Four-Step Design Process

Reference

Kimball, R., Ross, M. (2002). *The data warehouse toolkit, 2nd edition*. New York, NY. John Wiley & Sons, Inc.

ISBN: 0-471-20024-7

Four-Step Design Process

- select the business process to model
- declare the grain of the business process
- choose dimensions that apply to each fact table row
- identify the numeric facts that will populate each fact table row

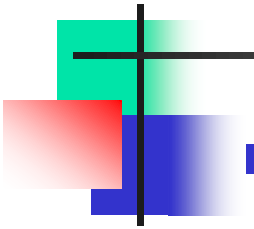
Selecting the Business Process

- a business process is a natural business activity, supported by a data-collection system
  - raw materials purchasing
  - order management
  - inventory
  - customer relationship management
  - budgeting
  - human resources management

Declare the Grain

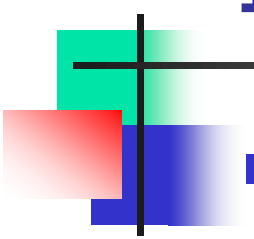
- describe *exactly* what an individual fact table row specifies
  - a line item on a retail sales ticket
  - a boarding pass to get on a flight
  - a daily snapshot of inventory levels for each product in a warehouse
  - a snapshot of account balances at end of each accounting period
  - an individual procurement transaction

Choose the Dimensions



- the business process determines the dimensions
- if the grain is clearly defined, the dimensions are normally easy to determine
- dimensions should supply a rich set of descriptive data for the business process being modeled

Identify the Facts



- to identify the facts, ask "what are we measuring?"
- facts must be compatible with the grain defined in step 2
- typical facts are numeric additive values such as quantity ordered or dollar cost amount

Retail Case Study

- we have a large grocery store chain
  - 100 grocery stores
  - 5-state area
  - each grocery store has many departments: grocery, frozen foods, meat, produce, bakery, floral, health and beauty, …
  - approximately 60,000 products per store, identified by SKUs

SKUs

- an SKU is a *stock-keeping unit*
- of the 60,000 products in the grocery store, about 55,000 are from outside manufacturers and have a *universal product code* (UPC), which is at the same grain as an SKU
- each different package variation of a product has a separate UPC and therefore also a separate SKU

SKUs

- the remaining 5,000 SKUs are from departments within the store (meat or bakery for example) and do not have UPCs
- the grocery store assigns SKUs to these products and sticks scanner labels, containing the SKU, on these products

Data Collection

- when a customer checks out at a cash register, the bar codes are scanned directly to the point-of-sale (POS) system of the grocery store
- data is also collected when vendors make deliveries, and inventory information is kept
- for now, we will only be concerned with POS transactions

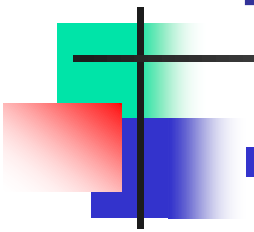
The Case Study

- management is interested in studying purchasing habits of customers to determine the most effective marketing strategies
- to this end, they want to build a data warehouse
- the business process to be modeled is customer purchases at POS terminals

Retail Sales Model – 4 Step Process

- business process: customer purchases as modeled by the POS system
- grain: an individual line item on a POS transaction
- dimensions: store, product, date, and transaction
- facts: sales quantity, sales dollar amount, cost dollar amount, profit dollar amount

Dimension Tables



- we will look at each of the dimension tables in turn
- note that all dimension tables will have a surrogate key
- if the dimension table also exists in the operational database and has a primary key, the primary key value is stored, but it is *not* used as the surrogate key in the date warehouse

Date Dimension Attributes

- date dimension is included in virtually every data mart
- this is necessary because SQL date function does not support many date attributes, such as fiscal periods and holidays
- the date dimension may be relatively small: 10 years worth of days is only 3,650 rows – relatively small for a dimension

A Typical Date Dimension

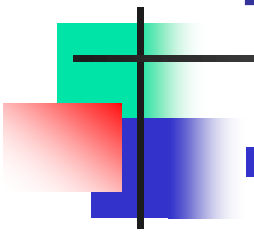
Product Dimension

- may include 50 or more attributes, including SKU (stock-keeping unit) number, product description, department code, department description, brand, weight, package type, and many more
- rich set of attributes makes it easy to drill down through the data

Typical Product Dimension

Product	
PK	**ProductKey**
	ProductDescription SKUNumber BrandName BrandDescription CategoryName CategoryDescription DepartmentName DepartmentDescription PackageTypeDescription PackageSize FatContent DietType Weight WeightUnitsOfMeasure StorageType ShelfLifeType ShelfWidth ShelfHeight ShelfDepth ...and more

