Data warehousing

S. Chou, NTU IIM, Spr03

- Overview
- Data warehouse basics
  - Multidimensional model and schemas
  - OLAP operations
- DW design process and examples
- System and products

References:
- Elmasri 26, DM book, and MS SQL server’s OLAP service
Sample Queries (actual)

- “How many customers have complained more than once for our new software”
- How many customers had to call the customer contact center more than once for the same problem”
- “How many people have claimed loss luggage on our airline”
- "Which products in my store are selling most quickly?"
- "Which products stay in the inventory the longest?"
- "How does one store's profitability compare to the rest of the chain?"
- "How many VISA card holders from our bank did not use their VISA card last year". The answer was: 500,000. This meant that the bank paid $12 million dollars to VISA for nothing
- "Which doctors in California charge more than the national average for a broken leg procedure?"
- "Which doctors filed the largest number of claims during the last quarter?"
- "How many purchase orders were placed in 1991, 1992, and 1993 for the Sector, and what were the corresponding dollars".
- "What are the top 10 purchased commodities and corresponding suppliers in the Sector"
Can these queries be satisfied by a common corporate data?

What type of technologies will make it possible?
Data Warehouse

Data Warehouse characteristics:
- Contains summary (light and heavy) and historical data
- Data may be from operational systems, external systems, unstructured data (memos, letters, pictures)
- Information repository Intended for decision support (primarily retrieval, discovery) and not by applications (order processing, purchasing and sales)
- Data is subject-oriented (customers, products)
- Support for tools:
  - Report writers
  - Data browsers
  - Spreadsheets
  - 4GL (e.g., focus)
  - “Drill down” applications
  - Data mining tools
  - Other planning and modeling tools
Why Data Warehouses for Integration

- Data of legacy apps can be extracted and loaded in a DW instead of re-eng the legacy app
- DW avoids interference with operations (queries may interfere with the day-to-day operation of the on-line transaction processing applications).
- DW can be used for data spread across multiple systems stored in diverse database managers requiring diverse access methods (may require expensive mediators).
- DW can be used for legacy data that is embedded in IMS databases and flat files that do not support adhoc queries
- DW can be used for more indexes for extensive queries
- And others
Data Warehouse: To do or not to do

Data warehouse is, in general, a good approach if:

- Demand for ad hoc queries and analysis is very high
- The needed data is used for decision support (it is easier to provide decision support through a data warehouse)
- The surround technology for access in place is not efficient and does not meet the requirements of new users
- The data does not change frequently (in such cases, data warehouse needs to be synchronized with the back-end system frequently)
- Needed data is embedded in too many legacy systems (it is difficult to directly access 30 or so data sources and perform joins among them, etc.)

Data warehouses are not a good choice if:

- Demand for data is low (access in place is better in this case)
- Most recent copy of data is essential (DWs give somewhat outdated view)
- Data changes frequently in the sources
**DW and Data Mining**

- **What is data mining**: Finding and discovering trends and patterns in data.
- Also known as knowledge discovery KDD (knowledge discovery in databases).
- Emphasis is on hidden patterns and relationships.
- Heavy use of AI and statistical analysis (more AI).
- Data to be mined is extracted and loaded in a mining file.
- **Examples:**
  - What products the customer is likely to buy (based on current purchases).
  - Which customers are likely to discontinue services.
  - What variables determine the customers who will go to a competitor.
  - Who is most likely to respond to mail.
  - How to detect fraudulent behaviour in credit card users.
  - Which emarkets are likely to succeed.
Typical DW applications

- Reposition products and manage product portfolios – compare the performance of sales by qtr, year, region
- Increase customer focus – analyze customer buying patterns such as buying preference, buying time, buying cycle
- Analyze operations and look for sources of profit
- Manage customer relationships
零售 – 該賣甚麼產品或是哪幾組產品？在哪裡賣？甚麼價錢？特定產品該佔多少貨架空間？每個產品的促銷比例為何？哪些產品的搭售效果較好？要庫存多少貨？

財務金融 – 依據分行別列出前一百名利潤最高的顧客，並說明他們對營收的貢獻。多少比例來自手續費？利息收入？預支費用？哪些是我們該採取直效行銷的對象？一組特定金融商品的合理價格為何？承保與授信人員作業的績效如何？那些顧客群組的風險較高？詐領事件有多少？

電信 – 針對上個月更換電話公司的所有顧客，列出他們在使用本公司的服務時，平均的通話數量與費用是多少？這些人在離開前的最後三個月內，前述項目的統計值又為何？

醫療保健 – 某一種特定醫療的成效範圍為何？這項治療多久開一次處方？哪一種藥、哪家醫院、哪個醫生、哪種保健計劃最有效率？目前那種病患群的風險最高？針對某項疾病，某一特定技術它的效率與效益如何？
資料倉儲是將不同部門或是企業資料庫裡的歷史資料萃取出來，彙整後以電子形式儲存。資料倉儲背後的想法其實很簡單：把公司所有的資料擺在一起，好讓使用者可以一次看得更多、學得更多、同時使組織運作得更好。

