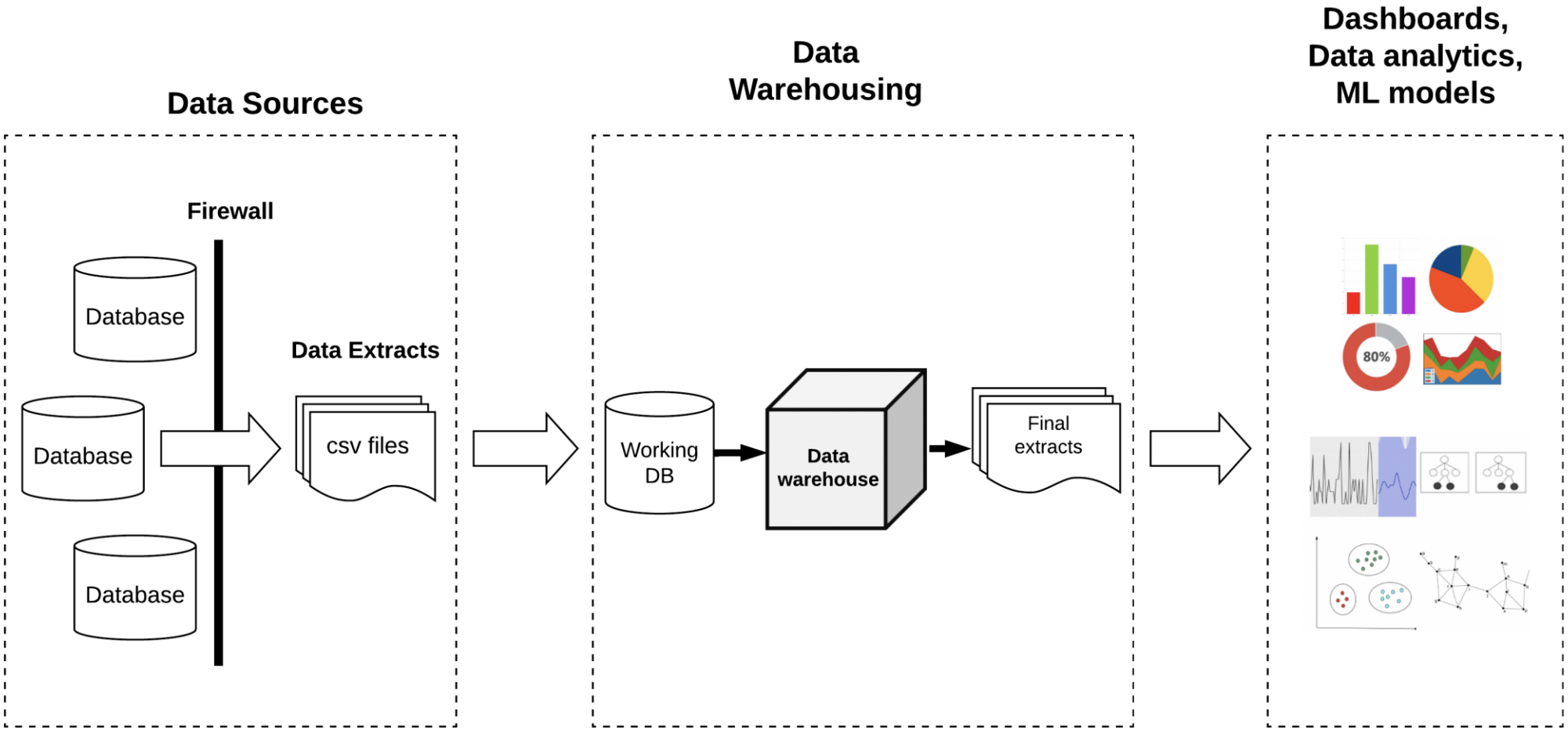
Data Warehousing:
Revision

Dr Isma Farah Siddiqui

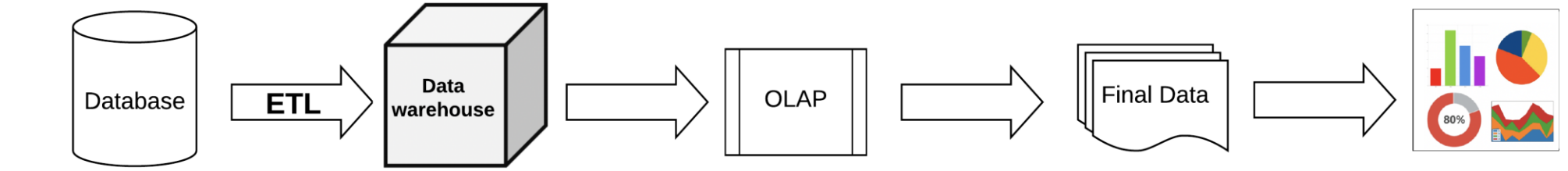
Data Engineering Framework



Data Engineering Architectures

Data Engineering Architectures

Model 1 – DW model (local databases)



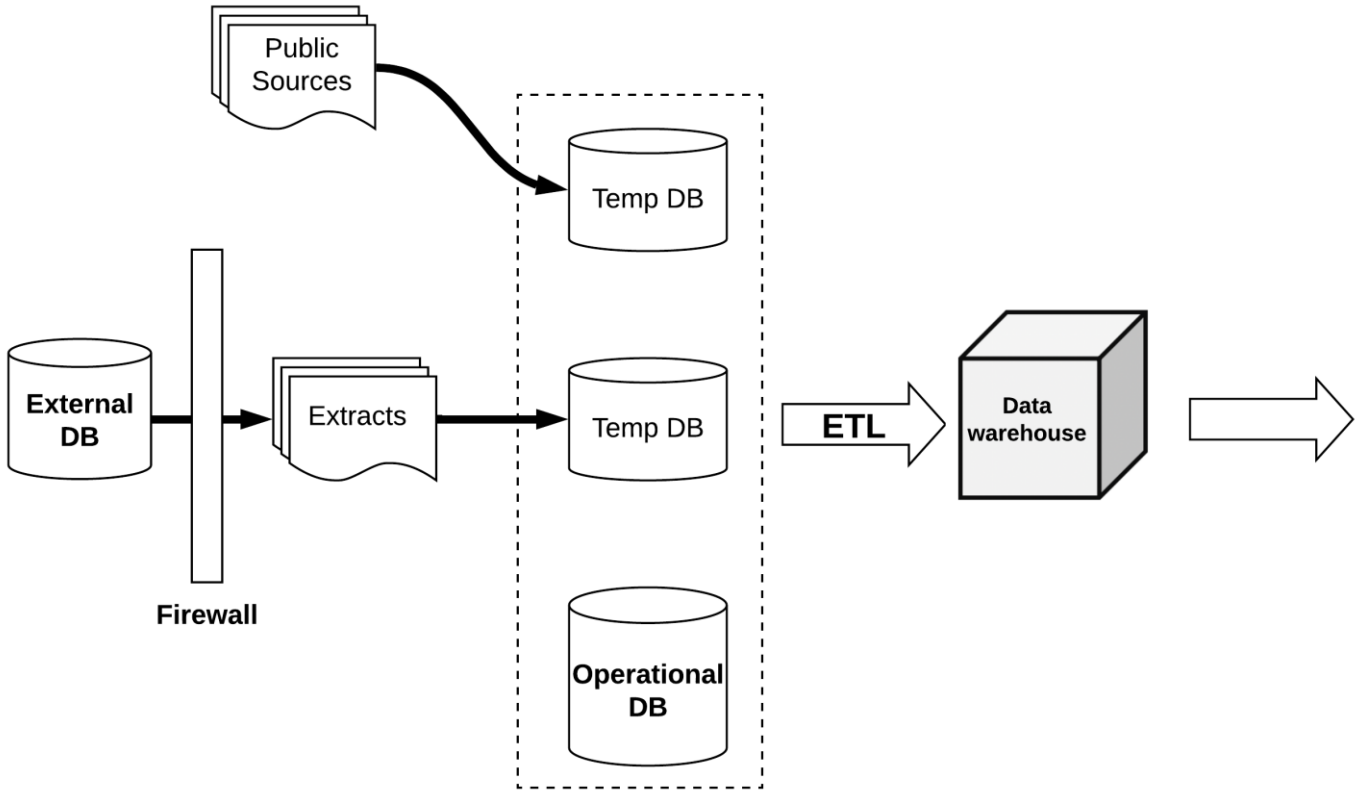
- Operational Databases
- Transactional Databases

- Data Cube
- Data Mart
- Extended Fact Table

- Dashboards
- ML models

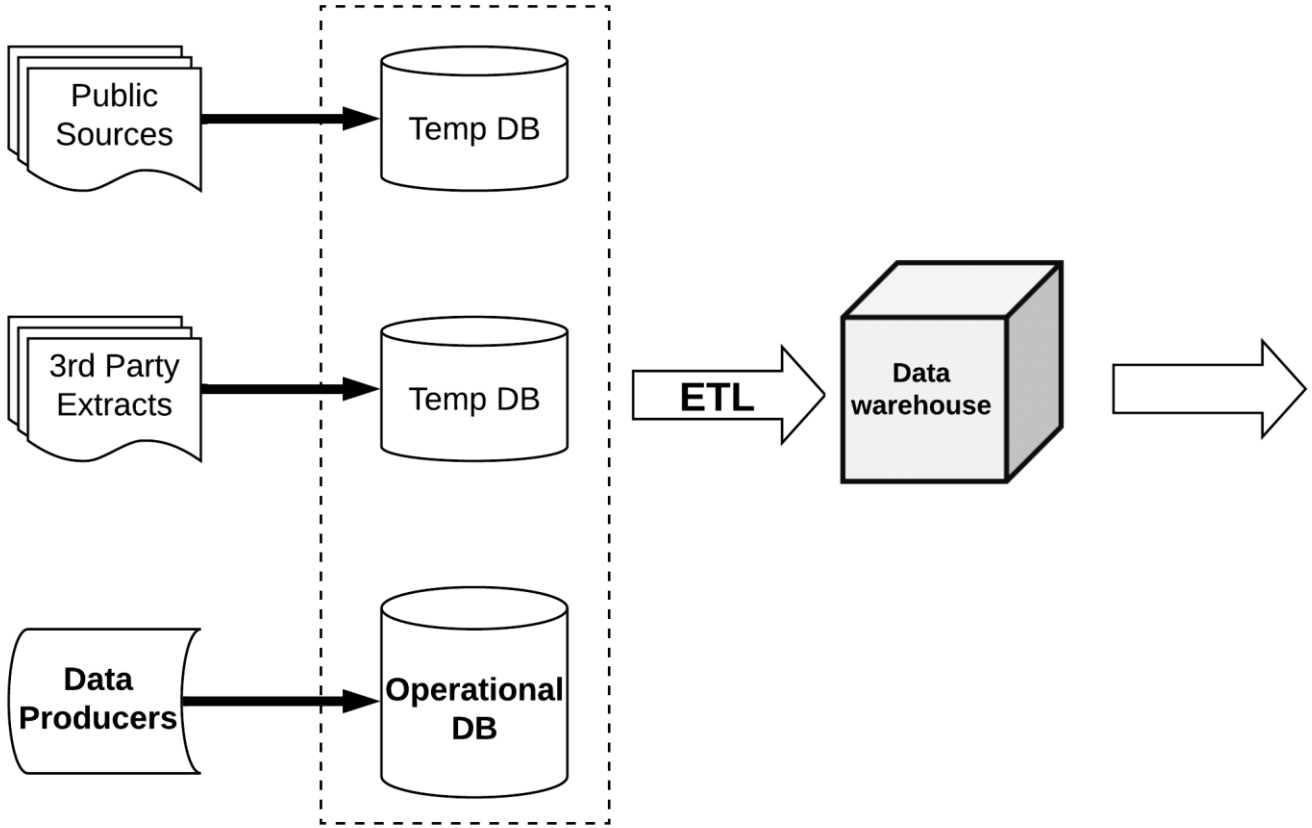
Data Engineering Architectures

Model 2 – DW model (external sources)



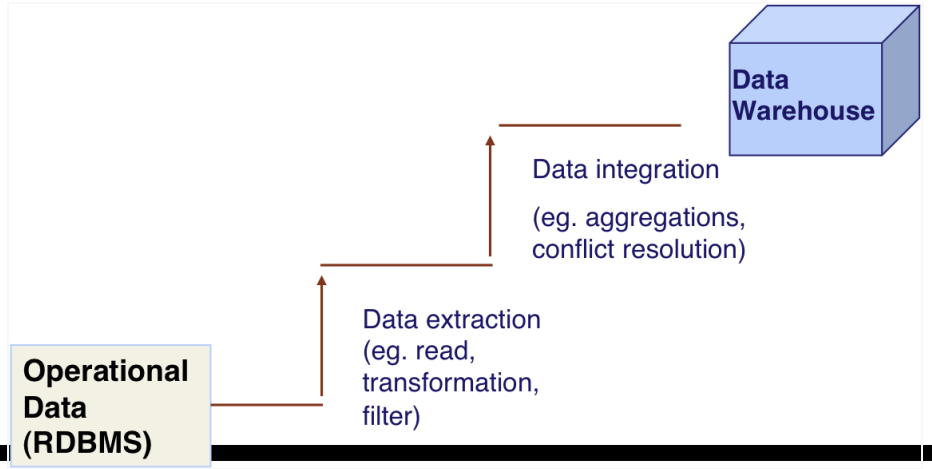
Data Engineering Architectures

Model 3 – DW model (other data producers)



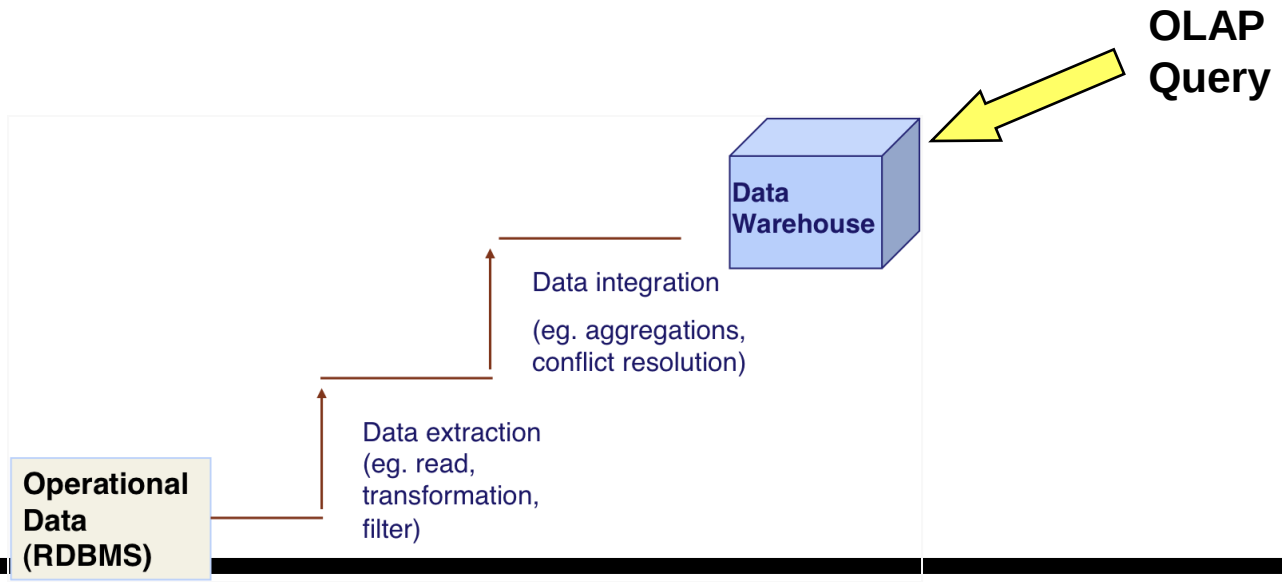
Data Warehouse

The need for an effective decision support system has motivated the emergence of the new data storage facility called Data Warehouse.



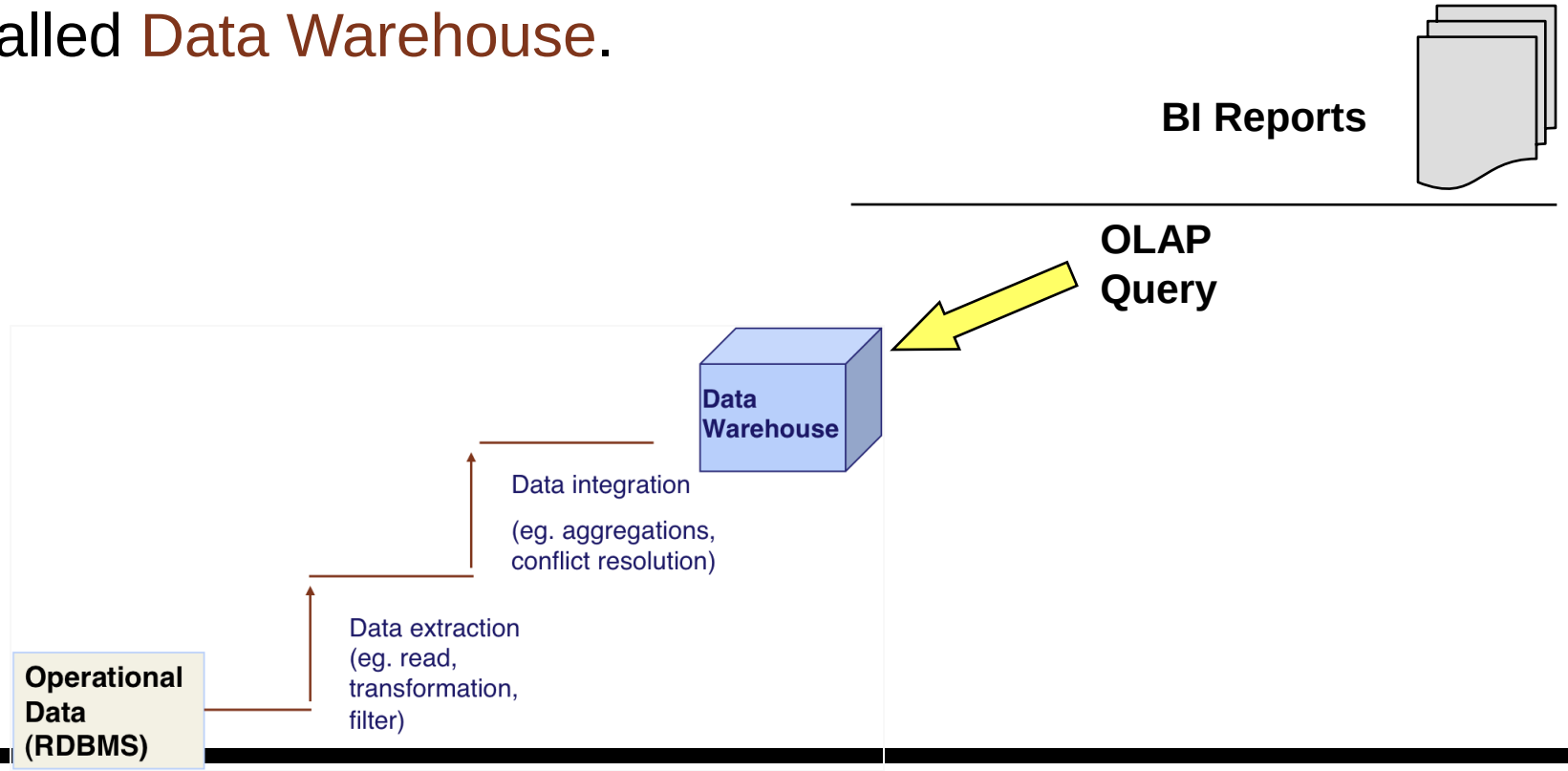
Data Warehouse

The need for an effective decision support system has motivated the emergence of the new data storage facility called Data Warehouse.



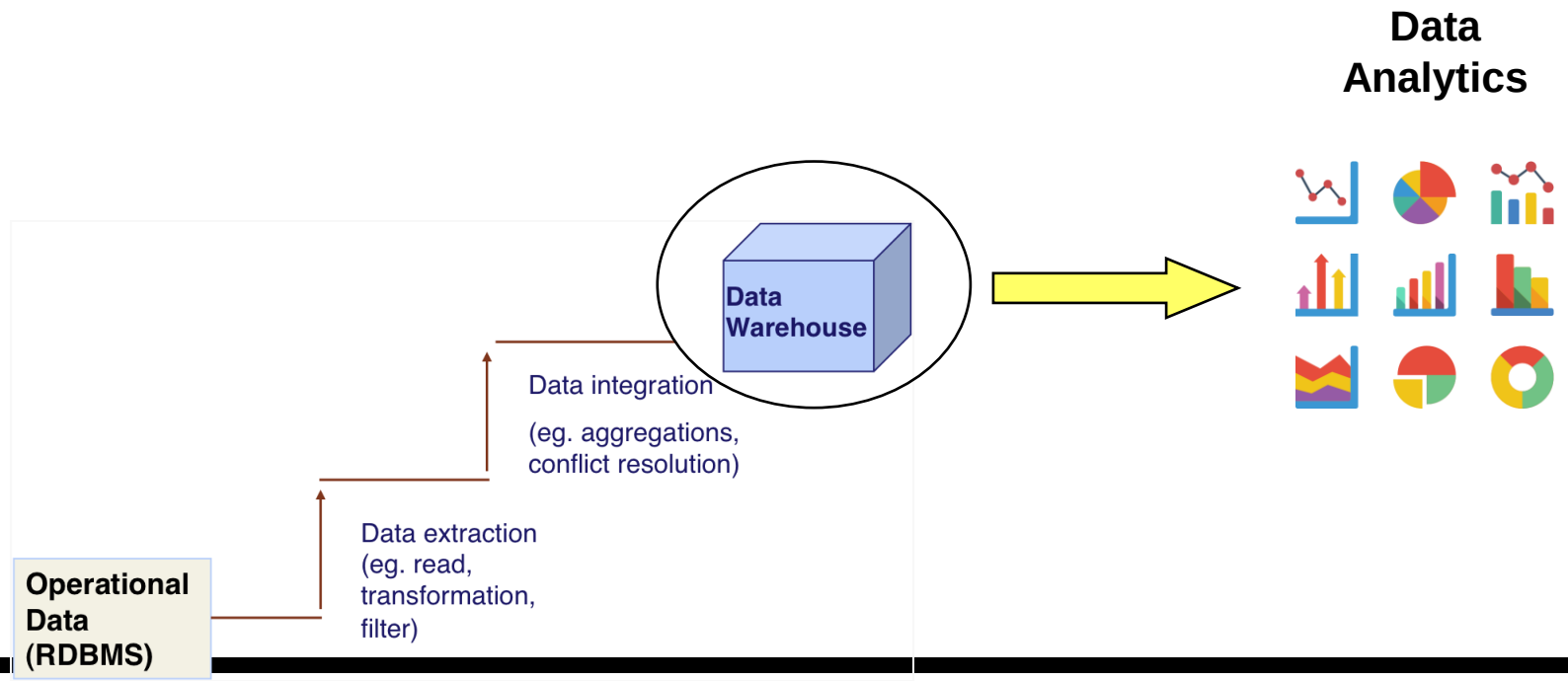
Data Warehouse

The need for an effective decision support system has motivated the emergence of the new data storage facility called Data Warehouse.



Data Warehouse

The need for an effective decision support system has motivated the emergence of the new data storage facility called Data Warehouse.



Star Schema

The Star Schema is a data modeling technique used to map multidimensional decision support data into a relational database.

The reason for the star schema's development is that existing relational modeling techniques: ER and normalization, did not yield a database structure that served the advanced data analysis requirements well.

There are Three main components of the Star Schema:

1. Facts
2. Dimensions
3. Attributes

Star Schema

1. Facts

   Facts are **numeric measurements** (values) that represent a specific business aspect or activity.

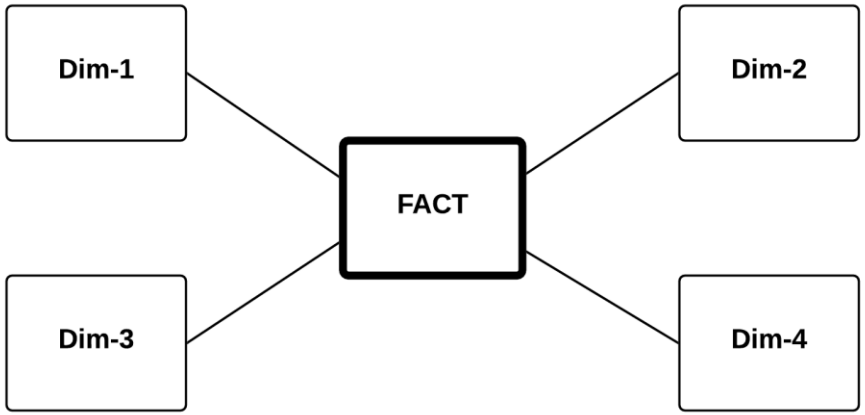
   For example, sales figures are numeric measurements that represent product and/or service sales.

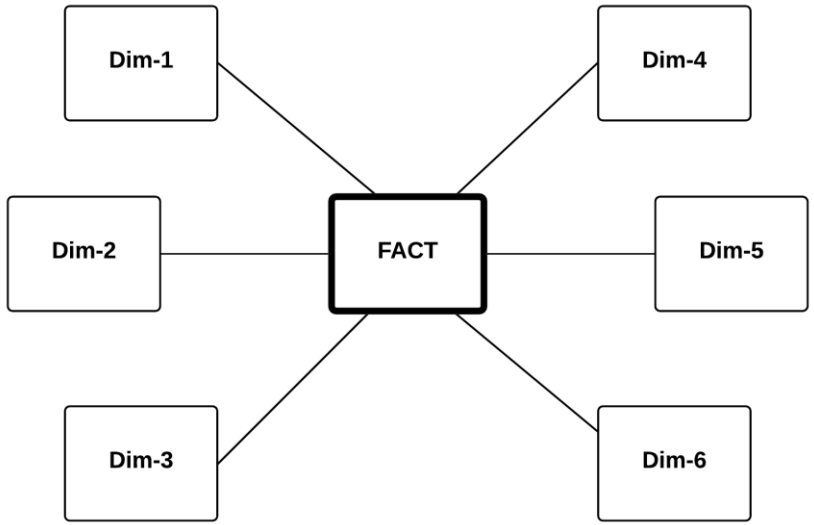
2. Dimensions

   Dimensions are qualifying characteristics that provide **additional perspectives** to a given fact.

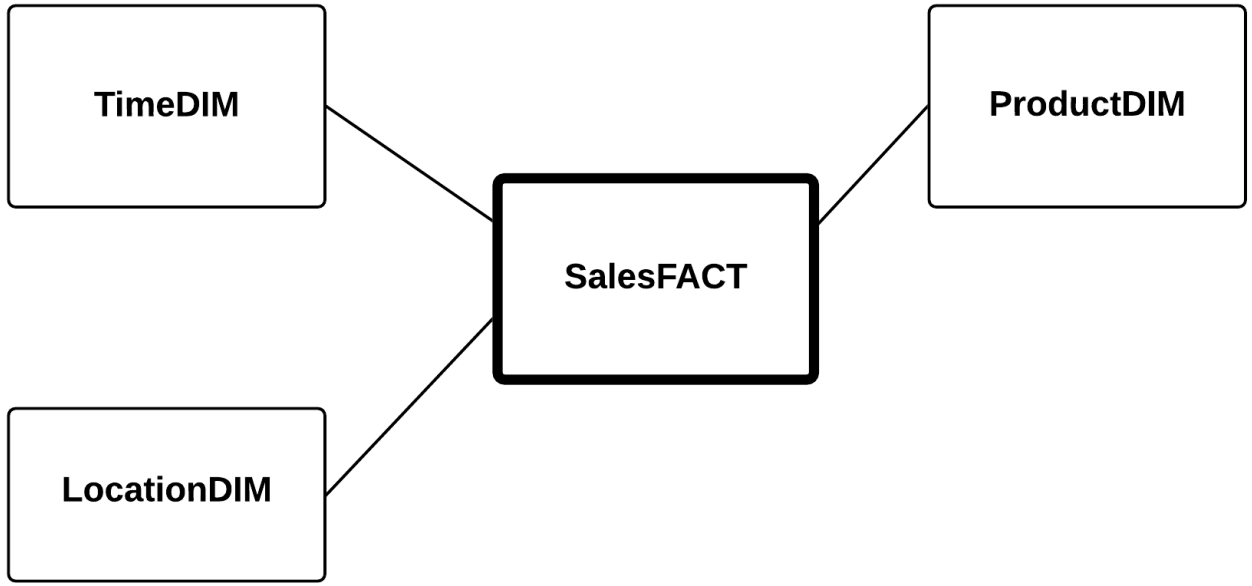
   For example, sales might be *viewed* from specific dimension(s), such as sales location, sales period, sales product, etc.

Star Schema



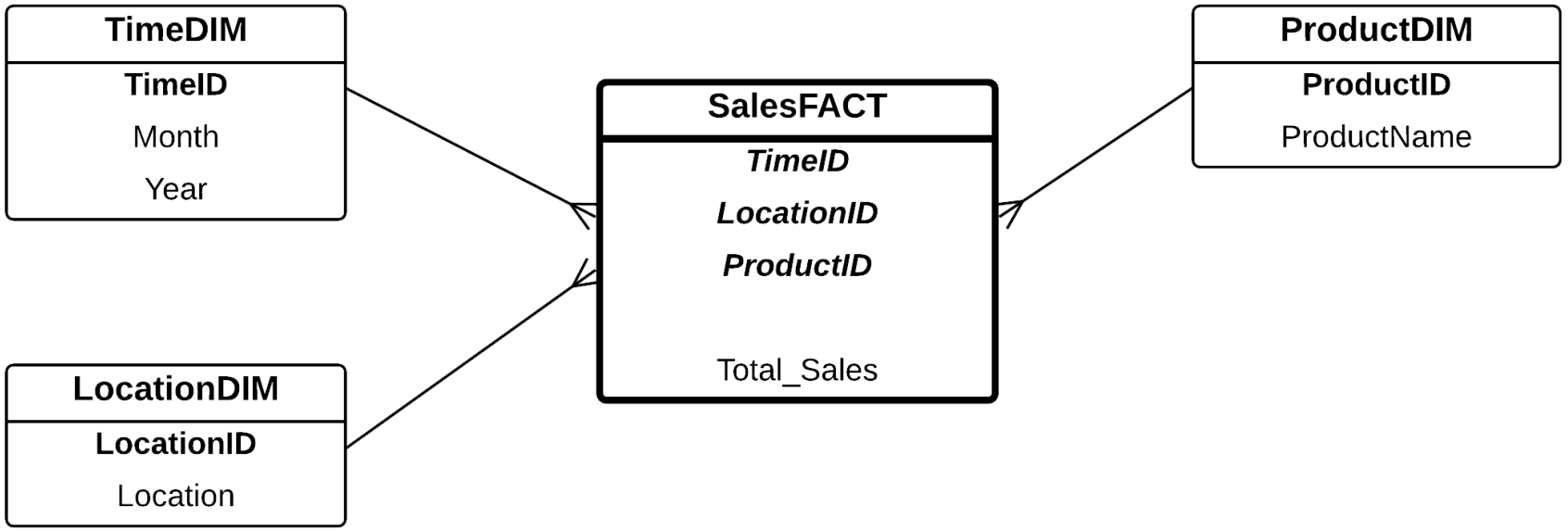


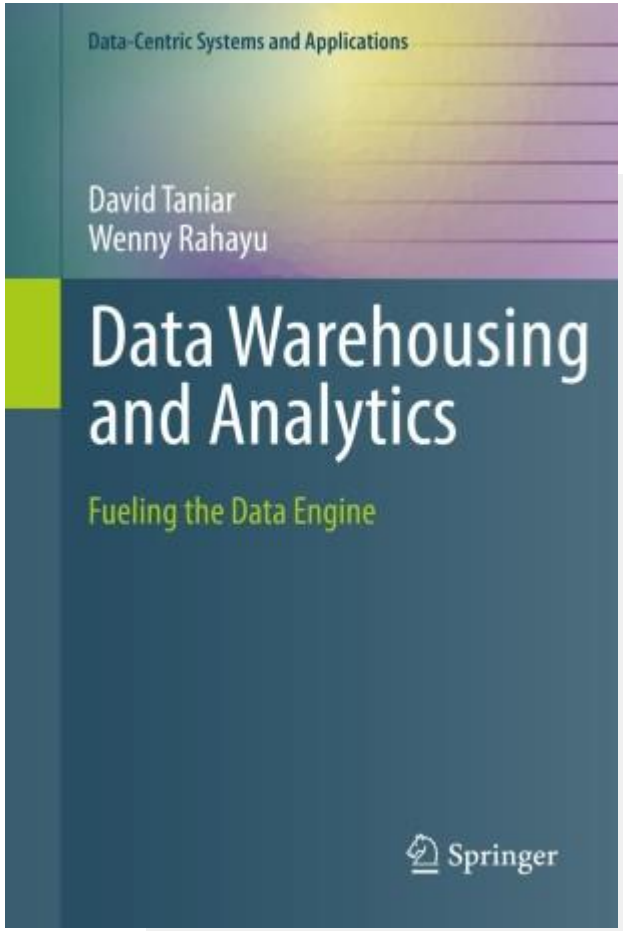
Star Schema



Star Schema

3. Attributes





Chapter 2
Simple Star Schemas

Overview

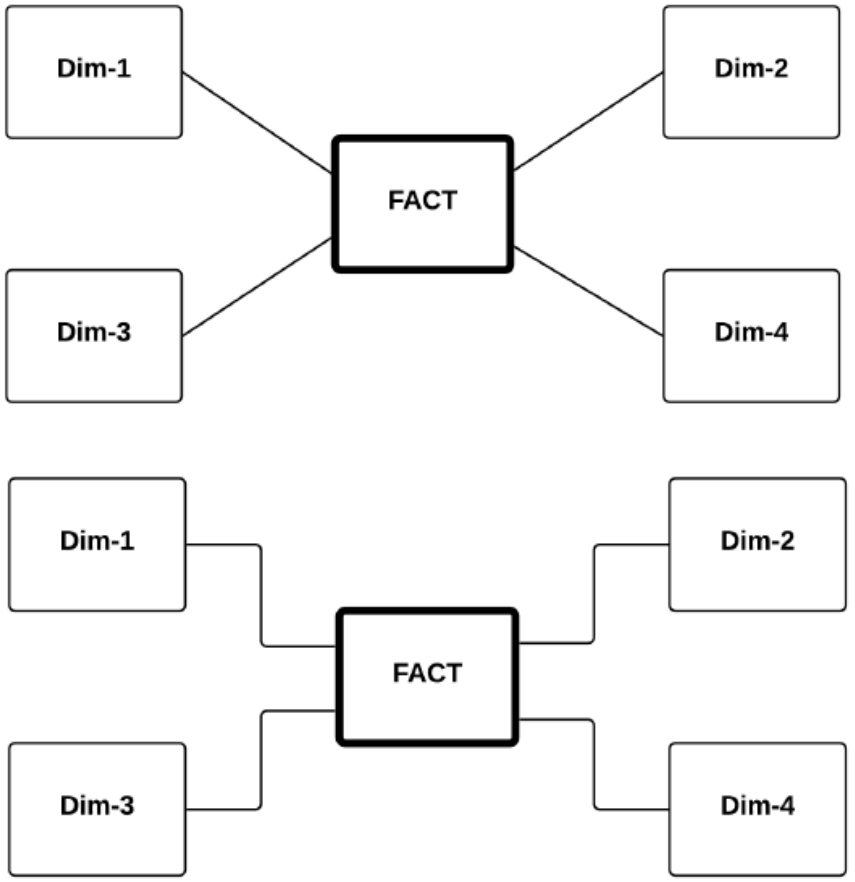
1. Notations and Processes
2. First Case Study: A College Star Schema
3. Another Simple Case Study: A Sales Star Schema
4. Two-Column Table Methodology

1. Notations and Process

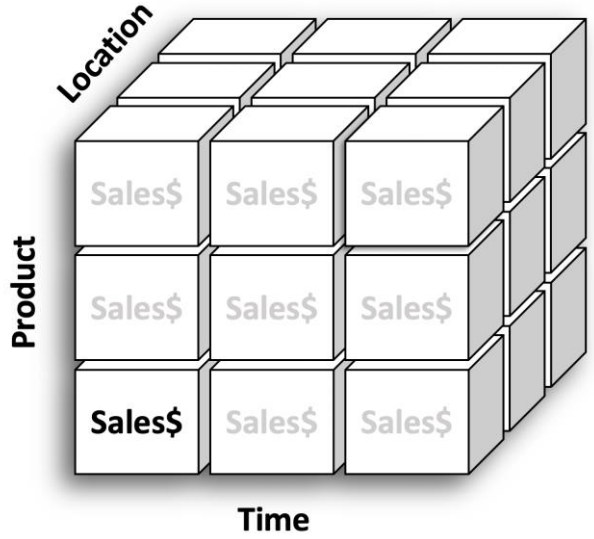
- Star Schema Notation
- E/R Diagram Notation
- Transformation Process

1.1. Star Schema Notation

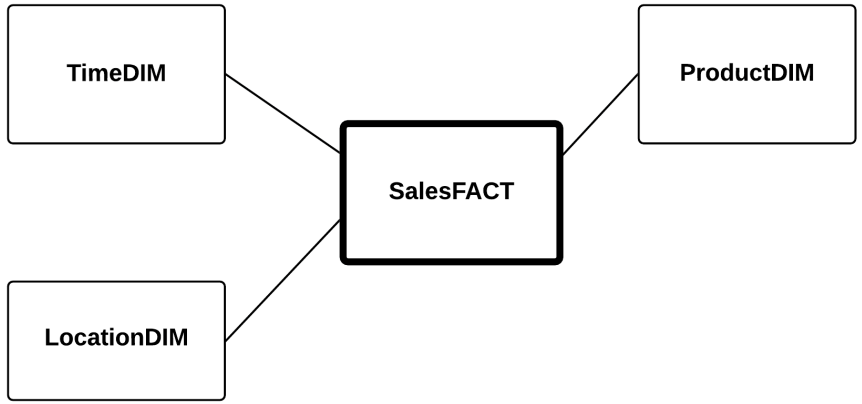
- A data warehouse is a multi-dimensional view of the database
- A Star Schema shows a multi-dimensional view
- A Star Schema consists of **Fact** and **Dimensions**
- The lines can be straight lines or bended lines



1.1. Star Schema Notation

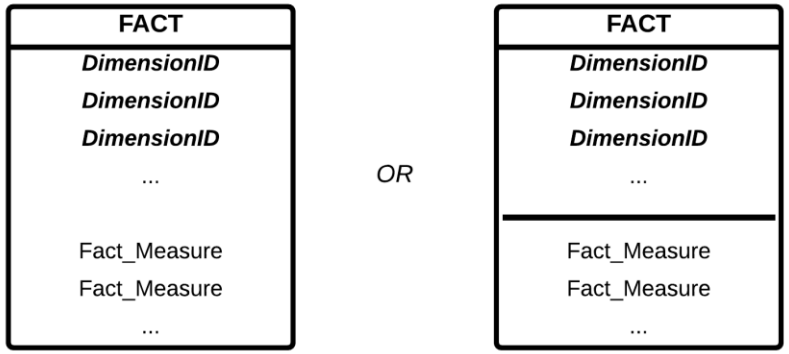


- A data warehouse is a **multi-dimensional view** of the database
- A Star Schema shows a multi-dimensional view
- A Star Schema consists of Fact and Dimensions
- The lines can be straight lines or bended lines

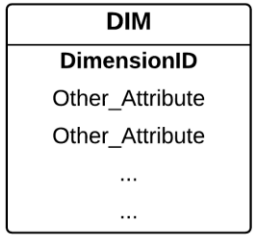


1.1. Star Schema Notation

- Fact and Dimensions contain attributes
  - Each Dimension has a Dimension ID (PK)
  - Dimension IDs in Fact are FK and PK
- Fact can only have numerical values



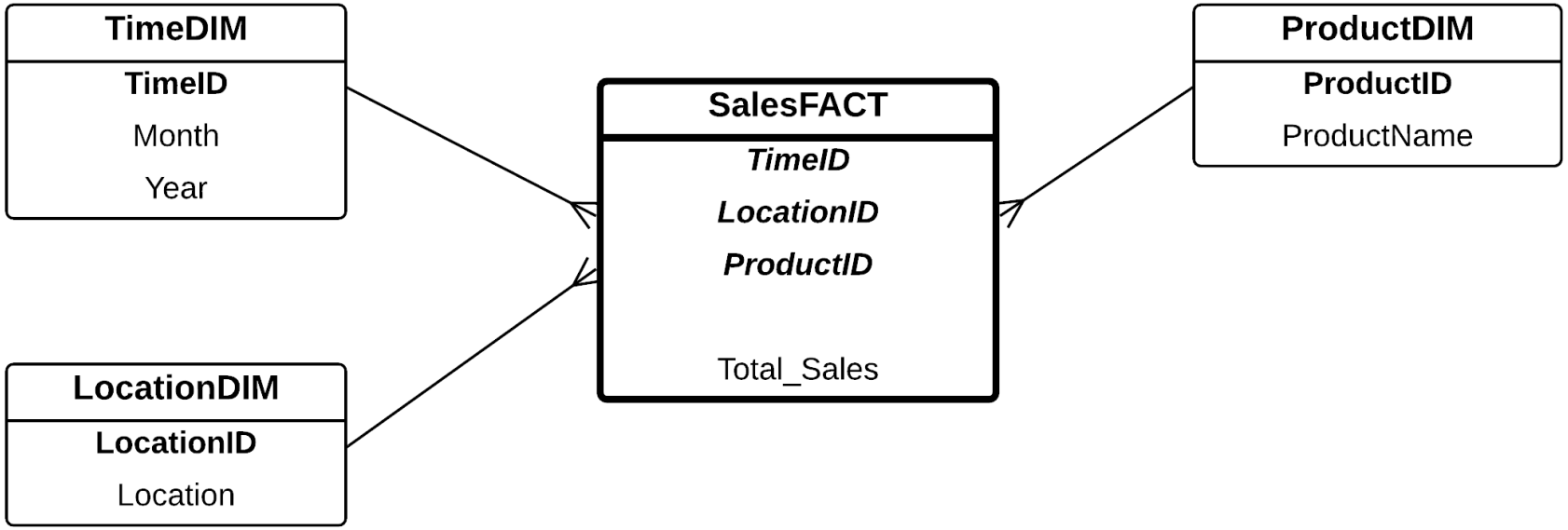
(*a*) Fact



(*b*) Dimension

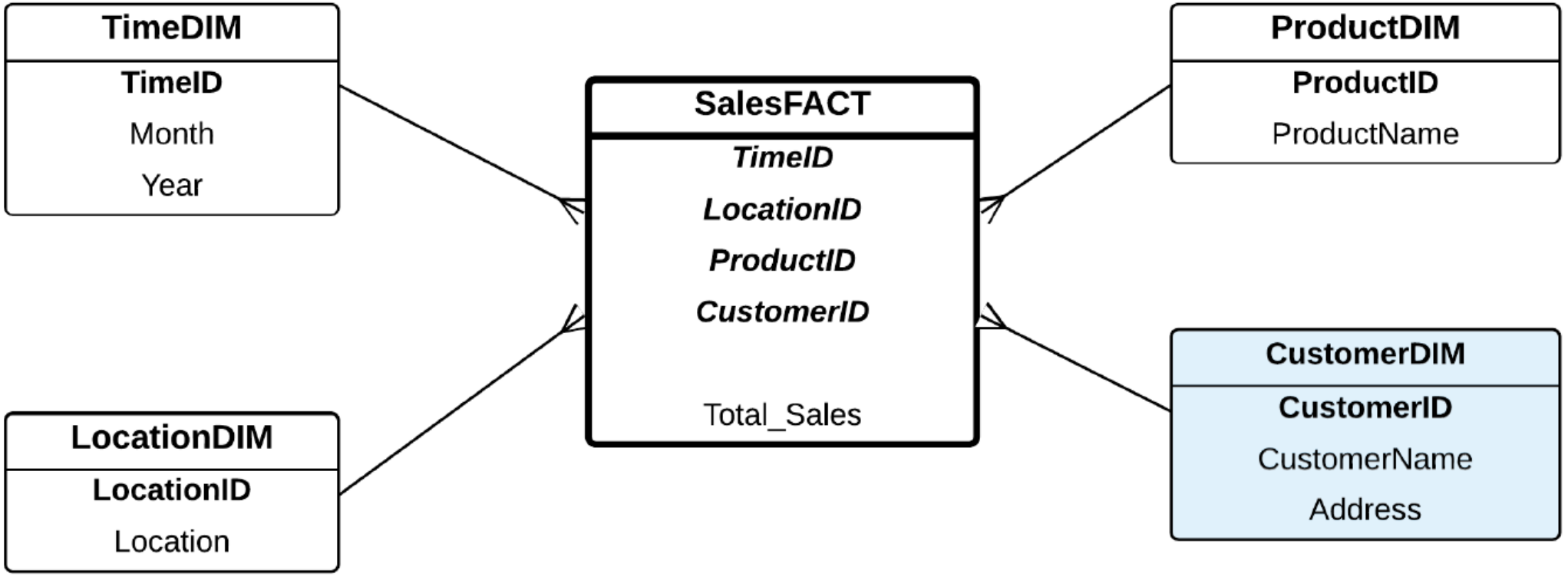
1.1. Star Schema Notation

- Sales star schema complete with attributes



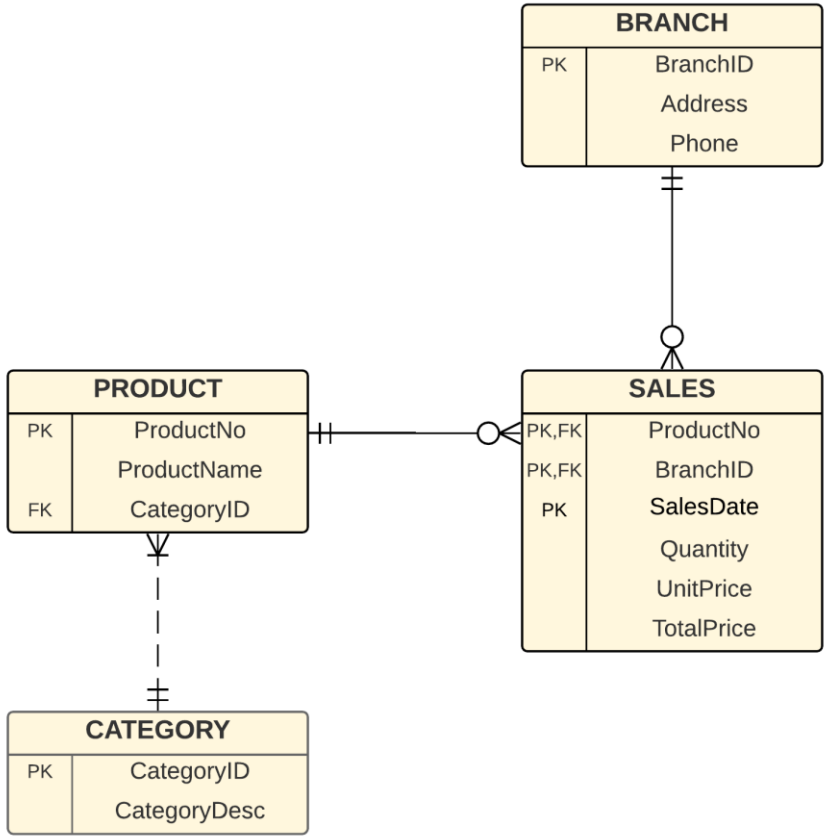
1.1. Star Schema Notation

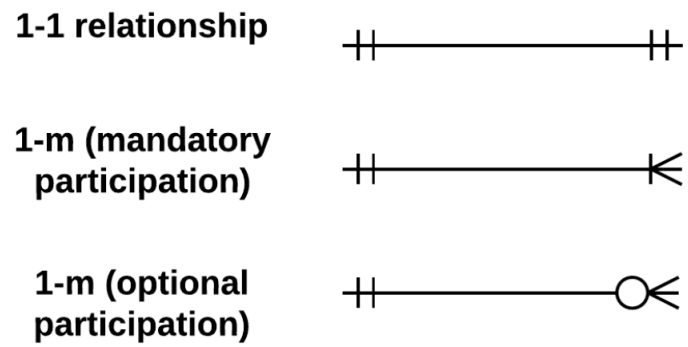
- When a particular dimension is being discussed, the dimension is highlighted in the star schema



1.2. E/R Diagram Notation

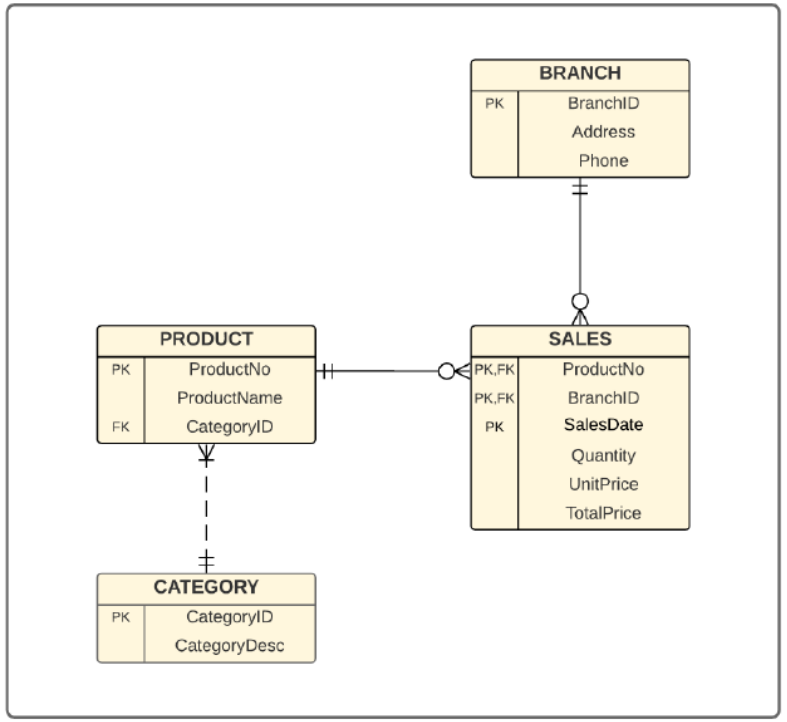
- Entity name is capitalised
- Primary Key: PK ; Foreign Key: FK
- Relationship notation uses a Crow-foot notation with participation
- Associative relationship (m-m), Non-associative relationship (1-m)





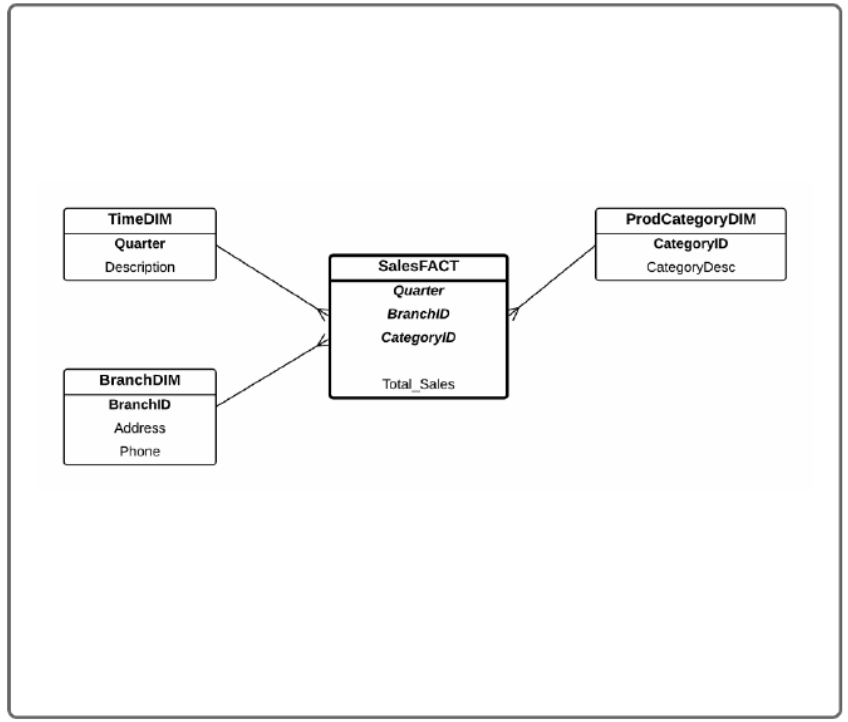
1.3. Transformation Process

Operational Database (E/R Diagram)



Transformation (ETL)

