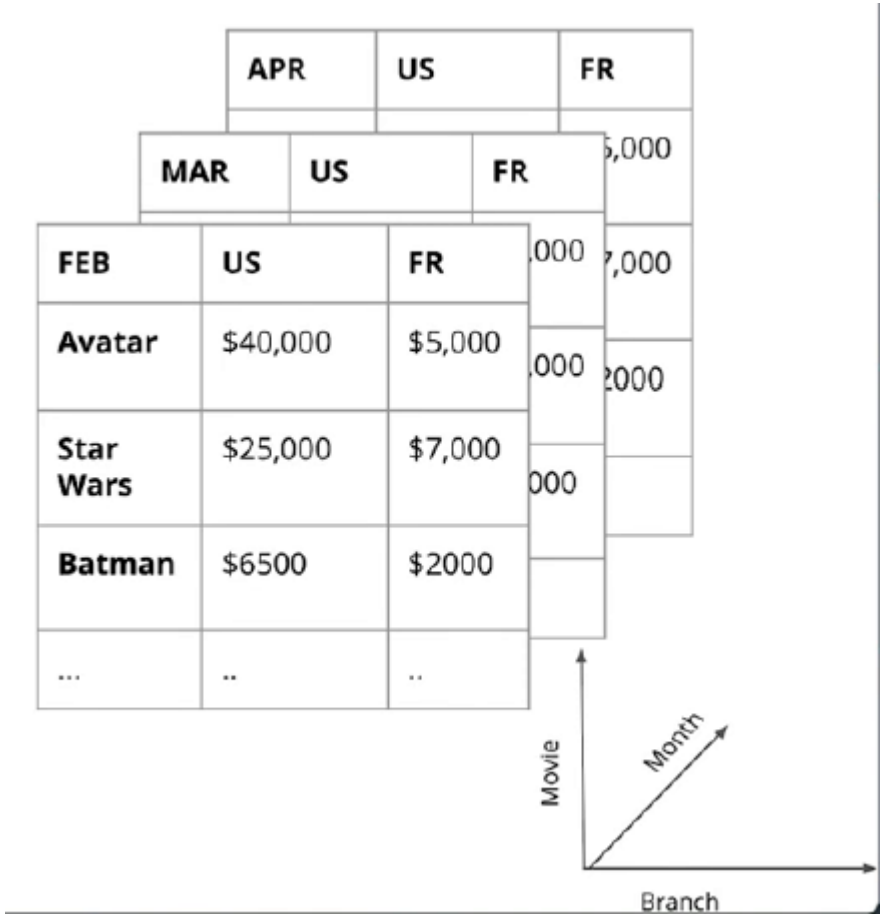
# OLAP Cubes

- An OLAP cube is an aggregation of a fact metric on a number of dimensions
- E.g. Movie, Branch, Month
- Easy to communicate to business users
- Common OLAP operations include: Rollup, drill-down, slice, & dice



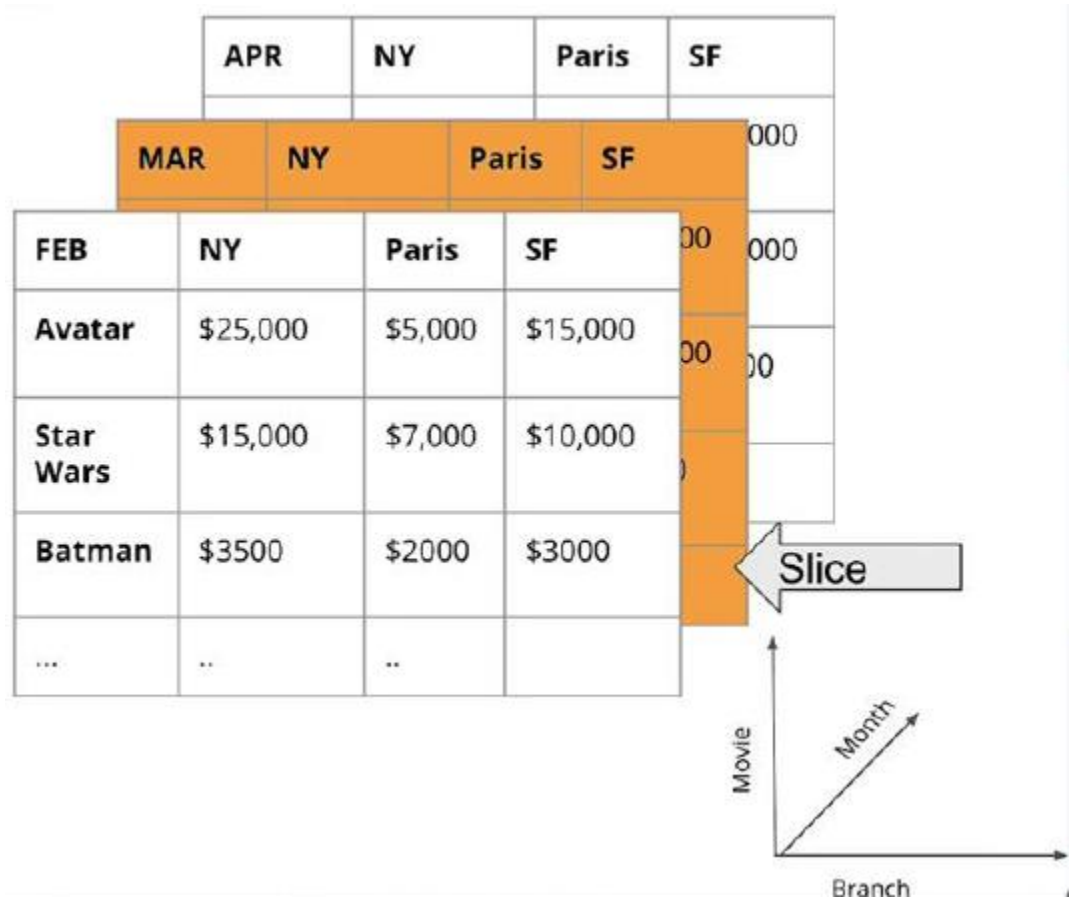
# OLAP Cubes Operations: Roll-up & Drill Down

