Data Warehouse

CS 5513

Outline

- Introduction
- Data warehouse
  - Architecture for a data warehouse
  - Data warehouse schema
  - Multidimensional data model
  - Partitioning
  - Aggregation
  - Indexing
  - Data marting
  - Maintenance

Introduction

- Data is growing at a phenomenal rate
- Users expect more sophisticated information
- How?

  UNCOVER HIDDEN INFORMATION

  DATA MINING

Data Mining Definition

- Finding hidden information in a database
- Fit data to a model
- Similar terms
  - Exploratory data analysis
  - Data driven discovery
  - Deductive learning

Data Mining Algorithm

- Objective: Fit Data to a Model
  - Descriptive
  - Predictive
- Preference – Technique to choose the best model
- Search – Technique to search the data
  - "Query"

Database Processing vs. Data Mining Processing

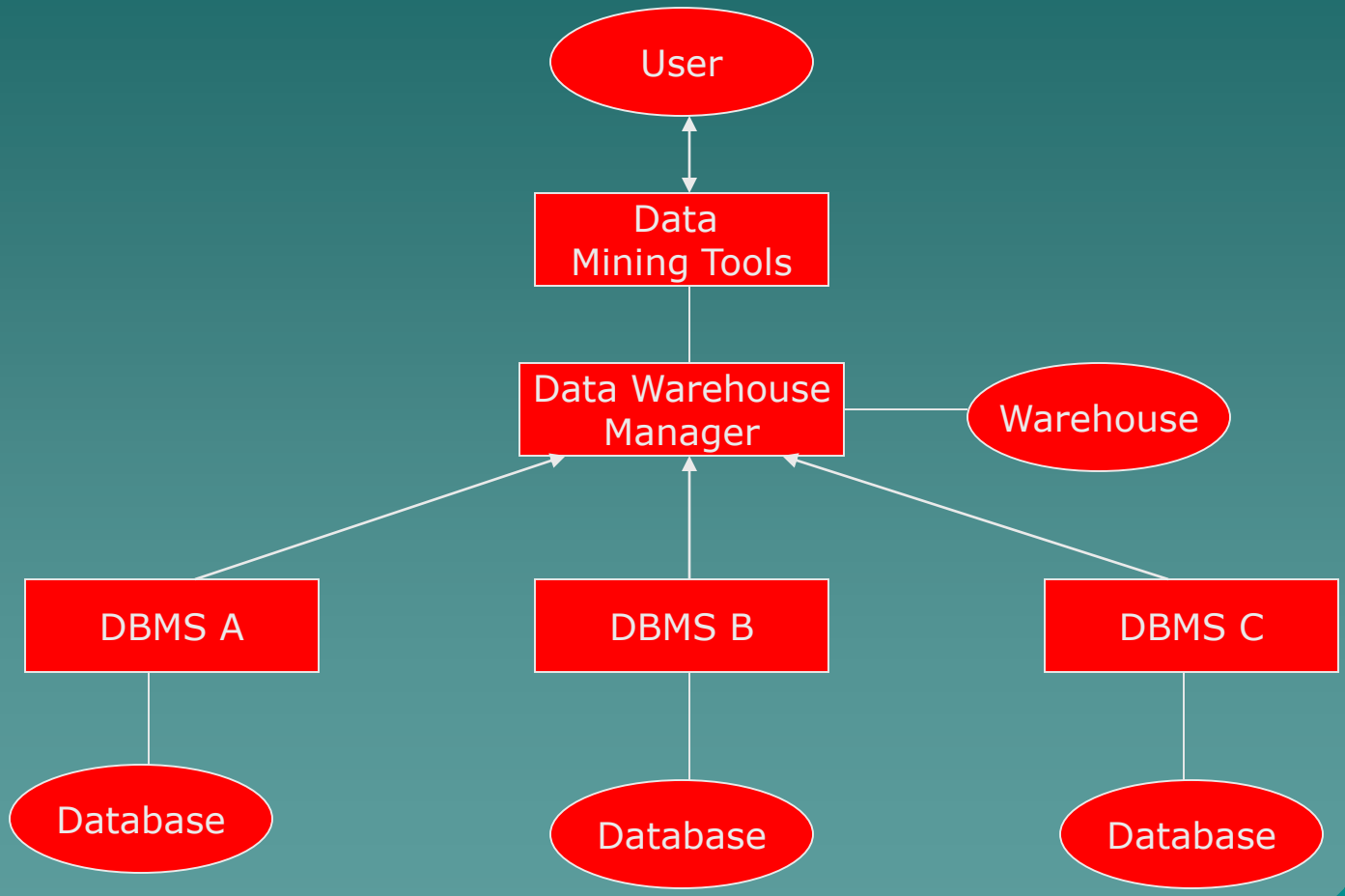
- Query
  - Well defined
  - SQL
- Data
  - Operational data
- Output
  - Precise
  - Subset of database

- Query
  - Poorly defined
  - No precise query language
- Data
  - Not operational data
- Output
  - Fuzzy
  - Not a subset of database

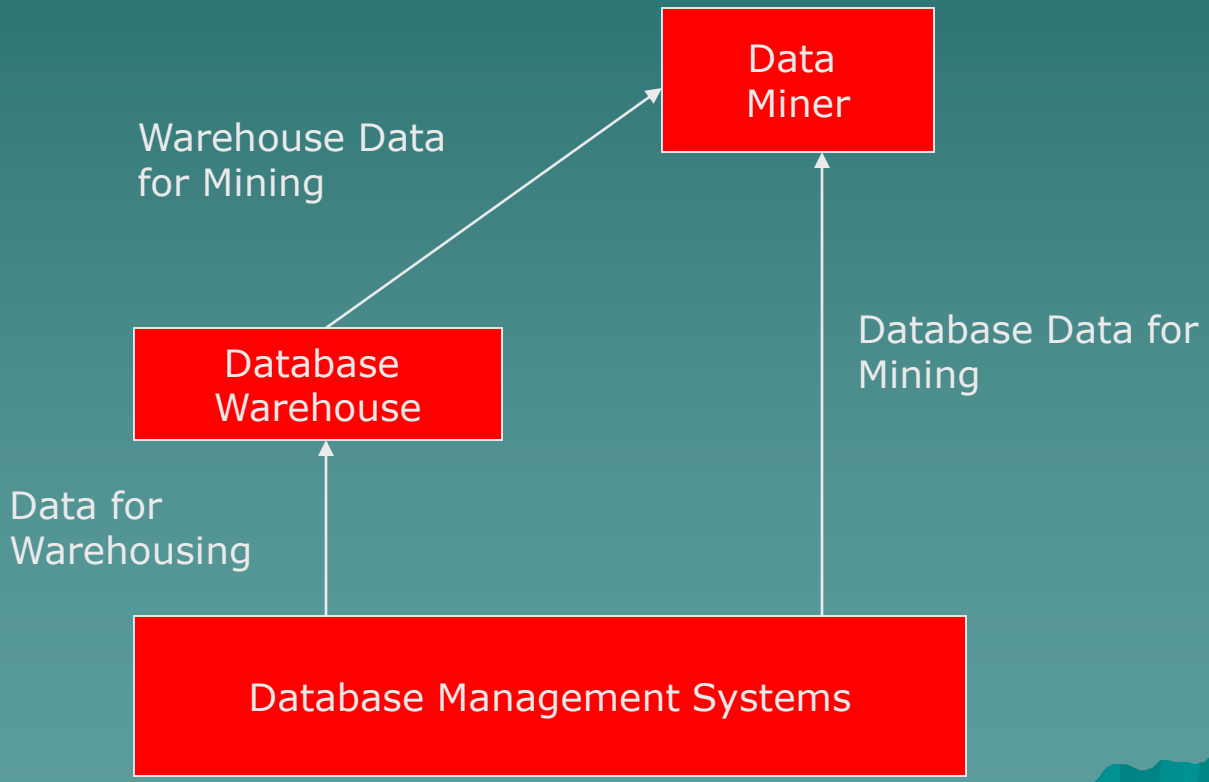
Query Examples

- Database
  - Find all credit applicants with last name of Smith.
  - Identify customers who have purchased more than $10,000 in the last month.
  - Find all customers who have purchased milk
- Data Mining
  - Find all credit applicants who are poor credit risks. (classification)
  - Identify customers with similar buying habits. (Clustering)
  - Find all items which are frequently purchased with milk. (association rules)

Data Mining vs. Data Warehousing



Database Systems, Data Warehousing and Mining



OLTP (Online Transaction Processing) vs. OLAP (Online Analytical Processing

	OLTP	OLAP
User	Clerk, DBA, Database professional	Knowledge worker (manager, executive, analyst)
Function	Day to day operations	Decision support
DB design	Application-oriented (ER based)	Subject-oriented (Star, Snowflake)
Data	Current, isolated	Historical, consolidated (note: must take care of inconsistency of different rows of data)
View	Detailed, flat relational	Summarized, multidimensional
Usage	Structured, repetitive	Ad hoc
Unit of work	Short, simple transaction	Complex query (note: lots of joins)
Access	Read/write	Read mostly
Operations	Index/hash on primary key	Lots of scans
# Records accessed	Tens	Millions
# Users	Thousands	Hundreds
DB size	GB to high-order GB	>= TB
Metric	Transaction throughput	Query throughput, response time

