Fundamentals of databases & Database design

IT 1400- Level 1 Semester 2
Lecture 2

Thilina Ranbaduge
tranbaduge@uom.lk

What we discuss Today……

- Data Models, Schemas, and Instances
- DBMS Architecture and Data Independence
- Database Languages and Interfaces
- The Database System Environment
- Classification of Database Management Systems

Data Models

- **Data Model:** A set of concepts to describe the *structure of a database, and certain constraints that the database should obey.*

- **Data Model Operations:** Operations for specifying database retrievals and updates by referring to the concepts of the data model. Operations on the data model may include *basic operations and user-defined operations.*

Example of User Defined Operation

- *Calculate GPA*

Sum(Result GPA * No. of Credits)/Sum(Credits)

Categories of data models

- **Conceptual** (high-level, semantic) data models: Provide concepts that are close to the way many users *perceive data.* *(Also called entity-based or object-based data models.)*

- **Physical** **(low-level**, internal) data models: Provide concepts that describe details of how data is stored in the computer.

- **Implementation** **(representational)** data models: Provide concepts that fall between the above two, balancing user views with some computer storage details.

History of Data Models

- **Relational Model**: proposed in 1970 by E.F.Codd(IBM), first commercial system in 1981-82. Now in several commercial products (DB2, ORACLE, SQL Server, SYBASE, INFORMIX).

- **Network Model**: the first one to be implemented by Honeywell in 1964-65 (IDS System). Adopted heavily due to the support by CODASYL (CODASYL -DBTG report of 1971).

- **Hierarchical Data Model**: implemented in a joint effort by IBM and North American Rockwell around 1965. Resulted in the IMS family of systems. The most popular model. Other system based on this model: System 2k (SAS inc.)

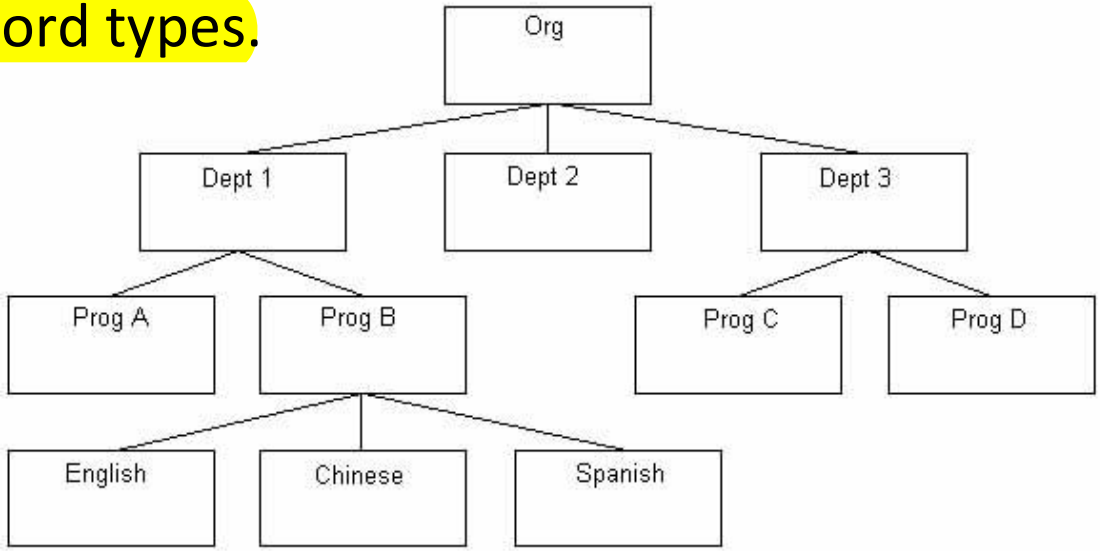
History of Data Models

- **Object-oriented Data Model(s):** several models have been proposed for implementing in a database system. One set comprises models of persistent O-O Programming Languages such as C++

- **Object-Relational Models**: Most Recent Trend. Started with Informix Universal Server. Exemplified in the latest versions of Oracle-10i, DB2, and SQL Server etc. systems

Hierarchical Model

- Organizes data in a tree structure.
- This structure implies that a record can have repeating information, generally in the child data segments.
- Collects all the instances of a specific record together as a record type.
- To create links between these record types, the hierarchical model uses Parent Child Relationships. These are a 1:N mapping between record types.



Hierarchical Model

ADVANTAGES:

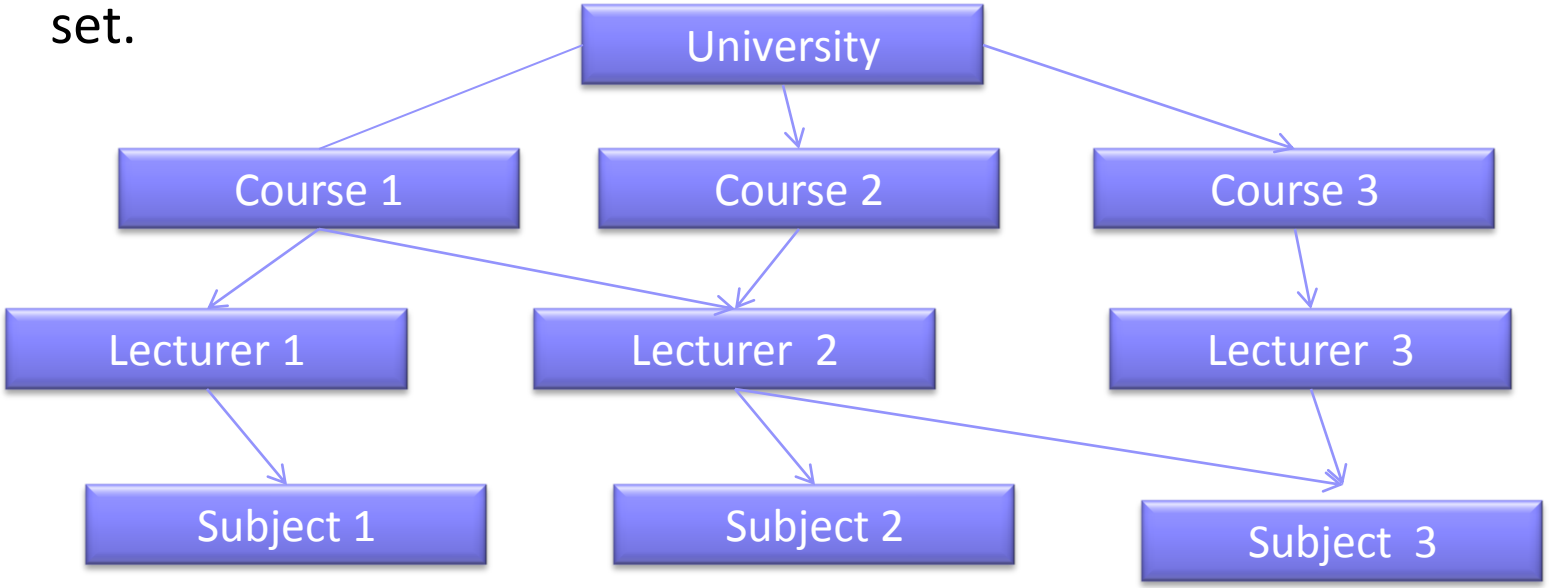
- Hierarchical Model is simple to construct and operate on
- Corresponds to a number of natural hierarchically organized domains -e.g., assemblies in manufacturing, personnel organization in companies
- Language is simple; uses constructs like GET, GET UNIQUE, GET NEXT, GET NEXT WITHIN PARENT etc.

