Part 14:

OLAP and Data Warehouse

Database System Concepts, 7th Ed.

OLAP

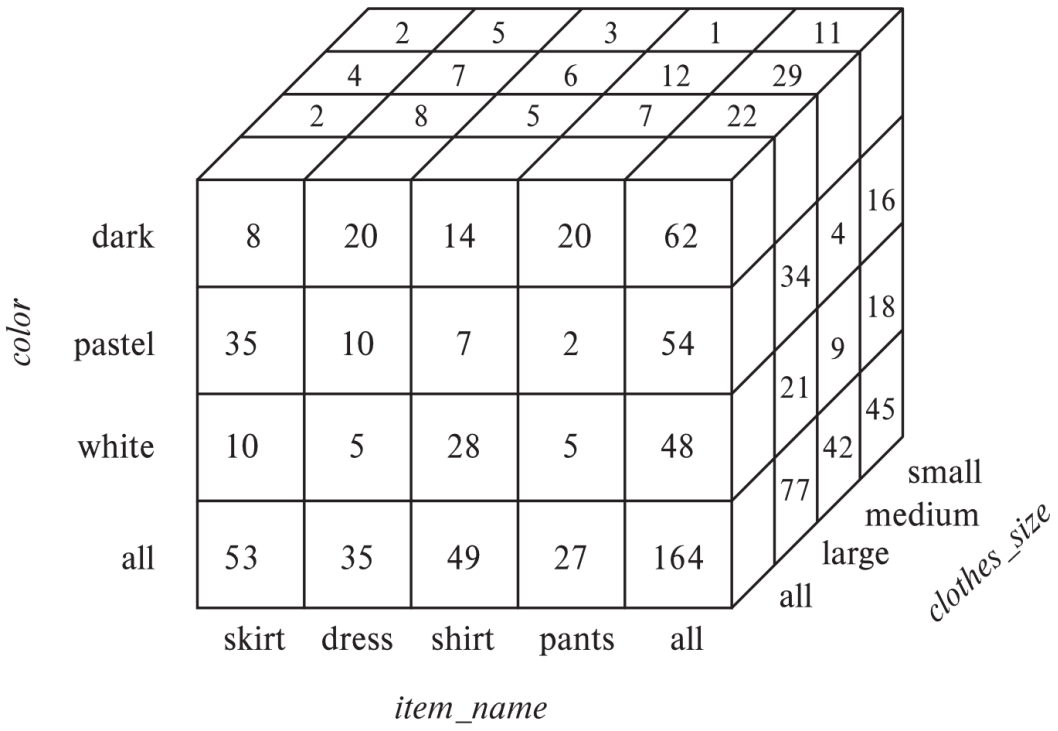
Data Analysis and OLAP

- **Online Analytical Processing (OLAP)**
  - Interactive analysis of data, allowing data to be summarized and viewed in different ways in an online fashion
- We use the **sales** relation to illustrate OLAP concepts
  - *sales* (*item_name*, *color*, *clothes_size*, *quantity*)

item_name	*color*	*clothes_size*	*quantity*
dress	dark	small	2
dress	dark	medium	6
dress	dark	large	12
dress	pastel	small	4
dress	pastel	medium	3
dress	pastel	large	3
dress	white	small	2
dress	white	medium	3
dress	white	large	0
pants	dark	small	14
pants	dark	medium	6
pants	dark	large	0
pants	pastel	small	1
pants	pastel	medium	0
pants	pastel	large	1
pants	white	small	3
pants	white	medium	0
pants	white	large	2
shirt	dark	small	2
shirt	dark	medium	6
shirt	dark	large	6
shirt	pastel	small	4
shirt	pastel	medium	1
shirt	pastel	large	2
shirt	white	small	17
shirt	white	medium	1
shirt	white	large	10
skirt	dark	small	2
skirt	dark	medium	5
skirt	dark	large	1
skirt	pastel	small	11
skirt	pastel	medium	9
skirt	pastel	large	15
skirt	white	small	2
skirt	white	medium	5
skirt	white	large	3

sales relation

Hypercube/Data Cube



A *conceptual view* of multi-dimensional information is a ***hypercube*** (also called ***data cube***)