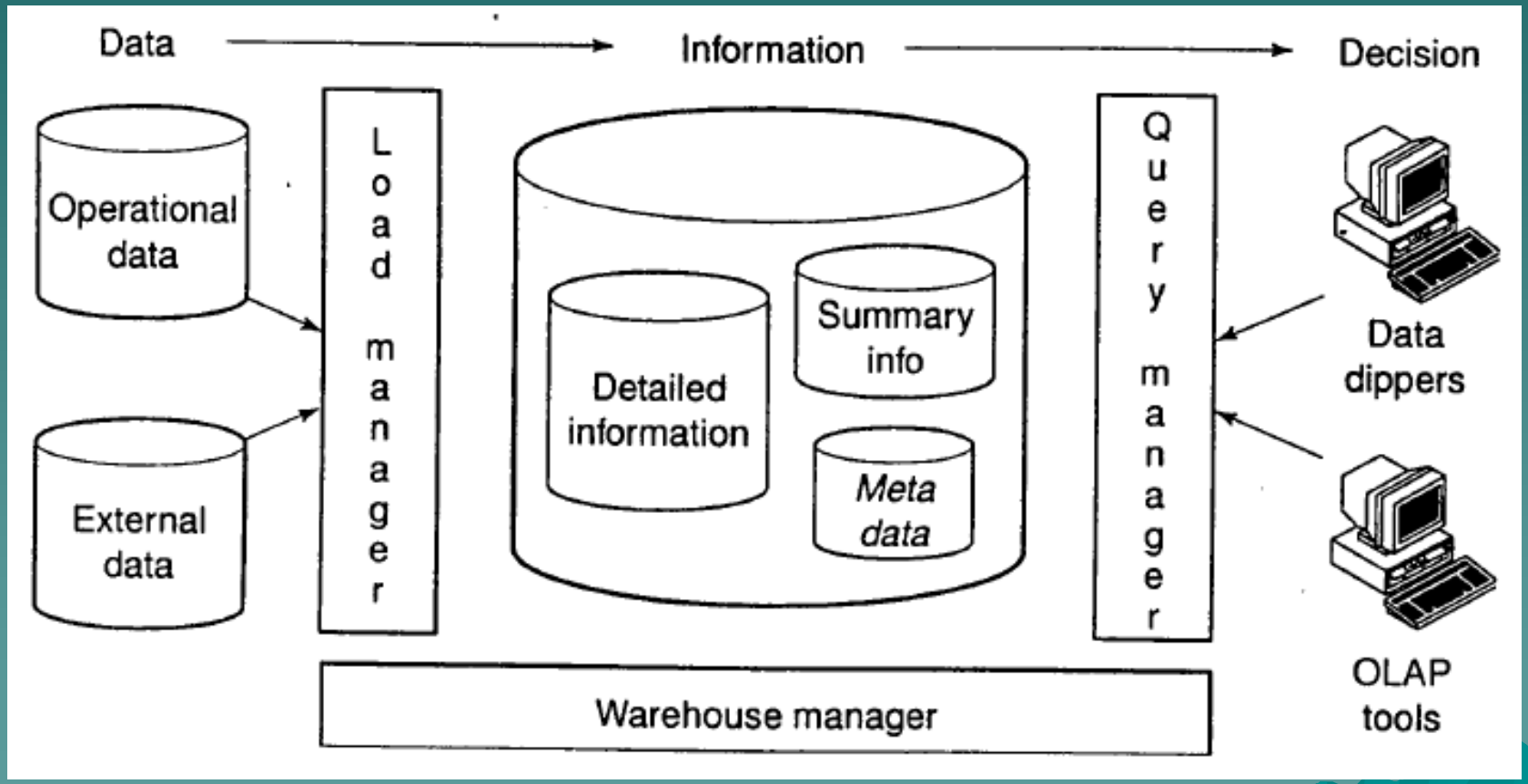
Data Warehouse

- A decision support database that is maintained separately from the organization's operational databases.
- A data warehouse is a
  - subject-oriented,
  - integrated,
  - time-varying,
  - non-volatile

  collection of data that is used primarily in organizational decision making.

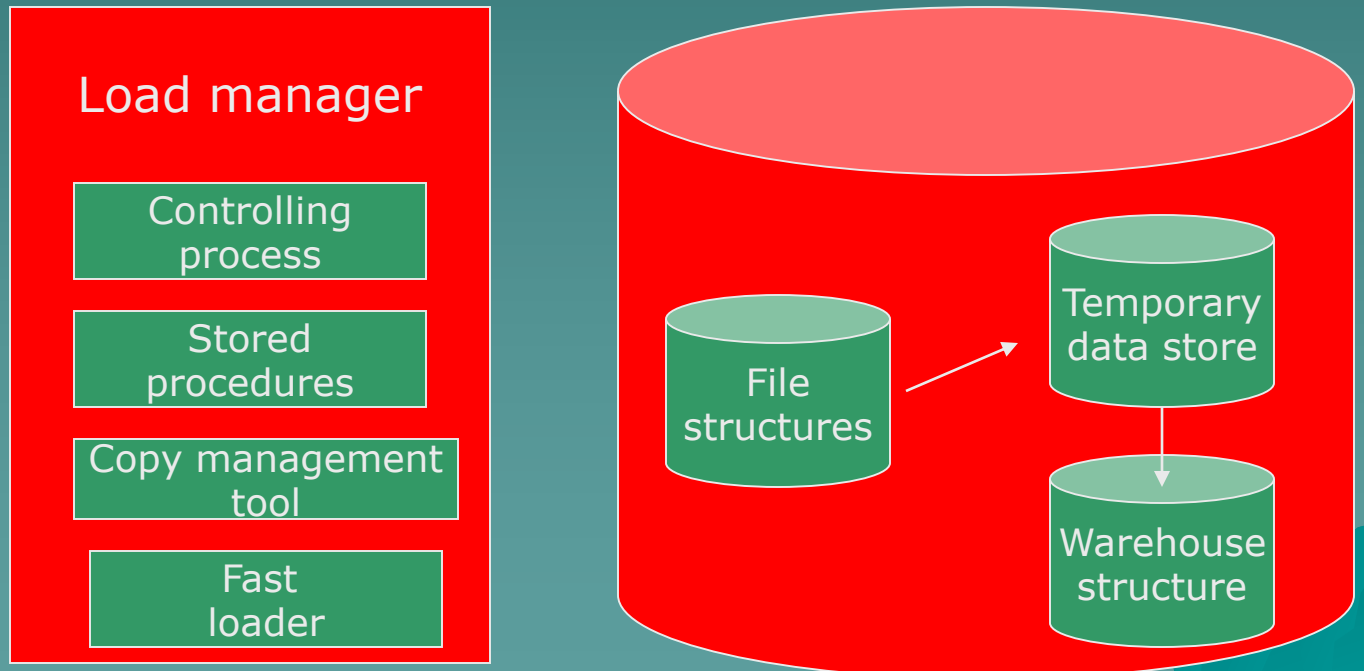
  -- W.H. Inmon, Building the Data Warehouse, John Wiley & Sons, 1996

Architecture for a Data Warehouse



Load Manager Architecture

- A load manager performs the following operations:
  - Extract the data from the source systems.
  - Fast-load the extracted data into a temporary data store.
  - Perform simple transformations into a structure similar to the one in the data warehouse.

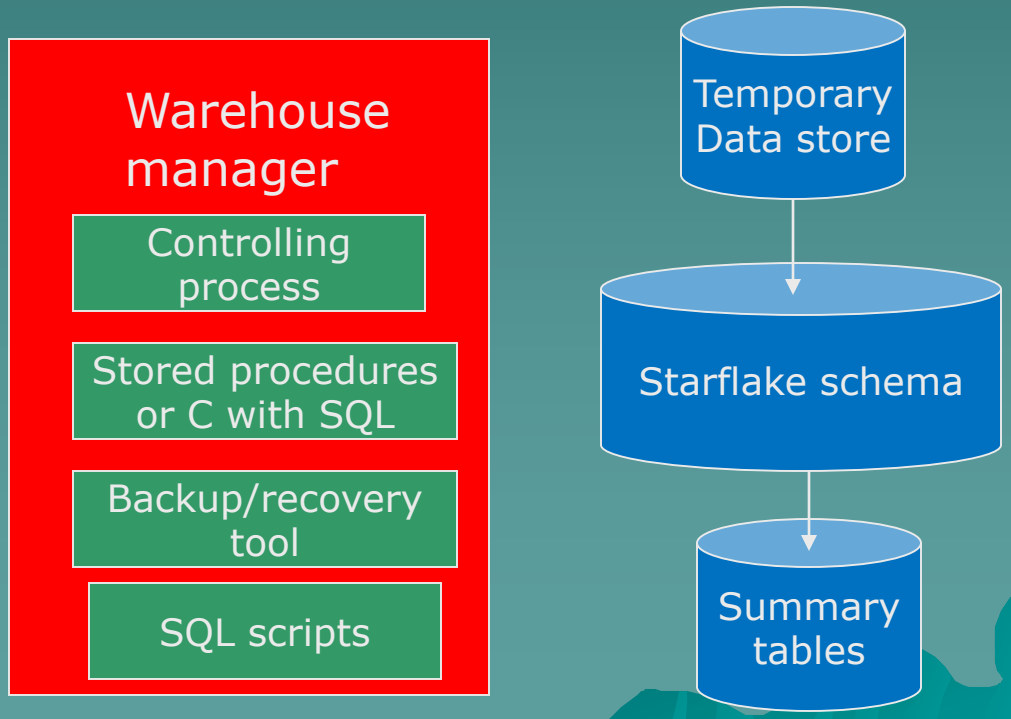


Warehouse Manager

- A warehouse manager performs the following operations:
  - Analyze the data to perform consistency and referential integrity checks.
  - Transform and merge the source data in the temporary data store into the published data warehouse.
  - Create indexes, business views, partition views, business synonyms against the base data.
  - Generate renormalizations if appropriate.
  - Generate any new aggregations that may be required.
  - Update all existing aggregations.
  - Back up incrementally or totally the data within the data warehouse.
  - Archive data that has reached the end of its capture life.

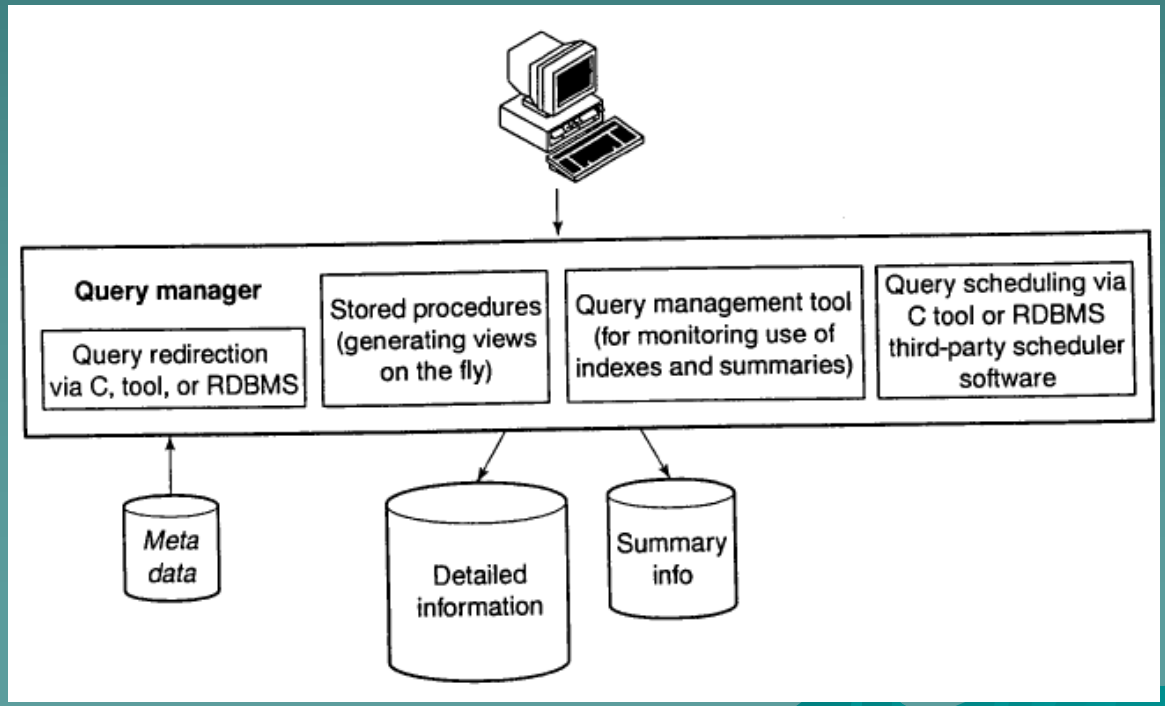
Warehouse Manager Architecture

- In some cases, the warehouse manager also analyzes query profiles to determine which indexes and aggregations are appropriate.
- As with the load manager, each of these functions has to operate to a very large extent with no human intervention.



