## Why Data Warehouses?

- A “style” or “model” of a Relational Database
  - Designed for a very specific purpose
  - Like “SUV” versus “Truck” versus “Luxury Sedan” versus “Racing”
- Designed for very large amounts of historical data
- Designed for analytical queries (DSS) rather than transactional (OLTP) queries

**Consider the Classic Models database (from HW # 3)**

- Database was designed for OLTP
  - Add/Change/Delete transactions
  - Keeping track of various entities related to their business
  - Queries:
    - What's my inventory in stock quantity of [….] right now?
    - From which suppliers can I buy [….] product?
    - What is a customer's current balanced owed?
    - Which materials are below the reorder point?

## Consider Classic Models

- Database was NOT designed for DSS queries
  - Over the past 5 years how are sales of [….] trending?
  - What is our YTD profitability for sales of XXX product by zip code within a 30 mile radius?
  - Which employees yield the highest margins for sales since January 1st this year? Rank them.
  - Which models have the lowest margins and the lowest sales. Should we discontinue them?

## The Data Warehouse

- Designed for DSS queries
  - Looking at trends over time
  - Supports strategic decision making
  - Large amounts of data
  - History kept forever

**SO – the 3NF design works well for OLTP databases, but NOT for DSS/OLAP databases !**

**The Data Warehouse**

- Queries must process massive amounts of data as efficiently as possible
- Lots of indexes
- Bulk Loads nightly rather than add/change/delete transactions throughout the day
- No or few updates
- Typically WORM – “Write Once Read Many”

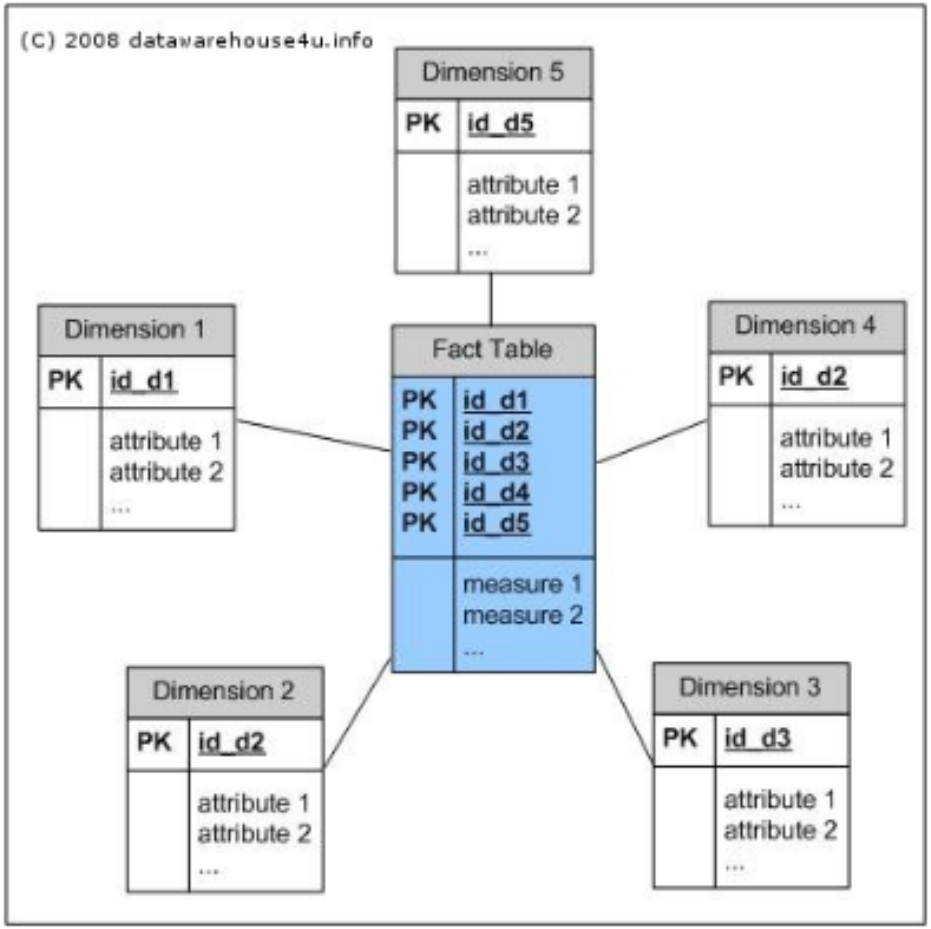
- Probably use a BI reporting/analytics tool against the data in the warehouse

**The Data Warehouse**

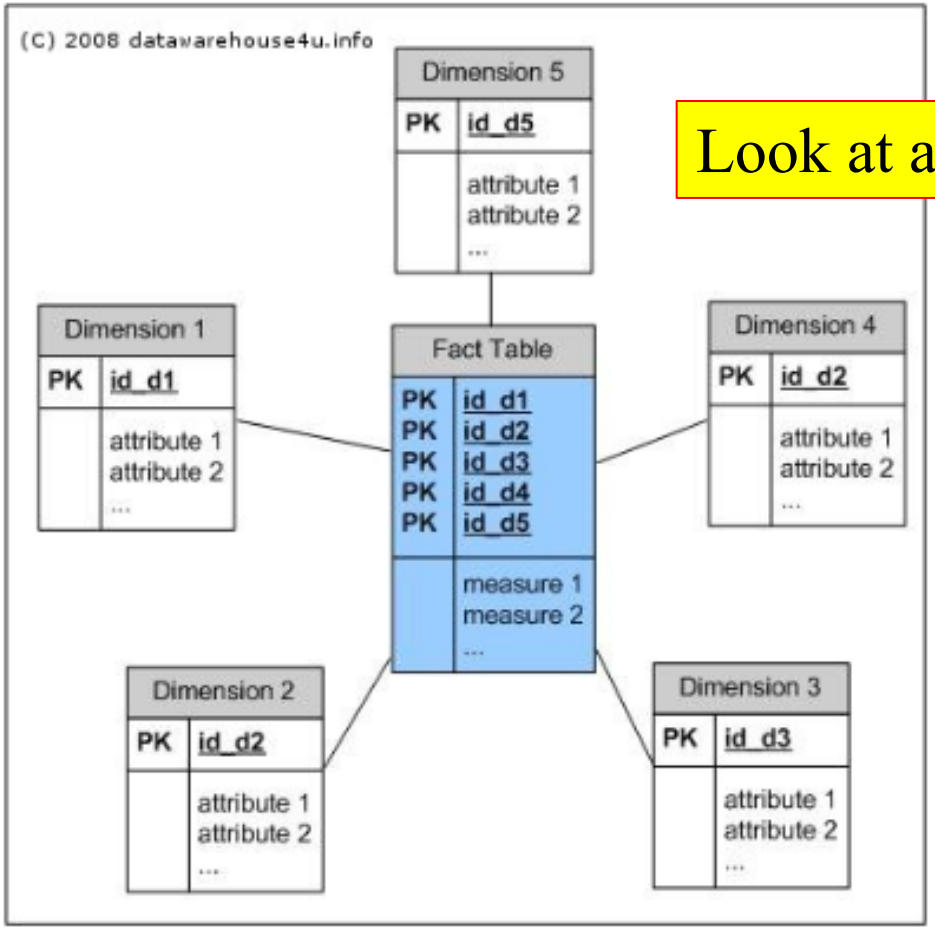
Why use a BI Tool?

- Typical analytical queries will require a lot of joins.
- The SQL gets rather complex due to the relatively high number of joins.

