Data Warehousing and Decision Support

M. Tamer Özsu

David R. Cheriton School of Computer Science
University of Waterloo

CS 348
Introduction to Database Management
Fall 2012

Notes

Outline

1. Introduction to Decision Support
2. On-Line Analytical Processing
   - Multidimensional Data
   - Multidimensional Queries
3. Data Warehousing
   - Creating and Maintaining a Warehouse
   - Materializing Views

Notes

Transaction Processing

The most common use of relational databases is for *operational data*.

- Examples:
  - Students enrolling in courses
  - Customers purchasing products
  - Passengers purchasing airline tickets

On-Line Transactional Processing (OLTP)

Databases that support the basic operations of a business are generally classified as OLTP systems.

- Workload characteristics:
  1. simple queries
  2. many short transactions making small changes
- Systems tuned to maximize throughput of concurrent transactions

Notes

Beyond Transaction Processing

More recent uses of operational data:

Decision Support Summarizing data to support high-level decision making

- Complex queries with much aggregation

Data Mining Searching for trends or patterns in data for a business to exploit

- Simple queries, but very data-intensive

Data Warehousing

A *data warehouse* is a separate copy of the operational data used for executing decision support and/or data mining queries.

Notes

On-Line Analytical Processing

On-Line Analytical Processing (OLAP)

OLAP is a particular type of decision support

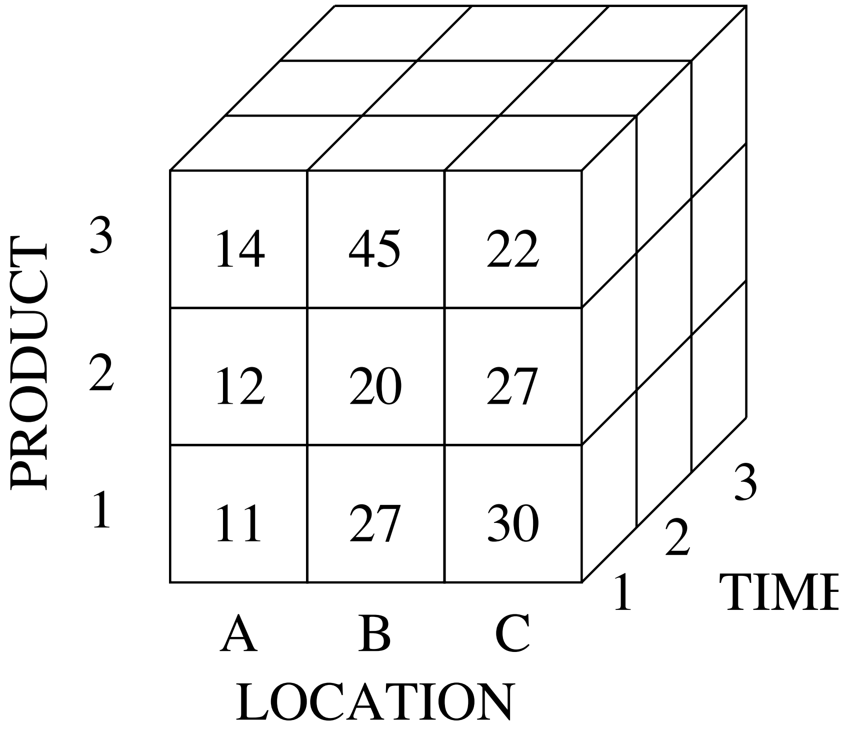
- Data is modeled as multidimensional array
- Queries are usually ad hoc
- Queries select and aggregate cells of the array

- OLAP systems are divided into two categories:
  1. Special-purpose OLAP systems
     - store data as multidimensional arrays ("MOLAP")
     - provide an OLAP-specific query language
  2. Relational databases
     - store data in relations ("ROLAP")
     - queries written in SQL

Notes

Multidimensional Data

- Example: Number of Sales



PRODUCT	A	B	C
3	14	45	22
2	12	20	27
1	11	27	30

Notes

Star Schemas

Location

lid	*store*	*city*	*province*	*country*
A	Weber	Waterloo	ON	CA
B	F-H	Kitchener	ON	CA
C	Park	Kitchener	ON	CA

Product

pid	*pname*	*category*	*price*
1	Bolt	Hardware	.10
2	Nut	Hardware	.05
3	Wrench	Tools	1.99

Time

tid	*date*	*week*	*month*	*quarter*	*year*
virtual relation					

Sales

lid	*pid*	*tid*	*sales*
A	1	1	11
A	2	1	12
A	3	1	14
B	1	1	27
B	2	1	20
B	3	1	45
C	1	1	30
C	2	1	27
C	3	1	22
A	1	2	16
A	2	2	20
A	3	2	55
⋮			

