

Announcement

- HW1 score out; TA office hour Tue 3:30-4:30 pm
 - Min 23, Max 100, Mode 80
 - **Mean 85.1**, Median 87, **Std Dev 12.9**
 - **Carefully read questions, come to class, read slides/textbook**
- Next HWs
 - We will have fewer questions
 - **PDF version!!! NDID-HWx.pdf**
 - **No extension!!!**
 - **Type your answers!!! Explain your answers!!!**
- Our TA was *really really generous* for HW1 ❤️❤️❤️
- Expect to see good project results: Surprise me!

How to Work

If you want to do something,
Be Determined and *Work Hard!*
and *Work Smart!*

Q-Q plot

- Data object and attributes
- Scatter plot
- Q-Q plot
 - An application



Chapter 4&5. Data Cube: Data Warehousing and OLAP

Meng Jiang

CSE 40647/60647 Data Science Fall 2017

Introduction to Data Mining

Review: Data Cube

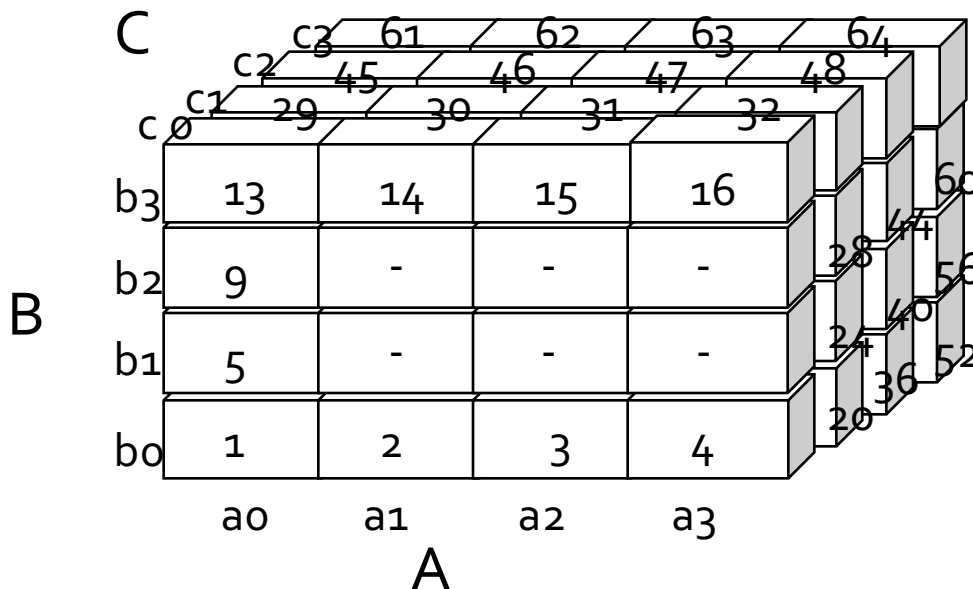
- Concepts
 - Cell, Cuboid, Cube
 - Dimension Value, Dimension Level, Dimension
 - Base/Aggregate Cell/Cuboid
- Components
 - Dimension tables and Fact tables
 - Concept hierarchy and Measures
 - Schemas: Star, Snowflake, Constellation
- Operations: Roll-up, drill-down, dice, slide, pivot...
- Materialization: Full, no, **partial**
 - **Cells: Iceberg cube**
 - **Dimensions: Aggregation?**

Efficient Computation

- General computation heuristics (Agarwal et al.'96)
- Computing full/iceberg cubes: 3 methodologies
 - Bottom-Up:
 - Multi-way array aggregation (Zhao, Deshpande & Naughton, SIGMOD'97)
 - Top-down:
 - **BUC** (Beyer & Ramakrishnan, SIGMOD'99)
 - Integrating Top-Down and Bottom-Up:
 - Star-cubing algorithm (Xin, Han, Li & Wah: VLDB'03)
- High-dimensional OLAP:
 - **A shell-fragment approach** (Li, et al. VLDB'04)
- Computing alternative kinds of cubes:
 - Partial cube, closed cube, approximate cube,

Multi-Way Array Aggregation

- Bottom-up: Partition a huge *sparse* array into *chunks* (a small subcube which fits in memory) and aggregation.
- Data addressing: Compressed *sparse array addressing* (chunk_id, offset)
- Compute **aggregates in “multiway”** by visiting cube cells in the order which **minimizes** the # of times to visit each cell, and **reduces** memory access and storage cost