The Data Warehouse is modeled in a “Star Schema” in a “Dimensional Model”, NOT in typical 3rd Normal Form

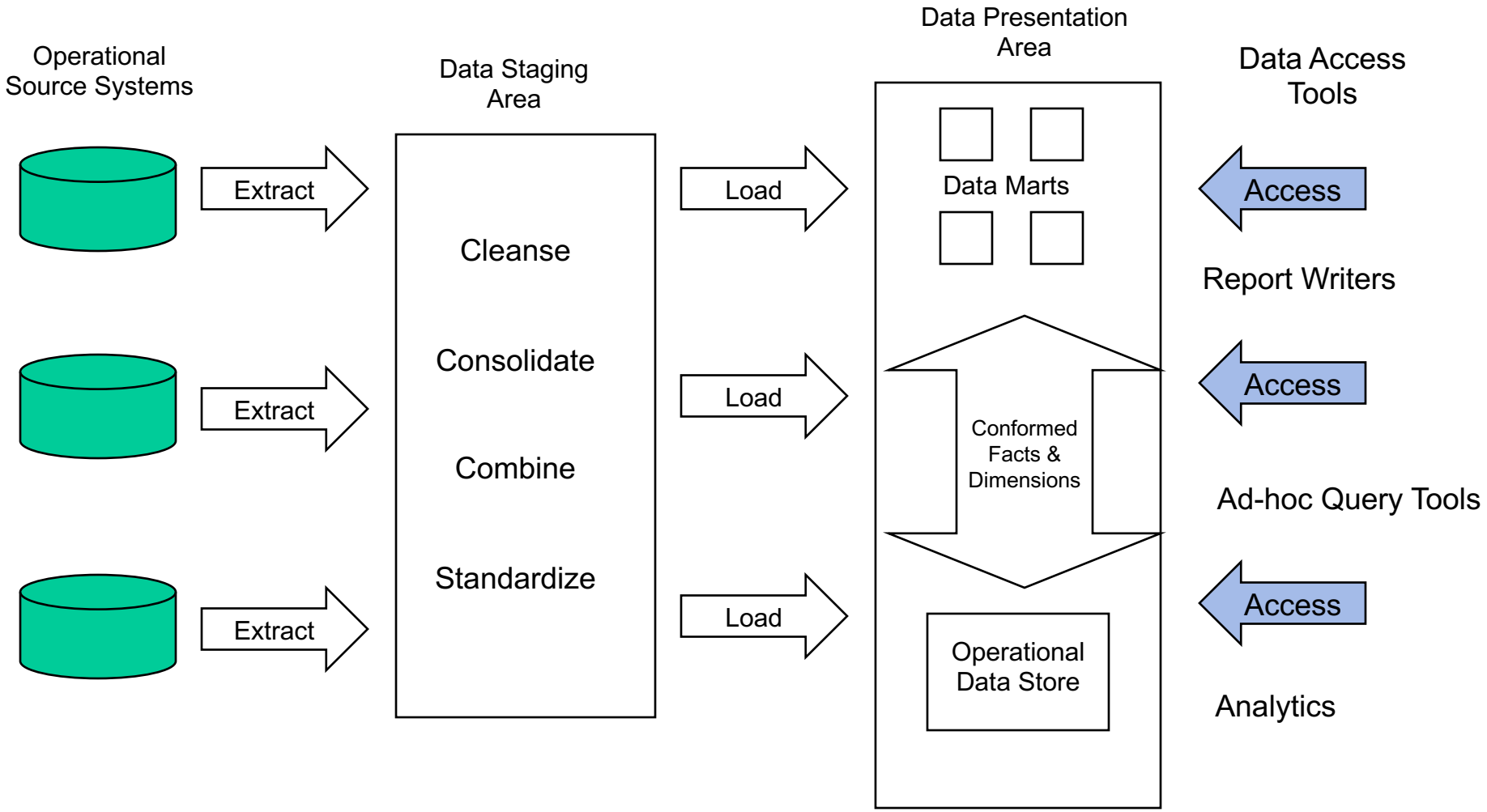


The Data Warehouse is modeled in a “Star Schema” in a “Dimensional Model”, NOT in typical 3rd Normal Form




Look at all those keys !!

# Data Warehousing



- **Data Warehouse Components**
  - Operational Source System
  - Data Staging Area
  - Data Presentation Area
  - Data Access Tools
  - Metadata
  - Operational Data Store

- **Extract Transform Load ("ETL")**

  – Software that
    - Retrieves data from Operational Systems
    - Stages data in temporary databases
    - Cleanses and standardizes the data
    - Provides metadata regarding data sources

https://www.glassdoor.com/Salaries/etl-developer-salary-SRCH_KO0,13.htm

- **ETL Tools**

https://www.etltool.com/list-of-etl-tools/

Why?

- Dirty data
- Missing values
- Inconsistent data
- Data not integrated
- Wrong format, wrong level of detail
  - Too fine
  - Not fine enough
- Too much data
  - Too many attributes
  - Too much volume -- summarize

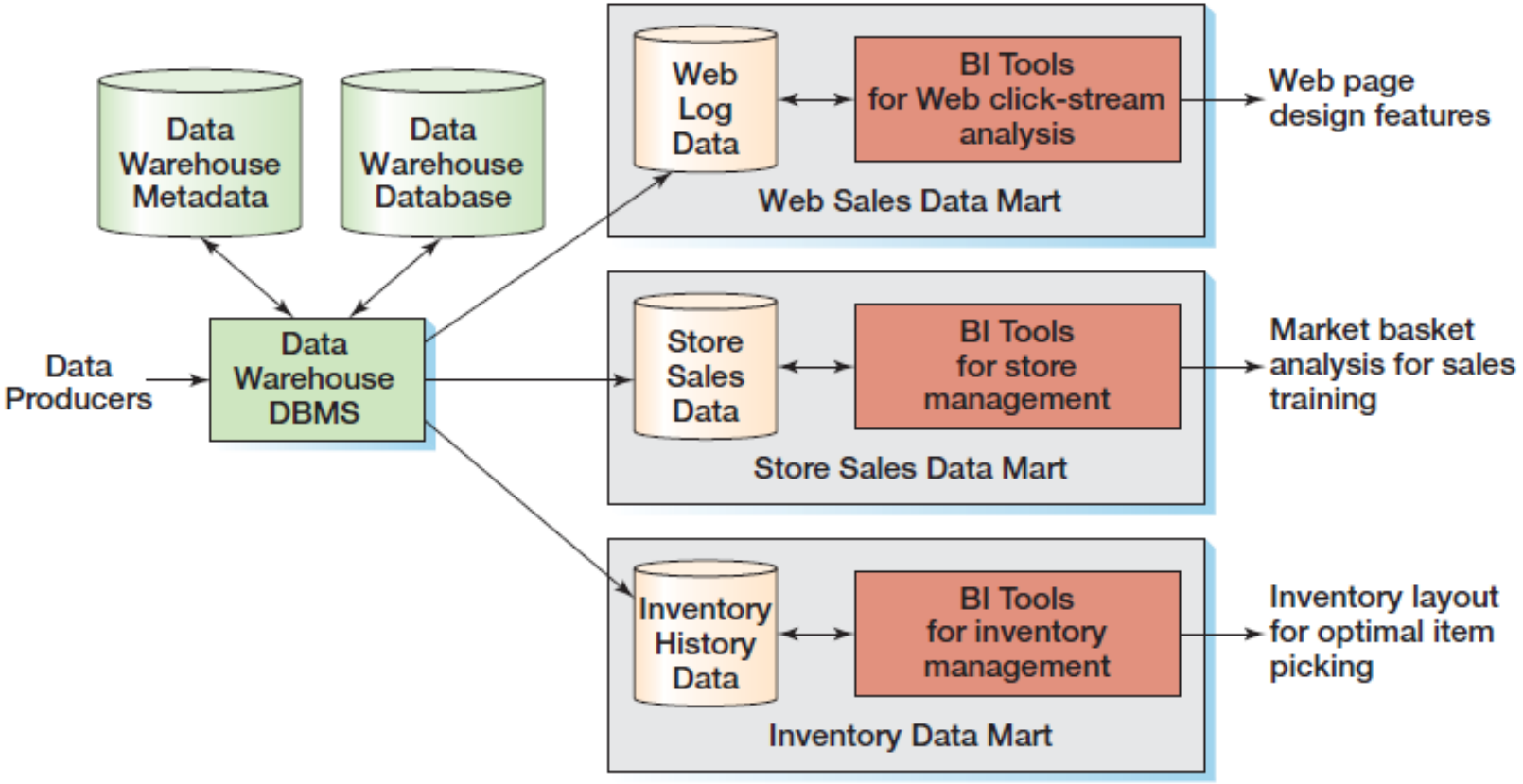
# *Data Warehousing*

Some Organizations Purchase Data to be inserted into their Data Warehouse

- Voter Registration Data
- Click data
- Name and Address lists
- Purchase History Data

## The Data Mart

A subset of a data warehouse for a group or department



**The Data Presentation area**

- **Data Marts**
- **Dimensional design**
  - Star schemas
    - Less complex
    - More easily optimized
- **Facts**
- **Dimensions**
- **Relational DBMS**

