MONASH University

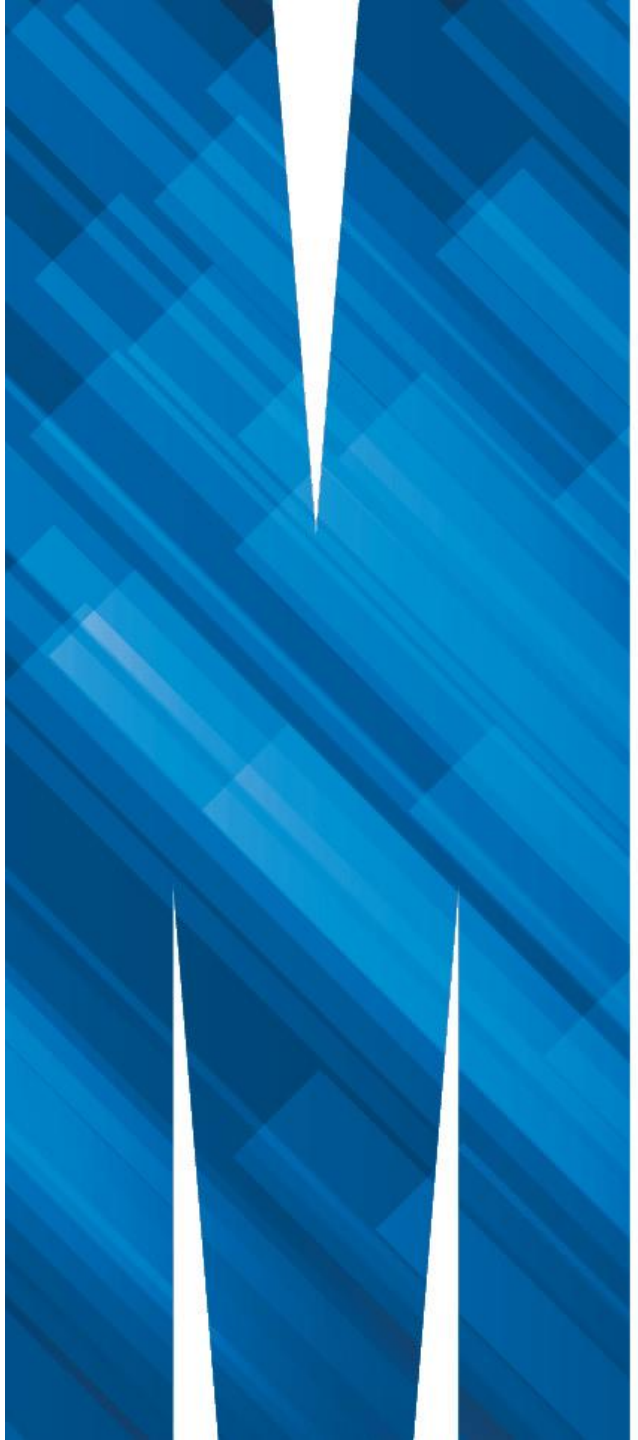
MONASH
INFORMATION
TECHNOLOGY

# FIT3003 – Business Intelligence and Data Warehousing

## Week 3 – Bridge Tables & Average in the Fact

Semester 2, 2022

Developed by:
Dr. Agnes Haryanto
Agnes.Haryanto@monash.edu



# Learning Objectives

1. Able to implement a star schema using SQL
2. Know when and why Bridge Tables are necessary to be used
3. Be familiar with the concepts of Weight Factor and ListAgg in Bridge Tables
4. Understand the concept of snowflakes and bridge tables
5. Understand why Average in a Fact should be avoided

# Agenda

1. Bridge Tables
    1. Product Sales Case Study
    2. Truck Delivery Case Study

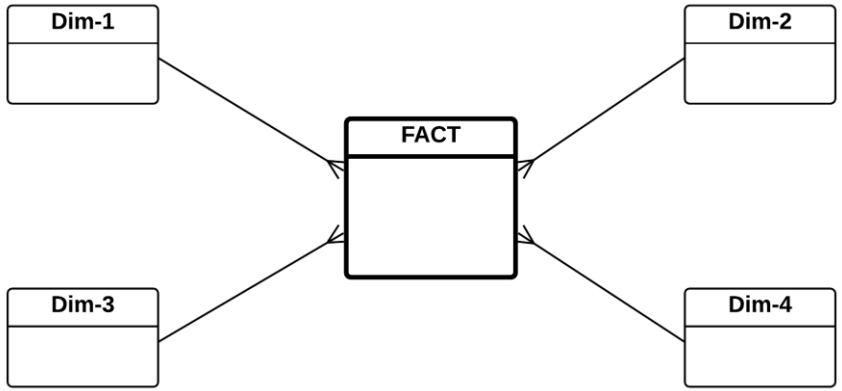
2. Average in the Fact

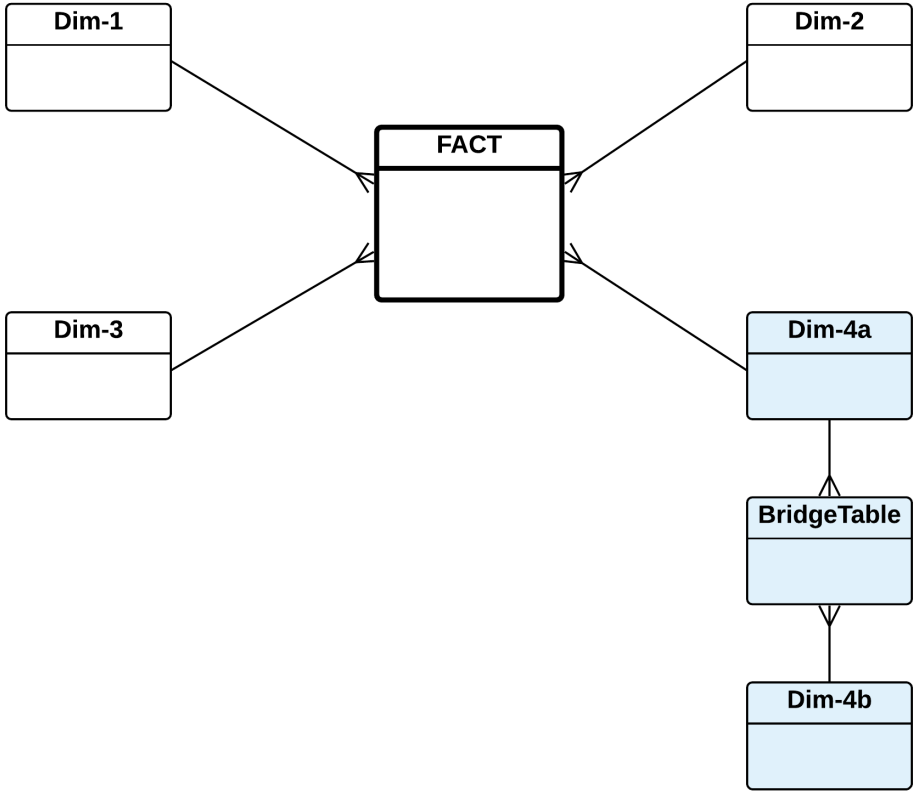
# Bridge Tables

MONASH University

# 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.
  - As a result, the star schema becomes a *snowflake schema*.



# Bridge Tables

- Two reasons on why a dimension cannot be connected directly to the Fact:

  a) The Fact table has a fact measure, and the dimension has a key identity. In order to connect a dimension to the Fact, the dimension's key identity must contribute directly to the calculation of the fact measure. Unfortunately, this cannot happen if the operational database does not have this data.

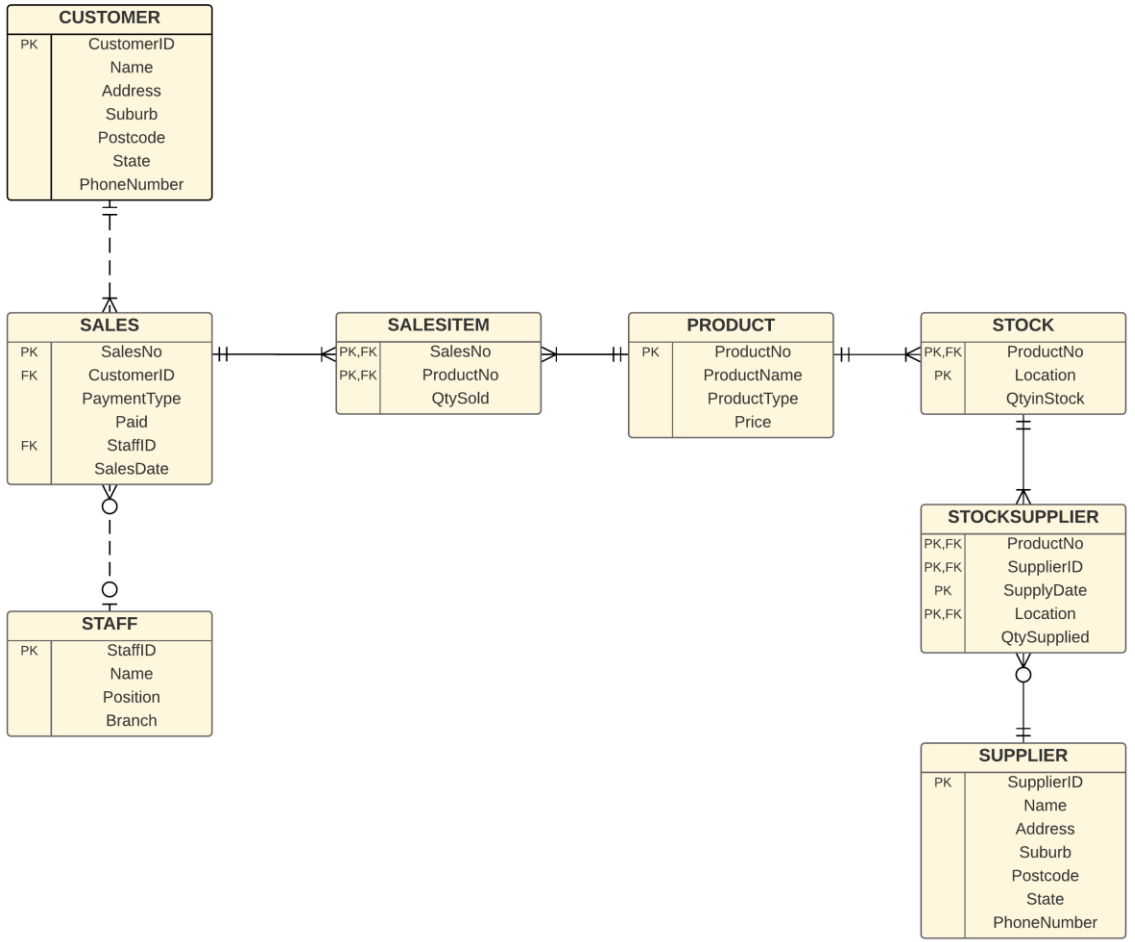
  b) The operational database does not have this data if the relationship between two entities in the operational database that hold the information about dimension's key identity and the intended fact measure is a *many-many* relationship.

# Bridge Tables
# Case Study #1

MONASH University

# Case Study #1 – A Product Sales Case Study

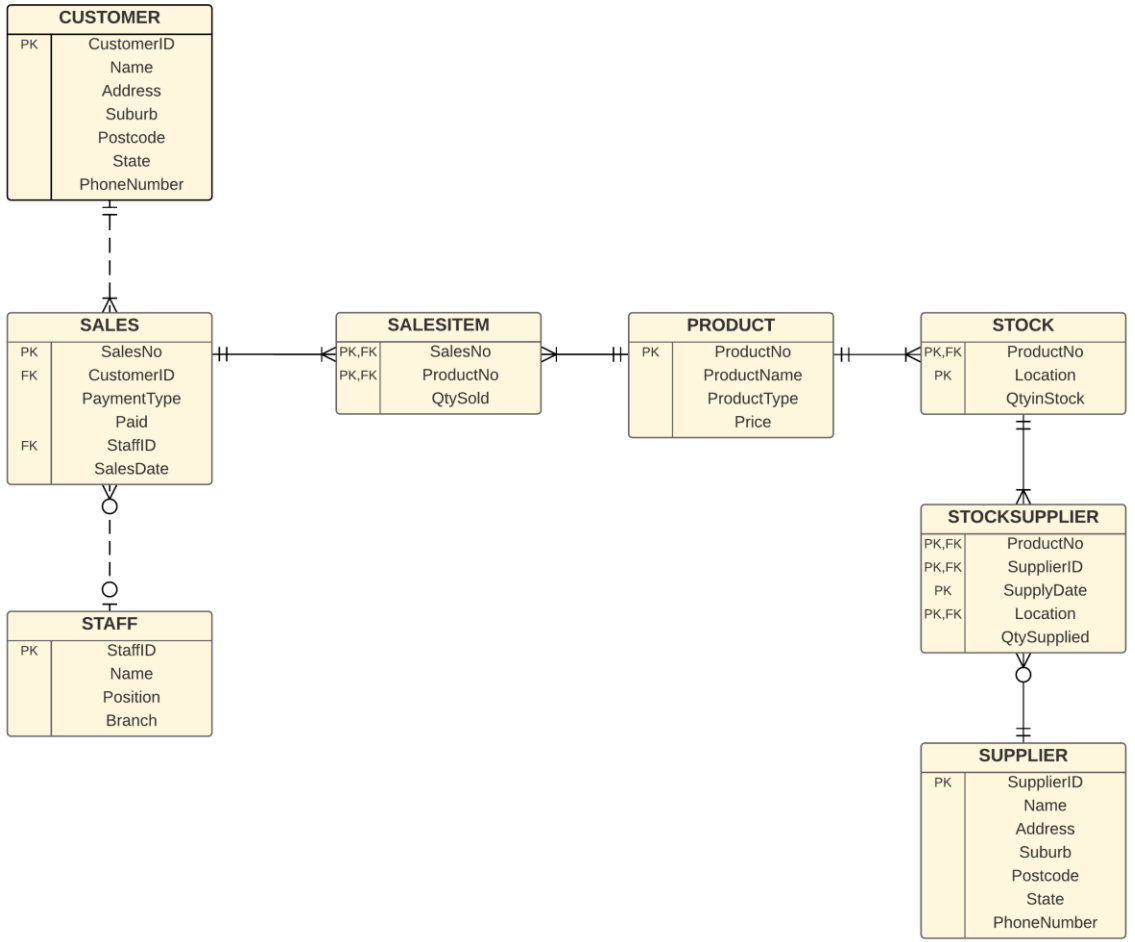
- A company management team would like to analyze the statistics of its **product sales history**. The analysis is needed to identify popular products, suppliers supplying those products, the best time to purchase more stock, etc.
- A small data warehouse is to be built to keep track of the statistics.
- The management is particularly interested in analyzing the *total sales (quantity * price)* by *product*, *customer suburbs*, *sales time periods (month and year)*, and *supplier*.




MONASH University

# Case Study #1 – A Product Sales Case Study

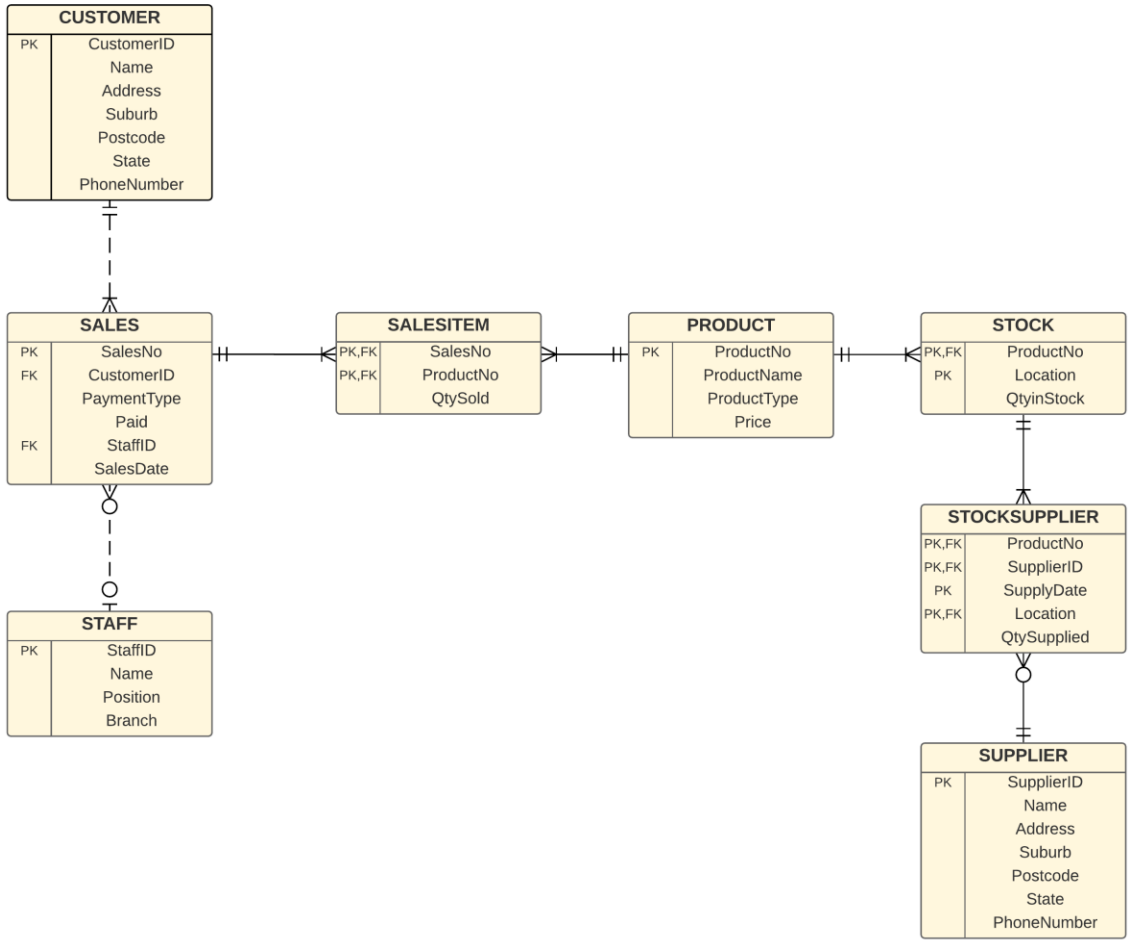
- The management is particularly interested in analyzing the *total sales (quantity * price)* by *product*, *customer suburbs*, *sales time periods (month and year)*, and *supplier*.