DISADVANTAGES:

- Navigational and procedural nature of processing
- Database is visualized as a linear arrangement of records
- Little scope for "query optimization"

Network Model

- Some data were more naturally modeled with more than one parent per child.
- Permitted the modeling of many-to-many relationships in data.
- Consists of an owner record type, a set name, and a member record type.
- A member record type can have that role in more than one set
- An owner record type can also be a member or owner in another set.



Network Model

ADVANTAGES:

- Network Model is able to model complex relationships and represents semantics of add/delete on the relationships.
- Can handle most situations for modeling using record types and relationship types.
- Language is navigational; uses constructs like FIND, FIND member, FIND owner, FIND NEXT within set, GET etc. Programmers can do optimal navigation through the database

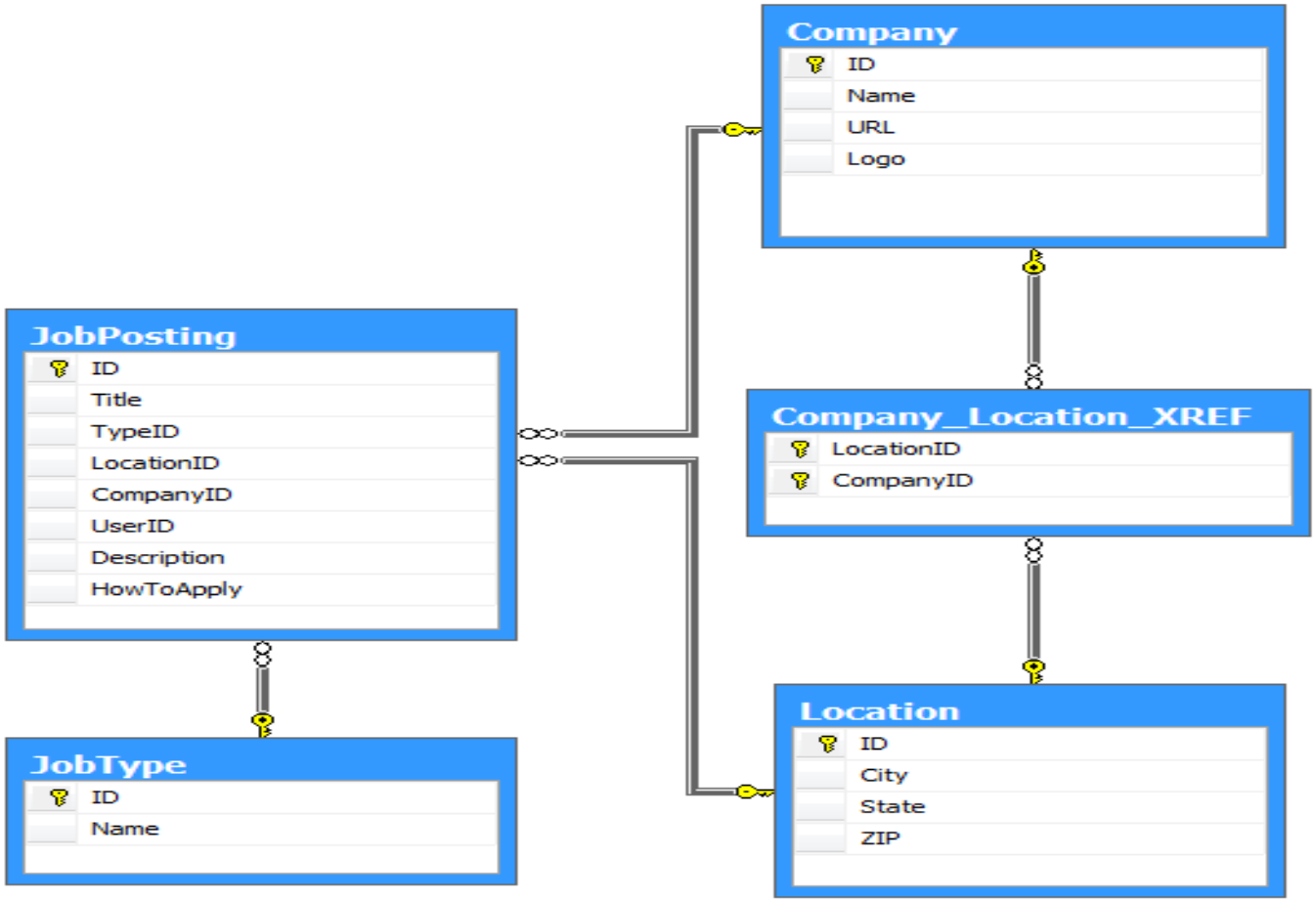
DISADVANTAGES:

- Navigational and procedural nature of processing
- Database contains a complex array of pointers that thread through a set of records.
- Little scope for automated "query optimization”