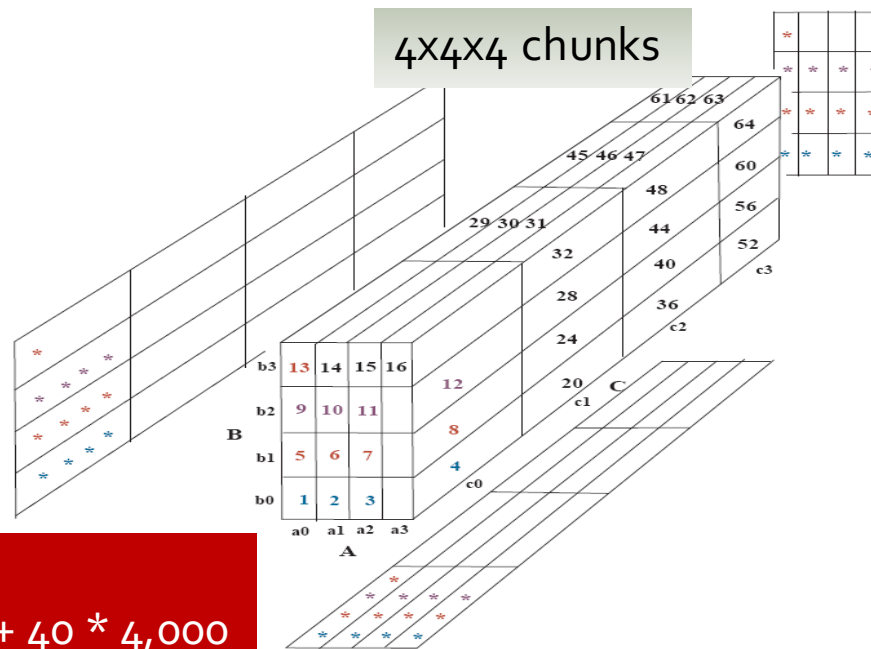
What is the best traversing order to do multi-way aggregation?

ABC → AB, BC and AC

A: 40 (location),
B: 400 (item),
C: 4,000 (time)

Multi-way Array Aggregation (3-D to 2-D)

- How much **memory cost of computation** (aggregation for **AB, AC, BC planes**) can we save?



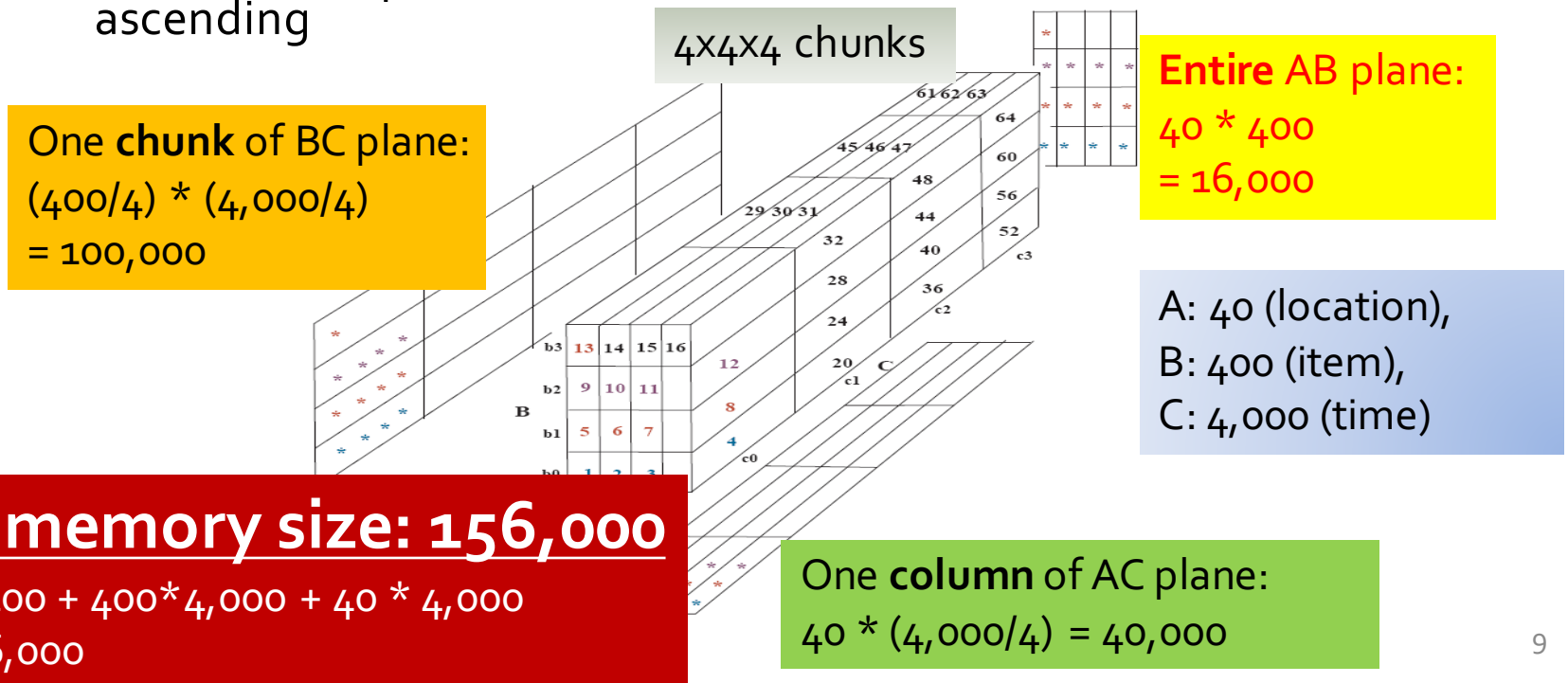
A: 40 (location),
B: 400 (item),
C: 4,000 (time)

