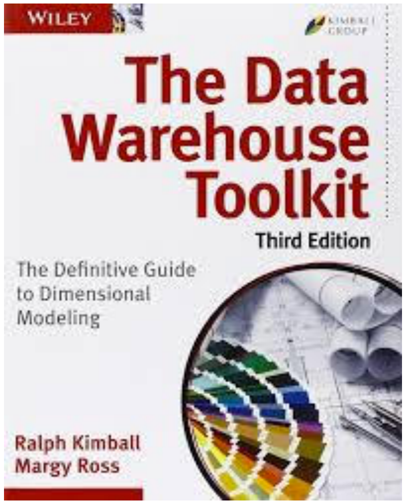
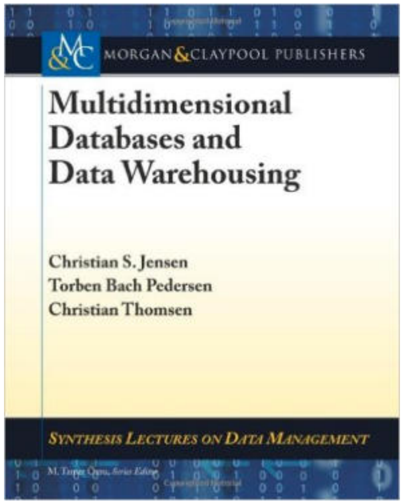
Resources

these slides cannot replace the textbooks by any means !

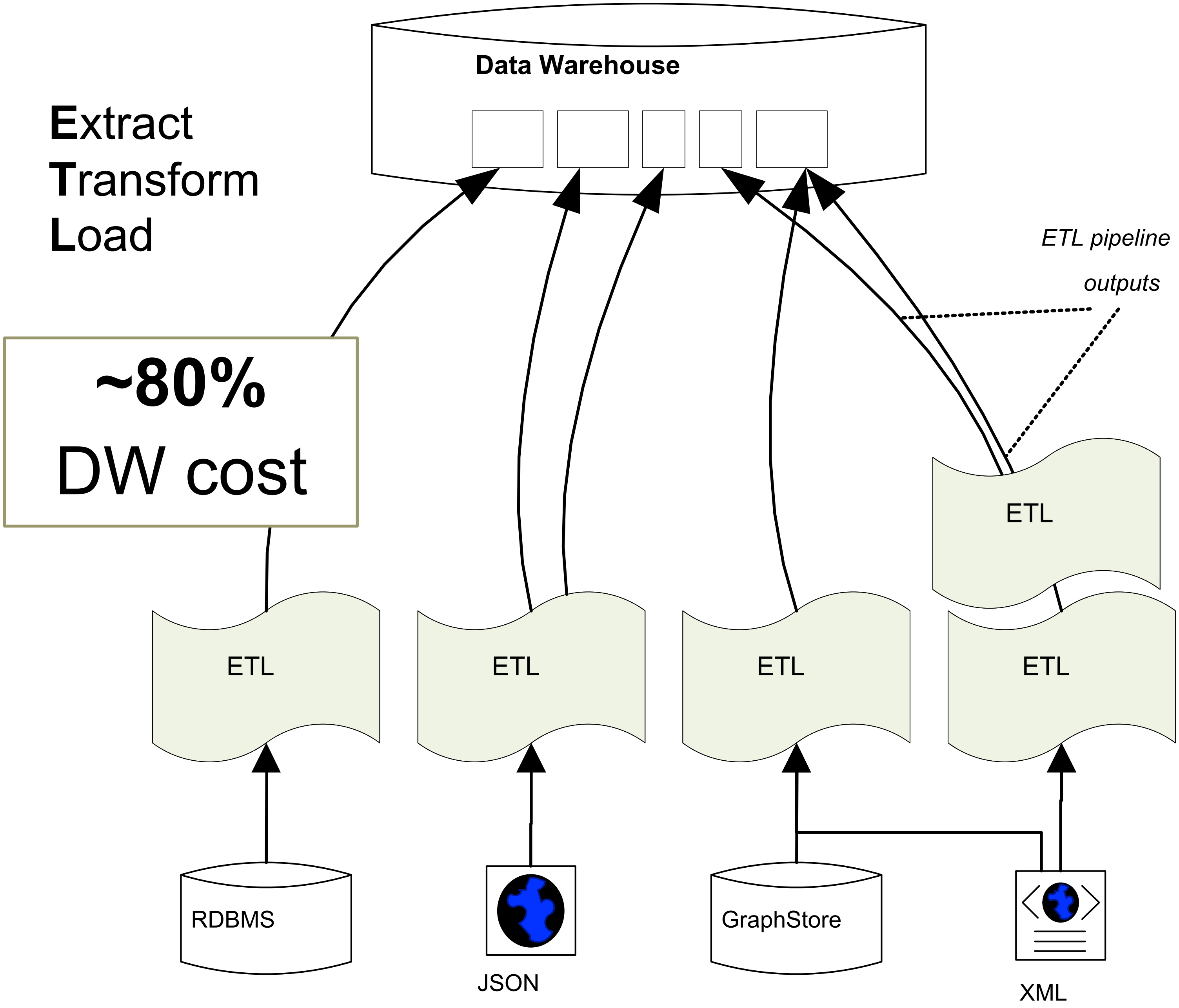
- Entrepôts de données, guide pratique de modélisation dimensionnelle. R.Kimball, M.Ross


WILEY
The Data Warehouse Toolkit
Third Edition
The Definitive Guide to Dimensional Modeling
Ralph Kimball
Margy Ross


MORGAN & CLAYPOOL PUBLISHERS
Multidimensional Databases and Data Warehousing
Christian S. Jensen
Torben Bach Pedersen
Christian Thomsen
SYNTHESIS LECTURES ON DATA MANAGEMENT

- Multidimensional databases and Data Warehousing C.S. Jensen, T.B.Pedersen and C.Thomsen

The Art of Designing a Datawarehouse : the Retail Case Part 1



Extract
Transform
Load

~80%
DW cost

Data Integration

http://www.financegirltoronto.com/wp-content/uploads/2017/07/getting-data-right.pdf

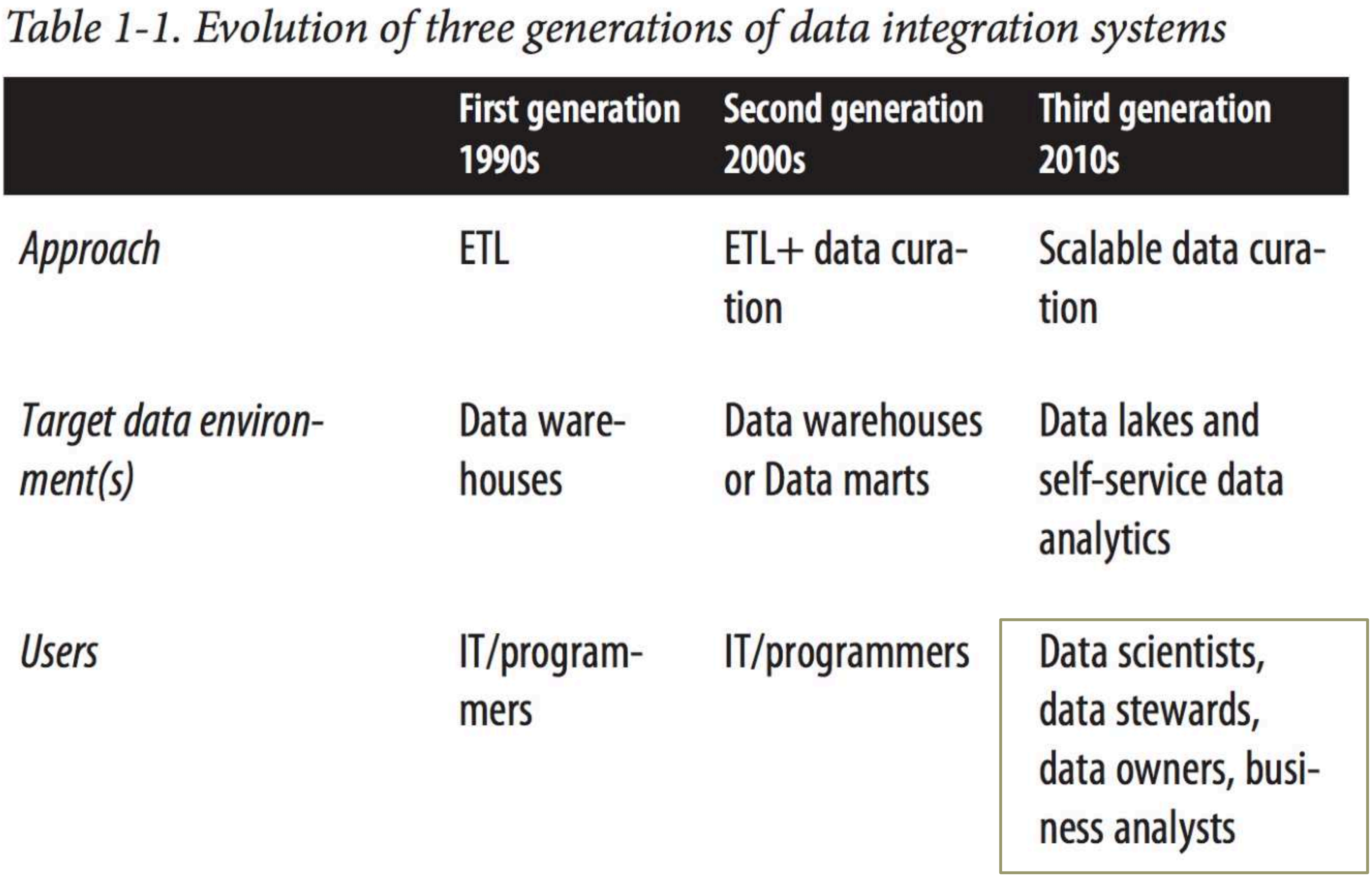


Table 1-1. Evolution of three generations of data integration systems

	First generation 1990s	Second generation 2000s	Third generation 2010s
Approach	ETL	ETL+ data curation	Scalable data curation
Target data environment(s)	Data warehouses	Data warehouses or Data marts	Data lakes and self-service data analytics
Users	IT/programmers	IT/programmers	Data scientists, data stewards, data owners, business analysts

Modelling for Data Analysis

- **Key question : which are the most important aspects to model inside a DW** ?
  (talking in terms of business)

- Treat then in order of importance !

- ETL procedures are expensive

- Ressources inside a company should be invested accordingly (*keep that in mind also for your project !!*)

Retail Case Study : Grocery Chain

The case :

- 100 Grocery stores spread over a five-state area.
- Each store has ~60,000 individual products on its shelves; 80% come from outside manufacturers.
- Grocery departments : frozen foods, dairy, meat, bakery, floral, and health/beauty aids.

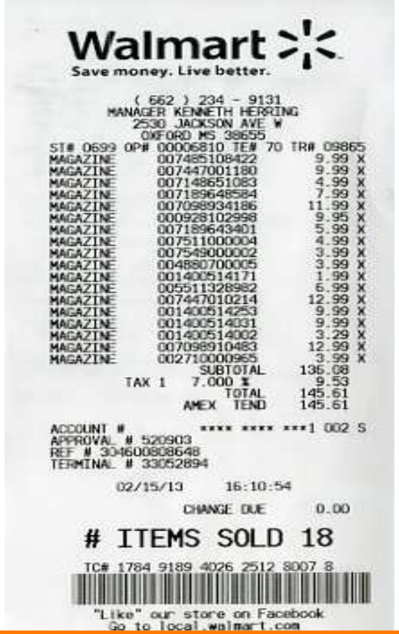
Retail Case Study : Grocery Chain

Data is collected at

- **cash registers** as customers purchase products
- **the back door**, where vendors make deliveries

- Question #1 : Which is the most important aspect (talking in business terms) to analyze ?

Facts



```
Walmart
Save money. Live better.

         ( 662 ) 234 - 9131
      MANAGER KENNETH HERRING
         2530 JACKSON AVE W
          OXFORD MS 38655
ST# 0699 OP# 00006810 TE# 70 TR# 09865
MAGAZINE    007485108422      9.99 X
MAGAZINE    007447001180      9.99 X
MAGAZINE    007148651083      4.99 X
MAGAZINE    007189648584      7.99 X
MAGAZINE    007098934186     11.99 X
MAGAZINE    000928102998      9.95 X
MAGAZINE    007189643401      5.99 X
MAGAZINE    007511000004      4.99 X
MAGAZINE    007549000002      3.99 X
MAGAZINE    004880700005      3.99 X
MAGAZINE    001400514171      1.99 X
MAGAZINE    005511328982      6.99 X
MAGAZINE    007447010214     12.99 X
MAGAZINE    001400514253      9.99 X
MAGAZINE    001400514031      9.99 X
MAGAZINE    001400514002      3.29 X
MAGAZINE    007098910483     12.99 X
MAGAZINE    002710000965      3.99 X
                   SUBTOTAL  136.08
         TAX 1   7.000 %       9.53
                      TOTAL  145.61
               AMEX   TEND   145.61

ACCOUNT #          **** **** ***1 002 S
APPROVAL # 520903
REF # 304600808648
TERMINAL # 33052894

     02/15/13     16:10:54

               CHANGE DUE      0.00

      # ITEMS SOLD 18

    TC# 1784 9189 4026 2512 8007 8

     "Like" our store on Facebook
       Go to local.walmart.com
```

sale

Step 1) Decide business process(es) to model

- **First dimensional model must have the most impact**
- Here, we want to understand customer purchases
- Vendors delivery is set aside for the moment

Step 2) Declare the Grain

- Data expressed at lowest possible grain of dimensions
  - Eases precise and complex analytic queries

- Here, tree choices for the grain
  - by transaction (too coarse)
  - by item type (just right)
  - by item (too fine, here brings no benefits)

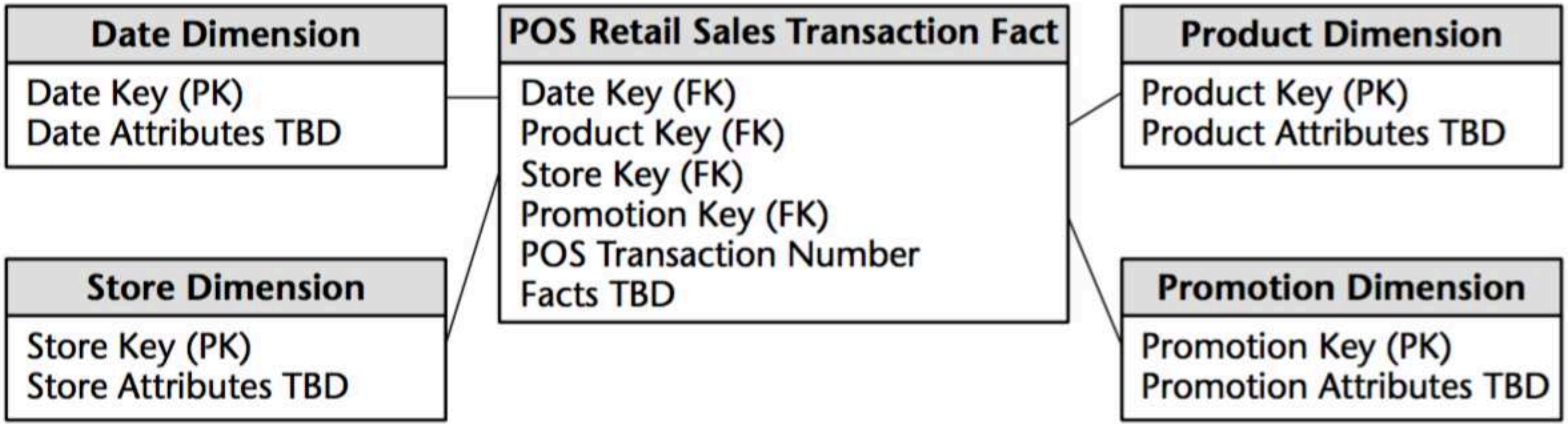
- We choose individual line item type on a selling transaction

Step 3) Choose the dimensions

- Primary dimensions main output of previous step
  - Date, Product, Store, Promotion

- Often possible to add more dimensions later

A first star-schema



Date Dimension

Date Key (PK)
Date Attributes TBD

POS Retail Sales Transaction Fact

Date Key (FK)
Product Key (FK)
Store Key (FK)
Promotion Key (FK)
POS Transaction Number
Facts TBD