Product Dimension



- recall that stores carry about 60,000 SKUs
- however, because merchandise changes and we store historical data for products that are no longer carried, we expect at least 150,000 rows in this table – but perhaps as many as a million

Product Dimension

- while we may have 150,000 distinct SKUs, there may be only 50 different values of the department attribute
- this means that a department description might be repeated an average of 3,000 times in the product table

Product Dimension

- this may tempt you to normalize by creating a Department table in a 1:M relation with product, but don't do it !
- dimension table space requirements are small in comparison to the space required by the fact table

Store Dimension

- a geographic dimension
- a store can be thought of as a location
- stores can be "rolled up" to any geographic attribute, such as zip, state, or city
- stores can also be rolled up to store districts and regions

Store Dimension

- it is acceptable to represent multiple hierarchies (zip, city, state and also districts, regions) in a dimension table

Typical Store Dimension

Store	
PK	**StoreKey**
	StoreNumber StoreName StreetAddress City State Zip FloorPlanType TotalSquareFeet GrocerySquareFeet FrozenSquareFeet MeatSquareFeet DistrictID DistrictDescription DistrictSquareMiles RegionID RegionDescription FirstOpenDate ... and more

The Transaction Dimension

- information from the transaction that we might want to store includes
  - date
  - store number
  - transaction number
- except for the transaction number, the information that we want to store for transactions is already present in other dimensions

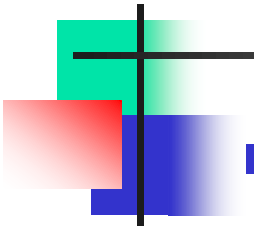
The Transaction Dimension

- this means the only item that needs to be stored in the transaction dimension is the transaction number
- because this dimension has only an identifier-like attribute and no other attributes, the POS transaction dimension is considered to be empty and is not included as a separate dimension table

The Transaction Dimension

- the transaction dimension is referred to as a *degenerate dimension*
- the POS transaction number is included in the fact table (with notation DD to indicate it is a degenerate dimension) and does not link to any dimension table
- if we do not include the POS transaction number, we cannot pull together all the line items on a particular transaction