Min memory size: ?

$$\begin{aligned}
 &< 40 * 400 + 400 * 4,000 + 40 * 4,000 \\
 &= 1,776,000
 \end{aligned}$$

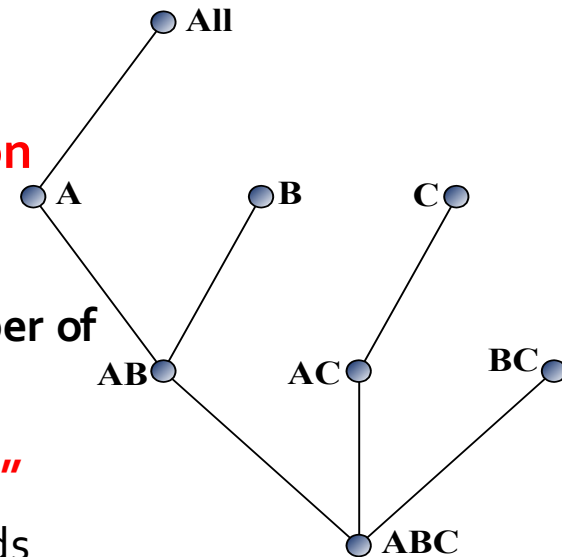
Multi-way Array Aggregation (3-D to 2-D)

- How to minimize the memory requirement and reduced I/Os?
 - Keep the **smallest** plane in **main memory**
 - Fetch and compute **only one chunk** at a time for the **largest** plane
 - The planes should be **sorted** and computed according to their **size** in ascending



Multi-Way Array Aggregation

- Array-based “**bottom-up**” algorithm (from ABC to AB,...)
- Using multi-dimensional **chunks**
- Simultaneous aggregation on multiple dimensions
- **Cannot do *Apriori* pruning: No iceberg optimization**
- Comments on the method
 - Efficient for computing the full cube for a **small number of dimensions**
 - If there are a large number of dimensions, “**top-down**” **computation** and **iceberg cube computation** methods should be used



