

COMP37332

**Introduction to
On-line analytical processing
(OLAP)**

Goran Nenadic
School of Computer Science

1

Aims

- Understand basic principles of OLAP
- Learn typical OLAP functionalities
- Understand the differences between OLAP and OLTP
- Review SQL extensions to support OLAP

2

Plan

- OLAP overview
- OLAP operations
- SQL extensions for OLAP
- Examples of commercial products
- Problems & challenges

- **Lab 1:** 9 March 2010, 13-14 (3rd year Lab)

3

Data warehouse applications

- Online analytical processing (OLAP)
- Decision support (DSS) and executive information systems (EIS)
- Data mining (DM)

Recall from the previous Lecture

4

OLAP

- Term coined in mid-1990's
- Main goal: support **ad-hoc** but **complex** querying performed by business analysts
- OLAP = interactive process of creating, managing, analysing and reporting on data
- Extends spreadsheet-like analysis to work with huge amounts of data in a data warehouse

5

continued

OLAP

- Data exploration & aggregation in various ways
- Typical applications include accessing the effectiveness of a marketing campaign, product sales forecasting (predictive analysis) and capacity planning
- Also, spot trends, pinpoint problems, perform "what-if" modelling
- Allows a sophisticated user to analyse data using complex, multi-dimensional views

6

Typical OLAP queries

- Write a multi-table join to compare sales for each product line year-to-date (YTD) this year vs. last year.
- Repeat the above process to find the top 5 product contributors to margin.
- Repeat the above process to find the sales of a product line to new vs. existing customers.
- Repeat the above process to find the customers that have had negative sales growth.

7

OLAP and data warehouses

- Place *key performance indicators* (measures) into context (*dimensions*)
 - measures are pre-aggregated
 - data retrieval is significantly faster
 - modelled in a DW
- The processed “cube” is made available to business analysts who can browse the data using a variety of visualisation tools, making ad hoc interactive and analytical processing/querying

8

Measures and dimensions

- Measures: key performance indicators that you would like to evaluate
 - typically numerical, e.g. volume, sales, and cost
 - a rule of thumb: if a number makes (business) sense when aggregated, then it is a measure
 - examples:
 - aggregate daily volume to month, quarter and year
 - aggregating telephone numbers would not make sense; therefore, telephone numbers are not measures
 - postcode: not a measure, but can be a dimension
 - [determines what should be stored/modelled in DW]

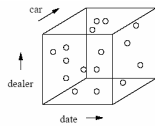
9

Measures and dimensions

- Dimensions: categories used in data analysis
 - typical dimensions include product, time, region
 - a rule of thumb: when a report is requested "by" something, that something is usually a dimension
 - e.g. in sales report: view sales by *month*, by *region*, so the two dimensions needed are time and region
 - dimensions = "bys"

DW schema

- Dimensions and measures are physically represented by a star schema (typically)
 - arrange the dimension tables around a central fact table that contains the measures
 - a fact table contains a column for each measure as well as a column for each dimension
- Build a cube (visualisation)



Example

		2	5	3	1	11	
		4	7	6	12	29	
		2	8	5	7	22	
dark	8	20	14	20	62	34	16
pastel	35	10	7	2	54	9	18
white	10	8	28	5	48	21	45
all	53	35	49	27	164	77	all
		skirt	dress	shirts	pant	all	small
							medium
							large
							size

OLAP operations

13

OLAP operations

- Include support for complex Boolean conditions, statistical functions and time-series analysis
- Based on the concept of multi-dimensional data model (hyper-cube)*

*recall the previous Lecture

14

Working example (1)

- Dimension tables
 - Market (Market_ID, City, Region)
 - Product (Product_ID, Name, Category, Price)
 - Time (Time_ID, Week, Month, Quarter)
- Fact table
 - Sales (Market_ID, Product_ID, Time_ID, Amount)

15

Operations in OLAP

- A very common operation is aggregating a measure over one of more dimensions
 - e.g. find the total sales (over time) of each product in each market
- ```
SELECT Market_ID, Product_ID, SUM (Amount)
FROM Sales
GROUP BY Market_ID, Product_ID
```

■ Output

| Market_ID | Product_ID | SUM(Amount) |
|-----------|------------|-------------|
| M1        | P1         | 500         |
| M1        | P2         | 200         |
| M2        | P1         | 250         |

