Johns Hopkins Engineering

Principles of Database Systems

Module 1 / Lecture 1

Introduction to Databases

JOHNS HOPKINS
WHITING SCHOOL
of ENGINEERING

Course Introduction

- Instructor: Dr. Dar-Ning Kung
  - W: 301-827-3688; C: 240-888-4442;
  - Email: darningkung@aol.com; dnkung@yahoo.com

- Course Syllabus
  - Course Structure: Modules, Lectures, Readings, Discussion (10%), Assignments (10%), Database Project (40%) and Exams (40%)
  - Textbook
  - Office Hours via Adobe Connect and emails

Common Uses of Database Systems

- Companies and schools
- Supermarket purchases
- Bank credit card transactions
- Library borrowing activities
- Airline and hotel reservations
- PII in state and federal systems
- Online or traditional in-store shopping

What Is A Database System?

- A computerized record-keeping system essentially
- A kind of electronic filing cabinet
- A repository for a collection of computerized data files

Basic Database Operations by Its Users

- Add new, empty files to databases
- Insert new data into existing files
- Retrieve data from existing files
- Update data in existing files
- Delete data from existing files
- Remove existing files, empty or otherwise, from databases

Why Databases?

- Compactness: Reduce potentially voluminous paper files
- Speed: Perform database operations faster using computers
- Less labor: Save manpower for maintaining files
- Currency: Accurate, up-to-date information is available any time a user requests it
- Data Sharing: Access the data at the same time for multiple users (e.g., airline reservations)
- Security: Maintain access control to keep data from unauthorized access

Benefits of the Database Approach of A Multi-user System

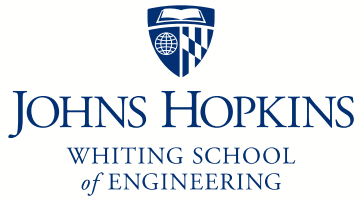
- Redundancy can be reduced
- Inconsistency can be avoided
- Data can be shared
- Standards can be enforced
- Security restrictions can be applied
- Integrity can be maintained
- Conflicting requirements can be balanced