- the operation of moving from finer-granularity data to coarser granularity is called **Rollup**
- the operation of moving from coarser-granularity data to finer granularity is called **Drill-down**
- fixing on a particular value of an attribute (e.g. year), one gets a slice from the cube – this is called **slicing**
- fixing on more values, one gets a dice from the cube – this is called **dicing**

OLAP Implementation

- The earliest OLAP systems used multidimensional arrays in memory to store data cubes, and are referred to as **multidimensional OLAP (MOLAP)** systems
- OLAP implementations using only relational database features are called **relational OLAP (ROLAP)** systems
- Hybrid systems, which store some summaries in memory and store the base data and other summaries in a relational database, are called **hybrid OLAP (HOLAP)** systems

OLAP IN SQL

Cube

- The **cube** operation computes union of **group by'**s on every subset of the specified attributes
- E.g., consider the query

  select *item_name, color, size,* **sum**(*quantity*)
  from *sales*
  group by cube(*item_name, color, size*)

  This computes the union of eight different groupings of the *sales* relation, exhausting all 2^3 combinations:

  { (*item_name, color, size*), (*item_name, color*),
  (*item_name, size*), (*color, size*),
  (*item_name*), (*color*),
  (*size*), ( ) }

- For each grouping, the result contains the value null (i.e., all) for attributes not present in the grouping, which is subtotaled
- The first (*item_name, color, size*) generates the original table, while ( ) subtotals over all attributes producing a grand total

Rollup

- The **rollup** generates selected subtotals

  select *item_name*, *color*, *size*, **sum**(*quantity*)
  from *sales*
  group by rollup(*item_name, color, size*)

  Generates the union of four groupings (dropping one attribute from the right each time):

  { (*item_name, color, size*), (*item_name, color*), (*item_name*), ( ) }

- For each grouping, the result contains the value null (i.e., all) for attributes not present in the grouping, which is subtotaled
- The first (*item_name, color, size*) generates the original table, while ( ) subtotals over all attributes producing a grand total

Rollup

- The **rollup** generates selected subtotals

  select *item_name*, *color*, **sum**(*quantity*)
  from *sales*
  group by rollup(*item_name, color*)

  Generates the union of three groupings (dropping one attribute from the right each time):

  { (*item_name, color*), (*item_name*), ( ) }

item_name	*color*	*quantity*
skirt	dark	8
skirt	pastel	35
skirt	white	10
dress	dark	20
dress	pastel	10
dress	white	5
shirt	dark	14
shirt	pastel	7
shirt	white	28
pants	dark	20
pants	pastel	2
pants	white	5
skirt	*null*	53
dress	*null*	35
shirt	*null*	49
pants	*null*	27
null	*null*	164

The first tuple sums over all sizes for dark skirt.
The second last tuple sums over all sizes and all colors for pants.
The last tuple sums over all sizes and all colors and all item_name, producing a grand total.

item_name	*color*	*clothes_size*	*quantity*
dress	dark	small	2
dress	dark	medium	6
dress	dark	large	12
dress	pastel	small	4
dress	pastel	medium	3
dress	pastel	large	3
dress	white	small	2
dress	white	medium	3
dress	white	large	0
pants	dark	small	14
pants	dark	medium	6
pants	dark	large	0
pants	pastel	small	1
pants	pastel	medium	0
pants	pastel	large	1
pants	white	small	3
pants	white	medium	0
pants	white	large	2
shirt	dark	small	2
shirt	dark	medium	6
shirt	dark	large	6
shirt	pastel	small	4
shirt	pastel	medium	1
shirt	pastel	large	2
shirt	white	small	17
shirt	white	medium	1
shirt	white	large	10
skirt	dark	small	2
skirt	dark	medium	5
skirt	dark	large	1
skirt	pastel	small	11
skirt	pastel	medium	9
skirt	pastel	large	15
skirt	white	small	2
skirt	white	medium	5
skirt	white	large	3

