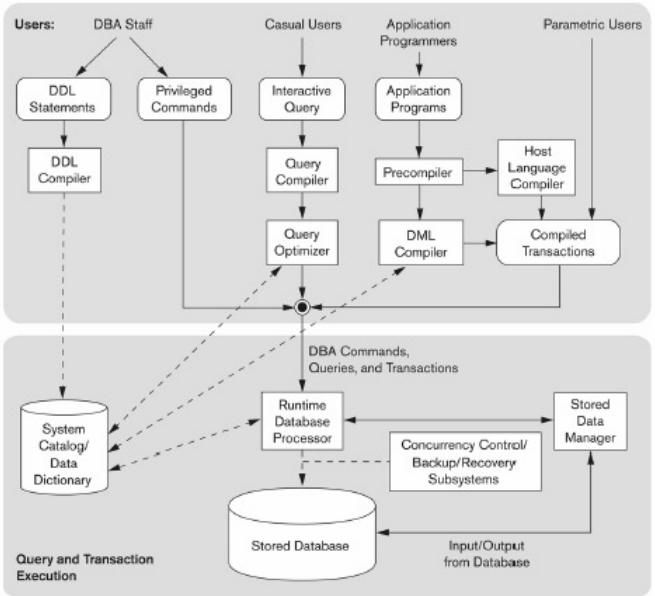
CSC 430/530 : DATABASE MANAGEMENT SYSTEMS/ DATABASE THEORY