Johns Hopkins Engineering

Principles of Database Systems

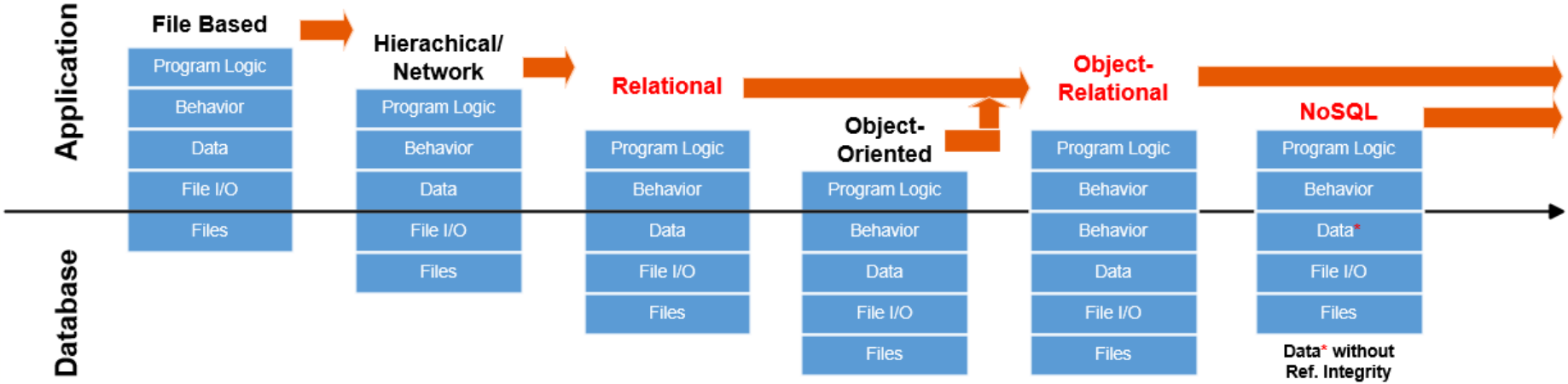
Module 1 / Lecture 2

Introduction to Databases



Database Management System (DBMS) Evolution

Database Systems
Database Applications and Databases



File Based DBMS

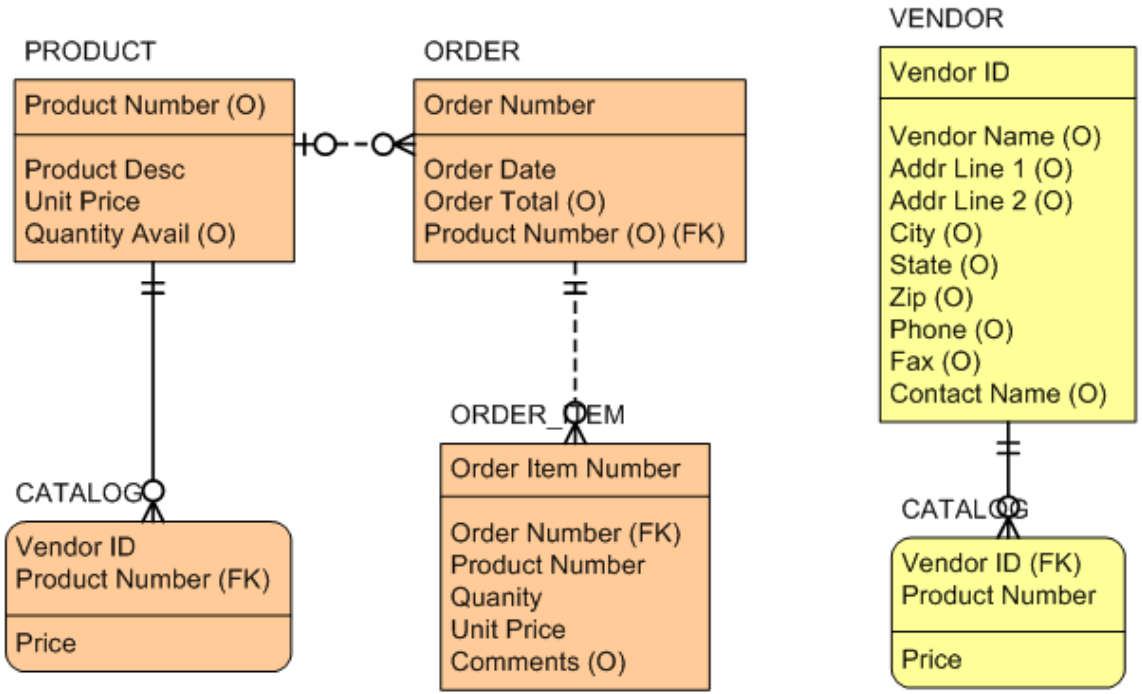
- File Based DBMS Characteristics
  - Access Method: Navigational
    - The data integrity problems due to duplication of data
    - The inability to represent logical data relationships easily
    - Program strongly depends data format (program-data dependence.)
  - Data is stored in text files or binary files

Hierarchical DBMS

- Hierarchical DBMS Characteristics
  - Access Method: Navigational (data access is only through the predefined relationships, and through the entity at the top of the hierarchy, the root, and must proceed in hierarchical order)
    - Fast access by following the predefined relationship
    - Do not provide multiple parents (or many-to-many relationships) for an entity that causes duplicated data and access restrictions
    - Do not have the ability to direct data access
    - Is not suitable for ad hoc query type applications
  - Commercial Product: IBM Information Management System (IMS), MUMPS

