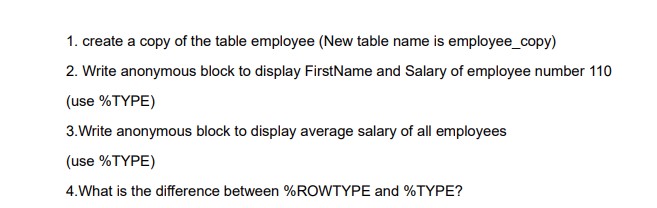
**Program Code CSD 4203**

**Student Name : Shreejana Shrestha**

**Student ID : c0930321**

**Assignment 12**

**# Question Assignment 1**



**Concept of Assignment**

In the context of the PL/SQL project, working with anonymous blocks, %TYPE, and %ROWTYPE has helped me better understand how to handle and manage data. Through this project, you can learn more about:

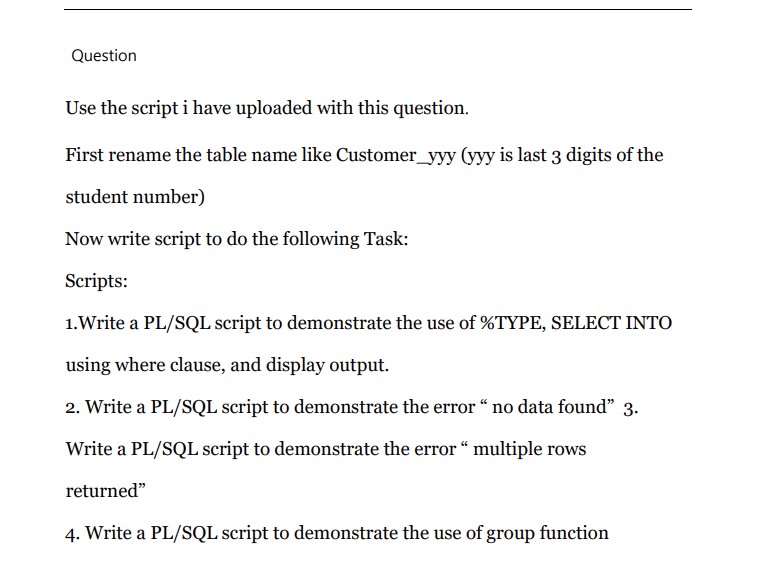
Blocks without names: By developing and executing anonymous blocks, I get experience executing PL/SQL code in real-time without the need to create stored procedures or functions. This helps me understand the best way to structure PL/SQL code for testing and immediate execution.

%TYPE: By going through the cases when you define variables using %TYPE, I learn how to ensure that your variables have the correct data types, matching the columns of the database tables. This is necessary to maintain consistency between your PL/SQL code and database schema and to avoid type-related errors.

%ROWTYPE: I discover how to use %ROWTYPE to manage entire data rows as a single record. This is useful for managing complex queries that call for you to manipulate multiple columns from a table at once. It ensures that your code maintains consistent with the table layout and facilitates handling data rows.

To sum up, this project has helped me understand how to use PL/SQL to query and modify data effectively while preserving type safety and optimizing the handling of table rows.

**# Question Assignment 2**



**Concept of Assignment**

I now grasp the following core ideas after completing this PL/SQL assignment:

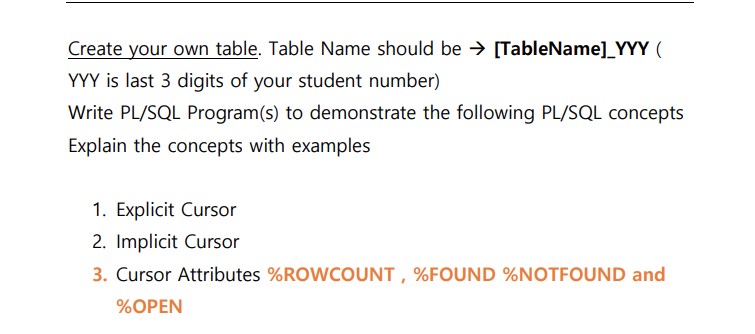
Understanding %TYPE: I now know how to use %TYPE to ensure that the PL/SQL variables and the data types of the table columns match. This promotes uniformity and lessens the chance of data type inconsistencies.

