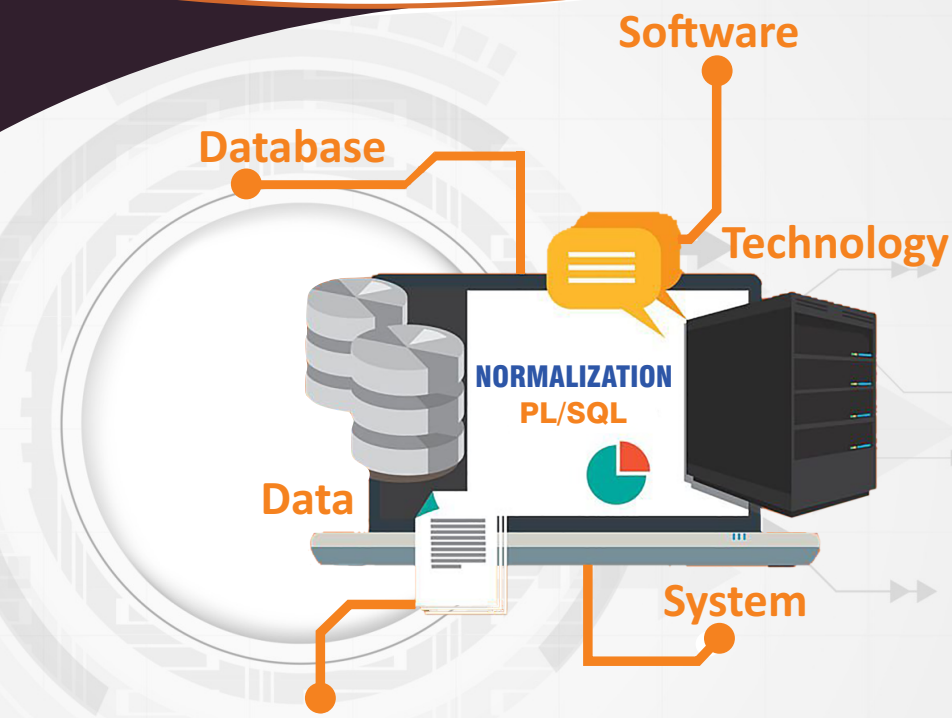
BAOU
Education for All

Dr. Babasaheb Ambedkar Open University

(Established by Government of Gujarat)



Relational Database Management System

Bachelor of Science (Hons.)
Information Technology

BSCIT - 301

ISBN:

Printed and published by: Dr. Babasaheb Ambedkar Open University, Ahmedabad

Dr. Babasaheb Ambedkar Open University

BSCIT-301

Relational Database Management System

BLOCK1: INTRODUCTION OF DATABASE

BLOCK-2: NORMALIZATION AND SERVER ARCHITECTURE

BLOCK-3: ORACLE

BLOCK-4: PL/SQL - ORACLE

BLOCK – 1

Introduction of Database

BLOCK 1: INTRODUCTION OF DATABASE

Block Introduction

In this block, we will learn about basic concepts of databases and database management systems. It provides an overview of the above topic with the help of various examples.

You will also get a basic idea of what is data, database, and its architecture. Various models have been discussed here in this block in a very brief and interesting way. The main topics that have been covered in this block are basic concepts, architecture, models, and database design.

This block aims to enable the reader to understand the basic concepts of this topic. The writer has tried his best to detail the concepts; apart from this, the sub-topics have even been discussed by him in detail. On the other hand, he has even discussed the database design in detail with sufficient examples.

The objective of this block includes detailing the student‘s about the various above mentioned topics in very detail with the help of sufficient and suitable examples.

Block Objective

After learning this block, you will be able to understand:

- Purpose of database
- Why database is important over file systems
- Database models
- The rows and columns within a table and their use
- The architecture of the database
- The database design process
- Anomalies within the database

Unit 1: Basic Concepts

1

Unit Structure

1.0 LEARNING OBJECTIVES

After learning this unit, you will be able to understand:

- Database concepts - Data and Information, Database systems and Database
- Management Systems and How DBMS is advantageous as compared to the traditional file systems.

1.1 INTRODUCTION

A database is a collection of related data that is saved to make it accessible to a large number of people for various purposes. The content of a database is created by merging information from multiple sources so that all users have access to data and redundancy of data is minimized or eliminated. Data management entails both the creation of a structure for data storage and the administration of data.

1.2 BASIC CONCEPTS

In this topic, we are going to learn about the basic concepts of what is data, how data is organized, and what are database systems.

How can these companies handle massive amounts of data, store it all, and then swiftly retrieve the information that decision-makers require? The solution is that databases are used. Virtually all current business systems rely on databases; consequently, every information systems expert must have a thorough grasp of how these structures are generated and how to utilize them properly. Even if your professional path does not lead you down the exciting route of database design and development, databases will be a critical component of the systems you work with. In any case, it's highly probable that you'll be making judgments based on data-driven knowledge. As a result, understanding the distinction between data and information is critical.

1.2.1 Data

Data are raw and unprocessed facts. The term "raw" denotes that the information has not been processed to disclose its meaning.

Information is the result of processing raw data to reveal its meaning. For example, if we write our daily expenses, we are collecting the raw facts and at the end of the day, if we process it and get the total expenses, we are generating the information out of the data collected.

Data constitute the building blocks of information, if the data is accurate the information generated will be correct and so the decisions taken on the basis of that information will be accurate, thus, accurate, relevant, and timely information is the key to good decision making and good decision making is the key to organizational success in a competitive global environment.

Data are the fundamental components of information. Data processing results in the creation of information. The purpose of information is to reveal the meaning of data. The key to making excellent decisions is having accurate, relevant, and timely information.

In a global world, good decision-making is essential for organizational survival and accurate data is required for timely and meaningful information. Such data must be generated correctly and saved in a format that is simple to use and understand. The data environment, like any other basic resource, must be carefully controlled.

1.2.2 Database

Accurate data is needed for timely and meaningful information. Such data must be stored correctly and kept in an accessible and processable way. The data environment, like any other essential resource, must be properly managed. So, there is a need for a proper storage and processing mechanism of data which is called a database.

Efficient data management typically requires the use of a computer database. A database is a shared and integrated computer structure that stores end-user data, that is, raw facts collected by the end-user and metadata, or data about data, through which

the end-user data are integrated and managed. The metadata describes the data qualities as well as the set of relationships that connect the data in the database.

Types of databases

Classification on the basis of number of users

The number of users in a database affects whether it is single-user or multiuser. Only one user can access a single-user database at a time. To put it another way, if user A is using the database, user B will have to wait till user A is finished. A desktop database is a single-user database that operates on a personal computer. A multiuser database, on the other hand, may accommodate several users at the same time. A workgroup database is a multiuser database that serves a small number of users (typically fewer than 50) or a specific department within an organisation. An enterprise database is one that is utilised by the entire organisation and supports a large number of users (greater than 50) across multiple departments.

Classification on the basis of location

The database could potentially be classified based on its location. A centralized database, for example, is one that supports data from a single location. A distributed database is a database that supports data that is distributed over multiple places.

Classification on the basis of usage

However, the most common method of categorizing databases nowadays is based on how they will be used and the time sensitivity of the data gathered from them. Transactions like product or service sales, payments, and supply purchases, for example, indicate important day-to-day operations. Such transactions must be correctly and promptly recorded. An operational database is a database that is primarily used to support a company's day-to-day operations (sometimes referred to as a transactional or production database). A data warehouse, on the other hand, is primarily concerned with storing data that is utilised to create information needed to make tactical or strategic decisions.

The databases now a day are created to cater to different types of data like

1) Structure data

2) Semi structured data

3) Unstructured data

The structured data is like the related tables stored in the database, semi structured includes the data which is stored as an xml or json formats in sql or no sql platforms. The unstructured data is like images and videos which are generating enormous data in recent days.

Check your Progress -1

1) Data refers to _____

a) raw facts b) processed facts c) only numbers d) files

2) Information refers to

a) processed data b) only numbers c) raw facts d) files

3) Database consists of

a) user data and metadata b) user data only

1.2.3. Database Systems

A database system is a logical data repository that stores logically connected data. A database system is a set of components that define and control data collection, storage, administration, and usage in a database environment. The word database system refers to a set of components that define and control how data is collected, stored, managed, and used in a database environment.

Due to the issues with file systems, a database system is a far better option. The database system, unlike the file system, which has many independent and unconnected files, is made up of logically related data kept in a single logical data repository. (The "logical" name refers to the fact that the data repository's contents may be physically distributed among numerous data storage facilities and/or locations, despite the fact that it appears to the end user to be a single unit.) The database marks a significant change

in the way end-user data is stored, accessed, and managed because the database's data repository is a single logical entity.

Check your Progress -2

1) Database systems consists of _____
   a) logically connected data b) data which is not connected

2) Procedures in database systems are
   a) rules and instructions b) processing of files

3) DBA refers to
   a) database administrator b) database activation

1.3 DATABASE MANAGEMENT SYSTEM

1.3.1 Introduction

Data management is a discipline concerned with the correct creation, storage, and retrieval of information. A database management system (DBMS) is a set of programs that manages the database's structure and restricts access to the data it stores. A database is similar to a well-organized electronic file cabinet in which strong software, known as a database management system, assists in the administration of the cabinet's contents.

Functions of DBMS

A database management system (DBMS) performs numerous critical operations that ensure the database's data's integrity and consistency. The majorities of these functions are invisible to end users and can only be accomplished
with the help of a database management system.

- **Data dictionary Management**

In a data dictionary, the DBMS holds definitions of data elements and their relationships (metadata). As a result, the DBMS is used by all programmes that access the database's data. The DBMS looks for the relevant data component structures and relationships in the data dictionary, saving you the trouble of having to code such complex relationships in each programme. Furthermore, any changes to a database structure are automatically stored in the data dictionary, saving you the trouble of having to update all of the programmes that use the modified structure.

- **Data Storage Management**

Management of data storage: The database management system (DBMS) develops and manages the complex structures required for data storage, saving you the time and effort of specifying and programming physical data properties.

A modern database management system (DBMS) stores not just data, but also related data entry forms or screen definitions, report definitions, data validation criteria, procedural code, and structures to handle video and picture formats, among other things. Database performance tuning also necessitates data storage management.

- **Data Transformation**

The database management system (DBMS) changes data so that it conforms to the appropriate data structures. You don't have to worry about distinguishing between logical and physical data formats because the DBMS does it for you. That is, the DBMS formats the physically retrieved data to meet the logical requirements of the user.

- **Security**

The database management system (DBMS) creates a security mechanism that ensures user safety and data privacy.

Which users can access the database, which data items each user can access, and which data activities (read, add, delete, or edit) each user can execute, are all determined by security rules. In multiuser database systems, this is very significant. Database Administration and Security delves deeper into data security and privacy concerns. All database users can log in to the DBMS using a username and password

or a biometric authentication method like a fingerprint scan. This information is used by the DBMS to provide access privileges to database components like queries and reports.

- **Multi user access control**

To ensure data integrity and consistency, the DBMS employs sophisticated algorithms to ensure that several users can access the database at the same time without jeopardizing the database's integrity.

- **Backup and recovery Management**

To maintain data safety and integrity, the DBMS supports backup and data recovery. Current DBMS systems have tools that enable the DBA to perform normal and custom backup and restoration procedures. Recovery management is concerned with the restoration of a database following a failure, such as a faulty disc sector or a power outage. This feature is necessary for maintaining the database's integrity.

- **Managing data integrity**

Integrity rules are promoted and enforced by the DBMS, reducing data redundancy and increasing data consistency. Data integrity is enforced using the data relationships stored in the data dictionary.

Manual File Systems:

Traditionally, manual paper-and-pencil systems were frequently used in such systems. The papers in these systems were arranged in chronological order to make the data available for intended use. Typically, this was performed via a filing system consisting of file folders and filing cabinets. As long as a data collection was small and a company's business users had limited reporting options. The manual system accomplished its purpose as a data warehouse well. However, as businesses grew more complex, keeping track of data in a manual file system became more challenging which gave rise to Database Management Systems.

Problems in traditional file systems

- Complexity of Managing files

- Lack of security
- Lack of data sharing
- Data Redundancy and Inconsistency
- Data isolation

To overcome the problems of traditional file systems the DBMS came into existence which was effective to store, manipulate and retrieve the data in the required format in less amount of time.

1.3.2 Purpose and Advantages of DBMS

Database Management System, or DBMS, refers to the understanding of storing and retrieving users' data with maximum efficiency while maintaining appropriate security. End users can create, read, update, and remove data in a database with the help of a database management system (DBMS). The database management system (DBMS) essentially creates a connection between the database and end users or application programmes, ensuring that data is organised and accessible at all times.

There are several advantages of DBMS over Traditional File System

1. Data Redundancy and Inconsistency - During file processing, different files with different formats are generated. The same information being stored in different files leads to inconsistency. The information in two files may be different if not properly updated which causes inconsistency.

2.Security: The DBMS comes with inbuilt security features and has authentication methods which are not possible in traditional file systems.

3. Data Sharing : The database management system (DBMS) aids in the creation of an environment in which end users have better access to more and better-managed data. End users can respond fast to changes in their environment with this kind of access.

4.Data Integration: Access to well-managed data allows for a more clear view of the organization's operations and a clearer understanding of the big picture. It's a lot easier to understand how activities in one part of the organisation affect the rest of the company.

5. Decision Making: Better-managed data and increased data access allow for the generation of higher-quality data, which may then be used to make better decisions.
The quality of the underlying data determines the quality of the information generated.
Thus increasing the end user productivity
6. Transaction Management and Concurrency Control: Transactions are set of operations - sql statements which are either entirely executed or entirely aborted. The database can be shared between multiple users and in multi-user environment the serializability or sequence of Transactions are crucial factors.
7. Backup and Recovery: DBMS has facilities of having complete backup of the data and it can be easily recovered in case of system failure.

Disadvantages of DBMS

Increase in Cost

Database systems necessitate advanced technology and software as well as highly trained staff. The cost of maintaining the hardware, software, and staff needed to run and manage a database system can be significant.

More complexity

Database systems interact with a wide range of technologies and have a major influence on a company's resources and environment. To guarantee that the changes brought on by the implementation of a database system assist the organisation achieve its goals, they must be appropriately managed

Frequent updations

You must keep your database system updated in order to enhance its efficiency.
As a result, you must keep all components up to date and apply the latest patches and security measures. Personnel training expenses are often high due to the fast advancement of database technology.

Check your Progress -3

1) Data redundancy refers to _____

a) data duplication b) data integration

2) Database management systems _____ redundancy

a) decreases b) increases

3) Data redundancy leads to _____

a) data inconsistency b) data consistency

1.4 LET US SUM UP

This unit gives unique learning concepts of Data, Database and Database management systems and it helps to understand how DBMS is useful for increasing user productivity.

1.5 ANSWERS FOR CHECK YOUR PROGRESS

Check your progress 1

Answers: (1-a), (2-a), (3-a)

Check your progress 2

Answers: (1-a), (2-a), (3-a)

Check your progress 3

Answers: (1-a), (2-a), (3-a)

1.6 ASSIGNMENT

1. Explain DBMS and its features.
2. Describe Database system environment.
3. Explain functions of DBMS
4. Explain advantages and disadvantages of DBMS

1.7 ACTIVITIES

1. Write a note on various applications of DBMS.
2. How implementation of DBMS improves productivity

1.8 CASE STUDY

1. Explain the benefits of studying DBMS.
2. Compare various Database Management systems.

1.9 FURTHER READING

1. Database Management Systems - Rajesh Narang -PHI Learning Pvt Ltd.
2. Database System Concepts by Silberschatz, Korth -Tata McGraw-Hill Publication.
3. An Introduction to Database Systems - Bipin Desai- Galgotia Publication.
4. Database Management System by Raghu Ramkrishnan- Tata McGraw-Hill Publication.

Unit 2: Architecture of DBMS

2

Unit Structure

2.0 LEARNING OBJECTIVES

After learning this unit, you will be able to understand:

- Database concepts
- Architecture of database

2.1 ARCHITECTURE OF DBMS

2.1.1 Architecture

A DBMS's design primarily relates to its architecture. This might be hierarchical, centralized, or decentralized. The DBMS architecture is discovered to be single-tier or multi-tier.

With N-tier architecture, the DBMS is separated into a comprehensive system that is linked to independent N modules that may be adjusted, modified or replaced individually.

Single Layer Architecture

In a single entity in a single layer DBMS system, if the user makes a change, it will immediately affect the database management system. As a result, it will not have any useful tools for end consumers to employ. The database's creators or designers can easily manage a simple single-tier architecture.

Two Layer Architecture

In 2-tier DBMS architecture, the application layer, and the database layer, the application

will pass through the DBMS to gain direct access. When using a two-tier architecture, programmers will simply access the database management system through its application layer.

Three Layer Architecture

In the case of 3-tier DBMS architecture, the differences between various tier layouts are distinct.

The Presentation layer, the middle layer - the application layer, and the database layer are the three tiers of three-tier architecture

The three-tier architecture is the most commonly utilized architecture in the development of DBMS.

● Database Tier - In this layer, the database, as well as its query processing languages, is available.

● Application -Middle Tier - In this layer, the database is accessible through application servers, which are having programs or functions to interact with the database tier. End-users do not know the details of the database tier.

● Presentation Tier - In this layer, the user utilizes the database without being aware of the presence of the database beyond this layer. With the help of applications, a variety of data views are displayed. The presentation tier communicates with the database only through the middle tier.

Check your progress -1

1) _____is simplest of all architectures

a) single tier b) middle tier c) presentation tier d) none of these

2) _____ is accessible through application servers

a) middle tier b) single tier c) presentation tier d) none of these

3) ____ layer communicates to database through middle tier

a) presentation tier b) single tier c) both a and b d) none

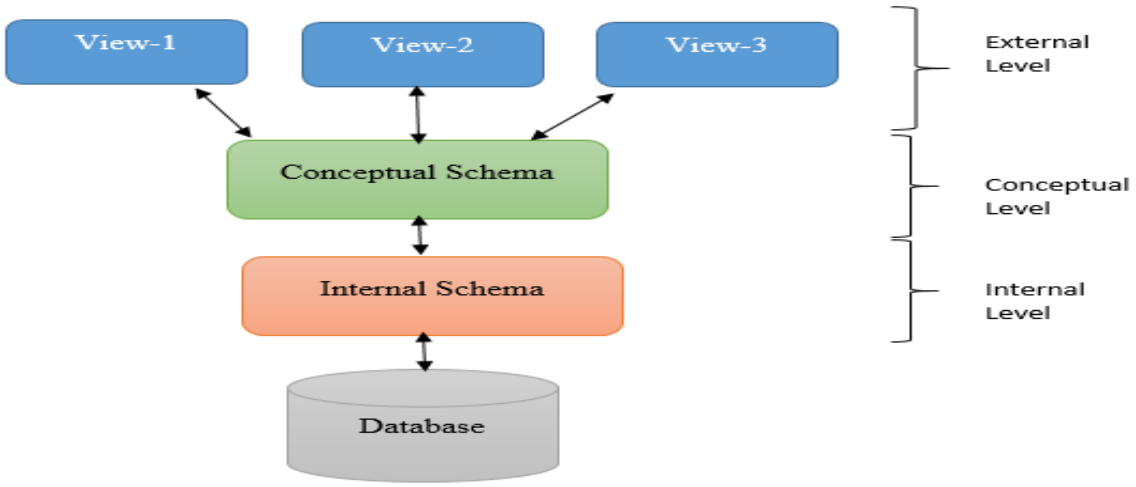


Figure describing the Architecture of DBMS

The DBMS architecture outlines how different users perceive the data in the database. By hiding the intricacies of its internal management processes, this design provides data to users at various levels of abstraction. The total database description can be defined at three levels in this architecture:
Levels: internal, conceptual, and external as a result, it's sometimes referred to as a Three-level architecture.

External Level: This is the highest level of abstraction and is also known as view level because it deals with the user's perspective of the database. The external level describes a database part to a specific group of users or to a single user. Every view provided to the user is tailored to their specific needs. It's possible that various consumers will see the same data through distinct interfaces. By hiding specific data from certain users, it also provides a robust and adaptable security tool. The data described at this level is hardware and programme agnostic. As the data is represented, an entity relationship diagram is typically utilized to depict the external view.

Conceptual Level: This level of abstraction, also known as logical view, deals with the logical structure of the entire database. The view explains the structure and type of data contained in the database, as well as the relationships that exist between the data. It's a broad overview of the database, taking into account the DBMS software that will be utilised.

Internal Level: Also known as physical level, this level describes data at the lowest level of abstraction and deals with the physical representation of the database on the computer. It specifies how data is arranged and stored on a physical storage device. Various aspects are addressed at this level in order to ensure optimal runtime performance and storage space usage.

Check your progress-2

1) ___ is available in Database tier.

a) Query processing b) database designing c) formatting d) all the above

2) ____is highest level of abstraction

a) external level b) internal level c) database level d) none of these

3) ____ is also known as logical view

a) conceptual level b) external level c) database level d) none of these

4) _____ is also known as physical level

a) physical level b) conceptual level c) database level d) all the above

2.1.2 Various components of DBMS

The database management system is composed of five major components namely hardware, software, people, procedures, and data. The implementation of a DBMS gives a foundation for enforcing rules and standards. As long as DBMS is built to make use of the available power, it is feasible to tackle considerably more sophisticated applications of data resources.

Hardware: Hardware includes all physical equipment in a system, such as computers, storage devices, printers, network devices like switches, routers, fiber optics and other components.

Software: DBMS is the most well-known software, the database system requires three types of software to function properly: operating system, DBMS software, and application programs.

- Operating system software is responsible for managing all hardware components and allowing all other applications to execute on computers like Microsoft Windows, Linux, Mac OS.
- Database management software is used to manage databases within a database system like Microsoft SQL Server, Oracle, MySQL.
- Application programs are used to access and alter data in the database systems, as well as to administer the data access and data manipulations.

People: On the basis of the job functions, various roles are assigned to the people to carry out smooth functioning of the database. System administrator looks after the

database general operations Database administrator (DBA) looks after the performance and tuning of the database operations. Database designers are the architects of the database; they design and develop the database objects and the structures where the data is actually stored. Programmers and the end-users are the persons who either use the database for application development and the end-users are going to actually enter the data during the daily operations.

Procedures: Procedures are the instructions and rules that control the database system's design and operation. They are an important part of the system that is sometimes overlooked. Procedures are vital in a business because they enforce the business rules.

Data: Data refers to the database's collection of facts. Because data is the raw material from which information is formed, determining what data should be added into the database and how that data should be arranged is done by the people according to the procedures defined.

Check your progress – 3

1) ____ includes physical components of database

a) hardware b) data c) procedures d) none

2) ___ looks after the performance and tuning of the database

a) database administrator b) end user c) procedures d) none

3) ____ refers to the collection of facts

a) data b) procedures c) users d) all the above

4) ____ are responsible for application development

a) programmers b) designers c) procedures d) none

5) ___ are instructions and rules that control the database

a) procedures b) data c) database design d) all of these

2.2 ANSWER FOR CHECK YOUR PROGRESS

Check your progress 1

Answers: (1-a), (2-a), (3-a) (4-a)

Check your progress 2

Answers: (1-a), (2-a), (3-a)

Check your progress 3

Answers: (1-a), (2-a), (3-a) (4-a), (5-a)

2.3 ASSIGNMENT

Explain various components of the database and its features.

2.4 ACTIVITIES

Write a note on architecture of DBMS.

2.5 CASE STUDY

Compare and contrast different components of DBMS.

2.6 FURTHER READING

1. Database Management Systems - Rajesh Narang -PHI Learning Pvt Ltd.

2. Database System Concepts by Silberschatz, Korth -Tata McGraw-Hill Publication.

3. An Introduction to Database Systems - Bipin Desai- Galgotia Publication.

4. Database Management System by Raghu Ramkrishnan- Tata McGraw-Hill Publication.

Unit 3: Data Models 3

Unit Structure

3.0 LEARNING OBJECTIVES

After learning this unit, you will be able to understand:

- Database Models
- Data modelling and mapping

3.1 DATA MODELS

3.1.1 Introduction

Data modelling is the initial step in the database design process, and it acts as a link between real-world objects and the computer's database. The first stage in developing a database is data modelling, which is the process of creating a specific data model for a given area. A data model is a graphical representation of more sophisticated real-world data structures that are relatively basic. A model is a simplified representation of a more complex real-world object or event. The fundamental purpose of a model is to assist you in understanding the complexities of the real-world environment.

The ability to model data is also an important aspect of the design process. It is critical to adequately document the design that is generated. To ease communication between the database designer, the end user, and the developer, design documentation is required. The most popular approach of documenting database designs is data modelling, which will be discussed later in this work. The implementation of a consistent data-modeling approach guarantees that the data model performs its function of easing communication between the designer, user, and developer. When it comes to maintaining or upgrading a database as business requirements change, the data model is a significant resource. End-user data designs are rarely documented, and they are never accompanied by a consistent data-modeling approach.

Properly build a database that assures data consistency, enforces integrity, and provides a reliable and versatile foundation for delivering fast, accurate information to consumers.

When database designers are deciding which entities, characteristics, and relationships to employ to develop a data model, they may begin by learning about the many types of data that exist in an organisation, how the data is utilised, and in what time periods the

data is used. However, such data and information do not provide the essential understanding of the entire firm on their own. The collection of data becomes useful from the standpoint of a database only when it follows correctly specified business rules.

Business rules, in a way, are mislabeled: they apply to any organisation, large or little, that keeps and utilises data to produce knowledge, whether it's a business, a government unit,

Business rules, which are formed from a thorough description of an organization's processes, aid in the creation and enforcement of actions inside that context. Business rules must be written down and modified when the operating environment of the company changes.

Entities, properties, relationships, and constraints are all defined by properly stated business rules.

3.1.2 Data Modelling and Mapping

The use of the database structure to store and manage end-user data is the subject of database design. The first stage in developing a database is data modelling, which is the act of creating a specific data model for a given issue area. (A problem domain is a well-defined area inside a real-world environment with well-defined scope and bounds that must be addressed in a methodical manner.) A data model is a graphical representation of more sophisticated real-world data structures that is relatively basic. A model is a simplified representation of a more complex real-world object or event. The fundamental purpose of a model is to assist you in comprehending the intricacies of the real-world environment.

A data model is a representation of data structures and their properties, relations, constraints, transformations, and other components in a database environment for the goal of supporting a certain problem area.

Data modelling is a continuous, iterative process. You begin by having a basic understanding of the problem domain, and as your understanding grows, so does the level of information in the data model. When done correctly, the final data model functions as a "blueprint" that contains all of the instructions for creating a database that

meets all end-user needs. This blueprint is narrative and pictorial in character, which means it includes written descriptions in plain, unambiguous language as well as clear, useful graphics displaying the primary data elements.

To construct a good data model, database designers have traditionally depended on their intuition. Unfortunately, sound judgment is often in the eye of the beholder, and it is often acquired through a process of trial and error. If each student in this class is asked to develop a data model for a online shop, for example, it's extremely probable that each of them will come up with a completely different model. Interaction between the designer, the applications programmer, and the end-user can be made easier by data models.

Data abstraction or highlighting only the most important characteristics while hiding the storage and data organization details from the user, is one of the fundamental goals of database systems. A model is an abstraction that focuses on the application's fundamental features while ignoring the non-essential aspects.A database model gives you the tools you need to abstract data.

Various data models allow us to visualize the relationship between entities and their properties. The data models are described as under

Importance of Data Models

Data are the most fundamental informational components used bya framework. Applications are made to help with data management and information transformation. However, data are shown in various ways and by many persons.

Another perspective on data comes from applications programmers, who are more focused on data location, formatting, and particular reporting requirements. In essence, applications programmers convert relevant interfaces, reports, and query screens from a range of sources into organisational policies and processes.

It is irrelevant whether a manager's and/or end user's perceptions of the data differ from an applications programmer's when a sound database blueprint is provided. On the other hand, issues are likely to arise in the absence of a solid database blueprint. You cannot extract the necessary data from the data model since it is an abstraction. You

are as unlikely to build a good database without first developing a suitable data model, just as you are unlikely to build a decent house without a plan.

Hierarchical Data Model

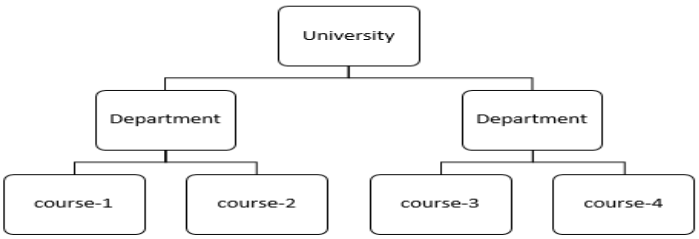
The hierarchical model was created to manage large volumes of data for complicated manufacturing projects. A top-down tree represents its core logical structure. There are levels, or segments, in the hierarchical structure. A segment is the equivalent of a record type in a file system. A higher layer is regarded as the parent of the part right underneath it, known as the child, within the hierarchy. A collection of one-to-many (1: M) relationships between parent and child segments is depicted in the hierarchical model. (Each parent can have multiple children, but each child only has one parent.)

Advantages:

1. It encourages the sharing of information.
2. The parent-child relation encourages conceptual clarity.
3. The database management system (DBMS) provides and enforces database security.
4. Data integrity is enhanced by the parent-child relationship.
5. It works well with one-to-many relationships.

Disadvantages:

1. Complicated implementation necessitates a thorough understanding of physical data storage features.
2. Structure modifications require changes in all application programmes.
3. There are drawbacks in terms of implementation (no multiparent or M:N relationships).
4. The DBMS lacks a data definition or data manipulation language.



Hierarchical Model

Network Data Model

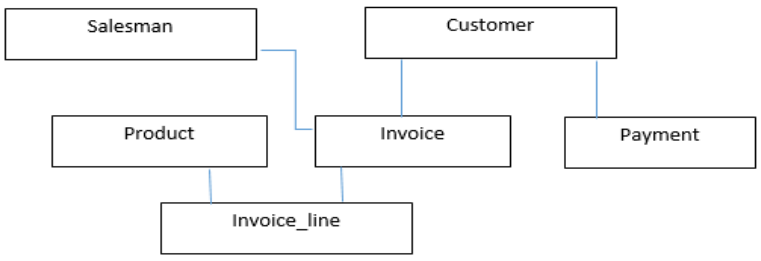
The network model was developed to express complicated data relationships than the hierarchical model, increase database performance, and establish a database standard. The user understands the network database as a collection of entries with 1:M associations in the network model. The network model, unlike the hierarchical approach, allows a record to have multiple parents.

