4.1 Data Warehousing Components

What is Data Warehouse?

- Defined in many different ways but mainly it is:
  - A decision support database that is maintained separately from the organization's operational database.
  - Supports information processing by providing a solid platform of consolidated, historical data for analysis.
- In the broad sense, "A Data Warehouse is a subject-oriented, integrated, time-variant, and Non-Volatile collection of data in support of management's decision-making process".
- Hence, Data Warehousing is a process of constructing and using data warehouses.
- Data Warehouse is Subject-Oriented:
  - Data organized around major subjects, such as customer, product and sales.
  - Focused on modeling and analysis of data for decision-makers, not on daily-operations or transaction-processing.
  - Provides a simple and concise view around a particular subject by excluding data that are not useful in the decision support process.
- Data Warehouse is Integrated:
  - Constructed by integrating multiple, heterogeneous data sources.
    - Relational databases, flat files, on-line transaction records…
  - Data cleaning and Data integration techniques are applied.
    - Ensures consistency in naming conventions, encoding structures, attribute measures etc. from different data sources.
      - Eg. Hotel_price: currency, tax, breakfast_covered etc.
    - When data is moved to the warehouse it is converted.
- Data Warehouse is Time-Variant:
  - The time-horizon for the data warehouse is significantly longer than that of operational systems.
    - Operational Database: Current value data
    - Data Warehouse: Stores data with historical perspective (Eg. Past 5 to 10 years).
  - Every key structure in the data warehouse contains an element of time explicitly or implicitly.
- Data Warehouse is Non-Volatile:
  - A physically separate store of data transformed from the operational environment.
  - Operational update of data does not occur in the data warehouse environment.
    - Does not require Transaction Processing, Recovery, Concurrency control mechanisms (functions of DBMS).
    - Requires only two operations in data accessing:
      - Initial Loading of Data / Periodic Refresh of Data
      - Access of Data
- Data Warehouse Vs Heterogeneous DBMS:
  - Traditional Heterogeneous DB Integration:

        - Query-driven approach
        - Build wrappers / mediators on top of heterogeneous databases
        - When a query is passed from a client site, a meta-dictionary is used to translate the query into queries appropriate for individual heterogeneous sites involved and the results are integrated into a global answer set.
        - Complex information filtering, compete for resources
    - Data Warehouse:
        - Update-driven approach
        - Has high performance
        - Information from heterogeneous sources are integrated in advance and stored in warehouses for direct query and analysis.

- Data Warehouse Vs Operational DBMS:
    - OLTP – Online Transactional Processing:
        - This includes the major tasks of traditional relational DBMS (Concurrency control, transaction procession, recovery etc…)
        - Does Day-to-Day operations such as purchasing, inventory, banking, manufacturing, payroll, registration, accounting…
    - OLAP – Online Analytical Processing:
        - This is the major task of Data Warehousing System.
        - Useful for complex data analysis and decision making.

    - Distinct Features – OLTP Vs OLAP:
        - User and System Orientation:
            - OLTP – Customer-Oriented – Used by Clerks, Clients and IT Professionals.
            - OLAP – Market-Oriented – Used by Managers, Executives and Data Analysts
        - Data Contents:
            - OLTP – has current data - data is too detailed so that it can not be used for decision making.
            - OLAP – has historical data – data summarized and aggregated at different levels of granularity – data easy for decision making
        - Database Design:
            - OLTP – E-R Data Model - Application Oriented DB design.
            - OLAP – Star or Snowflake Model – Subject oriented DB design.
        - View:
            - OLTP – Data pertaining to a department or an enterprise.
            - OLAP – Data pertaining to many departments of an organization or data of many organizations stored in a single data warehouse.
        - Access Patterns:
            - OLTP – Short atomic transactions.

- OLAP – Read only transactions (mostly complex queries).

▪ Other distinguishing Features:

- Database Size, Frequency of Operation and Performance Metrics.

○ Comparison between OLTP and OLAP systems:

Feature	OLTP	OLAP
Characteristic	operational processing	informational processing
Orientation	transaction	analysis
User	clerk, DBA, database professional	knowledge worker (e.g., manager, executive, analyst)
Function	day-to-day operations	long term informational requirements, decision support
DB design	E-R based, application-oriented	star/snowflake, subject-oriented
Data	current; guaranteed up-to-date	historical; accuracy maintained over time
Summarization	primitive, highly detailed	summarized, consolidated
View	detailed, flat relational	summarized, multidimensional
Unit of work	short, simple transaction	complex query
Access	read/write	mostly read
Focus	data in	information out
Operations	index/hash on primary key	lots of scans
# of records accessed	tens	millions
# of users	thousands	hundreds
DB size	100 MB to GB	100 GB to TB
Priority	high performance, high availability	high flexibility, end-user autonomy
Metric	transaction throughput	query throughput, response time

- Why Separate Data Warehouse?:

○ High performance for both systems:

▪ DBMS – Tuned for OLTP:

- access methods, indexing, concurrency control, recovery

▪ Warehouse – Tuned for OLAP:

- Complex OLAP Queries
- Multi dimensional view
- Consolidation

○ Different Functions and Different Data:

▪ Missing Data:

- Decision support requires historical data
- Historical data are not maintained by Operational DBs

▪ Data Consolidation:

- DS requires consolidation (aggregation and summarization) of data from heterogeneous sources.

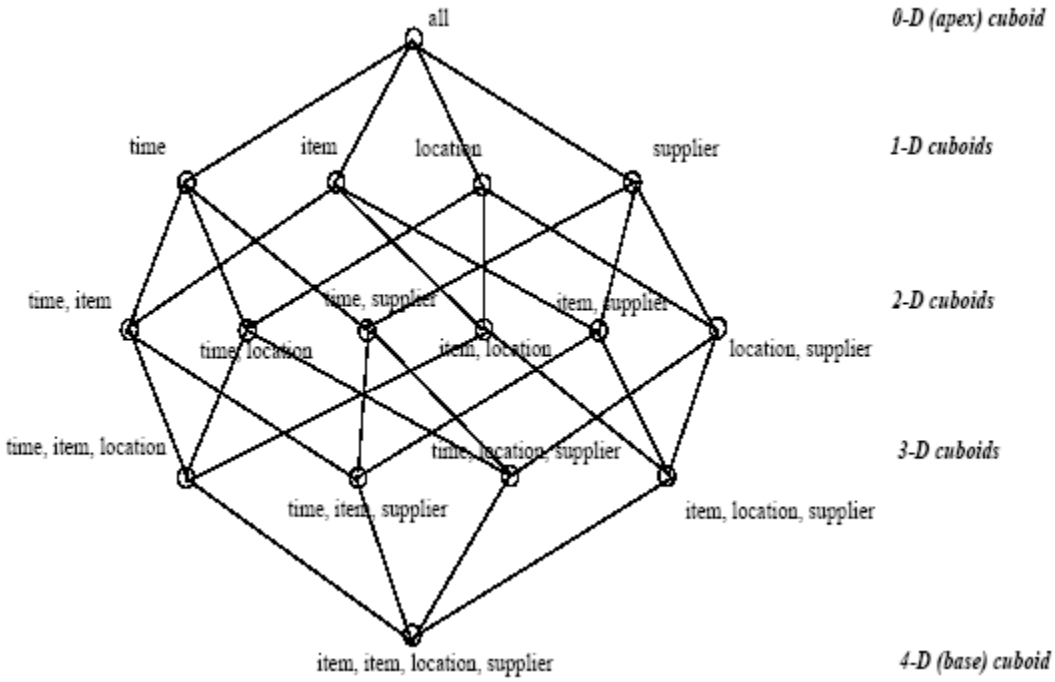
▪ Data Quality:

- Data from different sources are inconsistent
- Different codes and formats has to be reconciled