16

---

---

---

---

---

---

---

---

## Operations in OLAP: roll-up

- **Roll-up:** specific grouping on one dimension where we go from a lower level of aggregation to a higher
  - e.g. summing-up sales by month and then fiscal year
  - e.g. summarisation over aggregate hierarchy (total sales by region, by state, by continent)
- Gradually coarser aggregations
- Can include multiple dimensions
  - e.g. roll-up the sales data (already aggregated on city) additionally by product

17

---

---

---

---

---

---

---

---

example

## Operations in OLAP: roll-up

| <i>category</i> |          | <i>item-name</i> |        |       |       |     |
|-----------------|----------|------------------|--------|-------|-------|-----|
|                 |          | dark             | pastel | white | total |     |
| womenswear      | skirt    | 8                | 8      | 10    | 53    |     |
|                 | dress    | 20               | 20     | 5     | 35    |     |
|                 | subtotal | 28               | 28     | 15    |       | 88  |
| menswear        | pants    | 14               | 14     | 28    | 49    |     |
|                 | shirt    | 20               | 20     | 5     | 27    |     |
|                 | subtotal | 34               | 34     | 33    |       | 76  |
| total           |          | 62               | 62     | 48    |       | 164 |

18

---

---

---

---

---

---

---

---

example

## Operations in OLAP: roll-up

| Product_ID | Market_ID | SUM(Amount) |
|------------|-----------|-------------|
| P1         | M1        | 500         |
| P1         | M2        | 200         |
| P1         |           | 700         |
| P2         | M2        | 250         |
| P2         |           | 250         |

roll-up sales on product

19

---

---

---

---

---

---

---

---

example

## Operations in OLAP: roll-up

roll up sales on market, from city to region

| Product_ID | City       | SUM(Amount) |
|------------|------------|-------------|
| P1         | Manchester | 100         |
| P1         | Salford    | 250         |
| P1         | North-West | 700         |
| P1         | Hull       | 100         |
| P1         | Newcastle  | 200         |
| P1         | North-East | 300         |
| P2         | Manchester | 200         |
| P2         | North-West | 200         |
| P2         | Newcastle  | 100         |
| P2         | North-East | 100         |
| ...        |            | ...         |

20

---

---

---

---

---

---

---

---

continued

## Operations in OLAP: roll-up

- Doable in SQL, but not in a single SQL statement

```
SELECT S.Product_ID, M.City, SUM(S.Amount)
 INTO City_Sales
 FROM Sales S, Market M
 WHERE
 M.Market_ID = S.Market_ID
 GROUP BY S.Product_ID, M.City
```

```
SELECT T.Product_ID, M.Region, SUM(T.Amount)
 FROM City_Sales T, Market M
 WHERE T.City = M.City
 GROUP BY T.Product_ID, M.Region
```

21

---

---

---

---

---

---

---

---

## Operations in OLAP

- **Drill-down:** finer-grained view on aggregated data, i.e. going from higher to lower aggregation
  - converse of roll-up
  - e.g. disaggregate county sales by region/city
- **Pivoting** (or rotation)
  - select a different dimension (orientation) for analysis
  - e.g. from a presentation of sales by year and state, move to a presentation of sales by product and year

22

---

---

---

---

---

---

---

---

## Roll-up and drill down

Higher Level of Aggregation



23

---

---

---

---

---

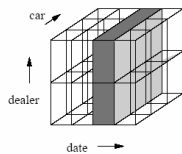
---

---

---

## Operations in OLAP: slice/dice

- **Projection operations**
  - typically define a sub-cube
  - **slicing:** selection on one or more dimensions, possibly with some dimensions projected out
  - **dicing:** a range selection in a hypercube (partition or group on one or more dimensions e.g., dealers by state and cars by colour)



24

---

---

---

---

---

---

---

---

## Example: slicing with SQL

- Slicing the data cube in the *time* dimension, e.g. choosing sales only in week 12

```
SELECT S.*
FROM Sales S, Time T
WHERE T.Time_ID = S.Time_ID AND T.Week = 'Week12'
```