Advantages:

1. The conceptual simplicity is almost similar to hierarchical models.
2. It supports new relationships, including M: N and multiparent relationships.
3. Access to data is more flexible than in hierarchical and file system architectures.
4. The relationship between the Data Owner and the Member improves data integrity.
5. A database management system (DBMS), supports data definition language (DDL) and data manipulation language (DML).

Disadvantages:

1. The complexity of the system reduces its efficiency
2. Navigation, application development, and management of network models are difficult.
3. Modifications in the structure require changes in all application programs.



Check your progress -1

1) The _____ model was created to manage large volumes of data for complicated manufacturing projects

a) hierarchical b) network c) both a and b d) none

2) _____ model supports multiparent relationship

a) network b) hierarchial c) both a and b c) none

3) _____ allow us to visualize the relationship between entities and their Properties

a) data models b) files c) entities d) all the above

Entity-Relationship Data Model

It's a graphical representation of a database's structure that uses entities and their relationships. The ER Model was a close match to the relational data model. The following components of the E-R model are listed below:

Entities: It's a physical object for which data is gathered and stored. It's merely one of many entities in a collection. Although the terms entity and entity set are distinct, they can be interchanged. In an E-R diagram, an entity set is represented by a rectangle. The entity's name is usually a singular noun.

Its generally represents a person, a place, a thing or an event.

Examples of Entities: salesman, warehouse, product, seminar

Attribute: An attribute is a term used to describe an entity. A single object can contain several properties, such as Student can have student_id, student_name, student_emailid

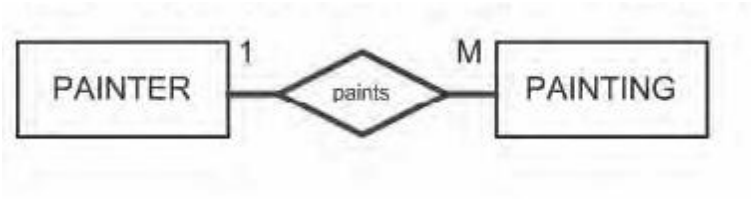
There are different types of attributed like:

Single valued attribute: A single-valued attribute is one that has just one possible value. A person's Vehicle license number is limited to one person, while manufactured product is limited to one product code. It's important to remember that a single-valued attribute isn't always a simple attribute.

Multi valued attribute: Attributes that have several values are known as multivalued attributes. For example, an individual may have many college degrees, and a family may have multiple phone numbers.

Derived attribute: A derived attribute is one that is obtained from another attribute. A derived attribute is one whose value is derived from the values of other properties. The derived property does not have to be physically recorded in the database; rather, it can be calculated using an algorithm.

Relationships: Relationships are the connections that exist between two entities. One-to-one (1:1), one-to-many (1:M), and many-to-many (M:N). are the three sorts of possible relations between the entities.



Example of One to Many relationship

A strong relationship, also known as an identifying relationship, exists when the Primary Key of the related entity contains Primary Key component of the parent entity.

Cardinality

The minimum and maximum number of entity occurrences connected with one occurrence of the related entity is known as cardinality. Cardinality is specified in the ERD by inserting the necessary numbers next to the entities.

Advantages:

1. Conceptual simplicity is achieved through visual modeling.
2. It is a powerful communication tool due to its visual depiction.
3. It works in conjunction with the prevailing relational model.

Disadvantages:

1. Constraint representation is restricted.
2. Relationship representation is minimal.
3. There is no language for manipulating data.

Check your progress 2

1) _____ is a graphical representation of a database's structure that uses entities and their relationships

a) ER model b) attributes c) entitites d) all the above

2) An _____ is a term used to describe an entity

a) attribute b) entity c) relationships d) all the above

3) ____ generally represents a person, a place, a thing or an event.

a) entity b) attributes c) relations d) none of these

4) A _____ relationship, also known as an identifying relationship

a) strong b) weak c) cordial d) none

Relational Data Model

A complex relational database management system is used to implement it (RDBMS). It not only accomplishes the same fundamental operations as the hierarchical and network models, but it also allows the end-user to be unaware of the complexity of the relational model. A table is a matrix made up of a sequence of row/column intersections that are linked by a common entity attribute. A relational diagram depicts the entities, characteristics inside those entities, and relationships between those entities in a relational database. A relational table is similar to a file in that it maintains a collection of connected things. A relational table is solely a conceptual structure, and it makes no difference where the data is physically stored in the database.

empid	empname	deptid
e1	mohan	d1
e2	rohan	d2

deptid	deptname
d1	sales
d2	marketing
d3	purchase

Advantages:

1. The usage of separate tables promotes structural independence. Data access and application programs are unaffected by changes in a table's structure.
2. The tabular view greatly enhances conceptual simplicity, making database design, implementation, maintenance, and use much easier.
3. Structured query language is used to do ad hoc queries.

Disadvantages:

1. The RDBMS needs a significant amount of hardware and system software.
2. Conceptual simplicity makes relatively inexperienced users misuse a good system, and if left unchecked, it may result in the same data anomalies that occur in file systems.

Check your progress – 3

1) relational table is a ____ structure

a) conceptual b) physical c) both d) none

2) RDBMS stands for

a) Relational database management system

b) Relational design management system

c) Redesigned database management system

d) Redesinged data management system

3) ___ is a matrix of rows and columns

a) table b) database c) tuple d) all of these

Object-Oriented Data Model

The object-oriented data model is a logical data model based on the object-oriented

programming idea. It was created to address the growing complexity of real-world applications that are not easily answered by existing models. A class represents both the object's properties and the entity's behavior. The data and their relationships are bothincluded in the instance of the class - object. Information describing the relationship between the facts inside an object, as well as information about relationships with other things are included in an object. All actions that may be done on an object are also stored in the object.

Object-Oriented Data Model is based on several components:

• An object is a representation of a physical entity. In general, an object may be thought of as the entity of an ER model. A single instance of an entity is represented by an object.

• The properties of an object are described via attributes.

• Classes are groups of objects with similar features. A class is a group of objects that have similar characteristics and actions.

• A class hierarchy is used to arrange classes. The class hierarchy resembles an upside-down tree with just one parent for each class.

• The capacity of an object inside the class hierarchy to inherit the properties and methods of the classes is known as inheritance.

Advantages:

1. The addition of semantic content.
2. Semantic material is included in the visual depiction.
3. Data integrity is aided via inheritance.

Disadvantages

1. Slow standard development led to vendors providing their own modifications, resulting in the abolition of a widely acknowledged standard.
2. It is a sophisticated navigation system.
3. Transactions are slowed by high system overhead.

There are a few traits that data models must share in order to be broadly accepted throughout the evolution of data models:

A data model must exhibit some conceptual simplicity without sacrificing the database's semantic completeness. Having a data model that is harder to imagine than the real world does not make sense.

A data model must be as accurate a representation of reality as feasible. More semantics in the model's data representation makes it easier to achieve this goal. Data representation is the static component of the real-world scenario, whereas semantics is concerned with the dynamic behaviour of the data.

Any data model's consistency and integrity requirements must be met in order for the representation of real-world transformations (behaviour) to be valid.

Check your progress – 4

1) An ____ is a representation of a physical entity.

a) object b) event c) model d) none of these

2) Classes are groups of objects with similar _____

a) features b) events c) models d) all the above

3) The properties of an object are described by _____.

a) attributes b) events c) models d) none of these

4) Object oriented data model was introduced to manage

a) Real world applications b) basic applications only c) both a and b d) none

Each new data model corrects the flaws of the prior models. Because the network model made it much easier to depict complicated (many-to-many) relationships, it supplanted the hierarchical approach. As a result of its simpler data representation, improved data independence, and easy-to-use query language, the relational model has become the preferred data model for commercial applications over the hierarchical and network models. Within a rich semantic framework, the OO data model provides support for complex data. The ERDM enhanced the relational model with several OO capabilities, allowing it to maintain its dominant market position in the commercial world.

It's vital to remember that not all data models are created equal; some are better suited to certain tasks than others. Conceptual models, for example, are better for high-level

data modelling, whereas implementation models are better for managing stored data for implementation. A conceptual model would be the entity relationship model, while implementation models would be the hierarchical and network models. Some models, such as the relational model and the OODM, can be utilised as both conceptual and implementation models at the same time.

3.2 LET US SUM UP

This unit gives unique learning different models of databases and its advantages and disadvantages. A data model is a representation of data structures and their properties, relations, constraints, transformations, and other components in a database environment for the goal of supporting a certain problem area. Then we learn about The hierarchical model. It was created to manage large volumes of data for complicated manufacturing projects

3.3 ANSWER FOR CHECK YOUR PROGRESS

Check your progress 1

Answers: (1-a), (2-a), (3-a)

Check your progress 2

Answers: (1-a), (2-a), (3-a) (4-a)

Check your progress 3

Answers: (1-a), (2-a), (3-a)

Check your progress 4

Answers: (1-a), (2-a), (3-a) (4-a)

3.4 ASSIGNMENT

1. Explain various components of the database and its features.
2. Write a note on architecture of DBMS
3. Compare and contrast different database models.

3.5 FURTHER READING

1. Database Management Systems - Rajesh Narang -PHI Learning Pvt Ltd.
2. Database System Concepts by Silberschatz, Korth -Tata McGraw-Hill Publication.
3. An Introduction to Database Systems - Bipin Desai- Galgotia Publication.
4.Database Management System by Raghu Ramkrishnan- Tata McGraw-Hill Publication.

Unit 4: Database Design

4

Unit Structure

4.0 LEARNING OBJECTIVES

After learning this unit, you will be able to understand:

- Database Design process
- Tables, rows, and columns
- Process of designing a database

4.1 DATABASE DESIGN

The efforts centered on the design of the database structure that will be used to store and manage end-user data are referred to as database design. A database that fits all of the user's needs does not appear out of nowhere; its structure must be meticulously planned. In fact, database design is an important component of database operations. With an improperly built database, even a decent DBMS will perform poorly.

In order to create a good database, the designer must first figure out what the database will be used for. The need of reliable and consistent data, as well as operating speed, is emphasised while designing a transactional database. The utilisation of historical and aggregated data is stressed while creating a data warehouse database.

The combined data must be correctly dissected into its component components, with each item being recorded in its own table. Furthermore, the links between these tables must be carefully researched and executed in order for the integrated view of the data to be re-created as information for the end user afterwards. A well-designed database makes data administration easier and delivers accurate and useful data. A badly constructed database is likely to become a breeding ground for hard-to-trace mistakes, which can lead to poor decision making—and poor decision making can lead to an organization's downfall. The design of a database is far too critical to be left to chance. That is why database design is too important of organisations of all sizes.

Database design is the process of building a complete database data model that includes all of the logical and physical design options as well as physical storage considerations required to create a database design. It should always reflect the

information system and should be evaluated and revised as part of a database life cycle strategy.

Database design can be divided into two categories:

Top-Down and Bottom-up approach

We identify the dataset and define the data pieces using a top-down method. We find data pieces first, then aggregate them into datasets, using a bottom-up strategy.

Centralised and Decentralised approach

In centralized database design is conducted by a single person or a small team on the contrary in decentralized database design large number of relationship and complex relations exits and are spread across multiple sites

4.1.1 Tables

The logical view of the relational database and creation of data relationships based on a logical construct known as tables. A table is a mathematical construct, users find it easier to think of a relation as a table. A table is a two-dimensional structure composed of rows and columns. It is also called a relation.

The development of data relationships based on a logical construct known as a relation helps to facilitate the logical view of a relational database. End users find it much easier to think of a relationship as a table because it is a mathematical construct. A table is a two-dimensional structure with rows and columns. Because E. F. Codd, the inventor of the relational model, adopted the term relation as a synonym for table, a table is also referred to as a relation. A table can be thought of as a persistent representation of a logical relation, one whose contents can be kept for future use.

A table is a collection of linked entity occurrences or an entity set. A STUDENT table, for example, has a collection of entity occurrences, each of which represents a student. As a result, a table is also referred to as an entity set. The table include rows and columns, also known as records and variables

Characteristics of a Relational Table

1. A table is perceived as a two-dimensional structure composed of rows and columns.
2. Each table row (tuple) represents a single entity occurrence within the entity set.
3. Each table column represents an attribute, and each column has a distinct name.
4. Each row/column intersection represents a single data value.
5. All values in a column must conform to the same data format.
6. Each column has a specific range of values known as the attribute domain.
7. The order of the rows and columns is immaterial to the DBMS.
8. Each table must have an attribute or a combination of attributes that uniquely Identifies each row.

Thus, a relational database's basic building blocks are tables. A table stores an entity set, which is a collection of connected entities. In terms of structure, a relational table is made up of intersecting rows (tuples) and columns. Each column reflects the entities' qualities (attributes), while each row represents a single entity.

Student id	Student name	Student contact	Student emailid
101	Tapan	982519825	101@baou.edu.in
102	Palak	786545678	102@baou.edu.in
103	Nikunj	912342349	103@baou.edu.in
104	Jaikishan	543212345	104@baou.edu.in
105	Om	789000789	105@baou.edu.in

i. The STUDENT table is considered as a two-dimensional structure with four columns and five rows.

ii. The STUDENT table shows a single entity occurrence inside the entity set for each row. For example, any student id=101 reflects the attributes in the table, such as the student's name in this caseidentifies a record

iii. Each column is treated as an attribute, and each one should be given a distinct name.

iv. Every attribute in a column must be of the same data type. The data type for the designating name field is character.

v. The table has student id as primary key which identifies all other attributes of a student

vi. The table can be related to all tables using a primary key foreign key relationship

vii The relational table is created in order to reduce the redundancy which was the disadvantage of the file systems. Thus relational table can be linked to the relational tables and can eliminate / reduce the duplication of data and thus remove the inconsistencies in the database

Check your progress-1

1) Each table ___ represents a single entity occurrence within the entity set.

a) row b) column c) both a and b d) none

2) Each row/column intersection represents a ____ data value

a) single b) multiple c) complex d) all the above

3) Each column has a specific range of values known as the _____

a) domain b) set c) attribute d) none

4) a relational database's basic building blocks are ___

a) tables b) entities c) rows d) none of these

4.1.2 Rows

Each row in the table is called a tuple which represents a single entity within the table.

Student id	Student name	Student contact	Student emailid
101	Tapan	982519825	101@baou.edu.in
102	Palak	786545678	102@baou.edu.in
103	Nikunj	912342349	103@baou.edu.in
104	Jaikishan	543212345	104@baou.edu.in
105	Om	789000789	105@baou.edu.in

In the above example, the row with student id 101 describes an entity of one student with

all its attributes

4.1.3 Domains

Each column has a specific range of values known as the attribute domain. For example student id contains numbers within a specific range which is called domain of that attribute.

A domain exists for attributes. A domain is the set of potential values for a given property, the domain for the salary attribute, for example, is number because the salary attribute is a number. There are just two options for the gender attribute's domain: M or F. (or some other equivalent code). All dates that fit within a given range make up the domain for a company's date of hire attribute.

A domain can be shared by attributes. A student address and a professor address, for example, both belong to the same domain of all conceivable addresses. In fact, if the

same attribute name is used, the data dictionary may allow a newly declared attribute to inherit the properties of an existing attribute.

4.1.4 Columns

Each table column represents an attribute, and each column has a distinct name and all values within a column should have the same data format or data type.

Student id	Student name	Student contact	Student emailid
101	Tapan	982519825	101@baou.edu.in
102	Palak	786545678	102@baou.edu.in
103	Nikunj	912342349	103@baou.edu.in
104	Jaikishan	543212345	104@baou.edu.in
105	Om	789000789	105@baou.edu.in

In above example the column studentid with values 101 to 105 represents a single attribute

The order of rows and columns is not important to the user. The primary key column may not always be the first column or student name can be the last column of the table.

Each column will have specific range of values which we call domain of the column. The columns are basically the properties of the entity which is represented by the table itself. There will be a column in each table which can uniquely identify other attributes of the same table. For example student id can identify all other details of the student

Student id	Student name	Student contact	Student emailid
101	Tapan	982519825	101@baou.edu.in
102	Palak	786545678	102@baou.edu.in

Student id	Student name	Student contact	Student emailid
103	Nikunj	912342349	103@baou.edu.in
104	Jaikishan	543212345	104@baou.edu.in
105	Om	789000789	105@baou.edu.in

Table : student_marks

Student id	Test	Marks1	Marks2	Marks3
101	Test1	10	20	20
102	Test1	20	10	15
103	Test1	20	15	10
104	Test2	15	20	10
105	Test2	10	10	15

Here the student_marks table has student_id as are related to the student table.

Relationships that are defined in relational database are of three types:
• One-to-many (1:M)
• One-to-one (1:1)
• Many-to-many (M:N)

1) **One to many**: The 1:M relationship is the relational database standard. To thishow this relationship is designed let us consider a simple example of student table and student marks.
Here one student can have many tests and respective marks,

2**) one to one**: This relationship describes that one value in a relational table is related to only one value in another table. For example one employee works in only one department or one painter paints a painting.

3) **Many to many**: This relationship describes that many values in one relational table are related to many values in another table. For example many customers purchase many products or many students study many subjects.

4.1.5 Database Design Process

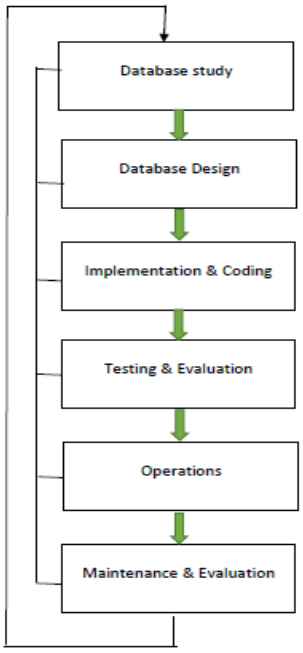
Database design Life Cycle

Phase-1: Database study

During the first phase we examine the structure of the organisation as well as its working environment. We start by defining the problem and making a list of all the limitations.
We must also outline the proposed system's key goals, as well as its scope and limitations.

Phase 2: Database design

During the second phase the database administrator must focus on data needs, establish a conceptual design, construct a logical design, and thereafter create a physical design during this phase.
*) Conceptual Design: We map the database to real-world elements through conceptual design.
We conduct data analysis here, construct an entity relationship model from the requirements, normalise it to its required form, and then verify the data model that has been developed.
*) The logical design converts the conceptual design into a model consisting of Tables, Indexes,
Views, Transactions, and other logical model design components.
*) In physical design the physical storage devices and the components used to access the database are described.



Phase 3: Implementation and coding

The developing storage structure for the end user tables is part of this step. During this phase performance, security, backup and recovery, transaction management and concurrency control are also addressed.

Phase 4: Testing and Evaluation

In this step, the database is tested and fine-tuned for performance, integrity, concurrent access, and security requirements.

Phase 5: Operations

During the operations phase the database is available for executions and the database performance is evaluated. During the execution, the database may be tuned for unforeseen problems.

Phase 6: Maintenance

We use several maintenance strategies in this phase, such as preventive maintenance, corrective maintenance, adaptive maintenance, database statistics are calculated for performance monitoring are carried out.

Check your progress -2

1) We map the database to real-world elements through _____ design.
a) conceptual b) physical c) both a and b d) none of these

2) ___ is not a type of maintenance
a) assistive b) corrective c) preventive d) adaptive

3) The ____ design converts the conceptual design into a model
a) logical b) physical c) both and b d) none of these

4.1.6 Anomalies in Database

Anomalies are problems that can occur in non-normalized databases that have been badly built. Non-normalized databases are those that are designed and developed without adhering to database standard rules. When referring to attributes in related tables, numerous types of anomalies can occur.

The basic reason for the generation of anomalies is data redundancy, the same data is duplicated in multiple locations. A data anomaly develops when not all of the required changes in the redundant data are made properly.
Consider two records
1) STUDENT
Student_name, Student_phone, Student_address, Student_pincode

2) MARKS
Student_Name, Student_phone, Subject, marks

Update anomalies. If a student has a new phone number, that number must be entered in the student file and not updated in marks records. It can happen if records are large.
If it is updated in the student file but not in the marks file then creates an update anomaly and ultimately leads to inconsistency.

Insertion anomalies, If data is inserted at one location and it is not inserted in another file. Again, the potential for creating data inconsistencies would be there. If a new student is added to the student file but the same record is not inserted in the marks file it leads to an insertion anomaly and again it leads to inconsistency of records in both the files.

Deletion anomalies. If you delete the record in one file eventually it is not deleted in the

related files, it is referred to as a deletion anomaly. If a student record is deleted from the student file, its corresponding record has to be deleted in the marks file. It again creates inconsistency of records in both the files.

From the above illustrations, we can conclude that the anomaly occur because of the redundancy of data and in turn leads to inconsistency.

Check your progress -3

1) Redundancy gives rise to _____
a) anomalies b) consistency c) serializability d) none
2) Anomalies give rise to _____
a) inconsistency b) redundancy c) serializability d) none
3) ____ is not a type of anomaly
a) creation b) insertion c) deletion d) all the above
4) ___ is evaluated in operations phase
a) performance b) testing c) tuning d) none of these

4.2 LET US SUM UP

This unit describes the concepts of redundancy, anomalies and inconsistencies occurring due to the redundancy of data. It also describes the database design process and database components.

A few definitions to remember:

Data
Raw details like a phone number, a birth date, a student name, Data is meaningless unless it has been structured in a reasonable manner.

Field A character (alphabetic or numeric) or a collection of characters with a defined meaning. A field is a type of variable that is used to define and store data.

Record A person, place, or object is described by a logically linked collection of one or more fields. The fields that make up a customer record, for example, might include the students name, address, phone number, date of birth.

File A grouping of tracks with a same theme. For example, a file may contain information on University's present students.

4.3 ANSWER FOR CHEK YOUR PROGRESS

Check your progress 1
Answers: (1-a), (2-a), (3-a) (4-a)

Check your progress 2
Answers: (1-a), (2-a), (3-a)

Check your progress 3
Answers: (1-a), (2-a), (3-a) (4-a)

4.4 ASSIGNMENT

1. Explain characteristics of relational tables.
2. Explain the terms table, domain, rows and columns

4.5 ACTIVITIES

Write a note on how redundancy creates anomalies and in turn inconsistencies with a suitable example.

4.6 CASE STUDY

Explain the benefits of studying Database design process

4.7 FURTHER READING

1. Database Management Systems - Rajesh Narang -PHI Learning Pvt Ltd.

2. Database System Concepts by Silberschatz, Korth -Tata McGraw-Hill Publication.

3. An Introduction to Database Systems - Bipin Desai- Galgotia Publication.

4.Database Management System by Raghu Ramkrishnan- Tata McGraw-Hill Publication.

BLOCK 2: NORMALISATION & SERVER ARCHITECTURE

This block will cover the fundamentals of databases and database management systems. With the aid of many examples, it offers an outline of the aforementioned topic.

You'll also gain a fundamental understanding of what data, databases, and database design are. In this lock, several models have been explained in a very concise and intriguing manner. Basic principles, architecture, models, and database design are the key subjects addressed in this lock.

The purpose of this lock is to help the reader know the fundamental notions of this area. The writer has done his best to explain the principles; in addition, the sub-topics have been thoroughly covered by him. On the other hand, he has gone into great length on database architecture, with many of examples.

This lock's objective is to describe the student's understanding of the numerous above-mentioned concepts in great depth, using sufficient and appropriate examples.

Block Objectives

After learning this block, you will be able to understand:

- Purpose of database
- Why database is important over file systems
- Database models
- The rows and columns within a table and their use
- The architecture of the database
- The database design process
- Anomalies within the database

Unit 1: Functional Dependencies

1

Unit Structure

1.0 LEARNING OBJECTIVES

After learning this unit, you will be able to understand:

- Database concepts
- Architecture of database
- Tables, rows, and columns
- Process of designing a database

1.1 INTRODUCTION

Functional dependencies (FDs) are an important factor in distinguishing between excellent and bad database architectures. A functional dependence is a kind of restriction that is a broadening of the concept of important functional dependencies. Formalism on relation infrastructure is represented by FDs, which are restrictions on well-formed relations. The identification of functional relationships is a critical component of relational database architecture, as well as database normalization and de-normalization. The functional dependencies, as well as the attribute domains, are chosen to establish constraints that will keep as much data out of the system that is irrelevant to the user domain. Normalization (NF) is a method of assuring that a database structure is acceptable for general-purpose querying and free of undesired features such as insertion, update, and deletion anomalies, which might result in data integrity being compromised. The usual forms of relational database theory give criteria for identifying a table's level of logical inconsistency and anomaly vulnerability. Individual tables are affected by the normal forms; to state that a database is in normal form n means that all of its tables are in normal form n.

Check your Progress 1

1 Functional Dependency is a relationship between _____.

(A) Attribute and Value

(B) Attribute and Attribute

(C) Attribute and Table

(D) None of these.

2. NF stands for

a) Normal form

b) Normal functions

c) Normal facts

d) All the above

3) A _____ dependence is a kind of restriction that is a broadening of the concept of important functional dependencies

a) functional

b) factual

c) non functional

d) none of these

1.2 FUNCTIONAL DEPENDENCIES

Functional Dependency (FD) is a constraint in a Database Management System that establishes the relationship of one attribute to another attribute. Functional Dependency aids in the preservation of database data quality. It is critical for distinguishing between excellent and bad database design.

An arrow → denotes a functional dependence. X →Y represents the functional relationship between X and Y. Let's look at an example of functional dependency in a database management system.

Student Number	Name	City	Pecentage
1	Ashok	Ahmedabad	88
2	Biren	Pune	67
3	Chirag	Jamnagar	78

In this example, if we know the value of Student number, we can obtain Student Name, city, Percentage, etc. By this, we can say that the Student Name, city and Percentage are functionally depended on Student number.

1.2.1 Rules of Functional Dependency

reflexive rule: If X is a set of attributes and Y is a subset of X, then X has the value Y.

Transitivity rule: This rule is very much similar to the transitive rule in algebra if x -> y and y -> z are true ,then x -> z is true as well. X -> y is the functional relationship that determines y.

Augmentation rule: When x -> y holds, and c is attribute set, then ac -> bc also holds. That is adding attributes which do not change the basic dependencies.

1.2.2Types of Functional Dependency

Trivial Function Dependency

A collection of attributes is considered a trivial dependence if all of the attributes in the set are contained in that attribute.

If Y is a subset of X, then X -> Y is a simple functional dependence. Let's look at a simple functional dependency example.

Consider this table of with two columns Student_id and Student_name.

{Student_Id,Student_name} ->Student_id is a trivial functional dependency as Student_id is a subset of {Student_id,Student_name}.

Non Trivial Function Dependency

When A->B holds true but B is not a subset of A, this is referred to as a functional dependency, also known as a nontrivial dependency. If attribute B is not a subset of attribute A in a relationship, it is termed a non-trivial dependence.

{Company} -> {CEO} (if we know the Company, we knows the CEO name)

But CEO is not a subset of Company, and hence it’s non-trivial functional dependency.

Transitive Functional Dependency

In transitive functional dependency, dependent is indirectly dependent on determinant.
i.e. If **X** → **Y** & **Y** → **Z**, then according to axiom of transitivity, **X** → **Z**. This is a **transitive**

Emp_Id → Department, Department→Bulding_no

Hence according to transitive dependency rules Emp_id→Building_no

Multivalued Dependency

When there are numerous independent multivalued attributes in a single table, multivalued dependence occurs. A full constraint between two sets of attributes in a relation is a multivalued dependence. It needs the presence of specific tuples in a relation. To grasp the concept, see the Multivalued Dependency Example below.

Car_model,Maf_year and Color are three attributes in single table.In this example, Maf_year and Color are independent of each other but dependent on Car_model. In this example, these two columns are said to be multivalue dependent on car_model.This dependence can be represented like this:Car_model -> Maf_yearCar_model-> Colour

Check your Progress 2

Q1 In _____ dependency one attribute is subset of fields of Table.

a) Trivial
b) Non-Trivial
c) Transitive
d) None of these

Q2 When A->B holds true but B is not a subset of A, this is referred to as a functional dependency, also known as a ____ dependency

a) Non trivial
b) Trivial
c) Transitive
d) None of these

Q-3 FD stands for

a) Functional dependency
b) Functional data
c) Functional database
d) None of these

1.3 ARMSTRONG AXIOMS OF FUNCTIONAL DEPENDENCIES

F+, the fastener of F, endures the set dependently all-inclusive functional dependencies theoretically intimated adjacent F if F continues to set relevantly functional dependencies. Armstrong's codes are a series of commands that, when approached, commonly causes blockage of functional dependencies.

reflexive rule: If X is a set of attributes and Y is a subset of X, then X has the value Y.

Transitivity rule: This rule is very much similar to the transitive rule in algebra if x -> y and y -> z are true ,then x -> z is true as well. X -> y is the functional relationship that determines y.

Augmentation rule: When x -> y holds, and c is attribute set, then ac -> bc also holds. That is adding attributes which do not change the basic dependencies.

Check your progress 3

Q1 In _____ rule N set of attribute and Z is a subset of N then N has all the value of Z.

(A) Transitivity
(B) Reflexive

(C) Augmentation

(D) All the Above

Q-2 When x -> y holds, and c is attribute set, then ac -> bc also holds is a___ rule

a) Augmentation rule
b) Reflexive rule
c) Transitive rule
d) None of these

Q-3 If X is a set of attributes and Y is a subset of X, then X has the value Y.

a) Reflexive rule
b) Augmentation rule
c) Transitive rule

1.4 LET US SUM UP

We spoke about database dependencies and the normalisation process in this chapter. We've looked at the process of functional dependence with a variety of people. We've learned about the FDs' Inference Rules. We've also broken down the Normalization Process into distinct Normal Forms. After completing this chapter, students will be able to properly normalise the database.

1.5 GLOSSARY

Functional Dependency: Functional Dependency (FD) is a constraint in a Database Management System that determines the relationship of one attribute to another (DBMS).

