An Overview of Data Warehousing and OLAP Technology

Presentation by Debojit

Discussion by Ali

Data Warehouse Motivation

- Businesses have a lot of data, operational data and facts.
- Data is usually in different databases and in different physical places.

- Decision makers need to access information (data that has been summarized) virtually on the single site.
- Access needs to be fast regardless of the size of data, and how data's age.

What is decision support

- Decision support systems are a class of computerized information systems that support decision making activities.
- Decision support systems usually require consolidating data form many heterogeneous sources: these might include external sources.

  -Such as stock market feeds.

What is data warehouse

- Data warehouse is a collection of decision support technologies, aimed at enabling the analysts to make better and faster decisions. It consists of subject-oriented, integrated, time-variant, and non-volatile collection of data.
  - It contains data from different sources.
  - It retains a long history.
  - Changes as new data is added to the repository.

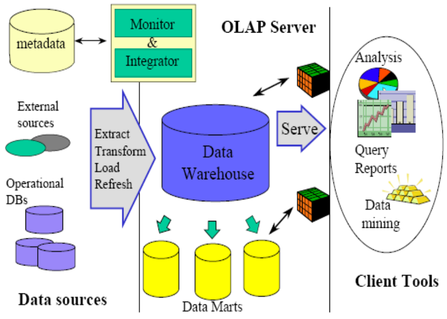
Difference between OLAP and OLTP

	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.	Historical, summarized, multidimensional,…
Usage	repetitive	Ad-hoc
Access	Read/write	Lots of scans
Unit of work	Short, simple transaction	Complex query
# rec accessed	tens	Millions
# users	thousands	Hundreds
DB size	100 MB-GB	100 GB-TB
Metric	Transaction throughput	Query throughput

Why do we separate DW from DB ?

- Performance reasons:
  - OLAP requires special data organization that supports multidimensional views.
  - OLAP queries would degrade operational DB.
  - OLAP is read only.
  - No concurrency control and recovery.
- Decision support requires historical data.
- Decision support requires consolidated data.

Typical OLAP architecture



metadata

Monitor & Integrator

OLAP Server

External sources

Operational DBs

Extract Transform Load Refresh

Data Warehouse

Serve

Analysis

Query Reports

Data mining

Data Marts

Data sources

Client Tools

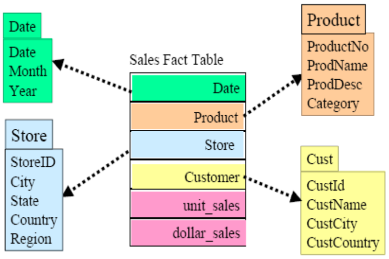
Utilities

- Data Cleaning
  - Data Migration: simple transformation rules (replace "gender" with "sex")
  - Data Scrubbing: use domain-specific knowledge (e.g. zip codes) to modify data.
  - Data Auditing: discover rules and relationships (or signal violations thereof).
- Load
  - Full load: like one big xact – change from old data to new is atomic.
  - Incremental loading ("refresh") makes sense for big warehouses, but transaction model is more complex.

Database Design Methodology

- Most data warehouses use a star schema to represent the multi-dimensional model.
- Each dimension is represented by a dimension-table that describes it.
- A fact-table connects to all dimension-tables with a multiple join. Each tuple in fact-table consists of a pointer to each of the dimension-tables.
- Links between the fact-table in the centre and the dimension-tables form a shape like a star. (Star Schema)

Star Schema Example



Date
Date
Month
Year

Store
StoreID
City
State
Country
Region

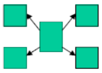
Sales Fact Table
Date
Product
Store
Customer
unit_sales
dollar_sales

Product
ProductNo
ProdName
ProdDesc
Category

Cust
CustId
CustName
CustCity
CustCountry

Database Design Methodology (contd.)

- Each dimension is represented by one table.



