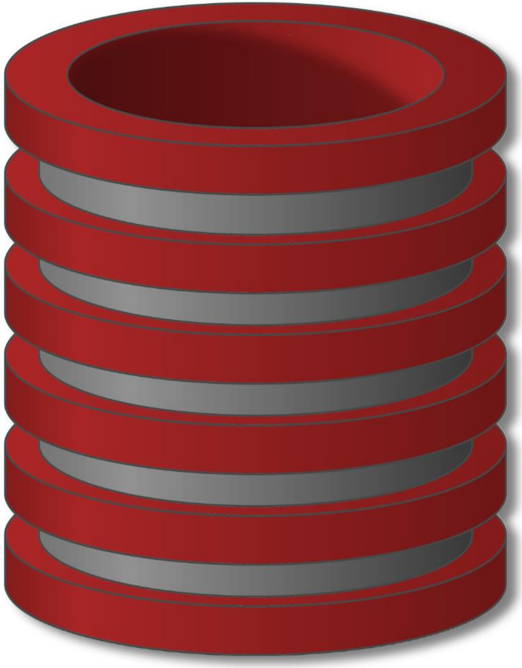


# On-Line Analytical Processing (OLAP)

## Introduction

Jennifer Widom

# Two broad types of database activity

- OLTP – Online Transaction Processing
  - Short transactions
  - Simple queries
  - Touch small portions of data
  - Frequent updates
- OLAP – Online Analytical Processing
  - Long transactions
  - Complex queries
  - Touch large portions of the data
  - Infrequent updates

# OLAP Vs. OLTP

**OLTP (On Line Transaction Processing)**

```
Select tx_date, balance from tx_table
Where account_ID = 23876;
```


Jennifer Widom

# OLAP Vs. OLTP

## OLAP

Select balance, age, sal, gender from
customer_table, tx_table
Where age between (30 and 40) and
Education = ‘graduate’ and
CustID.customer_table = Customer_ID.tx_table;

# Why a Data Warehouse?

## DBMS Approach

List of all items that were sold last month?

List of all items purchased by a customer?

The total sales of the last month grouped by branch?

How many sales transactions occurred during the month of January?

# Why a Data Warehouse?

## Intelligent Enterprise

Which items sell together? Which items to stock?

Where and how to place the items? What discounts to offer?

How best to target customers to increase sales at a branch?

Which customers are most likely to respond to my next promotional campaign, and why?

# Why a Data Warehouse?

- Businesses want much more…
  - What happened?
  - Why it happened?
  - What will happen?
  - What is happening?
  - What do you want to happen?

**Stages of Data Warehouse**

# What is a Data Warehouse?

A *complete repository* of *historical* corporate data extracted from *transaction systems* that is available for *ad-hoc* access by *knowledge workers.*

# What is a Data Warehouse?

- **Complete repository**
  - All the data is present from all the branches/outlets of the business.
  - Even the archived data may be brought online.
- **Transaction System**
  - Management Information System (MIS)
  - Could be typed sheets (NOT transaction system)
- **Ad-Hoc access**
  - Dose not have a certain access pattern
  - Queries not known in advance
  - Difficult to write SQL in advance
- **Knowledge workers**
  - Typically NOT IT literate (Executives, Analysts, Managers).
  - NOT clerical workers.
  - Decision makers