Top-N Queries

- Top-N queries ask for the *N* largest or smallest values of a column
  - What are the top ten best selling products in Canada?
  - What are the 10 worst selling products?
- Both largest-values and smallest-values sets are considered Top-N queries
- Top-N queries use a nested query structure with the following elements
  - Subquery to generate the sorted list of data
  - Outer Query to limit the number of rows in the final result set, which includes:
    - ROWNUM pseudo-column which assigns a sequential value starting with 1 to each of the rows returned from the subquery
    - WHERE clause used to specify the N returned rows

Top-N Queries

- To retrieve the *ID, GPA* of the top 10 students in order of GPA

```
select *
from (select ID, GPA
        from student_grades
        order by GPA desc)
where rownum <= 10;
```

- Depending on the implementation, other variants exist

```
select ID, GPA
from student_grades
order by GPA desc
limit 10;
```

or

```
select top 10 ID, GPA
from student_grades
order by GPA desc;
```

Other SQL OLAP Facilities

- **CUME_DIST** (cumulative distribution)
- **RANK, DENSE_RANK** (computes the rank)
- **NTILE** (percentile)
- **MOVING AVERAGE**
- **HISTOGRAM**

Data Warehouse

Database System Concepts, 7th Ed.

Data Warehouse

- Data warehouses are databases that store and maintain analytical data separately from transaction-oriented databases for the purpose of decision support
- **OLAP** (Online Analytical Processing) analyses complex data from the data warehouse
- **DSS** (Decision Support Systems) also known as **EIS** (Executive Information Systems) supports organization's leading decision makers for making complex and important decisions
- **BI** (Business intelligence) includes mechanisms for acquiring, storing, analyzing, and providing access to information to help users make sound business decisions
- **Data Mining** is used for knowledge discovery, the process of analyzing data to discover new knowledge and patterns

Operational vs Analytical Data

- Application oriented
- Detailed
- Accurate
- Small amount of data used in a process
- Serves the clerical community
- Frequently updated
- Run repetitively
- Transaction driven
- High availability
- Performance sensitive

- Subject oriented
- Summarized
- Represents values over time
- Large amount of data used in a process
- Serves the executive community
- Periodically refreshed
- Run heuristically
- Analysis driven
- Relaxed availability
- Performance relaxed

Extract, Transform, Load

- ETL (Extract, Transform, Load) is a data integration process that combines data from multiple data sources into a single, consistent data store that is loaded into the data warehouse
  - Extract data from operational systems and other sources
  - Cleanse the data to improve data quality and establish consistency
    - The resultant higher quality data may be used to enhance the source data (backflushing)
  - Load data into the data warehouse

Comparison with Traditional Databases

- Data warehouses are mainly optimized for data access
  - Traditional databases are transactional and are optimized for both transaction processing and integrity assurance
- Data warehouses emphasize more on historical data and support trend analysis
- In transactional databases, transaction is the mechanism of change to the database
- Information in data warehouse is relatively coarse grained and summarized
  - Data is periodically refreshed
  - Data in data warehouses are non-volatile

Characteristics of Data Warehouse

- A data warehouse is a
  - **subject-oriented**
  - **integrated**
  - **non-volatile**
  - **time variant**

  collection of data in support of management's decision

Subject Orientation

Operational

- Focus on applications areas of an organization
- Insurance Company
  - auto
  - life
  - health
  - accident

Data Warehouse

- Focus on major subject areas of an organization
- Insurance Company
  - customer
  - policy
  - premium
  - claim

Integration

- Converting data from several operational databases to data warehouse
- Issues
  - data encoding
  - attribute measurement
  - multiple sources
  - conflicting keys

