The picture can't be displayed.

Chapter 1: Introduction

The original presentation is infused with more information and slides by Verena Kantere

Database System Concepts, 6th Ed.

©Silberschatz, Korth and Sudarshan
See www.db-book.com for conditions on re-use

1

Outline

- The Need for Databases
- Data Models
- Relational Databases
- Database Design
- Storage Manager
- Query Processing
- Transaction Manager

2

Database Management System (DBMS)

- DBMS contains information about a particular enterprise
  - Collection of interrelated data
  - Set of programs to access the data
  - An environment that is both *convenient* and *efficient* to use
- Database Applications:
  - Banking: transactions
  - Airlines: reservations, schedules
  - Universities: registration, grades
  - Sales: customers, products, purchases
  - Online retailers: order tracking, customized recommendations
  - Manufacturing: production, inventory, orders, supply chain
  - Human resources: employee records, salaries, tax deductions
- Databases can be very large
- Databases touch all aspects of our lives

3

University Database Example

- Application program examples
  - Add new students, instructors, and courses
  - Register students for courses, and generate class rosters
  - Assign grades to students, compute grade point averages (GPA) and generate transcripts
- In the early days, database applications were built directly on top of file systems

4

Drawbacks of using file systems to store data

- Data redundancy and inconsistency
  - Multiple file formats, duplication of information in different files
- Difficulty in accessing data
  - Need to write a new program to carry out each new task
- Data isolation
  - Multiple files and formats
- Integrity problems
  - Integrity constraints (e.g., account balance > 0) become "buried" in program code rather than being stated explicitly
  - Hard to add new constraints or change existing ones

5

Drawbacks of using file systems to store data (Cont.)

- Atomicity of updates
  - Failures may leave database in an inconsistent state with partial updates carried out
  - Example: Transfer of funds from one account to another should either complete or not happen at all
- Concurrent access by multiple users
  - Concurrent access needed for performance
  - Uncontrolled concurrent accesses can lead to inconsistencies
    - Example: Two people reading a balance (say 100) and updating it by withdrawing money (say 50 each) at the same time
- Security problems
  - Hard to provide user access to some, but not all, data

Database systems offer solutions to all the above problems

6

Levels of Abstraction

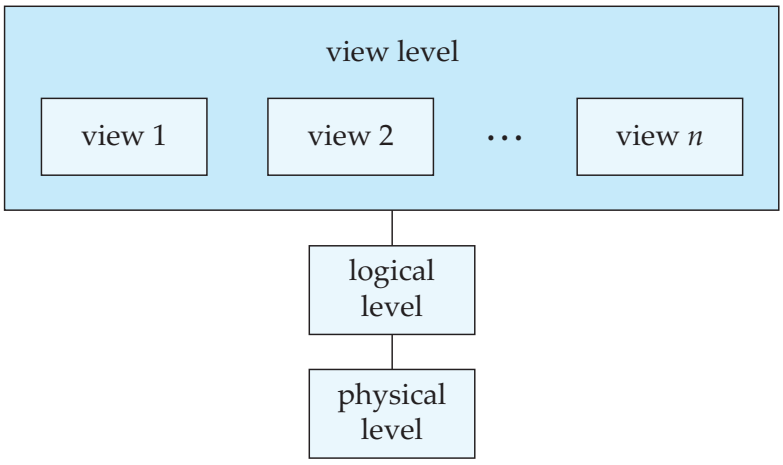
- **Physical level:** describes how a record (e.g., instructor) is stored.
- **Logical level:** describes data stored in database, and the relationships among the data.

```
type instructor = record
        ID : string;
        name : string;
        dept_name : string;
        salary : integer;
    end;
```

- **View level:** application programs hide details of data types. Views can also hide information (such as an employee's salary) for security purposes.

View of Data

An architecture for a database system



Instances and Schemas

- Similar to types and variables in programming languages
- **Logical Schema** – the overall logical structure of the database
  - Example: The database consists of information about a set of customers and accounts in a bank and the relationship between them
    - Analogous to type information of a variable in a program
- **Physical schema**– the overall physical structure of the database
- **Instance** – the actual content of the database at a particular point in time
  - Analogous to the value of a variable
- **Physical Data Independence** – the ability to modify the physical schema without changing the logical schema
  - Applications depend on the logical schema
  - In general, the interfaces between the various levels and components should be well defined so that changes in some parts do not seriously influence others.

Data Models

- A collection of tools for describing
  - Data
  - Data relationships
  - Data semantics
  - Data constraints
- Relational model
- Entity-Relationship data model (mainly for database design)
- Object-based data models (Object-oriented and Object-relational)
- Semistructured data model (XML)
- Other older models:
  - Network model
  - Hierarchical model

Relational Model

- All the data is stored in various tables.
- Example of tabular data in the relational model

Columns

ID	*name*	*dept_name*	*salary*
22222	Einstein	Physics	95000
12121	Wu	Finance	90000
32343	El Said	History	60000
45565	Katz	Comp. Sci.	75000
98345	Kim	Elec. Eng.	80000
76766	Crick	Biology	72000
10101	Srinivasan	Comp. Sci.	65000
58583	Califieri	History	62000
83821	Brandt	Comp. Sci.	92000
15151	Mozart	Music	40000
33456	Gold	Physics	87000
76543	Singh	Finance	80000

Rows

(a) The *instructor* table

A Sample Relational Database

ID	*name*	*dept_name*	*salary*
22222	Einstein	Physics	95000
12121	Wu	Finance	90000
32343	El Said	History	60000
45565	Katz	Comp. Sci.	75000
98345	Kim	Elec. Eng.	80000
76766	Crick	Biology	72000
10101	Srinivasan	Comp. Sci.	65000
58583	Califieri	History	62000
83821	Brandt	Comp. Sci.	92000
15151	Mozart	Music	40000
33456	Gold	Physics	87000
76543	Singh	Finance	80000

(a) The *instructor* table

dept_name	*building*	*budget*
Comp. Sci.	Taylor	100000
Biology	Watson	90000
Elec. Eng.	Taylor	85000
Music	Packard	80000
Finance	Painter	120000
History	Painter	50000
Physics	Watson	70000

(b) The *department* table

Data Definition Language (DDL)

- Specification notation for defining the database schema

  Example: **create table** *instructor* (
  ID **char**(5),
  name **varchar**(20),
  dept_name **varchar**(20),
  salary **numeric**(8,2))

- DDL compiler generates a set of table templates stored in a ***data dictionary***
- Data dictionary contains metadata (i.e., data about data)
  - Database schema
  - Integrity constraints
    - Primary key (ID uniquely identifies instructors)
  - Authorization
    - Who can access what

Data Manipulation Language (DML)

- Language for accessing and manipulating the data organized by the appropriate data model
  - DML also known as query language
- Two classes of languages
  - **Pure** – used for proving properties about computational power and for optimization
    - Relational Algebra
    - Tuple relational calculus
    - Domain relational calculus
  - **Commercial** – used in commercial systems
    - SQL is the most widely used commercial language

SQL

- The most widely used commercial language
- SQL is NOT a Turing machine equivalent language
- To be able to compute complex functions SQL is usually embedded in some higher-level language
- Application programs generally access databases through one of
  - Language extensions to allow embedded SQL
  - Application program interface (e.g., ODBC/JDBC) which allow SQL queries to be sent to a database

15

Database Design

The process of designing the general structure of the database:

- Logical Design – Deciding on the database schema. Database design requires that we find a "good" collection of relation schemas.
  - Business decision – What attributes should we record in the database?
  - Computer Science decision – What relation schemas should we have and how should the attributes be distributed among the various relation schemas?
- Physical Design – Deciding on the physical layout of the database

16

Database Design (Cont.)

