**UNIT-IV**

**DATABASE SYSTEM**

**Database Management System**

* A Database Management System (DBMS) is a collection of interrelated data and a set of programs to access these 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 system Applications;**

Database are widely used here are some representative applications.

**Banking:**  For customer information, accounts, loans and banking transactions.

**Airlines:** For reservations and schedule information, airlines were among the first to use database in a geographically distributed manner.

**Universities:** For student information, course registrations and grades.

**Credit card transactions:**  For purchases on credit cards and generation of monthly statements.

**Telecommunication:** For keeping records of calls made, generating monthly bills, maintaining balances on prepaid calling cards, and storing information about the communication networks.

**Finance:** For storing information about holding sales and purchases of financial instruments such as stocks and bonds also for storing real-time market data to enable on-line trading by customers and automated trading by the firm.

**Sales:** For customer, product and purchase information.

 **On-Line relations:** For sales data noted above plus on-line order tracking , generation of recommendation lists, and maintenance of on-line product evaluations.

**Manufacturing:** For management of the supply chain and for tracking production of items in factories, inventories of items in warehouse and stores, and orders for items.

**Human resources:** For information about employees, salaries, payroll taxes, benefits, and for generation of paychecks.

**Purpose of Database System**

Database systems arose in response to early methods of computerized management of commercial data.

**Example:**

* Consider part of a bank enterprise that, among other data, keeps information about all customers and savings accounts.
* 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 manipulates the files, including programs to
* Debit or credit an account
* Add a new account
* Find the balance of an account
* Generate monthly statements.
* The **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. Before database management systems (DBMS) came along, organizations usually stored information in such systems.

**Disadvantages:**

**Data redundancy and inconsistency:**

* 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.
* This redundancy leads to higher storage and access cost.
* It may lead to data inconsistency; that is, the various copies of the same data may no longer agree.

**Difficulty in accessing data:**

* Conventional file processing environment do not allow needed data to be retrieved in a convenient and efficient manner.
* Suppose that one of the bank officers needs to find out the names of all customers who live within a particular postal-code area. The officers ask the data processing department to generate such a list.
* Because the designers of the original system did not anticipate this request, there is no application program on hand to meet it.

**Data isolation:**

* Data are scattered in various files, and files maybe 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.

**Atomicity problems:**

* A computer System, like any other mechanical or electrical device, is subject to failure. If a failure occurs the data is restored to the consistent state that existed prior to the failure.
* Consider a program to transfer **$50** from account **A**, to account **B**. If a system failure occurs during the execution of the program. It is Possible that the **$50** was removed from account **A**, but was not credited to account **B**.

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

**View of Data:**

* A database system is a collection of interrelated data and a set of programs that allow users to access and modify these data.
* A major purpose of a database system is to provide users with an abstract view of the data, that is the system hides certain details of how the data are stored and maintained.

**Data abstraction:**

* It must retrieve data efficiently. The need for efficiency has led designers to use complex data structures to represent data in the database.
* Since many database system users are not computer trained, developers hide the complexity from users through several levels of abstraction, to simplify user’s interactions with the system.

**View level**

 ………..

**View 2**

**View n**

**View 1**

**Logical level**

**Physical level**

**The Three Level of Data abstraction**

**Physical Level:** The lowest level of abstraction describes **how** the data are actually stores.

**Logical Level:**

* The next-higher level of abstraction describes **what** data are stored in the database, and **what** relationships exist among those data.
* The logical level thus describes the entire database in terms of a small number of relatively simple structures.

**View Level:**

* The highest level of abstraction describes only part of the entire database.
* The view level of abstraction exists to simplify their interaction with the system. The system may provide many views for the same database.

**Example: Banking** 🡪 the relationship among the three levels of abstraction.

 Type Customer = record

 Customer-id : String;

 Customer- name : String;

 Customer- street : String;

 Customer - city : String;

 End;

* This code defines a new record type called customer with four fields. Each field has a name and a type associated with it.
* A banking enterprises may have several such record types, including
* **Account, with fields account-number and balance**
* **Employee, with fields employee-name and salary**
* The **physical level**, a customer, account, or employee record can be described as a block of consecutive storage locations **(for example, words and bytes).** The compiler hides this level of detail from programmers.
* The **Logical level**, each such record is described by a type definition, as in the previous code segment, and the interrelationship of these record types is defined as well programmers using a programming language work at this level of abstraction.
* Similarly database administrators usually work at this level of abstraction.
* The **logical level**, computer users see a set of application programs that hide details of the data types.
* The **view levels**, several views of the database are defined, and database users see these views.
* To hiding details of the logical level of the database, the views also provide a security mechanism to prevent users from accessing certain part of the database.

**Example:** A bank sees only that part of the database that has information on customer about salaries of employees.

**Instances and schemas:**

* The collection of information stored in database at a particular moment is called an **instance** of the database.
* The overall design of the database is called the database **schema**.
* A database schema corresponds to the variable declarations in a program.
* Each variable has a particular value at a given instant.
* Database systems have several schemas, portioned according to the levels abstraction.
* **Physical schema:** It describes the database design at the physical level.
* **Logical schema:** It describes the database design at the logical level.
* **Subschema:** A database may also have several schemas at the view level, called subschema, that describe different views of the database.

**Data models:**

* Data model is a collection of conceptual tools for describing data, data relationships, and consistency constraints.
* A data model provides a way to describe the design of a database at the **physical, logical,** and **view level**. The data models can be classified in four different categories.

**Relational Model:**

* The relational model uses a collection of tables to represent both data and the relationships among those data.
* Each table has multiple columns, and each column has a unique name.

|  |  |  |
| --- | --- | --- |
| C-Id | C-Name | Address |
|  |  |  |

|  |  |  |
| --- | --- | --- |
| C-Id | C-Name | Address |
|  |  |  |

* Each record type defines a fixed number of fields, or attributes.
* Single query uses to retrieve the data to multiple tables use the relation.

**The Entity – Relationship model:**

* The E–R model is based on a perception of a real world that consists of a collection of basic objects is called entities and of relationship among these objects.
* An entity is a **“thing” or “object”** in the real world that is noticeable from other objects.
* The entity – relationship model is widely used in database design.

**Object – Based Data Model:**

* The Object – Oriented model can be seen as extending the E–R model with notations of encapsulation, methods and object identity.
* The Object–relational data model combines features of the Object – Oriented data model and relational data model.

**Semi structured Data model:**

* The semi structured data model permits the specification of data where individual data items of the same type may have different sets of attributes.
* The Extensible Markup Language (XML) is widely used to represent semi structured data.
* The network data model and the hierarchical data model preceded the relational data model.
* These models were tied closely to the underlying implementation and complicated the task of modeling data.

**The Entity – Relationship Model:**

* The Entity – Relationship data model is based on a perception of a real world that consists of a collection of basic objects called **Entities** and of **Relationship** among these objects.
* Entities are described in a database by a set of **attributes.**
* A **relationship** is an association among several entities.
* The set of all entities of the same type and the set of all relationships of same type are termed an **entity set** and **relationship set,** respectively.
* The overall logical structure of a database can be expressed graphically by an **E–R** diagram, which is built up from the following components.

**customer**

**account**

**depositor**

**Rectangles**  🡪which represent entity set.

**Ellipses**  🡪 which represent attributes.

**Diamonds** 🡪 which represent sets of relationship among a member from each of several entity sets.

**Lines** 🡪 which link attributes to entity sets and entity sets to relationships.

* Each component is labeled with the entity or relationship that it represents.
* One important constraint is **mapping cardinalities,** which express the number of entities to which another entity can be associated via a relationship set.