➔Un-normalized (introduces redundancy)

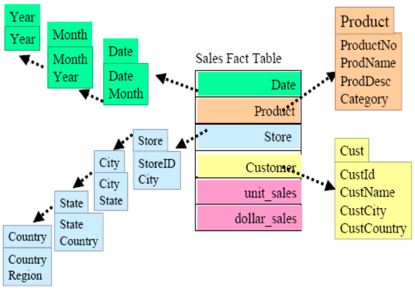
Ex: (Vancouver, BC, Canada, North America)
(Victoria, BC, Canada, North America)

Normalize dimension tables→
Snowflake Schema

Discussion

- Do you think that star schemas are more useful in data warehouses than in RDBMSs? Why or why not?

Metadata Example Snowflake Schema



Year
Year

Month
Month Year

Date
Date Month

Sales Fact Table: Date, Product, Store, Customer, unit_sales, dollar_sales

Product
ProductNo ProdName ProdDesc Category

Store
StoreID City

City
City State

State
State Country

Country
Country Region

Cust
CustId CustName CustCity CustCountry

Important considerations for DW servers

- Indexing
- Materialized Views
- Transformation of complex queries
- Parallel processing
- ROLAP/MOLAP servers
- SQL extensions

Materialized Views

- Materializing summary data can help to accelerate several queries.
- Some of the key challenges :
  - Identify the views to materialize
  - Exploit the materialized views to answer queries
  - Efficiently update the materialized views during load and refresh.

Metadata requirements

- Administrative metadata
- Source database and their contents
- Back-end and front-end tools
- Definitions of the warehouse schema
- Pre-defined queries and reports
- Data mark locations and contents
- Data refresh and purging policies
- User profiles and user access control policies

Metadata requirements

- Business metadata
- Business terms and definitions
- Ownership of data
- Charging policies

- Operational metadata
- Data lineage: history of migrated data and sequence of transformations applied
- Currency of data: active, archived, purged
- Monitoring information: warehouse usage statistics, error reports, audit trails

Discussion

- We can use materialized views both in relational data bases and in data warehouses. Using materialized views in which one is more crucial? Using materialized views in which one is easier? Why?

Data Cube: A Relational Aggregation Operator Generalizing Group-By, Cross-Tab, and Sub-Totals

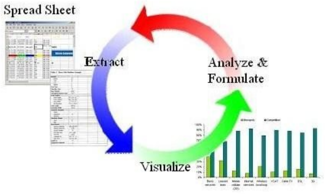
J. Gray, al, *Microsoft Research*
F. Pellow, al, *IBM Research*

Outline

- Data analysis
- Visualization and dimension reduction
- The relational representation of N-dimensional data
- What is CUBE
- Summary

Data analysis applications

- Looking for anomalies or unusual patterns.
- Extract statistical information
- Four steps to aggregate data across many dimensions



- Represent the dataset as an N-dimensional space

The sales example

Model	Year	Color	Number sold
Chevy	1994	Black	50
Chevy	1994	White	40
Ford	1994	Black	50
Ford	1994	White	10
Chevy	1995	Black	85
Chevy	1995	White	115
Ford	1995	Black	85
Ford	1995	White	75

We have ignored a few columns here such as the date of purchase and the dealer

"Dimensionality Reduction"

Analyze car sales

- Focus on the role of model, year and color of the cars
- Ignore differences between sales along dimensions of date of sale or car dealership
- As a result, extensive constructs are used, such as cross-tabulation, subtotals, roll-up and drill-down

Problems with SQL

The three most common problems faced by the SQL GROUP BY are:

1. Histograms
2. Roll-up and drill-down
3. Cross Tabulations

Histograms

- Standard SQL does not allow aggregation over computed categories.
- For example, if we had to sort the car sales by type and then perform aggregation functions on it, standard SQL would not support it.

```
SELECT day, nation, MAX(Temp)
FROM Weather
GROUP BY Day(Time) AS day,
    Nation(Latitude, Longitude) AS nation;
```