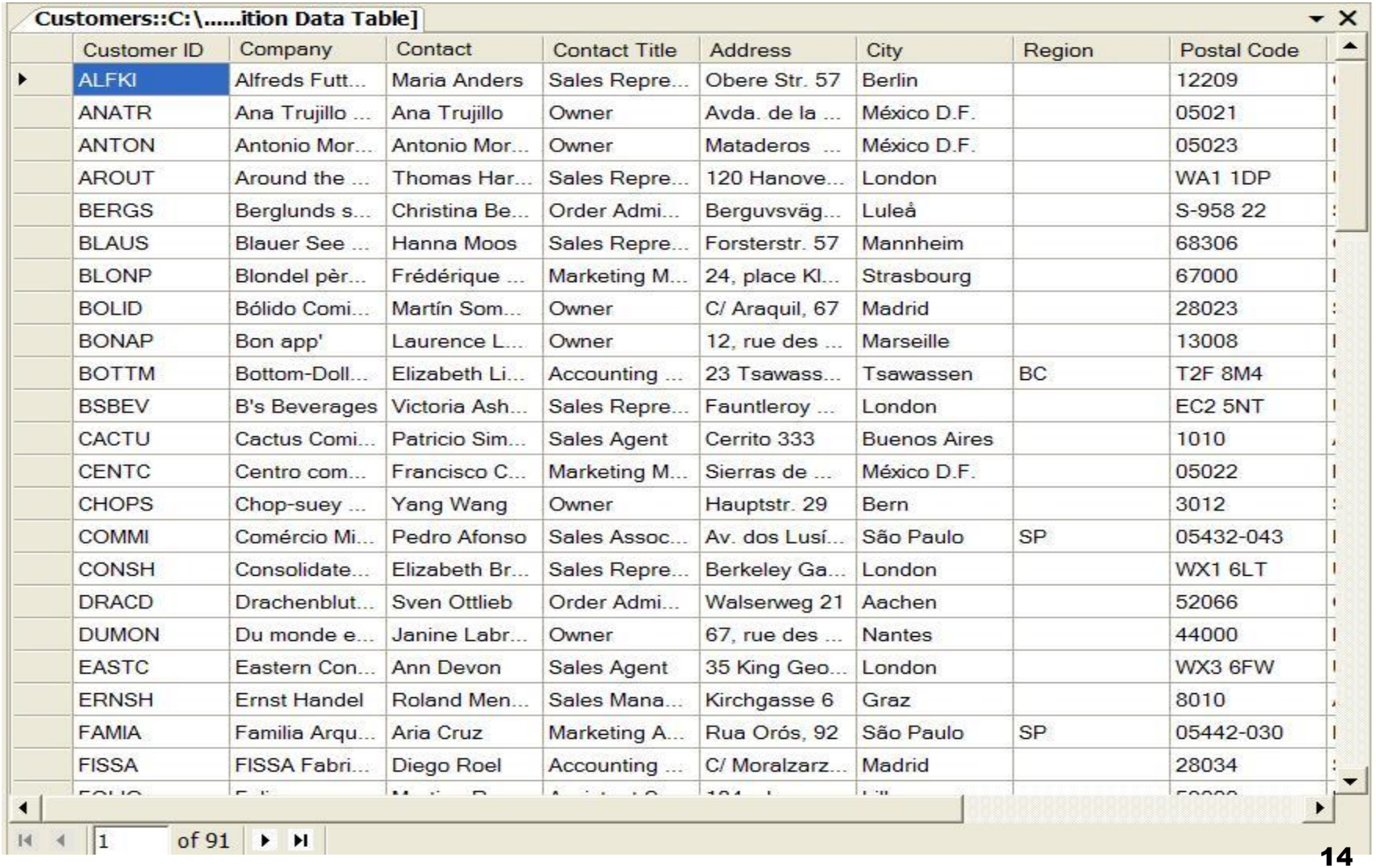
Schema VS Instances

- **Database Schema**: The description of a database. Includes descriptions of the database structure and the constraints that should hold on the database.
- **Schema Diagram**: A diagrammatic display of (some aspects of) a database schema.
- **Schema Construct**: A component of the schema or an object within the schema, e.g., STUDENT, COURSE.
- **Database Instance**: The actual data stored in a database at a particular moment in time. Also called database state(or occurrence).

Example of Schema



Example of Database Instance

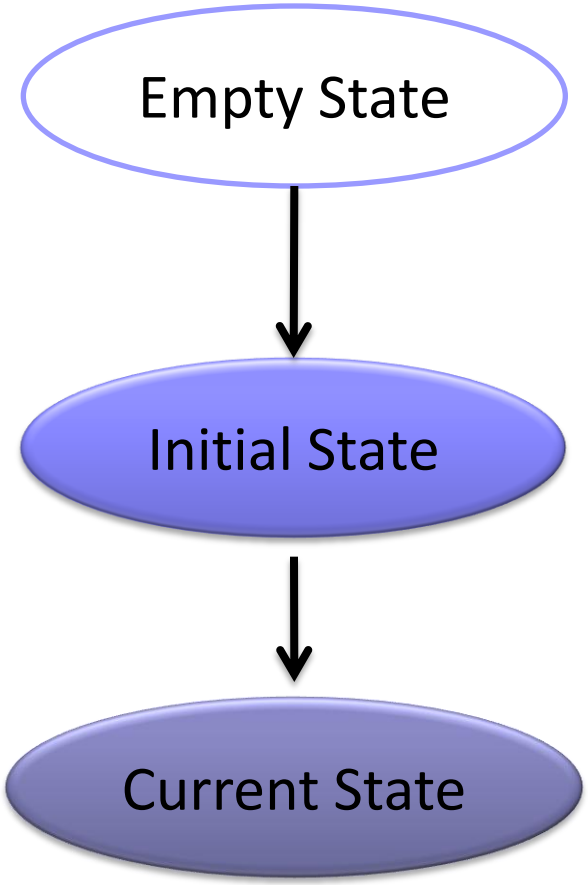


Customers::C:\.....ition Data Table]