Data Warehouse (Star Schema)



4. Two-Column Table Methodology

- To check the correctness of a star schema.
- *Imaginary Table* of our view to the fact measure from one particular dimension angle.
- First column represents category or dimension
- Second column represents fact
- Consists of two types:
  - One Fact Measure
  - Multiple Fact Measure

4.1. One Fact Measure

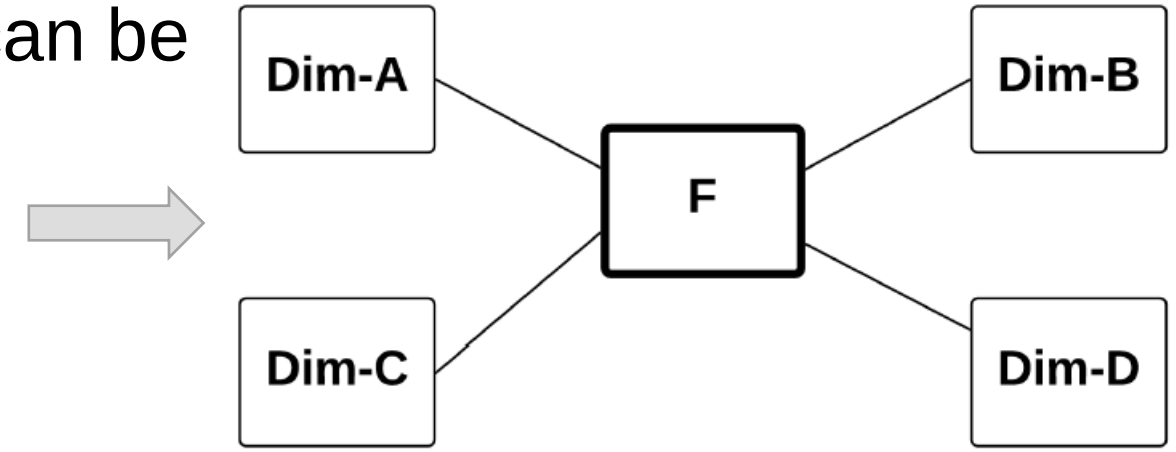
- First column contains a category
- Second column contains a statistical numerical figure
- E.g.: If all these four tables make sense, that is, the fact measure F is correctly viewed from each category (e.g. A, B, C, and D), then a star schema can be

A	F
x	4
y	3

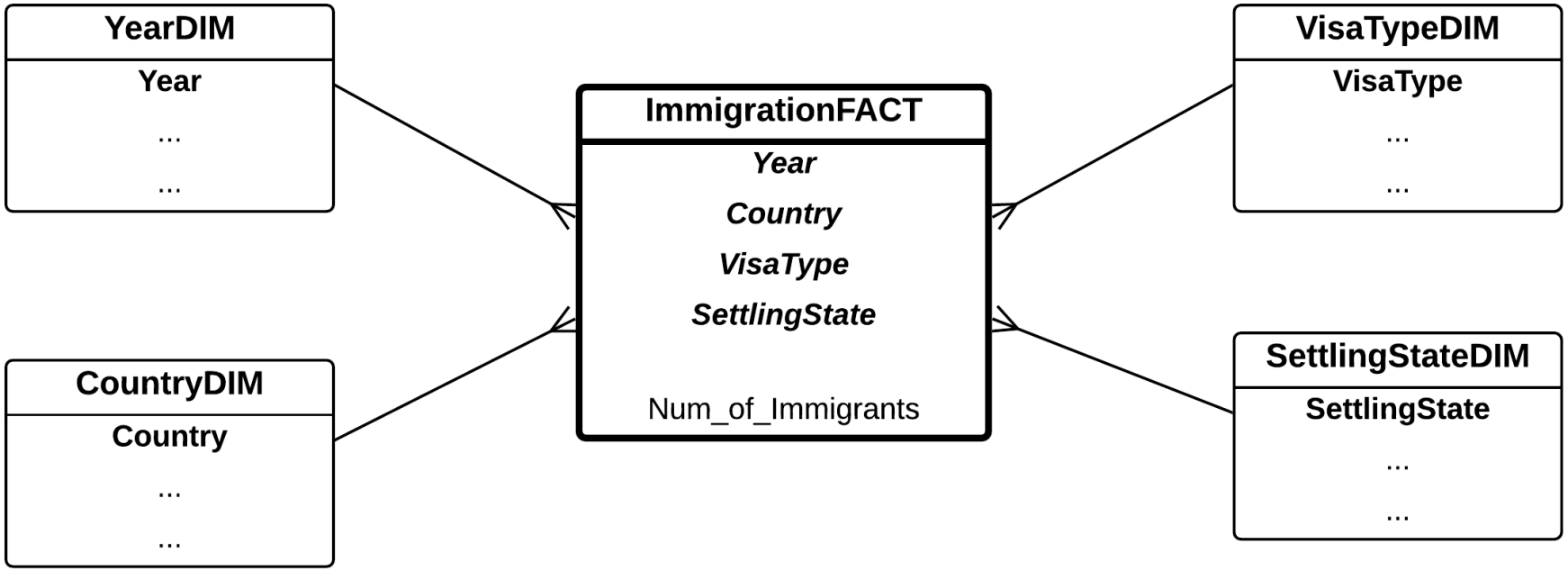
B	F
r	5
s	3

C	F
k	1
m	1

D	F
p	2
q	5



4.1. One Fact Measure: Example



4.2. Multiple Fact Measure

- Second column contains multiple facts F={F1,F2,F3…}
- All Fs must exist in all tables

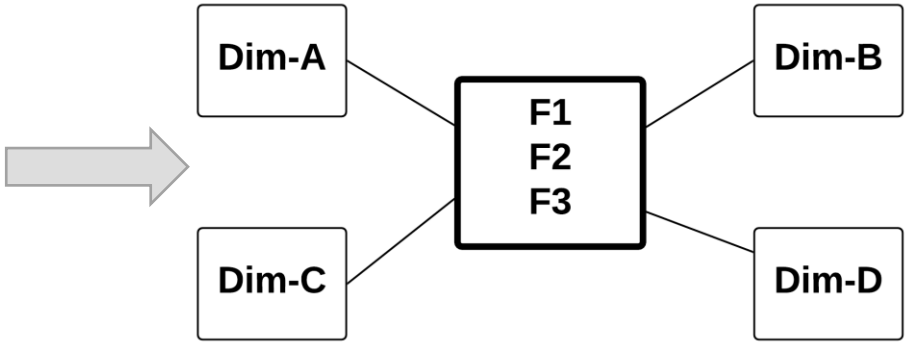
Column 1 Column 2

A	F1	F2	F3
x	4	0	1
y	3	2	7

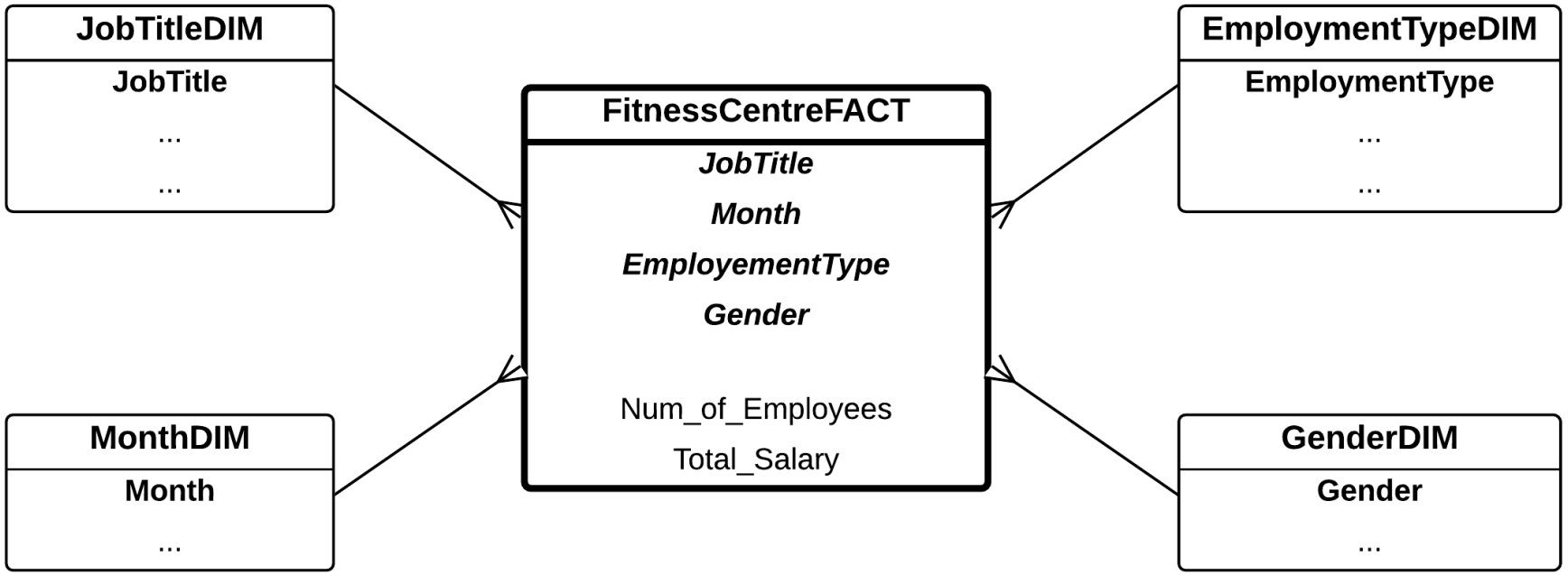
B	F1	F2	F3
r	5	2	8
s	3	5	9

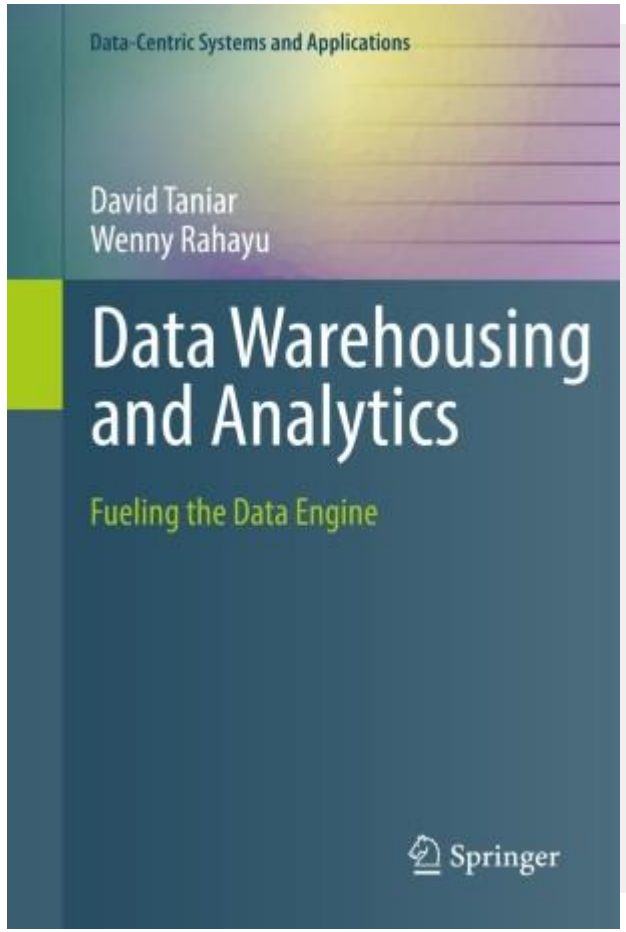
C	F1	F2	F3
k	1	0	4
m	1	0	6

D	F1	F2	F3
p	2	7	8
q	5	4	8



4.2. Multiple Fact Measure: Example





Chapter 3
Creating Facts and Dimensions:
More Complex Processes

Overview

1. Use of `count` Function
2. Average in the Fact
3. Outer Join
4. Creating Temporary Dimension Tables
5. Creating Temporary Tables in the Operational Database

1. Use of Count Function

- SQL:
  - `count(*)` → number of records
  - `count(attribute)` → exclude null values
  - `count(distinct attribute)` → remove duplication

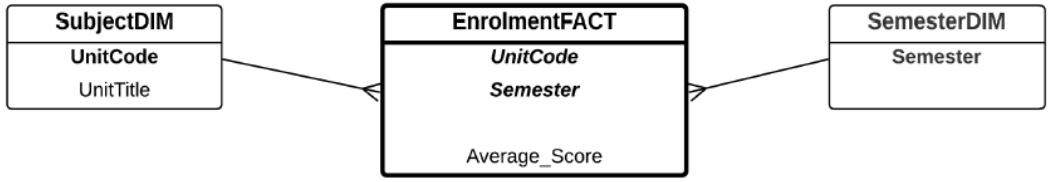
2. Average in the Fact

- Many aggregate functions can be used in fact table
- There are some pitfalls, especially when fact measures are highly aggregated
- Inappropriate use of average function will be highlighted

2. Average in the Fact:
Average of an Average **Example**

Unit Code	Unit Title	Semester	Student First Name	Score
IT001	Database	1	Mirriam	81
IT001	Database	1	Allan	41
IT001	Database	1	Ben	74
IT001	Database	1	Kate	85
IT001	Database	1	Larry	87
IT001	Database	1	Leonard	75
IT001	Database	2	Juan	64
IT001	Database	2	Andy	32
IT002	Java	1	Ally	65
IT002	Java	1	Menson	47
IT002	Java	2	Mirriam	78
IT002	Java	2	Ben	73
IT002	Java	2	Larry	64
IT003	SAP	1	Ally	63
IT004	Network	2	Juan	53
IT004	Network	2	Menson	52

- Fact: Average Score
- Dimension: Subject, Semester



2. Average in the Fact:
Average of an Average **Example**

Table 1.2 Table: Fact

Unit Code	Semester	Average Score
IT001	1	73.833
IT001	2	48
IT002	1	56
IT002	2	71.667
IT003	1	63
IT004	2	52.5

Table 1.3 Table: Subject Dimension

Unit Code	Unit Title
IT001	Database
IT002	Java
IT003	SAP
IT004	Network

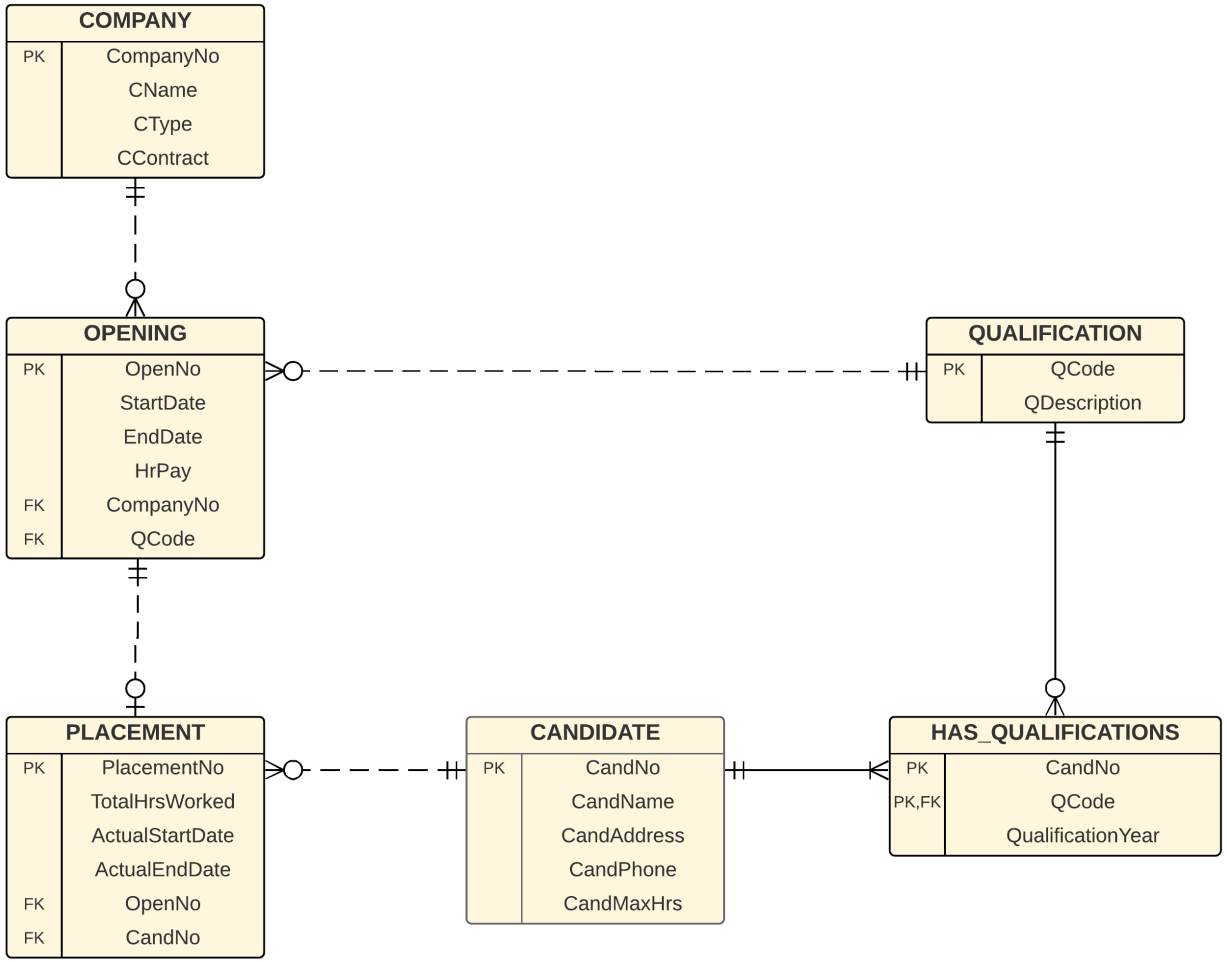
Table 1.4 Table: Semester Dimension

Semester
1
2

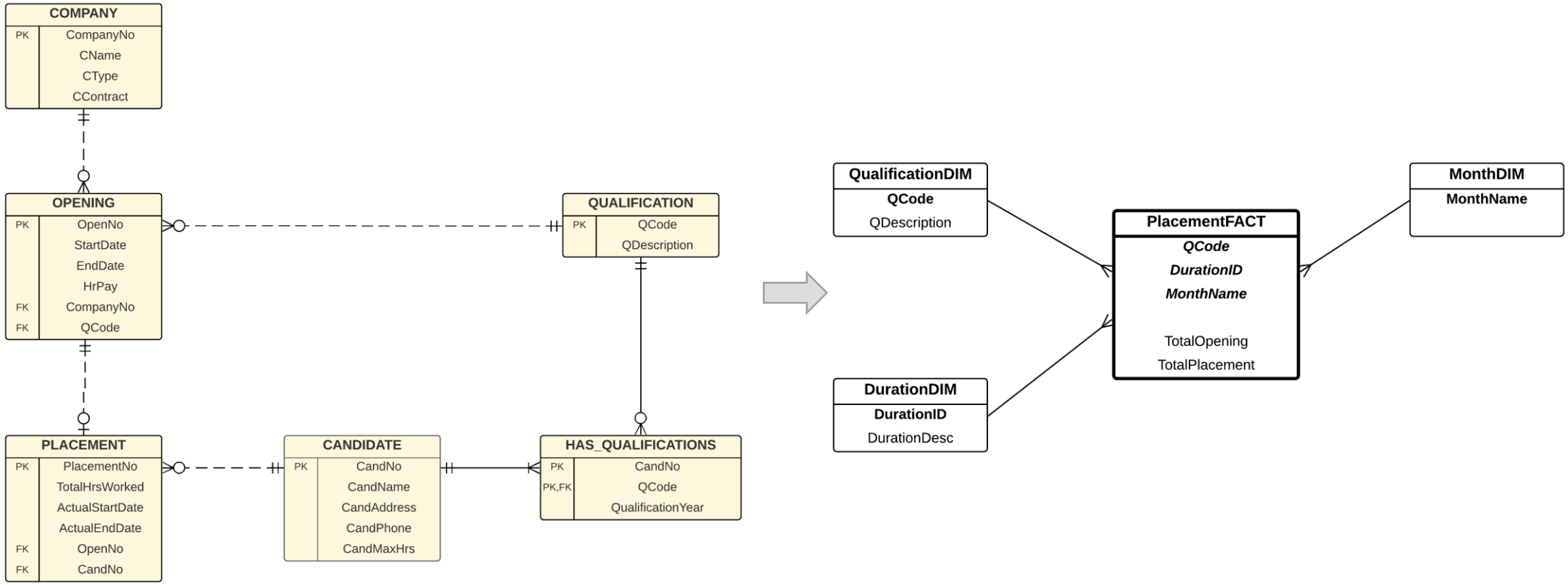
- Average Score for the Database Unit from Fact table (73.833+48)/2= **60.9165**
- Average Score for the Database unit from operational table 539/8=**67.375**
- Storing average as a fact measurement is not a good idea
- Replace **Average Score** with **Total Score** and **Number of Students**

3. Outer Join (Employment Agency)

- Employment Agency which places temporary workers in companies during peak periods
- Business process is described in the chapter

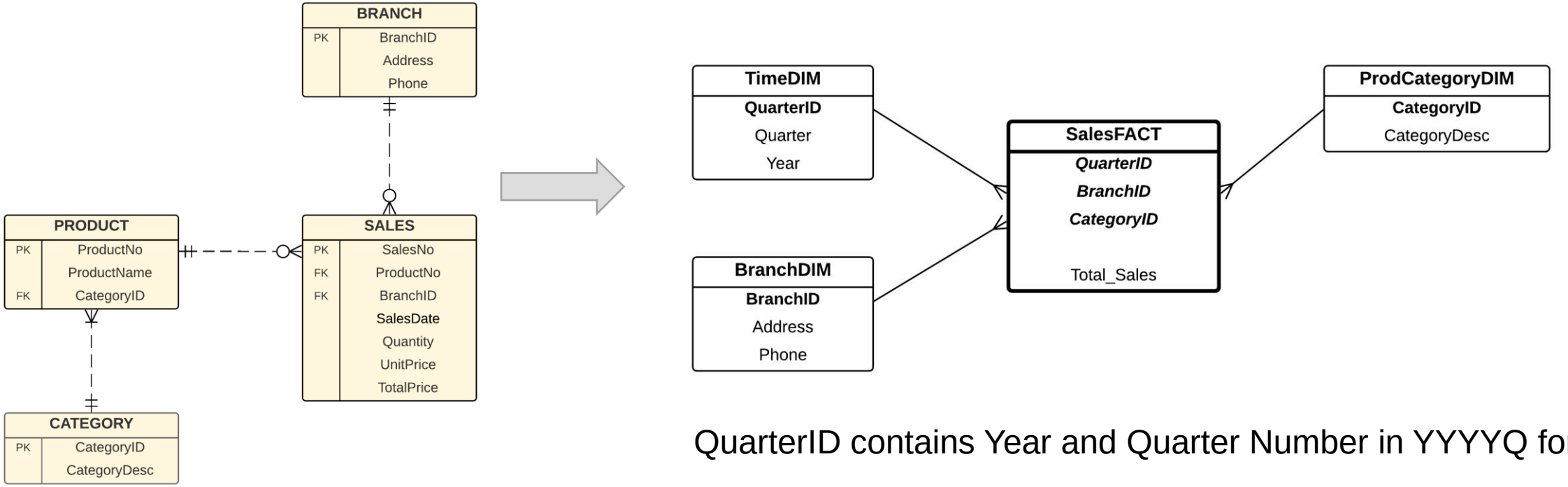


3. Outer Join
Example: Employment Agency



4. Creating *Temporary* Dimension Tables Sales Example

- Required dimension table cannot be created directly.

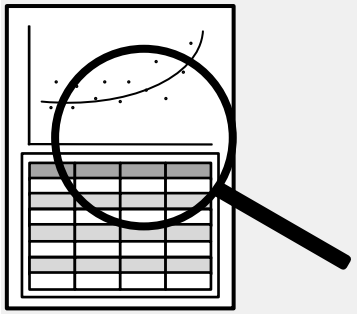


QuarterID contains Year and Quarter Number in YYYYQ format

4. Creating *Temporary* Dimension Tables Sales Example

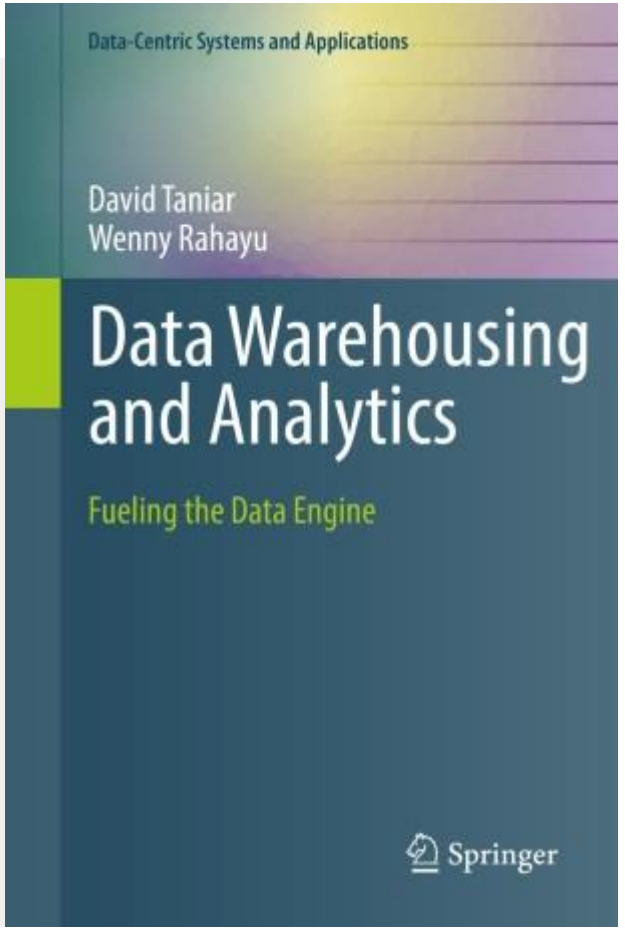
```
create table BranchDim as
select * from Branch;

create table ProdCategoryDim as
select * from Category;
```

- The Time Dimension that has QuarterID, Quarter, and Year attributes need to be created manually.
- Problems:
  - Number of records is unknown
  - Manual insertion in not efficient
- Temporary Time dimension is needed



Chapter 04

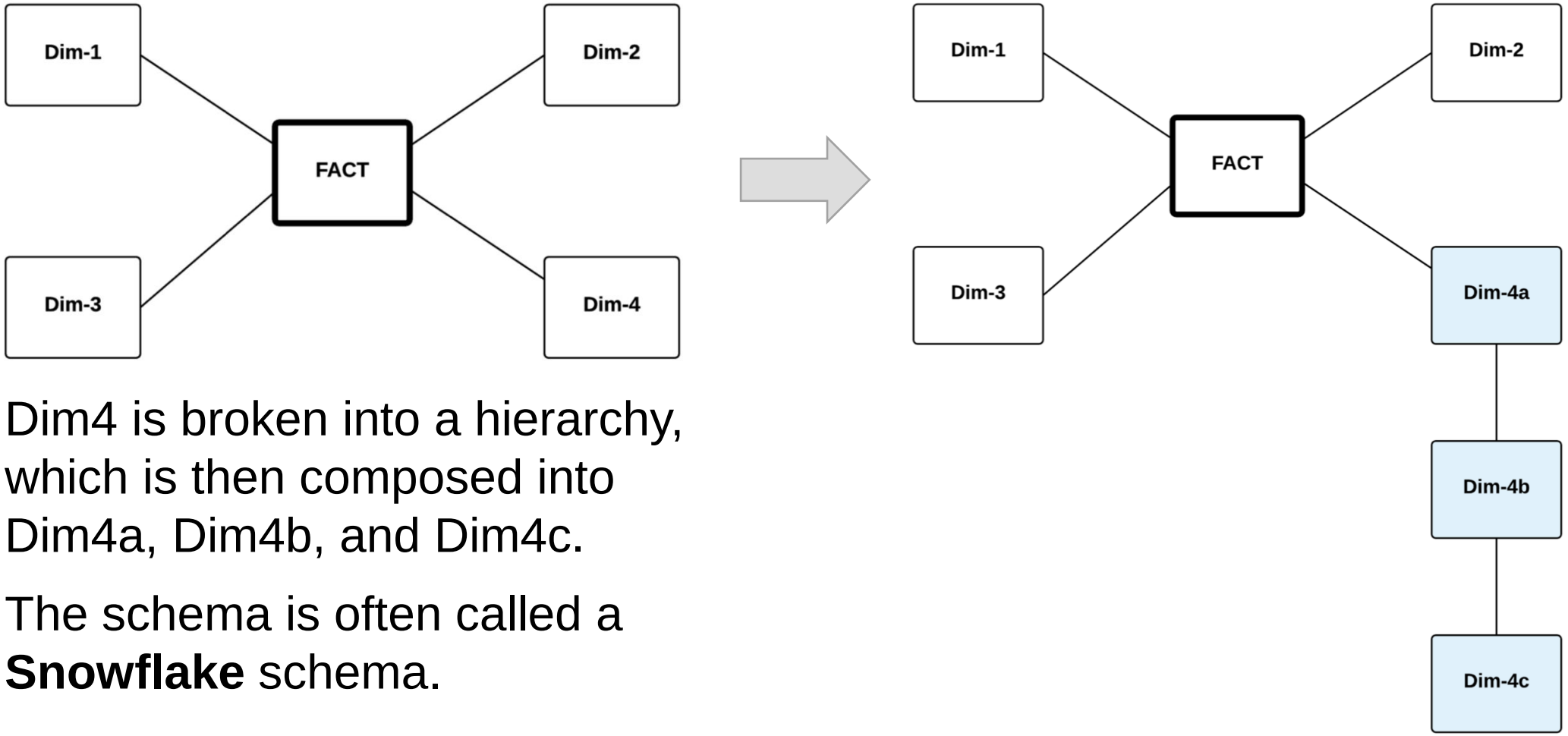
Hierarchy



Outline

- A **Hierarchy** is formed when a dimension is broken down to two or more dimensions in a hierarchical manner.
- This chapter focuses on how and when to design hierarchical dimensions in a star schema, and to compare the differences between the hierarchical version and the non-hierarchical version, which are:
    1) Hierarchy versus Non-Hierarchy Dimensions
    2) Hierarchy versus Multiple Independent Dimensions
    3) Linked Dimensions
    4) Hierarchy Design Considerations

1. Hierarchy versus Non-Hierarchy



- Dim4 is broken into a hierarchy, which is then composed into Dim4a, Dim4b, and Dim4c.
- The schema is often called a **Snowflake** schema.

1. Hierarchy versus Non-Hierarchy

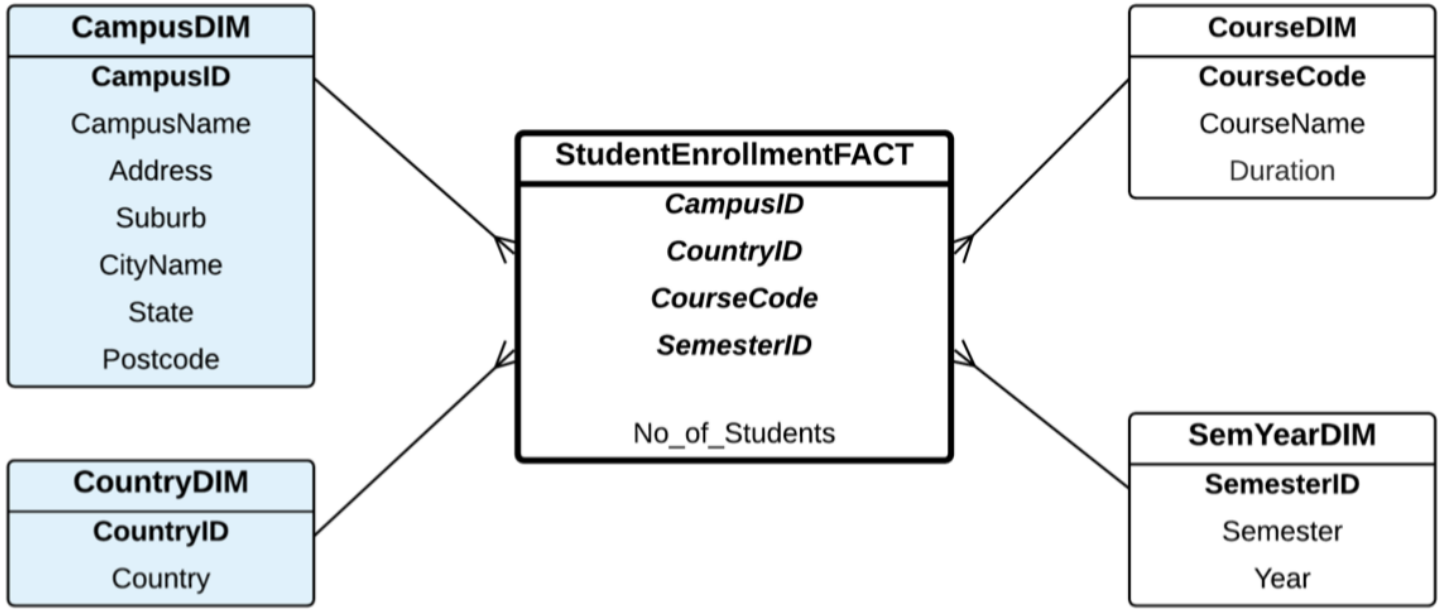
1) One tables vs. many tables
   - Without hierarchy, there is only one table.
   - With hierarchy, there are three tables

2) 3NF vs. lower than 3NF
   - With the hierarchy option, the tables are in 3NF.
   - With the non-hierarchy option, the table (e.g. the Campus Dimension table) is in 2NF.

3)Drilling down and rolling up
   - The hierarchy model is not about drilling down information exploration.
   - Rolling up is exploring information from the detail to a more general.

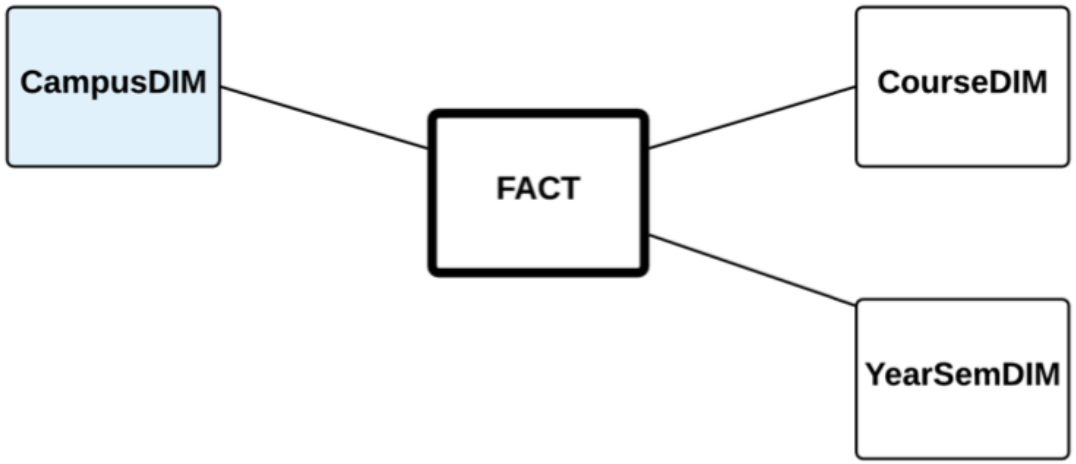
2. Hierarchy versus Multiple Independent Dimensions

- A few possibilities:
  - **Separate Dimensions**
  - Combined Dimension
  - Hierarchy Dimensions



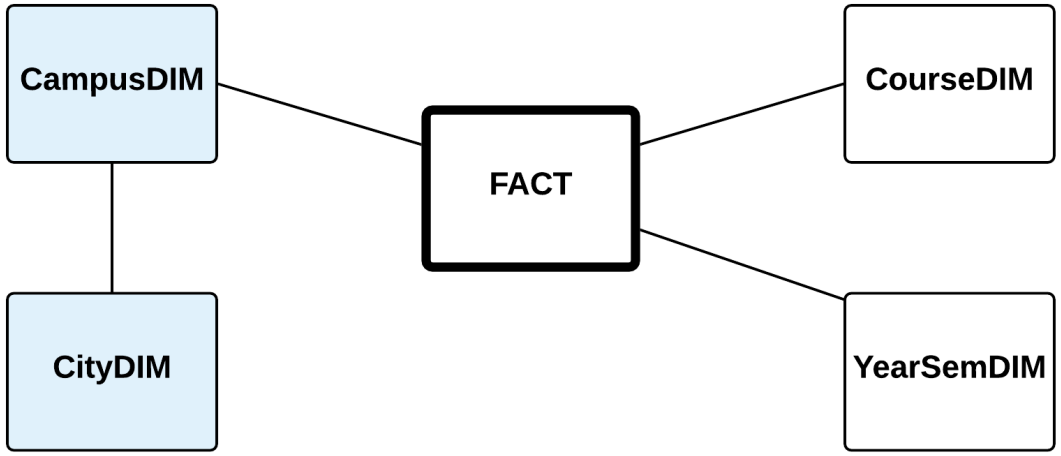
2. Hierarchy versus Multiple Independent Dimensions

- A few possibilities:
  - Separate Dimensions
  - **Combined Dimension**
  - Hierarchy Dimensions

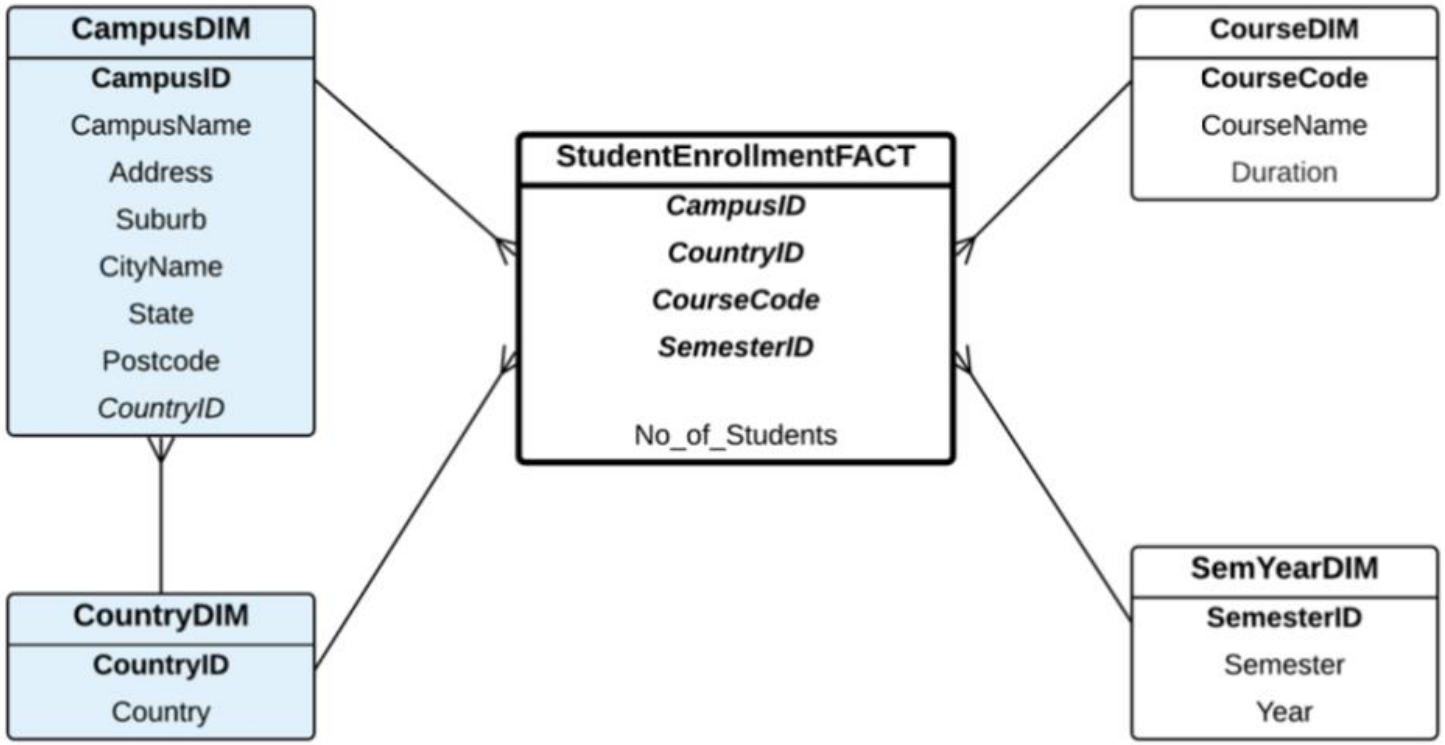


2. Hierarchy versus Multiple Independent Dimensions

- A few possibilities:
  - Separate Dimensions
  - Combined Dimension
  - **Hierarchy Dimensions**

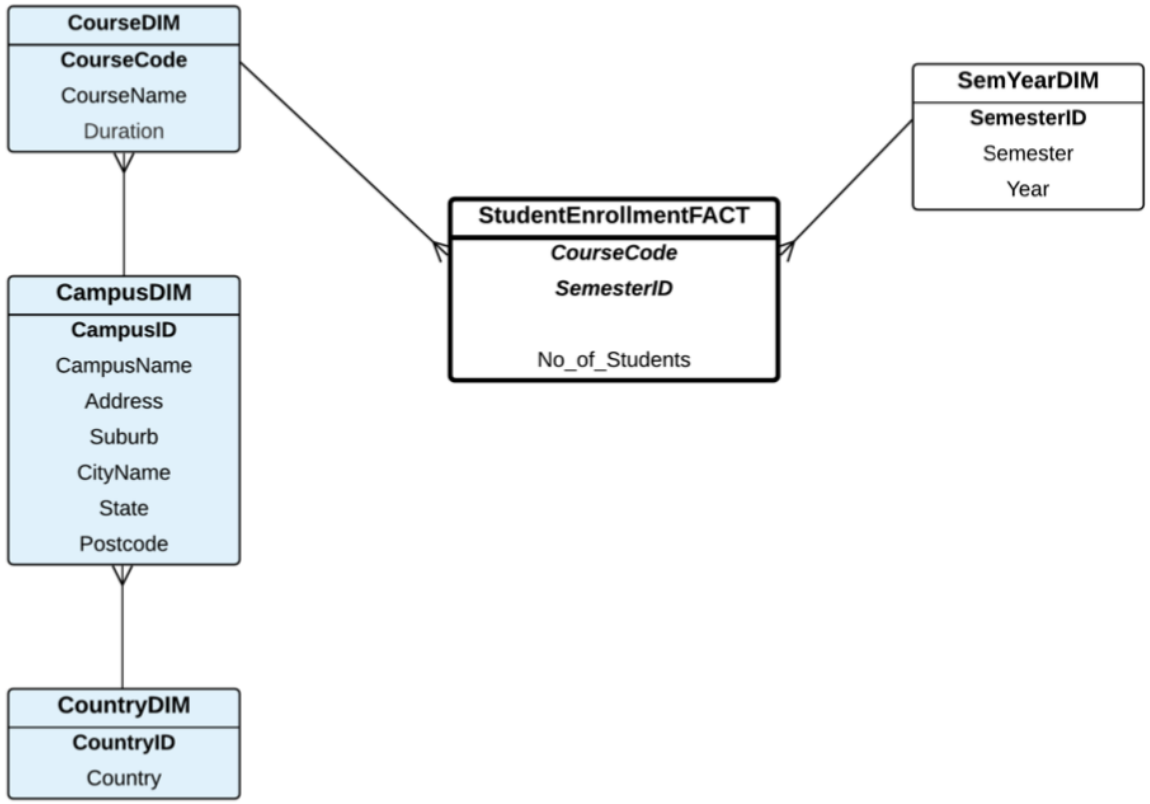


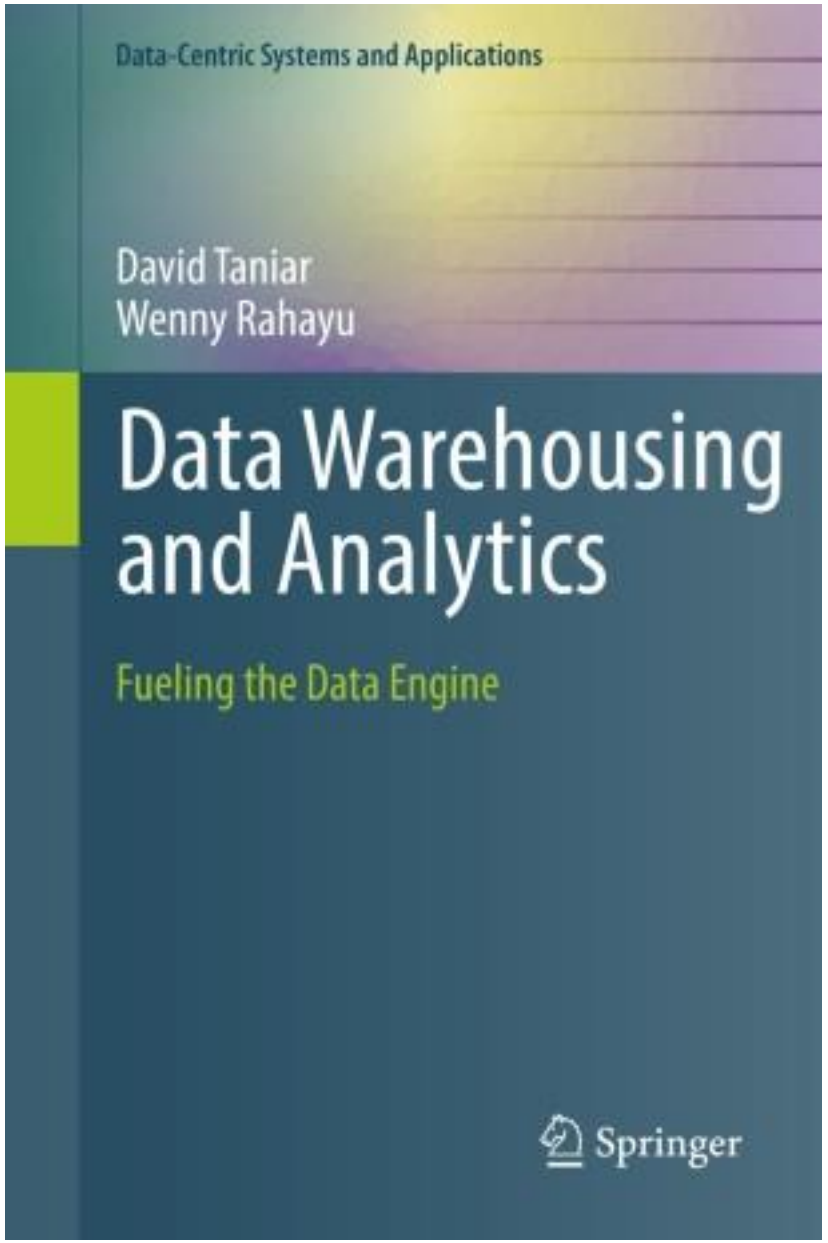
3. Linked Dimensions



4. Hierarchy Design Considerations

- Data access to the data warehouse is always through the Fact (e.g. fact measure)
- Fact measures are the focus of data retrieval
- All dimensions should be linked directly to the fact whenever possible





Chapter 5
Bridge Tables

Overview

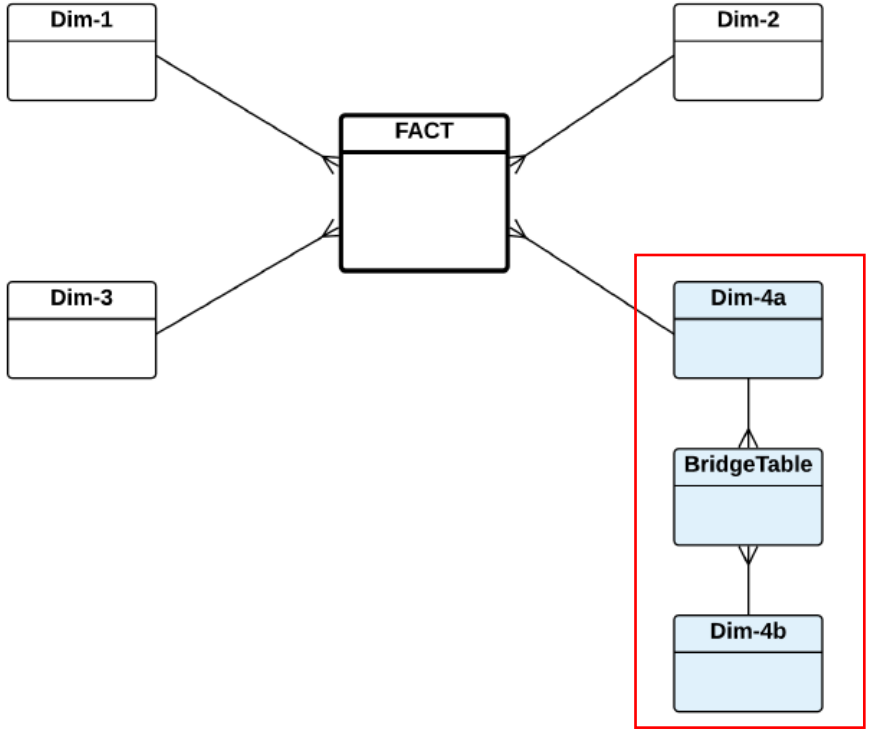
- Definition of bridge tables
- How and when to use bridge table
- Snowflakes schema in Data Warehousing

Bridge Tables

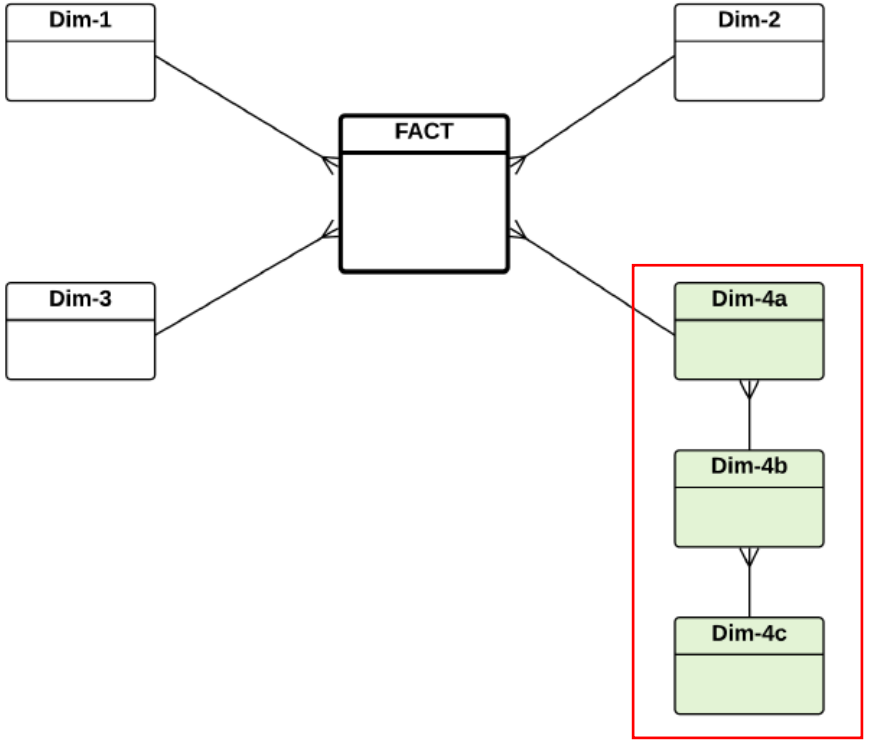
- A bridge table is a table that links between two dimensions; and only one of these two dimensions are linked to the fact.
- The star schema becomes a snowflake schema
- Bridge table is also a snowflake
- The relationship between the two dimensions that are linked through a bridge table has a cardinality of 1-many and many-1

Bridge Tables

Snowflake with a Bridge Table



Snowflake with a Hierarchy

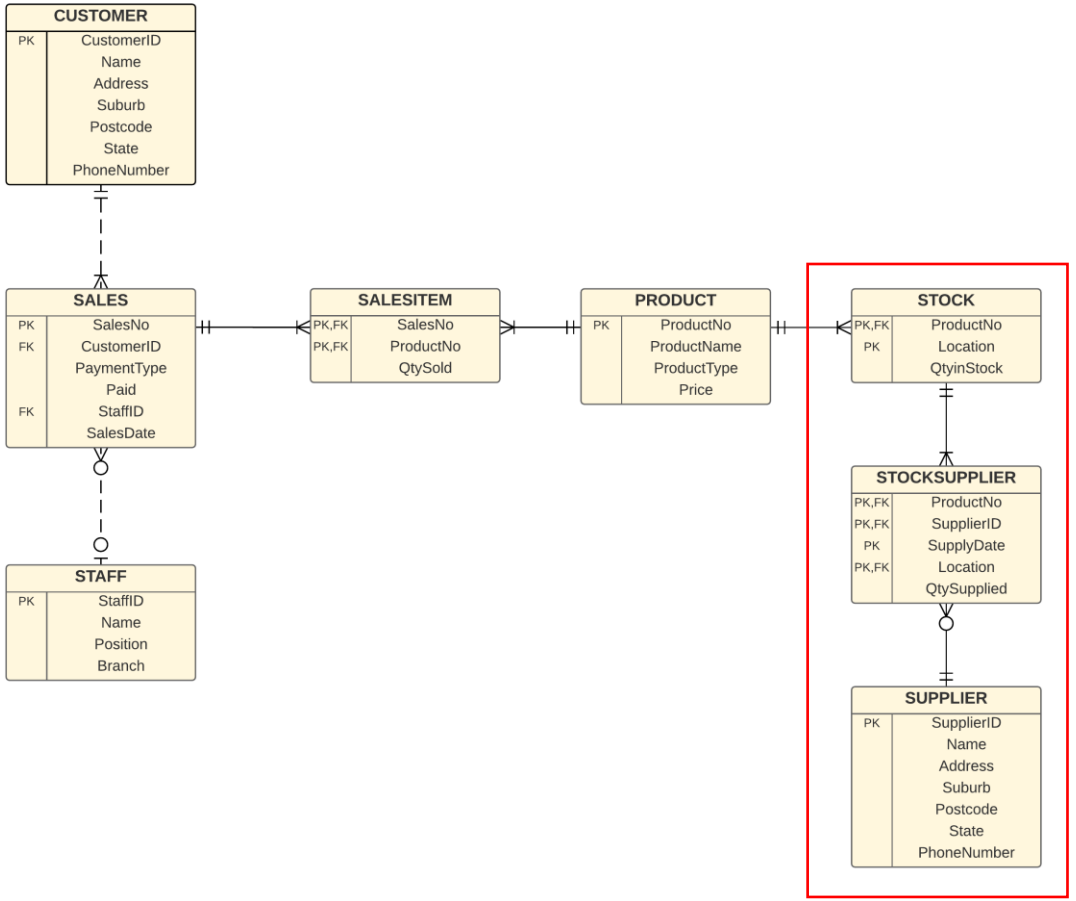


Bridge Tables

- There are at least two reasons, why a dimension cannot be connected directly to the Fact:
  - The Fact table has a fact measure, and the dimension has a key identity
  - The operational database does not have this data if the relationship between two entities in the operational database that hold the information about dimension's key identity and the intended fact measure is a *many-many* relationship.