One Dimensional Aggregation

Example: Car sales for year 1994 and 1995 showed in table_1:
Table_1:

Model	Sales
Chevy	290
Ford	220

If we need to know the sales for model, we can easily query it by:

```
SELECT      sales
FROM        table_1
GROUP BY model
```

Three Dimensional Aggregation

If we need more dimensional generalization of these operators
Table_2:

Model	Year	Color	Sales
Chevy	1994	black	50
Chevy	1995	black	85
Chevy	1994	white	40
Chevy	1995	white	115

Discussion

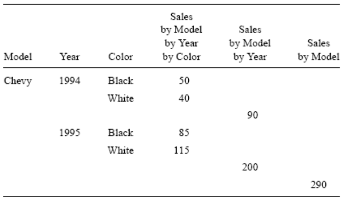
- How useful is multi-dimensional aggregation?
- Besides the data warehousing applications mentioned in the paper, can you think of any other application for multi dimensional aggregation and data cubes?

Roll up/Drill down

If we need to query the sales by model, by year, and by color, then how we can do it?

Typically, we can make a report as showed by

Table_3a:



Model	Year	Color	Sales by Model by Year by Color	Sales by Model by Year	Sales by Model
Chevy	1994	Black	50		
		White	40		
				90	
	1995	Black	85		
		White	115		
				200	
					290

Roll up/Drill down(contd.)

For Table_3a:

- Concepts: going up the levels is called rolling-up the data. Going down is called drilling-down into the data
- In this table, sales are rolled up by using totals and subtotals.
- Data is aggregated by Model, then by Year, then by Color.
- The report shows data aggregated at three levels, that is, at Model level, Year level, and Color level.
- Data aggregated at each distinct level produces a sub-total.

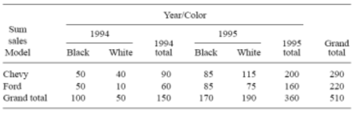
Problems

What problems with Table_3a approach?

- Table_3a suggests creating 2N aggregation columns for a roll-up of N elements.That is, there are six columns in table_3a
- Also, the representation of Table_3a is not relational, because the empty cells (presumably NULL values), cannot form a key

A pivot table in Excel

The approach by using a pivot table in Excel is showed by table_3c:
Table_3c:



Sum sales	Year/Color						
	1994			1995			
Model	Black	White	1994 total	Black	White	1995 total	Grand total
Chevy	50	40	90	85	115	200	290
Ford	50	10	60	85	75	160	220
Grand total	100	50	150	170	190	360	510

What problems with pivot table approach?

- The pivot operator typically aggregating cells based on values in the cells.
- Pivot creates columns based on subsets of column values-this is a much larger set!
- If one pivots on two columns containing N and M values, the resulting pivot table has N x M values, that's, so many columns and such obtuse column names!

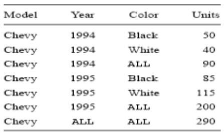
ALL value approach

One more approach by adding an ALL value is available

- Do not extend the result table to have many new columns
- Avoid the exponential growth of columns by overloading column values
- The dummy value "ALL" has been added to fill in the super-aggregation items

ALL value approach (contd.)

Table_3a: Sales summary



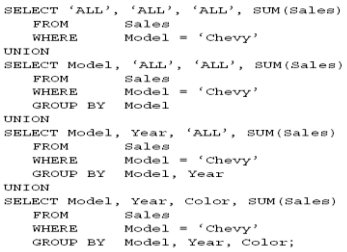
Model	Year	Color	Units
Chevy	1994	Black	50
Chevy	1994	White	40
Chevy	1994	ALL	90
Chevy	1995	Black	85
Chevy	1995	White	115
Chevy	1995	ALL	200
Chevy	ALL	ALL	290

For Table_3a:

- This is a 3_dimensional roll-up
- It have three unions
- The fact is that aggregating over N dimensions requires N such UNIONS!