- Is there any problem with this relation?

ID	*name*	*salary*	*dept_name*	*building*	*budget*
22222	Einstein	95000	Physics	Watson	70000
12121	Wu	90000	Finance	Painter	120000
32343	El Said	60000	History	Painter	50000
45565	Katz	75000	Comp. Sci.	Taylor	100000
98345	Kim	80000	Elec. Eng.	Taylor	85000
76766	Crick	72000	Biology	Watson	90000
10101	Srinivasan	65000	Comp. Sci.	Taylor	100000
58583	Califieri	62000	History	Painter	50000
83821	Brandt	92000	Comp. Sci	Taylor	100000
15151	Mozart	40000	Music	Packard	80000
33456	Gold	87000	Physics	Watson	70000
76543	Singh	80000	Finance	Painter	120000

17

Design Approaches

- Need to come up with a methodology to ensure that each of the relations in the database is "good"
- Two ways of doing so:
  - Entity Relationship Model
    - Models an enterprise as a collection of *entities* and *relationships*
    - Represented diagrammatically by an *entity-relationship diagram*
  - Normalization Theory
    - Formalize what designs are bad, and test for them

18

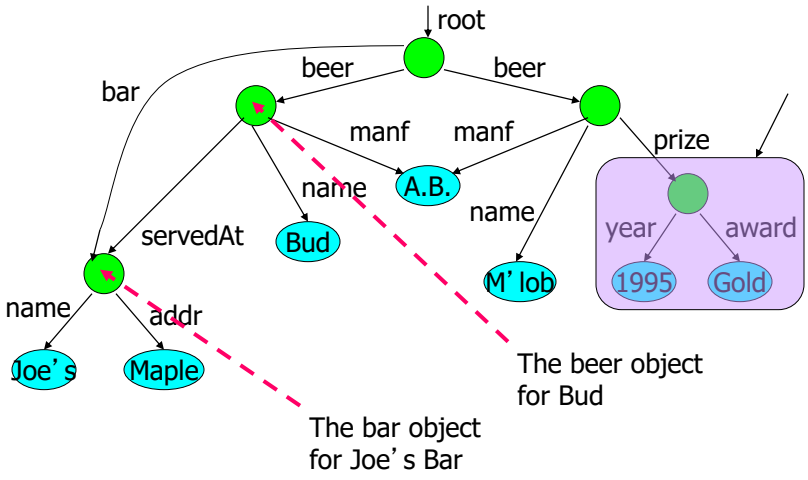
Object-Relational Data Models

- Relational model: flat, "atomic" values
- Object Relational Data Models
  - Extend the relational data model by including object orientation and constructs to deal with added data types.
  - Allow attributes of tuples to have complex types, including non-atomic values such as nested relations.
  - Preserve relational foundations, in particular the declarative access to data, while extending modeling power.
  - Provide upward compatibility with existing relational languages.

XML: Extensible Markup Language

- Defined by the WWW Consortium (W3C)
- Originally intended as a document markup language not a database language
- The ability to specify new tags, and to create nested tag structures made XML a great way to exchange **data**, not just documents
- XML has become the basis for all new generation data interchange formats.
- A wide variety of tools is available for parsing, browsing and querying XML documents/data
- XML is a language for semistructured data

Graph for semi-structured data



21

Example: Well-Formed XML

```
<? XML VERSION = "1.0" STANDALONE = "yes" ?>
<BARS>
   <BAR><NAME>Joe's Bar</NAME>
      <BEER><NAME>Bud</NAME>
         <PRICE>2.50</PRICE></BEER>
      <BEER><NAME>Miller</NAME>
         <PRICE>3.00</PRICE></BEER>
   </BAR>
   <BAR> …
</BARS>
```


22

Database Engine

- Storage manager
- Query processing
- Transaction manager

23

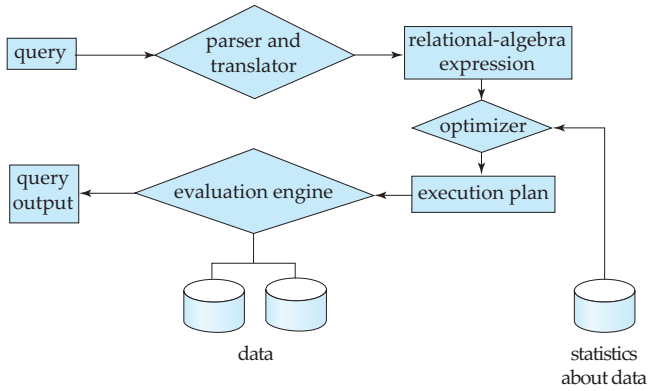
Storage Management

- **Storage manager** is a program module that provides the interface between the low-level data stored in the database and the application programs and queries submitted to the system.
- The storage manager is responsible to the following tasks:
  - Interaction with the OS file manager
  - Efficient storing, retrieving and updating of data
- Issues:
  - Storage access
  - File organization
  - Indexing and hashing

24

Query Processing

1. Parsing and translation
2. Optimization
3. Evaluation



25

Query Processing (Cont.)

- Alternative ways of evaluating a given query
  - Equivalent expressions
  - Different algorithms for each operation
- Cost difference between a good and a bad way of evaluating a query can be enormous
- Need to estimate the cost of operations
  - Depends critically on statistical information about relations which the database must maintain
  - Need to estimate statistics for intermediate results to compute cost of complex expressions

26

Transaction Management

- What if the system fails?
- What if more than one user is concurrently updating the same data?
- A **transaction** is a collection of operations that performs a single logical function in a database application
- **Transaction-management component** ensures that the database remains in a consistent (correct) state despite system failures (e.g., power failures and operating system crashes) and transaction failures.
- **Concurrency-control manager** controls the interaction among the concurrent transactions, to ensure the consistency of the database.

Concurrency Control

- Concurrent execution of user programs is essential for good DBMS performance.
  - Because disk accesses are frequent, and relatively slow, it is important to keep the CPU humming by working on several user programs concurrently.
- Interleaving actions of different user programs can lead to inconsistency: e.g., check is cleared while account balance is being computed.
- DBMS ensures such problems don't arise: users can pretend they are using a single-user system.

Transaction: An Execution Unit of a DB

- Key concept is *transaction*, which is an *atomic* sequence of database actions (reads/writes).
- Each transaction, executed completely, must leave the DB in a *consistent state* if DB is consistent when the transaction begins.
  - Users can specify some simple *integrity constraints* on the data, and the DBMS will enforce these constraints.
  - Beyond this, the DBMS does not really understand the semantics of the data. (e.g., it does not understand how the interest on a bank account is computed). Why not?
  - Thus, ensuring that a transaction (run alone) preserves consistency is ultimately the user's responsibility!

Scheduling Concurrent Transactions

- DBMS ensures that execution of {T1, ... , Tn} is equivalent to some *serial* execution T1' ... Tn'.
  - Before reading/writing an object, a transaction requests a lock on the object, and waits till the DBMS gives it the lock. All locks are released at the end of the transaction. (Strict 2PL locking protocol.)
  - Idea: If an action of Ti (say, writing X) affects Tj (which perhaps reads X), one of them, say Ti, will obtain the lock on X first and Tj is forced to wait until Ti completes; this effectively orders the transactions.
  - What if Tj already has a lock on Y and Ti later requests a lock on Y? What is it called? What will happen?

