Unit 11
Online Analytical Processing (OLAP)
Basic Concepts
OLAP in Context

User Level (View Level)
- Derived Tables
- Constraints, Privileges
  - Queries (DML)
  - Derived

Community Level (Base Level)
- Base Tables
- Constraints, Privileges
  - Application Data Analysis (ER)
  - Normalization (NFs)
  - Schema Specification (DDL)
  - Queries (DML)

Implemented

Physical Level
- Files
- Indexes, Distribution
  - Query Execution ($B^+$, ..., Execution Plan)

Relies on

DBMS OS Level
- Concurrency
- Recovery
  - Transaction Processing (ACID, Sharding)

Runs on

Centralized Or Distributed
- Standard OS
- Standard Hardware


OLAP vs. OLTP

◆ We have focused until now on OLTP: Online Transaction Processing
◆ This dealt with storing data both logically and physically and managing transactions querying and modifying the data
◆ We will now focus providing support for analytical queries, essentially statistical and summary information for decision-makers, that is on OLAP: Online Analytical Processing
◆ This may be accomplished by preprocessing, for efficiency purposes, and producing special types of views, which are also not necessarily up to date
  • Not up to date may not be a problem in OLAP
◆ Data for OLAP (and more generally for data mining) is frequently stored in a Data Warehouse
Example

- Our company has several stores and sells several products
- The stores are in different locations
- The locations, identified by (city, state) pairs are grouped into several regions
- We partition the times of sale into four quarters
- The quarters are grouped into two half-years
Our Company

<table>
<thead>
<tr>
<th>Store</th>
<th>Store#</th>
<th>City</th>
<th>State</th>
<th>Region</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alpha</td>
<td>New York</td>
<td>NY</td>
<td>NE</td>
<td></td>
</tr>
<tr>
<td>Beta</td>
<td>Albany</td>
<td>NY</td>
<td>NE</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Quarter</th>
<th>Quarter#</th>
<th>Half_Year</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>First</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>First</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Second</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Second</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Product</th>
<th>Product#</th>
<th>Weight</th>
<th>Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>Book</td>
<td>4</td>
<td>100</td>
<td></td>
</tr>
<tr>
<td>Glass</td>
<td>15</td>
<td>200</td>
<td></td>
</tr>
</tbody>
</table>
# Our Sales

<table>
<thead>
<tr>
<th>Sale</th>
<th>Store#</th>
<th>Product#</th>
<th>Quarter#</th>
<th>$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alpha</td>
<td>Book</td>
<td></td>
<td>1</td>
<td>70,000</td>
</tr>
<tr>
<td>Alpha</td>
<td>Glass</td>
<td></td>
<td>1</td>
<td>90,000</td>
</tr>
<tr>
<td>Beta</td>
<td>Book</td>
<td></td>
<td>1</td>
<td>90,000</td>
</tr>
<tr>
<td>Beta</td>
<td>Glass</td>
<td></td>
<td>1</td>
<td>80,000</td>
</tr>
<tr>
<td>Alpha</td>
<td>Book</td>
<td></td>
<td>2</td>
<td>90,000</td>
</tr>
<tr>
<td>Alpha</td>
<td>Glass</td>
<td></td>
<td>2</td>
<td>90,000</td>
</tr>
<tr>
<td>Beta</td>
<td>Book</td>
<td></td>
<td>2</td>
<td>60,000</td>
</tr>
<tr>
<td>Beta</td>
<td>Glass</td>
<td></td>
<td>2</td>
<td>50,000</td>
</tr>
<tr>
<td>Alpha</td>
<td>Book</td>
<td></td>
<td>3</td>
<td>60,000</td>
</tr>
<tr>
<td>Alpha</td>
<td>Glass</td>
<td></td>
<td>3</td>
<td>80,000</td>
</tr>
<tr>
<td>Beta</td>
<td>Book</td>
<td></td>
<td>3</td>
<td>50,000</td>
</tr>
<tr>
<td>Beta</td>
<td>Glass</td>
<td></td>
<td>3</td>
<td>90,000</td>
</tr>
<tr>
<td>Alpha</td>
<td>Book</td>
<td></td>
<td>4</td>
<td>50,000</td>
</tr>
<tr>
<td>Alpha</td>
<td>Glass</td>
<td></td>
<td>4</td>
<td>50,000</td>
</tr>
<tr>
<td>Beta</td>
<td>Book</td>
<td></td>
<td>4</td>
<td>70,000</td>
</tr>
<tr>
<td>Beta</td>
<td>Glass</td>
<td></td>
<td>4</td>
<td>70,000</td>
</tr>
</tbody>
</table>
Star Schema

- We want to support queries useful for statistical analysis by computing various sums, averages, etc.
- The structure we have is a **star schema**
- In the middle we have our **facts table**

![Star Schema Diagram]