Query Manager Architecture

- A query manager performs the following operations:
  - Direct queries to the appropriate table(s).
  - Schedule the execution of user queries.



Extraction, Transformation, and Loading (ETL)

(Han+Kamber, Data Miining: Concepts and Techniques, Morgan Kaufman, 2011)

- **Data extract**
  - get data from multiple, heterogeneous, and external sources
- **Data cleaning**
  - detect errors in the data and rectify them when possible
- **Data transformation**
  - convert data from legacy or host format to warehouse format
- **Load**
  - sort, summarize, consolidate, compute views, check integrity, and build indices and partitions
- **Refresh**
  - propagate the updates from the data sources to the warehouse

Data Warehouse Schemas

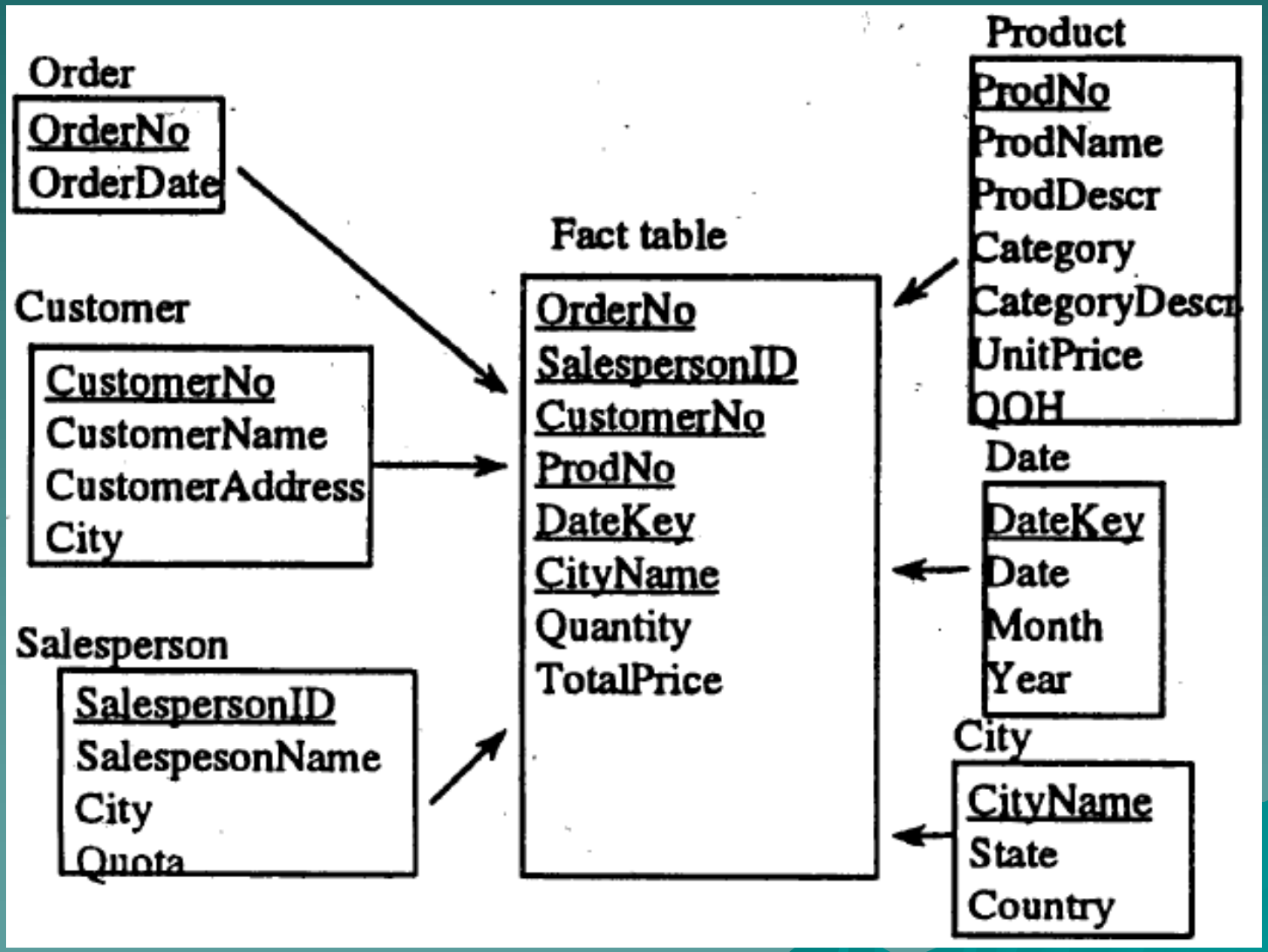
- ER design techniques are not appropriate
- Must structure data to exploit typical decision support queries
- Most decision support queries share similarities:
  - The queries examine a set of factual transactions.
    e.g. account transactions, EPOS (Electronic Point of Sale) transactions
  - The queries analyze facts in a variety of ways
    e.g. EPOS transactions by week or by store
- Schema alternatives: star schema, snowflake schema, starflake schema

Star Schema

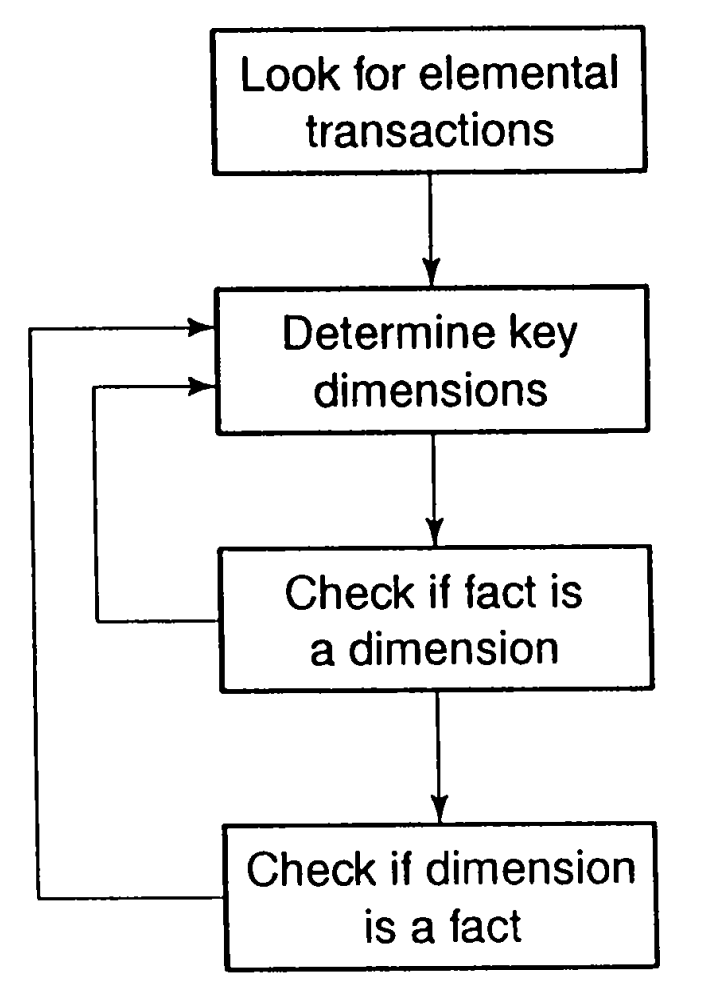
- Basic premise: information is classified into two groups: facts and dimensions.
- Facts: the core data elements being analyzed; transactions that have occurred at some point in the past and are unlikely to change in the future
- Dimensions: attributes about facts
- Every fact points to a tuple in each of dimensions and has additional attributes
- Represent data intuitively
- Does not capture hierarchies directly

FACTS	DIMENSION DATA
Millions or billions of rows	Tens to a few million rows
Multiple foreign keys	One primary key
Numeric	Textual description
Don't change	Frequently modified

Example of a Star Schema



Flowchart of Fact Table Identification Process



Candidate Fact Tables for each Industry/Business Requirement

Sector and Business	Fact Table
Retail	
Sales	EPOS transaction
Shrinkage analysis	Stock movement and stock position
Retail Banking	
Customer profiling	Customer events
Customer profitability	Account transactions
Insurance	
Product profitability	Customer claims and receipts
Telco	
Call analysis	Call event
Customer analysis	Customer events (e.g. installation, disconnection, payment)

Entities That Tend to Be Structured as Either Facts or Dimensions

Entity	Fact or dimension	Condition
Customer	Fact	In a customer profiling or customer marketing database, it is probably a fact table.
	Dimension	In a retail sales analysis data warehouse, or any other variation where the customer is not the focus, but is used as the basis for analysis.
Promotions	Fact	In a promotions analysis data warehouse, promotions are the focal transactions, so are probably facts.
	Dimension	In all other situations