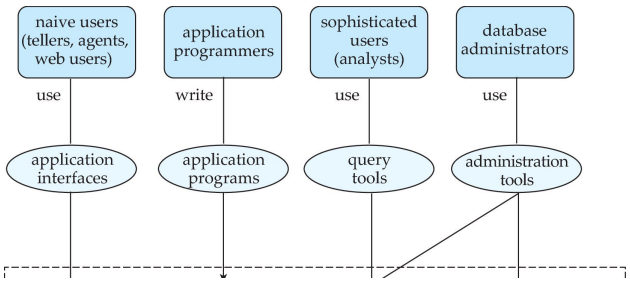
Ensuring Atomicity

- DBMS ensures *atomicity* (all-or-nothing property) even if system crashes in the middle of a Xact.
- Idea: Keep a *log* (history) of all actions carried out by the DBMS while executing a set of Xacts:
  - Before a change is made to the database, the corresponding log entry is forced to a safe location. (*WAL(Write-Ahead-Log) protocol*.)
  - After a crash, the effects of partially executed transactions are *undone* using the log. (Thanks to WAL, if log entry wasn't saved before the crash, corresponding change was not applied to database!)

The Log

- The following actions are recorded in the log:
  - *Ti writes an object*: the old value and the new value.
    - Log record must go to disk *before* the changed page!
  - *Ti commits/aborts*: a log record indicating this action.
- Log records chained together by Xact id, so it's easy to undo a specific Xact (e.g., to resolve a deadlock).
- Log is often *duplexed* and *archived* on "stable" storage.
- All log related activities (and in fact, all CC related activities such as lock/unlock, dealing with deadlocks etc.) are handled transparently by the DBMS.

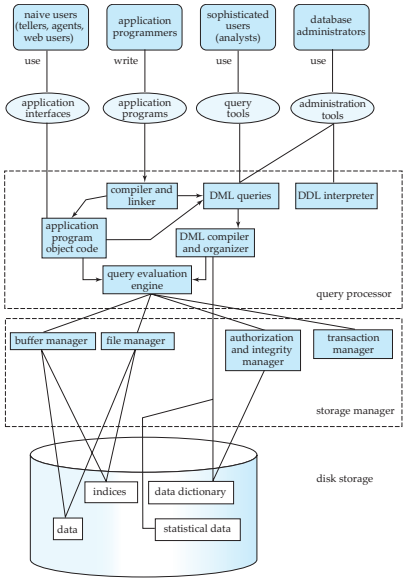
Database Users and Administrators



Databases make these folks happy ...

- End users and DBMS vendors
- DB application programmers
  - e.g. webmasters
- *Database administrator (DBA)*
  - Designs logical /physical schemas
  - Handles security and authorization
  - Data availability, crash recovery
  - Database tuning as needs evolve
  - ***Must understand how a DBMS works!***

Database System Internals



35

Database Architecture

The architecture of a database systems is greatly influenced by the underlying computer system on which the database is running:

- Centralized
- Client-server
- Parallel (multi-processor)
- Distributed

36

History of Database Systems

- 1950s and early 1960s:
  - Data processing using magnetic tapes for storage
    - Tapes provided only sequential access
  - Punched cards for input
- Late 1960s and 1970s:
  - Hard disks allowed direct access to data
  - Network and hierarchical data models in widespread use
  - Ted Codd defines the relational data model
    - Would win the ACM Turing Award for this work
    - IBM Research begins System R prototype
    - UC Berkeley begins Ingres prototype
  - High-performance (for the era) transaction processing

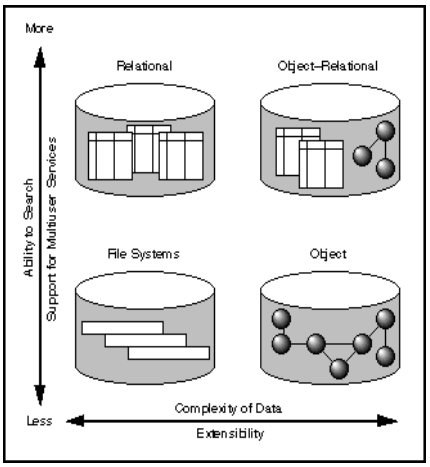
History (cont.)

- 1980s:
  - Research relational prototypes evolve into commercial systems
    - SQL becomes industrial standard
  - Parallel and distributed database systems
  - Object-oriented database systems
- 1990s:
  - Large decision support and data-mining applications
  - Large multi-terabyte data warehouses
  - Emergence of Web commerce
- Early 2000s:
  - XML and XQuery standards
  - Automated database administration
- Later 2000s:
  - Giant data storage systems
    - Google BigTable, Yahoo PNuts, Amazon, ..

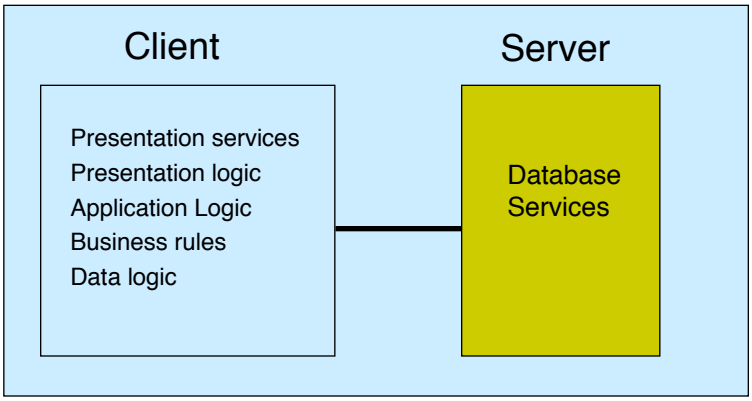
History (cont.)

- 1990-2000s (Meta-relational era)
  - It is the era of COMPLEX ENTITIES (engineering objects, multimedia, software objects)
    - Object-Relational Database Systems
  - Active Databases, Intelligent Systems, Multimedia databases
  - Client-Server
  - Data Warehouses
  - Data Mining
  - Distributed Databases, parallelization
    - DBMS in PCs
    - DBMS on the Internet (Web-based), Java, XML, …, Cloud

