Data Warehousing and OLAP Technology for Data Mining

- What is a data warehouse?
- A multi-dimensional data model
- Data warehouse architecture
- Data warehouse implementation
- Further development of data cube technology
- From data warehousing to data mining
What is 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.
- “A data warehouse is a subject-oriented, integrated, time-variant, and nonvolatile collection of data in support of management’s decision-making process.”—W. H. Inmon
- Data warehousing:
  - The process of constructing and using data warehouses

Data Warehouse—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.
Data Warehouse—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
    - E.g., Hotel price: currency, tax, breakfast covered, etc.
  - When data is moved to the warehouse, it is converted.

Data Warehouse—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”.
Data Warehouse—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, and concurrency control mechanisms
  - Requires only two operations in data accessing:
    - *initial loading of data* and *access of data*.

Data Warehouse vs. Heterogeneous DBMS

- Traditional heterogeneous DB integration:
  - Build *wrappers/mediators* on top of heterogeneous databases
  - *Query driven* approach
    - When a query is posed to 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**, high performance
  - Information from heterogeneous sources is integrated in advance and stored in warehouses for direct query and analysis
Data Warehouse vs. Operational DBMS

- **OLTP (on-line transaction processing)**
  - Major task of traditional relational DBMS
  - Day-to-day operations: purchasing, inventory, banking, manufacturing, payroll, registration, accounting, etc.

- **OLAP (on-line analytical processing)**
  - Major task of data warehouse system
  - Data analysis and decision making

- **Distinct features (OLTP vs. OLAP):**
  - User and system orientation: customer vs. market
  - Data contents: current, detailed vs. historical, consolidated
  - Database design: ER + application vs. star + subject
  - View: current, local vs. evolutionary, integrated
  - Access patterns: update vs. read-only but complex queries

### OLTP vs. OLAP

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

- Different functions and different data:
  - missing data: Decision support requires historical data which operational DBs do not typically maintain
  - data consolidation: DS requires consolidation (aggregation, summarization) of data from heterogeneous sources
  - data quality: different sources typically use inconsistent data representations, codes and formats which have to be reconciled

Data Warehousing and OLAP Technology for Data Mining

- What is a data warehouse?
- A multi-dimensional data model
- Data warehouse architecture
- Data warehouse implementation
- Further development of data cube technology
- From data warehousing to data mining
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.
- In data warehousing literature, an n-D base cube is called a base cuboid. The top most 0-D cuboid, which holds the highest-level of summarization, is called the apex cuboid. The lattice of cuboids forms a data cube.

Cube: A Lattice of Cuboids

0-D (apex) cuboid

1-D cuboids

2-D cuboids

3-D cuboids

4-D (base) cuboid
Conceptual Modeling of Data Warehouses

- Modeling data warehouses: dimensions & measures
  - **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 hierarchy is normalized into a set of smaller dimension tables, forming a shape similar to snowflake
  - **Fact constellations:** Multiple fact tables share dimension tables, viewed as a collection of stars, therefore called galaxy schema or fact constellation
A Data Mining Query Language: DMQL

- Cube Definition (Fact Table)
  
  define cube <cube_name> [<dimension_list>]:
  
  <measure_list>

- Dimension Definition (Dimension Table)
  
  define dimension <dimension_name> as
  
  <attribute_or_subdimension_list>

- Special Case (Shared Dimension Tables)
  
  – First time as “cube definition”
  
  – define dimension <dimension_name> as
  
  <dimension_name_first_time> in cube
  
  <cube_name_first_time>
Defining a Star Schema in DMQL

```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 (location_key, street, city, province_or_state, country)
```

CS490D 21

Defining a Snowflake Schema in DMQL

```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 (location_key, street, city(city_key, province_or_state, country))
```