4.2 Multi Dimensional Data Model

A Multi-Dimensional Data Model

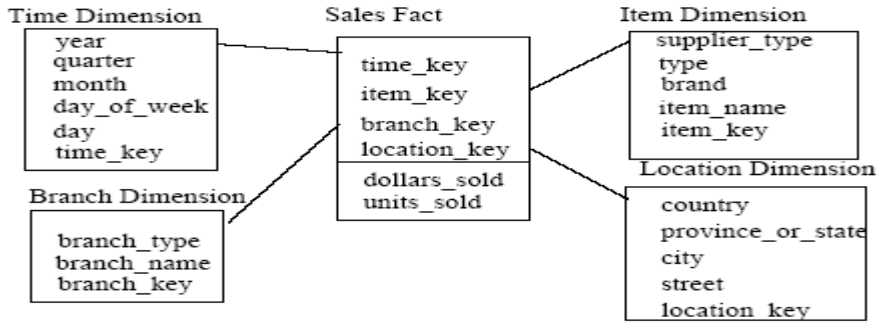
- From Tables and Spreadsheets to Data Cubes:
  - A data warehouse is based on a multi-dimensional data model which views data in the form of a data cube.
  - A data cube such as *Sales* allows data to be modeled and viewed in multiple dimensions.
    - Dimension tables such as *item(item_name, brand, type)* or *time(day, week, month, quarter, year)*
    - Fact tables consist of measures (such as *dollars_sold*) and keys to each of the related dimension tables.
  - In data warehousing we have:
    - Base Cuboid: Any n-D cube is called a base cuboid
    - Apex Cuboid: Top most 0-D cuboid that has the highest-level of summarization is called the apex cuboid.
    - Data Cube: Lattice of cuboids forms a data cube.



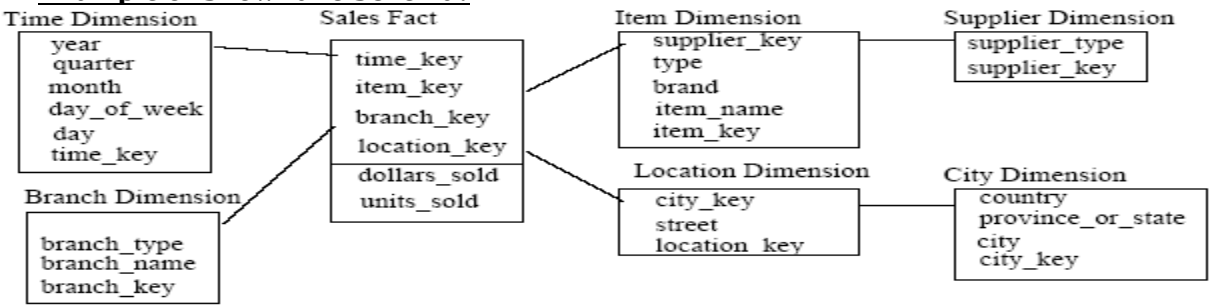
- Conceptual Modeling of Data Warehouses:
  - Data warehouse is modeled as dimensions and facts / measures.
  - There are three types of Modeling for Data Warehouses:
    - Star Schema:
      - A fact table in the middle connected to a set of dimension tables.
    - Snowflake Schema:
      - A refinement of star schema where some dimensional table is normalized into a set of smaller dimension tables, forming a shape similar to snowflake.
    - Fact Constellation:

- Multiple fact tables shares dimension tables. Viewed as a collection of star schema. Hence called Galaxy schema or fact constellation.

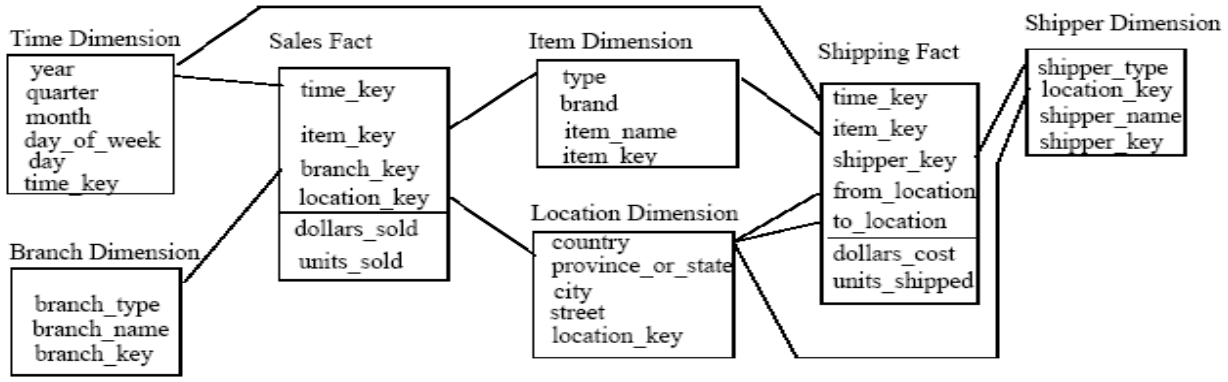
- Cube Definition Syntax in DMQL (Data Mining Query Language):
  - Cube Definition (Includes Fact table definition also):
    - Define cube <cube_name> [<dimension_list>]: <measure_list>
  - Dimension Definition (Dimension Table):
    - Define dimension <dimension_name> as (<attribute_list>)
  - Special Case – Shared Dimension Tables:
    - Define dimension <dimension_name> as <dimension_name_first_time> in cube <cube_name_first_time>
- Defining Star Schema in DMQL:
  - Define cube sales_star [time, item, branch, location]: dollars_sold = sum(sales_in_dollars), avg_sales = avg(sales_in_dollars), units_sold = count(*)
  - Define dimension time as (time_key, day, day_of_week, month, quarter, year)
  - Define dimension item as (item_key, item_name, brand, type, supplier_type)
  - Define dimension branch as (branch_key, branch_name, branch_type)
  - Define dimension location as (locaton_key, street, city, province_or_state, country)
- **Example of Star Schema:**



