1 Introduction to Decision Support

2 On-Line Analytical Processing
   Multidimensional Data
   Multidimensional Queries

3 Data Warehousing
   Creating and Maintaining a Warehouse
   Materializing Views
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
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.
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
Multidimensional Data

- Example: Number of Sales
### Star Schemas

#### Location

<table>
<thead>
<tr>
<th>lid</th>
<th>store</th>
<th>city</th>
<th>province</th>
<th>country</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Weber</td>
<td>Waterloo</td>
<td>ON</td>
<td>CA</td>
</tr>
<tr>
<td>B</td>
<td>F-H</td>
<td>Kitchener</td>
<td>ON</td>
<td>CA</td>
</tr>
<tr>
<td>C</td>
<td>Park</td>
<td>Kitchener</td>
<td>ON</td>
<td>CA</td>
</tr>
</tbody>
</table>

#### Product

<table>
<thead>
<tr>
<th>pid</th>
<th>pname</th>
<th>category</th>
<th>price</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Bolt</td>
<td>Hardware</td>
<td>.10</td>
</tr>
<tr>
<td>2</td>
<td>Nut</td>
<td>Hardware</td>
<td>.05</td>
</tr>
<tr>
<td>3</td>
<td>Wrench</td>
<td>Tools</td>
<td>1.99</td>
</tr>
</tbody>
</table>

#### Time

<table>
<thead>
<tr>
<th>tid</th>
<th>date</th>
<th>week</th>
<th>month</th>
<th>quarter</th>
<th>year</th>
</tr>
</thead>
</table>

**virtual relation**

#### Sales

<table>
<thead>
<tr>
<th>lid</th>
<th>pid</th>
<th>tid</th>
<th>sales</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>1</td>
<td>1</td>
<td>11</td>
</tr>
<tr>
<td>A</td>
<td>2</td>
<td>1</td>
<td>12</td>
</tr>
<tr>
<td>A</td>
<td>3</td>
<td>1</td>
<td>14</td>
</tr>
<tr>
<td>B</td>
<td>1</td>
<td>1</td>
<td>27</td>
</tr>
<tr>
<td>B</td>
<td>2</td>
<td>1</td>
<td>20</td>
</tr>
<tr>
<td>B</td>
<td>3</td>
<td>1</td>
<td>45</td>
</tr>
<tr>
<td>C</td>
<td>1</td>
<td>1</td>
<td>30</td>
</tr>
<tr>
<td>C</td>
<td>2</td>
<td>1</td>
<td>27</td>
</tr>
<tr>
<td>C</td>
<td>3</td>
<td>1</td>
<td>22</td>
</tr>
<tr>
<td>A</td>
<td>1</td>
<td>2</td>
<td>16</td>
</tr>
<tr>
<td>A</td>
<td>2</td>
<td>2</td>
<td>20</td>
</tr>
<tr>
<td>A</td>
<td>3</td>
<td>2</td>
<td>55</td>
</tr>
</tbody>
</table>

---

**CS 348** | **Warehousing** | **Fall 2012** | **7 / 16**

**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
OLAP Query Operations

- Slicing and Dicing

- Roll-up and Drill-down
Data Cube

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

![Data Cube Diagram]

CS 348 | Warehousing | Fall 2012 | 10 / 16

Notes

_________________________________________________________________

_________________________________________________________________

_________________________________________________________________

_________________________________________________________________

_________________________________________________________________

_________________________________________________________________

_________________________________________________________________

_________________________________________________________________
Data Cubes as Relations

<table>
<thead>
<tr>
<th>lid</th>
<th>pid</th>
<th>tid</th>
<th>sales</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>1</td>
<td>1</td>
<td>11</td>
</tr>
<tr>
<td>A</td>
<td>2</td>
<td>1</td>
<td>12</td>
</tr>
<tr>
<td>A</td>
<td>3</td>
<td>1</td>
<td>14</td>
</tr>
<tr>
<td>A</td>
<td>-</td>
<td>1</td>
<td>37</td>
</tr>
<tr>
<td>B</td>
<td>1</td>
<td>1</td>
<td>27</td>
</tr>
<tr>
<td>B</td>
<td>2</td>
<td>1</td>
<td>20</td>
</tr>
<tr>
<td>B</td>
<td>3</td>
<td>1</td>
<td>45</td>
</tr>
<tr>
<td>B</td>
<td>-</td>
<td>1</td>
<td>92</td>
</tr>
<tr>
<td>C</td>
<td>1</td>
<td>1</td>
<td>30</td>
</tr>
<tr>
<td>C</td>
<td>2</td>
<td>1</td>
<td>27</td>
</tr>
<tr>
<td>C</td>
<td>3</td>
<td>1</td>
<td>22</td>
</tr>
<tr>
<td>C</td>
<td>-</td>
<td>1</td>
<td>79</td>
</tr>
<tr>
<td>-</td>
<td>1</td>
<td>1</td>
<td>68</td>
</tr>
<tr>
<td>-</td>
<td>2</td>
<td>1</td>
<td>59</td>
</tr>
<tr>
<td>-</td>
<td>3</td>
<td>1</td>
<td>81</td>
</tr>
<tr>
<td>-</td>
<td>-</td>
<td>1</td>
<td>208</td>
</tr>
<tr>
<td>A</td>
<td>1</td>
<td>2</td>
<td>16</td>
</tr>
</tbody>
</table>

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)
```
Data Warehousing

- OLAP
- Data Visualization
- Data Mining

Operational Databases

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.*
• 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
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.