Error Handling: I learned how to handle common exceptions like "No Data Found" and "Multiple Rows Returned" from this part. Exception handling strategies need to be used to deal with situations where queries do not produce the expected results.

Working with Aggregate Functions: I am aware of how to assess and condense data sets using group functions, producing totals and averages, for example. This is necessary in order to compile data and draw meaningful conclusions.

All in all, this project has helped me better understand how to efficiently collect, handle, and process data in PL/SQL, which has helped me tackle a range of data-related situations.

**# Question Assignment 3**



**Concept of Assignment**

I feel like I now have a solid understanding of several key PL/SQL concepts and know how to apply them in practical settings after finishing this assignment:

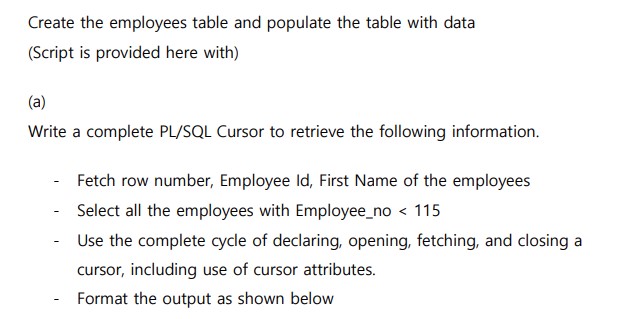
This course on explicit cursors teaches you how to declare, open, fetch from, and close cursors directly. With this experience, I can handle different result sets and examine data row by row, enabling me to accurately tackle complex data retrieval tasks.

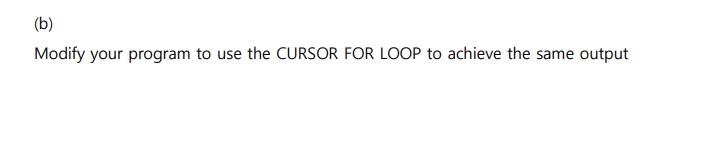
I now understand implicit cursors and how they enable PL/SQL to automatically execute fundamental SQL operations. With this information, I won't have to deal with cursor management alone, which will make it easier for me to handle simple queries and adjustments.

Cursor Attributes: I learn how to use cursor attributes like %ROWCOUNT, %FOUND, %NOTFOUND, and %OPEN to obtain important information about the state of the cursor and the results of SQL operations. This makes it easier to oversee the flow of my PL/SQL code, making it easier to handle different data scenarios and ensuring that my operations are carried out correctly.

All things considered, this assignment has improved my ability to manage and use cursors in PL/SQL, which has aided me in controlling flow and handling data in my PL/SQL programs.

**# Question Number 4**



**Concept of Assignment**

After doing the assignment on PL/SQL cursors, I now know the following:

Cursor Lifecycle Management: This section has taught me how to manage the declaration, opening, retrieval, and shutdown of a cursor. I am learning how to effectively organize and manage data retrieval procedures with this approach.

Data Retrieval and Processing: By using a cursor to select individual rows from a result set, I have learned how to retrieve and process data one row at a time. This is useful for handling complex queries that call for sequential data processing.

Application of Cursor Attributes: I need to become familiar with the following cursor attributes: %ROWCOUNT, %FOUND, %NOTFOUND, and %OPEN. These features have improved the management and control of data retrieval methods by providing me with useful information about the cursor's state and the results of your actions.

Formatting of Output: I am skilled at organizing and presenting the outcomes of various actions, ensuring that the data is displayed in a comprehensible and systematic manner.

When I modify my application to include a CURSOR FOR LOOP, I gain an advantage.

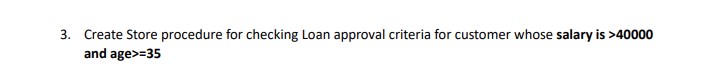
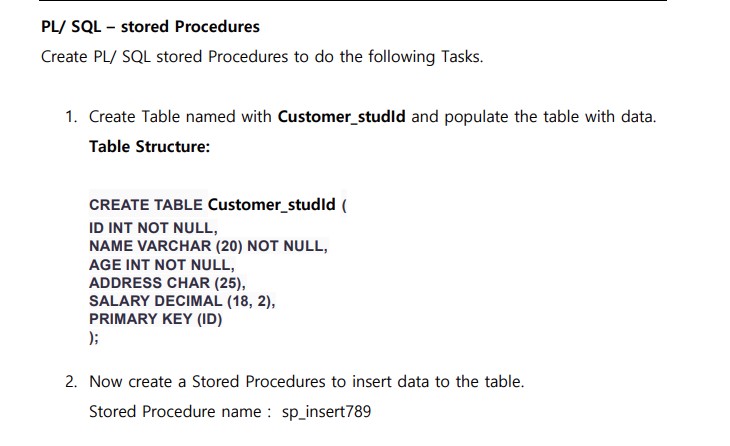
The CURSOR FOR LOOP simplifies cursor operations and reduces error risk by automatically handling cursor definition, opening, fetching, and shutting.

