

Database Management Systems

Winter 2004

CMPUT 391: Data Warehousing

Dr. Osmar R. Zaïane



University of Alberta

Chapter 19 of
Textbook

Objectives of Lecture 5

Data Warehousing and OLAP

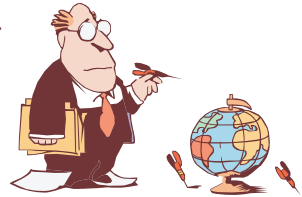
- Realize the purpose of data warehousing.
- Comprehend the data structures behind data warehouses and understand the OLAP technology.
- Get an overview of the schemas used for multi-dimensional data.
- See some implementations of OLAP operators with SQL

Data Warehouse and OLAP

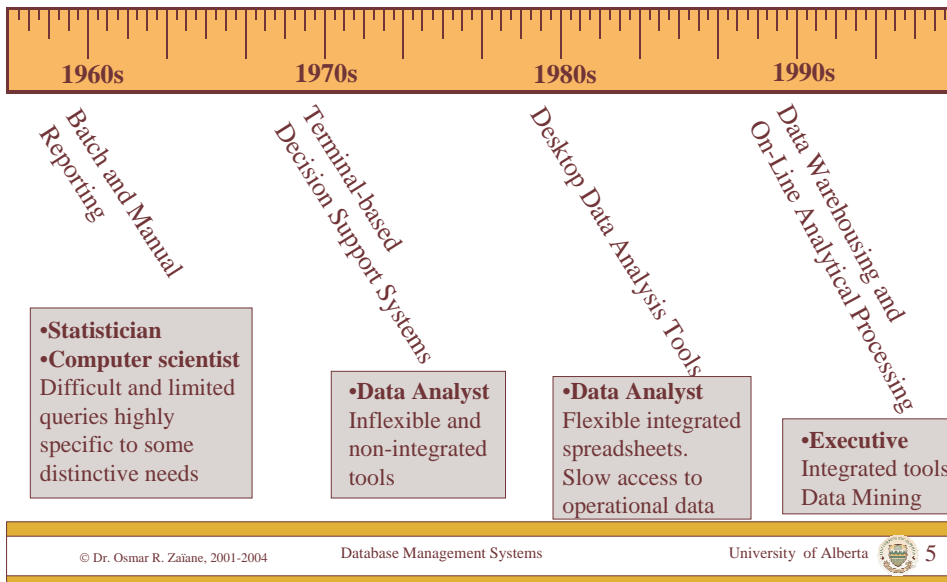


- What is a data warehouse and what is it for?
- What is the multi-dimensional data model?
- What is the difference between OLAP and OLTP?
- What is the general architecture of a data warehouse?
- How can we implement a data warehouse?
- Are there issues related to data cube technology?

Incentive for a Data Warehouse

- Businesses have a lot of data, operational data and facts.
 - This data is usually in different databases and in different physical places.
 - Data is available (or archived), but in different formats and locations. (heterogeneous and distributed).
- 
- Decision makers need to access information (data that has been summarized) virtually on one single site.
 - This access needs to be fast regardless of the size of the data, and how old the data is.

Evolution of Decision Support Systems



What Is Data Warehouse?

- A data warehouse *consolidates* different data sources.
- A data warehouse is a database that is *different and maintained separately* from an operational database.
- A data warehouse combines and merges information in a consistent database (not necessarily up-to-date) to help decision support.



Decision support systems access data warehouse and do not need to access operational databases → do not unnecessarily over-load operational databases.



Definitions

Data Warehouse is a subject-oriented, integrated, time-variant and non-volatile collection of data in support of management's decision making process. (*W.H. Inmon*)

Subject oriented: oriented to the major subject areas of the corporation that have been defined in the data model.

Integrated: data collected in a data warehouse originates from different heterogeneous data sources.

Time-variant: The dimension "time" is all-pervading in a data warehouse. The data stored is not the current value, but an evolution of the value in time.