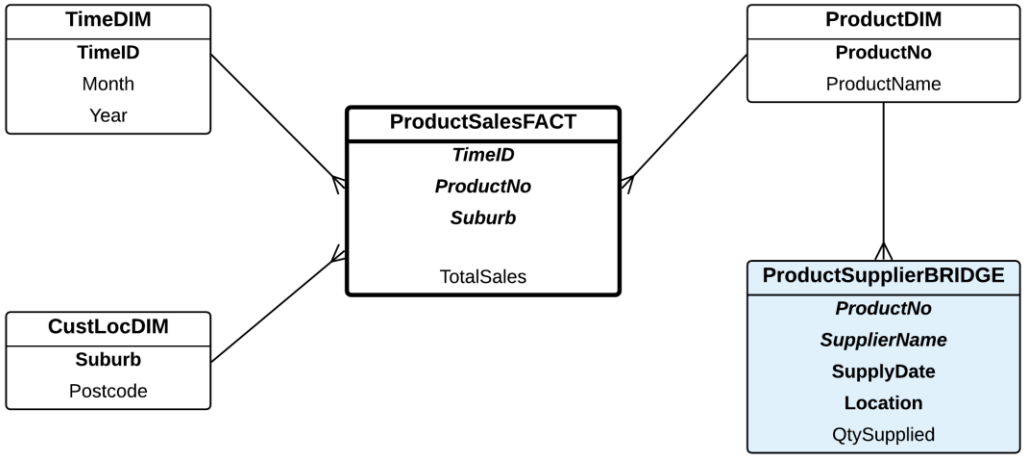
1. A Product Sales Case Study

- Analyse the statistics of its product sales history
- Analysing the *total sales* (*quantity * price*) by
  1. *product*
  2. *customer suburbs*
  3. *sales time periods* (month and year)
  4. *supplier*

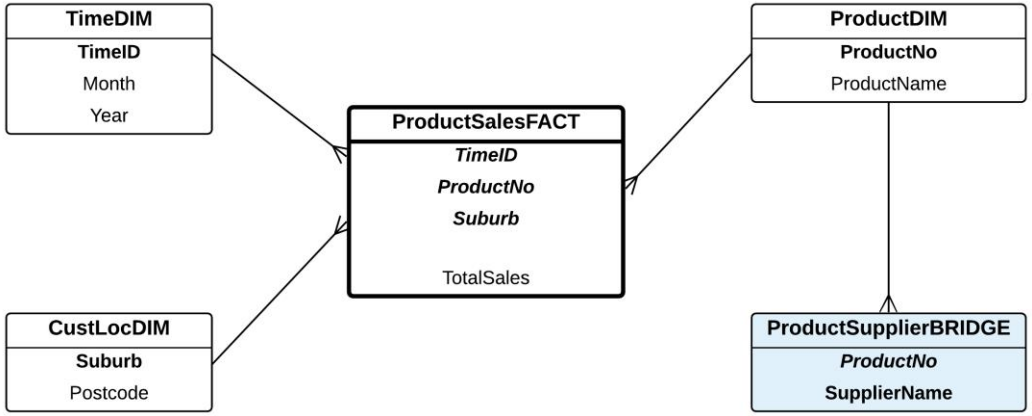


1. A Product Sales Case Study

Snowflake Schema with a Bridge Table, but without a Supplier Dimension



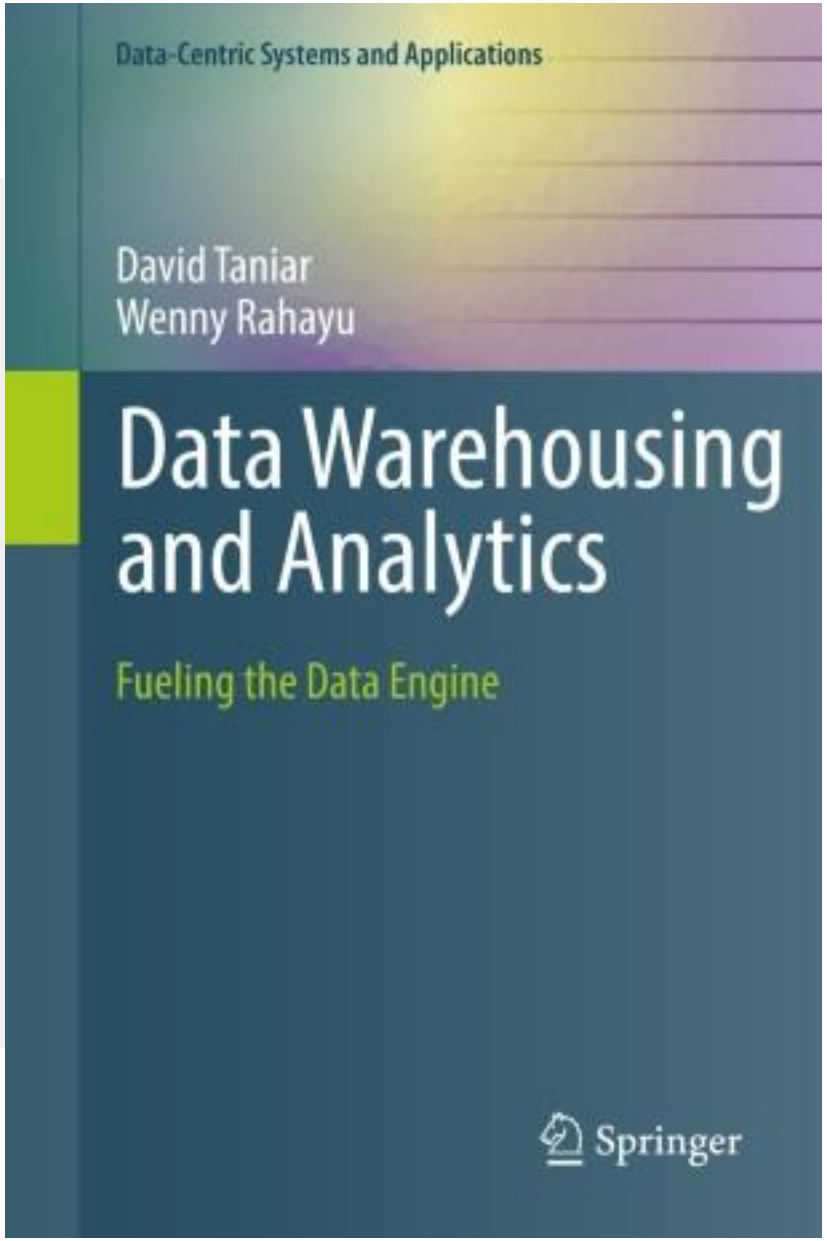
Snowflake Schema with a Bridge Table, but without maintaining the history of supplies



Summary

- Bridge Table is used:
  - There is no relationship between this dimension and the Fact table
  - When an entity (which will become a dimension) has a *many-many* relationship with another entity (dimension) in the E/R schema
  - When temporality aspect (data history) is maintained in the operational database and the bridge table can be used to accommodate the dimension that has temporal attributes
- When a Bridge Table is used in the schema, there are two additional options:
  - A Weight Factor is used to estimate the contribution of a dimension in the calculation of the fact measure.
  - Every snowflake schema (whether it has Weight Factor or not) can be implemented in two ways: a List Aggregate version, and a non-List Aggregate version.

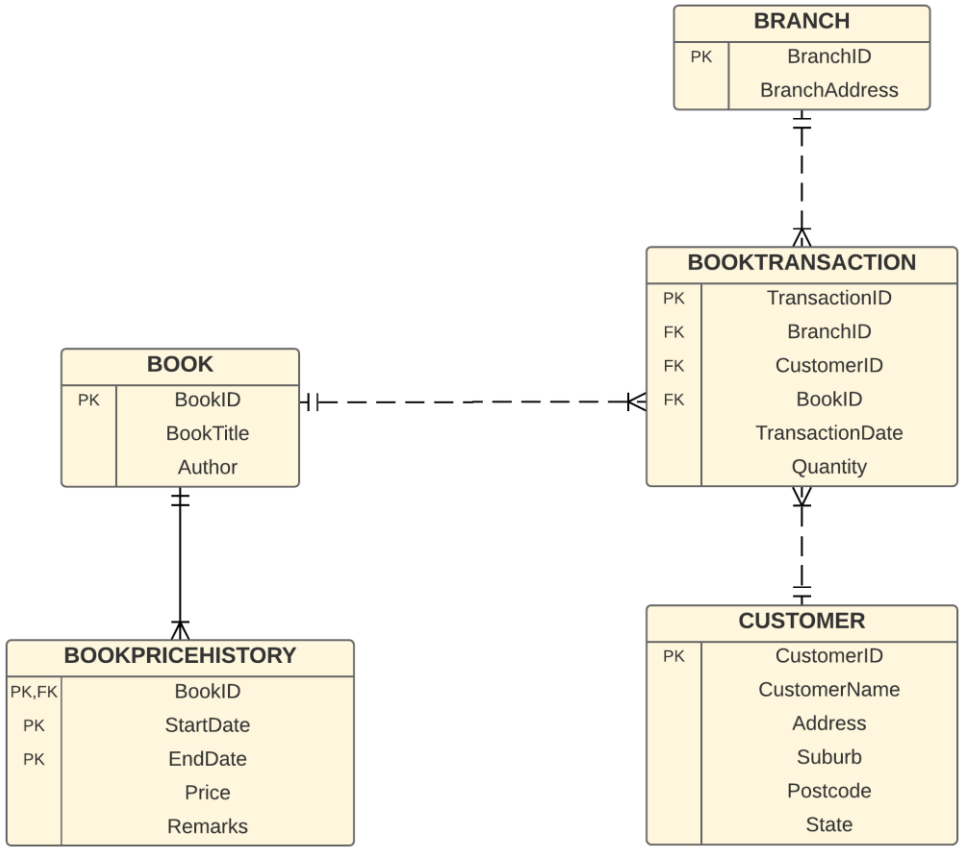
Chapter 6
Temporal Data Warehousing



Overview

- A warehousing technique whereby the temporal (or historical) aspect of records is incorporated into the data warehouse
- *Slowly Changing Dimensions* (*SCD*)

1. A Bookshop Case Study



- All book sales transactions are stored in an operational database
- Need to analyse book sales performance from various perspectives:
  - monthly basis
  - book basis
  - branch basis

1. A Bookshop Case Study

A Simple Star Schema

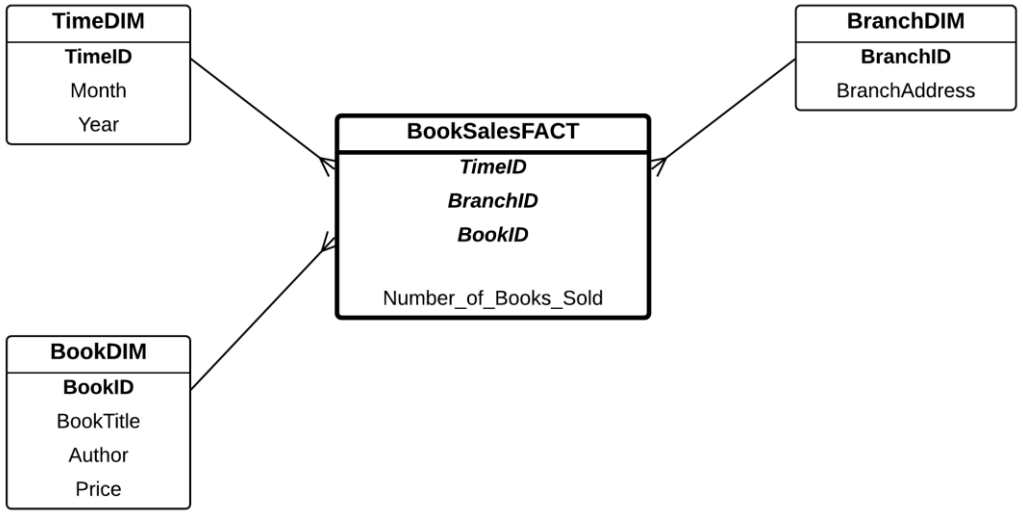


Table 6.3 Report 1 (Book Sales Fact with Book Dimension)

TimeID	BranchID	BookID	Book title	Author	Price	Number of books sold
Mar2008	City	C1	CSIRO Diet	CSIRO Team	$45.95	5
Mar2008	City	H6	Harry Potter 6	Rowling	$30.95	15
Mar2008	City	DV	Da Vinci Code	Dan Brown	$27.95	23
Mar2008	City	...	...	...	...	...
Mar2008	Chadstone	C1	CSIRO Diet	CSIRO Team	$45.95	15
Mar2008	Chadstone	H6	Harry Potter 6	Rowling	$30.95	3
Mar2008	Chadstone	DV	Da Vinci Code	Dan Brown	$27.95	2
Mar2008	Chadstone	...	...	...	...	...
Mar2008	Camberwell	C1	CSIRO Diet	CSIRO Team	$45.95	1
Mar2008	Camberwell	H6	Harry Potter 6	Rowling	$30.95	1
Mar2008	Camberwell	DV	Da Vinci Code	Dan Brown	$27.95	2
Mar2008	Camberwell	...	...	...	...	...
Mar2008	...	...	...	...	...	...
...	...	...	...	...	...	...
...	..	...	...	...	...	...
Dec2007	City	C1	CSIRO Diet	CSIRO Team	$45.95	15
Dec2007	City	H6	Harry Potter 6	Rowling	$30.95	6
Dec2007	City	DV	Da Vinci Code	Dan Brown	$27.95	6
Dec2007	City	..	...	...	...	...
Dec2007	Chadstone	C1	CSIRO Diet	CSIRO Team	$45.95	10
Dec2007	Chadstone	H6	Harry Potter 6	Rowling	$30.95	8
Dec2007	Chadstone	DV	Da Vinci Code	Dan Brown	$27.95	1
Dec2007	Chadstone	..	...	...	...	...
Dec2007	Camberwell	C1	CSIRO Diet	CSIRO Team	$45.95	18
Dec2007	Camberwell	H6	Harry Potter 6	Rowling	$30.95	3
Dec2007	Camberwell	DV	Da Vinci Code	Dan Brown	$27.95	2
Dec2007	Camberwell	..	...	...	...	...
Dec2007	...	...	...	...	...	...
...	..	...	...	...	...	...

1. A Bookshop Case Study

- Why C1 had more sales on Dec 2007?
- Price does not reflect actual selling prices
- Price column contains the *Current Price* of each book
- From time to time, price may change → temporal values

Table 6.3 Report 1 (Book Sales Fact with Book Dimension)

TimeID	BranchID	BookID	Book title	Author	Price	Number of books sold
Mar2008	City	C1	CSIRO Diet	CSIRO Team	$45.95	5
Mar2008	City	H6	Harry Potter 6	Rowling	$30.95	15
Mar2008	City	DV	Da Vinci Code	Dan Brown	$27.95	23
Mar2008	City	...	...	...	...	...
Mar2008	Chadstone	C1	CSIRO Diet	CSIRO Team	$45.95	15
Mar2008	Chadstone	H6	Harry Potter 6	Rowling	$30.95	3
Mar2008	Chadstone	DV	Da Vinci Code	Dan Brown	$27.95	2
Mar2008	Chadstone	...	...	...	...	...
Mar2008	Camberwell	C1	CSIRO Diet	CSIRO Team	$45.95	1
Mar2008	Camberwell	H6	Harry Potter 6	Rowling	$30.95	1
Mar2008	Camberwell	DV	Da Vinci Code	Dan Brown	$27.95	2
Mar2008	Camberwell	...	...	...	...	...
Mar2008	...	...	...	...	...	...
...	...	...	...	...	...	...
...	..	...	...	...	...	...
Dec2007	City	C1	CSIRO Diet	CSIRO Team	$45.95	15
Dec2007	City	H6	Harry Potter 6	Rowling	$30.95	6
Dec2007	City	DV	Da Vinci Code	Dan Brown	$27.95	6
Dec2007	City	..	...	...	...	...
Dec2007	Chadstone	C1	CSIRO Diet	CSIRO Team	$45.95	10
Dec2007	Chadstone	H6	Harry Potter 6	Rowling	$30.95	8
Dec2007	Chadstone	DV	Da Vinci Code	Dan Brown	$27.95	1
Dec2007	Chadstone	..	...	...	...	...
Dec2007	Camberwell	C1	CSIRO Diet	CSIRO Team	$45.95	18
Dec2007	Camberwell	H6	Harry Potter 6	Rowling	$30.95	3
Dec2007	Camberwell	DV	Da Vinci Code	Dan Brown	$27.95	2
Dec2007	Camberwell	..	...	...	...	...
Dec2007	...	...	...	...	...	...
...	..	...	...	...	...	...

1. A Bookshop Case Study

- Bridge Table to the Book Dimension, to store the history of book prices

Table 6.4 Book Dimension table

BookID	Book title	Author
C1	CSIRO Diet	CSIRO Team
H6	Harry Potter 6	Rowling
DV	Da Vinci Code	Dan Brown
---	---	---

Table 6.5 Book Price Dimension table

BookID	Start date	End date	Price	Remarks
C1	Jan2007	July2007	$45.95	Full Price
C1	Aug2007	Oct2007	$36.75	20% Discount
C1	Nov2007	Jan2008	$23.00	Half Price
C1	Feb2008	Dec9999	$45.95	Full Price
H6	Jan2007	Mar2007	$21.95	Launching
H6	Apr2007	Jan2008	$30.95	Full Price
H6	Feb2008	Dec9999	$10.00	End of Product Sale
DV	Jan2007	Dec9999	$27.95	Full Price
---	---	---	---	---

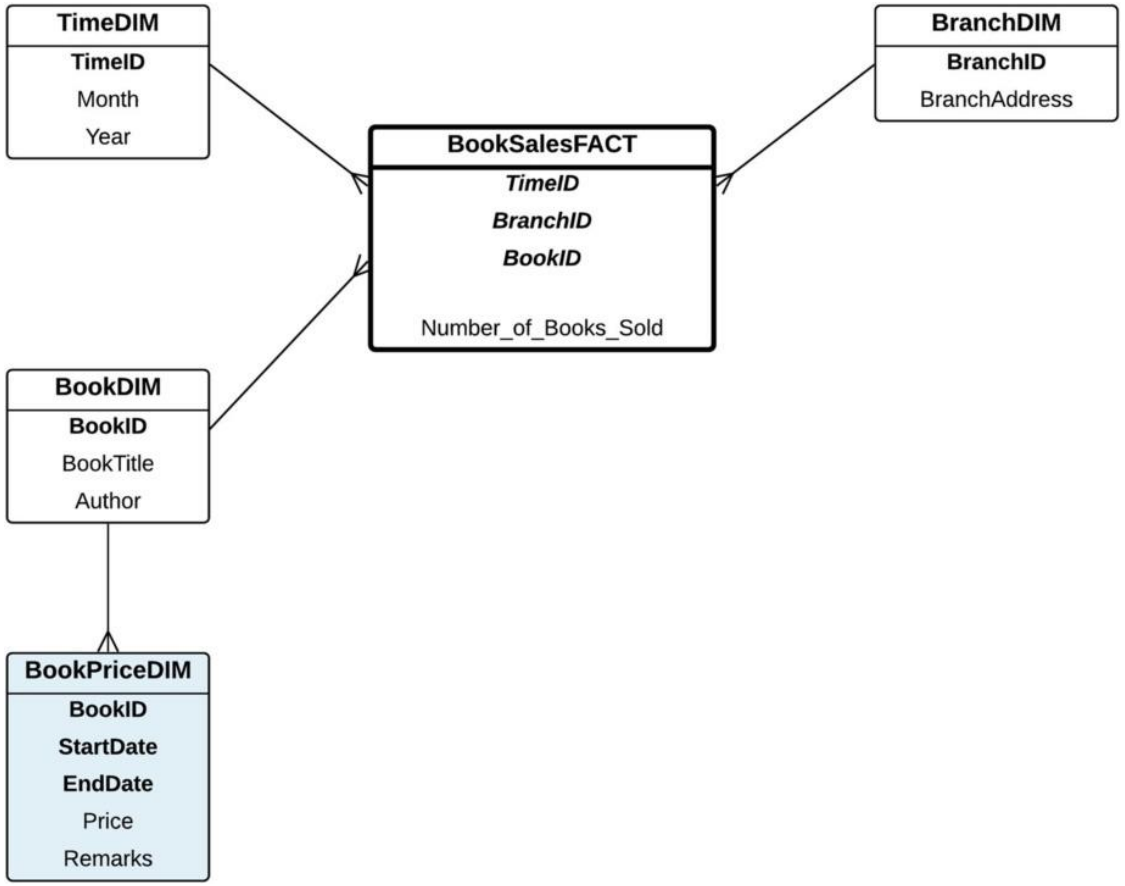


Fig. 6.2 A temporal data warehousing star schema

1. A Bookshop Case Study

Table 6.4 Book Dimension table

BookID	Book title	Author
C1	CSIRO Diet	CSIRO Team
H6	Harry Potter 6	Rowling
DV	Da Vinci Code	Dan Brown
---	---	---

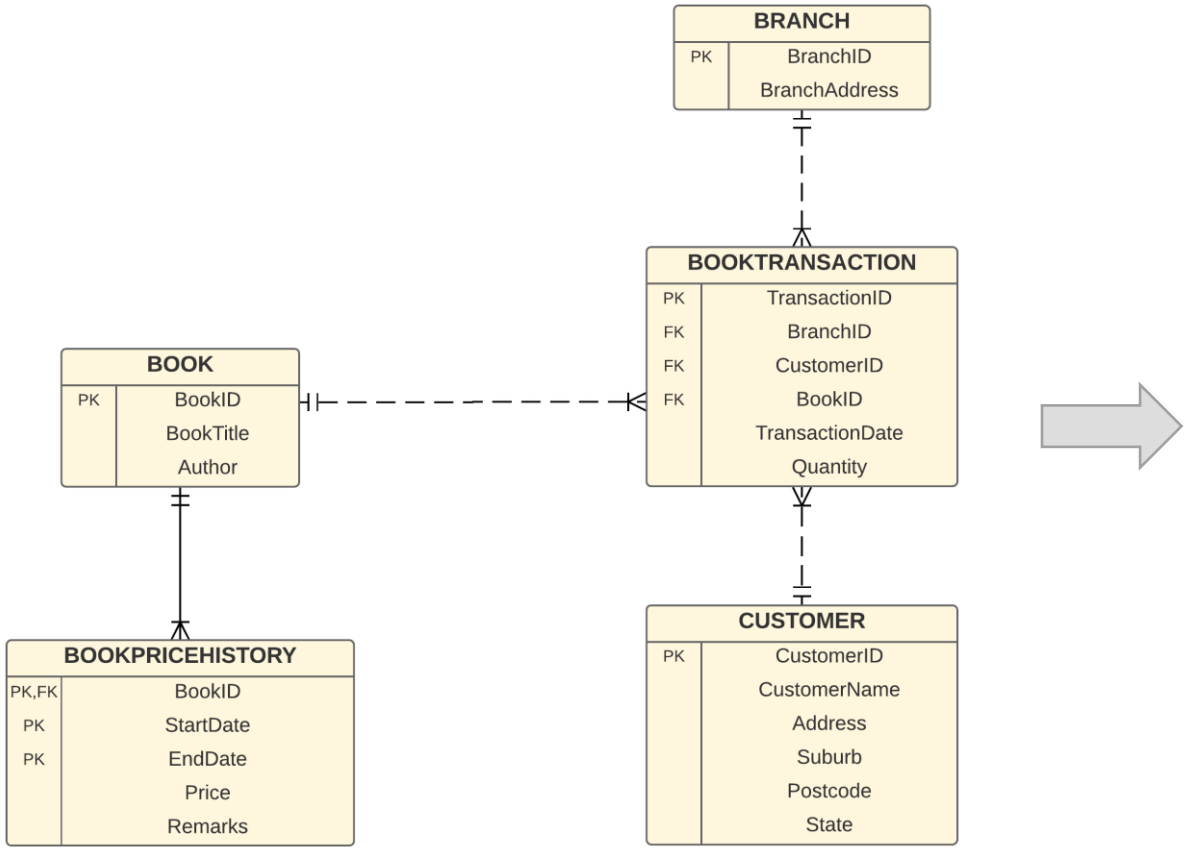
Table 6.5 Book Price Dimension table

BookID	Start date	End date	Price	Remarks
C1	Jan2007	July2007	$45.95	Full Price
C1	Aug2007	Oct2007	$36.75	20% Discount
C1	Nov2007	Jan2008	$23.00	Half Price
C1	Feb2008	Dec9999	$45.95	Full Price
H6	Jan2007	Mar2007	$21.95	Launching
H6	Apr2007	Jan2008	$30.95	Full Price
H6	Feb2008	Dec9999	$10.00	End of Product Sale
DV	Jan2007	Dec9999	$27.95	Full Price
---	---	---	---	---

Table 6.6 Report 2 with the correct Book Price

TimeID	BranchID	BookID	Book title	Author	Price	Number of books sold
Mar2008	City	C1	CSIRO Diet	CSIRO Team	$45.95	5
Mar2008	City	H6	Harry Potter 6	Rowling	$10.00	15
Mar2008	City	DV	Da Vinci Code	Dan Brown	$27.95	23
Mar2008	City	---	---	---	---	---
Mar2008	Chadstone	C1	CSIRO Diet	CSIRO Team	$45.95	15
Mar2008	Chadstone	H6	Harry Potter 6	Rowling	$10.00	3
Mar2008	Chadstone	DV	Da Vinci Code	Dan Brown	$27.95	2
Mar2008	Chadstone	---	---	---	---	---
Mar2008	Camberwell	C1	CSIRO Diet	CSIRO Team	$45.95	1
Mar2008	Camberwell	H6	Harry Potter 6	Rowling	$10.00	1
Mar2008	Camberwell	DV	Da Vinci Code	Dan Brown	$27.95	2
Mar2008	Camberwell	---	---	---	---	---
Mar2008	---	---	---	---	---	---
---	---	---	---	---	---	---
---	---	---	---	---	---	---
Dec2007	City	C1	CSIRO Diet	CSIRO Team	$23.00	15
Dec2007	City	H6	Harry Potter 6	Rowling	$30.95	6
Dec2007	City	DV	Da Vinci Code	Dan Brown	$27.95	6
Dec2007	City	--	---	---	---	---
Dec2007	Chadstone	C1	CSIRO Diet	CSIRO Team	$23.00	10
Dec2007	Chadstone	H6	Harry Potter 6	Rowling	$30.95	8
Dec2007	Chadstone	DV	Da Vinci Code	Dan Brown	$27.95	1
Dec2007	Chadstone	--	---	---	---	---
Dec2007	Camberwell	C1	CSIRO Diet	CSIRO Team	$23.00	18
Dec2007	Camberwell	H6	Harry Potter 6	Rowling	$30.95	3
Dec2007	Camberwell	DV	Da Vinci Code	Dan Brown	$27.95	2
Dec2007	Camberwell	--	---	---	---	---
Dec2007	---	---	---	---	---	---
---	---	---	---	---	---	---

2. Implementation of Temporal Data Warehousing



2 Implementation of Temporal Data Warehousing

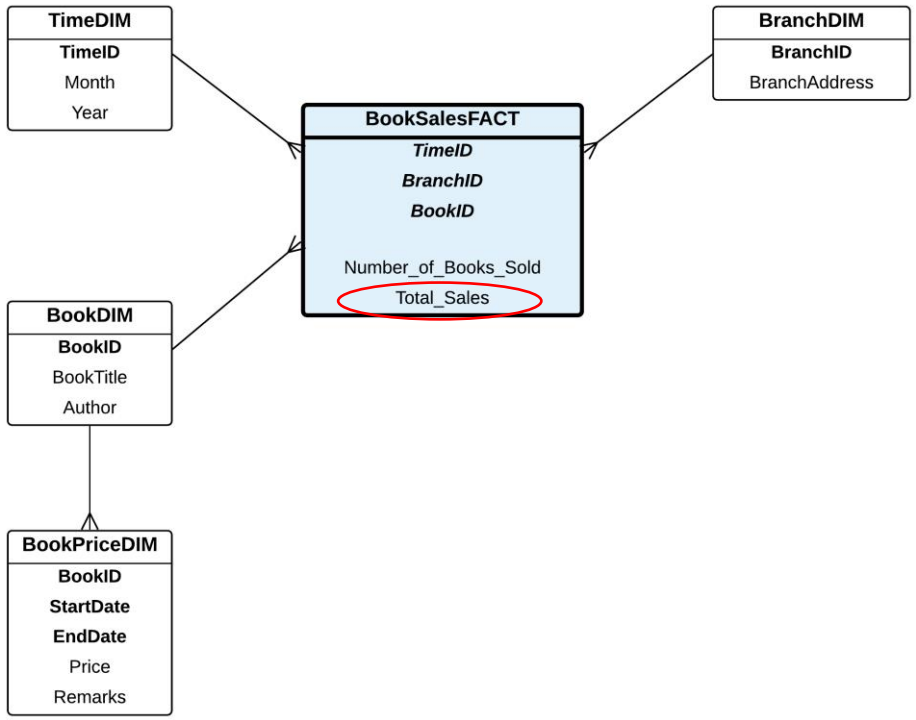
Table 1.6 Report 2 with the correct Book Price

TimeID	BranchID	BookID	Book Title	Author	Price	Number of Books Sold
Mar2008	City	C1	CSIRO Diet	CSIRO Team	$45.95	5
Mar2008	City	H6	Harry Potter 6	Rowling	**$10.00**	15
Mar2008	City	DV	Da Vinci Code	Dan Brown	$27.95	23
Mar2008	City	...	...	...	...	...
Mar2008	Chadstone	C1	CSIRO Diet	CSIRO Team	$45.95	15
Mar2008	Chadstone	H6	Harry Potter 6	Rowling	**$10.00**	3
Mar2008	Chadstone	DV	Da Vinci Code	Dan Brown	$27.95	2
Mar2008	Chadstone	...	...	...	...	...
Mar2008	Camberwell	C1	CSIRO Diet	CSIRO Team	$45.95	1
Mar2008	Camberwell	H6	Harry Potter 6	Rowling	**$10.00**	1
Mar2008	Camberwell	DV	Da Vinci Code	Dan Brown	$27.95	2
Mar2008	Camberwell	...	...	...	...	...
Mar2008	...	...	...	...	...	...
...	...	...	...	...	...	...
...	...	...	...	...	...	...
Dec2007	City	C1	CSIRO Diet	CSIRO Team	**$23.00**	15
Dec2007	City	H6	Harry Potter 6	Rowling	$30.95	6
Dec2007	City	DV	Da Vinci Code	Dan Brown	$27.95	6
Dec2007	City	..	...	...	...	...
Dec2007	Chadstone	C1	CSIRO Diet	CSIRO Team	**$23.00**	10
Dec2007	Chadstone	H6	Harry Potter 6	Rowling	$30.95	8
Dec2007	Chadstone	DV	Da Vinci Code	Dan Brown	$27.95	1
Dec2007	Chadstone	..	...	...	...	...
Dec2007	Camberwell	C1	CSIRO Diet	CSIRO Team	**$23.00**	18
Dec2007	Camberwell	H6	Harry Potter 6	Rowling	$30.95	3
Dec2007	Camberwell	DV	Da Vinci Code	Dan Brown	$27.95	2
Dec2007	Camberwell	..	...	...	...	...
Dec2007	...	...	...	...	...	...
...	...	...	...	...	...	...

- More sales on Dec 2007 was due to discount price

2. Implementation of Temporal Data Warehousing

A new fact attribute: Total Sales



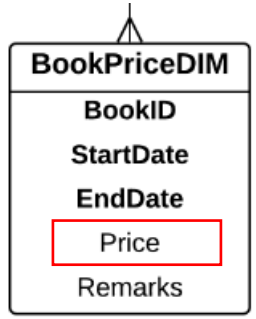
- A new additional fact measure is added → *Total Sales*
- Assumes that BookSalesFact1 has been created

3. Temporal Attributes and Temporal Dimensions

- **Temporal attribute**
  An attribute in which the value of that attribute has a life-time
- **Temporal Dimension**
  A dimension where the record of the dimension has a life-time

3.1. Temporal Attributes

- An attribute in which the value of that attribute has a life-time
- Each book price has a life-time, and the life-time is determined by the Start Date and End Date attributes in the Book Price Dimension table
- Example:
  The book price of $45.95 of Book C1 is only valid betw - 2007 and July-2007.



3.1. Temporal Attributes

The correct report

```
select
   F.TimeID,
   F.BranchID,
   F.BookID,
   B.BookTitle.
   B.Author,
   listagg(P.Price, ';') within group (order by P.Price)
     as Price,
   F.Number_of_Books_Sold
from BookSalesFact F, BookDim B, BookPriceDim P
where F.BookID = B.BookID
and B.BookID = P.BookID
and to_date(F.TimeID, 'MonYYYY') >=
  to_date(P.StartDate, 'MonYYYY')
and to_date(F.TimeID, 'MonYYYY') <=
  to_date(P.EndDate, 'MonYYYY')
group by
   F.TimeID,
   F.BranchID,
   F.BookID,
   B.BookTitle,
   B.Author,
   F.Number_of_Books_Sold;
```

Table 6.9 Report 4: multiple book prices on one month

TimeID	BranchID	BookID	Book title	Author	Price	Number of books Sold
Jan2008	City	C1	CSIRO Diet	CSIRO Team	$23.00;$45.95	25
Jan2008	City	H6	Harry Potter 6	Rowling	$30.95	10
Jan2008	City	DV	Da Vinci Code	Dan Brown	$27.95	7
Jan2008	City	...	...	...	...	...
Jan2008	Chadstone	C1	CSIRO Diet	CSIRO Team	$23.00;$45.95	30
Jan2008	Chadstone	H6	Harry Potter 6	Rowling	$30.95	15
Jan2008	Chadstone	DV	Da Vinci Code	Dan Brown	$27.95	5
Jan2008	Chadstone	...	...	...	...	...
Jan2008	Camberwell	C1	CSIRO Diet	CSIRO Team	$23.00;$45.95	20
Jan2008	Camberwell	H6	Harry Potter 6	Rowling	$30.95	5
Jan2008	Camberwell	DV	Da Vinci Code	Dan Brown	$27.95	5
Jan2008	Camberwell	...	...	...	...	...
Jan2008	...	...	...	...	...	...
...	...	...	...	...	...	...

4. Slowly Changing Dimensions (SCD)

- Temporal data warehousing = Slowly Changing Dimensions or SCD
- Dimensions where the records of these dimensions change slowly over a period of time
  - Not applicable to rapidly changed data
- Example:
  - The price of a book changes "slowly" over time
- Several SCD Types:
  - Original: Type 1, Type 2, Type 3
  - New Types: Type 0, Type 4, Type 6

4.1. SCD Type 0 and Type 1

Type 0

- Stores the "Original or Initial" value of the records

BookID	Book Title	Author	Price
C1	CSIRO Diet	CSIRO Team	$45.95
H6	Harry Potter 6	Rowling	$30.95
DV	Da Vinci Code	Dan Brown	$27.95
...	...	...	...

Type 1

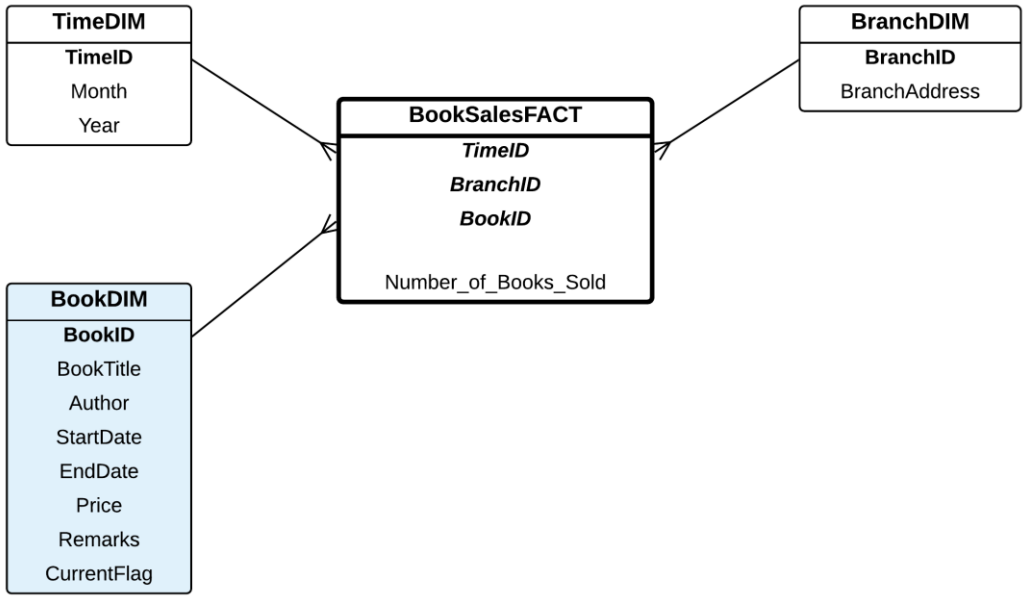
- Records the latest value of the records

BookID	Book Title	Author	Price
C1	CSIRO Diet	CSIRO Team	$45.95
H6	Harry Potter 6	Rowling	**$10.00**
DV	Da Vinci Code	Dan Brown	$27.95
...	...	...	...

Similarity: do not actually record the history of changes in the dimension

4.2. SCD Type 2

- Keeps track the history
- Not separating the history from the main dimension
- The new records keep added to the dimension



4.2. SCD Type 2

- Add sequence number to the original BookID to differentiate the same book in different time period

Table 6.15 Book Dimension table (SCD Type 2)

BookID	Book title	Author	Start Date	End Date	Price	Remarks	Current Flag
C1_1	CSIRO Diet	CSIRO Team	Jan2007	Jul2007	$45.95	Full Price	N
C1_2	CSIRO Diet	CSIRO Team	Aug2007	Oct2007	$36.75	20% Discount	N
C1_3	CSIRO Diet	CSIRO Team	Nov2007	Jan2008	$23.00	Half Price	N
C1_4	CSIRO Diet	CSIRO Team	Feb2008	Dec9999	$45.95	Full Price	Y
H6_1	Harry Potter 6	Rowling	Jan2007	Mar2007	$21.95	Launching	N
H6_2	Harry Potter 6	Rowling	Apr2007	Jan2008	$30.95	Full Price	N
H6_3	Harry Potter 6	Rowling	Feb2008	Dec9999	$10.00	End of Product Sale	Y
DV_1	Da Vinci Code	Dan Brown	Jan2007	Dec9999	$27.95	Full Price	Y
...	...	...	...	...	...	...	...

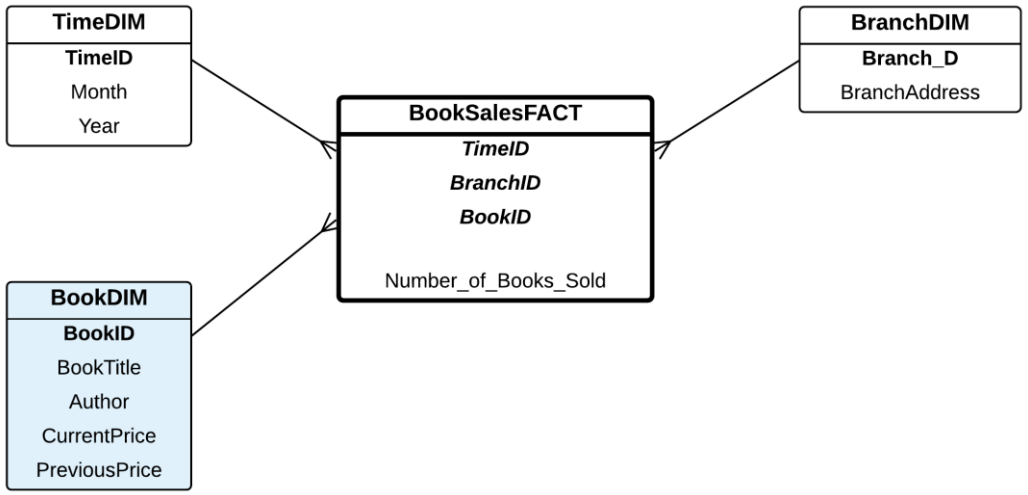
Table 6.16 Report 3 (SCD Type 2)

TimeID	BranchID	BookID	Book title	Author	Price	Number of books sold
Mar2008	City	C1_4	CSIRO Diet	CSIRO Team	$45.95	5
Mar2008	City	H6_3	Harry Potter 6	Rowling	$10.00	15
Mar2008	City	DV_1	Da Vinci Code	Dan Brown	$27.95	23
Mar2008	City	...	...	...	...	...
Mar2008	Chadstone	C1_4	CSIRO Diet	CSIRO Team	$45.95	15
Mar2008	Chadstone	H6_3	Harry Potter 6	Rowling	$10.00	3
Mar2008	Chadstone	DV_1	Da Vinci Code	Dan Brown	$27.95	2
Mar2008	Chadstone	...	...	...	...	...
Mar2008	Camberwell	C1_4	CSIRO Diet	CSIRO Team	$45.95	1
Mar2008	Camberwell	H6_3	Harry Potter 6	Rowling	$10.00	1
Mar2008	Camberwell	DV_1	Da Vinci Code	Dan Brown	$27.95	2
Mar2008	Camberwell	...	...	...	...	...
Mar2008	...	...	...	...	...	...
...	...	...	...	...	...	...
...	...	...	...	...	...	...
Dec2007	City	C1_3	CSIRO Diet	CSIRO Team	$23.00	15
Dec2007	City	H6_2	Harry Potter 6	Rowling	$30.95	6
Dec2007	City	DV_1	Da Vinci Code	Dan Brown	$27.95	6
Dec2007	City	..	...	...	...	...
Dec2007	Chadstone	C1_3	CSIRO Diet	CSIRO Team	$23.00	10
Dec2007	Chadstone	H6_2	Harry Potter 6	Rowling	$30.95	8
Dec2007	Chadstone	DV_1	Da Vinci Code	Dan Brown	$27.95	1
Dec2007	Chadstone	..	...	...	...	...
Dec2007	Camberwell	C1_3	CSIRO Diet	CSIRO Team	$23.00	18
Dec2007	Camberwell	H6_2	Harry Potter 6	Rowling	$30.95	3
Dec2007	Camberwell	DV_1	Da Vinci Code	Dan Brown	$27.95	2
Dec2007	Camberwell	..	...	...	...	...
Dec2007	...	...	...	...	...	...
...	...	...	...	...	...	...

4.3. SCD Type 3

- Simplification of Type 2
- Single value for each entry
- Assumed, complete history is not necessary
- Example: Records current and previous price

BookID	Book Title	Author	Current Price	Previous Price
C1	CSIRO Diet	CSIRO Team	**$45.95**	**$23.00**
H6	Harry Potter 6	Rowling	**$10.00**	**$30.95**
DV	Da Vinci Code	Dan Brown	**$27.95**	**Null**
...	...	...	...	...



4.4. SCD Type 4

- Create a new dimension to maintain the history of attribute value change.
- Book Dimension table is kept without the price attribute.
- The price attribute (and Start Date and End Date) are separated into another table
- Do not need to have a different BookID for the same book

Book Dimension Table

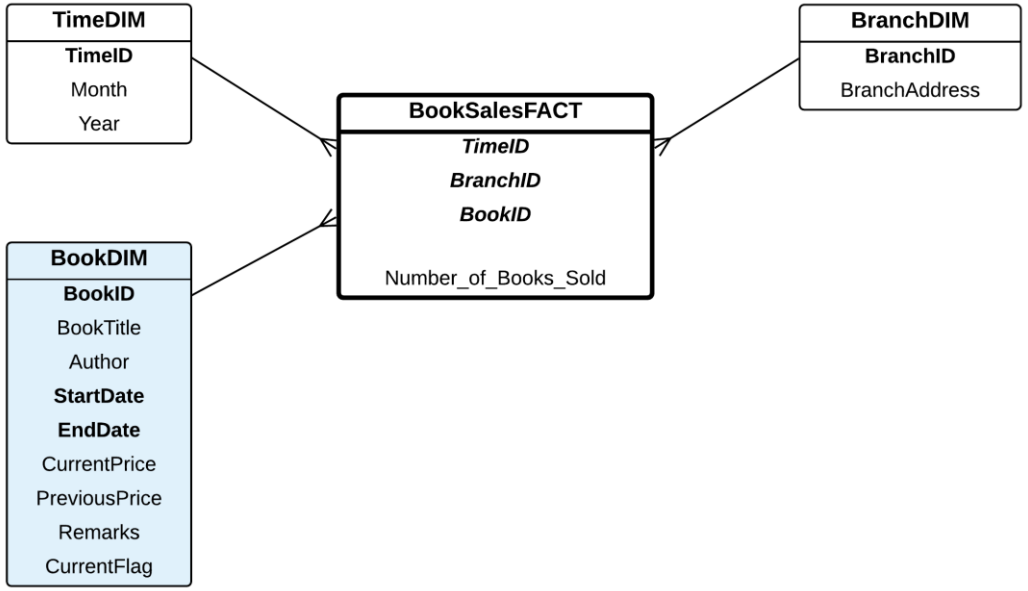
BookID	Book Title	Author
C1	CSIRO Diet	CSIRO Team
H6	Harry Potter 6	Rowling
DV	Da Vinci Code	Dan Brown
...	...	...

Book Price Dimension Table

BookID	Start Date	End Date	Price	Remarks
C1	Jan2007	July2007	$45.95	Full Price
C1	Aug2007	Oct2007	$36.75	20% Discount
C1	Nov2007	Jan2008	$23.00	Half Price
C1	Feb2008	Dec9999	$45.95	Full Price
H6	Jan2007	Mar2007	$21.95	Launching
H6	Apr2007	Jan2008	$30.95	Full Price
H6	Feb2008	Dec9999	$10.00	End of Product Sale
DV	Jan2007	Dec9999	$27.95	Full Price
...	...	...	...	...

4.5. SCD Type 6

- Combination between Type 2 and Type 3
- A separate identifier for the same book is not needed, but the entire history is kept



4.5 SCD Type 6

Table 6.15 Book Dimension table (SCD Type 2)

BookID	Book title	Author	Start Date	End Date	Price	Remarks	Current Flag
C1_1	CSIRO Diet	CSIRO Team	Jan2007	Jul2007	$45.95	Full Price	N
C1_2	CSIRO Diet	CSIRO Team	Aug2007	Oct2007	$36.75	20% Discount	N
C1_3	CSIRO Diet	CSIRO Team	Nov2007	Jan2008	$23.00	Half Price	N
C1_4	CSIRO Diet	CSIRO Team	Feb2008	Dec9999	$45.95	Full Price	Y
H6_1	Harry Potter 6	Rowling	Jan2007	Mar2007	$21.95	Launching	N
H6_2	Harry Potter 6	Rowling	Apr2007	Jan2008	$30.95	Full Price	N
H6_3	Harry Potter 6	Rowling	Feb2008	Dec9999	$10.00	End of Product Sale	Y
DV_1	Da Vinci Code	Dan Brown	Jan2007	Dec9999	$27.95	Full Price	Y
...	...	...	...	...	...	...	...

Book Dimension Table (SCD Type 3)

BookID	Book Title	Author	Current Price	Previous Price
C1	CSIRO Diet	CSIRO Team	**$45.95**	**$23.00**
H6	Harry Potter 6	Rowling	**$10.00**	**$30.95**
DV	Da Vinci Code	Dan Brown	**$27.95**	**Null**
...	...	...	...	...