Processing Data Effectively: It effortlessly manages each row in the result set, giving me up to focus on the data itself instead of fumbling with the position of the mouse all the time.

Streamlined Code: The loop pattern makes data retrieval easier to understand and maintain by handling it in a more concise and readable manner.

All things considered, this assignment strengthens my skills in PL/SQL data processing and cursor management, equipping me with the knowledge to handle a range of data retrieval situations and ensure effective and efficient data handling.

**# Question Number 5**



**Concept of Assignment**

Procedure for Data Insertion: I have learned how to automate repetitive tasks by creating a stored procedure (sp\_insert789) that is used to insert data. This reduces the requirement for human work by assuring consistent data entry and boosting productivity.

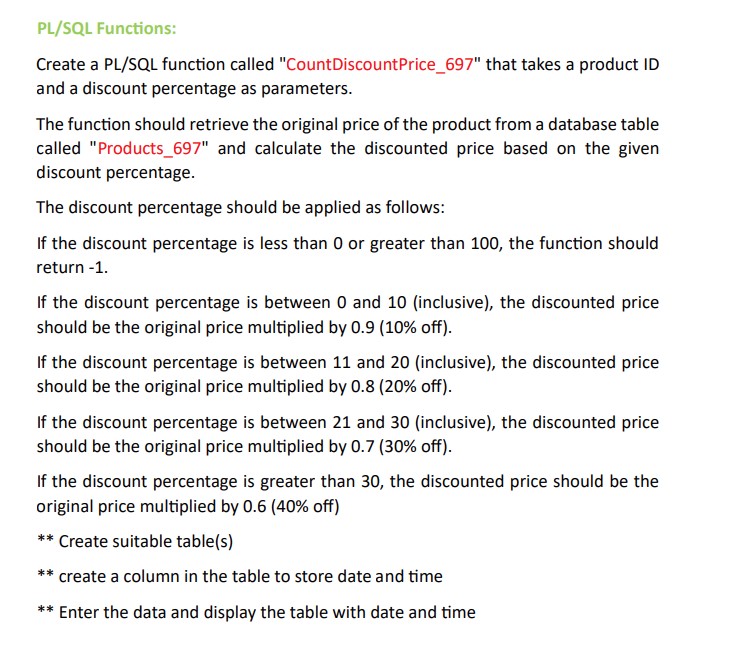
Procedure for Loan Approval: Establishing a documented process to assess a borrower's loan eligibility based on parameters such as age and income This educates me on applying business rules programmatically. This example shows how to add logic to database functions so that data can be used to inform decisions.

Table Design for Databases: I now know how to set up and manage tables for effective data storage and administration.

Automation with Stored Procedures: This course has taught me how to use stored procedures to automate data entry and business logic, which can improve consistency and streamline complex operations.

Putting Business Rules into Practice: I've been able to enhance decision-making and operational performance by integrating and utilizing certain business criteria in the database.

**#Question Assignment 6**



**Concept of Assignment**

Building Tables: Establish a Table, Define a table called Products\_325 to hold product data. Add columns for the product ID, starting price, and data entry timestamp.

In PL/SQL, function:

Creation of Functions: Make a PL/SQL function called CountDiscountPrice\_325 that computes a discounted price using a given discount percentage after retrieving the original price from the Products\_325 table. The function handles different discount ranges and edge scenarios.

Information Management:

Add Information: When adding sample product data to the Products 325 table, make sure to include timestamps.

