GPCET
Pioneering Innovative Education

G. PULLAIAH COLLEGE OF ENGINEERING AND TECHNOLOGY

Accredited by NAAC with 'A' Grade of UGC, Approved by AICTE, New Delhi

Permanently Affiliated to JNTUA, Ananthapuramu

(Recognized by UGC under 2(f) and 12(B) & ISO 9001:2008 Certified Institution)

Nandikotkur Road, Venkayapalli, Kurnool – 518452

Department of Computer Science and Engineering

Bridge Course
On
Database Management Systems

By

Dr. S. Prem Kumar

GPCET
Pioneering Innovative Education

Table of Contents

1.0 Data and Information

1.1 Data: Information in raw or unorganized form (such as alphabets, numbers, or symbols) that refer to, or represent, conditions, ideas, or objects. Data is limitless and present everywhere in the universe.

Data is measured, collected and reported, and analyzed, whereupon it can be visualized using graphs, images or other analysis tools. Data as a general concept refers to the fact that some existing information or knowledge is represented or coded in some form suitable for better usage or processing. Raw data ("unprocessed data") is a collection of numbers or characters before it has been "cleaned" and corrected by researchers. Raw data needs to be corrected to remove outliers or obvious instrument or data entry errors (e.g., a thermometer reading from an outdoor Arctic location recording a tropical temperature). Data processing commonly occurs by stages, and the "processed data" from one stage may be considered the "raw data" of the next stage. Experimental data is data that is generated within the context of a scientific investigation by observation and recording. Data has been described as the new oil of the digital economy.

1.2 Information: Data that is (1) accurate and timely, (2) specific and organized for a purpose, (3) presented within a context that gives it meaning and relevance, and (4) can lead to an increase in understanding and decrease in uncertainty. Information is valuable because it can affect behavior, a decision, or an outcome.

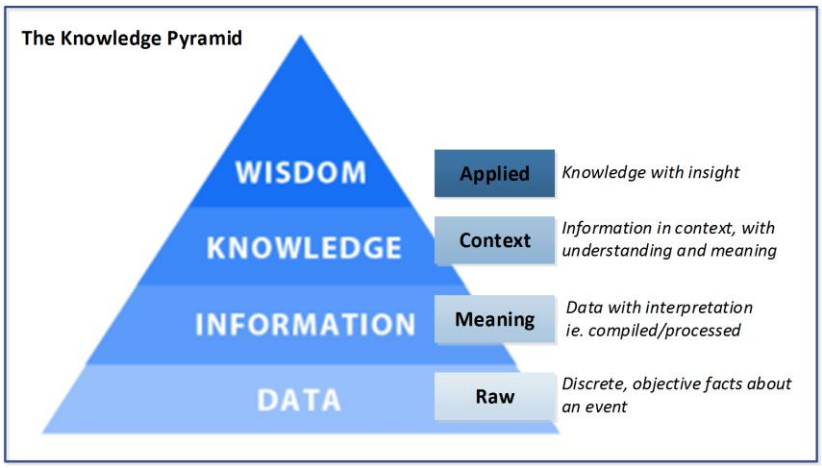

Fig 1.1: Knowledge Pyramid

1.3 DBMS: A database-management system is a collection of interrelated data and a set of programs to access those data. The collection of data, usually referred to as the database, contains information relevant to an enterprise. The primary goal of a DBMS is to provide a way to store and retrieve database information that is both convenient and efficient.

Database systems are designed to manage large bodies of information. Management of data involves both defining structures for storage of information and providing mechanisms for the manipulation of information.

In addition, the database system must ensure the safety of the information stored, despite system crashes or attempts at unauthorized access. If data are to be shared among several users, the system must avoid possible anomalous results.

1.4 RDBMS: A relational database is a collection of data items organized as a set of formally-described *tables* from which data can be accessed or reassembled in many different ways without having to reorganize the database tables. The relational database was invented by E. F. Codd at IBM in 1970.