Lecture 3: Languages and Interfaces

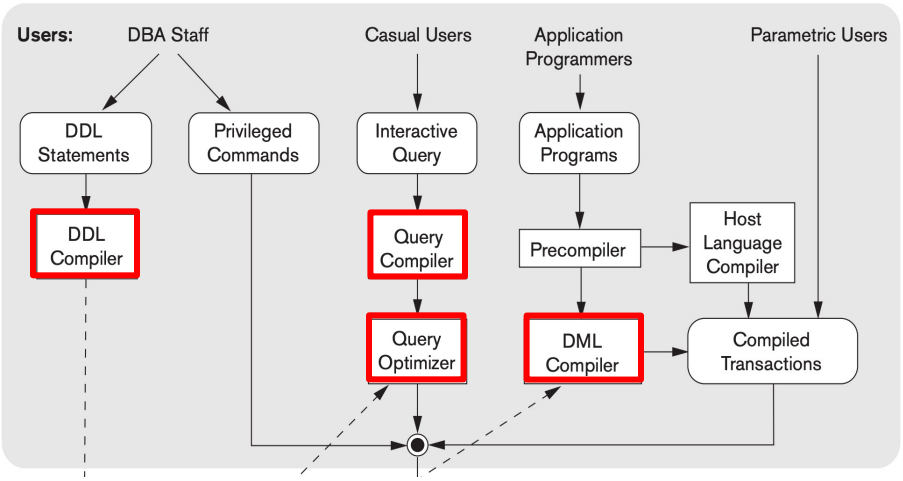
88

Component Modules of a DBMS



89

Component Modules of a DBMS



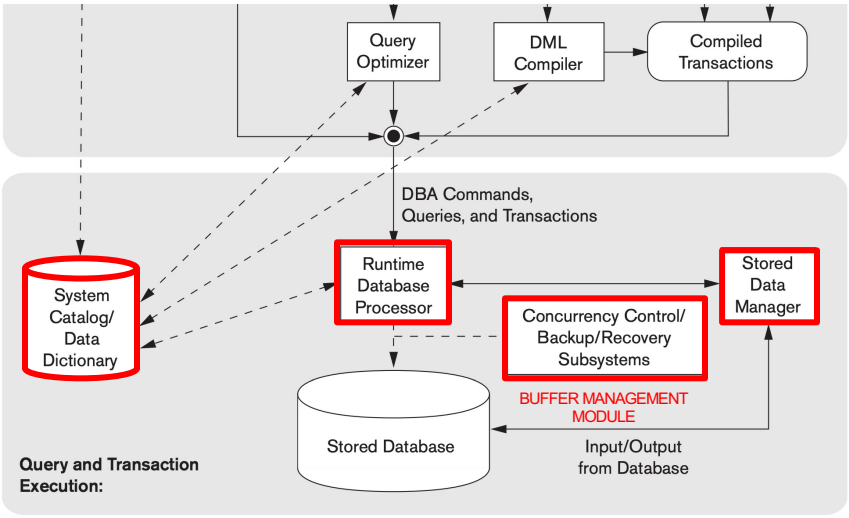
90

Component Modules of a DBMS

- **The DDL Compiler:** processes schema definitions, specified in the DDL, and stores descriptions of the schemas (meta-data) in the DBMS catalog.
- **Query Compiler:** compiles the queries into an internal form. This internal query is subject to query optimization.
- **Query Optimizer:** is concerned with the rearrangement and possible reordering of operations, removal of redundancies, and use of correct algorithms and indexes.
  - The query optimizer in consultation with the system catalog creates executable query plan to performs the necessary operations.
- **Runtime Processor** – executes the code from the query optimizer.
  - The privileged commands,
  - The executable query plans, and
  - The canned transactions with runtime parameters.

91

Component Modules of a DBMS



92

Component Modules of a DBMS

- **Buffer management module:** responsible for concurrency control and backup management.
  - Also responsible for scheduled disk read/write, as this has a considerable effect on performance (reducing disk read/write improves performance).
- **Stored data manager module** of the DBMS controls access to DBMS information.

93

DBMS Languages: Data Definition Language (DDL)

- **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 older DBMSs, separate **storage definition language** (**SDL**) and **view definition language** (**VDL**) are used to define internal and external schemas.
  - SDL is typically realized via DBMS commands provided to the DBA and database designers.

94

DBMS Languages: Data Manipulation Language (DML)

- **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 C, C++, Python or Java.
  - Alternatively, *stand-alone* DML commands can be applied directly (**query language**).
- **Types of DML**
  - **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.

95

DBMS Interfaces

1. Stand-alone query language interfaces. (**Casual User**)
2. User-friendly interfaces: (**Application oriented**)
   - Menu-based, popular for browsing on the web
   - Forms-based, designed for naïve users
   - Graphics-based (Point and Click, Drag and Drop etc.)
   - Voice-based (Alexa, Siri)
3. Programmer interfaces for embedding DML in programming languages (Embedded SQL): (**Application programmer**)
   - **Inline:** Pre-compiler Approach is based on compiler used. Could be C++, COBOL etc.
   - **Macros:** Procedure/function (Subroutine) Call Approach

96

Database System Utilities (by the DBA)

- To perform certain functions such as:
  - *Loading* data stored in files into a database. Includes data conversion tools. (**mysqlimport**)
  - *Backing up* the database periodically on tape. (**mysqldump**)
  - *Reorganizing* database file structures (**OPTIMIZE TABLE**).
  - *Report generation* utilities (**Many open-source tools**).
  - *Performance monitoring* utilities (**Many open-source tools**).
  - Other functions, such as *sorting*, *user monitoring*, *data compression*, etc.

97

Other Tools

1. **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 DBA only.
2. **Application Development Environments and CASE (computer-aided software engineering) tools:**
   - Examples – Power builder (Sybase), Builder (Borland)

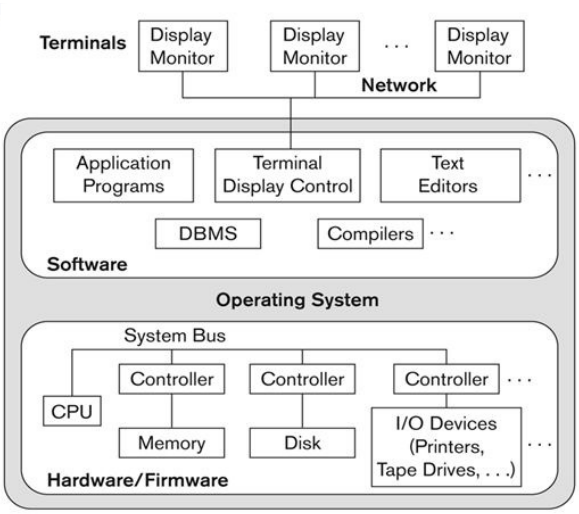
98

DBMS ARCHITECTURES

99

Centralized Architecture

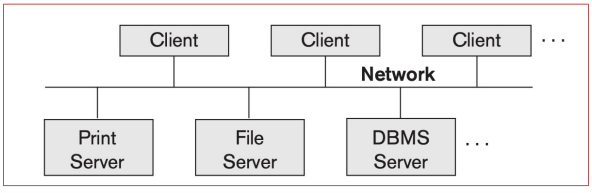
- **Centralized DBMS:** combines everything into single system including- DBMS software, hardware, application programs and user interface processing software.