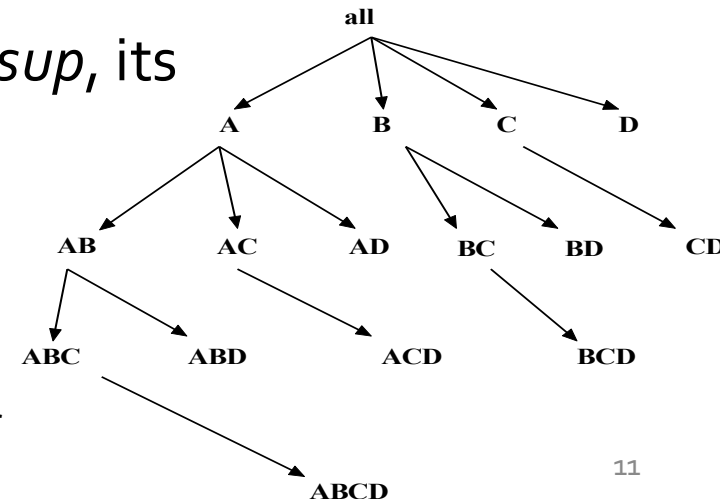
BUC (Top Down: From AB to ABC)

- BUC (Beyer & Ramakrishnan, SIGMOD'99)

BUC: acronym of Bottom-Up (cube) Computation

(Note: It is “**top-down**” in our view since we put Apex cuboid on the top!)

- Divides dimensions into partitions and facilitates **iceberg pruning (it works now!)**
 - If a partition does not satisfy *min_sup*, its **descendants** can be pruned



Data Warehouse: From Tables and Spreadsheets to Data Cubes

- A **data warehouse** is based on a multidimensional 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 table** contains **measures** (such as dollars_sold) and keys to each of the related dimension tables
- **Data cube**: A lattice of cuboids
 - In data warehousing literature, an **n-D base cube** is called a **base cuboid**
 - The top most **o-D cuboid**, which holds the highest-level of summarization, is called the **apex cuboid**
 - The lattice of cuboids forms a **data cube**

Data Warehouse

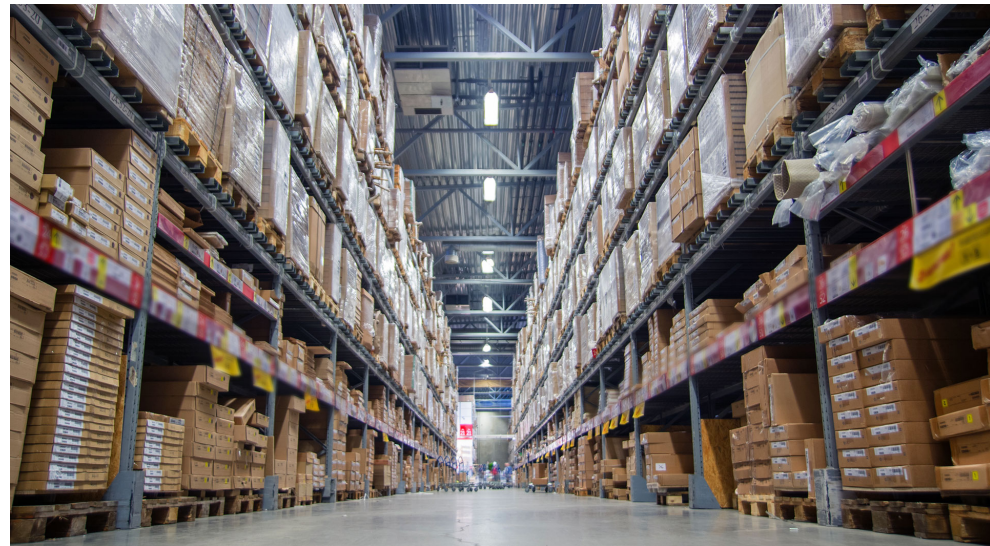
- Defined in many different ways, but not rigorously
 - A decision support database that is maintained separately from the organization's operational database