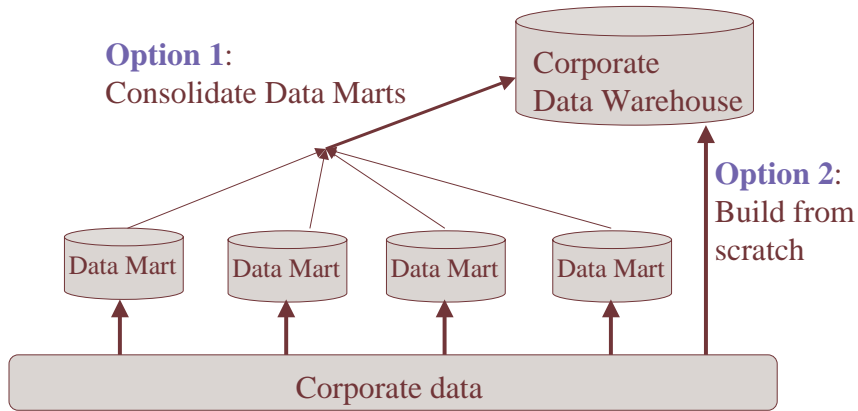
Non-volatile: update of data does not occur frequently in the data warehouse. The data is loaded and accessed.

Definitions (con't)

Data Warehousing is the process of constructing and using data warehouses.

A corporate data warehouse collects data about *subjects* spanning the **whole** organization. **Data Marts** are specialized, single-line of business warehouses. They collect data for a department or a specific group of people.

Building a Data Warehouse

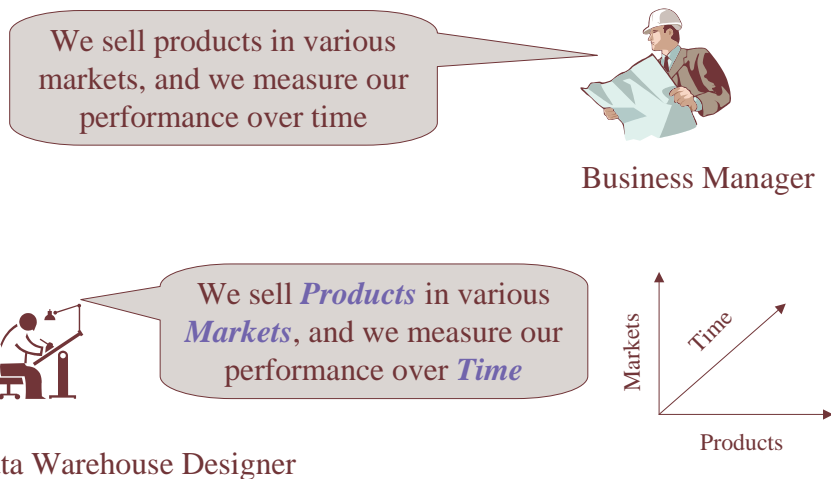


Data Warehouse and OLAP



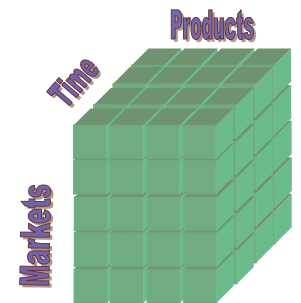
- What is a data warehouse and what is it for?
- What is the multi-dimensional data model?
- What is the difference between OLAP and OLTP?
- What is the general architecture of a data warehouse?
- How can we implement a data warehouse?
- Are there issues related to data cube technology?

Describing the Organization



Construction of Data Warehouse Based on Multi-dimensional Model

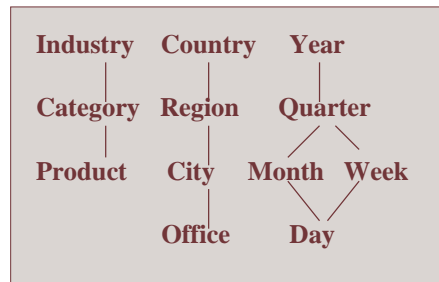
- Think of it as a *cube* with labels on each edge of the cube.
- The cube doesn't just have 3 dimensions, but may have many dimensions (N).
- Any point inside the cube is at the intersection of the coordinates defined by the edge of the cube.
- A point in the cube may store values (measurements) relative to the combination of the labeled dimensions.