Hierarchical DBMS Implementation

Implementation Example

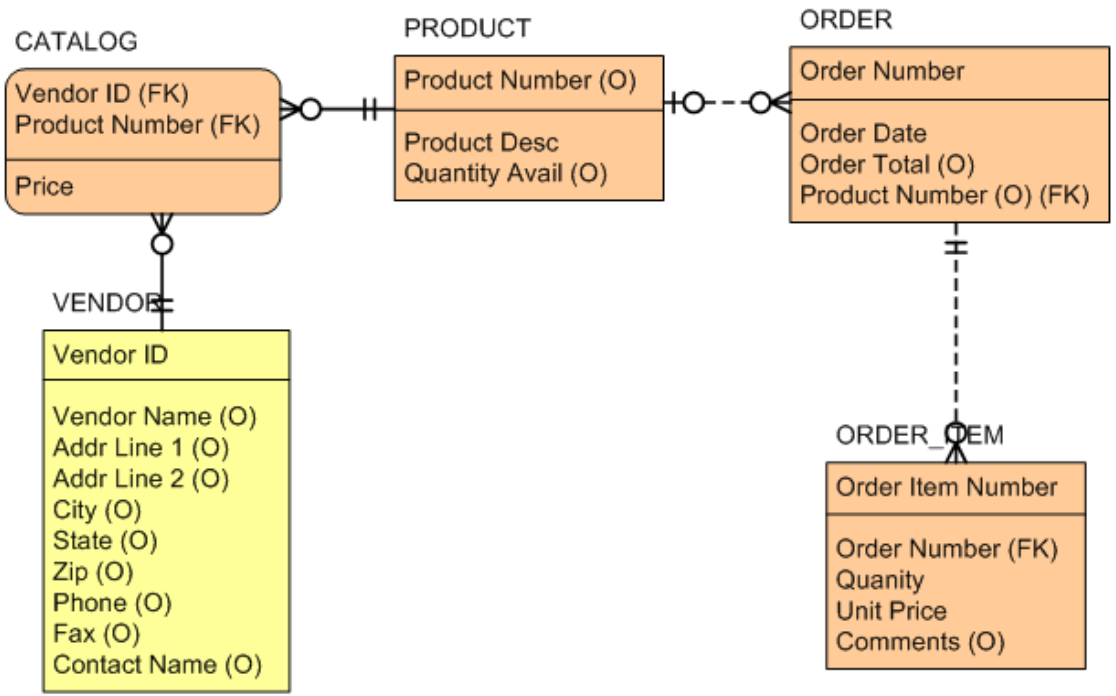


Network DBMS

- Network DBMS Characteristics
  - Access Method: Navigational
    - Design to eliminate the hierarchical database restrictions
    - Allow multiple parents for an entity
    - Fast access by following the predefined relationship
    - Allow direct access through hashing algorithm
    - Is not suitable for ad hoc query type applications
  - Commercial Product: CA-IDMS
    - Advantage CA-IDMS database is navigational and non-relational legacy sources
    - SQL is integrated in the database engine to enable SQL access to existing Advantage CA-IDMS

Network DBMS implementation

Implementation Example



Johns Hopkins Engineering

Principles of Database Systems

Module 1 / Lecture 3

Introduction to Databases

JOHNS HOPKINS
WHITING SCHOOL
of ENGINEERING

Relational DBMS

- Relational DBMS Characteristics
  - Access Method: Use primary key and foreign key for the logical relationships
    - Allow one-to-one or one-to-many relationships between entities
    - Use proper logical design with indexes, and careful query formulation to improve performance
    - Benefit from a sound theory and the SQL standard (Non-procedural, e.g., no IF-THEN-ELSE logic)
    - Vendor and platform independent
    - Provide superior ad hoc query support
    - Has been widely deployed and dominates the DBMS market

Relational DBMS (cont.)

- Relational DBMS Characteristics
  - Non-navigational: No need to follow chains or pointers. User specifies only "what", not "how". DBMS "optimizer" determines access path.
  - Properties: Atomicity, Consistency, Isolation, Durability (ACID)
  - Commercial/Open Source Products: IBM DB2 UDB, Microsoft SQL Server, Oracle, Sybase, MySQL, Postgre SQL, MariaDB, ...under mainframe, midrange, LAN, workstation, PC, …

  Note: Most vendors also support user-defined data types and other object-oriented functions, and their products become object-relational DBMS.

Object-Oriented DBMS

- Object-Oriented DBMS (ODBMS or OODBMS) Characteristics
  - Access Method: Navigational using object ids
    - Allow data and procedures to be stored together
    - Reuse the self-contained entities
    - Access data through relationships stored within the data themselves
    - Support complex objects, classes, methods, inheritance, and encapsulation
    - Is not suited for ad hoc querying as relational databases are
  - Commercial Products: ObjectDB, ObjectStore, Versant

Object-Relational DBMS

- Object-Relational DBMS (ORDBMS) Characteristics
  - Access Method: Using both key index and object ids (navigational)
    - Inherit all strengths from the RDBMS
    - Allow data and procedures to be stored together
    - Reuse the self-contained entities
    - Support complex objects, classes, methods, inheritance, and encapsulation
    - Support SQL (Structural Query Language) standards
  - Commercial Products: IBM DB2 UDB, Microsoft Server, and Oracle

NoSQL

- NoSQL Characteristics
  - Access Method: Navigational and non-navigational
  - Do not use fixed schema structures, referential integrity, defined joins, or a common storage model
  - Is good for scaling out horizontally, and is well suited for Big Data; while RDMBS is good for scaling up vertically
  - Follow BASE: Basically Available, Soft state, Eventual Consistency; do not follow the strict ACID rules of relational databases

NoSQL (cont.)