資料倉儲的目標是要幫助使用者找出趨勢、尋求企業問題的解答、以及從歷史與作業資料中找出合理的意義；所有這些都是在加強企業的決策支援。

資料倉儲通常會將生產性資料如顧客交易，進行清理、組織、然後擺在適當的地方，以利瀏覽、分析、及決策。在實務上，資料倉儲通常會將多個資料來源加以集中，或是延伸集中儲存的資料價值。這些儲存的資料有很多用途，例如可以加以挖掘、轉換、分析、或是以視覺化的方式呈現。它的成果是：非常實用的關鍵資料存取與決策支援。如果您知道企業每次要寫新的決策支援系統時，就得要到公司外面去尋找資料，就會發現資料倉儲的運作真的很了不起。
What is a data warehouse

“a subject-oriented, integrated, nonvolatile, time-variant collection of data in support of management’s decisions”

- Subject-oriented: organized around major subjects such as customer, product, sales
- Integrated: data from heterogeneous sources (db, flat files, …)
- Time-variant: historical perspective (5-10 years)
- Nonvolatile: a separate store of data transformed from the application data found in the operational environment; mostly read operations with incremental update (append)

“a collection of decision support technologies, aimed at enabling the knowledge worker to make better and faster decisions”
The overall process of data warehousing

- Databases
- Back flushing
- Cleaning
- Reformattting
- Other data input
- Updates/new data
- Data warehouse
  - Metadata
  - OLAP
  - DSS/EIS
  - Data mining
- Data mining
Meta Data

- serves as a directory and a catalog of the data warehouse
- Meta data can provide:
  - Data schema. Meta data shows the names, the attributes, the keys, and the formats of warehouse tables.
  - Semantic model. Business objects and their relationships
  - Mapping of operational data to warehouse data.
  - Common routines for summarization and access of data
  - Predefined queries, reports, and spreadsheets.
  - Extract history.
  - Information about external and unstructured data.
  - Relationship to other meta data stores.
  - Data location (what tables are located where in the network).
Data Extraction/Transformation/Load (ETL)

Data Selection /extraction
(done at source machines)

Data Transformation
(can be done at source and/or target and/or in between)

Data Load/Update
(done at target machine)

Many ETL Tools

www.datawarehousingonline.com/DWETLV.htm
Typical architecture of a data warehouse for AllElectronics.
DW characteristics

- distinct from traditional transaction-oriented databases in their structure, functioning, performance, and purpose.
- structure – multidimensional (see DW data model)
- functioning
  - provide access to data for complex analysis, knowledge discovery/data mining, and decision making
  - OLAP (online analytical processing) -- analysis of complex data from the data warehouse by skilled knowledge worker
- performance
  - very rapid access to a large volume of data
  - optimized for data retrieval
- purpose – support an organization’s leading decision makers with high-level data for complex and important decisions (DSS/EIS)
<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>historical; 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</td>
<td>tens</td>
<td>millions</td>
</tr>
<tr>
<td>accessed</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Number of users</td>
<td>thousands</td>
<td>hundreds</td>
</tr>
<tr>
<td>DB size</td>
<td>100 MB to GB</td>
<td>100 GB to 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 [CD97].
View data in the form of data cube, defined by dimensions and facts

Dimensions are perspectives such as time, item, branch, location

Facts are numerical measures – the quantities by which we want to analyze relationships between dimensions

The data cube is typically organized around a central theme, like sales
Fact table:
- Large dominant table in the middle
- Only table with multiple joins to other tables
- Can be designed at low or high granularity (for each product, daily activity, etc)
  - Can be very large (heavily indexed)
  - Stores numerical measurements of business
  - Facts should be numeric, continuously valued, and additive
  - Can be sparse (not every product is sold everyday)

Dimension tables
- Contain textual (descriptive) info
- Attached to fact table through a single join
- Should mostly consist of descriptions:
  - characters
  - numeric values (e.g., prices, sizes) that are used as descriptions
- Attributes used for column headings
Multidimensional databases capture and present data as arrays that can be arranged in multiple dimensions,

Multidimensional databases present large amounts of data to users in a manner that is easily comprehensible.

Many EISs such as Pilot Software's Lightship and Comshare's Commander use multi-dimensional databases.

These databases can be used to answer queries that would be extremely difficult cumbersome in SQL, such as

"List the top five sales regions based on the percentage increase in revenues this year relative to last year".