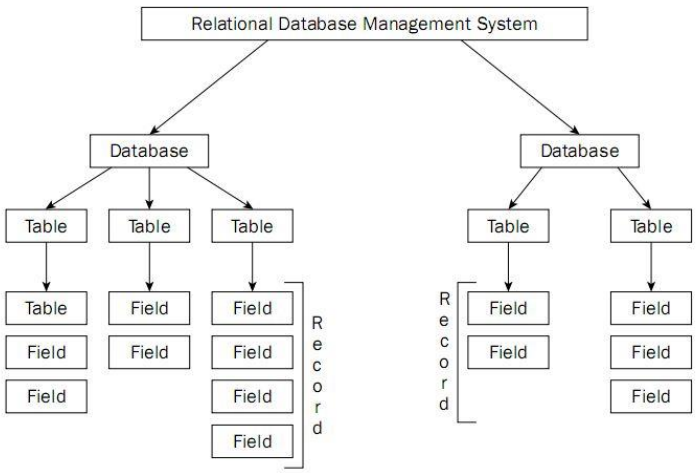

Fig 1.2: Structure of RDBMS

In RDBMS, Data bases are created with login and passwords. Inside the logins, various tables are created which contains rows and columns. The columns are created with fields. Each field contains data. The row of data becomes a record. The row of data at a particular moment of time is called *instance*.

1.4.1 Features of RDBMS: The various features supported by RDBMS are shown in the following diagram

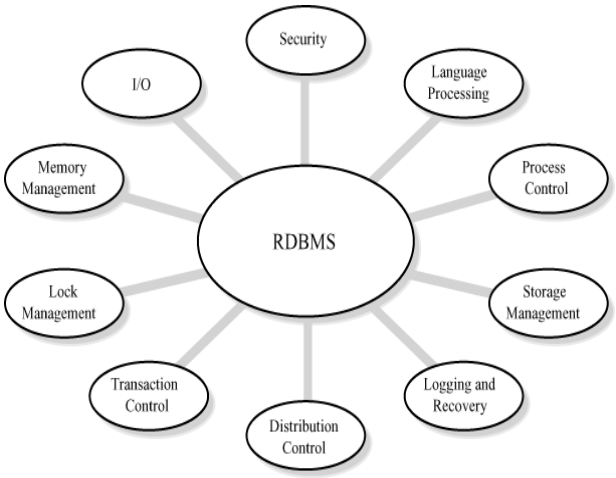

Fig 1.3: Feature of RDBMS

1.5 OORDBMS: An object-relational database (ORD), or object-relational database management system (ORDBMS), is a database management system (DBMS) similar to a relational database, but with an object-oriented database model: *objects*, *classes* and *inheritance* are directly supported in database schemas and in the query language.

1.6 MMDB: A Multimedia database (MMDB) is a collection of related multimedia data. The multimedia data include one or more primary media data types such as text, images, graphic objects (including drawings, sketches and illustrations) animation sequences, audio and video.

1.7 MDB: A mobile database is a database that resides on a mobile device such as a PDA, a smart phone, or a laptop. Such devices are often limited in resources such as memory, computing power, and battery power.

1.8 Spatial database is a database that is optimized to store and query data that represents objects defined in a geometric space. Most spatial databases allow representing simple geometric objects such as points, lines and polygons.

1.9 A **Web database** is a wide term for managing data online. A web database gives you the ability to build your own databases/data storage without you being a database guru or even a technical person. *FormLogix* web database solution will help you to easily create database driven web pages and host your database online.

2.0 History of Database Systems

1950s and early 1960s:

–Data processing using magnetic tapes for storage
Tapes provide only sequential access
–Punched cards for input

Late 1960s and 1970s:

–Hard disks allow direct access to data
–Network and hierarchical data models in widespread use
–Ted Codd defines the relational data model. Won 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

1980s:

–Research relational prototypes evolve into commercial systems
-SQL becomes industry 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
2000s:
–XML and XQuery standards
–Automated database administration
–Increasing use of highly parallel database systems
–Web-scale distributed data storage systems

3.0 Database-System Applications

Databases are widely used. Here are some representative applications:

- **Enterprise Information:**
  - Sales: For customer, product, and purchase information.
  - Accounting: For payments, receipts, account balances, assets and otheraccounting information.
  - Human resources: For information about employees, salaries, payroll taxes, and benefits, and for generation of paychecks.
  - Manufacturing: For management of the supply chain and for tracking production of items in factories, inventories of items inwarehouses and stores, and orders for items.
  - Online retailers: For sales data noted above plus online order tracking, generation of recommendation lists, and maintenance of online product evaluations.

- **Banking and Finance**
  - Banking: For customer information, accounts, loans, and banking transactions.
  - Credit card transactions: For purchases on credit cards and generation of monthly statements.
  - Finance: For storing information about holdings, sales, and purchases of financial instruments such as stocks and bonds; also for storing real-time market data to enable online trading by customers and automated trading by the firm.
- **Universities:** For student information, course registrations, and grades (in addition to standard enterprise information such as human resources and accounting).
- **Airlines:** For reservations and schedule information. Airlines were among the first to use databases in a geographically distributed manner.
- **Telecommunication:** For keeping records of calls made, generating monthly bills, maintaining balances on prepaid calling cards, and storing information about the communication networks.