Notes

OLAP Queries

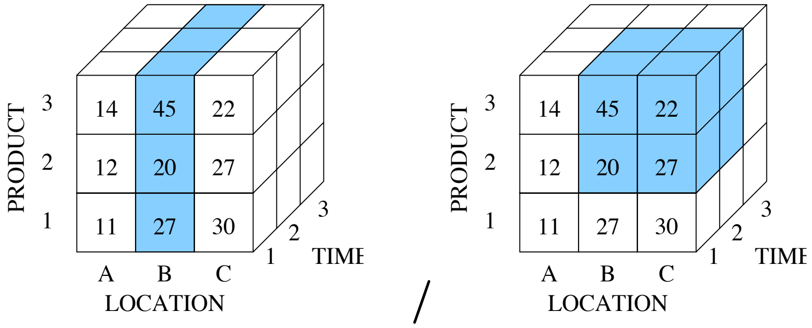
- OLAP queries typically aggregate over one or more dimensions. Examples:
  - Total sales
  - Total sales this year for each product category
  - Total sales for each store per quarter

- OLAP is a tool for *ad hoc* data exploration/visualization
  - Ad hoc queries tend to be iterative
  - Desirable to express queries using operations over previous result

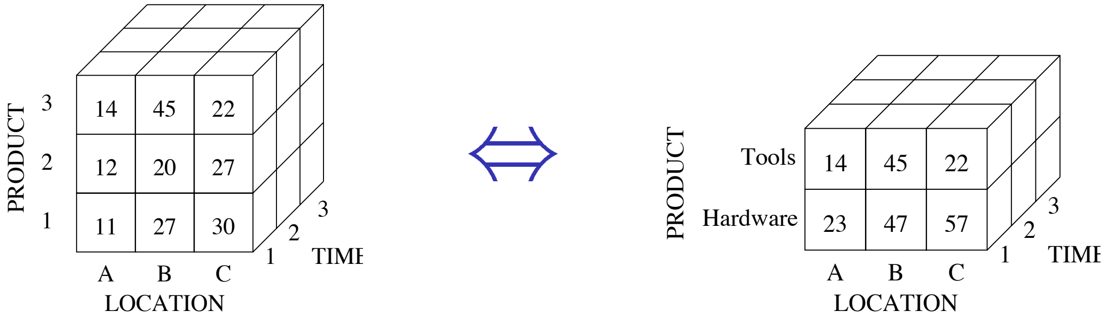
Notes

OLAP Query Operations

- Slicing and Dicing



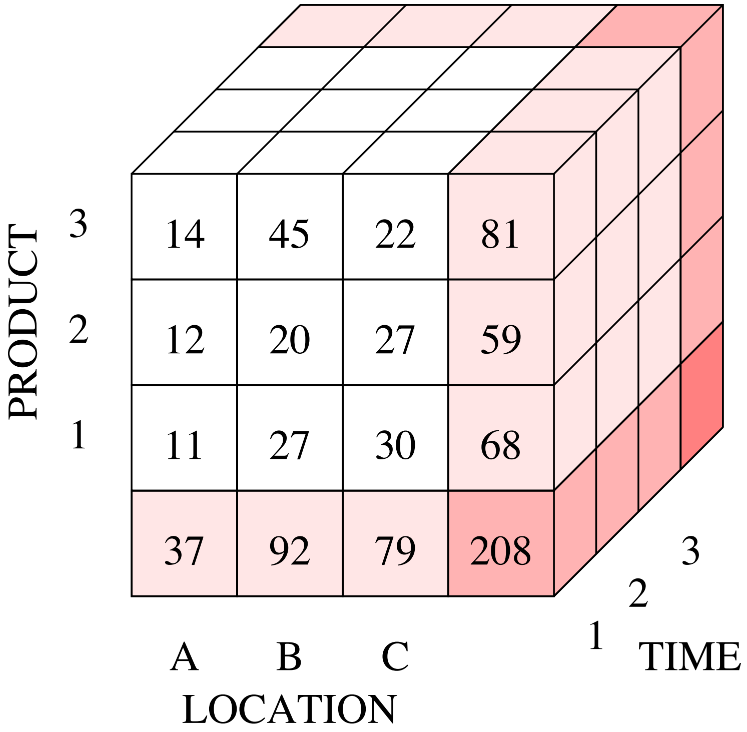
- Roll-up and Drill-down



Notes

Data Cube

- A *data cube* extends a multidimensional array of data to include all possible aggregated totals



Notes

Data Cubes as Relations

Sales

lid	*pid*	*tid*	*sales*
A	1	1	11
A	2	1	12
A	3	1	14
A	-	1	37
B	1	1	27
B	2	1	20
B	3	1	45
B	-	1	92
C	1	1	30
C	2	1	27
C	3	1	22
C	-	1	79
-	1	1	68
-	2	1	59
-	3	1	81
-	-	1	208
A	1	2	16
	⋮		