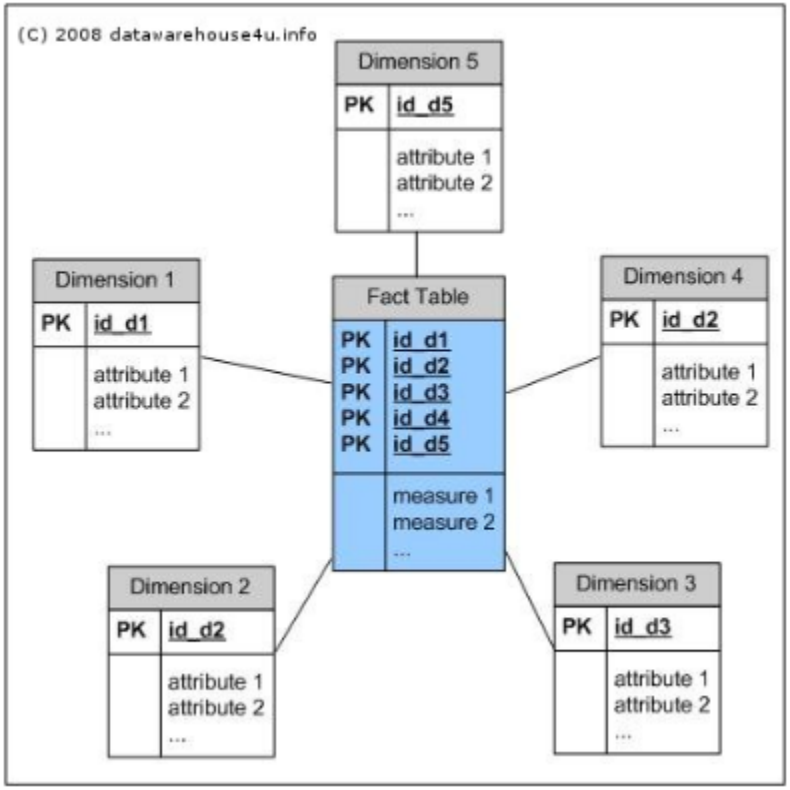
**The Dimensional Model**

- **Fills the need to store historical data**
  - Trend analysis over time
  - Keep it forever
  - Store data from many sources

| Operational Database | Dimensional Database |
|---|---|
| Used for structured transaction data processing | Used for unstructured analytical data processing |
| Current data are used | Current and historical data are used |
| Data are inserted, updated, and deleted by users | Data are loaded and updated systematically, not by users |

- **Facts**
  - What we are measuring
  - Numeric
  - Fact tables
- **Dimensions**
  - How we want to describe the facts
  - Text
  - Dimension tables

- **The STAR schema:**
  - Fact table in the middle
  - Dimensions around the outside



- **FACTS: Represent measurements**
  - Additive
    - Amounts & quantities
  - Semi-Additive
    - Point in time balances
  - Non-Additive
    - Ratios & percentages

- **FACTS examples**

- **Sales**
  - Amount
  - Quantity
- **Interest**
  - Paid
  - Received
- **Point-in-time balance**
- **Miles**
- **Length of Stay**

**Dimensions**

- **How we describe the facts**
- **Text information**
- **Stored in dimension tables**
- **Facts join to dimensions by foreign key**

- **DIMENSION examples**

- **Date**
  - Always found in a data warehouse
  - Measuring over time
- **Product**
- **Customer**
- **Location**
- **5-15 dimensions is good rule of thumb**

- **Granularity: "Grain" What does one row represent?**
  - Transaction
    - One row = one transaction
  - Periodic Snapshot
    - One row = one period of information
  - Accumulating Snapshot
    - One row = lifetime of a process

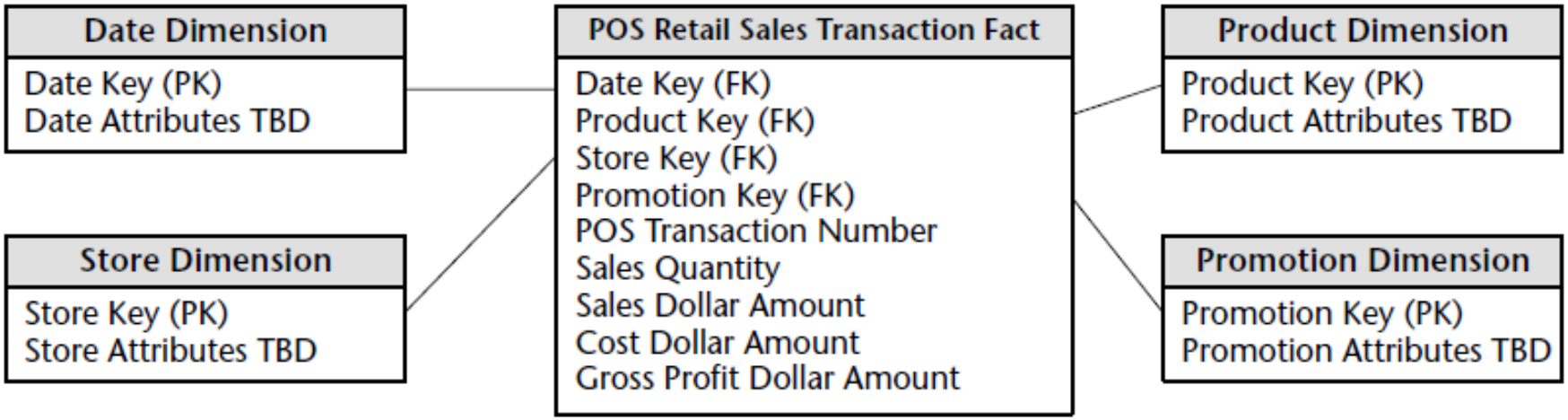
## Kimball's 4-Step Design Process

- **Select the Business Process**
  - What process are we modeling?
- **Declare the Grain**
  - What does one row of the fact table represent?
- **Choose the Dimensions**
  - How do we describe what we are measuring?
- **Identify the Facts**
  - What are we measuring?

- **Data Warehouse Design Principles**
  - Dimensions have fewer wide rows
  - Facts have many narrow rows
  - All PK and FK keys are surrogates
    - Meaningless (invisible) to users
    - Very fast for DBMS

- **Stopped here fri march 6**

- **The retail store example**



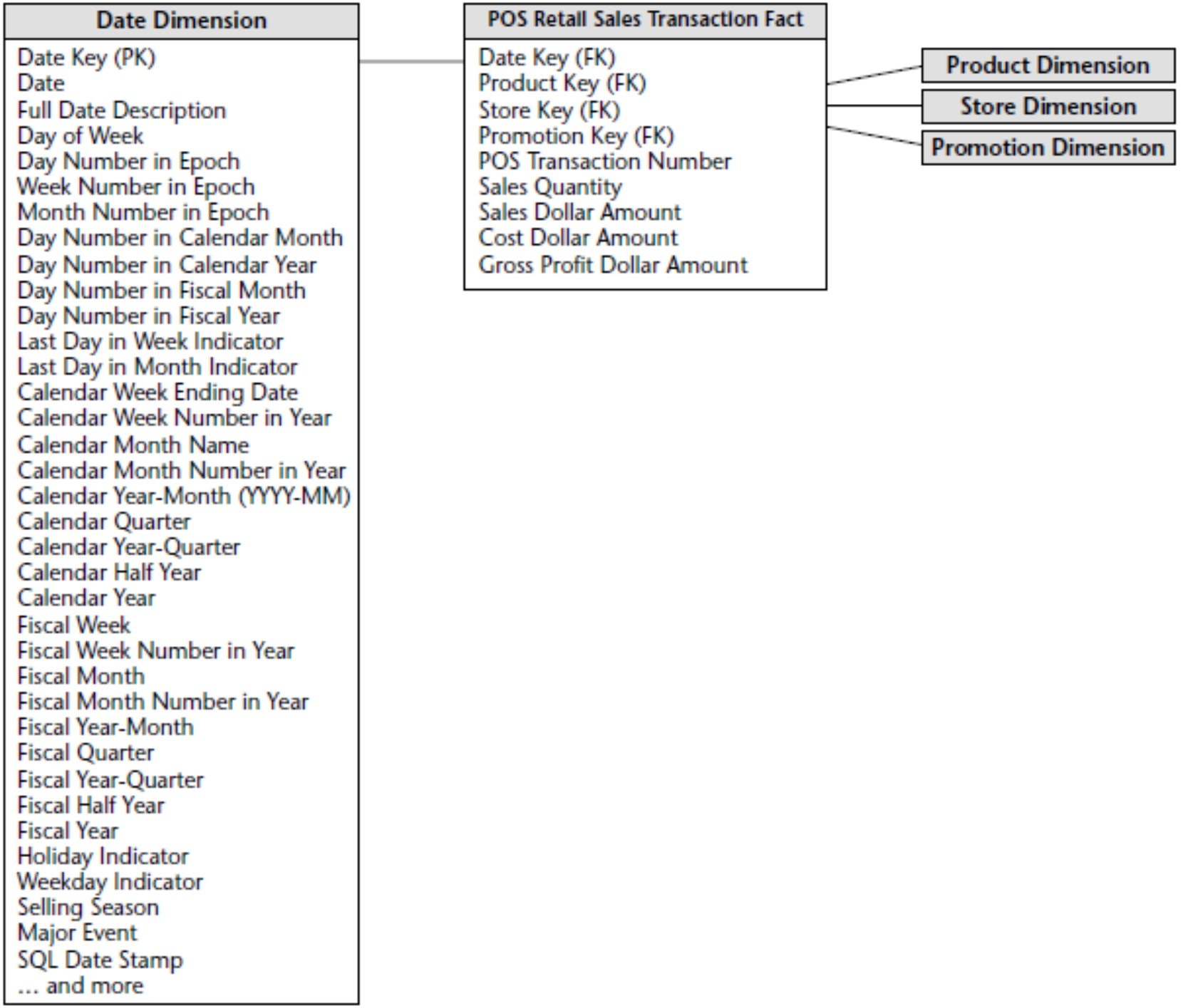
- **FACT table keys -- Designer's Choice:**