Relational and Object-oriented DBs



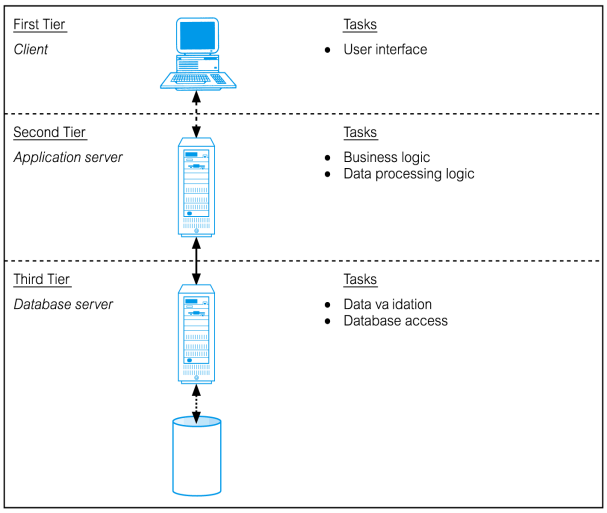
First generation of Client-Server



Architectural elements

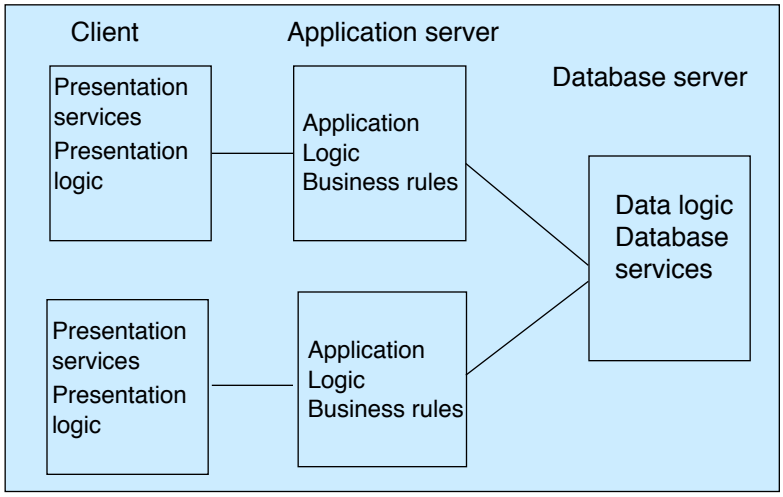
- **Presentation services**
  - Take inputs and show results
- **Presentation logic**
  - Controls the relation user-application
- **Application logic**
  - Calculations, decisions, actions for the application
- **Business rules**
  - From the whole business
- **Data logic**
  - Creation of queries (e.g. in SQL)
- **Database services**
  - Service reference to data

3-Tier architecture



43

Second generation Client-Server

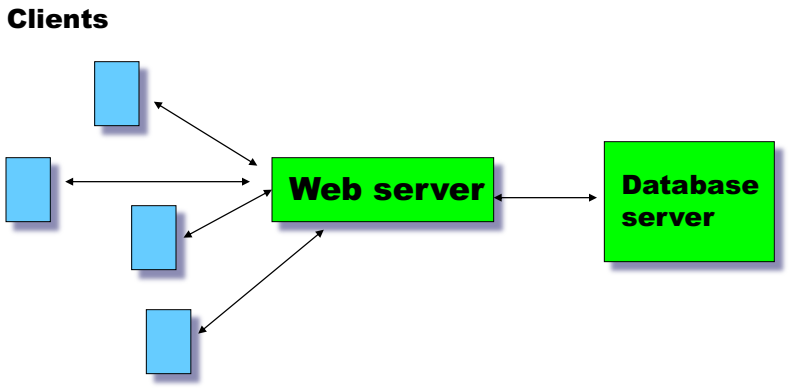


44

Infrastructure and operating system

- **Client**
  - A processor
  - PC (e.g. PowerPC, Power Mac)
  - Windows XP, … NT, … 2000, Apple Mac, OS/2, Unix
- **Server**
  - 2 to 100 users
  - Up to big machines (SMP (Symmetric Multiprocessors), clusters, MPP (Massively Parallel Processors))
  - Windows NT, 2000, Novell Netware, Unix

Today's typical architecture



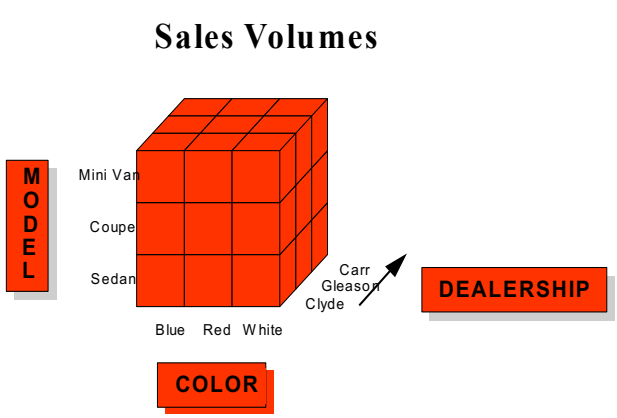
Technology for managing data

- **OLAP (Online Analytic Processing)**
  - Analysis in a multi-dimensional space
    - » Sales per (product, customer, time)
    - » Cube
  - drill down, rollup
  - Essbase, Commander, Oracle Express, SAS, Excel, SQL Server,…
- **Data Warehouses**
  - New form for Decision Support Systems
  - Redbrick, Oracle DW, Informix, Sybase, Micro Strategy
- **Data Marts (smaller Data Warehouses)**
- **Data Mining (extraction of information)**

OLAP - Applications

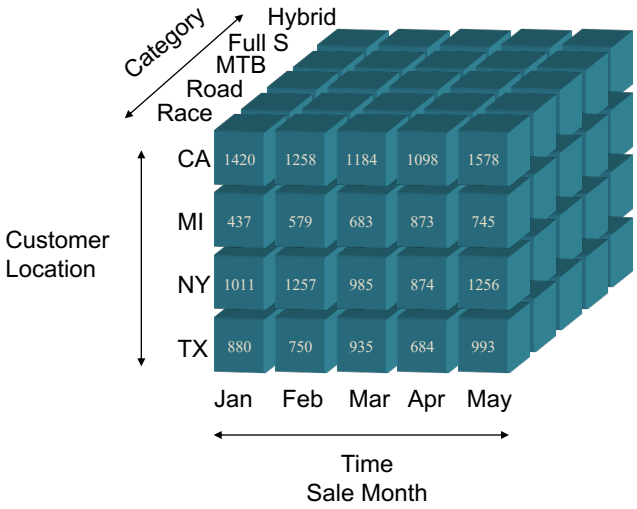
- Classical applications
  - Financial analysis, modeling ,reporting
  - Budgeting
  - Guarantee and control of Quality
  - Financial gain
  - Research analysis
  - Monitoring of products
- OLAP and OLTP cannot work at the same time on the same data!
  - Completely different and opposite requirements