# Case Study #1 – A Product Sales Case Study

- The management is particularly interested in analyzing the *total sales (quantity * price)* by *product*, *customer suburbs*, *sales time periods (month and year)*, and *supplier*.

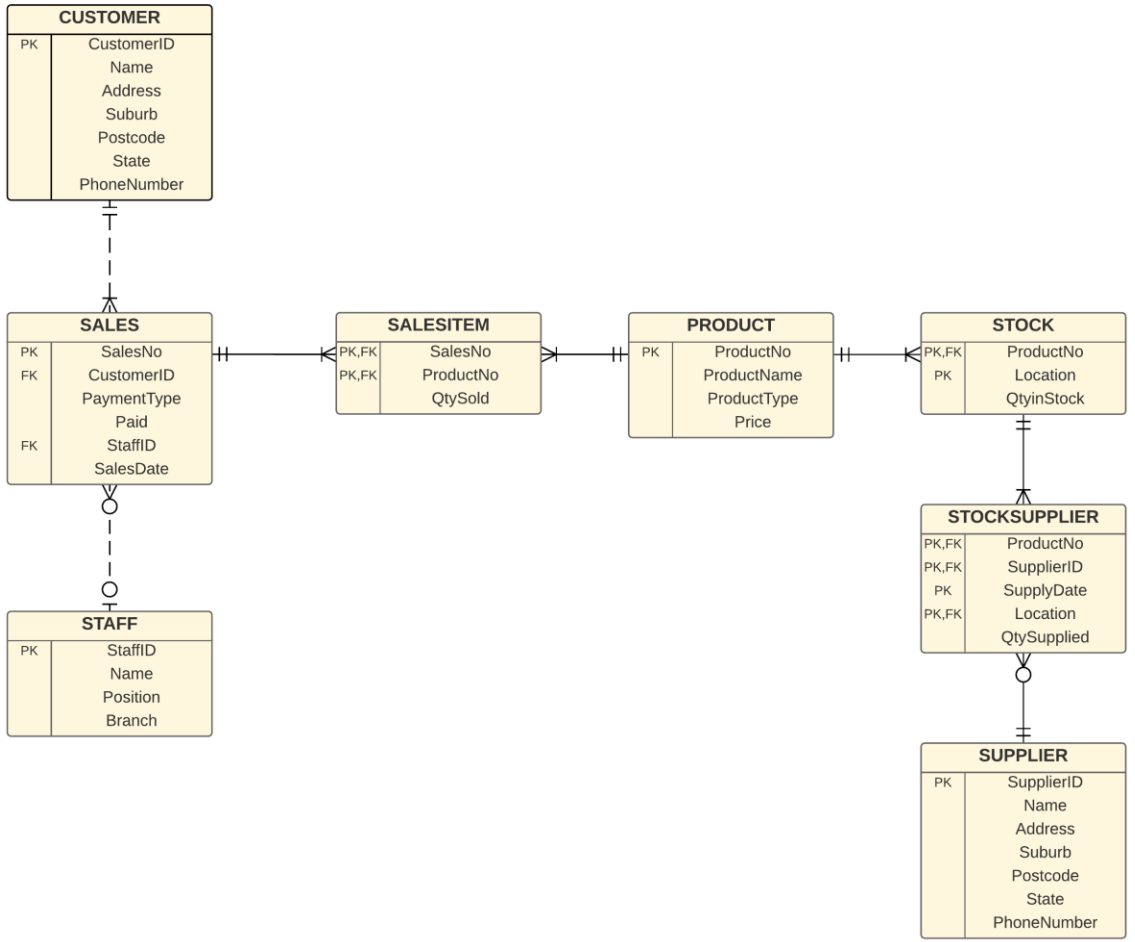
- Sales Star Schema
  - **Fact**:
    - Total Sales



# Case Study #1 – A Product Sales Case Study

- The management is particularly interested in analyzing the *total sales (quantity * price)* by *product*, *customer suburbs*, *sales time periods (month and year)*, and *supplier*.

- Sales Star Schema
  - **Fact**:
    - Total Sales
  - **Dimensions**:
    - Product
    - Customer locations/suburbs
    - Time period
    - Supplier



# Case Study #1 – A Product Sales Case Study

- Possible Two-Column Methodology Tables:

| ProductNo | TotalSales |
|---|---|
| A1 | $130,000 |
| B2 | $15,900 |
| C3 | $2,500,000 |
| … | … |

(a) Product point of view

| TimeID | TotalSales |
|---|---|
| 201801 | $25,000 |
| 201802 | $4,700 |
| 201803 | $3,500 |
| … | … |

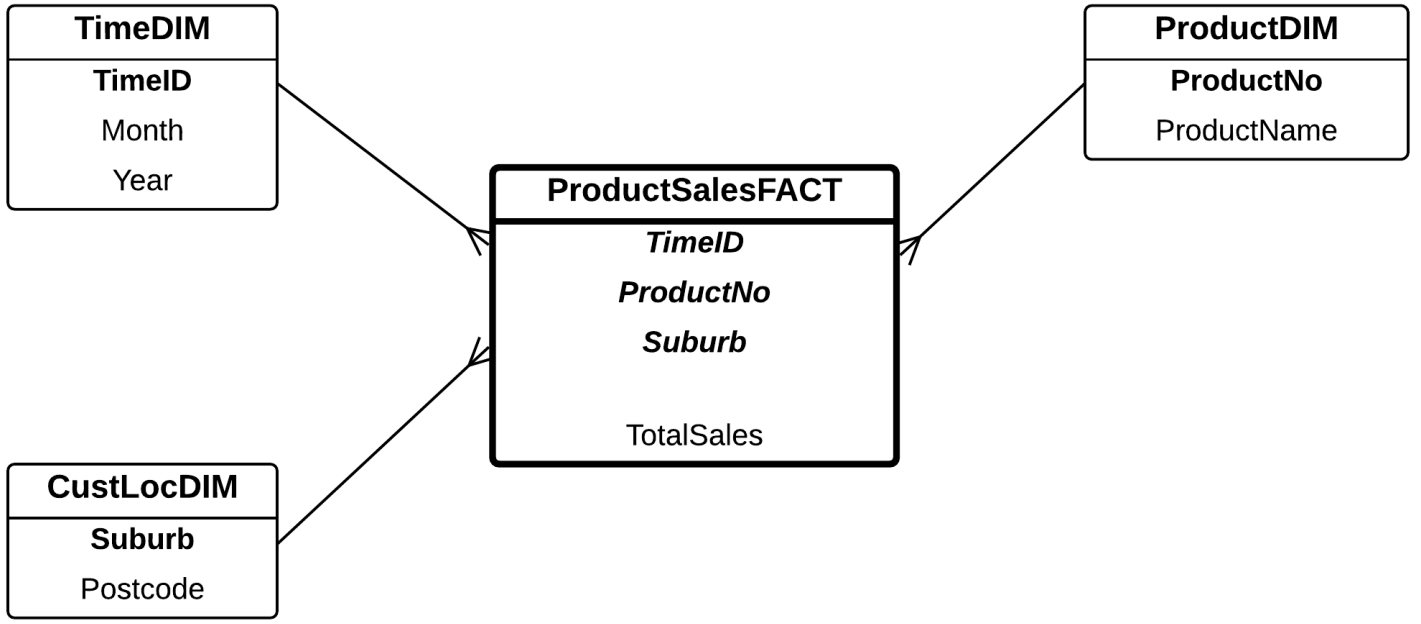
(b) Time point of view

| Suburb | TotalSales |
|---|---|
| Caulfield | $6,500 |
| Chadstone | $12,000 |
| Clayton | $1,800 |
| … | … |

(c) Suburb point of view

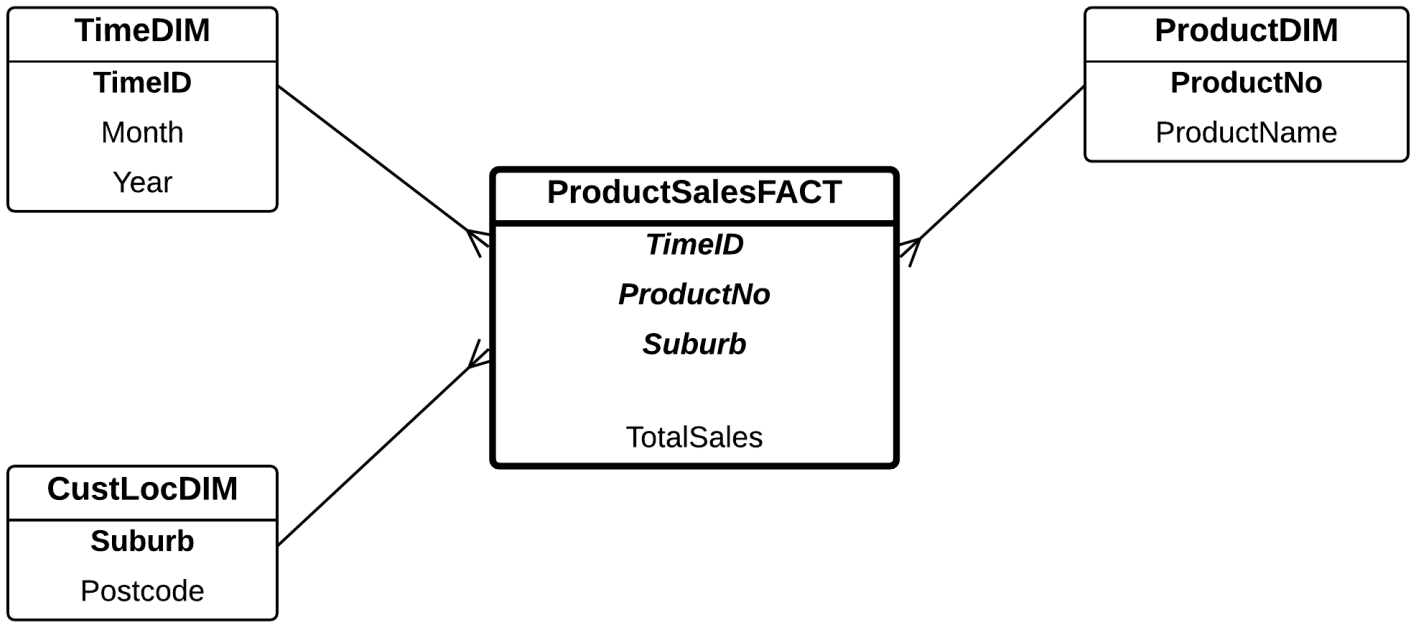
# Case Study #1 – A Product Sales Case Study

- Sales Star Schema
  - **Fact**:
    - Total Sales
  - **Dimensions**:
    - Product
    - Customer locations/suburbs
    - Time period
    - Supplier



# Case Study #1 – A Product Sales Case Study

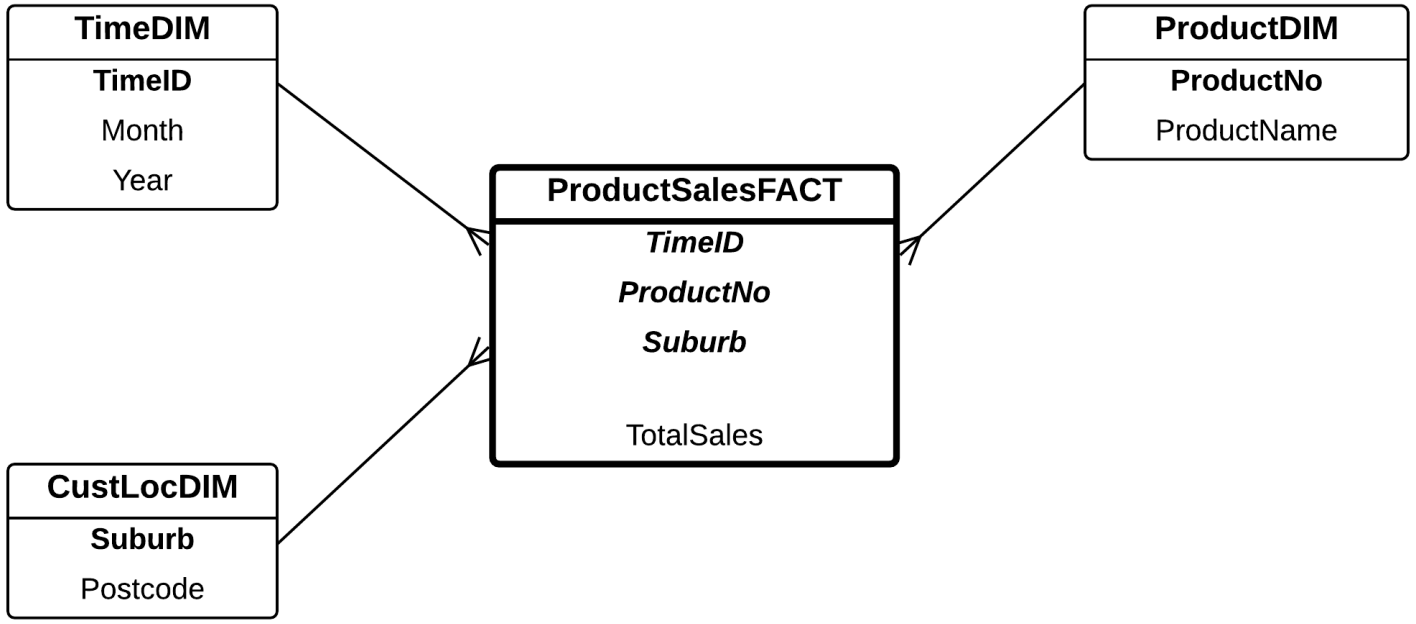
| TimeID | Suburb | ProductNo | TotalSales |
|---|---|---|---|
| 201801 | Caulfield | A1 | $450 |
| 201801 | Caulfield | B2 | $100 |
| 201801 | Caulfield | C3 | $320 |
| 201801 | Caulfield | ... | ... |
| 201801 | ... | ... | ... |
| 201801 | Chadstone | A1 | $75 |
| 201801 | Chadstone | B2 | $600 |
| 201801 | Chadstone | C3 | $55 |
| 201801 | Chadstone | ... | ... |
| 201801 | ... | ... | ... |
| 201801 | Clayton | A1 | $130 |
| 201801 | ... | ... | ... |
| 201802 | Caulfield | A1 | $500 |
| 201802 | Caulfield | B2 | $430 |
| 201802 | Caulfield | C3 | $120 |
| ... | ... | ... | ... |



# Case Study #1 – A Product Sales Case Study

| SupplierID | TotalSales |
|---|---|
| S1 | $77,000 |
| S2 | $5,700 |
| S3 | $12,500 |
| … | … |

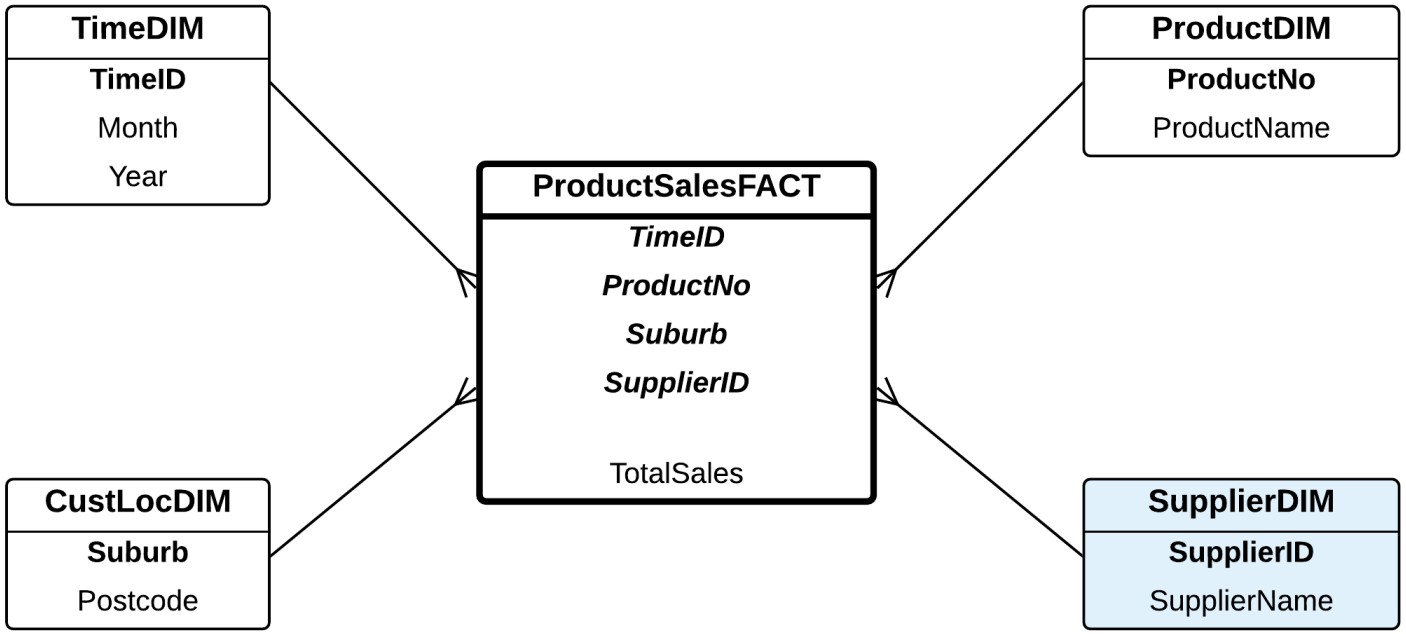
Supplier point of view



# Case Study #1 – A Product Sales Case Study

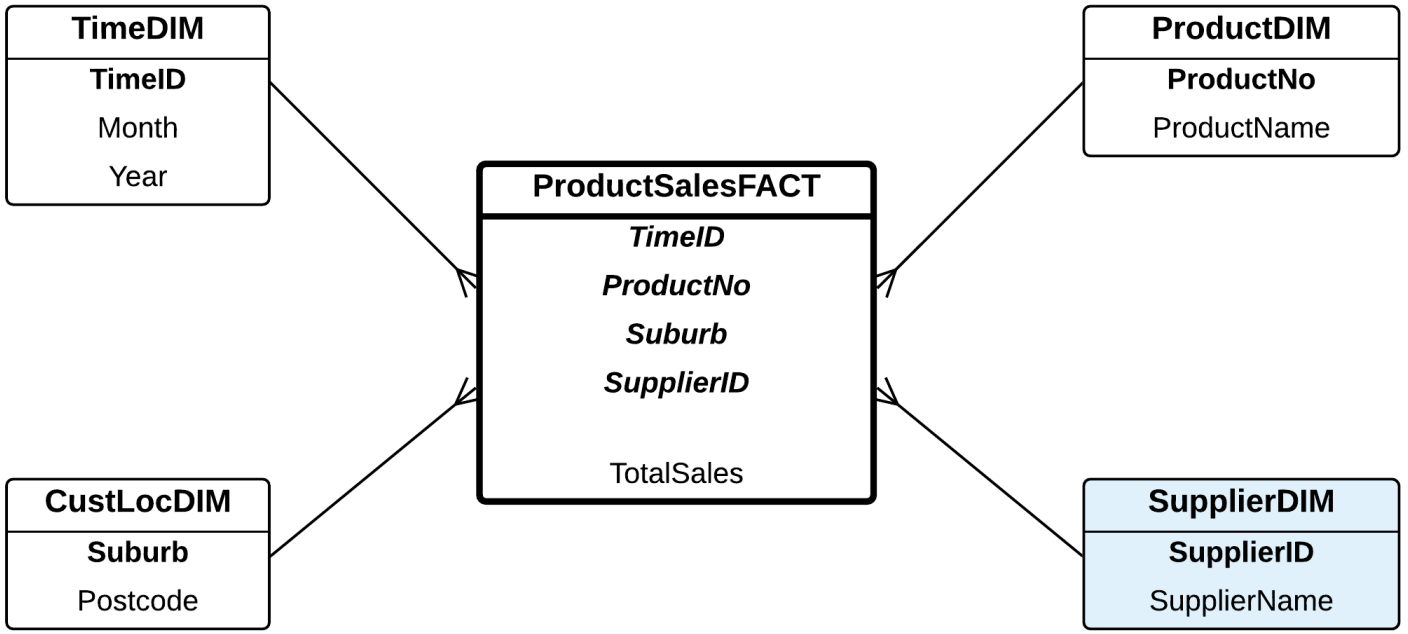
| SupplierID | TotalSales |
|---|---|
| S1 | $77,000 |
| S2 | $5,700 |
| S3 | $12,500 |
| … | … |