- NoSQL Characteristics
  - Four types NoSQL databases based on storage category:
    - Column-based Store: Cassandra, Hbase, Google - Big Table
    - Key Value-based Store: Amazon DynamoDB, Riak, Berkeley DB
    - Document-based Store: MongoDB, Couchbase, CouchDB, MarkLogic
    - Graph-based Store: Neo4j, OrientDB
  - Usage:
    - Big Data, schema-less design, better horizontal scalability, not mission critical applications
  - NoSQL does not replace SQL databases; it is a complementary addition to RDMBS and SQL to support business needs

Johns Hopkins Engineering

Principles of Database Systems

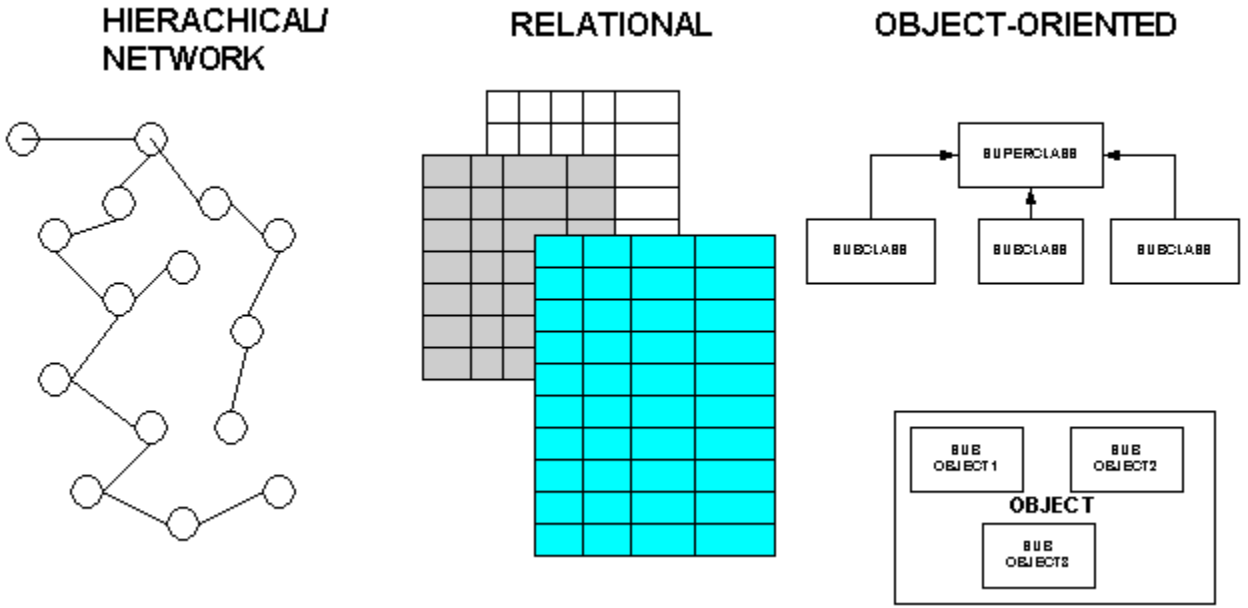
Module 1 / Lecture 4

Introduction to Databases

JOHNS HOPKINS
WHITING SCHOOL
of ENGINEERING

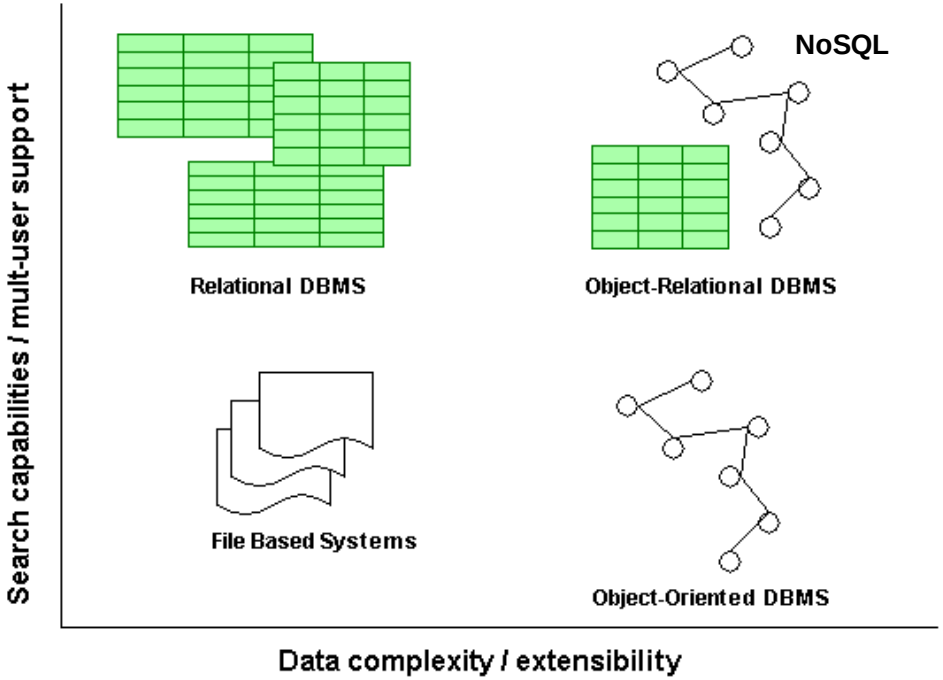
Graphical Representations for Various Data Models

Implementation Example



Data Complexity / Search Capabilities for Various Data Models

Implementation Example



Support of Multiple Views of Data

- Provide a means to present a different representation of data that resides within the base tables
- Are powerful because they allow the database designer or application developer to tailor the presentation of data to different types of users

Sharing of Data and Multi-user

- Transaction process
  - A logical unit of work on the database
  - An action or actions that perform database record reads or updates by a user or program

- Concurrency control
  - Managing simultaneous operations on the database without having them interface with one another
  - Without proper control, the updated data may be lost, cause inconsistent analysis, or cause uncommitted dependency problems

Types of Database Users

- Database Administrators (DBA)
  - Install and upgrade the DBMS on database server and application tools
  - Allocate system storage and plan for future storage requirements for the database system