Concept-Hierarchies

Most Dimensions are hierarchical by nature: total orders or partial orders
 Example: Location(continent → country → province → city)
 Time(year→quarter→(month,week)→day)

Dimensions: Product, Region, Time
 Hierarchical summarization paths



Data Warehouse and OLAP



- What is a data warehouse and what is it for?
- What is the multi-dimensional data model?
- What is the difference between OLAP and OLTP?
- What is the general architecture of a data warehouse?
- How can we implement a data warehouse?
- Are there issues related to data cube technology?

On-Line Transaction Processing

- Database management systems are typically used for on-line transaction processing (OLTP)
- OLTP applications normally automate clerical data processing tasks of an organization, like data entry and enquiry, transaction handling, etc. (access, read, update)
- Database is current, and consistency and recoverability are critical. Records are accessed one at a time.



- OLTP operations are structured and repetitive
- OLTP operations require detailed and up-to-date data
- OLTP operations are short, atomic and isolated transactions

Databases tend to be hundreds of Mb to Gb.

On-Line Analytical Processing



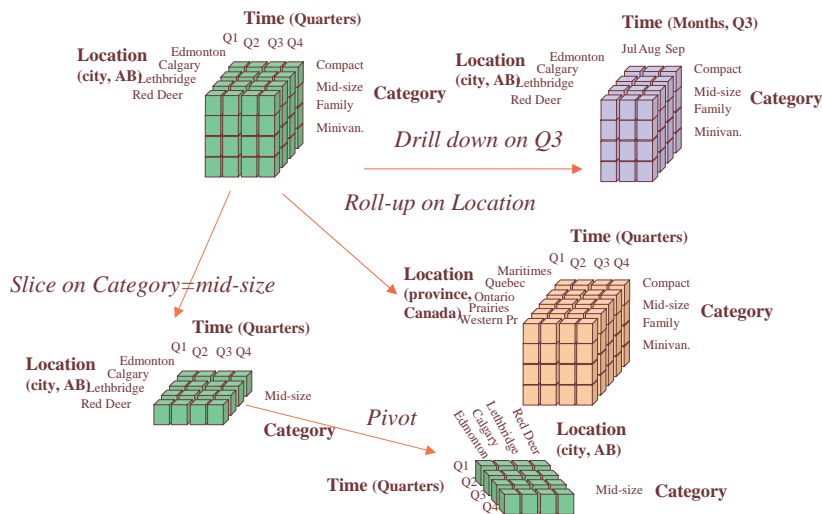
- On-line analytical processing (OLAP) is essential for decision support.
- OLAP is supported by data warehouses.
- Data warehouse consolidation of operational databases.
- The key structure of the data warehouse always contains some element of time.
- Owing to the hierarchical nature of the dimensions, OLAP operations view the data flexibly from different perspectives (different levels of abstractions).

•OLAP operations:

- **roll-up** (increase the level of abstraction)
- **drill-down** (decrease the level of abstraction)
- **slice** and **dice** (selection and projection)
- **pivot** (re-orient the multi-dimensional view)
- **drill-through** (links to the raw data)

DW tend to be in the order of Tb

Data Warehouse OLAP Example



OLTP vs OLAP

	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, flat relational isolated	historical, summarized, multidimensional integrated, consolidated
usage	repetitive	ad-hoc
access	read/write index/hash on prim. key	lots of scans
unit of work	short, simple transaction	complex query
# records accessed	tens	millions
#users	thousands	hundreds
DB size	100MB-GB	100GB-TB
metric	transaction throughput	query throughput, response

(Source: JH)

Why Do We Separate DW From DB?