100

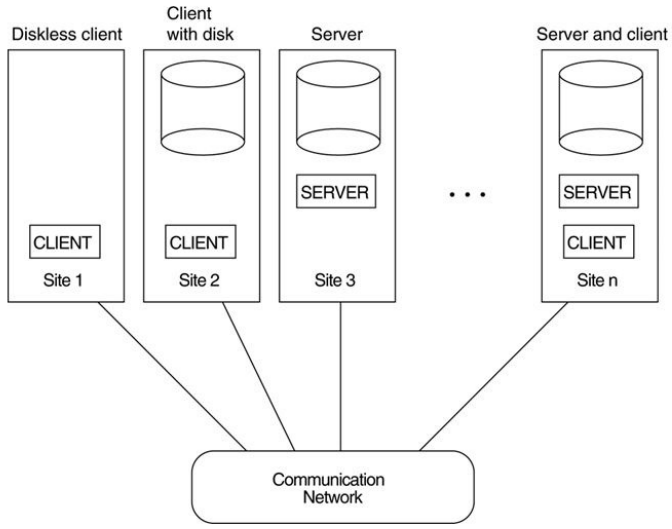
Client / Server Architectures

- **Client/Server Architecture:**
  - Specialized Servers with specialized functions
    - Clients
    - DBMS Server



101

Conceptual view:
Two-Tier Client / Server Architecture



102

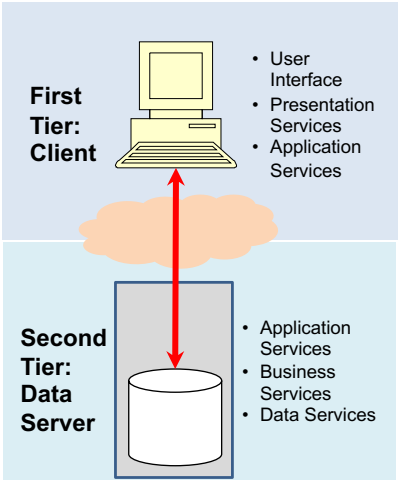
What are the roles Clients and Servers in a two-tier architecture?

- **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

103

Formal: Two Tier Client-Server Architecture

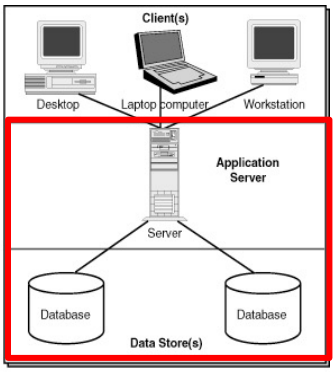
- **Client: User Interface Programs and Application Programs** run on the client side
- Interface called **ODBC (Open Database Connectivity)** provides an Application program interface (API) allow client-side programs to call the DBMS.
  - Most DBMS vendors provide ODBC drivers.
- **Data Server:** provides more functionality to clients by transferring information such as data dictionary functions, optimization and recovery across multiple servers, etc.



104

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



105

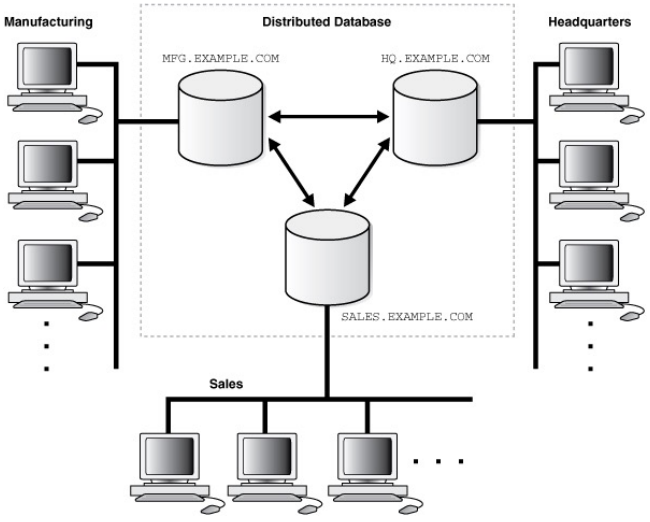
Classification of DBMSs

- **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)

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 servers supporting a set of clients.*