  - Your FACT table PK may be a composite key consisting of all dimension foreign keys
    - One less column, avoid overhead
  - But the combination may or may not be unique
    - Add a surrogate key (Auto-Increment) for uniqueness
    - Dimension keys are FKs
    - Turn off PK "unique" constraint
    - Adding a unique ID requires an index that is likely NOT ever used

**https://www.kimballgroup.com/2006/07/design-tip-81-fact-table-surrogate-key/**

- **FACT table indexes -- Designer's Choice:**
  - If there is a PK defined, the DBMS will create a clustering index
    - Rows arranged in physical clustering order.
  - With a surrogate key, the order is ascending on the chronological order of inserts
  - Without a surrogate key, putting the DATE Dimension FK first in a composite does the same thing
  - Without a surrogate key, index columns must be used in order
  - Best Practice: Define indexes based on users' query usage

- **The Date Dimension**
  - Populated once for every day in the organization's past, present, future operating horizon
  - Once written, a date dimension row should never change again
  - Not too big
    - 25 years * 365(366) days per year = 9,150 rows.
  - Allows for "Date Not Known" or Date TBD


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
Fiscal Year-Quarter
Fiscal Half Year
Fiscal Year
Holiday Indicator
Weekday Indicator
Selling Season
Major Event
SQL Date Stamp
... 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
Product Dimension
Store Dimension
Promotion Dimension

- **The DATE dimension**

| 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 |

- **The DATE dimension**
  - The need for unassigned dates

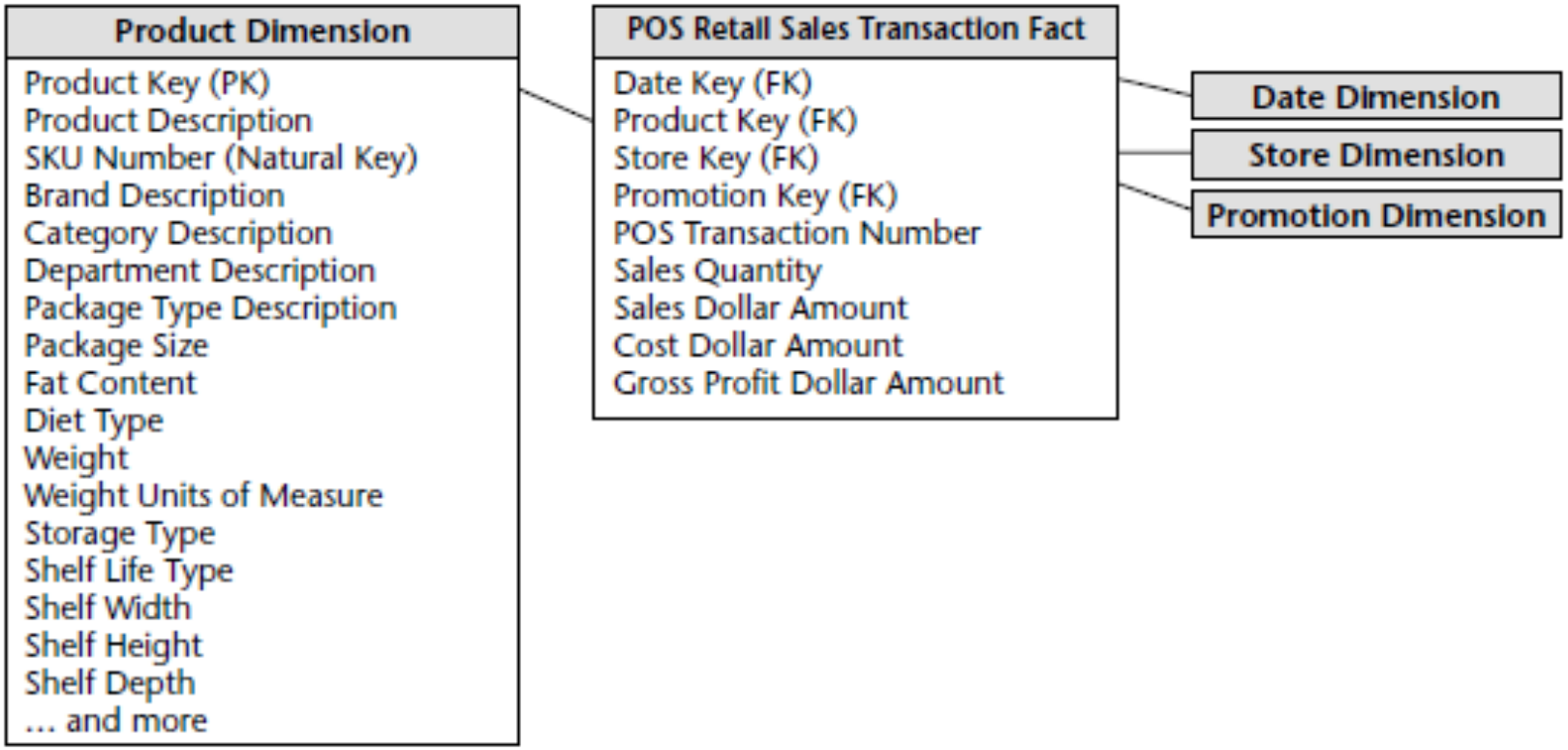
| Date Key | Date | Full Date Description | Day of Week | Calendar Month | Calendar Year | Fiscal Year-Month | Holiday Indicator | Weekday Indicator |
|---|---|---|---|---|---|---|---|---|
| 1 | 01/01/2002 | January 1, 2002 | Tuesday | January | 2002 | F2002-01 | Holiday | Weekday |
| 2 | 01/02/2002 | 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 |

  - Insert a row with key = 0
  - Date = 01/01/1900, Description = Unknown, etc.

