**EXPERIMENT NO. 01**

|  |  |
| --- | --- |
| **DATE OF PERFORMANCE:** | **GRADE:** |
| **DATE OF ASSESSMENT:** | **SIGNATURE OF LECTURER/ TTA:** |

**AIM: Introduction to RDBMS & ORACLE.**

**THEORY:**

**RELATIONAL DATABASE MANAGEMENT SYSTEM (RDBMS):**

**E.F. Codd of IBM research first introduces the relational data model in a paper in 1970. The relational data model is implemented using very sophisticated relational database management system (RDBMS). The relational data model simplified the user’s view of the database by using simple tables instead of the more complex tree and network structures. It is a collection of tables (also called relations).**

**ADVANTAGES OF RDBMS:**

***Simplicity*: A relational data model is even simpler than hierarchical and network models. It frees the designers from the actual physical data storage details, thereby allowing them to concentrate on the logical view of the database.**

**Structural Independence: Unlike hierarchical and network models, the relational data model does not depend on the navigational data access system. Changes in the database structure do not affect the data access.**

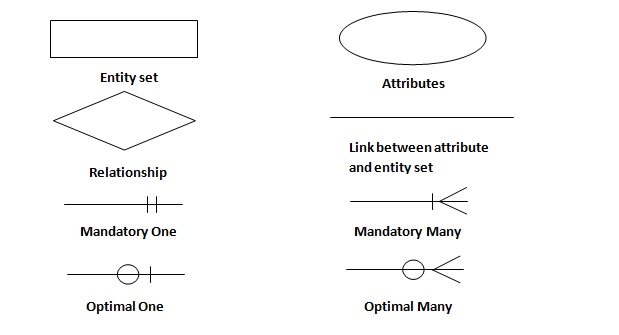
**Ease of design implementation, maintenance and uses: The relational model provides both structural independence and data independence. Therefore, it makes the database design, implementation, maintenance and usage much easier.**

**Flexible and powerful query capability: The relational database model provides very powerful, flexible, and easy to use query facilities. Its structured query language (SQL) capability makes ad hoc queries a reality.**

**ENTITY-RELATIONSHIP (E-R) DATA MODEL:**

**An entity-relationship (E-R) model is a logical database model, which has a logical representation of data for an enterprise of business establishment. It was introduced by Chen in 1976. E-R data model is a collection of objects of similar structures called an entity set.**

**The relationship between entity sets is represented on the basis of number of entities from entity set that can be associated with the number of entities of another set such as one-to-one(1:1), one-to-many(1:n) or many-to-many(n:n) relationships.**

****

***Figure 1.1***

**ADVANTAGES OF E-R DATA MODEL:**

* **Straightforward relational representation: Having designed an E-R diagram for a database application, the relational representation of the database model becomes relatively straightforward.**
* **Easy conversion for E-R to other data model: Conversion from E-R diagram to a network or hierarchical data model can easily be accomplished.**
* **Graphical representation for better understanding: An E-R model gives graphical and diagrammatical representation of various entities, its attributes and relationship between entities. This in turn helps in the clear understanding of the data structure and in minimizing redundancy and other problems.**

**BASIC E-R CONCEPTS:**

* **E-R modelling is a high level conceptual data model developed to facilitate database design.**
* **A conceptual data model is a set of concepts that describe the structure of a database and the associated retrieval and update transaction of database.**
* **E-R model is also defined as a logical representation of data for an enterprise.**
* **It is a top-down approach to database design.**

**An E-R model provides the following three main semantic concepts to the designers:**

* **Entities: which are distinct in a user enterprise.**
* **Relationships: which are meaningful interactions among the objects.**
* **Attributes: which describe the entities and relationships. Each such attribute is associated with a value set (also called domain) and can take a value from this value set.**
* **Constraints: on the entities, relationships and an attributes.**

**Entities:**

**An entity is an “object” or a “thing” in the real world with an independent existence and that is distinguishable from other objects. Each entity has attributes.**

**Person: student, patient, doctor, engineer, employee**

**Place: city, Country, State**

**Such things are modelled as entities that may be described by attributes. An entity is represented by a set of attributes. Each entity has a value for each of its attributes.**

**Entity type (set) and Entity instance:**

**An entity set (also called entity type) is a set of entities of the same type that share the same properties or attributes.**

**Relationships:**

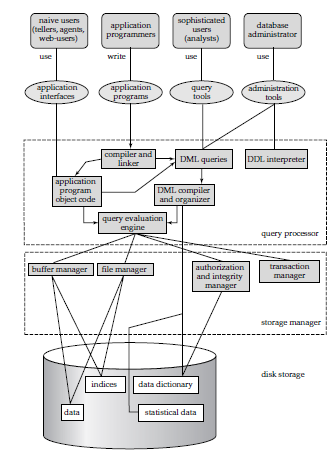
**A relationship is an association among two or more entities that is of interest to the enterprise.**