Customer ID	Company	Contact	Contact Title	Address	City	Region	Postal Code
ALFKI	Alfreds Futt...	Maria Anders	Sales Repre...	Obere Str. 57	Berlin		12209
ANATR	Ana Trujillo ...	Ana Trujillo	Owner	Avda. de la ...	México D.F.		05021
ANTON	Antonio Mor...	Antonio Mor...	Owner	Mataderos ...	México D.F.		05023
AROUT	Around the ...	Thomas Har...	Sales Repre...	120 Hanove...	London		WA1 1DP
BERGS	Berglunds s...	Christina Be...	Order Admi...	Berguvsväg...	Luleå		S-958 22
BLAUS	Blauer See ...	Hanna Moos	Sales Repre...	Forsterstr. 57	Mannheim		68306
BLONP	Blondel père...	Frédérique ...	Marketing M...	24, place Kl...	Strasbourg		67000
BOLID	Bólido Comi...	Martín Som...	Owner	C/ Araquil, 67	Madrid		28023
BONAP	Bon app'	Laurence L...	Owner	12, rue des ...	Marseille		13008
BOTTM	Bottom-Doll...	Elizabeth Li...	Accounting ...	23 Tsawass...	Tsawassen	BC	T2F 8M4
BSBEV	B's Beverages	Victoria Ash...	Sales Repre...	Fauntleroy ...	London		EC2 5NT
CACTU	Cactus Comi...	Patricio Sim...	Sales Agent	Cerrito 333	Buenos Aires		1010
CENTC	Centro com...	Francisco C...	Marketing M...	Sierras de ...	México D.F.		05022
CHOPS	Chop-suey ...	Yang Wang	Owner	Hauptstr. 29	Bern		3012
COMMI	Comércio Mi...	Pedro Afonso	Sales Assoc...	Av. dos Lusí...	São Paulo	SP	05432-043
CONSH	Consolidate...	Elizabeth Br...	Sales Repre...	Berkeley Ga...	London		WX1 6LT
DRACD	Drachenblut...	Sven Ottlieb	Order Admi...	Walserweg 21	Aachen		52066
DUMON	Du monde e...	Janine Labr...	Owner	67, rue des ...	Nantes		44000
EASTC	Eastern Con...	Ann Devon	Sales Agent	35 King Geo...	London		WX3 6FW
ERNSH	Ernst Handel	Roland Men...	Sales Mana...	Kirchgasse 6	Graz		8010
FAMIA	Familia Arqu...	Aria Cruz	Marketing A...	Rua Orós, 92	São Paulo	SP	05442-030
FISSA	FISSA Fabri...	Diego Roel	Accounting ...	C/ Moralzarz...	Madrid		28034

1 of 91

States of a Database



Database Schema Vs. Database State

- Database State: Refers to the content of a database at a moment in time.
- Initial Database State: Refers to the database when it is loaded
- Valid State: A state that satisfies the structure and constraints of the database.

Distinction

- The database schema changes very infrequently. The database state changes every time the database is updated.
- Schema is also called intension, whereas state is called extension.

Three-Schema Architecture

Proposed to support DBMS characteristics of:

- Program-data independence.
- Support of multiple views of the data.
- Use of a catalog to store the database description (schema).

Three-Schema Architecture

- **Internal schema** at the internal level to describe physical storage structures and access paths. Typically uses a physical data model.

- **Conceptual schema** at the conceptual level to describe the structure and constraints for the whole database for a community of users. Uses a conceptual or an implementation data model.

- **External schemas** at the external level to describe the various user views. Usually uses the same data model as the conceptual level.

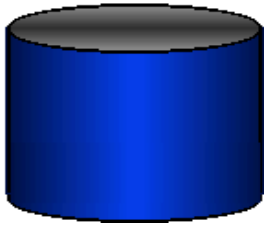
External Schema 1

External Schema 2

External Schema 3

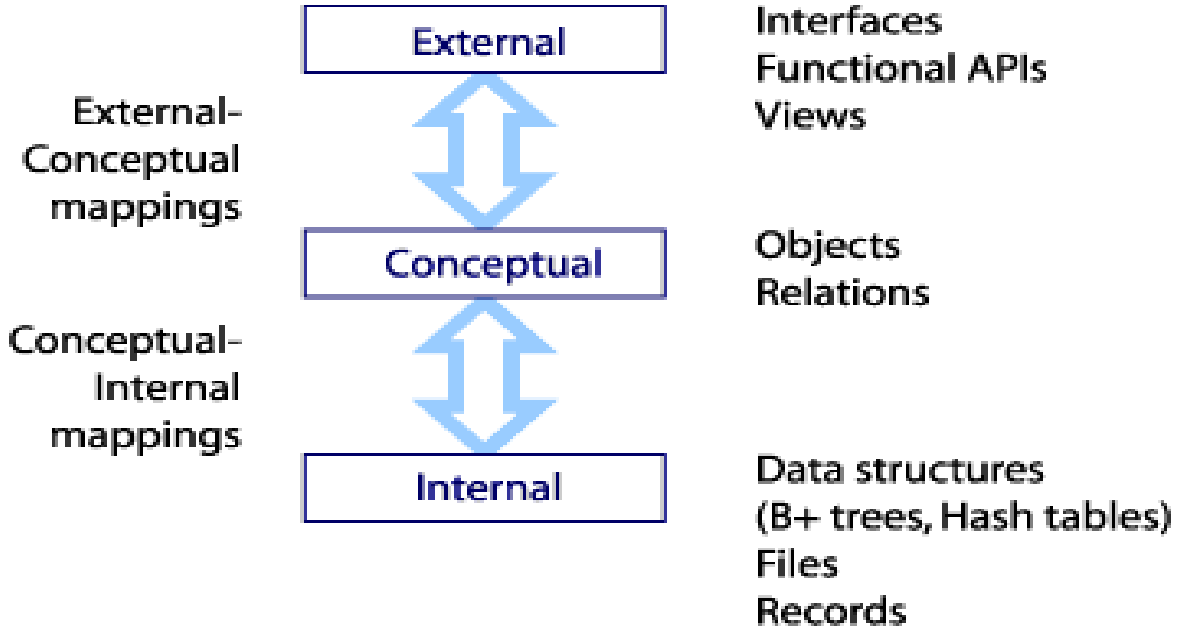
Conceptual Schema

Internal Schema



Three Schema Architecture

- **Mappings** among schema levels are needed to transform requests and data. Programs refer to an external schema, and are mapped by the DBMS to the internal schema for execution.



