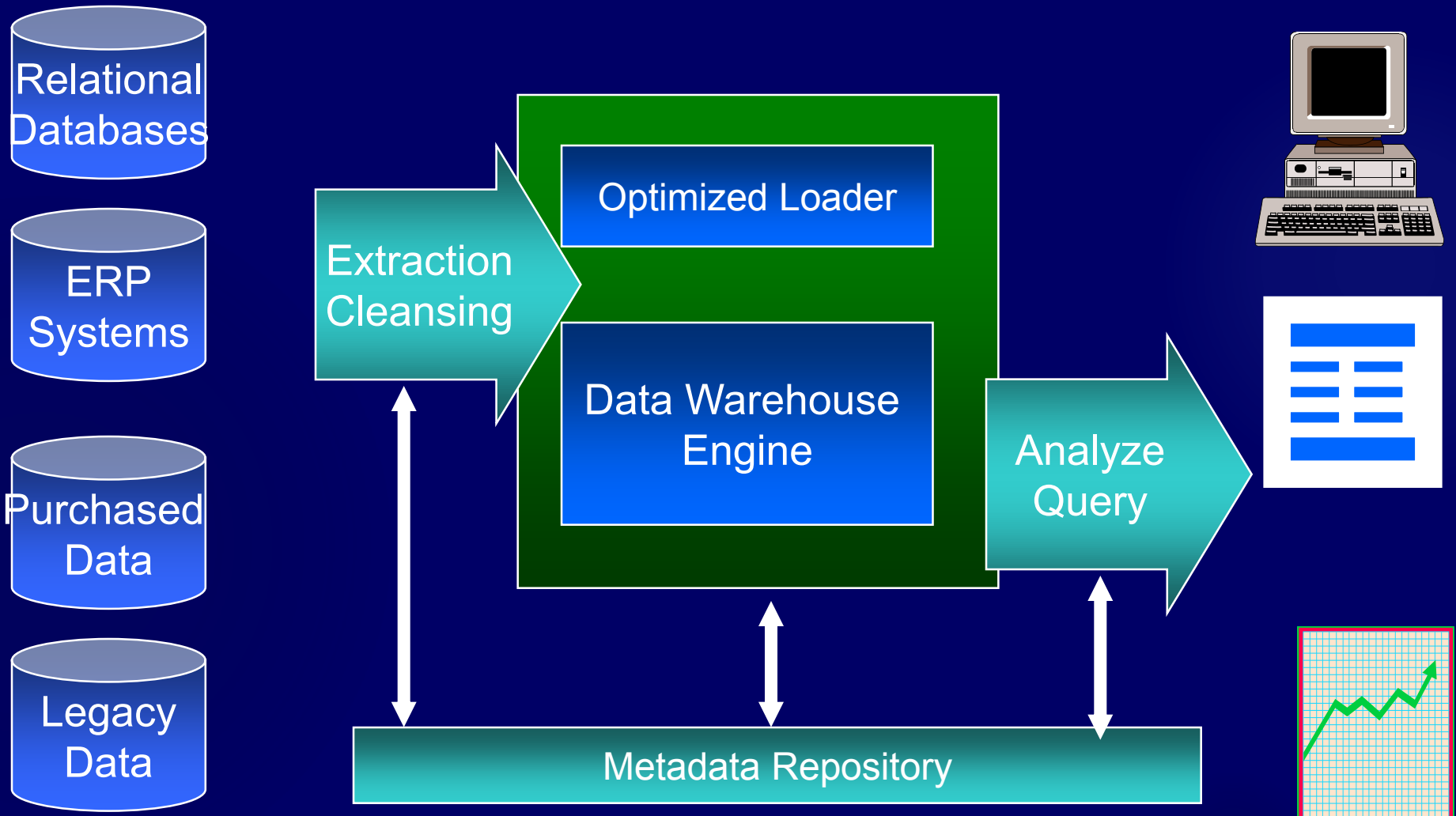
Course Overview

- 0. Introduction
- I. **Data Warehousing**
- II. Decision Support and OLAP
- III. Data Mining
- IV. Looking Ahead

- Demos and Labs

Data Warehouse Architecture



Components of the Warehouse

- Data Extraction and Loading
- The Warehouse
- Analyze and Query -- OLAP Tools
- Metadata

- Data Mining tools

Loading the Warehouse

- Cleaning the data before it is loaded

Source Data

- Typically host based, legacy applications
  - Customized applications, COBOL, 3GL, 4GL
- Point of Contact Devices
  - POS, ATM, Call switches
- External Sources
  - Nielsen's, Acxiom, CMIE, Vendors, Partners

Data Quality - The Reality

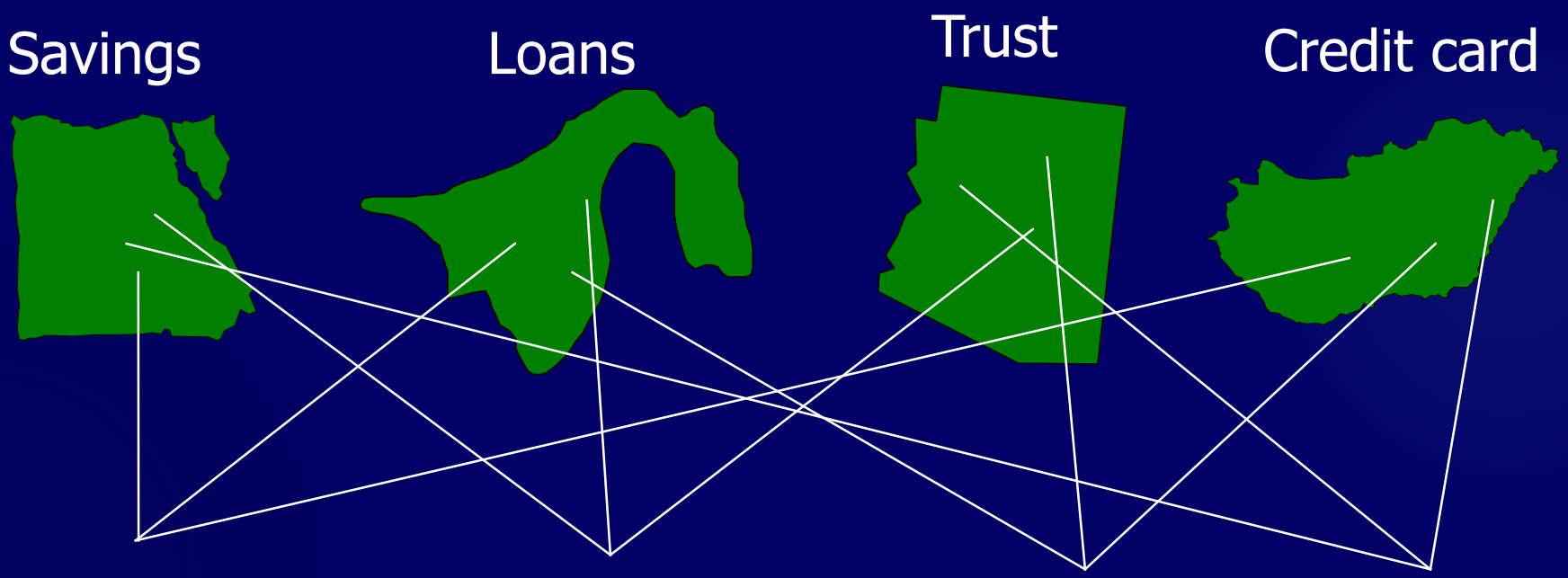
- Tempting to think creating a data warehouse is simply extracting operational data and entering into a data warehouse

- Nothing could be farther from the truth
- Warehouse data comes from disparate questionable sources

Data Quality - The Reality

- Legacy systems no longer documented
- Outside sources with questionable quality procedures
- Production systems with no built in integrity checks and no integration
  - Operational systems are usually designed to solve a specific business problem and are rarely developed to a a corporate plan
    - "And get it done quickly, we do not have time to worry about corporate standards..."

Data Integration Across Sources



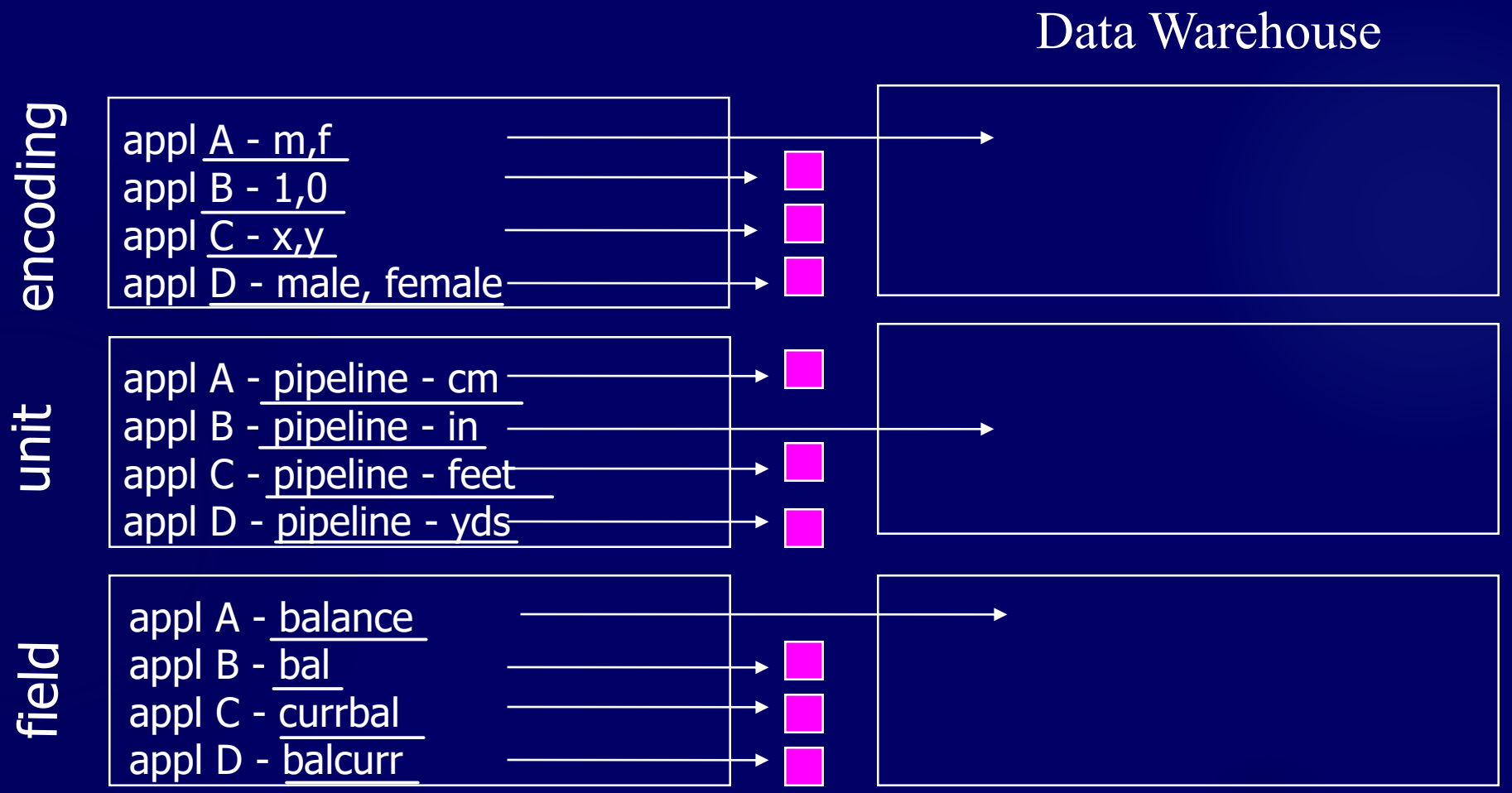
Same data different name

Different data Same name

Data found here nowhere else

Different keys same data

Data Transformation Example



Data Integrity Problems

- Same person, different spellings
  - Agarwal, Agrawal, Aggarwal etc...
- Multiple ways to denote company name
  - Persistent Systems, PSPL, Persistent Pvt. LTD.
- Use of different names
  - mumbai, bombay
- Different account numbers generated by different applications for the same customer
- Required fields left blank
- Invalid product codes collected at point of sale
  - manual entry leads to mistakes
  - "in case of a problem use 9999999"

Data Transformation Terms

- Extracting
- Conditioning
- Scrubbing
- Merging
- Householding
- Enrichment
- Scoring
- Loading
- Validating
- Delta Updating

Data Transformation Terms

- Extracting
  - Capture of data from operational source in "as is" status
  - Sources for data generally in legacy mainframes in VSAM, IMS, IDMS, DB2; more data today in relational databases on Unix
- Conditioning
  - The conversion of data types from the source to the target data store (warehouse) -- always a relational database

Data Transformation Terms

- Householding
  - Identifying all members of a household (living at the same address)
  - Ensures only one mail is sent to a household
  - Can result in substantial savings: 1 lakh catalogues at Rs. 50 each costs Rs. 50 lakhs. A 2% savings would save Rs. 1 lakh.