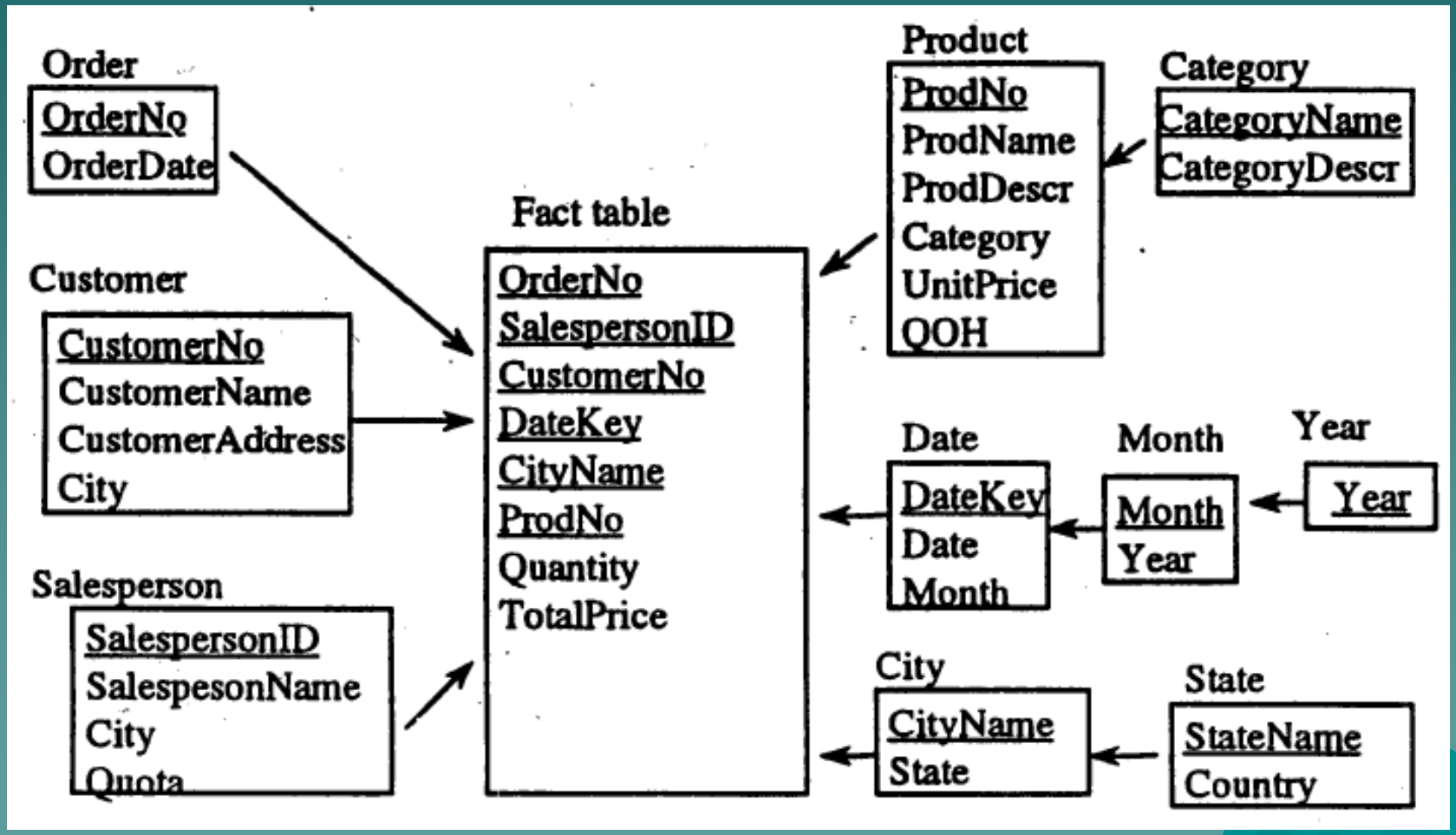
Star Schema: Designing Fact Tables

- Identify the significant historical period for each supported function.
- Select the appropriate columns.
- Minimize the column sizes.
- Determine the use of intelligent or non-intelligent keys as foreign keys
  - Intelligent keys: each key represents the unique identifier for the item in the real-world
  - Non-intelligent keys: automatically generated

Snowflake Schema

- Represent dimensional hierarchy directly by normalizing the dimension tables
- Easy to maintain
- Save storage

Example of a Snowflake Schema



Approaches to Building Warehouses

1. *ROLAP* = "relational OLAP": Tune a relational DBMS to support star schemas.
2. *MOLAP* = "multidimensional OLAP": Use a specialized DBMS with a model such as the "data cube."
3. *HOLAP* = "Hybrid OLAP": combine ROLAP and MOLAP.

Multidimensional Data Model (MOLAP)

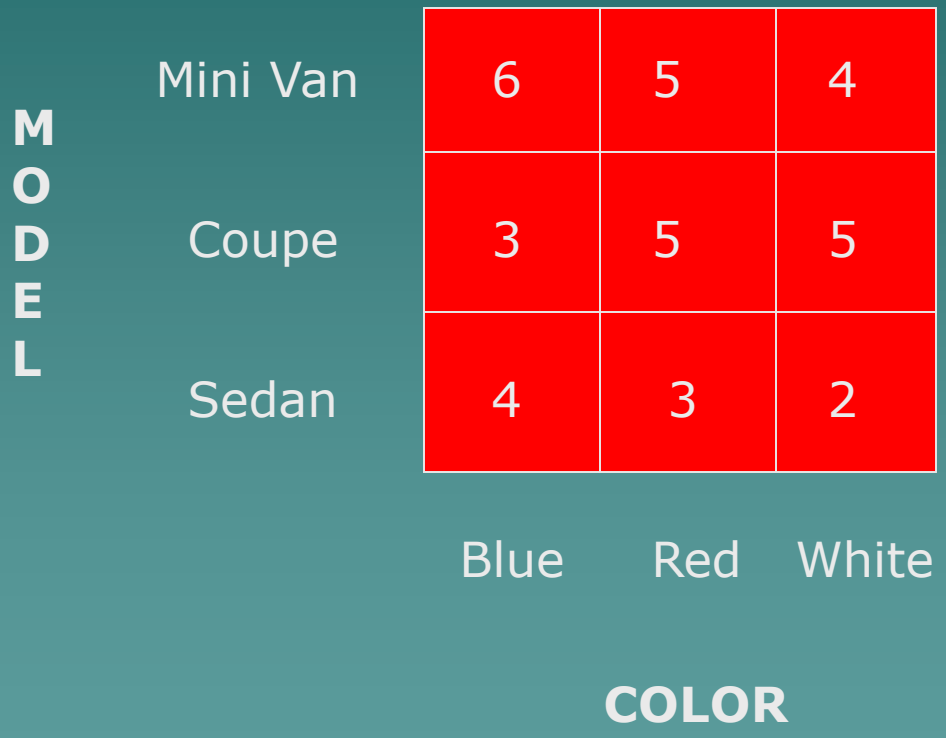
- Getting answers to typical business questions from raw data often requires viewing that data from various perspectives:
  - Sales volumes by model
  - Sales volumes by color
  - Sales volumes by dealership
  - Sales volumes over time
- Alternative way of representing this data: use a multidimensional array (data cube).
- An array is the fundamental component of a multidimensional database.

Example 1: Relational Table (Relational OLAP: ROLAP)

SALES VOLUMES FOR GLEASON DEALERSHIP

MODEL	COLOR	SALES VOLUME
MINI VAN	BLUE	6
MINI VAN	RED	5
MINI VAN	WHITE	4
SPORTS COUPE	BLUE	3
SPORTS COUPE	RED	5
SPORTS COUPE	WHITE	5
SEDAN	BLUE	4
SEDAN	RED	3
SEDAN	WHITE	2

Example 1: Corresponding Multidimensional Data Model



Example 2

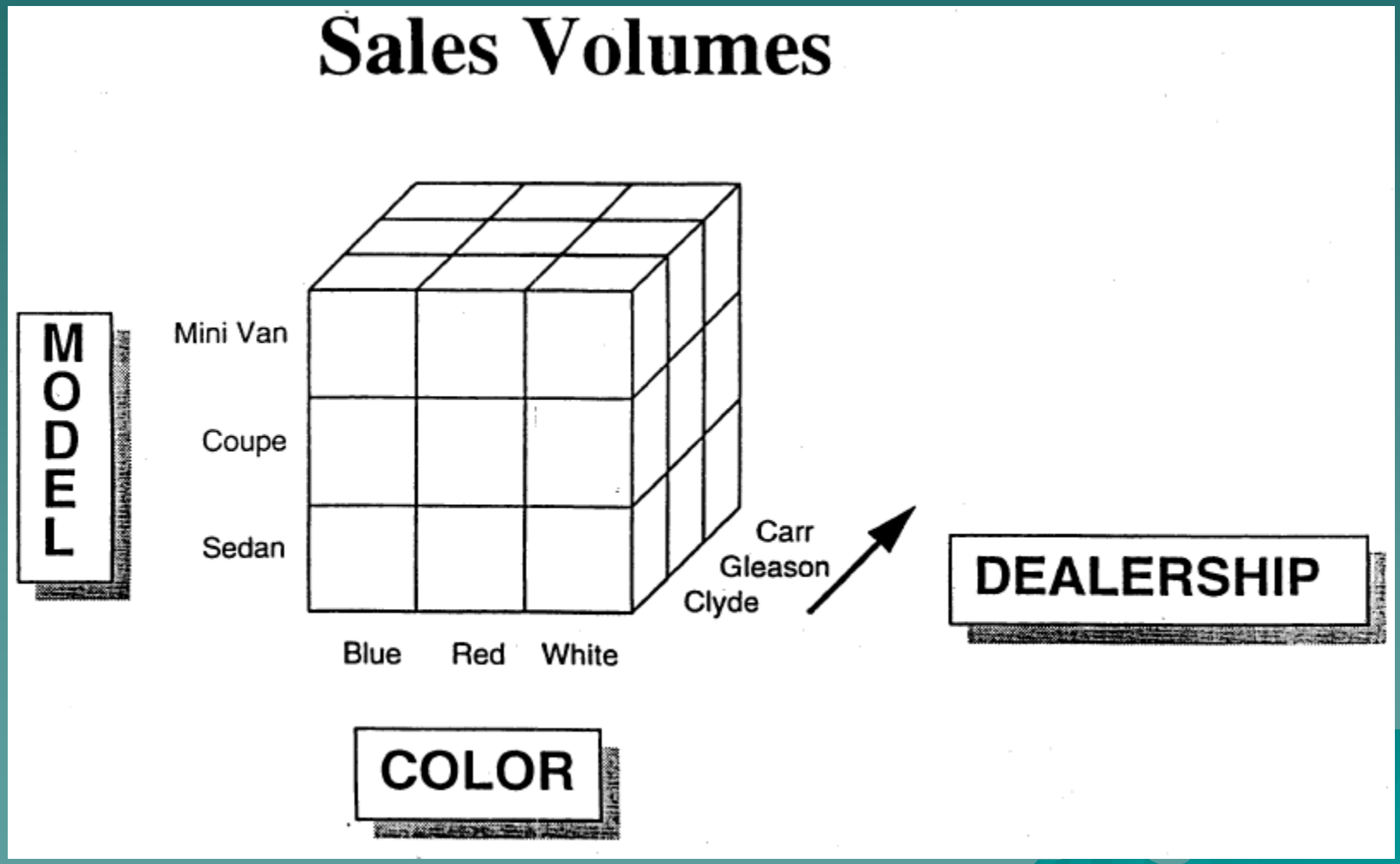
- Another example: increasingly complex relational tables
  - three possible values of model (mini van, sports coupe, sedan)
  - three possible values of color (blue, red, white)
  - three possible values of dealership (Clyde, Gleason, Carr)
- Corresponding three dimensional array structure

Example 2: More Complex Relational Table

SALES VOLUMES FOR ALL DEALERSHIPS

MODEL	COLOR	DEALERSHIP	VOLUME
MINI VAN	BLUE	CLYDE	6
MINI VAN	BLUE	GLEASON	6
MINI VAN	BLUE	CARR	2
MINI VAN	RED	CLYDE	3
MINI VAN	RED	GLEASON	5
MINI VAN	RED	CARR	5
MINI VAN	WHITE	CLYDE	2
MINI VAN	WHITE	GLEASON	4
MINI VAN	WHITE	CARR	3
SPORTS COUPE	BLUE	CLYDE	2
SPORTS COUPE	BLUE	GLEASON	3
SPORTS COUPE	BLUE	CARR	2
SPORTS COUPE	RED	CLYDE	7
SPORTS COUPE	RED	GLEASON	5
SPORTS COUPE	RED	CARR	2
SPORTS COUPE	WHITE	CLYDE	4
SPORTS COUPE	WHITE	GLEASON	5
SPORTS COUPE	WHITE	CARR	1
SEDAN	BLUE	CLYDE	6
SEDAN	BLUE	GLEASON	4
SEDAN	BLUE	CARR	2
SEDAN	RED	CLYDE	1
SEDAN	RED	GLEASON	3
SEDAN	RED	CARR	4
SEDAN	WHITE	CLYDE	2
SEDAN	WHITE	GLEASON	2
SEDAN	WHITE	CARR	3