Product Dimension

Product Key (PK)
Product Attributes TBD

Store Dimension

Store Key (PK)
Store Attributes TBD

Promotion Dimension

Promotion Key (PK)
Promotion Attributes TBD

TDB = to be defined yet

Step 4) Identify the Measures

- Selling transactions include

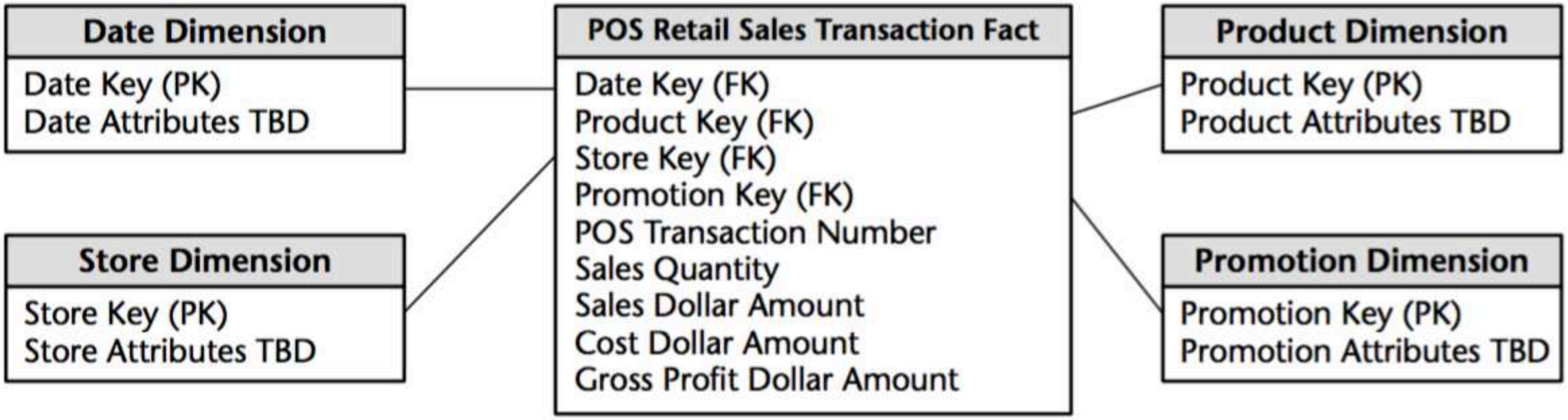
  1. sales quantity

  2. per unit sales price

  3. sales total amount (=1.* 2. , but still included)

- In this case facts are already available

Star-schema : Fact table



Date Dimension

Date Key (PK)
Date Attributes TBD

Store Dimension

Store Key (PK)
Store Attributes TBD

POS Retail Sales Transaction Fact

Date Key (FK)
Product Key (FK)
Store Key (FK)
Promotion Key (FK)
POS Transaction Number
Sales Quantity
Sales Dollar Amount
Cost Dollar Amount
Gross Profit Dollar Amount

Product Dimension

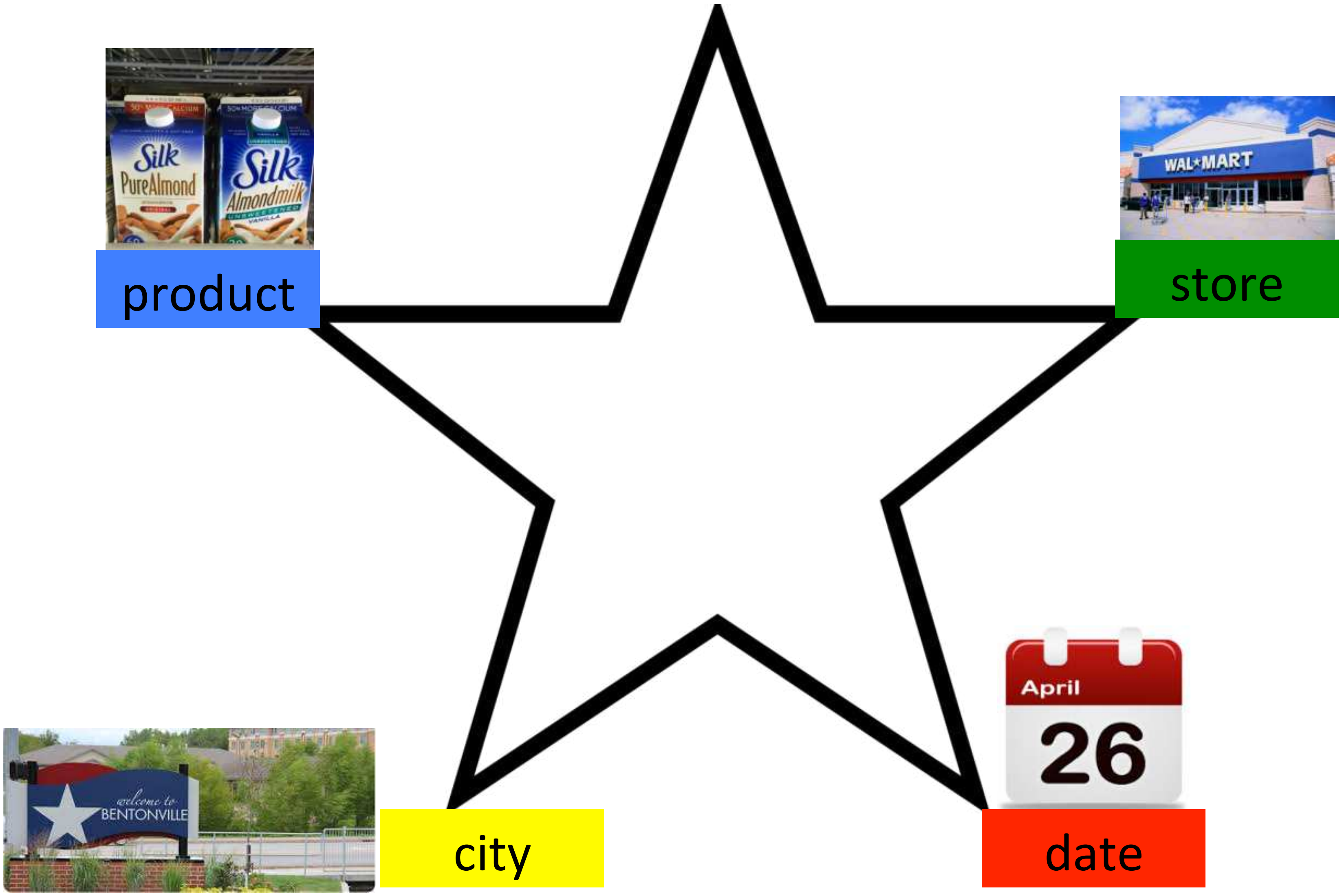
Product Key (PK)
Product Attributes TBD

Promotion Dimension

Promotion Key (PK)
Promotion Attributes TBD

Guidelines

Date vs Time Dimension



product

store

city

date

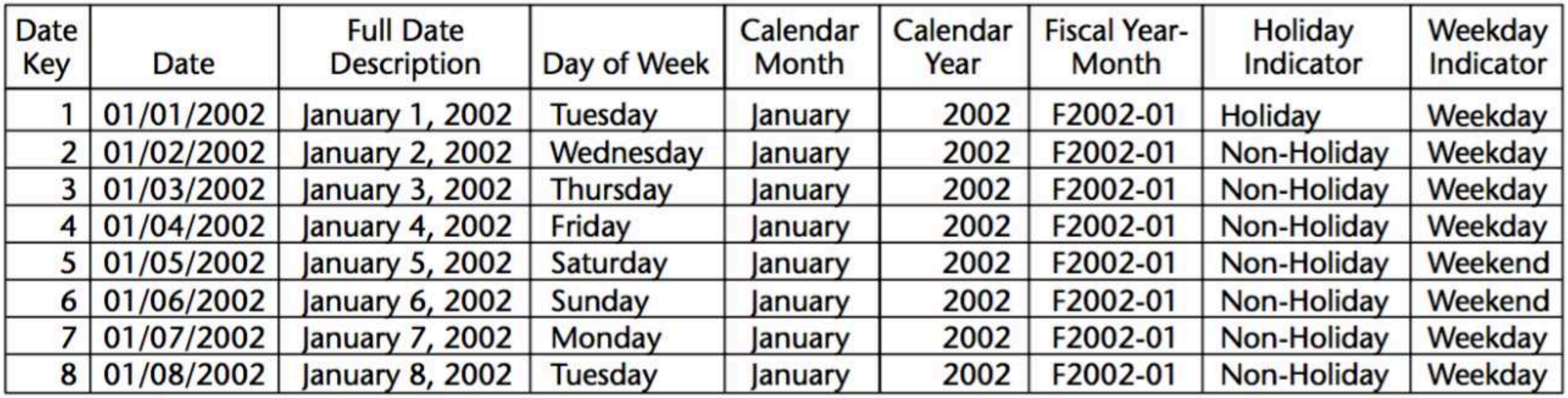
Date Dimension

- The only dimension to be in every datawarehouse

Counterintuitive (at first) but a date is :

- denoted by a key
- complex and fully described in dimensional table

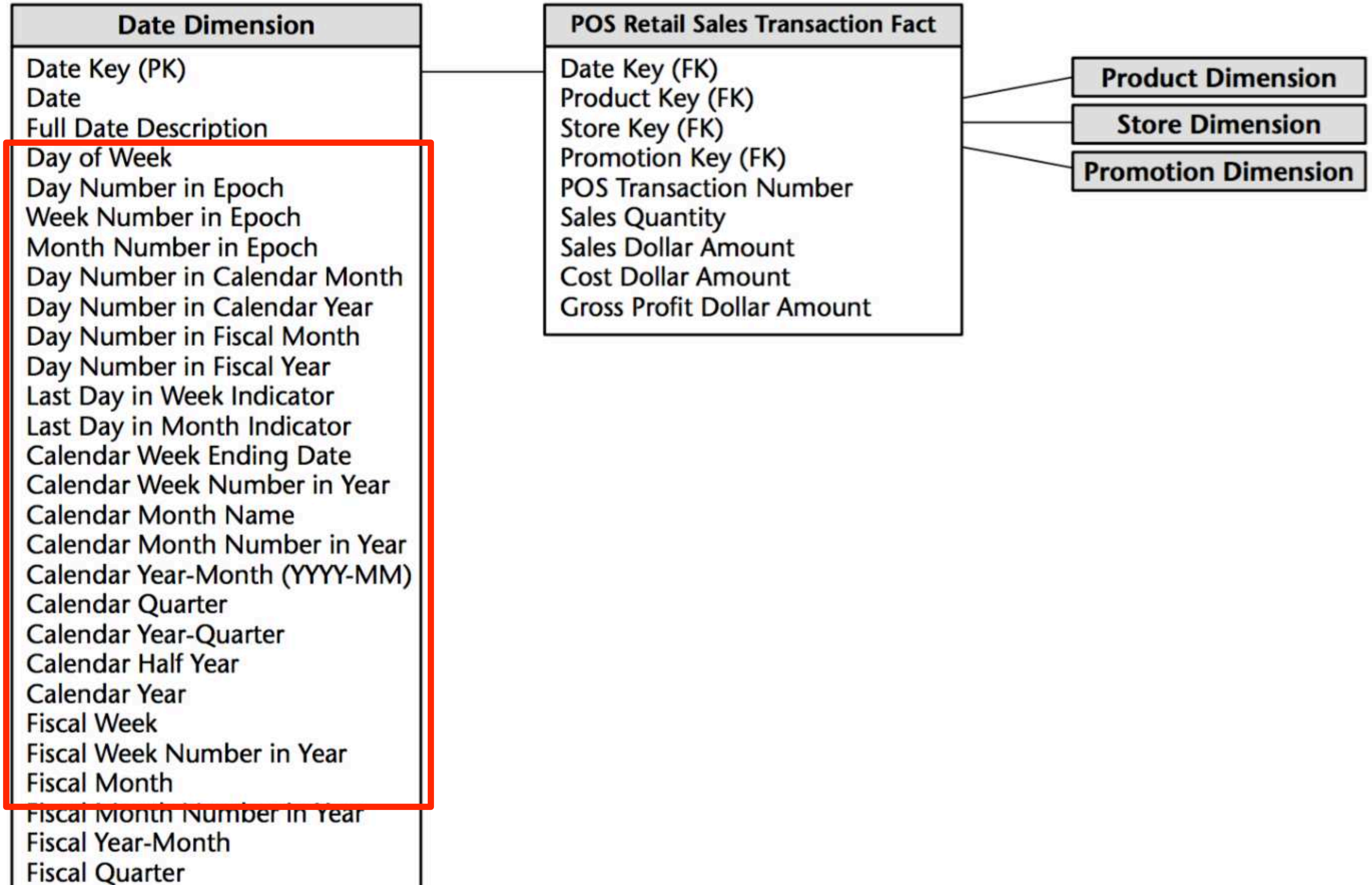
Date Dimension Table



Date Key	Date	Full Date Description	Day of Week	Calendar Month	Calendar Year	Fiscal Year-Month	Holiday Indicator	Weekday Indicator
1	01/01/2002	January 1, 2002	Tuesday	January	2002	F2002-01	Holiday	Weekday
2	01/02/2002	January 2, 2002	Wednesday	January	2002	F2002-01	Non-Holiday	Weekday
3	01/03/2002	January 3, 2002	Thursday	January	2002	F2002-01	Non-Holiday	Weekday
4	01/04/2002	January 4, 2002	Friday	January	2002	F2002-01	Non-Holiday	Weekday
5	01/05/2002	January 5, 2002	Saturday	January	2002	F2002-01	Non-Holiday	Weekend
6	01/06/2002	January 6, 2002	Sunday	January	2002	F2002-01	Non-Holiday	Weekend
7	01/07/2002	January 7, 2002	Monday	January	2002	F2002-01	Non-Holiday	Weekday
8	01/08/2002	January 8, 2002	Tuesday	January	2002	F2002-01	Non-Holiday	Weekday

- 10 years of rows representing days in date dimension table make only 3,650 rows (very small for a DW)

Date Dimension



Date Dimension

Date Key (PK)
Date
Full Date Description
Day of Week
Day Number in Epoch
Week Number in Epoch
Month Number in Epoch
Day Number in Calendar Month
Day Number in Calendar Year
Day Number in Fiscal Month
Day Number in Fiscal Year
Last Day in Week Indicator
Last Day in Month Indicator
Calendar Week Ending Date
Calendar Week Number in Year
Calendar Month Name
Calendar Month Number in Year
Calendar Year-Month (YYYY-MM)
Calendar Quarter
Calendar Year-Quarter
Calendar Half Year
Calendar Year
Fiscal Week
Fiscal Week Number in Year
Fiscal Month
Fiscal Month Number in Year
Fiscal Year-Month
Fiscal Quarter

POS Retail Sales Transaction Fact

Date Key (FK)
Product Key (FK)
Store Key (FK)
Promotion Key (FK)
POS Transaction Number
Sales Quantity
Sales Dollar Amount
Cost Dollar Amount
Gross Profit Dollar Amount

Product Dimension

Store Dimension

Promotion Dimension

Date Dimension

- The **day-of-week** and **calendar-month** columns contains the name of the day, such as Monday, and the name of the month, such as March.