- **Example of Snowflake Schema:**

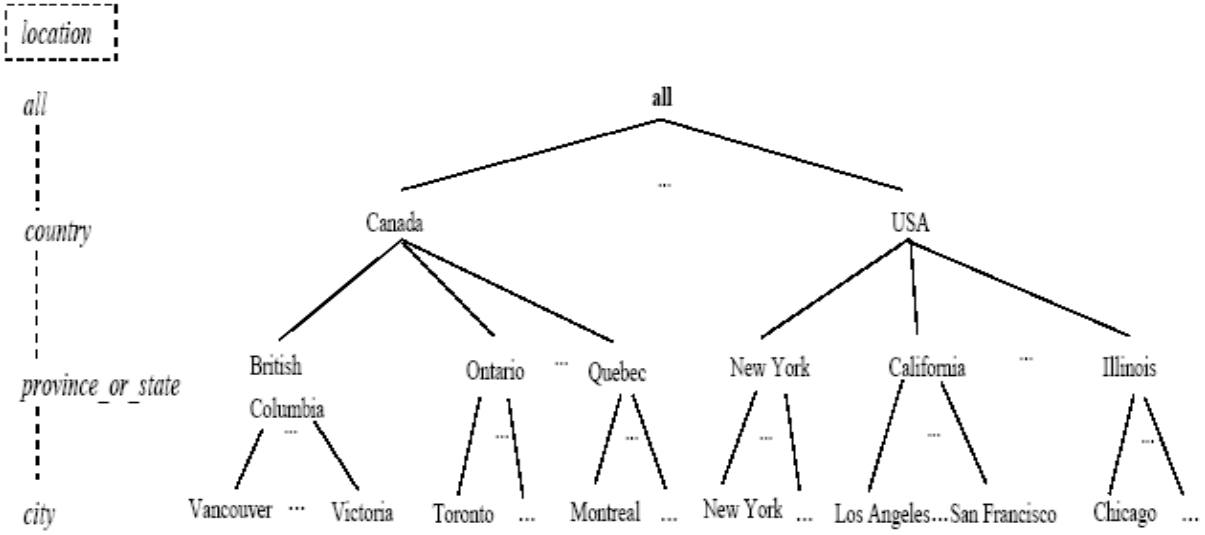


- **Example of Fact Constellation:**



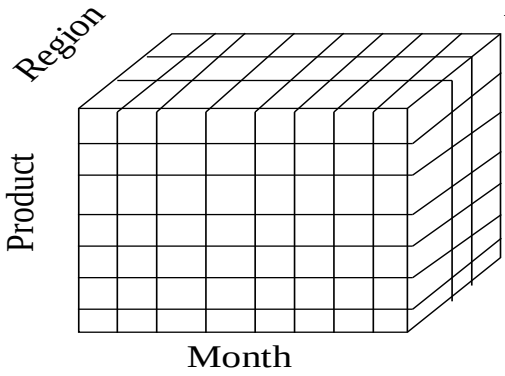
- Defining Snowflake Schema in DMQL:
  - Define cube sales_snowflake [time, item, branch, location]: dollars_sold = sum(sales_in_dollars), avg_sales = avg(sales_in_dollars), units_sold = count(*)
  - Define dimension time as (time_key, day, day_of_week, month, quarter, year)
  - Define dimension item as (item_key, item_name, brand, type, supplier(supplier_key, supplier_type))
  - Define dimension branch as (branch_key, branch_name, branch_type)
  - Define dimension location as (locaton_key, street, city(ciy_key, province_or_state, country))
- Defining Fact Constellation in DMQL:
  - Define cube sales [time, item, branch, location]: dollars_sold = sum(sales_in_dollars), avg_sales = avg(sales_in_dollars), units_sold = count(*)
  - Define dimension time as (time_key, day, day_of_week, month, quarter, year)
  - Define dimension item as (item_key, item_name, brand, type, supplier)
  - Define dimension branch as (branch_key, branch_name, branch_type)
  - Define dimension location as (locaton_key, street, city, province_or_state, country)
  - Define cube shipping [time, item, shipper, from_locaton, to_location]: dollar_cost = sum(cost_in_dollars), unit_shipperd = count(*)
  - Define dimension time as time in cube sales
  - Define dimension item as item in cube sales
  - Define dimension shipper as (shipper_key, shipper_name, location as location in cube sales, shipper_type)
  - Define dimension from_location as location in cube sales
  - Define dimension to_location as location in cube sales
- Measures of Data Cube: Three Categories: (based on the kind of aggregate functions used)
  - Distributive:
    - If the result derived by applying the function to n aggregate values is same as that derived by applying the function on all the data without partitioning.
    - Eg. Count(), Sum(), Min(), Max()
  - Algebraic:

        - If it can be computed by an algebraic function with M arguments (where M is a bounded integer), each of which is obtained by applying a distributive aggregate function.
        - Eg. Avg(), min_N(), standard_deviation()
    - Holistic:
        - If there is no constant bound on the storage size needed to describe a subaggregate
        - Eg. Median(), Mode(), rank()
- A Concept Hierarchy for the Dimension Location:
    - Concept Hierarchy – Defines a sequence of mappings from a set of low level concepts to higher level more general concepts.
    - There can b more than one concept hierarchy for a given attribute or dimension, based on different user view points.
    - These are automatically generated or pre-defined by domain-experts or users.

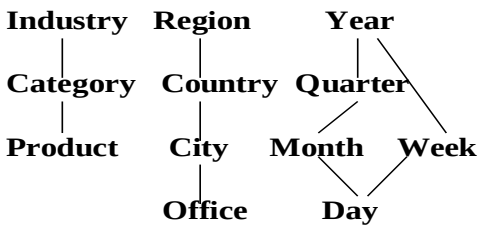


Multidimensional Data

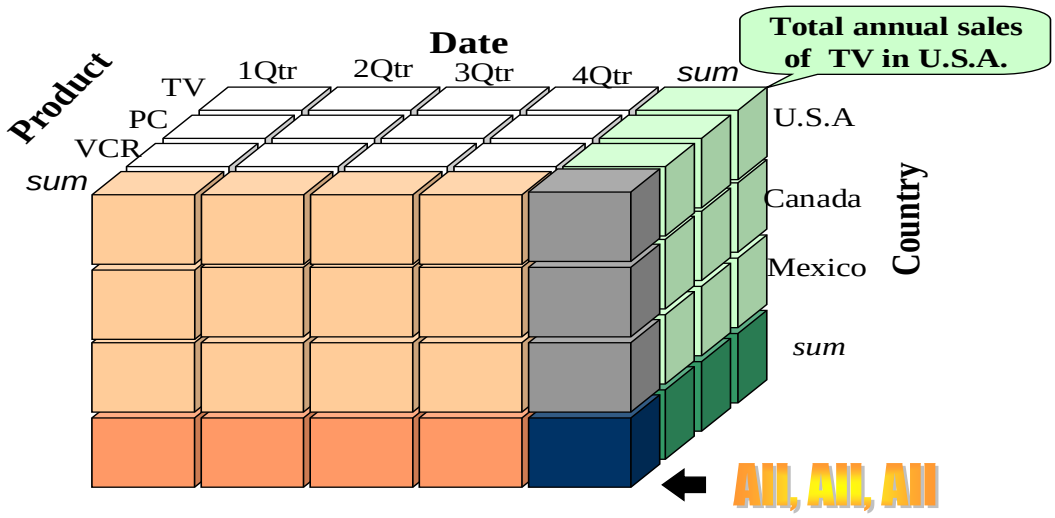
- Sales volume as a function of product, month, and region



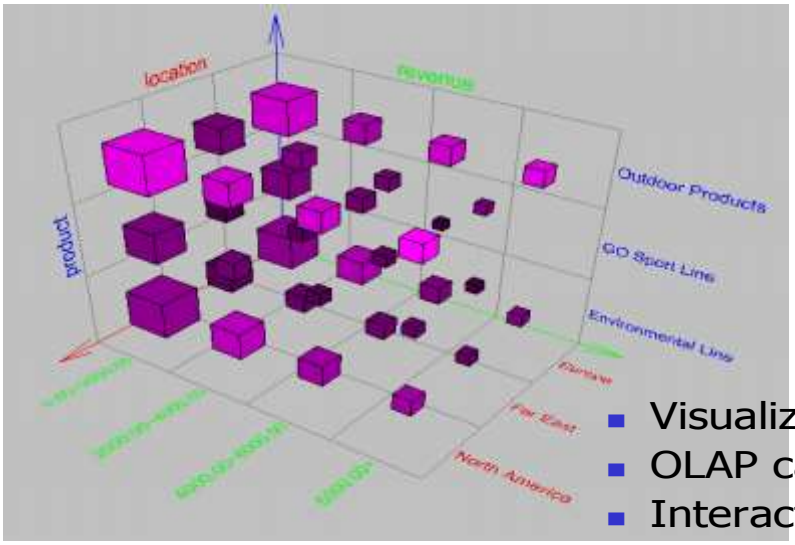
Dimensions: Product, Location, Time
Hierarchical summarization paths



A Sample Data Cube



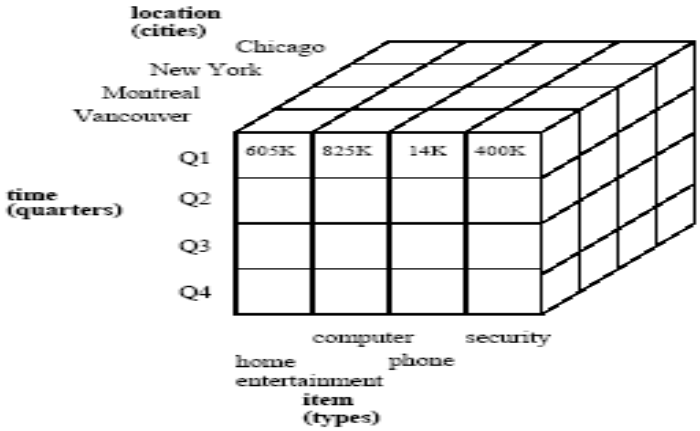
Browsing a Data Cube



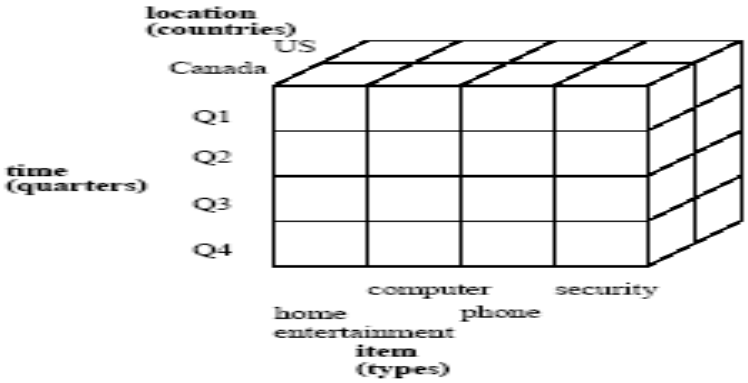
- Visualization
- OLAP capabilities
- Interactive manipulation