Data Transformation Terms

- Enrichment
  - Bring data from external sources to augment/enrich operational data. Data sources include Dunn and Bradstreet, A. C. Nielsen, CMIE, IMRA etc...
- Scoring
  - computation of a probability of an event. e.g..., chance that a customer will defect to AT&T from MCI, chance that a customer is likely to buy a new product

Loads

- After extracting, scrubbing, cleaning, validating etc. need to load the data into the warehouse
- Issues
  - huge volumes of data to be loaded → LOAD DATA - Schema on Read
  - small time window available when warehouse can be taken off line (usually nights)
  - when to build index and summary tables | materialized view.
  - allow system administrators to monitor, cancel, resume, change load rates
  - Recover gracefully -- restart after failure from where you were and without loss of data integrity

Load Techniques

- Use SQL to append or insert new data
  - record at a time interface
  - will lead to random disk I/O's
- Use batch load utility

Load Taxonomy

- Incremental versus Full loads
- Online versus Offline loads

Refresh

- Propagate updates on source data to the warehouse
- Issues:
  - when to refresh
  - how to refresh -- refresh techniques

When to Refresh?

- periodically (e.g., every night, every week) or after significant events
- on every update: not warranted unless warehouse data require current data (up to the minute stock quotes)
- refresh policy set by administrator based on user needs and traffic
- possibly different policies for different sources

Refresh Techniques

- Full Extract from base tables
  - read entire source table: too expensive
  - maybe the only choice for legacy systems

How To Detect Changes

- Create a snapshot log table to record ids of updated rows of source data and timestamp
- Detect changes by:
  - Defining after row triggers to update snapshot log when source table changes
  - Using regular transaction log to detect changes to source data

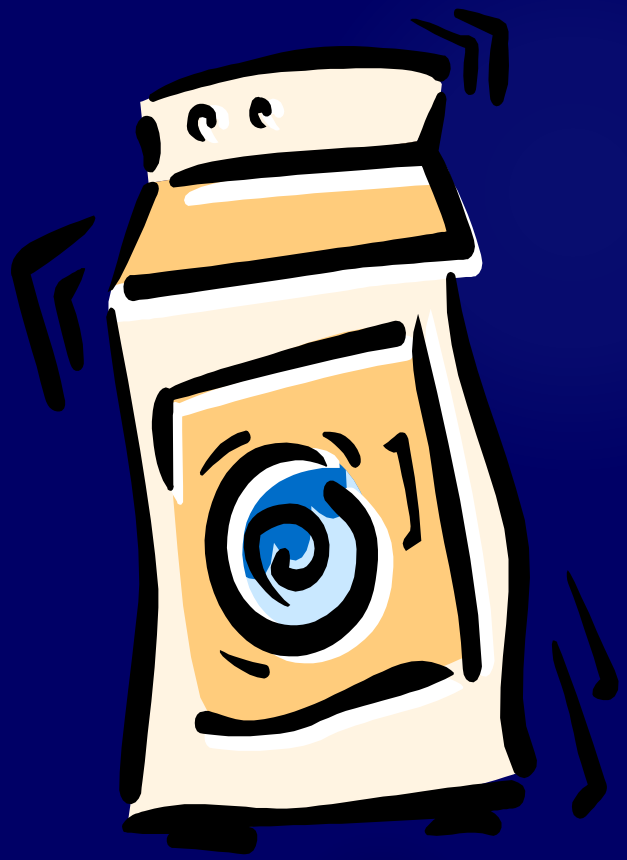
Sqoop import
↳ incremental import
↳ by id
↳ by timestamp

Data Extraction and Cleansing

- Extract data from existing operational and legacy data
- Issues:
  - Sources of data for the warehouse
  - Data quality at the sources
  - Merging different data sources
  - Data Transformation
  - How to propagate updates (on the sources) to the warehouse
  - Terabytes of data to be loaded

Scrubbing Data

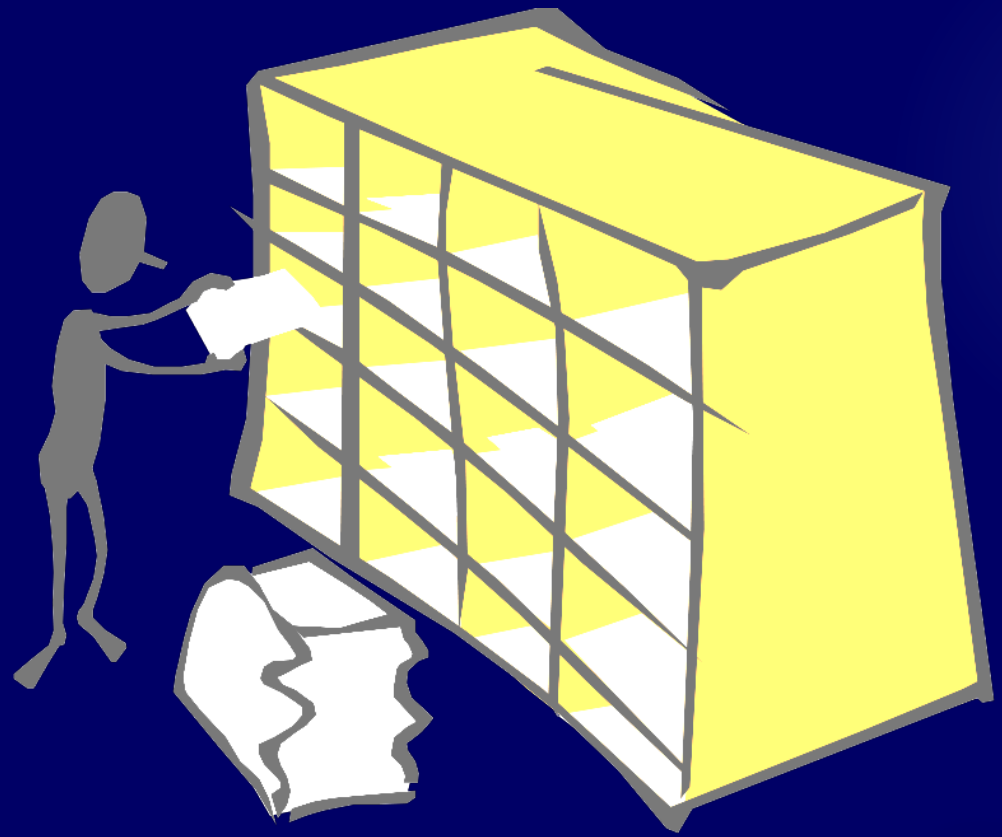
- Sophisticated transformation tools.
- Used for cleaning the quality of data
- Clean data is vital for the success of the warehouse
- Example
  - Seshadri, Sheshadri, Sesadri, Seshadri S., Srinivasan Seshadri, etc. are the same person



Scrubbing Tools

- Apertus -- Enterprise/Integrator
- Vality -- IPE
- Postal Soft

Structuring/Modeling Issues



Data -- Heart of the Data Warehouse

- Heart of the data warehouse is the data itself!
- Single version of the truth
- Corporate memory
- Data is organized in a way that represents business -- subject orientation

Data Warehouse Structure

- Subject Orientation -- customer, product, policy, account etc... A subject may be implemented as a set of related tables. E.g., customer may be five tables

Data Warehouse Structure

Time is part of key of each table

- base customer (1985-87)
  - custid, from date, to date, name, phone, dob
- base customer (1988-90)
  - custid, from date, to date, name, credit rating, employer
- customer activity (1986-89) -- monthly summary
- customer activity detail (1987-89)
  - custid, activity date, amount, clerk id, order no
- customer activity detail (1990-91)
  - custid, activity date, amount, line item no, order no

Data Granularity in Warehouse

- Summarized data stored
  - reduce storage costs
  - reduce cpu usage
  - increases performance since smaller number of records to be processed
  - design around traditional high level reporting needs
  - tradeoff with volume of data to be stored and detailed usage of data

Granularity in Warehouse

- Can not answer some questions with summarized data
  - Did Anand call Seshadri last month? Not possible to answer if total duration of calls by Anand over a month is only maintained and individual call details are not.
- Detailed data too voluminous

Granularity in Warehouse

- Tradeoff is to have dual level of granularity
  - Store summary data on disks
    - 95% of DSS processing done against this data
  - Store detail on tapes
    - 5% of DSS processing against this data

Vertical Partitioning

Acct. No	Name	Balance	Date Opened	Interest Rate	Address

Frequently accessed

Acct. No	Balance

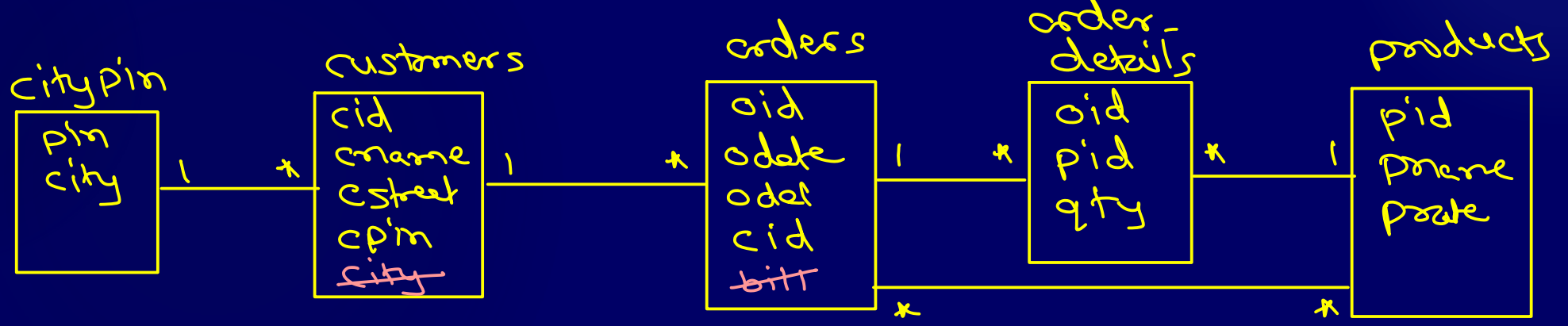
Rarely accessed

Acct. No	Name	Date Opened	Interest Rate	Address

Smaller table
and so less I/O

Derived Data

- Introduction of derived (calculated data) may often help
- Have seen this in the context of dual levels of granularity
- Can keep auxiliary views and indexes to speed up query processing



Schema Design

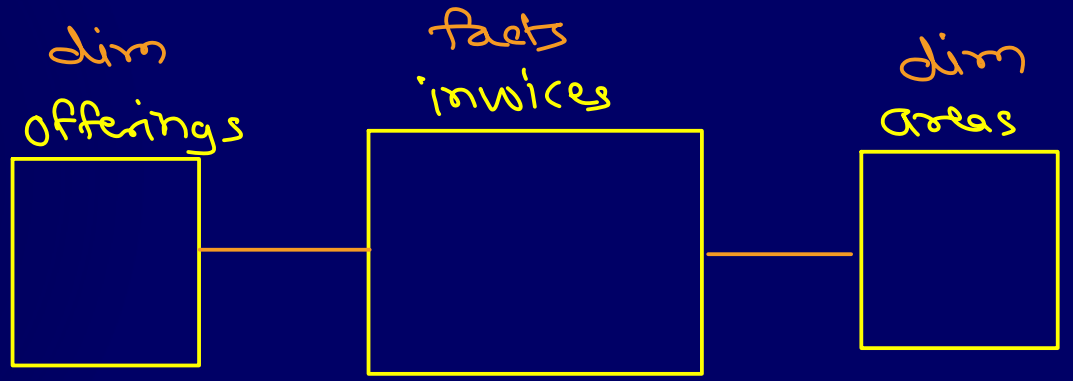
- Database organization
  - must look like business
  - must be recognizable by business user
  - approachable by business user
  - Must be *simple*
- Schema Types
  - Star Schema
  - Fact Constellation Schema
  - Snowflake schema

Dimension Tables

- Dimension tables
  - Define business in terms already familiar to users
  - Wide rows with lots of descriptive text
  - Small tables (about a million rows)
  - Joined to fact table by a foreign key
  - heavily indexed
  - typical dimensions
    - time periods, geographic region (markets, cities), products, customers, salesperson, etc.