Supplier point of view



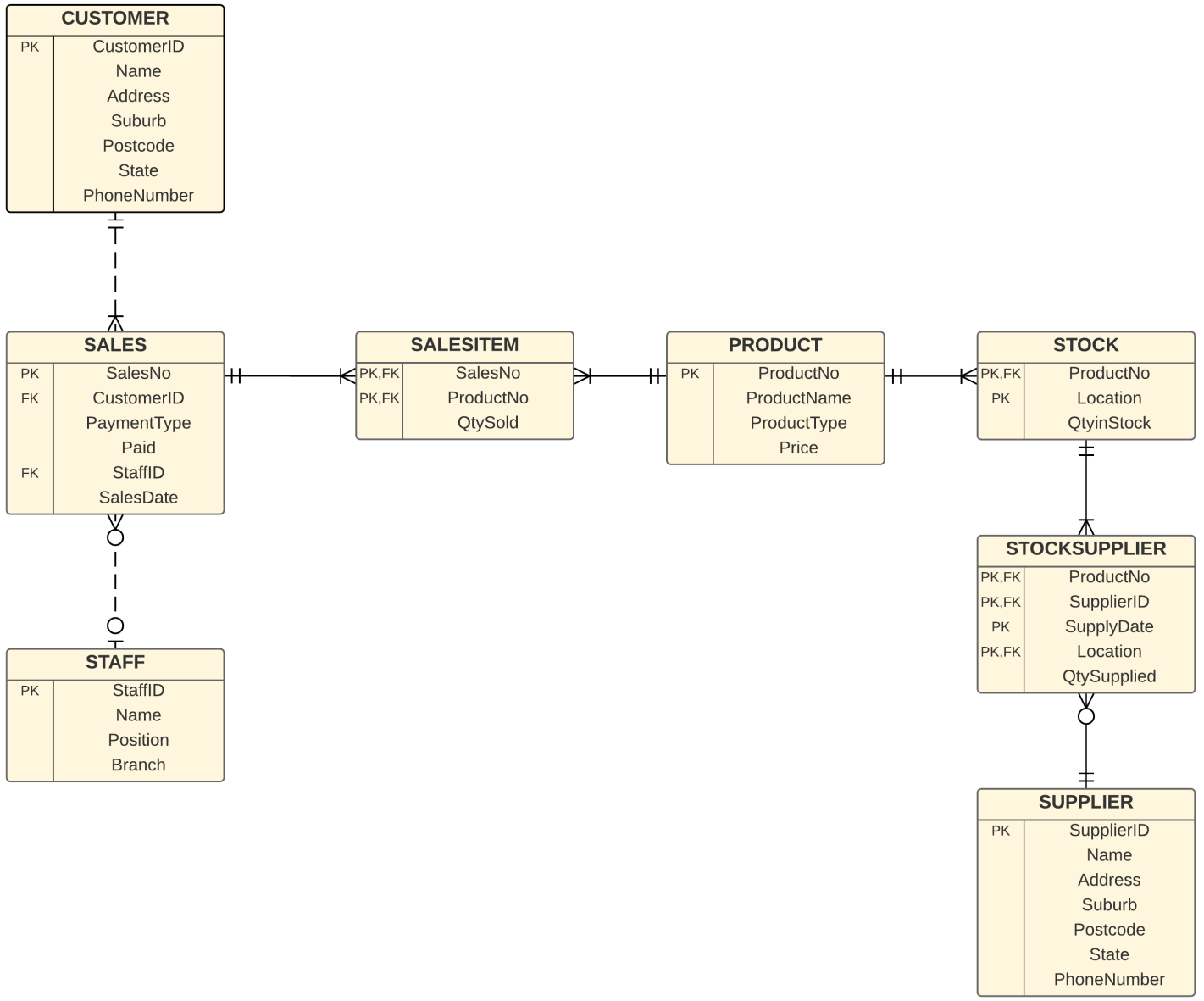
# Case Study #1 – A Product Sales Case Study

| TimeID | Suburb | ProductNo | SupplierID | TotalSales |
|---|---|---|---|---|
| 201801 | Caulfield | A1 | S1 | ... |
| 201801 | Caulfield | A1 | S2 | ... |
| 201801 | Caulfield | A1 | S3 | ... |
| 201801 | Caulfield | A1 | ... | ... |
| 201801 | Caulfield | B2 | S1 | ... |
| 201801 | Caulfield | B2 | S2 | ... |
| 201801 | Caulfield | B2 | S3 | ... |
| 201801 | Caulfield | B2 | ... | ... |
| 201801 | Caulfield | C3 | S1 | ... |
| 201801 | Caulfield | C3 | S2 | ... |
| 201801 | Caulfield | C3 | S3 | ... |
| 201801 | Caulfield | C3 | ... | ... |
| 201801 | ... | ... | ... | ... |
| 201801 | Chadstone | A1 | S1 | ... |
| 201801 | Chadstone | A1 | S2 | ... |
| 201801 | Chadstone | A1 | S3 | ... |
| 201801 | Chadstone | A1 | ... | ... |
| 201801 | ... | ... | ... | ... |
| 201802 | Caulfield | A1 | S1 | ... |
| 201802 | Caulfield | A1 | S2 | ... |
| 201802 | Caulfield | A1 | S3 | ... |
| 201802 | Caulfield | A1 | ... | ... |
| ... | ... | ... | ... | |



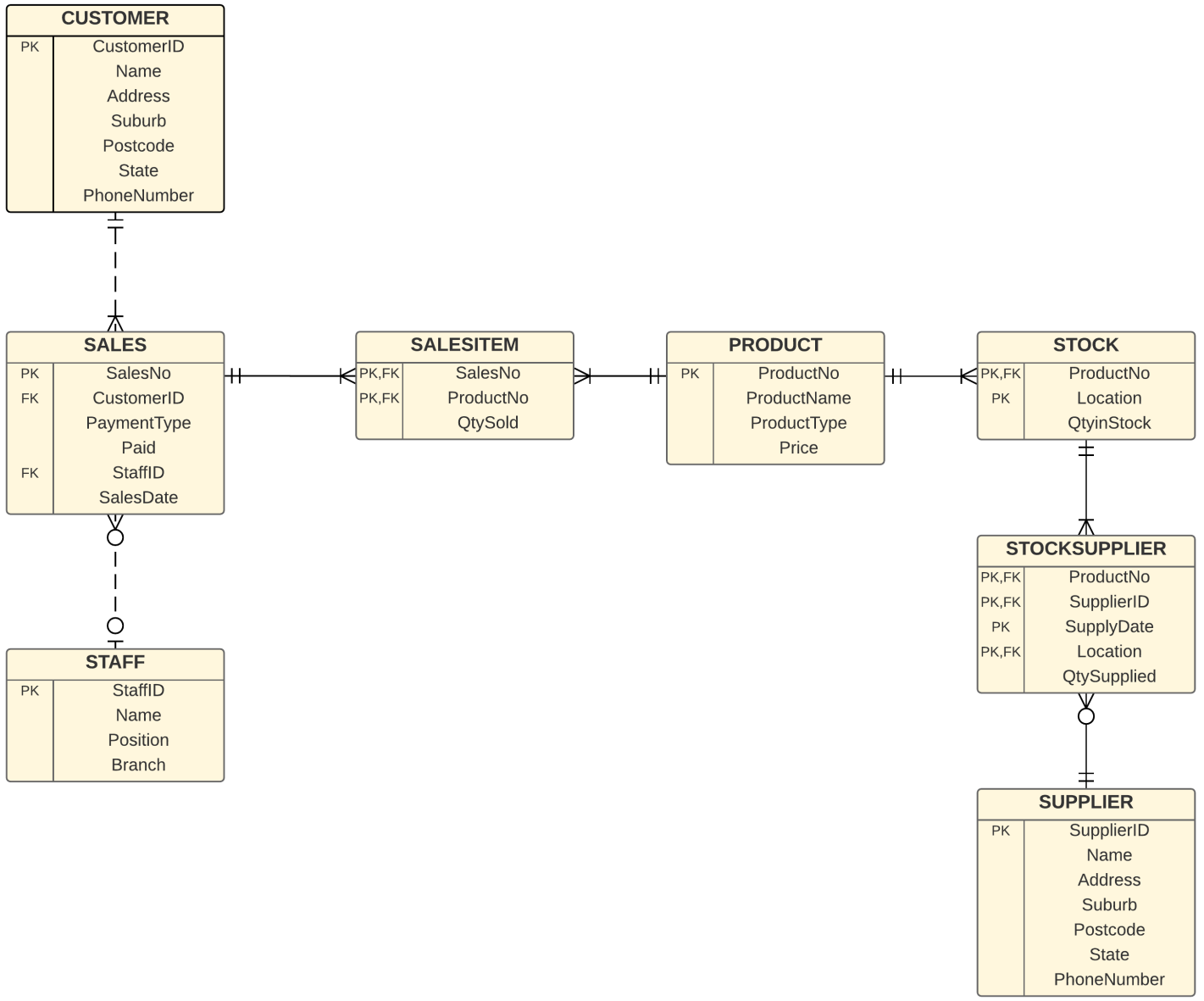
# Case Study #1 – A Product Sales Case Study

| TimeID | Suburb | ProductNo | SupplierID | TotalSales |
|---|---|---|---|---|
| 201801 | Caulfield | A1 | S1 | ... |
| 201801 | Caulfield | A1 | S2 | ... |
| 201801 | Caulfield | A1 | S3 | ... |
| 201801 | Caulfield | A1 | ... | ... |
| 201801 | Caulfield | B2 | S1 | ... |
| 201801 | Caulfield | B2 | S2 | ... |
| 201801 | Caulfield | B2 | S3 | ... |
| 201801 | Caulfield | B2 | ... | ... |
| 201801 | Caulfield | C3 | S1 | ... |
| 201801 | Caulfield | C3 | S2 | ... |
| 201801 | Caulfield | C3 | S3 | ... |
| 201801 | Caulfield | C3 | ... | ... |
| 201801 | ... | ... | ... | ... |
| 201801 | Chadstone | A1 | S1 | ... |
| 201801 | Chadstone | A1 | S2 | ... |
| 201801 | Chadstone | A1 | S3 | ... |
| 201801 | Chadstone | A1 | ... | ... |
| 201801 | ... | ... | ... | ... |
| 201802 | Caulfield | A1 | S1 | ... |
| 201802 | Caulfield | A1 | S2 | ... |
| 201802 | Caulfield | A1 | S3 | ... |
| 201802 | Caulfield | A1 | ... | ... |
| ... | ... | ... | ... | |



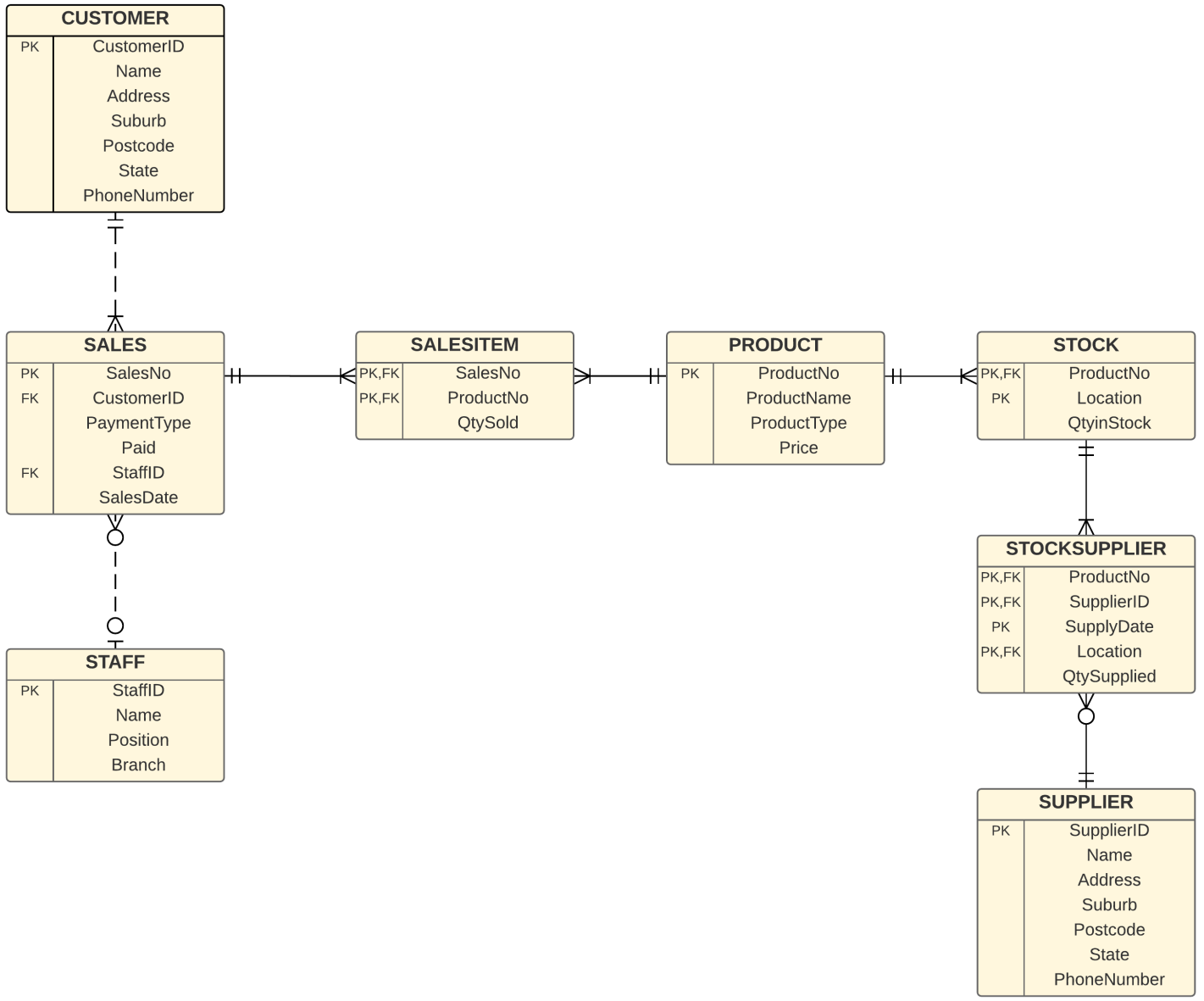
# Case Study #1 – A Product Sales Case Study

| SupplierID | TotalSales |
|---|---|
| S1 | $77,000 |
| S2 | $5,700 |
| S3 | $12,500 |
| … | … |