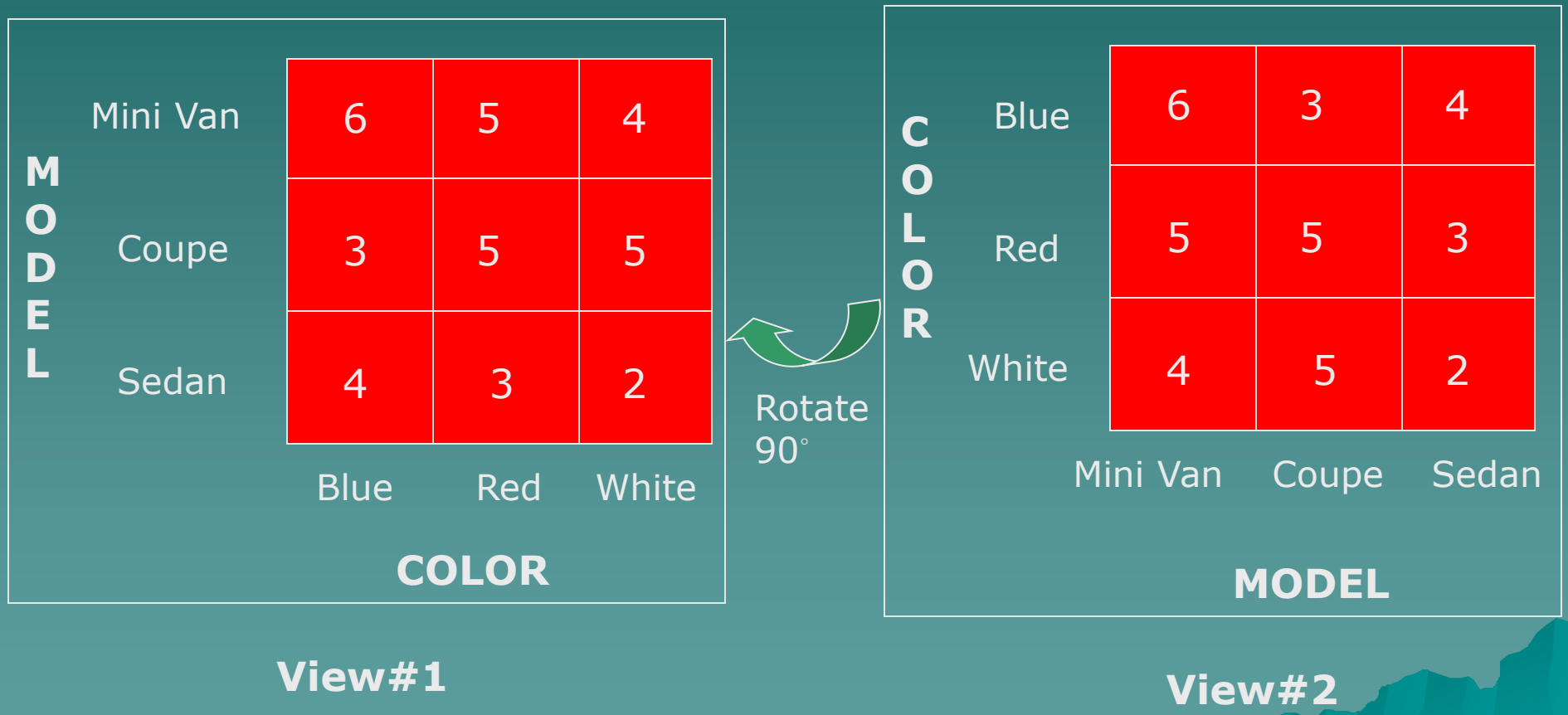
Example 2 (cont.): Corresponding Multidimensional Data Model



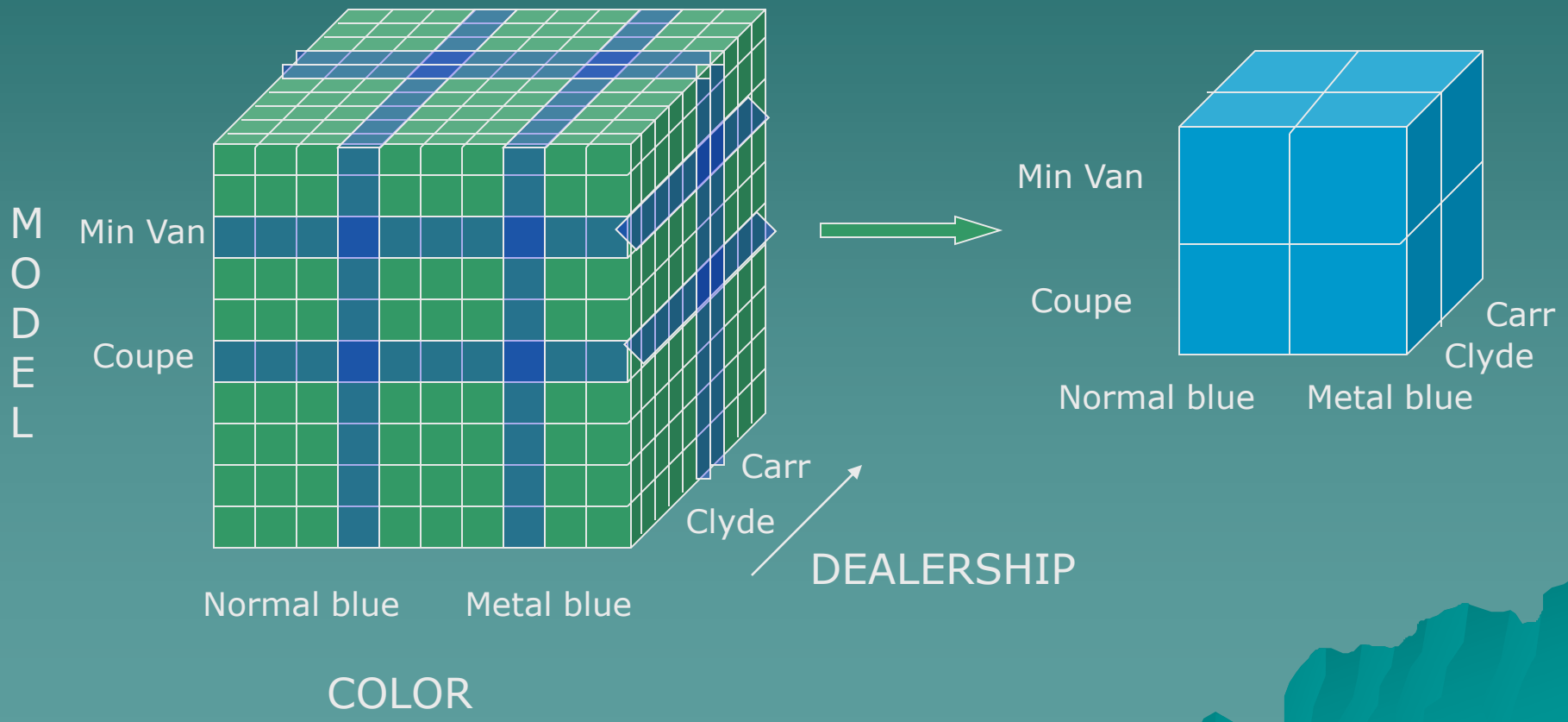
Example 2 (Cont.)

- Query: the user wants to know the sales volume figure when car type is sedan, color is blue, and dealership is Gleason. Consider a larger example: each dimension has 10 values.
- The corresponding relational table: 1000 records => may search 1000 records to answer
- The corresponding 10x10x10 array => search along three dimensions of 10 positions each => 30 position searches
- Multidimensional data views (OLAP operations): rotation, ranging, hierarchies, roll-ups, drill-downs, slice, dice

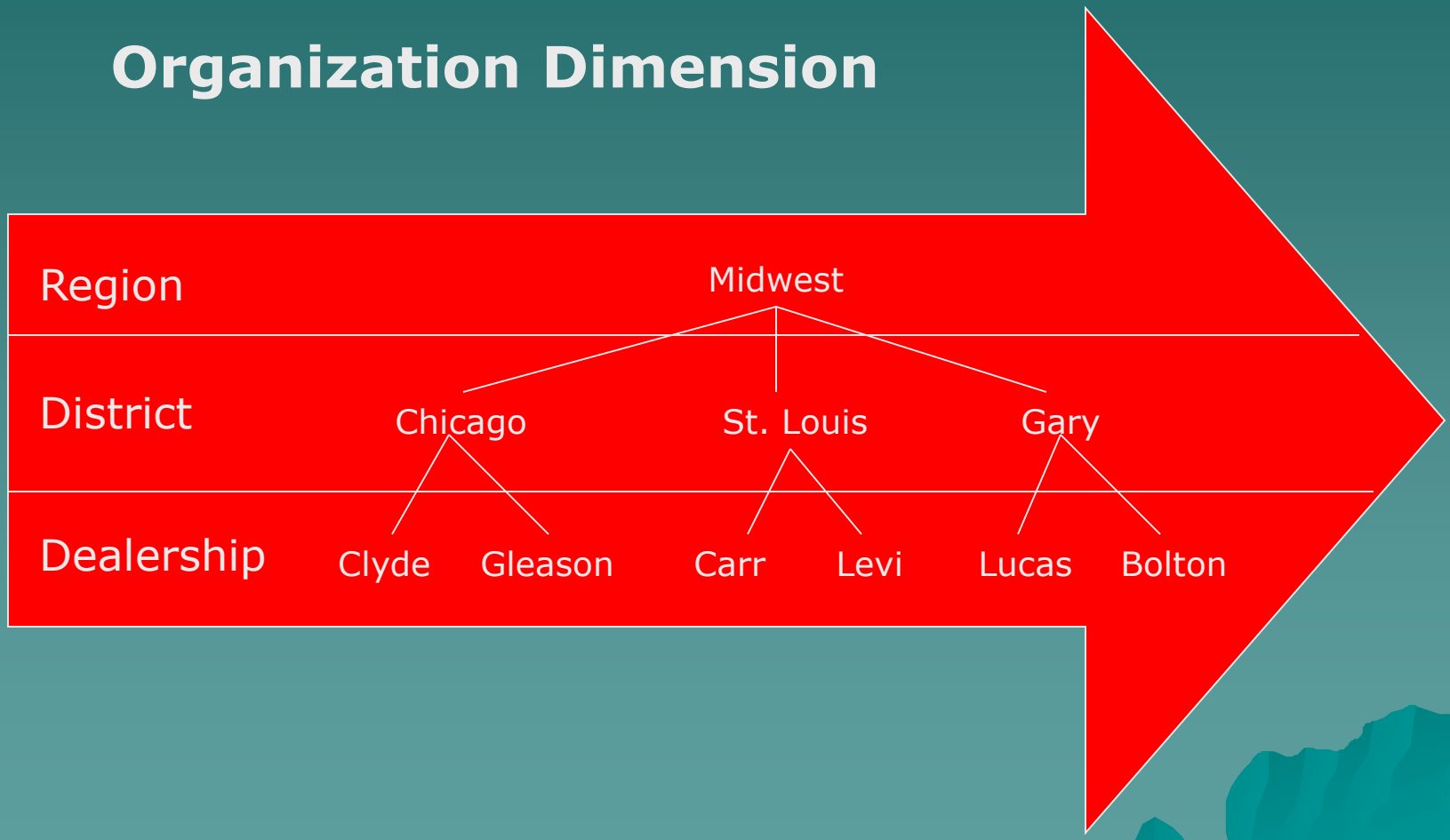
Example: Rotation



Example: Slice



Example: Hierarchies



Drill-Down

- *Drill-down* = "de-aggregate" = break an aggregate into its constituents.
- Example: having determined that Joe's Bar sells very few Anheuser-Busch beers, break down his sales by a particular Anheuser-Busch beer.