- **Roll-up:** Sum up the sales of each city by Country: e.g. US, France (less columns in branch dimension)
- **Drill-Down:** Decompose the sales of each city into smaller districts (more columns in branch dimension)
- The OLAP cubes should store the finest grain of data (atomic data), in case we need to drill-down to the lowest level, e.g. Country -City - District - Street etc..



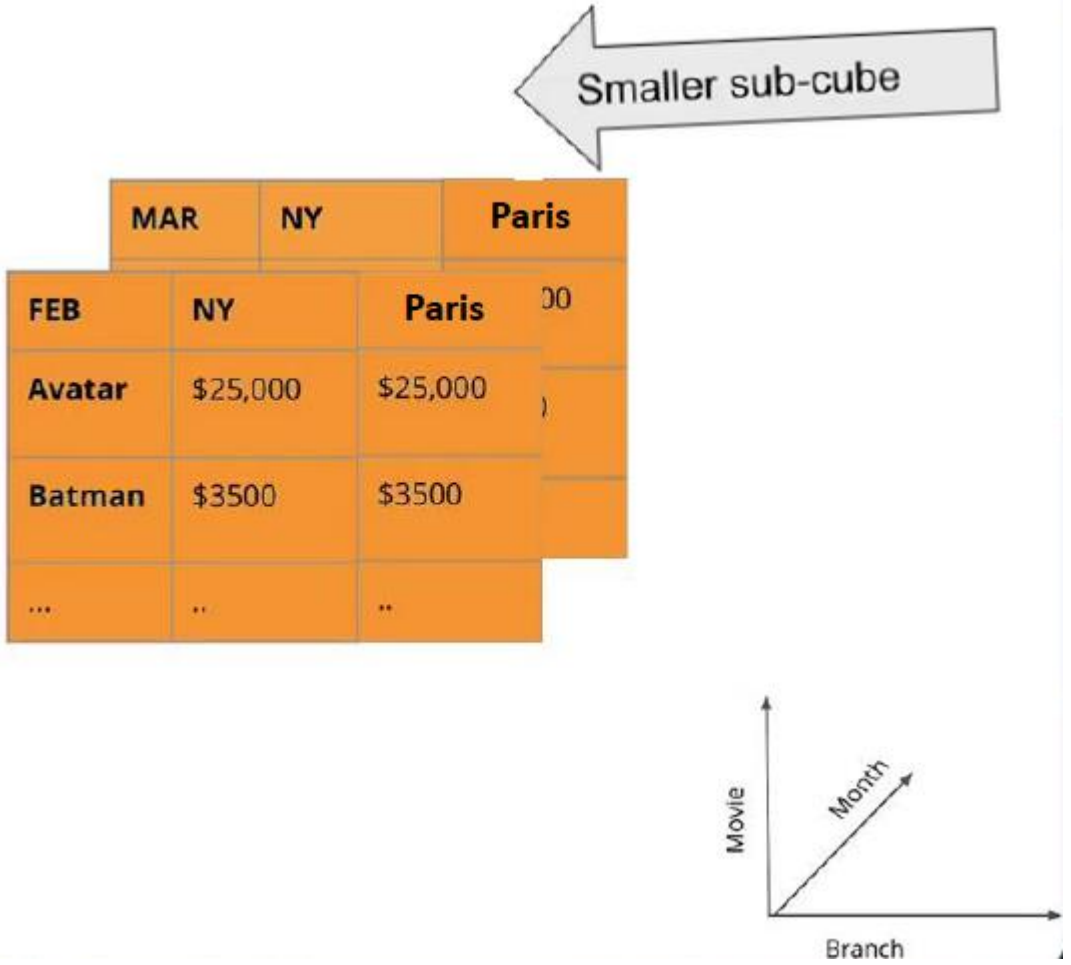
# OLAP Cubes Operations: Slice

- Reducing N dimensions to N-1 dimensions by restricting one dimension to a single value
- E.g. month='MAR'