4.0 QUERY AND OPERATORS

Query: A query is a question, often expressed in a formal way. A database query can be either a select query or an action query. A select query is a data retrieval query, while an action query asks for additional operations on the data, such as insertion, updating or deletion.

In order to perform operations on database with the help of query, the following operators are used

4.1 Unary Operators: A unary operator uses only one operand. A unary operator typically appears with its operand in the following format.

<Operator> <operand>

4.2 Binary Operators: A binary operator uses two operands. A binary operator appears with its operands in the following format.

<operand1><operator><operand2>

4.3 Set Operators: Set operators combine sets of rows returned by queries, instead of individual data items. All set operators have equal precedence. Oracle Database Lite supports the following set operators.

- UNION
- UNION ALL
- INTERSECT
- MINUS

Table shows Levels of Precedence of the Oracle Database Lite SQL Operators

Precedence Level	SQL Operator
1	Unary + - arithmetic operators, PRIOR operator
2	* / arithmetic operators
3	Binary + - arithmetic operators, || character operators
4	All comparison operators
5	NOT logical operator
6	AND logical operator
7	OR logical operator

4.4 Other Operators: Other operators with special formats accept more than two operands. If an operator receives a null operator, the result is always null. The only operator that does not follow this rule is CONCAT.

4.5 Arithmetic Operators: Arithmetic operators manipulate numeric operands. The '-' operator is also used in date arithmetic. Supported arithmetic operators are listed in below table.

Operator	Description	Example
+ (unary)	Makes operand positive	SELECT +3 FROM DUAL;
- (unary)	Negates operand	SELECT -4 FROM DUAL;
/	Division (numbers and dates)	SELECT SAL / 10 FROM EMP;
*	Multiplication	SELECT SAL * 5 FROM EMP;
+	Addition (numbers and dates)	SELECT SAL + 200 FROM EMP;

Operator	Description	Example
-	Subtraction (numbers and dates)	SELECT SAL - 100 FROM EMP;

4.6 Character Operators: Character operators used in expressions to manipulate character strings are listed in Table.

Operator	Description	Example
||	Concatenates character strings	SELECT 'The Name of the employee is: ' || ENAME FROM EMP;

4.7 Comparison Operators: Comparison operators used in conditions that compare one expression with another are listed in below table. The result of a comparison can be TRUE, FALSE, or UNKNOWN.

Operator	Description	Example
=	Equality test.	SELECT ENAME "Employee" FROM EMP WHERE SAL = 1500;
!=, ^=, <>	Inequality test.	SELECT ENAME FROM EMP WHERE SAL ^= 5000;
>	Greater than test.	SELECT ENAME "Employee", JOB "Title" FROM EMP WHERE SAL > 3000;
<	Less than test.	SELECT * FROM PRICE WHERE MINPRICE < 30;
>=	Greater than or equal to test.	SELECT * FROM PRICE WHERE MINPRICE >= 20;
<=	Less than or equal to test.	SELECT ENAME FROM EMP WHERE SAL <= 1500;
IN	"Equivalent to any member of" test. Equivalent to "=ANY".	SELECT * FROM EMP WHERE ENAME IN ('SMITH', 'WARD');
ANY/ SOME	Compares a value to each value in a list or returned by a query. Must be preceded by =, !=, >, <, <= or >=. Evaluates to FASLE if the query returns no rows.	SELECT * FROM DEPT WHERE LOC = SOME ('NEW YORK','DALLAS');
NOT IN	Equivalent to "!=ANY". Evaluates to FALSE if any member of the set is NULL.	SELECT * FROM DEPT WHERE LOC NOT IN ('NEW YORK', 'DALLAS');
ALL	Compares a value with every value in a list or returned by a query. Must be preceded by =, !=, >, <,	SELECT * FROM emp WHERE sal >= ALL (1400,

