



DBMS

database management system

tutorialspoint

SIMPLY EASY LEARNING

www.tutorialspoint.com



<https://www.facebook.com/tutorialspointindia>



<https://twitter.com/tutorialspoint>

About the Tutorial

Database Management System or DBMS in short refers to the technology of storing and retrieving users' data with utmost efficiency along with appropriate security measures. DBMS allows its users to create their own databases as per their requirement. These databases are highly configurable and offer a bunch of options.

This tutorial explains the basics of DBMS such as its architecture, data models, data schemas, data independence, E-R model, relation model, relational database design, and storage and file structure. In addition, it covers a few advanced topics such as indexing and hashing, transaction and concurrency, and backup and recovery.

Audience

This tutorial will especially help computer science graduates in understanding the basic-to-advanced concepts related to Database Management Systems.

Prerequisites

Before you start proceeding with this tutorial, it is recommended that you have a good understanding of basic computer concepts such as primary memory, secondary memory, and data structures and algorithms.

Copyright & Disclaimer

© Copyright 2015 by Tutorials Point (I) Pvt. Ltd.

All the content and graphics published in this e-book are the property of Tutorials Point (I) Pvt. Ltd. The user of this e-book is prohibited to reuse, retain, copy, distribute or republish any contents or a part of contents of this e-book in any manner without written consent of the publisher.

We strive to update the contents of our website and tutorials as timely and as precisely as possible, however, the contents may contain inaccuracies or errors. Tutorials Point (I) Pvt. Ltd. provides no guarantee regarding the accuracy, timeliness or completeness of our website or its contents including this tutorial. If you discover any errors on our website or in this tutorial, please notify us at contact@tutorialspoint.com

Table of Contents

About the Tutorial	i
Audience.....	i
Prerequisites.....	i
Copyright & Disclaimer	i
Table of Contents.....	ii
1. OVERVIEW	1
Characteristics	1
Users.....	2
2. ARCHITECTURE	4
3-tier Architecture	4
3. DATA MODELS	6
Entity-Relationship Model	6
Relational Model	7
4. DATA SCHEMAS	9
Database Schema.....	9
Database Instance.....	10
5. DATA INDEPENDENCE	11
Data Independence.....	11
Logical Data Independence	11
Physical Data Independence	12
6. ER MODEL – BASIC CONCEPTS	13
Entity	13
Attributes	13

Relationship	14
7. ER DIAGRAM REPRESENTATION	17
Entity	17
Attributes	17
Relationship	19
8. GENERALIZATION & SPECIALIZATION	22
Generalization	22
Specialization	22
Inheritance	23
9. CODD'S 12 RULES.....	25
Rule 1: Information Rule	25
Rule 2: Guaranteed Access Rule	25
Rule 3: Systematic Treatment of NULL Values.....	25
Rule 4: Active Online Catalog	25
Rule 5: Comprehensive Data Sub-Language Rule	25
Rule 6: View Updating Rule.....	26
Rule 7: High-Level Insert, Update, and Delete Rule	26
Rule 8: Physical Data Independence.....	26
Rule 9: Logical Data Independence	26
Rule 10: Integrity Independence	26
Rule 11: Distribution Independence.....	26
Rule 12: Non-Subversion Rule.....	26
10. RELATIONAL DATA MODEL.....	27
Concepts	27
Constraints.....	27

11. RELATIONAL ALGEBRA.....	29
Relational Algebra	29
Relational Calculus.....	31
12. ER MODEL TO RELATIONAL MODEL	33
Mapping Entity	33
Mapping Relationship	34
Mapping Weak Entity Sets	34
Mapping Hierarchical Entities	35
13. SQL OVERVIEW.....	37
Data Definition Language.....	37
Data Manipulation Language	38
14. NORMALIZATION	41
Functional Dependency	41
Armstrong's Axioms.....	41
Trivial Functional Dependency.....	41
Normalization	42
First Normal Form	42
Second Normal Form	43
Third Normal Form.....	44
Boyce-Codd Normal Form	45
15. JOINS.....	46
Theta (θ) Join	46
Equijoin.....	47
Natural Join (\bowtie)	47
Outer Joins.....	49

16. STORAGE SYSTEM	52
Memory Hierarchy	52
Magnetic Disks.....	53
RAID.....	53
17. FILE STRUCTURE.....	56
File Organization.....	56
File Operations.....	57
18. INDEXING	59
Dense Index	59
Sparse Index	60
Multilevel Index.....	60
B ⁺ Tree	61
19. HASHING.....	63
Hash Organization.....	63
Static Hashing	63
Bucket Overflow	64
Dynamic Hashing	65
Organization	66
Operation	66
20. TRANSACTION.....	68
ACID Properties.....	68
Serializability	69
Equivalence Schedules	69
States of Transactions	71
21. CONCURRENCY CONTROL	72

Lock-based Protocols	72
Timestamp-based Protocols.....	74
Timestamp Ordering Protocol	74
22. DEADLOCK	76
Deadlock Prevention.....	76
Deadlock Avoidance.....	77
23. DATA BACKUP	79
Loss of Volatile Storage.....	79
Database Backup & Recovery from Catastrophic Failure.....	79
Remote Backup.....	80
24. DATA RECOVERY	81
Crash Recovery	81
Failure Classification	81
Storage Structure.....	82
Recovery and Atomicity	82
Log-based Recovery	83
Recovery with Concurrent Transactions.....	83

1. OVERVIEW

Database is a collection of related data and data is a collection of facts and figures that can be processed to produce information.

Mostly data represents recordable facts. Data aids in producing information, which is based on facts. For example, if we have data about marks obtained by all students, we can then conclude about toppers and average marks.