- Slicing: use WHERE to specify a particular value for an axis (or several axes)

```
SELECT S.Product_ID, SUM (Amount)
FROM Sales S, Time T
WHERE T.Time_ID = S.Time_ID AND T.Week = 'Week12'
GROUP BY S.Product_ID
```

aggregate over sales per product, pivoting on Product\_ID

25

---

---

---

---

---

---

---

---

## Example: dicing with SQL

- **Dicing** sales in the *time* dimension: total sales for each product *in each quarter*

```
SELECT S.Product_ID, T.Quarter, SUM(S.Amount)
FROM Sales S, Time T
WHERE T.Time_ID = S.Time_ID
GROUP BY T.Quarter, S.Product_ID
```

- GROUP BY is used to specify part of a hierarchy i.e. to perform a range selection

26

---

---

---

---

---

---

---

---

## Example: slicing and dicing

- Slicing on *time* (quarter=4) and *market* (= M1) dimensions, then pivoting to Product\_ID and Week (in the *time* dimension)

```
SELECT S.Product_Id, T.Week, SUM (Amount)
FROM Sales S, Time T
WHERE T.Time_ID = S.Time_ID
AND T.Quarter = 4
AND S.Market_ID = 'M1'
GROUP BY S.Product_ID, T.Week
```

Problems: - needs other queries to get (sub)totals per product in Q4  
- needs another query to get the total amount for M1 in Q4  
- needs other queries to get (sub)totals for other markets

27

---

---

---

---

---

---

---

---

## Summary: OLAP operations

- Slice – a single dimension operation
- Dice – a multidimensional operation
- Roll-up – a higher level of generalization
- Drill-down – a greater level of detail
- Rotation/pivoting – view data from a new perspective

28

---

---

---

---

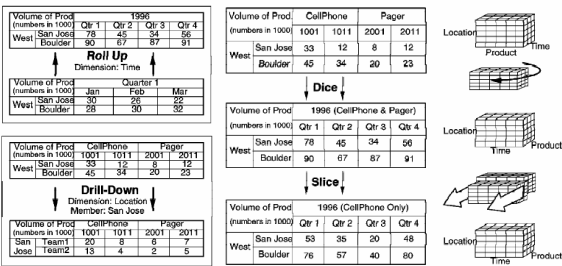
---

---

---

---

## Examples



29

---

---

---

---

---

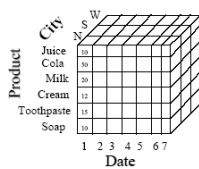
---

---

---

## Examples for you

- supermarket sales
  - roll-up: summarise over all cities, regions
  - slice the cube to select sales only in day 3
  - dice the cube to select sales only in week 2 (days 8-14), and group by regions



30

---

---

---

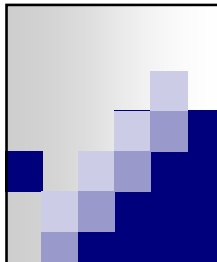
---

---

---

---

---



## SQL extensions for OLAP

31

---

---

---

---

---

---

---

---

## SQL extensions for OLAP

- Extended family of aggregate functions
  - *rank*, *percentile*, *ntile*, *limit* etc.
  - window queries ("window of rows around each tuple")
- SQL:1999 also introduced additional options in the GROUP BY clause:
  - easier and more efficient to define complex queries
    - GROUPING SETS
    - ROLLUP
    - CUBE

32

---

---

---

---

---

---

---

---

example

## Simple SQL extensions

aggregate functions such *rank*,  
*dense rank*, *ntile*, *limit* etc.

```

SELECT orderid, customerid,
 ROW_NUMBER() OVER(ORDER BY customerid) AS num,
 RANK() OVER(ORDER BY customerid) AS [rank],
 DENSE_RANK() OVER(ORDER BY customerid) AS [denserank],
 NTILE(5) OVER(ORDER BY customerid) AS ntile5
FROM orders
WHERE orderid < 10400 AND customerid <= 'BN'

```

33

---

---

---

---

---

---

---

---

## Simple SQL extensions

| orderid | customerid | num | rank | denserank | tile5 |
|---------|------------|-----|------|-----------|-------|
| 10308   | ANATR      | 1   | 1    | 1         | 1     |
| 10365   | ANTON      | 2   | 2    | 2         | 1     |
| 10355   | AROUT      | 3   | 3    | 3         | 2     |
| 10383   | AROUT      | 4   | 3    | 3         | 2     |
| 10278   | BERGS      | 5   | 5    | 4         | 3     |
| 10280   | BERGS      | 6   | 5    | 4         | 3     |
| 10384   | BERGS      | 7   | 5    | 4         | 4     |
| 10265   | BLONP      | 8   | 8    | 5         | 4     |
| 10297   | BLONP      | 9   | 8    | 5         | 5     |
| 10360   | BLONP      | 10  | 8    | 5         | 5     |