# More terminology

- Data warehousing

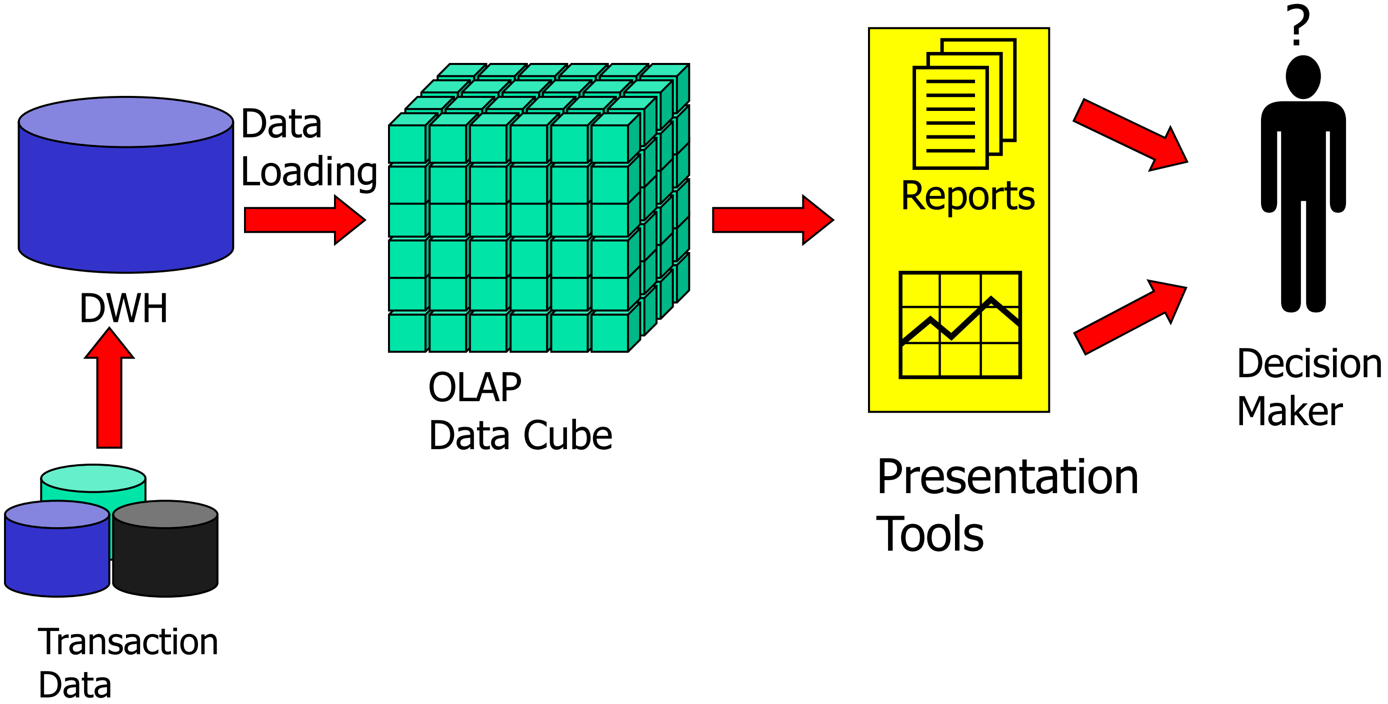
  Bring data from operational (OLTP) sources into a single "warehouse" for (OLAP) analysis

- Decision support system (DSS)

  Infrastructure for data analysis

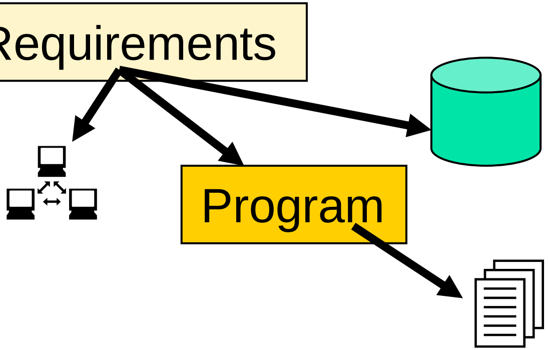
  E.g., data warehouse tuned for OLAP

# Where does OLAP fit in?



# How DW is Different?

- Does not follows the traditional development model

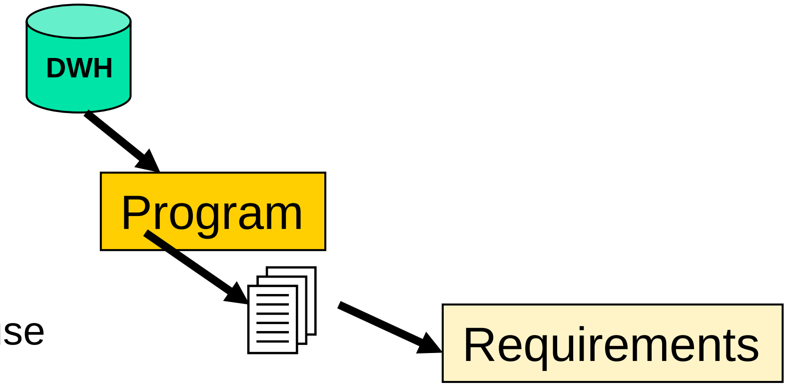



**Classical SDLC**

- Requirements gathering
- Analysis
- Design
- Programming
- Testing
- Integration
- Implementation

# How DW Different?

- **Does not follows the traditional development model**




**DWH SDLC (CLDS)**

- Implement warehouse
- Integrate data
- Test for biasness
- Program w.r.t data
- Design DSS system
- Analyze results
- Understand requirement

# Modeling Technique

- The entity-relationship data model is commonly used in the design of relational databases:
  - Where a database schema consists of a set of entities and the relationships between them. Such a data model is appropriate for online transaction processing.
- A data warehouse, requires a concise, subject-oriented schema that facilitates online data analysis.
  - The most popular data model for a data warehouse is a **dimensional model.**

# What is Dimensional Modeling

A simpler **logical model** optimized for decision support systems.