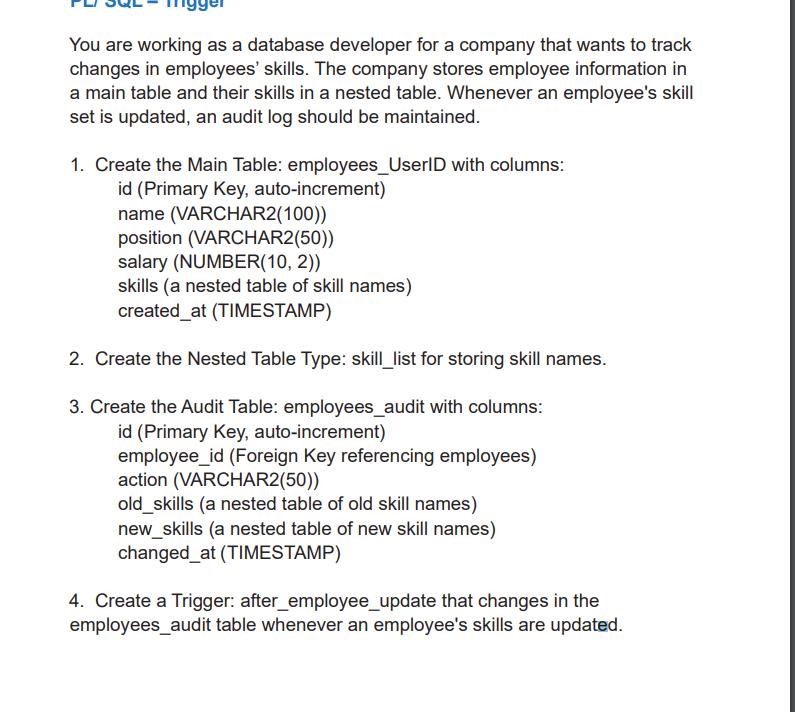
Show Information: Obtain and display the table data together with the timestamps to ensure accurate data insertion.

Design of Database Tables:

Table Structure: Recognize how to create and construct tables containing timestamps and product information. This entails deciding what kinds of data will be kept and how best to handle records.

Data Management: Data Insertion and Retrieval: Acquire the skills necessary to see and insert data in an organized manner. This will help you manage and retrieve product information from databases.

**# Question Assignment 7**



**Concept of Assignment**

Database table design:

Primary database Make a database called employees\_UserID to store important employee data like an ID, name, position, pay, skills, and creation timestamp. This table also includes a nested table to store skill names, which helps to illustrate the complex relationship that occurs between employees and their skills.

Nested Table Type:

List of abilities: Define a nested table of type skill\_list in order to manage and store lists of skill names associated with each employee. This makes it possible for the main employee table to hold data in a hierarchical and flexible way.

Audit Table:

Audit Log: To keep track of changes made to employees' skill sets, create an employees\_audit table. This entails recording the action taken, the timestamp of the adjustment, and both new and old talents. This audit trail is used to keep a historical record of skill modifications.

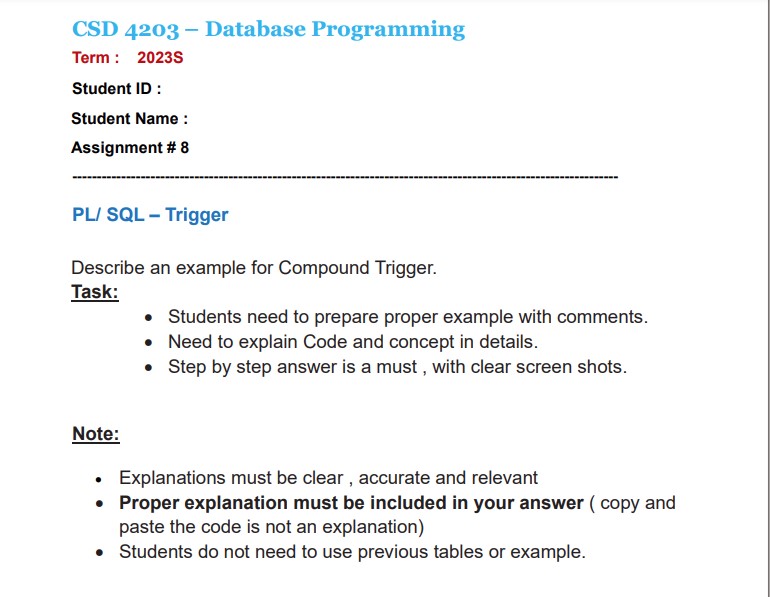
Trigger for Implementation: Track changes: To automatically record adjustments to an employee's skill set in the employees\_audit table, create a trigger named after\_employee\_update. This ensures that all changes are tracked and logged automatically, without the need for human intervention.