July 20, 2010 Data Mining: Concepts and Techniques 28

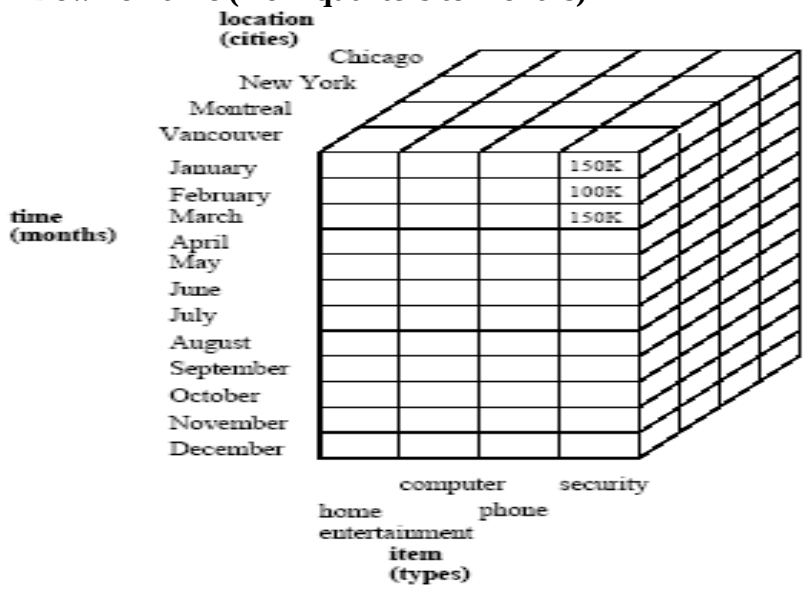
- Typical OLAP Operations:
  - Roll-Up (drill-up): Summarize data
    - By climbing up the hierarchy or by dimension reduction.
  - Drill-Down (roll down): Reverse of roll-up
    - From higher level summary to lower level summary or detailed data, or introducing new dimensions.
  - Slice and Dice: Project and Select
  - Pivot (rotate):
    - Re-orient the cube, visualization, 3D to series of 2D planes.
  - Other operations:
    - Drill across: Querying more than one fact table (Using SQL).
    - Drill through: Queries the back end relational tables through the bottom level of the cube. (Using SQL).



- **Roll-up on location (from cities to country)**

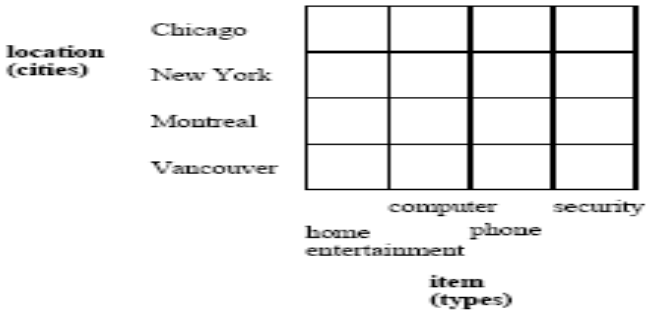


- **Drill-Down on time (from quarters to months)**

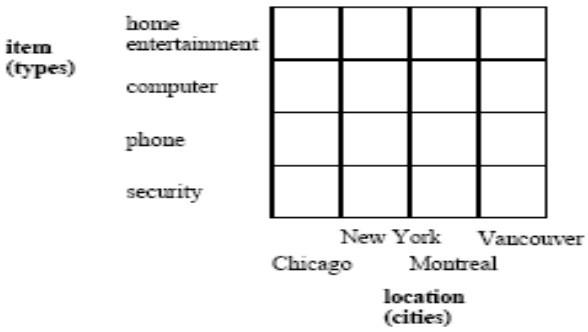


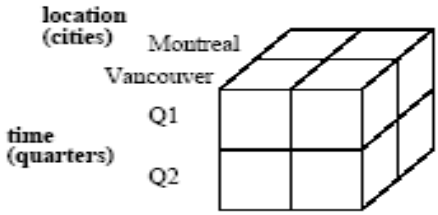
-

- **Slice for time = "Q2"**



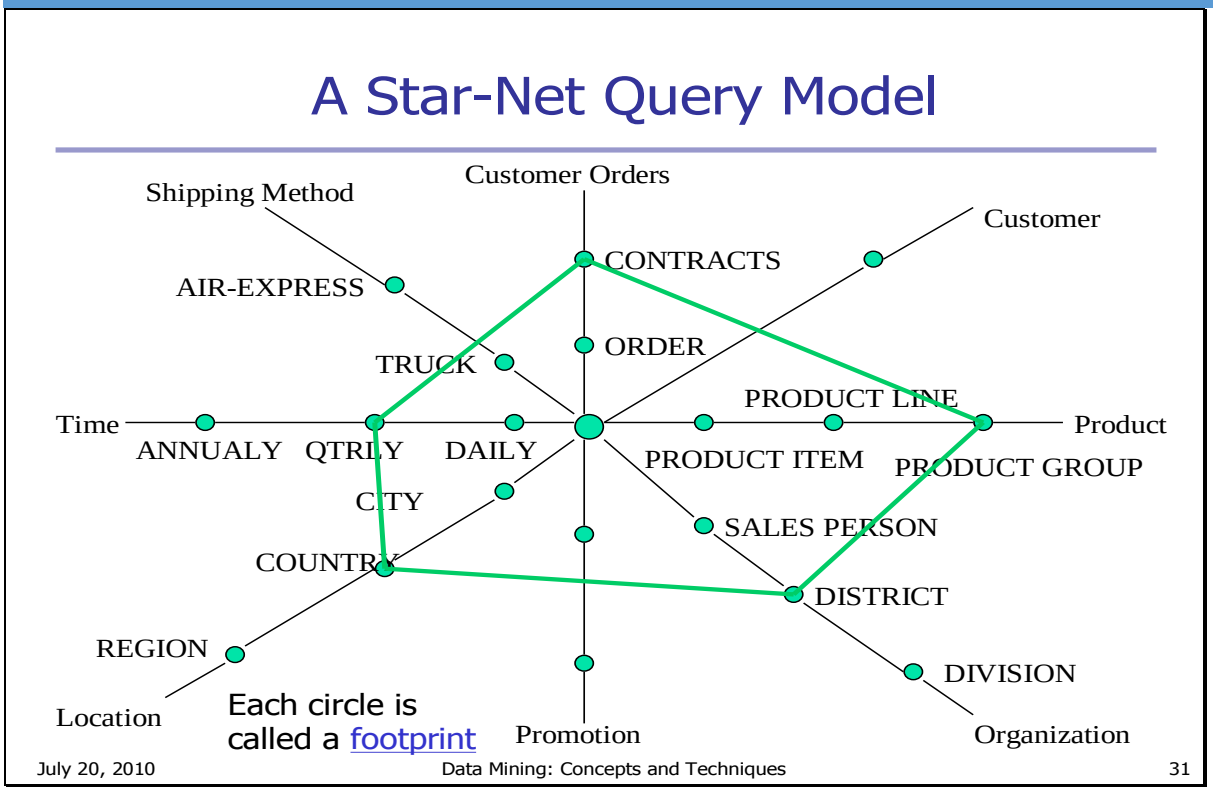
- **Pivot**





dice for
(location="Montreal" or "Vancouver") and
(time="Q1" or "Q2") and
(item="home entertainment" or "computer")

- A Star Net Query Model for querying multi dimensional databases:
  - A star net model consists of radial lines emanating from a central point where each line represents a concept hierarchy for a dimension.
  - Each abstraction level in the hierarchy is called a foot print.