Non-Volatility

Operational

- Highly changeable
- Lots of updates
- Record by record manipulation

Data Warehouse

- No update
- Mass load and access
- Refreshing the data warehouse

Time Variancy

Operational

- Time-horizon: current to 60-90 days
- Accurate as of moment of access

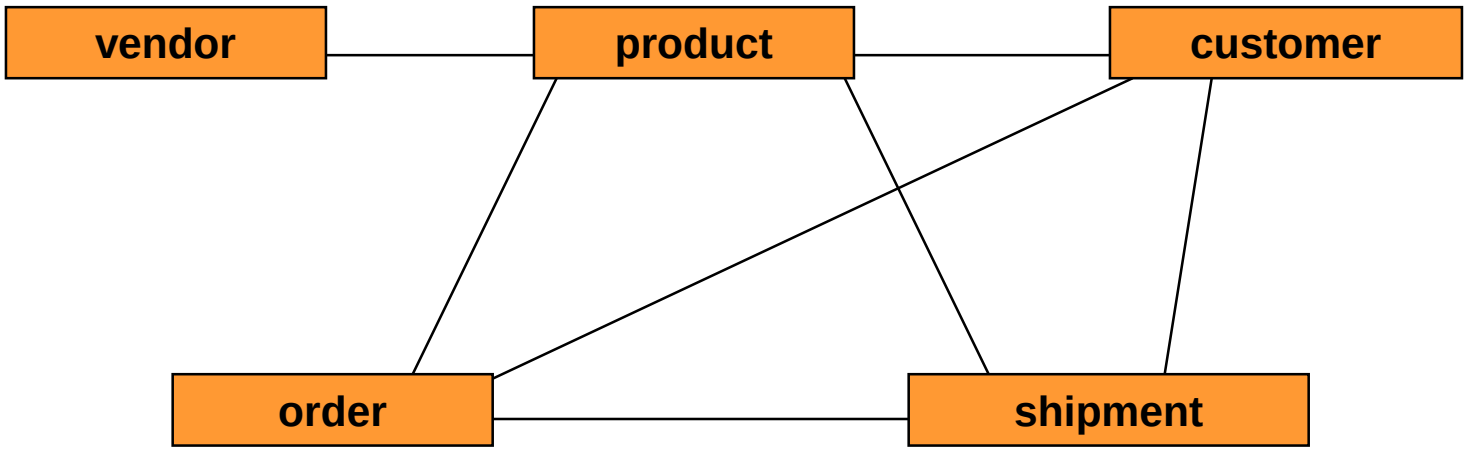
Data Warehouse

- Time-horizon: 5-10 years
- Sophisticated snapshots of data

Data Warehouse Schemas

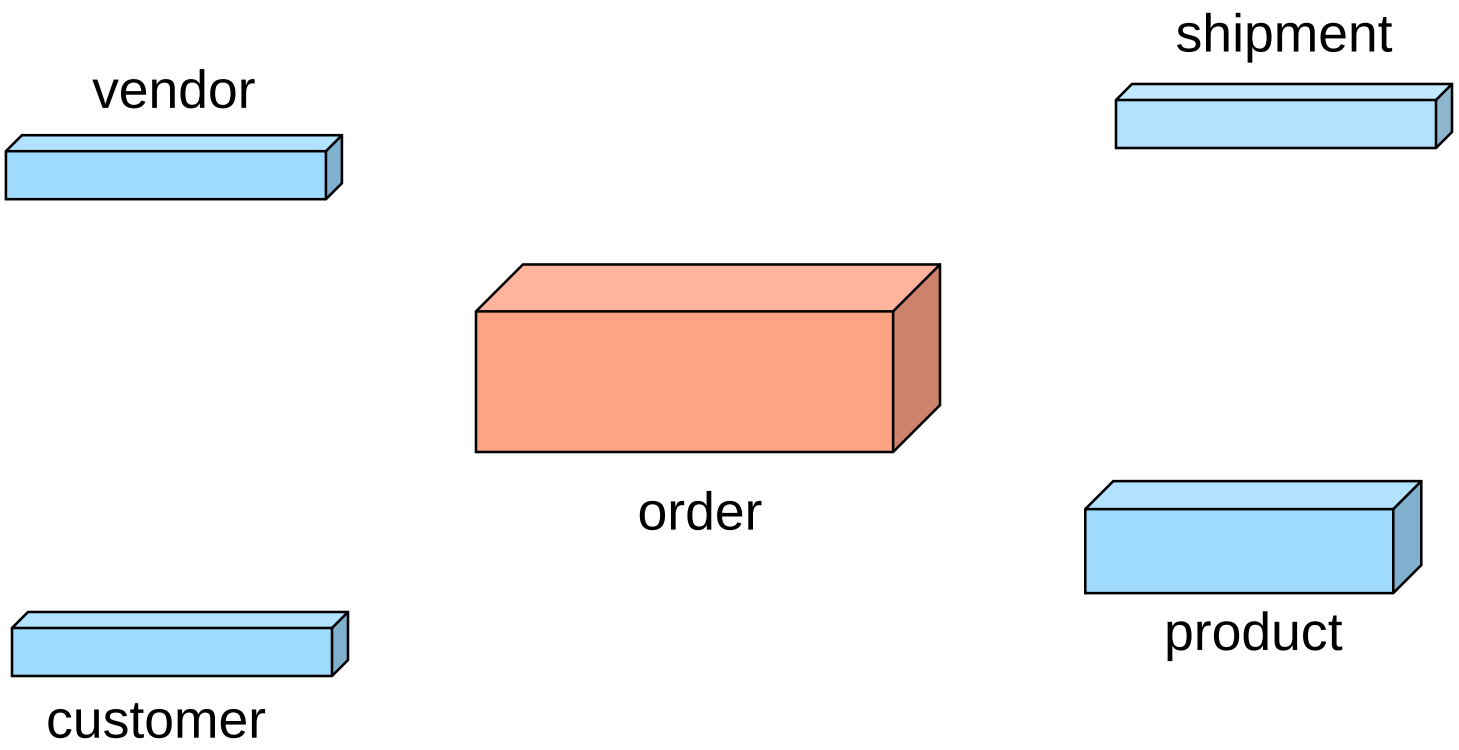
- Data Warehouse often uses a **star schema,** or sometimes a **snowflake schema**
  - fact table joined with dimension tables
  - group-by on dimension table attributes
  - aggregation on measure attributes of fact table
- Some applications do not find it worthwhile to bring data to a common schema
  - **Data marts** are generally focused on a subset of the organization, such as a single subject area or department
  - **Data lakes** are repositories which allow data to be stored in multiple formats, without schema integration