1.6 ANSWER FOR CHECK YOUR PROGRESS

Check your Progress 1

Q1 B Q-2 A Q-3 A

Check your Progress 2

Q1 AQ-2 A Q-3 A

Check your Progress 3

Q1 AQ-2 A Q-3 A

1.7 ASSIGNMENT

Q1 Explain Functional Dependency with Example.

Q2. What is the rules of functional Dependency?

1.8 Activity

Write a short note on Functional Dependency Types.

1.9 Case Study

Design a Table to show types of Functional Dependency.

1.10 Further Reading

1. Database Management Systems - Rajesh Narang -PHI Learning Pvt Ltd.

2. Database System Concepts, 6th Edition, Abraham Silberschatz, Henry F. Korth, S.Sudarshan, McGraw Hill.

Unit 2: Stages of Normalization

2

Unit Structure

2.0 LEARNING OBJECTIVES

After learning this unit, you will be able to understand:

- What is Normalization and why it is require?
- Types of Normalization, and how to implement in Database Design?
- Advantages and Disadvantages of Normalization.

2.1 NORMALIZATION

Normalization is a database design approach that avoids undesired features such as Insertion, Update, and Deletion Anomalies by reducing data redundancy. Normalization rules break huge tables into smaller tables and use relationships to connect them. The goal of SQL normalisation is to reduce duplicate (repetitive) data and guarantee logical data storage.

The major goal of normalising the relationships is to eliminate these inconsistencies. Failure to remove anomalies results in data redundancy, which can lead to data integrity and other issues as the database grows. Normalization is a set of rules that guide you through the process of building a decent database structure. The major goal of normalising the relationships is to eliminate these inconsistencies. Failure to remove anomalies results in data redundancy, which can lead to data integrity and other issues as the database grows. Normalization is a set of rules that guide you through the process of building a decent database structure.

Check your Progress 1

1 _____ is used to break large table In to small table and use relationship to connect them

(A) Dependency (B) Normalization (C) ERD (D) none of these

2.2 STAGES OF NORMALIZATION

Normalization is carried out via a succession of stages known as Normal forms. Individual relationships are subject to the normal forms. If a relation fulfils constraints, it is said to be in specific normal form.

Database Normal forms:

1NF First Normal Form

2NF First Normal Form

3NF First Normal Form

4NF First Normal Form

1 NF First Normal Form

A relation violates first normal form if it has composite or multi-valued attributes, or it is in first normal form if it does not contain composite or multi-valued attributes. If every attribute in a connection is a singly valued attribute, it is said to be in first normal form.

Faculty Name	Subject	Semester
Manish Patel	DS,C++,Java	3,4
Anand Patel	Python,DBMS	2,5
Sagar Bhavsar	PHP,OS	6,3

In above Table Subject and Semester are multivalued Attribute it is not 1NF normal form.

After we apply 1NF we will arrange Data in proper manner and Isolate the multivalued in single Column.

Faculty Name	Subject	Semester
Manish Patel	DS	3
Manish Patel	C++	3
Manish Patel	Java	4
Anand Patel	Python	5
Anand Patel	DBMS	2

2NF Normal Form

A relation must be in first normal form and contain no partial dependencies to be in second normal form. A relation is 2NF if it has No Partial Dependency, which means

that no non-prime attribute (attributes that are not part of any candidate key) of the table is dependent on any suitable subset of any candidate key.

Partial Dependency — Partial dependency occurs when the proper subset of candidate keys determines the non-prime attribute.

Faculty Name	Subject_ID	Hrs
Manish Patel	1	18
Manish Patel	2	12
Manish Patel	3	18
Anand Patel	4	14
Anand Patel	5	12

In above table Hrs is mention for each subject and that is similar in many subjects code so it is non-prime attribute. In 2 NF we define one more table with Subject Table and Subject_Id is a primary key and Hrs is dependent on Subject_Id so while we mention Subject id with faculty Table will automatic relate with hrs.

Faculty Table		Subject Table	
Faculty Name	**Subject_Id**	**Subject_Id**	**HRS**
Manish Patel	1	1	18
Manish Patel	2	2	12
Manish Patel	3	3	18
Anand Patel	4	4	14
Anand Patel	5	5	12

3 NF Normal Form

If there is no transitive dependency for non-prime attributes and the relation is in second normal form, it is in third normal form.

Every non-trivial function dependence must satisfy at least one of the following conditions for a relation to be in 3NF.

What is Transitive Dependency?

When an indirect relationship causes functional dependency it is called Transitive Dependency.
If P -> Q and Q -> R is true, then P-> R is a transitive dependency.
To achieve 3NF, eliminate the Transitive Dependency.

Student Table

Student_ID	Student Name	State	Country	Mobileno
1	Sandip Shah	Gujarat	India	9090876543
2	Ram	Maharashtra	India	9988776655
3	Shyam	Punjab	India	1234567890

In above table we define State and Country field and hrere trnsitive dependency is created throu Student_Id→State State→Country so Student_Id→Country. To remove Transitive Dependency we create one more table in 2nf and mention Country in that table with prime attribute State ID so we can direct relate that id with student table as a foreign key attribute.

After 3NF

Student Table

Student_ID	Student Name	State	Mobileno
1	Sandip Shah	Gujarat	9090876543
2	Ram	Maharashtra	9988776655
3	Shyam	Punjab	1234567890

State Table

State	Country
Gujarat	India
Maharashtra	India
Punjab	India

Definition: A relation is in BCNF, if it is in 3NF and if every determinant is a candidate key.

R is in BCNF if R is in Third Normal Form and LHS is super important for every FD. If X is a super key in every non-trivial functional dependency X –> Y, the relation is in BCNF.

BCF must be free from redundancy. If a relationship is in BCNF, it is also in 3NF.The relation is always in 3NF if all of its characteristics are prime attributes. Every Binary Relation (one with only two properties) exists in BCNF. After BCNF, there are many more Normal versions, such as 4NF and others. However, in most real-world database systems, going beyond BCNF is not required.

Check your Progress 2

1 In _____ Normal Form multi valued records are arrange in single column.

(A) 2NF (B) 3 NF (C) 1NF (D) 5NF

2_____ is must be free from redundancy.

(A) 1 NF (B) 2 NF (C)BCNF (D) 3NF

3 In 3NF form _____ is removed.

(A) Duplication (B) Error (C) Transitive Dependency (D) All the above

2.3 ADVANTAGES AND DISADVANTAGES OF NORMALIZATION

Advantages of Normalization

- Normalization helps in the decrease of data redundancy
- Improved database structure in general.
- Within the database, there is data consistency.
- Database design that is far more adaptable.
- The principle of relational integrity is highlighted.

Disadvantages of Normalization

• You can't begin developing the database until you know what the user requirement.

• When the relationships are normalised to higher normal forms, such as 4NF and 5NF, the performance suffers.

• Normalizing higher-degree relationships takes a long time and is challenging.

Check your Progress 3

1 _____ is helpful to decrease the data redundancy.

(A) Normalization (B) 1NF (C) 2NF (D) ERD

2.4 LET US SUM UP

In this unit, we learned that normalization is a key measurement in the database development process because it presents the genuine look of the database and the workings of the data, which is useful for a specific database.

We can update relation schema using the normalization method in order to reduce layoffs. It is clear that all normalization periods will result in more database relations.

Database normalisation is used to reduce redundancy and reliance, according to research. It will most likely aid in the division of large tables into smaller ones in order to reduce redundancy and improve data relationships.

We've seen how decomposition is used to change relationships by gathering smaller ones in this unit. As can be seen, using decomposition in a database causes tables to be broken down into different tables, resulting in a higher normal form.

When you've met the requirements for a functional dependency and the set of characteristics on the left side of the functional dependency statement can't be condensed any further, you've reached full functional dependency.

2.5 GLOSSARY

Dependency: Dependencies in DBMS is a relation between two or more attributes.

Foreign Key: A foreign key is an identifier that is used to join two tables or build connectivity between them.

Normalization: Normalization is the process of organizing data in a database.

Partial Dependency: When one primary key determines the value of another characteristic or attributes, this is known as partial dependency.

Primary key - It is an attribute or attributes that uniquely identify an instance of an entity.

Transitive Dependency: When an indirect relationship causes functional dependency it is called Transitive Dependency.

2.6 ANSWER FOR CHECK YOUR PROGRESS

Check your Progress 1

Answers: 1 (B)

Check your Progress 2

Answers: 1 (C),2 (C), 3 (C)

Check your progress 3

Answers: 1 (A)

2.7 ASSIGNMENT

Explain Normalization with all the steps.

2.8 ACTIVITIES

Draw Ecommerce Database Tables using Normalization.

2.9 CASE STUDY

Design Database online Admission system for university

2.10 FURTHER READING

1. Database Management Systems - Rajesh Narang -PHI Learning Pvt Ltd.
2. Database Management Systems, Raghu Ramakrishnan and Johannes Gehrke, McGrawHill Publication.

Unit 3: Oracle Server and Instances

3

Unit Structure

3.0 LEARNING OBJECTIVES

After learning this unit, you will be able to understand:

- Oracle Server and Instance Architecture should be learned and understood.
- To comprehend Oracle Processes
- To comprehend the Oracle database's memory structure.
- To become familiar with various storage structures.
- To become familiar with schema and schema objects.

3.1 INTRODUCTION

Oracle Server is a database management system that offers an open, integrated, and complete approach to data administration. In general, an Oracle server must be able to consistently manage a significant volume of data in a multi-user environment, allowing multiple users to access the same data at the same time. All of this must be done while maintaining a high level of performance. Unauthorized access must be prevented, and a reliable failure recovery mechanism must be provided. Physical components, memory components, operations, and logical structures are all part of the architecture.

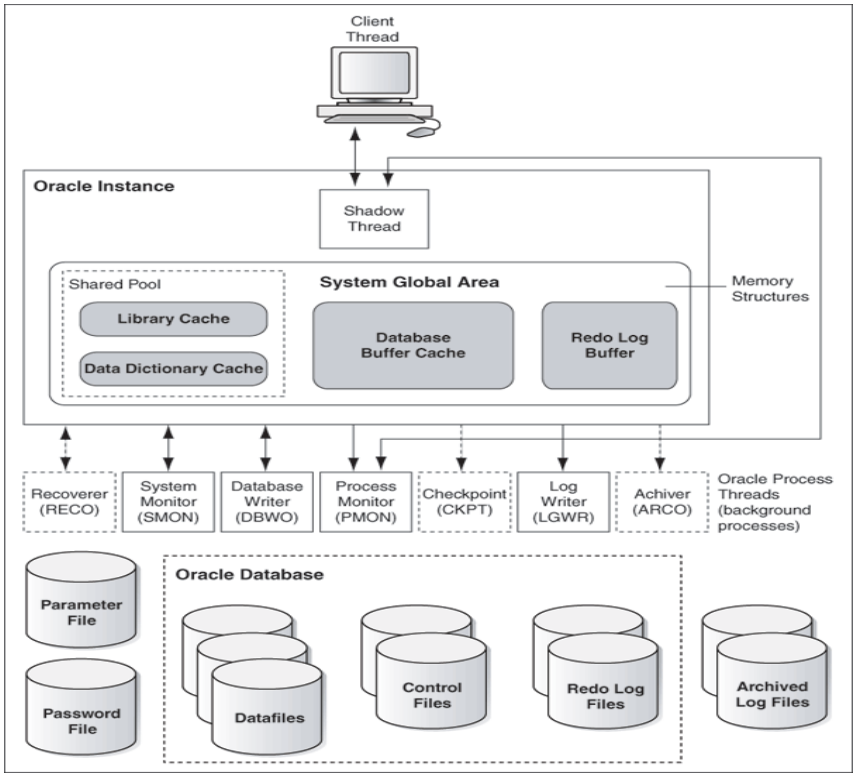


Fig. 3.1 Oracle Architecture

Oracle Server: An Oracle server consists of an Oracle instance and an Oracle database. The database contains a variety of file types, including data files, control files, redo log files, and archive redo log files. Parameter and password files are also accessed by the Oracle server. The following are some of the functions of this set of files:

- One goal is to allow system users to process SQL statements.
- Another is to boost system performance.
- And yet another is to ensure that the database can be recovered in the event of a software or hardware failure.

Oracle Instance: An Oracle instance is made up of two sets of components. Background processes such as SMON, PMON, DBW0/DBWR, RECO, LGWR, CKPT, D000, and others make up the initial component set. Each background process is essentially a computer programme. In order to provide good performance and database dependability, these processes execute input/output and monitor other Oracle processes.

The memory structures that make up the Oracle instance are included in the second component set. A memory structure called the System Global Area (SGA) is allocated when an instance starts up. The background processes begin at this stage as well. Access to an Oracle database is provided via the Oracle Instance. A single database is opened by an Oracle instance.

Oracle Database: An Oracle database is made up of files that are frequently referred to as operating system files, but are actually database files that store the database information that a company or organisation needs to function. A session is what happens when a user connects to an Oracle server. When the Oracle server validates the user for connection, the session begins. When the user signs out (disconnects) or the connection terminates abnormally, the session ends (network failure or client computer failure).A user can usually have multiple sessions open at the same time. The DBA sets the maximum number of concurrent session connections. Users can use this connection to run SQL commands. A dedicated server connection is a one-to-one contact between the User and Server Processes. Another option is to employ a Shared Server, which allows multiple User Processes to share a single Server Process.

Check your Progress 1

1 _____ is used to process SQL statements.

(A) Oracle Client (B) Oracle Server (C) Oracle Instance (D) All the above

2 DBA stands for

a) database administrator b) database activator c) data accessor d) none of these

3.2 DATABASE STRUCTURE

An Oracle Instance is connected with each operating Oracle database. When you launch a database on a database server, Oracle allocates a dedicated memory region called the System Global Area (SGA) and begins many Background processes. An Oracle Instance is a combination of SGA and Oracle Processes.

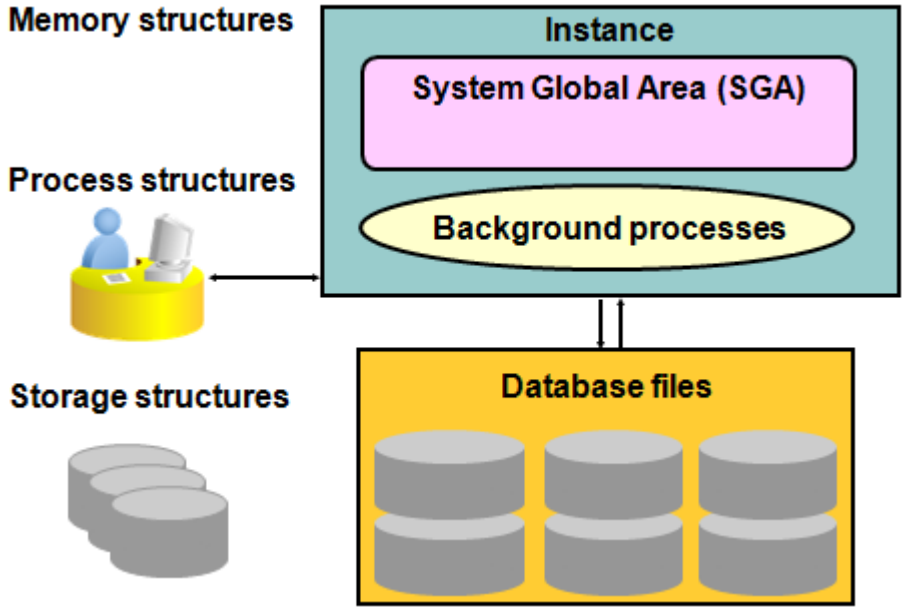


Fig. 3.2 Database Structure

Oracle associates an instance with a certain database after it is started. This is referred to as database mounting. The database is now ready to be accessed, allowing authorized individuals to access it. On the same computer, many instances can run simultaneously, each with its own actual database.

Check your Progress 2

1_____ is used to start background processes.

(A) System Global Area (B) Oracle Instance

(C) Program Global Area (D) none of these

3.3 STORAGE STRUCTURE

An Oracle database is made up of files that are frequently referred to as operating system files, but are actually database files that store the database information that a company or organisation requires to function. The following are the two parts of the database storage structures:

• Physical Arrangement.

• Logical Arrangement.

3.3.1 PHYSICAL DATABASE STRUCTURE

The files that make up an Oracle Database are divided into the following categories:

Control Files: This section contains information regarding the database itself. These files are necessary for the database to function. Without it, you won't be able to access data in the database since you won't be able to open data files. It's used to keep all database actions in sync.

Data Files: These files hold the database's actual data.

Redo Log Files: Store a record of database modifications and allow recovery in the event of a database failure. If the database fails but no data files are lost, the instance can recover the database using the information contained in these files.

Parameter file: The parameter file specifies how the instance will be configured when it starts up.

Password file: This file specifies which special users are allowed to start/stop an Oracle instance.

Allows users to connect to the database from after.

Archived Redo log files: Contain a running history of data changes caused by the instance. It's a copy of the redo log files that's required for recovery in an online transaction-processing environment in the case of a disc failure.

Backup files: These files are used to recover databases. Typically, backup files are restored after the original file has been destroyed or erased due to a media failure or user error.

Trace Files: A trace file can be created for each server and background process. When a process detects an internal issue, it writes information about the error to its trace file. Some of the data written to the trace file is for the database administrator's use.

Alert logs Files: Special trace files exist for alert log files. Alert logs are another name for them. A database's alert log is a chronological record of messages and problems.

3.3.2 Logical Structure

Understanding how an Oracle database is arranged in terms of a logical framework used to manage physical objects is beneficial.

Table space: Although a typical Oracle database will have several table spaces, an Oracle 10g database must always have at least two table spaces (SYSTEM and SYSAUX). Table spaces are logical storage facilities (logical containers) for storing database objects like tables, indexes, sequences, clusters, and other database objects.

At the operating system level, each table space has at least one physical data file that stores the table space. A big table space may have multiple data files dedicated to storing the table space’s items. A table space can only be used by one database. Except for the SYSTEM table space, which must always be online, table spaces can be brought online and taken offline for backup and administrative needs. Read-only and read-write table spaces are also possible.

Data file: Table spaces are stored on real disc objects known as data files. A data file can only hold objects for one table space, but a table space can have many data files — this happens when a disc drive device fills up and a table space needs to be expanded to a new disc drive. A data file’s size can be changed by the DBA to make it smaller or later. The file can also expand in size as the table space expands.

Segment: When logical storage objects, such as an employee table, are formed within a table space, a segment is assigned to the object. A table space, by definition, has numerous parts. A segment can't span table spaces, although it can span data files in the same table space.

Extent: Each item has a single segment, which is a collection of physical extents. Extents are essentially groups of disc storage blocks that are connected. A logical storage object, such as a table or index, always has at least one extent — ideally, the initial extend assigned to an object will be large enough to contain all data that is loaded at the start. Additional extents are added to the segment when a table or index increases. To tune the system's performance, a DBA can add extents to segments. A data file cannot be spanned by an extent.

Data Block: The Oracle Server organises data in what is known as a block or data block, which is the smallest unit. Blocks are used to store data. A physical block is the smallest addressable place for read/write operations on a disc drive. Because an Oracle data block is made up of one or more physical blocks (operating system blocks), the data block should be an even multiple of the operating system block size if it is larger than the operating system block size. For example, if the UNIX operating system block size is 2K or 4K, the Oracle data block size should be 2K, 4K, 8K, 12K, 16K, etc. This improves I/O performance.

The size of the data blocks is determined when the database is built and cannot be modified. The DB BLOCK SIZE parameter determines its size. The operating system determines the maximum data block size.

Check your Progress 3

1 In physical Database Structure_____ is allowed to access data in database.

(A) Data file (C) Redo log File (D) Control file (E) Backup File

2 _____ are logical storage containers for storing database objects like tables, indexes, sequences, clusters, and other database objects.

(A) Table Space (B) Data File (D) Control File (E) Segment

3.4 PROCESS STRUCTURE

The Oracle server creates a server process to execute the commands supplied by an application programme or an Oracle utility, such as Enterprise Manager, when you launch it. For each instance, the Oracle server establishes a series of background processes that interact with each other and with the operating system to maintain

memory structures, conduct asynchronous I/O to write data to disc, and do other essential duties. The features that are employed in the database determine which background processes a represent.

User Process: When a database user requests a connection to the Oracle Server, the user process begins. To utilise Oracle, you must first establish a connection to the database. Whether you're using SQL*Plus, an Oracle tool like Designer or Forms, or an application programme, this must happen. This establishes a User Process that makes programmatic calls through your user interface, creating a session and causing the creation of a dedicated or shared Server Process.

Server Process: When a user establishes a session and connects to an Oracle instance, the server process is initiated.

A User Process communicates with an Oracle Instance through a Server Process. In a Dedicated Server environment, each User Process is served by a single Server Process. A Server Process in a Shared Server context can serve several User Processes, albeit at a performance cost.

Background Process: When the Oracle Instance is specified, the background process begins. When an Oracle Instance starts up, both necessary and optional background processes are initiated, as seen here. All system users benefit from these background processes. For each instance, Oracle produces a group of background processes. Background processes asynchronously execute I/O and monitor other Oracle processes, resulting in increased symmetry and improved performance and reliability. DBWn, SMON, CKPT, PMON, ARCn, and RECO are the background processes.

The Database Writer: This (DBWn / DBWR) writes updated blocks to the data files from the database buffer cache. Although most systems only need one database writer process (DBW0), you can set up to 20 DBWn processes (DBW0 through DBW9 and DBWa through DBWj) to increase write performance for a system that alters data often. The number of DBWn processes is specified by the initialization parameter DB WRITER PROCESSES.

The goal of DBWn is to boost system performance by caching database block writes back to datafiles from the Database Buffer Cache. "Dirty blocks" are modified blocks that must be rewritten to disc. The DBWn additionally guarantees that the Database Buffer Cache has adequate free buffers to accommodate Server Processes reading data from datafiles.Because postponing publishing updated database blocks back to disc improves performance, a Server Process may discover that the data needed to meet a User Process request is already in memory.

Log Writer: The Log Writer (LGWR) writes the contents of the Redo Log Buffer to the currently active Redo Log File. Because the Redo Log Files capture database alterations based on the actual data, these are consecutive writes. the period during which the alteration takes place LGWR writes before DBWn and only certifies that a COMMIT action has succeeded after the contents of the Redo Log Buffer are successfully written to disc. LGWR can also use the DBWn to write the Database Buffer Cache's contents to disc.

The System Monitor (SMON): It is in charge of instance recovery, which is accomplished by applying entries from the online redo log files to the data files. All information in memory that hasn't been written to disc is lost if an Oracle Instance fails. When the database is restarted, SMON is in charge of recovering the instance. It performs the following functions:

• Rolls ahead to recover data from a Redo Log File that DBWn had not yet recorded to a data file. SMON examines the Redo Log Files and updates the data blocks as needed. This restores any committed transactions that were written to the Redo Log Files prior to the system failure.

• Allows system users to log in by opening the database.

• Rolls back uncommitted transactions.

SMON also manages a limited amount of space. It joins contiguous sections of free space in the database's data files for dictionary-managed table spaces. It also frees up space in the data files by de-allocating temporary segments.

The Process Monitor (PMON):It is a clean-up type of process that cleans up after failed processes such as a user connection being dropped due to a network failure or a user application program abending.

The Checkpoint (CPT): This process writes information to the database control files that specifies the point in time in the Redo Log Files where instance recovery should commence if necessary. This is done once every three seconds at the very least. Consider a checkpoint record as a place to start when it comes to healing. Prior to the checkpoint, DBWn will have finished writing all buffers from the Database Buffer Cache to disc, therefore those records will not need to be recovered. This accomplishes the following goals:

• Ensures that updated data blocks in memory are written to disc on a regular basis - CKPT can contact the DBWn process to do so, and it does so when writing a checkpoint record.

• Reduces Instance Recovery time by reducing the amount of work required for recovery by just recovering Redo Log File entries processed since the last checkpoint.

• During database shutdown, all committed data is written to data files.

Check Your Progress 4

1 When the database is restarted, _____ is in charge of recovering the instance.

(A) PMON (B) SMON (C) Log Writer (D) Background Process

2 CPT stands for _____

a) checkpoint b) check process c) check parts d) none of these

3 PMON stands for ___

a) process monitor b) processing memory c) partial monitor d) all the above

3.5 MEMORY STRUCTURE

System Global Area (SGA): All server and background processes share the System Global Area (SGA).

The System Global Area (SGA) is a storage area that holds the instance's data and control information. This information comprises both organisational data and Oracle

Server control information. The parameter SGA MAX SIZE in the parameter file determines the size of the SGA. When an Oracle instance is started up, the SGA is allocated depending on the values supplied in the initialization parameter file.

A Static SGA was utilised in earlier versions of Oracle Server. This meant that if memory management was to be tweaked, the database had to be shut down, the init. ora parameter file tweaked, and then the database had to be restarted. After Oracle 9i, a Dynamic SGA is used. Without shutting down the database instance, memory configurations for the system global area can be made.

There are several initialization options that determine the amount of random access RAM dedicated to an Oracle Instance's SGA, as follows:

SGA MAX SIZE: This sets a limits on the number of virtual memory memory allocated to the SGA – a typical setting might be 1GB; however, if the value for SGA MAX SIZE in the initialization parameter file or server parameter file is less than the sum of memory allocated for all components, either explicitly in the parameter file or by default, when the instance is initialized, the database ignores the setting for SGA MAX SIZE.

DB CACHE SIZE: In standard database blocks, this is the size of the Database Buffer Cache. The size of blocks varies depending on the operating system. The block sizes we employ are 8KB. On LINUX/UNIX and Windows operating systems, the total blocks in the cache are set to 48 MB and 52 MB, respectively.

LOG BUFFER: The Redo Log Buffer is allocated a certain number of bytes.

SHARED POOL SIZE: This is the number of bytes of shared SQL and PL/SQL memory allocated. The default size is 16 megabytes. The default size is 64 MB if the operating system is based on a 64-bit setup.

LARGE POOL SIZE: The size of the Large Pool defaults to zero because it is an optional memory object. The default size is calculated automatically if the init.ora parameter PARALLEL AUTOMATIC TUNING is set to TRUE.

JAVA POOL SIZE: This is another memory object that can be used. Memory is set to 24 MB by default.

Shared Pool

The Shared Pool is a memory structure that is shared by all system users. There are both fixed and flexible structures in it. The variable component expands and contracts in

response to the demands on memory size placed on it by system users and application applications. Library Cache and Data Dictionary Cache are included.

The option SHARED POOL SIZE in the parameter file determines how much memory is assigned to the Shared Pool. With the ALTER SYSTEM SET command, you can change the size of the shared pool on the fly. Keep in mind that the SGA MAX SIZE option determines the overall memory allotted to the SGA, and because the Shared Pool is part of the SGA, you cannot exceed the SGA's maximum size.

The most recently run SQL statements and used data definitions are stored in the Shared Pool. This is due to the fact that some system users and application applications will frequently run the identical SQL commands.

Library Cache:

When a SQL statement is parsed or a software unit is called, memory is allocated to the Library Cache. This allows the most recently used SQL and PL/SQL statements to be saved. If the Library Cache is too small, it must purge statement definitions to make room for new SQL and PL/SQL statements.The Least-Recently-Used (LRU) technique is used to manage this memory structure. When extra storage space is needed, the SQL and PL/SQL statements that are the oldest and least recently utilised are deleted.

The Library Queue is made up of two memory modules:

Shared SQL: This keeps and shares the SQL statement's execution plan and parse tree. If a system user executes an identical statement, the statement does not need to be parsed twice before being executed.

Procedures and Packages in Shared PL/SQL: This section saves and shares the most recently used PL/SQL statements, such as functions, packages, and triggers.

Data Dictionary Cache:

The Data Dictionary Cache is a memory structure that stores recently utilised data dictionary information. User account details, data file names, table descriptions, user privileges, and other data are included.

The size of the Data Dictionary Cache is managed internally by the database server, and it is determined by the size of the Shared Pool in which it is located. If the size is too small, information from the data dictionary tables on disc must be queried often, slowing performance.

Database Buffer Cache:

- The database buffers in the system global area keep track of the most recently used data blocks. The database buffer cache is made up of an instance's database buffers.
- It includes both changed and unmodified blocks.
- Data that has been utilised lately (and typically more frequently) is kept in memory, resulting in fewer I/O to and from the disc and higher performance.
- **Redo Log Buffer:**
- The system global area's redo log buffer holds the redo entries, which are a log containing database changes.
- The redo log buffer entries are written to an online redo log file, which is used when database recovery is required. It has a fixed size.
- **Large Pool:**
- The huge pool is an optional region of the SGA that provides large memory allocations for Oracle backup and restore activities, I/O server processes, and multithreaded server and oracle session memory.
- **Stream Pool:**
- The parameter STREAMS POOL SIZE is used to determine its size. To enable Oracle Streams, this pool holds data and control structures. In a distributed context, Oracle Steams manages data and event sharing.

Program Global Area (PGA): Private to each server and background processes is the Program Global Area (PGA). Each process has its own PGA.

The Process Global Area (PGA) is a chunk of memory that is allocated outside of the Oracle Instance and is also known as the Program Global Area (PGA). For a single Server Process or Background Process, the PGA holds data and control information. When a process is created, memory is allocated, and when the process is terminated, the operating system scavenges the memory. This is not a shared memory area; each process has its own PGA.

The PGA's content varies, but it usually includes:

Data for binding variables and runtime memory allocations are stored in the Private SQL Area. If more than one system user is executing the same SQL statement, the

user session has a Private SQL Area that can be linked to a Shared SQL Area. This is common in OLTP situations where multiple users are executing and using the same application programmed at the same time.

Dedicated Server Environment: The Private SQL Area is located in the Dedicated Server Environment Global Program Area

Shared Server Environment: The Private SQL Area is located in the Shared Server Environment. Global Area of the System

Session Memory: This is a type of memory that stores session variables and other information.

Software Code Area: Oracle executable files that execute as part of the Oracle instance are stored in software code areas. These code regions are static in nature and are kept distinct from other user programmes in privileged memory. When many Oracle instances run on the same server with the same software release level, the code can be shared.

Check your Progress 5

1 _____ is a storage area that holds the instance's data and control information.