4.3 Data Warehouse Architecture

Data Warehouse Architecture:

- Design of Data Warehouse: A Business Analysis Framework:
  - There are four views regarding the design of the data warehouse:
    - Top-Down View:
      - Allows the selection of relevant information necessary for the data warehouses that suits the business needs.
    - Data Source View:
      - Exposes the information being captured, stored and managed by operational systems.
      - This information may be at various levels of detail and accuracy, from individual data source table to integrated data source tables.
    - Data Warehouse View:
      - Includes Fact tables and dimension tables stored inside the data warehouse.
      - This includes pre-calculated totals and counts as well as the source information such as Date, Time to provide historical perspective.
    - Business Query View:

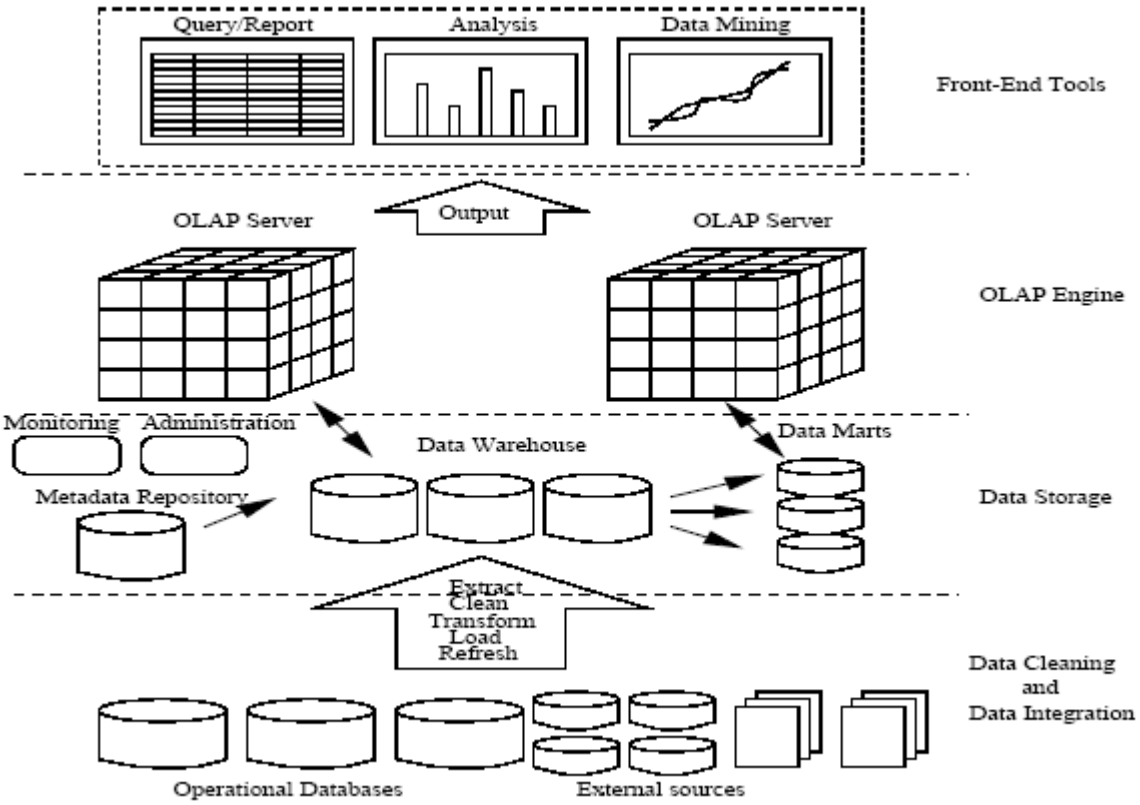
- Sees the perspectives of data in the warehouse from the view of end-user.

- Designing a Data warehouse is a complex task and it requires:
  - Business Skills: (Business Analysts)
    - Understanding and translating the business requirements into queries that the data warehouse can satisfy.
  - Technology Skills: (Data Analysts)
    - To understand how to derive facts and dimensions of the data warehouse.
    - Ability to discover patterns & trends based on history and to detect anomaly
    - Present relevant information as per managerial need based on such analysis
  - Program Management Skills: (Manager)
    - Interfacing with many technologies, vendors and end users so as to deliver results in a timely and cost effective manner.
- Extractors: Transfer data from operational system to the data warehouse.
- Warehouse Refresh Software: Keeps the data warehouse up to date with operational system's data.
- Building a Data warehouse requires understanding of how to store and manage data, how to build extractors and how to build warehouse refresh software.
- Data warehouse Design / Build Process:
  - Top-Down, Bottom-up and combination of both.
  - Top-Down:
    - Starts with overall design and planning (technology & business mature and well known)
    - Minimizes integration, Expensive, takes long time, lacks flexibility
  - Bottom-Up:
    - Starts with experiments and prototypes (rapid & less expensive)
    - Flexible, low cost, rapid return on investment, integration is a problem
- From software engineering point of view, design and construction of a data warehouse consists of the steps:
  - Planning, Requirements Study, Problem Analysis, Warehouse Design, Data integration and testing and Deployment of a data warehouse
- Data warehouse design and construction follows two methodologies:
  - Waterfall method:
    - Structured and systematic analysis at each step before proceeding to the next.
  - Spiral method:

- Rapid generation of increasingly functional systems, short turn around time, modifications done quickly (well suited for data warehouses or data marts).

- Typical data warehouse design process includes the below steps:
  - Choose a business process to model.
    - Eg. Invoice, accounts, sales, inventory…
    - If the business process is Organizational – model data warehouse
    - If the business process is departmental – model data mart
  - Choose the grain (atomic level of data in fact table) of the business process.
    - Eg. Daily_sales or monthly_sales
  - Choose the dimensions that will apply to each fact table record.
    - Eg. Time, Item, Customer, Supplier…
  - Choose the measure that will populate each fact table record.
    - Measures are numeric additive quantities. Eg. Dollars_sold, units_sold
- Goals of Data Warehouse implementation:
  - Specific, Achievable, Measurable
  - Determine time, budget, organizations to be modeled, and departments to be served.
- Steps after data warehouse implementation:
  - Initial installation, rollout planning, training, platform upgrades and maintenance.
- Data warehouse administration includes:
  - Data refreshment, data source synchronization, planning for disaster recovery, access control and security management, managing data growth, managing database performance, data warehouse enhancement and extension.

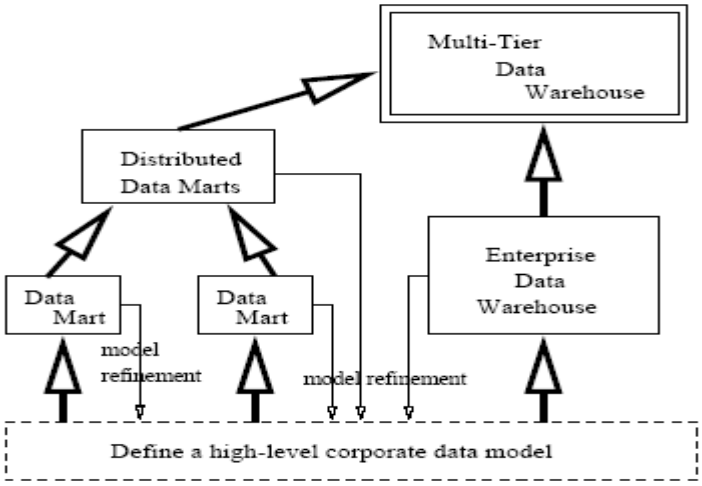
- A three-tier data warehouse architecture:
  - Bottom Tier = Warehouse Database Server
  - Middle Tier = OLAP Server (Relational OLAP / Multidimensional OLAP)
  - Top Tier = Client – Query, analysis, reporting or data mining tools