Table 6.20 Book Dimension (SCD Type 6)

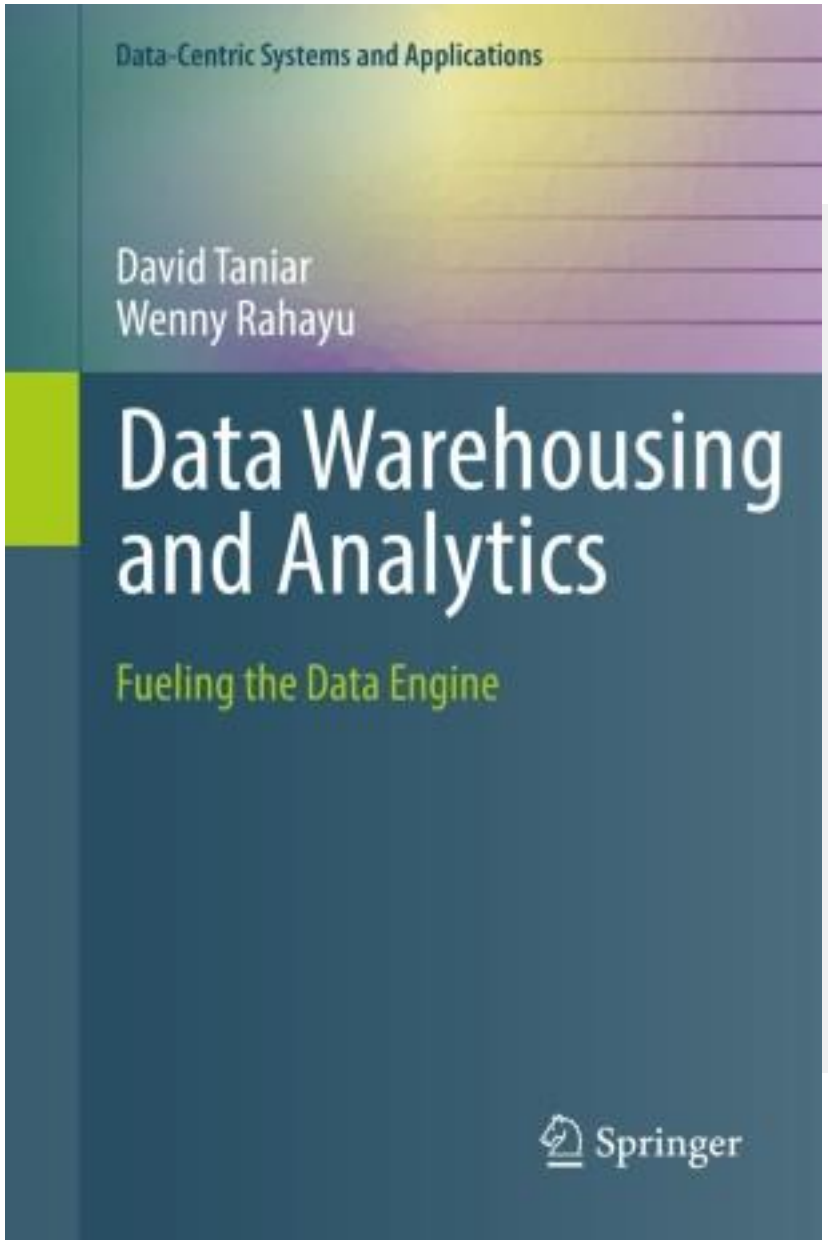
Book ID	Book Title	Author	Start Date	End Date	Current Price	Previous Price	Remarks	Current Flag
C1	CSIRO Diet	CSIRO Team	Jan2007	Jul2007	$45.95	Null	Full Price	N
C1	CSIRO Diet	CSIRO Team	Aug2007	Oct2007	$36.75	$45.95	20% Discount	N
C1	CSIRO Diet	CSIRO Team	Nov2007	Jan2008	$23.00	$36.75	Half Price	N
C1	CSIRO Diet	CSIRO Team	Feb2008	Dec9999	$45.95	$23.00	Full Price	Y
H6	Harry Potter 6	Rowling	Jan2007	Mar2007	$21.95	Null	Launching	N
H6	Harry Potter 6	Rowling	Apr2007	Jan2008	$30.95	$21.95	Full Price	N
H6	Harry Potter 6	Rowling	Feb2008	Dec9999	$10.00	$30.95	End of Product	Y
DV	Da Vinci Code	Dan Brown	Jan2007	Dec9999	$27.95	Null	Full Price	Y
---	---	---	---	---	---	---	---	---

4.5. SCD Type 6

- The cardinality relationship between Book Dimension table and Fact table is no longer 1-*m*, but *m*-*m*
- Three options to solve the problem:
  1. Add new surrogate key to Book Dimension table
  2. Add Start Date and End Date to Fact table, in addition to the BookID
  3. Add associative table (or a bridge table) between Book Dimension and Fact

Summary

- A temporal data warehousing uses the concept of the Bridge Table (or a Weak Entity)
- The history is maintained in a bridge table
- Maintaining the history of certain attributes is important when analysing the reports produced by the fact and dimensions
- Temporal data warehousing is known as *Slowly Changing Dimensions* (*SCD*)
- Several SCD types are discussed



Chapter 7

Determinant Dimensions

Outline

- In certain circumstances, a star schema may have a special dimension(or a key identifier) which must be used for all data retrieval or else the retrieved data will be meaningless. This dimension is called a **Determinant Dimension**, and the key identifier of this attribute is called a **Determinant Attribute**.
- In this chapter, we are going to learn about scenarios when a determinant dimension is used and when it is not necessarily to use a determinant dimension. Several case studies to explore the complexities of determinant dimensions.

Outline

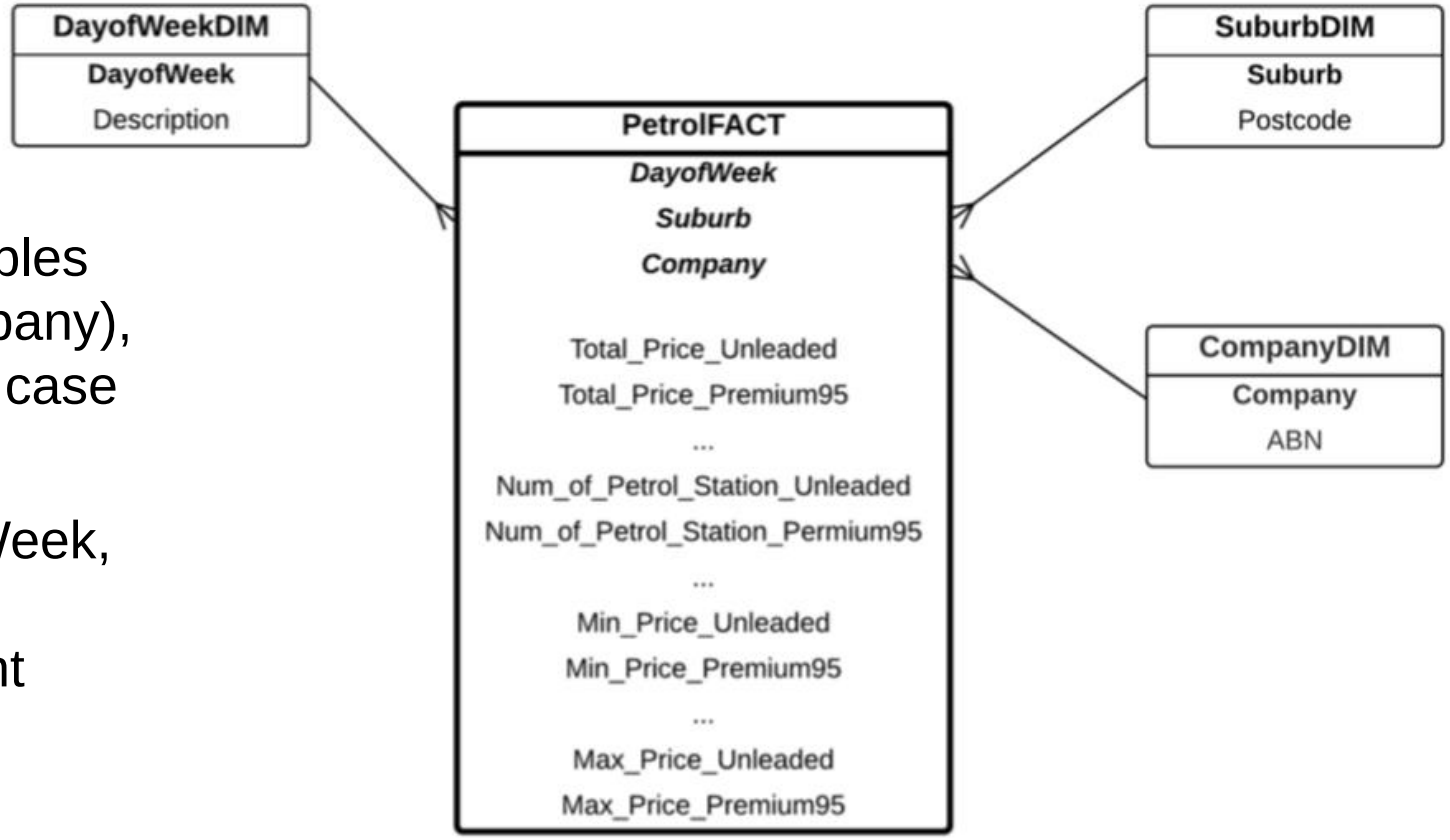
1. The **Petrol Station case study**, which illustrates the use of a determinant dimension
2. The **Olympic Games case study**, which compares and contrasts determinant and non-determinant dimensions
3. The **PTE Academic Test case study**, which explores the technical aspects and the complexities of both determinant and non-determinant dimensions
4. The **University's Student Population case study** to illustrate the use of year in the determinant dimension
5. The **Private Taxi Company case study** to explore the impact of complex dimensions on pivot tables

1. Introducing a Determinant Dimension – Petrol Station Case Study

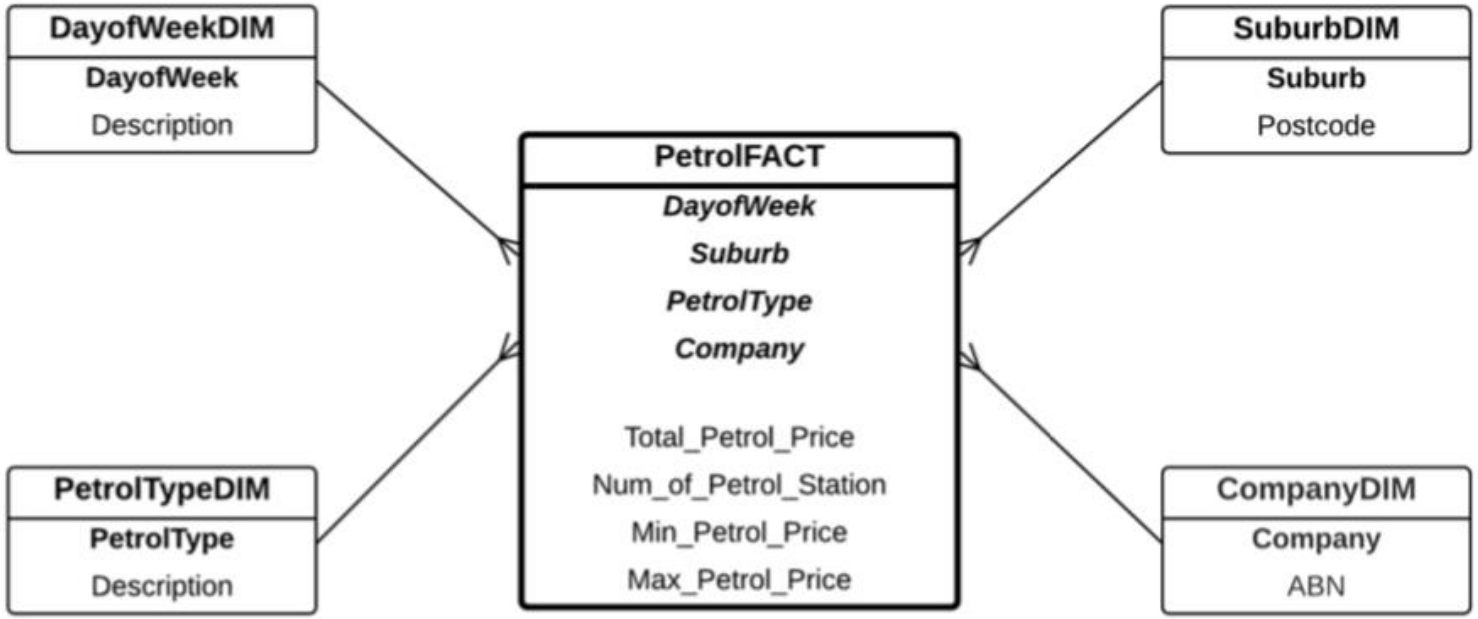
- The Petrol Station case study is about daily petrol prices in Victoria, Australia. Unlike petrol prices in other parts of the world, petrol prices in Victoria (Australia) fluctuate every day.
- A company that maintains a database of all petrol prices from all petrol stations in Victoria.
- In this section, we are going to examine two versions of the star schema for this Petrol Station case study.

1.1. Petrol Station Star Schema Version- 1

- Based on these three Two-Column tables (e.g. Day of Week, Suburb, and Company), the star schema for the Petrol Station case study is shown next page.
- There are three dimensions: Day of Week, Suburb, and Company and there are twenty-four fact measures, six different petrol types for total price, number of stations, and minimum and maximum prices.



1.2. Petrol Station Star Schema Version- 2

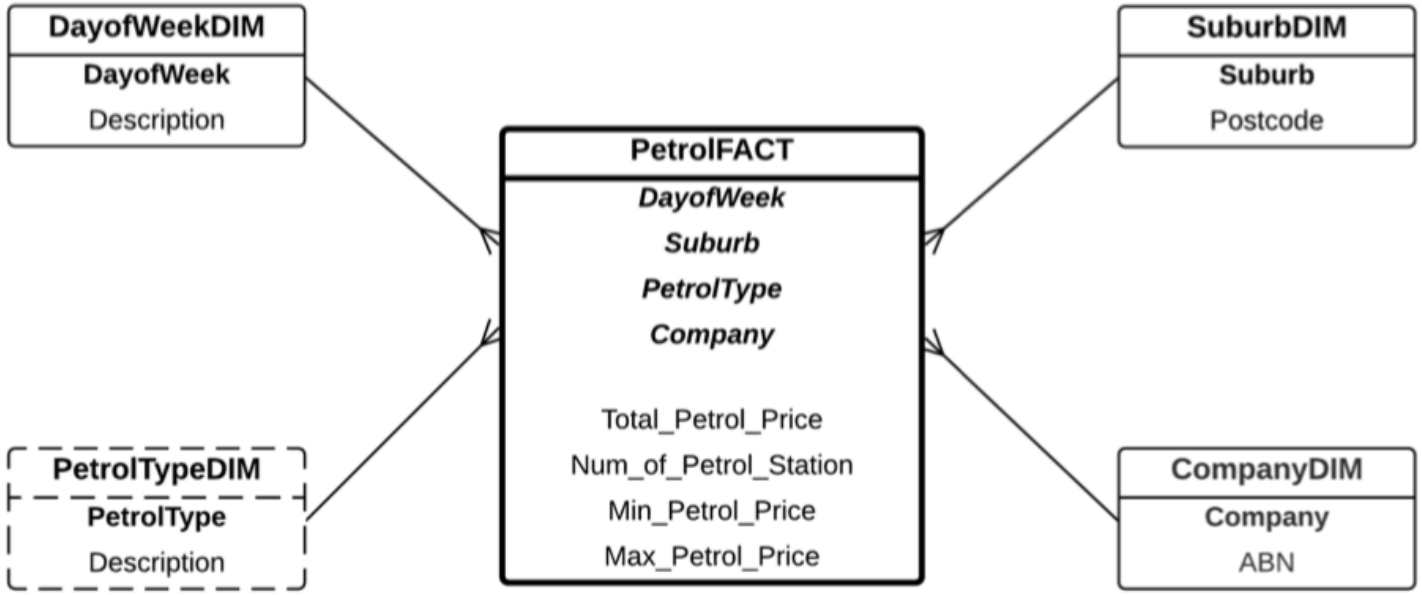


- Another way to reduce the number of fact measures, is to have a new dimension called **Petrol Type Dimension** which stores the different kinds of petrol.

1.2. Petrol Station Star Schema Version- 2

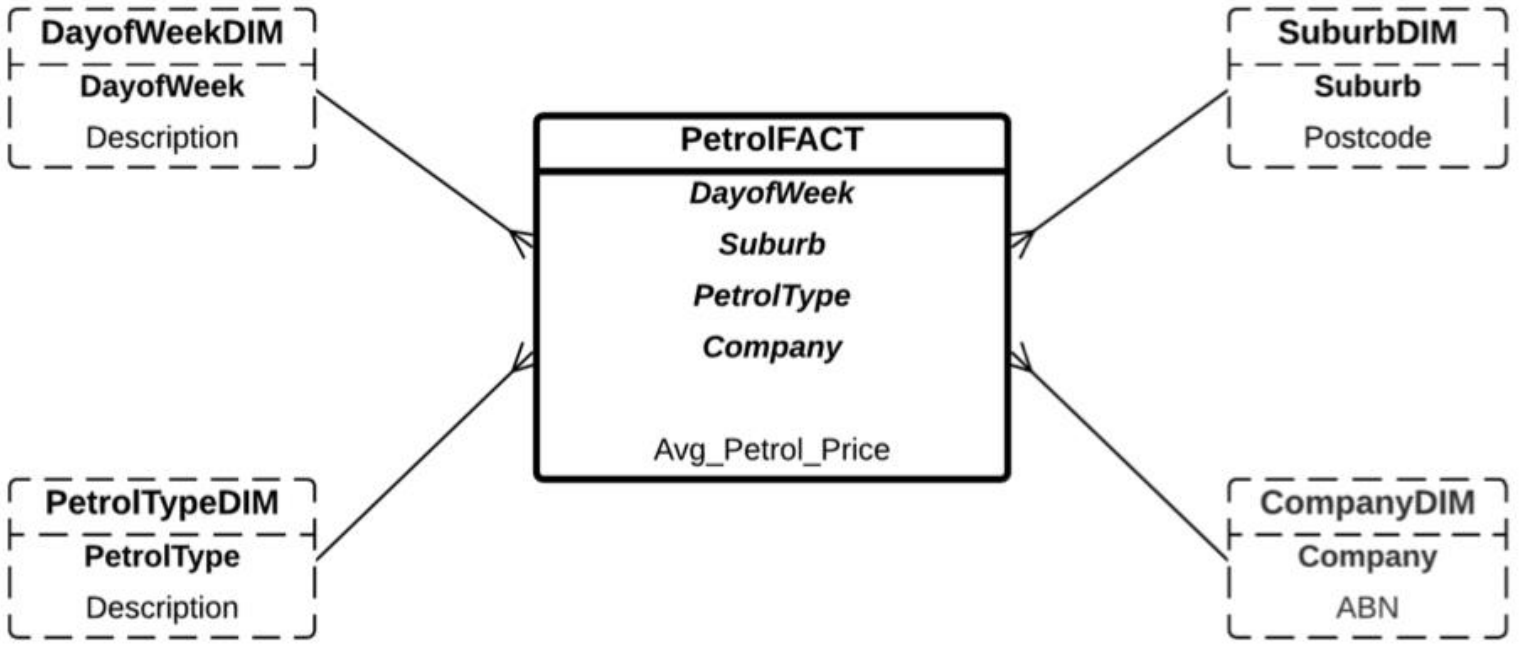
- Petrol Type Dimension, in this example, is called a **Determinant Dimension**, a dimension which must be used in the data retrieval because the fact measures are determined by this dimension and its key identifier (e.g. Petrol Type attribute) is called a **Determinant Attribute**.

1.2. Petrol Station Star Schema Version- 2



1.2. Petrol Station Star Schema Version- 2

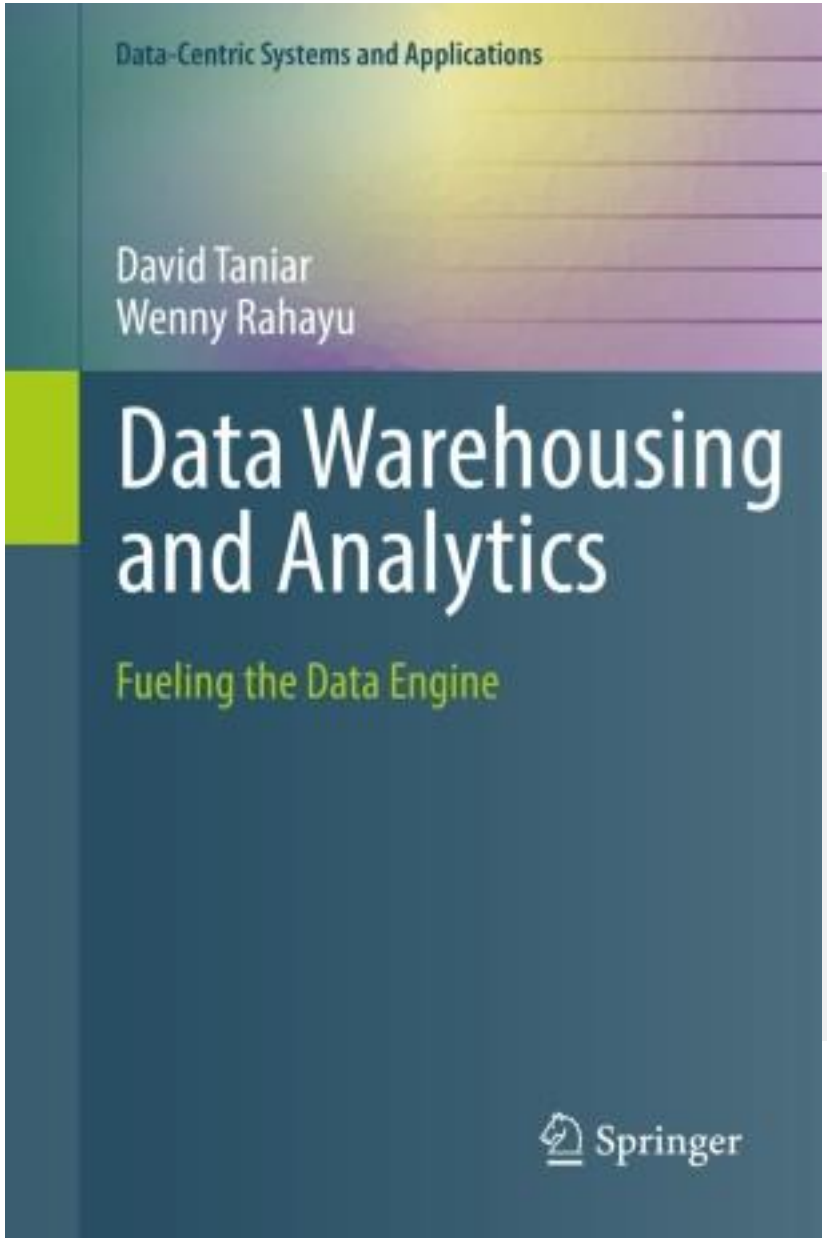
- If all dimensions are used in the data retrieval, the average fact measure is perfectly fine because there is no second average function to be used anymore.



1.2. Petrol Station Star Schema Version- 2

- ***In SQL, how can we ensure that a certain attribute is used?***
- ***How can we ensure that the Determinant Dimension or Determinant Attribute are used in data retrieval?***

- The first one is to covert the star schema that features a Determinant Dimension to a star schema where all the required information is captured in the fact measures ➔ **Pivoted Fact Table**
- The second method is to enforce the use of the Determinant Dimension or Determinant Attribute at the **User Interface Level**, where the users are forced to choose value(s) from the required Determinant Attribute.



Chapter 8

Junk Dimensions

Outline

- A **Junk Dimension** is a dimension that combines several “low cardinality” dimensions.
- This chapter focuses on how a junk dimension is used in a star schema, and to compare and to contrast two approaches, which are:
  1. Non-junk dimension
  2. Junk dimension
- Case Study on Junk Dimensions

1. A Real-Estate Case Study

- An established real estate agent has started their business many years ago and has implemented a very simple database system. The simple database system consists of one large table with the related attributes.

Table 1.4 Property Table

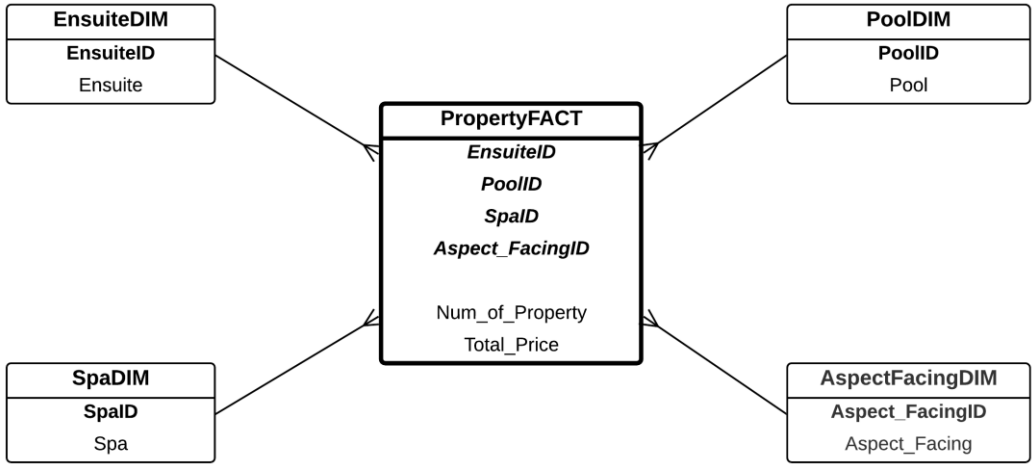
Attribute Name	Description
Key	Unique key
Date_offered	Date property offered to the public
Summary	Short description of the property
Adtext	Longer description of the property
URL	The URL of the advertisement
Address	Property address
Suburb	Property suburb name
Postcode	Property postcode
Longitude	Longitude of address
Latitude	Latitude of address
Category	Residential or Commercial
Zoning	Commercial Zoning Type
Property_type	Residential Property Type: House, Apartment or Lot
Houseprice	Price of property
Num_bedrooms	Number of bedrooms
Lot_size	Size of the lot
Heating	Ducted, Gas, Open Fireplace or Wood
Garage	Type of garage
Ensuite	yes or no
Balcony	yes or no
Pool	yes or no
Tennis_court	yes or no
Spa	yes or no
Aspect_facing	North, South, East, or West
School_distance	Distance to nearest school (in km)
Shop_distance	Distance to nearest shops (in km)
Train_distance	Distance to nearest train station (in km)
Bus_distance	Distance to nearest bus stop (in km)
Hospital_distance	Distance to nearest hospital (in km)
Major_road_distance	Distance to nearest major road (in km)

1. A Real-Estate Case Study

- A data warehouse for analysis purposes is required. To analyse number of properties using some variables, such as properties with pools or spa, or properties having a certain aspect facing (e.g. north facing), etc.

2. Option 1: The Non-Junk Dimension Version

- There could be many dimensions that can be derived from the abovementioned Property table.
- For simplicity, only four dimensions are chosen: *EnsuiteDim*, *PoolDim*, *SpaDim*, and *AspectFacingDim*, and only two fact measures: Number of Property, and Total Price.



2. Option 1: The Non-Junk Dimension Version

Table 8.5 EnsuiteDim table

EnsuiteID	Ensuite
1	no
2	yes

Table 8.6 PoolDim table

PoolID	Pool
1	no
2	yes

Table 8.7 AspectFacingDim table

Aspect_FacingID	Aspect_Facing
1	East
2	North
3	South
4	West

Table 8.8 SpaDim table

SpaID	Spa
1	no
2	yes

Table 8.9 PropertyFact1 table

EnsuiteID	PoolID	Aspect FacingID	SpaID	Num of property	Total price
1	1	1	1	1824	747,790,330
1	1	1	2	1911	779,736,773
1	1	2	1	832	332,080,307
1	1	2	2	863	348,569,427
1	1	3	1	3801	1,522,896,359
1	1	3	2	3697	1,502,161,380
1	1	4	1	1857	744,817,268
1	1	4	2	1903	766,424,589
1	2	1	1	384	207,206,147
1	2	1	2	375	206,016,159
1	2	2	1	176	92,994,827
1	2	2	2	190	101,747,332
1	2	3	1	640	353,440,961
1	2	3	2	764	415,248,088
1	2	4	1	358	196,980,885
1	2	4	2	331	187,747,229
2	1	1	1	1900	767,864,421
2	1	1	2	1953	780,174,960
2	1	2	1	940	385,170,902
2	1	2	2	950	370,731,351
2	1	3	1	3730	1,474,579,328
2	1	3	2	3741	1,523,864,755
2	1	4	1	1805	741,317,849
2	1	4	2	1854	745,630,764
2	2	1	1	331	185,814,783
2	2	1	2	387	216,223,145
2	2	2	1	175	95,384,840
2	2	2	2	145	82,550,189
2	2	3	1	671	367,434,044
2	2	3	2	716	395,235,770
2	2	4	1	334	182,518,099
2	2	4	2	334	189,319,617

3. Option 2: Junk Dimension Version

- **Junk dimension** is a type of dimension that consolidates all the low cardinality attributes or many small dimensions tables into a single dimension table.

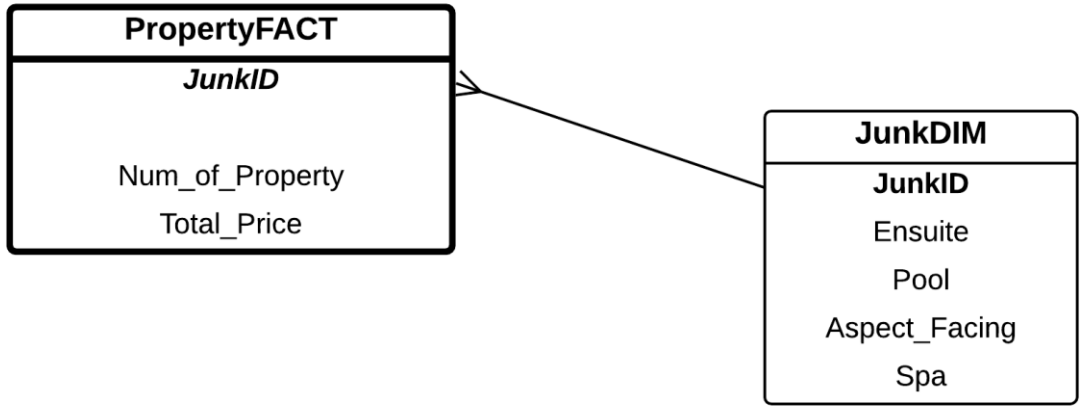


Table 8.10 JunkDim table

JunkID	Ensuite	Pool	Aspect_Facing	Spa
1	no	no	East	no
2	no	no	East	yes
3	no	no	North	no
4	no	no	North	yes
5	no	no	South	no
6	no	no	South	yes
7	no	no	West	no
8	no	no	West	yes
9	no	yes	East	no
10	no	yes	East	yes
11	no	yes	North	no
12	no	yes	North	yes
13	no	yes	South	no
14	no	yes	South	yes
15	no	yes	West	no
16	no	yes	West	yes
17	yes	no	East	no
18	yes	no	East	yes
19	yes	no	North	no
20	yes	no	North	yes
21	yes	no	South	no
22	yes	no	South	yes
23	yes	no	West	no
24	yes	no	West	yes
25	yes	yes	East	no
26	yes	yes	East	yes
27	yes	yes	North	no
28	yes	yes	North	yes
29	yes	yes	South	no
30	yes	yes	South	yes
31	yes	yes	West	no
32	yes	yes	West	yes

3. Option 2: Junk Dimension Version

Table 8.11 PropertyFact2 table

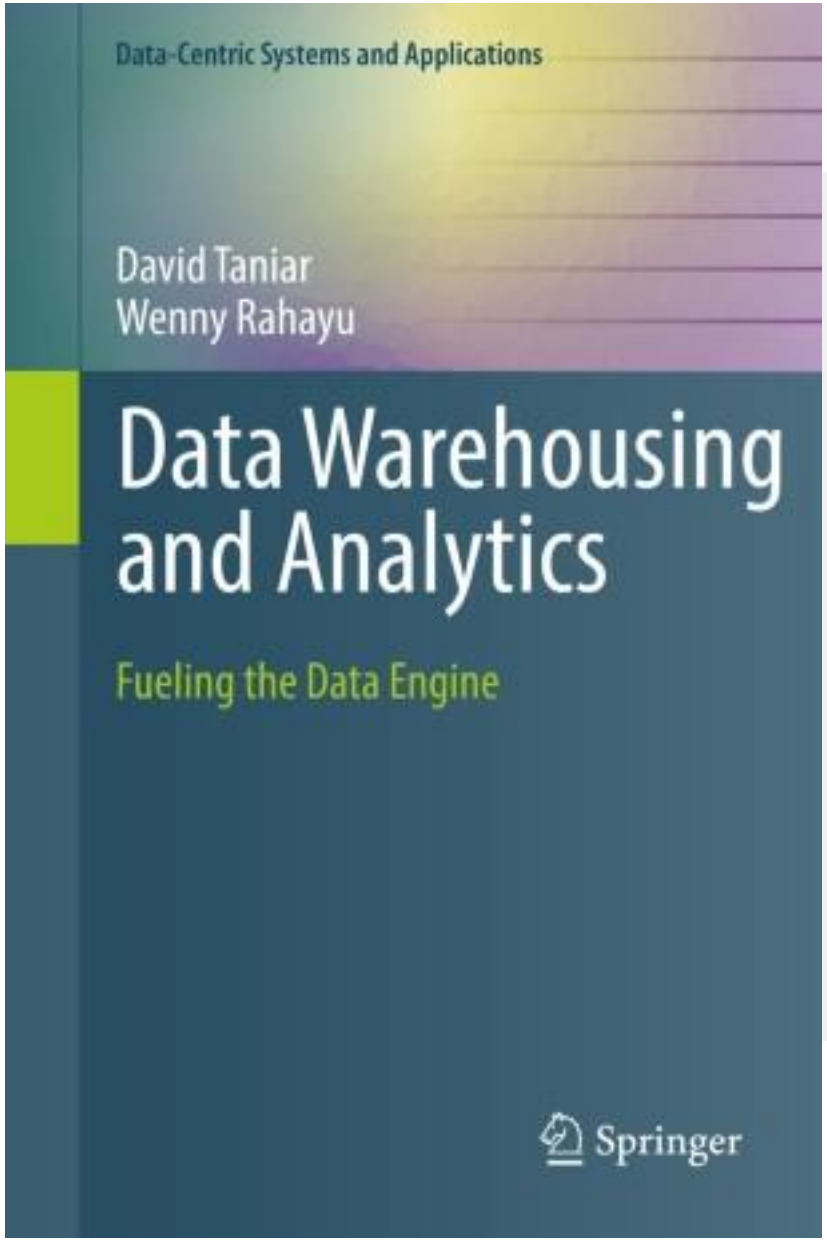
JunkID	Num_of_Property	Total_Price
1	1824	747,790,330
2	1911	779,736,773
3	832	332,080,307
4	863	348,569,427
5	3801	1,522,896,359
6	3697	1,502,161,380
7	1857	744,817,268
8	1903	766,424,589
9	384	207,206,147
10	375	206,016,159
11	176	92,994,827
12	190	101,747,332
13	640	353,440,961
14	764	415,248,088
15	358	196,980,885
16	331	187,747,229
17	1900	76,786,4421
18	1953	780,174,960
19	940	385,170,902
20	950	370,731,351
21	3730	1,474,579,328
22	3741	1,523,864,755
23	1805	741,317,849
24	1854	745,630,764
25	331	185,814,783
26	387	216,223,145
27	175	95,384,840
28	145	82,550,189
29	671	367,434,044
30	716	395,235,770
31	334	182,518,099
32	334	189,319,617

Summary

- Junk dimension is useful when there is a number of low cardinality dimension.

- The number of records in the junk dimension table is not more than the number of records in the fact table.

- There are two advantages of junk dimension:
  i. The star schema is simpler
  ii. The query is less complex



Chapter 9

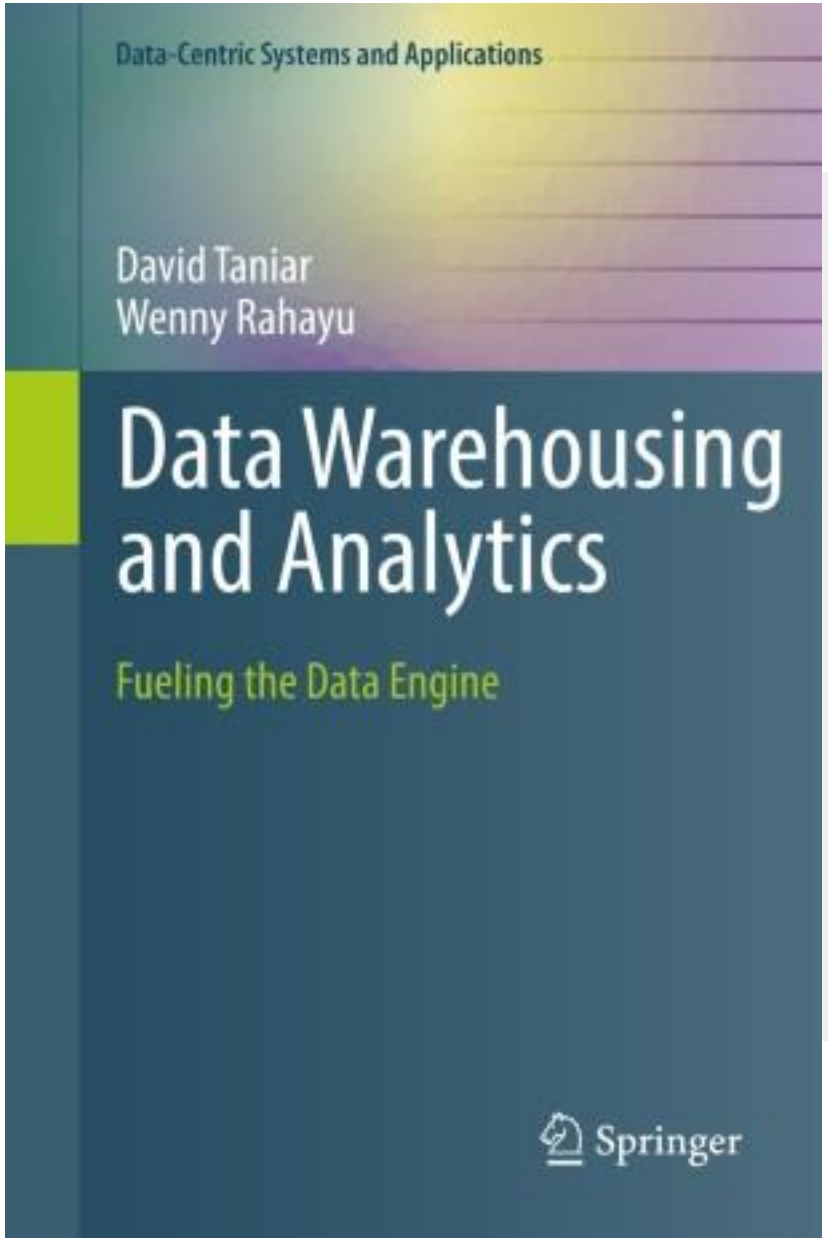
Dimension Keys

Outline

- This chapter focuses on two topics related to **Dimension Keys** which are:
    1) Surrogate Keys
    2) Dimension-less Keys

- A **surrogate key** is a key used locally in each dimension. If the primary key in the operational database is already unique across the systems in the operational databases, surrogate keys become optional in the data warehouse.

- A **dimension-less key** is an attribute in the Fact that does not refer to any dimension. This kind of dimension contains singular information which is generally widely accepted categorical codes.



Chapter 10

One Attribute Dimensions

Outline

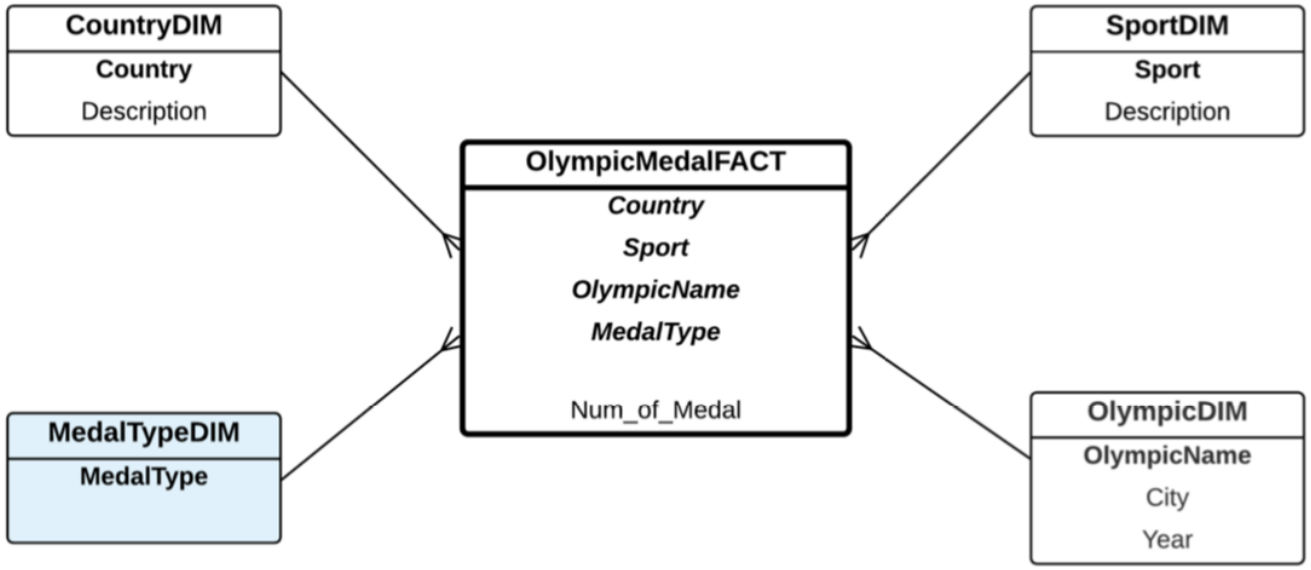
- **One Attribute Dimension** is a dimension with only one attribute.
- This chapter gives a comprehensive summary to dimensions with single attributes, and the two treatments to one-attribute dimensions, which are:
  1) Move it to the fact
  2) Keep it in the dimension

1. Move it to the Fact

1) Column-based Solution in the Fact
2) Row-based Solution in the Fact

1.1. Column-based Solution in the Fact

- In the Olympic Games Case Study, the Medal Type Dimension has only one attribute, which is a low cardinality dimension.



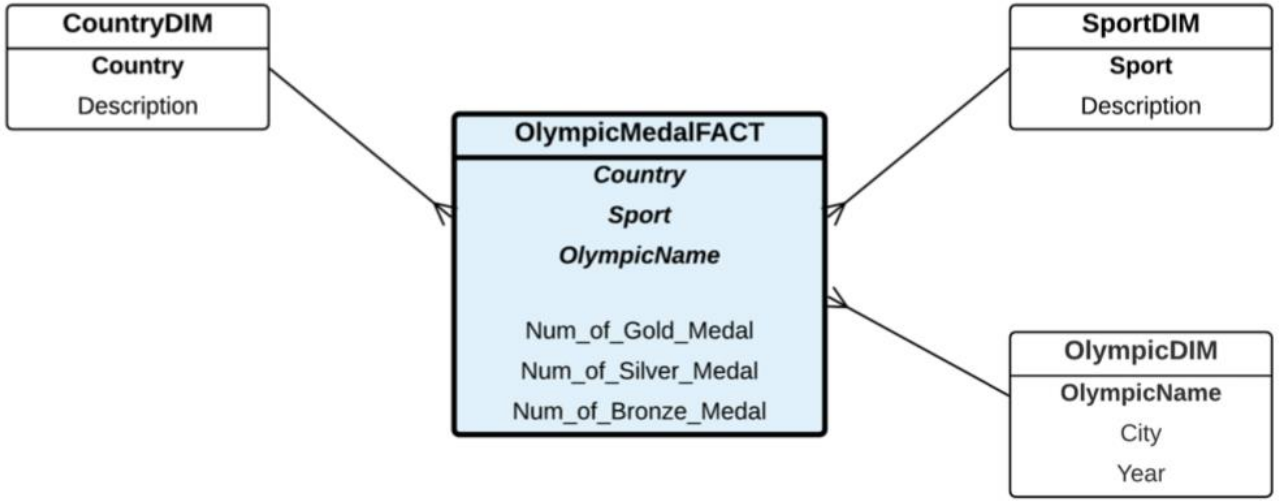
1.1. Column-based Solution in the Fact

Country	Sport	Olympic Name	Medal Type	Num of Medals
USA	Swimming	Rio 2016	Gold	16
USA	Swimming	Rio 2016	Silver	8
USA	Swimming	Rio 2016	Bronze	9
Australia	Swimming	Rio 2016	Gold	3
Australia	Swimming	Rio 2016	Silver	4
Australia	Swimming	Rio 2016	Bronze	2
China	Swimming	Rio 2016	Gold	1
China	Swimming	Rio 2016	Silver	2
China	Swimming	Rio 2016	Bronze	3
Italy	Swimming	Rio 2016	Gold	1
Italy	Swimming	Rio 2016	Bronze	2

Medal Type
Gold
Silver
Bronze

1.1. Column-based Solution in the Fact

- Hence, it is possible to expand the one fact measures to become three fact measures. This fact measure structure is called **Column-based Solution** *in the Fact*, because we are converting the dimension attribute into new columns (new fact measures) in the fact table.

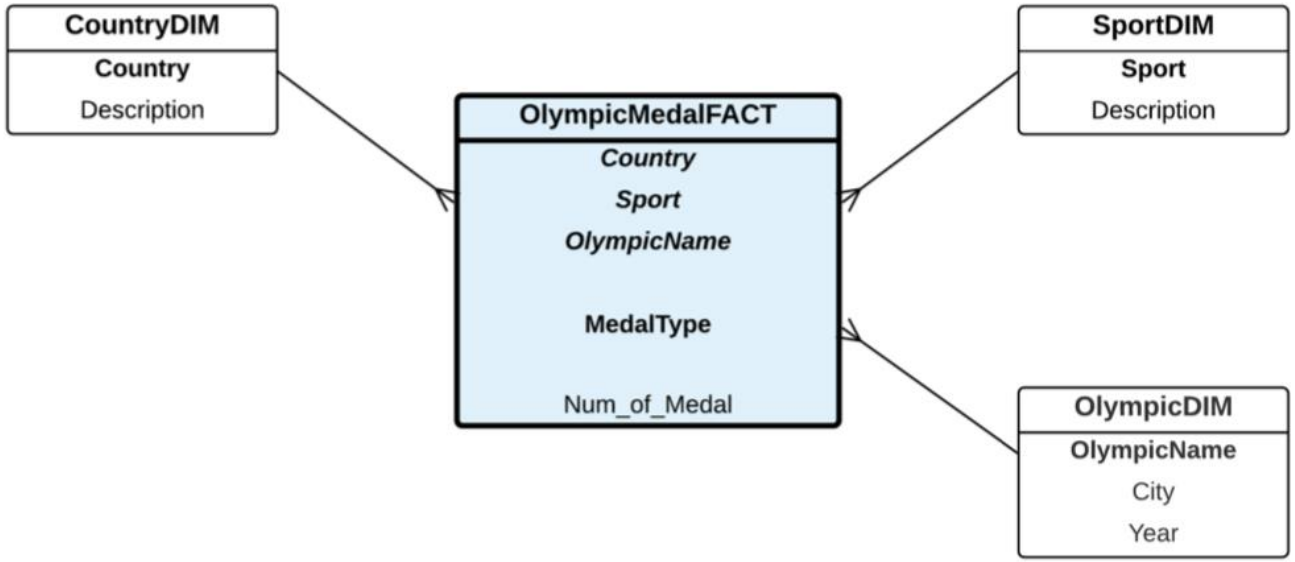


1.1. Column-based Solution in the Fact

Country	Sport	Olympic Name	Num of Gold	Num of Silver	Num of Bronze
USA	Swimming	Rio 2016	16	8	9
Australia	Swimming	Rio 2016	3	4	2
China	Swimming	Rio 2016	1	2	3
Italy	Swimming	Rio 2016	1	0	2

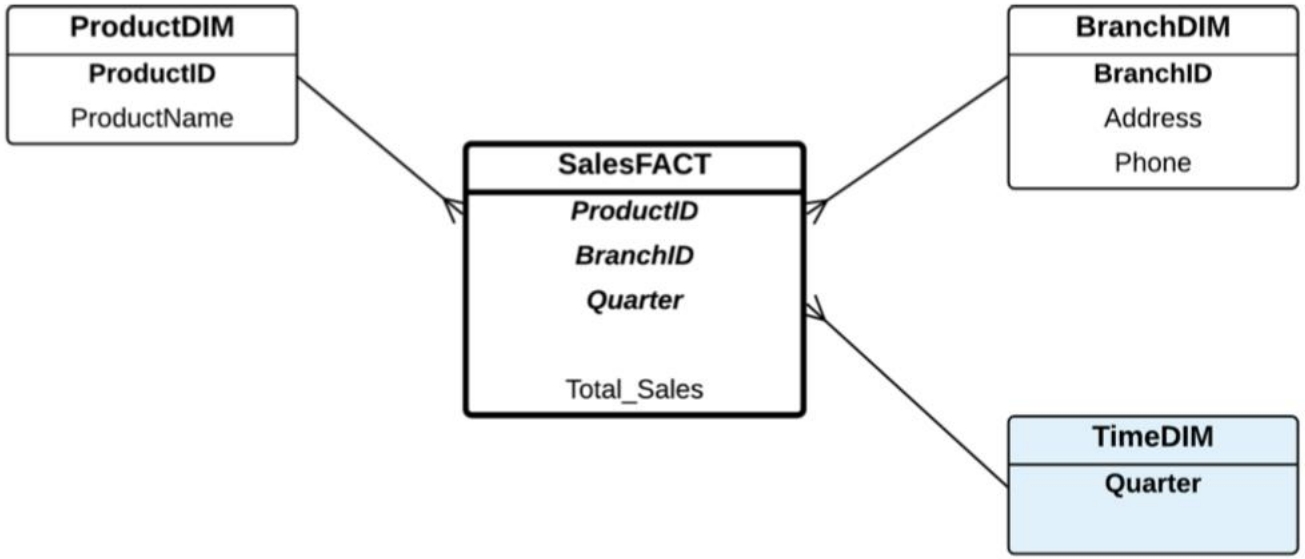
1.2. Row-based Solution in the Fact

- To remove the dimension and keep the attribute of the dimension as a dimension-less key in the fact table. This is how to present the **Row-based Solution**, which is already shown in the Dimension Key Chapter.



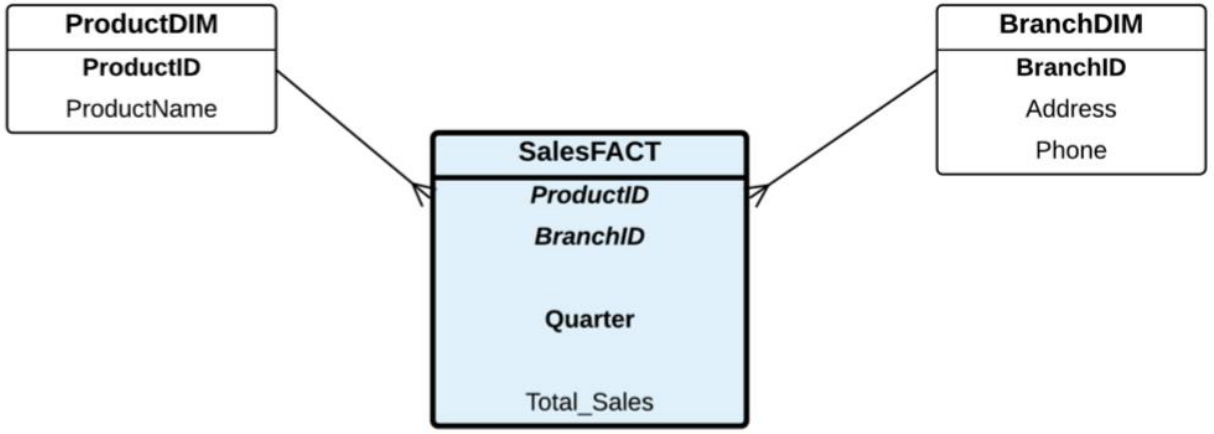
1.2. Row-based Solution in the Fact

- Take the Sales Case Study as an example:



1.2. Row-based Solution in the Fact

- The Time Dimension contains only the Quarter attribute. Assuming that there are only four quarters (e.g. Q1, Q2, ..., Q4).
- To remove the Time Dimension from the Star Schema, this is how to present a row-based solution.



2. Keep it in the Dimension

1) Combine all one-attribute dimensions
2) Combine with other normal dimensions
3) Determinant Dimension with one-attribute only
4) One-attribute dimension with bridge

2.1. Combine all one-attribute dimensions

- If there is more than one dimension with single attribute, and these dimensions are not related to each other, it is often desirable to combine them into a junk dimension.

2.1. Combine all one-attribute dimensions

- Take the Sales Case Study as an example, suppose other parameters are considered, such as delivery and payment modes. Hence, there are four dimensions: Gift Coupon, Sales Mode, Delivery, and Payment Mode and all of them have only one attribute.