  - Create primary database storage structures (tablespaces) once application developers have designed an application
  - Create primary objects (tables, views, indexes) once application developers have designed an application

Types of Database Users (cont.)

- Database Administrators (DBA)
  - Modify the database structure, as necessary, from information given by application developers
  - Enroll users and maintain system security
  - Control and monitor user access to the database
  - Monitor and optimize the performance of the database
  - Plan for back-up and recovery of database information
  - Back-up and restore the database
  - Create redundancy or replicate database instances

Types of Database Users (cont.)

- System Analysts / Database Designers / Application Developers
  - Design the database structure for an application
  - Design and develop the database application
  - Estimate storage requirements for an application
  - Specify modifications of the database structure for an application

Types of Database Users (cont.)

- System Analysts / Database Designers / Application Developers
  - Relay the above information to a database administrator
  - Tune the application during development
  - Establish an application's security measures during development
  - Remediate application vulnerabilities reported by static or dynamic scans

Types of Database Users (cont.)

- Database Users (ad-hoc, regular, or power users)
  - Enter, modify, and delete data, where permitted
  - Generate reports of data
  - Provide input of new features and enhancements for an application
  - Test new release of a database application and report issues

A DBMS Simplified Environment

- A simplified database system environment
  - Application, DBMS, Database Definition (Meta Data), Database
- Program-Data Independence
  - Programs to form an application do not depend on the data storage structure or access technique.