Operator	Description	Example
	<= or >=. Evaluates to TRUE if the query returns no rows.	3000);
[NOT] BETWEEN x and y	[Not] greater than or equal to x and less than or equal to y.	SELECT ENAME, JOB FROM EMP WHERE SAL BETWEEN 3000 AND 5000;
EXISTS	TRUE if a sub-query returns at least one row.	SELECT * FROM EMP WHERE EXISTS (SELECT ENAME FROM EMP WHERE MGR IS NULL);
x [NOT] LIKE y [ESCAPE z]	TRUE if x does [not] match the pattern y. Within y, the character "%" matches any string of zero or more characters except null. The character "_" matches any single character. Any character following ESCAPE is interpreted literally, useful when y contains a percent (%) or underscore (_).	SELECT * FROM EMP WHERE ENAME LIKE '%E%';
IS [NOT] NULL	Tests for nulls. This is the only operator that should be used to test for nulls.	SELECT * FROM EMP WHERE COMM IS NOT NULL AND SAL > 1500;

4.8 Logical Operators: Logical operators which manipulate the results of conditions are listed in below table.

Operator	Description	Example
NOT	Returns TRUE if the following condition is FALSE. Returns FALSE if it is TRUE. If it is UNKNOWN, it remains UNKNOWN.	SELECT * FROM EMP WHERE NOT (job IS NULL) SELECT * FROM EMP WHERE NOT (sal BETWEEN 1000 AND 2000)
AND	Returns TRUE if both component conditions are TRUE. Returns FALSE if either is FALSE; otherwise returns UNKNOWN.	SELECT * FROM EMP WHERE job='CLERK' AND deptno=10
OR	Returns TRUE if either component condition is TRUE. Returns FALSE if both are FALSE. Otherwise, returns UNKNOWN.	SELECT * FROM emp WHERE job='CLERK' OR deptno=10

4.9 Set Operators: Set operators which combine the results of two queries into a single result are listed in table.

Operator	Description	Example
UNION	Returns all distinct rows selected by either query.	SELECT * FROM (SELECT ENAME FROM EMP WHERE JOB = 'CLERK' **UNION** SELECT ENAME FROM EMP WHERE JOB = 'ANALYST');
UNION ALL	Returns all rows selected by either query, including all duplicates.	SELECT * FROM (SELECT SAL FROM EMP WHERE JOB = 'CLERK' **UNION** SELECT SAL FROM EMP WHERE JOB = 'ANALYST');
INTERSECT and INTERSECT ALL	Returns all distinct rows selected by both queries.	SELECT * FROM orders_list1 **INTERSECT** SELECT * FROM orders_list2
MINUS	Returns all distinct rows selected by the first query but not the second.	SELECT * FROM (SELECT SAL FROM EMP WHERE JOB = 'PRESIDENT' **MINUS** SELECT SAL FROM EMP WHERE JOB = 'MANAGER');

5.0 Popular databases

- **Oracle 12c:** Oracle is consistently at the top of lists of popular databases. The newest version of Oracle, 12c, is designed for the cloud and can be hosted on a single server or multiple servers, and it enables the management of databases holding billions of records. Some of the features of the latest version of Oracle include a grid framework and the use of both physical and logical structures.
- **MySQL:** MySQL is one of the most popular databases for web-based applications. This database engine allows you to select from a variety of storage engines that enable you to change the functionality of the tool and handle data from different table types.
- **Microsoft SQL Server**: It is one of the most popular databases for web-based applications. It's freeware. This database management engine works on cloud-based servers as well as local servers, and it can be set up to work on both at the same time. Not long after the release of Microsoft SQL Server 2016, Microsoft made it available on Linux as well as Windows-based platforms.
- **PostgreSQL:** It is one of several free popular databases, and it is frequently used for web databases. It was one of the first database management systems to be developed, and it allows users to manage both structured and unstructured data. It can also be used on most major platforms, including Linux-based ones, and it's fairly simple to import information from other database types using the tool.
- **MongoDB:** MongoDB is designed for applications that use both structured and unstructured data. The database engine is very versatile, and it works by connecting databases to applications via MongoDB database drivers. It's fast and easy to use. The engine supports JSON and other NoSQL documents. Data of any structure can be stored and accessed quickly and easily. Schema can be written without downtime.
- **MariaDB:** This database management system is free, and also offers paid versions. There are variety of plug-ins available for it, and it's the fastest growing open-source database available. The system is fast and stable. Extensible architecture and plug-ins allow you to

customize the tool to match your needs. Encryption is available at network, server and application levels.