CS490D 22
Defining a Fact Constellation in DMQL

```plaintext
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_type)
define dimension branch as (branch_key, branch_name, branch_type)
define dimension location as (location_key, street, city, province_or_state, country)
define cube shipping [time, item, shipper, from_location, to_location]:
    dollar_cost = sum(cost_in_dollars),
    unit_shipped = 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: Three Categories

- **distributive**: if the result derived by applying the function to \( n \) aggregate values is the same as that derived by applying the function on all the data without partitioning.
  - E.g., `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.
  - E.g., `avg()`, `min(N)`, `standard_deviation()`.
- **holistic**: if there is no constant bound on the storage size needed to describe a subaggregate.
  - E.g., `median()`, `mode()`, `rank()`.
A Concept Hierarchy: Dimension (location)

- all
  - region
    - Europe
    - North_America
  - country
    - Germany
    - Spain
    - Canada
    - Mexico
  - city
    - Frankfurt
    - Vancouver
    - Toronto
  - office
    - L. Chan
    - M. Wind

View of Warehouses and Hierarchies

Specification of hierarchies

- Schema hierarchy
  - day < {month < quarter; week} < year
- Set grouping hierarchy
  - {1..10} < inexpensive
Multidimensional Data

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

Dimensions: Product, Location, Time
Hierarchical summarization paths

A Sample Data Cube

Total annual sales of TVs in U.S.A.
Cuboids Corresponding to the Cube

- 0-D (apex) cuboid
- 1-D cuboids
- 2-D cuboids
- 3-D (base) cuboid

Browsing a Data Cube

- Visualization
- OLAP capabilities
- Interactive manipulation
Typical OLAP Operations

- **Roll up (drill-up):** summarize data
  - by climbing up 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):**
  - reorient the cube, visualization, 3D to series of 2D planes.
- **Other operations**
  - drill across: involving (across) more than one fact table
  - drill through: through the bottom level of the cube to its back-end relational tables (using SQL)

Data Warehousing and OLAP Technology for Data Mining

- What is a data warehouse?
- A multi-dimensional data model
- Data warehouse architecture
- **Data warehouse implementation**
- Further development of data cube technology
- From data warehousing to data mining
Efficient Data Cube Computation

• Data cube can be viewed as a lattice of cuboids
  – The bottom-most cuboid is the base cuboid
  – The top-most cuboid (apex) contains only one cell
  – How many cuboids in an n-dimensional cube with L levels?
    \[ T = \prod_{i=1}^{n} (L_i + 1) \]

• Materialization of data cube
  – Materialize every (cuboid) (full materialization), none (no materialization), or some (partial materialization)
  – Selection of which cuboids to materialize
    • Based on size, sharing, access frequency, etc.

Cube Operation

• Cube definition and computation in DMQL
  
  \textbf{define} \textit{cube} sales[item, city, year]: \textit{sum}(sales\_in\_dollars)
  
  \textbf{compute} \textit{cube} sales

  \textbf{Transform it into a SQL-like language (with a new operator \textit{cube by}, introduced by Gray et al.'96)}
  
  \textbf{SELECT} item, city, year, \textbf{SUM} (amount)
  \textbf{FROM} SALES
  \textbf{CUBE BY} item, city, year

• Need compute the following Group-Bys

