ST. XAVIER’S COLLEGE

**(Affiliated to Tribhuvan University)**

**Maitighar, Kathmandu**

****

**Database Management System**

**Theory Lab Assignment #1**

**SUBMITTED BY:**

**Siddhant Rimal**

**013BSCCSIT039**

**SUBMITTED TO**

|  |  |
| --- | --- |
| **Mr. Sanjay Kr. Yadav**  **( Lecturer )** |  |
| **Department of Computer Science** | |

Submission Date: July 24th 2015

Purpose of Database Systems

DBMS attempts to solve the following problems with conventional file-processing systems:

* Data redundancy and inconsistency
* Difficulty in accessing data
* Data isolation – multiple files and formats
* Integrity problems
* Atomicity of updates
* Concurrent access by multiple users
* Security problems [2]

These are some of the primary purposes of Database systems.

1. To see why database management systems are necessary, let's look at a typical “file-processing system” supported by a conventional operating system.

The application is a savings bank:

* + Savings account and customer records are kept in permanent system files.
  + Application programs are written to manipulate files to perform the following tasks:
    - Debit or credit an account.
    - Add a new account.
    - Find an account balance.
    - Generate monthly statements.[1]

1. Development of the system proceeds as follows:
   * New application programs must be written as the need arises.
   * New permanent files are created as required.
   * **but** over a long period of time files may be in different formats, and
   * Application programs may be in different languages. [1]
2. So we can see there are problems with the straight file-processing approach:
   * Data redundancy and inconsistency
     + Same information may be duplicated in several places.
     + All copies may not be updated properly.
   * Difficulty in accessing data
     + May have to write a new application program to satisfy an unusual request.
     + E.g. find all customers with the same postal code.
     + Could generate this data manually, but a long job...
   * Data isolation
     + Data in different files.
     + Data in different formats.
     + Difficult to write new application programs.
   * Multiple users
     + Want concurrency for faster response time.
     + Need protection for concurrent updates.
     + E.g. two customers withdrawing funds from the same account at the same time - account has $500 in it, and they withdraw $100 and $50. The result could be $350, $400 or $450 if no protection.
   * Security problems
     + Every user of the system should be able to access only the data they are permitted to see.
     + E.g. payroll people only handle employee records, and cannot see customer accounts; tellers only access account data and cannot see payroll data.
     + Difficult to enforce this with application programs.
   * Integrity problems
     + Data may be required to satisfy constraints.
     + E.g. no account balance below $25.00.
     + Again, difficult to enforce or to change constraints with the file-processing approach. [1]

These problems and others led to the development of **database management systems**. [1]

View of Data

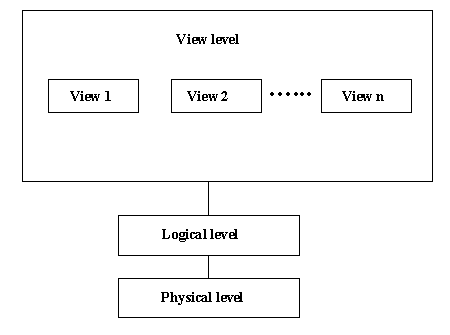


Figure 1: View of Data[2]

In database theory, a view is the result set of a stored query on the data, which the database users can query just as they would in a persistent database collection object.

Database Languages

Database language is a generic term referring to a class of languages used for defining and accessing databases. A particular database language will be associated with a particular database management system. There are two distinct classes of database language: those that do not provide complete programming facilities and are designed to be used in association with some general-purpose programming language (the host language), and those that do provide complete programming facilities (database programming languages). [3]

Data Definition Language(DDL)

* Specification notation for defining database schema
* DDL compiler generates a set of tables stored in data dictionary
* Data dictionary contains metadata (i.e. data about data)
* Data Storage and Definition Language – Special type of DDL in which the storage structure and success methods used by the database systems are specified[4]

Data Manipulation Language(DML)

* Language for accessing and manipulating the data organized by the appropriate data model
* Two classes of languages
  + Procedural : user specifies what data is required and how to get those data
  + Non-procedural : user specifies what data is required without specifying how to get those data