- **DB2:** Created by IBM, DB2 is a database engine that has NoSQL capabilities, and it can read JSON and XML files. Works on Windows, Linux and Unix. One, in particular, was an improvement of BLU Acceleration, which is designed to make this database engine work faster through data skipping technology. Blu Acceleration can make the most of available resources for enormous databases. It can be hosted from the cloud, a physical server or both at the same time. Multiple jobs can be run at once using the Task Scheduler. Error codes and exit codes can determine which jobs are run via the Task Scheduler.
- **SAP HANA:** Designed by SAP SE, SAP HANA is a database engine that is column-oriented and can handle SAP and non-SAP data. The engine is designed to save and retrieve data from applications and other sources across multiple tiers of storage. Along with being able to be hosted from physical servers, it can also be hosted from the cloud. It supports SQL, OLTP and OLAP. The engine reduces resource requirements through compression. Data is stored in memory, reducing access times, in some cases, significantly. Real-time reporting and inventory management are available. It can interface with a number of other applications.

6.0 Mathematical concepts for Database

6.1. Relational algebra is a procedural query language, which takes instances of relations as input and yields instances of relations as output. It uses operators to perform queries. An operator can be either unary or binary.

Relational algebra received little attention outside of pure mathematics until the publication of E.F. Codd's relational model of data in 1970. Codd proposed such an algebra as a basis for database query languages. Five primitive operators of Codd's algebra are the selection, the projection, the Cartesian product (also called the cross product or cross join), the set union, and the set difference.

6.2 Tuple calculus is a calculus that was introduced by Edgar F. Codd as part of the relational model, in order to provide a declarative database-query language for this data model. It formed the inspiration for the database-query languages QUEL and SQL, of which the latter, although far less faithful to the original relational model and calculus, is now the de facto standard database-query language. Lacroix and Pirotte proposed domain calculus, which is closer to first-order logic and together with Codd showed that both of these calculi (as well as relational algebra) are equivalent in expressive power. Subsequently, query languages for the relational model
were called relationally complete if they could express at least all of these queries.
Examples of query expressions are:

$$\{ t : \{name\} \mid \exists\, s : \{name, wage\} (Employee(s) \wedge s.wage = 50.000 \wedge t.name = s.name) \}$$

6.3 domain relational calculus (DRC) is a calculus that was introduced by Michel Lacroix and Alain Pirotte as a declarative database query language for the relational data model. In DRC, queries have the form:

$$\{\langle X_1, X_2, \ldots, X_n \rangle \mid p(\langle X_1, X_2, \ldots, X_n \rangle)\}$$

where each Xi is either a domain variable or constant. The result of the query is the set of tuples X1 to Xn that make the DRC formula true. This language uses the same operators as tuple calculus,

the logical connectives ∧ (and), ∨ (or) and ¬ (not). The existential quantifier (∃) and the universal quantifier (∀) can be used to bind the variables.

7.0 Purpose of Database Systems

One way to keep the information on a computer is to store it in operating system files. To allow users to manipulate the information, the system has a number of application programs that manipulate the files. This typical file-processing system is supported by a conventional operating system. The system stores permanent records in various files, and it needs different application programs to extract records from, and add records to, the appropriate files. Keeping organizational information in a file-processing system has a number
of major disadvantages:

Data redundancy and inconsistency: Since different programmers create the files and application programs over a long period, the various files are likely to have different structures and the programs may be written in several programming languages. Moreover, the same information may be duplicated in several places (files). This redundancy leads to higher storage and access cost. In addition, it may lead to data inconsistency; that is, the various copies of the same data may no longer agree.

Difficulty in accessing data: The conventional file-processing environments do not allow needed data to be retrieved in a convenient and efficient manner. More responsive data-retrieval systems are required for general use.

Data isolation: Because data are scattered in various files, and files may be in different formats, writing new application programs to retrieve the appropriate data is difficult.

Integrity problems: The data values stored in the database must satisfy certain types of consistency constraints. Developers enforce these constraints in the system by adding appropriate code in the various application programs. However, when new constraints are added, it is difficult to change the programs to enforce them. The problem is compounded when constraints involve several data items from different files.

Atomicity problems: A computer system, like any other device, is subject to failure. In many applications, it is crucial that, if a failure occurs, the data be restored to the consistent state that existed prior to the failure. It must happen in its entirety or not at all. It is difficult to ensure atomicity in a conventional file-processing system.

Concurrent-access anomalies: For the sake of overall performance of the system and faster response, many systems allow multiple users to update the data simultaneously. Indeed, today, the largest Internet retailers may have millions of accesses per day to their data by shoppers. In such an environment, interaction of concurrent updates is possible and may result in inconsistent data.

Security problems: Not every user of the database system should be able to access all the data.

But, since application programs are added to the file-processing system in an ad hoc manner, enforcing such security constraints is difficult.