- There are 3 data warehouse models based on the architecture:
  - Enterprise Warehouse:
    - Corporate wide data integration, spanning all subjects
    - One or more operational source systems; Takes years to design and build
    - Cross functional in scope; Size of data – gigabytes to terabytes
    - Implemented on mainframes / UNIX / parallel architecture platforms
  - Data Mart:
    - Subset of corporate wide data, spanning on selected subjects
    - Eg. Marketing data mart – subjects are customer, item and sales.
    - Implementation takes few weeks
    - Implemented on low cost UNIX / Windows/NT server.
    - 2 categories – based on source of data:
      - Independent data marts – Source from a department data
      - Dependent data marts – Source is an enterprise data warehouse
  - Virtual Warehouse:
    - Set of views from the operational databases
    - Summary views are materialized for efficient query processing

- Easy to build but requires excess capacity of operational db servers

- A recommended approach for Data Warehouse development:
  - Implement a warehouse in an incremental and evolutionary manner.
  - First – Define a high level corporate data model (in 1 or 2 months)
  - Second – independent data marts developed in parallel with enterprise data warehouse
    - Corporate data model refined as this development progresses.
  - Third – Multi-Tier-Data Warehouse constructed – Consists of enterprise data warehouse which in turn communicates with departmental data marts.



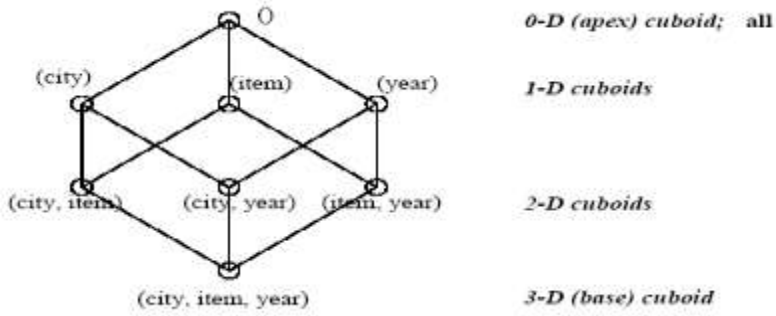
- OLAP Server Architectures:
  - Relational OLAP Servers (ROLAP):
    - Intermediate server between Relational back-end server and Client front-end tools.
    - Relational / Extended relational DBMS to store and manage warehouse data and OLAP middleware
    - Include optimization of DBMS backend, implementation of aggregation navigation logic, and additional tools and services.
    - Greater scalability
    - Eg. Metacube of Informix
  - Multidimensional OLAP Servers (MOLAP):
    - Supports multi dimensional views of data
    - Sparse array-based multi-dimensional storage engine.
    - Maps multidimensional views to data cube array structures
    - Fast indexing to pre-compute summarized data
    - Eg. Essbase of Arbor
  - Hybrid OLAP Servers (HOLAP):
    - Combines ROLAP and MOLAP architectures
    - Benefits from high scalability of ROLAP and fast computation of MOLAP
    - Eg. Microsoft SQL Server
  - Specialized SQL Servers:

- OLAP processing in relational databases (read only environment)
- Advanced query language and query processing support.

4.4 Data Warehouse Implementation

Data warehouse Implementation

- It is important for a data warehouse system to be implemented with:
  - Highly Efficient Computation of Data cubes
  - Access Methods; Query Processing Techniques



- Efficient Computation of Data cubes:
  - = Efficient computation of aggregations across many sets of dimensions.
  - Compute Cube operator and its implementations:
    - Extends SQL to include compute cube operator
    - Create Data cube for the dimensions item, city, year and sales_in_dollars:
      - Example Queries to analyze data:
        - Compute sum of sales grouping by item and city
        - Compute sum of sales grouping by item
        - Compute sum of sales grouping by city
      - Here dimensions are item, city and year; Measure / Fact is sales_in_dollars
      - Hence total number of cuboids or group bys for this data cube is $2^3 = 8$.
      - Possible group bys are {(city, item, year), (city, item), (city, year), (item, year), (city), (item), (year), ()}; These group bys forms the lattice of cuboid
      - 0-D (Apex) cuboid is (); 3-D (Base) cuboid is (city, item, year)

  - Hence, for a cube with n dimensions there are total 2^n cuboids.
  - The statement 'compute cube sales' computes sales aggregate cuboids for the eight subsets.
  - Pre-computation of cuboids leads to faster response time and avoids redundant computation.
  - But challenge in pre-computation is that the required storage space may explode.
  - Number of cuboids in an n-dimensional data cube if there are no concept hierarchy attached with each dimension = 2^n cuboids.
  - Consider Time dimension has the concept hierarchy $day < week < month < quarter < year.$
  - Then total number of cuboids are: $T = \prod_{i=1}^{n}(L_i + 1),$ where Li is the number of levels associated with the dimension i.
  - Eg. If a cube has 10 dimensions and each dimension has 4 levels, then total number of cuboids generated will be 5^{10}.
  - This shows it is unrealistic to pre-compute and materialize all cuboids for a data cube.

- Hence we go for Partial Materialization:
  - Three choices of materialization:
  - No Materialization:
    - Pre-compute only base cuboid and no other cuboids; Slow computation
  - Full Materialization: Pre-compute all cuboids; Requires huge space
  - Partial Materialization: Pre-compute a proper subset of whole set of cuboids
    - Considers 3 factors:
      - Identify cuboids to materialize – based on workload, frequency, accessing cost, storage need, cost of update, index usage. (or simply use greedy Algorithm that has good performance)
      - Exploit materialized cuboids during query processing
      - Update materialized cuboid during load and refresh (use parallelism and incremental update)
- Multiway array aggregation in the computation of data cubes:
  - To ensure fast online analytical processing we need to go for full materialization
  - But should consider amount of main memory available and time taken for computation.
  - ROLAP and MOLAP uses different cube computation techniques.
  - Optimization techniques for ROLAP cube computations:
    - Sorting, hashing and grouping operations applied to dimension attributes – to reorder and cluster tuples.
    - Grouping performed on some sub aggregates – 'Partial grouping step' – to speed up computations

        - Aggregates computed from sub aggregates (rather than from base tables).
        - In ROLAP dimension values are accessed by using value-based / key-based addressing search strategies.
    - Optimization techniques for MOLAP cube computations:
        - MOLAP uses direct array addressing to access dimension values
        - Partition the array into Chunks (sub cube small enough to fit into main memory).
        - Compute aggregates by visiting cube cells. The number of times each cell is revisited is minimized to reduce memory access and storage costs.
        - This is called as multiway array aggregation in data cube computation.
    - MOLAP cube computation is faster than ROLAP cube computation
- Indexing OLAP Data: Bitmap Indexing; Join Indexing
- Bitmap Indexing:
    - Index on a particular column; Each distinct value in a column has a bit vector
    - The length of each bit vector = No. of records in the base table
    - The i-th bit is set if the i-th row of the base table has the value for the indexed column.
    - This approach is not suitable for high cardinality domains

Base table

Cust	Region	Type
C1	Asia	Retail
C2	Europe	Dealer
C3	Asia	Dealer
C4	America	Retail
C5	Europe	Dealer

Index on Region

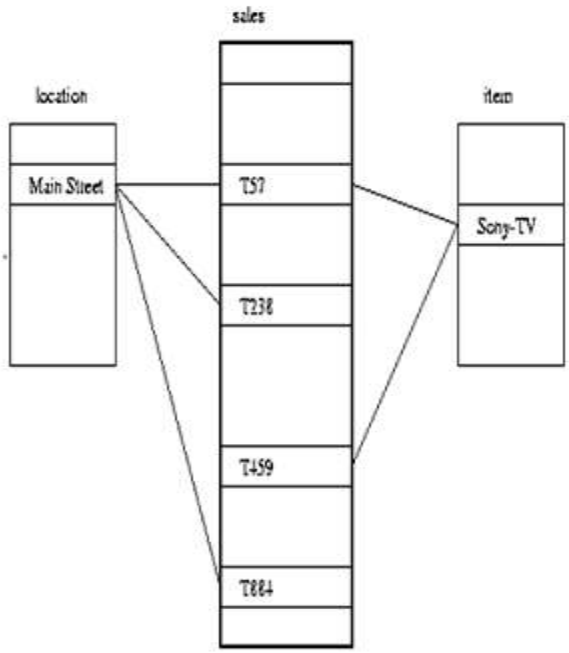
RecID	Asia	Europe	America
1	1	0	0
2	0	1	0
3	1	0	0
4	0	0	1
5	0	1	0