- Used to create reports comparing the business comparing selling in week-days or months

Date ≠ Time

Time Dimension

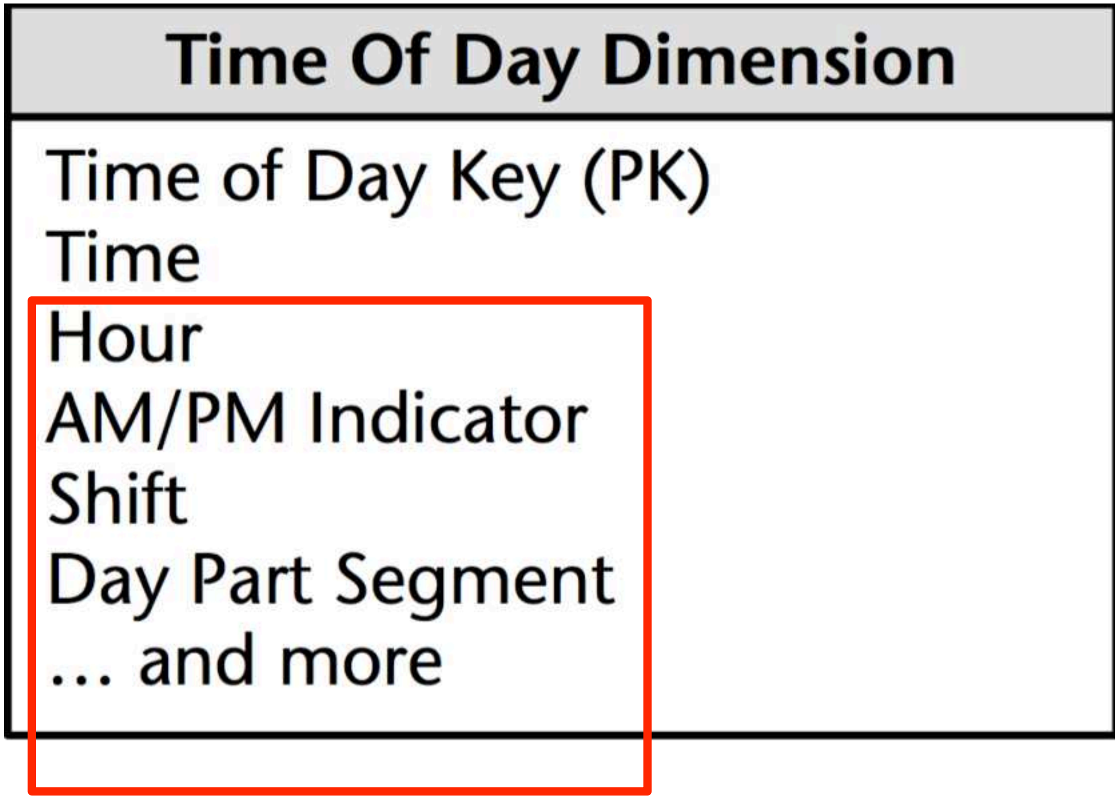
- Date and time are almost completely independent

- Separate time-of-day dimension, day-part analysis (eg, activity during the evening after-work)

- **Combining date and time in a single dimension would make undesirable cartesian product**

- 3,650-row date & 1,440-row time-of-day by minute better than 5,256,000 date-time rows

Time Dimension



Time Of Day Dimension

Time of Day Key (PK)
Time
Hour
AM/PM Indicator
Shift
Day Part Segment
… and more

Avoid
Too Many Dimensions

Correlated Dimensions

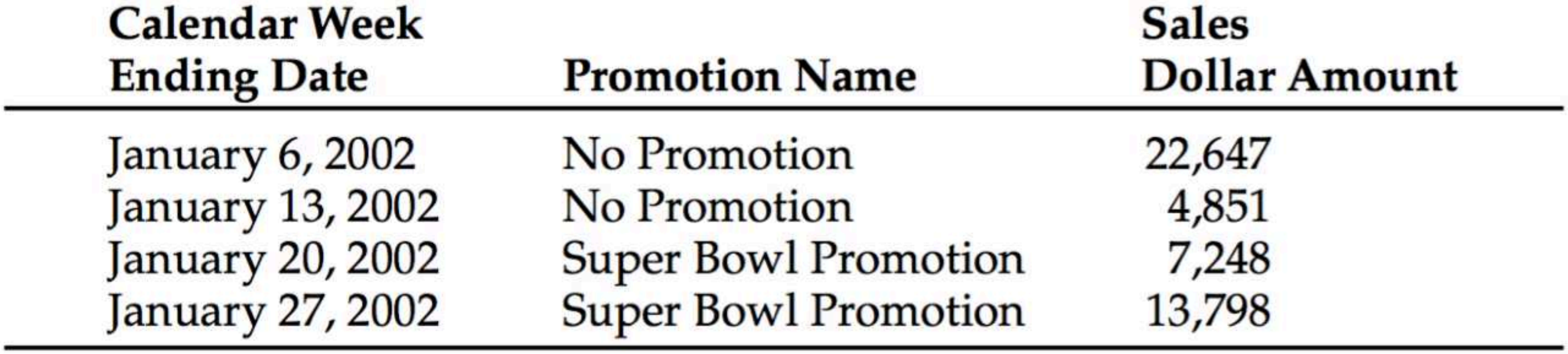
- **Promotion** : conditions under which a product was sold

- Can include :
  - temporary price reductions,
  - newspaper ads,
  - coupons
  - …

- This dimension is often called a *causal* dimension because it describes factors thought to cause a change in product sales.

Retail Schema in Action

- How to determine if a promotion is effective or not ?
- Weekly sales dollar volume by promotion for the snacks category during January 2002 for stores in the Boston district



Calendar Week Ending Date	Promotion Name	Sales Dollar Amount
January 6, 2002	No Promotion	22,647
January 13, 2002	No Promotion	4,851
January 20, 2002	Super Bowl Promotion	7,248
January 27, 2002	Super Bowl Promotion	13,798

Correlated Dimensions

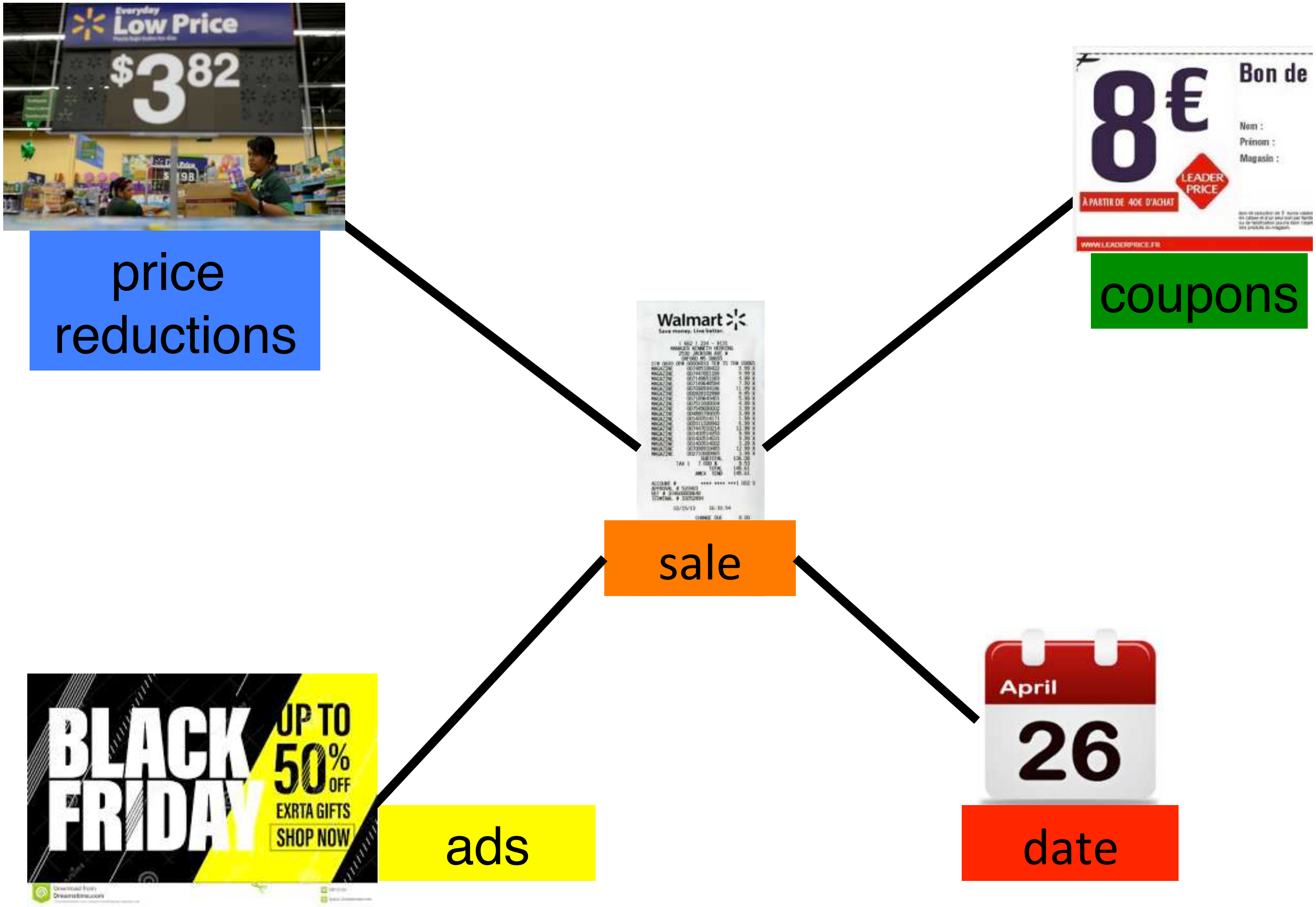
First **solution** : one dimension listing ***the*** promotions

- does not work when there is more than 1 promotion per item (multivalued dependency)

- **every line of the fact table is triplicated in this case !**

Second **solution** : one dimension for each type of promotion

- discouraged : too many dimensions : impossible to index effectively ; space waste in the fact table