A **database management system** stores data in such a way that it becomes easier to retrieve, manipulate, and produce information.

Characteristics

Traditionally, data was organized in file formats. DBMS was a new concept then, and all the research was done to make it overcome the deficiencies in traditional style of data management. A modern DBMS has the following characteristics:

- **Real-world entity:** A modern DBMS is more realistic and uses real-world entities to design its architecture. It uses the behavior and attributes too. For example, a school database may use students as an entity and their age as an attribute.
- **Relation-based tables:** DBMS allows entities and relations among them to form tables. A user can understand the architecture of a database just by looking at the table names.
- **Isolation of data and application:** A database system is entirely different than its data. A database is an active entity, whereas data is said to be passive, on which the database works and organizes. DBMS also stores metadata, which is data about data, to ease its own process.
- **Less redundancy:** DBMS follows the rules of normalization, which splits a relation when any of its attributes is having redundancy in values. Normalization is a mathematically rich and scientific process that reduces data redundancy.
- **Consistency:** Consistency is a state where every relation in a database remains consistent. There exist methods and techniques, which can detect attempt of leaving database in inconsistent state. A DBMS can provide greater

7

consistency as compared to earlier forms of data storing applications like file-processing systems.

- **Query Language:** DBMS is equipped with query language, which makes it more efficient to retrieve and manipulate data. A user can apply as many and as different filtering options as required to retrieve a set of data. Traditionally it was not possible where file-processing system was used.
- **ACID Properties:** DBMS follows the concepts of **A**tomicity, **C**onsistency, **I**solation, and **D**urability (normally shortened as ACID). These concepts are applied on transactions, which manipulate data in a database. ACID properties help the database stay healthy in multi-transactional environments and in case of failure.
- **Multiuser and Concurrent Access:** DBMS supports multi-user environment and allows them to access and manipulate data in parallel. Though there are restrictions on transactions when users attempt to handle the same data item, but users are always unaware of them.
- **Multiple views:** DBMS offers multiple views for different users. A user who is in the Sales department will have a different view of database than a person working in the Production department. This feature enables the users to have a concentrate view of the database according to their requirements.
- **Security:** Features like multiple views offer security to some extent where users are unable to access data of other users and departments. DBMS offers methods to impose constraints while entering data into the database and retrieving the same at a later stage. DBMS offers many different levels of security features, which enables multiple users to have different views with different features. For example, a user in the Sales department cannot see the data that belongs to the Purchase department. Additionally, it can also be managed how much data of the Sales department should be displayed to the user. Since a DBMS is not saved on the disk as traditional file systems, it is very hard for miscreants to break the code.

Users

A typical DBMS has users with different rights and permissions who use it for different purposes. Some users retrieve data and some back it up. The users of a DBMS can be broadly categorized as follows:



[Image: DBMS Users]

- **Administrators:** Administrators maintain the DBMS and are responsible for administrating the database. They are responsible to look after its usage and by whom it should be used. They create access profiles for users and apply limitations to maintain isolation and force security. Administrators also look after DBMS resources like system license, required tools, and other software and hardware related maintenance.
- **Designers:** Designers are the group of people who actually work on the designing part of the database. They keep a close watch on what data should be kept and in what format. They identify and design the whole set of entities, relations, constraints, and views.
- **End Users:** End users are those who actually reap the benefits of having a DBMS. End users can range from simple viewers who pay attention to the logs or market rates to sophisticated users such as business analysts.

2. ARCHITECTURE

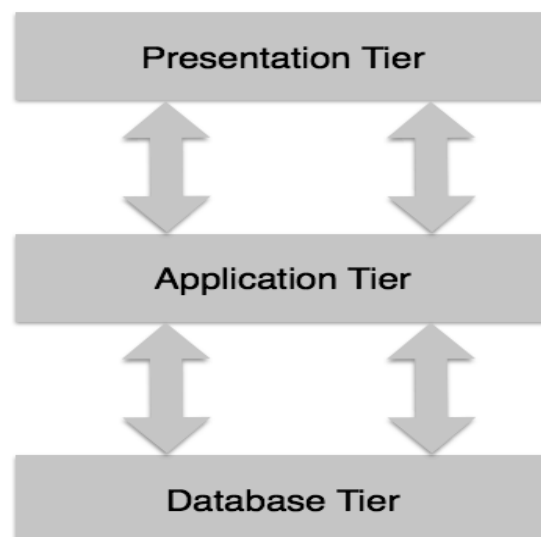
The design of a DBMS depends on its architecture. It can be centralized or decentralized or hierarchical. The architecture of a DBMS can be seen as either single tier or multi-tier. An n-tier architecture divides the whole system into related but independent **n** modules, which can be independently modified, altered, changed, or replaced.

In 1-tier architecture, the DBMS is the only entity where the user directly sits on the DBMS and uses it. Any changes done here will directly be done on the DBMS itself. It does not provide handy tools for end-users. Database designers and programmers normally prefer to use single-tier architecture.

If the architecture of DBMS is 2-tier, then it must have an application through which the DBMS can be accessed. Programmers use 2-tier architecture where they access the DBMS by means of an application. Here the application tier is entirely independent of the database in terms of operation, design, and programming.

3-tier Architecture

A 3-tier architecture separates its tiers from each other based on the complexity of the users and how they use the data present in the database. It is the most widely used architecture to design a DBMS.



[Image: 3-tier DBMS architecture]

- **Database (Data) Tier:** At this tier, the database resides along with its query processing languages. We also have the relations that define the data and their constraints at this level.

End of ebook preview
If you liked what you saw...
Buy it from our store @ <https://store.tutorialspoint.com>