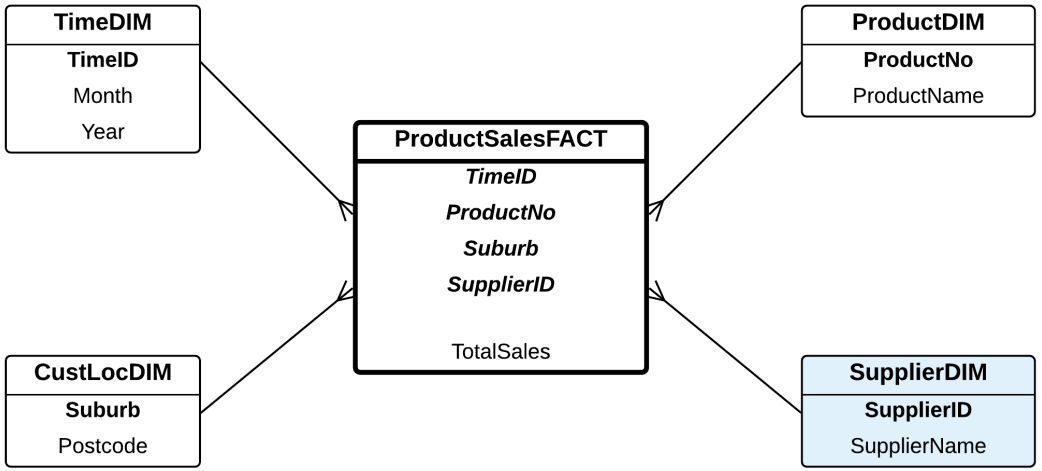


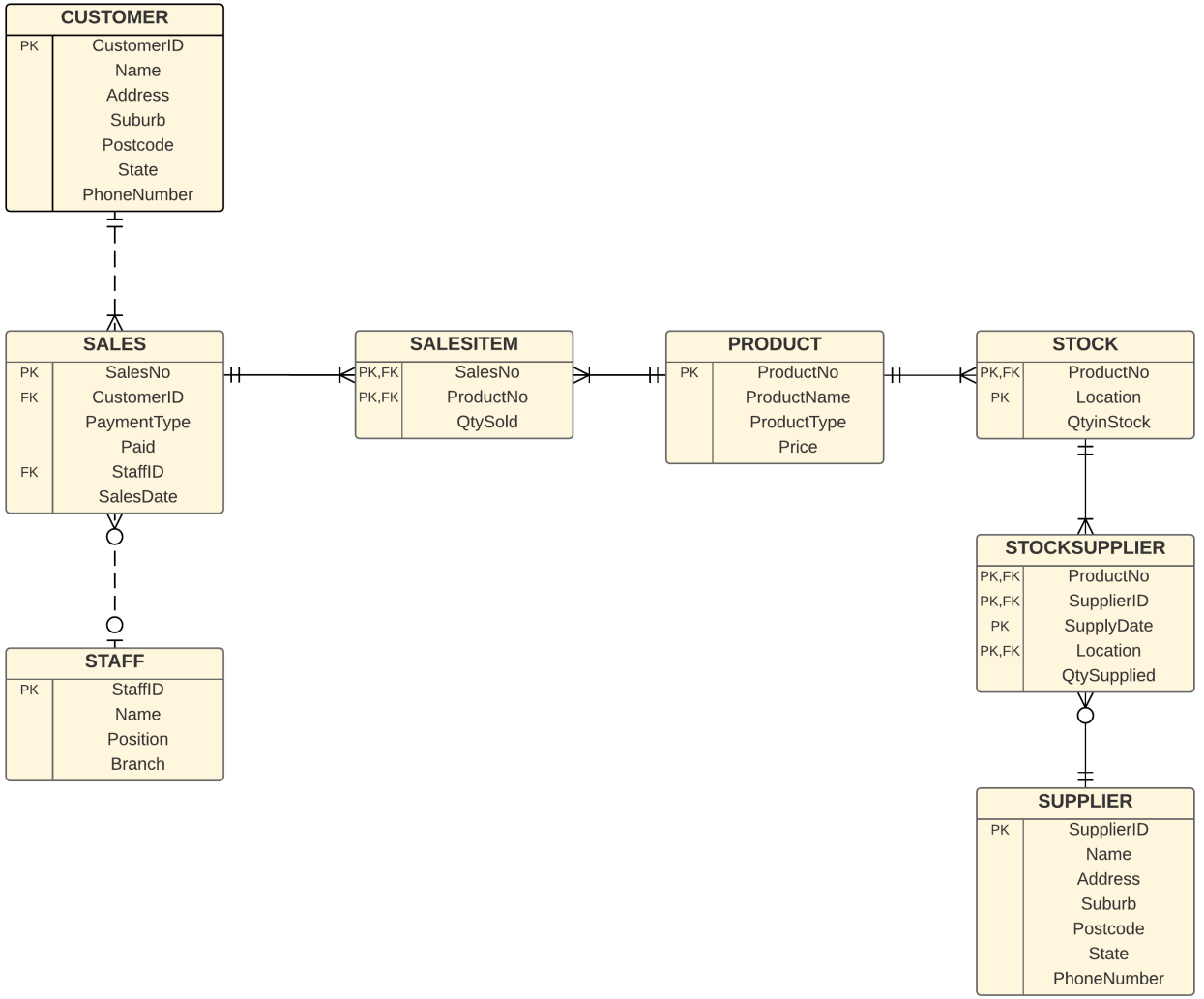
# Case Study #1 – A Product Sales Case Study

| SupplierID | TotalSales |
|---|---|
| S1 | $77,000 |
| S2 | $5,700 |
| S3 | $12,500 |
| … | … |

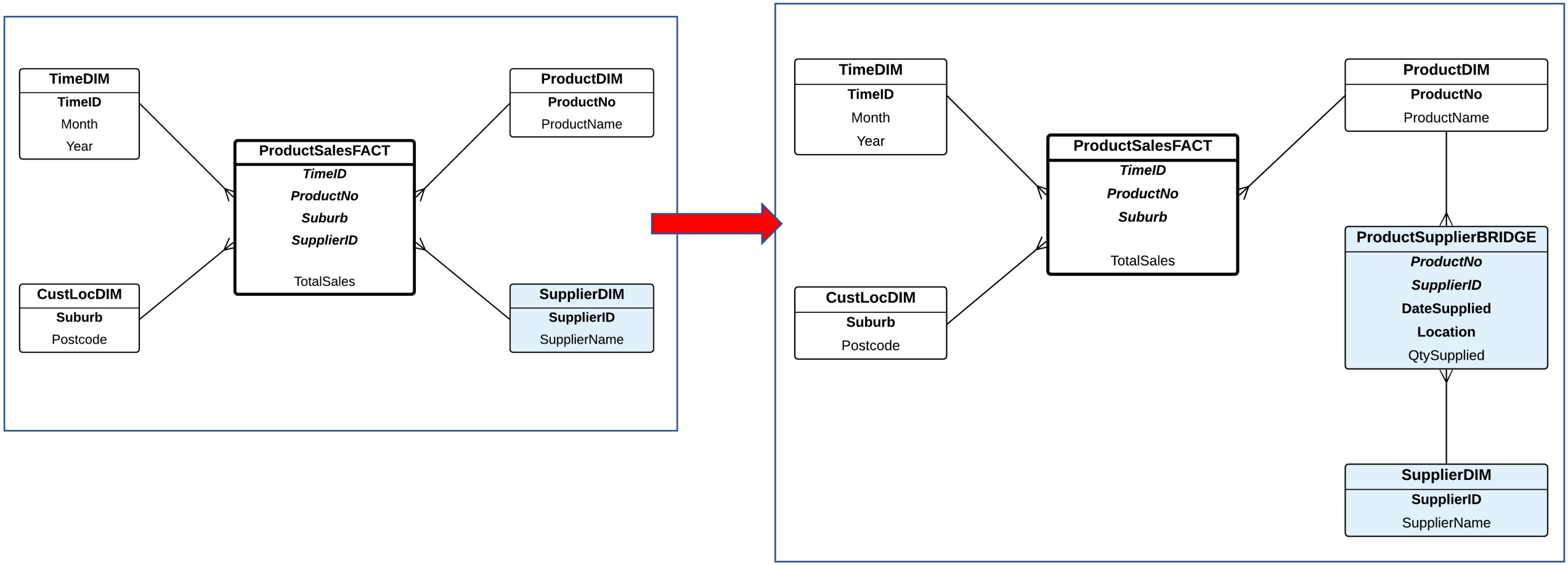


# Case Study #1 – A Product Sales Case Study

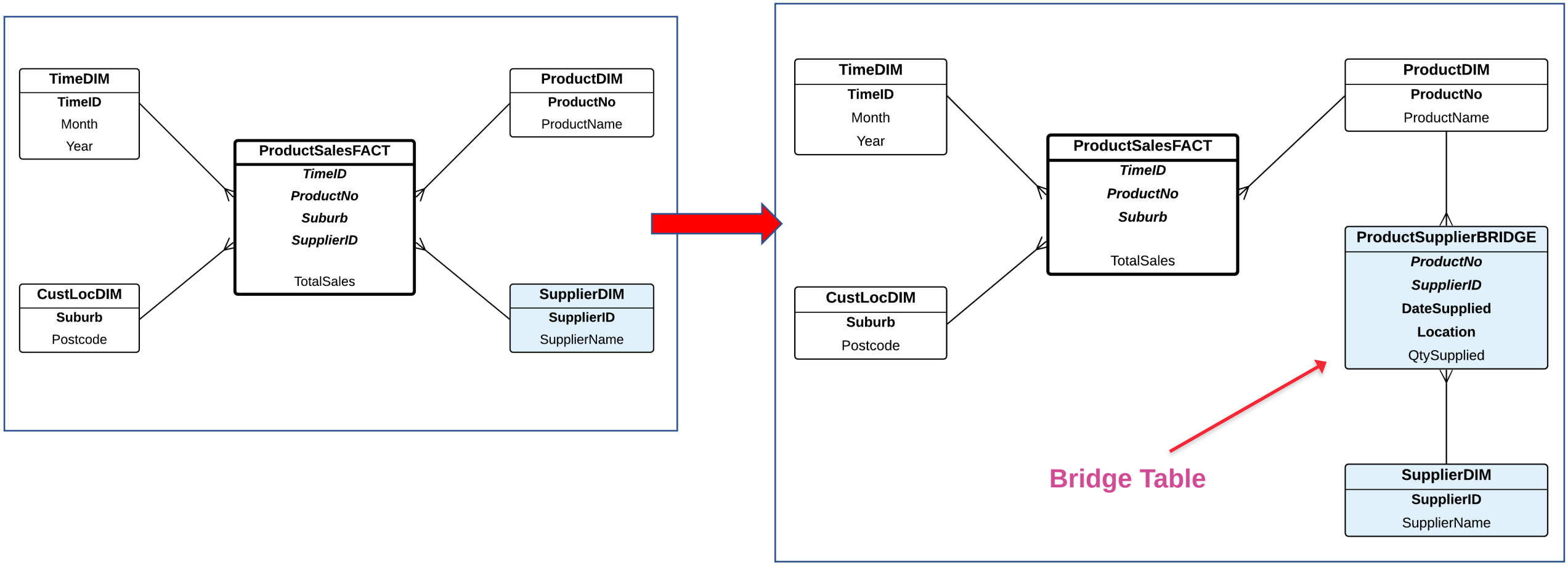


# Case Study #1 – A Product Sales Case Study



# Case Study #1 – A Product Sales Case Study



# Case Study #1 – A Product Sales Case Study

- To create Time Dimension:
  - 
```
create table TimeDim as
select
   distinct to_char(SalesDate, 'YYYYMM') as TimeID,
   to_char(SalesDate, 'YYYY') as Year,
   to_char(SalesDate, 'MM') as Month
from Sales;
```

- To create Customer Location Dimension:
  - 
```
create table CustLocDim as
select distinct Suburb, Postcode
from Customer;
```

# Case Study #1 – A Product Sales Case Study

- To create Product Dimension:
  - 
```
create table ProductDim as
select distinct ProductNo, ProductName
from Product;
```

- To create **Bridge Table**:
  - 
```
create table ProductSupplierBridge as
select *
from StockSupplier;
```

- To create Supplier Dimension:
  - 
```
create table SupplierDim as
select SupplierID, Name as SupplierName
from Supplier;
```

# Case Study #1 – A Product Sales Case Study

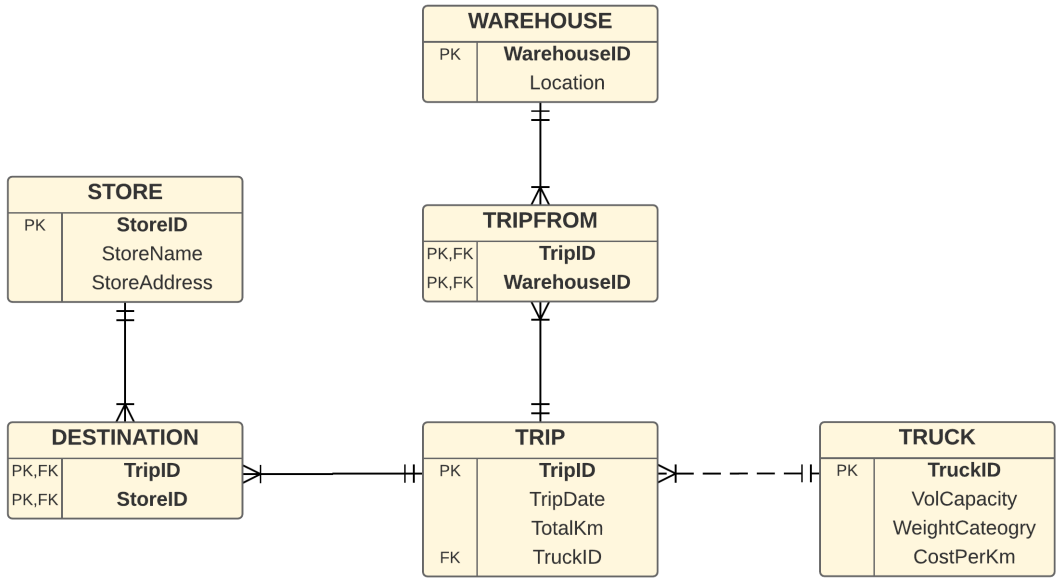
- To create Fact Table:
  - 
```
create table ProductSalesFact as
Select
     to_char(S.SalesDate, 'YYYYMM') as TimeID,
     P.ProductNo,
     C.Suburb,
     sum(SI.QtySold*P.Price) as TotalSales
from Sales S, Product P, Customer C, SalesItem SI
where S.SalesNo = SI.SalesNo
and SI.ProductNo= P.ProductNo
and C.CustomerID = S.CustomerID
group by
     to_char(S.SalesDate, 'YYYYMM'), P.ProductNo, C.Suburb;
```

# Bridge Tables
# Case Study #2

MONASH University

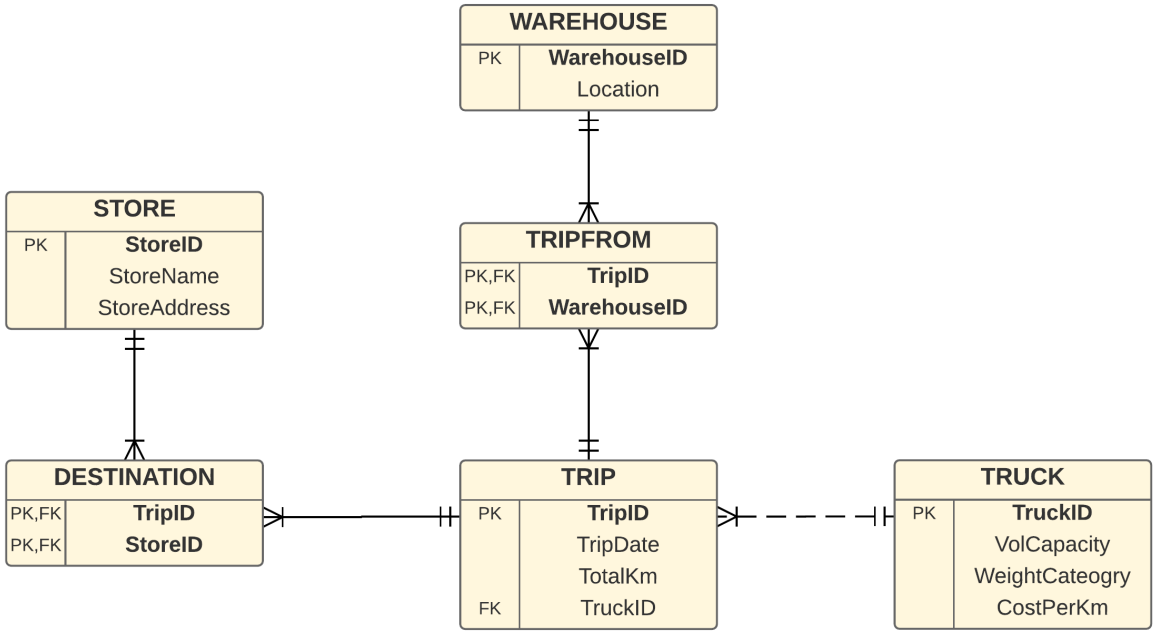
# Case Study #2 – A Truck Delivery Case Study

- A trucking company is responsible for picking up goods from warehouses of a retail chain company, and delivering the goods to individual retail stores.
- A truck carry goods during a single trip, which is identified by TripID, and delivers these goods to multiple stores. Trucks have different capacities for both the volumes they can hold and the weights they can carry.
- At the moment, a truck makes several trips each week. An operational database is being used to keep track the deliveries, including the scheduling of trucks, which provide timely deliveries to stores.



# Case Study #2 – A Truck Delivery Case Study

- A trip may pick up goods from many warehouses
  - i.e. a many-many relationship between Warehouse and Trip
- A trip uses one truck only, and a truck may have many trips in the history
  - i.e. a many-1 relationship between Trip and Truck
- A trip delivers goods (e.g. TVs, fridges, etc) potentially to several stores
  - a many-many relationship between Trip and Store, which is represented by the Destination table



# Case Study #2 – A Truck Delivery Case Study

- Sample data in the operational database:

(a) Warehouse Table

| WarehouseID | Location |
|---|---|
| W1 | Warehouse1 |
| W2 | Warehouse1 |
| W3 | Warehouse1 |
| … | … |

(b) Trip Table

| TripID | Date | TotalKm | TruckID |
|---|---|---|---|
| Trip1 | 14-Apr-2018 | 370 | Truck1 |
| Trip2 | 14-Apr-2018 | 570 | Truck2 |
| Trip3 | 14-Apr-2018 | 250 | Truck3 |
| Trip4 | 15-Jul-2018 | 450 | Truck1 |
| … | … | … | … |

(c) TripFrom Table

| TripID | WarehouseID |
|---|---|
| Trip1 | W1 |
| Trip1 | W2 |
| Trip1 | W3 |
| Trip2 | W1 |
| Trip2 | W2 |
| … | … |

(d) Truck Table

| TruckID | VolCapacity | WeightCategory | CostPerKm |
|---|---|---|---|
| Truck1 | 250 | Medium | $1.20 |
| Truck2 | 300 | Medium | $1.50 |
| Truck3 | 100 | Small | $0.80 |
| Truck4 | 550 | Large | $2.30 |
| Truck5 | 650 | Large | $2.50 |
| … | … | … | … |