Operational Databases



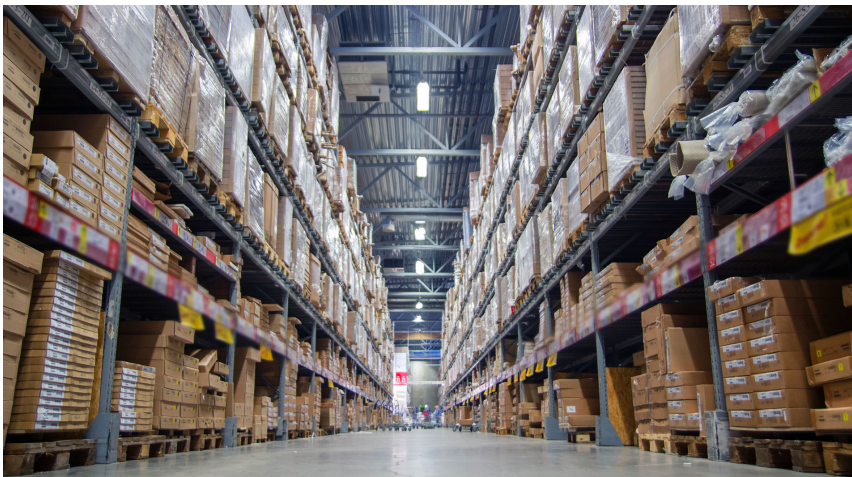
(Data) Marts

(Data) Warehouse



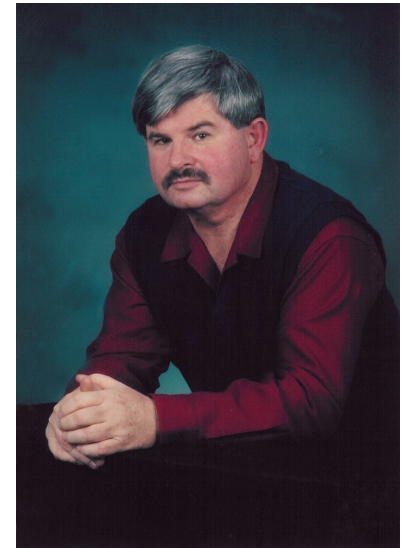
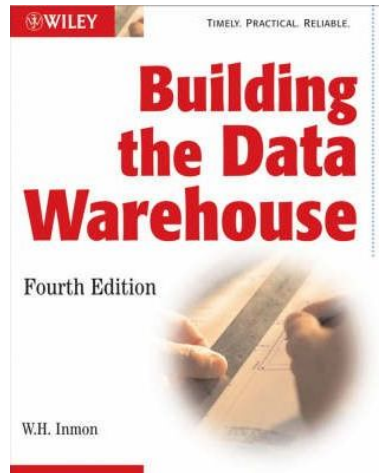
Data Warehouse

- Defined in many different ways, but not rigorously
 - A decision support database that is maintained **separately** from the organization's operational database
 - Support **information processing** by providing a solid platform of consolidated, **historical data for analysis**



Data Warehouse

- “A data warehouse is a subject-oriented, integrated, time-variant, and nonvolatile collection of data in support of management’s decision-making process.”—William H. (Bill) Inmon



- Data warehousing:
 - The process of constructing and using data warehouses

(1) Subject-Oriented

- Organized around major subjects, such as **customer, product, sales**
- Focusing on the modeling and analysis of data for decision makers, NOT on daily operations or transaction processing
- Provide **a simple and concise** view around particular subject issues by **excluding data that are not useful in the decision support process**

(2) Integrated

- Constructed by integrating multiple, heterogeneous data sources
 - relational databases, flat files, on-line transaction records
- Data cleaning and data integration techniques are applied
 - Ensure **consistency** in naming conventions, encoding structures, attribute measures, etc. among different data sources
 - Ex. Hotel price: differences on currency, tax, breakfast covered, and parking

(3) Time-Variant

- The time horizon for the data warehouse is significantly **longer** than that of operational systems
 - Operational database: current value data
 - Data warehouse data: provide information from a **historical** perspective (e.g., past 5-10 years)
- **Every key** structure in the data warehouse
 - Contains an element of **time**, explicitly or implicitly
 - But the key of operational data may or may not contain “time element”

(4) Nonvolatile

- Independence
 - A **physically separate store** of data transformed from the operational environment
- Static: **Operational update of data does NOT occur** in the data warehouse environment
 - Does not require transaction processing, recovery, and concurrency control mechanisms
 - Requires only two operations in data accessing:
 - **initial loading of data** and **access of data**