Index on Type

RecID	Retail	Dealer
1	1	0
2	0	1
3	0	1
4	1	0
5	0	1

- Join Indexing:
    - Registers joinable rows of two relations
    - Consider Relations R & S
    - Let R (RID, A) & S (SID, B); where RID and SID are record identifiers of R & S Respectively.
    - For joining the attributes A & B the join index record contains the pair (RID, SID).
    - Hence in traditional databases the join index maps the attribute values to a list of record ids
    - But, in data warehouses join index relates the values of the dimensions of a star schema to rows in the fact table.
    - Eg. Fact table: Sales and two dimensions city and product
    - A join index on city maintains for each distinct city a list of R-IDs of the tuples of the Fact table Sales
    - Join indices can span across multiple dimensions. – **Composite join indices**
    - To speed up query processing join indexing

and bitmap indexing can be integrated to form **Bitmapped join indices**.



- Efficient processing of OLAP queries:
  - Steps for efficient OLAP query processing:
  - 1. Determine which OLAP operations should be performed on the available cuboids:
    - Transform the OLAP operations like drill-down, roll-up,… to its corresponding SQL (relational algebra) operations.
    - Eg. Dice = Selection + Projection
  - 2. Determine to which materialized cuboids the relevant OLAP operations should be applied:
    - Involves (i) Pruning of cuboids using knowledge of "dominance" (ii) Estimate the cost of remaining materialized cuboids (iii) Selecting the cuboid with least cost
    - Eg. Cube: "Sales [time, item, location]: sum(sales_in_dollars)"
    - Dimension hierarchies used are:
      - "day < month < quarter < year" for time dimension
      - "Item_name < brand < type for item dimension
      - "street < city < state < country for location dimension
    - Say query to be processed is on {brand, state} with the condition year = "1997"
    - Say there are four materialized cuboids available
      - Cuboid 1: {item_name, city, year} ; Cuboid 2: {brand, country, year}
      - Cuboid 3: {brand, state, year} ;
      - Cuboid 4: {item_name, state} where year = 1997
    - Which cuboid selected for query processing?
    - Step 1: Pruning of cuboids – prune cuboid 2 as higher level of concept "country" can not answer query at lower granularity "state"
    - Step 2: Estimate cuboid cost; Cuboid 1 costs the most of the 3 cuboids as item_name and city are at a finer granular level than brand and state as mentioned in the query.
    - Step 3: If there are less number of years and there are more number of item_names under each brand then Cuboid 3 has the least cost. But if otherwise and there are efficient indexes on item_name then Cuboid 4 has the least cost. Hence select Cuboid 3 or Cuboid 4 accordingly.
- Metadata repository:
  - Metadata is the data defining warehouse objects. It stores:
    - Description of the structure of the data warehouse:
      - Schema, views, dimensions, hierarchies, derived data definition, data mart locations and contents
    - Operational meta data:
      - Data lineage: History of migrated data and its transformation path
      - Currency of data: Active, archived or purged
      - Monitoring information:
        - Warehouse usage statistics, error reports, audit trails
    - Algorithms used for summarization:
      - Measure and Dimension definition algorithm

- Granularity, Subject, Partitions definition
- Aggregation, Summarization and pre-defined queries and reports.

▪ Mapping from operational environment to the data warehouse:
- Source database information; Data refresh & purging rules
- Data extraction, cleaning and transformation rules
- Security rules (authorization and access control)

▪ Data related to system performance:
- Data access performance; data retrieval performance
- Rules for timing and scheduling of refresh

▪ Business metadata:
- Business terms and definitions
- Data ownership information; Data charging policies

- Data Warehouse Back-end Tools and Utilities:
  - Data Extraction: Get data from multiple, heterogeneous and external sources
  - Data Cleaning: Detects 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
    - Build indices and partitions
  - Refresh: Propagates the updates from data sources to the warehouse

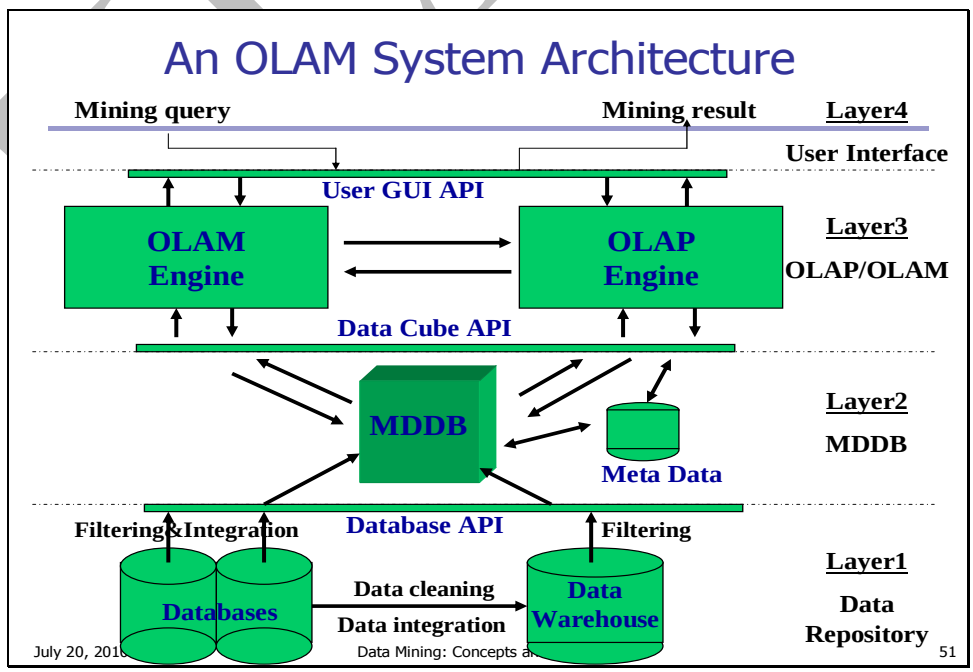
4.5 Mapping the Data Warehouse to Multiprocessor Architecture

From Data warehousing to Data Mining:

- Data Warehousing Usage:
  - Data warehouse and Data Marts used in wide range of applications;
  - Used in Feedback system for enterprise management – "Plan-execute-assess Loop"
  - Applied in Banking, Finance, Retail, Manufacturing,…
  - Data warehouse used for knowledge discovery and strategic decision making using data mining tools
  - There are three kinds of data warehouse applications:
    - Information Processing:
      - Supports querying & basic statistical analysis
      - Reporting using cross tabs, tables, charts and graphs
    - Analytical Processing:
      - Multidimensional analysis of data warehouse data
      - Supports basic OLAP operations slice-dice, drilling and pivoting
    - Data Mining:
      - Knowledge discovery from hidden patterns
      - Supports associations, classification & prediction and Clustering
      - Constructs analytical models
      - Presenting mining results using visualization tools

- From Online Analytical Processing (OLAP) to Online Analytical Mining (OLAM):
  - OLAM also called as OLAP Mining – Integrates OLAP with mining techniques
  - Why OLAM?
    - High Quality of data in data warehouses:
      - DWH has cleaned, transformed and integrated data (Preprocessed data)
      - Data mining tools need such costly preprocessing of data.
      - Thus DWH serves as a valuable and high quality data source for OLAP as well as for Data Mining
    - Available information processing infrastructure surrounding data warehouses:
      - Includes accessing, integration, consolidation and transformation of multiple heterogeneous databases ; ODBC/OLEDB connections;
      - Web accessing and servicing facilities; Reporting and OLAP analysis tools
    - OLAP-based exploratory data analysis:
      - OLAM provides facilities for data mining on different subsets of data and at different levels of abstraction
      - Eg. Drill-down, pivoting, roll-up, slicing, dicing on OLAP and on intermediate DM results
      - Enhances power of exploratory data mining by use of visualization tools
    - On-line selection of data mining functions:
      - OLAM provides the flexibility to select desired data mining functions and swap data mining tasks dynamically.