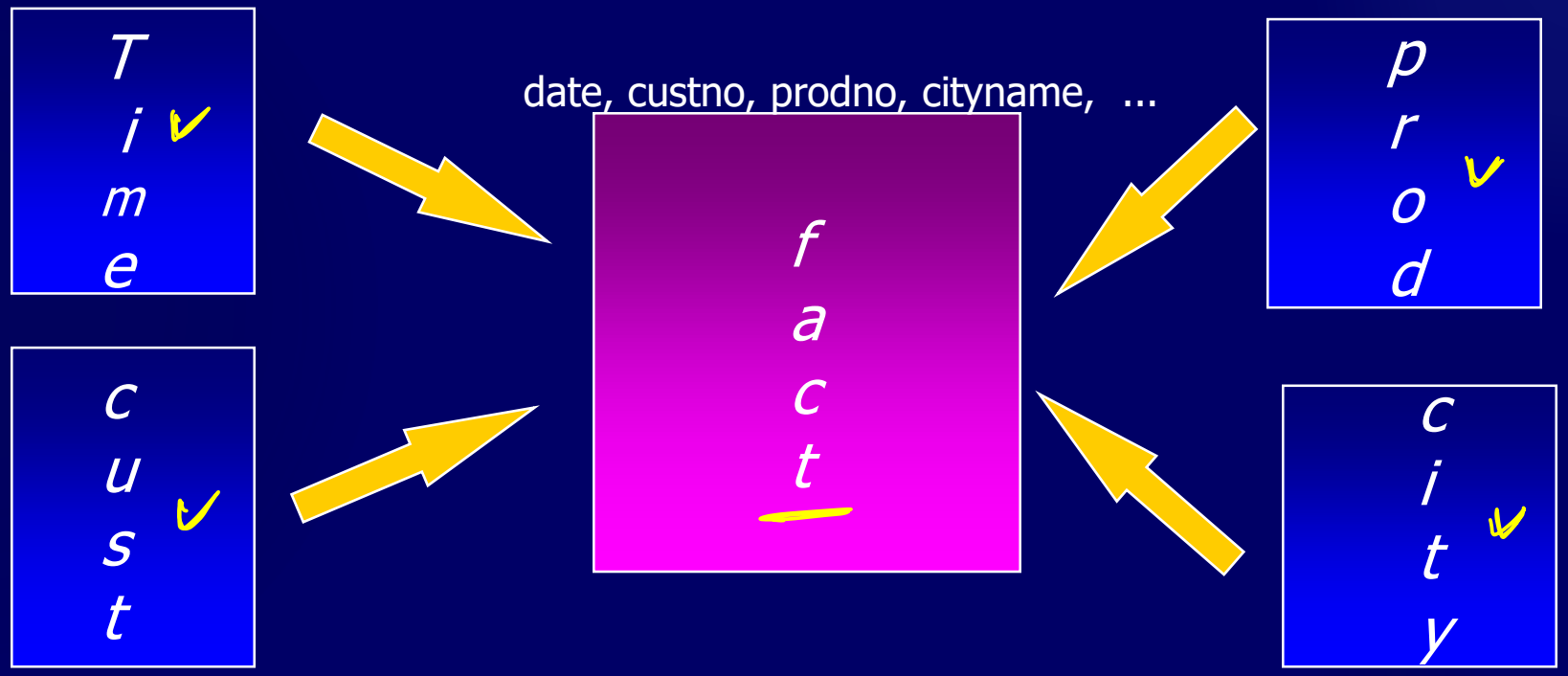
Fact Table

- Central table
  - mostly raw numeric items
  - narrow rows, a few columns at most
  - large number of rows (millions to a billion)
  - Access via dimensions



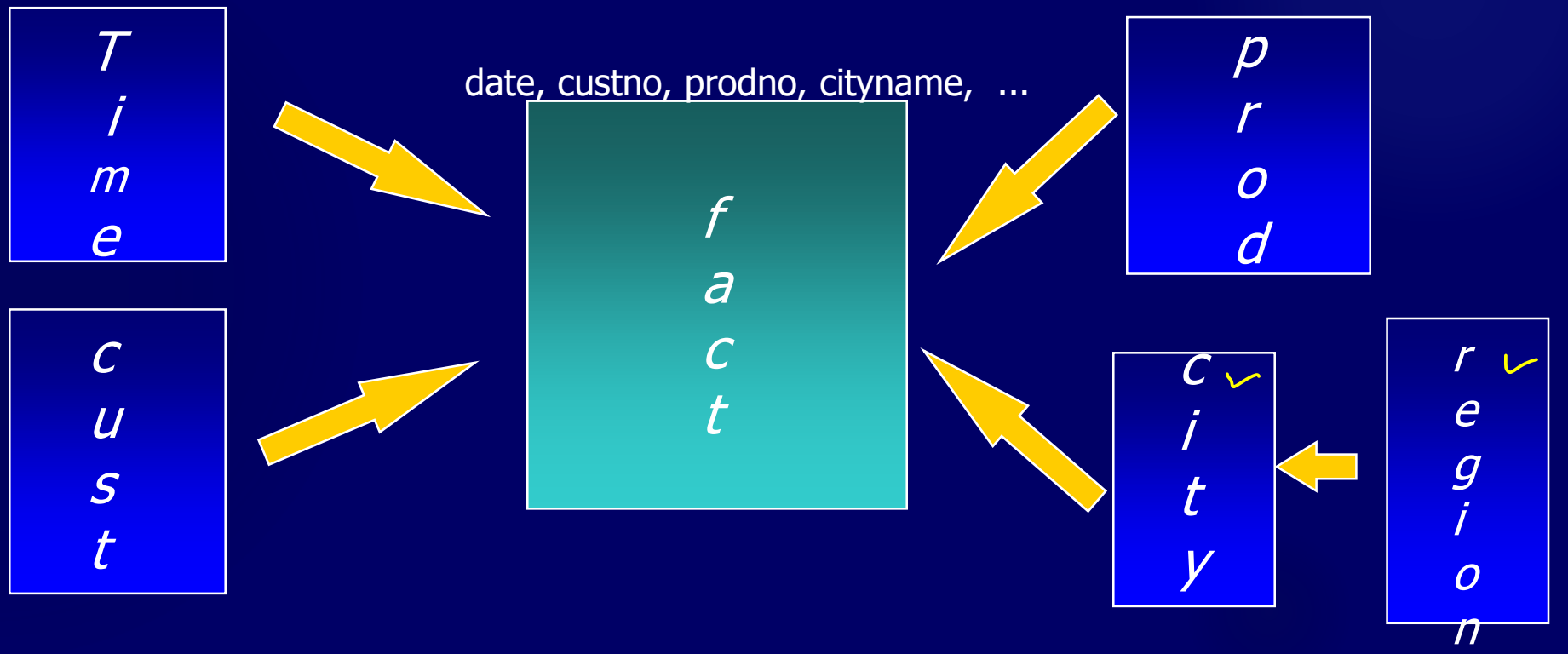
Star Schema

- A single fact table and for each dimension one dimension table
- Does not capture hierarchies directly



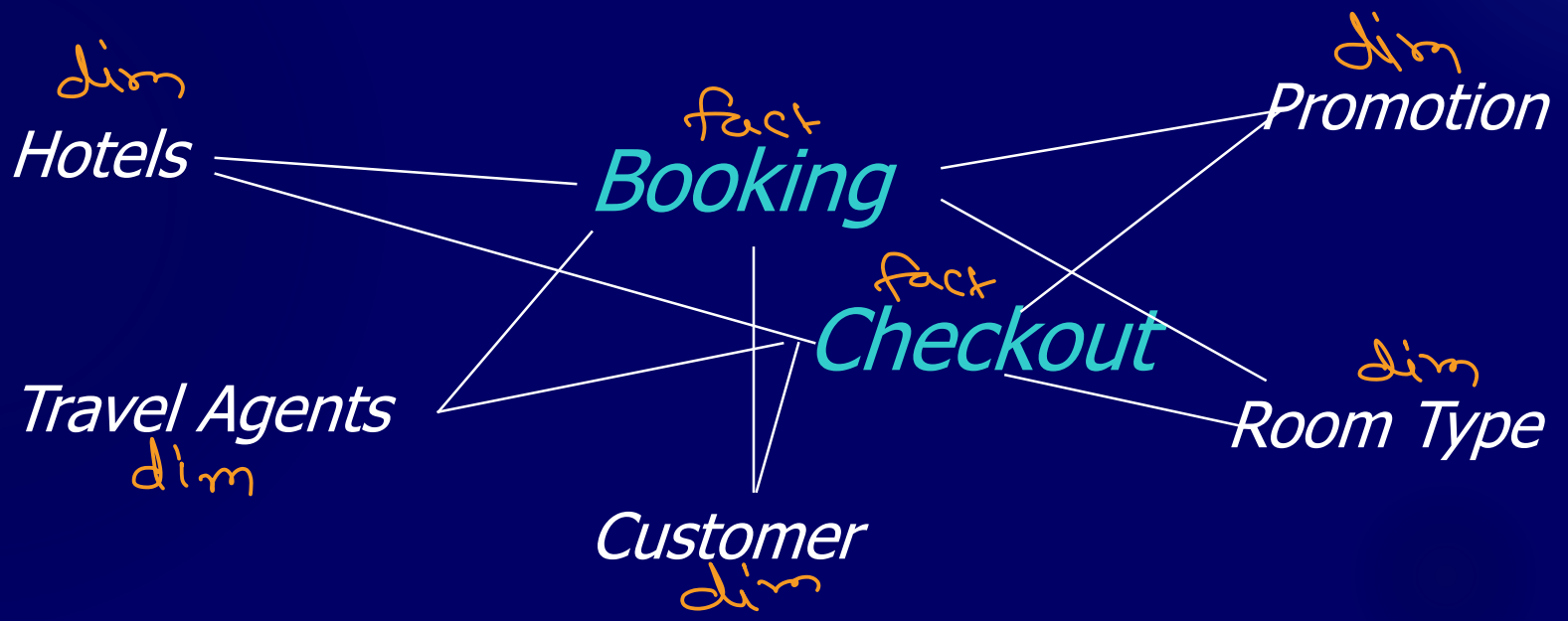
Snowflake schema

- Represent dimensional hierarchy directly by normalizing tables.
- Easy to maintain and saves storage



Fact Constellation

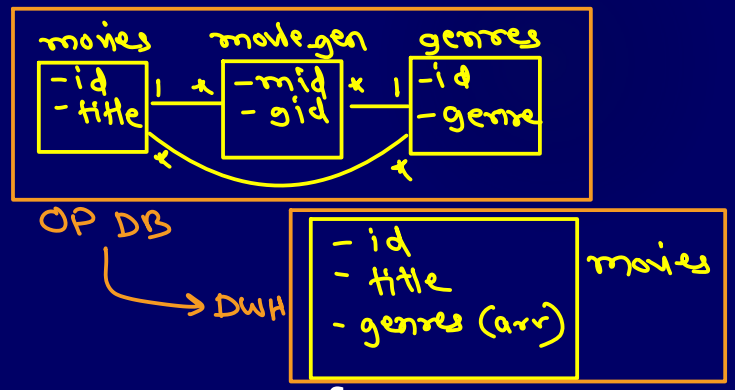
- Fact Constellation
  - Multiple fact tables that share many dimension tables
  - Booking and Checkout may share many dimension tables in the hotel industry



De-normalization

- Normalization in a data warehouse may lead to lots of small tables
- Can lead to excessive I/O's since many tables have to be accessed
- De-normalization is the answer especially since updates are rare

Creating Arrays



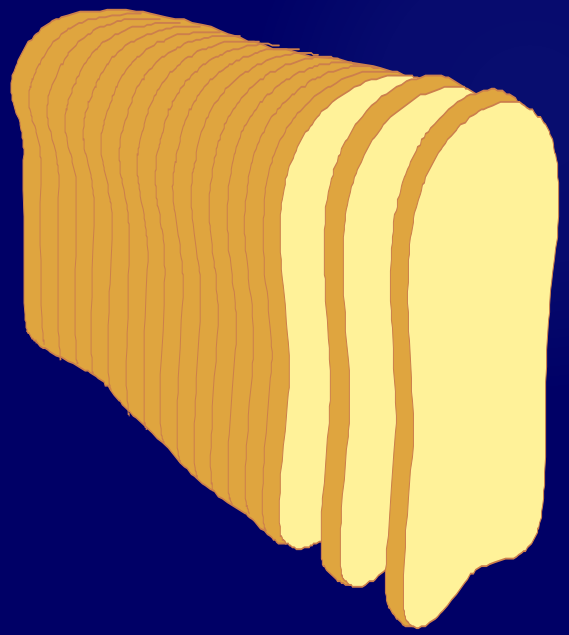
- Many times each occurrence of a sequence of data is in a different physical location
- Beneficial to collect all occurrences together and store as an array in a single row
- Makes sense only if there are a stable number of occurrences which are accessed together
- In a data warehouse, such situations arise naturally due to time based orientation
  - can create an array by month

Selective Redundancy

- Description of an item can be stored redundantly with order table -- most often item description is also accessed with order table
- Updates have to be careful

Partitioning

- Breaking data into several physical units that can be handled separately
- Not a question of *whether* to do it in data warehouses but *how* to do it
- Granularity and partitioning are key to effective implementation of a warehouse



Why Partition?