Relational Databases

A relational database (RDB) is a collective set of multiple data sets organized by tables, records and columns. RDBs establish a well-defined relationship between database tables. Tables communicate and share information, which facilitates data searchability, organization and reporting.[5]  
  
RDBs use Structured Query Language (SQL), which is a standard user application that provides an easy programming interface for database interaction.[5]  
  
RDB is derived from the mathematical function concept of mapping data sets and was developed by Edgar F. Codd.[5]

DBs organize data in different ways. Each table is known as a relation, which contains one or more data category columns. Each table record (or row) contains a unique data instance defined for a corresponding column category. One or more data or record characteristics relate to one or many records to form functional dependencies. [5]

These are classified as follows:

* One to One: One table record relates to another record in another table.
* One to Many: One table record relates to many records in another table.
* Many to One: More than one table record relates to another table record.
* Many to Many: More than one table record relates to more than one record in another table.[5]

RDB performs "select", "project" and "join" database operations, where select is used for data retrieval, project identifies data attributes, and join combines relations. [5]  
  
RDBs have many other advantages, including:

* Easy extendability, as new data may be added without modifying existing records. This is also known as scalability.
* New technology performance, power and flexibility with multiple data requirement capabilities.
* Data security, which is critical when data sharing is based on privacy. For example, management may share certain data privileges and access and block employees from other data, such as confidential salary or benefit information.[5]

Database Design

Database design is a process of modeling an enterprise in the real world. In fact, a database itself is a model of the real world that contains selected information needed by the enterprise. Many models and languages—some formally and mathematically defined, some informal and intuitive—are used by designers. Here are the ones that we present in this tutorial[6]:

• The **Unified Modeling Language** (UML) was designed for software engineering of large systems using object-oriented (OO) programming languages. UML is a very large language; we will use only a small portion of it here, to model those portions of an enterprise that will be represented in the database. It is our tool for communicating with the client in terms that are used in the enterprise. [6]

• The **Entity-Relationship** (ER) model is used in many database development systems. There are many different graphic standards that can represent the ER model. Some of the most modern of these look very similar to the UML class diagram, but may also include elements of the relational model. [6]

• The **Relational Model** (RM) is the formal model of a database that was developed for IBM in the early 1970s by Dr. E.F. Codd. It is largely based on set theory, which makes it both powerful and easy to implement in computers. All modern relational databases are based on this model. We will use it to represent information that does not (and should not) appear in the UML model but is needed for us to build functioning databases. [6]

• **Relational Algebra** (RA) is a formal language used to symbolically manipulate objects of the relational model. [6]

• The **table model** is an informal set of terms for relational model objects. These are the terms used most often by database developers. [6]

• The **Structured Query Language** (SQL, pronounced “sequel” or “ess-que-ell”) is used to build and manipulate relational databases. It is based on relational algebra, but provides additional capabilities that are needed in commercial systems. It is a declarative, rather than a procedural, programming language. There is a standard for this language, but products vary in how closely they implement it.[6]

Object-based and semi-structured databases

**Object DATABASE OR Object oriented database management system** is a database in which the information is represented in form of object as used in object-oriented programming. It is different from rational database. This type of database is used when there is complex data or/and multiple data relationships. It has a many-to-many object relationship. It should not be used when there are few join tables and there are large volums of simple transaction data. [7]

*It works well with the following application:*  
  
**-->** Multimedia Application.  
**-->** CAS Application [7]

### Features of Object Oriented Database:

* It support transactions.
* It supply querying in bulk data.
* Concurrent Access
* Security[7]

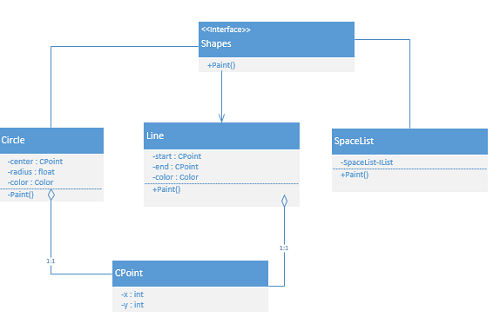


Figure 2: Object Oriented Database[7]