Inherently dimensional in nature, with a single central **fact table** and a set of smaller **dimensional tables**.

Results in a **star like** structure, called **star schema or star join.**

- All relationships mandatory 1-M.
- Single path between any two levels.

- Supports **ROLAP operations**.

# OLAP: Facts & Dimensions

- The foundation for OLAP is **dimensional modeling** techniques which focus on the concepts of "facts" and "dimensions" for organizing data.

  - **FACTS:** Quantitative values (numbers) or "measures."
    - e.g., units sold, sales $, C$^{o}$, Kg etc.

  - **DIMENSIONS:** Perspectives or entities with respect to which an organization wants to keep records.
    - Descriptive categories.
    - e.g., time, geography, product etc.

    - DIM often organized in **hierarchies** representing levels of detail in the data
      - e.g., week, month, quarter, year, decade etc.

# “Star Schema”

- Fact table

  Contains measurements, metrics, and facts about a business process

  Updated frequently, often append-only, very large

- Dimension tables

  Contains detailed data to be used constraining queries for the fact table

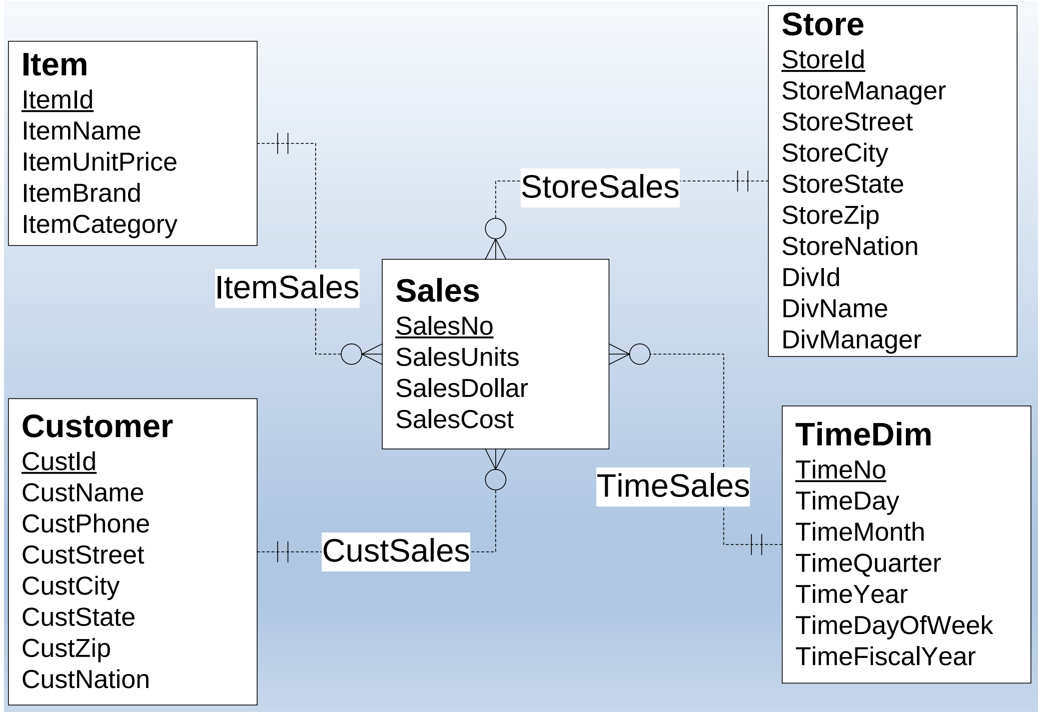
  Updated infrequently, not as large

Fact

“We need monthly sales
volume and price by
week, product and Zone”