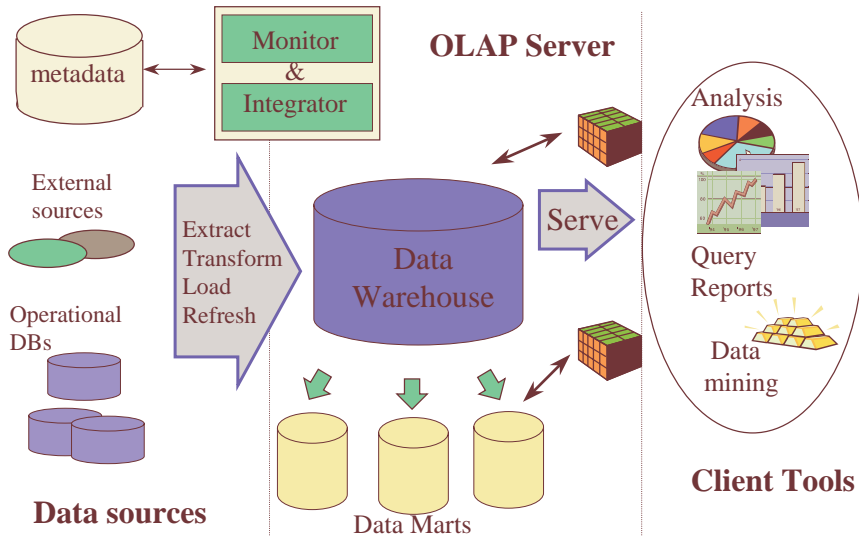
- Performance reasons:
 - OLAP necessitates special data organization that supports multidimensional views.
 - OLAP queries would degrade operational DB.
 - OLAP is read only.
 - No concurrency control and recovery.
- Decision support requires historical data.
- Decision support requires consolidated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

Data Warehouse and OLAP



- What is a data warehouse and what is it for?
- What is the multi-dimensional data model?
- What is the difference between OLAP and OLTP?
- What is the general architecture of a data warehouse?
- How can we implement a data warehouse?
- Are there issues related to data cube technology?

Three-tier Architecture



Data Warehouse and OLAP



- What is a data warehouse and what is it for?
- What is the multi-dimensional data model?
- What is the difference between OLAP and OLTP?
- What is the general architecture of a data warehouse?
- How can we implement a data warehouse?
- Are there issues related to data cube technology?
- Can we mine data warehouses?

Data Warehouse Design

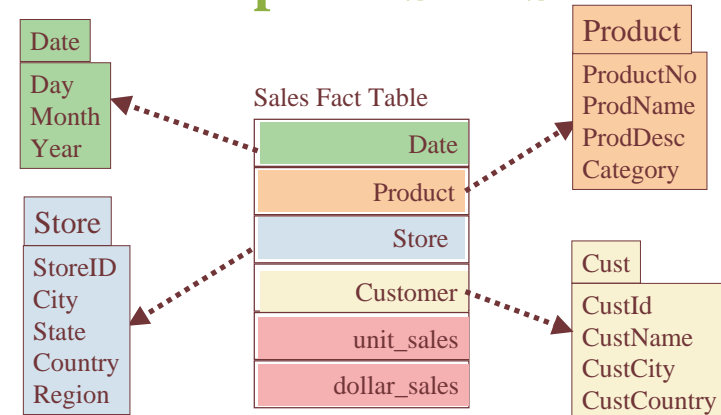
Most data warehouses use a **star schema** to represent the multi-dimensional model.

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 the fact-table consists of a pointer to each of the dimension-tables that provide its multi-dimensional coordinates and stores measures for those coordinates.

The links between the fact-table in the centre and the dimension-tables in the extremities form a shape like a star. (*Star Schema*)

Example of Star Schema



Star schema: A single object (fact table) in the middle connected to a number of objects (dimension tables)

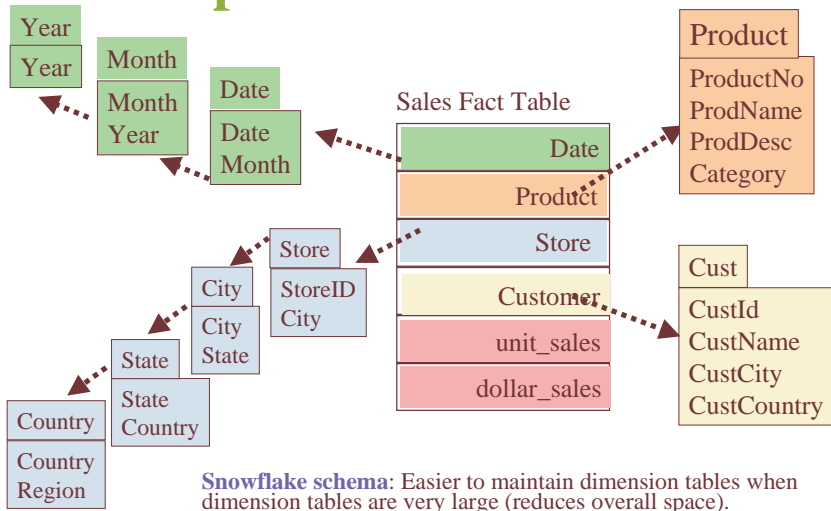
Each dimension is represented by one table

→ Un-normalized (introduces redundancy).