(e) Store Table

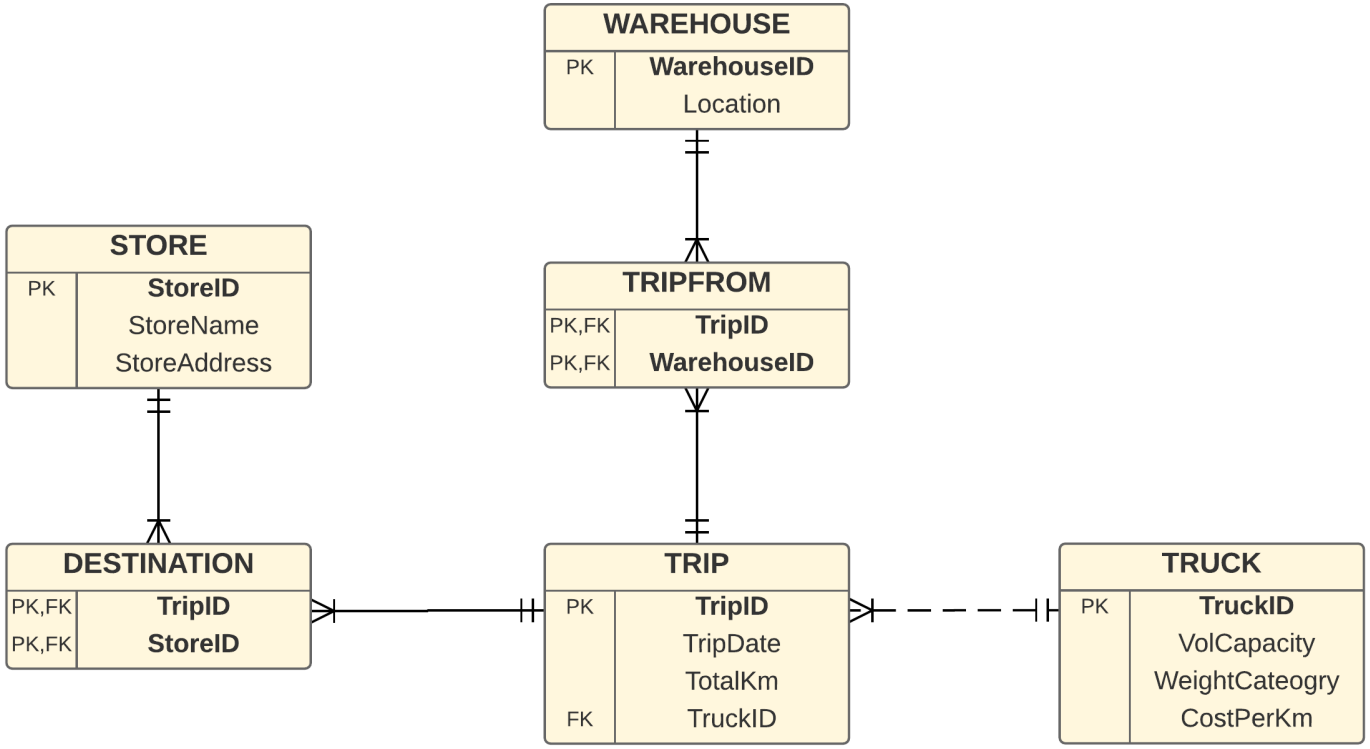
| StoreID | StoreName | Address |
|---|---|---|
| M1 | MyStore City | Melbourne |
| M2 | MyStore Chaddy | Chadstone |
| M3 | MyStore HiPoint | High Point |
| M4 | MyStore Donc | Doncaster |
| M5 | MyStore North | Northland |
| M6 | MyStore South | Southland |
| M7 | MyStore East | Eastland |
| M8 | MyStore Knox | Knox |
| … | … | |

(f) Destination Table

| TripID | StoreID |
|---|---|
| Trip1 | M1 |
| Trip1 | M2 |
| Trip1 | M4 |
| Trip1 | M3 |
| Trip1 | M8 |
| Trip2 | M4 |
| Trip2 | M1 |
| Trip2 | M2 |
| … | … |

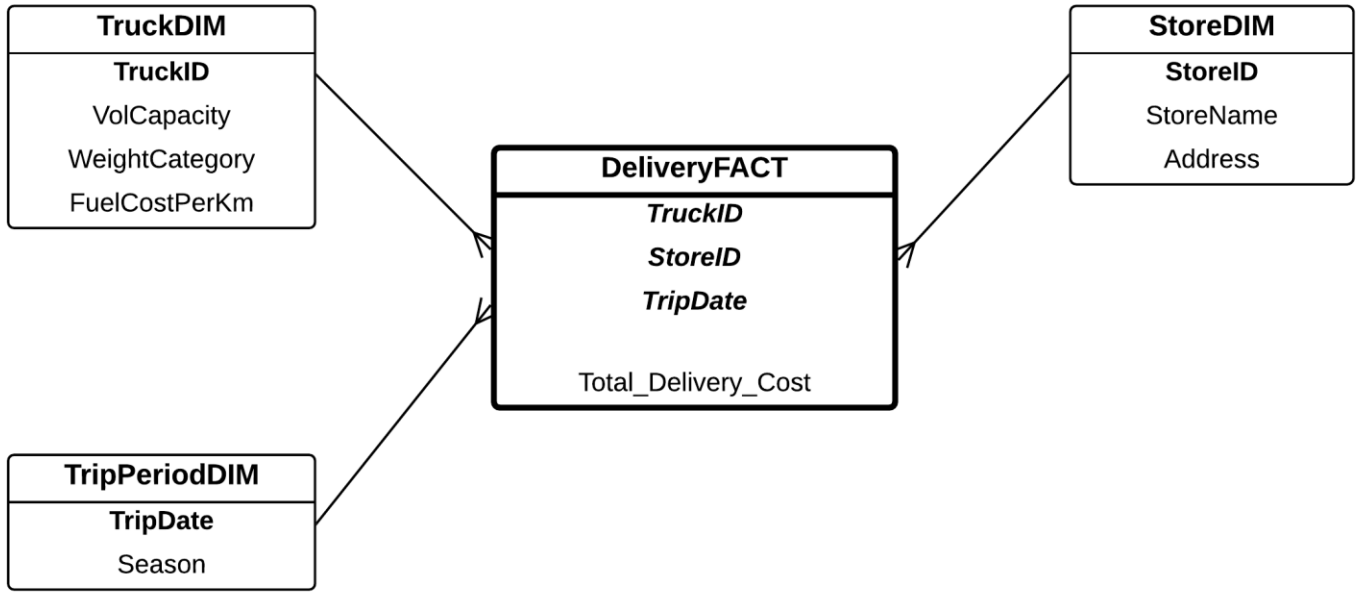
# Case Study #2 – A Truck Delivery Case Study

- The management of this trucking company would like to analyze the ***deliver cost***, based on ***trucks***, ***time period***, and ***store***.



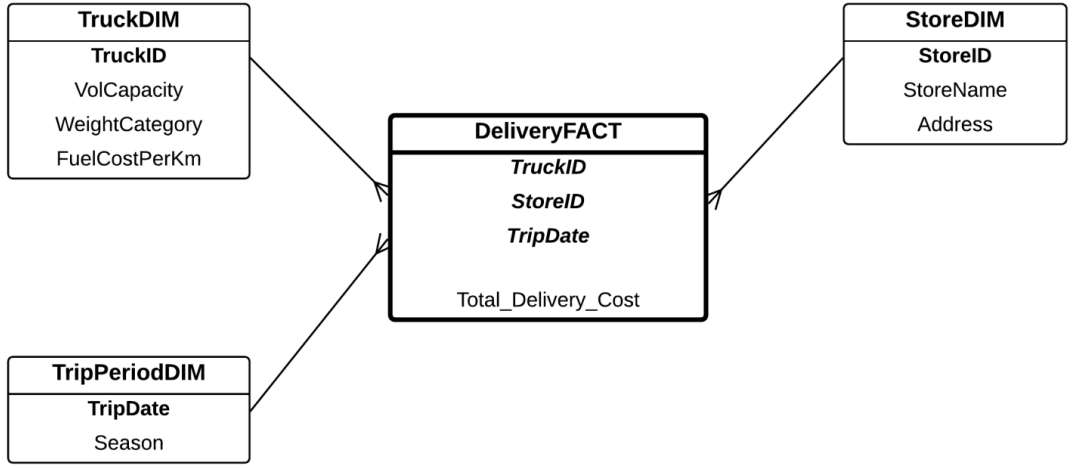
# Case Study #2 – A Truck Delivery Case Study

- Sales Star Schema
  - **Fact**:
    - Total Delivery Cost
      (distance * cost per kilometre)
  - **Dimensions**:
    - Truck
    - Time period
    - Store



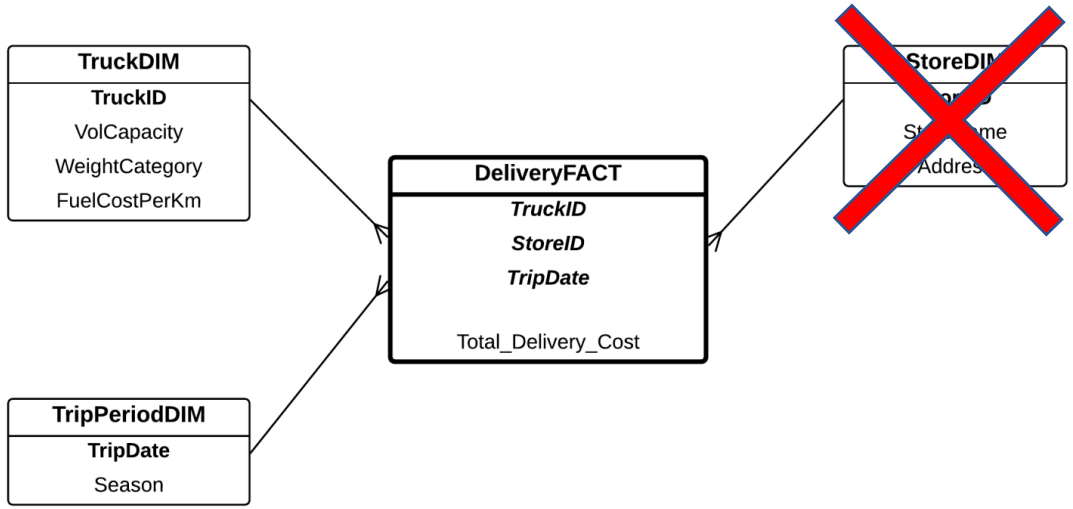
# Case Study #2 – A Truck Delivery Case Study

- From the **Truck** point of view, Truck1 has two trips (e.g. Trip1 and Trip4), with the total kilometres of 820km (370km + 450km). The cost for Truck1 is $1.20. Hence, calculating the cost for Truck1 is straightforward. Other trucks can be calculated this way.
- From the **Period** point of view, 14-Apr-2018 has three trips (e.g. Trip1,Trip2, and Trip3). Trip1 (370km) is delivered by Truck1 which costs $1.20/km. Trip2 and Trip 3, on the same day, can be calculated the same way. Hence, on 14-Apr-2018, the total cost can be calculated.
- From the **Store** point of view; The cost is calculated based on Trip, but a trip delivers goods to many stores. Therefore, the delivery cost for each store cannot be calculated. The delivery cost is for the trip – not for the store.

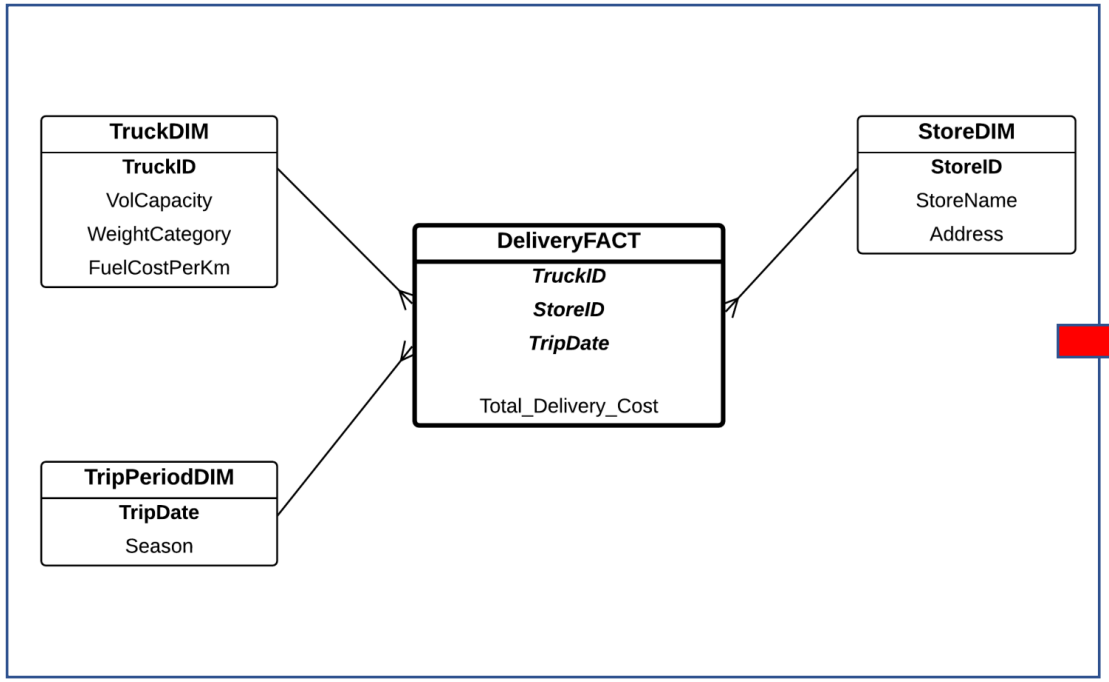


# Case Study #2 – A Truck Delivery Case Study

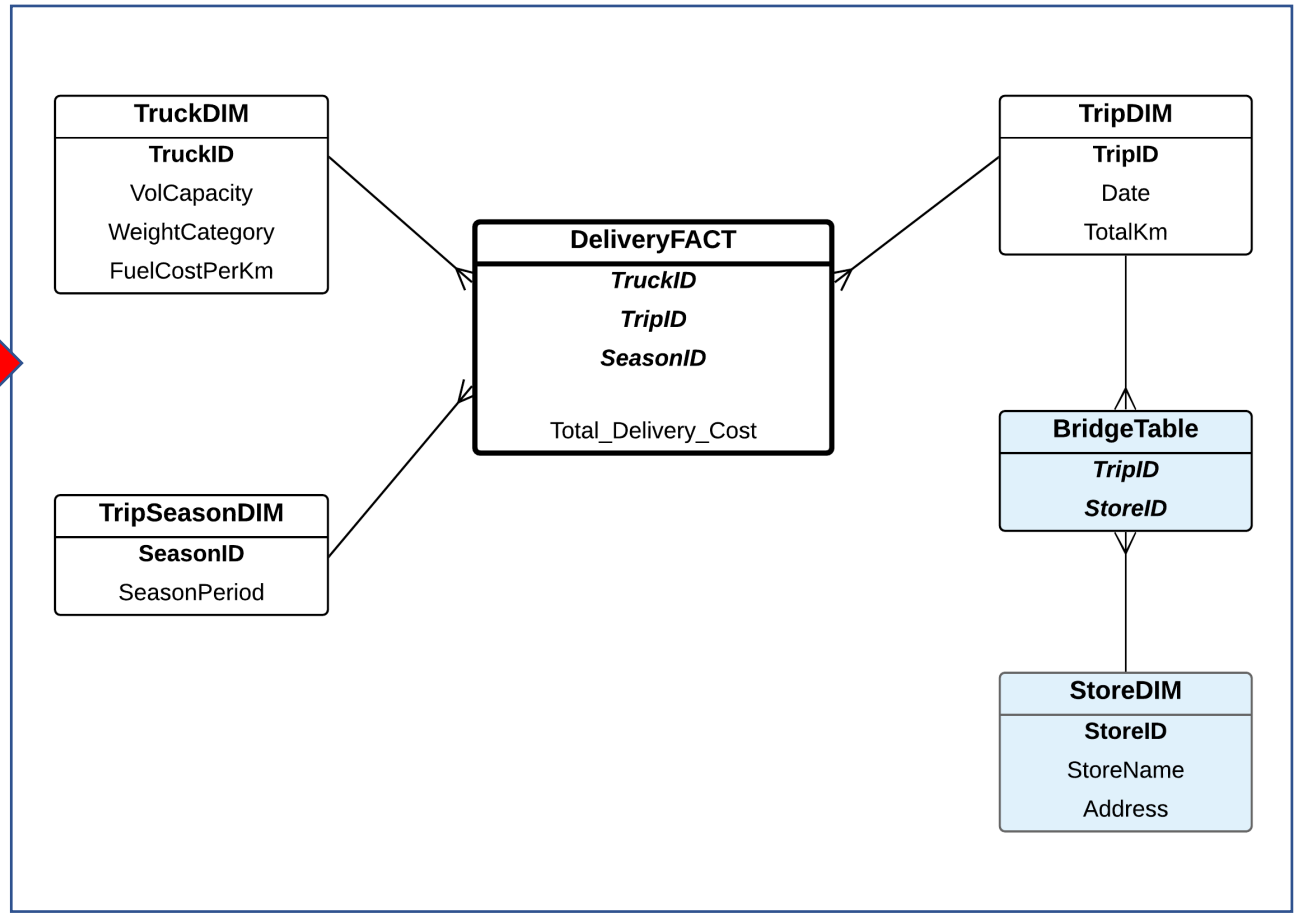
- From the **Truck** point of view, Truck1 has two trips (e.g. Trip1 and Trip4), with the total kilometres of 820km (370km + 450km). The cost for Truck1 is $1.20. Hence, calculating the cost for Truck1 is straightforward. Other trucks can be calculated this way.
- From the **Period** point of view, 14-Apr-2018 has three trips (e.g. Trip1,Trip2, and Trip3). Trip1 (370km) is delivered by Truck1 which costs $1.20/km. Trip2 and Trip 3, on the same day, can be calculated the same way. Hence, on 14-Apr-2018, the total cost can be calculated.
- From the **Store** point of view; The cost is calculated based on Trip, but a trip delivers goods to many stores. Therefore, the delivery cost for each store cannot be calculated. The delivery cost is for the trip – not for the store.