---

---

---

---

---

---

---

---

---

---

---

---

## Working example (2)

- shipments table (fact table)

SP(S#, P#, QTY, date)

- dimension tables for

- supplier: S#, ...
- product: P#, ...

| S#  | P#  | QTY | date   |
|-----|-----|-----|--------|
| S1  | P1  | 20  | 1.1.04 |
| S1  | P2  | 10  | 1.1.04 |
| S1  | P3  | 50  | 3.7.03 |
| S2  | P1  | 20  | 5.2.04 |
| S2  | P2  | 5   | 5.2.04 |
| S2  | P3  | 50  | 5.2.04 |
| S1  | P1  | 45  | 1.2.04 |
| ... | ... | ... | ...    |

---

---

---

---

---

---

---

---

---

---

---

---

## Simple aggregation queries

- total shipment quantities by supplier

```
SELECT S#, SUM(QTY)
FROM SP
GROUP BY (S#)
```

| S# | Tot. |
|----|------|
| S1 | 500  |
| S2 | 200  |

- total shipment quantities by supplier and product

```
SELECT S#, P#, SUM(QTY)
FROM SP
GROUP BY (S#, P#)
```

| S# | P# | Tot. |
|----|----|------|
| S1 | P1 | 200  |
| S1 | P2 | 100  |
| S1 | P3 | 200  |
| S2 | P1 | 100  |
| S2 | P2 | 50   |
| S2 | P3 | 50   |

---

---

---

---

---

---

---

---

---

---

---

---

## GROUPING SETS option

- Allows to specify particular groupings
- E.g. total shipment quantities by supplier and by product

```
SELECT
 S#, P#, SUM(QTY)
FROM SP
GROUP BY
 GROUPING SETS (S#, P#)
```

| S#   | P#   | Tot. |
|------|------|------|
| S1   | null | 500  |
| S2   | null | 200  |
| null | P1   | 300  |
| null | P2   | 150  |
| null | P3   | 250  |

group by S# and the by P#

37

---

---

---

---

---

---

---

---

## ROLLUP option

- roll-up along a given dimension

```
SELECT
 S#, P#, SUM(QTY)
FROM SP
GROUP BY
 ROLLUP (S#, P#)
```

GROUPING SETS ((S#, P#), S#, ( ))

| S#   | P#   | Tot. |
|------|------|------|
| S1   | P1   | 200  |
| S1   | P2   | 100  |
| S1   | P3   | 200  |
| S2   | P1   | 100  |
| S2   | P2   | 50   |
| S2   | P3   | 50   |
| S1   | null | 500  |
| S2   | null | 200  |
| null | null | 700  |

- aggregate with the finest granularity (GROUP BY S#, P#)
- then with the next level of granularity (GROUP BY S#)
- then the grand total (with no GROUP BY clause)

38

---

---

---

---

---

---

---

---

continued

## ROLLUP option

- Arguments are **ordered!**
- First, calculate the standard aggregate values specified in the GROUP BY clause
- Then, creates progressively higher-level subtotals, moving **from right to left** through the list of grouping column reference list.
  - ROLLUP(a, b, c) is equivalent to GROUPING SETS ((a, b, c), (a, b), (a), ( ))
- So, ROLLUP will create subtotals at n+1 levels, where n is the number of grouping columns
  - e.g. if there are 3 dimensions, the result set will include rows at four aggregation levels
  - note that the grouping "( )" above defines the grand total.

39

---

---

---

---

---

---

---

---

## ROLLUP option

- Widely used in tasks involving subtotals:
  - e.g. sub-totalling along a hierarchical dimension, such as time or geography. For instance, a query could specify a ROLLUP(year, month, day) or ROLLUP(country, state, city)
- ROLLUP simplifies and speeds the population and maintenance of summary tables
- Not sufficient for cross-tabular reports

---

---

---

---

---

---

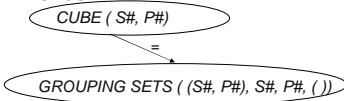
---

---

## CUBE option

- group by all possible subsets

```
SELECT
 S#, P#, SUM(QTY)
FROM SP
GROUP BY
 CUBE (S#, P#)
```