(A) SGA (B) PGA (C) Shared SQL (D) All of these

3.6 SCHEMA AND SCHEMA OBJECTS

A database schema is a grouping of database objects. A database user owns a schema, which has the same name as the user. A schema consists of a set of schema objects. Schema objects are data storage structures that are logical in nature. Schema objects do not have a one-to-one relationship with the physical file on disc that contain their data.Oracle, on the other hand, maintains a schema object logically within a database table space. Each object's data is physically stored in one or more data files in the table space. You can select how much disc space Oracle allocates for some objects, like as tables, indexes, and clusters, within the table space’s data files.

Following Objects are used in user Schema:

Tables: In an Oracle database, tables are the basic unit of data storage. Rows and columns are used to hold data.

Views: A view is a customised presentation of data from one or more tables. A view is sometimes known as a virtual table because it considers the output of a query as a table.

Synonyms: It is an alias for any table, view, snapshot, sequence, procedure, function, or package is a synonym. Because a synonym is just an alias, it doesn't need to be stored.

Indexes: Tables and clusters can have indexes, which are optional structures. It's used to make the execution of SQL statements on a table go faster.

Clusters: A cluster is a collection of tables that share data blocks and are frequently used together because they have common columns.

Hash Clusters: A hash cluster is a data structure that holds related rows in the same data block. The hash value of the rows in a hash cluster is used to group them.

3.7 LET US SUM UP

In this chapter, we have discussed about oracle architecture and instance. We havealso explored memory structure of Oracle Database. We have come to know vital processes, which is executes during database execution. We have also summarized storage structures and supported files and architectures. After completion of this chapter we came to know about schemas and various schema objects.

3.8 ANSWER OF CHECK YOUR PROGRESS

Check Your Progress 1

Answer :1-b, 2-A

Check Your Progress 2

Answer :1-a

Check Your Progress 3

Answer : 1-D , 2-A

Check Your Progress 4

Answer :1-B, 2-A , 3-A

Check Your Progress 5

Answer :1-A

3.9 ASSIGNMENT

Explain Oracle Architecture.

3.10 ACTIVITIES

Explain Memory Structure of Oracle Database.

3.11 CASE STUDY

Explain Schema Object in detail.

3.12 FURTHER READING

1. Database Management Systems - Rajesh Narang -PHI Learning Pvt Ltd.
2. Database Management Systems, Raghu Ramakrishnan and Johannes Gehrke, McGraw Hill Publication.

Unit 4: Distributed Database

4

Unit Structure

4.0 LEARNING OBJECTIVES

Objectives of this unit are:

- To learn about and comprehend various distributed database architectures.
- To comprehend the database architecture of a client/server system.
- To grasp the concept of database links and users.
- To gain an understanding of the security implications of the Distributed Database Environment.
- To gain knowledge about Database Application Development in a Distributed Environment

4.1 INTRODUCTION

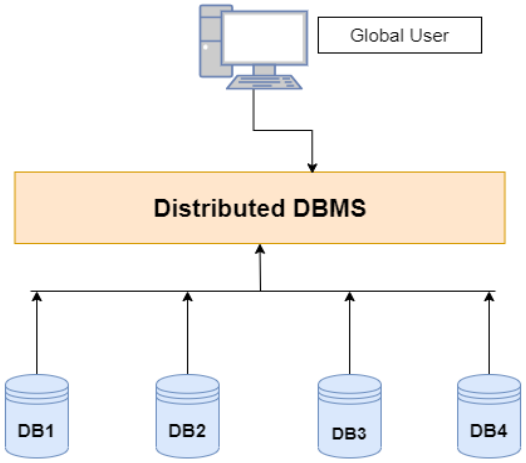
A distributed database is a database that is not restricted to a single system and is dispersed across numerous places, such as multiple computers or a network of computers. A distributed database system is made up of multiple sites with no physical components in common. This may be necessary if a database needs to be viewed by a large number of people all over the world. It must be administered in such a way that it appears to users as a single database.

A database is a logically organised collection of data. A database allows data to be easily accessed, maintained, amended, updated, controlled, and organised. Databases are divided into two categories: distributed databases and centralised databases.

The question here is why do we even need a distributed database?

Distributed databases address a variety of concerns, including availability, fault tolerance, throughput, latency, scalability, and a variety of other challenges that might develop when a single machine and database are used. That is why distributed databases are necessary. A distributed database is one that isn't confined to a single machine. It's similar to a database in that it's made up of two or more files that are stored on various computers or sites, either on the same network or on a different one.

There is no physical connection between these sites. When a specific piece of data in a database needs to be accessed by several people all over the world, distributed databases are required. It must be handled in such a way that it seems to a user to be a single database at all times.



Following are the different concepts of Distributed Database:

- Homogenous distributed database.
- Heterogeneous distributed database.

Check Your Progress 1

1 _____ Database is shared among the Network.

(A) Hierarchical (B) Distirbuted (C) Relational (D) Object

2 A _____ database system is made up of multiple sites with no physical components in common

a) Distributed b) hierarchical c) relational d) object

4.2 CLIENT SERVER ARCHITECTURE

The client-server paradigm is a distributed application structure that divides tasks or labour between servers, who supply a resource or service, and clients, who request that service. When a client computer submits a data request to the server over the internet, the server accepts the request and returns the data packets requested to the client. Clients do not share any of their assets with one another. Email, the World Wide Web, and other client-server models are examples.

How Client Server Architecture is work?

Client: Whenever we talk about a client, we're talking about people or an organisation who uses a specific service. A Client is a computer (Host) in the digital world, capable of receiving information or using a specific service from the service providers (Servers).

Servers: When we talk about servers, we're talking about a person or a medium who serves things. A server is a distant computer that offers information (data) or access to certain services in the digital world. So, its basically the Client requesting something and the Server serving it as long as its present in the database.

In a nutshell, the client-server model describes how a server provides services to clients. A server is a centralised computer that serves all connected clients. For example, file servers, web servers, and other types of servers all perform the same basic function of providing services to clients. A laptop computer, tablet, or Smartphone, for example, can be the client.

Clients have a variety of relationships with the server. Many servers have an excessive number of client relationships. Many clients were connected to one server in one too many relationships. When a client requests communication from the server. Client requests may be approved or rejected by the server. When a server computer accepts a client's request, the server maintains a connection using a protocol that has been defined. The protocol is in charge of the network.

By using this architecture structure this software is divided into three different tiers:

- **Presentation tier**
- **Logic tier**
- **Data-tier**

Presentation Tier:

This is the application's initial and highest level. This layer's primary function is to provide a user interface. A graphical user interface (GUI) is used. A graphical user interface (GUI) is a user interface that includes menus, buttons, and icons, among other things.

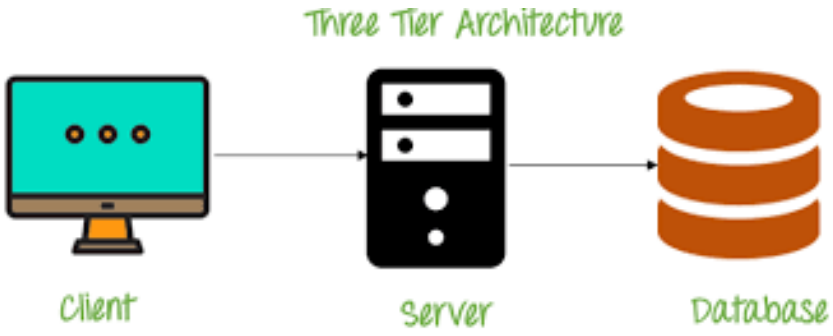
Logical Tier:

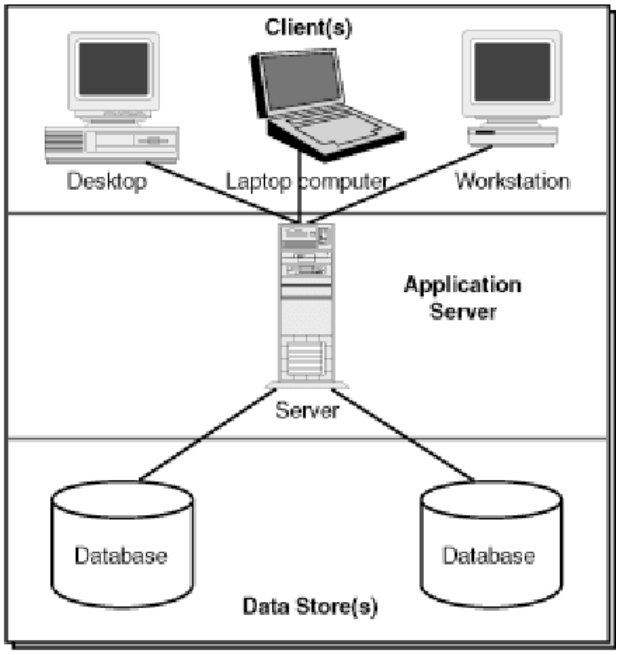
The logical tier is also known as the intermediate tier or data access tier. It is located between the display and data tiers. It essentially performs processing to govern the application's functions. The components that make up this layer are on the server, and they help with resource sharing. These components also establish business rules such

as government legal regulations, data rules, and business algorithms, all of which are intended to keep data structure consistent.

Data Tier:

The DBMS (database management system) layer is essentially what this is. Databases make up this layer. The business services layer can make advantage of it. Data is saved and retrieved in this layer. This layer's job is to ensure that data is consistent and self-contained. It also enhances scalability and performance by giving data its own tier. Data access components are found in this layer.





Check Your Progress 2

1 In Client Server architecture _____ is requesting for query.

(A) Server (B) Database (C) Client (D) All the above

2 In Client Server architecture _____ is provide GUI

(A) Presentation Tier (B) Logical Tier (C) Data tier

4.3 DISTRIBUTED DATABASE TYPES

4.3.1 Homogenous Distributed Database

A network of two or more Oracle Databases that exist on one or more systems is known as a homogeneous distributed database system. In a distributed environment, an application can access or alter data from multiple databases at the same time.

In a distributed system, you may also construct synonyms for distant objects so that users can access them using the same syntax as local objects. A distributed system can appear to have native data access in this fashion. Users of mfg do not need to be aware that the information they are accessing is stored in remote databases.

Oracle Database distributed database systems can use Oracle Databases from various versions. A distributed database system can use any supported release of Oracle Database. Nonetheless, programmes that interact with the distributed database must be aware of the capabilities of each node in the system. An Oracle7 database cannot be expected to understand SQL extensions that are only accessible with Oracle Database.

Distributed Database &Distributed Processing

Distributed Database:

A distributed database is a collection of databases that can seem to applications as a single data source in a distributed system.

Distributed Processing:

The operation that occurs when an application distributes its responsibilities among different machines in a network is known as distributed processing. A database application, for example, often distributes front-end presentation chores to client PCs

while allowing a back-end database server to manage shared database access. As a result, a client/server database application system is more typically used to describe a distributed database application processing system.

4.3.2 Heterogeneous Distributed Database

At least one of the databases in a heterogeneous distributed database system is not an Oracle database. The heterogeneous distributed database system appears to the application as a single, local Oracle Database. The data dispersion and heterogeneity are hidden by the local Oracle Database server.

A non-Oracle Database system is at least one of the databases in a heterogeneous distributed database system. The heterogeneous distributed database system appears to the application as a single, local Oracle database. The local Oracle Database server hides the data's heterogeneity and distribution. If you want to use a Sybase database in an Oracle Database distributed system, for example, you'll need to get a Sybase-specific transparent gateway so the Oracle Database can connect with it.If the non-Oracle Database system supports the ODBC or OLE DB protocols, you can utilise generic connectivity to access non-Oracle Database data stores.

Heterogeneous Services

Heterogeneous Operations are an Oracle Database server component that serves as the foundation for the current package of Oracle Transparent Gateway products. For Oracle Database gateway solutions and other heterogeneous access facilities, HS provides a standard design and administrative mechanisms. It also provides upwardly suitable functionality for Oracle Transparent Gateway users who have upgraded from earlier versions.

Transparent Gateway Agents

Heterogeneous Services can employ a transparent gateway agent to interface with each non-Oracle Database system that you access. Each type of system requires a different agent because the agent is particular to non-Oracle Database systems. The transparent gateway agent uses the Oracle Database server's Heterogeneous Services component to facilitate communication between Oracle Database and non-Oracle Database systems. On behalf of the Oracle Database server, the agent executes SQL and transactional queries on non-Oracle Database systems.

Generic Connectivity

You can use a Heterogeneous Services ODBC agent or a Heterogeneous Services OLE DB agent to connect to non-Oracle Database data stores with generic connectivity. Both are supplied as a basic feature with your Oracle product. A generic connectivity agent can connect to any data source that follows the ODBC or OLE DB standards. Generic connection offers the advantage of not necessitating the purchase and configuration of a separate system-specific agent. An ODBC or OLE DB driver that can communicate with the agent is used. Some data access functionalities, on the other hand, are only available through transparent gateway agents.

Check Your Progress 3

1 A network of two or more Oracle Databases that exist on one or more systems is known as a _____database system.

(A) Heterogeneous Distributed (B) Homogeneous Distributed (C) Both A and B

1 In a _____ database, different sites have different operating systems, DBMS products and data models.

(A) Heterogeneous Distributed (B) Homogeneous Distributed (C) Both A and B

4.4 DATABASE BACKUP AND RECOVERY

A backup is a replica of data that is similar to the original. Important portions of a database, including as the control file, redo logs, and data files, can be included in this copy. By offering a mechanism to restore original data, a backup protects data from programmed error and functions as a precaution against unexpected data loss.

Physical and logical backups are the two types of backups. Backups of physical database files are called physical backups. The term "backup and recovery" usually refers to the process of copying files from one location to another and performing various operations on them.

Logical backups, on the other hand, comprise data that has been exported using SQL instructions and saved in a binary file. In redo log buffers, Oracle captures both committed and uncommitted changes. Physical backups are supplemented with logical backups.The term "restore" refers to the process of rebuilding a physical backup and making it available to the Oracle server. Data is

updated using redo records from the transaction log to recover a restored backup. Changes to the database after the backup are recorded in the transaction log.

4.4.1 Logical Database Backup

The Oracle tool Import/Export is used to do Logical Database Operations, which allows us to perform data object exports and imports, as well as data transfer between databases on different hardware platforms and Oracle versions. The export (exp) and import (imp) programmes are used to backup and restore logical databases. Database objects are dumped to a binary file during the export process, which can then be imported into another Oracle database.

Users can choose between the old imp/exp utilities and the newly introduced Data pump utilities, expdp and impdp, starting with Oracle 10g. These new utilities add much-needed performance enhancements, network-based exports and imports, and other features.

There are several parameters that can be used to control which objects are exported or imported. Run the exp or imp utilities with the help=yes parameter to get a list of possible parameters.

Export and Import Functions are used for following tasks:

- Recovery and backup (Only for tiny databases)
- Data can be moved between Oracle databases on various platforms.
- Data reorganisation and database fragmentation elimination (export, drop and reimport tables)
- Upgrade databases from Oracle versions that are more than ten years old.
- Find out if your database is corrupt. Make certain that all of the data can be read.
- Table spaces are moved across databases.

Different Export/Import Utility Methods:

Full Export: To do a full export, you'll need the EXP FULL DATABASE and IMP FULL DATABASE variables. For a complete export, use the full export parameter.

Table space: To export a table space, use the table spaces export argument.

User: This mode allows you to export and import all of a user's objects. For a user (owner) export-import, use the owner export parameter and the from user import argument.

Table: Table export mode allows you to export/import certain tables (and partitions). For a table export, use the tables export argument.

4.4.2 Physical Database Backup

Backups can be mixed and matched in a number of ways. For example, we can backup the entire database once a week to ensure a relatively current copy of the original data, but only the most frequently visited table spaces once a day. As an added protection, the DBA can multiplex the critical control file and archived redo log.

Online Database Backup:

An online database backup, also known as an open backup, is a backup in which all read-write data files and control files are not check pointed to the same SCN. If you need your database to be available 24 hours a day, seven days a week, you'll have no choice but to do online backups of the entire database when it's in ARCHIVELOG mode.

Offline Database Backup:

All data files and control files in this backup are consistent with respect to the same SCN at the same point in time. Because the data is already consistent, the user can open the set of files created by the backup without having to apply redo logs. This type of backup can only be done by shutting down the database completely and performing the backup while it is closed.

A whole database backup: It contains the control file as well as all database files that belong to a database, making it the most frequent sort of backup. When using ARCHIVELOG mode, the DBA can back up different areas of the database over time, resulting in a piecemeal backup of the entire database.

Table space Backups: A table space backup is a portion of the database that has been backed up. Backups of table spaces are only valid if the database is in ARCHIVELOG mode. Only when a table space is read-only or offline-normal is a table space backup valid for a database running in NOARCHIVELOG mode.

Data file Backup: Backups of a single data file are known as data file backups. Data file backups are less popular than table space backups, and they can only be used if the database is running in ARCHIVELOG mode. If a data file is the sole file in a table space, a data file backup is valid for a database running in NOARCHIVELOG mode.

Control File Backups: A control file backup is a copy of the control file in a database. If a database is open, the following SQL command can be used to produce a legal backup: CHANGE THE DATABASE BACKUP CONTROLFILE TO 'location', or utilise Recovery Manager (RMAN).

Archived Redo Logs Backups: Successful media recovery requires backups of Archived Redo Logs. You want to preserve as many days of archive logs on disc as possible, depending on the disc space available and the amount of transactions done on the database, and you want to back them up regularly to ensure a more comprehensive recovery.

Configuration Files: When it comes to Oracle Database backups, we have two options. spfile or init.ora, password file, tnsnames.ora, and sqlnet.ora are examples of configuration files. Because these files do not change frequently, a less frequent backup schedule is required. A configuration file can be manually regenerated if it is lost. When time is of the essence, restoring a backup of the configuration file is preferable to manually creating a file in a certain format.

4.4.3 Recovery

Basic recovery consists of two steps: restoring a physical backup and then updating it with database changes since the last backup. The most crucial part of data recovery is ensuring that all data files are consistent with respect to the same time period. Oracle has integrity checks in place that restrict the user from accessing the database until all of the data files are in sync.

Recovery Process:

Oracle applies redo data to data blocks in a sequential manner in all types of recovery. Oracle determines if recovery is required based on information in the control file and data file headers. Rolling forward and rolling back are two components of the recovery process. Oracle applies redo records to the corresponding data blocks as it rolls forward. Oracle looks through the redo log step by step to identify which modifications need to be applied to which blocks, and then makes the necessary adjustments. For example, if a user adds a row to a table but the server fails before saving the change to disc, Oracle can update the data block to reflect the new row using the redo record for this transaction.

The Oracle database can be opened once Oracle has completed the rolling forward stage. After the database has been opened, the rollback step begins. The information about rollbacks is kept in transaction tables. Oracle looks for uncommitted transactions in the table and undoes any it finds. For example, if the user never committed the SQL statement that added the row, Oracle will find out and undo the modification in a transaction table.

Responding to the Loss of a Subset of the Current Control Files

In the event that a subset of the current control files is lost, what should you do? If a permanent media failure has damaged one or more control files in a database and at least one current

control file has not been harmed by the media failure, use the techniques below to recover the database.

Copying a Multiplexed Control File to a Default Location

You can simply copy one of the intact control files to the location of the missing control file if the disc and file system containing the lost control file are intact. You do not need to change the CONTROL FILES initialization parameter in this situation.

Shut down the instance if it is still running:

SQL>SHUTDOWN ABORT

Resolve the hardware issue that resulted in the media failure. If you can't fix the hardware issue right away, you can try database recovery by restoring broken control files to a different storage device. Copy over the damaged control files with an intact multiplexed copy of the database's current control file. Create a new instance of the database and mount and open it.

SQL>STARTUP

Identifying Which Data files Need to Be Recovered:

To decide which files to restore in preparation for media recovery, use the dynamic performance view V$RECOVER FILE. This view displays a list of all files that require recovery as well as an explanation of why they must be retrieved.

The following query shows the file ID numbers of datafiles that require media recovery, as well as the reason for recovery (if known), and the SCN and time when recovery should start:

SELECT * FROM V$RECOVER_FILE;

Query V$DATAFILE and V$TABLESPACE to obtain filenames and table space names
for data files requiring recovery.

Restore the Data file:

If a database's datafiles are permanently damaged due to a media failure, you must first restore backups of these datafiles before you can recover the damaged files. If you can't restore a damaged datafile to its original position (for example, if you need to repair a disc and restore the files to a different disc), you'll need to tell the control file where the files are now.

Check Your Progress 4

1 In _____you don't take the copies of any physical things,you only extract the data from the data files into dump files

(A) Physical Backup (B) Logical Backup (C) Both of these.

2 The most crucial part of data _____ is ensuring that all data files are consistent with respect to the same time period

a) recovery b) backup c) both a and b d) none of these

3 RMAN stands for

a) recovery manager b) restore manager c) redesign manager d) none of these

4) Physical and logical are the two types of ____

a) backups b) recovery c) both a and b d) none of these

4.5 ORACLE UTILITY EXPORT IMPORT

Export Utility:

This utility can be used to move data items from one Oracle database to another. The objects and data in an Oracle database can be transferred to another Oracle database that is running on different hardware and software.

The database definitions and actual data are copied into an operating system file via the export program (export file). The export file is an Oracle binary-format dump file (with.dmp extension), which is often saved to disc or tape. Before exporting, make sure there is adequate space on the disc or tape that will be utilized.

General the following parameters are used with the exp command:

- Complete: To choose full export mode, use this argument.
- Table spaces: This parameter specifies the export mode for table spaces.
- Owner: This field specifies the export mode for the user.
- Tables: This parameter is used to specify the export mode for tables.
- Query: Uses a where clause to limit the rows exported. Only the table export mode can use the query parameter. It must be applicable to all exported tables for obvious reasons.
- Par file: A par file is specified. A parameter file is a plain text file that may be created using any text editor.

Full, Owner, and Table are the three fundamental types of exports. For all schemas, a full export exports all objects, structures, and data in the database. Only the objects

owned by a certain user account are exported when using the owner export. Table export only exports tables that belong to a single user account.

We want to export the Student table from scott/tiger (username and password, respectively) users, and the exported data will be saved in a dump file named Student.

file=Student.dmp tables= EXP scott/tiger (Student)

We want to export the Student table from the scott/tiger (username and password) users, and the data will be saved in a dump file called Student in interactive mode.

Import Utility

Without any parameters, the IMP command can be used to invoke the import utility interactively. The import utility extracts items from an export dump file created by the export utility. With the exp command, we can use several parameter files at once. The following are some of the Import Utility's parameters:

Buffer: the size in bytes of the buffer through which data rows are transferred is specified for buffer.

Commit: Controls whether Import commits after each array insert. Import commits only after each table has been loaded, and it conducts a rollback if an error occurs before moving on to the next object.

Constraints: Indicates whether or not table constraints are to be imported. Constraints are imported by default. You must set the parameter value to n if you do not want constraints to be imported.

File: Specifies the names of the importable export files. Because Export supports multiple export files, you may need to specify multiple filenames to be imported. The default extension is.dmp.

Fromuser: This argument allows you to import a subset of schemas from a multi-schema export file.

Full: Indicates whether the whole export dump file should be imported.

Grants: Indicates whether or not object grants should be imported.

Parfile: Defines the name of a file containing a list of Import parameters. See Parameter Files for more information on how to use a parameter file.

Rows: Indicates whether or not to import the table data rows.

Tables: Indicates that the import is in table mode and provides a list of the tables, partitions, and sub partitions to import. You can import complete partitioned or non-partitioned tables using table-mode import.

Touser: This parameter specifies a list of user names whose schemas will be imported. Before the import operation, the user names must exist; otherwise, an error will be returned. To use this argument, you must have the IMP FULL DATABASE role. TOUSER.

UserId: Specifies the user's username/password (and optional connect string) for importing to a different schema from the one that originally held the item.

Check Your Progress 5

1 In _____ file an Oracle binary-format dump file (with.dmp extension), which is often saved to disc.

(A) Import (B) Export (C) Recovery (D) None of these

2 The _____ utility extracts items from an export dump file created by the export utility.

(A) Import (B) Export (C) Recovery (D) None of these

4.6 LET US SUM UP

We talked about oracle architecture and instances in this chapter. We've also looked at Oracle Database's memory structure. We've learned about important processes that take place during database operation. We've also compiled a list of storage structures, as well as the files and architectures that they support. We learned about schemas and numerous schema objects after finishing this chapter.

We explored numerous types of database backup strategies, such as logical backup and physical backup, in this chapter. Under what circumstances should we perform a logical backup? We also learned about the various parameters for Oracle's Import/Export function. We also have several sorts of physical backups, such as hot and cold backups, and we aim to cover all aspects of both types of backups and recovery procedures.

4.7 ANSWER FOR CHECK YOUR PROGRESS

Check Your Progress 1

Answers: 1-B, 2-A

Check Your Progress 2

Answers: 1-C , 2-A

Check Your Progress 3

Answers: 1-B , 2-A

Check Your Progress 4

Answers: 1-B, 2-A 3-A 4-A

Check Your Progress 5

Answers: 1-B , 2-A

4.8 ASSIGNMENT

Explain Distributed Database with its Types.

4.9 ACTIVITIES

Design a Client Server Architecture and explain in detail.

4.10 CASE STUDY

Explain Oracle Export and Import Utility.

4.11 FURTHER READING

1. Oracle Database 11g: Backup and Recovery User's Guide, Lance Ashdown, Oracle Press.

2. Expert Oracle Database Architecture, Third Edition, Darl Kuhn & Thomas Kyte,Apress Publishing.

3. Oracle Database 10g The Complete Reference, Kevin Loney, Oracle Press.

BLOCK 3: ORACLE

Block Introduction

In this block, we will learn about basic concepts of database and database management system. It provides an overview of the above topic with the help of various examples.

The aim of this block is to enable the reader to understand the basic concepts of this topic. The writer has tried his best to detail the concepts; apart from this, the sub-topics have even been discussed by him in detail. On the other hand, he has even discussed the database design in detail with sufficient examples.

The objective of this block includes detailing the student‘s about the various above mentioned topics in very detail with the help of sufficient and suitable examples.

Block Objectives

After learning this block, you will be able to understand:

- Basic Data Types of ORACLE,
- Inbuilt Functions, queries, Sub queries, Join,
- Indexes, Views, Sequences, and Synonyms.\
- ORACLE Database Object : Stored Procedures & Functions,
- Packages, Triggers, Users – Create & Delete User, Grant & Revoke Command
- ORACLE Database Privileges & Roles

Unit 1: PLSQL – Oracle Basics

1

Unit Structure

1.0 LEARNING OBJECTIVES

After learning this unit, you will be able to understand:

- Database concepts
- Architecture of database
- Tables, rows, and columns
- Process of designing a database

1.1 BASIC DATA TYPES IN ORACLE

Every field in the database has a specific domain of values like name will contain characters, pin code will contain numbers, so to store such values, different data types are created. These data types will be useful while creation of new tables. Each data type will have specific range of values that it can store. These data types and the range of values are described here

Data Types	**Description**
Char (N)	Fixed Length Character Data. Maximum size is 2000 bytes. Default or Minimum Size 1 Byte.
Varchar (N)	Variable Length Character Data. Maximum up to 2000 characters.
Varchar2 (N)	Variable Length Character Data. Maximum up to 4000 characters
Nvarchar2 (N)	Variable-length Unicode character string having maximum size is determined by the national character set definition, with an upper limit of 4000 bytes
Number (P,S)	Numeric data type for integers and Real Numbers. P = Overall number of Digits. Maximum values 38. S =

	Number of digits to the right of the decimal point.
FLOAT (p)	A subtype of the NUMBER data type. A FLOAT value requires from 1 to 22 bytes
LONG	Variable Length Character Data (Up to 2GB)
Date	Date data type for storing date and time. The size is fixed at 7 bytes.
BINARY_FLOAT	32-bit floating point number.
CLOB	Character Data (Up to 4GB)
NCLOB	Character Data containing Unicode characters. (Up to 4GB)
BLOB	Binary Data (Up to 4GB)
ROWID	A base-64 number system representing the unique address of a raw in its table.

1.2 DATA DEFINITION LANGUAGE (DDL)

In this topic, we are going to learn about the data definition language, it describes how the database structure can be created or modified.

DDL contains SQL statements for insert, query, update and delete query, schema creation and modification and data access control. DDL contains the following

CREATE

Create New Objects in Database like Table, View Index, etc.

Example

Create table student (studentid number(5), studentname varchar(20));

ALTER

Modify the existing database object.

Example

Alter table student

Add studentaddress varchar(20);

DROP Destroying an existing object.

Example

Drop table student;

RENAME Change the name of existing object.

Example

ALTER TABLE student RENAME TO student1;

In versions after 10g you can use

RENAME student to student1;

TRUNCATE Deleting an existing object. (Drop and Re-Create)

TRUNCATE table student;

COMMENT Provides Single line or multi line comment(s).

1.3 DATA MANIPULATION LANGUAGE (DML)

Data Manipulation language includes the sql statement for creation or modification of the contents of the table. It includes sql statements like:

SELECT

Retrieve specific or all records from one or more tables or views.

Example

Select * from employee;

Select empname from employee;

INSERT

Create new record into the table by specifying the values in corresponding fields

Example:

Employee table contains (empid, empname)

insert into employee values (1,'baouemp1');

UPDATE

Modify existing records in the table.

Update employee

Set empname = 'baouemp2'

Where empid = 2;

DELETE

Delete existing all records or specific records using a condition

Delete from employee;

Delete from employee where empid = 2;

Check your progress 1

1. The database schema is written in

a. HLL

b. DML

c. DDL

d. DCL

2. A logical schema

a. is the entire database.

b. is a standard way of organizing information into accessible parts.

c. describes how data is actually stored on disk.

d. both a. and c.

3. One of the advantages of database management is

a. data depends on programs.

b. data dismissal increases.

c. data is incorporated in addition to admittance by multiple programs.

d. none of the above.

4. Fix length character data type is supported by __Data type.

a. char

b. varchar

c. varchar2

d. int

5. Insert, update, delete are ___ statements

a. DML

b. DDL

c. Any of the above

d. None of the above

1.4 TRANSACTION PROCESSING LANGUAGE (TPL)

Transaction means a set of sql statements which should be either executed successfully or all must be aborted which is also called atomicity of transactions.

TPL statements are very useful in maintaining the atomicity of the transactions, which means that if one of the sql statement fails to execute other sql statements should be roll-backed and if the transaction is successful then the data is committed to the permanent storage.