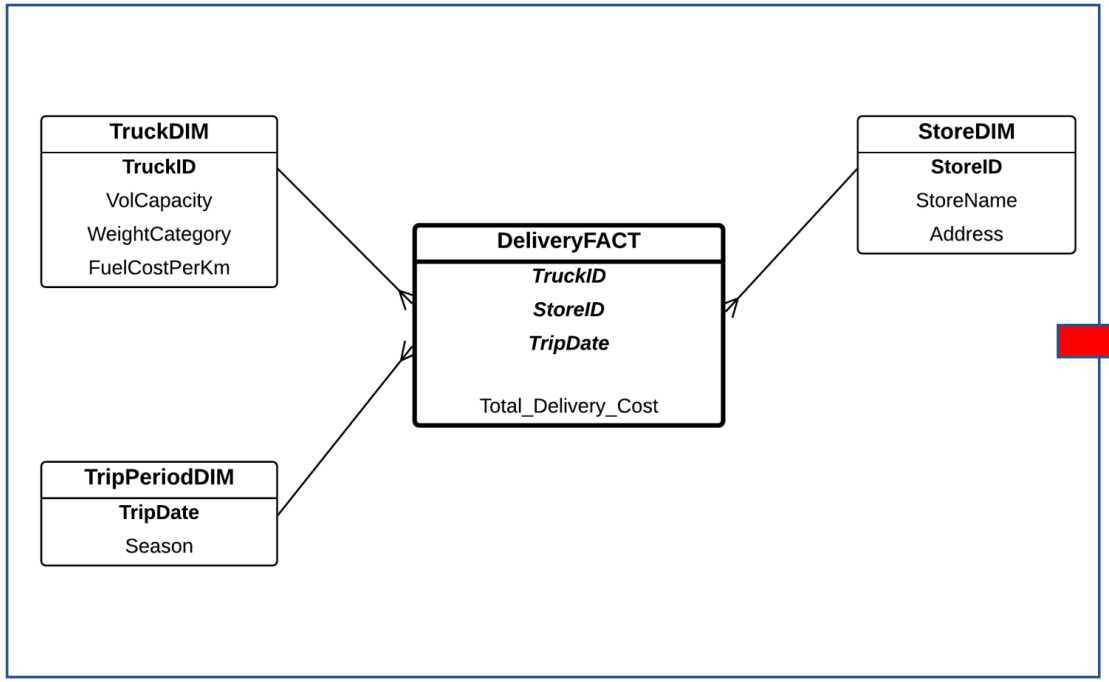
# Case Study #2 – A Truck Delivery Case Study



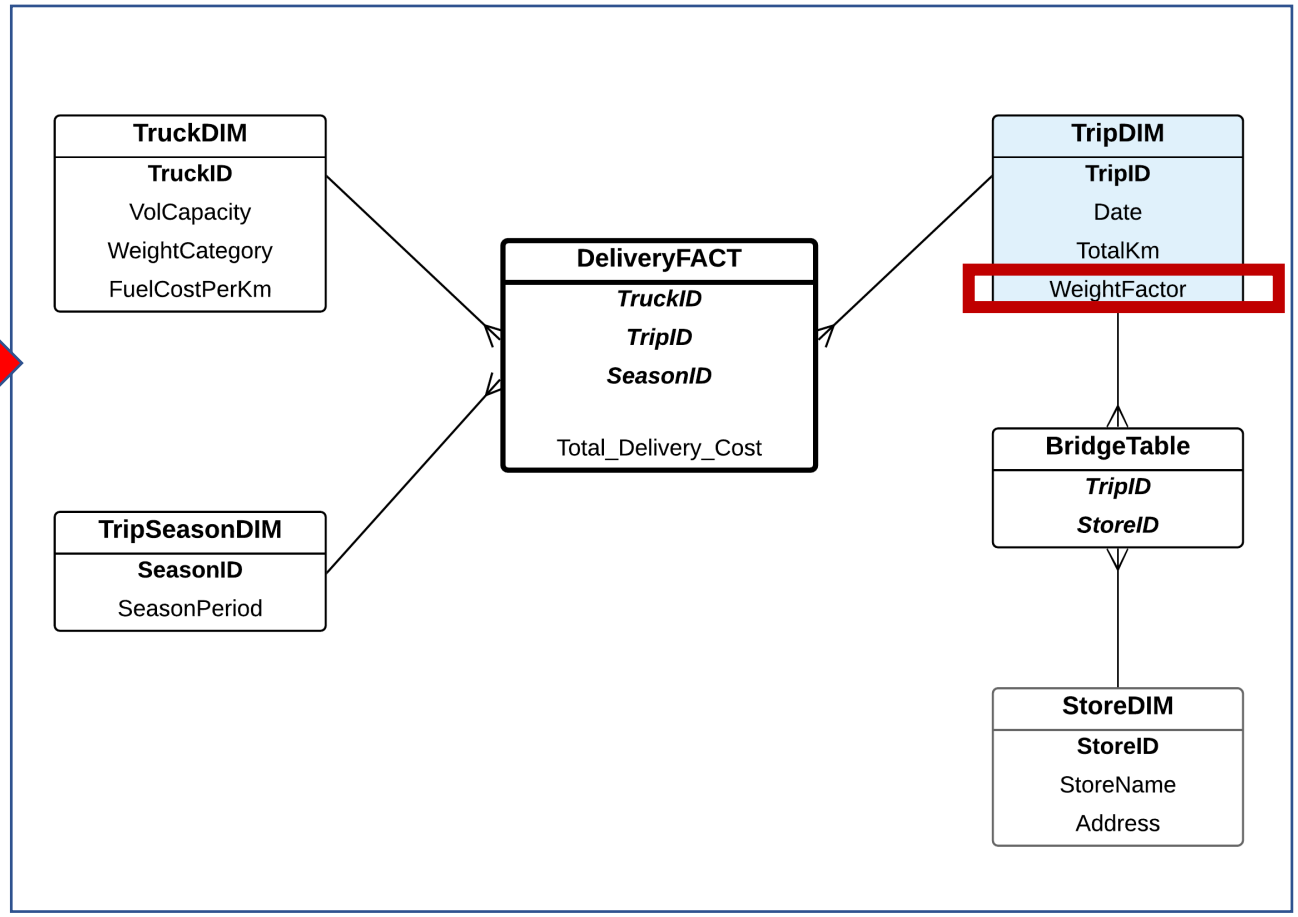

## Solution Model 1 – Using a Bridge Table



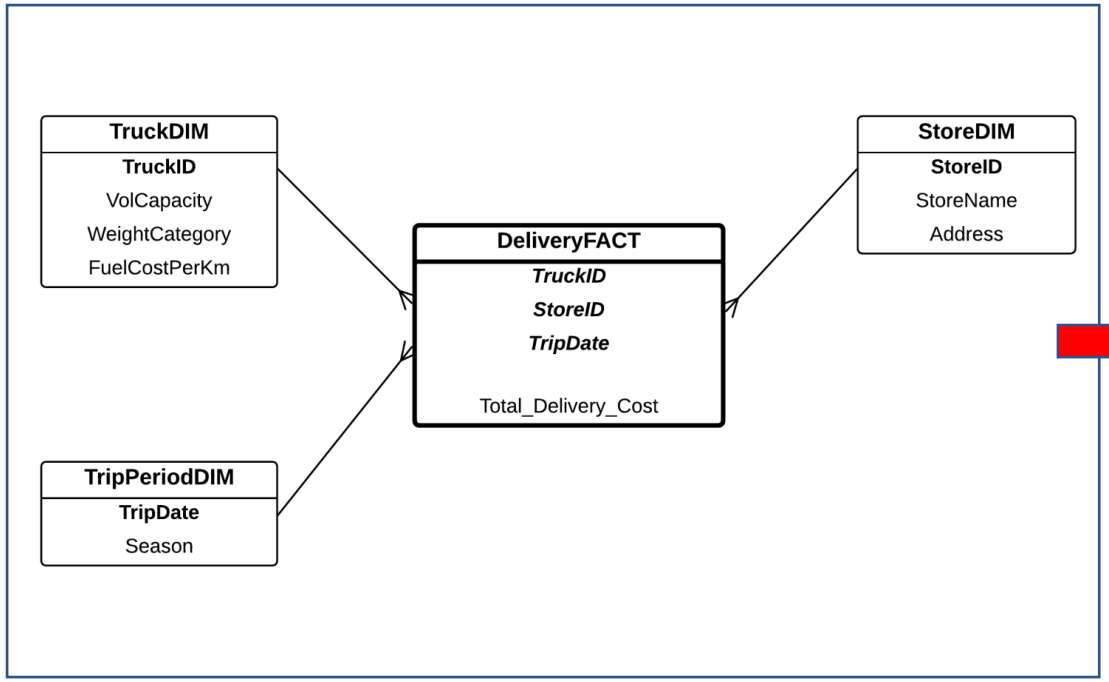
# Case Study #2 – A Truck Delivery Case Study



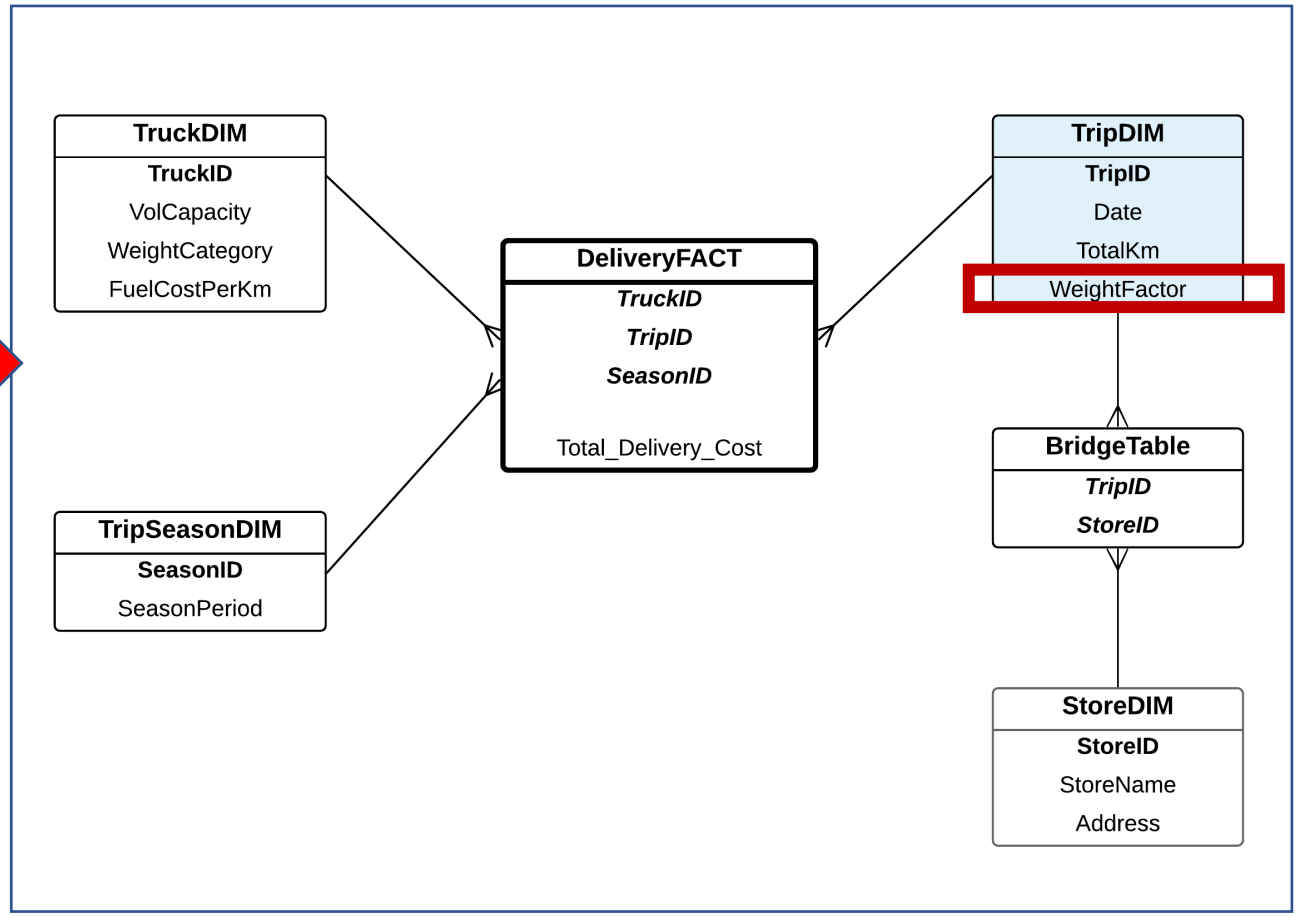

## Solution Model 2 – add a Weight Factor attribute



# Case Study #2 – A Truck Delivery Case Study




## Solution Model 2 – add a Weight Factor attribute




A **weight factor** is only needed if we want to estimate the contribution that a dimension made to the fact

# Case Study #2 – A Truck Delivery Case Study

(a) Trip Dimension Table

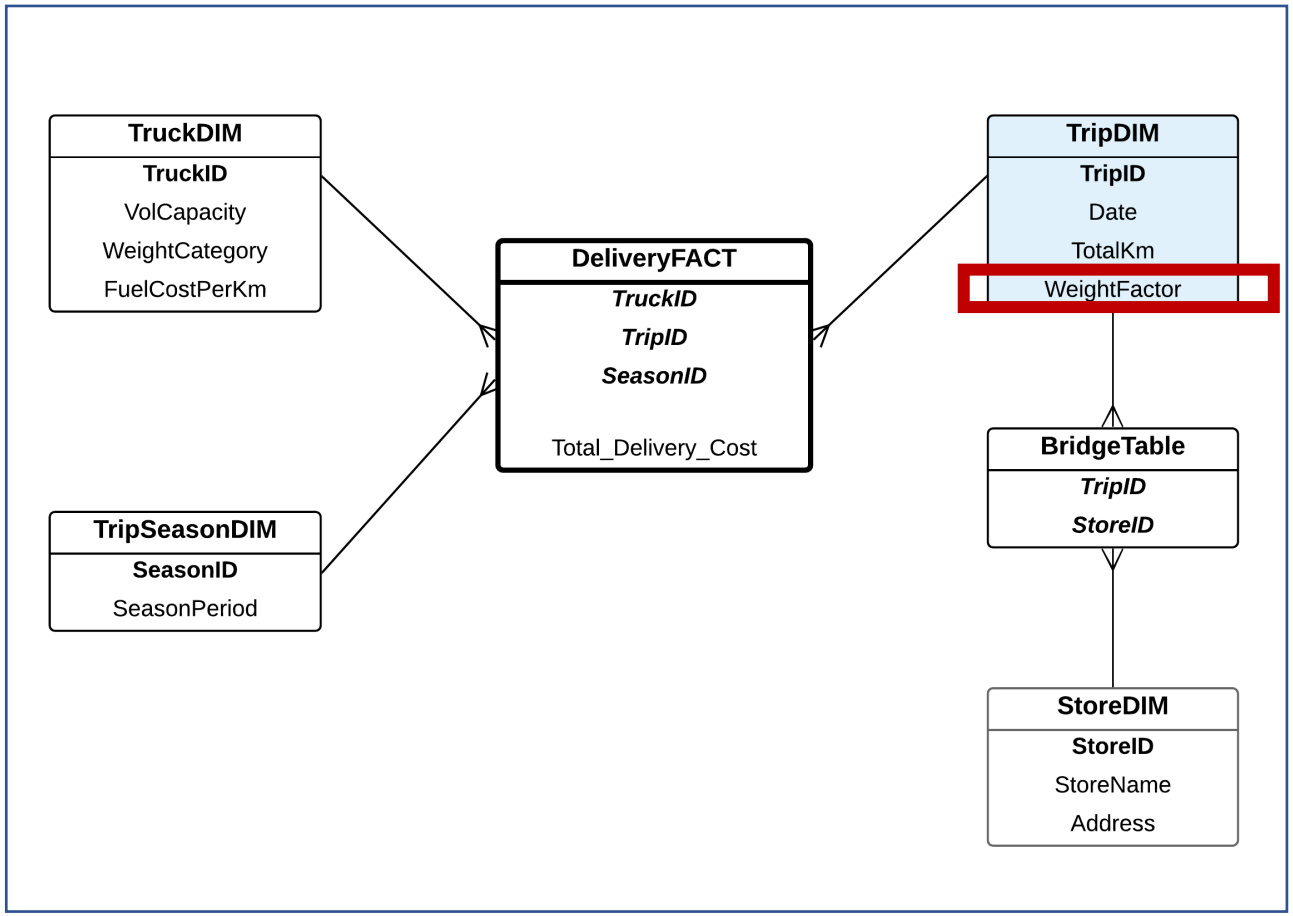
| TripID | Date | TotalKm | WeightFactor |
|---|---|---|---|
| Trip1 | 14-Apr-2018 | 370 | 0.20 |
| Trip2 | 14-Apr-2018 | 570 | 0.33 |
| ... | ... | ... | ... |

(b) Bridge Table

| TripID | StoreID |
|---|---|
| Trip1 | M1 |
| Trip1 | M2 |
| Trip1 | M4 |
| Trip1 | M3 |
| Trip1 | M8 |
| Trip2 | M4 |
| Trip2 | M1 |
| Trip2 | M2 |
| … | … |

(c) Store Table

| StoreID | StoreName | Address |
|---|---|---|
| M1 | MyStore City | Melbourne |
| M2 | MyStore Chaddy | Chadstone |
| M3 | MyStore HiPoint | High Point |
| M4 | MyStore Donc | Doncaster |
| M5 | MyStore North | Northland |
| M6 | MyStore South | Southland |
| M7 | MyStore East | Eastland |
| M8 | MyStore Knox | Knox |
| … | … | |

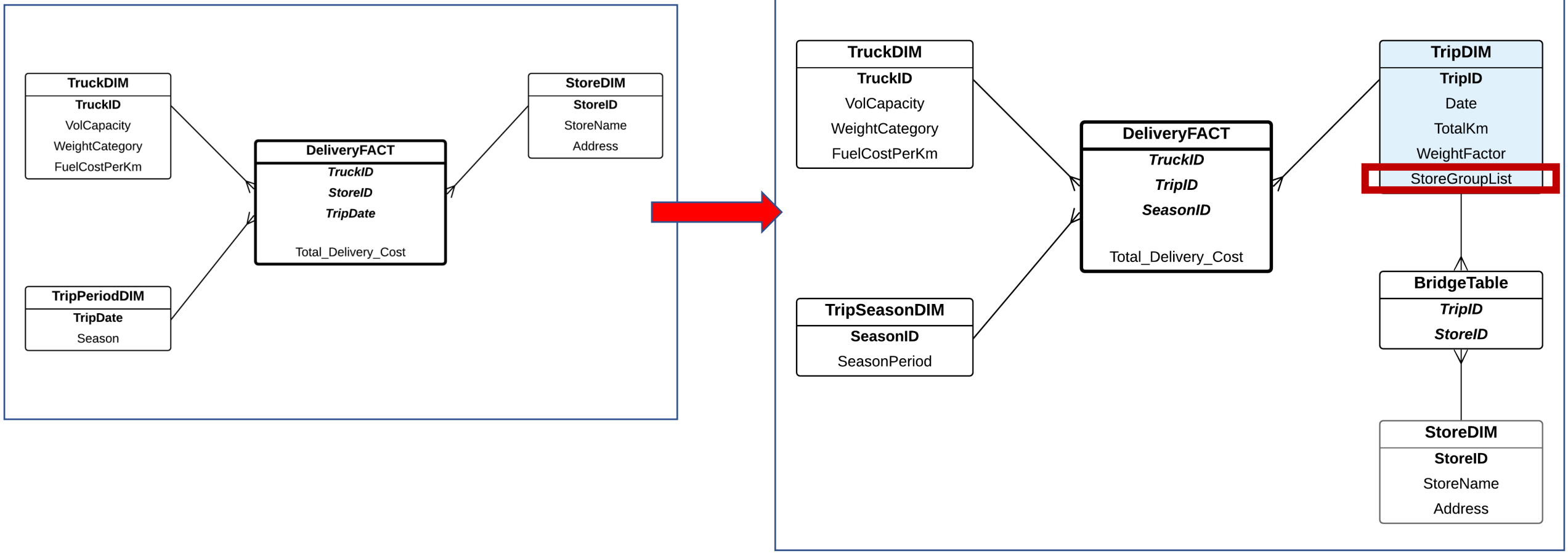


# Case Study #2 – A Truck Delivery Case Study

- To create Trip Dimension:
  - 
```
create table TripDim2 as
select T.TripID, T.TripDate, T.TotalKm,
    1.0/count(*) as WeightFactor
from Trip T, Destination D
where T.TripID = D.TripID
group by T.tripid, T.tripdate, T.totalkm;
```