Dimensions

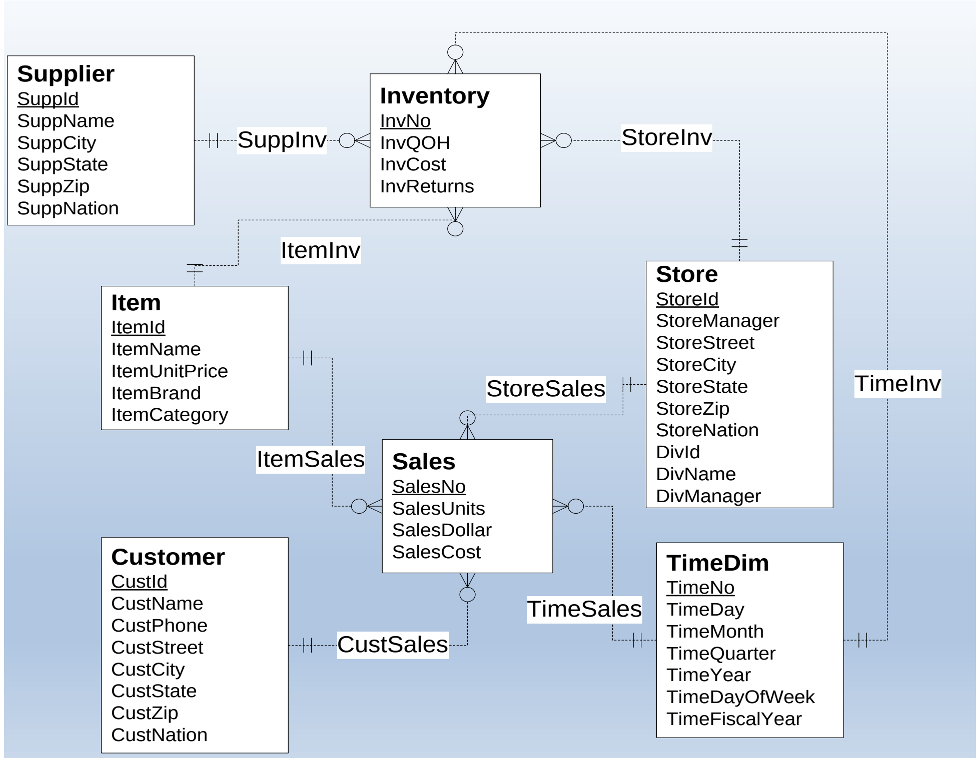
# Star Schema Example




Star schema:

- One fact table in the center
- Multiple dimension tables
- Represents one data cube
- DW may contain many star schemas

# Constellation Schema Example

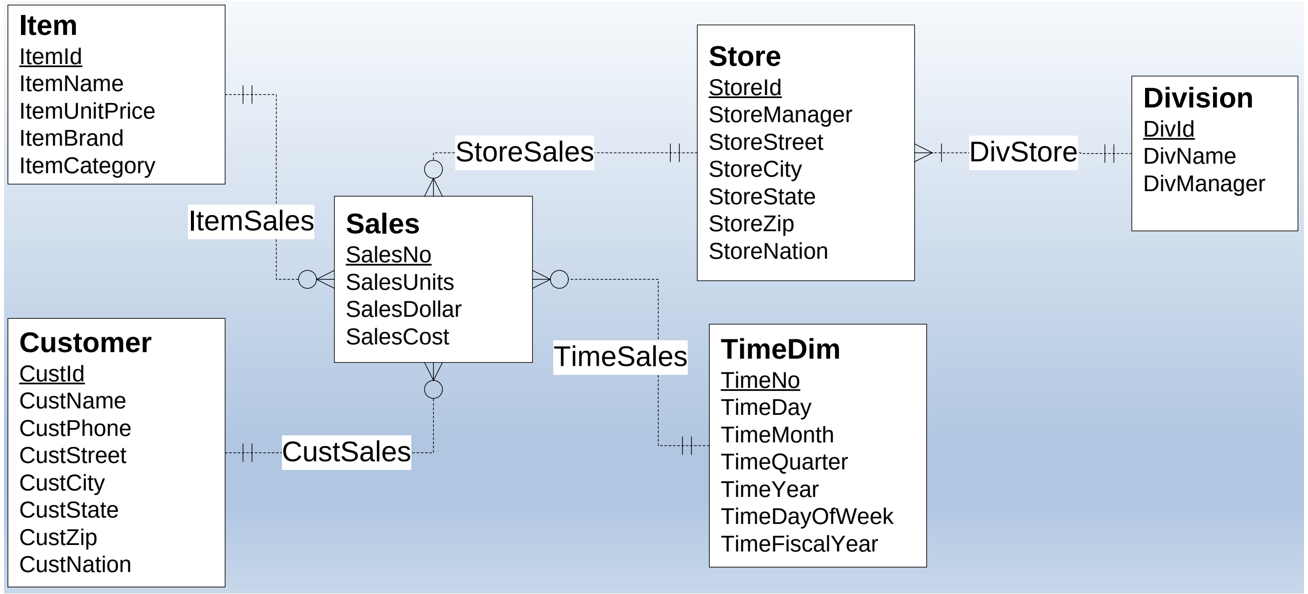



Constellation schema:

- Multiple fact tables
- Dimension tables share fact tables
- Relationship diagram looks like a constellation

Jennifer Widom

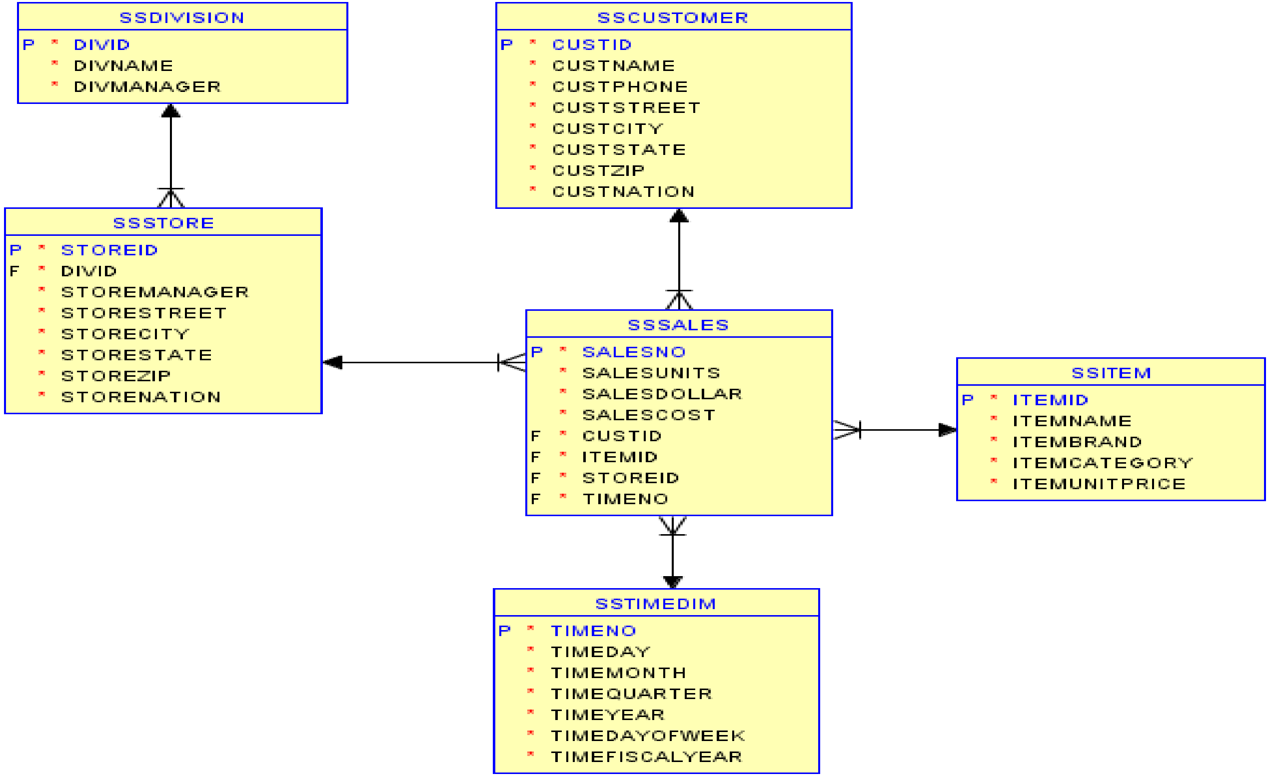
# Snowflake Schema Example




Snowflake schema:

- Multiple levels of dimension tables
- Use when dimension tables are small: little performance gain by deformalizing
- Relationship diagram looks like a snowflake

# Oracle Diagram for the Store Sales DW



# Star Schema – fact table references dimension tables

```
Sales(storeID, itemID, custID, qty, price)
Store(storeID, city, state)
Item(itemID, category, brand, color, size)
Customer(custID, name, address)
TimeDim(Timeday, timeMonth,…)
```