Use of RDBMS technology at the core of a multidimensional database (e.g., Microstrategy DSS Agent)

Limitation of multidimensional databases: cannot store large data amounts (typically more than 10 GB of data)
<table>
<thead>
<tr>
<th>time (quarters)</th>
<th>Chicago</th>
<th>New York</th>
<th>Toronto</th>
<th>Vancouver</th>
</tr>
</thead>
<tbody>
<tr>
<td>Q1</td>
<td>605</td>
<td>825</td>
<td>14</td>
<td>400</td>
</tr>
<tr>
<td>Q2</td>
<td>680</td>
<td>952</td>
<td>31</td>
<td>512</td>
</tr>
<tr>
<td>Q3</td>
<td>812</td>
<td>1023</td>
<td>30</td>
<td>501</td>
</tr>
<tr>
<td>Q4</td>
<td>927</td>
<td>1038</td>
<td>38</td>
<td>580</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>item (types)</th>
<th>computer</th>
<th>security</th>
</tr>
</thead>
<tbody>
<tr>
<td>home</td>
<td></td>
<td></td>
</tr>
<tr>
<td>phone</td>
<td></td>
<td></td>
</tr>
<tr>
<td>entertainment</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Multidimensional schemas

- Star schema – a large central table (facts) with a set of smaller dimension tables, one for each dimension
- Snowflake schema – a variant of the star schema model, where some dimension tables are normalized
- Galaxy schema or fact constellation – multiple fact tables to share dimension tables or viewed as a collection of stars
define cube sales [time, item, branch, location]:
    dollars_sold = sum(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]:
    dollars_cost = sum(cost_in_dollars), units_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
OLAP operations

- In the multidimensional model, data are organized into multiple dimensions.
- Each dimension contains multiple levels of abstraction defined by concept hierarchies (later).
- This organization provides users with the flexibility to view data from different perspectives.
- OLAP operations make use of background knowledge regarding the domain of the data being studied in order to allow the presentation of data at different levels of abstraction. Such operations accommodate different user viewpoints, allowing interactive querying and analysis of the data at hand.
- Typical OLAP operators are slice and dice, roll up, and drill down (examples later).
Concept hierarchy

- all
  - Canada
    - British Columbia
      - Vancouver
    - Ontario
      - Toronto
  - USA
    - New York
      - New York (repeated)
    - Illinois
      - Buffalo
      - Chicago
dice for
(location = "Toronto" or "Vancouver")
and (time = "Q1" or "Q2") and
(item = "home entertainment" or "computer")

roll-up on location
(from cities to countries)

slice for time = "Q1"

drill-down on time
(from quarters)
Category of measures

- **Distributive** – an aggregate function that can be computed in a distributed manner
  - e.g., sum(), min(), max(), count()
  - Aggr of aggr on partitions = aggr on the whole

- **Algebraic** – algebraic computation on distributive aggregate functions
  - E.g., avg() = sum()/count(), std_deviation()

- **Holistic** – taking as a whole
  - E.g., rank(), median()
Example applications

- Sales by region by qtr by product category for the past three years
- Drill down into product subcategories and even brands
- Drill up from qty to year

<table>
<thead>
<tr>
<th>x</th>
<th>y</th>
<th>z</th>
</tr>
</thead>
<tbody>
<tr>
<td>NE</td>
<td>NW</td>
<td>SE</td>
</tr>
</tbody>
</table>

Sales for product x, y, and z

y for different regions

Sales for y for NW region
Designing a dimensional database

Choose a business process to model
- processes are supported by systems from which data can be collected for the purpose of data warehouse
- examples processes are orders, invoices, shipments, sales, etc

- choose the *grain* of the business process
  - the grain is the fundamental, atomic level of data to be represented in the fact table
  - typical grains are individual transactions, daily or monthly snapshots

- choose the dimensions for each fact table record
  - typical dims are time, product, customer, promotion,
  - for each dim, describe all discrete attributes that fill out each dim table

- choose the measured facts
  - to populate each fact table record
  - examples measured facts are qty sold, dollars sold, and so on
An example

- Objective
  - to analyze the sales data for the upper management
  - typical DW OLAP applications such as trend analysis, explanation, etc.

- original data
  - transaction id, store id, date, product id, qty, amount

- grain size consideration
  - merged daily sales
  - transaction data from each store is collected and processed daily
Dimensional model

- Store dimension
  - derived from store id
  - store key, city, region, state, store type, size, etc.

- Time dimension
  - derived from data
  - time key, day of week, month, season, fiscal period, holiday flag, etc.

- Product dimension
  - derived from product id
  - product key, brand, category, subcategory, pkg type, pkg size, etc.
Data modification

- Data discarded
  - transaction id

- Data cleansing
  - such as store id, product id, exceptional qty or amount

- Data conversion
  - from original (transaction id, store id, date, product id, qty, amount)
  - after merged and pre-processing
  - to sales fact to be populated into the sales fact table (store key, product key, time key, daily qty, daily amount)
  - use of store, product, & time keys would reduce a lot of space needed for the sales fact table
Star scheme

Product dim
product key etc.