# Case Study #2 – A Truck Delivery Case Study

## Solution Model 3 – a List Aggregate version



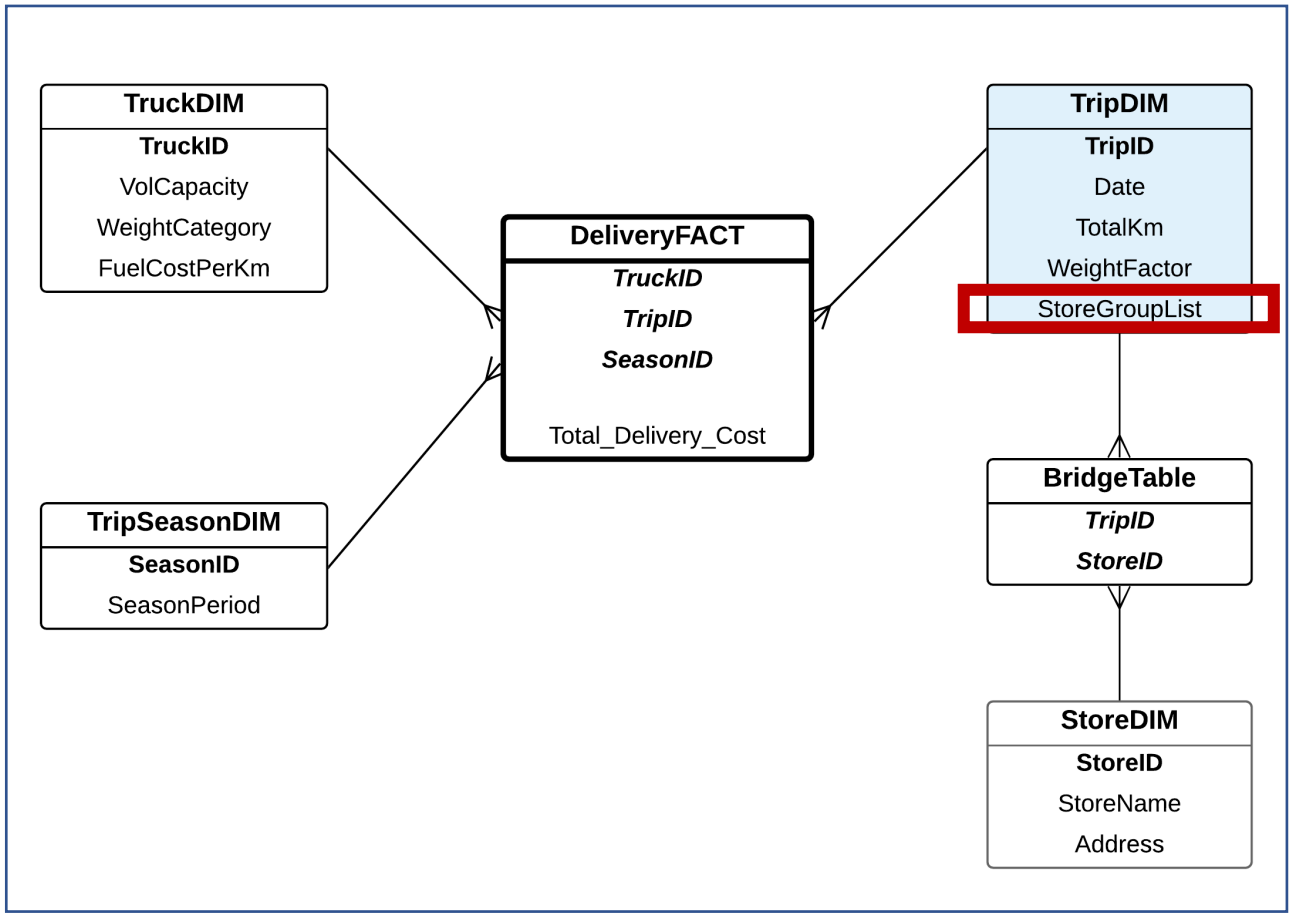
# Case Study #2 – A Truck Delivery Case Study

(a) Trip Dimension Table

| TripID | Date | TotalKm | WeightFactor | StoreGroupList |
|---|---|---|---|---|
| Trip1 | 14-Apr-2018 | 370 | 0.20 | M1_M2_M3_M4_M8 |
| Trip2 | 14-Apr-2018 | 570 | 0.33 | M1_M2_M4 |
| ... | ... | ... | ... | ... |

(b) Bridge Table

| TripID | StoreID |
|---|---|
| Trip1 | M1 |
| Trip1 | M2 |
| Trip1 | M4 |
| Trip1 | M3 |
| Trip1 | M8 |
| Trip2 | M4 |
| Trip2 | M1 |
| Trip2 | M2 |
| … | … |



# Case Study #2 – A Truck Delivery Case Study

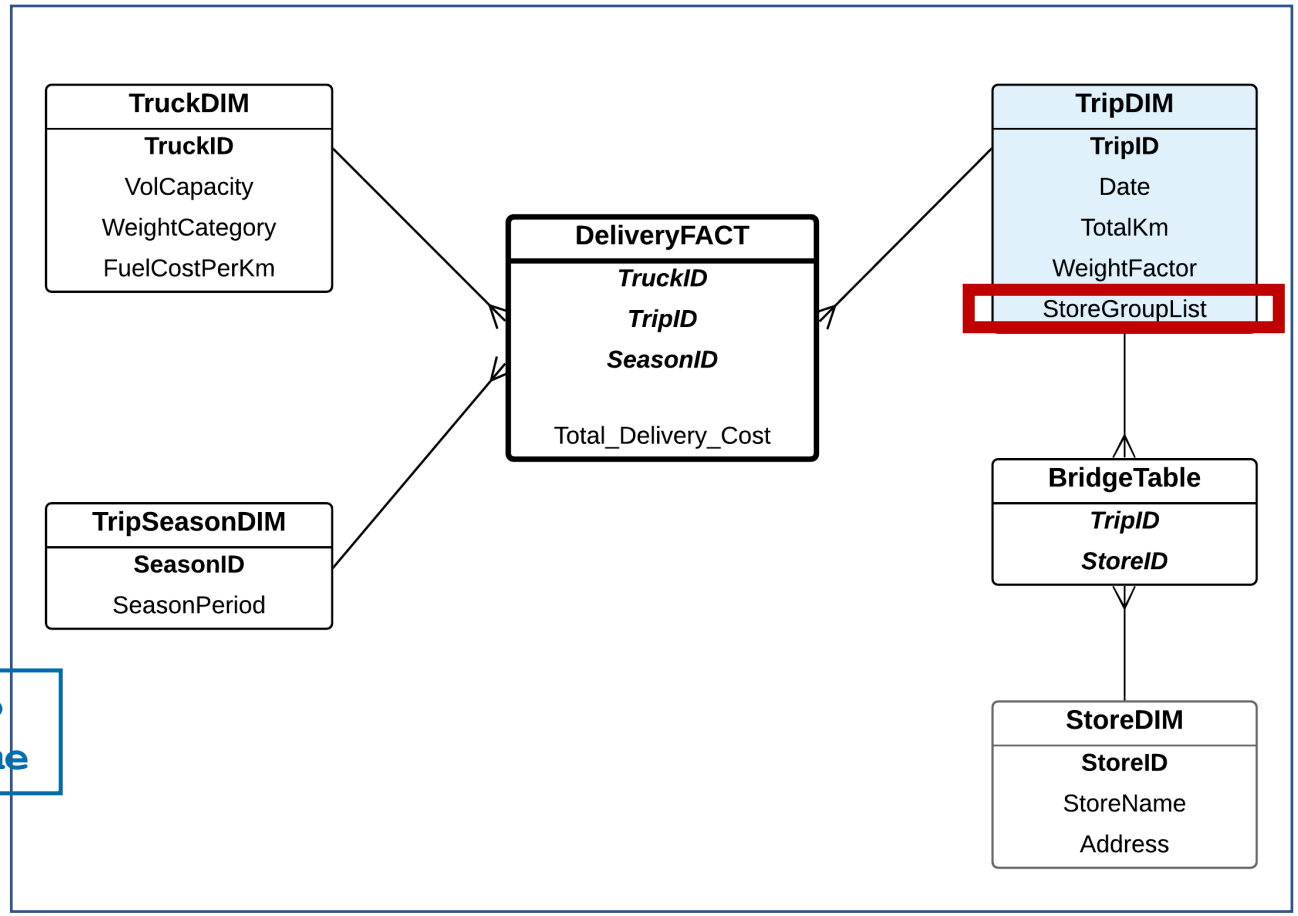
(a) Trip Dimension Table

| TripID | Date | TotalKm | WeightFactor | StoreGroupList |
|---|---|---|---|---|
| Trip1 | 14-Apr-2018 | 370 | 0.20 | M1_M2_M3_M4_M8 |
| Trip2 | 14-Apr-2018 | 570 | 0.33 | M1_M2_M4 |
| ... | ... | ... | ... | ... |

(b) Bridge Table

| TripID | StoreID |
|---|---|
| Trip1 | M1 |
| Trip1 | M2 |
| Trip1 | M4 |
| Trip1 | M3 |
| Trip1 | M8 |
| Trip2 | M4 |
| Trip2 | M1 |
| Trip2 | M2 |
| … | … |

```
listagg (Attr1, '_') within group
    (order by Attr1) as ColumnName
```



# Case Study #2 – A Truck Delivery Case Study

- To create Trip Dimension:
  - 
```
create table TripDim3 as
select T.TripID, T.TripDate, T.TotalKm,
    1.0/count(D.StoreID) as WeightFactor,
    listagg (D.StoreID, '_') within group
    (order by D.StoreID) as StoreGroupList
from Trip T, Destination D
where T.TripID = D.TripID
group by T.TripID, T.TripDate, T.TotalKm;
```

# Case Study #2 – A Truck Delivery Case Study

- Joining based on the *StoreGroupList* attribute in the Trip dimension table and the *StoreID* in the Store dimension table:
  - 
```
select *
from TripDim3 T, StoreDim3 S
where T.StoreGroupList like '%'||S.StoreID||'%';
```

- Without the *StoreGroupList* attribute in the Trip dimension, we need to join three tables:
  - 
```
select *
from TripDim3 T, BridgeTable3 B, StoreDim3 S
where T.TripID = B.TripID
and B.StoreID = S.StoreID;
```

# Bridge Tables Summary

MONASH University

# Bridge Tables – Summary

- In principal, a Bridge Table is used:

  a) When it is impossible to have a dimension connected directed to the Fact table, because simply there is no relationship between this dimension and the Fact table (e.g. in the Product Sales case study, it is impossible to have a direct link from SupplierDim to ProductSalesFact)

  b) When an entity (which will become a dimension) has a many-many relationship with another entity (dimension) in the E/R schema of the operational database (e.g. Supplier and Stock has a many-many relationship).

  c) 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 (e.g. product supply history is maintained in the second snowflake schema example).

# Bridge Tables – Summary

- When a Bridge Table is used in the schema, there are two additional options:

  a) A Weight Factor is used to estimate the contribution of a dimension in the calculation of the fact measure. Because this is only an estimate, a weight factor is option.

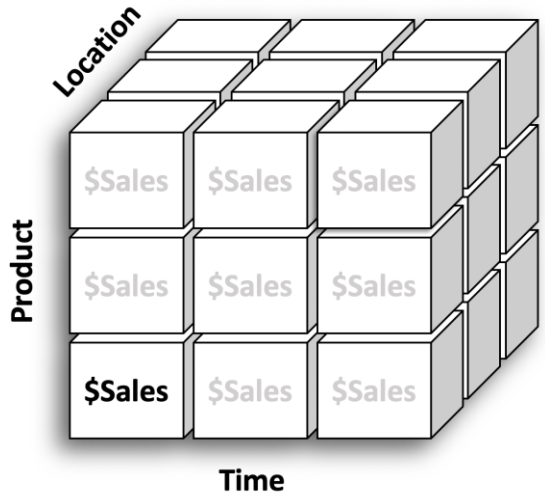
  b) Every snowflake schema (whether it has Weight Factor or not) can be implemented in two ways: a List Aggregate version, and a non-List Aggregate version.

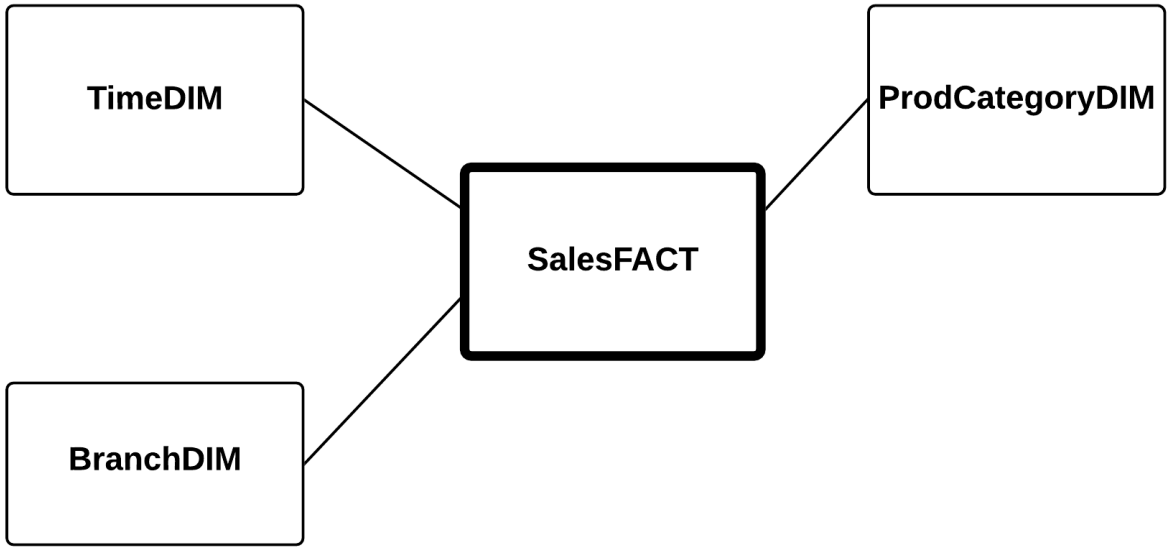
# More Complex Processes in Creating Fact

MONASH University

# Recall – Star Schema Components

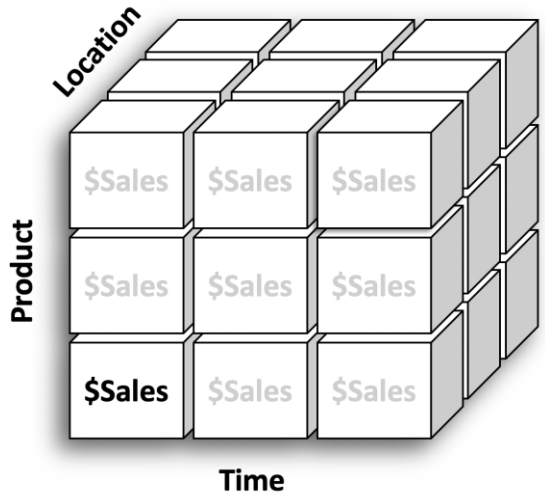
- There are **Three** main components of the Star Schema:
  1. Facts
  2. Dimensions
  3. Attributes

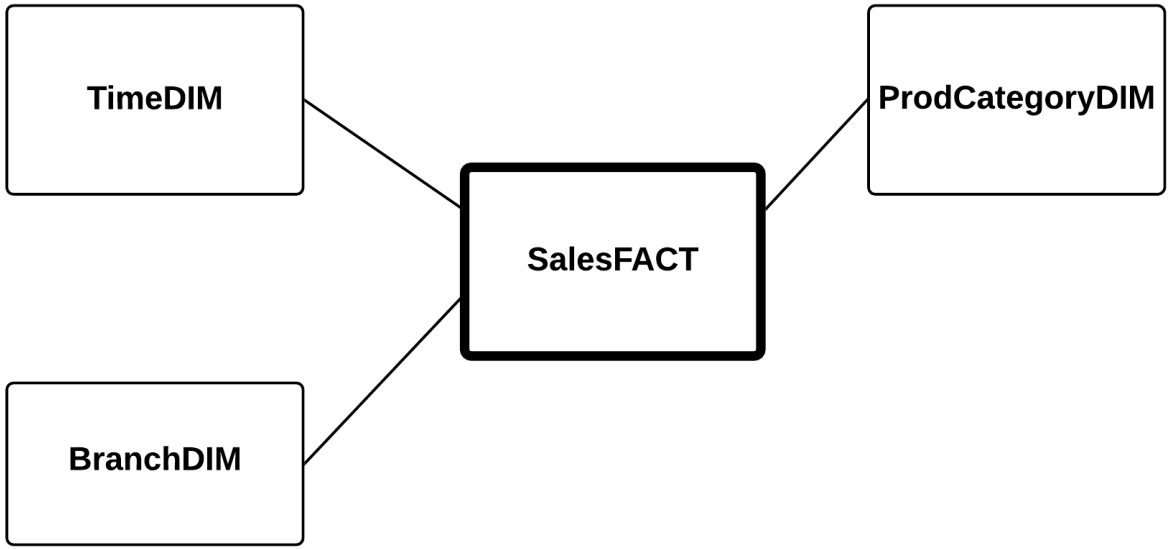


# Recall – Star Schema Components

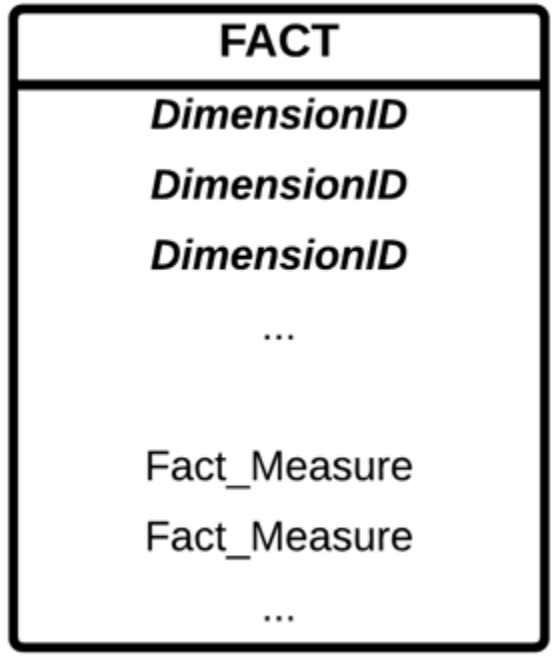
- There are **Three** main components of the Star Schema:
  1. Facts
  2. Dimensions
  3. Attributes



# Recall – Fact

- A Fact Table consists of key attributes from each dimension, and fact measures.
- A Fact Table is created by a join operation, that joins several tables from the operational database.
- Fact tables are created either through *TempFact* or directly retrieval from the tables in the operational database.
- The fact measure itself is an aggregated value.
  - In the SQL command, the fact measure attribute in the Fact Table is created using an aggregate function, such as `count` or `sum`, and the `group by` operation.