Architecture of Online Analytical Mining:



- OLAP and OLAM engines accept on-line queries via User GUI API

- And they work with the data cube in data analysis via Data Cube API
- A meta data directory is used to guide the access of data cube
- MDDB constructed by integrating multiple databases or by filtering a data warehouse via Database API which may support ODBC/OLEDB connections.
- OLAM Engine consists of multiple data mining modules – Hence sophisticated than OLAP engine.
- Data Mining should be a human centered process – users should often interact with the system to perform exploratory data analysis

4.6 OLAP Need

OLAP systems vary quite a lot, and they have generally been distinguished by a letter tagged onto the front of the word OLAP. ROLAP and MOLAP are the big players, and the other distinctions represent little more than the marketing programs on the part of the vendors to distinguish themselves, for example, SOLAP and DOLAP. Here, we aim to give you a hint as to what these distinctions mean.

4.7 Categorization of OLAP Tools

Major Types:

Relational OLAP (ROLAP) –Star Schema based

Considered the fastest growing OLAP technology style, ROLAP or "Relational" OLAP systems work primarily from the data that resides in a relational database, where the base data and dimension tables are stored as relational tables. This model permits multidimensional analysis of data as this enables users to perform a function equivalent to that of the traditional OLAP slicing and dicing feature. This is achieved thorough use of any SQL reporting tool to extract or 'query' data directly from the data warehouse. Wherein specifying a 'Where clause' equals performing a certain slice and dice action.

One advantage of ROLAP over the other styles of OLAP analytic tools is that it is deemed to be more scalable in handling huge amounts of data. ROLAP sits on top of relational databases therefore enabling it to leverage several functionalities that a relational database is capable of. Another gain of a ROLAP tool is that it is efficient in managing both numeric and textual data. It also permits users to "drill down" to the leaf details or the lowest level of a hierarchy structure.

However, ROLAP applications display a slower performance as compared to other style of OLAP tools since, oftentimes, calculations are performed inside the server. Another demerit of a ROLAP tool is that as it is dependent on use of SQL for data manipulation, it may not be ideal for performance of some calculations that are not easily translatable into an SQL query.

Multidimensional OLAP (MOLAP) –Cube based

Multidimensional OLAP, with a popular acronym of MOLAP, is widely regarded as the classic form of OLAP. One of the major distinctions of MOLAP against a ROLAP tool is that data are pre-summarized and are stored in an optimized format in a multidimensional cube, instead of in a relational database. In this type of model, data are structured into proprietary

formats in accordance with a client's reporting requirements with the calculations pre-generated on the cubes.
This is probably by far, the best OLAP tool to use in making analysis reports since this enables users to easily reorganize or rotate the cube structure to view different aspects of data. This is done by way of slicing and dicing. MOLAP analytic tool are also capable of performing complex calculations. Since calculations are predefined upon cube creation, this results in the faster return of computed data. MOLAP systems also provide users the ability to quickly write back data into a data set. Moreover, in comparison to ROLAP, MOLAP is considerably less heavy on hardware due to compression techniques. In a nutshell, MOLAP is more optimized for fast query performance and retrieval of summarized information.
There are certain limitations to implementation of a MOLAP system, one primary weakness of which is that MOLAP tool is less scalable than a ROLAP tool as the former is capable of handling only a limited amount of data. The MOLAP approach also introduces data redundancy. There are also certain MOLAP products that encounter difficulty in updating models with dimensions with very high cardinality.

Hybrid OLAP (HOLAP)

HOLAP is the product of the attempt to incorporate the best features of MOLAP and ROLAP into a single architecture. This tool tried to bridge the technology gap of both products by enabling access or use to both multidimensional database (MDDB) and Relational Database Management System (RDBMS) data stores. HOLAP systems stores larger quantities of detailed data in the relational tables while the aggregations are stored in the pre-calculated cubes. HOLAP also has the capacity to "drill through" from the cube down to the relational tables for delineated data.
Some of the advantages of this system are better scalability, quick data processing and flexibility in accessing of data sources.

Other Types:

There are also less popular types of OLAP styles upon which one could stumble upon every so often. We have listed some of the less famous types existing in the OLAP industry.

Web OLAP (WOLAP)

Simply put, a Web OLAP which is likewise referred to as Web-enabled OLAP, pertains to OLAP application which is accessible via the web browser. Unlike traditional client/server OLAP applications, WOLAP is considered to have a three-tiered architecture which consists of three components: a client, a middleware and a database server.
Probably some of the most appealing features of this style of OLAP are the considerably lower investment involved, enhanced accessibility as a user only needs an internet connection and a web browser to connect to the data and ease in installation, configuration and deployment process.
But despite all of its unique features, it could still not compare to a conventional client/server machine. Currently, it is inferior in comparison to OLAP applications which involve deployment in client machines in terms of functionality, visual appeal and performance.

Desktop OLAP (DOLAP)

Desktop OLAP, or "DOLAP" is based on the idea that a user can download a section of the data from the database or source, and work with that dataset locally, or on their desktop. DOLAP is easier to deploy and has a cheaper cost but comes with a very limited functionality in comparison with other OLAP applications.

Mobile OLAP (MOLAP)

Mobile OLAP is merely refers to OLAP functionalities on a wireless or mobile device. This enables users to access and work on OLAP data and applications remotely thorough the use of their mobile devices.

Spatial OLAP (SOLAP)

With the aim of integrating the capabilities of both Geographic Information Systems (GIS) and OLAP into a single user interface, "SOLAP" or Spatial OLAP emerged. SOLAP is created to facilitate management of both spatial and non-spatial data, as data could come not only in an alphanumeric form, but also in images and vectors. This technology provides easy and quick exploration of data that resides on a spatial database.

Other different blends of an OLAP product like the less popular 'DOLAP' and 'ROLAP' which stands for Database OLAP and Remote OLAP, 'LOLAP' for Local OLAP and 'RTOLAP' for Real-Time OLAP are existing but have barely made a noise on the OLAP industry.

Review Questions

Two Marks:

1. What is a data warehouse?
2. Compare Data Warehouse with Heterogeneous DBMS.
3. Distinguish between OLTP and OLAP systems.
4. Why do we need a data warehouse?
5. Write about the three types of modeling for data warehouses.
6. Define a Star Schema using DMQL?
7. Define a Snowflake Schema using DMQL?
8. Define a Fact Constellation Schema using DMQL?
9. What are the three measures of a data cube?
10. What is a (i) Base Cuboid (ii) Apex Cuboid (iii) Data Cube. Give Examples.
11. What is Concept hierarchy? Explain with an example.
12. What is a star net query model? Depict using diagram.
13. Detail on available OLAP server architectures.
14. Explain about the three choices of Materialization.

Sixteen Marks:

1. (i) Compare the features of OLTP and OLAP systems. (6)
   (ii) Explain the various OLAP operations with examples. (10)
2. Explain in detail about the Data warehousing architecture with suitable architecture diagram. (16)
3. (i) Detail on how OLAP data is indexed using bitmap indexing and join indexing. (4)
   (ii) Discuss the steps for efficient OLAP Query processing. (4)
   (iii) Write about Metadata Repository (4)
   (iv) Write notes on Data Warehouse Tools and Utilities. (4)
4. (i) Write notes on Data Warehousing Usage. (4)
   (ii) Why do we go from OLAP to OLAM? (6)
   (iii) Discuss the cube computation techniques used by ROLAP and MOLAP. (6)

Assignment Topic:

1. Write about OLAP Need and Categorization of OLAP Tools.