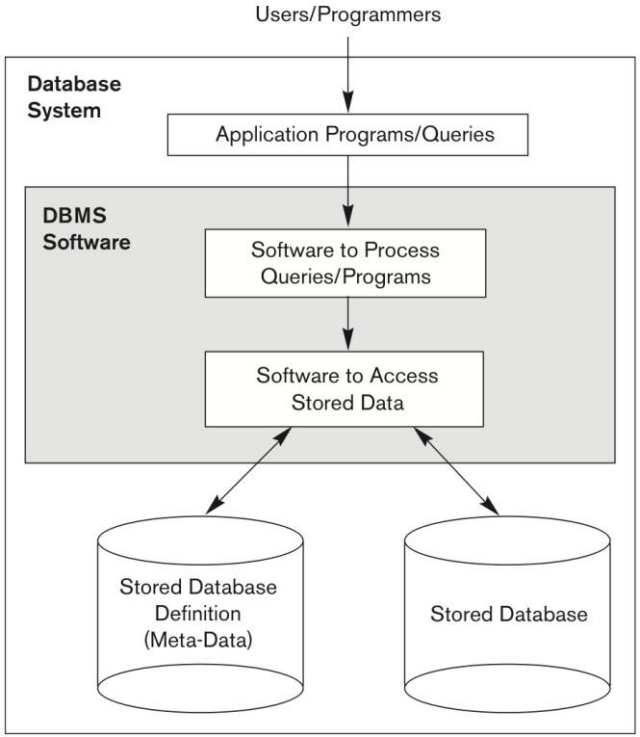


Figure 1.1 A simplified database system environment.

Data Abstraction

- Program-data independence and program-operation (function) independence.
- DBMS allows a conceptual representation of data.
- Database can grow without considering the applications.

Johns Hopkins Engineering

Principles of Database Systems

Module 1 / Lecture 5

Introduction to Databases

JOHNS HOPKINS
WHITING SCHOOL
of ENGINEERING

Data Models, Schemas, and Instances

- A data model is a set of concepts that can be used to describe the structure of a database including data types, relationships, and constraints of the data.

- The description of a database is called the database schema (meta-data). It is a collection of schema objects, such as tables, views, clusters, procedures, and packages used by a database.

Data Models, Schemas, and Instances (cont.)

- The data in the database at a particular moment is called database state (or set of occurrences or instances.) For example, STUDENT or ORDER is an instance construct. They can have multiple occurrences or instances.

- However, an Oracle instance, as a database engine, represents a set of background processes and allocated memory structure that are shared by all users to access the data in the database.

Data Models, Schemas, and Instances (cont.)

- An Oracle *database* is a collection of data that is treated as a unit. The purpose of a database is to store and retrieve related information.

- The database has *logical structures* and *physical structures*. Because the physical and logical structures are separate, the physical storage of data can be managed without affecting the access to logical storage structures.

Database Architecture - Three-schema Architecture

- External level
  - Include a number of external schemas or user views
- Conceptual level
  - Has a conceptual schema, which describes the structure of the entire database for a community of users
- Internal level
  - Has an internal schema, which describes the physical storage structure (data storage, and access path) of the database

Database Architecture - Three-schema Architecture (cont.)