- Flexibility in managing data
- Smaller physical units allow
  - easy restructuring
  - free indexing
  - sequential scans if needed
  - easy reorganization
  - easy recovery
  - easy monitoring

Criterion for Partitioning

- Typically partitioned by
  - date
  - line of business
  - geography
  - organizational unit
  - any combination of above

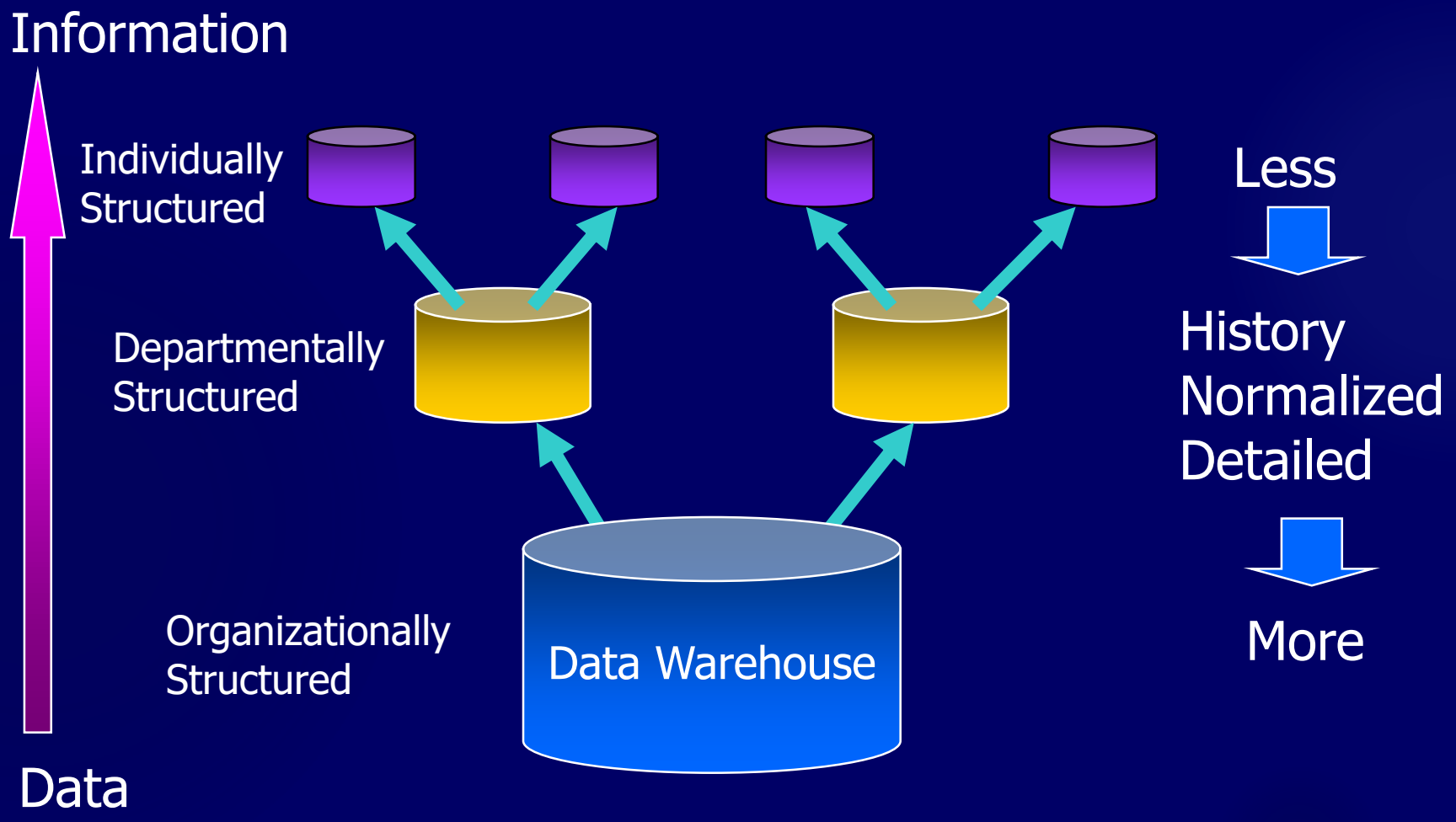
Where to Partition?

- Application level or DBMS level
- Makes sense to partition at application level
  - Allows different definition for each year
    - Important since warehouse spans many years and as business evolves definition changes
  - Allows data to be moved between processing complexes easily

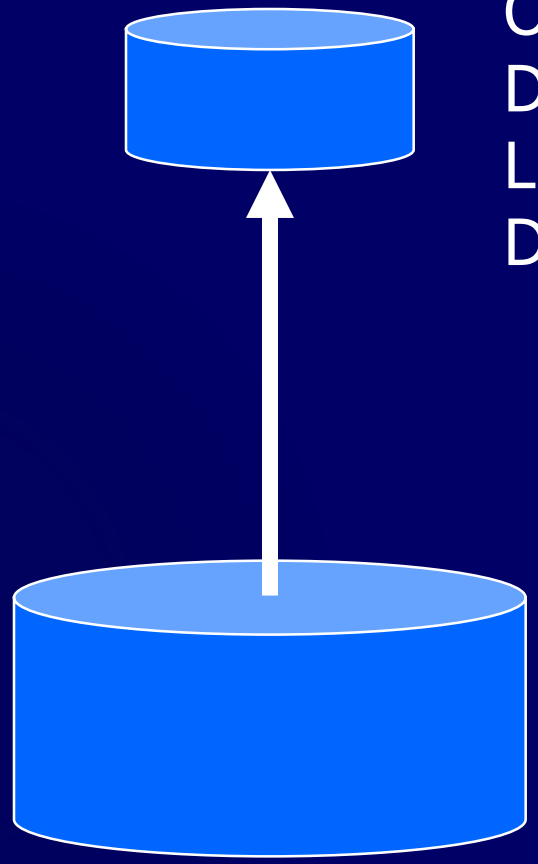
Data Warehouse vs. Data Marts

- What comes first

From the Data Warehouse to Data Marts



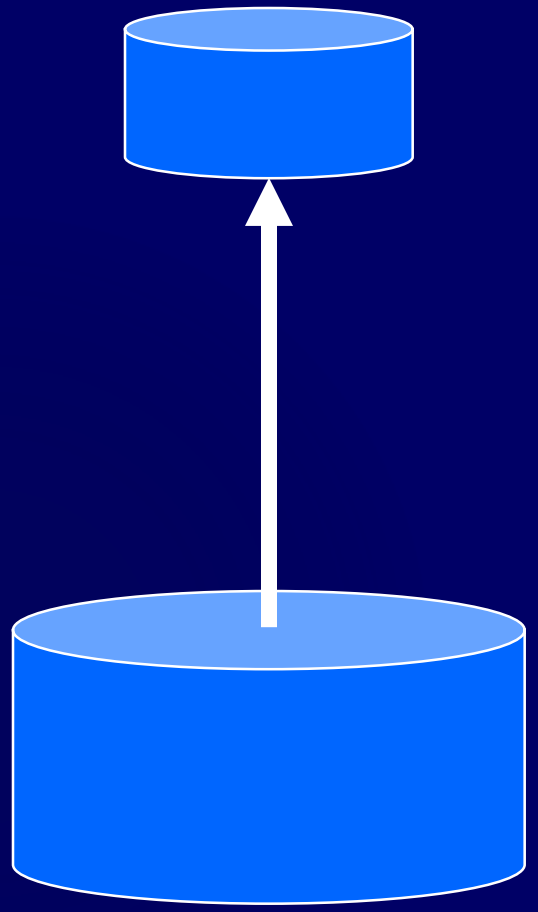
Data Warehouse and Data Marts



OLAP
Data Mart
Lightly summarized
Departmentally structured

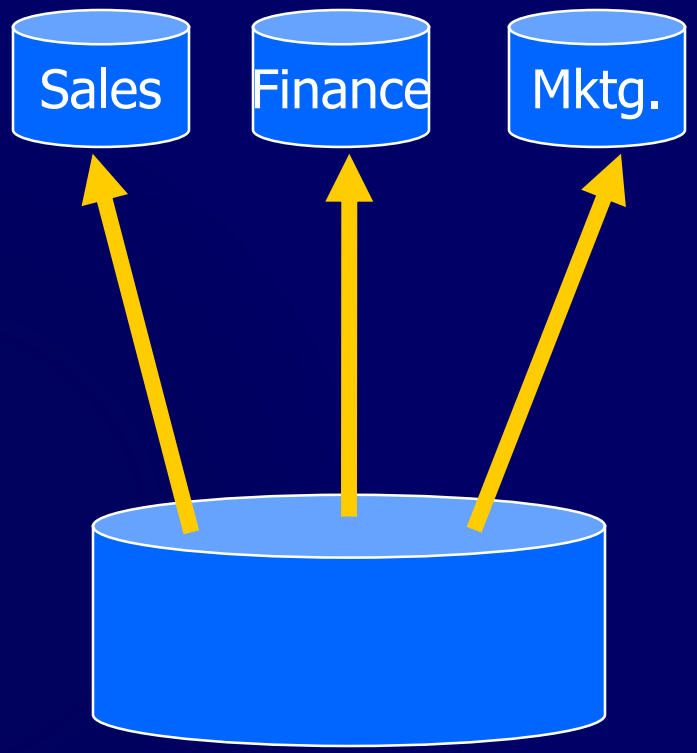
Organizationally structured
Atomic
Detailed Data Warehouse Data

Characteristics of the Departmental Data Mart



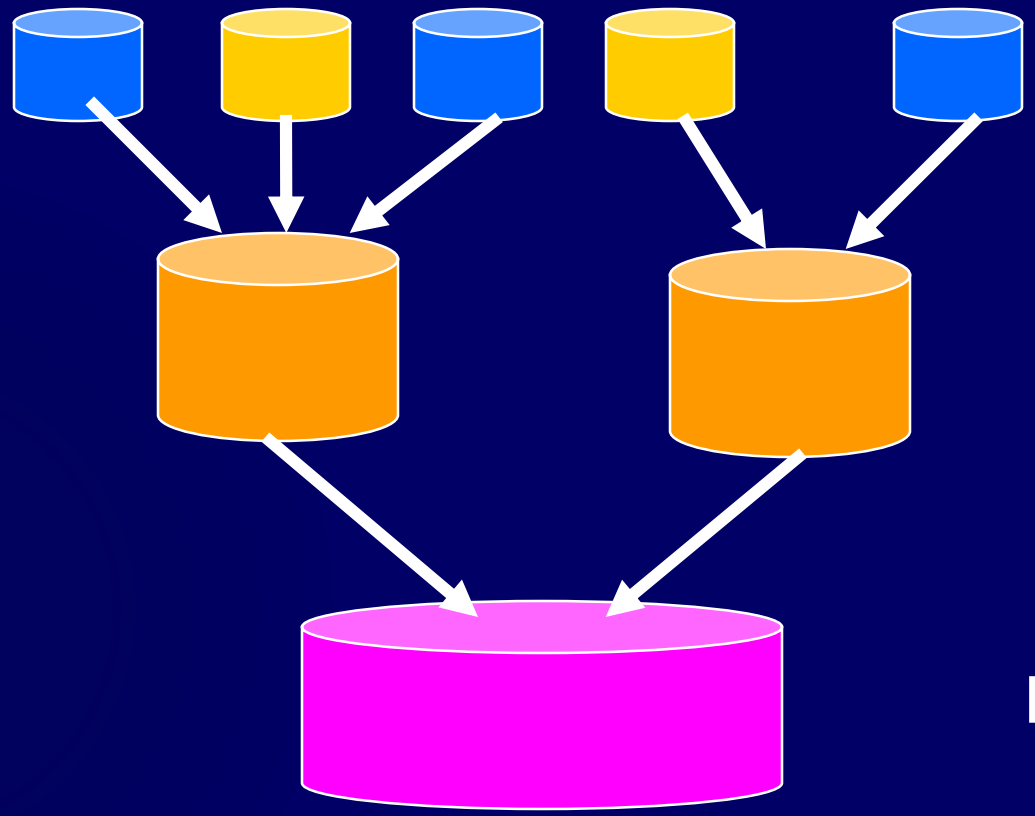
- OLAP
- Small
- Flexible
- Customized by Department
- Source is departmentally structured data warehouse