Data Independence

Capacity to change the schema at one level of a database system without having to change the schema at the next higher level.

- **Logical Data Independence**: The capacity to change the conceptual schema without having to change the external schemas and their application programs.
- **Physical Data Independence**: The capacity to change the internal schema without having to change the conceptual schema.

Data Independence

- Data independence is accomplished because, when the schema is changed at some level, the schema at the next higher level remains unchanged; only the *mapping* between the two levels is changed.

- However, the two levels of mappings create an overhead during compilation or execution of a query or program, leading to inefficiencies in the DBMS.

DBMS Languages

- Data Definition Language(DDL): Used by the DBA and database designers to specify the conceptual schema of a database.

- In many DBMSs, the DDL is also used to define internal and external schemas (views).

- In some DBMSs, separate storage definition language(SDL) and view definition language(VDL) are used to define internal and external schemas.

DBMS Languages

- Data Manipulation Language(DML): Used to specify database retrievals and updates.

- DML commands (data sublanguage) can be embedded in a general-purpose programming language (host language), such as COBOL, C or an Assembly Language.

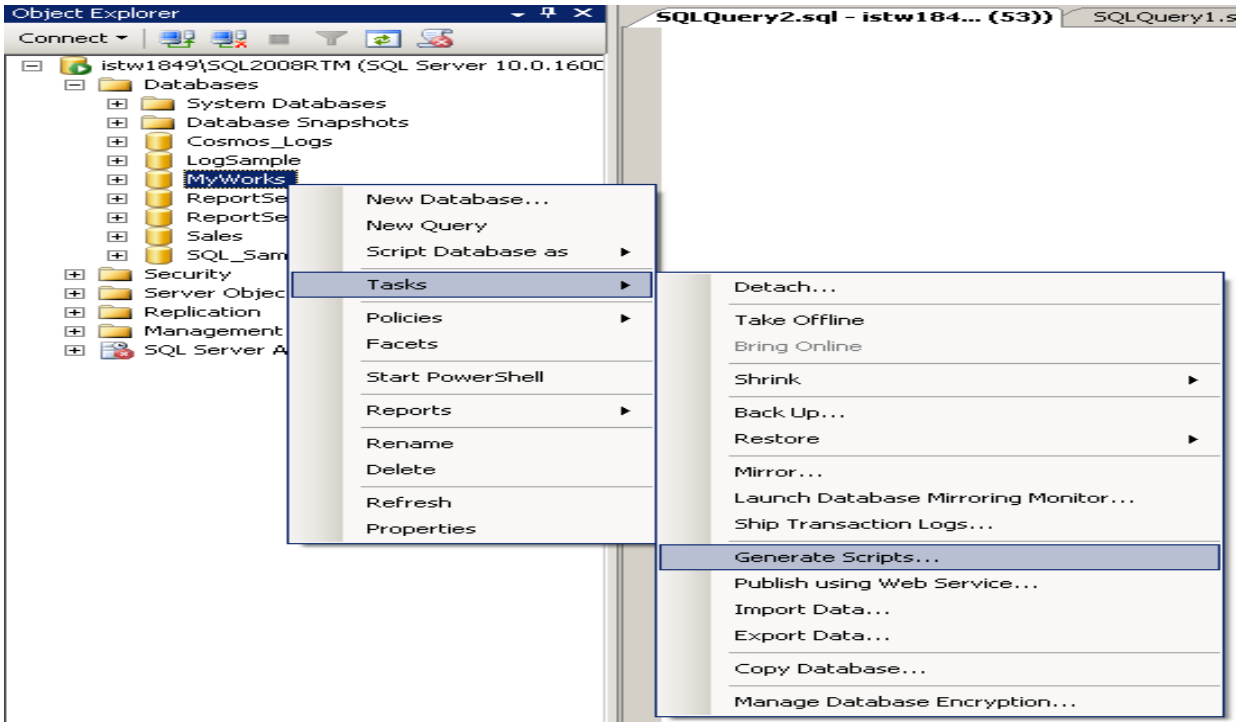
- Alternatively, stand-alone DML commands can be applied directly (query language).