- **Dates as a Natural Key?**
  - Dates make bad keys
    - Not unique – must be combined with a unique ID
    - CPU cycles to convert between text ➔ binary ➔ text
    - If it includes TIME: Time zone issues?
    - Format issues: U.S. versus other countries
  - Storage:
    - Date types typically use 8 bytes
    - Integer key typically use 4 bytes or less

- **SQL Date doesn't handle**
  - Date to be determined
  - Date not yet happened

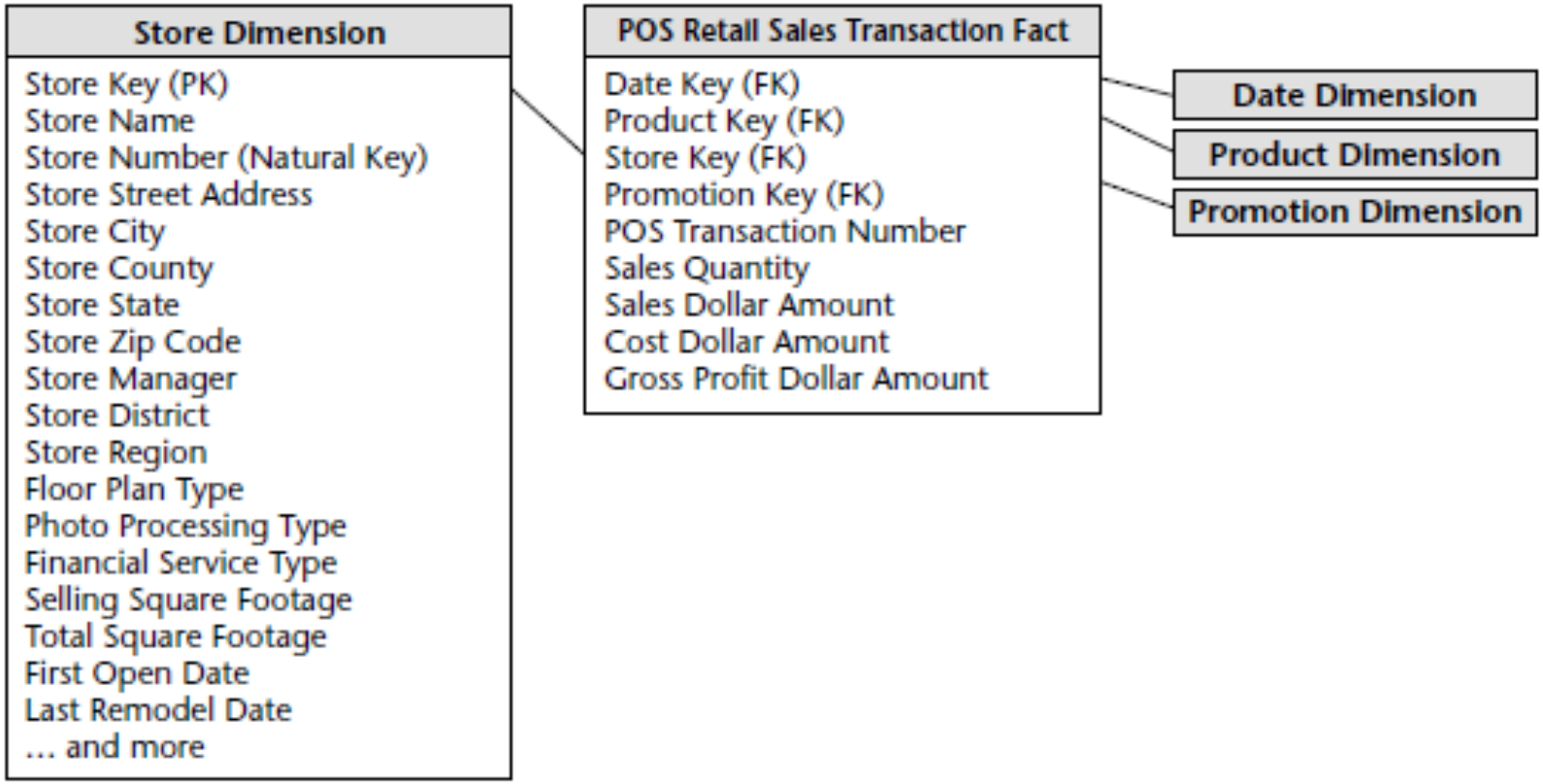
- **The Product dimension**



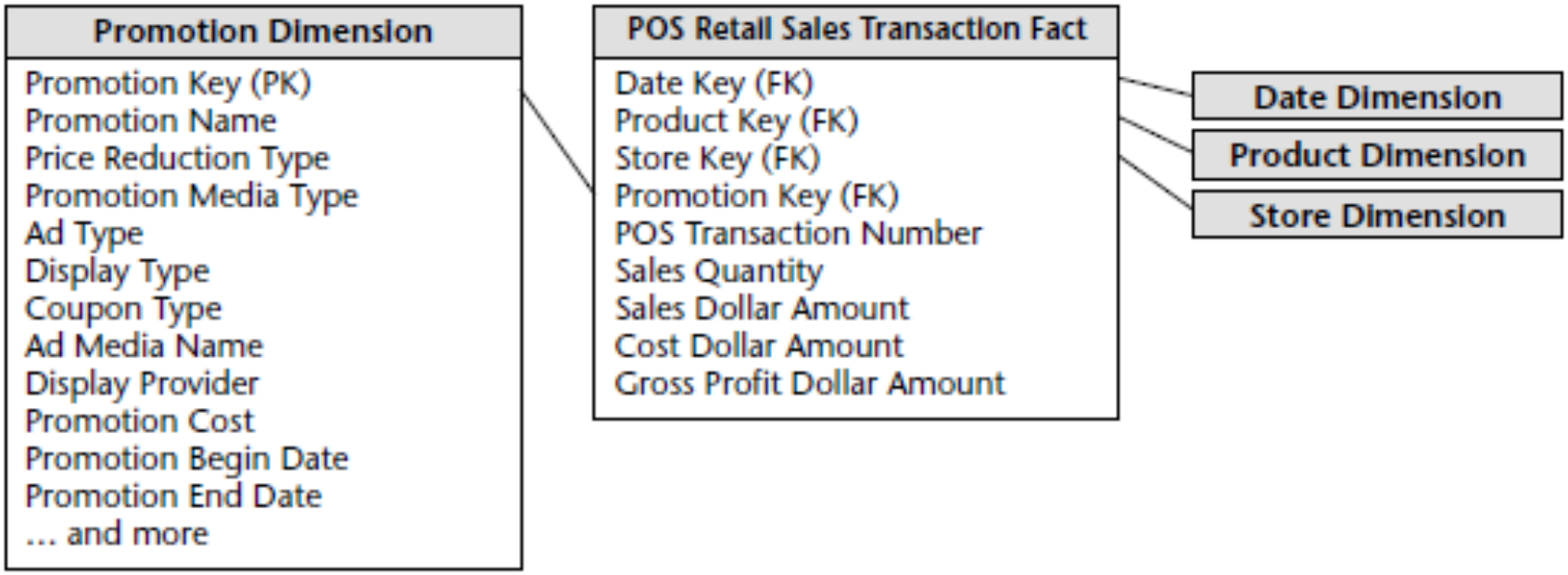
- **The Product dimension**

| 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 |

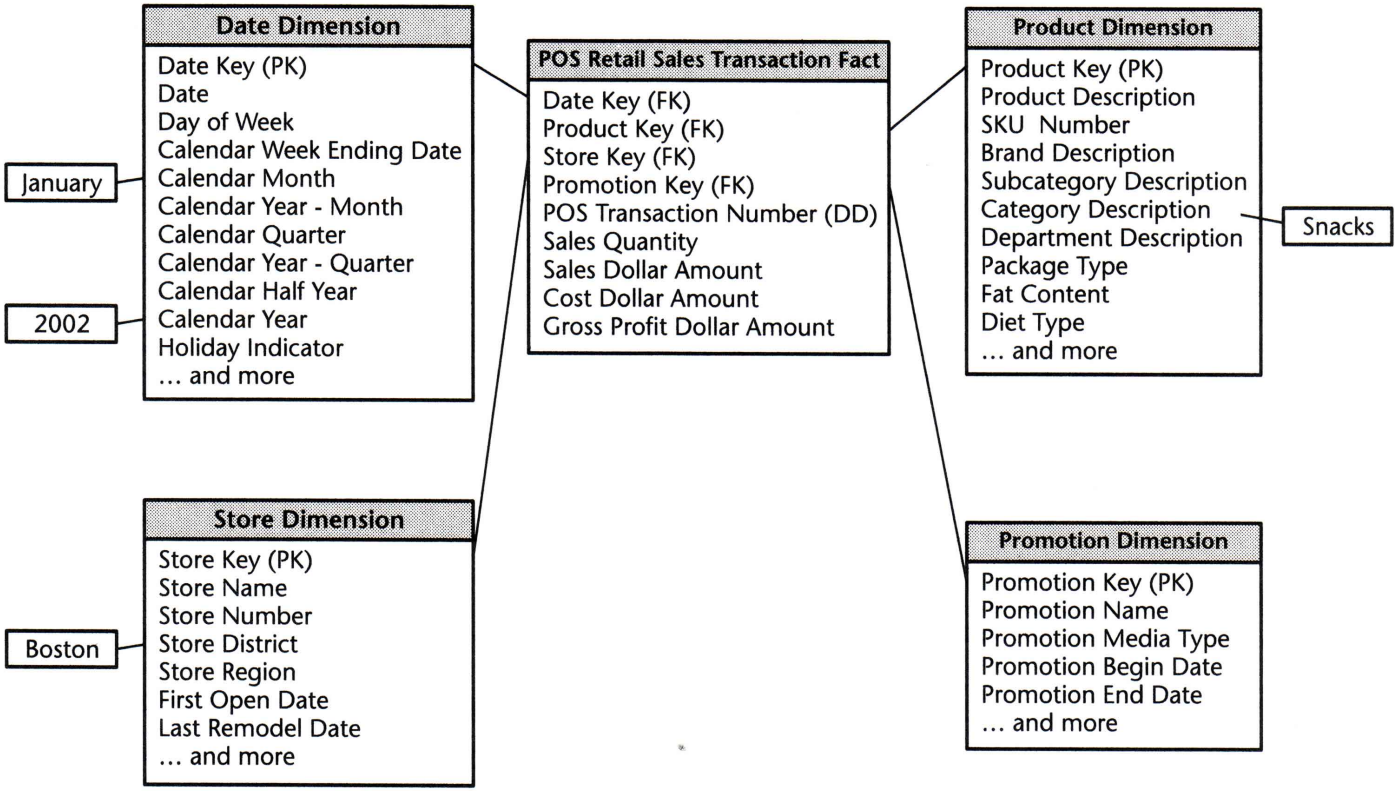
- **The Store dimension**



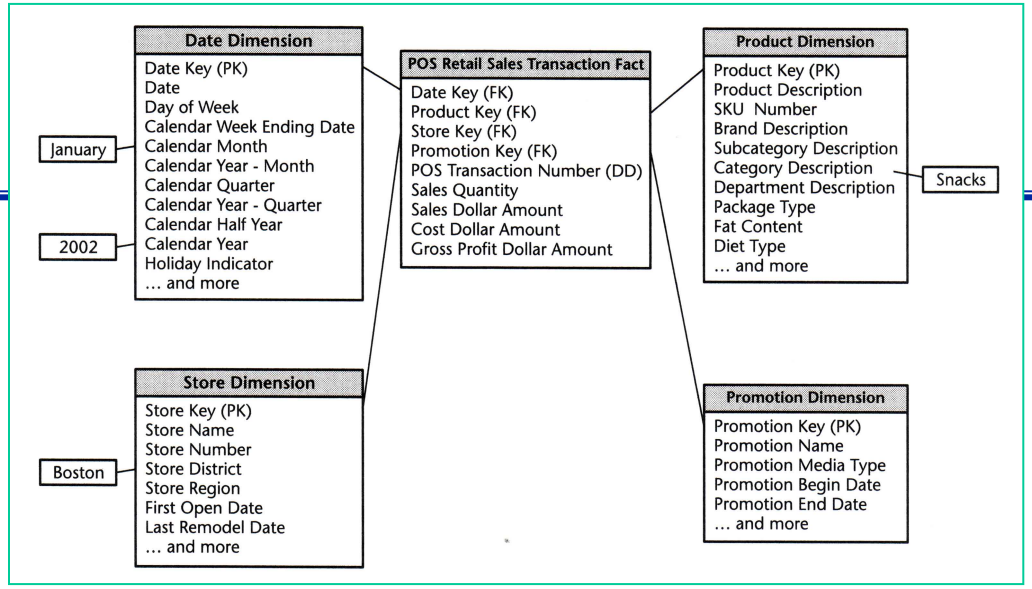
- **The Promotion dimension**



- **Using the design**




What are the total sales for snacks
in the Boston store for January 2002?




```
Select D.Year, D.Month, Sum(SalesQuantity),
          Sum(SalesDollarAmount)
    From   POSRetailSalesTransactionFact F,
          DateDimension D, ProductDimension P,
          StoreDimension S
    Where D.Month = 'January'
    And   D.Year = '2002'
    And   P.CategoryDescription = 'Snacks'
    And    S.StoreDistrict = 'Boston'
    And   F.DateKey = D.DateKey
    And   F.ProductKey = P.ProductKey
    And   F.StoreKey = S.StoreKey
Group By D.Year, D.Month
```

# *More Dimensions*

- **Degenerate Dimension**
- **Junk Dimensions**
- **Multiple Dates or Time Stamps (dimension role playing)**

# *Degenerate Dimension*

- **Useful information**
  - Grouping line items by POS Transaction Number
- **Belongs in Fact table**
- **Does not link to a Dimension table**

- **When?**
  - You need the ability to group together items from one purchase
  - You store NO DATA about the dimension

# *Leftover Stuff*

- **What to do with flags and indicators?**
  - Retail Store Examples:
    - Did they pay cash, credit or debit, gift card?
    - Did they use a coupon?
    - Local Customer?
  - Leave in Fact table row
    - Increases row size
  - Make each a separate dimension
    - Increases dimensions
  - Remove from design