COMMIT

Make permanent all changes performed in the transaction.

ROLLBACK

Undo all uncommitted works done by the transaction(s).

SAVEPOINT

Identify a point in a transaction to which you can later rollback.

1.5 DATA CONSTRAINTS

Constraints are the rules enforced on data columns on table. These are used to limit the types of data that can go into the table. Constraint could be applied at **column Level** or **table level**.

Types of Constraints

1) **I/O Constraints** -Primary key constraint , Referential / Foreign key constraint
2) **Business Rule Constraints** – Not NULL, Unique, Check constraint and Default value constraint.

Let us study them in detail:

I/O Constraints:

Primary Key: The rate at which data may be added or removed from a table is determined by this data constraint.

A primary key is a field in a database table that uniquely identifies each row (or record). The primary key field must be required, which implies that it cannot include null values and must have unique values. A table can only have one primary key, which can be made up of one or more fields. Single Field Primary Key refers to a primary key established on a single field, whereas Composite Primary Key refers to a primary key created on more than one fields.

Primary key can be implemented at **columnlevel**as :

```
Create table student
( student_id number(3) primary key,
```

```
sem_idnumber(3),
student_namevarchar(20)
);
```

Primary key can be implemented at **table level**as :

```
Create table student
( student_id number(3),
sem_idnumber(3),
student_namevarchar(20),
  primary key (student_id)
);
```

Composite key can be implemented at table level as:

```
Create table student
( student_id number(3),
sem_idnumber(3),
student_namevarchar(20),
  primary key (student_id, sem_id)
);
```

Referential / Foreign Key Constraint

A foreign key also known as a reference key is a column or set of columns whose values match those of a primary key in another table.

Foreign key column can contain null values, duplicate values but it can only contain the values which are present in its referred primary key column

Example:

```
Create table student
```

(student_id number(3) primary key,
sem_idnumber(3) **references**semester_master,
student_namevarchar(20)
);

Semester_Master

Sem_id	Semester
1	SEM_1_BSCIT
2	SEM_2_BSCIT

Student

Student_id	Sem_id	Student_name
1	1	Maulik
2	1	Akshay
3	2	Sanjay
4		Manish

The sem_id can contain duplicate values, null values but it cannot contain values which are not present in reffered primary key sem_id in semester_master, that means it cannot contain sem_id as 3 or 4 etc.. as sem_id in semester_master contains only 1 and 2 as sem_id values.

Check your progress 2

1. Commit and Rollback are ___ statements.
    a. TPL
    b. DML
    c. DDL
    d. DCL
2. ___Identify a point in a transaction to which you can later rollback
    a. Save point

b. Commit.

c. RollBack.

d. None of the above

3. Which is not a sql constraint

a. check

b. primary key

c. foriegn key

d. alternate key

4. I/O constrains are / is___

a. Primary Key

b. Foreign Key

c. None of the above

d. Both a and b

5. ___ key is also known as Reference key.

a. Primary

b. Foreign

c. Unique

d. None of the above

1.6 IN-BUILT FUNCTIONS

There are many inbuilt functions of oracle which are classified as follows

Aggregate Functions: These type of functions are returning only one value after execution of the query.

SUM (Values|Column)	Returns Sum of given Values.
AVG (Values|Column)	Return the Average Value.
COUNT (Values|Column)	Return Number of rows where the value of the column is not NULL
COUNT (*)	Return Number of rows including duplicates and NULLs
MAX (Values|Column)	Returns Maximum Value.

MIN (Values|Column)	Returns Minimum Value.
MEDIAN (Values|Column)	Returns Median (Middle value in the sorted column, interpolating if necessary
STDDEV (Values|Column)	Returns Standard deviation of the column ignoring NULL values
VARIANCE (Values|Column)	Returns Variance of the column ignoring NULL values
CORR (Column-1,Column-2)	Returns Correlation coefficient between the two columns after eliminating nulls.

Example:
Write a query to find the maximum salary of the employees

Select max(empsalary) from employee;

Numeric Functions: These are functions that take a number as input and return a number.

ABS (m)	Absolute value of m
MOD (m, n)	Remainder of m divided by n
POWER (m, n)	m raised to the nth power
ROUND (m , n)	m rounded to the nth decimal place
TRUNC (m, n)	m truncated to the nth decimal place
CEIL (n)	Smallest integer greater than or equal to n
FLOOR (n)	Greatest integer smaller than or equal to n

SQRT (n)	Positive square root of n

String Functions: These are functions that accept character input and can return both character and number values

LPAD (str1, n , str2)	Returns str1 right justified and padded left with n characters from str2
RPAD (str1, n, str2)	Returns str1 left justified and padded right with n characters from str2
LTRIM (str,set)	Returns str with characters removed up to the firstcharacter not in set
RTRIM (str, set)	Returns str with final characters removed after thelast character not in set
REPLACE (str, find_str, replace_str)	Returns s with every occurrence of find_str in str replaced by replace_str
SUBSTR (str, m, n)	Returns a substring from str, beginning in positionm and n characters long; default returns to endof str.
LENGTH (str)	Returns the number of characters in str.
INSTR (str1, str2, m, n)	Returns the position of the nth occurrence

	of str2
	in str1, beginning at position m, both m and n

Date Functions

SYSDATE	Current date
LAST_DAY (Date)	Date of the last day of the month containing date
NEXT_DAY (Date, day)	Date of the first day of the week after date
ADD_MONTHS (Date, No. of Month)	Add No. of Months in Date
MONTHS_BETWEEN (Date-1, Date-2)	Returns Difference in Month between two dates.

Conversion Functions

TO_NUMBER (String, Format)	Character String converted to a Number as Specified by Format.
TO_CHAR(Value, Format)	Convert Number or Date into Character string as specified by Format.
TO_DATE (String, Format)	String Value converted in a Date as specified by given Format.
ROUND (Date, Format)	Date Rounded as specified by the Format.

TRUNC (Date, Format)	Date truncated as Specified by the Format.

1.7 QUERIES

The queries can be classified into three main parts

1) Select query
2) Update query
3) Delete query

Select query

The select query is used to retrieve the data in required format from the stored database

using where conditions and various operators

select * from table_name;

it retrieves all rows and all columns from the table

select column_name, column_name from table_name;

it retrieves specific columns from the table

select * from table_name where condition;

it retrieves specific rows which satisfies the condition

SQL>**SELECT * FROM student;**

OUTPUT:

```
LASTNAME       FIRSTNAME       AREACODE PHONE    ST ZIP
-------------- --------------- --------- -------- -- -----
BHATT          AL                  100 555-1111 GJ 22333
MEHTA          AC                  200 555-2222 SU
PANDYA         BH                  300 555-6666 CO 80212
PATEL          JD                  381 555-6767 AM 23456
SHAH           FG                345 555-3223 VD 23332
```

SQL>**SELECT * FROM FRIENDS WHERE FIRSTNAME = 'JD';**

```
LASTNAME       FIRSTNAME       AREACODE PHONE    ST ZIP
-------------- -------------- --------- -------- -- -----
PATEL          JD                 381 555-6767 AM 23456
```

Update Query

Update queries are used to modify the existing contents of the table using a specific condition. If the condition is not applied all rows are updated

Update employee

Set salary = salary + salary * 0.2

Where empid =1;

Here the salary of employee with id =1 will be updated by 20% of the existing salary.

Delete query

The existing contents of the table can be deleted using specific conditions. If the condition is not specified all rows are deleted

Delete from employee where empid = 1;

Here the row with empid =1 will be deleted

Delete from employee;

Here all rows are deleted

Check your progress 3

1. Aggregate function always returns ___ value(s).

    a. only one

    b. more than one

    c. two

    d. depends on query

2. Which aggregate function is used to find the average of the values

a. AVG

b. Average.

c. AVGE.

d. AVER

3. Which of the following is not an aggregate function.

a. sum

b. min

c. max

d. join

4. ___ query is used to modify the existing contents of table.

a. Update.

b. Insert

c. Delete

d. None of the above

5. ____ function returns middle value of stored column.

a. MEDIAN

b. MAX

c. MIN

d. None of the above

1.8 SUB-QUERIES

A sub query, also known as an inner query or nested query, is a query that is contained within another query.

Parentheses must be used to surround sub queries. A sub query is used to return data that will be utilised as a condition in the main query to further limit the data that may be retrieved. Sub queries can be used with the operators and the SELECT, INSERT, UPDATE, and DELETE statements.

The outer query is called the parent query and the inner query is known as child query.

The child query executes first and its output is given to the parent query for execution

• A sub query can have only one column in the SELECT clause, unless multiple columns are in the main query for the sub query to compare its selected columns.

• An ORDER BY cannot be used in a sub query, although the main query can use an ORDER BY. The GROUP BY can be used to perform the same function as the ORDER BY in a sub query.

• Sub queries that return more than one row can only be used with multiple values Operators, such as the IN operator.

• The BETWEEN operator cannot be used with a sub query; however, the BETWEEN operator can be used within the sub query.

Example

Select * from order_master where customer_id IN (select customer_id from customer_master);

1.9 JOIN

Joins are used to retrieve data from two or more tables having a common attribute having a primary key – foreign key relationship

According to the usage of joins they are classified as follows

1) Self join

You may need to attach a table to itself just on occasion. The join is known as a Self Join when a table is attached to itself. To avoid column ambiguity, make sure that the join statement specifies as alias for both versions of the table.

2) Left outer join

A LEFT OUTER JOIN adds back all the rows that are dropped from the first (left)table in the join condition, and output columns from the second (right) table are set toNULL.

3) Right outer join

A RIGHT OUTER JOIN adds back all the rows that are dropped from the second(right) table in the join condition, and output columns from the first (left) table are setto NULL

4) Equi join

The two or more tables having the common attribute can be combined using an condition having (=) equal to sign.

Equijoin, linkstables on the basis of an equality condition thatcompares specified columns of each table. Theoutcome of the equijoin does not eliminate

duplicate columns, and the condition or criterionused to join the tables must be explicitlydefined. The equijoin takes its name from the equality comparison operator (=) used in the condition.

5) Theta join

It is similar to equi join but it uses comparision operator other than (=) equal to sign like > , <, >=, <=

1.10 INDEXES

Indexes are special lookup tables that may be used by the database search engine to speed up data retrieval, searching and sorting of the database. The usage of an index speeds up SELECT searches and WHERE clauses

The CREATE INDEX statement is used to create an index. It allows you to name the index, provide the table and the column(s) to index, and define whether the index should be in ascending or descending order.

Indexes can also be unique, in the sense that they prevent duplicate entries in the column or combination of columns on which they're based.

• An index is a pointer to the location of data in a table.

• An index can be applied to one or more columns of a table.

• Sequential scan is done on tables if indexing is not done.

• If indexing is done sequential scan is replaced by access of index entry
• Index entry is combination of key value and ROWID

Need for Indexing

• Reduce the access time of data fetched by a query.
• Table without index is scanned sequentially which is time consuming
• A query processor searches for indexes associated with the data being accessed.
• The ROWID is used directly to go to a specific location where data is contained.
• If more than one indexes are created on a table then insert update and delete operations performance would degrade.
• Oracle manages the changes in indexes after it is created.
• There is no change in syntax of SQL statements.
• Indexes should be created after inserting data into the table on which you want to create the index.
• If index is created before inserting the data, oracle updates the index entries every time when data is inserted, deleted or updated.
• It consumes extra resources and system time.
• Therefore it is desirable to create index after data is inserted in the table.

Syntax

Creating an index

```
CREATE INDEX INDEX_NAME ON TABLE_NAME;
```

Deleting an index

```
DROP INDEX INDEX_NAME;
```

1.11 VIEWS

A view is a virtual table based on a SELECT query. The query can contain columns, computed columns, aliases, and aggregate functions from one or more tables. The tables on which the view is based are called base tables.

There are numerous distinct aspects of a relational view:

• In a SQL query, you can use the name of a view anywhere a table name is required.

• Views are updated in real time. That is, each time the view is invoked, it is re-created on the go.

• Views provide a level of security in the database because the view can restrict users to only specified columns and specified rows in a table.

Creating a view

Create view view_name as select_query;

Example:

Create view view_products as select product_id, product_name from Product_master;

Deleting a view

Drop view view_products;

1.12 SEQUECES

Sequence is an Oracle object that is used to produce unique numbers and may be used to automatically construct primary keys. By picking the most often generated value and incrementing it, a new primary key value may be created. Serialization necessitates a lock during the transaction and forces other users to wait for the next value of the primary key. Users must first gain CREATE SEQUENCE system rights before they may create a sequence.

Syntax:

CREATE SEQUENCE SEQUENCE_NAME
STARTWITH INITIAL_VALUE
INCREMENT BY INCREMENT_VALUE
MAXVALUE MAXIMUM_VALUE
CYCLE |NOCYCLE
CACHE | NOCACHE;

Example :

CREATE SEQUENCE SEQ_PRODUCTID
STARTWITH 1
INCREMENT BY1
MAXVALUE 9999
NOCYCLE;

To insert Sequence Value in Product_Id of Product table, query will be
INSERTINTO PRODUCT VALUES (SEQ_PRODUCTID.nextval, ‘iphone13’);

Product Table contains Product_id and Product_name

Check your progress 4

1. Sub query is also known as ___.
a. inner query
b. nested query
c. None of the above
d. Both of the above
2. Sequence generate ___numbers.
a. unique
b. duplicate.
c. Both of the above.
d. None of the above
3.__ is virtual table.
a. view
b. sequence
c. both of the above
d. None of the above
4. To increase data retrieval ___ is used..
    a. Index.
    b. sequence
    c. view

d. None of the above

5. To join table itself ___ join is used.

a. self

b. left outer

c. right outer

d. None of the above

1.13 SEQUECES

Any table, view, sequence, procedure, function, or package can have a synonym. Because a synonym is only an alias, it is not stored in the database separately.

A synonym indicates the use of different names to describe the same attribute. For example, car and auto refer to the same object

1.14 LET US SUM UP

This unit gives unique learning on various important concepts of basic datatypes in oracle , data definition language (ddl) data manipulation language (dml), transaction processing language (tpl), data constraints, inbuilt functions, queries, sub-queries, joins, indexes, views, sequences and synonyms. This unit provides basic understanding of these most important concepts which can be useful for all types of software applications.

1.15 ANSWERS FOR CHECK YOUR PROGRESS

Check your progress -1
Answers :(1-C) , (2- a), (3- C), (4 -a),(5- a)

Check your progress -2
Answers :(1-a) , (2- a), (3- d), (4 -d),(5- b)

Check your progress -3
Answers :(1-a) , (2- b), (3- d), (4 -a),(5-a)

Check your progress -4
Answers :(1-d) , (2- a), (3- a), (4 -a),(5-a)

1.16 GLOSSARY

1. DDL - Data Definition Language
2. DML–Data Manipulation Language
3. TPL – Transaction Processing Language

1.17 ASSIGNMENT

1. Explain various data types used for creating a database
2. Explain various sql functions used in sql queries

1.18 ACTIVITIES

Write a note on DML, DDL and TPL and explain the difference between them

1.19 CASE STUDY

Explain the benefits of creating index, join and sequences.

1.20 FURTHER READING

1. Database Management Systems - Rajesh Narang -PHI Learning Pvt Ltd.

2. Database System Concepts bySilberschatz, Korth -Tata McGraw-Hill Publication.

3. An Introduction to Database Systems - Bipin Desai- GalgotiaPublication.

4. Database Management System by Raghu Ramkrishnan- Tata McGraw-Hill Publication.

5. SQL, PL/SQL: The Programming Language Oracle - Ivan Bayross- BPB Publication.

Unit 2: Oracle Database Objects

2

Unit Structure

2.0 LEARNING OBJECTIVES

After learning this unit, you will be able to understand:

- Stored Procedures
- The create procedure command and Delete or replacing a stored procedure
- Functions

2.1 ORACLE DATABASE OBJECTS

A database object is basically any object created by end users; for example, tables, views, indexes, stored procedures, functions and triggers.

A logical grouping of database objects that are related to each other is called schema. Usually, a schema belongs to a single user or application.

Important aspect of managing a database is monitoring the database objects that were created in the database.

A named object of PL/SQL block is a procedure or function. Procedures and Functions are the two types of subprograms in PL/SQL. Every subprogram will have a declaration block, an execution block (or body), and an optional exception handling block.

When a user calls a procedure or function, it is executed on the server side. Obviously, this lowers network traffic. The subprograms are compiled programmes that are stored in the Oracle database and may be called at any time.

Because the sub programmes are saved in compiled form, they only need to run when they are called. As a result, they reduce the amount of time necessary to compile the sub programme.

2.2 STORED PROCEDURES

A stored procedure is a combination of SQL statements and control and condition handling statements that provides an interface to the Database. The term stored procedure refers to a set of SQL statements including some programming constructs.

A stored procedure is a named collection of procedural and SQL statements. Just like database triggers, stored procedures are stored in the database. One of the major advantages of stored procedures is that they can be used to encapsulate and represent

business transactions. For example, you can create a stored procedure to represent product sale, a credit update, or the addition of a new customer. By doing that, you can encapsulate SQL statements within a single stored procedure and execute them as a single transaction

Types of Stored Procedure

Stored procedure can be classified in two groups.

- SQL Stored procedure
- External Stored procedure

SQL Stored procedure

In this type of stored procedure, SQL language is used. This allows porting stored procedures from other database management systems (DBMS) to the iSeries server, as well as from the iSeries server to other DBMS, easier. The SQL stored procedures are implemented using SQL99's procedural SQL standard

External Stored procedure

One of the programming language is used to create an external stored procedure. It is written by the user iSeries server. You can generate *PGM objects or Service Programs by compiling host language programmes. To build an external stored procedure, the host language's source code must be compiled to create a programme object. The system is then told where to find the software object that implements this stored procedure using the CREATE PROCEDURE statement.

The SQL stored procedure and external stored procedure is almost same excluding following differences:

1. The database must have the privilege to CREATE EXTERNAL PROCEDURE
2. The nesting restriction for stored procedures is unaffected by calling an external stored procedure from a client application.

Registering Stored Procedure

A stored procedure must be registered with the database using the DECLARE PROCEDURE or CREATE PROCEDURE statements before it can be called by a client programme. Either of these statements can be used to define the stored procedure. The CREATE PROCEDURE statement varies from the DECLARE PROCEDURE

statement in that it populates the system catalogue tables with procedure and parameter definitions (SYSROUTINES and SYSPARMS).A stored procedure becomes accessible to any client software executing on the local or distant system in this manner. Because the stored procedure's information is stored in the systemcatalog tables, the CREATE PROCEDURE command is only needed once during the procedure's lifetime.To destroy the stored procedure Catalog information entry, use the DROP PROCEDURE statement.The statement DECLARE PROCEDURE is rarely used. It's primarily used for registering stored routines on a temporary basis.

Applications of Stored Procedure

Stored procedures can be used for many different application purposes such as:

- Distributing the logic between a client and a server
- Performing a sequence of operations at a remote site
- Combining results of query functions at a remote site
- Controlling access to database objects
- Performing non-database functions

Components of Stored Procedure

The procedure is mainly divided in two parts: Procedure head and Procedure body.

- Procedure Head

The Procedure head or signature refers to the code that comes before the "IS" keyword. The following are some of the components of the PL/SQL Procedure Head:

A. Schema

This field is optional and specifies the schema in which the process will be constructed. The current user is the default schema. If a separate user is specified, that user must have the ability to construct a procedure in his or her schema.

B. Name

The NAME parameter specifies the procedure's name. A procedure's name should be more meaningful and readable.

C. Parameters

The parameters are not required. To pass and receive values from a PLSQL process, these will be necessary. There are three types of parameters.

- IN : In a PLSQL procedure, this is the default style of parameter. When we want the parameter to be read-only (i.e., we can't modify its value in the PLSQL procedure), we use the IN mode.
- OUT : The values are returned to the calling subprogram or subroutines through the OUT parameter. We can't provide the OUT parameter a default value, thus we can't make it optional. Before exiting the process, we must assign a value to the OUT parameter; otherwise, the value of the OUT parameter will be NULL. We must ensure that we pass variables for the corresponding OUT parameters when calling a procedure with OUT parameters.
- INOUT : In this manner, the actual parameter is supplied to the PLSQL process with starting values, and the parameter's value can then be altered or reassigned within the PLSQL procedure. Finally, the IN OUT parameter is passed back to the caller function.

D. Authid

This is an optional parameter that specifies whether the procedure will run with the privileges of the procedure's Creator / Definer or with the privileges of the Current User.

- Procedure Body

The body of the procedure is anything after the "IS" keyword. In the procedure body, The declaration of variables in the declaration portion, the code to be executed in the executable statements part, and the code to handle any exceptions in the exception handling part. The declaration and exception handling elements of the PLSQL procedure body are optional.

In the executable statement section, we must have at least one executable statement. The business logic must be written during the execution phase.

The procedure's Return statement is used to stop the procedure from running any further and return control to the calling function. A user with the Construct Procedure system privilege is able to create a stored procedure. In order to build the procedure properly, the user must have required object privileges on the objects that are referred in the procedure.

Types of parameters in Stored Procedure

In stored procedure there are two types of parameters. Actual Parameters and Formal Parameters.

- Actual Parameters

The parameters which are passed while procedure is invoked is called actual parameters. For example

myProc (‘Hello Dr.’, ‘Bhavik’);

in above syntax , ‘Hello Dr’ and ‘Bhavik’ are actual parameters and these are passed to fomal parameters message and name.

- Formal Parameters

Formal parameters are the parameters that are declared in the method definition.They get the values that were sent when the operation was called. For example

procedure myProc(message varchar2, name varchar2)

in above code message and name are two formal parameters of varchar type.

Syntax of stored procedure

CREATE [OR REPLACE] PROCEDURE procedure_name
[(parameter_1 [IN] [OUT] parameter_data_type_1, ..)]

IS
— declaration_statements

BEGIN
— executable_statements
return {return_data_type};

[EXCEPTION

-the exception-handling statements]

END [procedure_name];

Example of Stored Procedure

Create or Replace Procedure update_product_price (prod_id IN Number, price IN Number)
IS
Begin
Update product
Set product_price = price
Where product_id = prod_id;
dbms_output.put_line('Procedure executed successfully');
End update_product_price;

Executing a stored procedure

Exec update_product_price(1,300);

Deleting a stored procedure

Drop procedure update_product_price;

Explanation

All the code before the "IS" keyword is called the Procedure head or signature. Various parts of PL/SQL Procedure Head are procedure_name and parameters
The parameters are of three types: IN , OUT and IN OUT

IN parameter : These are read only parameters, where the value of the parameter can be read but cannot be modified inside the procedure. It is the default parameter type.
Create or Replace Procedure procname(p1 in varchar2)

```
IS
Begin
dbms_output.put_line('IN parameter is: ' || p1 );
End;
```

Executing the procedure

```
exec procname ('baou');
```

OUT parameter: It is the parameter which is returned to the calling program. Procedures cannot return the values as function does but it can set the value of out parameter which can be returned to the calling program.

```
Create or Replace Procedure procname(p1 out varchar2)
IS
Begin
p1:= 'bscit';
End;
```

PL SQL block to call the proceducre

```
Declare
p1 varchar2(20);
Begin
procname(p1);
dbms_output.put_line(p1);
End;
```

IN OUT parameter : Its acts as both IN and OUT parameter such that the value can be passed as IN parameter and can be modified inside the procedure and can be returned to the calling program as OUT parameter.

```
Create or replace Procedure procname(p1 IN OUT varchar2)
IS
Begin
p1 := ‘baou ‘ || p1;
End;
```

PL SQL block to execute the procedure

```
Declare
p1 varchar2(100) := ‘bscit’;
Begin
procname (p1);
dbms_output.put_line(p1);
End;
```

Stored Procedure with DML Statement

1. Insert Statement

First we create Emp_data table in Oracle database as shown below.

```
Create Table Emp_data( Emp_id number (5) not null, empname varchar2 (20) not null, designation varchar2 (20) not null, primary key (Emp_id) );
```

After creating the Emp_data table, stored procedure can be created. The procedure will accept different parameters and insert it into table “Emp_data”

```
Create OR Replace Procedure insertEMPDATA(
 empid IN Emp_data.EMP_ID%TYPE,
empname IN Emp_data.EMPNAME%TYPE,
designation IN Emp_data.DESIGNATION%TYPE)
 IS
Begin
 Insert INTO Emp_data (“Emp_Id”, “empname”, “designation”)
          Values (empid, empname,designation);
```

```
 Commit;
End;
/
```

After creation of the procedure called insertEMPDATA, it can be executed by the PL/SQL block as per the following code

Example:

```
Begin
InserEMPDATA(111,'Dhyani','manager');
End;
```

After executing the above statement, the table record will be changed.

2. Update Statement

Same as insert statement update statement can be written to update the table through stored procedure. In following update statement the Emp_data table is updated.

Example:

```
Create or Replace Procedure updateEMPDATA(
 empid IN Emp_data.EMP_ID%TYPE,
empname IN Emp_data.EMPNAME%TYPE)
IS
Begin
Update EMP_data SET empname = newempname where EMP_ID = empid;
Commit;
End;
/
```

In above-stored procedure two IN parameters are accepted andupdate the username field based on the provideduserId.

After creation of the procedure called updateEMPDATA, it can be executed by the PL/SQL block as per the following code

Example:

```
        Begin
updateEMPDATA(111,'Joyal');
        End;
```

After execution of above block the verify the data of Emp_data table.

3. Delete Statement

The records of Emp_data table can be deleted with use of procedure called deleteEMPDATA.

```
Create or Replace Procedure deleteEMPDATA(
 empid IN Emp_data.EMP_ID%TYPE,
)
IS
Begin
Delete EMP_data  where EMP_ID = empid;
Commit;
End;
/
```

After creation of the procedure call deleteEMPDATA, it can be executed by the PL/SQL block as per the following code

Example:

```
        Begin
deleteEMPDATA(111);
        End;
         /
```

After execution of above block the verify the data of Emp_data table.

Check your progress 1

1. Stored procedure is a _____ set of one or more SQL statements.

   a.interpreted

b.compiled

c.precompiled

d. None of the above

2. Point out the wrong statement.

a. Stored procedure can accepts input and output parameters

b. Stored procedure can returns multiple values using input parameters

c. Using stored procedure, we can Select,Insert,Update,Delete data in database

d. None of the above

3Which of the following procedures are created by user for own actions?

a. User Defined Stored Procedure

b. Extended Procedure

c.CLR Stored Procedure

d. All of the above

4. Which of the following stored procedure is already defined in Sql Server?

a. User Defined Stored Procedure

b. Extended Procedure

c. CLR Stored Procedure

d. System defined Procedure

5.System defined Procedure logically appear in _____

a. sys schema

b. stor schema

c. proc schema

d. all of the mentioned

6. Extended procedures starts with the _____ prefix.

a. sp_

b. xp_

c. clr_

d. all of the above

7. Nesting limit OF stored procedure is up to _____level.

a. 30

b. 32

c. 34

d. 36

8. Type of procedure that are based on the Common Language Runtime is _____

a. User Defined Stored Procedure

b. Extended Procedure

c. CLR Stored Procedure

d. None of the above

9. Data return using output parameter is _____

a. Return codes, which are always an integer value

b. A global cursor that can be referenced outside the stored procedure

c. A single cursor that can be referenced inside the stored procedure

d. None of the mentioned

10. _____ provides details on any database object.

a. sp_changeowner

b. sp_owner

c. sp_change

d. none of the mentioned

2.3 FUNCTIONS

A stored function is basically a named group of procedural and SQL statements that returns a value. A stored function is similar to a procedure, except it returns a value.
To create a stored function, use the Create Function command.

Syntax to create function

Create [OR Replace] Function function_name
[(parameter_1 [IN] [OUT] parameter_data_type_1, …)]
RETURN return_datatype
IS | AS
— declaration_statements

BEGIN
— executable_statements
return {return_data_type};

[EXCEPTION
— the exception-handling statements]

END [function_name];

The name supplied to the PLSQL function is function name.
The parameter name specifies the name of the parameter to be sent to the function.In the code execution section of every Oracle PL/SQL function, there must be a Return statement.The data-type of the result returned by the Oracle PL/SQL function is indicated by RETURN in the header element of the function.

Stored functions can be invoked only from within stored procedures or triggers and cannot
be invoked from SQLstatementsRemember not to confuse built-in SQLfunctions (such as MIN, MAX, and AVG) with stored functions and not to use the built in function name for user defined functions.

Parameters passing to a function

• IN: This is the default parameter style in the PLSQL function. This has the same capabilities
as a Stored Procedure.
• OUT: The OUT parameter returns the values to the subprogram or subroutines that called it.
This has the same capabilities as a Stored Procedure.
• IN OUT: In this mode, the actual parameter is supplied to the PL/SQL function with starting

values, and the parameter's value can then be altered or reassigned within the PL/SQL function.

Finally, the IN OUT parameter is sent back to the caller function. This has the same capabilities

as a Stored Procedure.

Example1 of Function

```
Create or Replace Function view_products(pid IN number)
RETURN varchar2
IS
product_name varchar2(30);
Begin
Select product_name into p_name from product
Where product_id = pid;
Return p_name;
End view_products;
```

The function can be invoked by using a select statement

```
Select view_products (1) from dual;
```

It can also be invoked using a plsql block like:

```
Declare
p_name varchar2(30);
Begin
p_name := view_products(1);
dbms_output.put_line(p_name);
End;
```

Deleting a function

To delete a function drop command is used.

```
Drop function function_name;
```

Drop function view_products;

Example2 of function

The following example describes how we can create function to find minimum value from two values.

```
DECLARE
        X number;
           Y number;
Z number;
Function findMin( a IN number, b IN number)
RETURN number
Is
ans number;
BEGIN
  IF a < b THEN
ans: = a;
ELSE
ans:= b;
ENDIF;
 RETURN ans;
END;

BEGIN
    X:=10;
    Y:=20;
     Z:=findMin(X,Y);
dbms_output_line('MIN is : ' || z);
 END;
/
```

Check your progress 2

1.Function cannot be used for _____ statement.

a.Create

b.Drop

c.Select

d. Insert

2.The function in SQL can return___

a. set of values

b. result set

c. scalar value

d. All of the above

3.The benefit of user defined functions in SQL Server are__.

a. modular programming is possible

b. reduce network traffic

c. increase execution

d. All of the above

4. To create function __ syntax is used.

a. CREATE FUNCTION

b. CREATE FUNCTIONS

c. CREATE FUN

d. None of the above

5.Which of the following is not user defined function.

a. MIN()

b. Inline table value function

c. Scalar value function

d. All of the above

6. Scalar function can return ___ data type values.

a. Float

b. Numerical

c. String

d. All of the above

7. ___ type parameter(s) can be passed to function.

a. In

b. Out

c. InOut

d. All of the above

8. To delete function __ command is used.

a. delete

b. drop

c. none of the above

d. any of the above

9. __ is default parameter style in function.

a. In

b. out

c. InOut

d. Any of the above

10. The Basic difference between function and stored procedure is

a. function returns value

b. function does not returns value

c. stored procedure returns value

d. none of the above

2.4 LET US SUM UP

This unit gives unique learning on database objects including the stored procedure and the functions. In the previous unit we had studied about built in sql functions.