price reductions

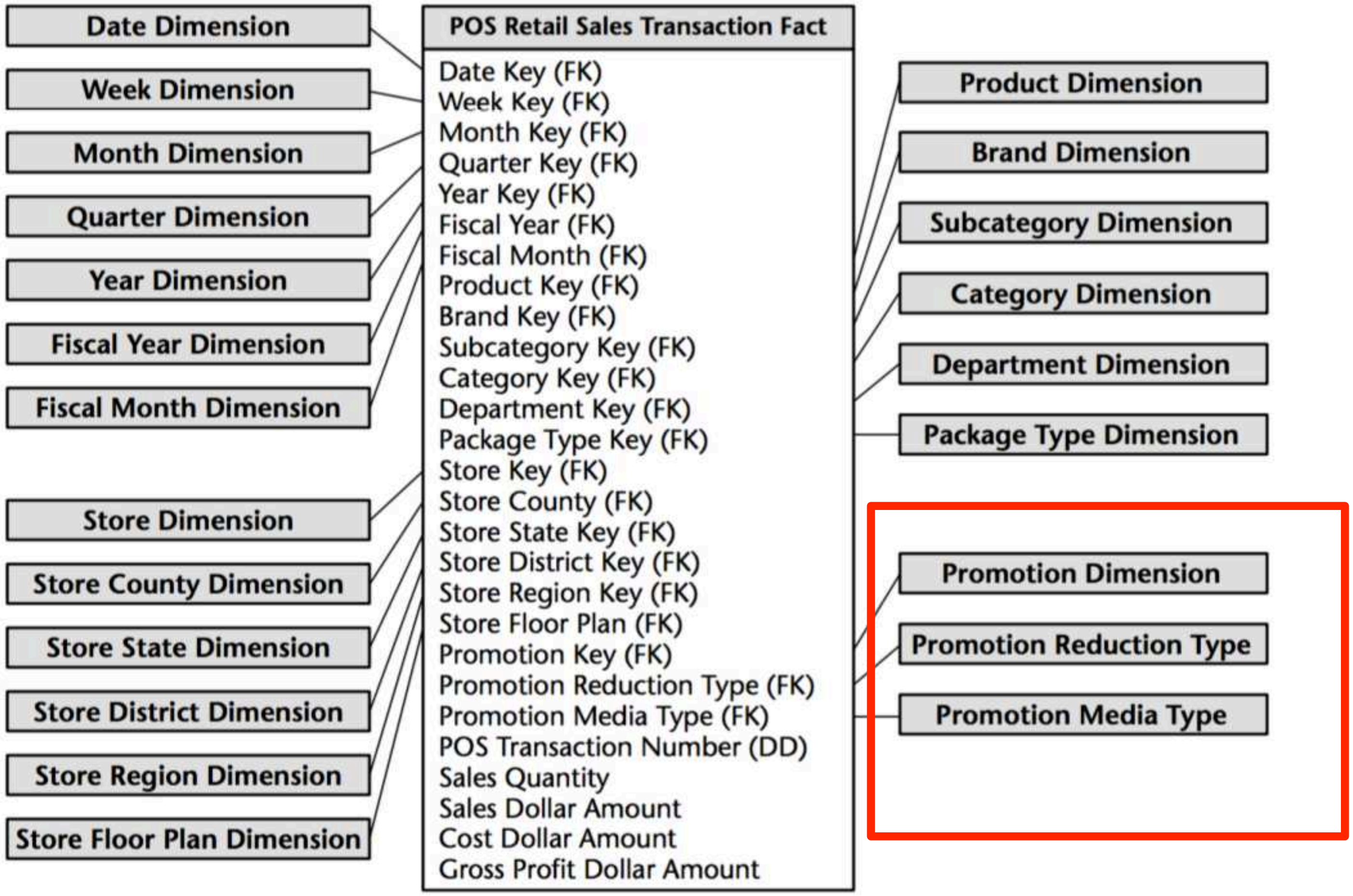
coupons

sale

ads

date

AVOID Too Many Dimensions

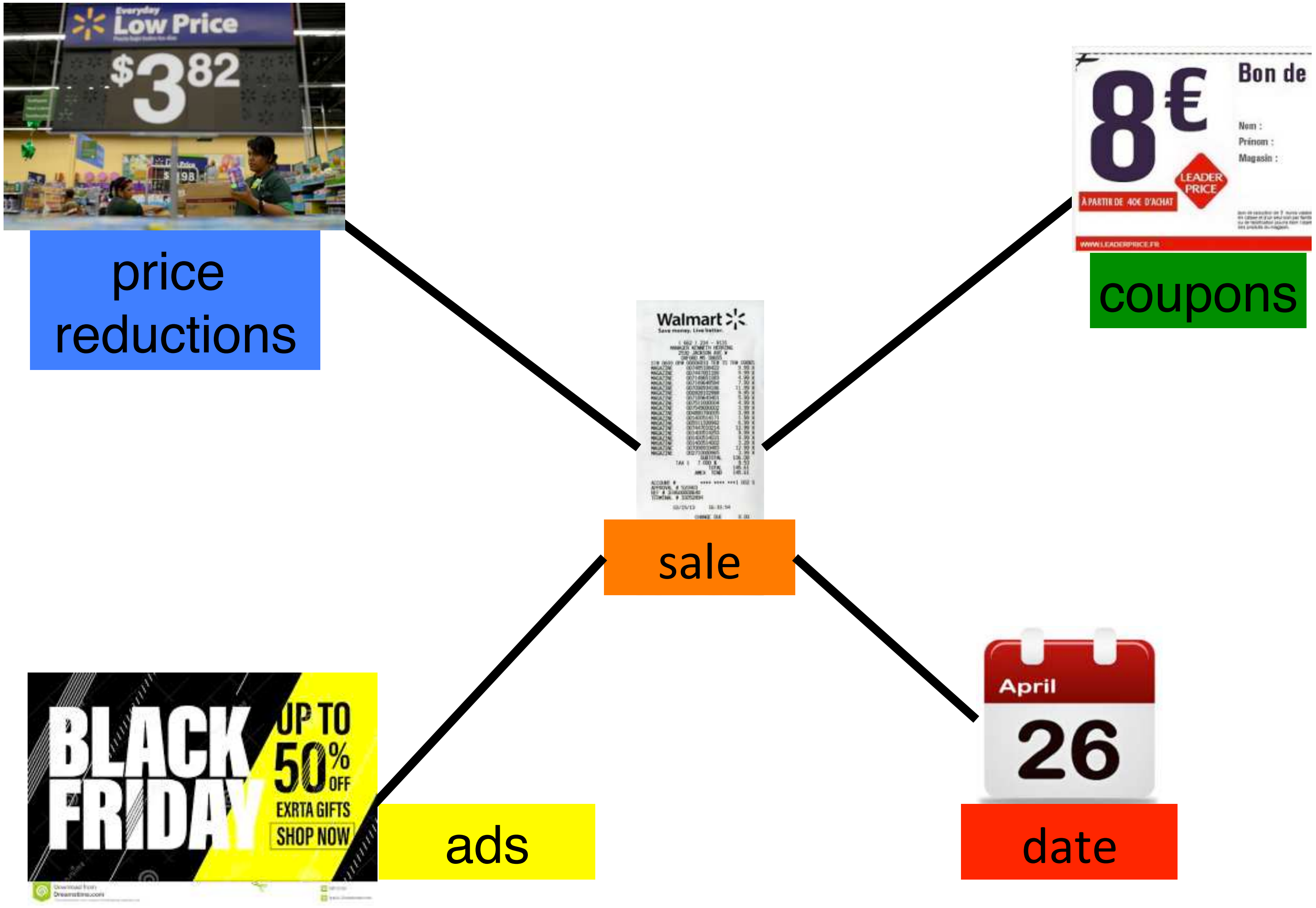


centipede fact table

Correlated Dimensions

- **Third** (the) **solution** : correlated dimensions merged

  into a single dimension



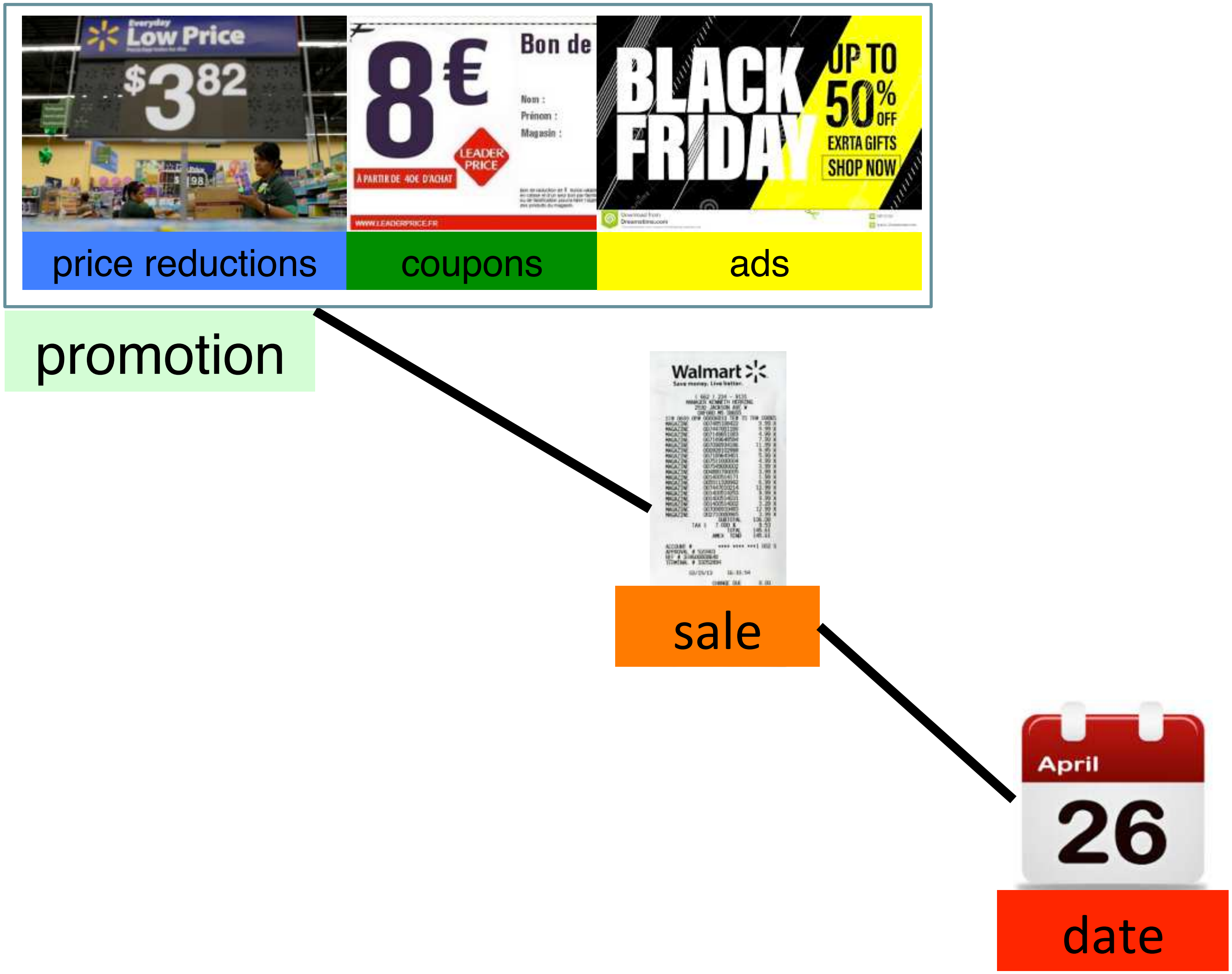
price reductions

coupons

sale

ads

date



price reductions

coupons

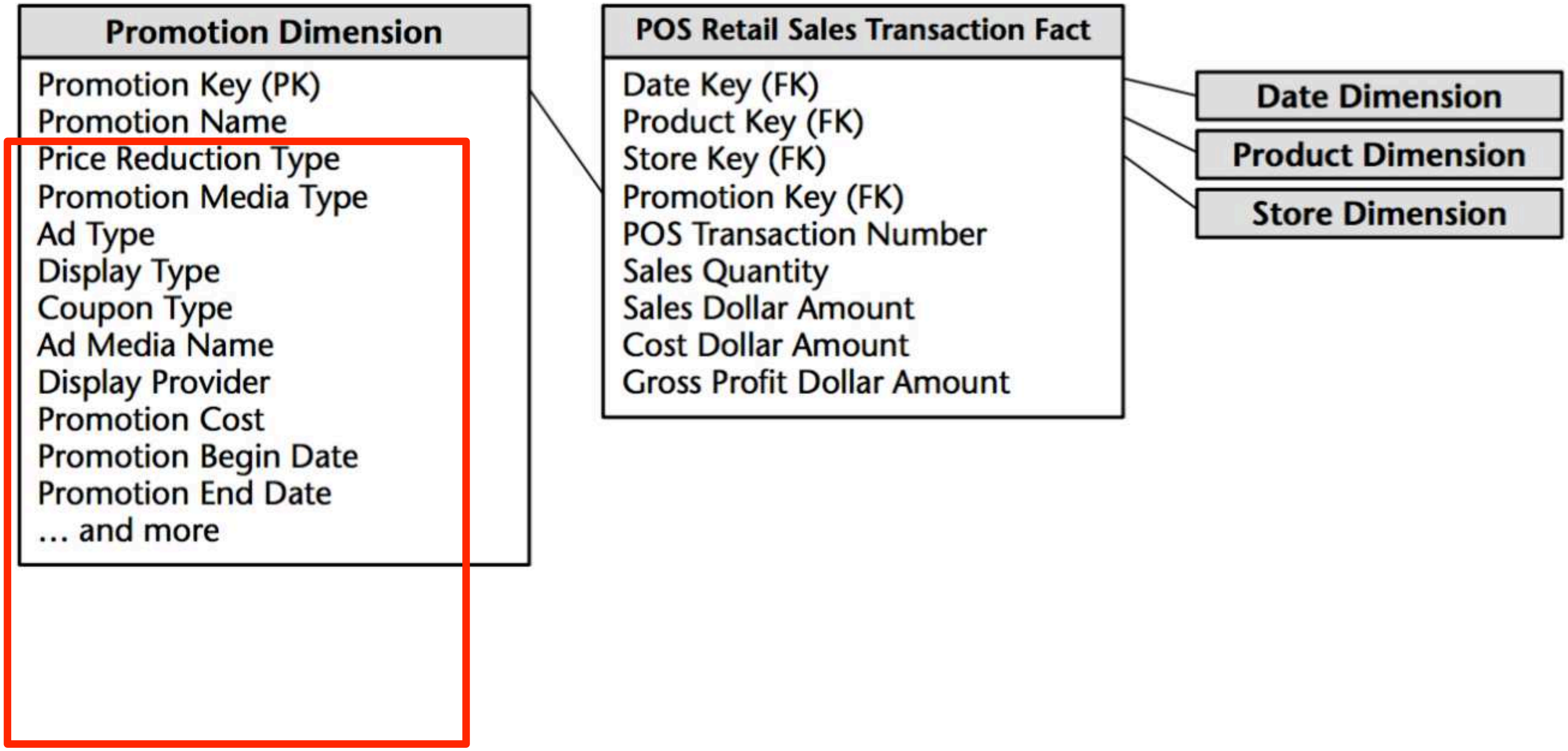
ads

promotion

sale

date

Promotion Dimension



Promotion Dimension

Promotion Key (PK)
Promotion Name
Price Reduction Type
Promotion Media Type
Ad Type
Display Type
Coupon Type
Ad Media Name
Display Provider
Promotion Cost
Promotion Begin Date
Promotion End Date
… and more

POS Retail Sales Transaction Fact

Date Key (FK)
Product Key (FK)
Store Key (FK)
Promotion Key (FK)
POS Transaction Number
Sales Quantity
Sales Dollar Amount
Cost Dollar Amount
Gross Profit Dollar Amount

Date Dimension

Product Dimension

Store Dimension

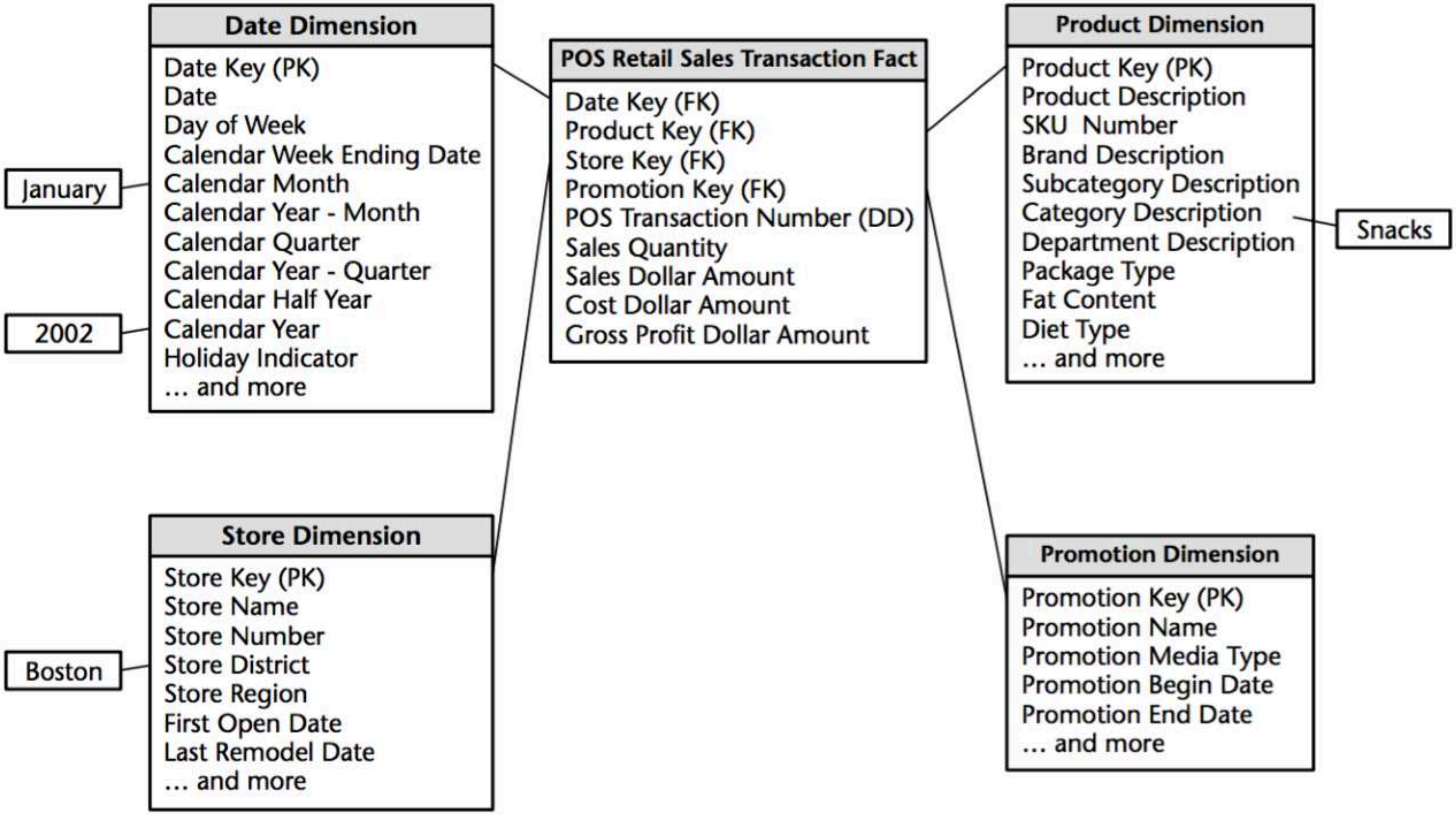
Promotion Dimension

- One row for each combination of promotion conditions
  = exponential table size (in theory)

- **But** : causal conditions are highly correlated:
  A temporary price reduction goes with an ad a
  Coupons often are associated with ads.

- For 1K ads & 5K temporary price reductions we find 2K combinations (out of the possible 5M): store only these

Retail (Star) Schema



Date Dimension

Date Key (PK)
Date
Day of Week
Calendar Week Ending Date
Calendar Month
Calendar Year - Month
Calendar Quarter
Calendar Year - Quarter
Calendar Half Year
Calendar Year
Holiday Indicator
… and more

POS Retail Sales Transaction Fact

Date Key (FK)
Product Key (FK)
Store Key (FK)
Promotion Key (FK)
POS Transaction Number (DD)
Sales Quantity
Sales Dollar Amount
Cost Dollar Amount
Gross Profit Dollar Amount

Product Dimension

Product Key (PK)
Product Description
SKU Number
Brand Description
Subcategory Description
Category Description
Department Description
Package Type
Fat Content
Diet Type
… and more

Store Dimension

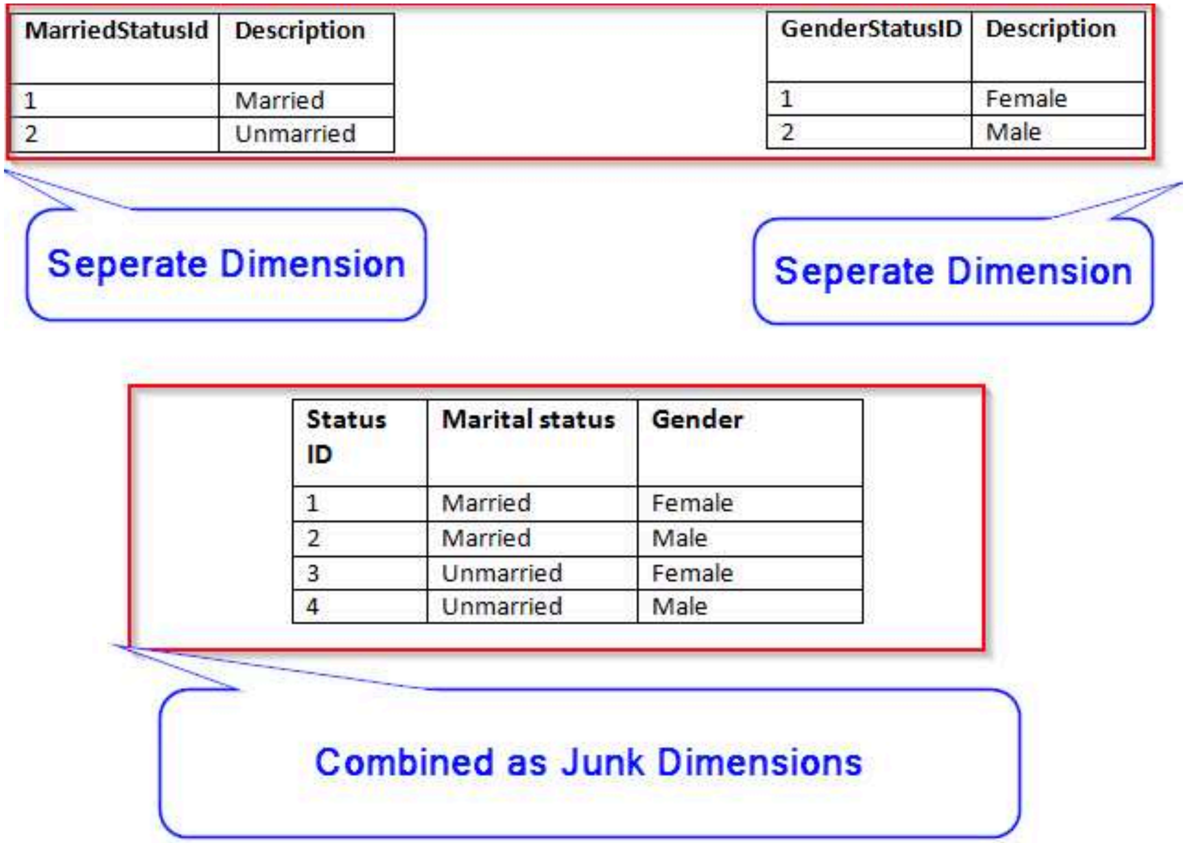
Store Key (PK)
Store Name
Store Number
Store District
Store Region
First Open Date
Last Remodel Date
… and more

Promotion Dimension

Promotion Key (PK)
Promotion Name
Promotion Media Type
Promotion Begin Date
Promotion End Date
… and more

Junk Dimensions

- The result of merging **uncorrelated** **small** dimensions



MarriedStatusId	Description
1	Married
2	Unmarried

Seperate Dimension

GenderStatusID	Description
1	Female
2	Male

Seperate Dimension

Status ID	Marital status	Gender
1	Married	Female
2	Married	Male
3	Unmarried	Female
4	Unmarried	Male

Combined as Junk Dimensions

Avoid Too Many Dimensions

- Group **correlated** dimensions together

- Group **low-cardinality** **independent** dimensions together (junk dimensions)