# OLAP Cubes Operations: Dice

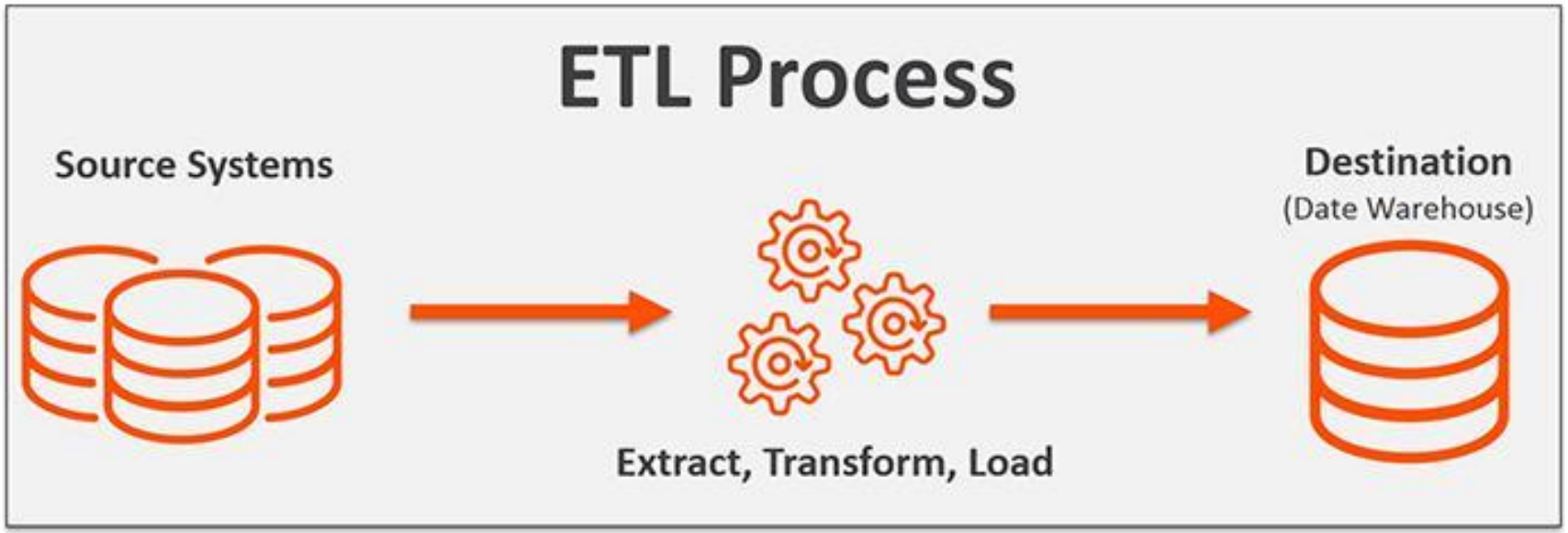
- Same dimensions but computing a sub-cube by restricting some of the values of the dimensions
- E.g. month in [‘FEB’, ‘MAR’], movie in [‘Avatar’, ‘Batman’] and branch in [‘NY’, ‘Paris’]



# Getting Data into the Data Warehouse

# ETL

- Data transferred from source applications to the DWH or Data Mart
- Done with a process called ETL



# ETL

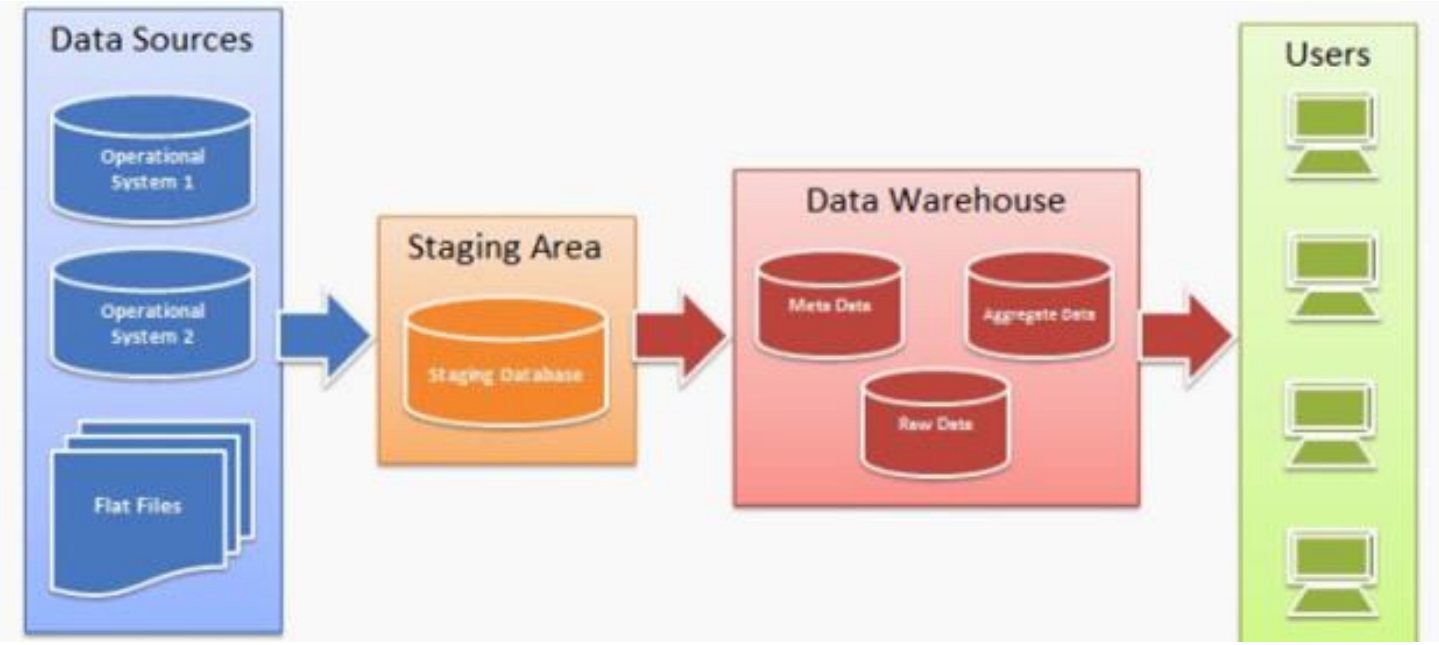
- Extract
- Transform
- Load

# Extract

- Pull data from **multiple** source systems
- Traditionally done in “batches” (can be hourly, weekly  etc.)
- Raw data is loaded including any existing errors
- Data transferred to a **staging area**