Gift Coupon
Y
N

Sales Mode
In-Store
Online

Delivery Mode
Normal
Express
Pick-up in store

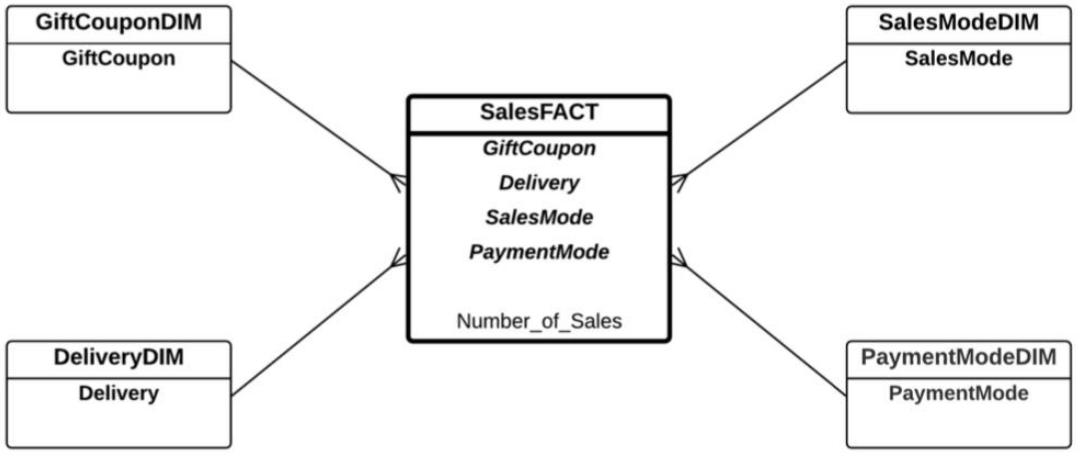
Payment Mode
Cash
Credit Card

2.1. Combine all one-attribute dimensions

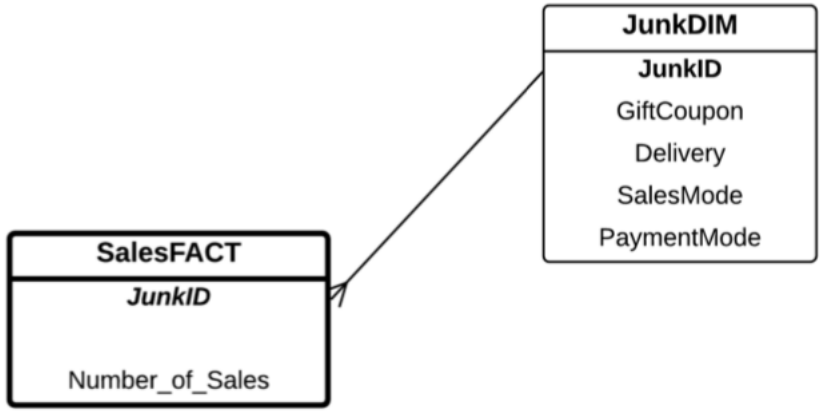
Gift Coupon	Sales Mode	Delivery	Payment Mode	Num of Sales
Y	Online	Express	Credit-Card	...
Y	In-Store	Pick-up in Store	Credit-Card	...
N	In-Store	Pick-up in Store	Cash	...
N	Online	Normal	Credit-Card	...
...	...	...	...	...

2.1. Combine all one-attribute dimensions

- Non-Junk Version



- Junk Version



2.1. Combine all one-attribute dimensions

- Junk Dimension

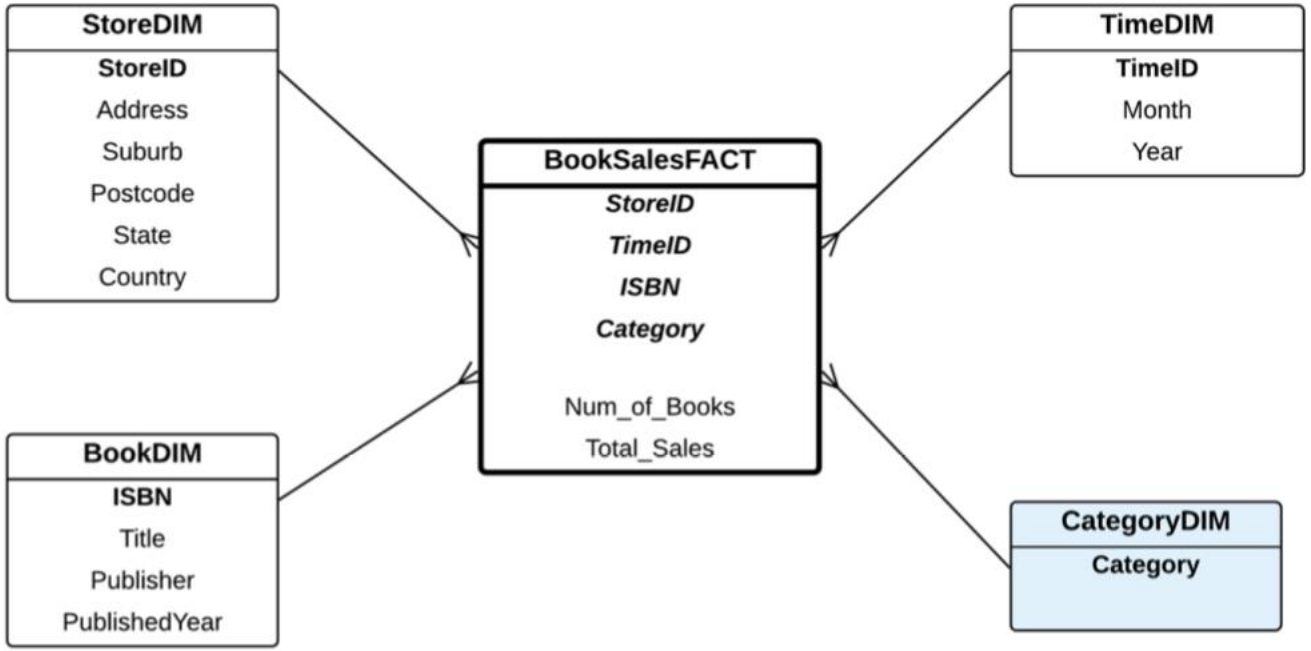
JunkID	Gift Coupon	Sales Mode	Delivery	Payment Mode
1	Y	In-Store	Normal	Cash
2	Y	In-Store	Normal	Credit Card
3	Y	In-Store	Express	Cash
4	Y	In-Store	Express	Credit Card
5	Y	In-Store	Pick-up in Store	Cash
6	Y	In-Store	Pick-up in Store	Credit Card
7	Y	Online	Normal	Cash
8	Y	Online	Normal	Credit Card
9	Y	Online	Express	Cash
10	Y	Online	Express	Credit Card
11	Y	Online	Pick-up in Store	Cash
12	Y	Online	Pick-up in Store	Credit Card
13	N	In-Store	Normal	Cash
14	N	In-Store	Normal	Credit Card
15	N	In-Store	Express	Cash
16	N	In-Store	Express	Credit Card
17	N	In-Store	Pick-up in Store	Cash
18	N	In-Store	Pick-up in Store	Credit Card
19	N	Online	Normal	Cash
20	N	Online	Normal	Credit Card
21	N	Online	Express	Cash
22	N	Online	Express	Credit Card
23	N	Online	Pick-up in Store	Cash
24	N	Online	Pick-up in Store	Credit Card

2.2. Combine with other normal dimension

- It might be possible that the one-attribute dimension has a relationship (or related) with another normal dimension, and in this case, it is possible to combine the one- attribute dimension with the normal dimension.

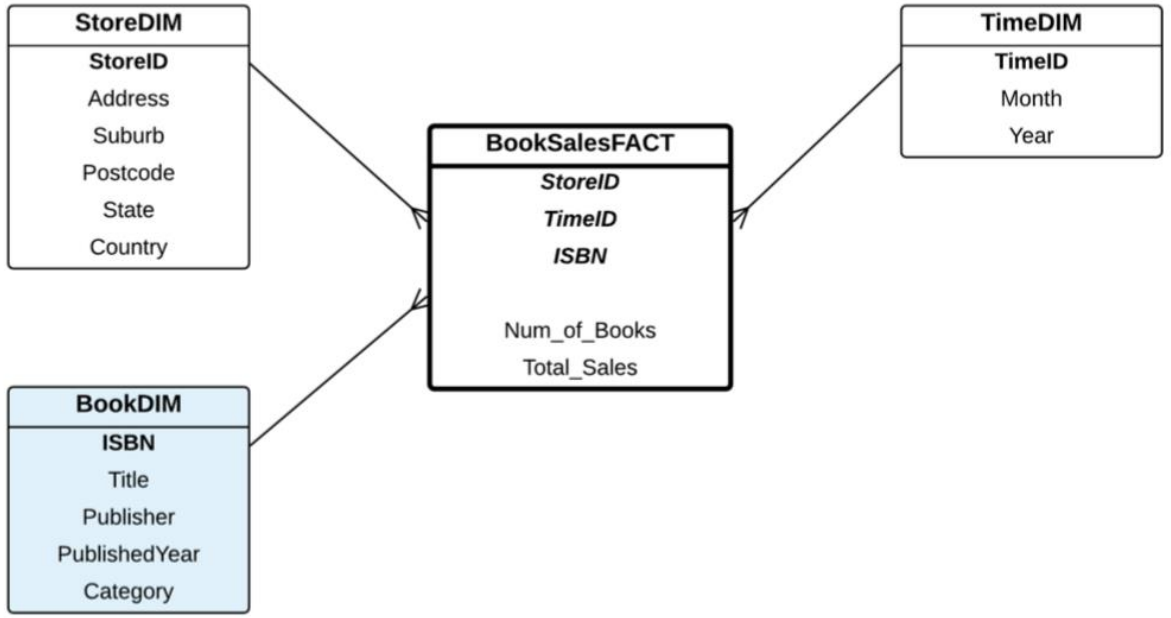
2.2. Combine with other normal dimension

- Take the Book Sales Case Study as an example, the Category dimension has only one attribute.



2.2. Combine with other normal dimension

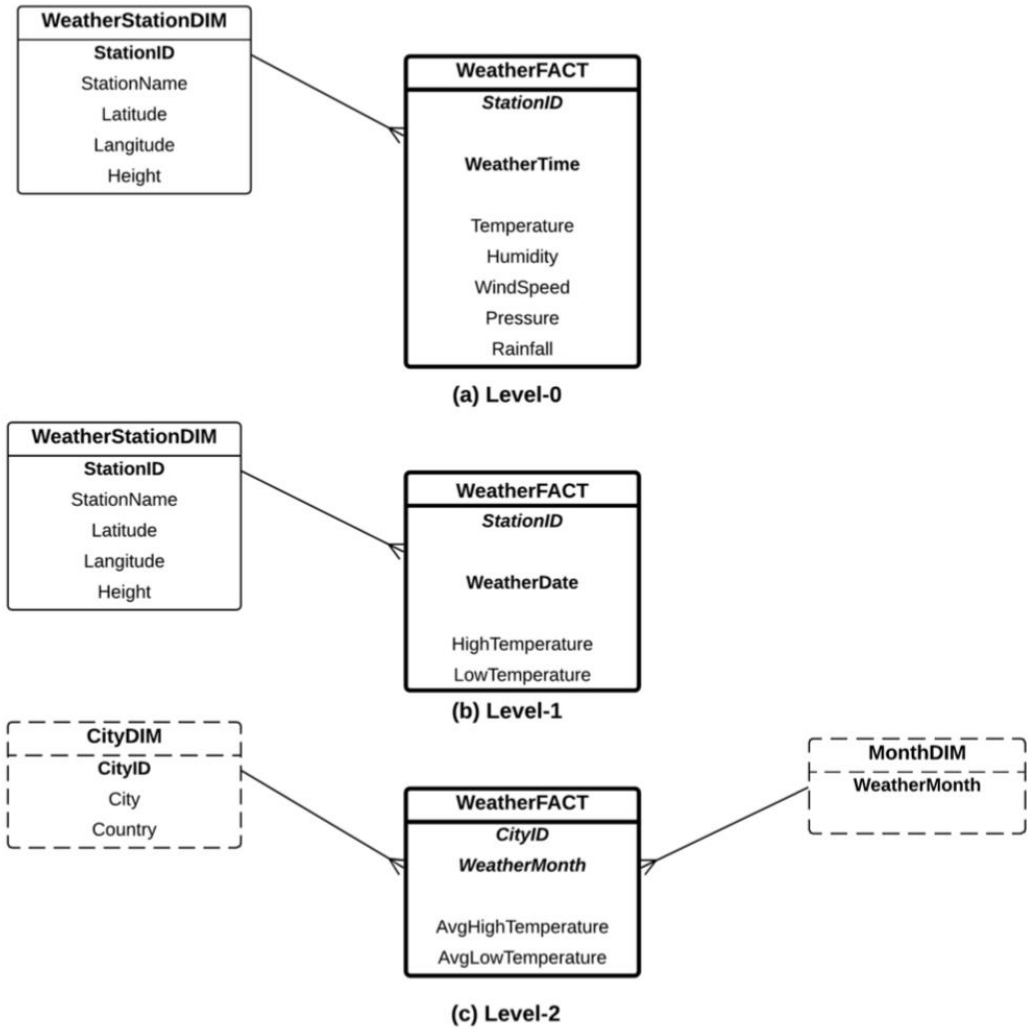
- Assuming that each book can only have one category, then Category is related to Book. Then the category can be moved into the book.



2.3. Determinant Dimension with one-attribute only

- Take the Weather Date Case Study as an example, the Category dimension has only one attribute. Weather Month Dimension must be kept in the star schema to enforce that Weather Month Dimension must be used in all queries to this star schema.

2.3. Determinant Dimension with one-attribute only



2.3. Determinant Dimension with one-attribute only

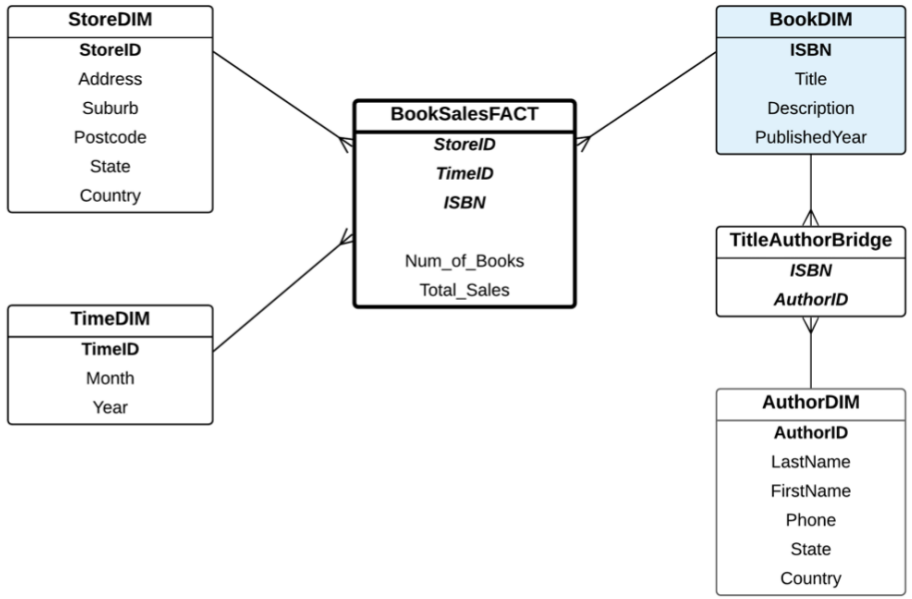
- Star schema a is level-0, containing one dimension called Weather Station.
- Star schema b is level-1, whereby the Weather Time in level-0 is now converted into Weather Date.
- Star schema c is even more general than star schema b, because the Weather Date is now changed to Weather Month. The fact measures are Average High (Max) Temperature, and Average Low (Min) Temperature.

2.4. One-attribute dimension with bridge

- When the one-attribute dimension needs to be kept intact. That is when the dimension is connected to a bridge table.
- Take the Book Sales Case Study as an example.

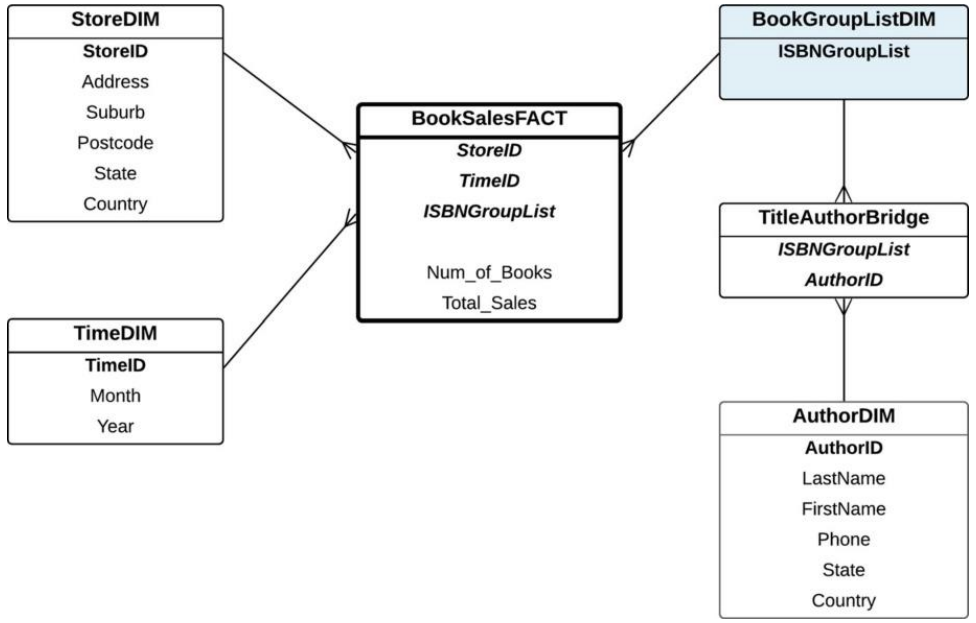
2.4. One-attribute dimension with bridge

- The Book Dimension is connected to the Author Dimension through a Bridge Table, called Title Author Bridge.
- The reason for using the bridge table is because one book may have multiple authors, and each author may have written several books.
- Therefore, analysing total sales must be based on books, not based on authors.



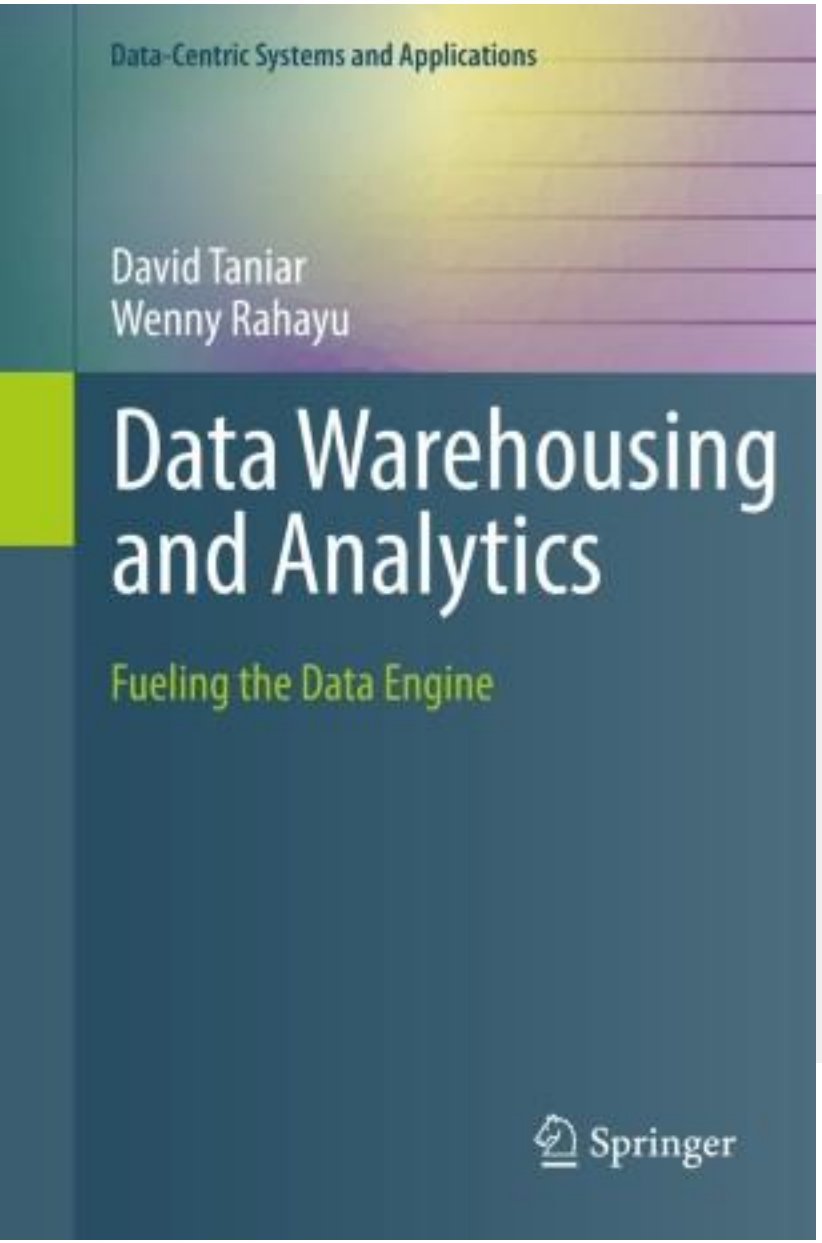
2.4. One-attribute dimension with bridge

- If the granularity is on Book Group List, whereby books authored by the same group of authors, then the dimension becomes BookGroupListDIM with only one attribute.
- But this dimension cannot be removed because it connects with another dimension: AuthorDIM through a bridge.



Summary

- One-attribute dimensions are special dimensions. In real world, it might be rare, but may exist occasionally.
- Two solutions to one-attribute dimensions:
  1) Move the dimension to the fact
  2) Combined the one-attribute dimension with other dimensions



Chapter 11

Multi-Fact Star Schemas

Outline

- A data warehouse has four basic features:

(i) Integrated,

(ii) Subject Oriented,

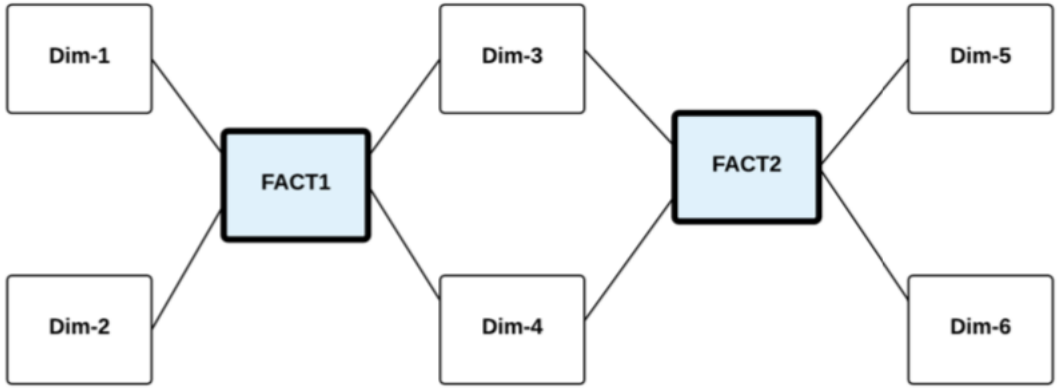
(iii) Time Variant, and

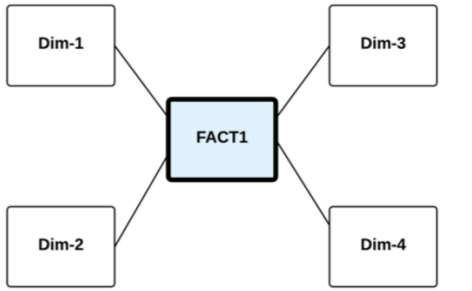
(iv) Non-Volatile.

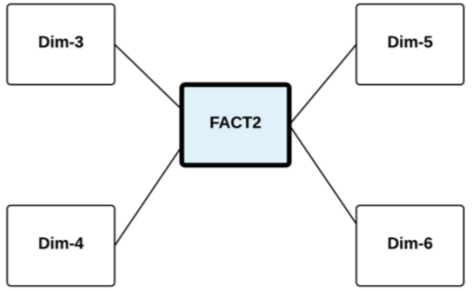
- A **Subject-Oriented** data warehouse means that one star schema focuses on one subject only.

Outline

- This chapter focuses on **Multi-Fact** Star Schemas where a star schema has multiple Fact Tables, as the figure shown:







Outline

- We are going to focus on how to create multi-fact star schemas through several case studies:
- The first case study is a Book Sales case study which shows how multi-facts are used.
- Then, the Private Taxi case study and the Tutorials and the PhD supervision case studies.
- Finally, this chapter will end with a case study where a multi-fact is caused by the different granularities of the dimensions.

Outline

- There are two main causes for a multi-fact:

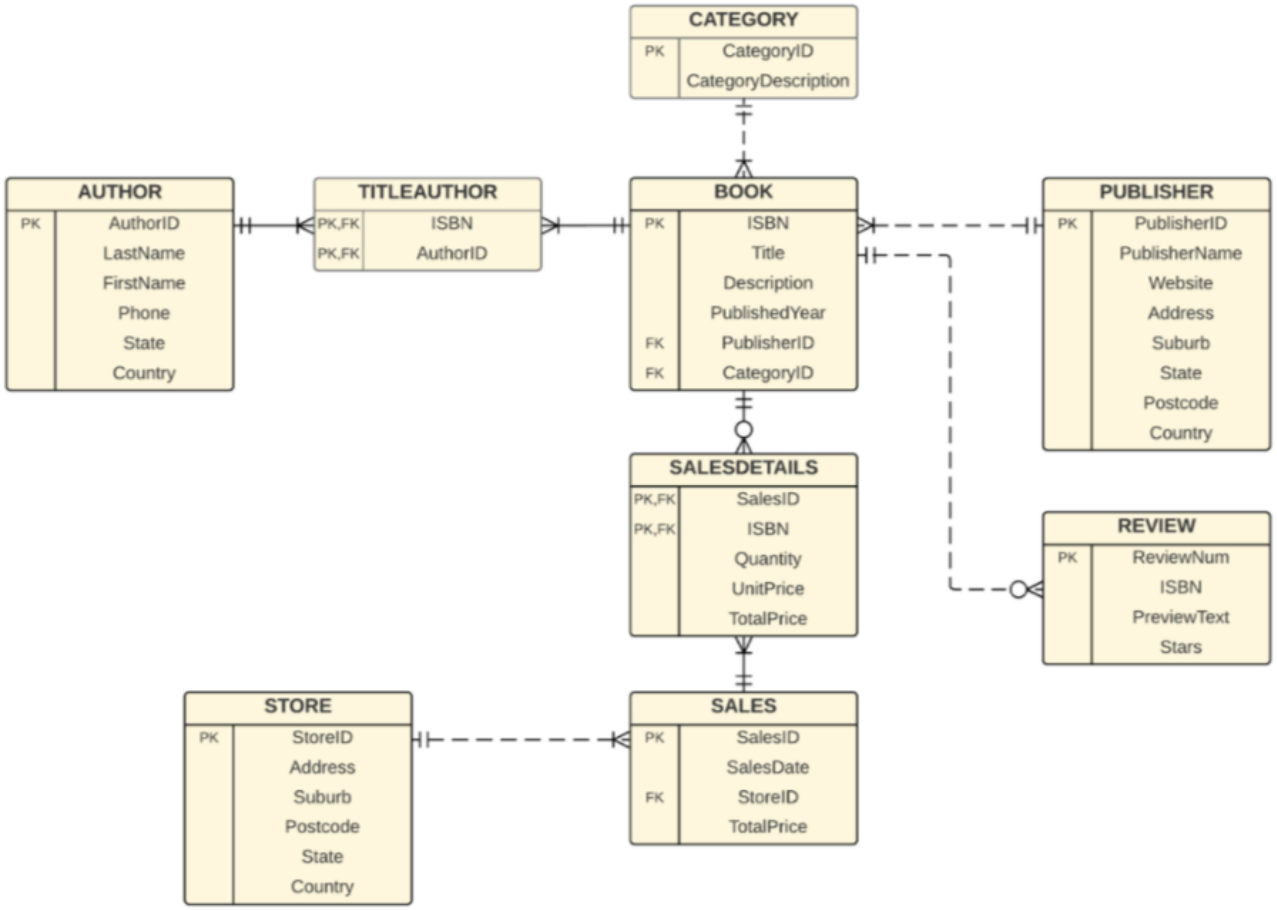
(i) different **Subject**, or

(ii) different **Granularity**

1. Different Subject Multi-Fact: The Book Sales Case Study

- The case study is about a bookshop that has several stores and they sell books.
- The system stores information about books, including the authors, publishers, book categories, and the reviews that each book has received.
- The "stars" attribute in the Review entity records the star rating for each review (e.g. 5 stars for excellent to 1 star for poor, etc).
- The E/R diagram also includes entities related to the sales of books and the stores which sell the books.

1. Different Subject Multi-Fact: The Book Sales Case Study



1. Different Subject Multi-Fact: The Book Sales Case Study

- The requirements for the data warehouse are quite simple.
  - What are the total sales for each bookstore in a month?
  - What is the number of books sold for each category?
  - Which book category has the highest total sales?
  - How many reviews are there for each category?
  - How many 5-star reviews are there for each category?

1. Different Subject Multi-Fact: The Book Sales Case Study

- Based on the above requirements, it is clear that there are three fact measures, namely:
  - Total sales,
  - Number of books sold, and
  - Number of reviews.

- We limit the dimensions to the following four:
  - Store dimension,
  - Time dimension,
  - Book category dimension, and
  - Star rating dimension.

1. Different Subject Multi-Fact: The Book Sales Case Study

- To examine the three fact measures against the four dimensions.

- For the **Store Dimension**, we would like to find the total sales for each store and the number of books sold for each store. However, it doesn't make any sense if we ask how many customer reviews (or 5-star reviews) there are for a certain store because review ratings are not applicable to the store, they are only applicable to books.

- Therefore, the store dimensions should not be connected to the fact measure: number of reviews.

1. Different Subject Multi-Fact: The Book Sales Case Study

• To examine the three fact measures against the four dimensions.

- For the **Time Dimension**, we would like to find total sales and the number of books sold each month, for example. This clearly makes sense. To find the number of customer reviews given each month seems to make sense too. However, if we inspect the E/R diagram closely, the month (or the date) is associated with the purchase of a book, not when a review is given by a customer.

- Consequently, the time dimension should not be connected to the fact measure: number of reviews.

1. Different Subject Multi-Fact: The Book Sales Case Study

• To examine the three fact measures against the four dimensions.

- For **Book Category Dimension**. Finding the total sales and the number of books sold for each book category seems to be sensible. Finding how many 5-star reviews there were for fiction novels is also reasonable because each book (or each novel) will receive reviews.

- Therefore, calculating how many reviews for each particular book category is also reasonable. In other words, the book category dimension is applicable to all of the three fact measures, total sales, number of books sold, and number of reviews.

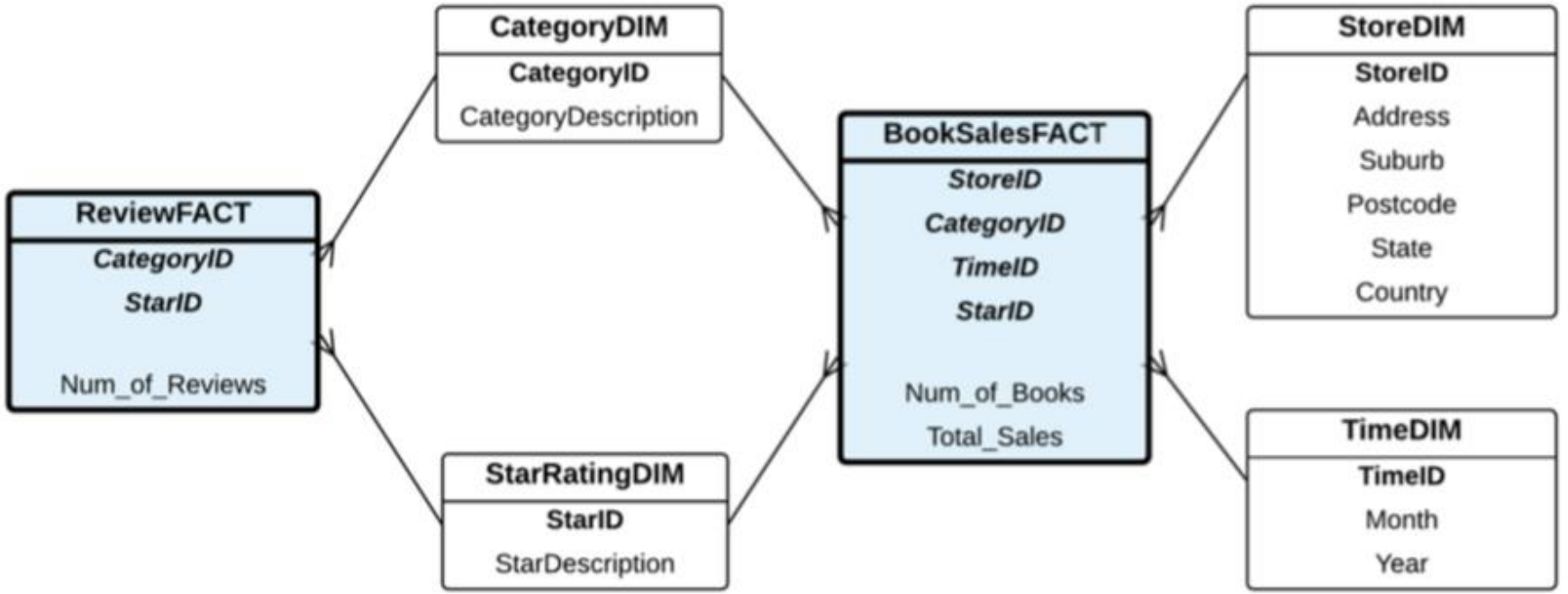
1. Different Subject Multi-Fact: The Book Sales Case Study

- To examine the three fact measures against the four dimensions.
- For **Star Rating Dimension**, describes the meaning of each star rating, that is from one star to five stars, as an example. Finding the total sales for books with 5-star ratings seems to be fine. Because one book may receive many different kinds of star ratings, we need to aggregate these and come up with a one-star rating, such as 3.6 stars. Then this will be classified to 3.0 to 3.9 stars in the Star Rating Dimension.
- Hence, the Review Star Dimension is also applicable to all of the fact measures.

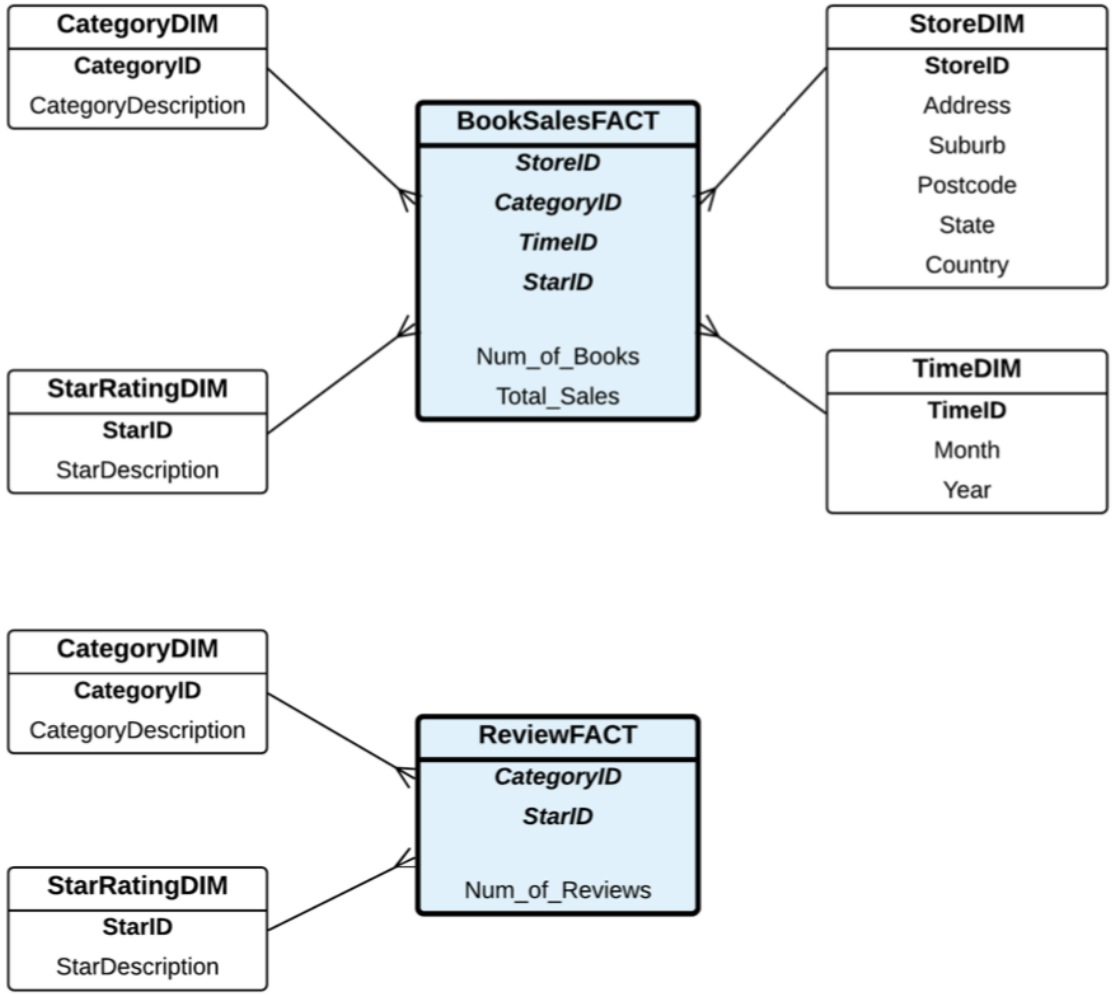
1. Different Subject Multi-Fact: The Book Sales Case Study

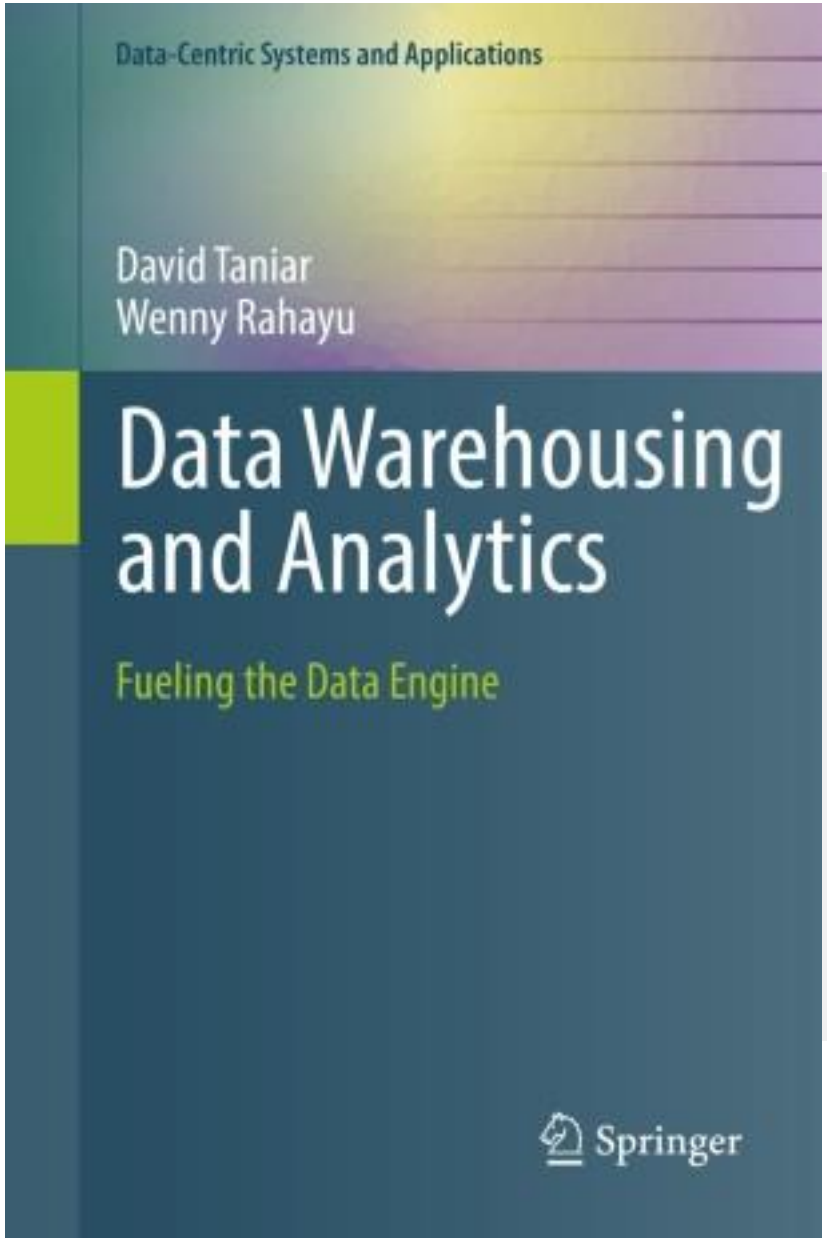
- In summary, the first subject, "Book Sales" will be the first fact, with all four dimensions (e.g. Store, Time, Category, and Star Rating dimensions). The fact measures are total sales and number of books sold.
- The second subject is "Reviews", with only two dimensions, Category and Star Rating Dimensions. There is only one fact measure, which is the number of reviews.
- The star schema will then have two fact entities, Book Sales Fact, and Review Fact.

1. Different Subject Multi-Fact: The Book Sales Case Study



1. Different Subject Multi-Fact: The Book Sales Case Study





Chapter 12

Slicing a Fact

Outline

- Slicing a fact is to slice the Fact Table into two or more Fact Tables which has the effect of creating multi-fact star schemas. So, the topic of slicing a fact is almost the opposite of a multi-fact.
- This chapter focuses on how to slice a Fact Table into multiple Fact Tables, creating a multi-fact star schema.
- There are two slicing methods to be discussed with the star schema features a Determinant Dimension:
  (i) Vertical Slice, and
  (ii) Horizontal Slice.

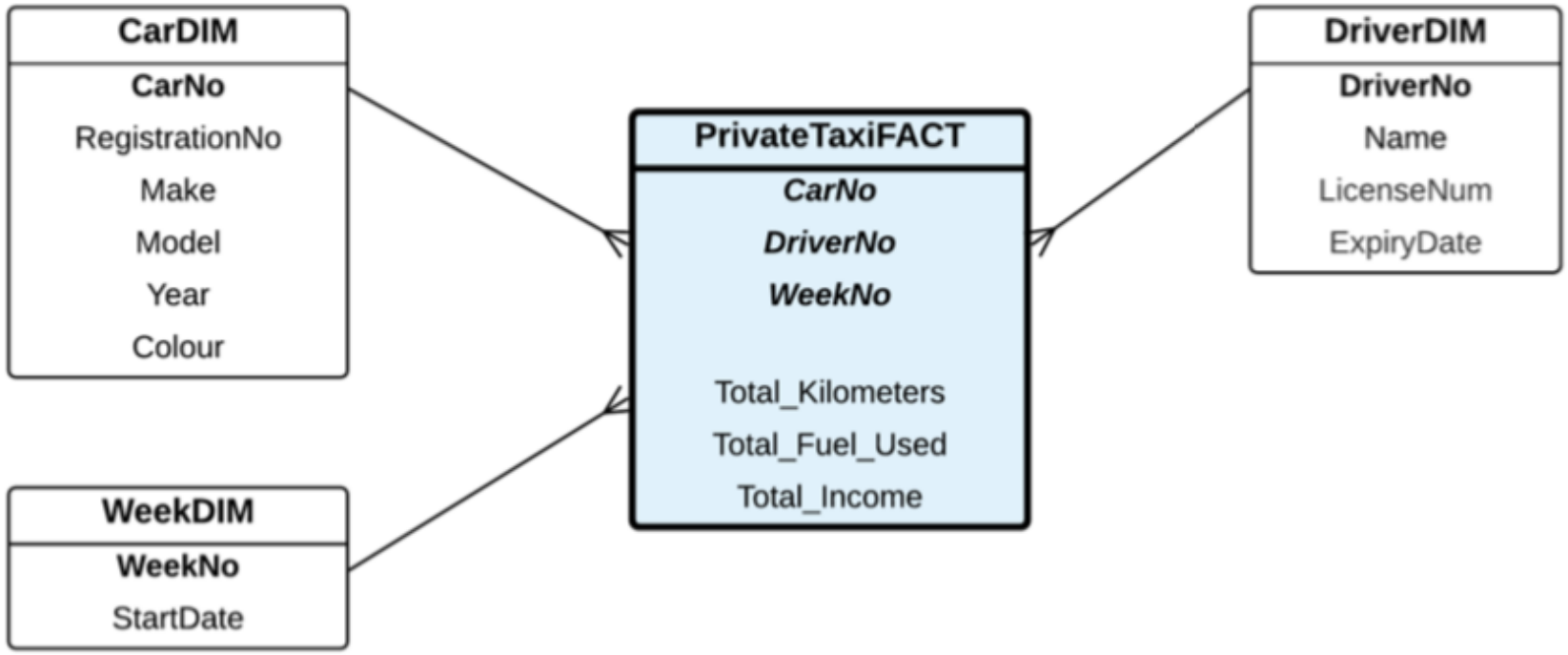
1. Vertical Slice

- A Vertical Slice is known as **Vertical Data Partitioning** in Distributed Databases, whereby a table is vertically partitioned by dividing the attributes of a table into multiple tables.
- Consequently, one Fact Table is sliced or partitioned into multiple Fact Tables, creating a multi-fact star schema.

1. Vertical Slice

- In the Private Taxi Company case study. a star schema presents with a number of taxis for private charter.
- The star schema has three fact measures, Total Kilometres, Total Fuel Used, and Total Income and three dimensions, Car Dimension, Week Dimension, and Driver Dimension.

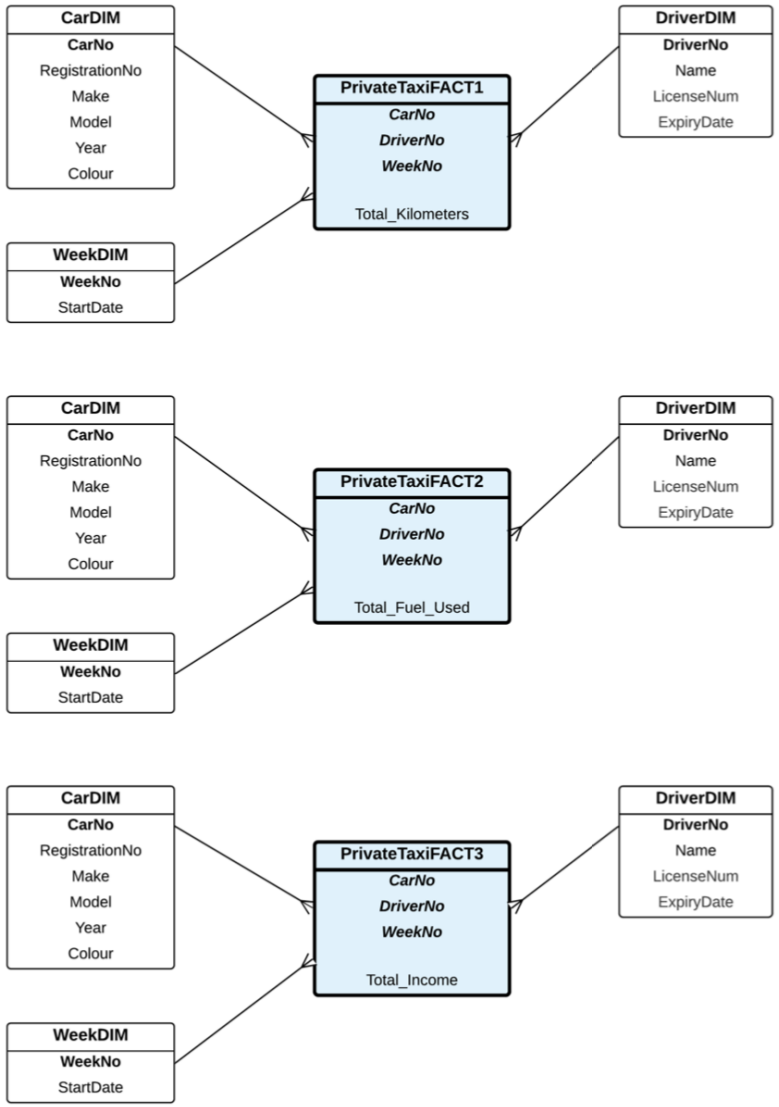
1. Vertical Slice



1. Vertical Slice

- Assuming that the fact has three different kinds of fact measures, one about distance travelled, one about fuel consumption, and the other about income.
- If we want to split the Fact Table into three Fact Tables, we need to vertically slice the Fact Table.
- In this example, the three fact measures will be separated into three Fact Tables.
- As a result, three star schemas are created, each star schema having one fact measure.

1. Vertical Slice



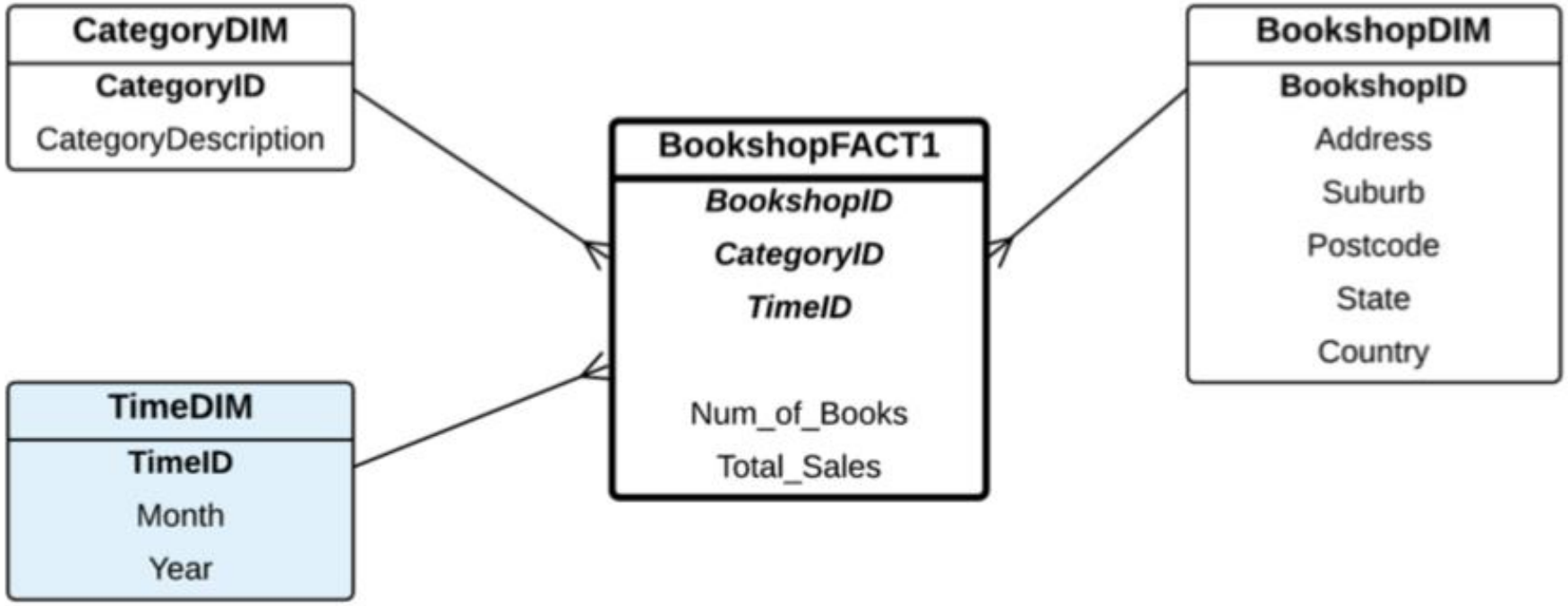
2. Horizontal Slice

- A Horizontal Slice is known as **Horizontal Data Partitioning** commonly found in Parallel Databases, where a table is horizontally partitioned by allocating different records into a different table, following a certain range of an attribute.

2. Horizontal Slice

- In the Bookshop Case Study, a star schema shown on next page.
- In this case, the Bookshop star schema has three dimensions: Category Dimension which consists of categories of books, Time Dimension which consists of month and year, and Bookshop Dimension which lists all the bookshops owned by this particular book company.