Normalize dimension tables → **Snowflake schema**

(Source: JH)

Example of Snowflake Schema



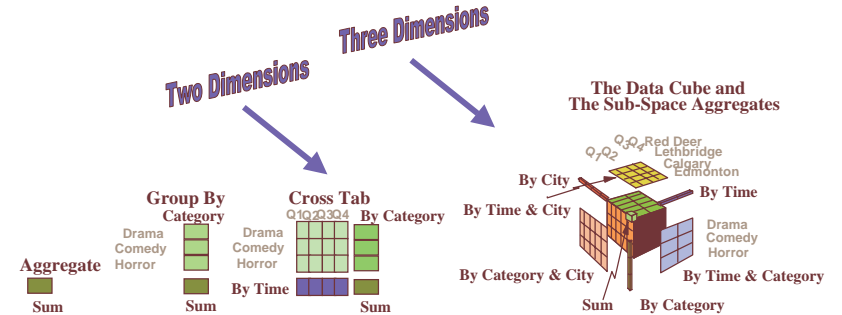
Snowflake schema: Easier to maintain dimension tables when dimension tables are very large (reduces overall space).

Star schema: More effective for data cube browsing (less joins) can affect performance.

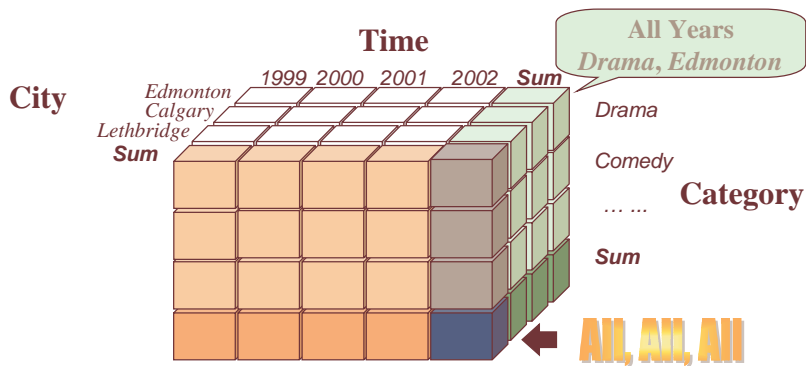
(Source: JH)

Aggregation in Data Warehouses

Multidimensional view of data in the warehouse:
Stress on aggregation of measures by one or more dimensions



Construction of Multi-dimensional Data Cube



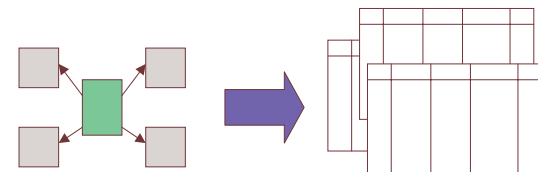
Implementation of the OLAP Server

ROLAP: Relational OLAP - data is stored in tables in relational database or extended-relational databases. They use an RDBMS to manage the warehouse data and aggregations using often a star schema.

- They support extensions to SQL
- A cell in the multi-dimensional structure is represented by a tuple.

Advantage: Scalable (no empty cells for sparse cube).

Disadvantage: no direct access to cells.



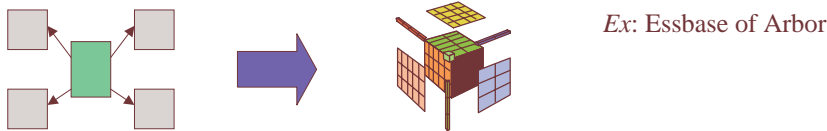
Ex: Microstrategy Metacube (Informix)

Implementation of the OLAP Server

MOLAP: Multidimensional OLAP – implements the multidimensional view by storing data in special multidimensional data structures (MDDS)

Advantage: Fast indexing to pre-computed aggregations. Only values are stored.