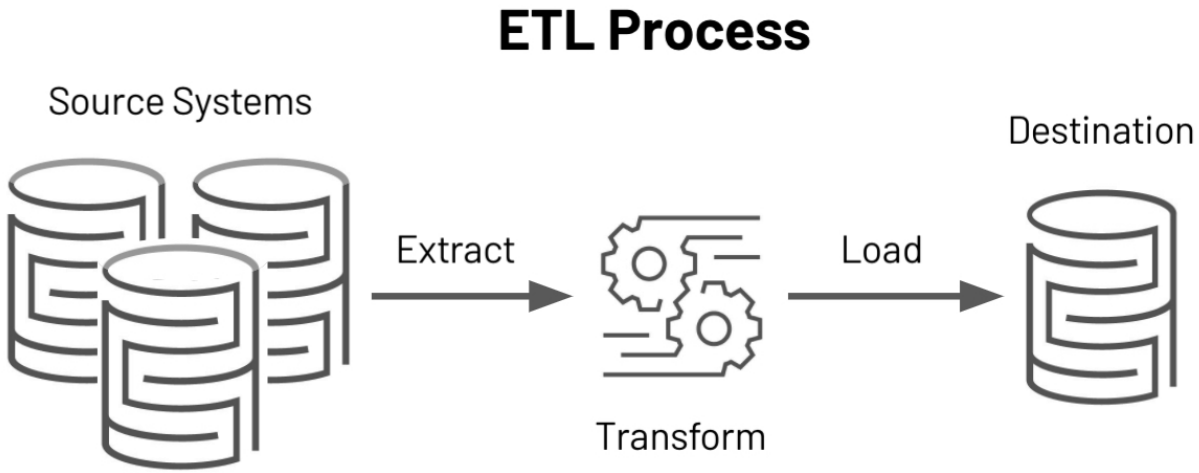
# Extract – Staging Area

- An intermediate storage area between the data sources and DWH
- The initial data is in different formats and may contain errors so it cannot be transferred directly to the DWH



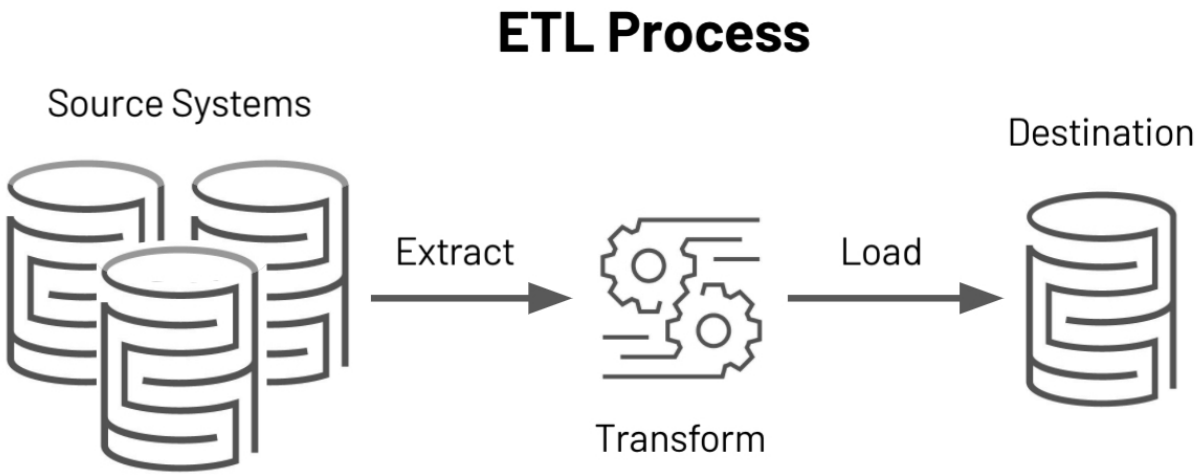
# Transform

- Convert data from multiple sources into a **uniform** format
- Performed on the extracted data in the staging area
- Transforms include
  - Cleaning
  - Filtering
  - Joining
  - Sorting
  - Splitting
  - Deduplication