We can create our own procedures and functions which can be very useful in Database applications.

2.5 ANSWER FOR CHECK YOUR PROGRESS

Check your progress -1

Answers :(1-C), (2- b), (3- a), (4 -d),(5- a),(6-a),(7-b),(8-c),(9-d),(10-d)

Check your progress -2

Answers :(1-d), (2- d), (3- a), (4 -a),(5- a),(6-d),(7-d),(8-b),(9-a),(10-a)

2.6 GLOSSARY

A database object is basically any object created by end users; for example, tables, views, indexes, stored procedures, functions and triggers.

2.7 ASSIGNMENT

1. Explain Stored Procedure and its features.
2. Explain functions and its features
3. Write the difference between Stored Procedure and function.

2.8 ACTIVITIES

Differentiate functions and procedures

2.9 CASE STUDY

Explain where functions and procedures can be useful in real life database applications

2.10 FURTHER READING

1. Database Management Systems - Rajesh Narang -PHI Learning Pvt Ltd.
2. Database System Concepts bySilberschatz, Korth -Tata McGraw-Hill Publication.
3. An Introduction to Database Systems - Bipin Desai- GalgotiaPublication.
4. Database Management System by Raghu Ramkrishnan- Tata McGraw-Hill Publication.
5. SQL, PL/SQL: The Programming Language Oracle - Ivan Bayross- BPB Publication.

Unit 3: Oracle Packages and Triggers

3

Unit Structure

3.0 LEARNING OBJECTIVES

After learning this unit, you will be able to understand:

- Database concepts
- Creating packages
- Triggers
- Creation and deletion of users

3.1 PACKAGES

A package is a schema object that groups PL/SQL types, items, and sub programs that are conceptually connected. Packages normally consist of two parts: a specification and a body, though the body is not always required. The specification defines the types, variables, constants, exceptions, cursors, and sub programs that are accessible for usage in your applications. The body implements the requirements by completely defining cursors and sub programs. Once compiled and stored, a package comes to life in a session when one of its components is referenced in the session. The variables, arrays, cursors, and sub programs defined in the package are then allocated and loaded in memory for that session, and any changes made to the components are live for the whole session, across all operations, sub programs, and triggers. This differs from a sub programs, where variables only affect the sub programs and not other programmes in the session. Packages, unlike sub programs, cannot be invoked, parameterized, or nested.

A package's format, however, is similar to that of a sub programs, as demonstrated below.

```
CREATE PACKAGE nameofpackage AS -- specification (visible part)
 -- public type and item declarations
 -- sub programs specifications
END [nameofpackage];

CREATE PACKAGE BODY nameofbody AS -- body (hidden part)
-- private type and item declarations
```

-- sub programs bodies [BEGIN -- initialization statements]
END [nameofbody];

The specification contains public statements that the application can see. The body contains implementation information as well as private declarations that are not visible to the application. When building modular applications, this approach comes in handy because it allows for a server-centric application with all of the application's related programmes and variables saved as part of the package.

You can debug, improve, or replace a package body without affecting the package body's interface (package specification).

Construct PACKAGE and CREATE PACKAGE BODY statements are used to create packages and store them permanently in an Oracle database.

SQL*Plus or Enterprise Manager can be used to run these statements interactively.
In following example package is created with record type, cursor and procedures. The procedure used proc_hireemployee has used the sequence called seq_empno and SYSDATE function to insert new employee number and hire date.

```
CREATE PACKAGE emp_actions AS – specification
PROCEDURE proc_hireemployee (
ename VARCHAR2,
 job VARCHAR2,
mgr NUMBER,
sal NUMBER,
comm NUMBER,
deptno NUMBER);
PROCEDURE fire_employee (emp_id NUMBER);
END emp_actions;
```

```
CREATE PACKAGE BODY emp_actions AS
-- body PROCEDURE proc_hireemployee (
ename VARCHAR2,
job VARCHAR2,
mgr NUMBER,
sal NUMBER,
comm NUMBER,
deptno NUMBER) IS
BEGIN
INSERT INTO emp VALUES (seq_empno.NEXTVAL, ename, job, mgr, SYSDATE, sal,
comm, deptno);
END proc_hireemployee;

PROCEDURE proc_fireemployee (emp_id NUMBER) IS
BEGIN
DELETE FROM emp WHERE empno = emp_id;
END proc_fireemployee;

 END emp_actions;
```

In this package there are two procedures proc_hireemployee and proc_fireemployee. To execute this procedure following code is used:

```
EXEC
EMP_ACTIONS.PROC_HIREEMPLOYEE('BHAVIK','MANAGER',7902,20000,200,10);
EXEC EMP_ACTIONS.PROC_FIREEMPLOYEE(4);
```

Applications can only see and access the declarations in the package definition.
The package body contains implementation details that are hidden and inaccessible.

As a result, the body (implementation) can be modified without requiring the calling programmes to be recompiled.

Benefits of Package:

Packages provide various benefits, including modularity, ease of programme design, information concealment, increased functionality, and improved speed.

Modularity

Packages logically linked types, items, and support programmes in a named PL/ SQL module. The interfaces between packages are basic, clear, and well specified, and each package is easy to understand. This facilitates in the creation of applications.

Application Design is Easy

All that is required to start designing an application is the interface information found in the package specifications. Without body, we can code and compile the specifications. Then, stored subprograms that reference the package can be compiled as well. The package bodies fully until the user is ready to complete the application.

Information Hiding

Package provides accessibility facilities. With use of package content can be make private or public. The continent that must be most secure can be made private and not accessible to the sub programs. The public content can be accessed everywhere. So the security can be managed with use of package.

Added Functionality

Package provide extra facilities of public access. If the cursor and variables are public then can be shared by the subprograms that execute the environment and data can be maintained across the transactions without storing it in the database.

Better Performance

When a packaged subprogram is initially called, it loads the entire package into memory. As a result, further calls to linked subprograms in the package do not necessitate disc I/O. Packages also prevent dependencies from cascading and hence reduce wasteful recompilation. For instance, if a packaged function's definition is updated because the calling subprograms are not dependent on the package body, Oracle does not need to recompile them.

3.2 TRIGGERS

A stored procedure linked to a database table, view, or event is referred to as a database trigger.

The trigger is fired each time for each row affected by an Insert, Update, or Delete command

Whenever an event happens, the trigger can be called once. To avoid unexpected operations, the trigger might be activated prior to the occurrence. Procedural and SQL statements can both be included in the trigger's executable portion. Whenever the table is affected by any DML actions, the stored procedure and functions must be invoked explicitly or implicitly while the database triggers are executed implicitly.

A trigger is a piece of procedural SQL code that is automatically executed by the RDBMS when a certain data manipulation event occurs. It's important to keep in mind that:

Before or after a data row is entered, changed, or removed, a trigger is called.

1. A database table is linked with a trigger.
2. One or more triggers may exist for each database table.
3. A trigger is executed as part of the transaction that executed it

Triggers and Their Applications

1. Trigger allows you to enforce business rules that you can't create using integrity constants.

2. Trigger permits us to exert high-security control.

3. We can also collect statistical data on table access using triggers.

4. We can avoid invalid transactions by using triggers.

Parts of a Trigger

A Trigger contains 3 parts. They are

- Triggering event or statement
- Trigger Restriction
- Trigger Action

Triggering Event or Statement

It is a SQL statement that causes the trigger to be fired. The valid statements are the DML statements – Insert, Update or Delete. Additionally, trigger can be made to be fired when a particular column is updated. Also it can contain multiple DML statements.

Triggering Restriction

This is a Boolean expression which must be TRUE for the trigger to be fired.

Trigger Action

This is a procedure containing SQL and PL/SQL statements to be executed when a triggering statement is issued and also the triggering restriction is true.

Types of Triggers

Triggers are classified into different types depending on when the trigger is to be fired.

- BEFORE
- AFTER
- INSTEAD OF

Based on how many records are to be affected by the triggers, triggers can be classified as

- Row Level
- Statement Level

Syntax:

```
CREATE [OR REPLACE] TRIGGER Trigger_Name
{BEFORE | AFTER | INSTEAD OF }
{INSERT [OR] | UPDATE [OR] | DELETE}
[OF col_name]
ON table_name
[REFERENCING OLD AS o NEW AS n]
[FOR EACH ROW]
WHEN (condition)
BEGIN
--- SQL statements
END;
```

• CREATE [OR REPLACE] TRIGGER trigger_name : This creates a trigger with the given name or overwrites an existing trigger with the same name.

• {BEFORE | AFTER | INSTEAD OF}: It describes at what time the trigger will get fired. i.e before or after updating a table.

Before triggers are used when the triggering action determines whether the triggering statement must be allowed to be completed or not. This eliminates un–necessary processing of the triggering statement.

After triggers: This trigger is executed after the trigger statement is issued. It is used when the trigger statement has to be completed before the trigger action.

INSTEAD OF is used to create a trigger on a view. Before and aftercannot be used to create a trigger on a view.

• {INSERT [OR] | UPDATE [OR] | DELETE} : This determines the triggering event.
There are more than one triggering events that can be used together separatedby OR keyword.
The trigger gets fired at all the specified triggering event.

• [OF col_name] : This clause is used with update triggers. This clause is used when we want the trigger to fire only when a specific column is updated.

• [ON table_name] : This clause specifies the name of the table or view to which the trigger is associated.

• [REFERENCING OLD AS o NEW AS n] : This clause is used to reference the old and new values of the data being changed. By default, we reference the values as :old.column_name or :new.column_name.
• [FOR EACH ROW] : This clause is used to determine whether a trigger must fire

when each row gets affected (i.e. a Row Level Trigger) or just once when the entire SQL statement is executed (i.e. statement level Trigger).

• WHEN (condition) : This clause is valid only for row level triggers. The trigger is fired only for rows that satisfy the specified condition

Example:

Before Triggers

```
Create or replace triggertrigger_name
before update
on Employee
Begin
dbms_output.put_line('one record updated successfully.....');
End;
```

This trigger will be fired when update query is executed on employee table like

```
update employee
set empsalary = 10000
where empid =1
```

After Triggers

```
Create or replace triggertrigger_name
after delete
on Employee
for each row
Begin
dbms_output.put_line(:old.id||' record deleted successfully.....');
End;
```

The trigger will be fired whenever the corresponding query is executed. If after delete trigger is created it will be fired whenever delete query is executed on the table on which the trigger is created like

delete from employee where id = 1;

Check your progress 1

1) A _____ is a schema object that groups PL/SQL types, items, and sub programs that are conceptually connected

a) package

b) function

c) procedure

d) none of these

2) The _____ is fired each time for each row affected by an Insert, Update, or Delete command

a) trigger

b) function

c) package

d) procedure

3) A _____ is a piece of procedural SQL code that is automatically executed by the RDBMS when a certain data manipulation event occurs

a) trigger

b) function

c) package

d) procedure

4) _____ triggers are used when the triggering action determines whether the triggering statement must be allowed to be completed or not.

a) before

b) after

3.3 USER – CREATE AND DELETE USERS

Creating a user

You may use the CREATE USER command to create a new database user that you can use to log into the Oracle database.

The CREATE USER statement has the following basic syntax:

```
CREATE USER username
   IDENTIFIED BY password
[DEFAULT TABLESPACE tablespace]
   [QUOTA {size | UNLIMITED} ON tablespace]
   [PROFILE profile]
[PASSWORD EXPIRE]
   [ACCOUNT {LOCK | UNLOCK}];
```

CREATE USER username
Provide a name for the user that will be created.

IDENTIFIED BY password
Set a password for the local user who will access the database. It's worth noting that you can also establish a global or external user, which isn't addressed in this article.

[DEFAULT TABLESPACE tablespace]
Specify the tablespace for the objects the user will construct, such as tables and views.

If you omit this phrase, the user's objects will be saved in the database default tablespace, which is usually USERS, or the SYSTEM tablespace if no database default tablespace is available.

[QUOTA {size | UNLIMITED} ON tablespace]
Set the maximum amount of tablespace space that a user can utilise. Multiple QUOTA clauses can be used, each for a different tablespace.
If you don't want to limit the extent of the tablespace that a user may utilise, use UNLIMITED.

[PROFILE profile]
The database resources or password that a user can use are limited by a user profile. A profile can be assigned to a newly established user. If you leave this clause blank, Oracle will assign the user the DEFAULT profile.

[PASSWORD EXPIRE]
If you want to compel the user to update their password the first time they log in to the database, use the PASSWORD EXPIRE command.

[ACCOUNT {LOCK | UNLOCK}]
Use ACCOUNT LOCK if you want to lock user and disable access. On the other hand, specify ACCOUNT UNLOCK to unlock user and enable access.

Example

CREATE USER baou IDENTIFIED BY pwd123;

baou user will be created with password pwd123, other parameters are optional.

Username should begin with a letter and can contain max of 30 chars.
Username is not case sensitive if specific case is required use double quotes “ “ .
Username contain letters from A-Z , 0-9, underscore _ , pound or dollar.
When user is created, schema is automatically created for user

Deleting the user

The DROP USER command is used to remove a user from the oracle database and remove all objects owned by that user.
Syntax:
DROP USER user_name [CASCADE];
Where:

user_name: It specifies the name of the user to remove from the Oracle database.

CASCADE: It is optional. It specifies that if user_name owns any objects (i.e. tables or views in its schema), we must specify CASCADE to drop all of these objects.
Example:

If the user has created database objects like tables or views etc use :

drop user baou cascade;

if the user has not created any database objects it can be simply deleted using :

drop user baou;

Altering the user details

ALTER USER baou
IDENTIFIED BY pwdbaou1

ALTER USER baou
QUOTA 50M ON tablespace1

ALTER USER baou
ACCOUNT LOCK;

ALTER USER baou
PROFILE yourprofile

Check your progress -2

1) _____ is used to set a password for the local user who will access the database

a) IDENTIFIED BY password

b) CREATED BY password

c) REQUIRED BY password

d) none of these

2) If you want to compel the user to update their password the first time they log in to the database ____ should be set

a) PASSWORD EXPIRE

b) PASSWORD RESET

c) PASSWORD CHANGE

d) PASSWORD FORGET

3) Username should begin with a letter and can contain max of 30 chars.

a) true b) false

4) When user is created, schema is automatically created for user

a) true b) false

3.4 GRANT AND REVOKE COMMAND

GRANT

SQL Grant command is used to provide access or privileges on the database objects to the users.

Syntax :

```
GRANT privilege_name
ON object_name
TO {user_name | PUBLIC |role_name}
[with GRANT option];
```

privilage_name: The user's access permission or privilege is named privilege name.
object name : It refers to the name of a database object, such as a table or a view.
user_name : The name of the user to whom an access permission is being provided is user name.
PUBLIC: The term "public" is used to refer to the ability to offer privileges to all users.
GRANT option: The Grant option allows users to grant other users access permissions.

Example:
GRANT select, insert, update to baou;

The grant option can be used for giving update privilege on a specific column of a table to a user.
Grant update (empname) on employee to baou;

Check your progress – 3

1) ____ is used to delete a user
a) DROP USER USERNAME
b) DELETE USER USERNAME
c) DROP USERNAME
d) DELETE USERNAME

2) _____ specifies that if user_name owns any objects (i.e. tables orviews in its schema) to drop all of these objects
a) CASCADE
b) DROP
c) ROLLBACK
d) CHECKPOINT

3) The ____ option allows users to grant other users access permissions
a) GRANT
b) GET

c) REVOKE
d) DROP

REVOKE

The revoke command removes a user's database object access permissions or privileges.

The REVOKE command has the following syntax:

REVOKE privilege_name ON object_name FROM {User_name | PUBLIC | Role_name}

Example

REVOKE insert, update, delete FROM baou;

REVOKE update(empname) on employee from baou;

The SQL command revoke can be used to revoke system rights or responsibilities.
Any database user who has admin privileges or roles can withdraw those privileges or roles from other database users. The user does not have to be the same who gave the permission or privilege.

Check your progress -4

1) Any database user who has admin privileges or roles can withdraw those privileges or roles from other database users (True/False)
2) _____ is used to revoke privileges from a user

3.5 LET US SUM UP

This unit gives unique learning on Granting and Revoking user roles and privileges Creating packages Triggers and Creation and deletion of users

3.6 GLOSSARY

1. GRANT – This keyword is used to assign roles or privileges to the user
2. REVOKE - This keyword is used to withdraw roles or privileges from the user.

3.7 ANSWER FOR CHECK YOUR PROGRESS

Check your progress -1

Answers:(1-a) , (2- a), (3- a), (4 -a)

Check your progress -2

Answers:(1-a) , (2- a), (3- a),

Check your progress -3

Answers:(1-a) , (2- a), (3- a)

Check your progress -4

Answers:(1-a) , (2- a)

3.8 ASSIGNMENT

1. Explain creation of roles.
2. Explain assigning privileges to roles
3. Explain assigning roles to

3.9 ACTIVITIES

Write a note on various system roles and privileges in RDBMS

3.10 CASE STUDY

Explain the benefits of creation of roles in database applications

3.11 FURTHER READING

1. Database Management Systems - Rajesh Narang -PHI Learning Pvt Ltd.

2. Database System Concepts by Silberschatz, Korth -Tata McGraw-Hill Publication.

3. An Introduction to Database Systems - Bipin Desai- GalgotiaPublication.

4. Database Management System by Raghu Ramkrishnan- Tata McGraw-Hill Publication.

5. SQL, PL/SQL: The Programming Language Oracle - Ivan Bayross- BPB Publication users.

Unit 4: Oracle Roles and Privileges

4

Unit Structure

4.0 LEARNING OBJECTIVES

After learning this unit, you will be able to understand:

- Database concepts
- Oracle Roles and Privileges
- Granting and revoking roles and privileges

4.1 ORACLE DATABASE PRIVILEGES AND ROLES

A user privilege is a right to execute a particular type of SQL statement, or a right to access another user's object. The types of privileges are defined by Oracle.

Roles are created by users (usually administrators) and are used to group together privileges or other roles. They are a means of facilitating the granting of multiple privileges or roles to users.

4.2 PRIVILEGES – SYSTEM AND OBJECT PRIVILEGES

There are more than 100 system privileges and system privilege allows a user to perform a particular database operation or class of database operations System privileges should be granted only when necessary to roles and trusted users of the database.

A privileges a right to execute a particular type of SQL statement or to access another User's object. Some examples of privileges include the right to

- connect to the database (create a session)
- create a table
- select rows from another user's table
- execute another user's stored procedure

Types of Privileges

1. System Privileges
2. Object Privileges

System Privileges

A system privilege is the ability to perform a certain action or actions on all schema objects of a specific kind. System privileges, for example, include the ability to create tablespaces and remove records from any table in a database.

There are approximately 100 different system privileges to choose from. A user can conduct a specific database operation or a class of database operations with each system privilege.

Users and roles can have system privileges granted or revoked. You can use roles to manage system privileges if you provide them system privileges. Users and roles can be granted or removed system rights using the graphical dialog box or using grant and revoke commands.

Only the user with ADMIN privilege can grant or revoke system privileges

Some of the system privileges include

1. CREATE ANY TABLE Enables a user to create a table owned by any user in the database.

2. CREATE ANY VIEW Enables a user to create a view owned by any user in the database.

3. CREATE PROCEDURE Enables a user to create a PL/SQL procedure, function or package owned by that user.

4. CREATE ANY INDEX Enables a user to create an index on any table or view in the database

5. DROP ANY TABLE Enables a user to drop any table in the database.

6. DROP ANY VIEW Enables a user to drop any view in the database

7. ADMIN Enables a user to perform administrative tasks including check pointing, backups, migration, and user creation and deletion.

Object Privileges

A schema object privilege is a privilege or right to perform a particular action ona specific table, view, sequence, procedure, function, or package. Different object privileges are available for different types of schema objects

The privileges include ability of the user to perform following operations on database table

1. SELECT Enables a user to select from a table,

2. INSERT Enables a user to insert into a table

3. UPDATE Enables a user to update a table.

4. DELETE Enables a user to delete from a table.

5. REFERENCES Enables a user to create a foreign key dependency on a table

6. ALL All privileges

The privileges can be assigned or revoked from a role and the role can be assigned to a user.

Granting the privileges to a Role
GRANT SELECT on employee_table to role_clerk;

Revoking the privileges from a role
REVOKE all on employee_table from role_clerk;

Check your progress -1

1) A _____ is the ability to perform a certain action or actions on all schema objects of a specific kind

a) system privilege b) object role c) object privilege d) all the above

2) Only the user with _____ privilege can grant or revoke system privileges

a) admin b) system c) user d) none

3) A schema _____ privilege is a privilege or right to perform a particular action ona specific table

a) object b) system c) user d) none

4.3 ROLES – CREATE, GRANT, VIEW AND DELETE ROLES

A role groups several privileges and roles, so that they can be granted to and revoked from users simultaneously. A role must be enabled for a user before it can be used by the user.

A role is a named collection of database access privileges that authorize a user to connect to the database anduse the database system resources. Examples of few predefined roles are as follows:

CONNECT allows a user to connect to the database and create and modify tables, views, and other
data-related objects.

RESOURCE allows a user to create triggers, procedures, and other data management objects.

DBA gives the user database administration privileges.

You can create a role so that you can logically group the users' permissions, it requires CREATE ROLE system privileges.

Creating a role

CREATE ROLE ROLE_NAME
[NOT IDENTIFIED | IDENTIFIED {BY password | USING [schema.] package |
EXTERNALLY | GLOBALLY }] ;

ROLE NAME: This is the name of the new role you're making. This is how you'll refer to the privilege grouping.

NOT IDENTIFIED: This indicates that the role is turned on right away. The role can be enabled without a password.

IDENTIFIED: This indicates that a user must be approved by a certain way before the role may be activated.
BY password: This signifies that a user must provide a password in order for the role to be enabled.

USING PACKAGE: This signifies you're establishing an application role that's only available to apps that use an approved package.

EXTERNALLY: To activate the role, a user must be approved by an external service. An operating system or a third-party service might be considered an external service.

GLOBALLY: It indicates that the enterprise directory service must allow a user to enable the role.

Check your progress-2

1) Any database user can be assigned any role
a) true b) false

2) You provide a _____ role all the permissions it needs to execute a database application

a) application b) user c) database d) all the above

3) System or schema object privileges can be provided to a role. (True/False)

Granting privileges to a Role

CREATE ROLE USER_ROLE;

It will create New Role called USER_ROLE;

Once the role is created, your next step is to grant privileges to thatrole. GRANT insert, update, delete, select on employee_table to USER_ROLE;

The privileges can be removed from a role as

REVOKE insert, update, delete on employee_table from USER_ROLE;

If all privileges have to be removed you can use

REVOKE ALL on employee_table from USER_ROLE;

Granting role to a user

After the roles are created and privileges are assigned to a role, the role can be assigned to an existing user.

Syntax

GRANT ROLE_NAME TO USER_NAME;

Example:

GRANT USER_ROLE to SCOTT;

SCOTT is the default user in the oracle database. So here the USER_ROLE will be assigned to the user SCOTT.

Check your progress-3

1) _____ specifies system resources that are available to a database user while working with a database

a) profile b) roles c) users d) all the above

2)_____ indicates total number of simultaneous sessions per user

a) SESSIONS_PER_USER

b) SESSIONS_PER_DATA

c) CPU_PER_SESSION

d) none of these

3) A default profile is assigned to the user if no profile is mentioned in creation of the user

a) true b) false

Deleting the roles

It may be necessary to remove a role from the database under specific circumstances. All users and roles who have been given a dropped role's security domains are instantly updated to reflect the absence of the dropped role's rights. The discarded role's indirectly given roles are likewise deleted from the impacted security domains. When you delete a role from a user's default role list, it's gone forever.

Tables and other objects are not dropped when a role is dropped since their creation is not reliant on the rights granted by the role.

Syntax

DROP ROLE ROLE_NAME;

Example

DROP ROLE USER_ROLE;

Check your progress-4

1) ____ indicates database user can use as much space on disk as required.

a) UNLIMITED b) UNSPECIFIED c) UNDECLARED d) all the above

2) Tables and other objects are not dropped when a role is dropped since their creation is not reliant on the rights granted by the role.

a) true b) false

3) After the roles are created and privileges are assigned to a role, the role can be assigned to an existing user.

a) True b) false

4) ___ is used to delete a role

a) drop role role_name;

b) delete role role_name;

c) drop role_name;

d) delete role_name;

4.4 LET US SUM UP

In this unit we have learnt about creation of roles, assigning privilages to the roles and assigning roles to the uses. The roles can be granted and revoked from a user and privileges can be dropped from a role. This helps in grouping the privileges to a role and assigning it to users having the same work area.

4.5 GLOSSARY

1. A role is a named collection of database access privileges that authorize a user to connect to the database and use the database system resources

2. A privilege is a right to execute a particular type of SQL statement or to access another user's object.

4.6 ANSWER FOR CHECK YOUR PROGRESS

Check your progress -1

Answers :(1-a) , (2- a), (3- a),

Check your progress -2

Answers :(1-a) , (2- a), (3- a),

Check your progress -3

Answers :(1-a) , (2- a), (3- a)

Check your progress -4

Answers :(1-a) , (2- a), , (3- a), (4- a)

4.7 ASSIGNMENT

1. Explain Roles and its features.
2. Explain types of privileges

4.8 ACTIVITIES

1. Write a note on creation privileges and assigning them to a role.
2. How to assign a role to a user?

4.9 CASE STUDY

Prepare a list of roles and privileges defined by oracle.

4.10 FURTHER READING

1. Database Management Systems - Rajesh Narang -PHI Learning Pvt Ltd.
2. Database System Concepts bySilberschatz, Korth -Tata McGraw-Hill Publication.
3. An Introduction to Database Systems - Bipin Desai- Galgotia Publication.
4. Database Management System by Raghu Ramkrishnan- Tata McGraw-Hill Publication.
5. SQL, PL/SQL: The Programming Language Oracle - Ivan Bayross- BPB Publication.

BLOCK 4: PL/SQL - ORACLE

Block Introduction

In this block, we will learn about basic concepts of PL/SQL block, Cursor Locking Methods and different types of Exceptions. It provides an overview and various examples of the various above tasks.

You will also get the concept of the different types of cursor, which provides control on the program by using the cursor. A cursor is a pointer to this context part. In PL/SQL, the context part is controlled by Cursor. A cursor contains information on a select statement and the rows of data accessed by it.

The aim of this block is to enable the reader to understand the basic concepts of these topics as PL/SQL blocks, types of cursors, different types locking methods for lock tables of the database. As well as also discussed exceptions handling which is used by the programmer to protect and handle the errors occurred in the running program. The writer has tried his best to detail the concepts; apart from this, the sub-topics have even been discussed by him in detail. On the other hand, he has even discussed the PL/SQL block, Cursor Locking Methods and different types of Exceptions in detail with sufficient examples.

The objective of this block includes detailing the various above mentioned topics in very detail with the help of sufficient and suitable understandable examples.

Unit 1: PL-SQL ORACLE Basics

1

Unit Structure

1.0 LEARNING OBJECTIVES

After learning this unit, you will be able to understand:

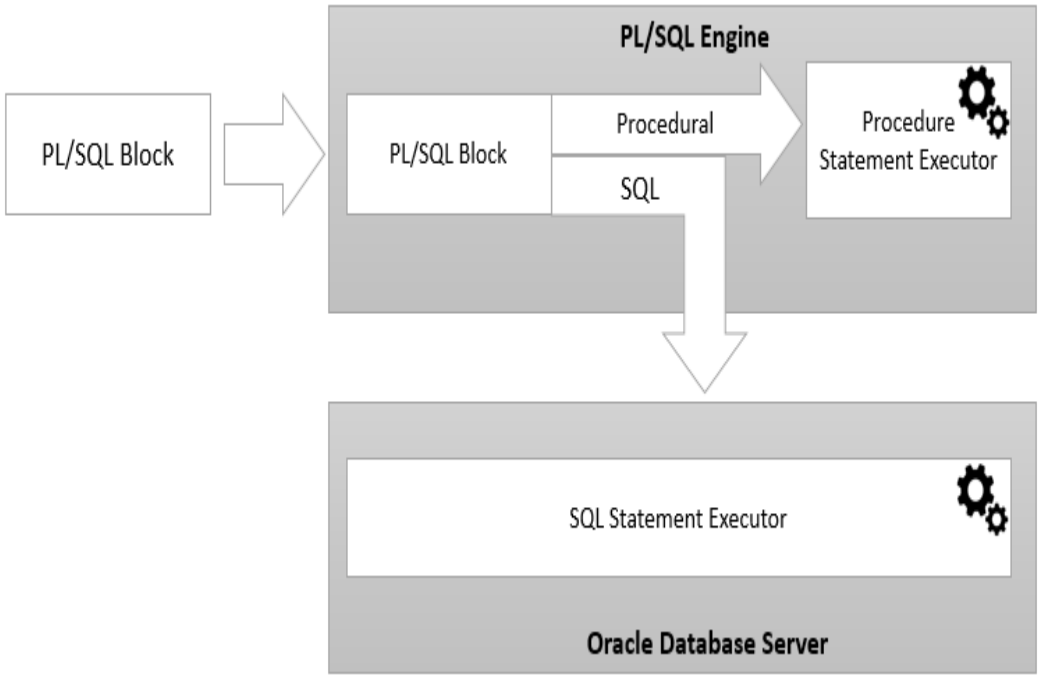
- PL/SQL concepts
- Architecture of the PL/SQL
- Importance and implementation of PL/SQL block
- Students will be able to declare, initialize and access local and global variables
- Students will be able to write a PL/SQL block and execute it
- Students will be able to print the message or value from the PL/SQL block

1.1 INTRODUCTION

PL/SQL is a block structured language that enables developers to combine the power of SQL with procedural statements. All the statements of a block are passed to oracle engine all at once which increases processing speed and decreases the traffic. PL/SQL is basically a procedural language, which provides the functionality of decision making, iteration and many more features of procedural programming languages. PL/SQL can execute a number of queries in one block using single command. One can create a PL/SQL unit such as procedures, functions, packages, triggers, and types, which are stored in the database for reuse by applications. PL/SQL provides a feature to handle the exception which occurs in PL/SQL block known as exception handling block. Applications written in PL/SQL are portable to computer hardware or operating system where Oracle is operational. PL/SQL Offers extensive error checking

PL/SQL is a highly structured and readable language. Its constructs express the intent of the code clearly. Also, PL/SQL is a straightforward language to learn. Also L/SQL is an embedded language. PL/SQL only can execute in an Oracle Database. It was not designed to use as a standalone language like Java, C#, and C++. In other words, you cannot develop a PL/SQL program that runs on a system that does not have an Oracle Database. Besides PL/SQL, you can use other programming languages such as Java, C#, and C++. However, it is easier to write efficient code in PL/SQL than other programming languages when it comes to interacting with the Oracle

The following picture illustrates the PL/SQL architecture:



PL/SQL engine is in charge of compiling PL/SQL code into byte-code and executes the executable code. The PL/SQL engine can only be installed in an Oracle Database server or an application development tool such as Oracle Forms.

