Data Warehousing and Data Mining

Lecture 2 Overview to Data Warehousing and OLAP Technology

Acknowledgement: The Lecture Slides are adapted from the original slides of Han’s textbook.
Lecture Outline

• What is a data warehouse?
• Difference between Operational & Transactional Database Systems
• A multi-dimensional data model
• Data warehouse architecture
• Data warehouse implementation
What is 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.
Data Warehouse – Subject Oriented

- Organized around major subjects, such as customer, supplier, product, sales, time.
- 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 making 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
• 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:
    
    • *initial loading of data* and *access of data*. 
Lecture Outline

• What is a data warehouse?

• **Difference between Operational & Transactional Database Systems**

• A multi-dimensional data model

• Data warehouse architecture

• Data warehouse implementation
Data Warehouse(OLAP) vs. Operational DBMS(OLTP)

- **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
  - Can organize and present data in various forms and combinations
## Comparison of OLTP and OLAP

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

*Note: Table is partially based on Chaudhuri and Dayal [CD97].*
A three-tier data warehousing architecture
Data Warehouse Models

- **Enterprise warehouse:**
  - An enterprise warehouse collects all of the information about subjects spanning the entire organization.
  - Contains both detailed and summerised data

- **Data Mart**
  - A data mart contains a subset of corporate-wide data that is of value to a specific group of users.
  - The scope is confined to specific selected subjects, e.g. marketing data mart (customer, item, and sales)
  - Summerised (sometimes due to privacy concerns)

- **Virtual warehouse:**
  - A virtual warehouse is a set of views over operational databases.
  - For efficient query processing, only some of the possible summary views may be materialized.
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
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.
Lecture Outline

• What is a data warehouse?
• Difference between Operational & Transactional Database Systems
  • A multi-dimensional data model
• Data warehouse architecture
• Data warehouse implementation
• **Data Cube:** (base cube, apex cube, concept of hierarchies)

• **Schemas:** (Star, Snowflakes, Fact Constellations)

• **OLAP Operations:** (Roll up, Drill down, Slice & Dice, Pivot)
**Multi-dimensional view of data**

### Table 4.2 2-D View of Sales Data for *AllElectronics* According to *time* and *item*

<table>
<thead>
<tr>
<th>location = &quot;Vancouver&quot;</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>item (type)</strong></td>
</tr>
<tr>
<td><strong>time (quarter)</strong></td>
</tr>
<tr>
<td>Q1</td>
</tr>
<tr>
<td>Q2</td>
</tr>
<tr>
<td>Q3</td>
</tr>
<tr>
<td>Q4</td>
</tr>
</tbody>
</table>

*Note: The sales are from branches located in the city of Vancouver. The measure displayed is *dollars_sold* (in thousands).*

### Table 4.3 3-D View of Sales Data for *AllElectronics* According to *time*, *item*, and *location*

<table>
<thead>
<tr>
<th>location = &quot;Chicago&quot;</th>
<th>location = &quot;New York&quot;</th>
<th>location = &quot;Toronto&quot;</th>
<th>location = &quot;Vancouver&quot;</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>item</strong></td>
<td><strong>item</strong></td>
<td><strong>item</strong></td>
<td><strong>item</strong></td>
</tr>
<tr>
<td><strong>time</strong></td>
<td>home</td>
<td>comp.</td>
<td>phone</td>
</tr>
<tr>
<td>Q1</td>
<td>854</td>
<td>882</td>
<td>89</td>
</tr>
<tr>
<td>Q2</td>
<td>943</td>
<td>890</td>
<td>64</td>
</tr>
<tr>
<td>Q3</td>
<td>1032</td>
<td>924</td>
<td>59</td>
</tr>
<tr>
<td>Q4</td>
<td>1129</td>
<td>992</td>
<td>63</td>
</tr>
</tbody>
</table>

*Note: The measure displayed is *dollars_sold* (in thousands).*
• A data warehouse is based on a **multidimensional data model** which views data in the form of a data cube.

• A data cube, is organised around a central theme, 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) or **location** (branch, city, state, country).
  
  – Fact table contains measures of central theme (such as **dollars_sold**, **units sold**) and keys to each of the related dimension tables.
Data Cube is a metaphor don’t confines data to 3-D.
Multidimensional Data

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

Dimensions: Product, Location, Time
Hierarchical summarization paths

Industry | Region | Year
---|---|---
Category | Country | Quarter
Product | City | Month
Office | Day | Week

Month

Product

Region
Sample Data Cube

Product: TV, VCR, PC
Date: 1Qtr, 2Qtr, 3Qtr, 4Qtr, sum
Country: U.S.A, Canada, Mexico, sum

All, All, All
Schema

- **Star Schema**
  - A fact table in the middle connected to a set of dimension tables

- **Snowflake Schema**
  - Some dimensional hierarchy is normalized into a set of smaller dimension tables, forming a shape similar to snowflake.
  - Reduces redundancy, however at the cost of effectiveness of browsing.