  \textit{(date, product, customer)},
  \textit{(date, product), (date, customer)},
  \textit{(product, customer)},
  \textit{(date), (product), (customer)}

  ()
Cube Computation: ROLAP-Based Method

- Efficient cube computation methods
  - ROLAP-based cubing algorithms (Agarwal et al’96)
  - Array-based cubing algorithm (Zhao et al’97)
  - Bottom-up computation method (Beyer & Ramarkrishnan’99)
  - H-cubing technique (Han, Pei, Dong & Wang: SIGMOD’01)

- ROLAP-based cubing algorithms
  - Sorting, hashing, and grouping operations are applied to the dimension attributes in order to reorder and cluster related tuples
  - Grouping is performed on some sub-aggregates as a "partial grouping step"
  - Aggregates may be computed from previously computed aggregates, rather than from the base fact table

Multi-way Array Aggregation for Cube Computation

- Partition arrays into chunks (a small subcube which fits in memory).
- 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?
Multi-way Array Aggregation for Cube Computation

Multi-way Array Aggregation for Cube Computation
Multi-Way Array Aggregation for Cube Computation (Cont.)

- Method: the planes should be sorted and computed according to their size in ascending order.
  - See the details of Example 2.12 (pp. 75-78)
  - Idea: keep the smallest plane in the main memory, fetch and compute only one chunk at a time for the largest plane
- Limitation of the method: computing well only for a small number of dimensions
  - If there are a large number of dimensions, “bottom-up computation” and iceberg cube computation methods can be explored

Data Warehousing and OLAP Technology for Data Mining

- What is a data warehouse?
- A multi-dimensional data model
- Data warehouse architecture
- Data warehouse implementation
- Further development of data cube technology
- From data warehousing to data mining
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.
  • Differences among the three tasks

From On-Line Analytical Processing to On Line Analytical Mining (OLAM)

• Why online analytical mining?
  – High quality of data in data warehouses
    • DW contains integrated, consistent, cleaned data
  – Available information processing structure surrounding data warehouses
    • ODBC, OLEDB, Web accessing, service facilities, reporting and OLAP tools
  – OLAP-based exploratory data analysis
    • mining with drilling, dicing, pivoting, etc.
  – On-line selection of data mining functions
    • integration and swapping of multiple mining functions, algorithms, and tasks.
Discovery-Driven Exploration of Data Cubes

- Hypothesis-driven
  - exploration by user, huge search space

- Discovery-driven (Sarawagi, et al.’98)
  - Effective navigation of large OLAP data cubes
  - pre-compute measures indicating exceptions, guide user in the data analysis, at all levels of aggregation
  - Exception: significantly different from the value anticipated, based on a statistical model
  - Visual cues such as background color are used to reflect the degree of exception of each cell

Examples: Discovery-Driven Data Cubes

<table>
<thead>
<tr>
<th>Item</th>
<th>Avg sales</th>
<th>IBM home computer</th>
</tr>
</thead>
<tbody>
<tr>
<td>region</td>
<td>month</td>
<td></td>
</tr>
<tr>
<td>North</td>
<td>Jan 1%</td>
<td>-3%</td>
</tr>
<tr>
<td></td>
<td>Feb -1%</td>
<td>-1%</td>
</tr>
<tr>
<td></td>
<td>Mar 1%</td>
<td>0%</td>