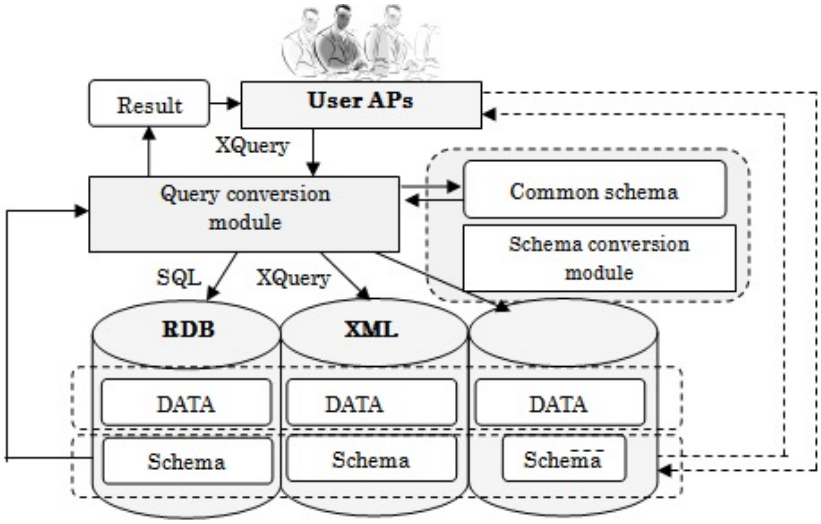
106

Distributed Database Management Systems



107

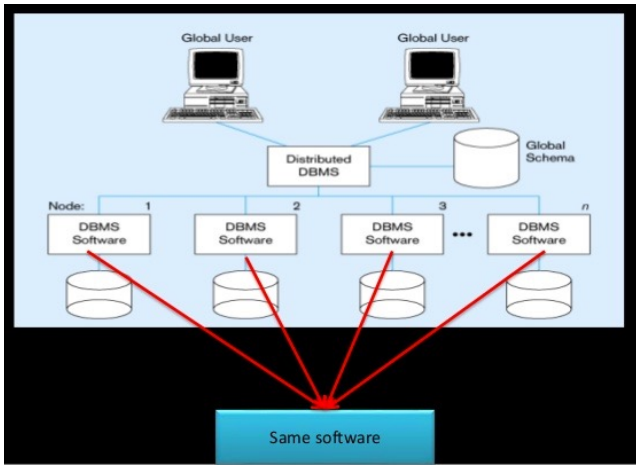
Example: Virtual Database Technology using XML Query Language



108

Variations of Distributed Environments:

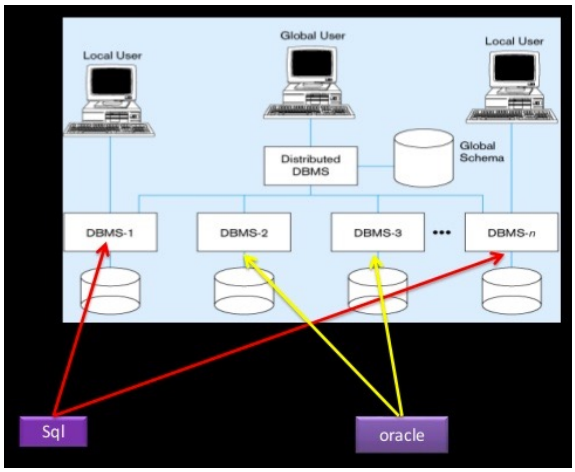
- **Homogeneous DDBMS**



109

Variations of Distributed Environments:

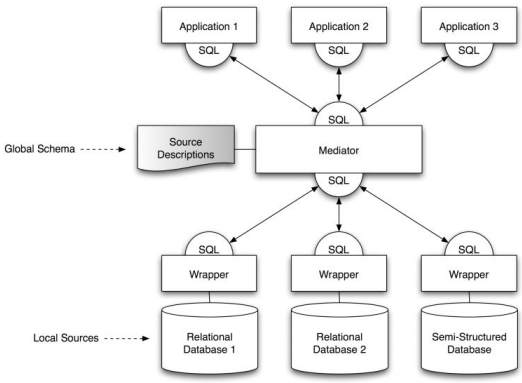
- **Heterogeneous DDBMS**



110

Variations of Distributed Environments:

- **Federated:** Each site may run different database system, but the data access is managed through a single conceptual schema
  - The use of wrappers to ensure consistency across the environment
  - Each site must adhere to a centralized access policy
  - There may be a global schema



111