- 1) aggregate with the finest granularity (GROUP BY S#, P#)
- 2) then with all subsets (GROUP BY S#; GROUP BY P#)
- 3) then the grand total (with no GROUP BY clause)

| S#   | P#   | Tot. |
|------|------|------|
| S1   | P1   | 200  |
| S1   | P2   | 100  |
| S1   | P3   | 200  |
| S2   | P1   | 100  |
| S2   | P2   | 50   |
| S2   | P3   | 50   |
| S1   | null | 500  |
| S2   | null | 200  |
| null | P1   | 300  |
| null | P2   | 150  |
| null | P3   | 250  |
| null | null | 700  |

---

---

---

---

---

---

---

---

## CUBE option

- CUBE finds subtotals based on all possible combinations of grouping columns
  - all the subtotals that could be calculated for a data cube with the specified dimensions
  - i.e. all cross-tabular aggregations with a single SELECT statement

---

---

---

---

---

---

---

---

## CUBE option

- If there are n columns specified for a CUBE, there will be 2<sup>n</sup> combinations of subtotals returned
- E.g. CUBE(a, b, c) is equivalent to:  
 GROUPING SETS ((a, b, c),  
 (a, b), (a, c), (b, c),  
 (a), (b), (c)  
 ())

---

---

---

---

---

---

---

---

## Example

retrieve total salary, with subtotals for each department and subtotals for each job type

| dept. | name  | job       | salary |
|-------|-------|-----------|--------|
| 10    | Alex  | manager   | 2450   |
| 10    | Jack  | president | 5000   |
| 20    | Jill  | manager   | 2975   |
| 10    | Ann   | clerk     | 1300   |
| 20    | Mike  | clerk     | 900    |
| 20    | Mary  | clerk     | 1000   |
| 20    | Matt  | analyst   | 2800   |
| 20    | Sarah | analyst   | 3200   |

```

SELECT
dept, job, count(*), sum(salary)
FROM employees
GROUP BY ROLLUP(dept, job)

dept job count(*) sum(salary)

10 CLERK 1 1300
10 MANAGER 1 2450
10 PRESIDENT 1 5000
(10) 3 8750
20 ANALYST 2 6000
20 CLERK 2 1900
20 MANAGER 1 2975
(20) 5 10875
8 19625

```

---

---

---

---

---

---

---

---

## Example

```

SELECT
dept, job, count(*), sum(salary)
FROM employees
GROUP BY CUBE(dept, job)

```

| dept. | name  | job       | salary |
|-------|-------|-----------|--------|
| 10    | Alex  | manager   | 2450   |
| 10    | Jack  | president | 5000   |
| 20    | Jill  | manager   | 2975   |
| 10    | Ann   | clerk     | 1300   |
| 20    | Mike  | clerk     | 1200   |
| 20    | Mary  | clerk     | 1400   |
| 20    | Matt  | analyst   | 2800   |
| 20    | Sarah | analyst   | 3200   |

```

dept job count(*) sum(salary)

10 CLERK 1 1300
10 MANAGER 1 2450
10 PRESIDENT 1 5000
10 3 8750
20 ANALYST 2 6000
20 CLERK 2 1900
20 MANAGER 1 2975
20 5 10875
ANALYST 2 6000
CLERK 3 3200
MANAGER 2 5425
PRESIDENT 1 5000
8 19625

```

---

---

---

---

---

---

---

---

## Example for you

- Create a simple fact table  
Sales(RegionID, StoreID, ClerkID, hourlyPay)
- Explain the results of the following query  
SELECT RegionID, StoreID, AVG(hourlyPay)  
FROM Sales  
GROUP BY ROLLUP(RegionID,StoreID)

46

---

---

---

---

---

---

---

---

## OLAP systems (overview)

47

---

---

---

---

---

---

---

---

## OLAP server architectures

- ROLAP = relational OLAP
  - OLAP data in a conventional/relational DB server
  - e.g. star schema for design
  - easily scalable
- MOLAP = multidimensional OLAP
  - multidimensional storage engine in database
  - data represented as multi-dimensional arrays
  - fast computation for small to medium volumes
- HOLAP = hybrid OLAP

48

---

---

---

---

---

---

---

---

## OLAP server architectures

### ■ DOLAP = Desktop OLAP

- data in client-based files, distributed in advance or on demand to clients
  - only relatively small extracts held on client machines
- OLAP multi-dim processing on a client engine
  - desktop PC are more and more powerful
- administration of a DOLAP database performed by a central server
  - some basic processing done on the server, e.g. preparing a data cube for each client, managing security

---

---

---

---

---

---

---

---

## Commercial OLAP systems

### ■ IBM DB2 Data Warehouse Enterprise Edition

- a business intelligence platform that includes DB2, federated data access, data partitioning, integrated OLAP, advanced data mining, enhanced extract, transform, and load (ETL), workload management.