OLAP: The notion of cube



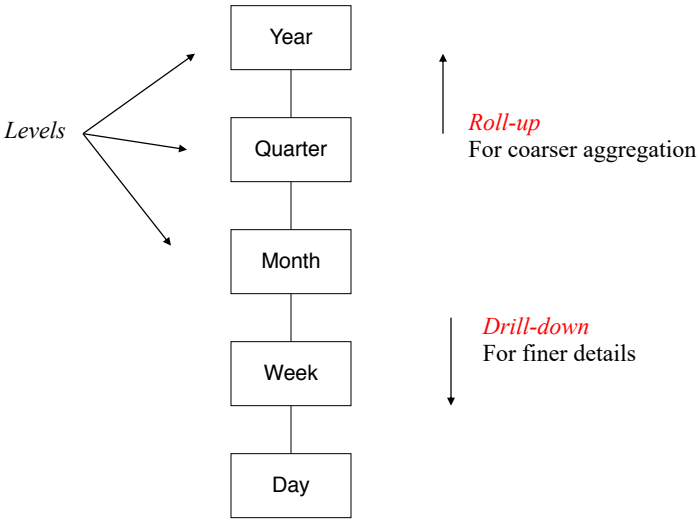
49

Example of Multidimensional cube



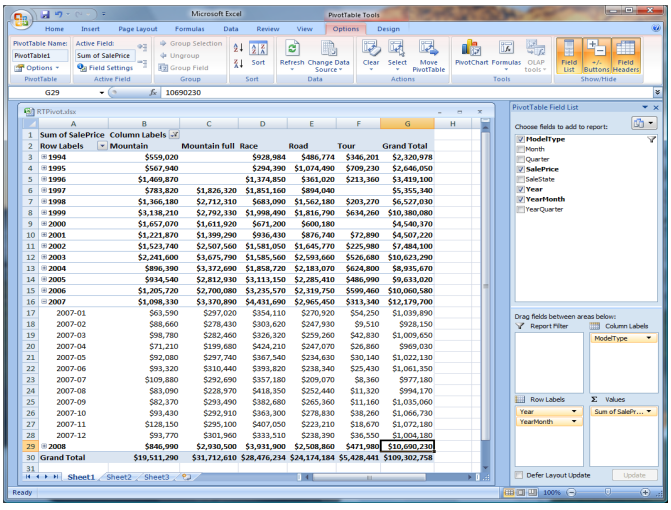
50

Basic actions / exploration



51

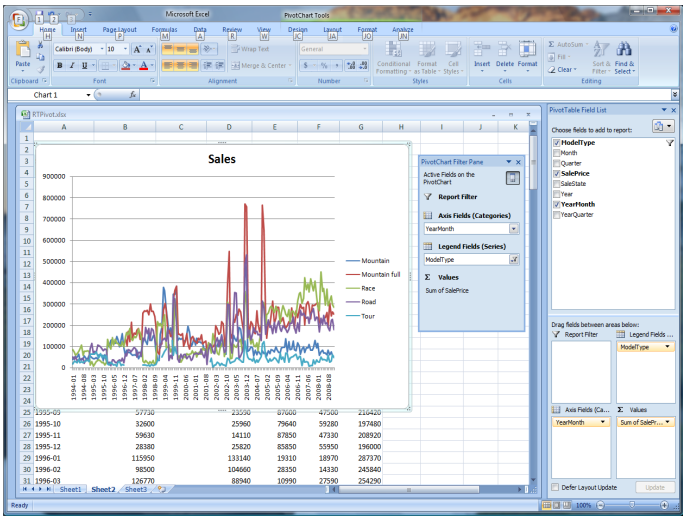
Microsoft Pivot Table



Sum of SalePrice	Column Labels					
Row Labels	Mountain	Mountain full	Race	Road	Tour	Grand Total
1994	$559,020		$928,984	$486,774	$346,201	$2,320,978
1995	$567,940		$294,390	$1,074,490	$709,230	$2,646,050
1996	$1,469,870		$1,374,850	$361,020	$213,360	$3,419,100
1997	$783,820	$1,826,320	$1,851,160	$894,040		$5,355,340
1998	$1,366,180	$2,712,310	$683,090	$1,562,180	$203,270	$6,527,030
1999	$3,138,210	$2,792,330	$1,998,490	$1,816,790	$634,260	$10,380,080
2000	$1,657,070	$1,611,920	$671,200	$600,180		$4,540,370
2001	$1,221,870	$1,399,290	$936,430	$876,740	$72,890	$4,507,220
2002	$1,523,740	$2,507,560	$1,581,050	$1,645,770	$225,980	$7,484,100
2003	$2,241,600	$3,675,790	$1,585,560	$2,593,660	$526,680	$10,623,290
2004	$896,390	$3,372,690	$1,858,720	$2,183,070	$624,800	$8,935,670
2005	$934,540	$2,812,930	$3,113,150	$2,285,410	$486,990	$9,633,020
2006	$1,205,720	$2,700,080	$3,235,570	$2,319,750	$599,460	$10,060,580
2007	$1,098,330	$3,370,890	$4,431,690	$2,965,450	$313,340	$12,179,700
2007-01	$63,590	$297,020	$354,110	$270,920	$54,250	$1,039,890
2007-02	$88,660	$278,430	$303,620	$247,930	$9,510	$928,150
2007-03	$98,780	$282,460	$326,320	$259,260	$42,830	$1,009,650
2007-04	$71,210	$199,680	$424,210	$247,070	$26,860	$969,030
2007-05	$92,080	$297,740	$367,540	$234,630	$30,140	$1,022,130
2007-06	$93,320	$310,440	$393,820	$238,340	$25,430	$1,061,350
2007-07	$109,880	$292,690	$357,180	$209,070	$8,360	$977,180
2007-08	$83,090	$228,970	$418,350	$252,440	$11,320	$994,170
2007-09	$82,370	$293,490	$382,680	$265,360	$11,160	$1,035,060
2007-10	$93,430	$292,910	$363,300	$278,830	$38,260	$1,066,730
2007-11	$128,150	$295,100	$407,050	$223,210	$18,670	$1,072,180
2007-12	$93,770	$301,960	$333,510	$238,390	$36,550	$1,004,180
2008	$846,990	$2,930,500	$3,931,900	$2,508,860	$471,980	$10,690,230
Grand Total	$19,511,290	$31,712,610	$28,476,234	$24,174,184	$5,428,441	$109,302,758