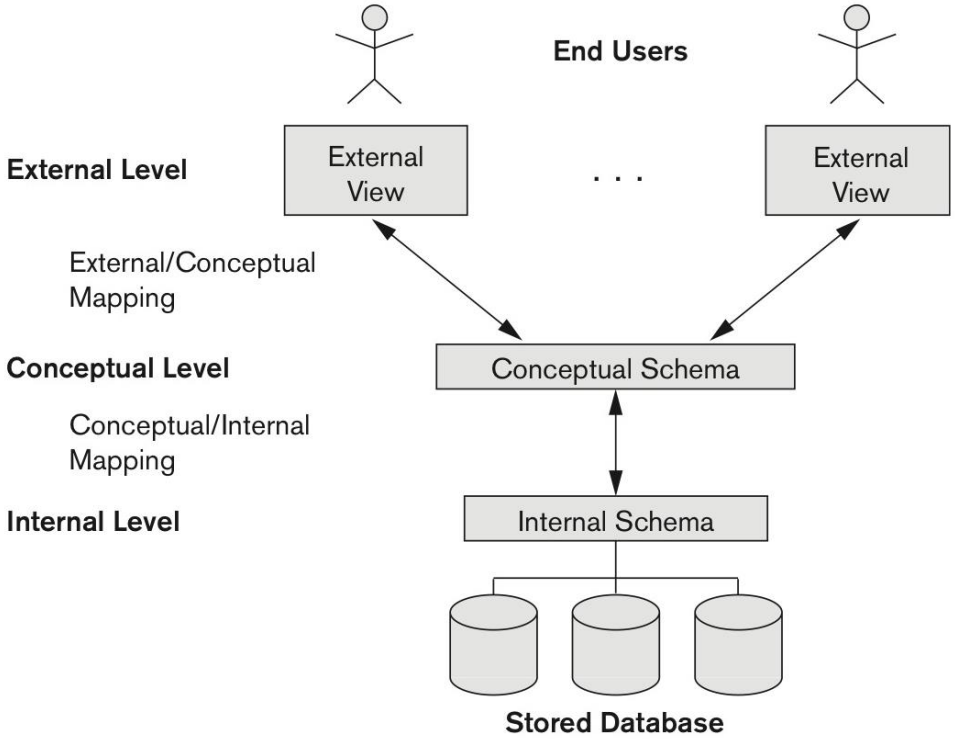


Figure 2.2 The three-schema architecture.

A DBMS Components And Their Interactions

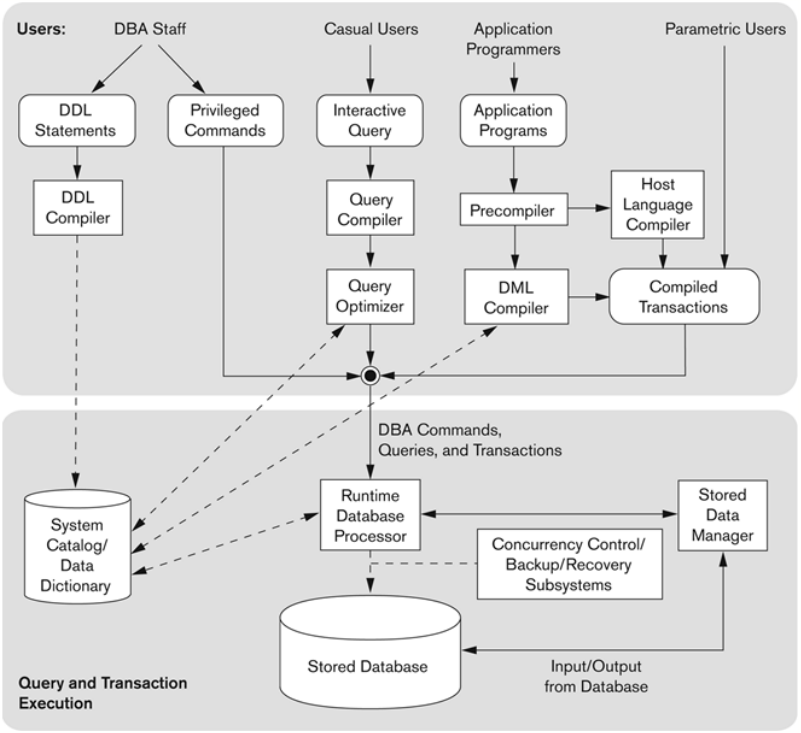
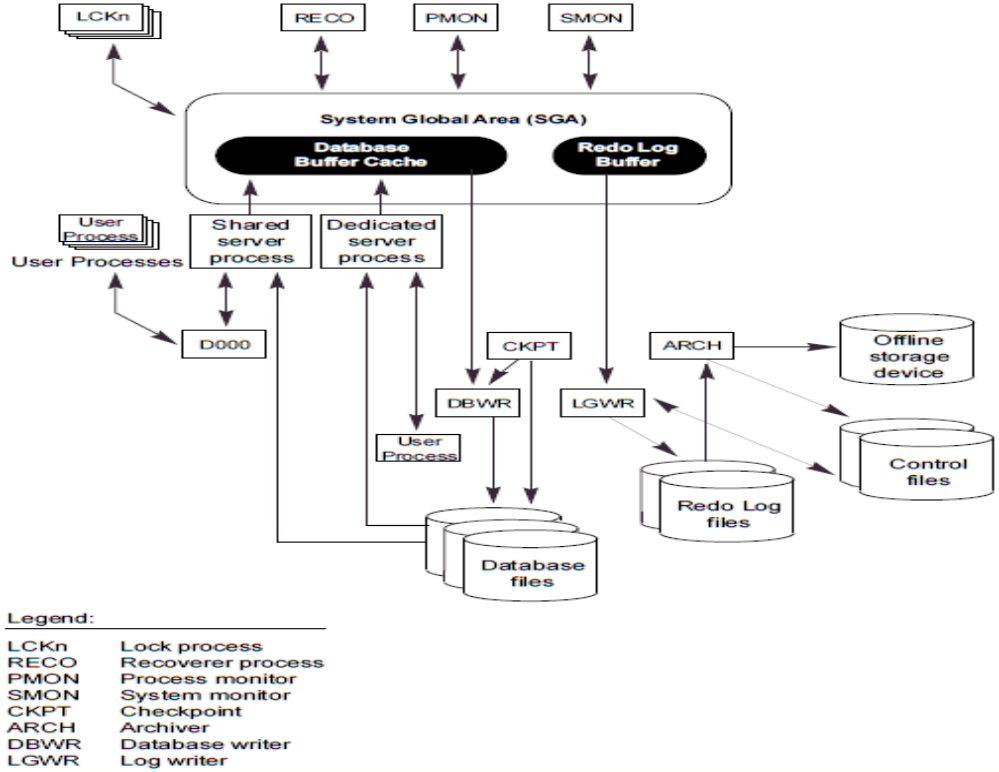