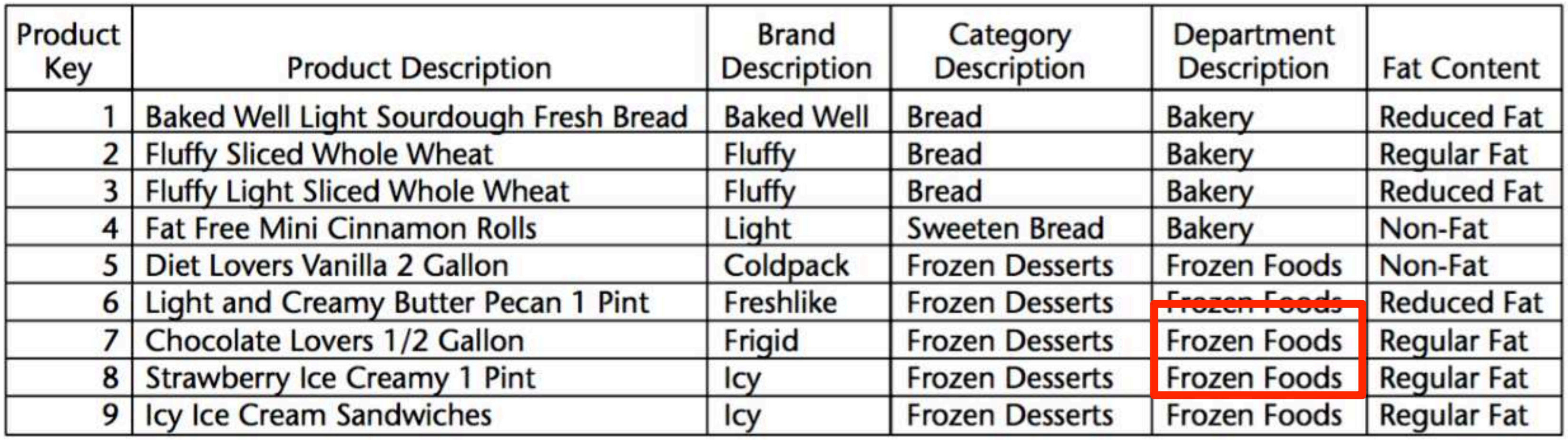
- But always verify that there is no real risk of making big cartesian products

- **Do not store combinations that are impossible !!**

Normalization Theory is over

Dimensions : Redundance is OK

- Assume 30 distinct values for department
- If dimension table has 60K lines, each department is repeated (on average) 2K times



Product Key	Product Description	Brand Description	Category Description	Department Description	Fat Content
1	Baked Well Light Sourdough Fresh Bread	Baked Well	Bread	Bakery	Reduced Fat
2	Fluffy Sliced Whole Wheat	Fluffy	Bread	Bakery	Regular Fat
3	Fluffy Light Sliced Whole Wheat	Fluffy	Bread	Bakery	Reduced Fat
4	Fat Free Mini Cinnamon Rolls	Light	Sweeten Bread	Bakery	Non-Fat
5	Diet Lovers Vanilla 2 Gallon	Coldpack	Frozen Desserts	Frozen Foods	Non-Fat
6	Light and Creamy Butter Pecan 1 Pint	Freshlike	Frozen Desserts	Frozen Foods	Reduced Fat
7	Chocolate Lovers 1/2 Gallon	Frigid	Frozen Desserts	Frozen Foods	Regular Fat
8	Strawberry Ice Creamy 1 Pint	Icy	Frozen Desserts	Frozen Foods	Regular Fat
9	Icy Ice Cream Sandwiches	Icy	Frozen Desserts	Frozen Foods	Regular Fat

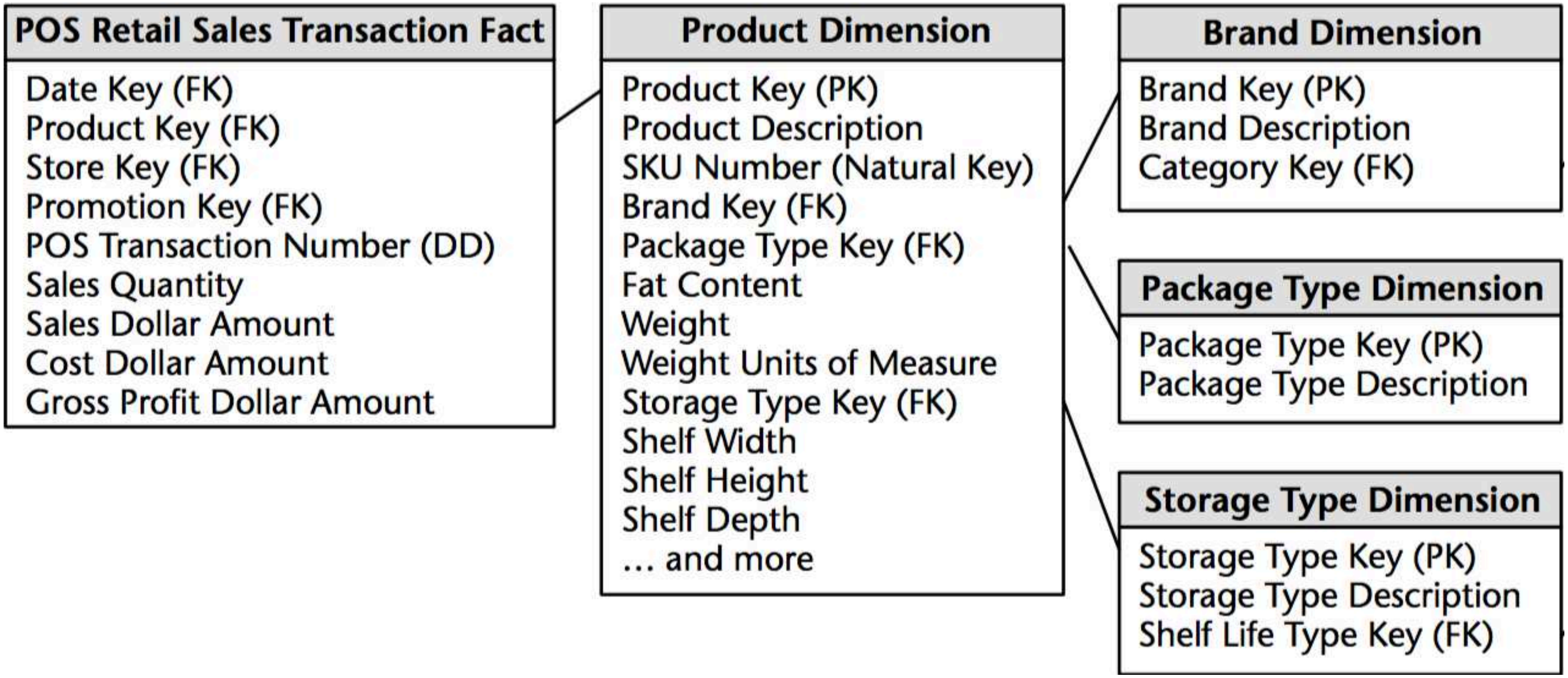
Dimension Table Normalization (Snowflaking)

Dimension Table Normalization (Snowflaking)

- Reduces redundancy in storage

- Allows better mainteinance of dimension values

Dimension Table Normalization (Snowflaking)



POS Retail Sales Transaction Fact
Date Key (FK)
Product Key (FK)
Store Key (FK)
Promotion Key (FK)
POS Transaction Number (DD)
Sales Quantity
Sales Dollar Amount
Cost Dollar Amount
Gross Profit Dollar Amount

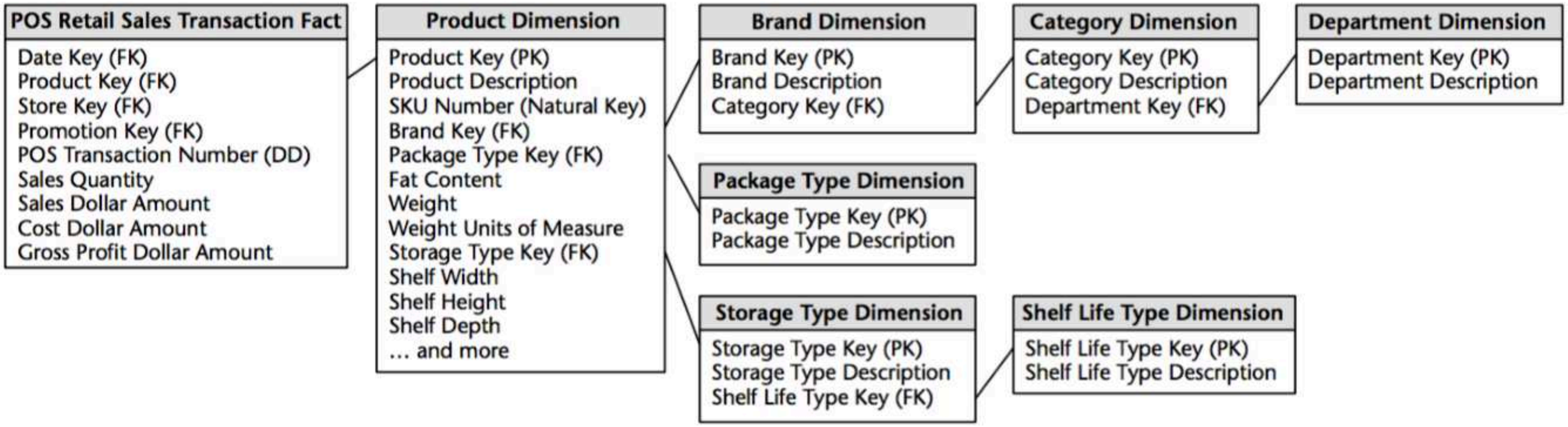
Product Dimension
Product Key (PK)
Product Description
SKU Number (Natural Key)
Brand Key (FK)
Package Type Key (FK)
Fat Content
Weight
Weight Units of Measure
Storage Type Key (FK)
Shelf Width
Shelf Height
Shelf Depth
… and more

Brand Dimension
Brand Key (PK)
Brand Description
Category Key (FK)

Package Type Dimension
Package Type Key (PK)
Package Type Description

Storage Type Dimension
Storage Type Key (PK)
Storage Type Description
Shelf Life Type Key (FK)

Dimension Table Normalization (Snowflaking)



POS Retail Sales Transaction Fact

Date Key (FK)
Product Key (FK)
Store Key (FK)
Promotion Key (FK)
POS Transaction Number (DD)
Sales Quantity
Sales Dollar Amount
Cost Dollar Amount
Gross Profit Dollar Amount

Product Dimension

Product Key (PK)
Product Description
SKU Number (Natural Key)
Brand Key (FK)
Package Type Key (FK)
Fat Content
Weight
Weight Units of Measure
Storage Type Key (FK)
Shelf Width
Shelf Height
Shelf Depth
... and more

Brand Dimension

Brand Key (PK)
Brand Description
Category Key (FK)

Category Dimension

Category Key (PK)
Category Description
Department Key (FK)

Department Dimension

Department Key (PK)
Department Description

Package Type Dimension

Package Type Key (PK)
Package Type Description

Storage Type Dimension

Storage Type Key (PK)
Storage Type Description
Shelf Life Type Key (FK)

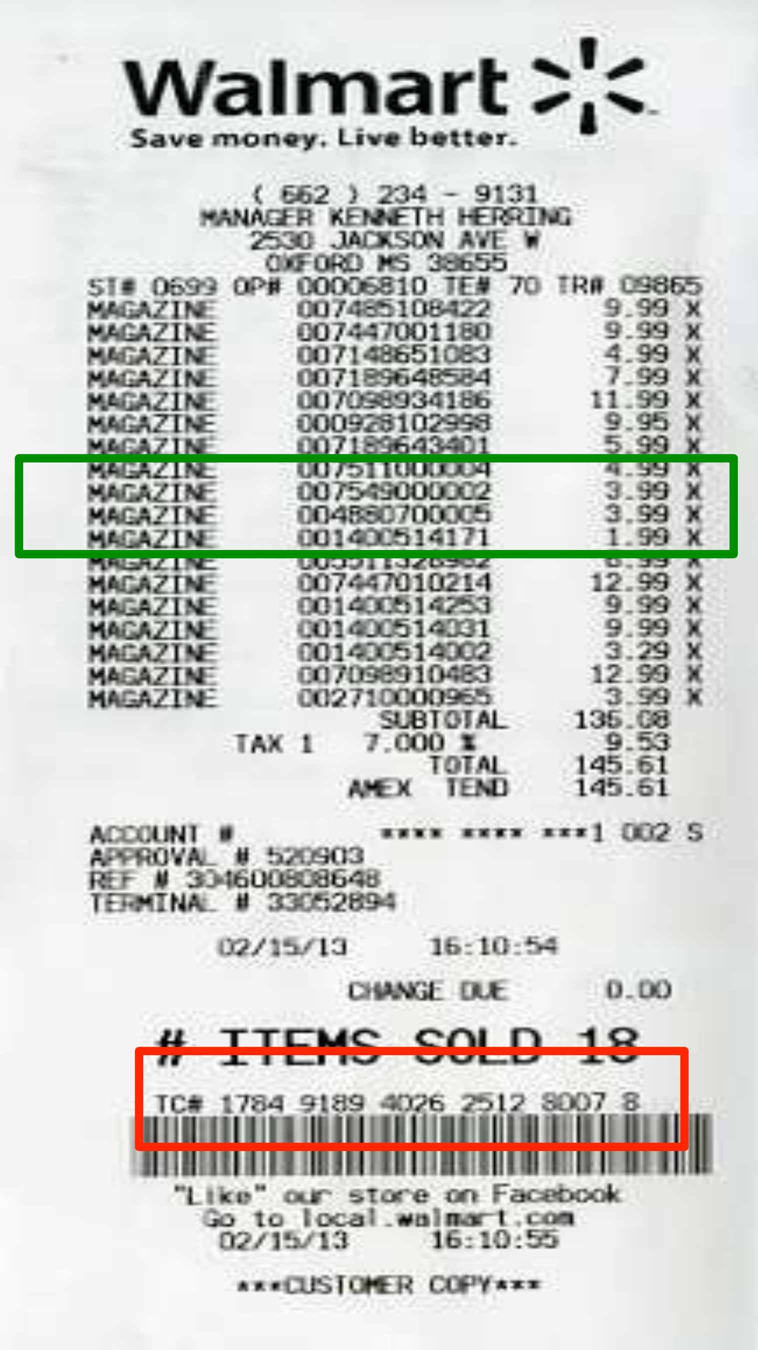
Shelf Life Type Dimension

Shelf Life Type Key (PK)
Shelf Life Type Description

Snowflaking is legal but..

- The dimension tables should remain as flat tables physically.
- Normalized, snowflaked dimension tables **penalize cross-attribute browsing (joins required)** and **prohibit the use of bit-mapped indexe**s (we'll see that later!)
- **Disk space savings** gained by normalizing the dimension tables typically are **less than 1 percent** of the total disk space needed for the overall DW !
- We knowingly sacrifice this dimension table space in the spirit of performance and ease-of-use advantages.

Recognize Degenerate Dimensions

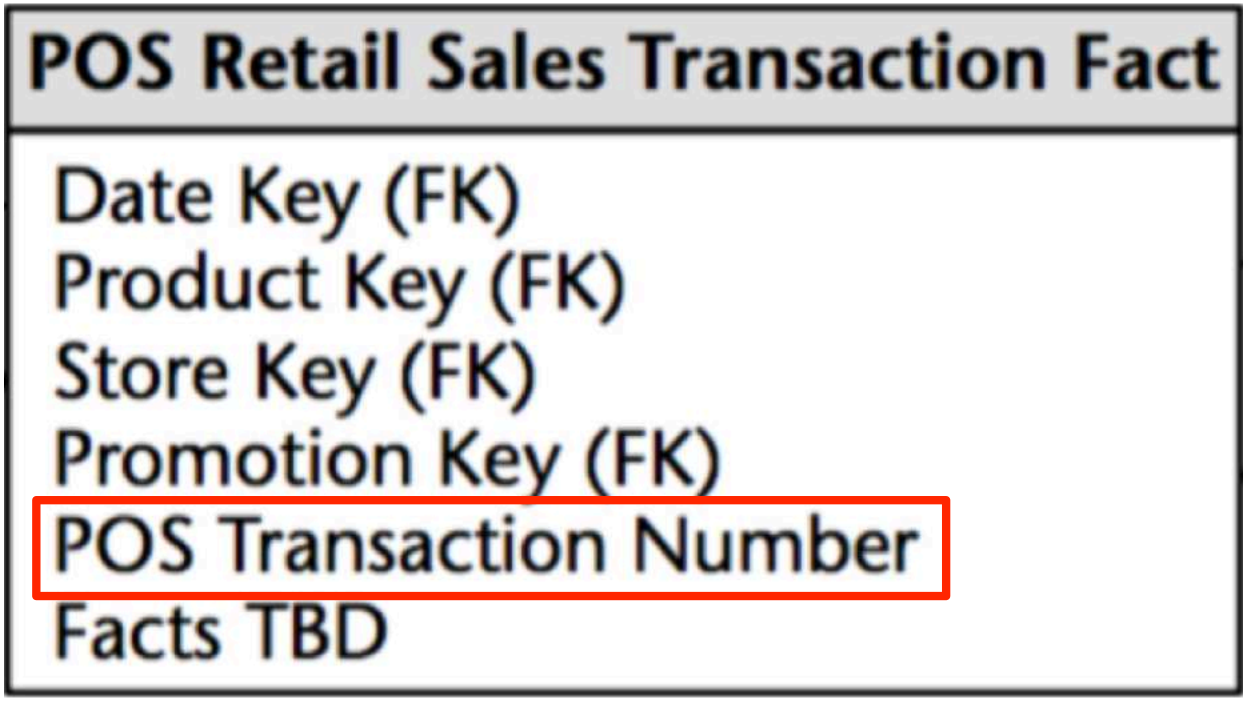


```
Walmart
Save money. Live better.

           ( 662 ) 234 - 9131
      MANAGER KENNETH HERRING
          2530 JACKSON AVE W
           OXFORD MS 38655
ST# 0699 OP# 00006810 TE# 70 TR# 09865
MAGAZINE       007485108422        9.99 X
MAGAZINE       007447001180        9.99 X
MAGAZINE       007148651083        4.99 X
MAGAZINE       007189648584        7.99 X
MAGAZINE       007098934186       11.99 X
MAGAZINE       000928102998        9.95 X
MAGAZINE       007189643401        5.99 X
MAGAZINE       007511000004        4.99 X
MAGAZINE       007549000002        3.99 X
MAGAZINE       004880700005        3.99 X
MAGAZINE       001400514171        1.99 X
MAGAZINE       [illegible]         5.99 X
MAGAZINE       007447010214       12.99 X
MAGAZINE       001400514253        9.99 X
MAGAZINE       001400514031        9.99 X
MAGAZINE       001400514002        3.29 X
MAGAZINE       007098910483       12.99 X
MAGAZINE       002710000965        3.99 X
                     SUBTOTAL    135.08
           TAX 1   7.000 %         9.53
                        TOTAL    145.61
                  AMEX  TEND     145.61

ACCOUNT #          **** **** ***1 002 S
APPROVAL # 520903
REF # 304600808648
TERMINAL # 33052894

          02/15/13      16:10:54

                  CHANGE DUE       0.00

       # ITEMS SOLD 18

     TC# 1784 9189 4026 2512 8007 8

      "Like" our store on Facebook
       Go to local.walmart.com
        02/15/13      16:10:55

         ***CUSTOMER COPY***
```