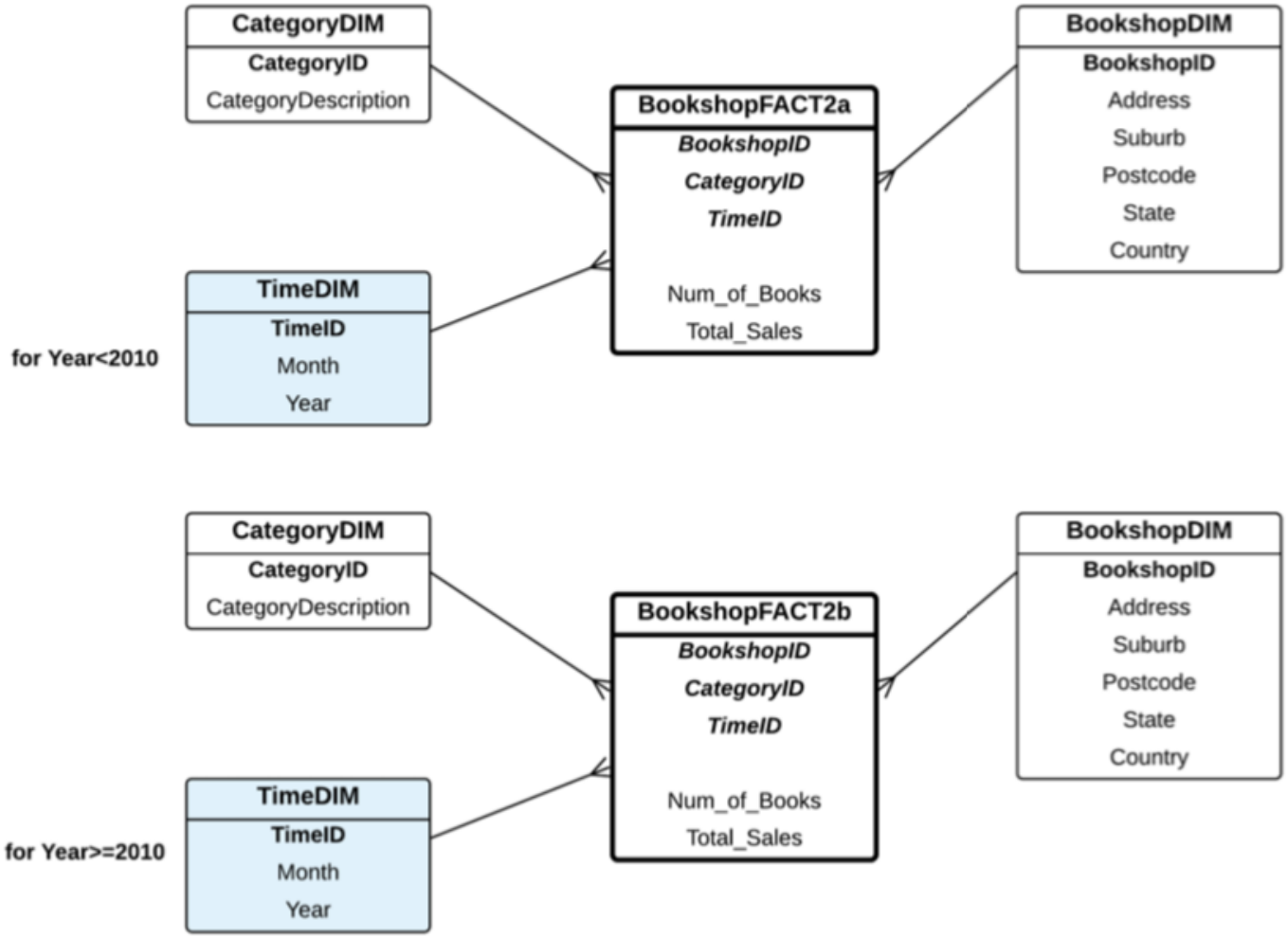
2. Horizontal Slice



2. Horizontal Slice

- A typical example of a Horizontal Slice is slicing the Fact Table based on the year.
- For example, old fact records are stored in one Fact Table, and recent fact records are stored in another Fact Table.
- As a result of this horizontal slice, the Fact Table is sliced into two Fact Tables, creating two separate star schemas.

2. Horizontal Slice



2. Horizontal Slice

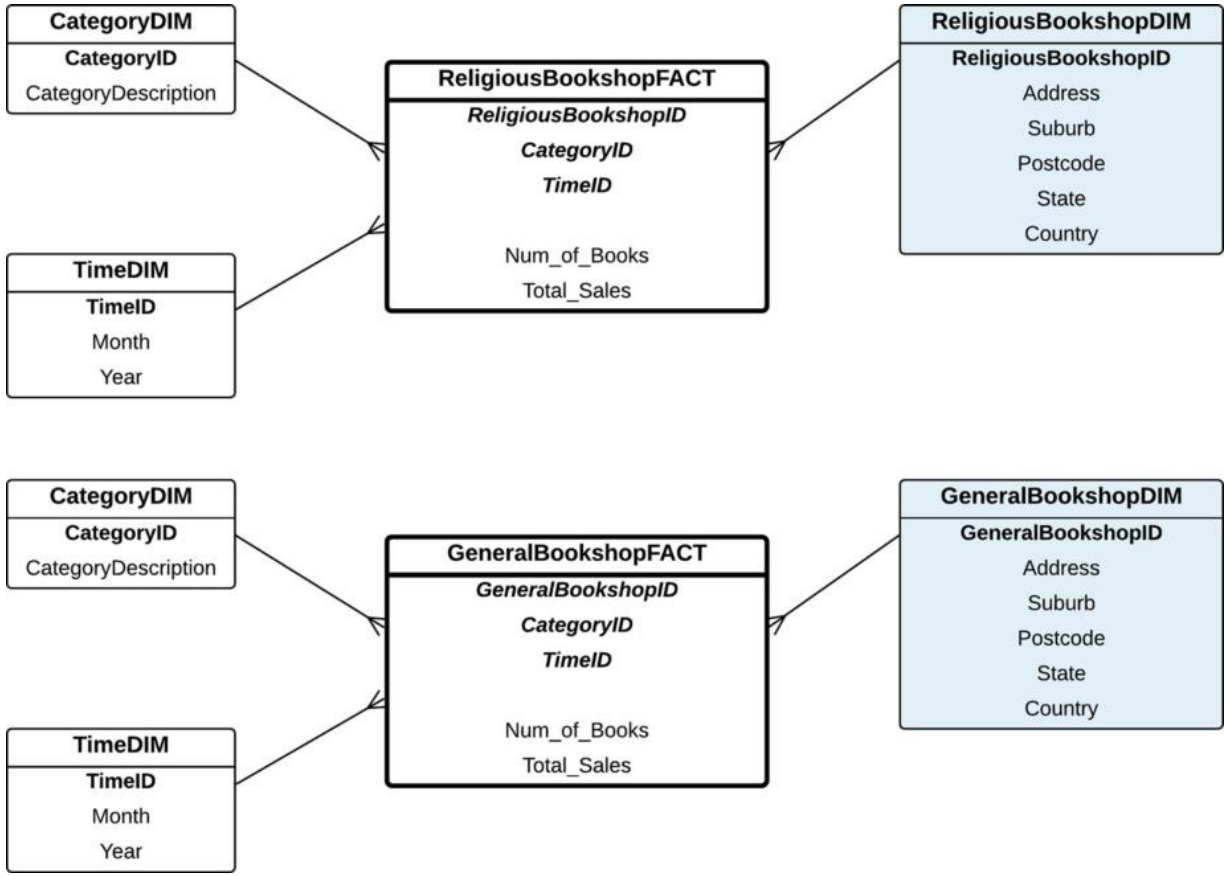
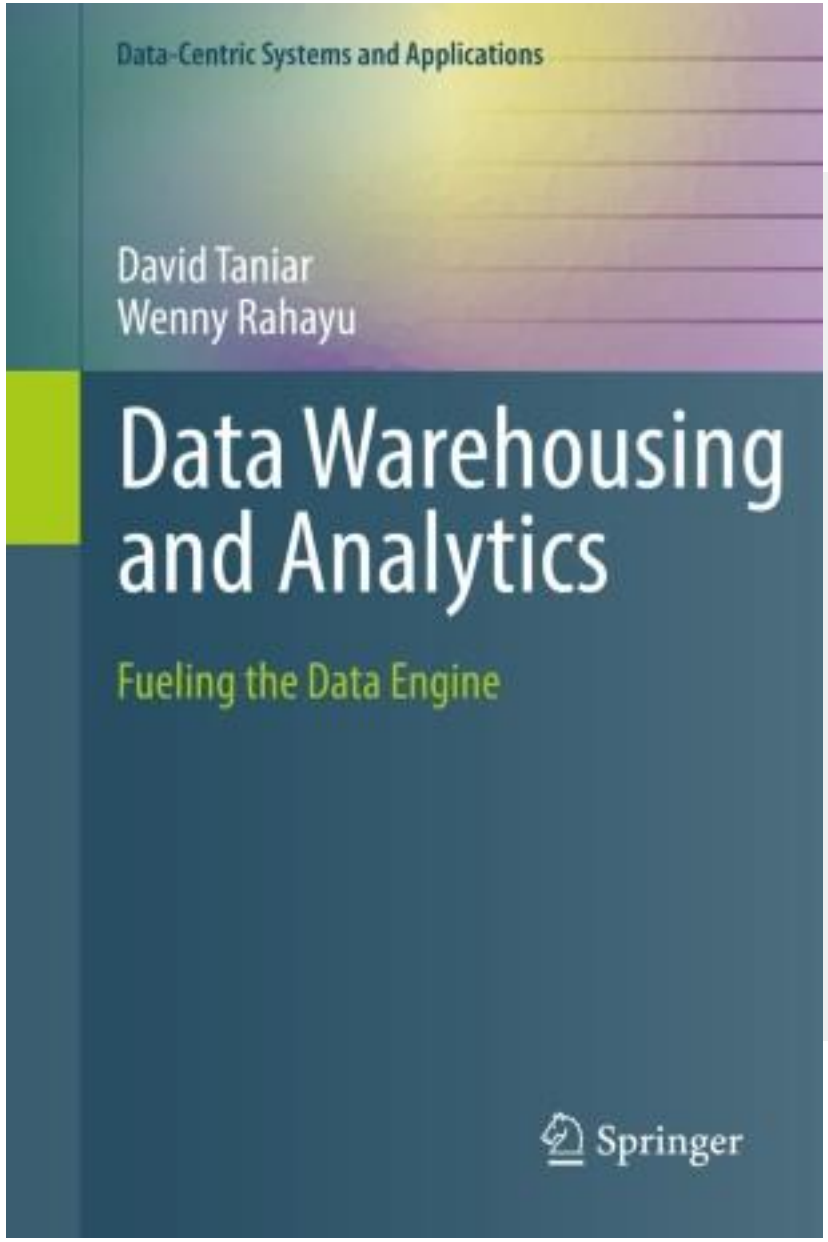


Fig. 12.5 Two bookshop star schemas sliced by bookshop type

Summary

- This chapter studies in more detail the concept of the Slicing Fact, using a number of case studies. There are two main methods:
  (i) Vertical Slice, and
  (ii) Horizontal Slice
- A Vertical Slice is vertical data partitioning as in Distributed Databases, where each fact measure attribute is placed into a new star schema.
- A Horizontal Slice is horizontal data partitioning as in Parallel Databases, where each record of the Fact Table is placed in or distributed to a new Fact Table.
- If the star schema has a **Pivoted Fact**, then it is a Vertical Slice.
- If the star schema has a **Type Dimension**, then it is a Horizontal Slice.



Chapter 13

Multi-Input Operational Databases

Outline

- The operational database is an input to the transformation process to create a data warehouse
- Multi-Input Operational Databases is when there is not one input operational database, but multiple.
- There are two methods:
  1. Vertical Stacking
  2. Horizontal Stacking

1. Vertical Stacking: University Student Clubs Case Study

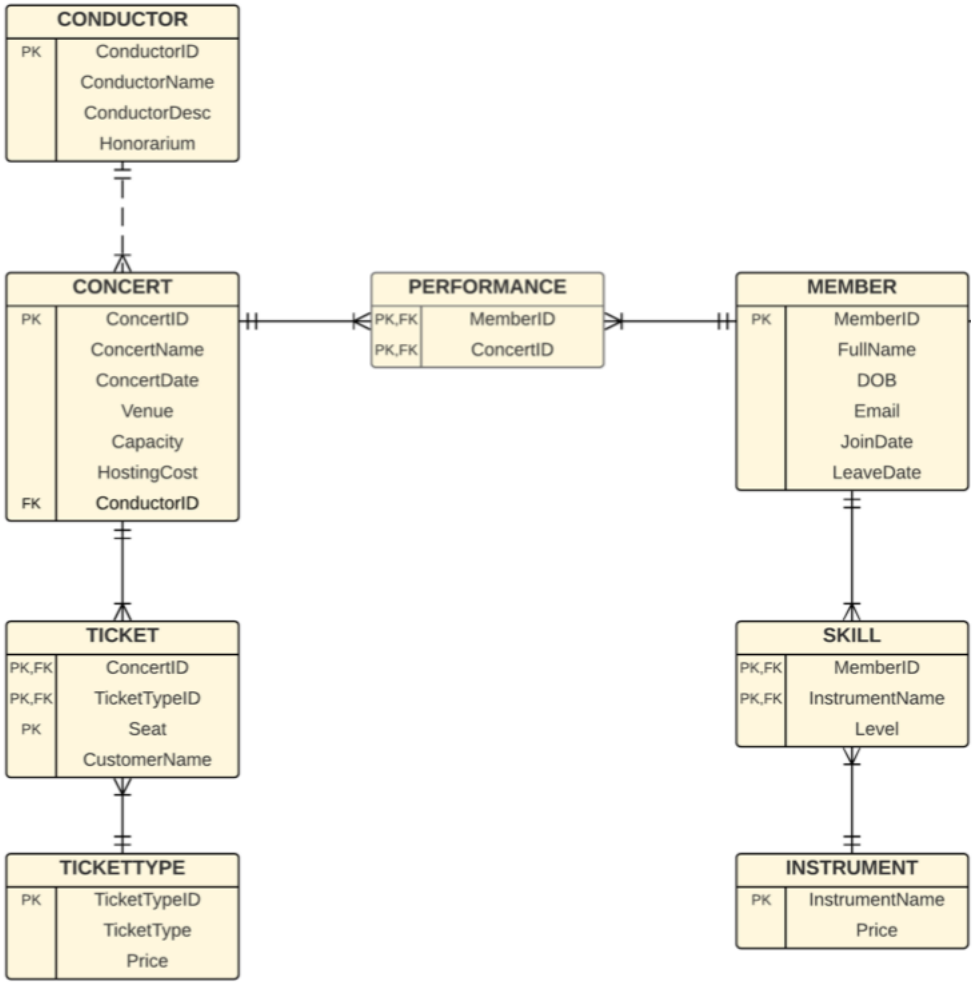
- A university typically has many student clubs. The aim of these clubs is to offer students extracurricular activities by bringing together students who share the same interests.
- All clubs receive a small amount of annual funding from the university.
- Each club maintains its own operational database, hence, to build a data warehouse, the input in this case would be **Multi-Input Operational Databases** and the output is an integrated star schema.

1. Vertical Stacking: University Student Clubs Case Study

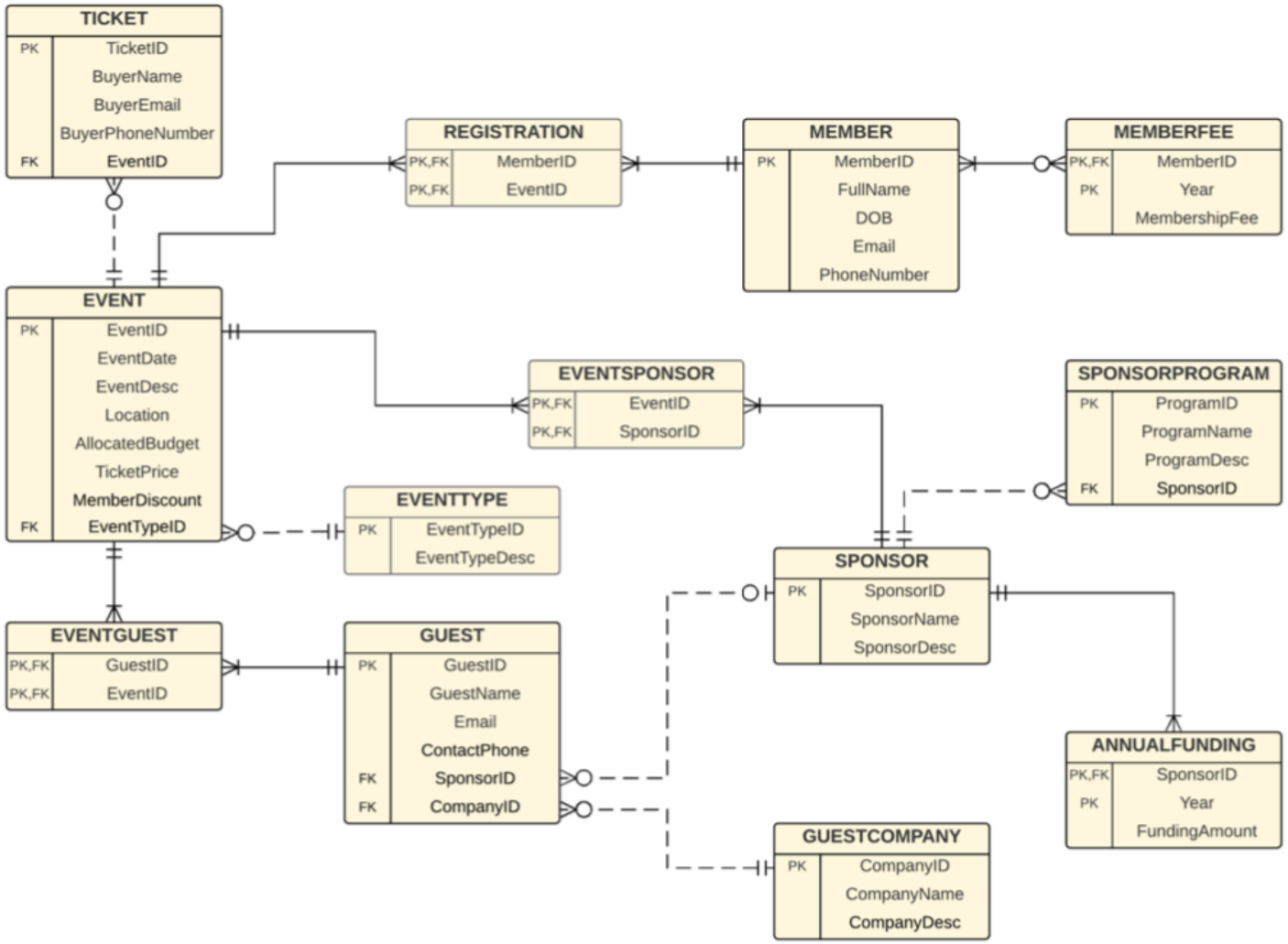
- Three clubs will be involved:

1) A Student Orchestra Club, is a kind of hobby club that students and faculty members who play classical music instruments may join.
2) A Business and Commerce Club, is a more academic type of club, focusing on academic activities to support business and commerce students.
3) A Japanese Club, is a social club where Japanese and other students can practise their Japanese language skills.

1.1. Student Orchestra Club

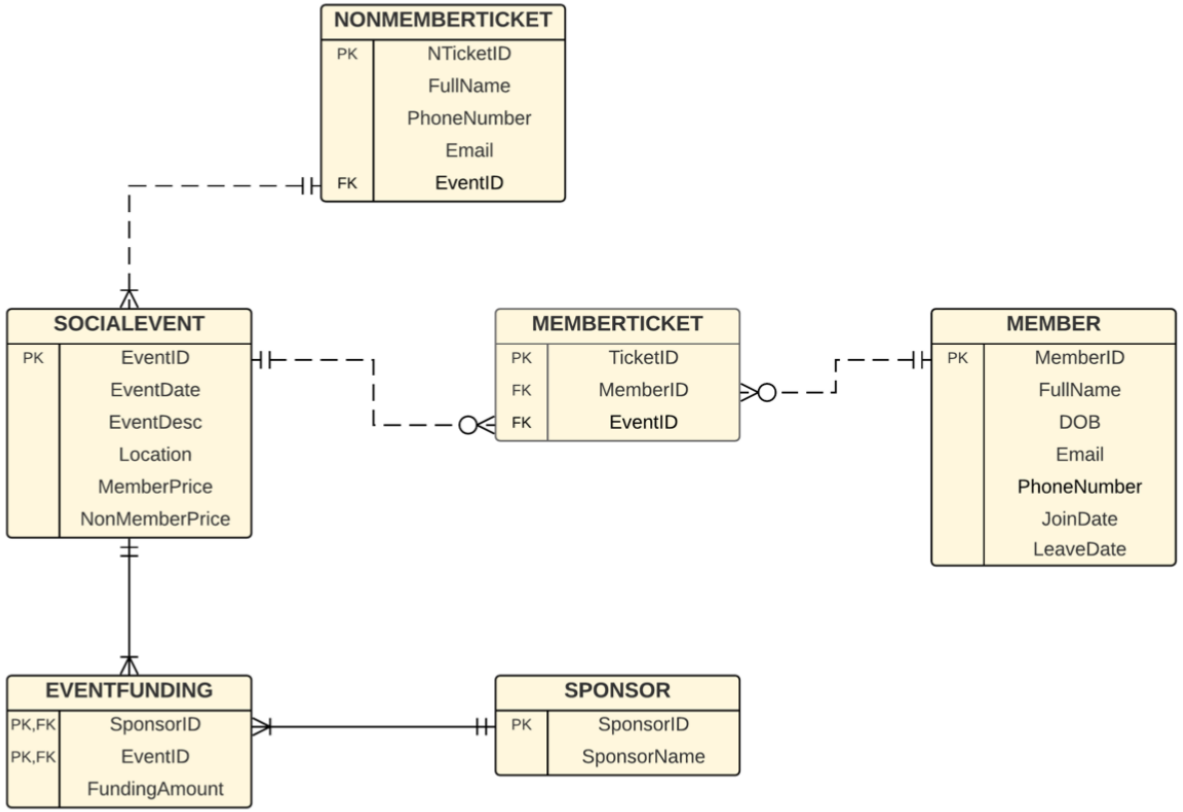


1.2. Business and Commerce Student Society



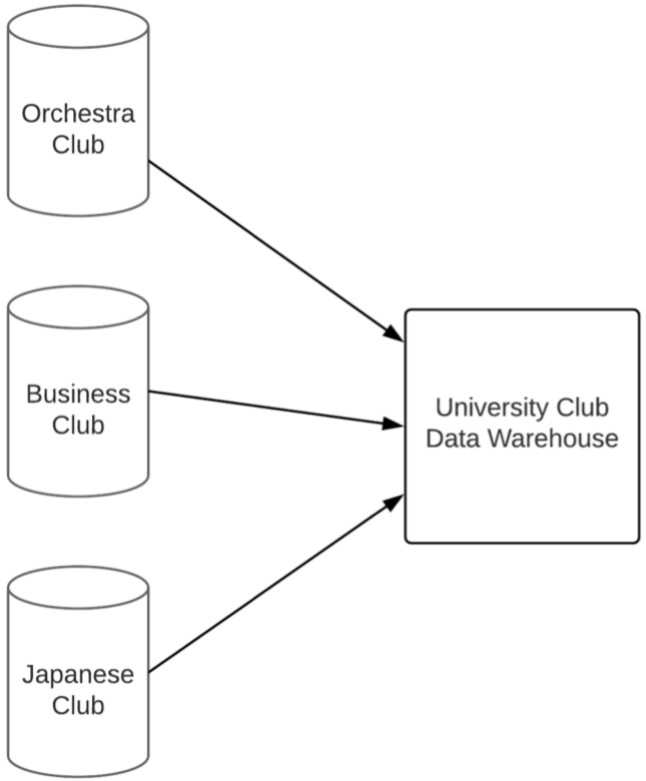
1.3. Japanese Club

- Based on the Excel file, the equivalent E/R diagram is shown as below:



1.4. Building an Integrated Data Warehouse

- To build an integrated data warehouse from three different input operational databases.

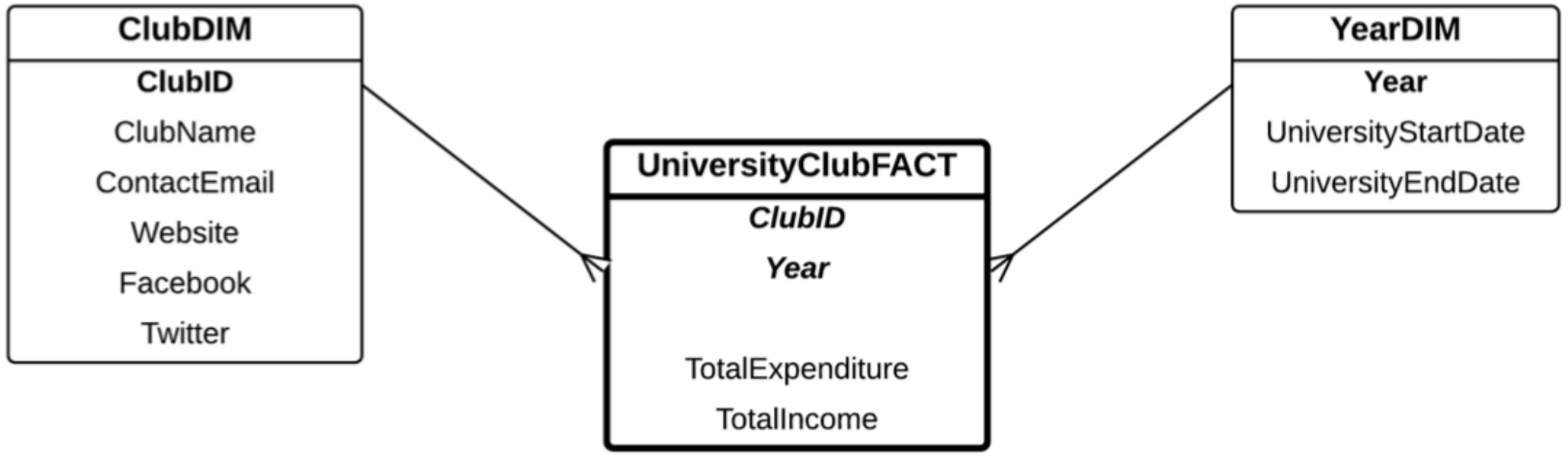


1.4. Building an Integrated Data Warehouse

- The information stored in each of these operational databases are members, activities, and financial expenditure.
- Although they are different operational databases, they have the same theme, which is activities and expenditure.
- So, it becomes possible to create one data warehouse which combines information from these different operational databases.

1.4. Building an Integrated Data Warehouse

- For simplicity, after combining the three operational databases, the star schema has only two dimensions: Club and Year Dimensions. Two fact measures: Total Expenditure and Total Income.



1.4. Building an Integrated Data Warehouse

- As a conclusion, creating an integrated star schema in this case study simply involves combining the individual star schemas.
- Combining the individual star schemas from each club is possible because the dimensions and the fact measures are identical. It uses only two dimensions, Club and Year and it has two fact measures, Total Expenditure and Total Income.
- Therefore, when combining the individual star schemas into one integrated star schema, we can simply use **union** and the **sum** aggregate function.

1.4. Building an Integrated Data Warehouse

- The method to combine the initial multiple star schema is a ***Union***.
- The union operator is like stacking multiple tables on top of each other.
- Consequently, it is like **Vertical Stacking**, where all the tables are collapsed vertically and combined into one table.

2. Horizontal Stacking: Real-Estate Property Case Study

- Another case study illustrates how an integrated data warehouse (or star schema) is created from multiple input operational databases.
- This is a case study of a real-estate company which sells properties (e.g. houses, apartments, townhouses, units, etc.).
- This company has a number of operational databases, each for different purposes.

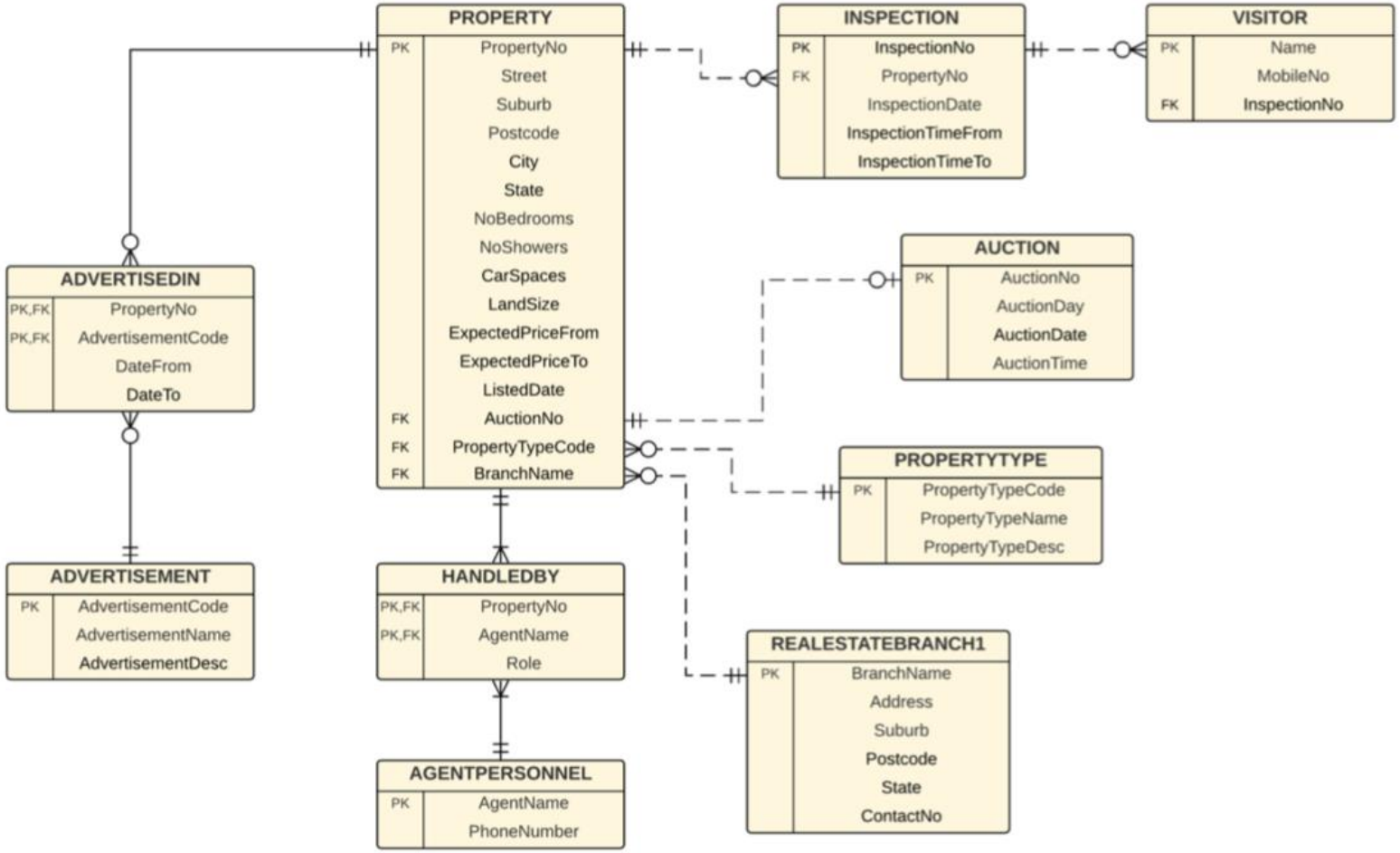
2. Horizontal Stacking: Real-Estate Property Case Study

- The **first** operational database is the **Inspection Database**.
- In this database, all properties registered for sale are recorded.
- The main table is **Property**, which records all the properties for sale.
- Each property falls into a category, which is indicated by the Property Type.
- Each property is normally open for inspection, and at every inspection, the details of the visitors are recorded.

2. Horizontal Stacking: Real-Estate Property Case Study

- The cardinality between **Property** and **Inspection** and **Inspection** and **Visitor**, is *1 − m*. If the same visitor visits the same property on different inspection dates, it will be recorded twice.
- Table **Visitor** simply logs all the visitors who attended the inspection.
- Each property is usually assigned to two staff from the branch to handle the property.

2. Horizontal Stacking: Real-Estate Property Case Study



2. Horizontal Stacking: Real-Estate Property Case Study

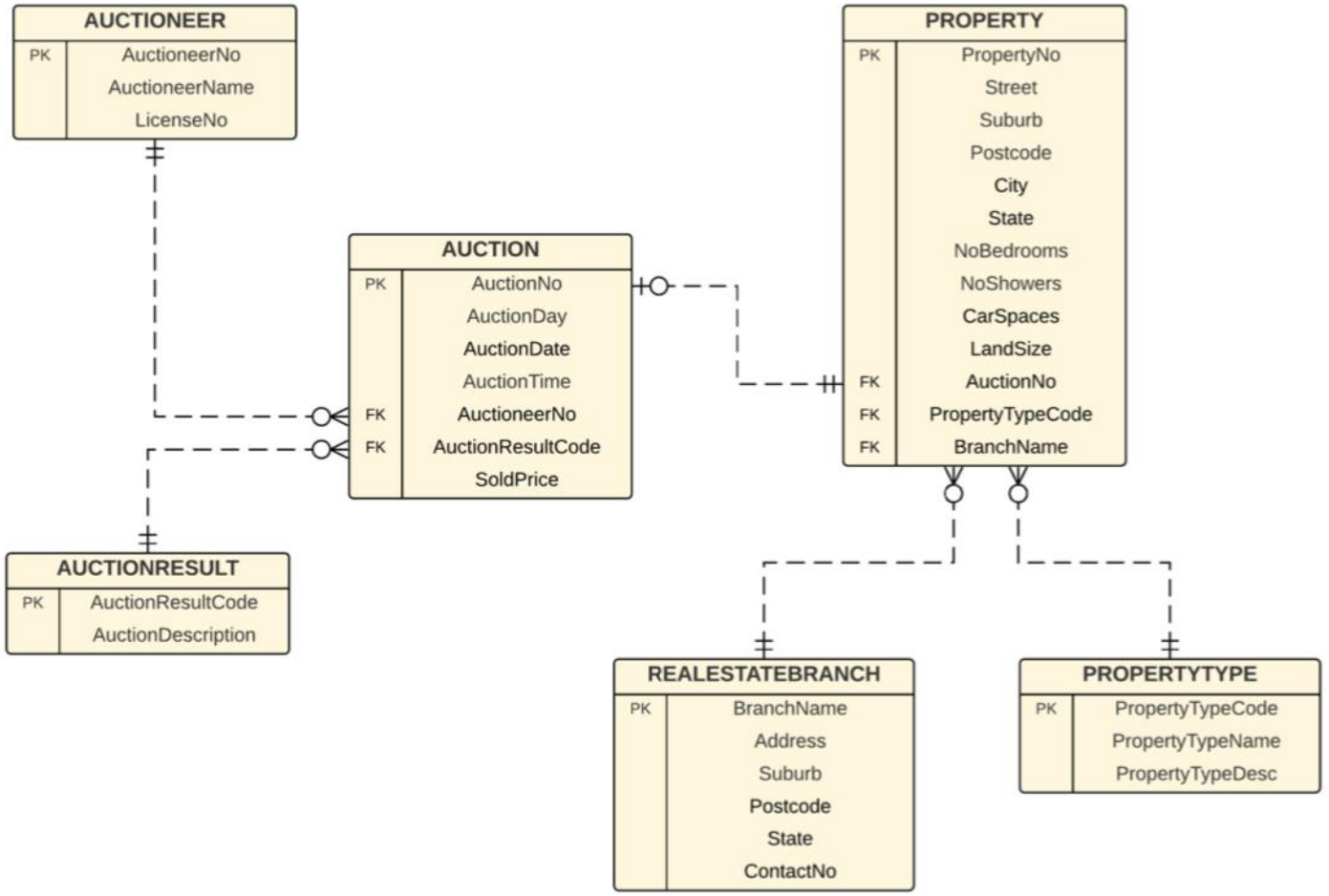
- The **second** operational database is the **Auction Database**.
- This database records the properties that have been auctioned and their auction results, for example: *sold at auction*, *sold after auction*, or *not sold*.

- "*sold after auction*" : some properties which are not sold at auction may be sold to the highest bidder, after negotiation with the owner of the property.
- If the negotiation fails, then the property's auction result is "*not sold*".

2. Horizontal Stacking: Real-Estate Property Case Study

- In this database, it has its own Table **Property**, which may not necessarily have the same attributes as the Table Property in the **Inspection Database**.
- For example, the Listed Date attribute is not included in Table **Property** in the **Auction Database**. Table **Auction** in this **Auction Database** has more information compared to Table **Auction** in the **Inspection Database**.
- The auctioneer and auction result are of course irrelevant in the **Inspection Database**.
- Only records on properties for auction are recorded in Table **Property**. Properties that are not for auction (e.g. for private sale,) are not recorded.

2. Horizontal Stacking: Real-Estate Property Case Study



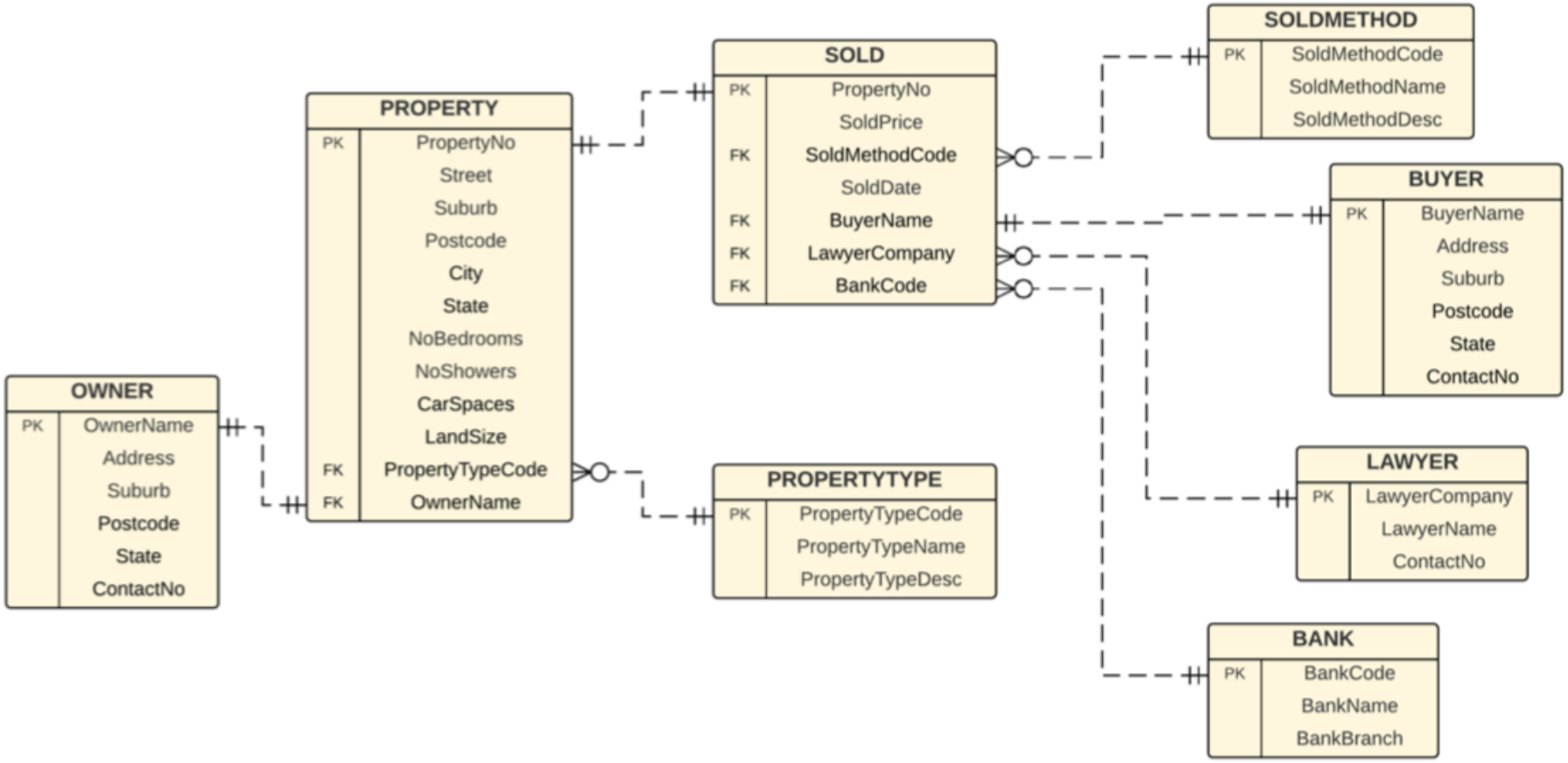
2. Horizontal Stacking: Real-Estate Property Case Study

- The **third** operational database is the **Sold Property Database**.
- This database keeps the records of all properties that are sold, either through auction or private sale.
- Table Property here is unlike the other two databases, this table does not have auction information. But all the sold properties are listed here.

2. Horizontal Stacking: Real-Estate Property Case Study

- In the **Sold Property Database**, the Table **Owner** does not exist in the other two databases. If there are joint owners, the joint owners will be listed as one record in Table **Owner**.
- The cardinality relationship between Table **Property** and **Table** Owner is *1 − 1*. If the same owner owns another property, this information will exist as different records in Table **Owner**.
- Other tables related to the selling and buying of properties, such as the Buyer, the Bank, and the Lawyer, are maintained.

2. Horizontal Stacking: Real-Estate Property Case Study



2. Horizontal Stacking: Real-Estate Property Case Study

- As a summary, the three databases in this real-estate company are all independent of each other because each of them focuses on different aspects of the properties that they manage.

- **The Inspection Database** contains information on the inspection of properties for sale;
- **The Auction Database** contains information on the auction itself;
- **The Property Sold Database** contains information on properties which have sold.

2. Horizontal Stacking: Real-Estate Property Case Study

- This real-estate company would like to build an integrated data warehouse which takes all three operational databases as input for the data warehouse.
- There are several analyses that they would like to make.

2. Horizontal Stacking: Real-Estate Property Case Study

- The first is **Auction Clearance Rate***;*
- **Auction Clearance Rate** shows the percentage of properties being auctioned that are actually sold.
- **Auction Clearance Rate** is Total Successful Auctions divided by Total Properties in Auction.

2. Horizontal Stacking: Real-Estate Property Case Study

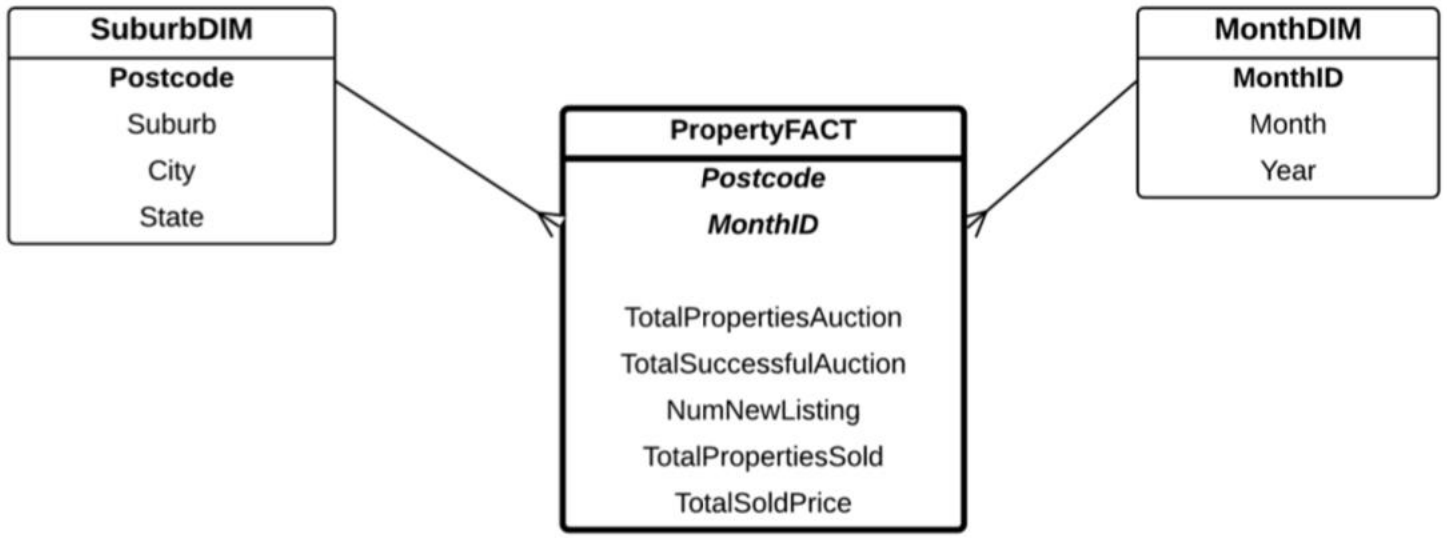
- The second indicator is **Number of New Listings**.
- **Number of New Listings** indicates how many properties are newly listed every month.
- During a property boom era, this indicator usually shows an increase, meaning that there are many people selling their houses.
- During a slow time when the property market is weak, many people hold onto their properties and do not sell, hence the **number of newly listed** properties declines.

2. Horizontal Stacking: Real-Estate Property Case Study

- The last indicator is **Price**.
- For simplicity, we only need to keep track of the mean price of the property prices. Hence, we need Total Properties Sold and Total Sold Price.

2. Horizontal Stacking: Real-Estate Property Case Study

- The integrated data warehouse is shown as below:



2. Horizontal Stacking : Real-Estate Property Case Study

- It includes two dimensions: Suburb Dimension and Month Dimension.
- The fact measures are :
  - Total Property Auction and Total Successful Auction (for Auction Clearance Rate),
  - Number of New Listings, and Number of Properties on the Market (for Property Auction Ratio), and
  - Total Properties Sold and Total Sold Price (for Property Mean Price)

2. Horizontal Stacking: Real-Estate Property Case Study

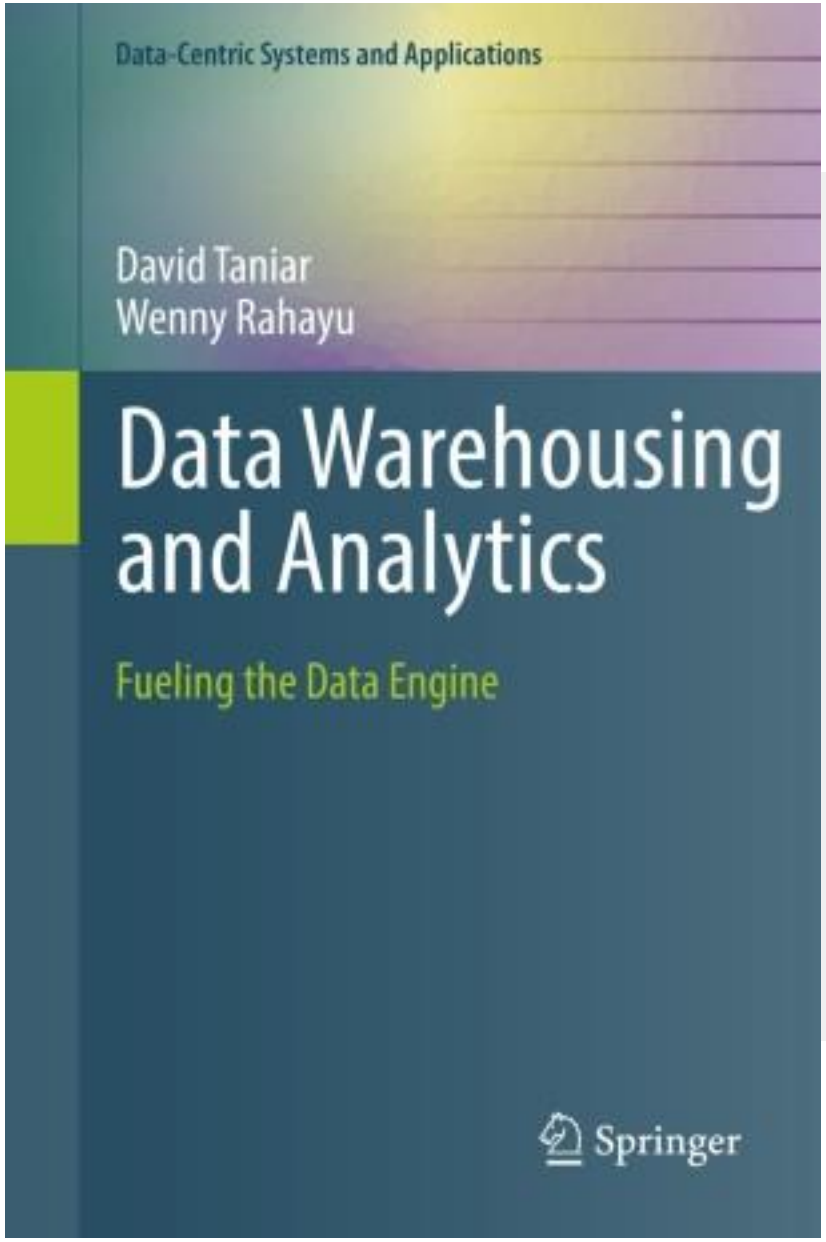
- The method to combine multiple star schemas is **Horizontal Stacking**, where each original Fact Table has its own fact measures and these fact measures are different from those of other Fact Tables.
- All the fact measures from their original Fact Table are pushed into the new Fact Table.

Summary

- This chapter discusses how a star schema is created using multiple input operational databases.
- Two case studies are studied in this chapter:
  - In the University Student Club Case Study, each operational database creates its own star schema. Then all star schemas are combined using a union command in SQL.

    *The University Student Club is a **Vertical Stacking**, where multiple Fact Tables are collapsed vertically into one Fact Table.*
  - In the Real-Estate Property Case Study, each star schema has its own fact measure. Therefore, when creating the integrated star schema, all the fact measures must be included in the Fact Table.

    *The Real-Estate Property Case Study is a **Horizontal Stacking**, where multiple tables are collapsed horizontally into one Fact Table.*



Chapter 14

Data Warehousing Granularity and Levels of Aggregation

Outline

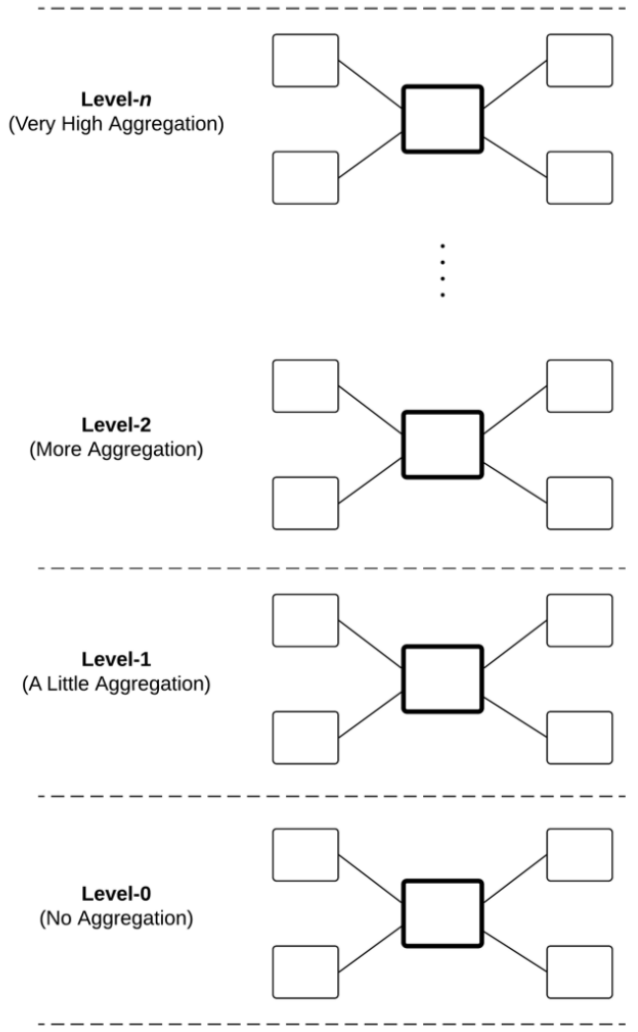
- The central concept of data warehousing is **Aggregation**.
- Digging deeper into the nature of aggregate values, aggregate values have different levels of granularity.
- **Levels of Aggregation** is about levels of granularity, or level of detail and in the context of data warehousing
- This chapter focuses Level of Aggregation, which is the foundation of Data Warehousing Granularity

1. Levels of Aggregation

- A data warehousing granularity consists of a number of granularity layers of a star schema.
- The highest granularity star schema, which contains the most detailed data is known as **Level-0**.
- **Level-1** is built on top of Level-0 star schemas, and Level-1 star schemas have a lower granularity of the fact measure.
- **Level-2** star schemas are built on top of Level-1 star schemas and have an even lower granularity of the fact measures.