</tr>
<tr>
<td></td>
<td>Apr 5%</td>
<td>0%</td>
</tr>
<tr>
<td></td>
<td>May 3%</td>
<td>0%</td>
</tr>
<tr>
<td></td>
<td>Jun 1%</td>
<td>0%</td>
</tr>
<tr>
<td></td>
<td>Jul 1%</td>
<td>0%</td>
</tr>
<tr>
<td></td>
<td>Aug 1%</td>
<td>0%</td>
</tr>
<tr>
<td></td>
<td>Sep 1%</td>
<td>0%</td>
</tr>
<tr>
<td></td>
<td>Oct 2%</td>
<td>2%</td>
</tr>
<tr>
<td></td>
<td>Nov 4%</td>
<td>4%</td>
</tr>
<tr>
<td></td>
<td>Dec 5%</td>
<td>5%</td>
</tr>
<tr>
<td>South</td>
<td>Jan 1%</td>
<td>-1%</td>
</tr>
<tr>
<td></td>
<td>Feb -1%</td>
<td>-1%</td>
</tr>
<tr>
<td></td>
<td>Mar 3%</td>
<td>0%</td>
</tr>
<tr>
<td></td>
<td>Apr 1%</td>
<td>0%</td>
</tr>
<tr>
<td></td>
<td>May 1%</td>
<td>0%</td>
</tr>
<tr>
<td></td>
<td>Jun 1%</td>
<td>0%</td>
</tr>
<tr>
<td></td>
<td>Jul 1%</td>
<td>0%</td>
</tr>
<tr>
<td></td>
<td>Aug 1%</td>
<td>0%</td>
</tr>
<tr>
<td></td>
<td>Sep 1%</td>
<td>0%</td>
</tr>
<tr>
<td></td>
<td>Oct 2%</td>
<td>2%</td>
</tr>
<tr>
<td></td>
<td>Nov 4%</td>
<td>4%</td>
</tr>
<tr>
<td></td>
<td>Dec 5%</td>
<td>5%</td>
</tr>
<tr>
<td>East</td>
<td>Jan 1%</td>
<td>-1%</td>
</tr>
<tr>
<td></td>
<td>Feb -2%</td>
<td>-2%</td>
</tr>
<tr>
<td></td>
<td>Mar -3%</td>
<td>-3%</td>
</tr>
<tr>
<td></td>
<td>Apr 1%</td>
<td>1%</td>
</tr>
<tr>
<td></td>
<td>May 1%</td>
<td>1%</td>
</tr>
<tr>
<td></td>
<td>Jun 1%</td>
<td>1%</td>
</tr>
<tr>
<td></td>
<td>Jul 1%</td>
<td>1%</td>
</tr>
<tr>
<td></td>
<td>Aug 1%</td>
<td>1%</td>
</tr>
<tr>
<td></td>
<td>Sep 1%</td>
<td>1%</td>
</tr>
<tr>
<td></td>
<td>Oct 2%</td>
<td>2%</td>
</tr>
<tr>
<td></td>
<td>Nov 4%</td>
<td>4%</td>
</tr>
<tr>
<td></td>
<td>Dec 5%</td>
<td>5%</td>
</tr>
<tr>
<td>West</td>
<td>Jan 4%</td>
<td>0%</td>
</tr>
<tr>
<td></td>
<td>Feb 4%</td>
<td>0%</td>
</tr>
<tr>
<td></td>
<td>Mar 4%</td>
<td>0%</td>
</tr>
<tr>
<td></td>
<td>Apr 4%</td>
<td>0%</td>
</tr>
<tr>
<td></td>
<td>May 4%</td>
<td>0%</td>
</tr>
<tr>
<td></td>
<td>Jun 4%</td>
<td>0%</td>
</tr>
<tr>
<td></td>
<td>Jul 4%</td>
<td>0%</td>
</tr>
<tr>
<td></td>
<td>Aug 4%</td>
<td>0%</td>
</tr>
<tr>
<td></td>
<td>Sep 4%</td>
<td>0%</td>
</tr>
<tr>
<td></td>
<td>Oct 4%</td>
<td>0%</td>
</tr>
<tr>
<td></td>
<td>Nov 4%</td>
<td>0%</td>
</tr>
<tr>
<td></td>
<td>Dec 4%</td>
<td>0%</td>
</tr>
</tbody>
</table>
Summary

- **Data warehouse**
- A **multi-dimensional model** of a data warehouse
  - Star schema, snowflake schema, fact constellations
  - A data cube consists of dimensions & measures
- **OLAP operations:** drilling, rolling, slicing, dicing and pivoting
- **OLAP servers:** ROLAP, MOLAP, HOLAP
- **Efficient computation of data cubes**
  - Partial vs. full vs. no materialization
  - Multiway array aggregation
  - Bitmap index and join index implementations
- **Further development of data cube technology**
  - Discovery-drive and multi-feature cubes
  - From OLAP to OLAM (on-line analytical mining)

References (I)

- 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
- G. Dong, J. Han, J. Lam, J. Pei, K. Wang. Mining Multi-dimensional Constrained Gradients in Data Cubes. VLDB’2001
References (II)

- J. Han, J. Pei, G. Dong, K. Wang. Efficient Computation of Iceberg Cubes With Complex Measures. SIGMOD’01
- V. Harinarayan, A. Rajaraman, and J. D. Ullman. Implementing data cubes efficiently. SIGMOD’96
- Y. Zhao, P. M. Deshpande, and J. F. Naughton. An array-based algorithm for simultaneous multidimensional aggregates. SIGMOD’97