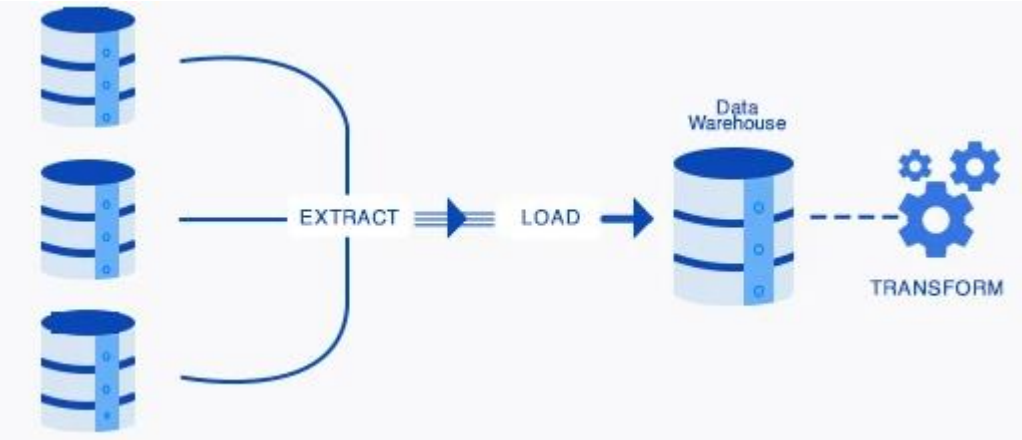
# Load

- Final stage in the ETL process
- Involves transferring data into the DWH



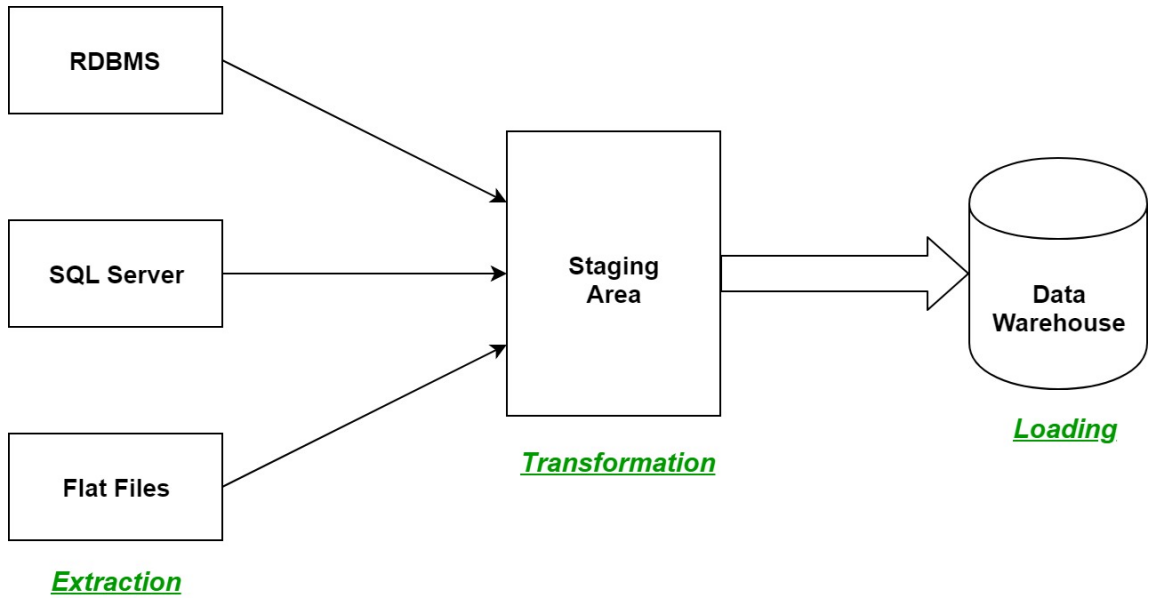
# ELT

- ELT – Extract, Load, Transform
- Raw data stored in Hadoop HDFS, AWS S3 etc.
- No staging area
- Use big data environment computing power to **transform when needed**
- Used in cases where massive amounts of data need to be ingested quickly