1. Levels of Aggregation

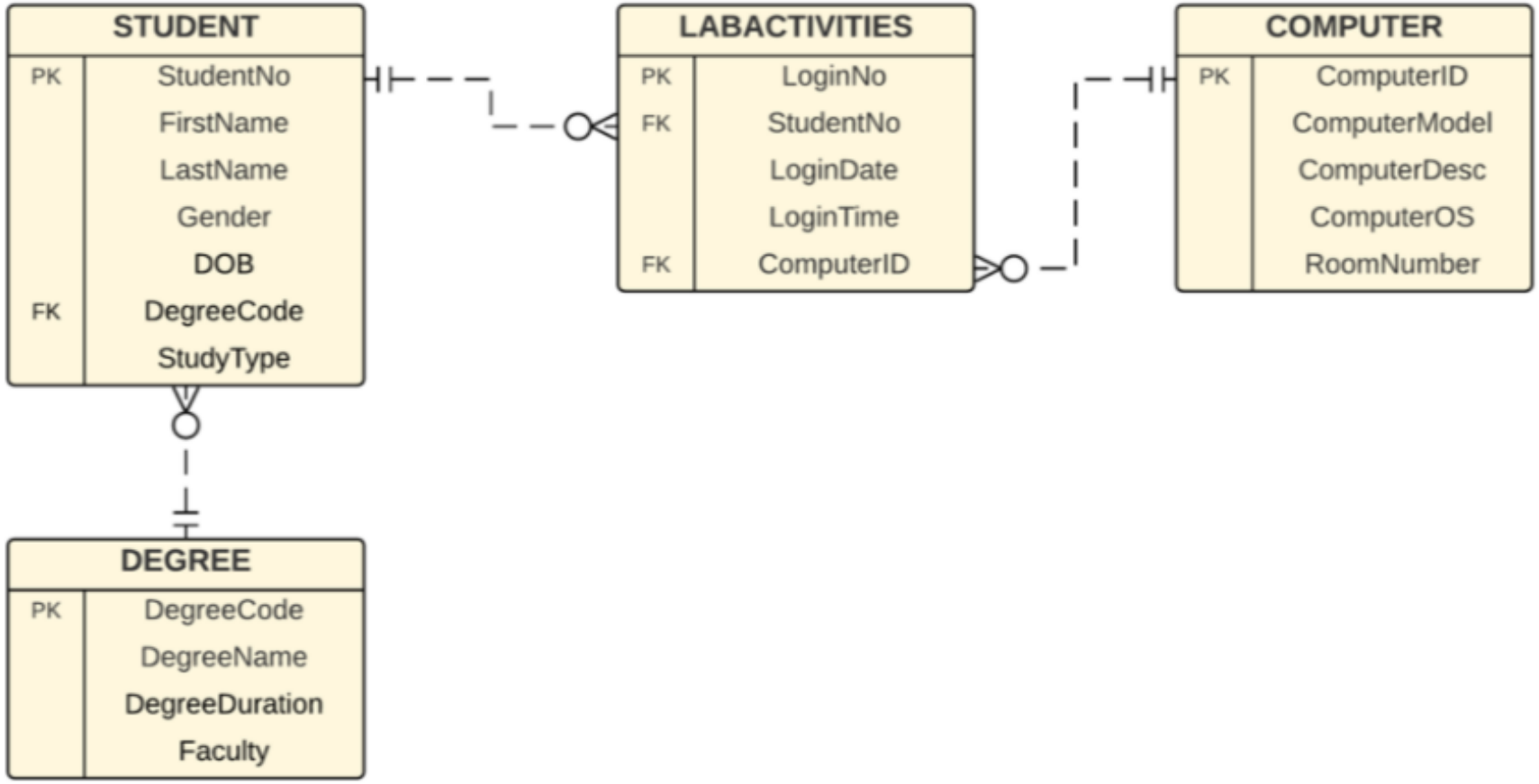
- Level-0 has the highest level of granularity where no aggregation exists. All domain tables are incorporated as dimension tables and there is no grouping in the dimension attributes.
- The lower the level of granularity, the higher the level of aggregation. Because star schemas and data warehousing are about aggregation and aggregate values, the term **Level of Aggregation** is used to refer to Level-0, Level-1, Level-2, etc.
- The higher the level of aggregation, the higher the level number.
- When given a star schema, we can only determine whether it is Level-0 or not. If it is not Level-0, then it depends on how many non-Level-0 there are below this star schema and this will determine the position of this star schema in the data warehousing architecture.



1. Levels of Aggregation

- In the Computer Lab Usage Case Study, a simple operational database is designed to keep track of the usage of computers in the labs.
- This operational database has only four tables.
  - The Student Table stores student details, including the Degree in which the student is enrolled and the Study Type.
  - The Degree Table stores a list of degrees offered by the university.
  - The Computer Table stores the details of each computer in the labs, including Computer Model, Operating Systems, etc.
  - The Lab Activities Table keeps the details of every student who logged in to each computer.

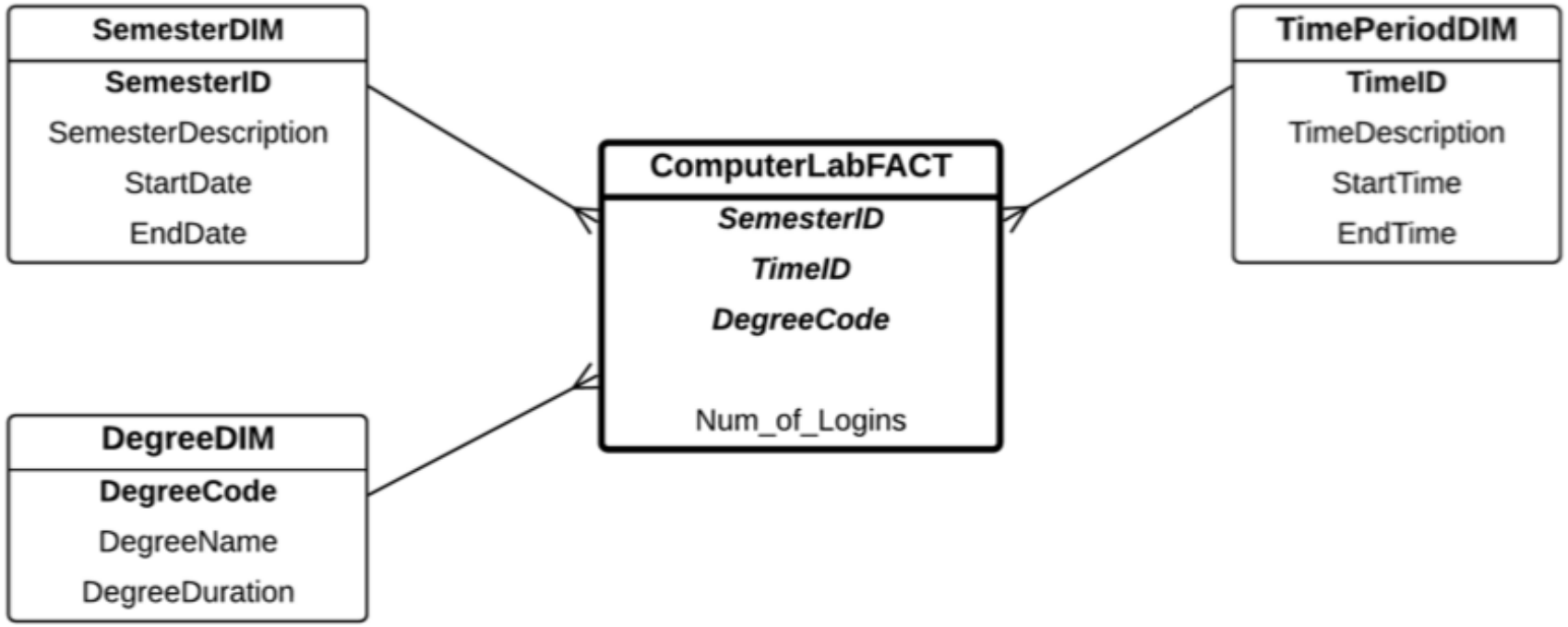
1. Levels of Aggregation



1. Levels of Aggregation

- A star schema is created to analyse the usage of computers in the labs.
- One fact measure is included in the star schema, that is Num of Logins.
- The dimensions are Semester Dimension (e.g. semester 1, semester 2), Time Period Dimension (e.g. morning, afternoon, night), and Degree Dimension.

1. Levels of Aggregation



1. Levels of Aggregation

- Data warehousing is basically precomputed aggregate values.
- In this case study, the fact measure, Num of Logins, is basically a precomputed value by counting the number of records in the transaction table, namely the Lab Activities Table, grouped by Semester, Time Period, and Degree.

1. Levels of Aggregation

- A data warehouse is built through a transformation process from the operational database by extracting, cleaning and aggregating the original data in the operational databases.
- This is why decision making does not normally query operational databases because decision-making queries usually focus on aggregate values and it will be more convenient if the required information is readily available in the data warehouse because the data warehouse is already cleaned, transformed, aggregated, etc.

1. Levels of Aggregation

- When management finds interesting data in the Fact Table, they often want to drill down to find more detailed information.
- For example, in the Computer Lab Usage Case Study, if the number of logins at night in semester 1 is rather high, we might want to see the breakdown, such as the number of logins per hour at night, etc.
- To address this issue, the concept of **Data Warehousing Granularity** is introduced.

1. Levels of Aggregation

- To be able to drill down to a lower level of aggregation, we need to query the lower level of aggregation of this star schema.
- There are two ways to lower the level of aggregations of a star schema:

1) Add a new dimension

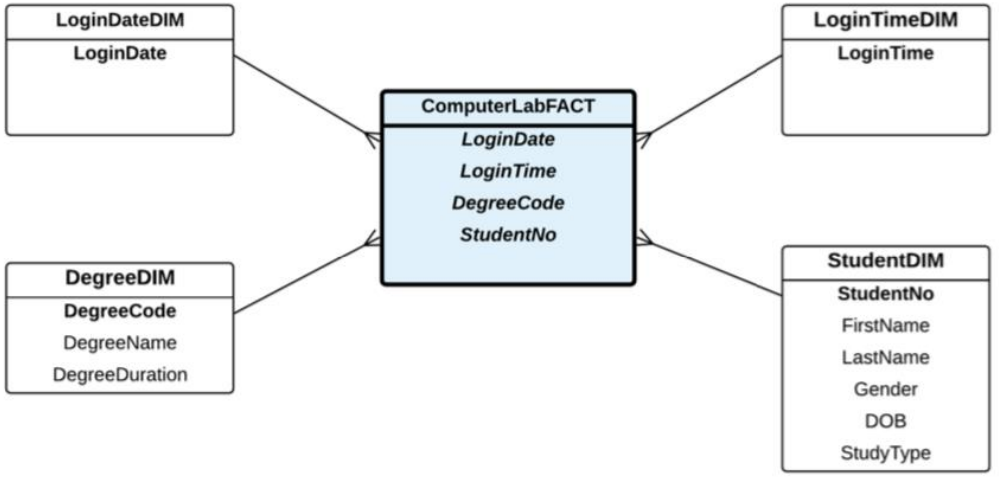
- When we add a new dimension, each value in the fact measure will literally be broken down more details on each record of the new dimension.

2) **Replace an existing dimension with a higher granularity dimension**

- The values of the fact measures will also be broken down more details because the fact measure has a lower detail dimension.

2. Facts without Fact Measures

- If the fact measure only has the value of 1, then the fact measure may be excluded from the fact.
- Consequently, the Level-0 star schema of the Computer Lab Usage case study has no fact measure.

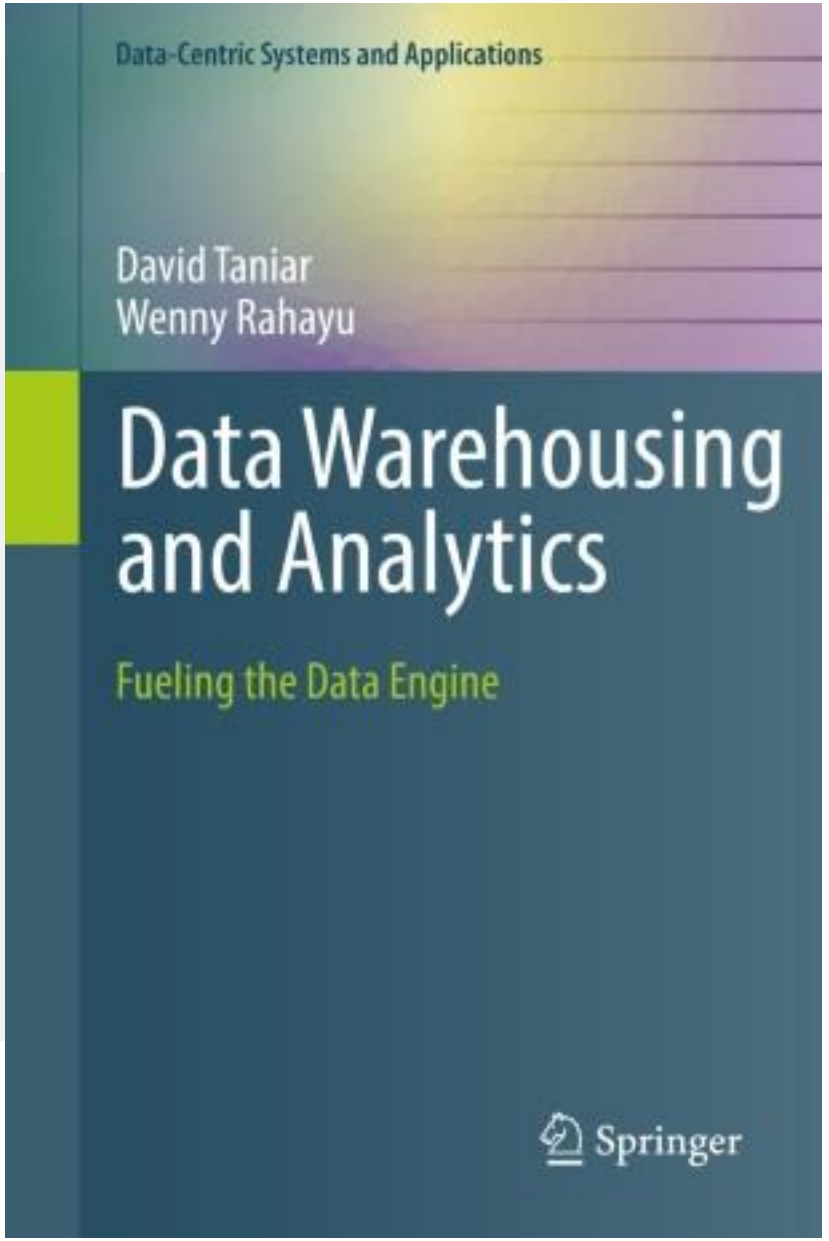


2. Facts without Fact Measures

- Facts without fact measures are only possible if the fact measure is a count, such as Num of Logins in this case. If the fact measure is not a count, but some other aggregate functions such as sum (e.g. Total Sales), then Level-0 star schema must still include the fact measure (e.g. the amount of sales of each sales).

Chapter 21

Data Analytics for Data Warehousing

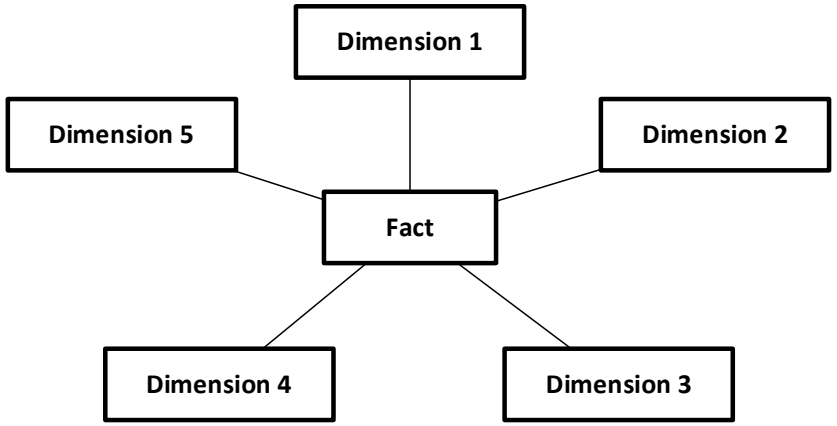


Outline

- Data analytics for data warehousing focuses on numerical value on fact in the star schema.
- This chapter focuses on three main data analysis techniques suitable in a data warehousing context, which are:
  1. Regression
  2. Clustering
  3. Classification
- Review on Traditional Data Mining Techniques

1.2. Data Analytics Requirements in Data Warehousing

- Use the data in data warehouse using Star Schema.
- Fact measures contains numerical values
- Data analysis in data warehousing is data analysis of numerical values.
- Techniques:
  - Regression
  - Clustering
  - Classification



1.2. Data Analytics Requirements in Data Warehousing

Regression

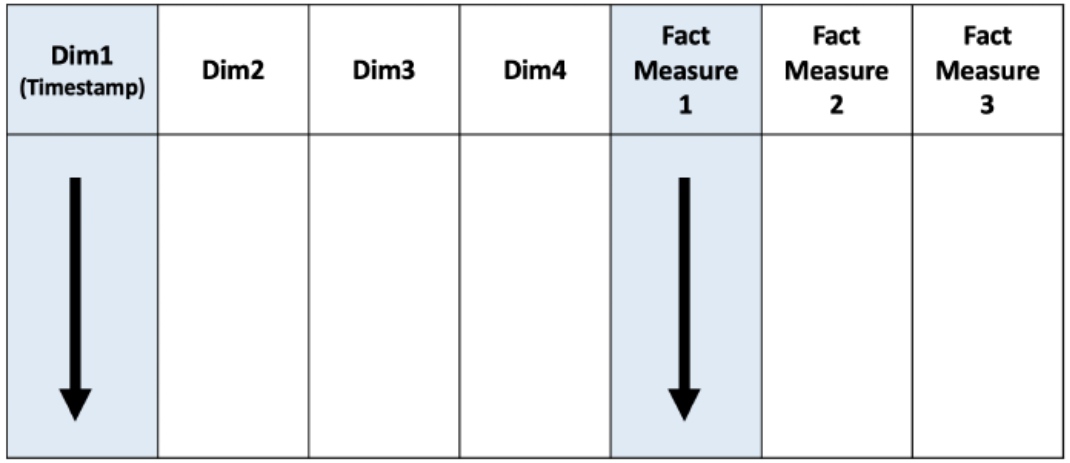


Figure 1.5: Fact Table and Time-Series Regression

Clustering and Classification

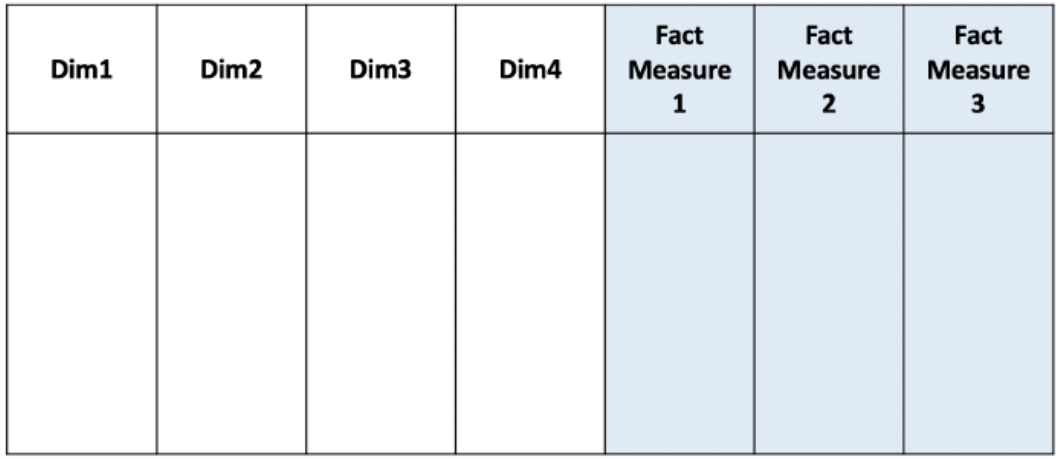


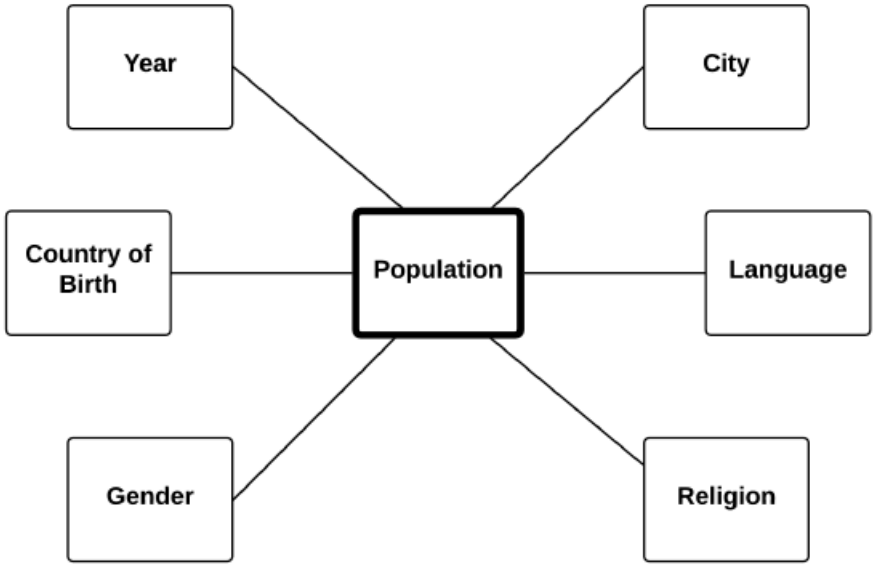
Figure 1.6: Data Analysis of Fact Measures

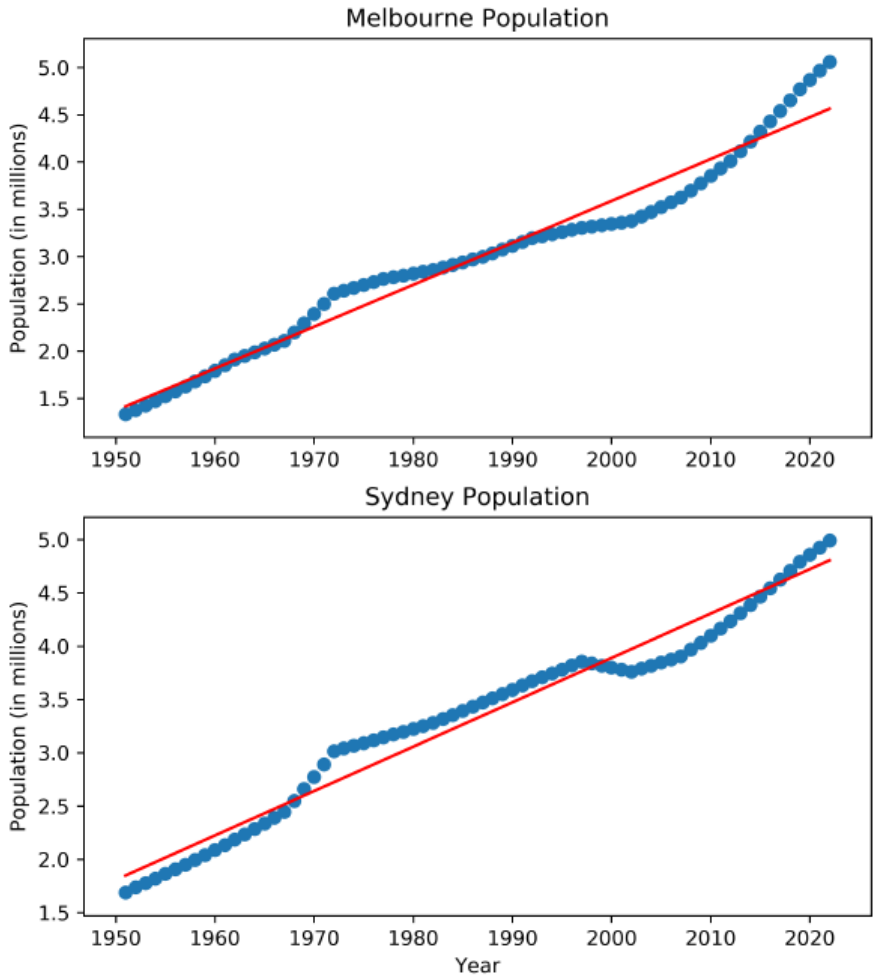
2. Statistical Method: Regression

- Regression Types:
  - Simple Linear Regression
  - Polynomial Regression
- Based on Fact measures:
  - Time Series
  - Non Time-Series

2.1. Simple Linear Regression: Example 1

Population Star Schema



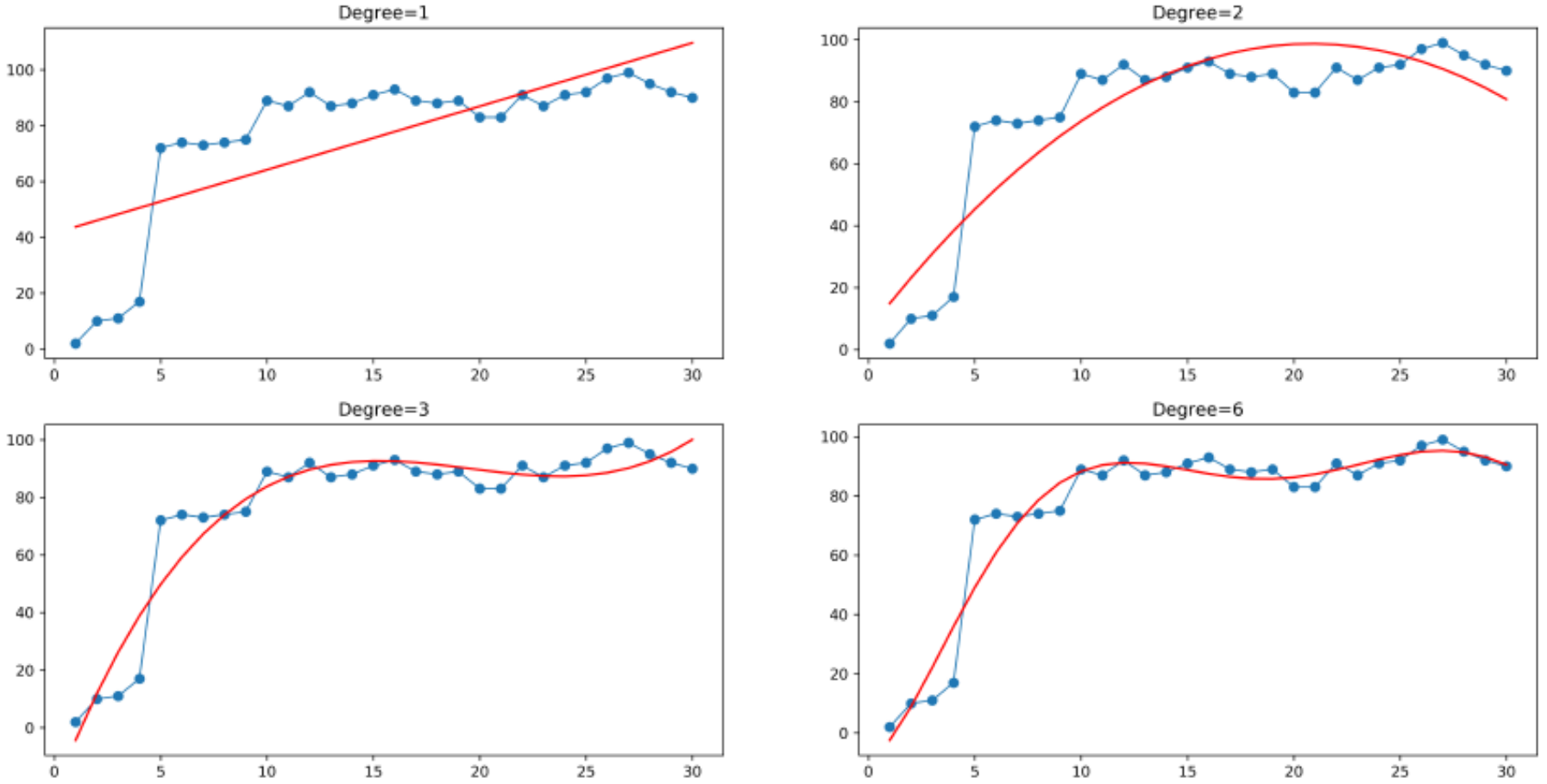


2.2. Polynomial Regression

- Data distribution is more complex and may not be linear.
- The degree is indicated by n
- The model is represented by curve.

$$y = b_0 + (b_1 \times x) + (b_2 \times x^2) + (b_3 \times x^3) + \ldots + (b_n \times x^n)$$

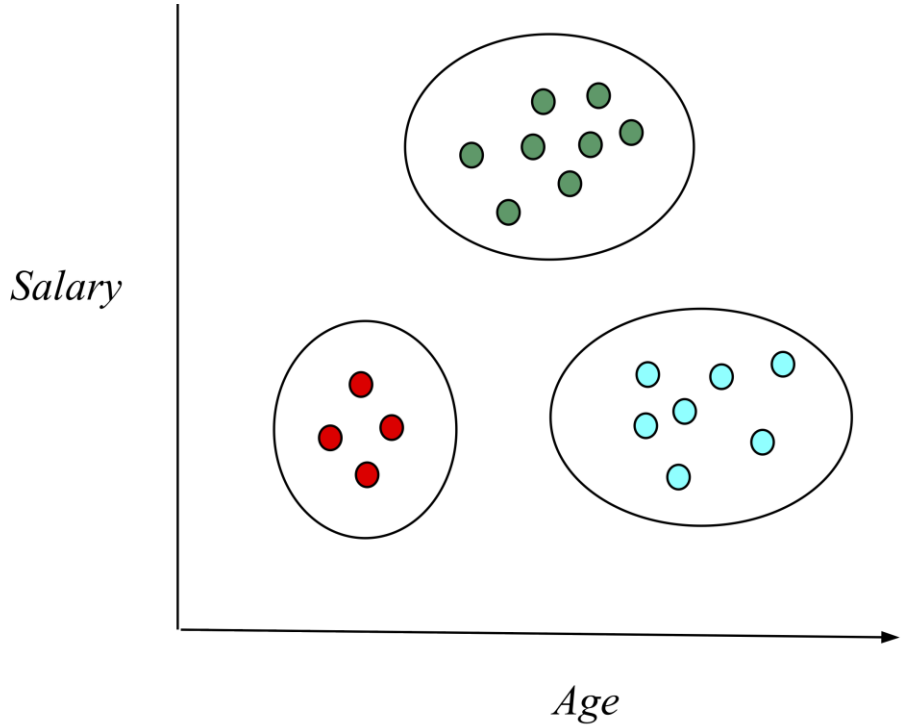
2.2. Polynomial Regression: Example



3. Clustering Analysis

- Finding groups or clusters in data.
- Members within a cluster are considered to be closer or similar
- Distance formula is used to define the closeness
- No category label
- unsupervised learning

$$dist(x_i, x_j) = \sqrt{\sum_{k=1}^{h} (x_{ik} - x_{jk})^2}$$



3. Clustering Analysis

Two types of Clustering Analysis

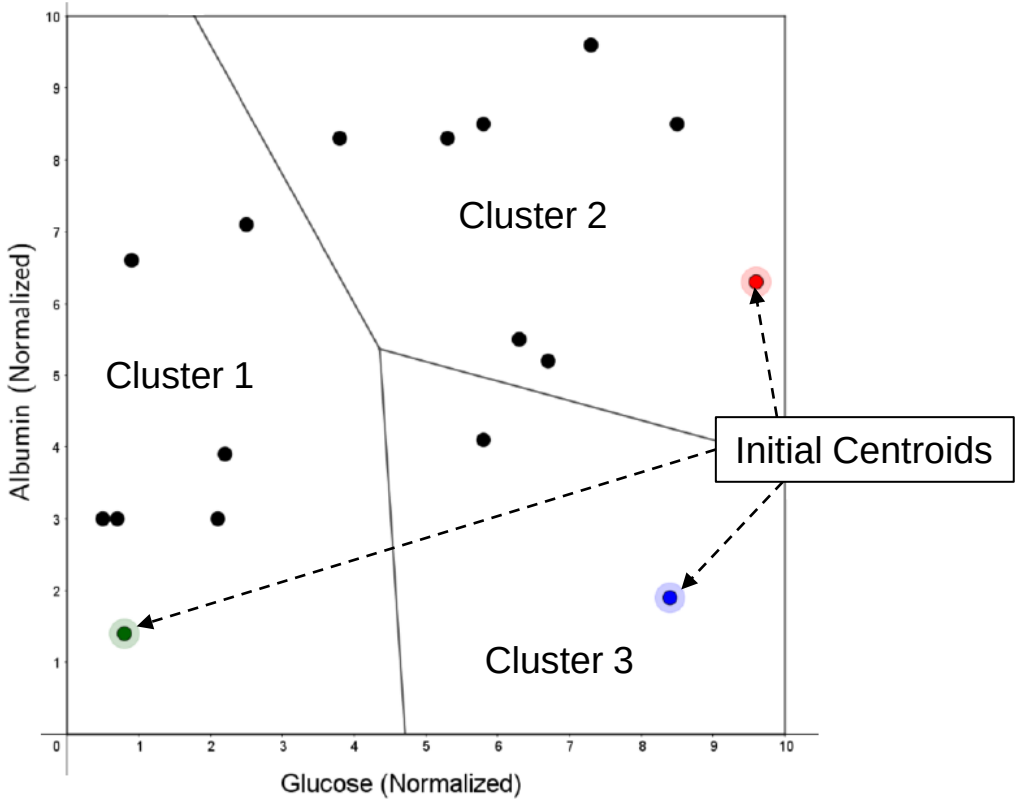
- Centroid-based clustering
  - The number of desired clusters is predefined
  - To divide the objects into the predefined number of clusters
- Density-based clustering
  - Does not require to predefine the number of clusters
  - To determine the ideal number of clusters

3.1. Centroid-based Clustering

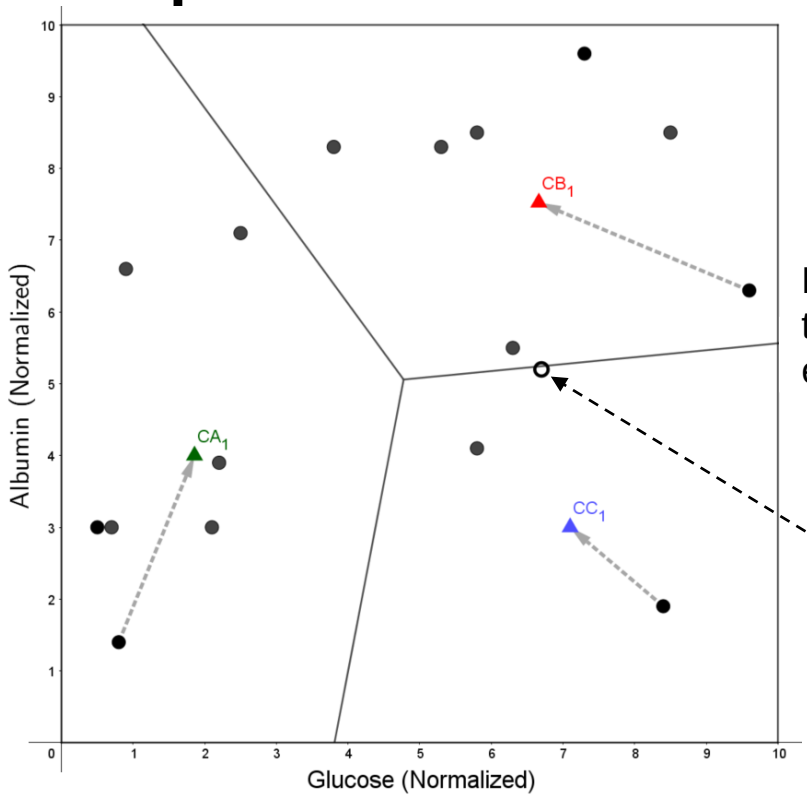
- Objects are mutual-exclusively partitioned into the predefined clusters
- Each cluster has a centroid
- Objects will be assigned to the nearest centroid
- Example: *k*-means clustering

3.1. Centroid-based Clustering: k-means: Example 2

Initial Clusters



Step-1

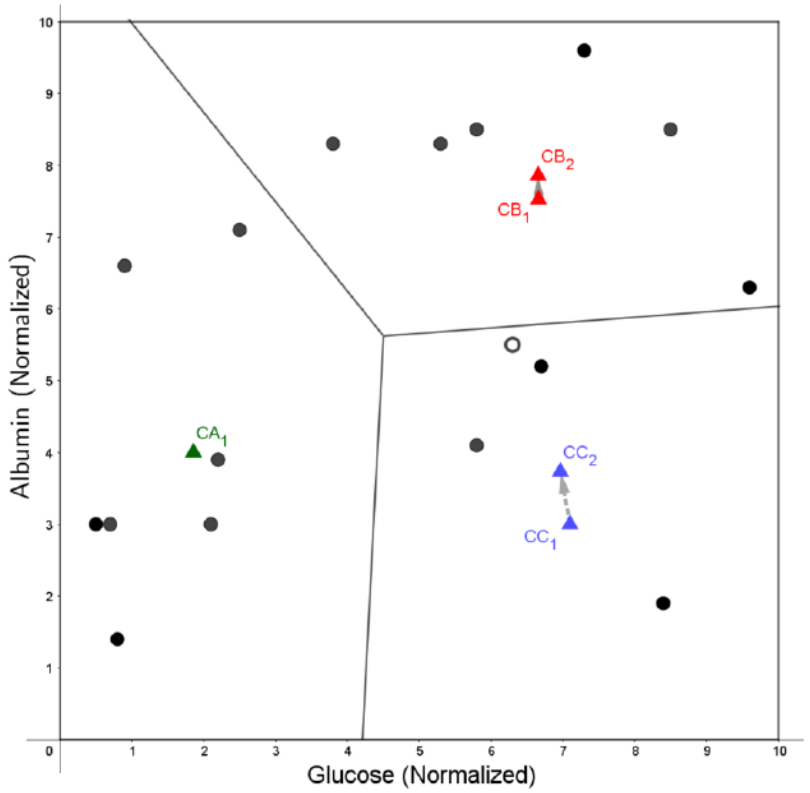


Each cluster will recalculate the new centroids from existing members

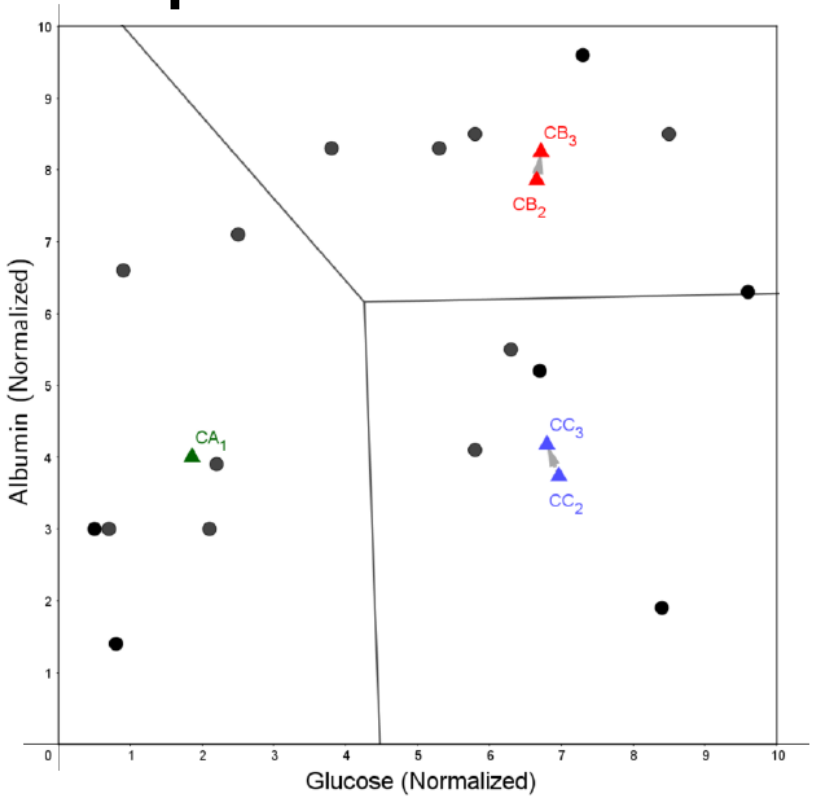
Object movement

3.1. Centroid-based Clustering: k-means: Example 2

Step-2

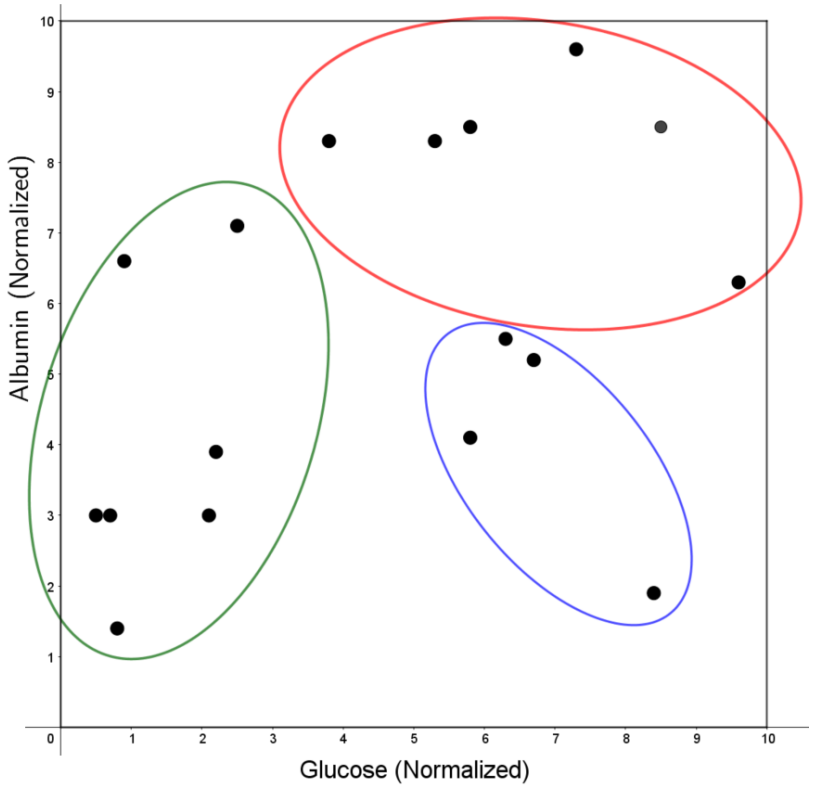


Step-3



3.1. Centroid-based Clustering: k-means: Example 2

Result



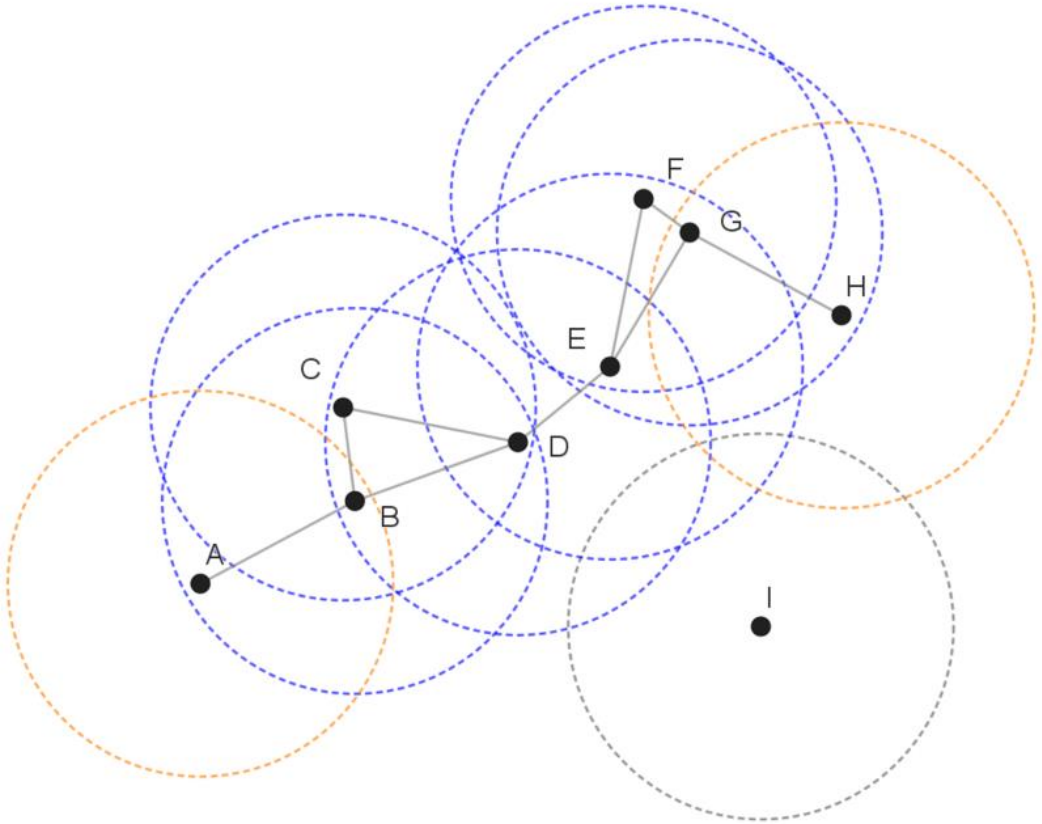
- The process terminates when no more member movements from the clusters

3.2. Density-based Clustering

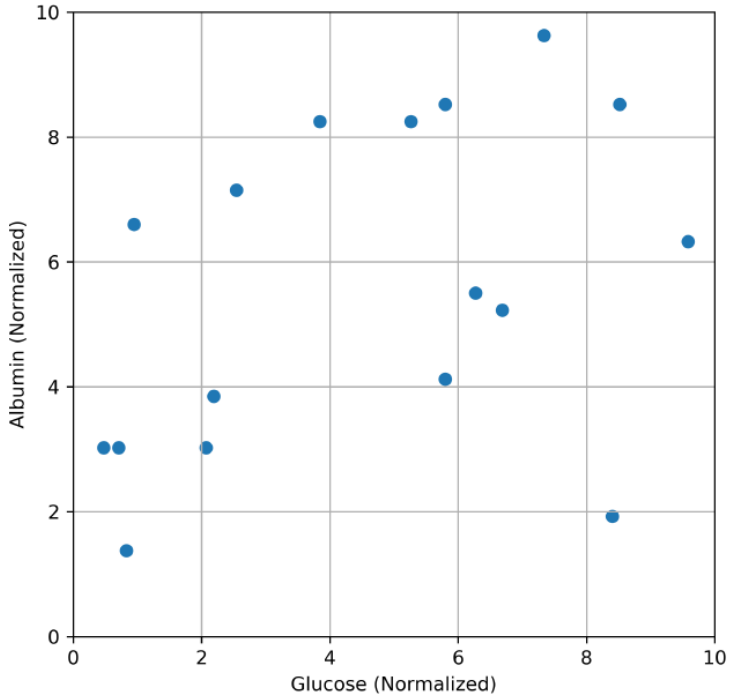
- To create a chain of object neighbourhood
- The feature of this cluster is that it is dense
- Tight proximity between one object and another
- Popular method: DBSCAN

3.2. Density-based Clustering: Example

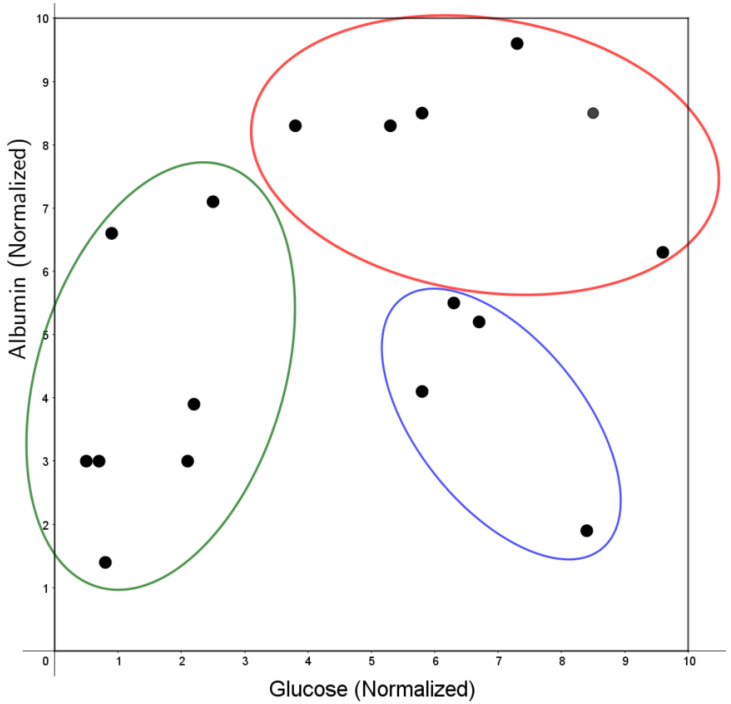
- Five important elements of DBSCAN:
  1. MaxDist
  2. MinPts
  3. Core points
  4. Border points
  5. Outliers.