Disadvantage: Not very scalable and sparse



HOLAP: Hybrid OLAP - combines ROLAP and MOLAP technology. (Scalability of ROLAP and faster computation of MOLAP)

Example of Fact and Dimension tables for ROLAP

- The dimensions of the fact table are further described with *dimension tables*

- Fact table:

Sales (*Market_id*, *Product_Id*, *Time_Id*, *Sales_Amt*)

- Dimension Tables:

Market (*Market_Id*, *City*, *Province*, *Region*)

Product (*Product_Id*, *Name*, *Category*, *Price*)

Time (*Time_Id*, *Week*, *Month*, *Quarter*)

Aggregation

- Many OLAP queries involve *aggregation* of the data in the fact table
- For example, to find the total sales (over time) of each product in each market, we might use


```
SELECT  S.Market_Id, S.Product_Id, SUM (S.Sales_Amt)
FROM    Sales S
GROUP BY S.Market_Id, S.Product_Id
```
- The aggregation is over the entire time dimension and thus produces a two-dimensional view of the data

Aggregation over Time

- The output of the previous query

		<i>Market_Id</i>			
		M1	M2	M3	M4
<i>Product_Id</i>	SUM(<i>Sales_Amt</i>)				
	P1	3003	1503	...	
	P2	6003	2402	...	
	P3	4503	3	...	
	P4	7503	7000	...	
	P5	

Drilling Down and Rolling Up

- Some dimension tables form an **aggregation hierarchy**
Market_Id → *City* → *Province* → *Region*
- Executing a series of queries that moves down a hierarchy (e.g., from aggregation over regions to that over provinces) is called **drilling down**
 - Requires the use of the fact table or information more specific than the requested aggregation (e.g., cities)
- Executing a series of queries that moves up the hierarchy (e.g., from provinces to regions) is called **rolling up**
 - Note: In a rollup, coarser aggregations can be computed using prior queries for finer aggregations

Drilling Down

- Drilling down on market: from *Region* to *Province*
Sales (*Market_Id*, *Product_Id*, *Time_Id*, *Sales_Amt*)
Market (*Market_Id*, *City*, *Province*, *Region*)
1.

```
SELECT  S.Product_Id, M.Region, SUM (S.Sales_Amt)
FROM    Sales S, Market M
WHERE   M.Market_Id = S.Market_Id
GROUP BY S.Product_Id, M.Region
```
 2.

```
SELECT  S.Product_Id, M.Province, SUM (S.Sales_Amt)
FROM    Sales S, Market M
WHERE   M.Market_Id = S.Market_Id
GROUP BY S.Product_Id, M.Province,
```

Rolling Up

- Rolling up on market, from *Province* to *Region*
 - If we have already created a table, *Province_Sales*, using
1.

```
SELECT  S.Product_Id, M.Province, SUM (S.Sales_Amt)
FROM    Sales S, Market M
WHERE   M.Market_Id = S.Market_Id
GROUP BY S.Product_Id, M.Province
```

then we can roll up from there to:
 2.

```
SELECT  T.Product_Id, M.Region, SUM (T.Sales_Amt)
FROM    Province_Sales T, Market M
WHERE   M.Province = T.Province
GROUP BY T.Product_Id, M.Region
```

Pivoting

- When we view the data as a multi-dimensional cube and group on a subset of the axes, we are said to be performing a **pivot** on those axes
 - Pivoting on dimensions D_1, \dots, D_k in a data cube $D_1, \dots, D_k, D_{k+1}, \dots, D_n$ means that we use **GROUP BY** A_1, \dots, A_k and aggregate over A_{k+1}, \dots, A_n , where A_i is an attribute of the dimension D_i
 - *Example*: Pivoting on **Product** and **Time** corresponds to grouping on *Product_id* and *Quarter* and aggregating *Sales_Amt* over *Market_id*:

```
SELECT  S.Product_Id, T.Quarter, SUM (S.Sales_Amt)
FROM    Sales S, Time T
WHERE   T.Time_Id = S.Time_Id
GROUP BY S.Product_Id, T.Quarter
```

Pivot

Slicing-and-Dicing

- When we use WHERE to specify a particular value for an axis (or several axes), we are performing a *slice*
 - Slicing the data cube in the Time dimension (choosing sales only in week 12) then pivoting to *Product_id* (aggregating over *Market_id*)

```
SELECT S.Product_Id, SUM (Sales_Amt)
FROM Sales S, Time T
WHERE T.Time_Id = S.Time_Id AND T.Week = 'Wk-12'
GROUP BY S.Product_Id
```

Slice

Pivot

Slicing-and-Dicing

- Typically slicing and dicing involves several queries to find the “right slice.”