- **Galaxy schema (Fact Constellation)**
  - Multiple fact tables share dimension tables.
  - Viewed as a collection of stars - Galaxy schema.
Star Schema

- **time**
  - time_key
  - day
  - day_of_the_week
  - month
  - quarter
  - year

- **branch**
  - branch_key
  - branch_name
  - branch_type

- **item**
  - item_key
  - item_name
  - brand
  - type
  - supplier_type

- **location**
  - location_key
  - street
  - city
  - state_or_province
  - country

- **Sales Fact Table**
  - time_key
  - item_key
  - branch_key
  - location_key
  - units_sold
  - dollars_sold
  - avg_sales

- **Measures**
Snowflake Schema

- **time**
  - time_key
  - day
  - day_of_the_week
  - month
  - quarter
  - year

- **branch**
  - branch_key
  - branch_name
  - branch_type

- **Sales Fact Table**
  - time_key
  - item_key
  - branch_key
  - location_key
  - units_sold
  - dollars_sold
  - avg_sales

- **item**
  - item_key
  - item_name
  - brand
  - type
  - supplier_key

- **supplier**
  - supplier_key
  - supplier_type

- **location**
  - location_key
  - street
  - city_key

- **city**
  - city_key
  - city
  - state_or_province
  - country
Fact Constellation – Galaxy Schema

**Sales Fact Table**
- **time_key**
- **item_key**
- **branch_key**
- **location_key**
- **units_sold**
- **dollars_sold**
- **avg_sales**

**Measures**
- **item_key**
- **item_name**
- **brand**
- **type**
- **supplier_type**

**Shipping Fact Table**
- **time_key**
- **item_key**
- **shipper_key**
- **from_location**
- **to_location**
- **dollars_cost**
- **units_shipped**

**Branch**
- **branch_key**
- **branch_name**
- **branch_type**

**Location**
- **location_key**
- **street**
- **city**
- **province_or_state**
- **country**

**Shipper**
- **shipper_key**
- **shipper_name**
- **location_key**
- **shipper_type**
Cube Definition Syntax (BNF) in 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>
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)
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))
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
The Role of Concept Hierarchies

- **Concept Hierarchy**
  - Defines a sequence of mappings from a set of low-level concepts to high-level, more general concepts.

- **Schema Hierarchy**
  - A concept hierarchy that is a total or partial order among attributes in a database schema
    - Total order: street < city < province_or_state < country
    - Partial order: day < {month < quarter; week} < year

- **Set-grouping Hierarchy**
  - Defined by discretizing or grouping values for a given dimension or attribute.
Hierarchical and lattice structures of attributes in warehouse dimensions: (a) a hierarchy for location and (b) a lattice for time.
Measures

- A data cube *measure* is a numeric function that can be evaluated at each point in the data cube space.
- A measure value is computed for a given point by aggregating the data corresponding to the respective dimension–value pairs defining the given point.
Types of Measures

- **Types of measures**
  - **Distributive:**
    - An aggregate function is distributive if it can be computed in a distributed manner by applying the same function on partitioned sets.
    - `count()`, `min()`, and `max()` are distributive aggregate functions.
  - **Algebraic:**
    - An aggregate function is algebraic if it can be computed by an algebraic function with $M$ arguments (where $M$ is a bounded positive integer), each of which is obtained by applying a **distributive** aggregate function.
    - `avg()` (average) can be computed by `sum()/count()`, where both `sum()` and `count()` are distributive
    - `standard_deviation()`.
  - **Holistic:**
    - `median()`, `mode()`, and `rank()`.
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)
Example of OLAP Operations
A Starnet Model of Business Queries
Each circle is called a footprint
Lecture Outline

• What is a data warehouse?
• Difference between Operational & Transactional Database Systems
• A multi-dimensional data model
• Data warehouse architecture
• Data warehouse implementation
Data Warehouse – Design Process

• **Top-down, bottom-up** approaches or a combination of both
  – **Top-down**: Starts with overall design and planning (mature)
  – **Bottom-up**: Starts with experiments and prototypes (rapid)

• **From software engineering point of view**
  – **Waterfall**: structured and systematic analysis at each step before proceeding to the next
  – **Spiral**: rapid generation of increasingly functional systems, short turn around time, quick turn around
• Chose a **business process** to model
  – E.g. orders, invoices, shipments, sales …

• **Choose the grain (atomic level of data) of the business process**
  – E.g. individual transactions, individual daily snapshots

• **Choose the dimensions that will apply to each fact table record**
  – Typical dimensions are time, item, customer, supplier, warehouse, transaction type and status

• **Choose the measure that will populate each fact table record**
  – Typical measures are numeric additive quantities like dollars_sold and units_sold
Lecture Outline

- What is a data warehouse?
- Difference between Operational & Transactional Database Systems
- A multi-dimensional data model
- Data warehouse architecture
- Data warehouse implementation
What is the total number of cuboids?