Limitations of Entity Relationship Modelling

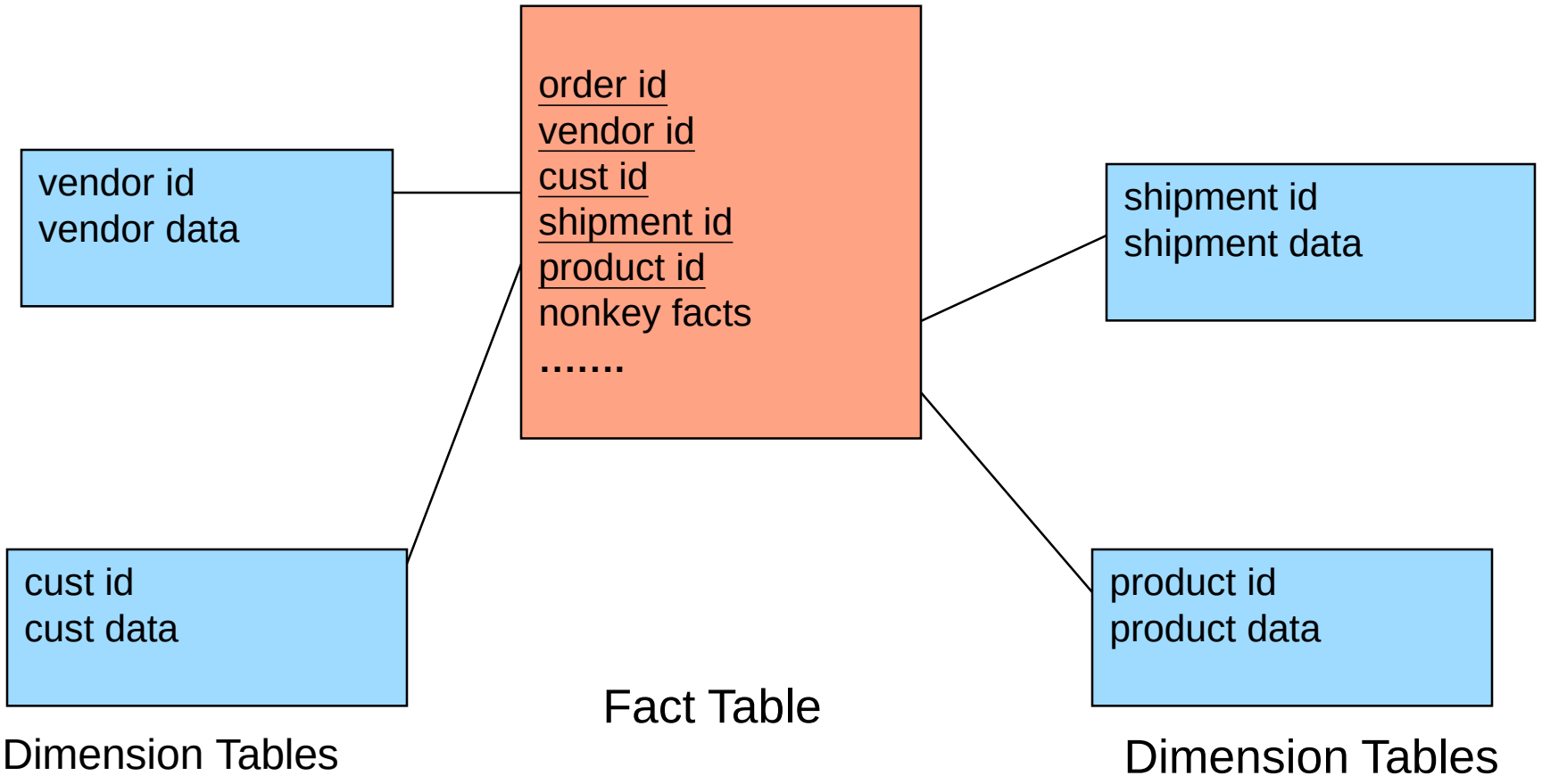


- Very symmetric
- Cannot tell which table is most important or largest
- Cannot tell which tables hold static or dynamic business information
- Joining of any tables is possible by user

A 3-D Perspective: Orders are Much More Numerous



Star Schema: Fact and Dimension Tables



Star Schema

- The centre of the star consists of a fact table and the points of the star are the dimension tables
- A star schema is characterized by very large *fact* tables that contain the primary information in the data warehouse, and a number of much smaller dimension tables, each of which contains information about the entries for a particular attribute in the fact table
- Fact table tends to contain additive facts and have composite keys
- Fact table is naturally highly normalized
- Each dimension table is joined to the fact table using a primary-key to foreign-key join, but the dimension tables are not joined to each other
- Dimension table tends to contain textual or non-additive information
- Dimension tables should not be normalized
- Normalized dimension tables destroy the ability to browse (can lead to the undesirable snowflake schema)

Snowflake Schema

- **Snowflake Schema is Undesirable**
- Snowflake schemas normalize dimensions to eliminate redundancy, and makes the star schema look like a snowflake
- For example, a product dimension table in a star schema might be normalized into a Product table, a Product_Category table, and a Product_Manufacturer table in a snowflake schema
  - A given product may be a small household product but is encoded as product Category 5 in the Product table, and the separate Product_Category table would also give other information such as weight, colour, safety to children etc. (and a foreign key join on category_id is necessary to obtain such information)
  - The manufacturer information may be stored in a separate table giving details of the manufacturer (and a foreign key join on manufacturer_id is necessary to obtain the full manufacturer information)
- While this saves space, it increases the number of dimension tables and requires more foreign key joins