Figure 2.3 Component modules of a DBMS and their interactions.

Oracle RDBMS Architecture - An Instance



Source : Oracle

Database Languages

- Data definition language (DDL)
  - After a DBMS is chosen and the design of a database is completed, DDL is used to create a database. For RDBMS, basic commands include CREATE, ALTER, DROP, and RENAME.

- Data manipulation language (DML)
  - DML is used to perform basic operations on a database such as retrieve, insert, update, delete data; commit or rollback changes of data. These commands can be embedded as data sublanguage (DSL) into a host language.

Database Languages (cont.)

- Date query language (DQL)
  - A query language only involves data retrieval. Users may need complex queries with set operations or aggregate functions for reporting.
  - The database state remains the same.

- Data control language (DCL)
  - DCL provides user access control to a database. Basic commands are GRANT, REVOKE, AUDIT, and LOCK.

Database Interfaces

- Create forms-based applications that provide users access to information stored in a database
- Create applications with graphical user interface (GUI) environments
- Increase productivity for users by creating menu driven and mouse driven applications that do not require users to remember function keys, or understand commands and programming

Johns Hopkins Engineering

Principles of Database Systems

Module 1 / Lecture 6

Introduction to Databases

JOHNS HOPKINS
WHITING SCHOOL
of ENGINEERING

Centralized and Client/Server Architectures for DBMSs

- Centralized
  - Terminals and a DBMS server
- Basic two-tier client/server architecture
  - Clients and servers
  - Clients running programs interact with DBMS
- Three-tier client/server architecture
  - Client – GUI, Web Interfaces
  - Application server and/or Web server
  - Database server with DBMS

Classification of DBMSs

- Based on data model or DBMS implementation:
  - Hierarchical, network, relational, object-oriented and object-relational databases, NoSQL databases (based on column-based, key-value, document-based, and graph-based)
- Based on number of users:
  - Single-user, multi-user
- Based on number of sites:
  - Centralized, distributed

Classification of DBMSs (cont.)

- Based on purpose:
  - General – On-Line Transaction Processing (OLTP)
  - Data warehouse – On-Line Analytical processing (OLAP) for decision support or data mining
  - Special – XML DB, GIS, NoSQL, and others

DBMS Market

- Operational DBMSs – Oracle, Microsoft, IBM, and SAP as leaders
  - Dominate RDBMS and ORDBMS market shares
- Open-source relational DBMSs have matured significantly and can be used to replace commercial RDBMSs
  - Can be chosen for a substantial savings in Total Cost of Ownership (TCO)
  - MySQL, MariaDB, PostgreSQL and others

Continue Growth in the DBMS Market

- Maintain or increase levels of DBMS spending for most organizations

- Look for cost-savings for better business competitiveness
  - Increase the use of “open-source” RDBMSs
  - May use for mission-critical and internal applications
  - May have few functions, DB admin tools, DBA resources – can serve the needed features
  - Linux has become a leading DBMS platform
  - Oracle MySQL and Postgres SQL

Big Data

- Is a large and complex collection of data sets
  - Data characteristics/dimensions: volume, velocity, variety, and complexity
  - Data sources
    - Sensor/machine/device data (Internet of Things)
    - Unstructured content from email, office documents, etc.
    - Large audio/video/photographic data
    - Scientific/genomic data
  - NoSQL Databases and Big Data (See Ch 24 and Ch 25)