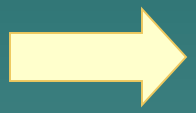
Roll-Up

- *Roll-up* = aggregate along one or more dimensions.
- Example: given a table of how much Bud each drinker consumes at each bar, roll it up into a table giving total amount of Bud consumed by each drinker.

Example: Roll Up and Drill Down

$ of Anheuser-Busch (A-B) beers by drinker/bar

	Jim	Bob	Mary
Joe's Bar	45	33	30
Nut-House	50	36	42
Blue Chalk	38	31	40



Roll up by Bar

$ of A-B beers / drinker

Jim	Bob	Mary
133	100	112

Drill down by Beer

$ of A-B beers / drinker

	Jim	Bob	Mary
Bud	40	29	40
M'lob	45	31	37
Bud Light	48	40	35

Partitioning Strategy (1)

- Performed for a number of performance-related and manageability reasons: fact tables can be very large => management, backup, and daily processing problems => partition each fact table into separate partitions.
- Horizontal Partitioning:
  - Partitioning by time into equal segments

    e.g. if the majority of user queries: month-to-date values => monthly segments
  - Partitioning by time into different-sized segments

    e.g. if aged data is accessed infrequently => small partitions for relatively current data, larger partitions for less active data.
  - Partitioning on a different dimension

    e.g. if each region tends to query on info. captured within its region => partition the fact table by region
  - Using round-robin partitions: archive the oldest partition prior to creating a new one and reuse the old partition for the latest data

Partitioning Strategy (2)

- Vertical Partitioning
  - Data is split vertically
  - Can be used to split less used column information out from a frequently accessed fact table.
- Which Key to Partition By?
  - If choose the wrong key => may have to totally reorganize the fact data.

    e.g. consider the following fact table

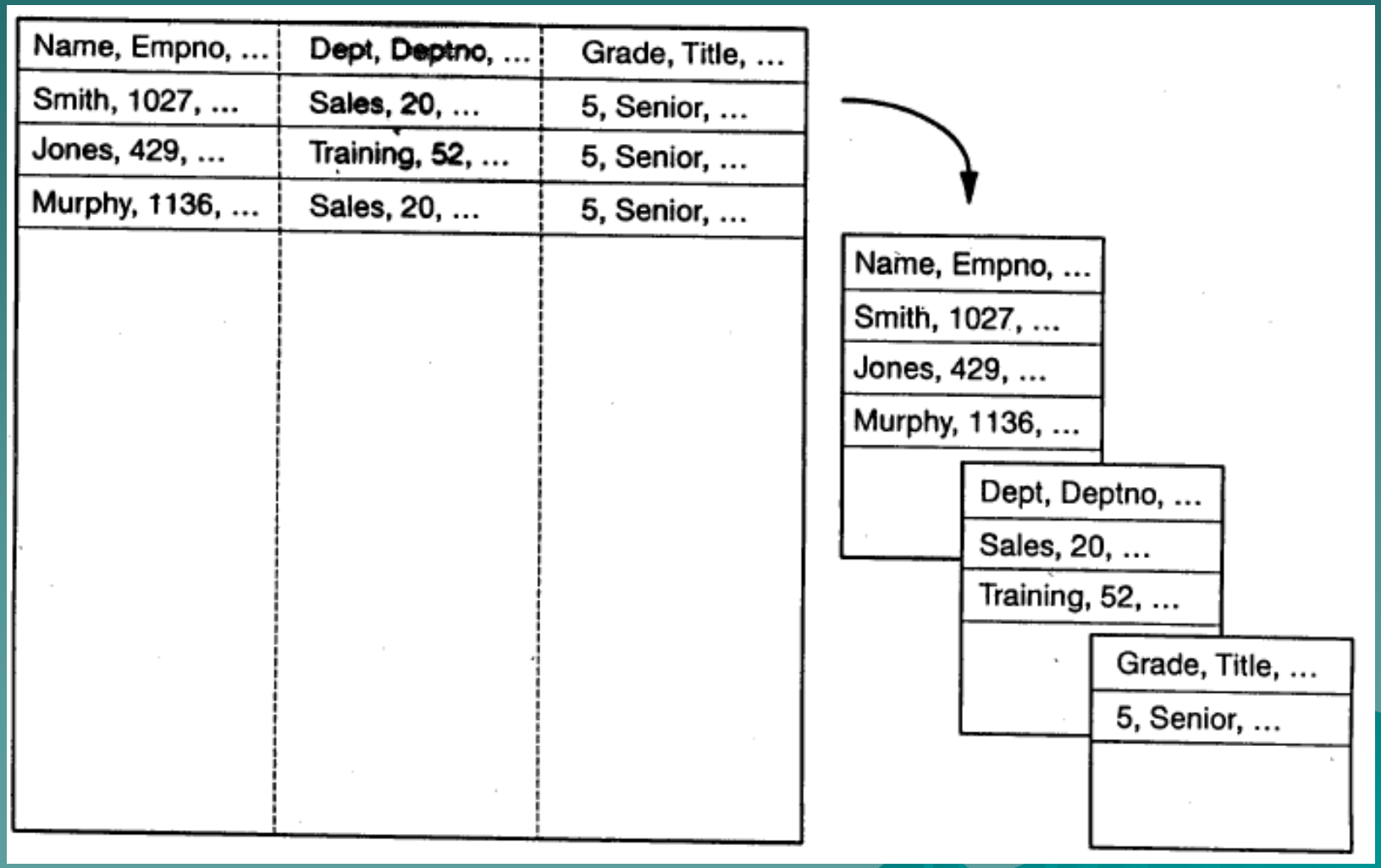
    Account_Transaction_Table
    - transaction_id
    - account_id
    - transaction_type
    - value
    - transaction_date
    - region
    - branch_name

    Assume the majority of queries are restricted to user's own region

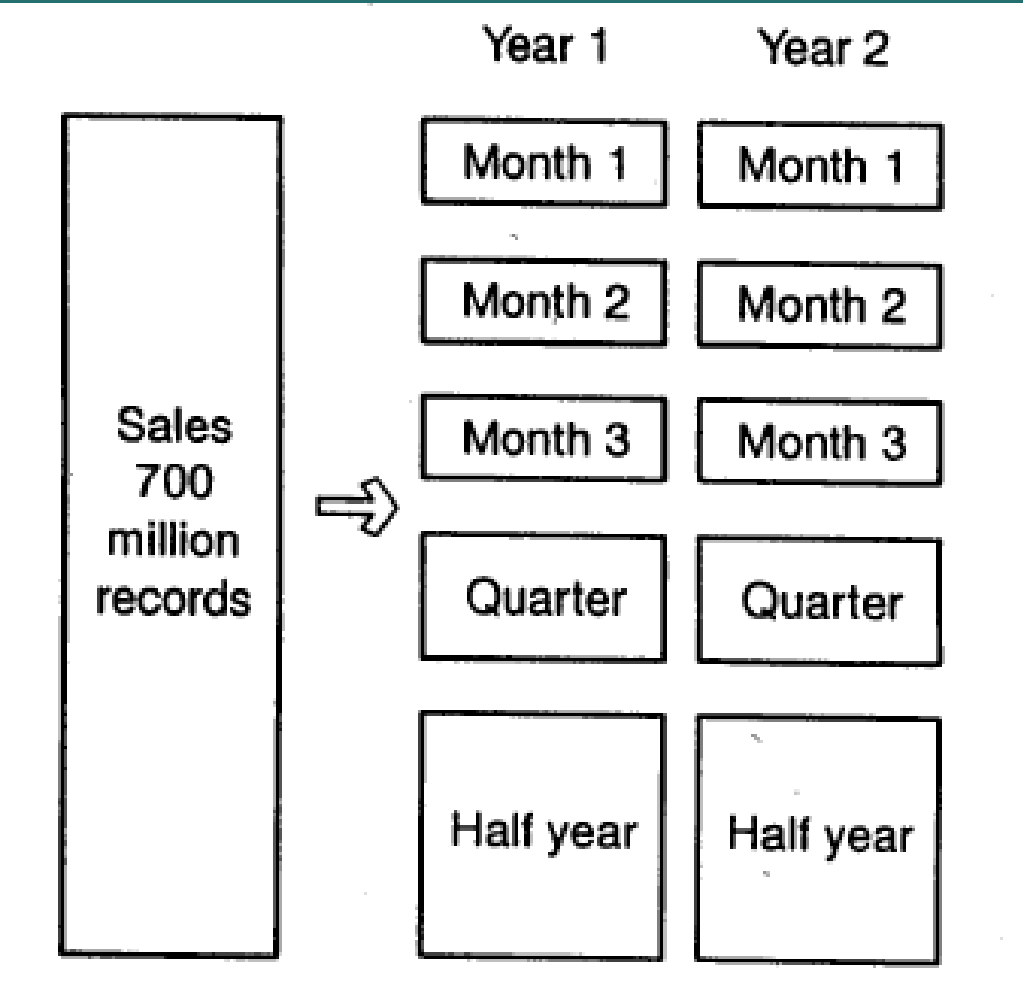
    if partition on region => OK

    if partition on transaction_date => latest transactions from every region will be in one partition => not OK
- Sizing the Partition
  - Considerations affecting the decision: backing up, managing, monitoring, moving, restoring, query performance...

Example of Vertical Partitioning



Partitioning Fact Tables into Different-sized Segments



Aggregations (Materialized Views) --Why Needed?