Fundamental Concepts of the PL/SQL

I. Lexical Units

PL/SQL is not case-sensitive language, so lower-case letters are equivalent to corresponding upper-case letters except within string and character literals. A line of PL/SQL text contains groups of characters known as lexical units, which can be classified as follows:

II. Delimiters (Simple and Compound Symbols)

A delimiter is a simple or compound symbol that has a special meaning to PL/SQL. For example, we can use delimiters to represent arithmetic operations such as addition and subtraction.

III. Identifiers (include Reserved Words)

We can use identifiers to name PL/SQL program objects and units, which include constants, variables, exceptions, cursors, subprograms and packages. Some identifiers called Reserved Words, have a special syntactic meaning to PL/SQL and so cannot be redefined. For flexibility, PL/SQL lets us to enclose identifiers within double quotes. Quoted identifiers are seldom needed, but occasionally they can be useful.

IV. Literals

A literal is an explicit numeric, character, string, or Boolean value not represented by an identifier. Two kinds of numeric literals can be used in arithmetic expressions: integers and reals.

- String literal is a sequence of zero or more characters enclosed by single quotes. All string literals except the null string (`') belong to type CHAR. PL/SQL is case-sensitive within string literals. •Boolean literals are the predefined values TRUE and FALSE and the non-value NULL (which stands for a missing, unknown, or inapplicable value). Boolean literals are not strings.

V. Comments

The PL/SQL compiler ignores comments. Adding comments to our program enhances readability and guides the user in understanding the code. PL/SQL supports two types of comment styles, single-line and multiline.

- Single-line comments begin with a double hyphen (--) anywhere on a line and extend to the end of the line.
- Multiline comments begin with a slash asterisk (/*), end with an asterisk-slash (*/), and can span multiple lines. We cannot nest comments.

Check your progress 1

1. PL/SQL supports _____ types of comment styles

A. One

B. Two

C. Three

D. None of these

2. Which portion is optional in IF statements?

E. ELSE

F. THEN

G. IF

H. None of these

3. PL/SQL Variables are by default

A. Not Case Sensitive

B. Case Sensitive

C. Upper Case Sensitive

D. All of these

1.2 ADVANTAGES OF PL/SQL

The basic advantages of the PL/SQL are,

1. Block structures: It consists of blocks of code, which can be nested within each other. Each block forms a unit of a task or a logical module. PL/SQL blocks are often kept within the info and reused.

2. Procedural language capability: It consists of procedural language constructs like conditional statements (if-else statements) and loops like (FOR loops).

3. Better performance: PL/SQL engine processes multiple SQL statements at the same time as one block, thereby reducing network traffic.

4. Error handling: PL/SQL handles errors or exceptions effectively throughout the execution of a PL/SQL program. Once an associate degree exception is caught, specific actions can be taken depending upon the type of the exception or it can be displayed to the user with a message. PL/SQL saves time on design and debugging by strong features, such as exception handling, encapsulation, data hiding, and object-oriented data types.

5. Integrated: SQL is the standard database language and PL/SQL is strongly integrated with SQL. PL/SQL supports both static and dynamic SQL. Static SQL supports DML operations and transaction control from PL/SQL block. In Dynamic SQL, SQL allows embedding DDL statements in PL/SQL blocks.

6. High productivity: PL/SQL gives high productivity to programmers as it can query, transform, and update data in a database.

7. Security: PL/SQL provides support for Object-Oriented Programming. • PL/SQL provides support for developing Web Applications and Server Pages.

8. Portable: Applications written in PL/SQL are fully portable.

9. OOS Support: PL/SQL provides support for Object-Oriented Programming. As well as PL/SQL provides support for developing Web Applications and Server Pages.

Check your progress 2

1. PL/SQL is a

A. Brick Structured Language

B. Block Structured Language

C. Banner Structured Language

D. Build Structured Language

2. Oracle Database's _____ are inherited in PL/SQL.

A. Portability

B. Robustness

C. Security

D. All of the above

3. A Variable in PL/SQL should not exceed

A. 10

B. 20

C. 30

D. 40

4. ____ are values used in PL/SQL blocks that do not change during execution.

A. Variables

B. Constants

C. Functions

D. Cursor

1.3 GENERIC PL/SQL BLOCK

1) Declaration section

A PL/SQL block has a declaration section where you declare variables, this section starts with the keyword DECLARE. It is an optional section and defines all variables, cursors, subprograms, allocate memory for cursors, and define data types. And other elements to be used in the program

2) Executable commands section

A PL/SQL block has an executable section. This section is enclosed between the keywords BEGIN and END and it is a mandatory section. The executable section must have a least one executable statement, which may be just a NULL command to indicate that nothing should be executed.

3) Exception-handling section

A PL/SQL block has an exception-handling section that starts with the keyword EXCEPTION. This optional section contains exception(s) that handle errors in the program. The exception-handling section is where you catch and handle exceptions raised by the code in the execution section.

PL/SQL Block

In PL/SQL, the code is not executed in single line format, but it is always executed by grouping the code into a single element called Blocks. In this tutorial, you are going to learn about these blocks.

Blocks contain both PL/SQL as well as SQL instruction. All these instruction will be executed as a whole rather than executing a single instruction at a time.

PL/SQL blocks have a pre-defined structure in which the code is to be grouped. Below are different sections of PL/SQL blocks.

E. Declaration section

F. Execution section

G. Exception-Handling section

Declaration Section

This is the first section of the PL/SQL blocks. This section is an optional part. This is the section in which the declaration of variables, cursors, exceptions, subprograms, pragma instructions and collections that are needed in the block will be declared. Below are few more characteristics of this part.

• This particular section is optional and can be skipped if no declarations are needed.

• This should be the first section in a PL/SQL block, if present.

• This section starts with the keyword 'DECLARE' for triggers and anonymous block. For other subprograms, this keyword will not be present. Instead, the part after the subprogram name definition marks the declaration section.

• This section should always be followed by execution section.

Execution Section

Execution part is the main and mandatory part which actually executes the code that is written inside it. Since the PL/SQL expects the executable statements from this block this cannot be an empty block, i.e., it should have at least one valid executable code line in it. Below are few more characteristics of this part.

• This can contain both PL/SQL code and SQL code.

• This can contain one or many blocks inside it as a nested block.

• This section starts with the keyword 'BEGIN'.

• This section should be followed either by 'END' or Exception-Handling section (if present)

Exception-Handling Section

The exception is unavoidable in the program which occurs at run-time and to handle this Oracle has provided an Exception-handling section in blocks. This section can also contain PL/SQL statements. This is an optional section of the PL/SQL blocks.

• This is the section where the exception raised in the execution block is handled.

• This section is the last part of the PL/SQL block.

• Control from this section can never return to the execution block.

• This section starts with the keyword 'EXCEPTION'.

• This section should always be followed by the keyword 'END'

DECLARE--- Optional

<Declarations Sections>

BEGIN --- Mandatory

< Executable Commands>

EXCEPTION--- Optional

<Exception Handling>

END; --- Mandatory

Check your progress 3

1. PL/SQL basically procedural language

A. True

B. False

2. Which of the following is a Character literal?

A. 4

B. B

C. %

D. All of these

3. PL/SQL Constant is a/an Constants literal value

A. true

B. false

Check your progress 4

1 Oracle is a _____

A. Oracle is a OOS language

B. Oracle is a database

C. Oracle is a scripting language

D. Oracle is an operating system2.

2. Which one is an advantage of Oracle database?

A. Portability

B. Flashback Technology

C. Performance

D. All of the mentioned

3. Which one is a disadvantage of Oracle database?

A. Difficult to manage

B. Cost of level

C. Complexity

D. All of the mentioned

1.4 LET US SUM UP

This unit gives unique learning on PL/SQL block and it helps to make practice on combined SQL commands which provides groups of result.

1.5 ANSWER FOR CHECK YOUR PROGRESS

Check your progress 1

Answers: (1-b), (2-a), (3-a)

Check your progress 2

Answers: (1-b), (2-d), (3-c), (4-b)

Check your progress 3

Answers: (1-a), (2-d), (3-a)

Check your progress 4

Answers: (1-b), (2-d), (3-d)

1.6 GLOSSARY

Pl/SQL Block- PL/SQL is basically a procedural language, which provides the functionality of decision making, PL/SQL can execute a number of queries in one block using single command. One can create a PL/SQL unit such as procedures, functions, packages, triggers, and types, which are stored in the database for reuse by applications

Exception-handling -Exception(s) that handle errors in the program, the exception-handling section is where you catch and handle exceptions raised by the code in the execution section.

Error handling: PL/SQL handles errors or exceptions effectively throughout the execution of a PL/SQL program. Once an associate degree exception is caught, specific actions can be taken depending upon the type of the exception or it can be displayed to the user with a message.

Exception: PL/SQL saves time on design and debugging by strong features, such as exception

Exception-handling section: A PL/SQL block has an exception-handling section that starts with the keyword EXCEPTION. This optional section contains exception(s) that handle errors in the program. The exception-handling section is where you catch and handle exceptions raised by the code in the execution section.

1.7 ASSIGNMENT

1. Explain PL/SQL Block sections with details
2. Define the generic PL/SQL block

1.8 ACTIVITIES

1. Write a note PL/SQL block
2. Implements the PL/SQL blocks as requirements

1.9 ACTIVITIES

1. Explain the Features and benefits of the PL/SQL block
2. Explain the practical benefits of the blocks

1.10 FURTHER READING

1. Database System Concepts bySilberschatz, Korth -Tata McGraw-Hill Publication.
2. An Introduction to Database Systems - Bipin Desai- Galgotia Publication.
3. Database Management System by Raghu Ramkrishnan- Tata McGraw-Hill Publication
4. SQL, PL/SQL: The Programming Language Oracle - Ivan Bayross- BPB Publication

Unit 2: Cursor 2

Unit Structure

2.0 LEARNING OBJECTIVES

After learning this unit, you will be able to understand:

- Meaning of Cursor
- Use of Cursor
- Types of Cursor
- Practical implementation of cursor

2.1 IMPLICIT CURSOR

A cursor is a pointer to this context part. It contains all information needed for processing the statement. In PL/SQL, the context part is controlled by Cursor. A cursor contains information on a select statement and the rows of data accessed by it.

Cursor Actions:

Declare Cursor: A cursor is declared by defining the SQL statement that returns a result set.

- Open: A Cursor is opened and populated by executing the SQL statement defined by the cursor.
- Fetch: When the cursor is opened, rows can be fetched from the cursor one by one or in a block to perform data manipulation.
- Close: After data manipulation, close the cursor explicitly.
- De-allocate: Finally, delete the cursor definition and release all the system resources associated with the cursor.

Types of Cursors:

Cursors are classified depending on the circumstances in which they are opened.

Implicit Cursor: If the Oracle engine opened a cursor for its internal processing it is known as an Implicit Cursor. It is created “automatically” for the user by Oracle when a query is executed and is simpler to code.

Explicit Cursor: A Cursor can also be opened for processing data through a PL/SQL block, on demand. Such a user-defined cursor is known as an Explicit Cursor.

Implicit cursors:

The implicit cursors are automatically generated by Oracle while an SQL statement is executed; these are created by default to process the statements when DML statements like INSERT, UPDATE, and DELETE etc. are executed. Implicit cursor's attributes to check the status of DML operations. Some of them are: %FOUND, %NOTFOUND, %ROWCOUNT and %ISOPEN.

The following table specifies the context of the cursor with each of its attribute to implicit cursor.

SQL%ISOPEN	Always returns FALSE for implicit cursors, because the SQL cursor is automatically closed after executing its associated SQL statements.
SQL%FOUND	It return value TRUE if DML statements like INSERT, DELETE or UPDATE affect at least one row or more rows or a SELECT INTO statement returned one or more rows. Otherwise it returns FALSE.
SQL%NOTFOUND	It is a just opposite of %FOUND. It return value is TRUE if DML statements like INSERT, DELETE and UPDATE affect no row, or a SELECT INTO statement return no rows. Otherwise it returns FALSE.
SQL%ROWCOUNT	It returns the number of rows affected by DML statements like INSERT, DELETE, and UPDATE or returned by a SELECT INTO statement.

Practical example:

```
DECLARE
total_rows number(2);
BEGIN
  UPDATE employee
  SET salary = salary + 2000;
  IF sql%notfound THEN
    dbms_output.put_line('no employees selected');
```

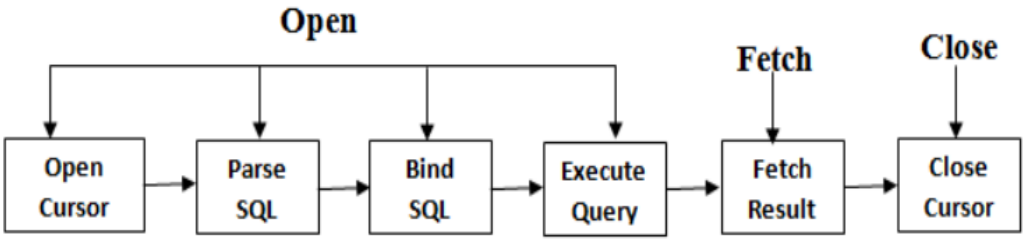
```
  ELSIF sql%found THEN
total_rows := sql%rowcount;
    dbms_output.put_line( total_rows || ' employee selected');
  END IF;
END;
```

The above program will update the table and increase the salary of each employee by 2000 and use the SQL%ROWCOUNT attribute to determine the number of rows affected

The important steps in the cursor execution cycle are OPEN, FETCH and CLOSE.

A cursor execution cycle refers to the stages which a cursor follows to process and execute the query.

The phases of cursor execution cycle are listed below:



The activity carried out by the server in the key phases is:

1. OPEN Phase In this phase, PGA memory is allocated for cursor processing, SELECT statement is parsed, Variable binding takes place, SELECT Query executes and finally pointer moves to the first record.

2. FETCH Phase In this phase, the record to which the record pointer points, is retrieved from the result set. The record pointer will move only in the forward direction. The FETCH phase lives until the last record is reached.

3. CLOSE Phase After the last record of the result set is reached, cursor is closed and allocated memory will be garbage collected and returned back to SGA. If an open cursor is not closed, oracle automatically closes it after the execution of its parent block.

Check your progress 1

1. Static cursors are always read only

A. True

B. False

2. Cursor Life cycle involves _____ steps.

A. 2

B. 3

C. 6

D. 5

3.. cursor for its internal processing it is known as true

A. Implicit cursor

B. Explicit cursor

C. Raised cursor

D. None of these

2.2 EXPLICIT CURSOR

The explicit cursor is used to process more than one record individually. The Explicit cursors are defined by the programmers to gain more control over the context area. These cursors should be defined in the declaration section of the PL/SQL block. It is created on a SELECT statement which returns more than one row.

The steps required to manage explicit cursor and manipulate data.

1. Declare a cursor to initialize in the memory.
2. Open a cursor to allocate memory.
3. Fetching a cursor to retrieve data.
4. Processing data manipulations
5. Closing cursor to release allocated memory.

Steps:

You must follow these steps while working with an explicit cursor.

1. Declare the cursor to initialize in the memory.
2. Open the cursor to allocate memory.
3. Fetch the cursor to retrieve data.
4. Close the cursor to release allocated memory.

1 Declare the cursor:

It defines the cursor with a name and the associated SELECT statement.

Syntax for explicit cursor declaration

CURSOR name IS

SELECT statement;

2 Open the cursor:

It is used to allocate memory for the cursor and make it easy to fetch the rows Returned by the SQL statements into it

Syntax for cursor open:

OPEN cursor_name;

3 Fetch the cursor:

It is used to access one row at a time. You can fetch rows from the above-opened cursor as follows:

Syntax for cursor fetches:

FETCH cursor_name INTO variable list;

4 Close the cursor:

It is used to release the allocated memory. The following syntax is used to close the above-opened cursors.

Syntax for cursor close:

Close cursor_name;

Syntax of explicit cursor

Following is the syntax to create an explicit cursor:

CURSOR cursor_name IS select_statement;;

Attribute of the Explicit Cursor

%ISOPEN	If explicit cursor if Open, return true, else return false
SQL%FOUND	If record fetched successfully then return true, else return false,

SQL%NOTFOUND	If record Not fetched successfully then return true, else return false,
SQL%ROWCOUNT	Return the number of records fetched from the active data set. It is set to zero when the cursor is opened.

Example of the Explicit Cursor

Explicit cursors are defined by programmers to gain more control over the context area. It is defined in the declaration section of the PL/SQL block. It is created on a SELECT statement which returns more than one row. Let's take an example to demonstrate the use of explicit cursor. In this example, we are using the already created EMPLOYEE table.

Create employee table and have records:

ID	NAME	AGE	ADDRESS	SALARY
1	Suresh	20	Ahmedabad	25,000
2	Pankaj	24	Vadodara	27,000
3	SOM	28	Vadodara	29,000
4	Sima	24	Ahmedabad	31,000
5	Ajay	23	Surat	35,000
6	Sunita	21	Ahmedabad	42,000

DECLARE

e_id employee.id%type;

*e_name employee.**name**%type;*

e_address employee.address%type;

CURSOR *e_ employee* **is**

```
    SELECT id, name, address FROM employee;
BEGIN
  OPEN e_ employee;
  LOOP
    FETCH e_ employee into e_id, e_name, e_addr;
    EXIT WHEN c_ employee %notfound;
    dbms_output.put_line(e_id || ' ' || e_name || ' ' || e_address);
  END LOOP;
  CLOSE e_ employee;
END;
/
```

Execute the above program to retrieve the employee name and address.

OUTPUT:

E-ID	E-NAME	E-ADDRESS
1	Suresh	Ahmedabad
2	Pankaj	Vadodara
3	SOM	Vadodara
4	Sima	Ahmedabad
5	Ajay	Surat
6	Sunita	Ahmedabad

Check your progress 2

1_____ contains a pointer

A. cursor

B. block

C. trigger

D. All of these

2. The Explicit cursors are defined by

A. User

B. Programmer

C. Engineer

D. None of these

3. Cursors are declared and used by the user to process multiple rows, returned by SELECT statement.

A. Implicit cursor

B. Explicit cursor

C. internal cursor

D. All of these

2.3 CURSOR FOR LOOP

The cursor for Loop can be used to process multiple records. There are two benefits with cursor for Loop.

1. It implicitly declares a %ROWTYPE variable.

2. Cursor for loop itself opens a cursor, read records and then closes the cursor automatically. So, Open, Fetch and Close statements are not necessary in it. To process a cursor, we can use cursor FOR loop to automate the following steps.

- Opening cursor
- Fetching rows from the cursor
- Terminating loop when all rows in the cursor are fetched
- Closing cursor

The following is the syntax of cursor for loop. This for loop is specifically meant to process cursors.

A fetch statement can fetch data from single row of an active data set. But there is a need to fetch more than one row from the database; the fetched statement is enclosed in loop to process multiple rows. Oracle provide another loop statement to control loops specially for cursors, this called cursor for loop, the syntax is,

```
DECLARE
CURSOR <cursor_name> IS <SELECT statement>;
BEGIN
FORI IN<cursor_name>
LOOP
  < - Execute Command -->
  .
  END LOOP;
END;
```

Practical example

```
DECLARE
CURSOR <cursor_Name> IS SELECT emp_name FROM empoyee;
BEGIN
FOR lv_emp_name IN <cursor_Name>
LOOP
Dbms_output.put_line ('Employee Fetched: '|| lv_emp_name.emp_name);
END LOOP;
END;
```

Using this example, will project all the employee name from employee table using a cursor-FOR loop.

Check your progress 3

1 cursor for Loop can be used to process _____records

A. multiple

B. Single

C. Double

D. none

2. The Explicit cursors are defined by

A. User

B. Programmer

C. Engineer

D. None of these

3. %notfound is an implicit cursor attributes

A. true

B. false

2.4 PARAMETERISED CURSOR

Using Parameterised cursor, we can also pass parameters to the cursors same like a subprogram IN parameter, these types of cursor are called as parameterized cursor, in parameterized cursor we are defining formal parameters when we are declaring a cursor and passing actual parameter when we are opening the cursor.

Parameterised cursor assign default values is assigned to the Cursor parameters, it allows us to create dynamic SQL queries with conditions containing the variables. Parameterized cursor passes the parameters into a cursor and uses them in to query. PL/SQL Parameterized cursor define only data type of parameter and not need to define its length. Parameterized cursors are also saying static cursors that can pass parameter value when cursor is opened.

We have to remember that In Oracle when we are defining formal parameters in cursors, procedure, functions, then we are not allow to use data types size in formal parameter declaration.

Note:

1. Scope of the parameters are locally
2. You can assign default value to a cursor parameter.

Here we have the syntax for declaration of Parameterized Cursor:

CURSOR cursor_name (variable_name Data type) IS <SELECT statement.

Example of the parameterised cursor

```
SQL>set serveroutput on
SQL>edit parameter_cursor_example
DECLARE
cursor c(no number) is select * from emp_information
whereemp_no = no;
tmpemp_information%rowtype;
BEGIN
  OPEN e(4);
  FOR tmp IN c(4) LOOP
 dbms_output.put_line('EMP_No:    '||tmp.employee_no);
 dbms_output.put_line('EMP_Name:  '||tmp. employee _name);
 dbms_output.put_line('EMP_Dept:  '||tmp. employee _dept);
  dbms_output.put_line('EMP_Salary:'||tmp. employee _salary);
  END Loop;
CLOSE e;
END;
/

SQL>@parameter_cursor_example
```

In the cursor query, each parameter in the parameter list can be used anywhere which

a constant is used. The cursor parameters cannot be referenced outside of the cursor query.

Check your progress 4

1. Which line in the following statement will produce an error?

A. Cursor action_cursor is

B. Select name, rate, action

C. Into action_record

D. From action_table;

2. The command used to open a CURSOR FOR loop is

A. open

B. fetch

C. parse

D. None

3. Parameterised cursor assign default values

A. True

B. false

2.5 LET US SUM UP

In this unit we have learnt that the major task of a cursor is to fetch data, one row at a time, from the result set. Cursors are used whenever the user wants to manipulate or update records in a singleton fashion or in a row by row manner, in a database table. The information stored in the Cursor is known as Active Data Set. Cursors are opened in predefined area of Oracle's DBMS in the main memory set, where the cursors are opened. We have also discussed cursor with for loop and parameter. Cursor plays an important role in accessing data one row at a time unlike sql commands.

2.6 ANSWER FOR CHECK YOUR PROGRESS

Check your progress 1

Answers: (1-a), (2-d), (3-a)

Check your progress 2

Answers: (1-a), (2-b), (3-b)

Check your progress 3

Answers: (1-a), (2-b), (3-a)

Check your progress 4

Answers: (1-c), (2-d), (3-a)

2.7 GLOSSARY

Cursor - A cursor is a pointer to this context part. It contains all information needed for processing the statement. In PL/SQL, the context part is controlled by Cursor.

A cursor contains information on a select statement and the rows of data accessed by it.

Implicit Cursor - If the Oracle engine opened a cursor for its internal processing it is known as an Implicit Cursor. It is created “automatically” for the user by Oracle when a Query is executed and is simpler to code.

Explicit Cursor - A Cursor can also be opened for processing data through a PL/SQL block, on demand. Such a user-defined cursor is known as an Explicit Cursor.

Cursor for Loop- A fetch statement can fetch data from single row of an active data set.

But there is a need to fetch more than one row from the database

Parameterised cursor - assign default values is assigned to the Cursor parameters, it allows us to create dynamic SQL queries with conditions containing the variables. Parameterized cursor passes the parameters into a cursor and uses them in to query. PL/SQL Parameterized cursor define only data type of parameter and not need to define its length. Parameterized

cursors are also saying static cursors that can pass parameter value when cursor is opened.

2.8 ASSIGNMENT

1. Explain Cursor for loop and Parameterised cursor with example
2. Define the explicit cursor
3. Implements the use of implicit cursor
4. Discuss various advantages of PL/SQL.
5. Define Cursor. Explain Cursor Cycle.
6. Discuss the types of cursor with proper syntax.
7. How do we use While Loop and For Loop in Cursor? Discuss with example.
8. Differentiate Cursor declared in a procedure and Cursor declared in a packageSpecification
9.What are PL/SQL cursor exceptions?
10. Write a PLSQL code to check whether a number is prime or not

2.9 ACTIVITIES

Write a note on various types of cursor

2.10 CASE STUDY

1. Explain the benefits of the cursor and benefit of cursor for loop
2. Explain Parameterised cursor
3. Implements the cursor on the database as requirements

2.11 FURTHER READING

1. An Introduction to Database Systems - Bipin Desai- Galgotia Publication.
2. Database Management System by Raghu Ramkrishnan- Tata McGraw-Hill Publication.
3. SQL, PL/SQL: The Programming Language Oracle - Ivan Bayross- BPB Publication.

Unit 3: Locking Strategy 3

Unit Structure

3.0 LEARNING OBJECTIVES

After learning this unit, you will be able to understand:

- Concept of Locking
- Types of locking
- Use of locking
- Implementation of locking

3.1 IMPLICIT LOCKING

Locking Strategies

The database maintains different types of locks based on the operation that hold the lock. Locks have direct impact on the interaction of read and write operation. The following rules summarize the locking behaviour of oracle database for reads and writes:

• A row is locked whenever modified by a write operation. When a transaction updates one row, the transaction acquires a lock for this row only. The contention can be minimized by locking table data at the row level.

• When one transaction is updating a row, then a row lock prevents a different transaction from updating the same row concurrently.

• A read operation never blocks a write operation. A reading of a row does not lock that row; a write operation can update this row. The only exception is a SELECT ... FOR UPDATE statement that will lock the row being read.

• A write operation never blocks a read operation. When a row is being changed by a write transaction, the database applies undo data to provide readers with a consistent view of the row data.

Locks in PL/SQL are an approach to maintain data integrity of the database. As in oracle is a multi-user platform where tables used in a database acts as a global resource being shared by multiple users at the same time.There is a possibility that the data may become inconsistent due to concurrent processing of data by multiple users at the same time on the database. Therefore, locks play an important role to maintain concurrency control ensuring data integrity of stored data in the database.

Scope of locks

The amount of data locked by a statement can vary.

Table locks

The statement can lock the entire table. Table-level locking systems always lock entire tables.

Row-level locking systems can lock entire tables if the WHERE clause of a statement cannot use an index. For example, UPDATES that cannot use an index lock the entire table.

Row-level locking systems can lock entire tables if a high number of single-row locks would be less efficient than a single table-level lock. Choosing table-level locking instead of row-level locking for performance reasons is called lock escalation. For more information about this topic, see "About the system's selection of lock granularity" and "Transaction-based lock escalation" in Tuning Java DB.

Single-row locks

A statement can lock only a single row at a time.

For row-level locking systems:

- For TRANSACTION_REPEATABLE_READ isolation, the locks are released at the end of the transaction.
- For TRANSACTION_READ_COMMITTED isolation, Derby locks rows only as the application steps through the rows in the result. The current row is locked. The row lock is released when the application goes to the next row.
- For TRANSACTION_SERIALIZABLE isolation, however, Derby locks the whole set before the application begins stepping through.
- For TRANSACTION_READ_UNCOMMITTED, no row locks are requested.

Derby locks single rows for INSERT statements, holding each row until the transaction is committed. If there is an index associated with the table, the previous key is also locked.

Range locks

A statement can lock a range of rows (range lock).

For row-level locking systems:

- For any isolation level, Derby locks all the rows in the result plus an entire range of rows for updates or deletes.
- For the TRANSACTION_SERIALIZABLE isolation level, Derby locks all the rows in the result plus an entire range of rows in the table for SELECTs to prevent no repeatable reads and phantoms.

For example, if a SELECT statement specifies rows in the Employee table where the salary is BETWEEN two values, the system can lock more than just the actual rows it returns in the result. It also must lock the entire range of rows between those two values to prevent another transaction from inserting, deleting, or updating a row within that range.

An index must be available for a range lock. If one is not available, Derby locks the entire table.

In implicit locking, oracle engine automatically locks the below types of approach for applies the locks

1. Type of lock to be applied.
2. Level of lock to be applied.

1. Type of lock to be applied

Using this type of operation (read or write) to be performed on a database, there are two types of locks.

Shared Lock	**1.** Applied on a table when data is being viewed example, the stored data is just being read. **2.** Multiple shared locks can be applied on the table at the same time. **3.** Used in case of SELECT statements.
Exclusive Lock	**1.** Applied on a table when the data is being changed example, the data is being written. **2.** Only one exclusive lock can be positioned at

	one time. 3. Used in INSERT, UPDATE, DELETE statements.

2. Level of lock to be applied:

There are basis of three levels of locking:

Row Level: It is used when a condition is applied in a query on a single row (or record) using WHERE clause.