Discover how to build and manage tables with nested data structures, which illustrate the connections that exist in the real world between things like workers and their abilities. We call this sophisticated data modeling.

Historical Tracking: Learn how to set up audit procedures to maintain a historical record of modifications in order to keep track of changes and ensure data integrity over time.

Learn how to use triggers to automatically update audit logs and handle data changes for automated change management. By ensuring that adjustments are regularly tracked, this will save time and effort.

**# Question Assignment 8**



**Concept of Assignment**

PL/SQL compound triggers are used to handle complex scenarios involving many types of database actions (such INSERT, UPDATE, and DELETE) on a single table. It makes things more efficient and improves consistency by integrating several timing points into one trigger.

Key Concepts of Unified Management:

Handling a Single Trigger: By merging many activities (BEFORE, AFTER) into a single trigger, this method reduces the need for separate triggers and streamlines management.

Issues with Table Mutations:

Error Avoidance: This solution solves issues that come up when querying or editing the same table with traditional triggers by segmenting logic into multiple steps.

State Preserving:

Handling Data Consistently: Maintaining consistency in processing by keeping data in its original state during different stages of data changes.

Benefits

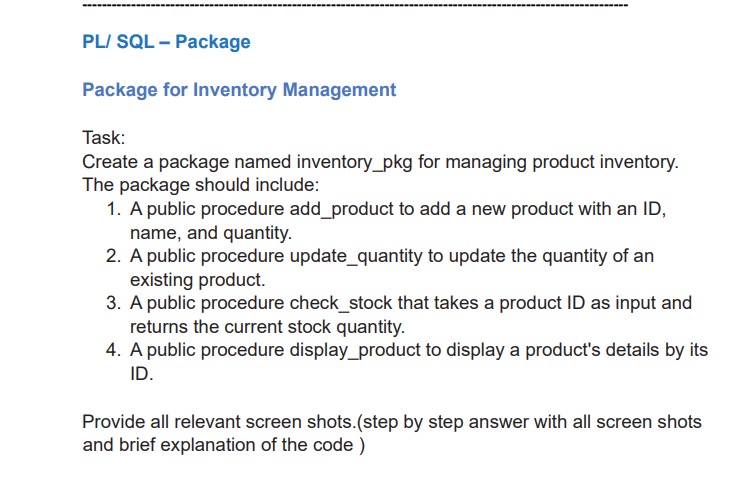
Simplified Design: To cut down on complexity, multiple operations are combined into a single trigger.

Ensures that data modifications are handled uniformly.

Efficiency: Reduces expenses and streamlines procedures to improve performance.

In summary, compound triggers help avoid common trigger management issues and streamline the handling of intricate data processing procedures.

**#Question Assignment 9**



**Concept of Assignment**

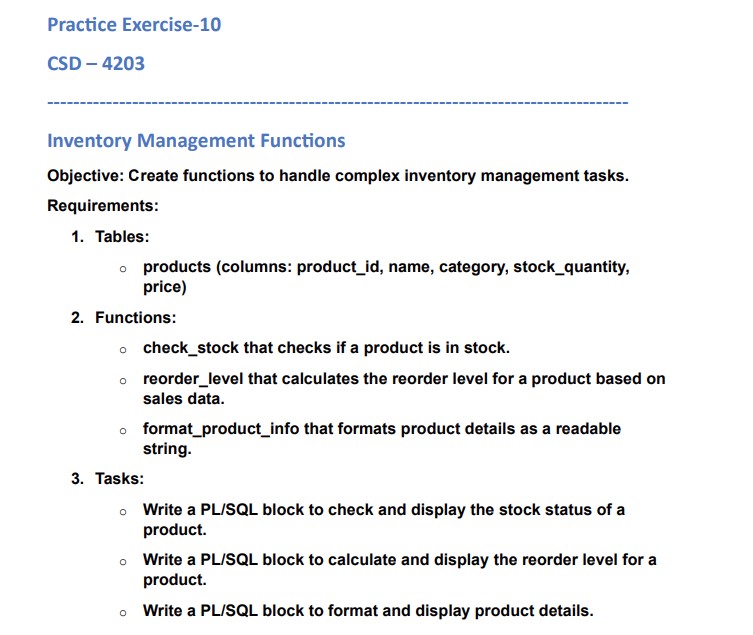
The creation of the inventory\_pkg PL/SQL package, which will be utilized to efficiently manage product inventory, is the primary objective of this task. Organizing and maintaining code is made easier in PL/SQL by combining related functions and procedures into packages.