- Performed in order to speed up common queries

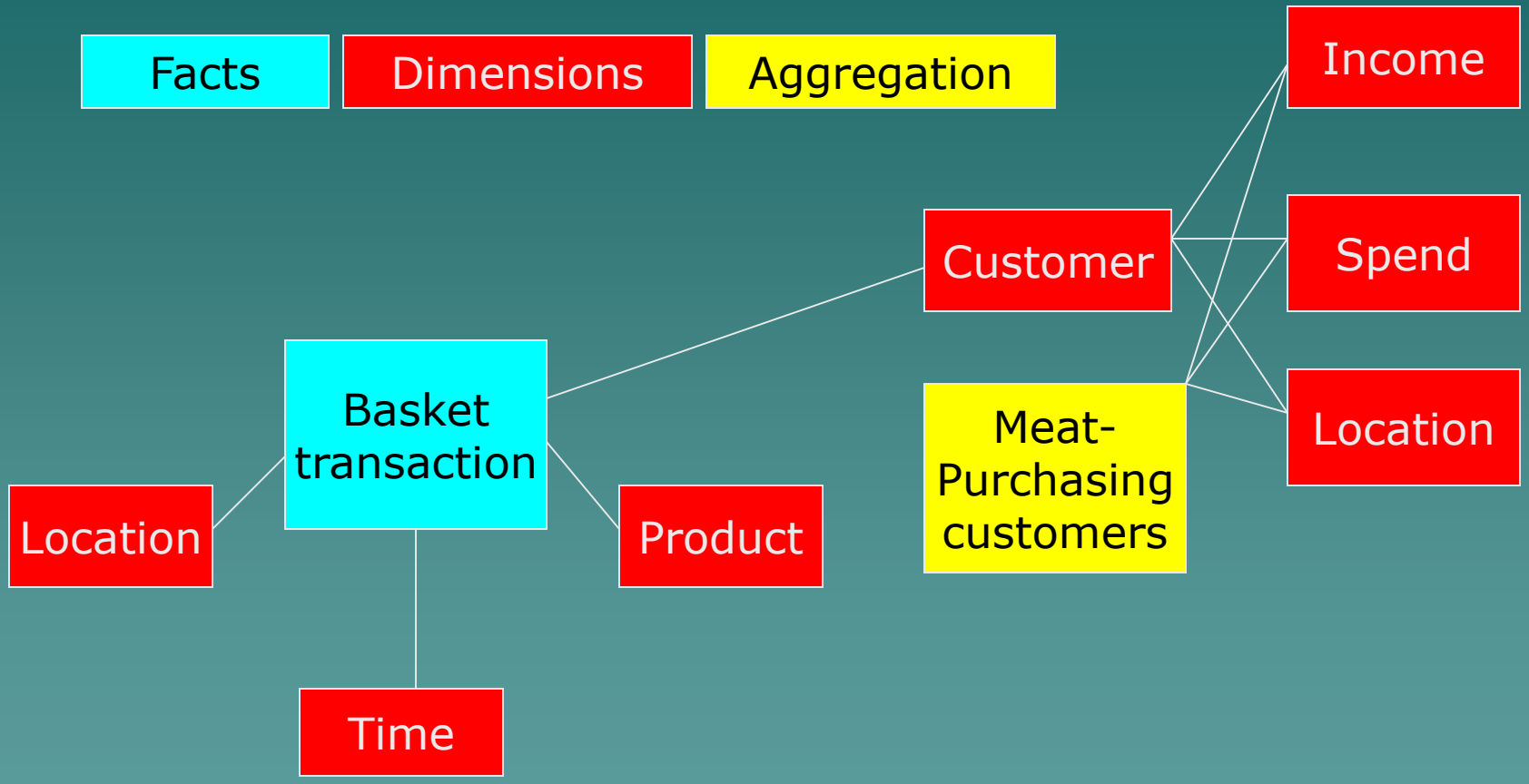
  e.g. A query to identify customers who purchased meat regularly over the past 2 years in the Orange County. Operations to perform in sequence:

  1. Identify customers within Orange County
  2. Identify the subset of customers who purchase meat regularly by scanning purchases over the past 2 years.
  3. Identify income and spend for each customer

  => speed up the query by having a pre-aggregated summary table that identifies all meat-buying customers in advance.
- The aggregations are geared around the popular queries at a specific point in time.
- The aggregations also make trends more apparent

  e.g. can analyze other trends about the meat-buying customers.

Example 1 of Aggregation

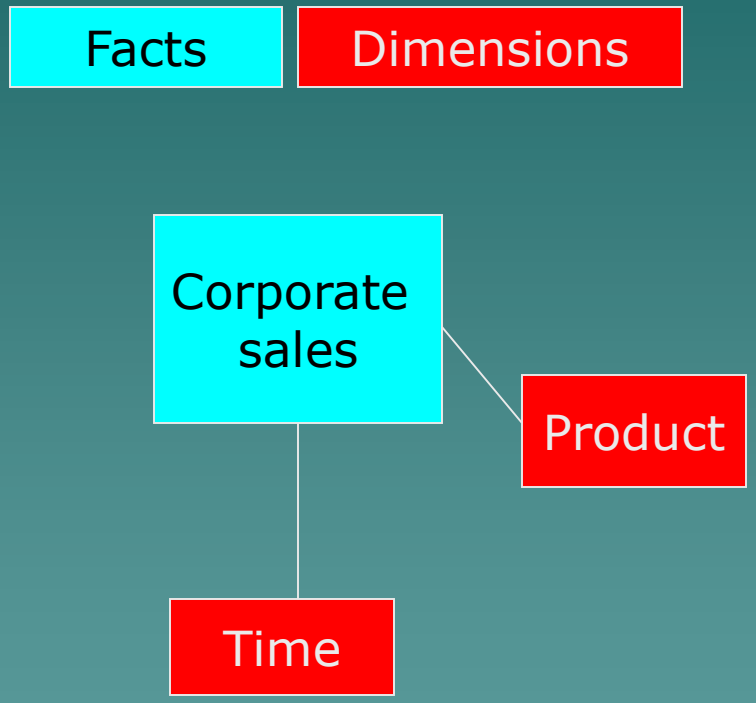


Summary table that identifies meat-purchasing customers

Aggregations--Designing Summary Tables

- Determine which dimensions should be aggregated
  - Summary tables should be created to make full use of the existing starflake schema structures by being designed to act as a pseudo fact table.
  - Summary tables should be designed to retain all the dimensions that have not been aggregated => subsuming the dimension
  - e.g. aggregate daily product sales data across the country => location is subsumed into the summary table

Example 2 of Aggregation



Starflake schema for a sales summary table aggregated by location

Determination of Aggregation (1)

- Determine the aggregation of multiple values

  e.g. want the following aggregated values for each product during each week:

  - total number of sales
  - peak number of daily sales
  - lowest number of sales
  - average number of sales

  Each of these items relates to aggregated values along the same dimension (e.g. product dimension)

  If majority of queries look at more than one of the aggregated values => store all these aggregations into a set of columns within the same summary table to speed up these queries

Determination of Aggregation (2)

- Determine the level of aggregation
  - e.g. weekly aggregation v.s. monthly aggregation of customer transactions. (aggregate at a level below the one required and then aggregate up on fly).
- Large summary tables should be partitioned.
- Which summaries to create: must take into account queries in the workload, their frequencies, and their costs.

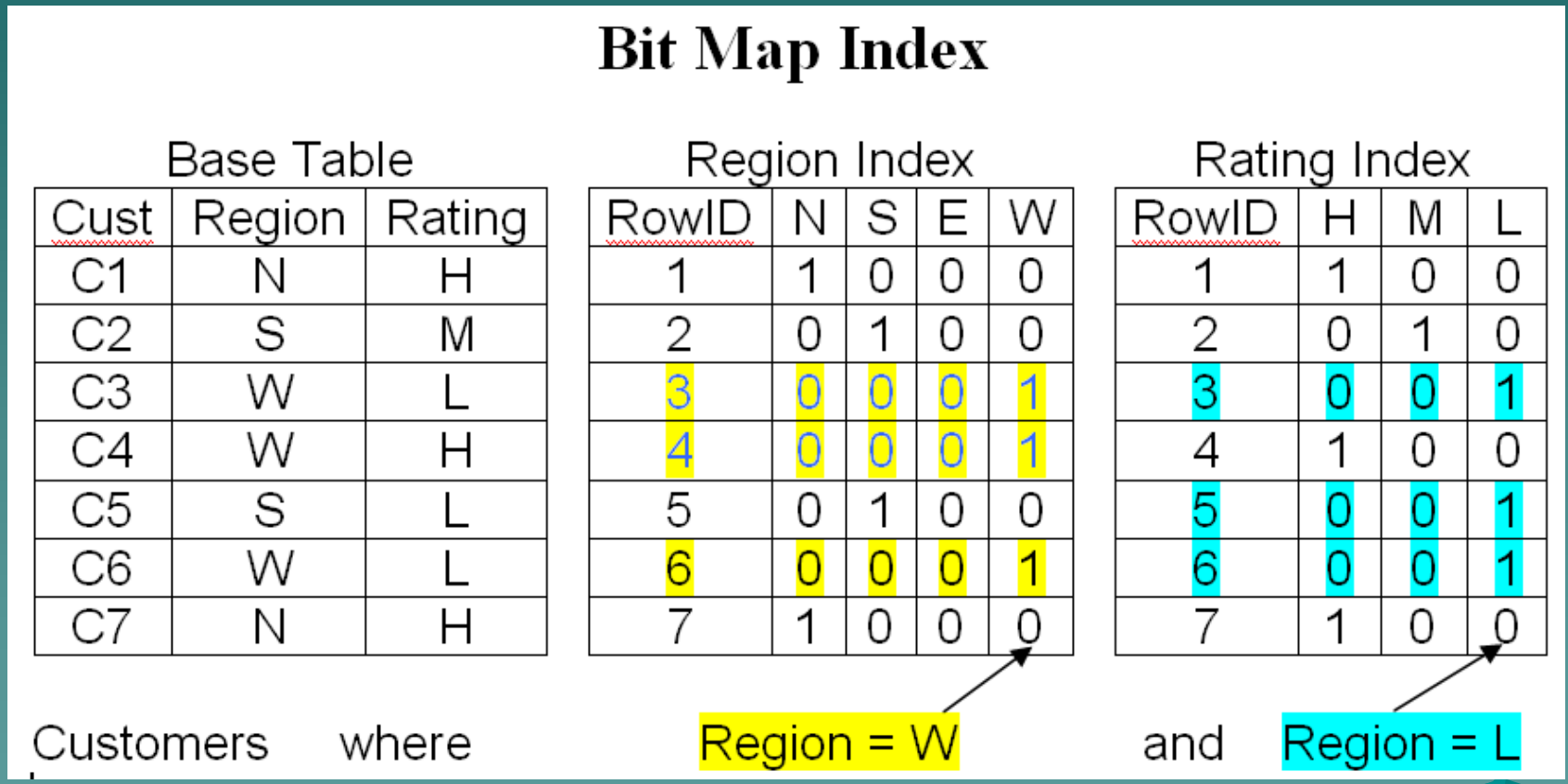
Indexing

- Exploiting indexes to reduce scanning of data
- Bitmap indexes
- Join Indexes

Bitmap Indexes

- Specially advantageous for low-cardinality attributes
- There is a distinct bit vector Bv for each value v in the attribute's domain
  - e.g. The attribute sex has values M and F. A table of 100 million people needs 2 bit vectors of 100 million bits each
  - If the attribute has the value v for a given row in the data table, then the bit representing that value is set to 1 in the corresponding row of the bitmap index; all other bits for that row are set to 0.
- Comparison, join, and aggregation operations are reduced to bit arithmetic with dramatic improvement in processing time.
- Significant reduction in space and I/O