In **Semi-Structured Database** the data are in the form of structured data that edoes not conform with the formal structure of data models associated with rational databases or other form of data. Therefore, it is also known as self-describing structure.[7]

### Types of Semi-Structured Database:

* XML semi-structured database
* JSON (JavaScript Object Notation)semi-structured database[7]

**Advantages of Semi-Structured Database**

* It can show the information of data source that is not constrained by schema.
* It is used to view structured data as semi-structured data.
* The data transfer format may be portable. [7]

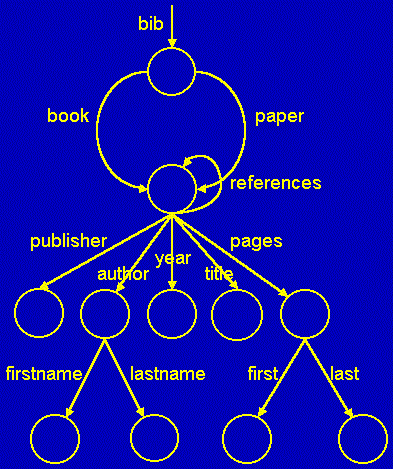


Figure 3: Semi-Structured Database[7]

Data Storage and Querying

**Data storage:** Database storage is the container of the physical materialization of a database. It comprises the *internal* (physical) *level* in the database architecture. It also contains all the information needed (e.g., metadata, "data about the data", and internal data structures) to reconstruct the *conceptual level* and *external level* from the internal level when needed. Putting data into permanent storage is generally the responsibility of the database engine a.k.a. "storage engine". Though typically accessed by a DBMS through the underlying operating system (and often utilizing the operating systems' file systems as intermediates for storage layout), storage properties and configuration setting are extremely important for the efficient operation of the DBMS, and thus are closely maintained by database administrators. A DBMS, while in operation, always has its database residing in several types of storage (e.g., memory and external storage). The database data and the additional needed information, possibly in very large amounts, are coded into bits. Data typically reside in the storage in structures that look completely different from the way the data look in the conceptual and external levels, but in ways that attempt to optimize (the best possible) these levels' reconstruction when needed by users and programs, as well as for computing additional types of needed information from the data (e.g., when querying the database).[9]

Some DBMSs support specifying which character encoding was used to store data, so multiple encodings can be used in the same database.

Various low-level database storage structures are used by the storage engine to serialize the data model so it can be written to the medium of choice. Techniques such as indexing may be used to improve performance. Conventional storage is row-oriented, but there are also column-oriented and correlation databases.

**Querying:**  A query is a request for information from a database. There are three general methods for posing queries:

* **Choosing parameters from a menu**: In this method, the database system presents a list of parameters from which you can choose. This is perhaps the easiest way to pose a query because the menus guide you, but it is also the least flexible.
* **Query by example (QBE):** In this method, the system presents a blank record and lets you specify the fields and values that define the query.
* **Query language:** Many database systems require you to make requests for information in the form of a stylized query that must be written in a special query language. This is the most complex method because it forces you to learn a specialized language, but it is also the most powerful.[9]

Transaction Management

A transaction is a collection of operations that performs a

* single logical function in a database application
* Transaction-management component ensures that thedatabase 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.

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

Database users and Administrators

Coordinates all the activities of the database system; the database administrator has a good understanding of the enterprise’s information resources and needs.

* Database administrator’s duties include:
* Schema definition
* Storage structure and access method definition
* Schema and physical organization modification
* Granting user authority to access the database
* Specifying integrity constraints
* Acting as liaison with users
* Monitoring performance and responding to changes in Requirements.