ALL value approach (contd.)

Since table-3a is a relation, it could be built using SQL, like this statement:



```
SELECT 'ALL', 'ALL', 'ALL', SUM(Sales)
    FROM        Sales
    WHERE       Model = 'Chevy'
UNION
SELECT Model, 'ALL', 'ALL', SUM(Sales)
    FROM        Sales
    WHERE       Model = 'Chevy'
    GROUP BY    Model
UNION
SELECT Model, Year, 'ALL', SUM(Sales)
    FROM        Sales
    WHERE       Model = 'Chevy'
    GROUP BY    Model, Year
UNION
SELECT Model, Year, Color, SUM(Sales)
    FROM        Sales
    WHERE       Model = 'Chevy'
    GROUP BY    Model, Year, Color;
```

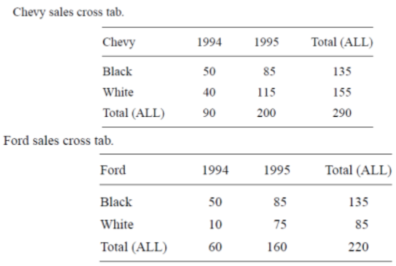
ALL value approach (contd.)

How is ALL value approach ?

- Expressing roll-up and cross-tab queries with conventional SQL is daunting! Why?
- A six dimension cross tab requires a 64-way union of 64 different GROUP BY operators to build the underlying representation.
- The resulting representation of aggregation is too complex to analyze for optimization. On most SQL systems this will result in 64 scans of the data, 64 sorts or hashes, and a long wait

Cross tab

Symmetric aggregation result in a table.



Chevy sales cross tab.

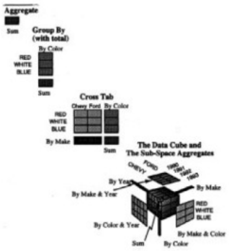
Chevy	1994	1995	Total (ALL)
Black	50	85	135
White	40	115	155
Total (ALL)	90	200	290

Ford sales cross tab.

Ford	1994	1995	Total (ALL)
Black	50	85	135
White	10	75	85
Total (ALL)	60	160	220

The CUBE operator

- N-dimensional generalization of simple aggregate functions
- N-1 lower-dimensional aggregates are points, lines, planes, cubes
- data cube operator builds a table containing all these aggregate values
- OD data cube: a point.
- 1D data cube: a line & a point.
- 2D data cube: a cross tabulation, a plane, two lines, and a point.
- 3D data cube: a cube with three intersecting 2D cross tabs



The CUBE operator (contd.)

- For example:

SELECT Model, Year, Color, SUM (Sales) AS sales
FROM Sales
WHERE Model in ['Ford', 'Chevy']   AND year BETWEEN 1994 AND 1995
GROUP BY CUBE Model, Year, Color

- A relational operator
- GROUP BY and ROLL UP are degenerate forms of the operator.
- Aggregates over all <select list> attributes in GROUP BY clause as in standard GROUP BY
- It UNIONs in each super-aggregate of global cube—substituting ALL for the aggregation columns
- If there are N attributes in the <select list>, there will be 2N -1 super-aggregate value
- The super-aggregates are produced by ROLLUP, like running sum or average

Summary

- The cube operator generalizes and unifies several common and popular concepts: such as aggregates, group by, histograms, roll-ups and drill-downs and, cross tabs.
- The cube operator is based on a relational representation of aggregate data using the ALL value to denote the set over which each aggregation is computed.
- The data cube is easy to compute for a wide class of functions
- SQL's basic set of five aggregate functions needs careful extension to include

Discussion

- How hard did you find it to understand the CUBE operator?
- As a query writer, would you feel comfortable using it? Or, would you rather use the "solutions" described in the previous slides?

Thanks