    - Decreases usability of information

# *Junk Dimension*

- **Convenient grouping of low-cardinality flags and indicators**

| DimID | Payment Type | Coupon | Local Customer |
|---|---|---|---|
| 1 | Cash | Yes | Yes |
| 2 | Credit | Yes | Yes |
| 3 | Debit | Yes | Yes |
| 4 | Gift | Yes | Yes |
| 5 | Cash | Yes | No |
| 6 | Credit | Yes | No |
| 7 | Debit | Yes | No |
| 8 | Gift | Yes | No |
| 9 | Cash | No | Yes |
| 10 | Credit | No | Yes |
| 11 | Debit | No | Yes |
| 12 | Gift | No | Yes |
| 13 | Cash | No | No |
| 14 | Credit | No | No |
| 15 | Debit | No | No |
| 16 | Gift | No | No |

# *Dimension Role Playing*

- **Single dimension appears multiple times in same fact table**
- **Single physical dimension table**
- **Use *views* to allow joins with different foreign keys**
- **Most common with Date dimension**

# *Dimension Role Playing*

- **What's wrong with this query?**

```
Select P.Product_Description,
    'Date Received', Calendar_Month,
    'Date Shipped', Calendar_Month
from Warehouse_Inventory_Fact F,
     Product_Dimension P,
     Date_Dimension D
Where F.Product_Key = P.Product_Key
And   F.Date_Received = D.Date_Key
And   F.Date_Shipped = D.Date_Key
```

# *Dimension Role Playing*

- **Use the Date Dimension Twice**

```
Select P.Product_Description,
    'Date Received', R.Calendar_Month,
    'Date Shipped', S.Calendar_Month
from Warehouse_Inventory_Fact F,
      Product_Dimension P,
      Date_Dimension R,
      Date_Dimension S
Where F.Product_Key = P.Product_Key
And   F.Date_Received = R.Date_Key
And   F.Date_Shipped = S.Date_Key
```

# *Dimension Role Playing*

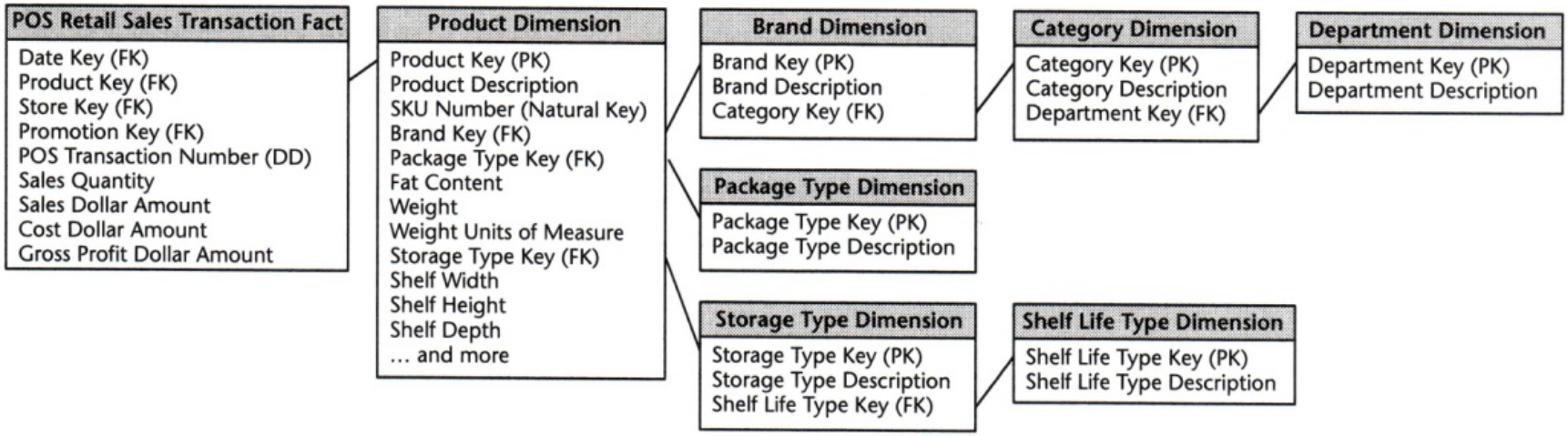
- **Use Views**

```
Create View ReceiveDate
    As Select * From Date_Dimension
Create View ShipDate
    As Select * From Date_Dimension

Select P.Product_Description,
    'Date Received', R.Calendar_Month,
    'Date Shipped', S.Calendar_Month
from Warehouse_Inventory_Fact F,
      Product_Dimension P,
      ReceiveDate R,
      ShipDate S
Where F.Product_Key = P.Product_Key
And   F.Date_Received = R.Date_Key
And   F.Date_Shipped = S.Date_Key
```