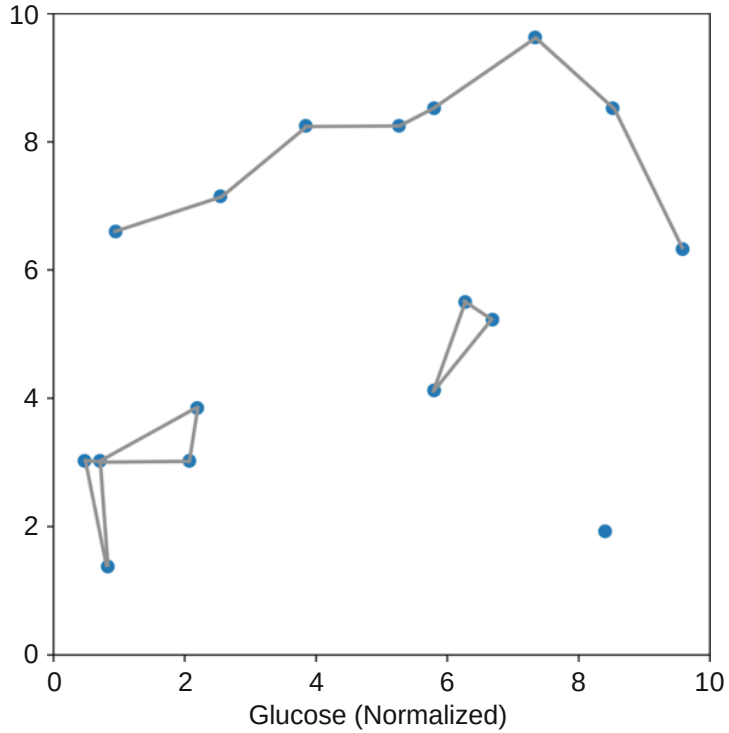
3.2. Density-based Clustering: Example



Original Normalized Data



Centroid-based Clustering (k-Means)



Density Clustering (DBSCAN)

4. Classification using Regression Trees

- Similar with Decision Tree
- Works with continues/numerical values
- Built using a training dataset
- Predicting the target class of incoming data can use the regression model

4. Classification using Regression Trees

- The main process of Regression Trees:
  1. Selecting Root Node
  2. Repeat
     a) Processing the Left Sub-Tree
     b) Processing the Right Sub-Tree
  3. Finalizing the Regression Tree
- Termination Condition
  i. The objects within a partition are cohesive enough.
  ii. The number of objects in a partition is very small

4. Classification using Regression Trees: Example

- 17 patients data
- Fact measures:
  - Glucose
  - Albumin
- Target class:
  - Mortality Prediction

Table 1.6: Emergency Patient Extended Fact Table

Patient ID	...	Glucose	Albumin	...	...	Mortality Prediction
A		162.2	3.5			0.573189504
B		93.7	2.9			0.22
C		68.5	2.6			0.217082562
D		155.0	3.0			0.534815242
E		121.0	4.5			0.475139465
F		198.0	2.2			0.969279952
G		99.1	4.1			0.552492172
H		180.2	5.0			0.752011091
I		169.0	3.4			0.799263517
J		64.9	2.6			0.383771494
K		72.1	3.9			0.45
L		155.0	4.6			0.862259153
M		218.0	3.8			0.99
N		146.0	4.5			0.813710339
P		91.9	2.6			0.496873792
Q		200.0	4.6			0.745266898
R		70.3	2.0			0.132516482

4.1. Selecting the Root Node

- Two aspects in selecting a root node:
  - Which attribute to be used as the root node?
    What condition to be applied to the branches of the root node?
- If the partition has large differences in their target value → sub-optimal partition
- Preserves Cohesiveness using ***Sum of Squared Residual (SSR)***

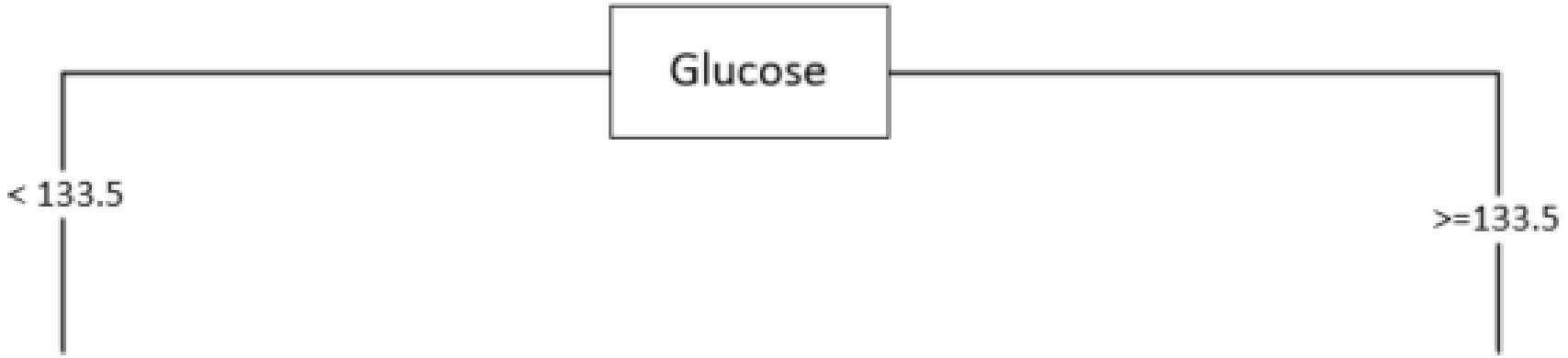
4.1. Selecting the Root Node

- A Residual value:
  The difference between an object and the average value of all objects in the same partition
- Residual is squared → the difference between each object and it's average is always a positive value.
- The lowest SSR indicates the best partitioning method.

$$SSR = \sum_{i=1}^{n}(r_i - \bar{r})^2 + \sum_{j=1}^{m}(s_j - \bar{s})^2$$

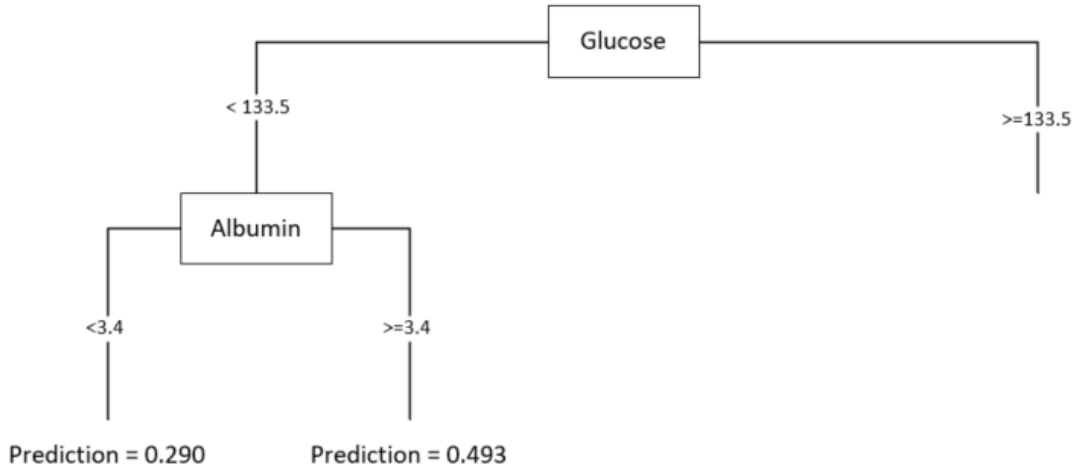
4.1. Selecting the Root Node: Example (Regression Tree)

- Min (SSR Glucose) (**0.3622)** < Min (SSR Albumin) (**0.7748**)
- Root = Glucose



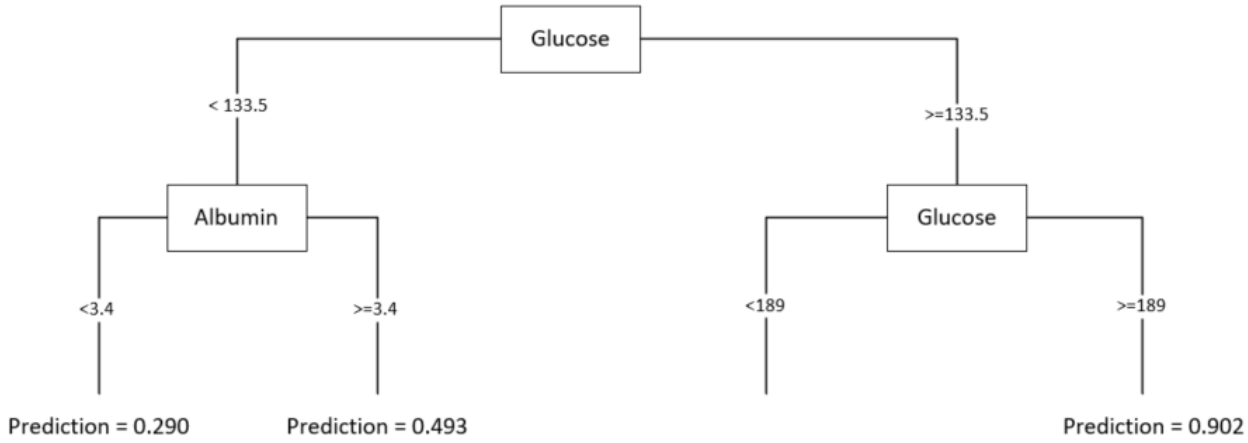
4.2. Level 1: Processing the Left Sub-Tree: Example (Regression Tree)

- Min (SSR Glucose) (**0.0983**) > Min (SSR Albumin) (**0.0923**)
- The Split Node = Albumin.



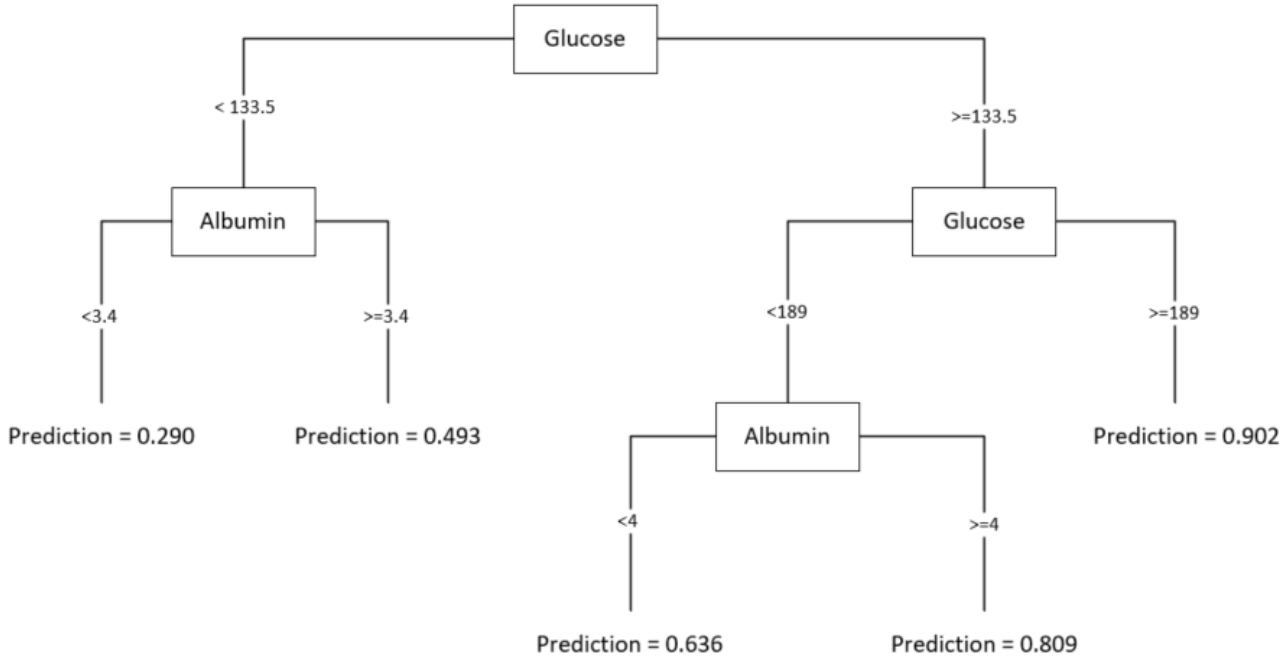
4.2. Level 1: Processing the Right Sub-Tree: Example (Regression Tree)

- Min (SSR Glucose) (**0.129**) < Min (SSR Albumin) (**0.1537**)
- The Split Node = Glucose.

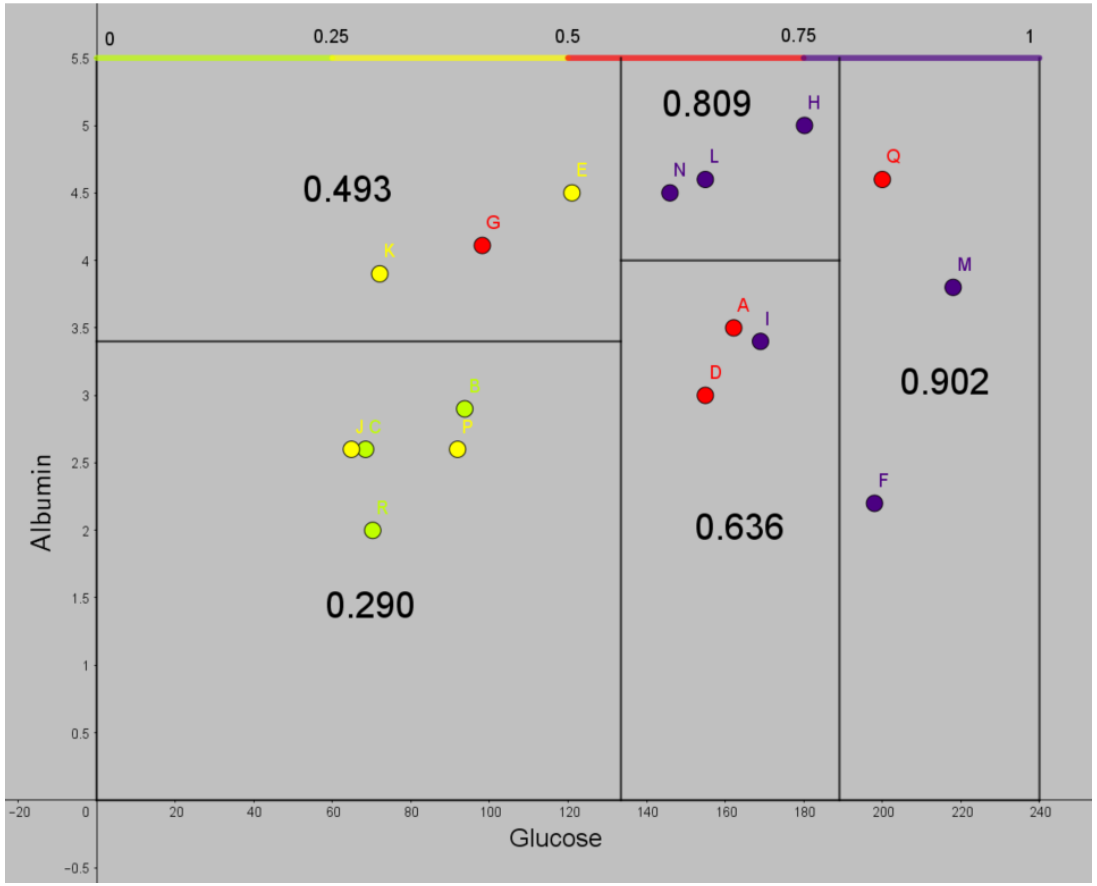


4.4. Level 2: Finalizing the Regression Tree Example (Regression Tree)

- Min(SSR Glucose) (**0.082**) > Min (SSR Albumin) (**0.047**)
- The Split Node = Albumin.



4.4. Level 2: Finalizing the Regression Tree Example (Final Regression Tree)



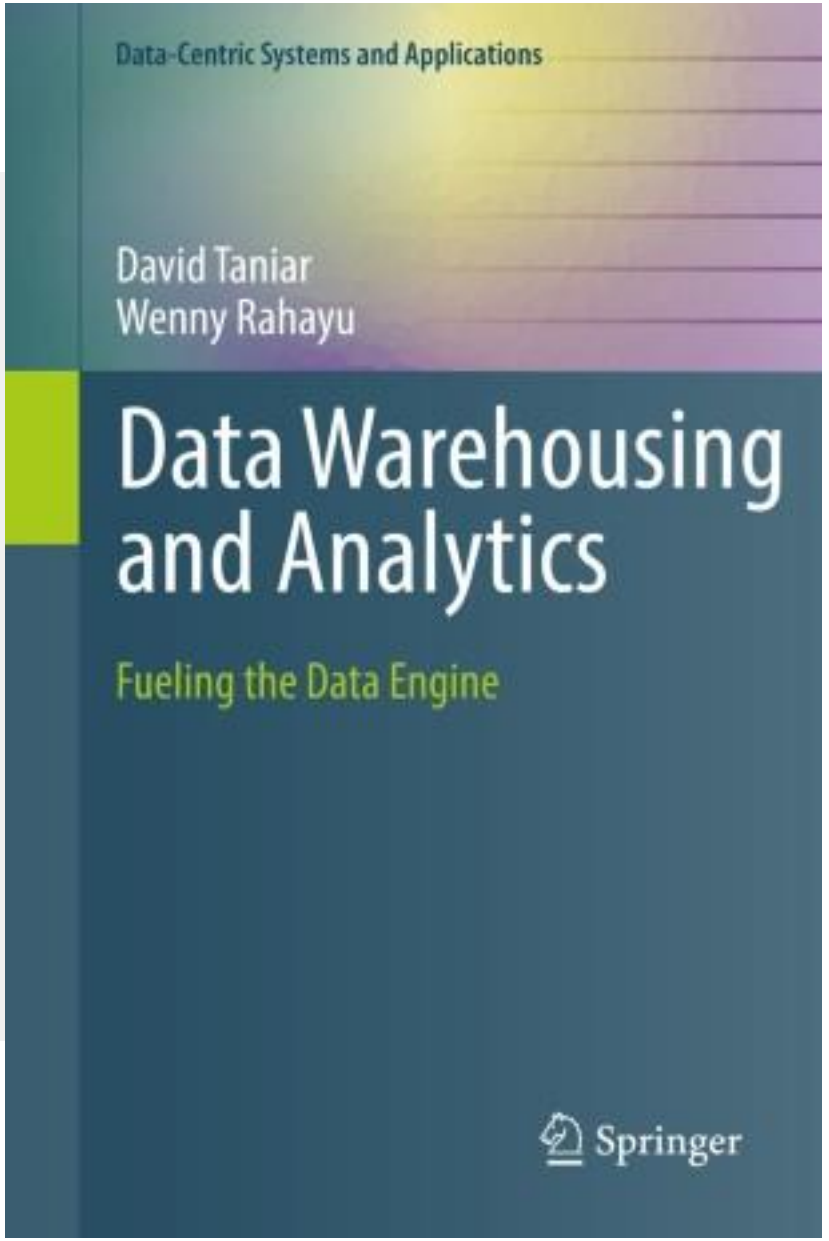
- Mortality prediction value is represented with colour (low-high)
- The root → vertical line (Glucose = 133.5)

Summary

- Data analytics for data warehousing focuses on data in the star schema.
- The focus on data analytics in data warehousing is primarily on fact measures which are numerical values.
- Three data analytics techniques suitable for data warehousing:
  i. Regression
  ii. Clustering
    a. Centroid-based
    b. Density-based
  iii. Classification using Regression Tree

Chapter 18

Active Data Warehousing



Outline

- An **Active Data Warehousing** is where the data warehouse is immediately updated when the operational database is updated.
- This chapter discusses the complexities in three parts, which are:
  1) **Incremental updates**
  2) **Data warehousing schema evolution**
  3) **Operational database evolution**

1. Passive vs. Active Data Warehousing

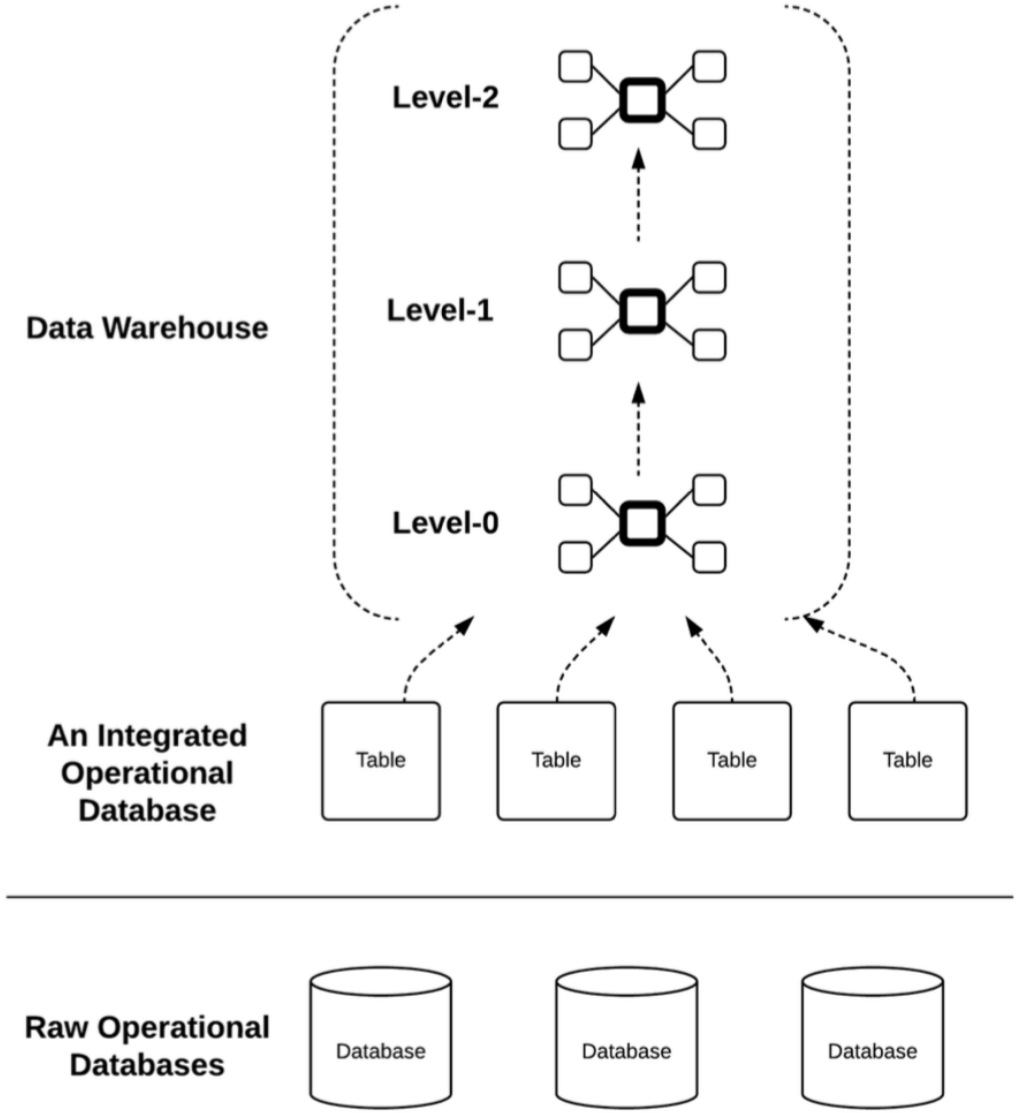
- A **Passive** **Data Warehousing** is a data warehouse that, once built, will remain unchanged.

- An **Active** **Data Warehousing** is dynamic.

- When the operational database is updated, the data warehouse is immediately updated. The data warehouse is always up-to-date, and it is no longer purely historical.

1. Passive vs. Active Data Warehousing

- The architecture of Active Data Warehousing



1. Passive vs. Active Data Warehousing

- Designing an **Active Data Warehousing** adopts a **bottom-up approach**, where the star schema is built from Level-0, and the upper levels (e.g. Level-1, Level-2, etc) are built on top of the immediate lower levels.
- Passive data warehouses may be built using the same approach, but is often easier to build a data warehouse **top-down**, that is, to start from a reasonable upper level (e.g. Level-2) and move down level-by-level to the very bottom level (e.g. Level-0).

2. Incremental Updates

- **Incremental updates** in Active Data Warehousing are not to automatically update the data warehouse.

- Other elements need to be considered, such as the "recentness" of the data in the data warehouse, as all data in the data warehouse need to be removed as they may not be as relevant as before.

- For example: data has an expire date; rules changed

2.1. Automatic Updates of Data Warehouse

- **Automatic updates** refer to updates that occur immediately once the operational database has changed.
- In the Computer Lab Activities case study, It is a simple database that consists of four tables: Lab Activities, Computer, Student, and Degree.

2.1. Automatic Updates of Data Warehouse - Level 0

- The first step in star schema implementation is to create dimension tables, and then followed by the TempFact and Fact Tables.
- Assuming that the operational database that contains the tables are within the same system, when new records are being added to the tables in the operational database, the dimension tables of the data warehouse must be updated automatically.
- Therefore, **create view** can be used to create the dimension tables, instead of create table.

2.1. Automatic Updates of Data Warehouse - Level 0

- If we would like to keep the dimension table as a table rather than a view (or a virtual table), we can use the **create table** command, but then we must have a **database trigger** that triggers an insertion of a new record.

2.1. Automatic Updates of Data Warehouse - Level 0

- If we opt for **Fact without fact measure**, then **create view** is sufficient and there is *no need for a database trigger*.

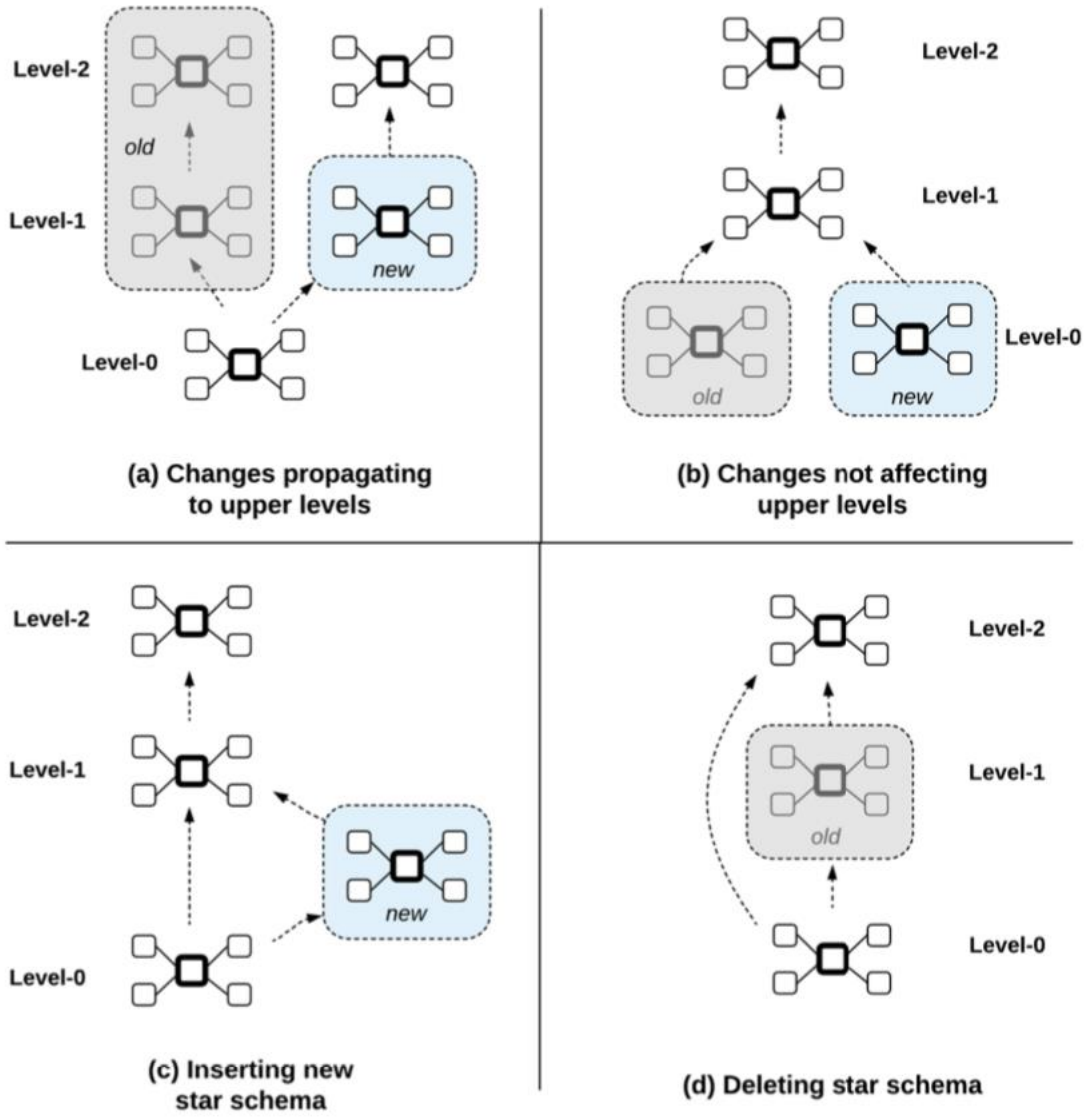
```
create or replace view ComputerLabFactLevel0 as
select distinct
  LoginDate, LoginTime,
  S.DegreeCode, L.StudentNo
from Student S, LabActivities L
where S.StudentNo = L.StudentNo;
```

3. Data Warehousing Schema Evolution

- As Active Data Warehousing is time-boundless, the data warehousing requirements first used to design the data warehouse might have evolved over time.
- There might be a need to change the requirements.
- Consequently, data warehousing schema evolves over time. This is known as **Data Warehousing Schema Evolution**.

3. Data Warehousing Schema Evolution

- There are **four types** of data warehousing schema evolution:
  1) Changes to a star schema at one level propagating to upper levels
  2) Changes to a star schema at one level which do not affect the upper levels
  3) Inserting a new star schema into the data warehouse, and
  4) Deleting a star schema from a data warehouse



3.1. Changes Propagating to the Next Levels

- In a data warehouse consisting of star schemas of various levels of granularity, when one star schema changes the structure, the changes may propagate to the upper levels of the star schema.
- In Active Data Warehousing, star schemas are reactive to changes, so change propagation must be dealt with in the next levels of the star schema.

3.1. Changes Propagating to the Next Levels

- In the Lab Activities case study, the Semester Dimension does not contain any information about the Year. This means that the analysis is purely based on semester.
- The way to deal with this situation is by including the **Year** information into the **semester**. Therefore, there will be a separate record for each semester/year.

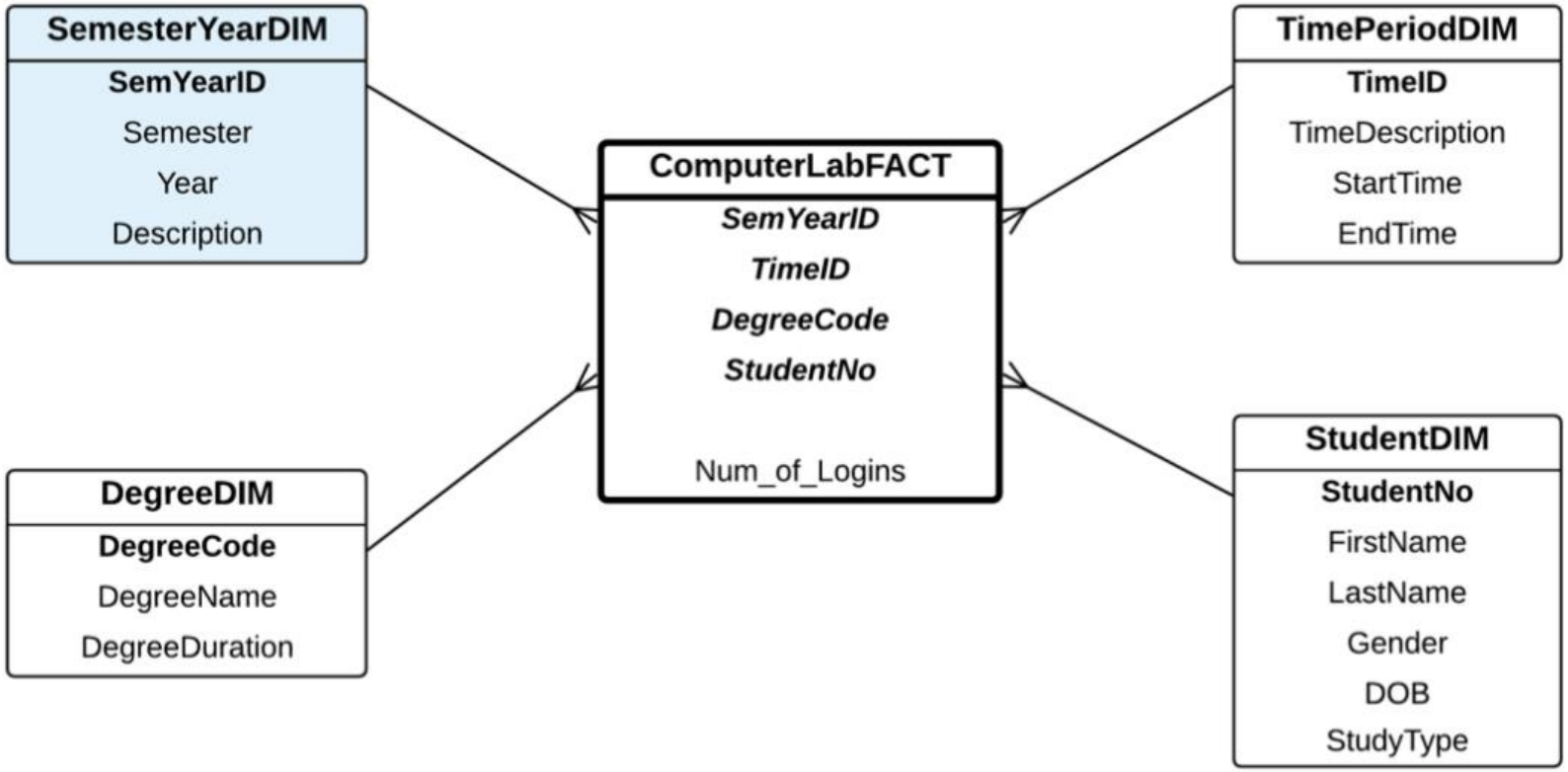
3.1. Changes Propagating to the Next Levels - Level 0

- The semester and year information will not appear until Level-1, so there are no changes to the **Level-0** star schema.

3.1. Changes Propagating to the Next Levels - Level 1

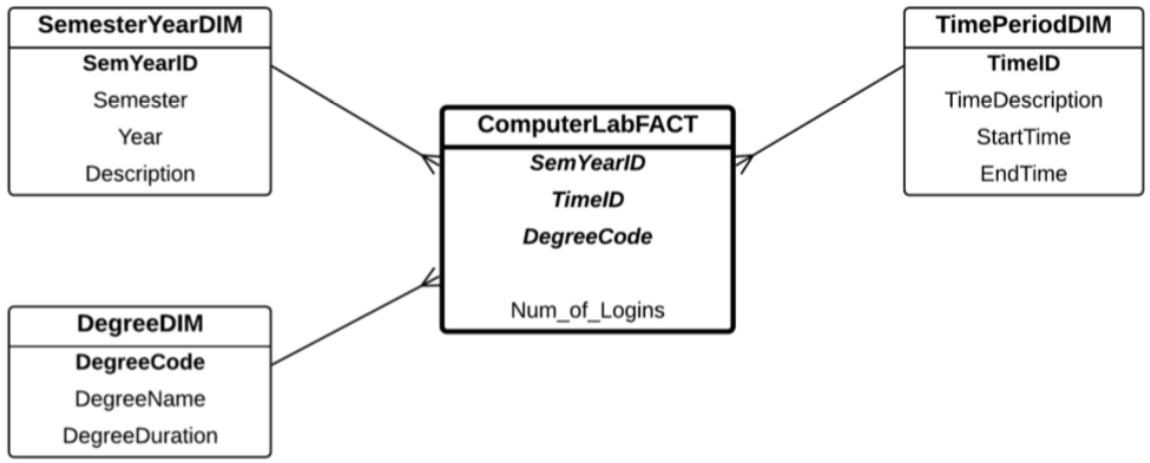
- In the **Level-1** star schema, the Semester Dimension is now changed to the **Semester Year Dimension**, incorporating the year information to the dimension.
- Other dimensions remain unchanged.

3.1. Changes Propagating to the Next Levels - Level 1



3.1. Changes Propagating to the Next Levels - Level 2

- The changes in the Level-1 star schema, namely the changes of the Semester Year Dimension, are carried forward to the upper level: Level-2.
- The **Level-2** star schema basically removes the Student Dimension from Level-1.



3.2. Changes **Not Affecting** the Next Levels

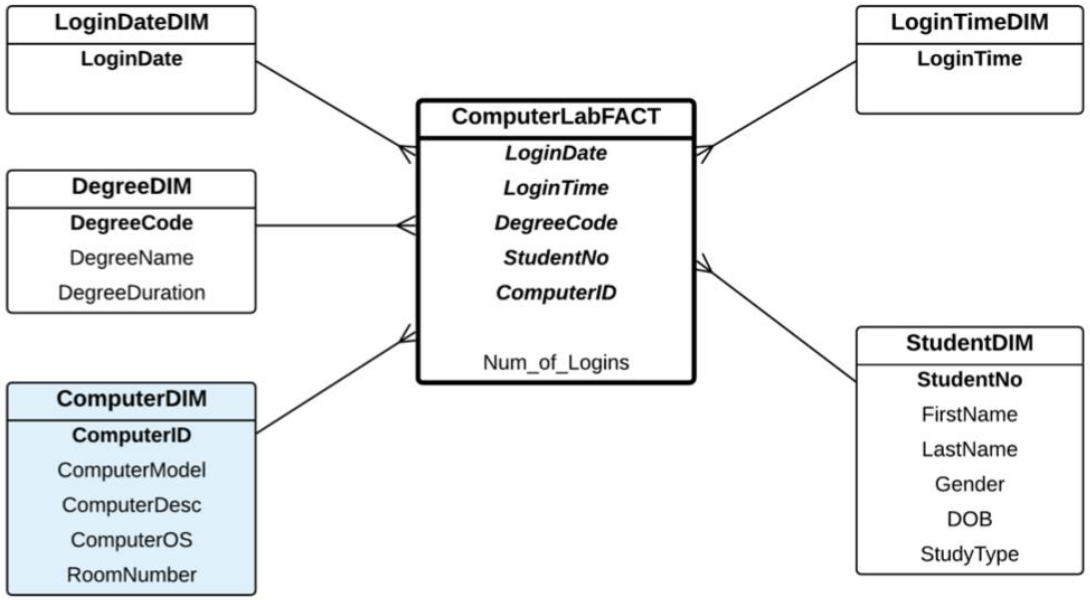
- There are cases where the changes to a star schema on one level **will not affect the next levels**. Hence, the changes are isolated to that particular star schema.
- In the Lab Activities case study, assuming now that the requirement needs to include the Computer information in the data warehouse, the **Computer Dimension** needs to be added to the **Level-0** star schema.
- The other four dimensions are unchanged and the fact measure also remains the same.

3.2. Changes Not Affecting the Next Levels

The coding to maintain the new Computer Dimension will be done in the same way as for the other dimensions. For the fact, we will need to get the ComputerID from the Lab Activities table.

```
create table ComputerLabFactLevel0 as
select distinct
  LoginDate, LoginTime,
  S.DegreeCode, L.StudentNo, L.ComputerID,
  count(L.StudentNo) as Num_of_Logins
from Student S, LabActivities L
where S.StudentNo = L.StudentNo
group by
  LoginDate, LoginTime,
  S.DegreeCode, L.StudentNo, L.ComputerID;
```

The FK and PK-FK constraints with the dimension tables need to be done. Also, a database trigger to automatically insert records into the fact needs to be revised to accommodate the ComputerID attribute in the Fact Table.

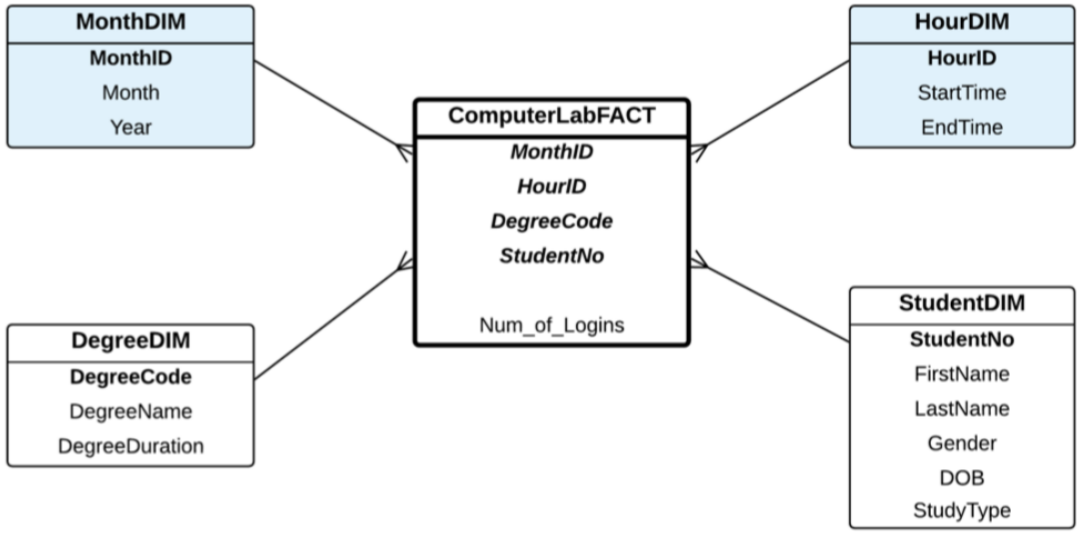


3.2. Changes Not Affecting the Next Levels

- From the schema point of view, **adding the Computer Dimension** to the Level-0 star schema will not affect the next levels since the Computer Dimension will not be used in the next levels.
- In the Level-1 star schema, granularity is reduced by focusing on semester and time period, the fact measure is now aggregated.
- There is no need to include the Computer Dimension in Level-1 as the granularity of the star schema in Level-1 is already reduced.

3.3. Inserting New Star Schema

- In the Lab Activities case study, the jump in the granularity level from Level-0 to Level-1, that is, from Login Date and Login Time to Semester Year and Time Period (e.g. morning, afternoon, night) might be seen to be wide.
- It is possible to insert a new star schema between these levels. Suppose the *proposed star schema* is at **the month and hour granularity level**.



3.3. Inserting New Star Schema - **Creation** of the new star schema

- The creation of this new star schema could be done in the usual way, that is, the **Month Dimension**, and the **Hour Dimension** can be created manually.
- The other two dimensions, Degree and Student Dimensions, can be reused from the lower level.

3.3. Inserting New Star Schema - Impact on the next levels

- The **Level-1** star schema has the **Semester Year Dimension** and **Time Period Dimension**. *These two dimensions were created directly using the Lab Activities table in the operational database*. They do not rely on the lower level star schemas.

- The **TempFact** was created based on the Fact Table from the Level-0 star schema. Therefore, *there won't be any inter-dependency between the Level-1 star schema and the new Sub-Level-1 star schema.*

3.3. Inserting New Star Schema - Impact on the next levels

- The **Fact Table** will not be affected either because the **Level-2** star schema is based on its TempFact.

- The **Level-2** star schema will not be impacted because the dimensions are reused, and the Fact Table is created using the create view command based on the Fact Table from the Level-1 star schema.

3.4. Deleting Star Schema

- In the Lab Activities case study, there are a couple of implications for the Level-2 star schema when **the Level-1 star schema is deleted**.

- **Dimensions**: The Level-2 star schema shares three dimensions with the Level-1 star schema. When deleting the Level-1 star schema, we **only remove the Fact Table**; the dimension tables are not removed as they are still being used by the Level-2 star schema.

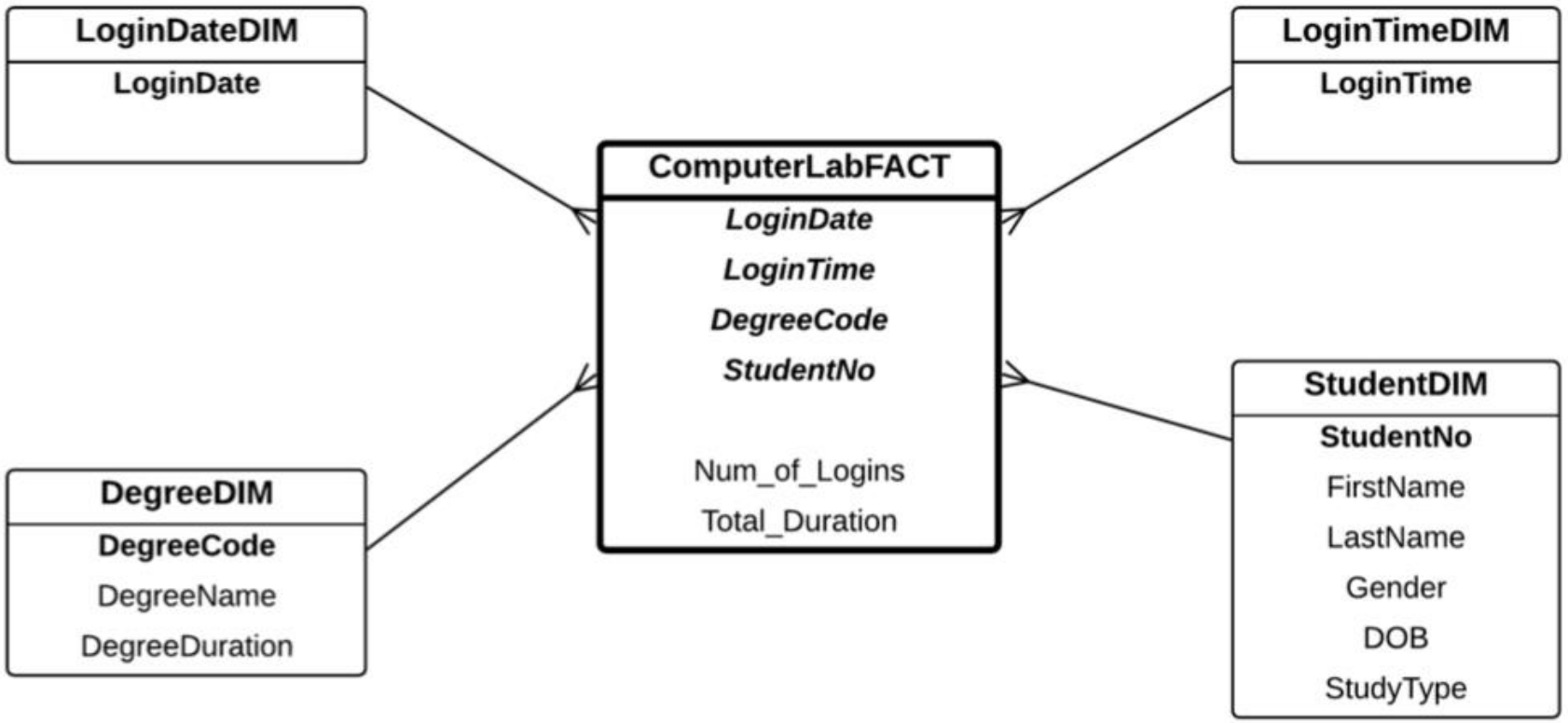
4. Operational Database Evolution

- As the operational database is changed, the data warehousing requirements may also change.

4.1. Changes in the **Table Structure**

- In the Computer Lab Activities case study, in the original system, the Login Time is recorded but not the **Logout Time**.
- Supposing the Logout Time is recorded in the Lab Activities table in the operational database, this information (e.g. Logout Time) or the derived information (e.g. Duration of Login) can be included in the data warehouse.
- In this case, a **new fact measure**: **Duration** (or **Total Duration**), can be added to the star schema.

4.1. Changes in the Table Structure



4.1. Changes in the Table Structure

There are two possible options to deal with this:

- Option 1 is to **update the existing star schemas**.

- Option 2 is to **build a new data warehouse**, since the star schema now has new requirements.

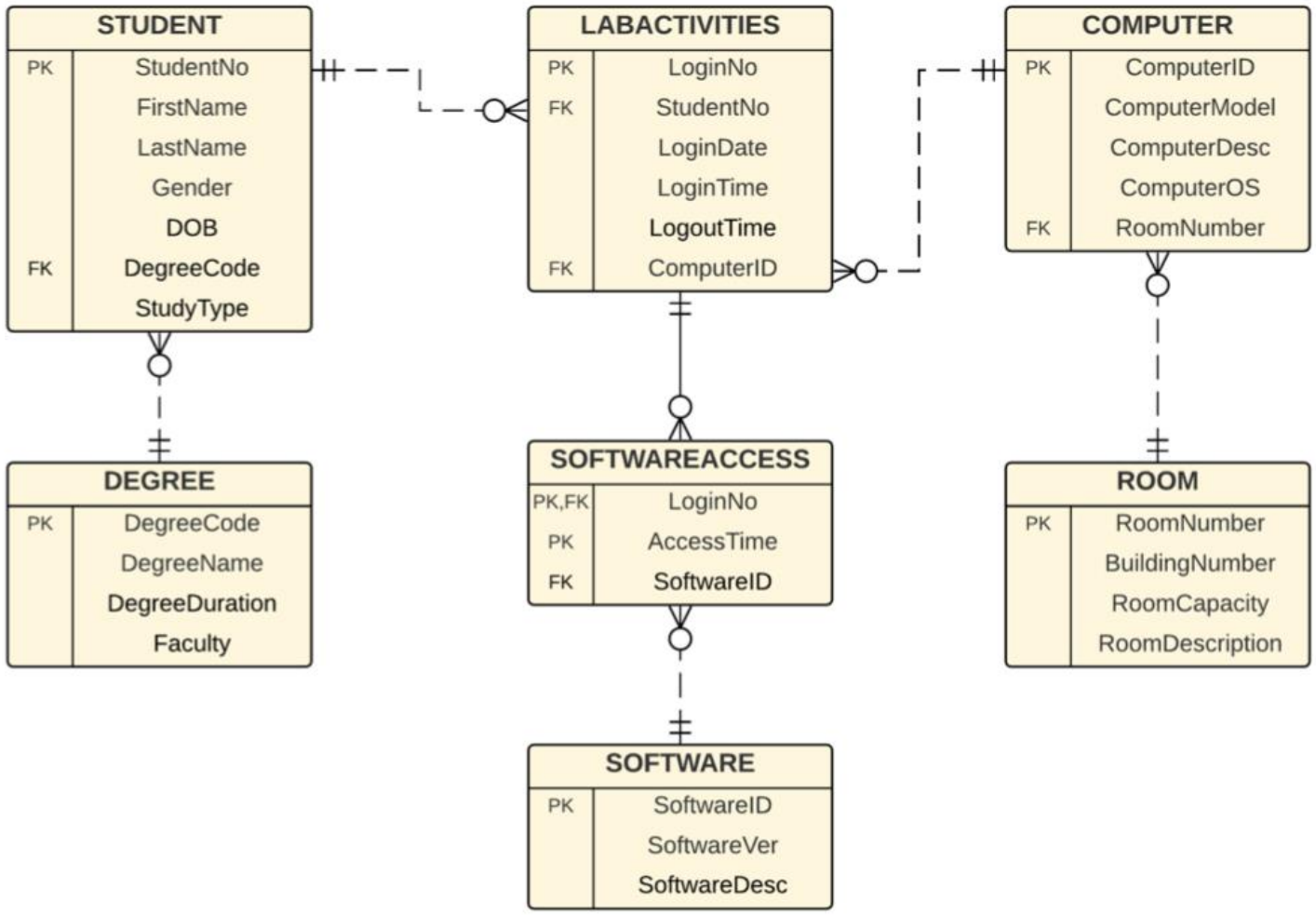
4.2. Changes in the **E/R Schema**

- Another type of change in the operational database is the addition of new entities (and hence new relationships) in the E/R diagram.
- Two areas of extension are applied in the Computer Lab Activities case study

- 1) Create an entity to store the **Room** information

- 2) Add **Software Access** and **Software** entities that related to the lab

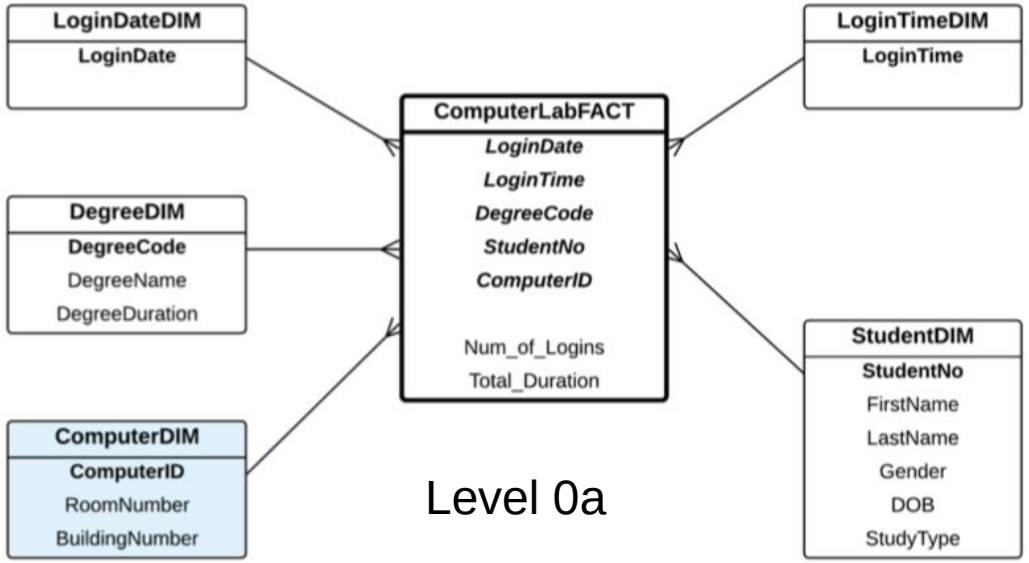
4.2. Changes in the E/R Schema



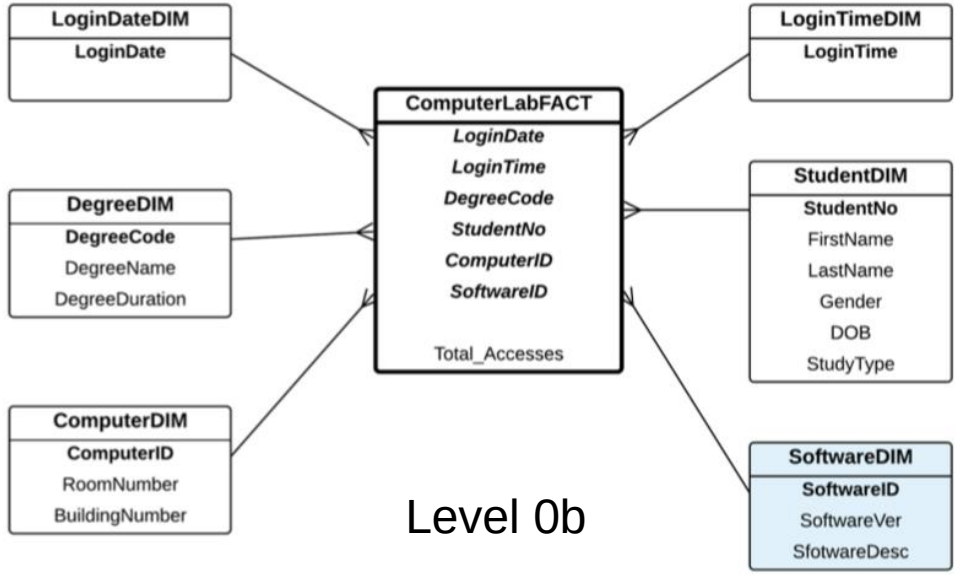
4.2. Changes in the E/R Schema

The data warehouse is now expanded into **two Level-0s**:

- One is at the **login granularity**
- The second is at the **software access granularity**



Level 0a



Level 0b

4.3. Changes in the Operational Database

- All the operational databases, in which the data warehouse is based, may change.
- The **old system is decommissioned** and the **new system is deployed**.
- **Table names may change**, **table structures may change**; the **entire design may change**.

4.3. Changes in the Operational Database

There are two options to deal with Active Data Warehousing when the operational databases have changed.

- Option 1: **Update existing star schema**
- Option 2: **Create new star schema**

Summary (Active Data Warehousing)

- Active Data Warehousing is solely about **inter-dependency** between the operational database, which is the source of the data warehouse and the data warehouse itself, as well as among the star schemas of various levels of granularity in the data warehouse.
- There are three types of inter-dependencies studied in this chapter :
  1) **Incremental updates**
  2) **Changes in the data warehouse**
  3) **Changes in the operational database**
- Since inter-dependency can be quite complex in many cases, Active Data Warehousing is used only for applications that really need active data in the data warehouse.