### ■ IBM DB2 Data Warehouse Standard Edition

- a data mart infrastructure product that includes DB2, integrated OLAP, advanced data mining, ETL.

### ■ IBM DB2 Data Warehouse Base Edition

- mid- to large-scale data warehouse and data mart infrastructure that includes DB2 and integrated OLAP capability

<http://www-306.ibm.com/software/data/db2/udb/dwe/>

---

---

---

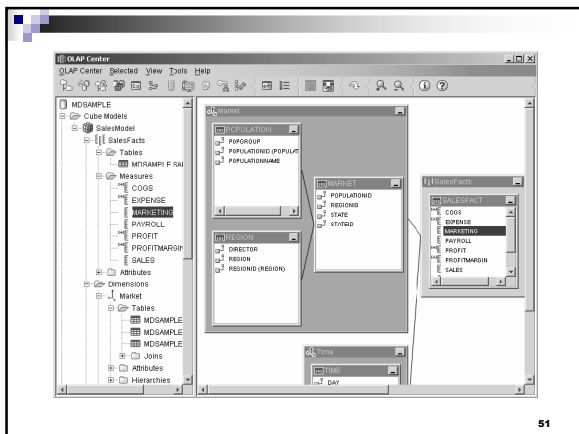
---

---

---

---

---



---

---

---

---

---

---

---

---

## Commercial OLAP systems

- **HP/Sybase Industry Warehouse Studio**
  - integration and analysis of customer data from the multiple collection points
  - focused on financial services, healthcare, communications, and government business problems
  - data warehouse design and meta data analysis
- **Oracle 9i Enterprise Edition**
  - support for warehouse building
    - Oracle9i Warehouse Builder; ETL from various sources
  - support for OLAP
    - summary management
    - analytical functions
    - sophisticated SQL optimisation

---

---

---

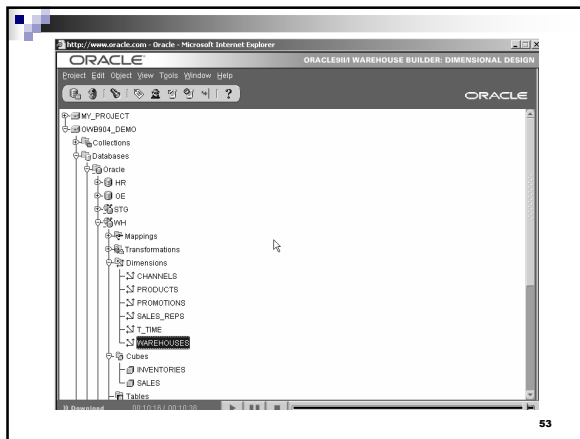
---

---

---

---

---




---

---

---

---

---

---

---

---

## Commercial OLAP systems

- **Microsoft SQL Server 2005 Business Intelligence Workbench Platform**
  - **SQL Server Relational Database:** used to create relational database
  - **Analysis Services:** used to create multidimensional model (measures, dimensions and schema)
  - **Data Transformation Services (DTS):** used to extract, transform and load data from source(s) to the DW
  - **Reporting Services:** used to build and manage enterprise reporting using the relational or multidimensional sources
  - **Data Mining:** used to extract information based on predetermined algorithms