DBMS Languages

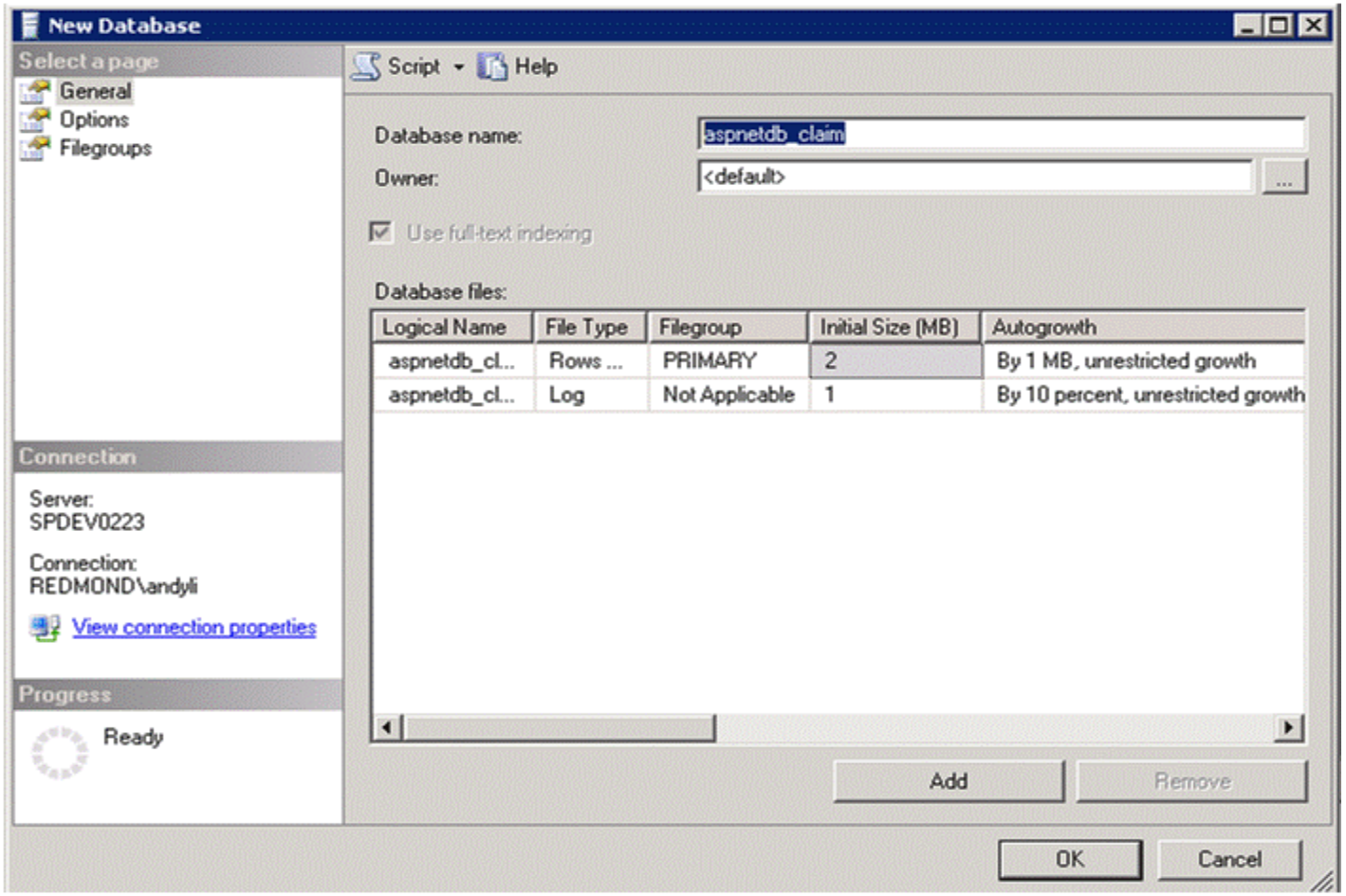
- High Level or Non-procedural Languages : e.g., SQL, are set-oriented and specify what data to retrieve than how to retrieve. Also called declarative languages.
- Low Level or Procedural Languages: record-at-a-time ;they specify how to retrieve data and include constructs such as looping.

DBMS Interfaces

- **Menu-Based Interfaces for Browsing** :- These interfaces present the user with lists of options, called **menus,** that lead the user through the formulation of a request .



- **Forms-Based Interfaces** :- A forms-based interface displays a **form** to each user. Users can fill out all of the form entries to insert new data, or they fill out only certain entries, in which case the DBMS will retrieve matching data for the remaining entries.

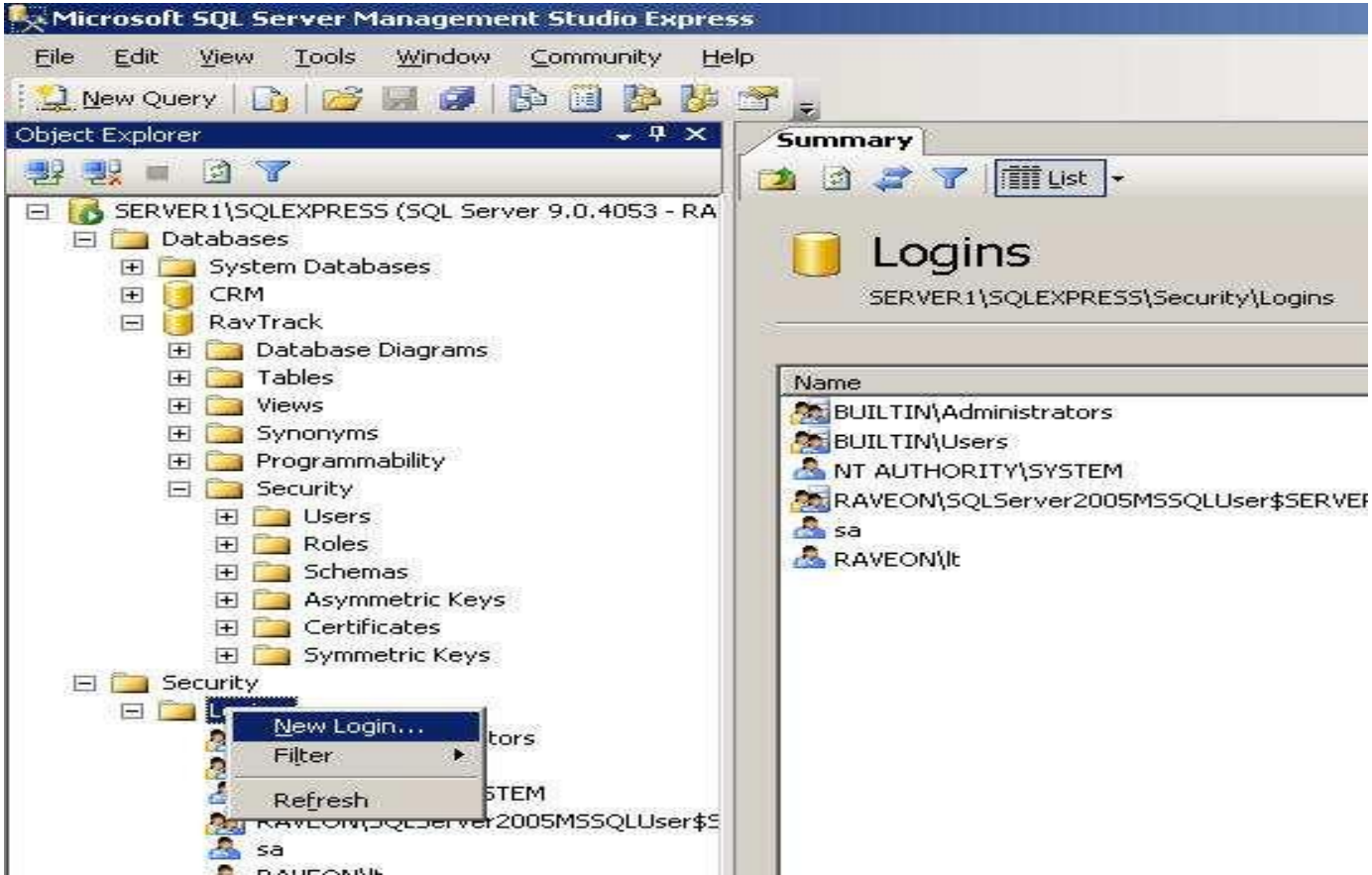


- **Graphical User Interfaces :-** A graphical interface (GUI) typically displays a schema to the user in diagrammatic form. The user can then specify a query by manipulating the diagram. In many cases, GUIs utilize both menus and forms.