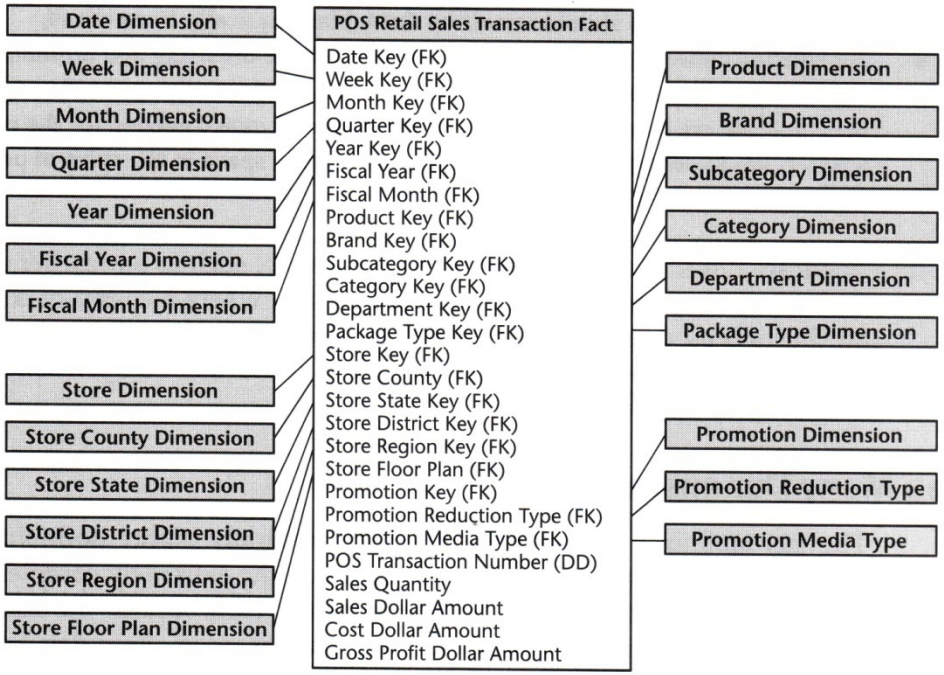
# *Snow Flaked Dimension*

- **Don't Over-Normalize Design**



# *Too Many Dimensions*

- **Centipede Fact Table**

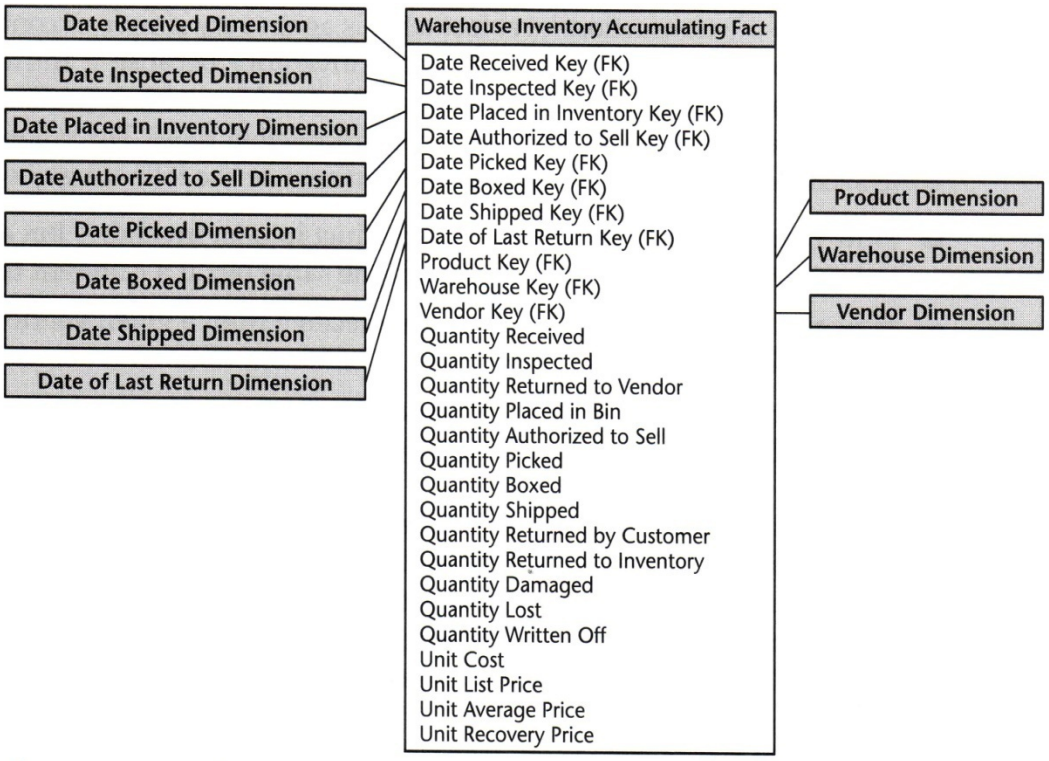


# *Natural or Meaningless Keys?*

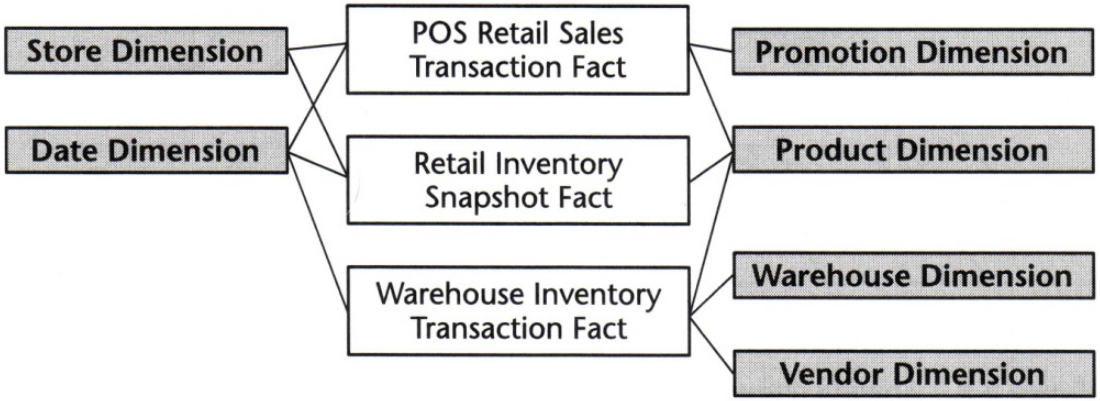
- **Integer, artificial, synthetic, surrogate**
  - Generated by sequence number
- **Much smaller (4 bytes) than character keys**
  - Provide faster searching
- **Insulates DW from operational changes**
  - Avoids re-use of dormant or unused codes
  - DW timeframe is longer than OLTP systems
  - Not vulnerable to acquisition or consolidations

# *Warehouse Inventory*

- **Accumulating Snapshot Fact**



# *Sharing Dimensions*




| BUSINESS PROCESSES | COMMON DIMENSIONS: Date | Product | Store | Promotion | Warehouse | Vendor | Contract | Shipper |
|---|---|---|---|---|---|---|---|---|
| Retail Sales | X | X | X | X | | | | |
| Retail Inventory | X | X | X | | | | | |
| Retail Deliveries | X | X | X | | | | | |
| Warehouse Inventory | X | X | | | X | X | | |
| Warehouse Deliveries | X | X | | | X | X | | |
| Purchase Orders | X | X | | | X | X | X | X |