- A data cube is a lattice of cuboids. Suppose that you want to create a data cube for AllElectronics sales that contains the following: *city*, *item*, *year*, and *sales in dollars*.

- Possible queries such as the following:
  - “Compute the sum of sales, grouping by city and item.”
  - “Compute the sum of sales, grouping by city.”
  - “Compute the sum of sales, grouping by item.”

Curse of dimensionality!
• 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 DMQL

- Cube definition and computation in DMQL
  
  ```
  define cube sales[item, city, year]:
    sum(sales_in_dollars)
  
  compute cube sales
  ```

- Transform it into a SQL-like language (with a new operator `cube by`, introduced by Gray et al.’96)
  
  ```
  SELECT item, city, year, SUM (amount)
  FROM SALES
  CUBE BY item, city, year
  ```
In the AllElectronics data warehouse,

- dimension item has four values (representing item types): “home entertainment (H),” “computer (C),” “phone (P),” and “security (S).”

- Suppose that the cube is stored as a relation table with 100,000 rows. Because the domain of item consists of four values, the bitmap index table requires four bit vectors (or lists) for each record. We have a total 100,000 vectors.

<table>
<thead>
<tr>
<th>Base table</th>
<th>item bitmap index table</th>
<th>city bitmap index table</th>
</tr>
</thead>
<tbody>
<tr>
<td>RID</td>
<td>item</td>
<td>city</td>
</tr>
<tr>
<td>R1</td>
<td>H</td>
<td>V</td>
</tr>
<tr>
<td>R2</td>
<td>C</td>
<td>V</td>
</tr>
<tr>
<td>R3</td>
<td>P</td>
<td>V</td>
</tr>
<tr>
<td>R4</td>
<td>S</td>
<td>V</td>
</tr>
<tr>
<td>R5</td>
<td>H</td>
<td>T</td>
</tr>
<tr>
<td>R6</td>
<td>C</td>
<td>T</td>
</tr>
<tr>
<td>R7</td>
<td>P</td>
<td>T</td>
</tr>
<tr>
<td>R8</td>
<td>S</td>
<td>T</td>
</tr>
</tbody>
</table>

Another Example

<table>
<thead>
<tr>
<th>Cust</th>
<th>Region</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>C1</td>
<td>Asia</td>
<td>Retail</td>
</tr>
<tr>
<td>C2</td>
<td>Europe</td>
<td>Dealer</td>
</tr>
<tr>
<td>C3</td>
<td>Asia</td>
<td>Dealer</td>
</tr>
<tr>
<td>C4</td>
<td>America</td>
<td>Retail</td>
</tr>
<tr>
<td>C5</td>
<td>Europe</td>
<td>Dealer</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>RecID</th>
<th>Asia</th>
<th>Europe</th>
<th>America</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>RecID</th>
<th>Retail</th>
<th>Dealer</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>
Indexing on OLAP Data - Join indexing
Given four materialized cuboids, the query to be processed is on \{\text{brand, province or state}\}, with the selection constant “year = 2010.”

- Cuboid 1: \{\text{year, item name, city}\}
- Cuboid 2: \{\text{year, brand, country}\}
- Cuboid 3: \{\text{year, brand, province or state}\}
- Cuboid 4: \{\text{item name, province or state}\}, where year = 2010

Which one to choose?
- Cuboids 1, 3, and 4 can be used to process the query because
  - they have the same set or a superset of the dimensions in the query,
  - the selection clause in the query can imply the selection in the cuboid, and
  - the abstraction levels for the item and location dimensions in these cuboids are at a finer level than brand and province or state, respectively.
Summary

- Understand data warehouse from the angle of analytical processing requirements.
- Differences between OLTP and OLAP
- Star, Snowflake and Galaxy Schema
- Concept Hierarchies
- Structure: Lattice of data cuboids, schemas, concept hierarchy
- Roll-up, Drill-down, Slice & Dice, Pivot
- 3-tier architecture
- Efficient implementation
Sample Questions

• What is a data cube? Explain the OLAP operations roll up and drill down in relation to a data cube.
• Explain the concept of a data warehouse and the main steps required for constructing a data warehouse.
• Explain the meaning of star schema and snowflake schema in relation to a data warehouse.
Suppose that a data warehouse consists of three dimensions time, doctor and patient, and two measures count (the number of patients examined) and charge (fee that a doctor charges a patient for a visit).

- Draw either a star or a snowflake schema for the above data warehouse.
- Starting with the base cuboid [day, doctor, patient], what specific OLAP operations should be performed in order to list the total fee collected by each doctor in 2010?
- Starting with the base cuboid [day, doctor, patient], what specific OLAP operations should be performed in order to list the total fee paid by patient John Citizen in the years 2009 and 2010 combined?