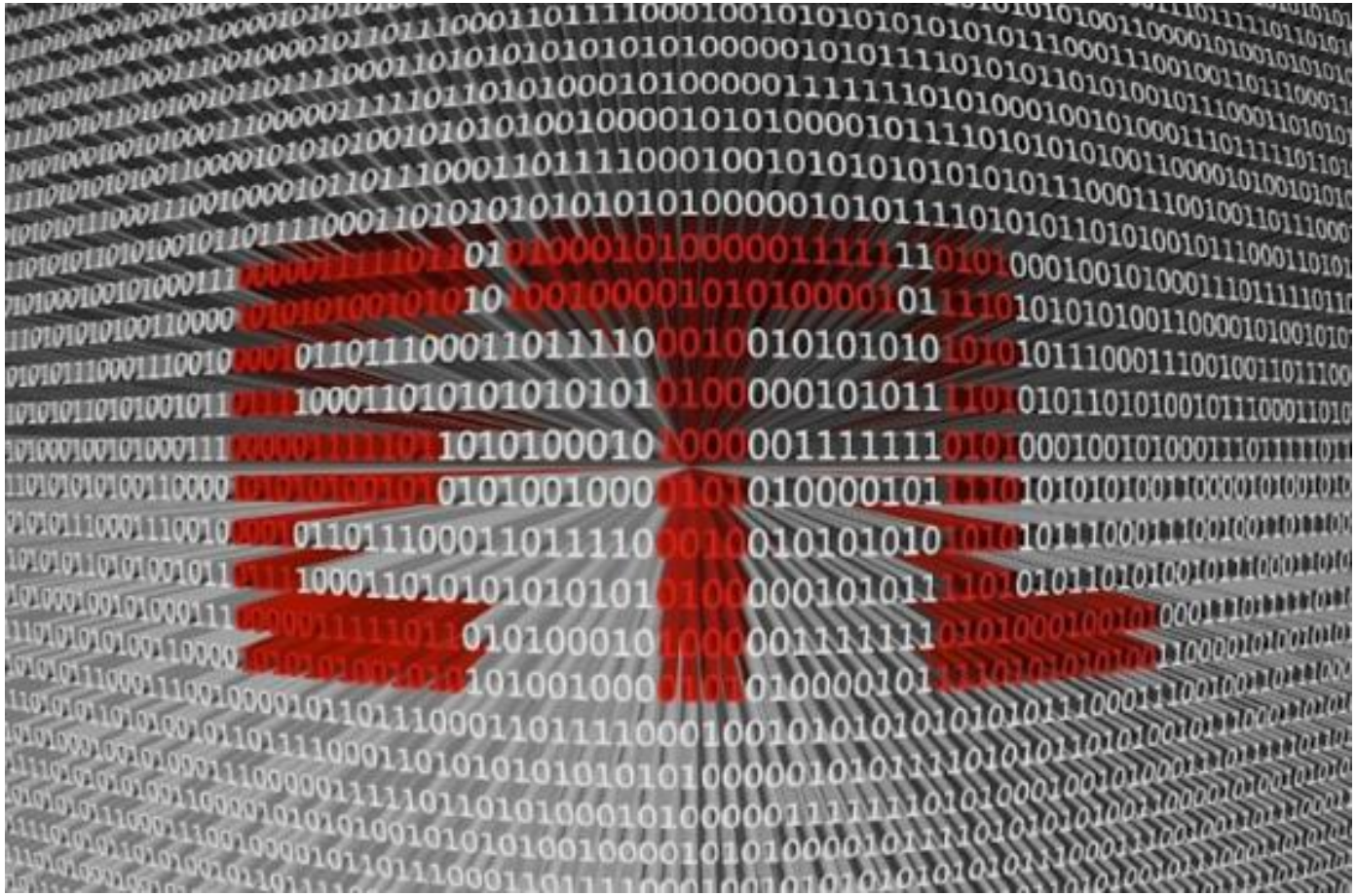
# ETL and Datawarehouse

- Data Warehouses work with relational SQL-like data structures
- Data must be transformed into a relational structure before it can be loaded into the Data Warehouse
- ETL used in Data Warehouses as **transformation must happen before loading**



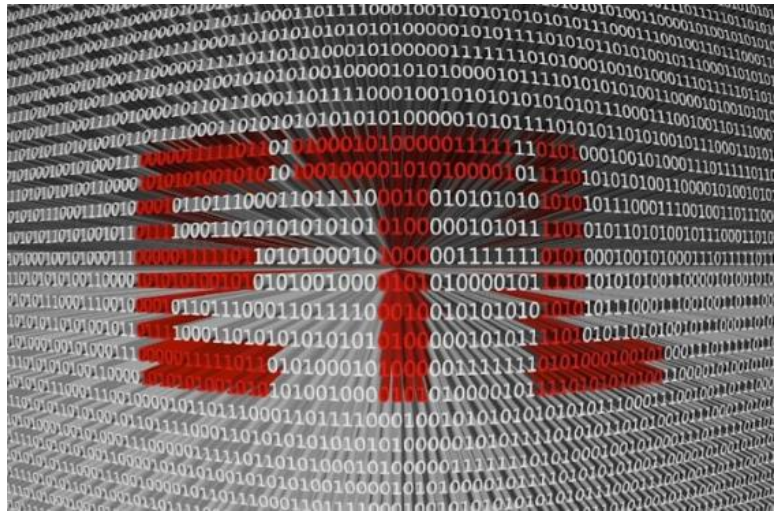
# Variations of ETL

- **Initial**
- **Incremental**



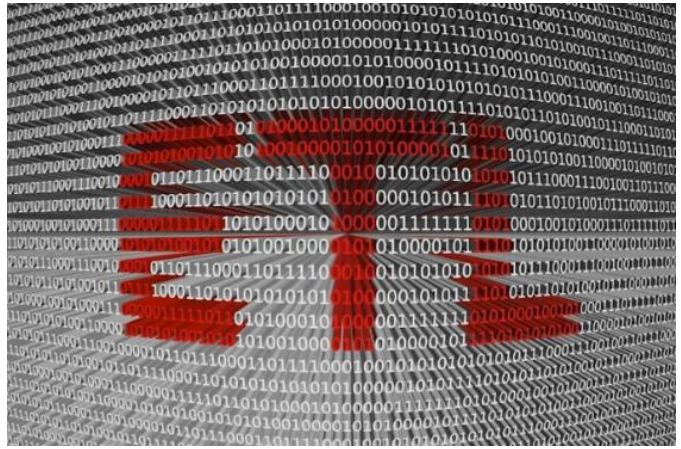
# Initial Load ETL

- Done right before the Data Warehouse goes live
- Normally one time only
- Load all **relevant** data necessary for Analytics
- Redo if Data Warehouse corrupted



# Incremental ETL

- Incrementally “refreshes” the data warehouse
- New data: new employees, products, …
- Modified data: employee promotions, product price change, …
- Deleted data: employee resigns, customer unsubscribes
- Load only updated data instances