Page Level: It is used when the condition is applied in a query on a certain set of data (certain records) using WHERE clause.

Table Level: It is used when the condition is applied in a query on the entire table of data (certain records) using WHERE clause.

Check your progress 1

1. Locks in PL/SQL are an approach to maintain data

A. Security

B. Integrity

C. Availability

D. Modification

2. The Record are locked using

A. Field Level locking

B. Row Level locking

C. Table Level locking

D. None

3. Table-level locking systems always lock entire tables.

A. True

B. false

3.2 EXPLICIT LOCKING

What is an "explicit" table lock?

Oracle DDL commands are required to hold exclusive locks on internal structures. If these internal locks are unavailable when DDL commands are issued, the consequence will be that the DDL commands will fail even though

they may have succeeded if they had been sent only a few seconds later. The WAIT option allows a DDL command to wait for its locks for a specified period of time before failing.

The lock table command has new syntax that lets the DBA specify the maximum number of seconds the statement should wait to obtain a DML lock on the table. For instance, the following Oracle syntax to perform these options would be as follows:

LOCK TABLE Ã¢â,¬Â¦ IN {lock_mode} MODE [NOWAIT | WAIT {integer_value}

If the database return control is desired without a wait period, use the NOWAIT clause for the lock table command.

In the following example, there is a table that the DBA does not want to lock for any longer than necessary.

SQL> LOCK TABLE LAB.NOWAITING IN EXCLUSIVE MODE NOWAIT;

Output

Table(s) Locked.

Use the WAIT clause to allow Oracle to lock tables with the lock table statement so that it will wait for the number of seconds in the command to acquire the DML lock. In the event that one does not specify an option for NOWAIT or WAIT, then Oracle will wait forever until the table is available, lock the object, and finally return control back to the user.When the database is executing DDL statements concurrently with DML statements, timeouts and deadlocks may occur from time to time on rare occasions. Oracle has an automatic detection mechanism for such timeouts and deadlocks. If a deadlock or deadly embrace condition occurs, an error message is written to

the alert log and Oracle returns the error message for the deadlock. As will soon be covered, there are some ways to resolve these conditions.

This is a user-defined locking done on a database by the user according to their granted table privileges. Explicit Locking can be done by using one of the following ways:

Select...For Update statement

SELECT * FROM <tablename> WHERE <condition> For Update No wait;

Suppose two clients, client SOM and client TOM are performing the transactions on the same database table employee_detail

Following is the query run by Client SOM:

SELECT * from employeet_detail where city='Ahmedabad' For Update;

When the above SELECT statement is executed, the oracle engine automatically locks the record with city as Ahmedabad and this lock can be released only when Client SOM will execute COMMIT or ROLLBACK command.

Now let's see the query run by Client TOM:

SELECT * from employee_detail where city='Ahmedabad' For Update;

Now the client TOM executes the SELECT statement for the same record that has been already locked by the oracle engine for Client SOM. So in this case the client TOM has to wait till the client SOM release the lock by executing the COMMIT statement

Therefore, to overcome the unnecessary waiting time NOWAIT option can be used to inform the oracle engine that the record has already been locked and it can terminate the statement.

WAIT indicates that the oracle engine will wait till the resource is freely available. Where NOWAIT indicates that the oracle engine will not wait for

resource to be available and display the message to the user that Resource is Busy.

Check your progress2

1. Exclusive Lock refer for

A. open for all

B. open for one user only

C. open for two user only

D. open for reading only

2. The Record are locked using

A. Field Level locking

B. Row Level locking

C. Table Level locking

D. None

3. Table-level locking systems always lock entire tables.

A. True

B. false

3.3 LOCK TABLE

How LOCK TABLE Statement works in Oracle?

- When we use LOCK Table statement and after it gets executed the database overrides the manual automatic locking available in Oracle and permits or denies other users to view or update the table for a specified time as mentioned in the statement.
- The permission actually depends on the type of mode of Lock that the user has chosen.
- In case suppose the user has chosen the mode EXCLUSIVE then other users can only use queries on that table and the database will not allow any update on that table from other users.
- In case it is ROW SHARE mode then the database will not allow the user to lock the whole table and will allow concurrent access to the table.

- The duration of time is also specified in the statement or queries itself.
- So, it actually depend on the type of mode u provide to the database and depending on that the database will lock your table or tables.

We can also acquire lock on a table by executing the Lock table statement. Following is the syntax:

LOCK TABLE <tablename>**in lock mode <WAIT/NOWAIT>;**

Lock mode can be one of the following:

EXCLUSIVE: allow the queries on the locked table.
SHARE: allow queries but restricts *UPDATE* on a table.
ROW EXCLUSIVE: allow concurrent access to the table by multiple users but restricts from locking table in exclusive or share mode.
SHARE ROW EXCLUSIVE: allow viewing the entire table records but restricts locking the table in share mode and also restricts *UPDATE* on a table.

Examples of Oracle LOCK TABLE

Practical examples of the Locking

Example 1:

EXCLUSIVE MODE WITH NO WAIT

Here we will look at the Exclusive mode of operation. We will lock the table employee in the Exclusive mode with NO WAIT which means that it will not wait if another user has already locked the table.

Code:

```
LOCK TABLE employee
IN EXCLUSIVE MODE NOWAIT;
```

Output:

Lock Succeeded

Example 2:

SHARE MODE WITH NO WAIT.

In this case, we will slightly change the situation we will issue a lock on the employee table present in the database in SHARE MODE with NO WAIT. One important point to note is that once the share lock is obtained, exclusive locks cannot be obtained. NO WAIT means that it will not wait for a lock to be released.

Code:

```
LOCK TABLE employee
IN SHARE MODE NOWAIT;
```

Output:

Lock Succeeded

Example 3

EXCLUSIVE MODE WITH WAIT

In this example, we will look at the Exclusive mode of operation. We will lock the table employee in the Exclusive mode with WAIT which means that the database will wait until the table is available and then lock the table.

Code:

LOCK TABLE employee
IN EXCLUSIVE MODE WAIT 5;

In the above query, the wait time is mentioned as 5 seconds.

Output:

Lock Succeeded

Check your progress 3

1. Shared locking mean

A. open for all in read mode

B. open for all in write mode

C. open for one user only

D. all of these

2. The Record are locked using

A. Field Level locking

B. Row Level locking

C. Table Level locking

D None

3. ROW EXCLUSIVE mode are used for multiple user in write mode

A. True

B. false

3.4 LET US SUM UP

This unit gives unique learning on different types of locking with implicit and explicit locking. Also lock table maintain the exclusive and shared locking. Oracle Database provides data concurrency, consistency and integrity among transactions through a locking mechanism. The locks are performed automatically and require no user

interaction. It is directly associated with a session. Database Locks are mechanisms that prevent destructive interaction between transactions accessing the shared resource or objects. These resources can be tables, data rows, data blocks, cached items, connections and entire systems.

There are many types of locks that can occur such shared locks, exclusive locks, transaction locks, DML locks, and backup-recovery locks. Oracle database automatically obtains required locks when performing SQL transactions. For example, before a session is permitted to update data, the session must first lock the data. The lock empowers the session exclusive control over the data so that no other transaction can update the locked data until the lock is released.

Oracle Database always performs locking automatically to ensure data concurrency, data integrity, and statement-level read consistency. However, you can override the Oracle default locking mechanisms. This can be useful in situations such as the following:

When your application requires consistent data for the duration of the transaction, not reflecting changes by other transactions, you can achieve transaction-level read consistency by using explicit locking, read-only transactions, serializable transactions, or by overriding default locking.

When your application requires that a transaction have exclusive access to a resource so that the transaction does not have to wait for other transactions to complete, you can explicitly lock the data for the duration of the transaction.

Automatic Locks in DML Operations that the purpose of a DML lock, also called a data lock, is to guarantee the integrity of data being accessed concurrently by multiple users. For example, a DML lock can prevent multiple customers from buying the last copy of a book available from an online bookseller. DML locks prevent destructive interference of simultaneous conflicting DML or DDL operations.DML statements automatically acquire locks at both the table level and the row level. In the sections that follow, the acronym in parentheses after each type of lock or lock mode is the abbreviation used in the Locks Monitor of Oracle Enterprise Manager. Enterprise Manager might display "TM" for any table lock, rather than indicate the mode of table lock (such as RS or SRX).

Check your progress 4

1. DML lock is also called

A. Field lock

B. Data lock

C. Record lock

D. Database lock

2. Database will wait until the table is available and then lock the table.

A. True

B. False

3.Oracle database automatically obtains required locks when performing SQL transactions

A. True

B. False

3.5 ANSWER FOR CHECK YOUR PROGRESS

Check your progress 1

Answers: (1-b), (2-b), (3-a)

Check your progress 2

Answers: (1-b), (2-b), (3-a)

Check your progress 3

Answers: (1-a), (2-b), (3-a)

Check your progress 4

Answers: (1-b), (2-a), (3-a)

3.6 GLOSSARY

Implicit locking -As in oracle is a multi-user platform where tables used in a database acts as a global resource being shared by multiple users at the same time. There is a possibility that the data may become inconsistent due to concurrent processing of data by multiple users at the same time on the database. Therefore, locks play an important

role to maintain concurrency control ensuring data integrity of stored data in the database.

Explicit locking -This is a user-defined locking done on a database by the user according to their granted table privileges.

Lock Table - We can also acquire lock on a table by executing the Lock tableStatement

Table Name - It refers to the name of the table.

Lock Mode - It refers to the mode on which we are going to lock the table. There are many modes.

ROW_SHARE - This mode allows concurrent access to the table in which it is applied. Here concurrent access means that it allows multiple users to access the table simultaneously and the users who are accessing the table are not allowed to lock the entire table which means no exclusive access.

ROW_EXCLUSIVE - This mode is just like the above allows concurrent access to the table. It means multiple users can access it simultaneously and the users who are accessing the table are not allowed to lock the entire table. The difference of this mode with ROW_SHARE is that it also does not allow locking in share mode.

SHARE UPDATE - It is similar like ROW_SHARE. It allows concurrent access to the table. It allows multiple user access to the table. It does not allow users to lock the entire table.

SHARE - It is similar to SHARE UPDATE. It allows concurrent access to the table. It allows multiple user access to the table. It does not allow users to lock the entire table. It also prohibits updates to the locked table.

SHARE ROW EXCLUSIVE - This mode allows Users to view records in the table but it does not allow the users to update the table or from locking the table in SHARE mode.

EXCLUSIVE - This mode allows only queries to be executed on the Locked table. No other activities are allowed other than that.

WAIT - This keyword when used allows the database to wait until the table is available and then lock the table and returns the control to the user.

NOWAIT - This condition refers to the condition when the database does not wait for the lock to be released. This is useful when the user wants the database to return control to him/her immediately.

3.7 ASSIGNMENT

1. Explain different types of locks applied by the programmer requirements
2. When we use shard and exclusive locking
3. Explain about Lock table importance
4. Explain difference between Wait and Nowait
5. Explain difference between Share and Share Update

3.8 ACTIVITIES

Write a note on various locking techniques.

3.9 CASE STUDY

Explain the different locking schemes.
\

3.10 FURTHER READING

1. Database System Concepts bySilberschatz, Korth -Tata McGraw-Hill Publication.
2. Database Management System by Raghu Ramkrishnan- Tata McGraw-Hill Publication.
3. SQL, PL/SQL: The Programming Language Oracle - Ivan Bayross- BPB Publication.

Unit 4: Exception Handling

4

Unit Structure

4.0 LEARNING OBJECTIVES

After learning this unit, you will be able to understand:

- Meaning of Exception
- Use of Exception
- Implementation of Exception

4.1 EXCEPTION HANDLING

What is PL/SQL Exceptions?

PL/SQL treats all errors that occur in an anonymous block, procedure, or function as exceptions. The exceptions can have different causes such as coding mistakes, bugs, even hardware failures. It is not possible to anticipate all potential exceptions, however, you can write code to handle exceptions to enable the program to continue running as normal. The code that you write to handle exceptions is called an exception handler. A PL/SQL block can have an exception-handling section, which can have one or more exception handlers.

There are three types of exceptions:

- Predefined exceptions are error conditions that are defined by PL/SQL.
- Non-predefined exceptions include any standard TimesTen errors.
- User-defined exceptions are exceptions specific to your application.

In PL/SQL, a warning or error condition is called an exception. Exceptions can be internally defined (by the run-time system) or user defined. Examples of internally defined exceptions include division by zero and out of memory. Some common internal exceptions have predefined names, such as ZERO_DIVIDE and STORAGE_ERROR. The other internal exceptions can be given names.

You can define exceptions of your own in the declarative part of any PL/SQL block, subprogram, or package. For example, you might define an exception named insufficient funds to flag overdrawn bank accounts. Unlike internal exceptions, user-defined exceptions must be given names.

When an error occurs, an exception is raised. That is, normal execution stops and control transfers to the exception-handling part of your PL/SQL block or subprogram. Internal exceptions are raised implicitly (automatically) by the run-time system. User-defined exceptions must be raised explicitly by RAISE statements, which can also raise predefined exceptions.

To handle raised exceptions, you write separate routines called exception handlers. After an exception handler runs, the current block stops executing and the enclosing block resumes with the next statement. If there is no enclosing block, control returns to the host environment.

In the example below, you calculate and store a price-to-earnings ratio for a company with ticker symbol ABC. If the company has zero earnings, the predefined exception ZERO_DIVIDE is raised. This stops normal execution of the block and transfers control to the exception handlers. The optional OTHERS handler catches all exceptions that the block does not name specifically.

An exception is an error which disrupts the normal flow of program instructions. It is also called error condition during a program execution. PL/SQL supports programmers to catch such conditions using EXCEPTION block in the program and an appropriate action is taken against the error condition. There are two types of exceptions User defined exception and System defined exceptions

Syntax to write an exception

```
WHEN exception THEN
Statement;
```

```
DECLARE
declarations section;
BEGIN
executable command(s);
EXCEPTION
WHEN exception1 THEN
statement1;
```

WHEN exception2 THEN
statement2;
[WHEN others THEN]
END;

When an exception occurs in the executable section, the execution of the current block stops and control transfers to the exception-handling section if the exception e1 occurred, the exception_handler1 runs. If the exception e2 occurred, the exception_handler2 executes. In case any other exception rises, then the other_exception_handler runs.

After an exception handler executes, control transfers to the next statement of the enclosing block. If there is no enclosing block, then the control returns to the invoker if the exception handler is in a subprogram or host environment (SQL Developer or SQL*Plus) if the exception handler is in an anonymous block.

The following is a listing of Oracle Error Messages:
ORA-00001ORA-00018ORA-00020ORA-00023ORA-00028ORA-00034

Check your progress 1

1. How many types of exception are there in PL/SQL?
A. 1
B. 2
C. 3
D. 4
2. Exception raised when an arithmetic
A. ZERO_DIVIDE
B. VALUE_ERROR
C. TOO_MANY_ROWS
D. SELF_IS_NULL

3. ZERO_DIVIDE exception is used for checking divide by zero

A. True

B. false

4.2 PREDEFINED EXCEPTIONS

PL/SQL provides number of pre-defined exceptions, which are executed when any database rule is violated by a program. For example, the predefined exception ZERO_DIVIDE It is raised when an attempt is made to divide a number by zero. The following table lists few of the important pre-defined exceptions.

ZERO_DIVIDE	It is raised when an attempt is made to divide a number by zero.
LOGIN_DENIED	It is raised when a program attempts to log on to the database with an invalid username or password.
NOT_LOGGED_ON	It is raised when a database call is issued without being connected to the database.
NO_DATA_FOUND	It is raised when a SELECT INTO statement returns no rows.
INVALID_NUMBER	It is raised when the conversion of a character string into a number fails because the string does not represent a valid number.
PROGRAM_ERROR	It is raised when PL/SQL has an internal problem.
VALUE_ERROR	It is raised when an arithmetic, conversion, truncation, or size constraint error occurs.
ACCESS_INTO_NULL	It is raised when a null object is automatically assigned a value.
COLLECTION_IS_NULL	It is raised when a program attempts to apply collection methods other than EXISTS to an uninitialized nested table or array, or the program attempts to assign values to the elements of an uninitialized nested table or array.

DUP_VAL_ON_INDEX	It is raised when duplicate values are attempted to be stored in a column with unique index.
INVALID_CURSOR	You tried to reference a cursor that does not yet exist. This may have happened because you've executed a FETCH cursor or CLOSE cursor before OPENing the cursor.
ROWTYPE_MISMATCH	It is raised when a cursor fetches value in a variable having incompatible data type.
TOO_MANY_ROWS	It is raised when a SELECT INTO statement returns more than one row.
STORAGE_ERROR	It is raised when PL/SQL ran out of memory or memory was corrupted.
SELF_IS_NULL	It is raised when a member method is invoked, but the instance of the object type was not initialized.
CASE_NOT_FOUND	It is raised when none of the choices in the WHEN clause of a CASE statement is selected, and there is no ELSE clause.
CURSOR_ALREADY_OPEN	You tried to open a cursor that is already open.
TRANSACTION_BACKED_OUT	The remote portion of a transaction has rolled back.

Syntax of the pre-defined of the exception

We will take a look at the syntax for Named System Exceptions in both procedures and functions.

The syntax for the Named System Exception in a procedure is:

```
CREATE [OR REPLACE] PROCEDURE <procedure_name>
[ (parameter [,parameter]) ]
IS
   [declaration_section]
```

```
BEGIN
executable_section

EXCEPTION
  WHEN exception_name1 THEN
    [statements]

  WHEN exception_name2 THEN
    [statements]

  WHEN exception_name_n THEN
    [statements]

  WHEN OTHERS THEN
    [statements]

END [procedure_name];
```

Syntax for Functions

The syntax for the Named System Exception in a function is:

```
CREATE [OR REPLACE] FUNCTION function_name
[ (parameter [,parameter]) ]
  RETURN return_datatype
IS | AS
  [declaration_section]

BEGIN
executable_section
EXCEPTION
```

```
  WHEN exception_name1 THEN
    [statements]

  WHEN exception_name2 THEN
    [statements]

  WHEN exception_name_n THEN
    [statements]

  WHEN OTHERS THEN
    [statements]

END [function_name];
```

Example 1:

For example here ZERO_DIVIDE = raises exception WHEN dividing with zero

```
DECLARE
  a int:=10;
  b int:=0;
  answer int;

BEGIN
  answer:=a/b;
  dbms_output.put_line('the output after division is'||answer);

EXCEPTION

  WHEN zero_divide THEN
    dbms_output.put_line('dividing by zero error check the denominator');
```

```
    dbms_output.put_line('the value of a is '||a);
    dbms_output.put_line('the value of b is '||b);
END;
```

Output:

```
dividing by zero error check the denominator
the value of a is 10
the value of b is 0
```

Example 2:

VALUE_ERROR:This error is raised WHEN a statement is executed that resulted in an arithmetic, numeric, string, conversion, or constraint error. This error mainly results from programmer error or invalid data input.

```
DECLARE
  temp no;

BEGIN
  SELECT g_name  into temp from geeks where g_name='SOMj';
  dbms_output.put_line('the g_name is '||temp);

EXCEPTION
  WHEN value_error THEN
  dbms_output.put_line('Error occurred');
  dbms_output.put_line('Change data type of temp to varchar(20)');

END;
```

Output:

```
Error occurred
```

Change data type of temp to varchar(20)

Check your progress 2

1. Which statements can be checked for handling errors.

A. DDL

B. DML

C. DCL

D. TCL

2. Following is a global variable for error handling

A. "@@ERRORS"

B. "@ERRORS"

C. "@@ERR"

D. "@@ERRORS"

3.. Exception handling is possible in SQL Server using throw exception

A. True

B. false

4.3 USER DEFINED EXCEPTIONS

What is User defined Exception?

User-defined exception is a custom exception created and throws that exception using a keyword 'throw'. It is done by extending a class 'Exception'. An exception is a problem that arises during the execution of the program.

PL/SQL allows you to define your own user defined exceptions according to the requirements of the program. A user-defined exception must be declared and then raised explicitly, using either a RAISE statement or the procedure DBMS_STANDARD.RAISE_APPLICATION_ERROR.

Syntax of user defined exception

DECLARE

user_define_exception_name EXCEPTION;

```
BEGIN
Statement(s);
   IF condition THEN
      RAISE user_define_exception_name;
END IF;
EXCEPTION
   WHEN user_define_exception_name THEN
      User defined statement (action) will be taken;
END;
```

Practical example of the user defined exception:

```
QL>edit user_exception
DECLARE
   myException EXCEPTION;
i NUMBER;
BEGIN
   FOR i IN (SELECT * FROM enum) LOOP
      IF i.eno = 10 THEN
         RAISE myException;
      END IF;
   END LOOP;
EXCEPTION
   WHEN myException THEN
      dbms_output.put.line ('Employee number already exists in enum table.');
END;
/
OUTPUT
SQL>@user_ exception
Employee number already exists in enum table.
```

PL/SQL procedure successfully operation

Check your progress3

1. User-defined exception is a custom exception created using _____a keyword

A. Exception

B. throw

C. raise

D. none

2. Which is implicit cursor attribute

A. %rowcount

B. %rowtype

C. %notfound

D. none

3. User define exception is a custom defined exception

A. True

B. False

4.3 USER DEFINED EXCEPTIONS

What is User defined Exception?

User-defined exception is a custom exception created and throws that exception using a keyword 'throw'. It is done by extending a class 'Exception'. An exception is a problem that arises during the execution of the program.

PL/SQL allows you to define your own user defined exceptions according to the requirements of the program. A user-defined exception must be declared and then raised explicitly, using either a RAISE statement or the procedure DBMS_STANDARD.RAISE_APPLICATION_ERROR.

Syntax of user defined exception

DECLARE

user_define_exception_name EXCEPTION;

```
BEGIN
Statement(s);
   IF condition THEN
      RAISE user_define_exception_name;
END IF;
EXCEPTION
   WHEN user_define_exception_name THEN
      User defined statement (action) will be taken;
END;
```

Practical example of the user defined exception:

```
QL>edit user_exception
DECLARE
   myException EXCEPTION;
i NUMBER;
BEGIN
   FOR i IN (SELECT * FROM enum) LOOP
      IF i.eno = 10 THEN
         RAISE myException;
      END IF;
   END LOOP;
EXCEPTION
   WHEN myException THEN
      dbms_output.put.line ('Employee number already exists in enum table.');
END;
/
```

OUTPUT

```
SQL>@user_ exception
Employee number already exists in enum table.
```

PL/SQL procedure successfully operation

Check your progress 3

1. User-defined exception is a custom exception created using _____a keyword

A. Exception

B. throw

C. raise

D. none

2. Which is implicit cursor attribute

A. %rowcount

B. %rowtype

C. %notfound

D. none

3. User define exception is a custom defined exception

A. True

B. False

4.4 HANDLING RAISED EXCEPTIONS

Exceptions are raised by the database server automatically whenever internal database error occurred, but also exceptions can be raised explicitly by the programmer by using the command RAISE. Following is the simple syntax for raising an exception. The RAISE statement stops normal execution of a PL/SQL block or subprogram and transfers control to an exception handler. RAISE statements can raise predefined exceptions, such as ZERO_DIVIDE or NO_DATA_FOUND, or user-defined exceptions whose names you decide.

How we can handle exceptions raised in Declare section?

The exception section of a PL/SQL block can only possibly handle an exception raised in the executable section. An exception raised in the declaration section (in an attempt

to assign a default value to a variable or constant) always propagates out unhandled to the enclosing block

Following is the simple syntax for raising an exception

```
DECLARE
   exception_name EXCEPTION;
BEGIN
IF condition THEN
RAISE exception_name;
   END IF;
EXCEPTION
WHENexception_nameTHEN
Statement;
END;
```

Syntax Explanation:

- In the above syntax, the keyword RAISE is used in the exception handling block.
- Whenever program encounters exception “exception_name”, the exception is handled and will be completed normally
- But the keyword ‘RAISE’ in the exception handling part will propagate this particular exception to the parent program.
- In function, an exception should always either return value or raise the exception further. Else Oracle will throw ‘Function returned without a value’ error at run-time.
- Transaction control statements can be given at exception handling block.
- SQLERRM and SQLCODE are the in-built functions that will give the exception message and code.
- If an exception is not handled then by default all the active transaction in that session will be rolled back.

• RAISE_APPLICATION_ERROR(-<error_code>, <error_message>) can be used instead of RAISE to raise the error with user code and message. Error code should be greater than 20000 and prefixed with '-'.

Check your progress 4

1. Exceptions are raised by the database server

A. automatically

B. Internally

C. raise

D. none

2. A user-defined exception must be declared and then _____

A. Raised explicitly

B. Raised implicitly

C. Raised internally

D. Raised externally

3. The RAISE statement stops normal execution of a PL/SQL block

A. True

B. false

4.5 LET US SUM UP

This unit gives the concept of the different types of exceptions, used by the programmer For the handling errors, with system defined or its own defined exception, for handling errors.

PL/SQL exception categories: PL/SQL has three exception categories:

Internally defined exceptions are errors which arise from the Oracle Database environment. The runtime system raises the internally defined exceptions automatically. ORA-27102 (out of memory) is one example of internally defined exceptions. Note that internally defined exceptions do not have names, but an error code.

Predefined exceptions are errors which occur during the execution of the program. The predefined exceptions are internally defined exceptions that PL/SQL has given names e.g., NO_DATA_FOUND, TOO_MANY_ROWS.

User-defined exceptions are custom exception defined by users like you. User-defined exceptions must be raised explicitly.

From above code we can conclude that exception handling

1. Improves readability by letting us isolate error-handling routines and thus providing robustness.

2. Provides reliability, instead of checking for different types of errors at every point we can simply write them in exception block and IF error exists exception will be raised thus helping the programmer to find out the type of error and eventually resolve it.

Uses: One of the real lIFeuse of exception can be found in online train reservation system.

While filling the station code to book the ticket IF we input wrong code it shows us the exception that the code doesn't exist in database.

4.6 ANSWERS FOR CHECK YOUR PROGRESS

Check your progress 1

Answers: (1-b), (2-b), (3-a)

Check your progress 2

Answers: (1-b), (2-a), (3-a)

Check your progress 3

Answers: (1-a), (2-c), (3-a)

Check your progress 4

Answers: (1-a), (2-a), (3-a)

4.7 GLOSSARY

1. Exception - An exception is an error which disrupts the normal flow of program instructions. It is also called error condition during a program execution

2.RAISE - Exceptions are raised by the database server automatically whenever internal database error occurred, but also exceptions can be raised explicitly by the programmer by using the command RAISE

3. System Defined Exceptions - PL/SQL provides number of pre-defined exceptions, which are executed when any database rule is violated by a program

4. User Defined Exception - User-defined exception is a custom exception created and throws that exception using a keyword 'throw'. It is done by extending a class 'Exception'. An exception is a problem that arises during the execution of the program.

5. Raised Exception - An internal exception is raised automatically if your PL/SQL program violates an Oracle rule or exceeds a system-dependent limit. PL/SQL predefines some common Oracle errors as exceptions. For example, PL/SQL raises the predefined exception NO_DATA_FOUND if a SELECT INTO statement returns no rows.

6. INVALID_NUMBER Exception - It is raised when the conversion of a character string into a number fails because the string does not represent a valid number.

7. ZERO_DIVIDE - It is raised when an attempt is made to divide a number by zero.

8. NO_DATA_FOUND - It is raised when a SELECT INTO statement returns no rows.

9. LOGIN_DENIED - It is raised when a program attempts to log on to the database with an invalid username or password.

4.8 ASSIGNMENT

a. Explain Different types of Exceptions

b. What is the use of RAISE in exception?

c. Explain any five system defined exceptions

4.9 ACTIVITIES

Write a note on practical example of the different types of exceptions

4.10 CASE STUDY

1. Explain the use of the different types of exceptions.
2. Defined the exceptions on the employee database.

3. Implement user defined exception.

4. Define Raise exception

5. Define different types of system defined functions on employee database

6. Define User defined exceptions on employee database

4.11 FURTHER READING

1. Database Management System by Raghu Ramkrishnan- Tata McGraw-Hill Publication.

2. SQL, PL/SQL: The Programming Language Oracle - Ivan Bayross- BPB Publication.

BAOU
Education for All

યુનિવર્સિટી ગીત

સ્વાધ્યાયઃ પરમં તપઃ
સ્વાધ્યાયઃ પરમં તપઃ
સ્વાધ્યાયઃ પરમં તપઃ

શિક્ષણ, સંસ્કૃતિ, સદ્‌ભાવ, દિવ્યબોધનું ધામ
ડૉ. બાબાસાહેબ આંબેડકર ઓપન યુનિવર્સિટી નામ;
સૌને સૌની પાંખ મળે, ને સૌને સૌનું આભ,
દશે દિશામાં સ્મિત વહે હો દશે દિશે શુભ-લાભ.

અભણ રહી અજ્ઞાનના શાને, અંધકારને પીવો ?
કહે બુદ્ધ આંબેડકર કહે, તું થા તારો દીવો;
શારદીય અજવાળા પહોંચ્યાં ગુર્જર ગામે ગામ
ધ્રુવ તારકની જેમ ઝળહળે એકલવ્યની શાન.

સરસ્વતીના મયૂર તમારે ફળિયે આવી ગહેકે
અંધકારને હડસેલીને ઉજાસના ફૂલ મહેંકે;
બંધન નહીં કો સ્થાન સમયના જવું ન ઘરથી દૂર
ઘર આવી મા હરે શારદા દૈન્ય તિમિરના પૂર.

સંસ્કારોની સુગંધ મહેંકે, મન મંદિરને ધામે
સુખની ટપાલ પહોંચે સૌને પોતાને સરનામે;
સમાજ કેરે દરિયે હાંકી શિક્ષણ કેરું વહાણ,
આવો કરીયે આપણ સૌ
ભવ્ય રાષ્ટ્ર નિર્માણ...
દિવ્ય રાષ્ટ્ર નિર્માણ...
ભવ્ય રાષ્ટ્ર નિર્માણ

●

DR. BABASAHEB AMBEDKAR OPEN UNIVERSITY
(Established by Government of Gujarat)
'Jyotirmay' Parisar,
Sarkhej-Gandhinagar Highway, Chharodi, Ahmedabad-382 481
Website : www.baou.edu.in | Email : office.scs@baou.edu.in