Techniques for Creating Departmental Data Mart

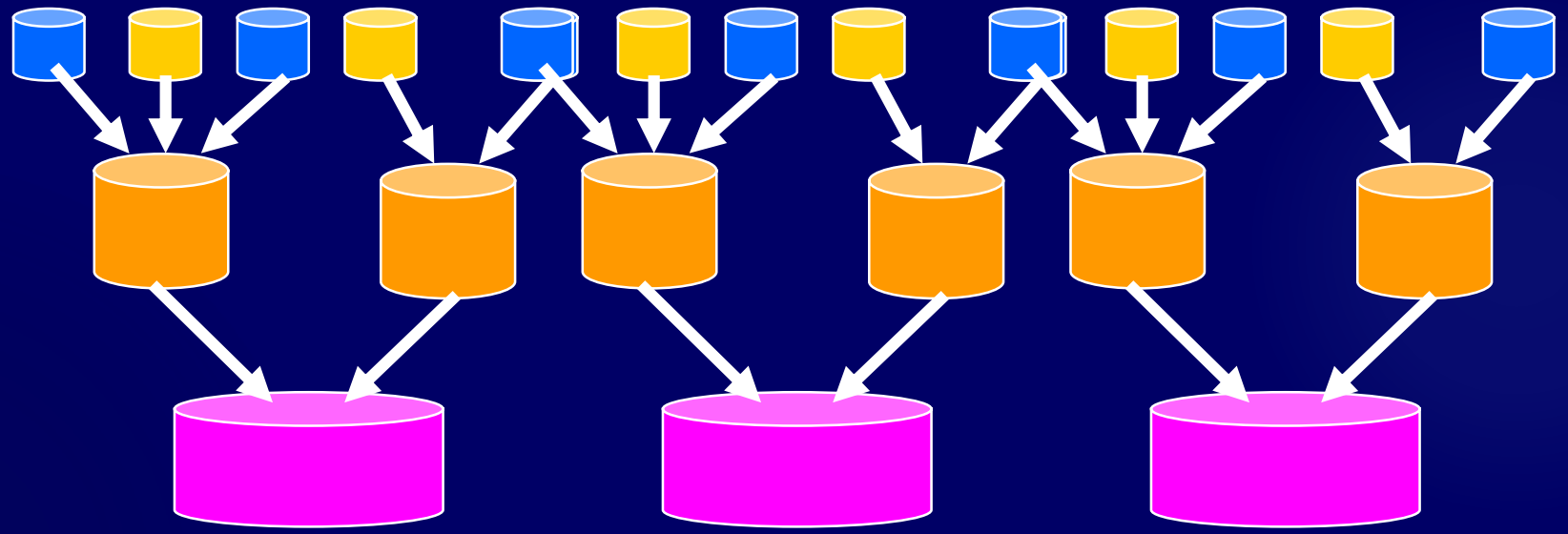


- OLAP
- Subset
- Summarized
- Superset
- Indexed
- Arrayed

Data Mart Centric



Problems with Data Mart Centric Solution



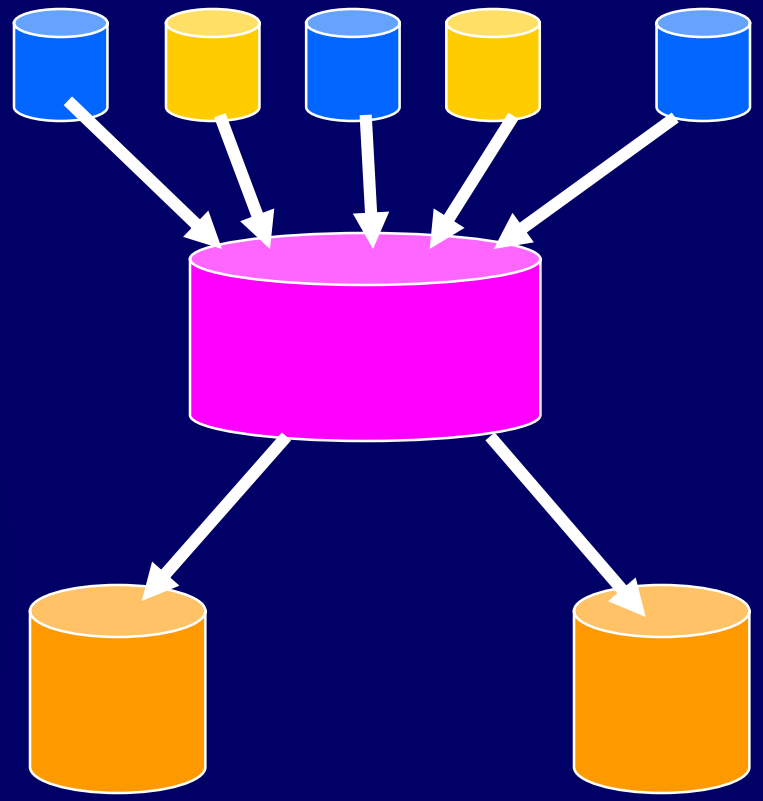
If you end up creating multiple warehouses, integrating them is a problem

True Warehouse

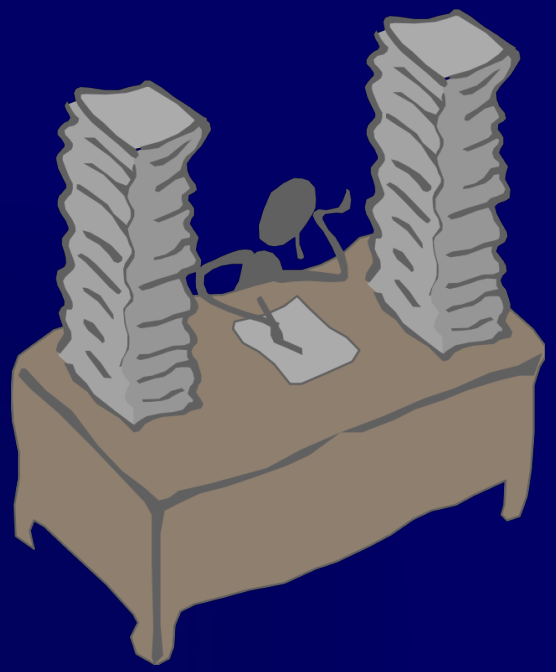
Data Sources

Data Warehouse

Data Marts



Query Processing



- Indexing
- Pre computed views/aggregates
- SQL extensions

Indexing Techniques

- Exploiting indexes to reduce scanning of data is of crucial importance
- Bitmap Indexes
- Join Indexes
- Other Issues
  - Text indexing
  - Parallelizing and sequencing of index builds and incremental updates

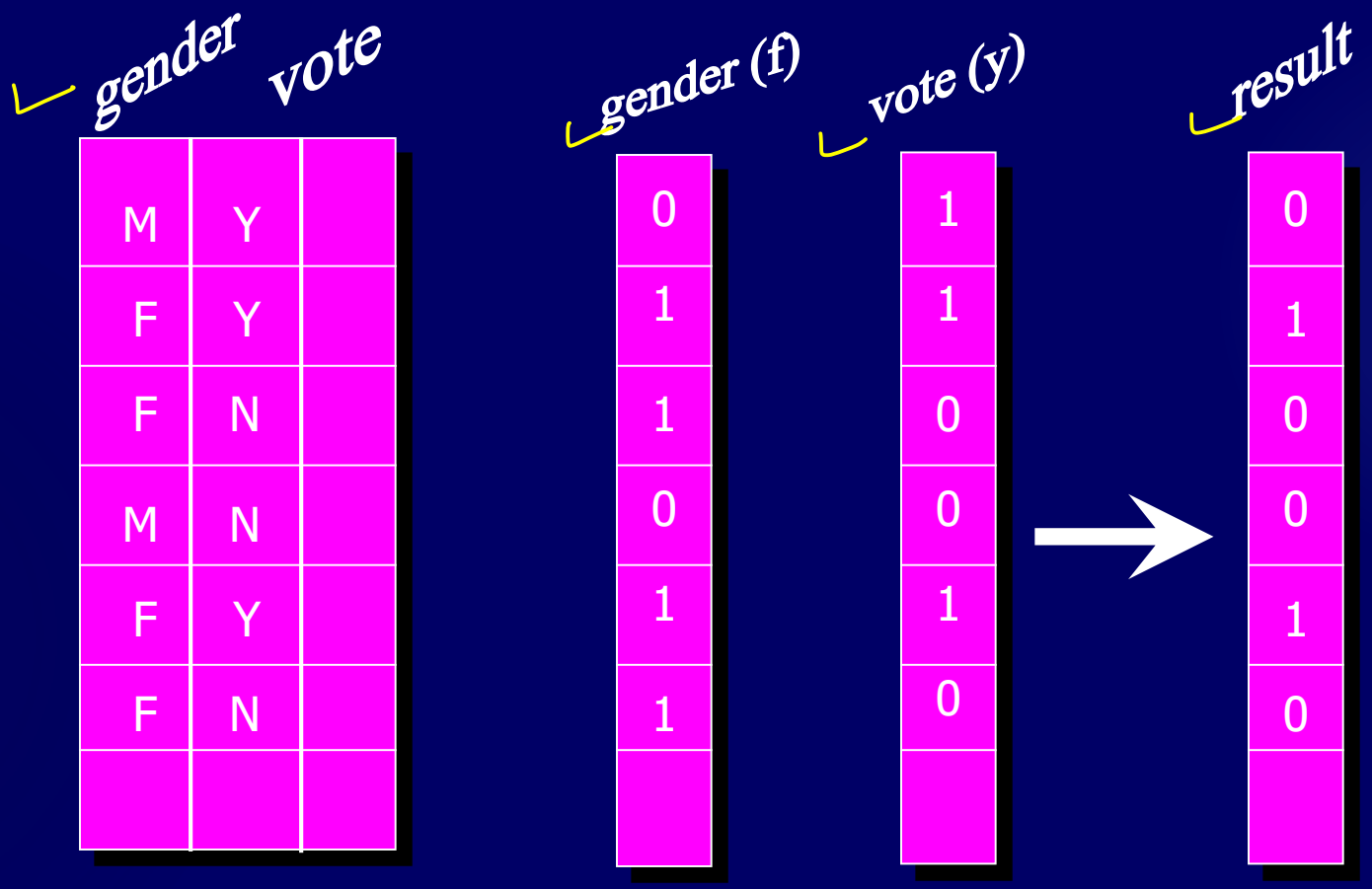
Indexing Techniques

- Bitmap index:
  - A collection of bitmaps -- one for each distinct value of the column
  - Each bitmap has N bits where N is the number of rows in the table
  - A bit corresponding to a value v for a row r is set if and only if r has the value for the indexed attribute

BitMap Indexes

- An alternative representation of RID-list
- Specially advantageous for low-cardinality domains
- Represent each row of a table by a bit and the table as a bit vector
- There is a distinct bit vector Bv for each value v for the domain
- Example: the attribute sex has values M and F. A table of 100 million people needs 2 lists of 100 million bits

Bitmap Index



Customer

Query : select * from customer where gender = 'F' and vote = 'Y'

Bit Map Index

Base Table

Cust	Region	Rating
C1	N	H
C2	S	M
C3	W	L
C4	W	H
C5	S	L
C6	W	L
C7	N	H

Region Index

Row ID	N	S	E	W
1	1	0	0	0
2	0	1	0	0
3	0	0	0	1
4	0	0	0	1
5	0	1	0	0
6	0	0	0	1
7	1	0	0	0

Rating Index

Row ID	H	M	L
1	1	0	0
2	0	1	0
3	0	0	0
4	0	0	0
5	0	1	0
6	0	0	0
7	1	0	0

Customers where **Region = W** ***And*** **Rating = M**

BitMap Indexes

- Comparison, join and aggregation operations are reduced to bit arithmetic with dramatic improvement in processing time
- Significant reduction in space and I/O (30:1)
- Adapted for higher cardinality domains as well.
- Compression (e.g., run-length encoding) exploited
- Products that support bitmaps: Model 204, TargetIndex (Redbrick), IQ (Sybase), Oracle 7.3