# OLAP queries

```
Sales(storeID, itemID, custID, qty, price)
Store(storeID, city, state)
Item(itemID, category, brand, color, size)
Customer(custID, name, address)
```

Join → Filter → Group → Aggregate

## Performance

– Inherently very slow:
  special indexes, query processing techniques

# Data Cube (a.k.a. multidimensional OLAP)

- Complete set of subtotals

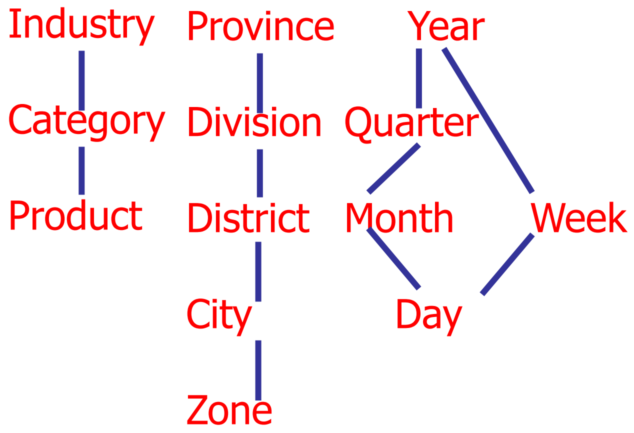
- Dimension data forms axes of “cube”
- Fact (dependent) data in cells
- Aggregated data on sides, edges, corner

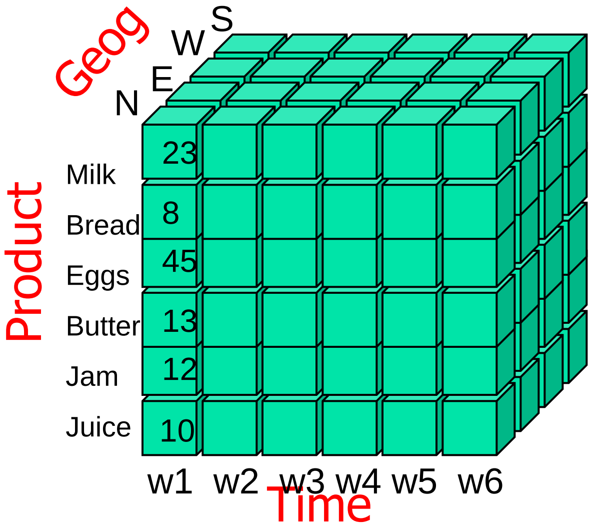
# Aggregations in MOLAP

- Sales volume as a function of (i) product, (ii) time, and (iii) geography
- A cube structure created to handle this.

Dimensions: Product, Geography, Time

Hierarchical summarization paths



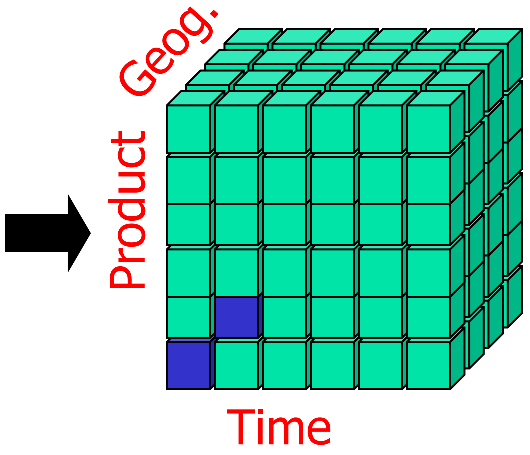



# ROLAP as a "Cube"

- OLAP data is stored in a relational database (e.g. a star schema)
- The **fact table** is a way of *visualizing as* a "un-rolled" cube.
- So where is the cube?
  - It's a matter of perception
  - Visualize the fact table as an elementary cube.

Fact Table

| Month | Product | Zone | Sale K Rs. |
|---|---|---|---|
| M1 | P1 | Z1 | 250 |
| M2 | P2 | Z1 | 500 |



# How to create "Cube" in ROLAP

- Cube is a logical entity containing values of a certain fact at a certain aggregation level at an intersection of a combination of dimensions.
- The following table can be created using 3 queries

Month_ID

Product_ID

| SUM (Sales_Amt) | M1 | M2 | M3 | **ALL** |
|---|---|---|---|---|
| P1 | | | | |
| P2 | | | | |
| P3 | | | | |
| **Total** | | | | |