The Fact Table

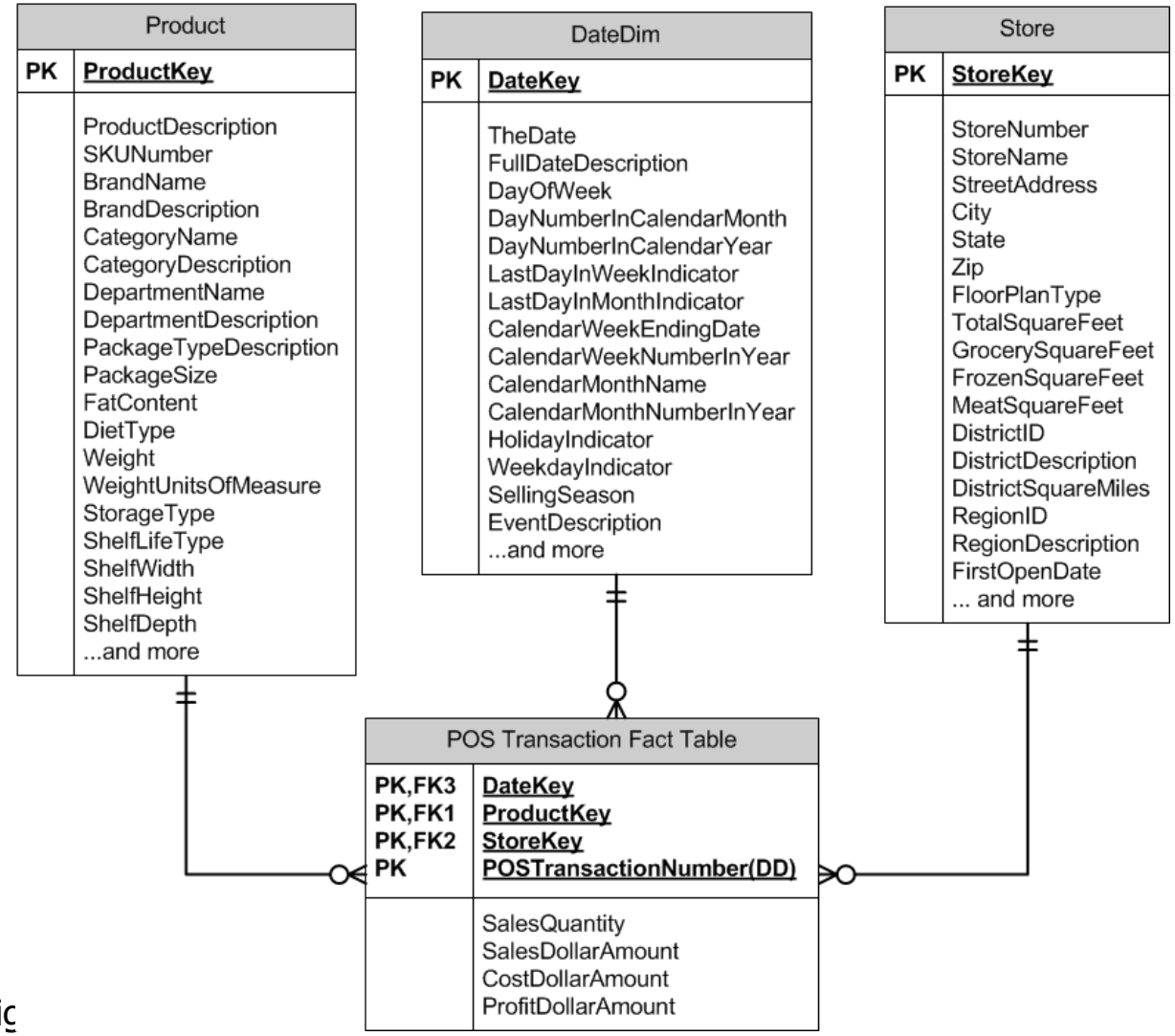


- unlike the dimension tables, the fact table will normally have a composite key that includes the primary keys of the dimension tables
- other attributes may also be part of the primary key of the fact table
- in addition, “facts” are stored in the fact table
- the primary key attributes are not facts

The Fact Table

POSFactTable	
PK,FK1 PK,FK2 PK,FK3 PK	**DateKey** **ProductKEY** **StoreKey** **POSTransactionNumber(DD)**
	SalesQuantity SalesDollarAmount CostDollarAmount ProfitDollarAmount

The Complete Model



Additive Facts

- sales quantity, dollar sales amount, cost dollar amount, and profit dollar amount are additive across *all* dimensions
- for example, consider sales quantity
  - we can add sales quantity for a particular product and a particular date across all stores and the result is meaningful ("the chain sold 375 8-ounce packages of Kraft's shredded Swiss cheese on 1/5/2007")

Additive Facts

- we can add sales quantity for a particular store and a particular date across all products and the result is meaningful ("Piggly Wiggly Store 203 sold 3,456 individual items on 1/5/2007")

Additive Facts

- we can add sales quantity for a particular store and a particular product across all dates and the result is meaningful ("Piggly Wiggly Store 203 has sold 4,523 8-ounce packages of Kraft's shredded Swiss cheese since it opened"

Additive Facts

- we can add sales quantity for all stores and all products across all dates and the result is meaningful ("The stores in the chain have sold a total of 108,234,567 items since the chain opened its first store in 1995")

Semi-Additive Facts

- a semi-additive fact can be added across some, but not all, dimensions
- bank balances are semi-additive
  - you can add across accounts (you can add the amounts of your savings and checking accounts and the total makes sense)
  - you can’t add across dates (adding the amounts you have in a savings account today and tomorrow does not give any meaningful information

Non-Additive Facts

- some facts cannot be meaningfully added across any dimension
- measures of intensity, such as temperatures or blood pressure, for example, are usually non-additive
  - it does not make sense to add the temperatures in Dallas and New York City
  - it does not make sense to add yesterday's and today's high temperature

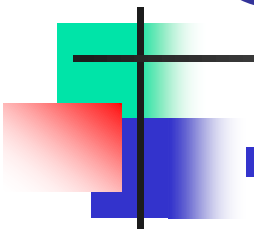
Calculated Facts

- profit can be calculated by subtracting the cost from the sales
- in operational databases, calculated facts are generally not stored
- in a data warehouse, it is common to store calculated facts – the storage cost is minor and storing it removes the possibility of user error in making the calculation

Things to Avoid

- dimension normalization (snowflaking)
- too many dimensions (centipedes)
  - rule of thumb is to have less than 15 dimensions
  - 25 or more dimensions is almost always wrong

Surrogate Keys



- dimensional tables should use surrogate keys
  - surrogate keys should be meaningless
  - do not use “smart” keys where you can tell something about the contents of the row simply by looking at the key

Surrogate Keys Exceptions

- surrogate keys should be used for the date dimension, but unlike other surrogate keys, the date dimension keys should be assigned in a meaningful, sequential order
- typically, surrogate keys are not assigned to degenerate dimensions
  - in the example in these slides, the actual transaction number is stored in the fact table, instead of a surrogate key

Four-Step Design Process

The End