Join Indexes

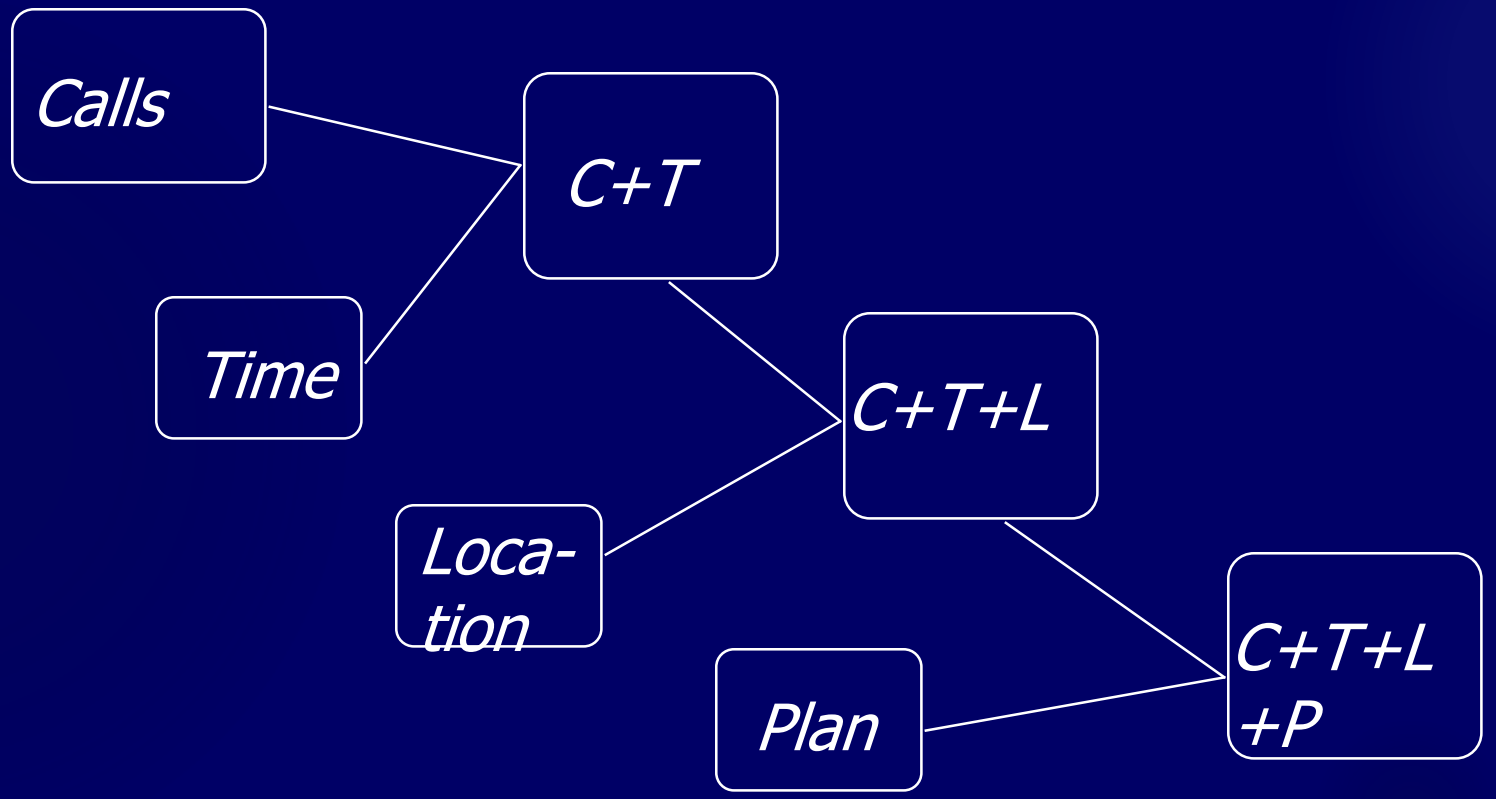
- Pre-computed joins
- A join index between a fact table and a dimension table correlates a dimension tuple with the fact tuples that have the same value on the common dimensional attribute
  - e.g., a join index on *city* dimension of *calls* fact table
  - correlates for each city the calls (in the *calls* table) from that city

Join Indexes

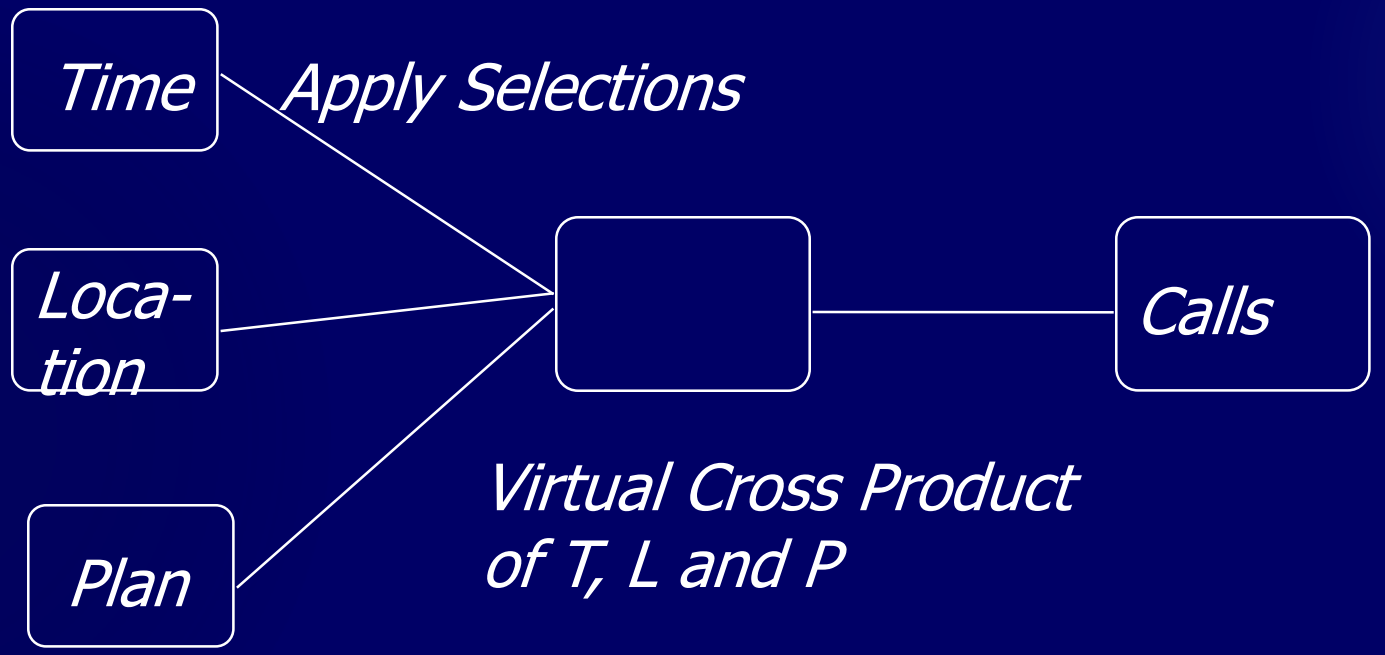
- Join indexes can also span multiple dimension tables
  - e.g., a join index on *city* and *time* dimension of *calls* fact table

Star Join Processing

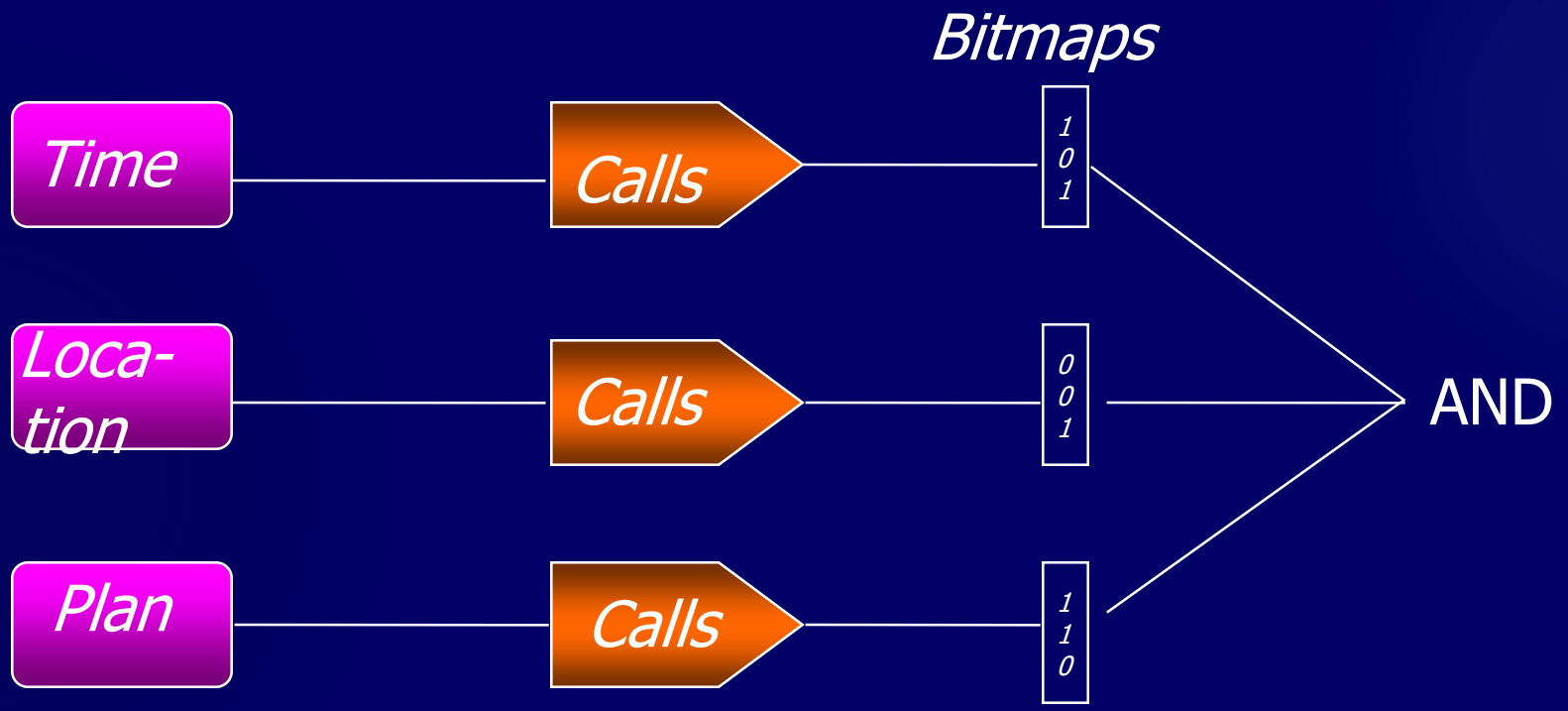
- Use join indexes to join dimension and fact table



Optimized Star Join Processing



Bitmapped Join Processing



Pre-computed Aggregates

Materialized View

- Keep aggregated data for efficiency (pre-computed queries)
- Questions
  - Which aggregates to compute?
  - How to update aggregates?
  - How to use pre-computed aggregates in queries?

Pre-computed Aggregates

- Aggregated table can be maintained by the
  - warehouse server
  - middle tier
  - client applications
- Pre-computed aggregates -- special case of materialized views -- same questions and issues remain

SQL Extensions

- Extended family of aggregate functions
  - rank (top 10 customers)
  - percentile (top 30% of customers)
  - median, mode
  - Object Relational Systems allow addition of new aggregate functions

SQL Extensions

- Reporting features
  - running total, cumulative totals
- Cube operator Roll up
  - group by on all subsets of a set of attributes (month,city)
  - redundant scan and sorting of data can be avoided

Course Overview

- The course: what and how

- 0. Introduction
- I. Data Warehousing
- II. Decision Support and OLAP
- III. Data Mining
- IV. Looking Ahead

- Demos and Labs

Limitations of SQL

- "A Freshman in Business needs a Ph.D. in SQL"

-- Ralph Kimball

What Is OLAP?

- Online Analytical Processing - coined by EF Codd in 1994 paper contracted by Arbor Software*
- Generally synonymous with earlier terms such as Decisions Support, Business Intelligence, Executive Information System
- OLAP = Multidimensional Database
- MOLAP: Multidimensional OLAP (Arbor Essbase, Oracle Express)
- ROLAP: Relational OLAP (Informix MetaCube, Microstrategy DSS Agent)

* Reference: http://www.arborsoft.com/essbase/wht_ppr/coddTOC.html

The OLAP Market

- Rapid growth in the enterprise market
  - 1995: $700 Million
  - 1997: $2.1 Billion
- Significant consolidation activity among major DBMS vendors
  - 10/94: Sybase acquires ExpressWay
  - 7/95: Oracle acquires Express
  - 11/95: Informix acquires Metacube
  - 1/97: Arbor partners up with IBM
  - 10/96: Microsoft acquires Panorama
- Result: OLAP shifted from small vertical niche to mainstream DBMS category