# How to create "Cube" in ROLAP using SQL

- For the table entries, without the totals

```
SELECT      S.Month_Id, S.Product_Id,
            SUM(S.Sales_Amt)
FROM        Sales
GROUP BY    S.Month_Id, S.Product_Id;
```

- For the row totals

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

- For the column totals

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

# CUBE / GROUP BY Comparison

SELECT State, Month, SUM(Sales)
GROUP BY CUBE(State, Month)

| State | Month | SUM(Sales) |
|---|---|---|
| CA | Dec | 100 |
| CA | Feb | 75 |
| CO | Dec | 150 |
| CO | Jan | 100 |
| CO | Feb | 200 |
| CN | Dec | 50 |
| CN | Jan | 75 |
| CA | - | 175 |
| CO | - | 450 |
| CN | - | 125 |
| - | Dec | 300 |
| - | Jan | 175 |
| - | Feb | 275 |
| - | - | 750 |

SELECT State, Month, SUM(Sales)
GROUP BY State, Month

| State | Month | SUM(Sales) |
|---|---|---|
| CA | Dec | 100 |
| CA | Feb | 75 |
| CO | Dec | 150 |
| CO | Jan | 100 |
| CO | Feb | 200 |
| CN | Dec | 50 |
| CN | Jan | 75 |

The CUBE operator clause produces all possible subtotal combinations in addition to the normal totals shown in a GROUP BY clause.

# CUBE / GROUP BY Comparison

```
Select CUSTID, STOREID,
SUM(SALESDOLLAR)
from SSSALES
group by CUSTID, STOREID;
```

```
Select CUSTID, STOREID,
SUM(SALESDOLLAR)
from SSSALES
group by CUBE (CUSTID, STOREID)
order by CUSTID, STOREID;
```

# CUBE Example

- Summarize (sum, min, and count) store sales for USA and Canada in 2016 by store zip code and month
- Generate all possible subtotals by zip code and month

```
SELECT StoreZip, TimeMonth, SUM(SalesDollar) AS
SumSales,
       MIN(SalesDollar) AS MinSales, COUNT(*) AS
RowCount
 FROM SSSales, SSStore, SSTimeDim
 WHERE SSSales.StoreId = SSStore.StoreId
   AND SSSales.TimeNo = SSTimeDim.TimeNo
   AND (StoreNation = 'USA' OR StoreNation =
'Canada')
   AND TimeYear = 2016
 GROUP BY CUBE (StoreZip, TimeMonth)
 ORDER BY StoreZip, TimeMonth;
```

# CUBE Operator Calculations

- GROUP BY CUBE(Col1, Col2)
  - $M$ unique values in Col1
  - $N$ unique values in Col2
- Result rows
  - Maximum of $M \times N$ rows: GROUP BY Col1, Col2
  - Maximum subtotal rows of $M + N + 1$ (CUBE)
- Subtotal groups
  - Three groups of subtotal rows (Col1, Col2, grand total)

# Cube operations

- **Rollup:** summarize data
  - e.g., given sales data, summarize sales for last year by product category and region
- **Drill down:** get more details
  - e.g., given summarized sales as above, find breakup of sales by city within each region
- **Slice and dice:** select and project
  - e.g.: Sales of soft-drinks in any city during last quarter

# ROLLUP Operator Characteristics

- Partial set of subtotals
- Appropriate for hierarchical dimensions
- Order dependent, coarsest to finest

# ROLLUP/GROUP BY Comparison

SELECT Year, Month, SUM(Sales)
GROUP BY ROLLUP(Year, Month)

| Year | Month | SUM(Sales) |
|---|---|---|
| 2016 | Jan | 100 |
| 2016 | Feb | 75 |
| 2016 | Mar | 150 |
| 2017 | Jan | 100 |
| 2017 | Feb | 200 |
| 2017 | Mar | 50 |
| 2016 | - | 325 |
| 2017 | - | 350 |
| - | - | 675 |

SELECT Year, Month, SUM(Sales)
GROUP BY Year, Month

| Year | Month | SUM(Sales) |
|---|---|---|
| 2016 | Jan | 100 |
| 2016 | Feb | 75 |
| 2016 | Mar | 150 |
| 2017 | Jan | 100 |
| 2017 | Feb | 200 |
| 2017 | Mar | 50 |