OLTP vs OLAP

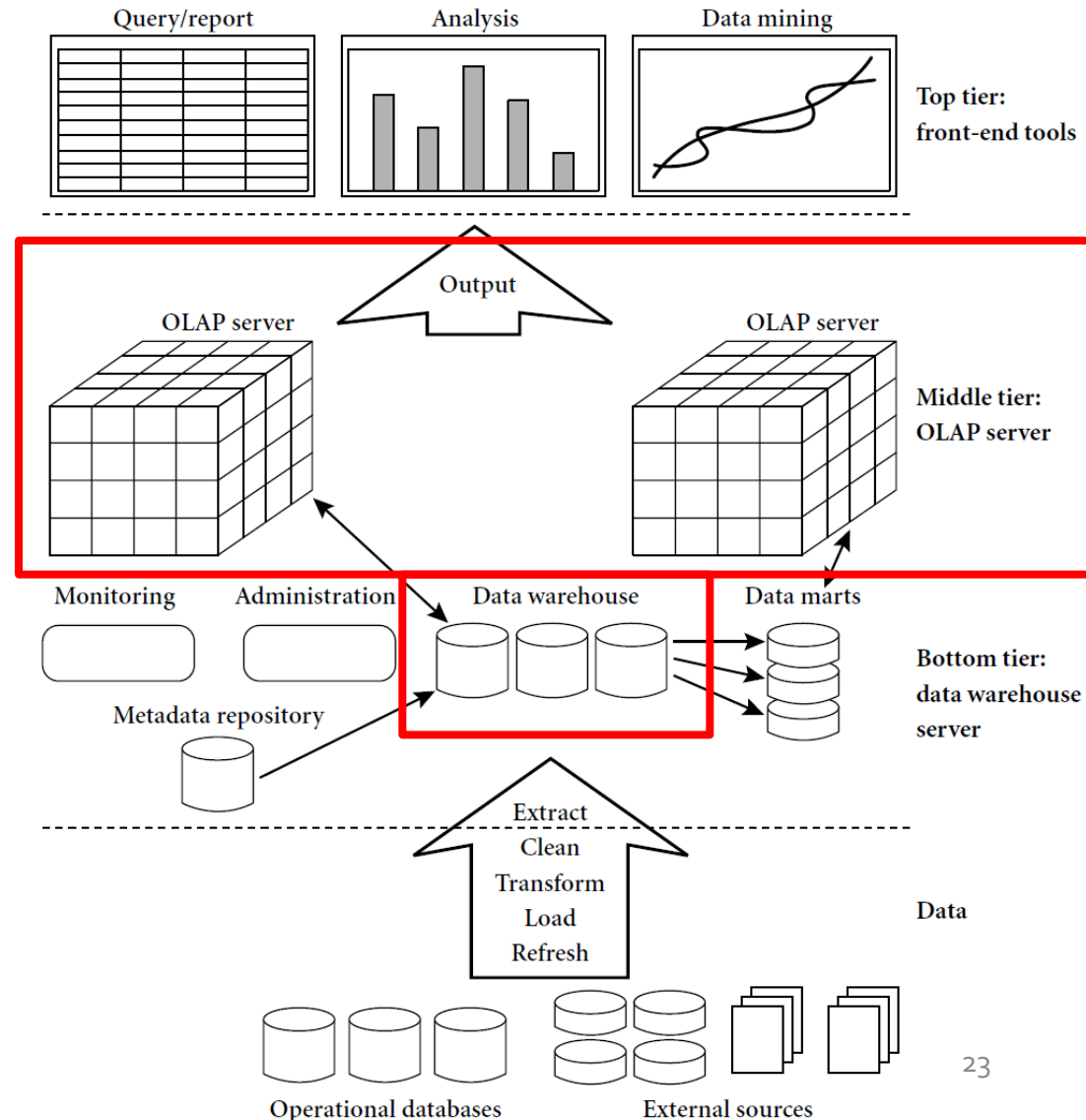
- OLTP: **Online** transactional processing
 - DBMS operations
 - Query and transactional processing
- OLAP: **Online** analytical processing
 - Data warehouse operations (drilling, slicing, dicing, etc.)
 - Data analysis to support decision making

OLTP vs OLAP

	OLTP	OLAP
users	clerk, IT professional	knowledge worker
function	day to day operations	decision support
DB design	application-oriented	subject-oriented
data	current, up-to-date detailed, flat relational isolated	historical, summarized, multidimensional integrated, consolidated
usage	repetitive	ad-hoc
access	read/write index/hash on prim. key	lots of scans
unit of work	short, simple transaction	complex query
# records accessed	tens	millions
#users	thousands	hundreds
DB size	100MB-GB	100GB-TB
metric	transaction throughput	query throughput, response