# Average in the Fact

# Average in the Fact

| 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 |

# Average in the Fact

| 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 |

The operational database contains:

- 9 records of Semester one
- 7 records of Semester two

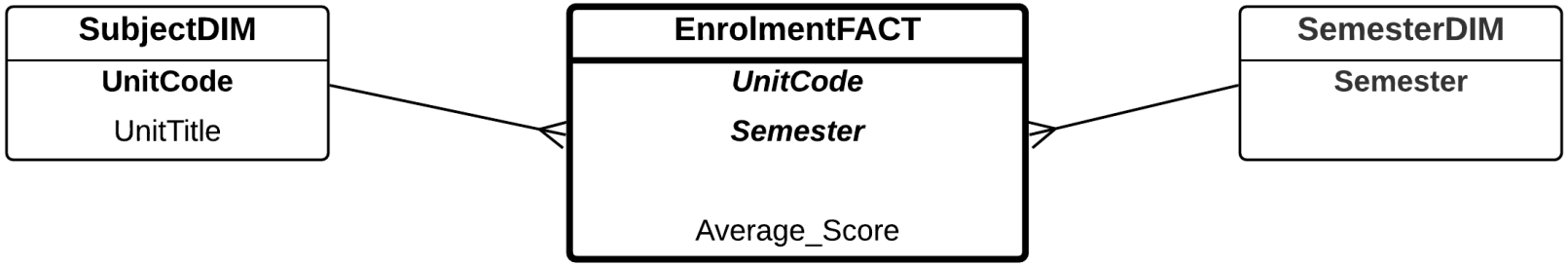
# Average in the Fact

| 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 |

The operational database contains:

- 9 records of Semester one
- 7 records of Semester two
- 8 records of Database Unit (6 Semester one and 2 Semester two)

# Average in the Fact




(a) 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 |

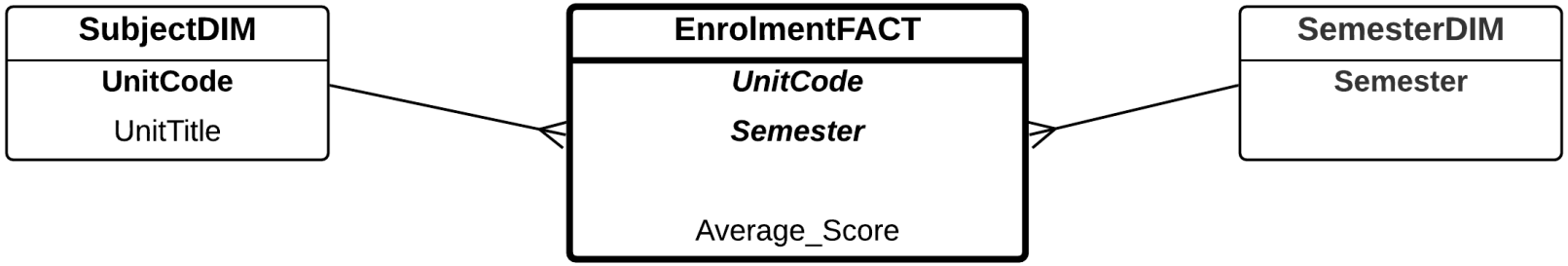
(b) Subject Dimension

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

(c) Semester Dimension

| Semester |
|---|
| 1 |
| 2 |

# Average in the Fact




(a) 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 |

(b) Subject Dimension

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

(c) Semester Dimension

| Semester |
|---|
| 1 |
| 2 |

# Average in the Fact

Average Score for the Database Unit in:

- Semester one: (81+41+74+85+87+75)/6 = **73.833**

(a) 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 |

(b) Subject Dimension

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

(c) Semester Dimension

| Semester |
|---|
| 1 |
| 2 |

# Average in the Fact

Average Score for the Database Unit in:

- Semester one: (81+41+74+85+87+75)/6 = **73.833**
- Semester two: (64+32)/2 = **48**

(a) 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 |

(b) Subject Dimension

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

(c) Semester Dimension

| Semester |
|---|
| 1 |
| 2 |

# Average in the Fact

Average Score for the Database Unit in:

- Semester one: (81+41+74+85+87+75)/6 = **73.833**
- Semester two: (64+32)/2 = **48**

**These are actually incorrect!**

(a) 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 |

(b) Subject Dimension

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

(c) Semester Dimension

| Semester |
|---|
| 1 |
| 2 |

# Average in the Fact

**Query:** Calculate Average Score for the Database Unit:

- (73.833 + 48) / 2 = 60.9165

The SQL command:

```
select avg(Average_Score)
from EnrolmentFact
where UnitCode = 'IT001';
```

(a) 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 |

(b) Subject Dimension

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

(c) Semester Dimension

| Semester |
|---|
| 1 |
| 2 |

# Average in the Fact

**Query:** Calculate Average Score for the Database Unit.

Calculation using **Fact**:

- (73.833 + 48) / 2 = 60.9165

Calculation based on the **Operational Database**:

- (81+41+74+85+87+75+64+32) / 8 = 539 / 8 = 67.375

(a) 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 |

(b) Operational Database

| 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 |

# Average in the Fact

**Query:** Calculate Average Score for the Database Unit.

Calculation using **Fact**:

- (73.833 + 48) / 2 = **60.9165**

Calculation based on the **Operational Database**:

- (81+41+74+85+87+75+64+32) / 8 =
  539 / 8 = **67.375**

(a) 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 |

(b) Operational Database

| 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 |

# Average in the Fact

**Query:** Calculate Average Score for the Java Unit in both Semesters.

Calculation using **Fact**:

- (56+71.667) / 2 = **63.833**

Calculation based on the **Operational Database**:

- (65+47+78+73+64) / 5 = **65.4**

(a) 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 |

(b) Operational Database

| 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 |

# Average in the Fact

**Query:** Calculate Average Score for Semester One.

Calculation using **Fact**:

- (73.833+56+63) / 3 = **64.287**

Calculation based on the **Operational Database**:

- (81+41+74+85+87+75+65+47+63) / 9 = **68.667**

(a) 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 |

(b) Operational Database

| 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 |

# Average in the Fact

**Query:** Calculate Average Score for Semester Two.

Calculation using **Fact**:

- (48+71.667+52.5) / 3 = **57.389**

Calculation based on the **Operational Database**:

- (64+32+78+73+64+53+53) / 7 = **59.4286**

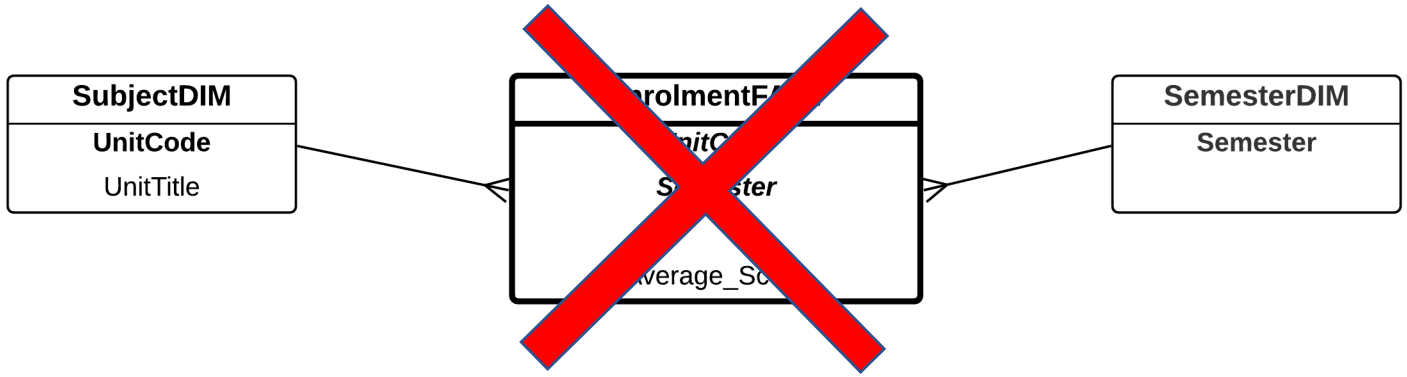
(a) Fact

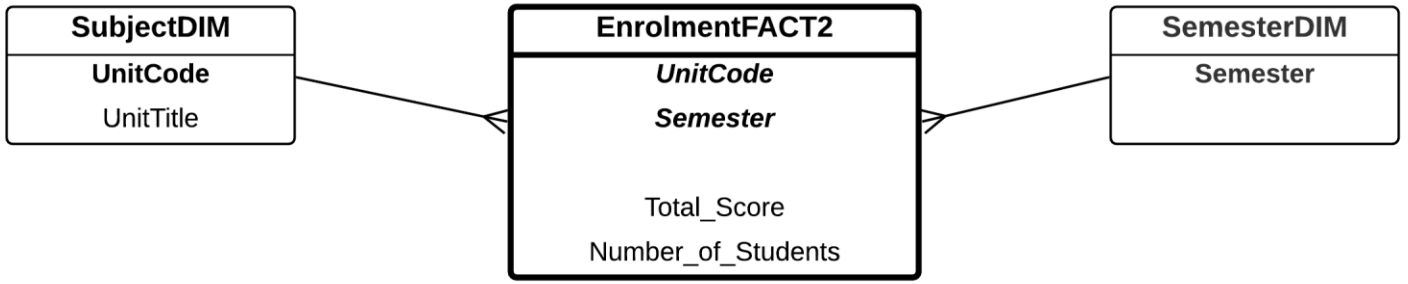
| Unit Code | Semester | Average_Score |
|---|---|---|
| IT001 | 1 | 73.833 |
| IT001 | 2 | 48 |
| IT002 | 1 | 56 |
| IT002 | 2 | 71.667 |
| IT003 | 1 | 63 |
| IT004 | 2 | 52.5 |

(b) Operational Database

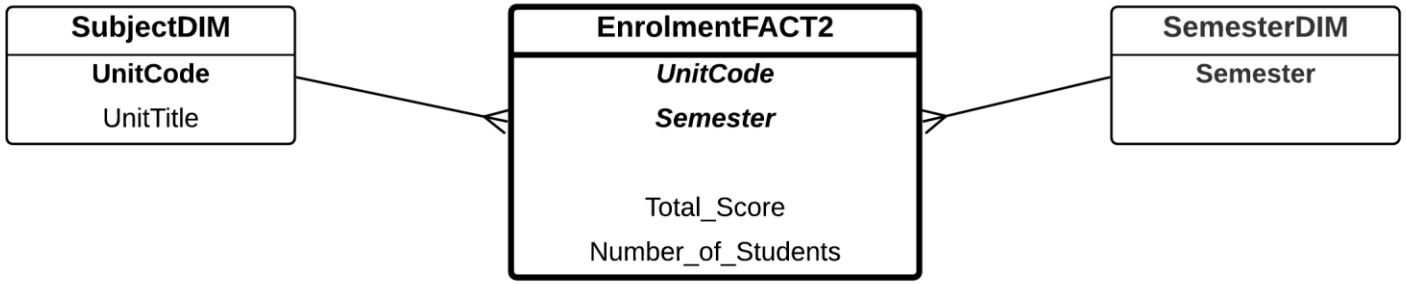
| Unit Code | Unit Title | Semester | Student First Name | Score |
|---|---|---|---|---|
| **IT001** | Database | 1 | Mirriam | 81 |
| **IT001** | Database | 1 | Allan | 41 |
| **IT001** | Database | 1 | Ben | 74 |
| **IT001** | Database | 1 | Kate | 85 |
| **IT001** | Database | 1 | Larry | 87 |
| **IT001** | 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 |

# Average in the Fact



# Average in the Fact




(a) Fact Version 2

| Unit Code | Semester | Total_Score | Number_of_Students |
|---|---|---|---|
| IT001 | 1 | 443 | 6 |
| IT001 | 2 | 96 | 2 |
| IT002 | 1 | 112 | 2 |
| IT002 | 2 | 215 | 3 |
| IT003 | 1 | 63 | 1 |
| IT004 | 2 | 105 | 2 |

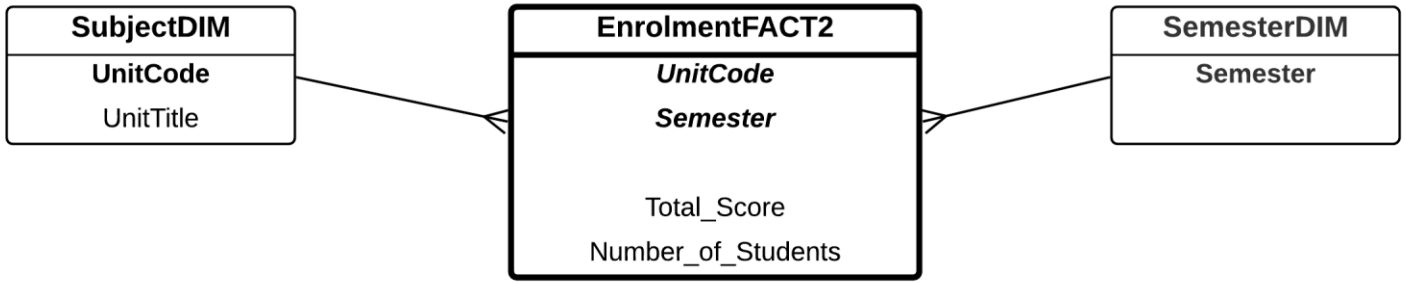
(b) Subject Dimension

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

(c) Semester Dimension

| Semester |
|---|
| 1 |
| 2 |

# Average in the Fact




(a) Fact Version 2

| Unit Code | Semester | Total_Score | Number_of_Students |
|---|---|---|---|
| IT001 | 1 | 443 | 6 |
| IT001 | 2 | 96 | 2 |
| IT002 | 1 | 112 | 2 |
| IT002 | 2 | 215 | 3 |
| IT003 | 1 | 63 | 1 |
| IT004 | 2 | 105 | 2 |

**Query:** Calculate Average Score for the Database Unit:

- (443+96)/(6+2) = 67.375

The SQL command:

```
select sum(Total_Score)/
    sum(Number_of_Students)
    as Average_Score
from EnrolmentFact2
where UnitCode = 'IT001';
```

# Average in the Fact

- The problem of Average in the Fact is known as the ***Average of an Average*** problem.
  - This problem is well known in Mathematics and Statistics.
  - Average of an average will simply produce an incorrect average result.

- Hence, it is not desirable to have an average measure in the fact.
  - *Exceptional case*: when the analysis ALWAYS uses all the dimensions (e.g. Determinant Dimensions).

# Min & Max in the Fact

- If Average should not be used in the fact, how about Min or Max?

# Min & Max in the Fact

- If Average should not be used in the fact, how about Min or Max?
  - Yes, we can.
  - Because Max of Max is always a global Max, and Min of Min is always a global Min.

# Min & Max in the Fact

- If Average should not be used in the fact, how about Min or Max?
  - Yes, we can.
  - Because Max of Max is always a global Max, and Min of Min is always a global Min.

(a) Fact

| Ucode | Semester | Min_Score | Max_Score |
|---|---|---|---|
| IT001 | 1 | 41 | 87 |
| IT001 | 2 | 32 | 64 |
| IT002 | 1 | 47 | 65 |
| IT002 | 2 | 64 | 78 |
| IT003 | 1 | 63 | 63 |
| IT004 | 2 | 52 | 53 |

(b) Subject Dimension

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

(c) Semester Dimension

| Semester |
|---|
| 1 |
| 2 |

# Min & Max in the Fact

- Query: Find the Maximum Score of Database Unit.
  - Max of {87, 64} is **87**.
- The SQL command:

```
select max(Max_Score) from EnrolmentFact
where UnitCode = 'IT001';
```

(a) Fact

| Ucode | Semester | Min_Score | Max_Score |
|---|---|---|---|
| IT001 | 1 | 41 | 87 |
| IT001 | 2 | 32 | 64 |
| IT002 | 1 | 47 | 65 |
| IT002 | 2 | 64 | 78 |
| IT003 | 1 | 63 | 63 |
| IT004 | 2 | 52 | 53 |

(b) Subject Dimension

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

(c) Semester Dimension

| Semester |
|---|
| 1 |
| 2 |

# Min & Max in the Fact

- Query: Find the Minimum Score of Database Unit.
  - Min of {41, 32} is **32**.
- The SQL command:

```
select min(Min_Score) from EnrolmentFact
where UnitCode = 'IT001';
```

(a) Fact

| Ucode | Semester | Min_Score | Max_Score |
|---|---|---|---|
| IT001 | 1 | 41 | 87 |
| IT001 | 2 | 32 | 64 |
| IT002 | 1 | 47 | 65 |
| IT002 | 2 | 64 | 78 |
| IT003 | 1 | 63 | 63 |
| IT004 | 2 | 52 | 53 |

(b) Subject Dimension

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

(c) Semester Dimension

| Semester |
|---|
| 1 |
| 2 |

# Average in the Fact – Conclusion

- **Average in the Fact is not desirable**, although technically it satisfies the two criteria of the fact (e.g. must be a numerical and aggregate value).
- *Min* and *Max* in the Fact can still be used, since Min Score and Max Score are valid fact measures (e.g. they are numerical and aggregated values).
- In general, `count` and `sum` are more common.