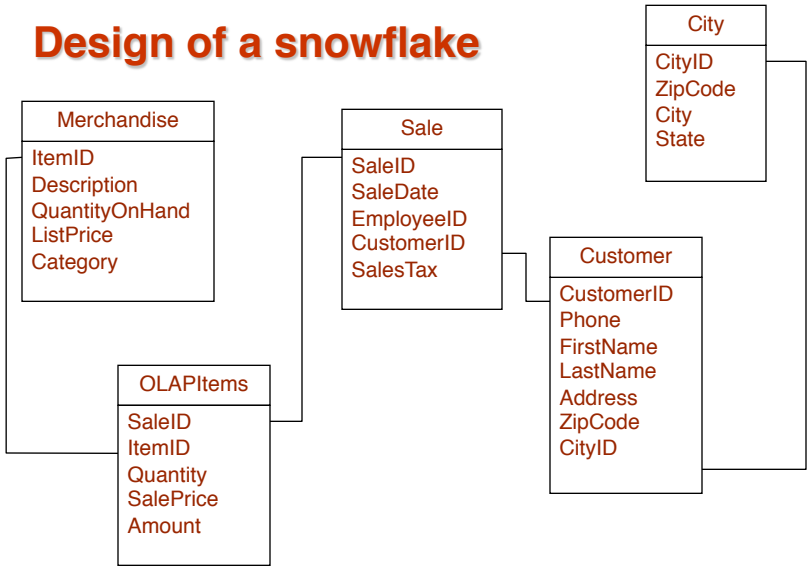
52

Microsoft Pivot Chart



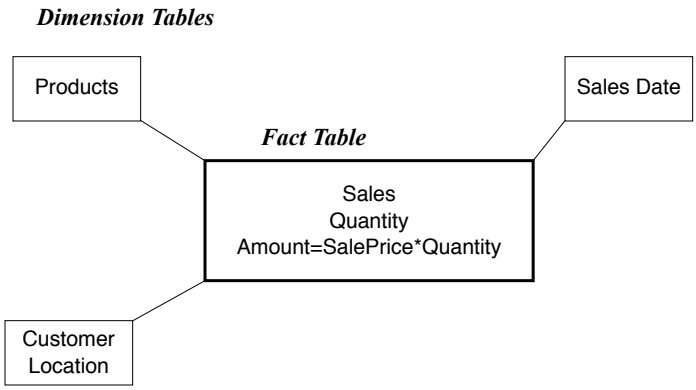
53

Design of a snowflake



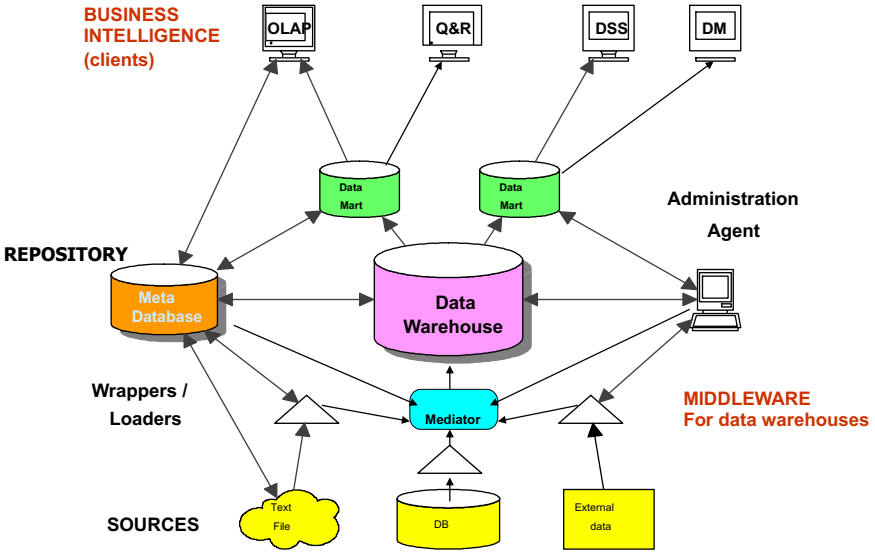
54

Design of a star



55

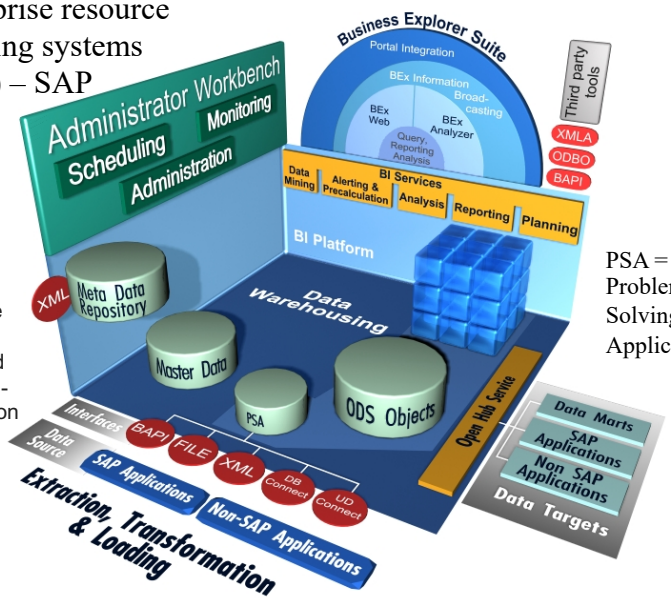
Classical architecture for data warehouse



56

Enterprise resource planning systems (ERP) – SAP

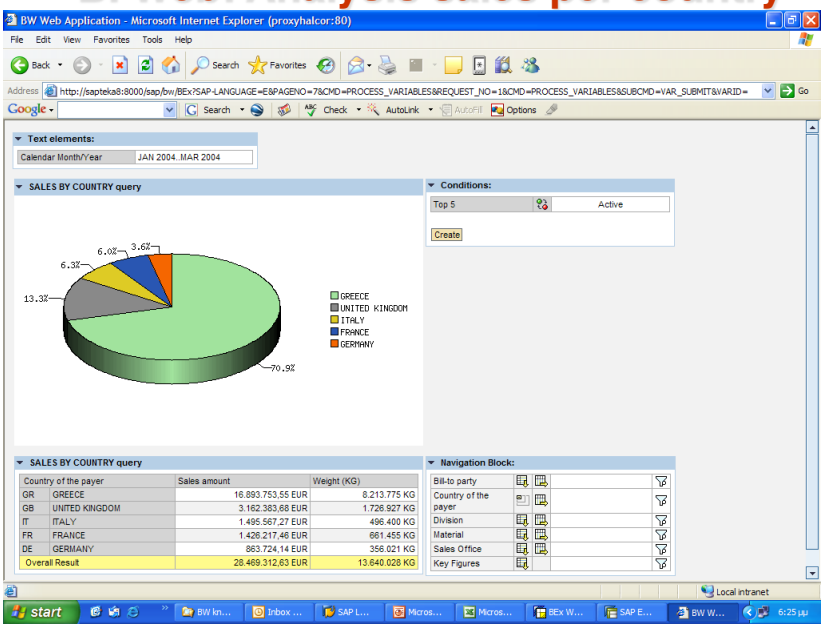
An **ODS object** acts as a storage location for consolidated and cleaned-up transaction data