Data Warehouse: A Multi-Tiered Architecture

- Top Tier: Front-End Tools
- Middle Tier: OLAP Server
- Bottom Tier: Data Warehouse Server
- Data



From Data to Data Warehouse: Extraction, Transformation, and Loading (ETL)

- **Data extraction**
 - 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 Usage

- Three kinds of data warehouse applications
 - Information processing
 - supports querying, basic statistical analysis, and reporting using crosstabs, tables, charts and graphs
 - Analytical processing
 - multidimensional analysis of data warehouse data
 - supports basic OLAP operations, slice-dice, drilling, pivoting
 - Data mining
 - knowledge discovery from hidden patterns
 - supports associations, constructing analytical models, performing classification and prediction, and presenting the mining results using visualization tools

Efficient Processing OLAP Queries

- **Determine which operations** should be performed on the available cuboids
 - Transform drill, roll, etc. into corresponding SQL and/or OLAP operations, e.g., dice = selection + projection
- **Determine which materialized cuboid(s)** should be selected for OLAP op.
 - Let the query to be processed be on $\{brand, province_or_state\}$ with the condition “ $year = 2004$ ”, and there are 4 materialized cuboids available:
 - 1) $\{year, item_name, city\}$
 - 2) $\{year, brand, country\}$
 - 3) $\{year, brand, province_or_state\}$
 - 4) $\{item_name, province_or_state\}$ where $year = 2004$Which should be selected to process the query?

Efficient Processing OLAP Queries

- **Determine which operations** should be performed on the available cuboids
 - Transform drill, roll, etc. into corresponding SQL and/or OLAP operations, e.g., dice = selection + projection
 - **Determine which materialized cuboid(s)** should be selected for OLAP op.
 - Let the query to be processed be on $\{brand, province_or_state\}$ with the condition “ $year = 2004$ ”, and there are 4 materialized cuboids available:
 - 1) $\{year, item_name, city\}$
 - 2) $\{year, brand, country\}$
 - 3) $\{year, brand, province_or_state\}$ ✓
 - 4) $\{item_name, province_or_state\}$ where $year = 2004$
- Which should be selected to process the query?

References

- S. Agarwal, R. Agrawal, P. M. Deshpande, A. Gupta, J. F. Naughton, R. Ramakrishnan, and S. Sarawagi. On the computation of multidimensional aggregates. VLDB'96
- D. Agrawal, A. E. Abbadi, A. Singh, and T. Yurek. Efficient view maintenance in data warehouses. SIGMOD'97
- R. Agrawal, A. Gupta, and S. Sarawagi. Modeling multidimensional databases. ICDE'97
- **S. Chaudhuri and U. Dayal. An overview of data warehousing and OLAP technology. ACM SIGMOD Record, 26:65-74, 1997**
- J. Gray, et al. Data cube: A relational aggregation operator generalizing group-by, cross-tab and sub-totals. Data Mining and Knowledge Discovery, 1:29-54, 1997.
- A. Gupta and I. S. Mumick. Materialized Views: Techniques, Implementations, and Applications. MIT Press, 1999
- J. Han. Towards on-line analytical mining in large databases. *SIGMOD Record*, 1998
- V. Harinarayan, A. Rajaraman, and J. D. Ullman. Implementing data cubes efficiently. SIGMOD'96

References (cont.)

- C. Imhoff, N. Galemme, and J. G. Geiger. Mastering Data Warehouse Design: Relational and Dimensional Techniques. John Wiley, 2003
- W. H. Inmon. Building the Data Warehouse. John Wiley, 1996
- R. Kimball and M. Ross. The Data Warehouse Toolkit: The Complete Guide to Dimensional Modeling. 2ed. John Wiley, 2002
- P. O'Neil and D. Quass. Improved query performance with variant indexes. SIGMOD'97
- S. Sarawagi and M. Stonebraker. Efficient organization of large multidimensional arrays. ICDE'94
- P. Valduriez. Join indices. ACM Trans. Database Systems, 12:218-246, 1987.
- J. Widom. Research problems in data warehousing. CIKM'95.
- K. Wu, E. Otoo, and A. Shoshani, Optimal Bitmap Indices with Efficient Compression, ACM Trans. on Database Systems (TODS), 31(1), 2006, pp. 1-38.