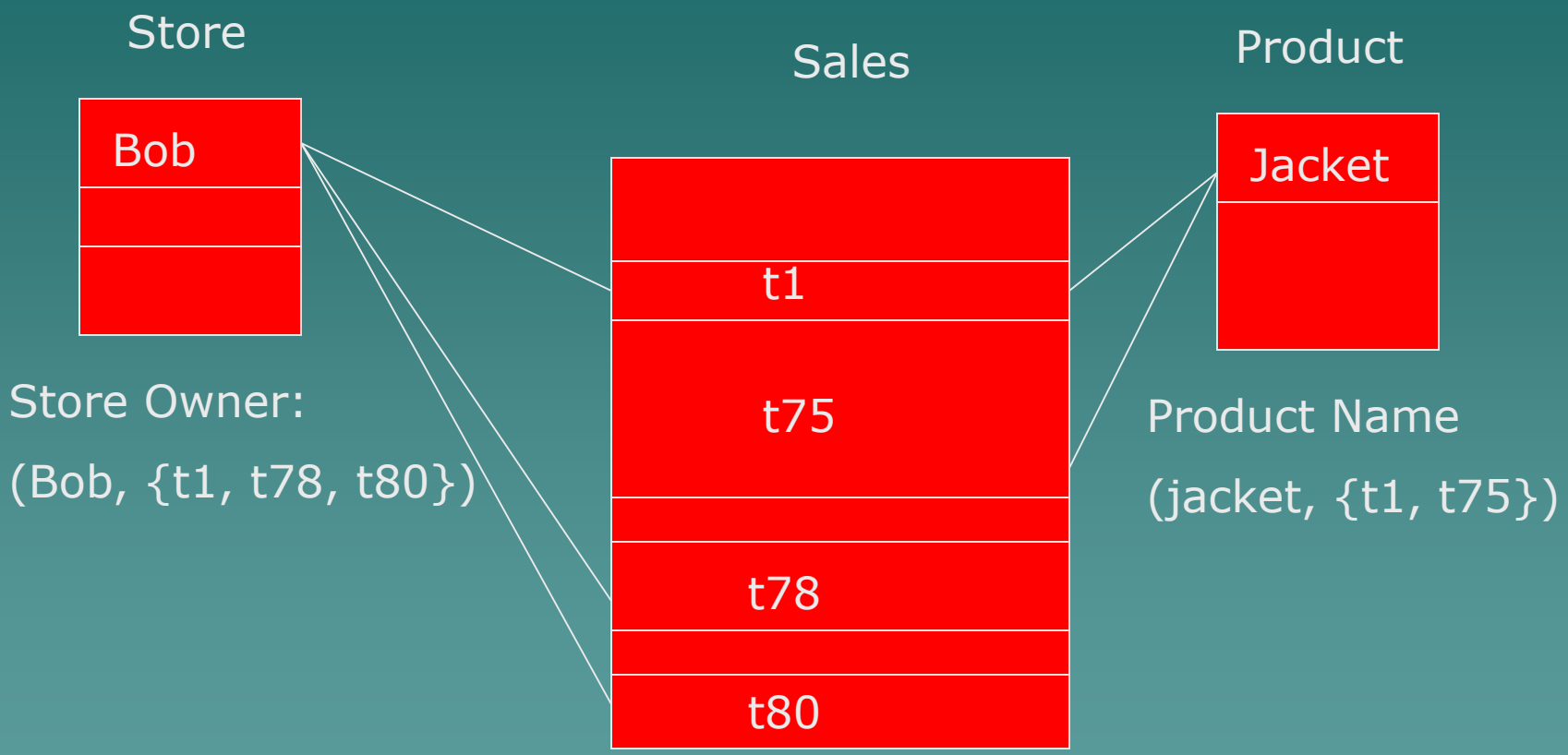
Example of Bitmap Index



Join Indexes

- Traditional indexes map the value in a column to a list of rows with that value.
- Join indexes maintain relationship between attribute value of a dimension and the matching rows in the fact table
- Join indexes may span multiple dimensions (composite join indexes)

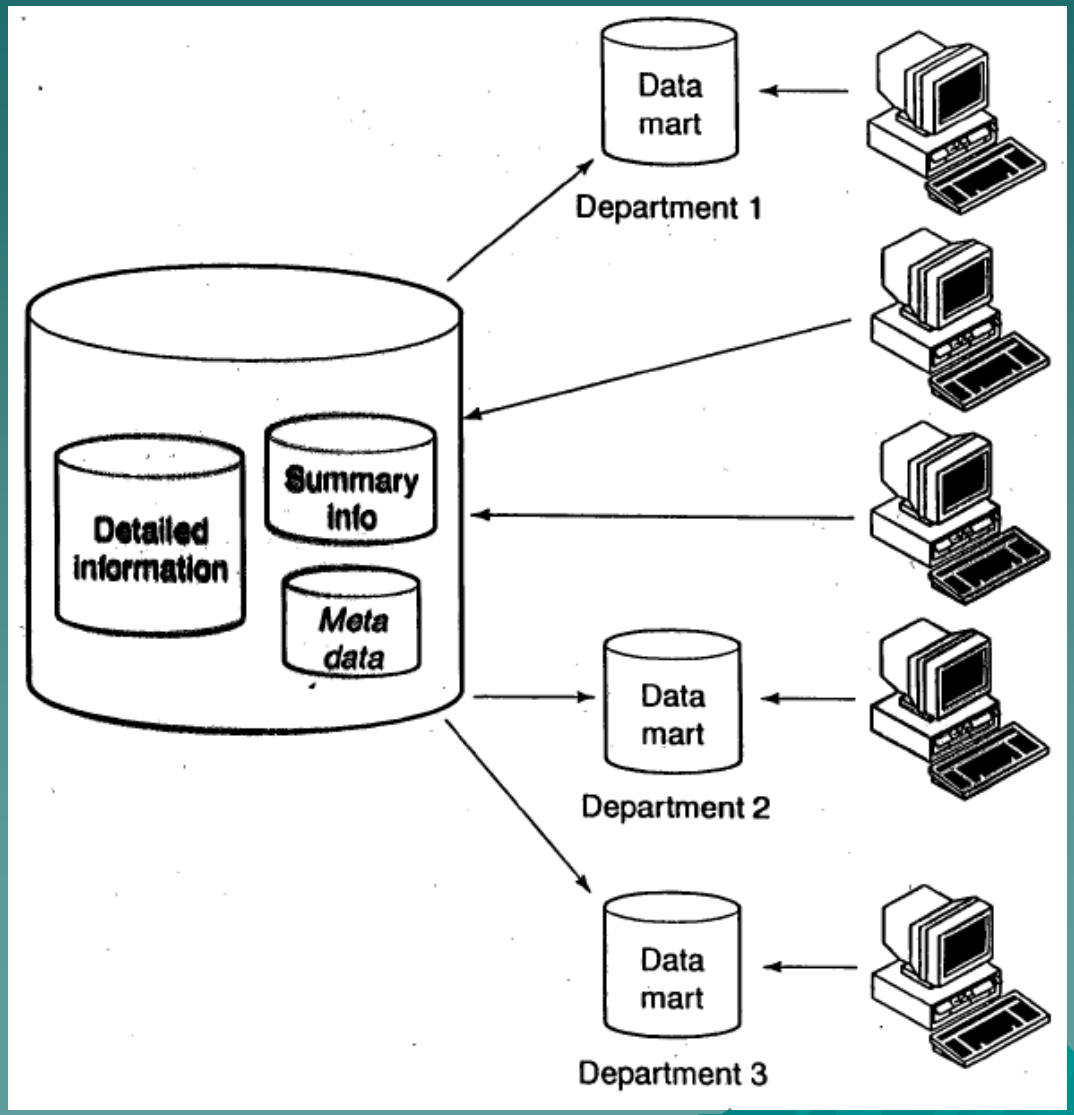
Example of Join Index



Data Marting (1)

- Data marts hive off a subset of the data in the data warehouse, into a separate database, possibly in a different locations
- Data marts are created for the following reasons:
  - To speed up queries by reducing the volume of data to be scanned.
  - To structure data in a form suitable for a user access tool.
  - To partition data in order to impose access control strategies.
  - To segment data into different hardware platforms

Example of Data Marting



Data Marting (2)

- Steps to decide if a data mart is appropriate:
  - Identify whether there is a natural functional split within the organization and whether there is natural split of the data.
    - determine department splits, its use of data, business benefits and technical feasibility.
  - Identify whether the proposed user access tool uses its own database structures
    - Data within these structures is outside the control of the data warehouse, but they need to be populated and updated on a regular basis.
  - Identify access control issues
    - information within the data warehouse may be subject to privacy laws => data marts allow us to separate segments of data in the data warehouse.

Data Marting (3)

- Cost of data marting
  - Hardware and software
  - network access
  - Time window constraints
    - calculate the data volumes being shipped into each data mart.
    - calculate elapsed time to transfer across the LAN or WAN into each mart
    - calculate elapsed time to load the data into each mart.
    - allow additional elapsed time if the load process is performing data transformations.

Maintenance

- Data extraction
- Data cleaning
- Data transformation
  - Convert from host format to warehouse format
- Load
  - Sort, summarize, consolidate, compute views, check integrity, build indexes, partition
- Refresh
  - Propagate updates from sources to the warehouse

Maintenance--Data Cleaning

- Why?
  - Data warehouse contains data that is analyzed for business decision
  - More data and multiple sources could mean more errors in the data and harder to trace such errors.
  - Results in incorrect analysis
- Detecting data anomalies and rectifying them early has huge payoffs.
- Data cleaning techniques:
  - Transformation rules: e.g. translate “gender” to “sex”
  - Uses domain-specific knowledge to do scrubbing
  - Parsing and fuzzy matching
  - Discover facts that flag unusual patterns

Maintenance--Load

- Issues:
  - Huge volumes of data to be loaded
  - Small time window (usually when the system load is low) when the warehouse can be taken off-line
  - When to build indexes and summary tables
  - Allow system administrator to monitor status, cancel suspend, resume load, or change load rate.
  - Restart after failure with no loss of data integrity.
- Techniques:
  - Batch load utility: sort input records on clustering key and use sequential I/O; build indexes and derived tables
  - Sequential load may be too long
  - Use parallelism and incremental techniques

Maintenance--Refresh

- When to refresh?
  - On every update: too expensive, only necessary if OLAP queries need current data (e.g. up-to-the-minute stock quotes)
  - Periodically or after significant events
  - Refresh policy set by the administrator based on user needs and traffic
  - Possibly different policies for different sources
- Refresh Techniques
  - Full extract from base tables
  - Incremental techniques
    - Detect and propagate changes on base tables
    - Logical correctness: computing changes to star tables, computing changes to summary tables, optimization: only significant changes

Metadata Repository

- Administrative metadata
  - source databases and their contents
  - warehouse schema, and view definitions
  - hierarchies
  - pre-defined queries
  - data mart locations and contents
  - data partitions
  - data extraction, cleansing, transformation rules
  - data refresh and purging rules
  - user profiles
  - security: user authorization, access control
- Business data
  - business terms and definitions
  - ownership of data
  - charging policies
- Operational metadata
  - history of migrated data and sequence of transformations applied
  - currency of data: active, archived, purged
  - monitoring information: warehouse usage statistics, error reports.

Research Issues

- Data cleaning
  - Use of data mining techniques
- Physical design
  - Summary tables, partitions, indexes
- Query processing
  - Selecting appropriate summary tables
- Warehouse management
  - Failure recovery during load and refresh
  - Scheduling queries, load, and refresh
  - Computing summary tables during load

THE END