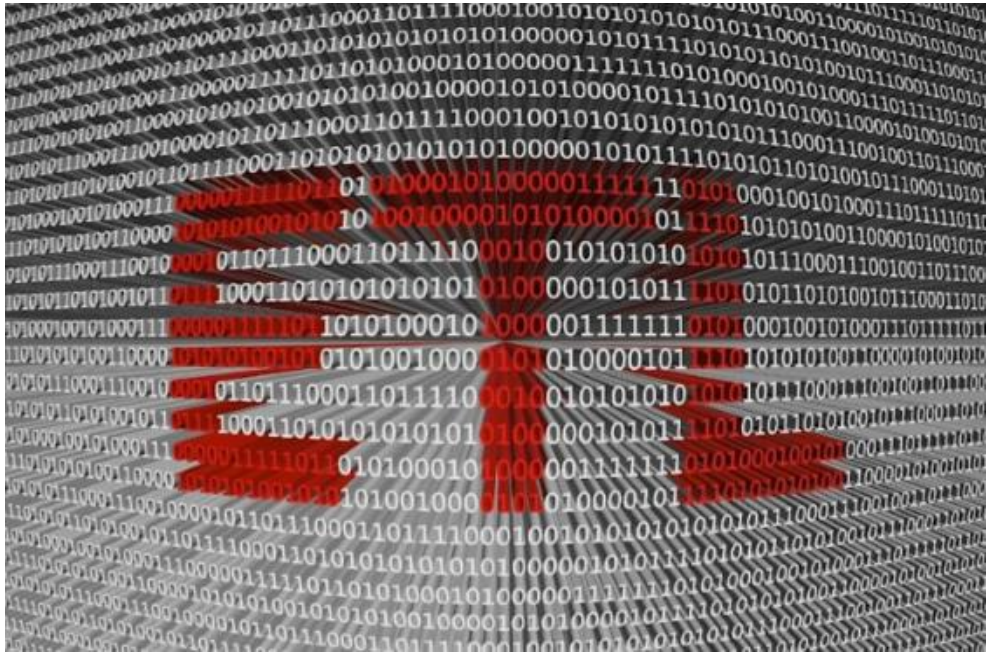


# Incremental ETL Patterns

Modern data warehouses use

- ✓ Append
- ✓ In-place Update

- *Complete replacement*
- *Rolling Append*



# Incremental ETL Patterns

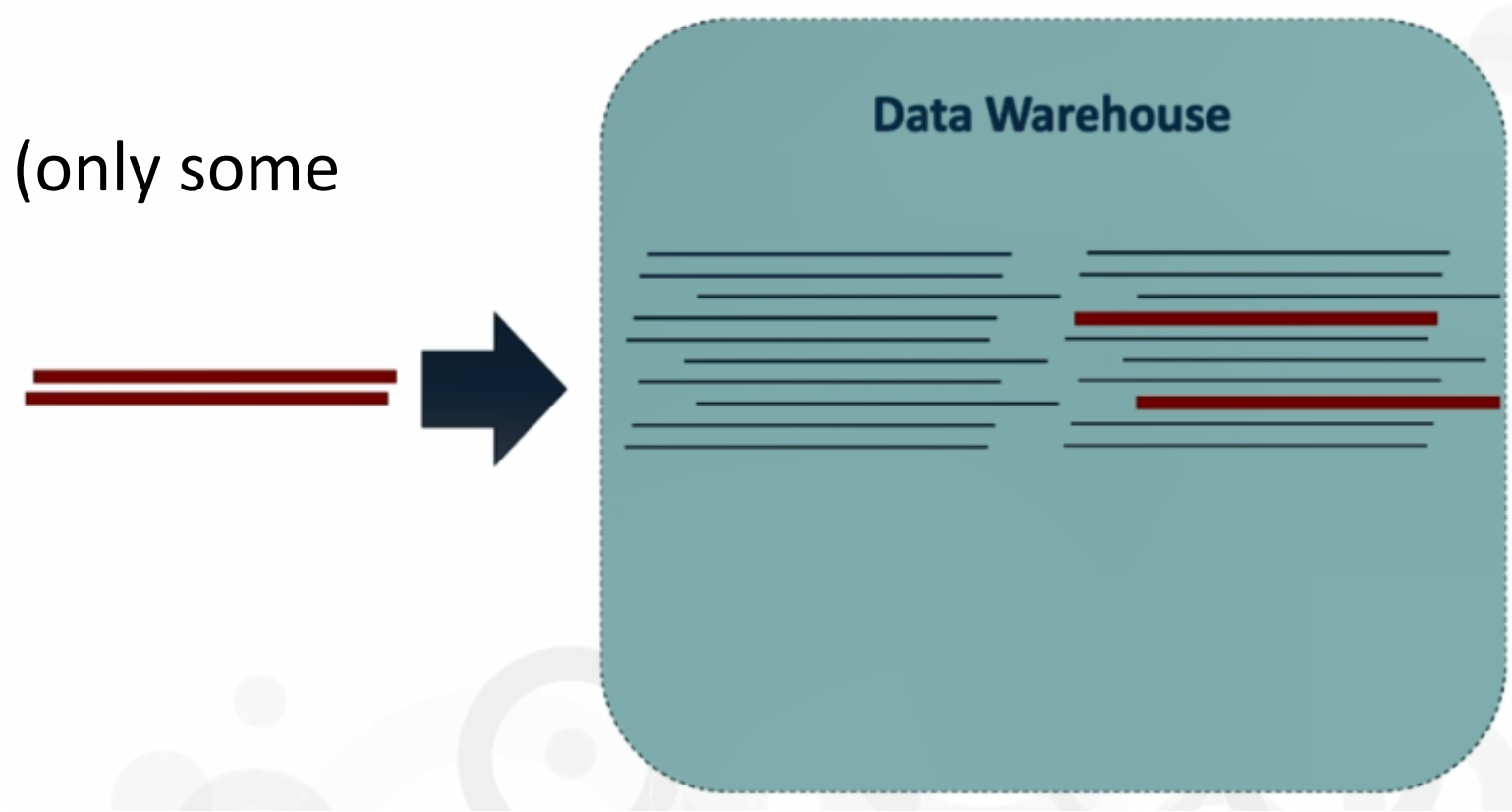
**Append**

- New data added at the end

# Incremental ETL Patterns

**In-place update**

- Modify existing data (only some rows)