Candidate granularities

- **By item type**
  - **Precise analysis** of each item type in a purchase
- **By transaction**
  - Just a **summary** of the purchase **(#items, total)**
  - Limited business analysis value
- **By item** (**too fine,** here brings no benefits)

We pick the first



POS Retail Sales Transaction Fact
Date Key (FK)
Product Key (FK)
Store Key (FK)
Promotion Key (FK)
POS Transaction Number
Facts TBD

Transaction number is a Degenerate Dimension

- **It is a dimension because it caracterizes the item sold**

- But, although the transaction number may look like a dimension key in the fact table …

- … all the descriptive items of a transaction are in the other dimensions or in the fact table

- Transaction is thus an **empty dimension** that we refer as *degenerate dimension* (DD)

Degenerate = Empty (no attributes)

- Transaction number serves as grouping key for all the products purchased in a single transaction
  - **find items sold together**
  - **compute total amount of a purchase**

- **Order numbers**, **invoice numbers**, almost always appear as degenerate dimensions

- Finally, degenerate dimensions often play an integral role in the fact table's primary key

Use Surrogate Keys

Surrogate Keys

- Business analysts may want to navigate the fact table based on the **operational code** of a product avoiding a join to the dimension table. Eg **2014FR2510d**
- This is not a good idea, as the dimension tables are the entry point of the system
- Surrogate keys are integers that are assigned sequentially as needed to populate a dimension.
- Example Product 1,2,3,4,

Surrogate Keys

- *Every join between dimension and fact tables in the data warehouse should be based on* ***meaningless integer surrogate keys****.*

- Avoid using the natural operational production codes. None of the data warehouse keys should be smart, where you can tell something about the row just by looking at the key.

- This gives better performances

The Art of Designing a Datawarehouse : the Retail Case Part 2

Retail Case Study : Grocery Chain

- 100 stores spread over a five-state area
- ~60,000 individual products on a store
- 80% come from outside manufacturers

Retail Case Study : Grocery Chain

Data is collected at

- cash registers as customers purchase products
- the back door, where vendors make deliveries

- Sales are much more important than deliveries
  - This is why we treated it first !
  - Now, we can move on.

Inventory

Having products at the right store at the right time :

- minimizes out-of-stocks (sale more)
- reduces overall inventory carrying costs

Inventory Models

- Inventory comes after sales, in terms of importance.
- A company is likely to invest less resources on the analysis of these data.
- **Less resources** = **less detailed information** in the datawarehouse

Inventory Models

1. Periodic Snapshot

2. Transaction-grain

3. Updated Records

PERIODIC SNAPSHOT

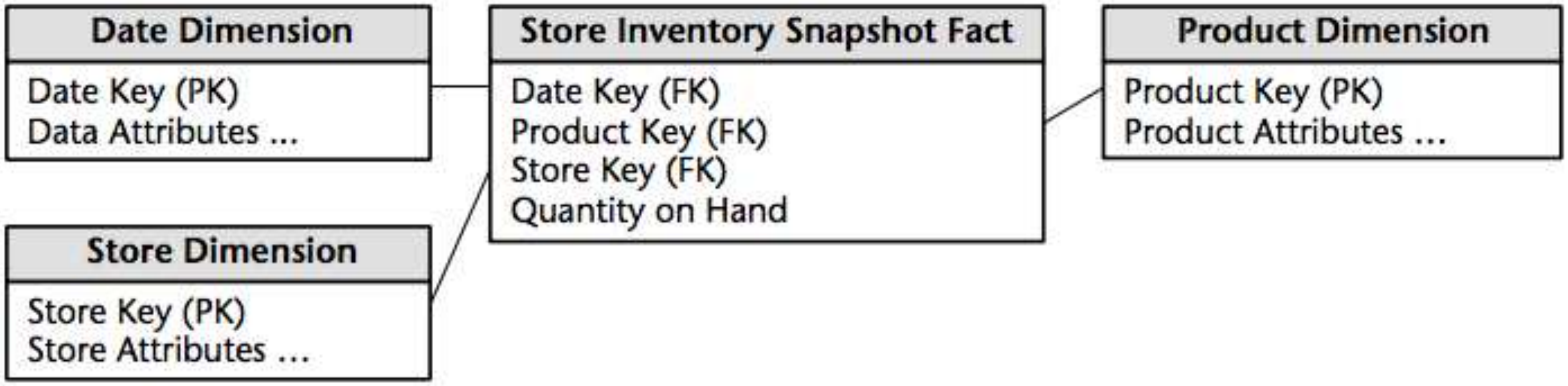
Periodic Snapshot

- **Regularly** record the **full state of the inventory**
  - every day
  - or every week
  - or every 2 hours (etc...)

- Example : record a summary of the status of the inventory at the end of each day

- **Granularity** : higher than real-life actions
  - but should be just right for profitable analysis

Periodic Snapshot Schema



Date Dimension

Date Key (PK)
Data Attributes ...

Store Inventory Snapshot Fact

Date Key (FK)
Product Key (FK)
Store Key (FK)
Quantity on Hand

Product Dimension

Product Key (PK)
Product Attributes ...

Store Dimension

Store Key (PK)
Store Attributes ...

Snapshot Fact Table

date	product	store	quantity
1	21	1	11
1	21	2	65
1	21	3	2332
1	21	4	53

Snapshot Fact Table

date	product	store	quantity
1	21	1	11
1	21	2	65
1	21	3	2332
1	21	4	53
1	31	1	234
1	31	2	23
1	31	3	4332
1	31	4	66

Snapshot Fact Table

date	product	store	quantity
1	21	1	11
1	21	2	65
1	21	3	2332
1	21	4	53
1	31	1	234
1	31	2	23
1	31	3	4332
1	31	4	66
2	21	1	33
2	21	2	234
2	21	3	44
2	21	4	22
2	31	1	44
2	31	2	544
2	31	3	445
2	31	4	22

Snapshot Fact Table

date	product	store	quantity
1	21	1	11
1	21	2	65
1	21	3	2332
1	21	4	53
1	31	1	234
1	31	2	23
1	31	3	4332
1	31	4	66
2	21	1	33
2	21	2	234
2	21	3	44
2	21	4	22
2	31	1	44
2	31	2	544
2	31	3	445
2	31	4	22

Inconvenient : dense snapshot tables

- Dense means: 1 row for each (product,store,day)

  - 60K products * 100 stores = 6M lines

  - assume 1 row (previous table) = 14 bytes
  ➔ 30 GB/year

- Compromise : reduce snapshot frequency
  daily (for last 60 days) weekly (for hystorical data)
  - question : how much storage on 12 months ?
  - always estimate the size of your tables

Now, which analytical queries can we answer ?

« today overall quantity of a given product »

« today overall quantity on a given store »

« overal quantity of a given product in july »

Now, which analytical queries can we answer ?

« today overall quantity of a given product » OK

« today overall quantity on a given store » OK

« overal quantity of a given product in july » NO

Semiadditive facts

Q : « overal quantity of a given product in july » NO

- Why ? Cannot SUM inventory levels !

July 1st	:	10010	product 21 store 1
July 2nd	:	13016	product 21 store 1
July 3rd	:	19016	product 21 store 1

…

- Semiadditive facts : facts that are additive across some dimensions, but not all
  – store-dim, product-dim are ok, time is not ok!

Semiadditive facts

M	**T**	**W**	**T**	**F**
$50	**$50**	**$100**	**$100**	**$100**

- On Monday have $50, on Tuesday no deposit. Deposit $50 on Wednesday, then no actions.

- Friday night : cannot pretend we have $**400.**

Semiadditive facts

Q: «average bank account weekly balance» YES

M	T	W	T	F
$50	$50	$100	$100	$100

$80

Semiadditivity does not exclude mean

  - At 10am we have 10deg
  - At 11am we have 12deg
  - At 12am we have 15deg

- During last two hours AVG deg temp.was 12.3

Quiz

- Un fait (j, p, c, m, x) existe lorsque
  - un produit p
  - est acheté par un client c
  - le jour j
  - au magasin m
- La mesure x correspond au prix total.

- Fait snapshot ou transactionnel ?

Fait 1 : si Snapshot

- toutes les combinaisons (produit,client,date,magasin)
- intervalles reguliers (chaque jour)

id_produit	**id_client**	**id_date**	**id_magasin**	**prix_tot_vente**
1	1	**1**	1	6565
1	1	**1**	2	0
1	2	**1**	1	0
1	2	**1**	2	45654
2	1	**1**	1	0
2	1	**1**	2	0
2	2	**1**	1	0
2	2	**1**	2	0

Fait 1 : si Snapshot

- toutes les combinaisons (produit,client,date,magasin)
- intervalles reguliers (chaque jour)

id_pr duit
1
1
1
1
2
2
2
2

id_produit	id_client	id_date	id_magasin	prix_tot_vente
1	1	**2**	1	0
1	1	**2**	2	0
1	2	**2**	1	0
1	2	**2**	2	0
2	1	**2**	1	0
2	1	**2**	2	0
2	2	**2**	1	0
2	2	**2**	2	0

Fait 1 : si Snapshot

- toutes les combinaisons (produit,client,date,magasin)
- intervalles reguliers (chaque jour)

id_pr duit
1
1
1
1
2
2
2
2

id_pro duit
1
1
1
1
2
2
2
2

id_produit	id_client	id_date	id_magasin	prix_tot_vente
1	1	**3**	1	0
1	1	**3**	2	111
1	2	**3**	1	0
1	2	**3**	2	0
2	1	**3**	1	0
2	1	**3**	2	0
2	2	**3**	1	188

Fait 1 : si Transactionnel

- Tous les achats

id_produit	id_client	id_date	id_magasin	prix_tot_vente
1	1	1	1	6565
1	2	1	2	45654
1	1	3	2	111
2	2	3	1	188

Comparaison

- Quelques valeurs possibles
  - 60K produits
  - 10K clients au programme fidélité
  - 30 jours
  - 100 magasins
  - 100 produits achetés par semaine

- Transactionnel = 400**M** lignes * mois
- Snapshot = 1.8**T** lignes * mois (dont +99% à 0)

RECORD TRANSACTIONS

Record Transactions

The «expensive» solution (like, Amazon)

Record every transaction that affects inventory

1. Receive product
2. Place product into inspection hold
3. Release product from inspection hold
4. Return product to vendor due to inspection failure
5. Place product in bin
6. Authorize product for sale

Record Transactions

The «expensive» solution (like, Amazon)

Record every transaction that affects inventory

7. Pick product from bin
8. Package product for shipment
9. Ship product to customer
10. Receive product from customer
11. Return product to inventory from customer return
12. Remove product from inventory

Record Transactions

- Needs a special dimension for **transaction-type**

- Other dimensions : product, order, status, date

Receive

Product_id	Order_id	Transaction_type	Status	Date
1	**No_order**	Receive product	**COMPLETED**	**2015/12/11**

Inspect

Product_id	**Order_id**	**Transaction_type**	**Status**	**Date**
1	**No_order**	Receive product	**COMPLETED**	**2015/12/11**
1	**No_order**	Inspection Hold	**COMPLETED**	**2015/12/11**

Ask for authorization

Product_id	Order_id	Transaction_type	Status	Date
1	**No_order**	Receive product	**COMPLETED**	**2015/12/11**
1	**No_order**	Inspection Hold	**COMPLETED**	**2015/12/11**
1	**No-order**	Authorized for sale	**PENDING**	**2015/12/11**

Authorize

Product_id	Order_id	Transaction_type	Status	Date
1	**No_order**	Receive product	**COMPLETED**	**2015/12/11**
1	**No_order**	Inspection Hold	**COMPLETED**	**2015/12/11**
1	**No-order**	Authorized for sale	**PENDING**	**2015/12/11**
1	**No-order**	Authorized for sale	**COMPLETED**	**2015/12/12**

Ship

Product_id	Order_id	Transaction_type	Status	Date
1	**No_order**	Receive product	**COMPLETED**	**2015/12/11**
1	**No_order**	Inspection Hold	**COMPLETED**	**2015/12/11**
1	**No-order**	Authorized for sale	**PENDING**	**2015/12/11**
1	**No-order**	Authorized for sale	**COMPLETED**	**2015/12/12**
1	**20**	Ship to Customer	**COMPLETED**	**2016/12/01**

Record Transaction

- When selling thousands of items of the same type : even more cumbersome than before
- But detailed…

Can answer analytical queries that periodic snapshots can't

- How many separate shipments did we receive from a given vendor?

- On which products have we had more than one round of inspection failures ?

UPDATED RECORDS

(previous) Record Transactions

1 fact table row = 1 movement of 1 product

- Movement : receiving, inspection, bin placement, authorization to sell, shipping

Updated Records

1 fact table row = **ALL** movements of 1product

- **UPDATE the fact table row over and over** until the product leaves the warehouse
  - – Eg., shipping can take values : { NO, pending, OK }

Updated Records

Product_id	Order_id	Status_id	Boxing_id	Shipping_date_id
1	10	INVALID	Package_1	undefined
2	10	INVALID	Package_2	undefined

Updated Records

Product_id	Order_id	Status_id	Boxing_id	Shipping_date_id
1	10	INVALID	Package_3	undefined
2	10	INVALID	Package_3	undefined

Updated Records

Product_id	Order_id	Status_id	Boxing_id	Shipping_date_id
1	10	VALIDATED	Package_3	undefined
2	10	VALIDATED	Package_3	undefined

Updated Records

Product_id	Order_id	Status_id	Boxing_id	Shipping_date_id
1	10	VALIDATED	Package_3	2015/17/12
2	10	VALIDATED	Package_3	2015/17/12

Updated Records

Product_id	Order_id	Status_id	Boxing_id	Shipping_date_id
1	10	VALIDATED	Package_3	2018/17/12
2	10	VALIDATED	Package_3	2018/17/12
3	20	WAITING	Package_1	undefined
4	20	WAITING	Package_1	undefined