Strengths of OLAP

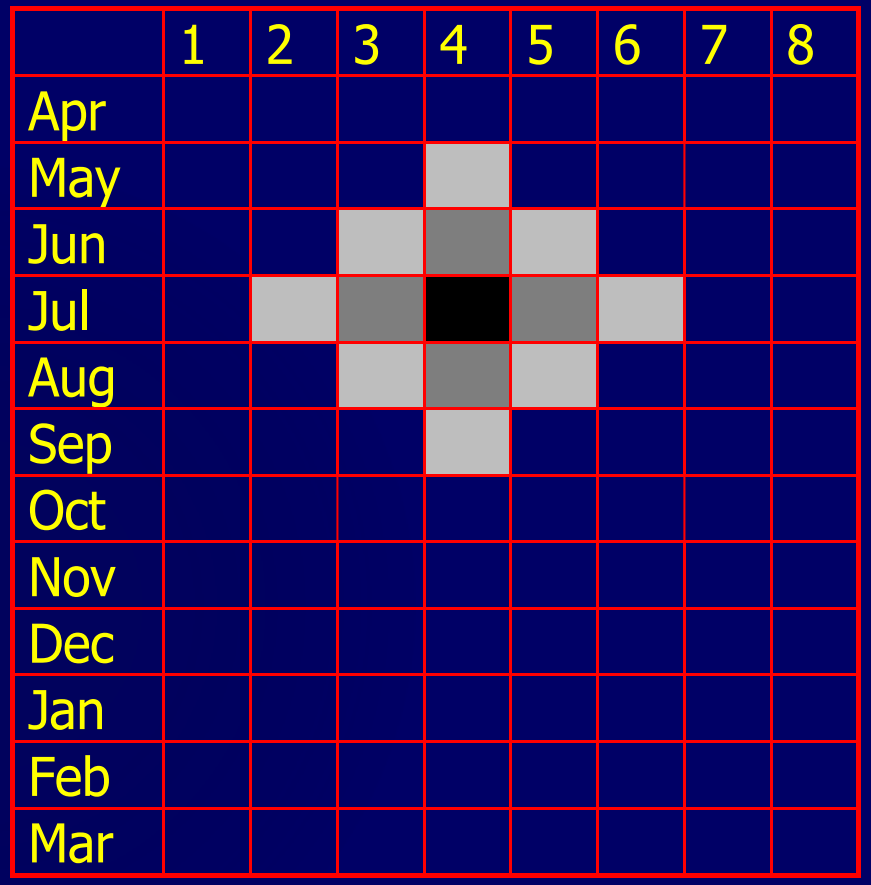
- It is a powerful visualization paradigm
- It provides fast, interactive response times
- It is good for analyzing time series
- It can be useful to find some clusters and outliers
- Many vendors offer OLAP tools

OLAP Is FASMI

- Fast
- Analysis
- Shared
- Multidimensional
- Information

Nigel Pendse, Richard Creath - The OLAP Report

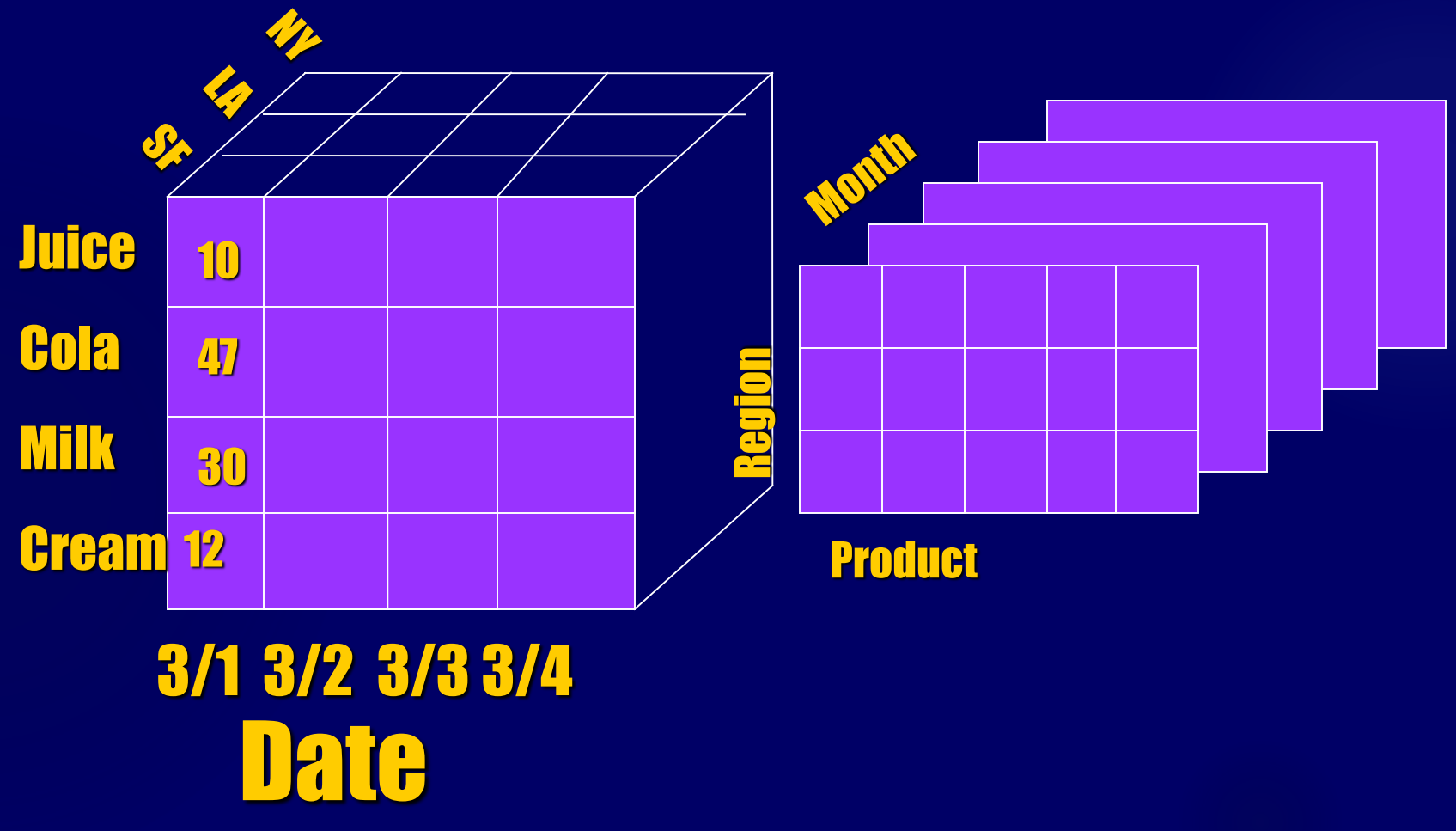
Visualizing Neighbors is simpler



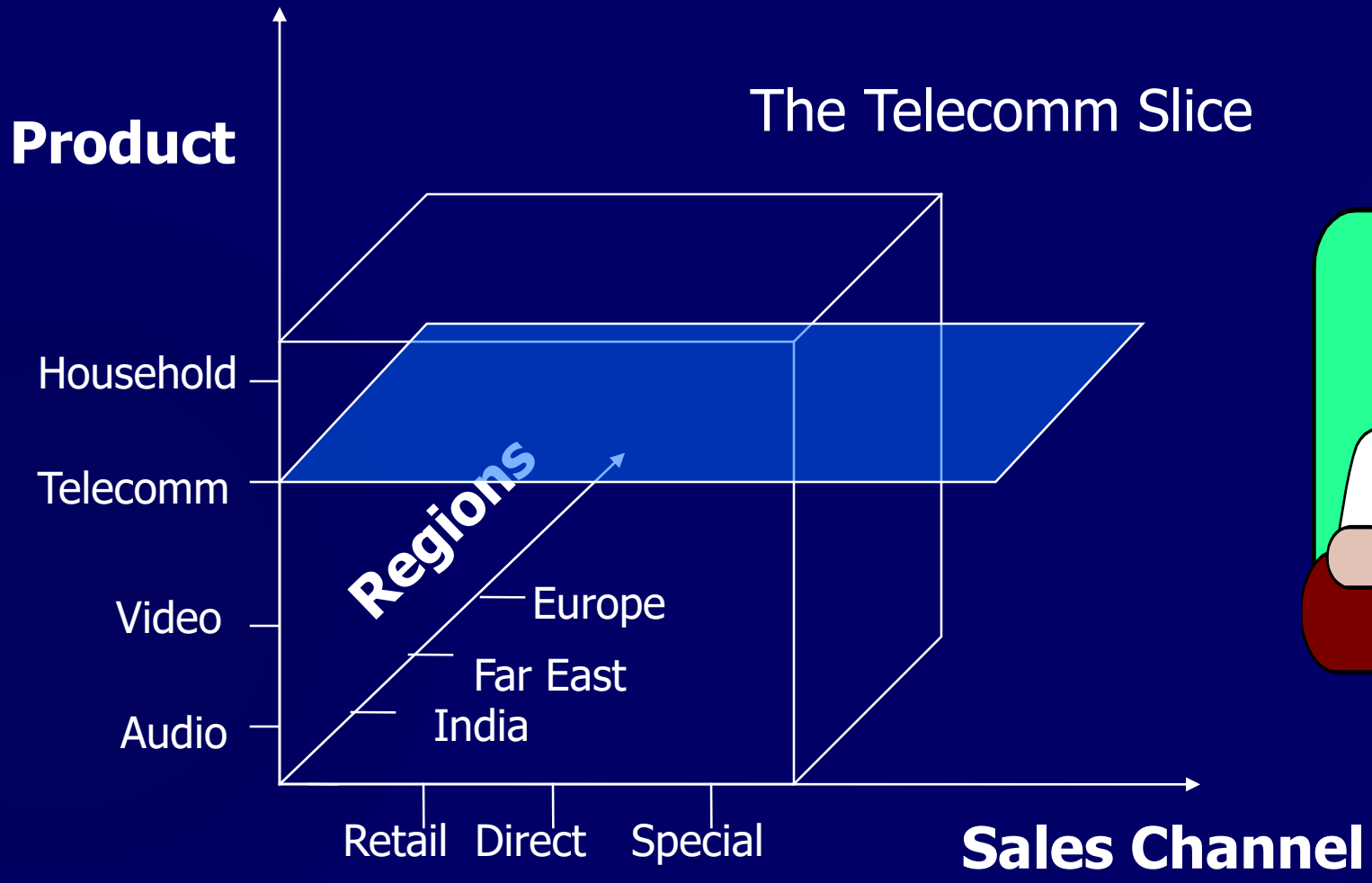
	1	2	3	4	5	6	7	8
Apr								
May								
Jun								
Jul								
Aug								
Sep								
Oct								
Nov								
Dec								
Jan								
Feb								
Mar								

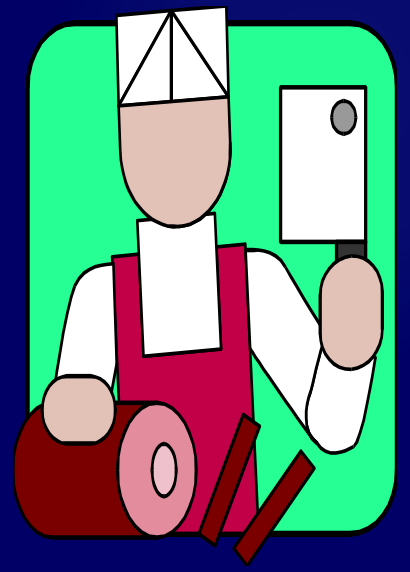
Month	Store	Sales
Apr	1	
Apr	2	
Apr	3	
Apr	4	
Apr	5	
Apr	6	
Apr	7	
Apr	8	
May	1	
May	2	
May	3	
May	4	
May	5	
May	6	
May	7	
May	8	
Jun	1	
Jun	2	

A Visual Operation: Pivot (Rotate)

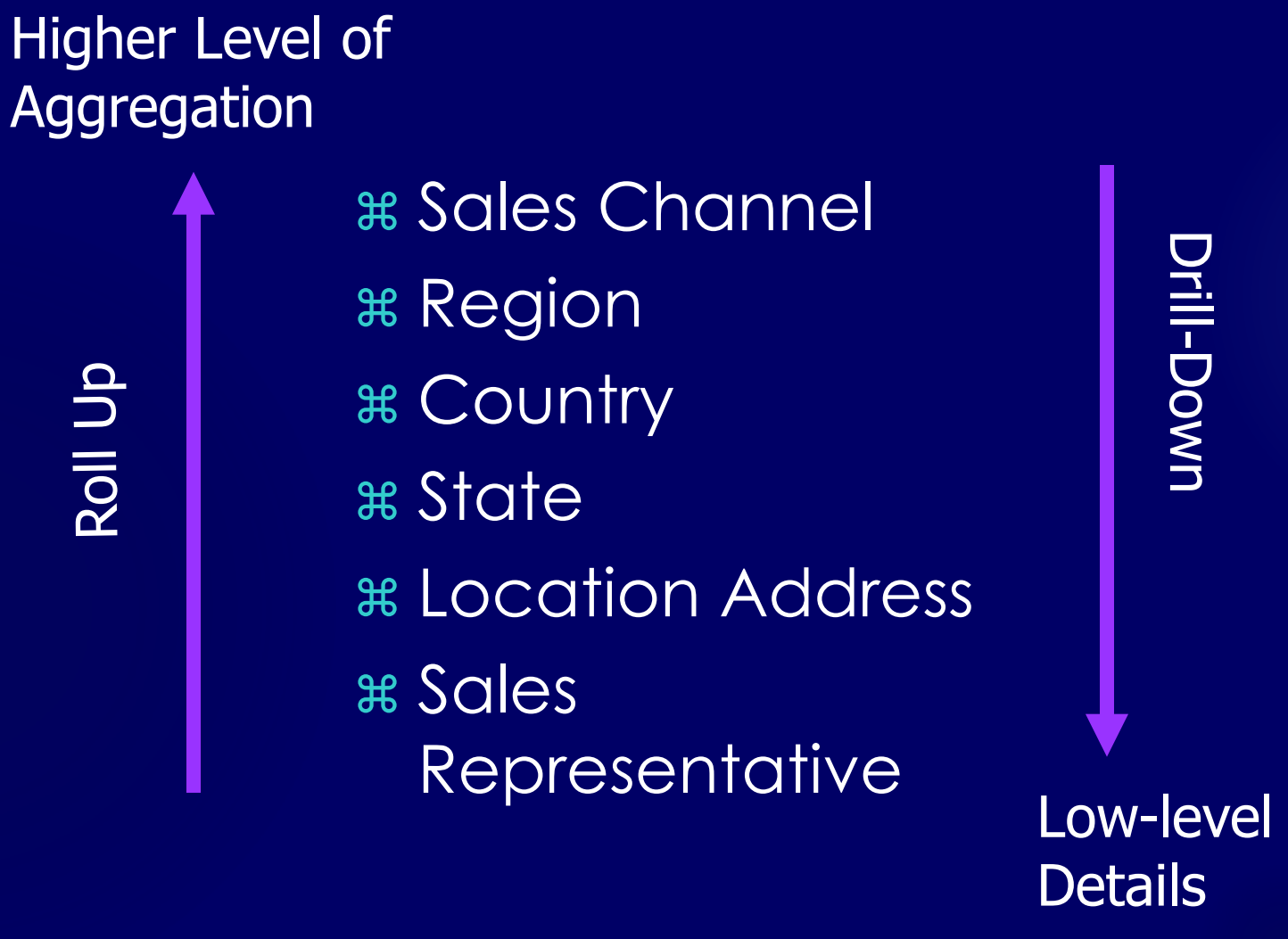


"Slicing and Dicing"