For instance, change the slice and the axes:

- Slicing on Time and Market dimensions then pivoting to *Product_id* and *Week* (in the time dimension)

```
SELECT S.Product_Id, T.Quarter, SUM (Sales_Amt)
FROM Sales S, Time T
WHERE T.Time_Id = S.Time_Id
      AND T.Quarter = 4
      AND S.Market_Id = 12345
GROUP BY S.Product_Id, T.Week
```

Slice

Pivot

The CUBE Operator

- To construct the following table, would take 3 queries (next slide)

		Market_Id			
		M1	M2	M3	Total
Product_Id	SUM(Sales_Amt)				
	P1	3003	1503
	P2	6003	2402
	P3	4503	3
	P4	7503	7000
Total	

The Three Queries

- For the table entries, without the totals (aggregation on time)


```
SELECT S.Market_Id, S.Product_Id, SUM (S.Sales_Amt)
FROM Sales S
GROUP BY S.Market_Id, S.Product_Id
```
- For the row totals (aggregation on time and supermarkets)


```
SELECT S.Product_Id, SUM (S.Sales_Amt)
FROM Sales S
GROUP BY S.Product_Id
```
- For the column totals (aggregation on time and products)


```
SELECT S.Market_Id, SUM (S.Sales)
FROM Sales S
GROUP BY S.Market_Id
```

Definition of the CUBE Operator

- Doing these three queries is wasteful
 - The first does much of the work of the other two: if we could save that result and aggregate over *Market_Id* and *Product_Id*, we could compute the other queries more efficiently
- The CUBE clause is part of SQL:1999
 - GROUP BY CUBE (*v1*, *v2*, ..., *vn*)
 - Equivalent to a collection of GROUP BYs, one for each of the 2^n subsets of *v1*, *v2*, ..., *vn*

Example of CUBE Operator

- The following query returns all the information needed to make the previous products/markets table:

```
SELECT S.Market_Id, S.Product_Id, SUM (S.Sales_Amt)
FROM Sales S
GROUP BY CUBE (S.Market_Id, S.Product_Id)
```

ROLLUP

- ROLLUP is similar to CUBE except that instead of aggregating over all subsets of the arguments, it creates subsets moving from right to left
- GROUP BY ROLLUP (*A₁*, *A₂*, ..., *A_n*) is a series of these aggregations:
 - GROUP BY *A₁*, ..., *A_{n-1}*, *A_n*
 - GROUP BY *A₁*, ..., *A_{n-1}*
 -
 - GROUP BY *A₁*, *A₂*
 - GROUP BY *A₁*
 - No GROUP BY
- ROLLUP is also in SQL:1999

Example of ROLLUP Operator

```
SELECT S.Market_Id, S.Product_Id, SUM (S.Sales_Amt)
FROM Sales S
GROUP BY ROLLUP (S.Market_Id, S.Product_Id)
– first aggregates with the finest granularity:
  GROUP BY S.Market_Id, S.Product_Id
– then with the next level of granularity:
  GROUP BY S.Market_Id
– then the grand total is computed with no GROUP BY clause
```

ROLLUP vs. CUBE

- The same query with CUBE:
 - first aggregates with the finest granularity:
GROUP BY S.Market_Id, S.Product_Id
 - then with the next level of granularity:
GROUP BY S.Market_Id
 - and
GROUP BY S.Product_Id
 - then the grand total with *no* GROUP BY

Materialized Views

The CUBE operator is often used to pre-compute aggregations on all dimensions of a fact table and then save them as a *materialized views* to speed up future queries

Data Warehouse and OLAP



- What is a data warehouse and what is it for?
- What is the multi-dimensional data model?
- What is the difference between OLAP and OLTP?
- What is the general architecture of a data warehouse?
- How can we implement a data warehouse?
- Are there issues related to data cube technology?

Issues



- Scalability
- Sparseness
- Curse of dimensionality
- Materialization of the multidimensional data cube (total, virtual, partial)
- Efficient computation of aggregations
- Indexing