**Attributes:**

**An attribute is a property of an entity or a relationship type. An entity is described using a set of attributes. Each attribute is associated with a set of values called a domain. For example, If the age of an employee in an enterprise is between 18 and 60 years, we can define a set of values for age attributes of the ‘employee’ entity as the set of integers between 18 and 60. For example, Domain for the date of birth attribute is made up of sub-domains namely, day, month and year. Attributes may share a domain and is called the attribute domain.**

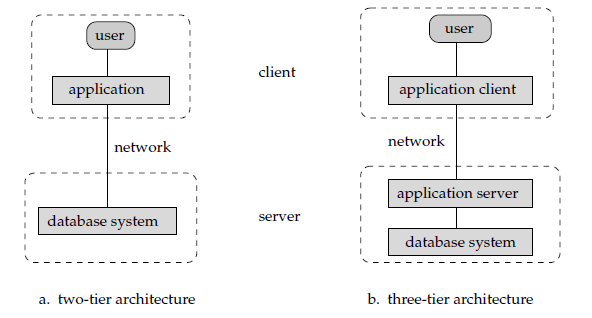
**DATABASE ARCHITECTURE:**

**Database applications are usually partitioned into two or three parts, as in figure. In a two-tier architecture, the application is partitioned into a component that resides at the client machine, which invokes database system functionality at the server machine through query language statements. Application program interface standards like ODBC and JDBC are used for interaction between the client and the server.**

****

***Figure 1.2***

**In contrast, in a three-tier architecture, the client machine acts as merely a front end and does not contain any direct database calls. Instead, the client end communication with an application server, usually through a forms interface. The application server in turn communicates with a database system to access data. The business logic of the application, which says what actions to carry out under what conditions, is embedded in the application server, instead of being distributed across multiple clients. Three-tier applications are more appropriate for large applications, and for applications that run on the World Wide Web.**

****

***Figure 1.3***

**What is Oracle?**

**Oracle is a relational database management system.**

* + **It is a management system which uses the relational data model.**
  + **In the relational data model, data is seen by the users in form of tables alone.**

**Oracle Server:**

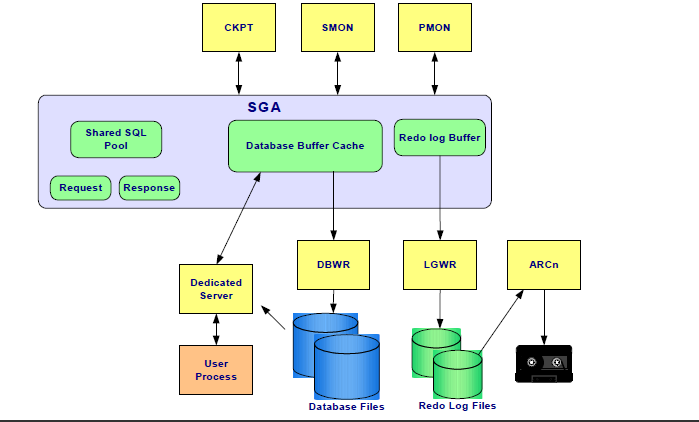
* + **Is a database management system that provides an open, comprehensive, integrated approach to information management.**
  + **Consists of an Oracle Instance and an Oracle database**

**Oracle in Industry:**

* **In today’s world, data is the key for business.**
* **Every organization stores its data in multiple databases.**
* **One of the most widely used database in industry is Oracle.**
* **Oracle can work on various Operating Systems (Windows, UNIX, etc.).**
* **The demand for Oracle in today’s world is immense.**
* **Many projects across the industry use Oracle as back-end for deploying its various applications.**

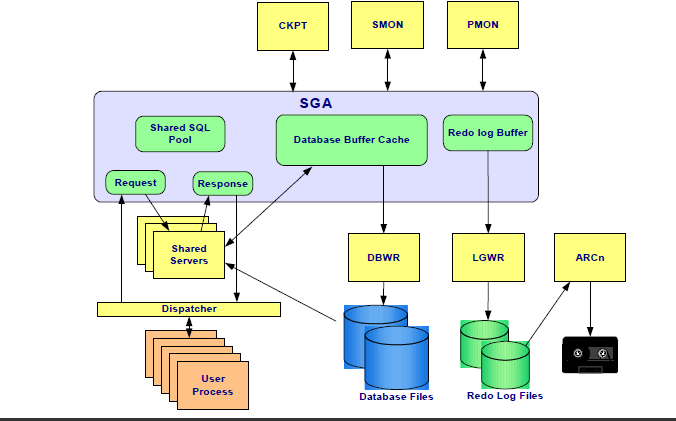
**ORACLE ARCHITECTURE:**

**Interaction with the Database (Dedicated server):**

****

***Figure 1.4***

**Interaction with the Database (Shared server):**

****

***Figure 1.5***

**Internal Memory Structures SGA:**

**System or ‘shared’ Global Area (SGA)**

* **Database buffer cache**
* **Redo log buffer**
* **Shared pool**
* **Request & response queues (shared server)**