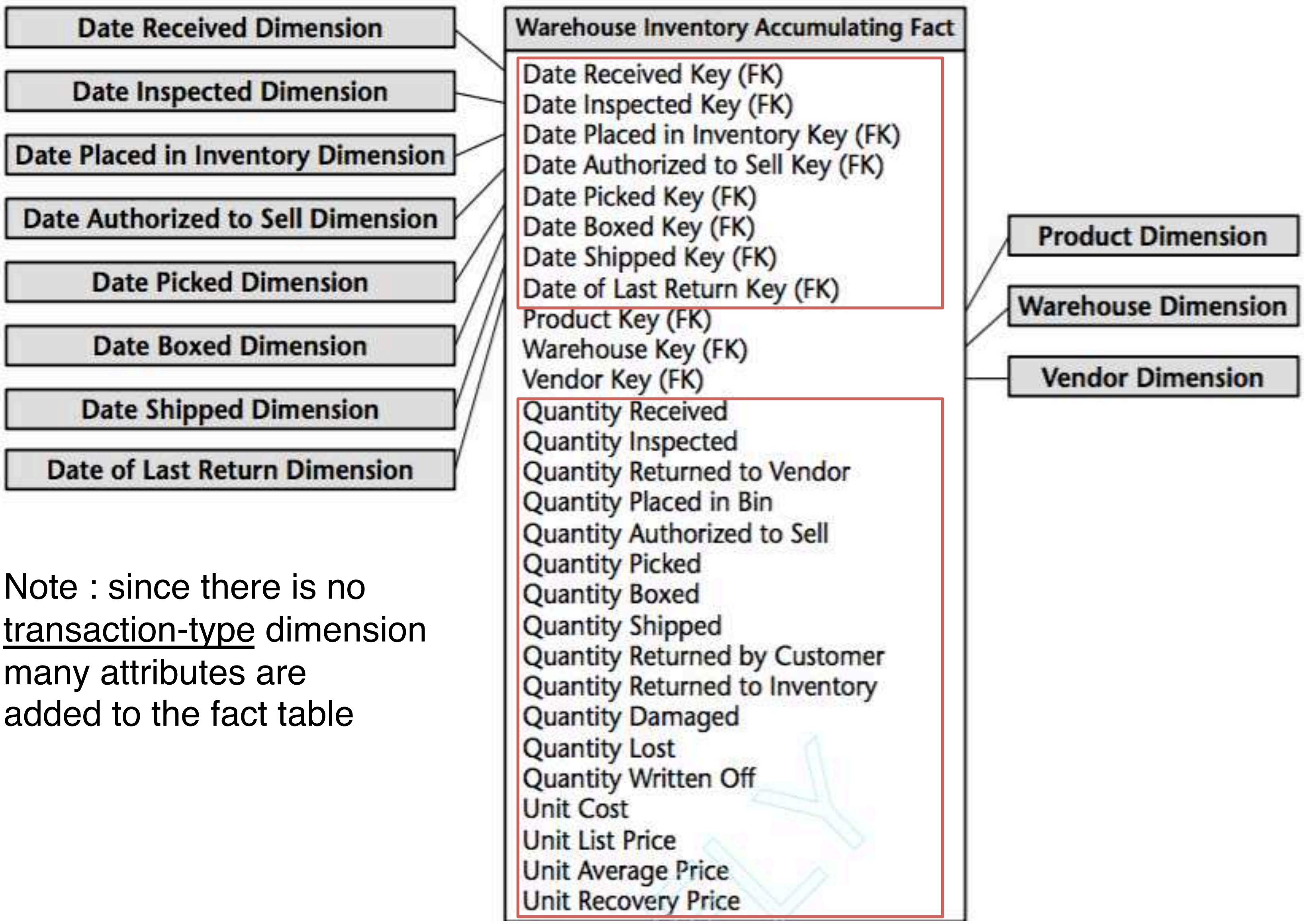
Updated Records

Product_id	Order_id	Status_id	Boxing_id	Shipping_date_id
1	10	VALIDATED	Package_3	2018/17/12
2	10	VALIDATED	Package_3	2018/17/12
3	20	VALIDATED	Package_1	undefined
4	20	VALIDATED	Package_1	undefined

Updated Records

Product_id	Order_id	Status_id	Boxing_id	Shipping_date_id
1	10	VALIDATED	Package_3	2018/17/12
2	10	VALIDATED	Package_3	2018/17/12
3	20	VALIDATED	Package_1	2019/03/01
4	20	VALIDATED	Package_1	2019/03/01

In reality : a much more complex schema



Date Received Dimension

Date Inspected Dimension

Date Placed in Inventory Dimension

Date Authorized to Sell Dimension

Date Picked Dimension

Date Boxed Dimension

Date Shipped Dimension

Date of Last Return Dimension

Warehouse Inventory Accumulating Fact

Date Received Key (FK)
Date Inspected Key (FK)
Date Placed in Inventory Key (FK)
Date Authorized to Sell Key (FK)
Date Picked Key (FK)
Date Boxed Key (FK)
Date Shipped Key (FK)
Date of Last Return Key (FK)
Product Key (FK)
Warehouse Key (FK)
Vendor Key (FK)
Quantity Received
Quantity Inspected
Quantity Returned to Vendor
Quantity Placed in Bin
Quantity Authorized to Sell
Quantity Picked
Quantity Boxed
Quantity Shipped
Quantity Returned by Customer
Quantity Returned to Inventory
Quantity Damaged
Quantity Lost
Quantity Written Off
Unit Cost
Unit List Price
Unit Average Price
Unit Recovery Price

Product Dimension

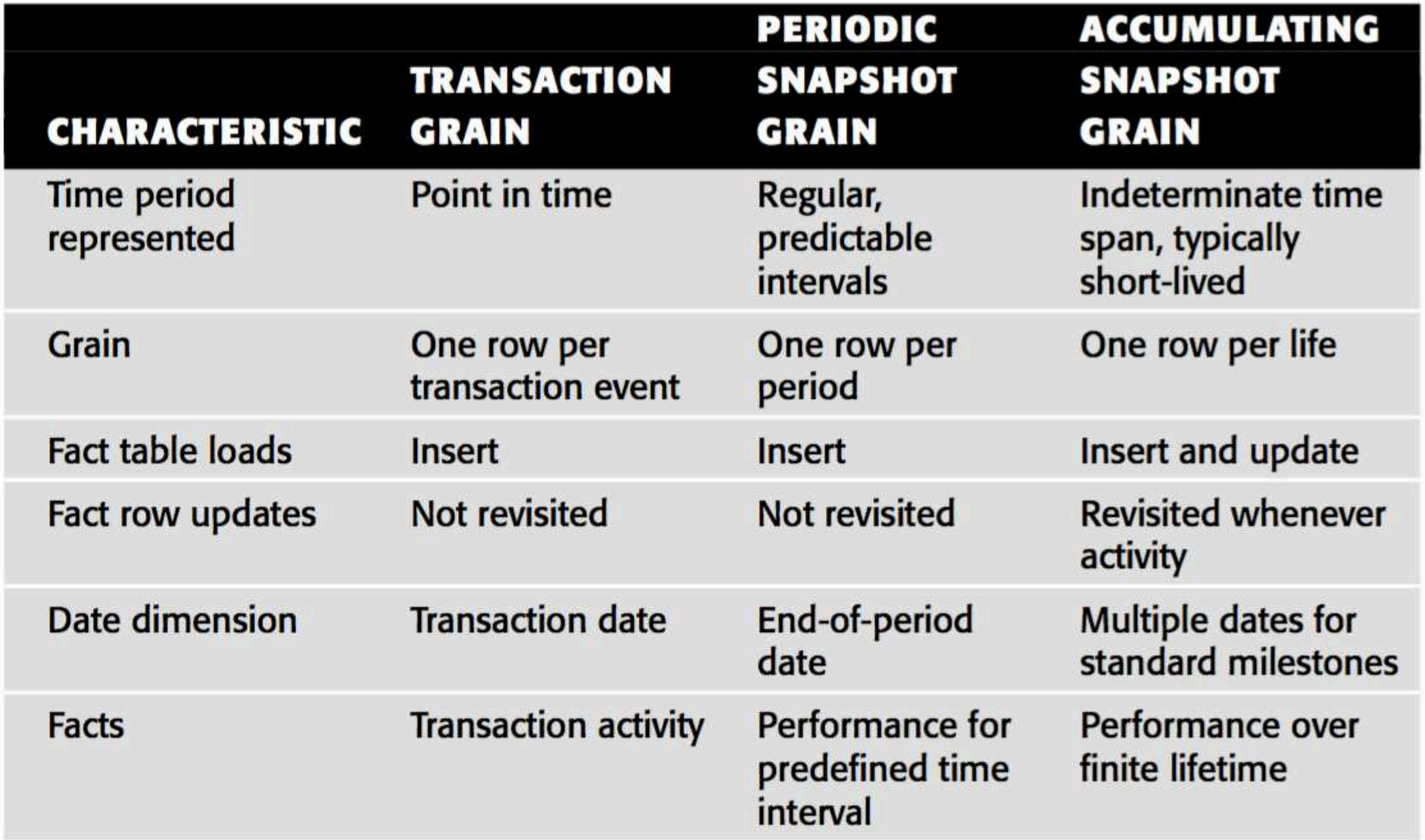
Warehouse Dimension

Vendor Dimension

Note : since there is no <u>transaction-type</u> dimension many attributes are added to the fact table

Summing up : types of fact tables

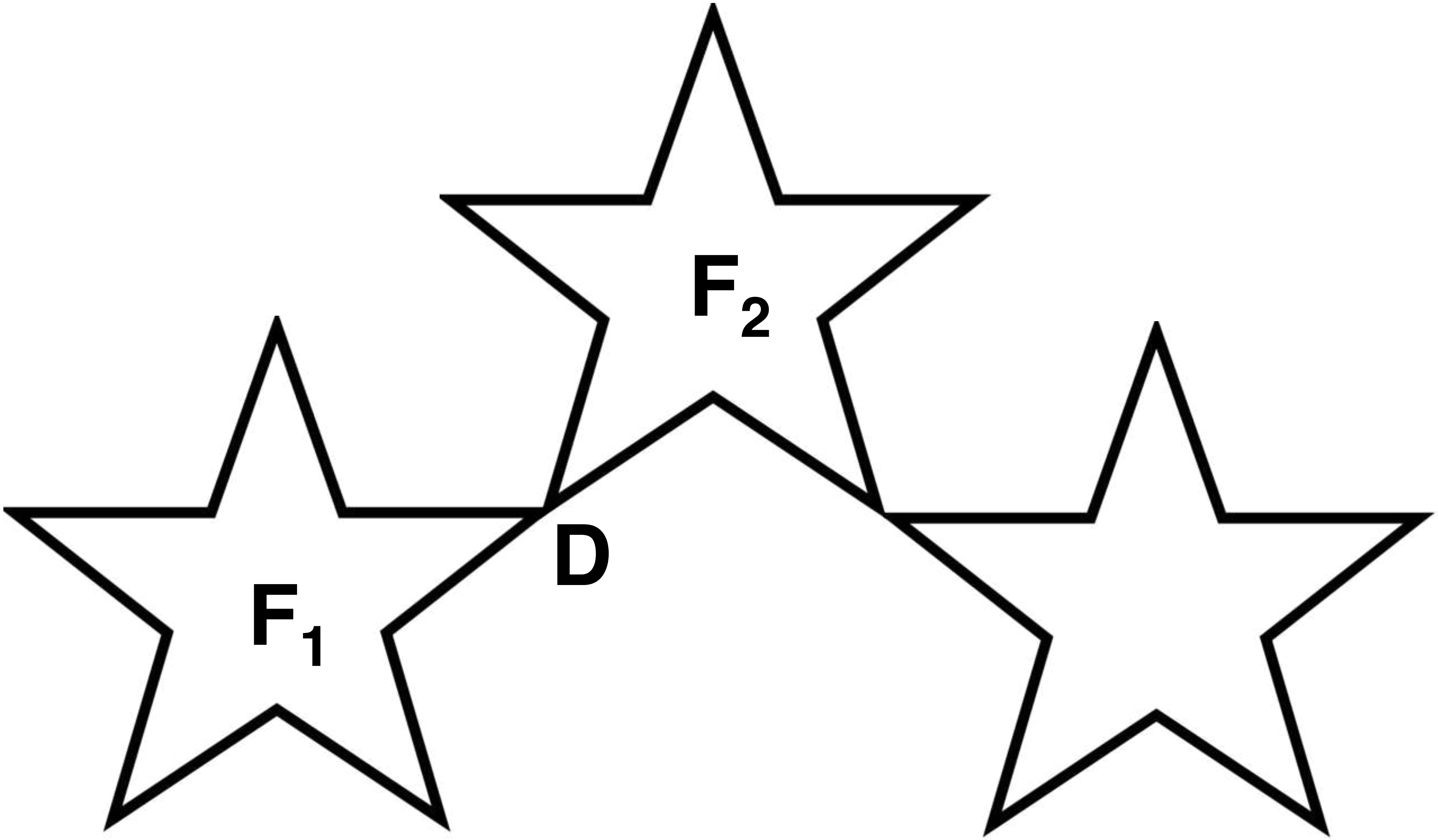
updated records



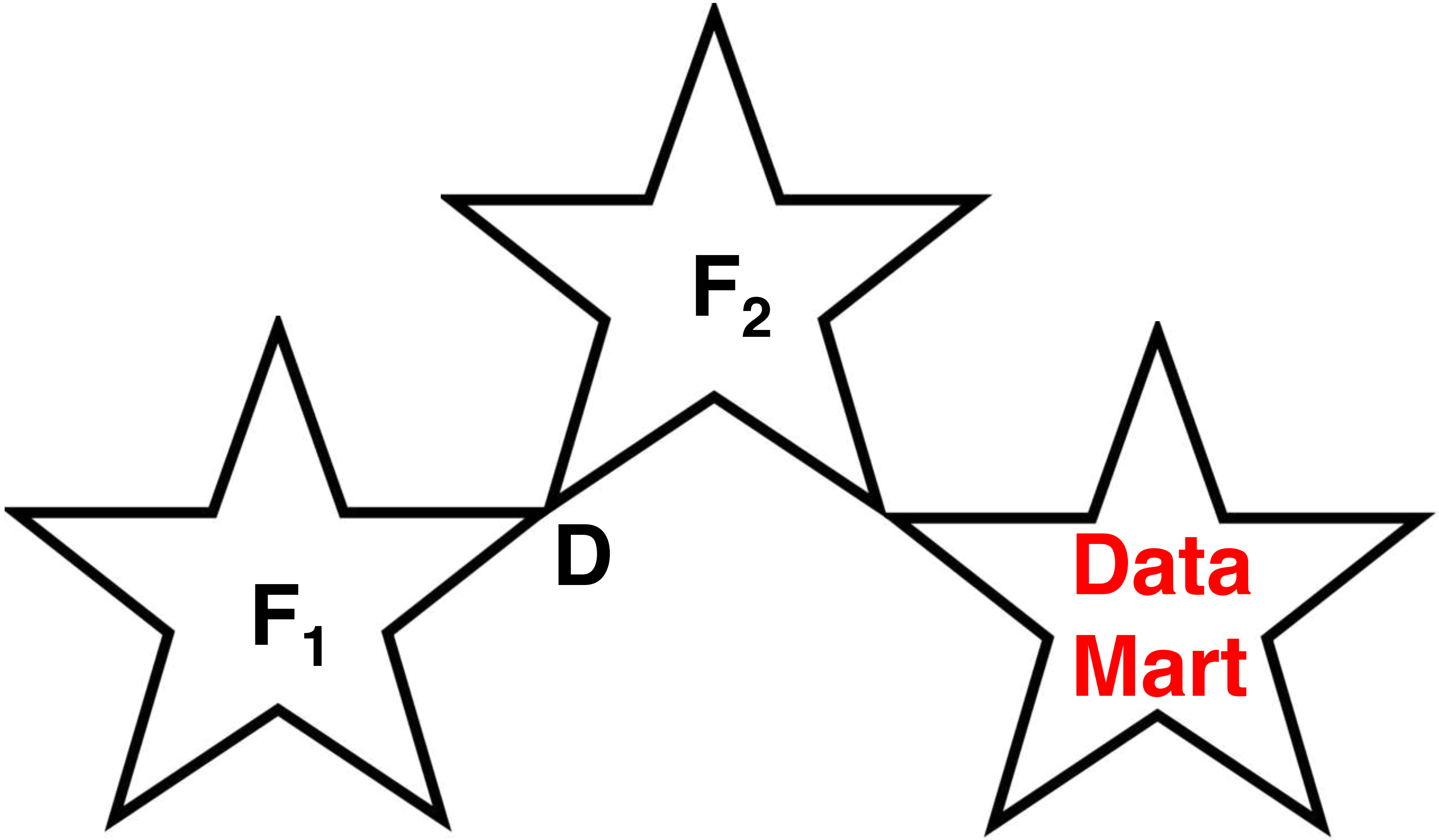
		PERIODIC	ACCUMULATING
	TRANSACTION	SNAPSHOT	SNAPSHOT
CHARACTERISTIC	GRAIN	GRAIN	GRAIN
Time period represented	Point in time	Regular, predictable intervals	Indeterminate time span, typically short-lived
Grain	One row per transaction event	One row per period	One row per life
Fact table loads	Insert	Insert	Insert and update
Fact row updates	Not revisited	Not revisited	Revisited whenever activity
Date dimension	Transaction date	End-of-period date	Multiple dates for standard milestones
Facts	Transaction activity	Performance for predefined time interval	Performance over finite lifetime