PSA = Problem Solving Applications

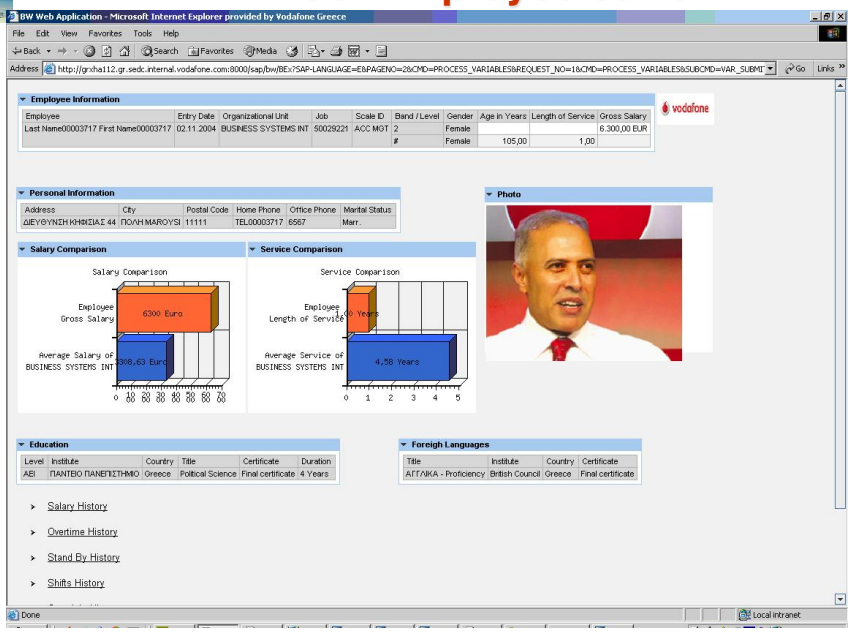
57

BI Web: Analysis sales per country



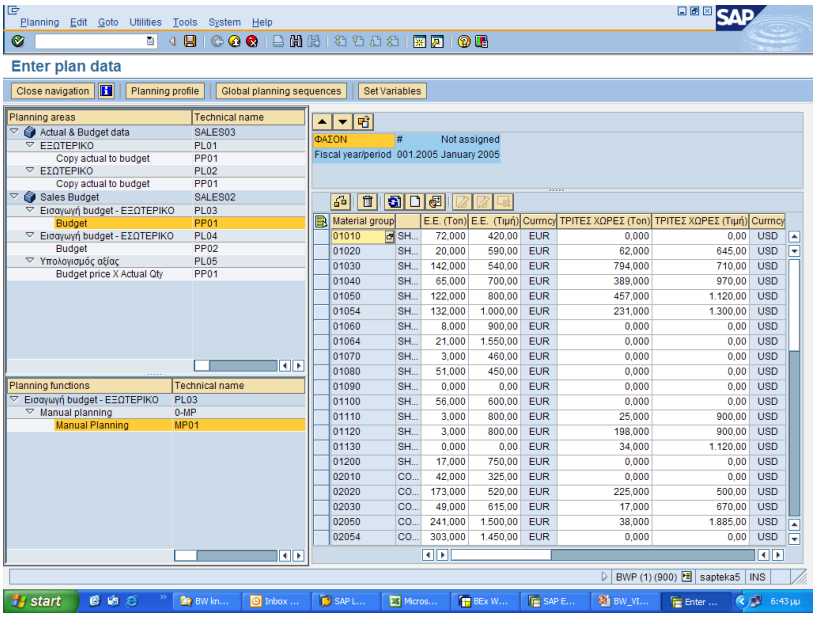
58

BI Web: Employee carte



59

Planning & Budgeting

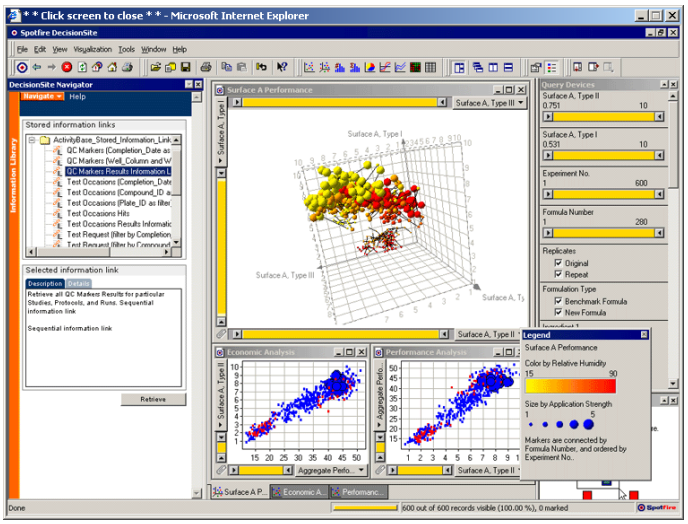


60

Data Mining

- **Automatic analysis of data**
- **Statistics**
  - Correlation
  - Regression (multiple correlation)
  - Clustering
  - Classification
  - Nonlinear relationships
- **More automatizations**
  - Analysis of the market basket
- **Numerical data and non-numerical data**
  - Language analysis

Data Mining Tools: Spotfire



http://www.spotfire.com

Market Basket Analysis

What do customers buy at the same time?

Data Mining: Market Basket Analysis

- **Goal: Measure the relation between two objects**
  - What products do customer buy together?
  - Which web pages do users visit together?
- **Classical examples**
  - In convenient stores that are open on Sunday, it was found that customers buy together beers and baby pampers.
  - Amazon.com: shows correlated purchases
- **Strategic use of such information**
  - Decide to place products together in order to increase cross selling
  - Alternatively, put them at the start an the end of the corridor in order for customers to buy even more products

DBMS : Old protagonists

- Systems that were popular in the 80s
  - IMS (IBM) -- Hierarchical Model (γλώσσα DL/1)
  - I-D-S (Honeywell) -- Network DBTG (Integrated Data Store)
  - IDMS (Cullinane) - Network (Integrated Data Mgmnt System)
  - TOTAL (Cincom) - Network
  - IMAGE (Hewlett-Packard) - Network
  - SYSTEM 2000 (Intel-MRI) - Inverted (ad-hoc model)
    Other Inverted: ADABAS (Software AG), Model 204 (CCA)
  - ...

DBMS : Old protagonists

- **SYBASE**
- **INGRES** now it is called Computer Associates-Ask Group and has become and open source system
- Others:
  - **Rdb, Gupta Quadbase, Ralma, Watcom, XDB**, ...

DBMS : Old protagonists

Today there are 3 very popular systems:

- **IBM DB2**, (also in Unix, Linux, Windows)
- **ORACLE 18 (18C),** almost everywhere – first in the market especially for Unix and large installations
- **Microsoft SQL Server 2017,** in Microsoft platforms
  - **INFORMIX** (Bought by IBM!)

DBMS : Special protagonists

- Massive Parallel Processors (**MPP**) :
  - **Terradata (the biggest), Tandem (NonStop SQL), Oracle Parallel Server, Informix, Sybase (Navigator), DB2, DEC,…(some already closed)**

- **The biggest in Windows / PC**
  - **MICROSOFT ACCESS**
  - **Powersoft, Gupta…**

  (a) SQL access πρόσβαση (gateways)

  (b) Excellent for Client-Server (DBMS)

  (c) Look very much like the big DBMS

OODBMS Companies

GemStone Systems, Inc.
Hewlett-Packard, Inc. (OpenODB)
IBEX Corporation, SA.
Illustra (Informix, Inc.)
Matisse Software, Inc.
O2 Technology, Inc.
Objectivity, Inc.
Object Design, Inc.
ONTOS, Inc.
POET Software Corporation
UniSQL
Unisys Corporation (OSMOS)
Versant Object Technology

69

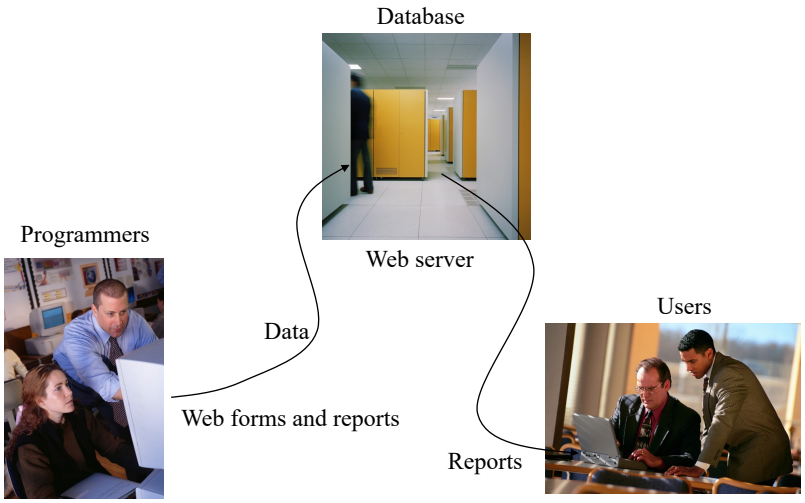
OPEN Source DBMS

- MySQL
- PostgreSQL + EnterpriseDB
- Berkeley DB, Firebird, etc.

There are also memory DBMSs, e.g.:

- SQLite (classical opensource DBMS)

Web data bases



71

Web data bases

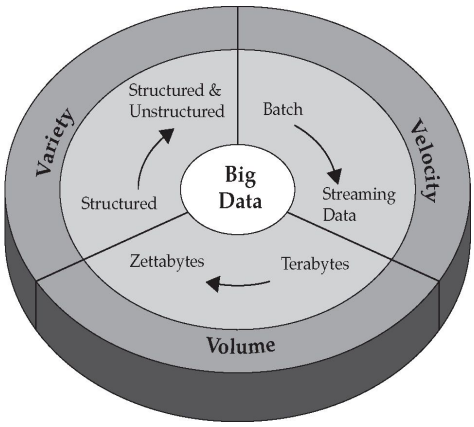
- More than 1B HTML pages, more than 15 TB
- Extraordinary amount of information
  - Bookstores, restaurants, trip advisors, purchases, news, financial, guides, maps etc etc
  - Many forms: text, image, voice, video…
  - Many formats: HTML, XML, postscript, pdf, JPEG, MPEG, MP3
- With special dynamic characteristics
  - More than 1M new pages a day
  - Graph structure among pages
- 100M queries a day