Overall Structure

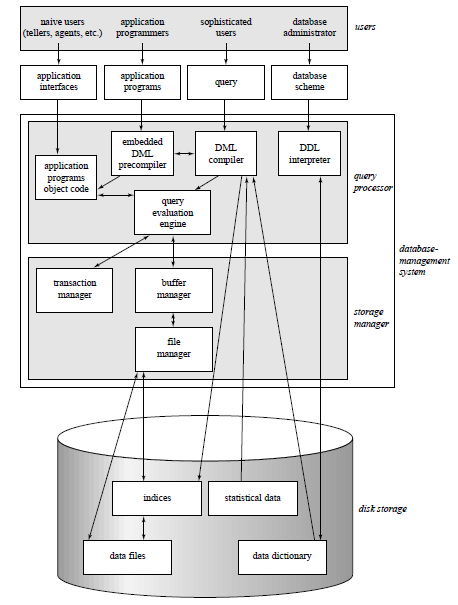


Figure 4: Overall Structure [1]

History of Database Systems

**1980’s**

* Birth of IBM PC.
* RDBMS market begins to boom.
* SQL becomes standardized through ANSI (American National Standards Institute) and ISO (International Organization for Standardization)
* By Mid 80’s it had become apparent that there were some fields(medicine, multimedia, physics) where relational databases were not practical, due to the types of data involved.
* More flexibility was needed in how their data was represented and accessed.
* This led to research in Object Oriented Databases in which users could define their own methods of access to data and how to represent and manipulate it. This coincided with the introduction of Object Oriented Programming languages such as C++ which started to appear[8]

**1990’s**

* First OODBMS’ start to appear from companies like Objectivity. Object Relational DBMS’ hybrids also begin to appear.
* Industry shakeout begins with fewer surviving companies offering increasingly complex products at higher prices. Much of the development centers on client tools for application development such as: PowerBuilder(Sybase), Oracle Developer, Visual Basic, etc
* Development of personal/small business productivity tools such as Excel and Access from Microsoft.
* New application areas: Data warehousing and OLAP(Online Analytical Processing, a category of software tools that provides analysis of data stored in a database), internet, multimedia, etc[8]

**Late 90’s – 2000’s**

* Large investment in internet companies fuels tools-market boom for Web/Internet/DB connectors:
  + Active Server Pages, Front page, Java Servlets, JDBC, Java Beans, ColdFusion, Dream Weaver, Oracle Developer 2000, etc
* Open source projects come online with widespread use of gcc,cgi, Apache, MySQL
* Three main companies dominate in the large DB market: IBM, Microsoft, and Oracle[8]

References:

1. “Purpose of Database Systems”. PDF. Simon Fraser University Website. Osmar R. Zaï ane. url: <http://www.cs.sfu.ca/CourseCentral/354/zaiane/material/notes/Chapter1/node3.html> 10/09/1995 [24/07/2015]
2. “Purpose of Database Systems”. PPT Slide. University at Buffalo. url: <http://www.cse.buffalo.edu/~bina/cse462/lectures/lec1/sld003.htm> 2015 [24/07/2015]
3. “A dictionary of Computing”. Book. John Dantith url: <http://www.encyclopedia.com/doc/1O11-databaselanguage.html> 24/07/2004 [24/072015]
4. “Database Systems Concepts”. PDF. Silberschatz, Korth and Sudarshan. url: <http://cs.nyu.edu/courses/spring01/G22.2433-001/mod1.2> 1997 [24/07/2015]
5. “Relational Database(RDB)”. Internet. Cary Janssen. url: <http://www.techopedia.com/definition/1234/relational-database-rdb> 2015[24/07/2015]
6. “Models and Languages”. Intenet. url: <http://www.tomjewett.com/dbdesign/dbdesign.php?page=models.html> 2015 [24/07/2015]
7. “Object and Semi-structure based database”. Internet. url: <http://www.assignmenthelp.net/assignment_help/object-based-and-semistructured-database-assignment-help> 2015 [24/07/2015]
8. “Database Management Systems”. Internet. url: [*http://www.techopedia.com/definition/24361/database-management-systems-dbms*](http://www.techopedia.com/definition/24361/database-management-systems-dbms) 2015 [24/07/2015]
9. “Role And Advantages Of The Dbms Information Technology Essay”. Internet. url: <http://www.uniassignment.com/essay-samples/information-technology/role-and-advantages-of-the-dbms-information-technology-essay.php> 2015 [24/07/2015]