Notes

CUBE operator in SQL:1999

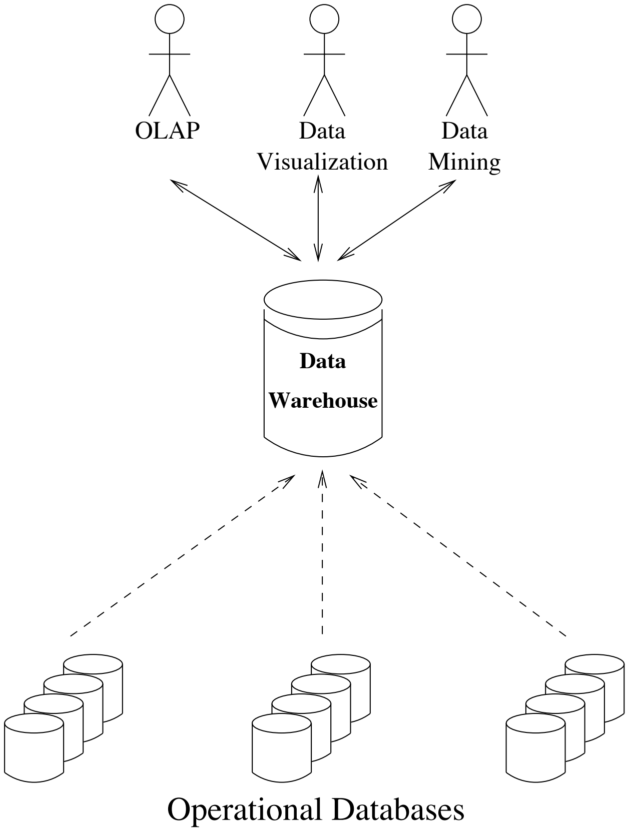
- Generating the data cube:
  1. SUM(sales) GROUP BY location, product, time (raw cells)
  2. SUM(sales) GROUP BY location, time
  3. SUM(sales) GROUP BY product, time
  4. SUM(sales) GROUP BY product, location
  5. SUM(sales) GROUP BY product
  6. SUM(sales) GROUP BY location
  7. SUM(sales) GROUP BY time
  8. SUM(sales)

- CUBE operator in SQL:1999 groups by all combinations

```
SELECT lid, pid, tid, SUM(sales)
FROM Sales
GROUP BY CUBE(lid, pid, tid)
```

Notes

Data Warehousing



Notes

Creating and Maintaining a Warehouse

Necessary steps when creating a warehouse:

Extract Run queries against the operational databases to retrieve necessary data

Clean Delete or repair tuples with missing or invalid information

Transform Reorganize the data to fit the conceptual schema of the warehouse

Load Populate the warehouse tables; build indexes and/or materialized views

Note

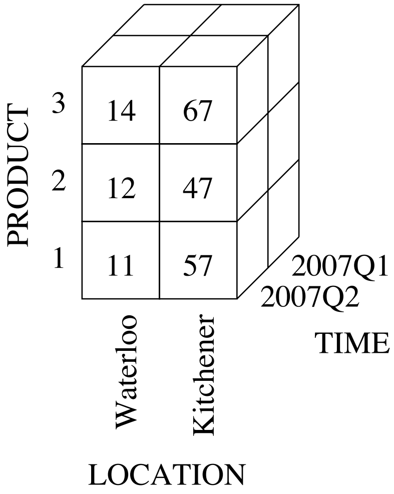
The data in the warehouse needs to be refreshed periodically (typically nightly or weekly). To make this process efficient, the above steps need to be executed *incrementally*.

Notes

Materializing Views

- Consider the following view of the Sales data:

```
CREATE VIEW
ByCityQuarter(city, pid, quarter, sales) AS
 SELECT city, pid, QUARTER(tid), SUM(sales)
 FROM Sales s, Location l
 WHERE s.lid = l.lid
 GROUP BY city, pid, QUARTER(tid)
```



- View `ByCityQuarter` is useful for any query that
  1. Rolls-up the Location dimension to *at least* City; and
  2. Rolls-up the Time dimension to *at least* Quarter

Notes

Materializing Views (cont'd)

- Issues related to using materialized views:
  1. Which views to materialize (*view selection*)
  2. Which views are useful to answer a query (*view matching*)
  3. Which indexes to build on the views
  4. How to refresh the data in the view. Options:
     - Synchronous incremental maintenance
     - Asynchronous incremental maintenance
     - No synchronization (periodic re-creation)

Observation

These are the very same issues that apply to the entire data warehouse, relative to the data in the operational databases.

Notes