- This, of course is just a standard ternary relationship among 3 entity sets, but each community uses its own jargon and some may not understand what entity sets and relationships are.
Snowflake Schema: Normalized Star Schema

- One could also normalize, as table Store is not normalized, since State $\rightarrow$ Region

- Then, one could get, which we will not consider further, a snowflake schema
**Cube**

◆ We could think of each row of fact table as occupying a voxel (volume element) in a *cube*

◆ Cube, in general, can have any number of dimensions; in our example there are three

◆ This cube can then be *sliced and diced*
Slice

- SELECT Store#, Product#, SUM($)
  FROM Sale
  GROUP BY Store#, Product#

- We can do all kinds of such slices

- As you can see, this is a standard SQL statement but “visualized” differently
Dimension Hierarchies

- Dimensions could have hierarchies (or more generally even lattices)
- We have two very simple hierarchies
  - One temporal: quarters are in half years
  - One geographical: cities are in states are in regions
Using Hierarchies

- SELECT Sale.Product#, Quarter.Half_Year, SUM($)
  FROM Sale, Quarter
  WHERE Sale.Quarter# = Quarter.Quarter#
  GROUP BY Half_Year;

- Will produce summaries by half years, not quarters

- As you can see, this is a standard SQL statement but “visualized” differently
**New Operator: CUBE**

- SELECT Store#, Product#, SUM($) FROM Sale GROUP BY CUBE (Store#,Product#);

- Will produce all possible aggregations based on subsets of {Store#,Product#}, best explained by looking at what will come out

<table>
<thead>
<tr>
<th>Store#</th>
<th>Product#</th>
<th>$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alpha</td>
<td>Book</td>
<td>270,000</td>
</tr>
<tr>
<td>Alpha</td>
<td>Glass</td>
<td>310,000</td>
</tr>
<tr>
<td>Beta</td>
<td>Book</td>
<td>270,000</td>
</tr>
<tr>
<td>Beta</td>
<td>Glass</td>
<td>290,000</td>
</tr>
<tr>
<td>Alpha</td>
<td>NULL</td>
<td>580,000</td>
</tr>
<tr>
<td>Beta</td>
<td>NULL</td>
<td>560,000</td>
</tr>
<tr>
<td>NULL</td>
<td>Book</td>
<td>540,000</td>
</tr>
<tr>
<td>NULL</td>
<td>Glass</td>
<td>600,000</td>
</tr>
<tr>
<td>NULL</td>
<td>NULL</td>
<td>1,140,000</td>
</tr>
</tbody>
</table>
New Operator: ROLLUP

- ROLLUP produces only some of the aggregate operators produced by CUBE, best explained by example
- SELECT Store#, Product#, SUM($) FROM Sale GROUP BY ROLLUP (Store#, Product#);

<table>
<thead>
<tr>
<th>Store#</th>
<th>Product#</th>
<th>$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alpha</td>
<td>Book</td>
<td>270,000</td>
</tr>
<tr>
<td>Alpha</td>
<td>Glass</td>
<td>310,000</td>
</tr>
<tr>
<td>Beta</td>
<td>Book</td>
<td>270,000</td>
</tr>
<tr>
<td>Beta</td>
<td>Glass</td>
<td>290,000</td>
</tr>
<tr>
<td>Alpha</td>
<td>NULL</td>
<td>580,000</td>
</tr>
<tr>
<td>Beta</td>
<td>NULL</td>
<td>560,000</td>
</tr>
<tr>
<td>NULL</td>
<td>NULL</td>
<td>1,140,000</td>
</tr>
</tbody>
</table>
**ROLL and MOLAP**

- **ROLAP**: Relational OLAP
- That is what we have been doing: OLAP information was stored as a set of star (or more generally snowflakes) schemas

- **MOLAP**: Multidimensional OLAP
- Information not stored as a relational database, but essentially as a cube
Oracle

- Oracle supports OLAP
- Following is an SQL code
Oracle: Defining the Database

create table store(
    sid char(20) primary key,
    city char(20),
    state char(20),
    region char(20)
);

create table product(
    pid char(20) primary key,
    weight number,
    price number
);

create table quarter(
    qid number primary key,
    half_year char(10)
);
create table sale(
    sid char(20),
    pid char(20),
    qid number,
    profit number,
    primary key(qid, pid, sid),
    foreign key(qid) references quarter(qid),
    foreign key(pid) references product(pid),
    foreign key(sid) references store(sid)
);
Oracle: OLAP Query

```sql
select sid, pid, sum(profit)
from sale
group by rollup(sid, pid);

select sid, pid, sum(profit)
from sale
group by cube(sid, pid);
```
Key Ideas

◆ OLAP vs. OLTP
◆ Star schema
◆ Snowflake schema
◆ Cube
◆ Slicing and dicing
◆ Dimension hierarchies
◆ ROLAP
◆ MOLAP
◆ Oracle support