- **Natural Language Interfaces :-** These interfaces accept requests written in English or some other language and attempt to "understand" them.

- **Interfaces for Parametric Users :-** Parametric users, such as bank tellers, often have a small set of operations that they must perform repeatedly.

- **Interfaces for the DBA :-** Most database systems contain privileged commands that can be used only by the DBA's staff. These include commands for creating accounts, setting system parameters, granting account authorization, changing a schema, and reorganizing the storage structures of a database.



Database System Utilities

To perform certain functions such as:

- *Loading* data stored in files into a database. Includes data conversion tools.
- *Backing up* the database periodically on tape.
- *Reorganizing* database file structures.
- *Report generation* utilities.
- *Performance monitoring* utilities.
- Other functions, such as *sorting*, *user monitoring*, *data compression*, etc.

Other Tools

Data dictionary / repository

- Used to store schema descriptions and other information such as design decisions, application program descriptions, user information, usage standards, etc.
- Active data dictionary is accessed by DBMS software and users/DBA.
- Passive data dictionary is accessed by users/DBA only.

Application Development Environments and CASE (computer-aided software engineering) tools:

- Examples –Power builder (Sybase), Builder (Borland)

communications software

function is to allow users at locations remote from the database system site to access the database through computer terminals, workstations, or their local personal computers.

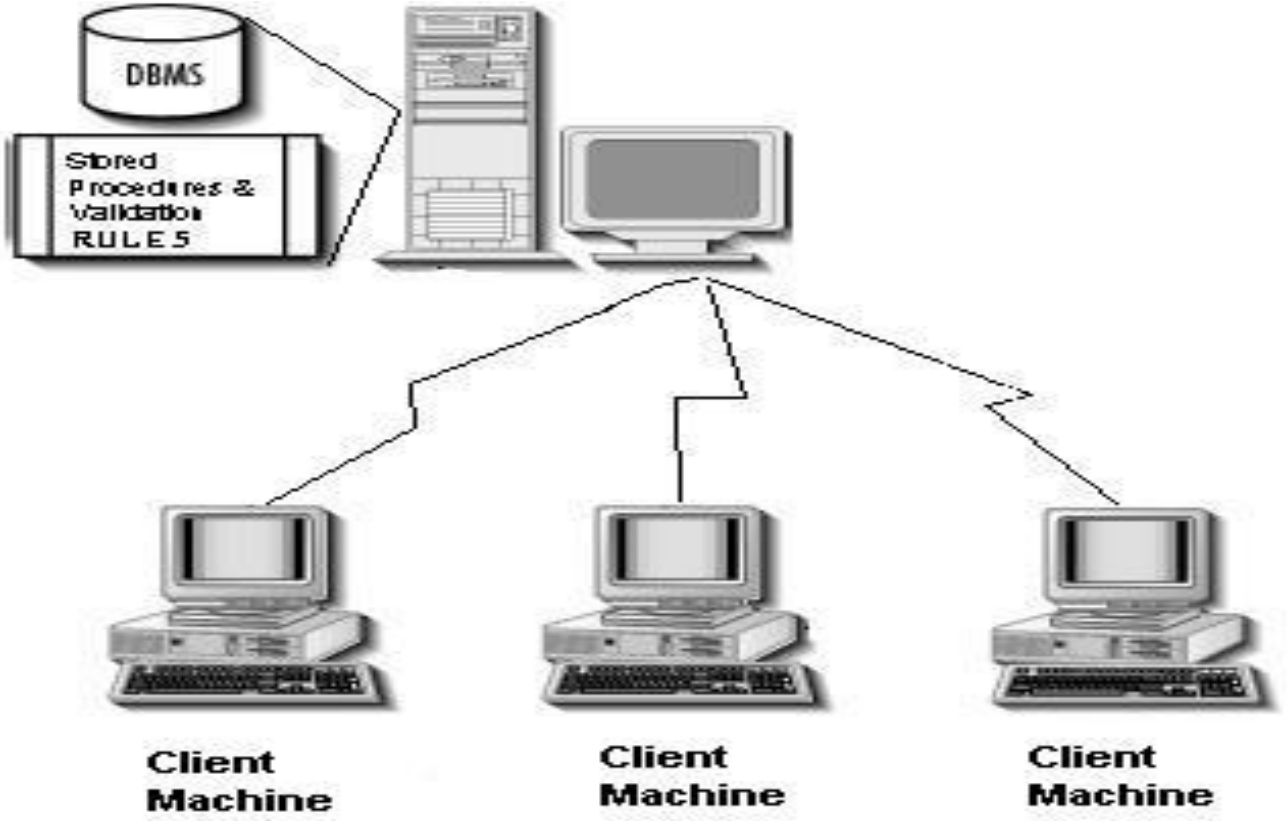
Centralized and Client-Server Architectures

- **Centralized DBMS:** combines everything into single system including- DBMS software, hardware, application programs and user interface processing software.

Basic Client-Server Architectures

- **Specialized Servers with Specialized functions**
- **Clients**
- **DBMS Server**

Client Server Architecture



Specialized Servers with Specialized functions:

- File Servers
- Printer Servers
- Web Servers
- E-mail Servers

Clients:

- Provide appropriate interfaces and a client-version of the system to access and utilize the server resources.
- Clients maybe diskless machines or PCs or Workstations with disks with only the client software installed.
- Connected to the servers via some form of a network.

  (LAN: local area network, wireless network, etc.)

DBMS Server

- Provides database query and transaction services to the clients
- Sometimes called query and transaction servers

Two Tier Client-Server Architecture

- **User Interface Programs and Application Programs** run on the client side

- Interface called **ODBC (Open Database Connectivity)** provides an Application program interface (API) allow client side programs to call the DBMS. Most DBMS vendors provide ODBC drivers.