Sharing Dimensions : beyond star schemas

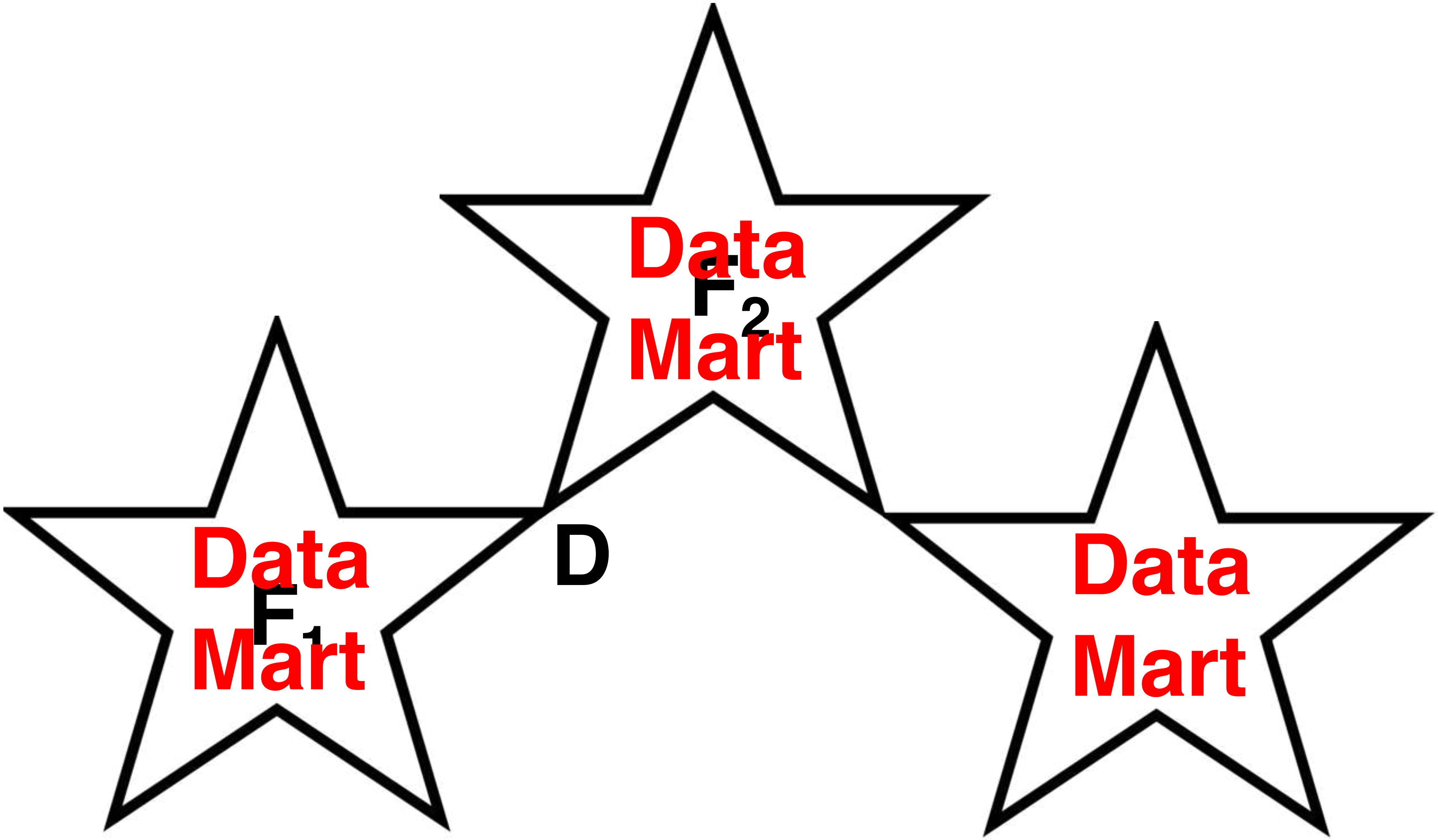
DW Model : Constellation Schemas



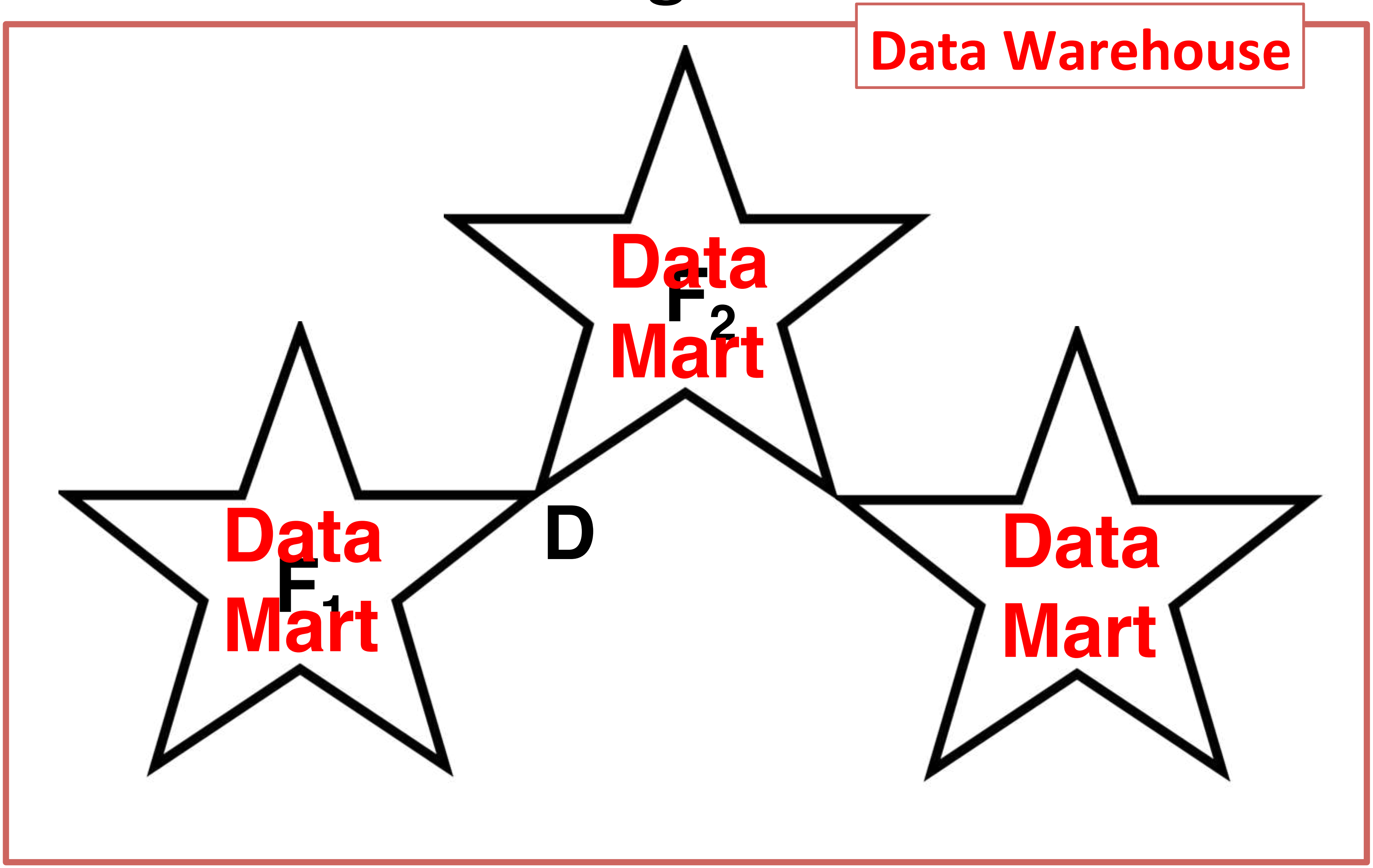
Data Mart : Single Fact-table DW



Data Mart : Single Fact-table DW



Data Mart : Single Fact-table DW



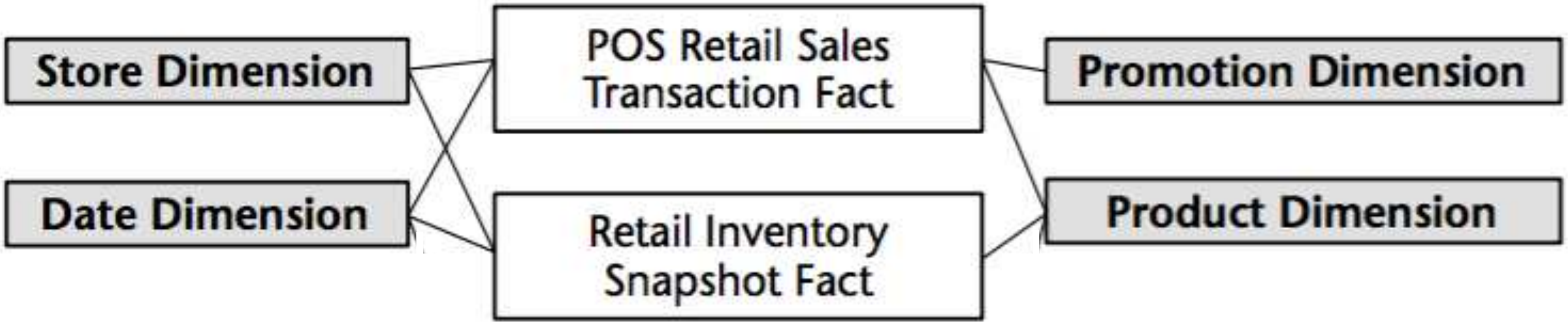
Data mart

- Business oriented star-schema (or few of them)

- Virtual (users have views on data) or Materialized (different database instances; more expensive)

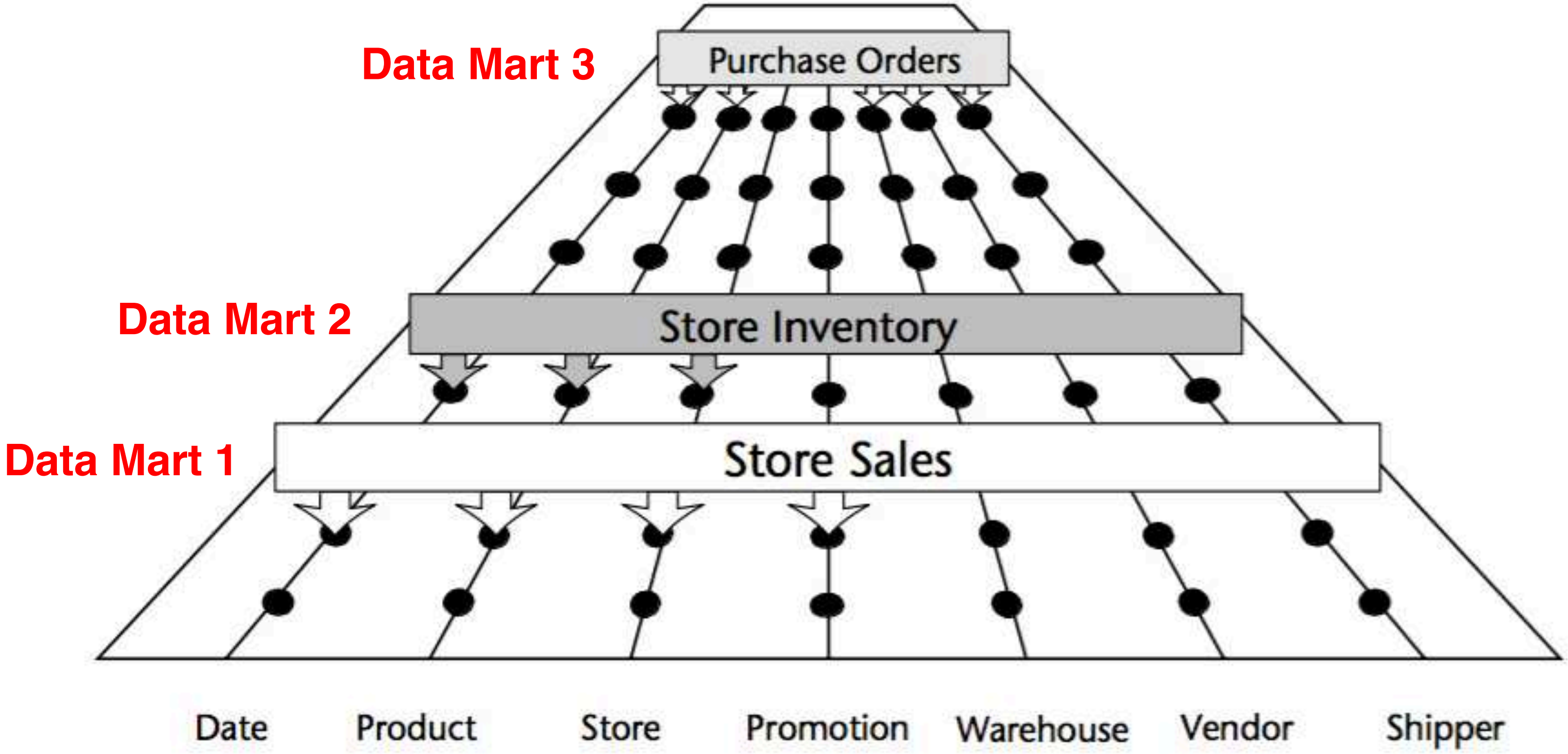
- Important notion also for privacy reasons : data should not be visible to any user !

Sharing Dimensions



- Star-Constellation Schema **!=** Snowflake Schema

Shared Dimensions



Sharing Dimensions : Which Level of Detail ?

- *Date* dimension identical for **retail** and **inventory**
  - also *product* and *store*

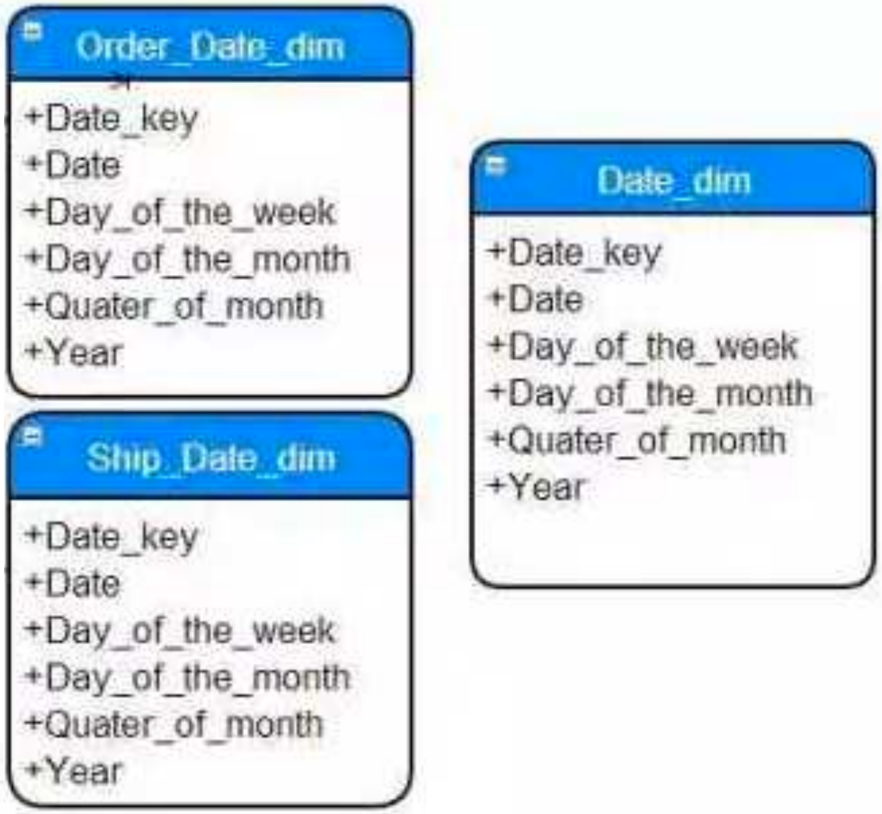
- Use the **more detailed** version of the dimension !

  - Previously defined tables for retail **may be not enough detailed** and lack of attributes useful for inventory analysis

  - Eg product dimension: <u>minimum reorder quantity</u>
  - Eg store dimension : <u>storage square footages</u>

Sharing Date Dimension

- Problem : order date and shipping date **both** dates
- Better to specify order and ship date dimension



Order_Date_dim

+Date_key
+Date
+Day_of_the_week
+Day_of_the_month
+Quater_of_month
+Year

Ship_Date_dim

+Date_key
+Date
+Day_of_the_week
+Day_of_the_month
+Quater_of_month
+Year

Date_dim

+Date_key
+Date
+Day_of_the_week
+Day_of_the_month
+Quater_of_month
+Year

Create illusion of independent date-tables using (virtual) views

-
  ```
  CREATE VIEW ORDER_DATE
  AS SELECT * FROM DATE
  ```

-
  ```
  CREATE VIEW SHIP_DATE
  AS SELECT * FROM DATE
  ```

Create illusion of independent date-tables using (virtual) views

- CREATE VIEW ORDER_DATE AS SELECT $X_1, \ldots, X_n$ FROM DATE

  more detailed

- CREATE VIEW SHIP_DATE AS SELECT $X_1, \ldots, X_{m<n}$ FROM DATE

  less detailed