# Incremental ETL Patterns

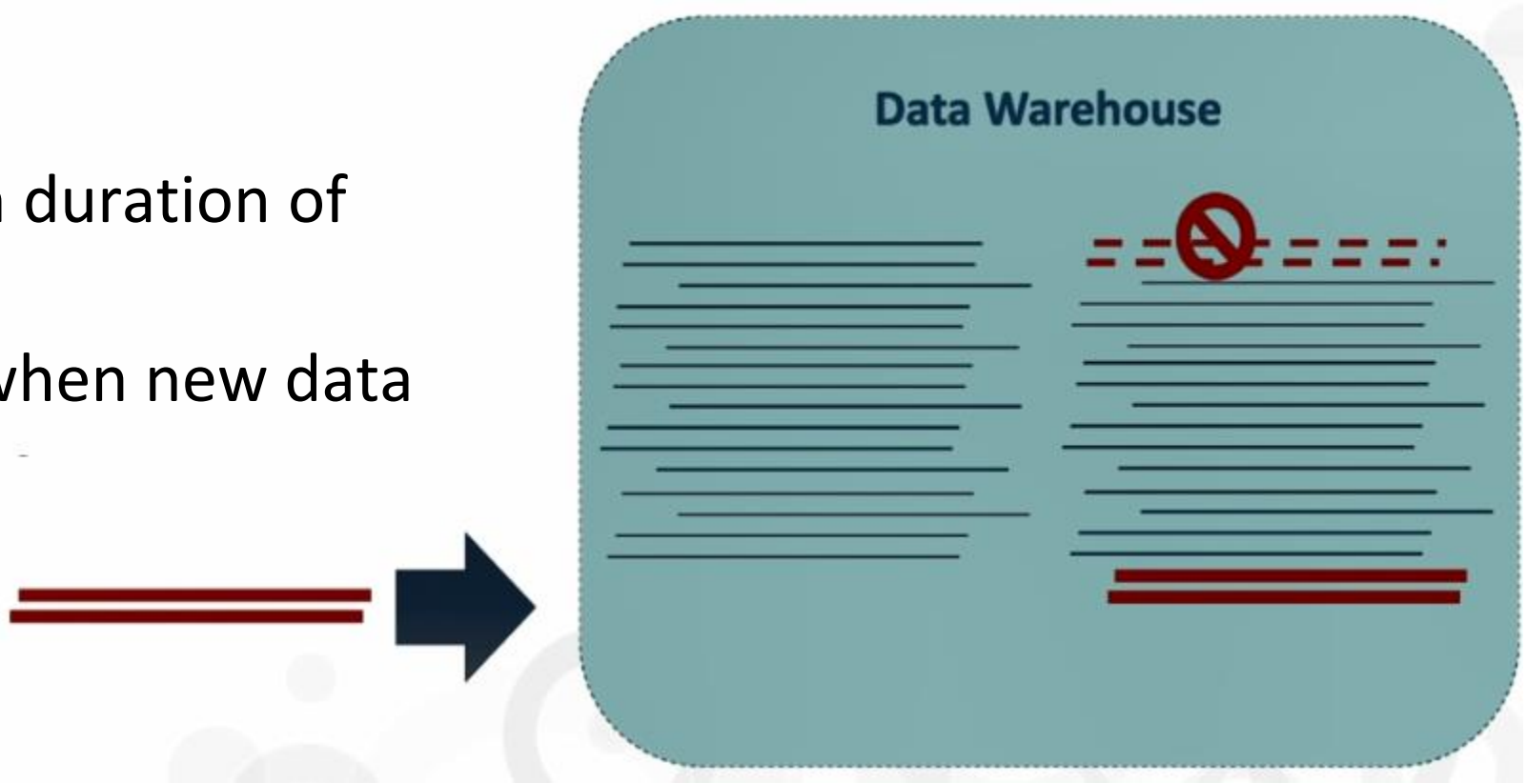
**Complete replacement**

- Overwrite existing data

# Incremental ETL Patterns

**Rolling Append**

- Maintain certain duration of history
- Wipe old data, when new data is appended



# References & Further Reading

- Kimball, The Data Warehouse Toolkit [LINK]
- Inmon, Building the Data Warehouse [LINK]
- Rainardi, Building a Data Warehouse [LINK]