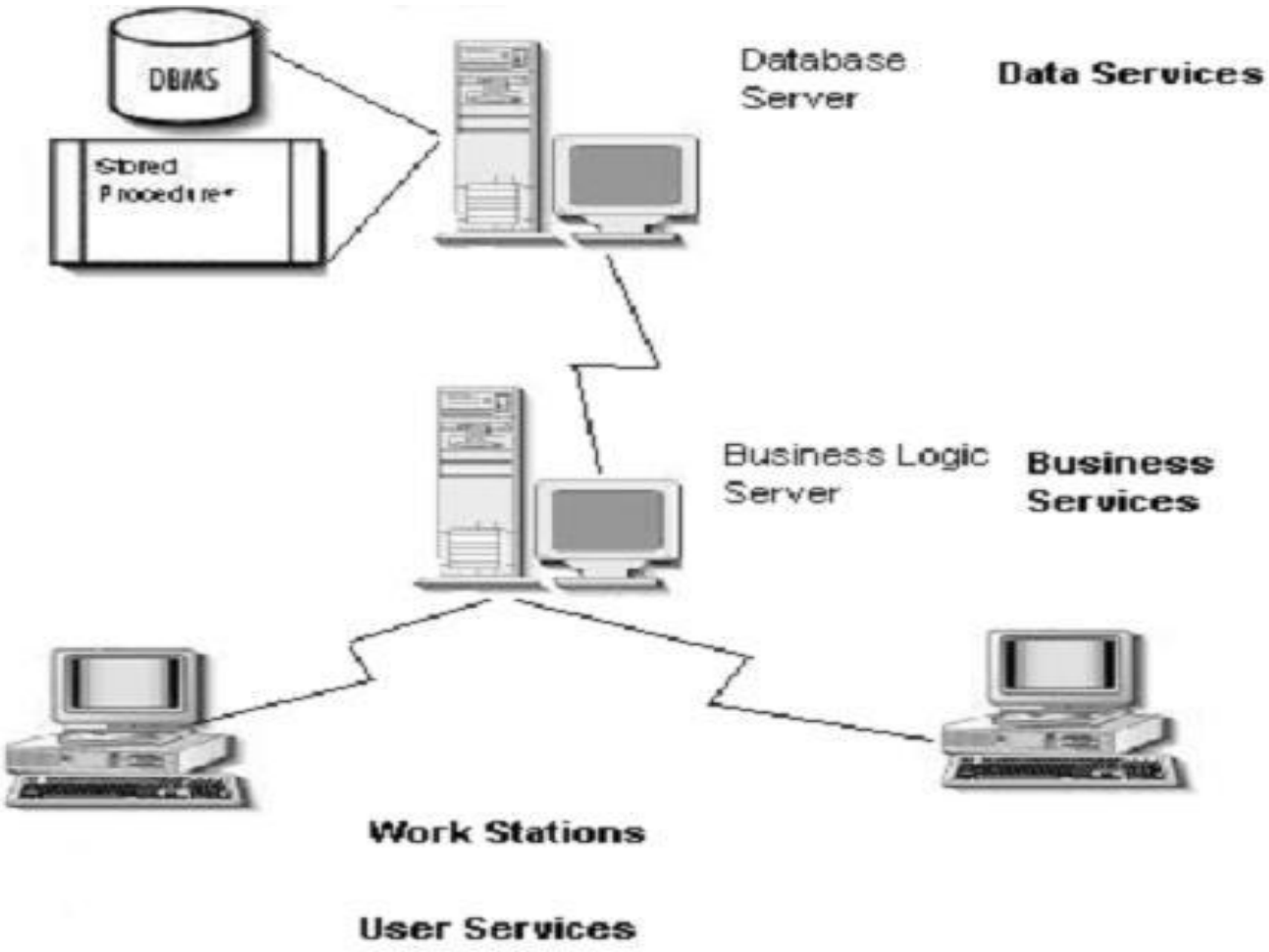
Two Tier Client-Server Architecture

- A client program may connect to several DBMSs.
- Other variations of clients are possible: e.g., in some DBMSs, more functionality is transferred to clients including data dictionary functions, optimization and recovery across multiple servers, etc.
- In such situations the server may be called the **Data Server**.

Three Tier Client-Server Architecture

- Common for **Web applications**
- Intermediate Layer called **Application Server** or **Web Server:**
  - stores the web connectivity software and **the rules and business logic (constraints)** part of the application used to access the right amount of data from the database server
  - acts like a conduit for sending partially processed data between the database server and the client.
- **Additional Features- Security:**
  - encrypt the data at the server before transmission
  - decrypt data at the client

Three Tier Architecture



Classification of DBMSs

Based on the data model used:

- Traditional: Relational, Network, Hierarchical.
- Emerging: Object-oriented, Object-relational.

Other classifications:

- Single-user (typically used with micro-computers)vs. multi-user (most DBMSs).
- Centralized (uses a single computer with one database)vs. distributed (uses multiple computers, multiple databases)

Classification of DBMSs

Distributed Database Systems *have now come to be known as client server based database systems because they do not support a totally distributed environment, but rather a set of database*

Variations of Distributed Environments:

- **Homogeneous DDBMS**
- **Heterogeneous DDBMS**
- **Federated or Multi database Systems**

Homogeneous DDBMS

- In homogeneous DDBMS, all sites use the same DBMS product.
- Much easier to design and manage.
- This design provides incremental growth by making additional new sites to DDBMS easy
- Allows increased performance by exploiting the parallel processing capability of multiple sites.

Heterogeneous DDBMS

- In heterogeneous DDBMS, all sites may run different DBMS products, which need not to be based on the same underlying data model and so the system may be composed of RDBMS, ORDBMS and OODBMS products.
- communication between different DBMS are required for translations.
- Data from the other sites may have different hardware, different DBMS products and combination of different hardware and DBMS products
- Task for locating those data and performing any necessary translation are the abilities of heterogeneous DDBMS

Referance

- Chapter 2 : *Fundamentals of Database Systems*

  By Remez Elmasri & Shamkant B. Navathe

Questions ???