**Database buffer cache:**

* **Used to hold data blocks read from data files by server processes**
* **Contains ‘dirty’ or modified blocks and ‘clean’ or unused or unchanged bocks**
* **Size is managed by the parameter DB\_BLOCK\_BUFFERS**

**Least Recently Used (LRU):**

**LRU and the database buffer cache:**

* **Every time a data block is read from disk it is placed in the database buffer cache at the head of the LRU list.**
* **If a block is already in the cache and it is read again**

**Redo Log Buffer:**

* **A circular buffer that contains redo entries**
* **Redo entries take up contiguous, sequential space in the buffer**
* **Data stored in the redo log buffer is periodically written to the online redo log files**

**Shared Pool:**

**Consists of multiple smaller memory areas**

* **Library cache**
* **Shared SQL area**
* **Contains parsed SQL and execution plans for statements already run against the database**
* **Procedure and package storage**
* **Dictionary cache**
* **Names of all tables and views in the database**
* **Names and datatypes of columns in the database tables**
* **Privileges of all users**

**Least Recently Used (LRU):**

**LRU and the shared pool**

* **Every time a SQL statement is parsed it is placed in the shared pool for reuse**
* **If a SQL statement is already in the shared pool it will not re-parse but it is placed at the head of the LRU**

**Internal Memory Structures PGA:**

**Program or ‘process’ Global Area (PGA)**

* **Used for a single process**
* **Not shareable with other processes**
* **Writable only by the server process**
* **Allocated when a process is created and**
* **deallocated when a process is terminated**
* **Contains**
* **Sort area – Used for any sorts required by SQL processing**
* **Session information – Includes user privileges**
* **Cursor state – Indicates stage of SQL processing**
* **Stack space – Contains session variables**

**Background Processes – DBWR:**

* **Writes contents of database buffers to datafiles.**
* **Primary job is to keep the database buffer ‘clean’ .**

**Background Processes – LGWR:**

* **Writes contents of redo log buffers to online redo log files.**
* **Primary job is to keep the redo log buffer ‘clean’.**

**Background Processes – SMON:**

* **Performs automatic instance recovery.**
* **Reclaims space used by temporary segments no longer in use.**

**Background Processes - PMON:**

**Performs automatic process recovery**

* **Cleans up abnormally terminated connections**
* **Rolls back non committed transactions**
* **Releases resources held by abnormally terminated transactions**

**Restarts failed shared server and dispatcher processes**

**Background Processes – CKPT:**

* **Forces all modified data in the SGA to be written to datafile Updates the datafile headers.**

**Background Processes – ARCH:**

* **Automatically copies online redo log files to designated storage once they have become full**

**Server Processes**

* **Services a single user process in the dedicated server configuration or many user processes in the shared server configuration**

**User Processes**

* **Run on the client machine**
* **Are spawned when a tool or an application is invoked**

**DATA INDEPENDENCE:**

**Data independence is a major objective of implementing DBMS in an organization. It may be defined as the immunity of application programs to change in physical representation and access techniques. Alternatively, data independence is a characteristic of a database system to change the schema at one level without having to change the schema at the next higher level.**

**There are two types of data independence:**

1. **Physical Data Independence**
2. **Logical Data Independence**

**Physical Data Independence:**

**Immunity of the conceptual (or external) schemas to changes in the internal schema is referred to as physical data independence. In physical data independence, the conceptual schema insulates the users from changes in the physical storage of the data. Changes to the internal schema such as using different file organizations or storage structures, using different storage devices, modifying indexes and hashing algorithms, must be possible without changing the conceptual or external schemas. In other words, physical data independence indicates that the physical storage structures or devices used for storing the data could be changed without necessitating a change in the conceptual views or any of the external views.**

**Logical Data Independence:**

**Immunity of the external schemas (or application programs) to changes in the conceptual schemas is referred to as logical data independence. In logical data independence, the users are shielded from changes in the logical structure of the data or changes in the choice of relations to be stored. Changes to the conceptual schema, such as the addition and deletion of entities, addition and deletion of attributes, or addition and deletion of relationships, must be possible without changing existing external schemas or having to rewrite application programs. Only the view definition and mapping need be changed in a DBMS that supports logical data independence.**