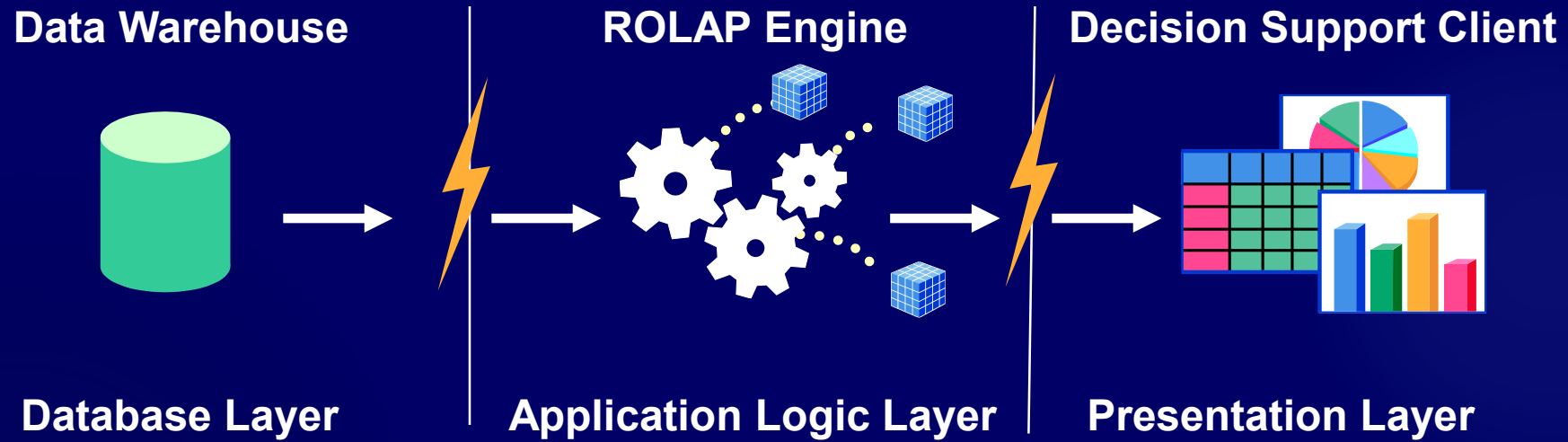
Roll-up and Drill Down



Nature of OLAP Analysis

- Aggregation -- (total sales, percent-to-total)
- Comparison -- Budget vs. Expenses
- Ranking -- Top 10, quartile analysis
- Access to detailed and aggregate data
- Complex criteria specification
- Visualization

Relational OLAP: 3 Tier DSS

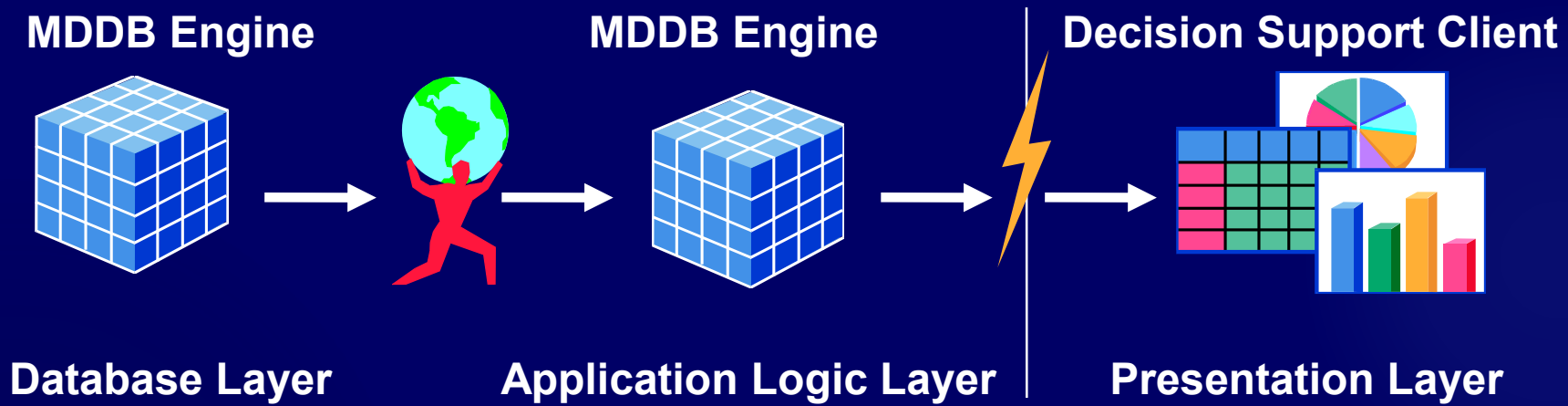


Store atomic data in industry standard RDBMS.

Generate SQL execution plans in the ROLAP engine to obtain OLAP functionality.

Obtain multi-dimensional reports from the DSS Client.

MD-OLAP: 2 Tier DSS

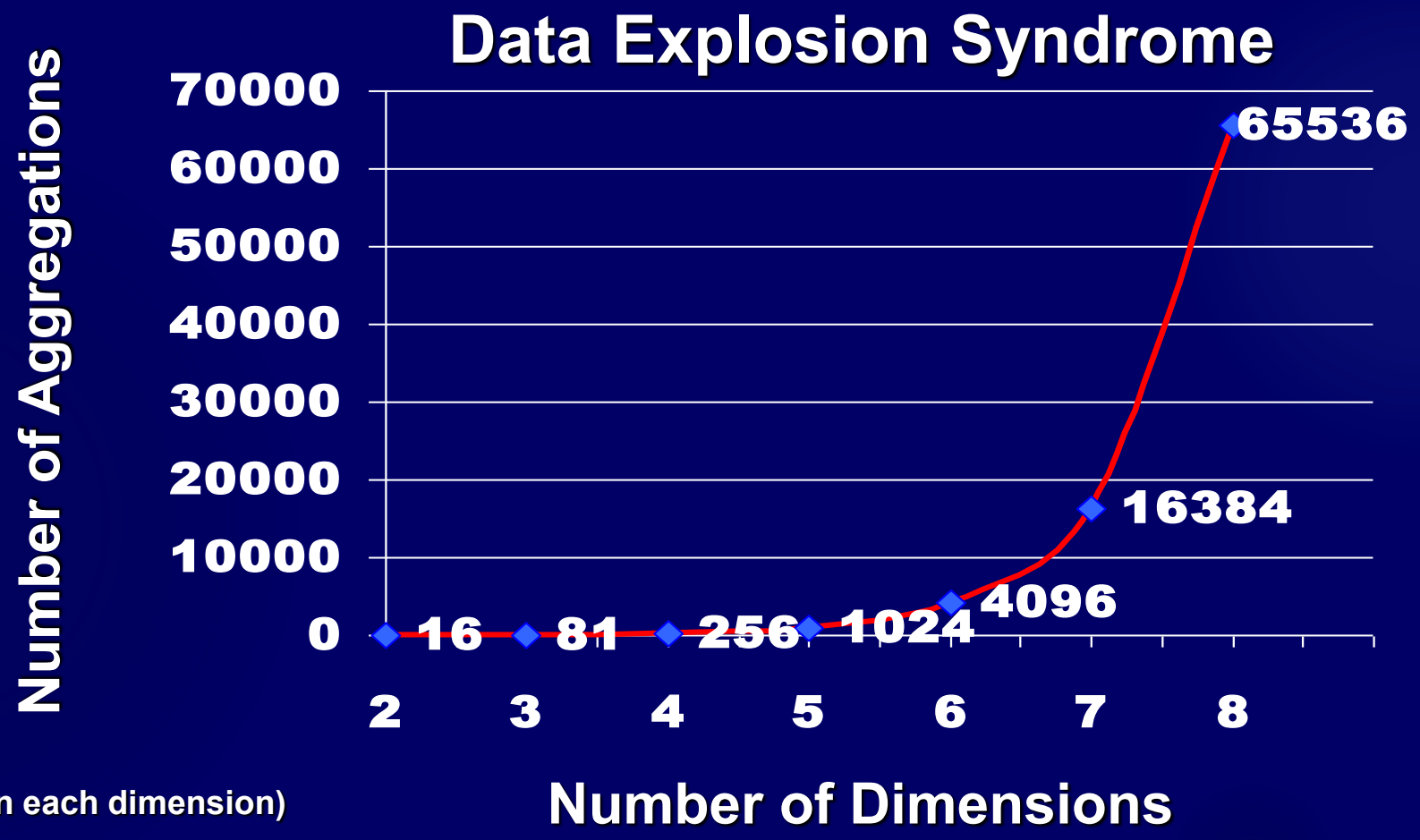


Store atomic data in a proprietary data structure (MDDB), pre-calculate as many outcomes as possible, obtain OLAP functionality via proprietary algorithms running against this data.

Obtain multi-dimensional reports from the DSS Client.

Typical OLAP Problems

Data Explosion



(4 levels in each dimension)

Recipe for a Successful Warehouse

For a Successful Warehouse

From Larry Greenfield, http://pwp.starnetinc.com/larryg/index.html

- From day one establish that warehousing is a joint user/builder project
- Establish that maintaining data quality will be an *ONGOING* joint user/builder responsibility
- Train the users one step at a time
- Consider doing a high level corporate data model in no more than three weeks

For a Successful Warehouse

- Look closely at the data extracting, cleaning, and loading tools
- Implement a user accessible automated directory to information stored in the warehouse
- Determine a plan to test the integrity of the data in the warehouse
- From the start get warehouse users in the habit of 'testing' complex queries

For a Successful Warehouse

- Coordinate system roll-out with network administration personnel
- When in a bind, ask others who have done the same thing for advice
- Be on the lookout for small, but strategic, projects
- Market and sell your data warehousing systems

Data Warehouse Pitfalls

- You are going to spend much time extracting, cleaning, and loading data
- Despite best efforts at project management, data warehousing project scope will increase
- You are going to find problems with systems feeding the data warehouse
- You will find the need to store data not being captured by any existing system
- You will need to validate data not being validated by transaction processing systems

Data Warehouse Pitfalls

- Some transaction processing systems feeding the warehousing system will not contain detail
- Many warehouse end users will be trained and never or seldom apply their training
- After end users receive query and report tools, requests for IS written reports may increase
- Your warehouse users will develop conflicting business rules
- Large scale data warehousing can become an exercise in data homogenizing

Data Warehouse Pitfalls

- 'Overhead' can eat up great amounts of disk space
- The time it takes to load the warehouse will expand to the amount of the time in the available window... and then some
- Assigning security cannot be done with a transaction processing system mindset
- You are building a HIGH maintenance system
- You will fail if you concentrate on resource optimization to the neglect of project, data, and customer management issues and an understanding of what adds value to the customer

Useful URLs

- Ralph Kimball's home page
  - http://www.rkimball.com
- Larry Greenfield's Data Warehouse Information Center
  - http://pwp.starnetinc.com/larryg/
- Data Warehousing Institute
  - http://www.dw-institute.com/
- OLAP Council
  - http://www.olapcouncil.com/