# ROLLUP Example

- Summarize (SUM, COUNT, and MIN) store sales for USA and Canada between 2016 and 2017 by year and month
- Generate partial subtotals for year and month

```
SELECT TimeYear, TimeMonth, SUM(SalesDollar) AS SumSales,
       MIN(SalesDollar) AS MinSales, COUNT(*) AS RowCount
 FROM SSSales, SSStore, SSTimeDim
 WHERE SSSales.StoreId = SSStore.StoreId
   AND SSSales.TimeNo = SSTimeDim.TimeNo
   AND (StoreNation = 'USA' OR StoreNation = 'Canada')
   AND TimeYear BETWEEN 2016 AND 2017
 GROUP BY ROLLUP(TimeYear, TimeMonth)
 ORDER BY TimeYear, TimeMonth;
```

# ROLLUP Calculations

- Two grouping columns
  - $N$ distinct values in outer most column
  - Maximum subtotal rows: $N + 1$
- Two grouping columns
  - ROLLUP (Col1, Col2) where Col1 has $N$ distinct values, Col2 has $M$ distinct values
  - Maximum subtotal rows: $N \times M + N + 1$
- $k$+1 subtotal groups for $k$ columns

# Drill-down

Examining summary data, break out by dimension attribute

```
Select CUSTID, STOREID, ITEMID, SUM(SALESDOLLAR)
from SSSALES
group by CUSTID, STOREID, ITEMID;
```

# Slicing

Analyze a slice of the cube, it does that by constraining one of the dimensions.

```
Select F.STOREID, ITEMID, CUSTID, SUM(SALESDOLLAR)
from SSSALES F, SSSTORE S
WHERE F.STOREID=S.STOREID and STORESTATE='CO'
group by F.STOREID,ITEMID,CUSTID;
```

# Dicing

Project slice of a cube in more than one dimension, it does that by constraining two or more dimensions of a cube and display chunk of the cube.

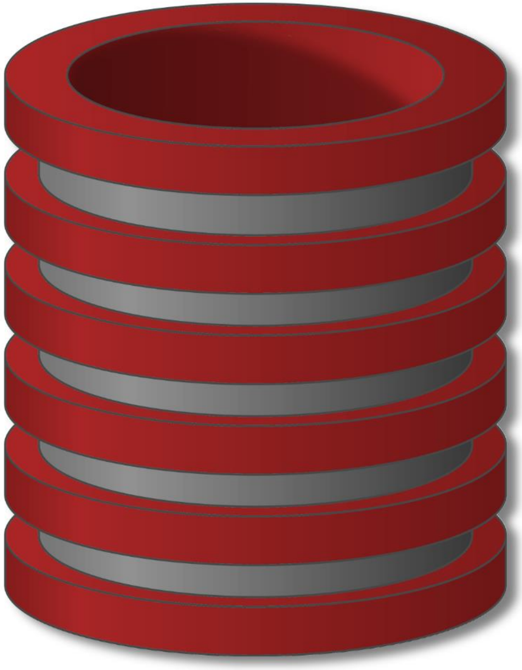
```
select F.STOREID, I.ITEMID, CUSTID, sum(SALESDOLLAR)
from SSSALES F, SSSTORE S, SSITEM I
where F.STOREID = S.STOREID and F.ITEMID = I.ITEMID
and STORESTATE = 'CO' and ITEMCATEGORY = 'Printing'
group by F.STOREID, I.ITEMID, CUSTID;
```

# Two broad types of database activity

- OLTP – Online Transaction Processing
  - Short transactions
  - Simple queries
  - Touch small portions of data
  - Frequent updates
- OLAP – Online Analytical Processing
  - Long transactions
  - Complex queries
  - Touch large portions of the data
  - Infrequent updates

# Two broad types of database activity

- OLTP – Online Transaction Processing
  - Short transactions
  - Simple queries
  - Touch small portions of data
  - Frequent updates
- OLAP – Online Analytical Processing
  - Star schemas
  - Data cubes
  - `Cube` and `Rollup`
  - Special indexes and query processing techniques



# On-Line Analytical Processing (OLAP)

## Demonstration

Jennifer Widom

# SQL Constructs for OLAP Operations

```
Select dimension-attrs, aggregates
From tables
Where conditions
Group By dimension-attrs
```

# SQL Constructs

## Cube and Rollup

```
Select dimension-attrs, aggregates
From tables
Where conditions
Group By CUBE/ROLL UP dimension-attrs
```

Add to result: faces, edges, and corner of cube using NULL values

- Star Schema
  - Fact table
  - Dimension tables
- OLAP Queries
  - Star join
  - `Cube` and `Rollup`
  - Drill-down and roll-up
  - “Slice” and “dice”