Time dim
time key etc.

Sales fact
product key
time key
store key
daily qty
daily amount

Store dim
store key etc.

Typical Time dim table
T03001 2 3 1 … (Tuesday, March, first qtr, ... T03002 3 3 1 … (Wednesday, March, first qtr, ...
A three-tier data warehousing architecture.
Three types of DW

- enterprise-wide data warehouses are huge projects requiring massive investment of time and resources
- virtual data warehouses provide views of operational databases that are materialized for efficient access
- data marts generally are targeted to a subset of the organization, such as a department, and are more tightly focused
Plusses:
- same consistent and complete data for all users.
- users logon to one environment (no multiple data sources)
- Works well where most of the processing is done at the corporate

Minusses:
- very difficult to develop a global data model for most organizations
- difficult to agree on a corporate wide level of detail and naming conventions.
- need to carefully manage the performance and end-user access
Local Functional Warehouses (Data Marts)

- Typically created by departments/divisions to support their own decisions
- May be created to support specific products (e.g., automobile parts) or function (e.g., loan management)
- May be created for user populations/environments (DW for PC users)
- May be fed by a centralized DW
- **Plus**: can be developed quickly to serve the local needs without having to wait for the large corporate data warehouse.
- **Minus**: proliferation of DWs that are not consistent with each other.
- **In practice**, data marts can be used by independent departments as a starting point in an overall strategy for a centralized corporate data warehouse.
"Data warehouse mediator" contains a global data dictionary
Mediator can send the requests directly to the operational databases
"Virtual" data warehouse (VDW) routes the user queries to the data sources (essentially a read-only distributed data management technology)
VDW intelligence to automatically migrate data to the data warehouse (grow as you go)
**Plusses:**
- Flexibility, performance, scaling, and load balancing.
- Can send local queries to regional and corporate queries corporate DW
- Can support heterogeneity (one data mart may use a specialized DBMS suitable)
**Minusses:**
- Global data model may still be needed
- For widely distributed data, performance degradation & service outages can occur
**System and products**

- **OLAP servers**
  - Relational OLAP (ROLAP)
  - Multidimensional OLAP (MOLAP)
  - Hybrid OLAP (HOLAP), e.g., MS SQL server supports a HOLAP server

- DW products include MS OLAP server, NCR’s, Powerplay …

- **System issues**
  - Some degree of pre-computation of multidimensional aggregates
  - Indexing
Indexing

- Bitmap indexing
  - for low cardinality domain
  - one bit vector for each domain value
  - good comparison, aggregation, and join performance

<table>
<thead>
<tr>
<th>RID</th>
<th>item</th>
<th>city</th>
</tr>
</thead>
<tbody>
<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>

<table>
<thead>
<tr>
<th>RID</th>
<th>H</th>
<th>C</th>
<th>P</th>
<th>S</th>
</tr>
</thead>
<tbody>
<tr>
<td>R1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>R2</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>R3</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>R4</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>R5</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>R6</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>R7</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>R8</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>RID</th>
<th>V</th>
<th>T</th>
</tr>
</thead>
<tbody>
<tr>
<td>R1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>R2</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>R3</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>R4</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>R5</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>R6</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>R7</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>R8</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

Join indexing

- an indexing scheme
- relate the values of a dimension to rows in the fact table
- For example, if there is a join index on city of the store dimension, then for each city the join index maintains the tuple IDs of fact table tuples containing that city.

<table>
<thead>
<tr>
<th>SF</th>
<th>t1</th>
<th>t5</th>
<th>t8</th>
</tr>
</thead>
<tbody>
<tr>
<td>LA</td>
<td>t2</td>
<td>t15</td>
<td>t28</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SD</td>
<td>t11</td>
<td>t33</td>
<td>t57</td>
</tr>
</tbody>
</table>

Think: Use examples to show why/how bit indexing and join indexing are important for the efficiency of multidimensional DW processing
Key Data Mining References

- **Books**
  - "Data Warehousing and Data Mining: Implementing Strategic Knowledge Management" by Elliot King, computer technology research
  - "Business Intelligence: The IBM Solution" by Mark Whitehorn, Mary Whitehorn. (1999)
  - "Building Data Mining Applications for CRM" by Alex Berson, et al. 1999
  - "Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations" by Ian H. Witten, Eibe Frank
  - "Advances in Knowledge Discovery and Data Mining" by Usama M. Fayyad(Editor), et al.

- Annual conference, KDD (Knowledge Discovery and Data Mining): ACM SIGKDD

- White papers by many data mining vendors. Examples:
  - [www.bluemartini.com](http://www.bluemartini.com), [www.netgen.com](http://www.netgen.com)