---

---

---

---

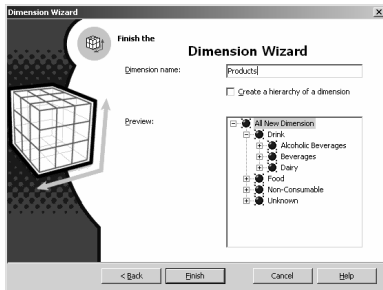
---

---

---

---

## Commercial OLAP systems



55

---

---

---

---

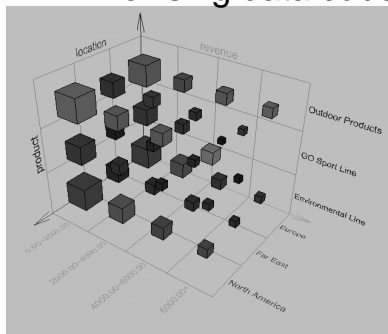
---

---

---

---

## Browsing data cubes



- Visualization
- OLAP capabilities
- Interactive manipulation

56

---

---

---

---

---

---

---

---

## Open source OLAP support

- Palo OLAP server (Jedox)
- Mondrian (Pentaho open source business intelligence)
- Cubulus OLAP
- openi.org (Open Intelligence)

See tutorial/lab materials on the web

57

---

---

---

---

---

---

---

---



## OLAP versus OLTP

- **Type of processing**
  - OLTP – day-to-day typical processing of operational transactions
  - OLAP – for high-level analysis; ad-hoc queries
- **Data access**
  - OLTP – numerous access to small amounts of data
  - OLAP – occasional access to large amounts of data
- **Frequency of updates**
  - OLTP – frequent updates; transaction integrity highly important
  - OLAP – updates do not need to be frequent

61

---

---

---

---

---

---

---

---

## OLAP versus OLTP

|                           | OLTP                  | OLAP                  |
|---------------------------|-----------------------|-----------------------|
| <b>users</b>              | operators             | analysts              |
| <b># of users</b>         | thousands             | ten-hundred           |
| <b>type of processing</b> | day-to-day operations | analytical processing |
| <b>data</b>               | detailed              | summarised            |
| <b>access</b>             | read/write            | read mostly           |
| <b>data volume</b>        | medium                | huge                  |

62

---

---

---

---

---

---

---

---

## OLAP versus OLTP

|                                 | OLTP             | OLAP          |
|---------------------------------|------------------|---------------|
| <b>performance requirements</b> | strict           | varying       |
| <b>workload</b>                 | predictable      | unpredictable |
| <b>units of work</b>            | small            | large         |
| <b>utilisation</b>              | high, repetitive | erratic       |
| <b>granularity</b>              | fine             | coarse        |

63

---

---

---

---

---

---

---

---

Wrapping -up

## OLAP and data warehouses

- DW schema should organise data in a way that supports OLAP queries
- Recall DW design steps
  1. Choose relevant business processes
  2. Choose the grain (granulation) of DW
  3. Identify measures and dimensions
  4. ETL the facts
  5. Store pre-calculations in the fact table
  6. Round out the dimension tables
  7. Choose the refresh of the DW
  8. Track slowly changing dimensions
  9. Decide the query priorities

64

---

---

---

---

---

---

---

---

Wrapping -up

## OLAP: challenges and problems

- Analytical complexity
  - business questions can be rarely answered by a single query
  - complex queries hard to understand, write and execute efficiently
  - need for good business analysts
- Data cubes can be huge
  - but also can be sparse
  - can compute in advance, compute on demand, or some combination

65

---

---

---

---

---

---

---

---

Wrapping -up

## OLAP –further comments

- Decision support is not part of database technology *per se* – but it uses it, and relies on DW schema and data model.
- OLAP practice is mostly *ad-hoc*, not so “systematic” and “scientific” as we may expect
  - it depends on a skilled and experienced analyst to formulate “interesting” queries

66

---

---

---

---

---

---

---

---

## Summary: OLAP

- Supports ad-hoc but complex querying for business analysts
- Allows a sophisticated user to analyse data using complex, multi-dimensional views
- Data aggregation in various ways
- SQL-99 extensions
  - RANK, ROLLUP, CUBE
  - many tools have specific additions
  - visualisation, reporting services

67

---

---

---

---

---

---

---

---

## Reading for this lecture

- Main text: Chapter 33 in [Connolly & Begg]
- Chapter 28 (28.3, 28.5) in [Elmasri & Navathe]

Also:

- Chaudhuri, Dayal: *An overview of data warehousing and OLAP* technology (<http://portal.acm.org/citation.cfm?id=248616>)
- See on-line materials – solve the tutorial questions and prepare for the lab.

68

---

---

---

---

---

---

---

---

## Lab 1

Next Tuesday 9 March 2010

69

---

---

---

---

---

---

---

---