72

BIG DATA – Characteristics

- Volume -- In 2000 we had about 800,000 petabytes (PB) stored data in the whole world. We anticipate that this number will reach 35 zettabytes (ZB) until 2020. (Twitter alone produces more than 7 terabytes (TB) per day, Facebook 10 TB, and several enterprises many terabytes an hour.
- Variety -- Structures, semi-structures, unstructured
- Velocity – speed of production (Batch to streams)

BIG DATA – Characteristics

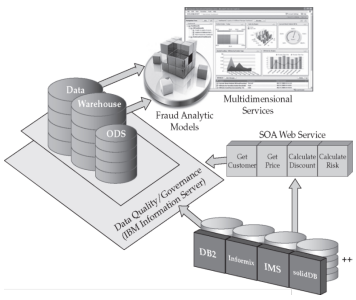


Later on:
4th v: Veracity
5th v: Value

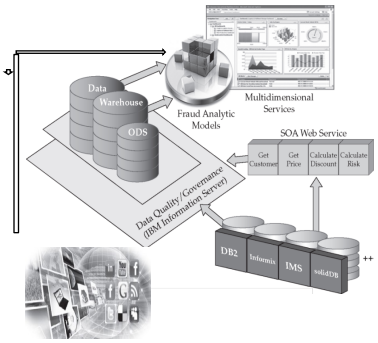
Today:
6th v: Variability
7th v: Visualization

Some also add:
8th v: Validity
9th v: Vulnerability
10th v: Volatility

Usage of 80% of data (non-relational)



Classical model



New model

HADOOP

- Hadoop (http://hadoop.apache.org/) is a very important Apache project in Apache Software Foundation written in Java.
- Hadoop is a software environment built on top of a distributed file system that was created specifically for very high intensity processing of data.
- Hadoop was inspired from Google File System (GFS) and the programming environment MapReduce, in which each job is broken into Mapper and Reducer tasks for the processing of data on a cluster of servers for parallelization.

Characteristics of NoSQL Systems

- They do not use SQL
- They do not use relations
- They do not support relationships
- They do not support classical transaction properties ACID

- They do not have a formal schema
- They are designed for the support of web applications
- They are designed for the support of very big applications
- They are Open Source

Types of NoSQL Systems (a)

- **DOCUMENT STORES**
  - **Examples: CouchDB, MongoDB**

```
{
  "title" : "Foundations of Databases"
  "rating" : 10
}
```

Types of NoSQL Systems (b)

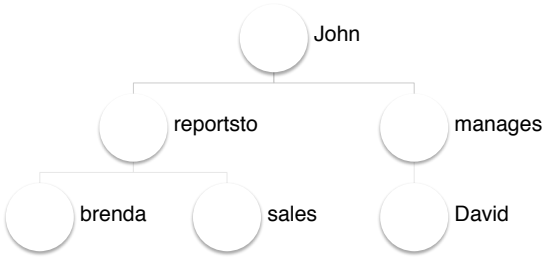
- KEY – VALUE Stores
  - Everything is stored in pairs (Key, Value)
  - Examples: Memcached, Riak, …

KEY	VALUE
Name1	bob
Course_44	DBMS
V32_phote	{binary data}
…	

Types of NoSQL Systems (c)

- GRAPH DATABASES
  - Everything is stored as a graph

    Examples: Neo4j, AlegroGraph, DB2 NoSQL, …



Why NoSQL???

- If you want a FLEXIBLE SCHEMA
- If you have HUGE AMOUNTS OF DATA
- **If you are NOT interested in data consistency so much!**

Cloud Computing with Databases

Google: BigTable, for internal storage
AppEngine: http://code.google.com/appengine/
Excellent for complex objects
Generic: Hadoop (Apache) - Open source for Cloud
Amazon:

S3	Big files http://aws.amazon.com/s3/
SimpleDB	Similar to BigTable http://aws.amazon.com/simpledb/
RDS	Service for relational data MySQL or Oracle 12g http://aws.amazon.com/rds/

Microsoft:

Azure	SQL Server http://www.microsoft.com/windowsazure/

Advantages of using cloud services for DBMS

- There are no standard costs
  - No cost for infrastructure or software
  - No maintenance
  - Easy management
- Pricing is depends on usage
  - Monthly cost proportional to the DB
  - Monthly cost based on data transfer
- Extensibility
  - Multiple distributed servers
  - Multiple high speed Internet connections
- Reliability
  - Distributed
  - Managed by specialists
  - Security

Limitations for cloud DBs

- Cost may become very high
- Then it may be better to buy in-house

Cloud Database Pricing

Example: Amazon RDS (MySQL), U.S. East
- 1 Extra large instance
- 20 hours/day
- 20 GB/month at 50 million I/O per month
- 10 GB/month data transfer in
- 500 GB/month data transfer out
- 20 GB/month regional transfer
- => $616 per month ($7400/year)

All values are estimates and might not include all fees.

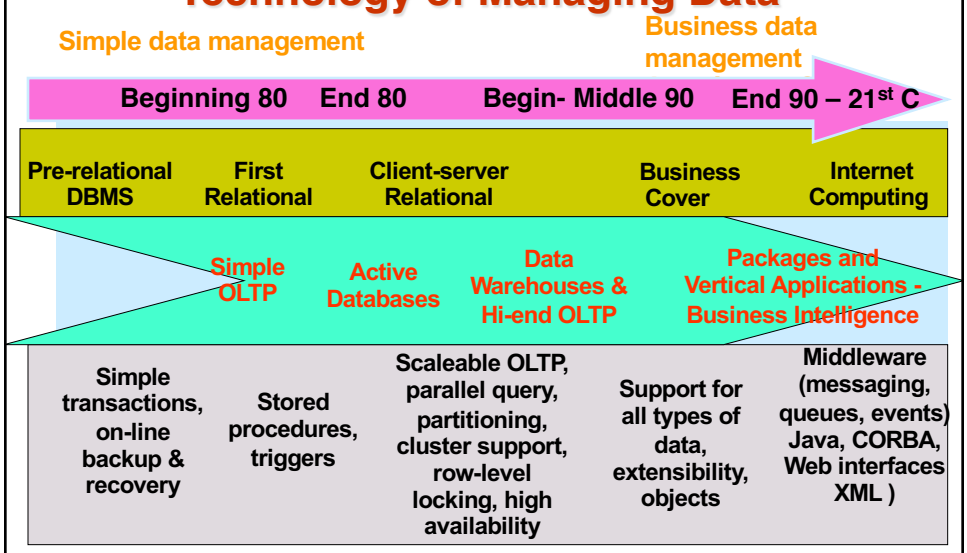
Example: Microsoft SQLAzure Business Edition
- 1 Extra large instance ($0.96/hour = $576/month)
- 20 GB/month ($200/month)
- 10 GB/month data transfer in ($1/month)
- 500 GB/month data transfer out ($75/month)
- => $852 per month ($10,224/year)

You get a relatively large database with T1-level data transfer for less than 10 percent of the cost of a DBA.

Cloud and Databases

"It was much nicer before people started storing all their data in the Cloud."

Evolution in the Application and Technology of Managing Data



Final Thoughts

- The DBMS is used for the management of Big data bases.
- Among the advantages is the recovery from system failures, fast application development, concurrency, integrity, security.
- Abstraction levels help in the independence of applications for physical entities.
- DB administrations have very important tasks and are usually well paid.
- DBMS R&D is one of the most attractive computer science domains (perfect combination of theory and practice)

End of Chapter 1