# *Data Warehousing Mistakes To Avoid*

1. **Putting text attributes in fact tables**
2. **Limiting descriptions to save space**
3. **Splitting hierarchies into multiple dimensions**
4. **Ignoring need to track dimension changes**
5. **Solving query performance problems by adding hardware**
6. **Using “smart” keys in dimension tables**
7. **Not complying with fact table grain**
8. **Designing the model based on one specific report**
9. **Having users query atomic data in normalized format**

- **A-B and Teradata**
- **Stop here**

# *Slowly Changing Dimensions*

- **Additional slides on Slowly Changing Dimensions**

# *Slowly Changing Dimensions*

- **Want to handle changes gracefully**
- **Dimension maintenance**
  - Should accurately reflect present
  - What about the past?
- **What happens when dimension change info not received on time?**

# *Type 1 SCD*

- **Overwrite the Value**

| Product Key | Product Description | Department | SKU Number (Natural Key) |
|---|---|---|---|
| 12345 | IntelliKidz 1.0 | Education | ABC922-Z |
| | | | |

| Product Key | Product Description | Department | SKU Number (Natural Key) |
|---|---|---|---|
| 12345 | IntelliKidz 1.0 | Strategy | ABC922-Z |
| | | | |

# *Type 2 SCD*

- **Add a new row in the dimension table**
  - Good for partitioning history

| Product Key | Product Description | Department | SKU Number (Natural Key) |
|---|---|---|---|
| 12345 | IntelliKidz 1.0 | Education | ABC922-Z |
| | | | |

| Product Key | Product Description | Department | SKU Number (Natural Key) |
|---|---|---|---|
| 12345 | IntelliKidz 1.0 | Education | ABC922-Z |
| 25984 | IntelliKidz 1.0 | Strategy | ABC922-Z |

# *Type 3 SCD*

- **Add a new column**
  - Good for presenting alternate realities

| Product Key | Product Description | Current Department | Historical Department | SKU Number (Natural Key) |
|---|---|---|---|---|
| 12345 | IntelliKidz 1.0 | Education | Education | ABC922-Z |
| | | | | |

| Product Key | Product Description | Department | Prior Department | SKU Number (Natural Key) |
|---|---|---|---|---|
| 12345 | IntelliKidz 1.0 | Strategy | Education | ABC922-Z |
| | | | | |

# *Hybrid SCD*

- Preserve historical accuracy (Type 2)
- Report historical data according to current values (Type 3)

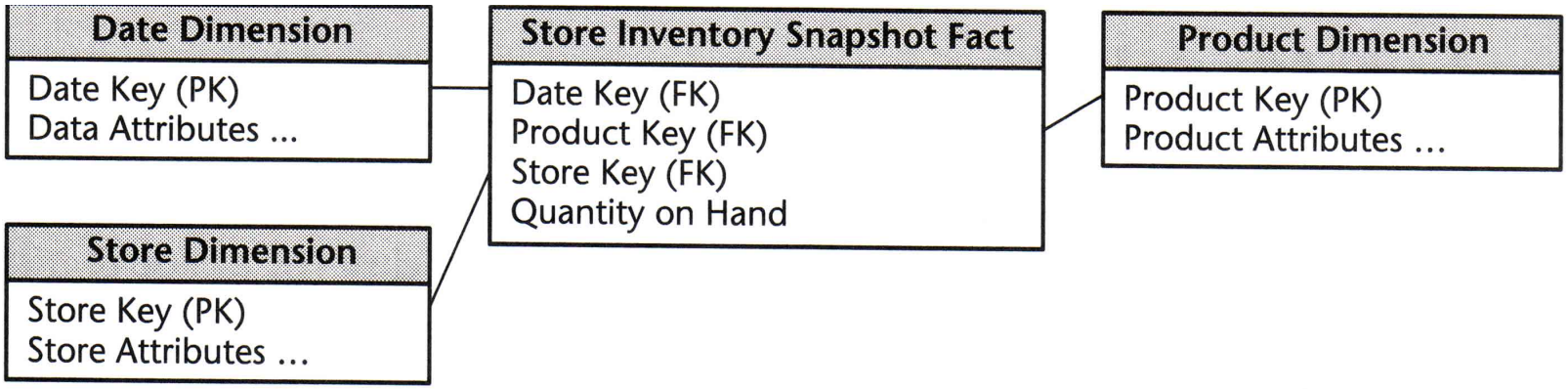
| Product Key | Product Description | Current Department | Historical Department | SKU Number (Natural Key) |
|---|---|---|---|---|
| 12345 | IntelliKidz 1.0 | Education | Education | ABC922-Z |
| | | | | |

| Product Key | Product Description | Current Department | Historical Department | SKU Number (Natural Key) |
|---|---|---|---|---|
| 12345 | IntelliKidz 1.0 | Strategy | Education | ABC922-Z |
| 25984 | IntelliKidz 1.0 | Strategy | Strategy | ABC922-Z |

| Product Key | Product Description | Current Department | Historical Department | SKU Number (Natural Key) |
|---|---|---|---|---|
| 12345 | IntelliKidz 1.0 | Critical Thinking | Education | ABC922-Z |
| 25984 | IntelliKidz 1.0 | Critical Thinking | Strategy | ABC922-Z |
| 31726 | IntelliKidz 1.0 | Critical Thinking | Critical Thinking | ABC922-Z |

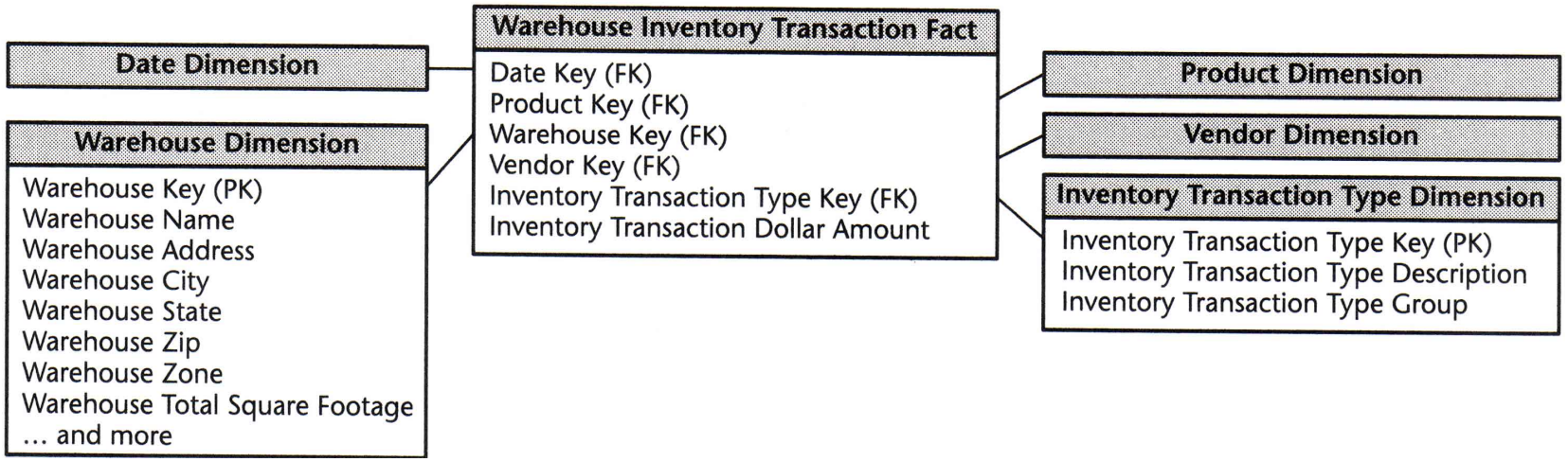
# *Inventory Periodic Snapshot*

- **Periodic Snapshot Fact**



# *Warehouse Inventory*

- **Transaction Fact**



# *Fact Table Comparisons*

| | Transaction | Periodic Snapshot | Accumulating Snapshot |
|---|---|---|---|
| ***Time Period*** | Point in Time | Regular Intervals | Short lived, indeterminate |
| ***Grain*** | One row per transaction (event) | One row per period | One row per process life |
| ***Loads*** | Insert | Insert | Insert and Update |
| ***Updates*** | Only for Error correction | Only for Error correction | Whenever Activity |
| ***Date Dim*** | Transaction Date | End of Period Date | Multiple Dates representing Milestones |
| ***Facts*** | Transaction Activity | Performance for Interval | Performance over finite lifetime |