The major goal is to create a package body and specification. Other program components can access the public functions and procedures that are stated in the package specification, which acts as an interface. This includes how to add new products, change how much of them to buy, figure out how much stock is available, and display product details. Inserting new products is done with add\_product; updating stock levels is done with update\_quantity; retrieving the stock quantity is done with check\_stock; and product information is shown with display\_product based on its ID. Every technique has a unique objective.

The package body contains the implementation details for these processes. It performs the actual logic required to do these tasks, such as getting product specifications or adding information to the inventory table.

By completing this project, I will get understanding on how to organize and encapsulate code using PL/SQL packages. Developing my ability to design and implement clear interfaces for my processes will help me be more capable of managing and maintaining complex database operations. The benefits of modular design, which unifies related functions into a single package to simplify code management and improve maintainability, are also demonstrated by this study. Taking everything into account, this assignment provides valuable knowledge on creating reusable and orderly PL/SQL code for inventory control.

**#Question Assignment 10**



**Concept of Assignment**

Uses: In PL/SQL, functions are designed to perform certain calculations or data manipulations and return the result. In this assignment, functions are used to automate and simplify challenging inventory management tasks.

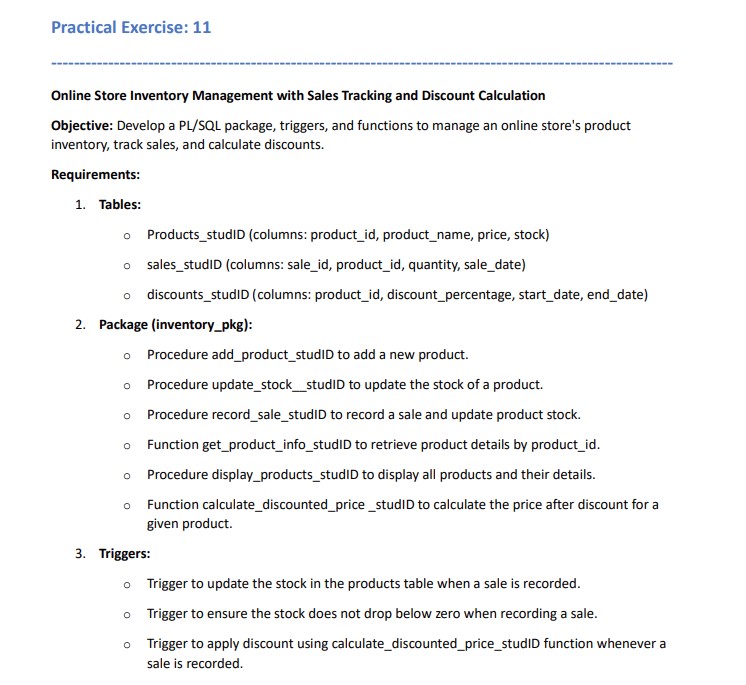
Advantages of Modular Architecture: Because distinct functions are allocated to different tasks (such formatting product descriptions, figuring out reorder levels, and checking stock), the code is modular and simple to maintain. Each function improves clarity and maintainability by attending to a different criterion.

Data Administration: Effectiveness: Functions that directly work with data stored in tables enable data processing and retrieval. They support the maintenance of correct and consistent handling of inventory data.

User Interaction: Text Readability: Formatting features improve the presentation of product information, making it easier for people to obtain and understand.

This assignment concludes by demonstrating how to use PL/SQL functions appropriately for job management with inventories. It involves establishing capabilities to assess stock levels, determine reorder points, and format product information in order to enhance inventory management and display data more effectively.

**#Question Assignment 11**



**Concept of Assignment**

Design and Management of Databases: Idea: To manage and store product, sales, and discount data, one must have a strong grasp of table design and implementation.

Learning: Get an understanding of how to effectively organize data and preserve data integrity.

Package Development in PL/SQL: Concept: Learn how to use and group related functions and procedures for sales and inventory management into packages.

Learning: Develop your knowledge of modular programming and reusable code management.

Method and Goal of Execution: Concept: The ability to create functions and procedures that do tasks like adding products, calculating discounts, and recording sales.

Use of Trigger: Concept: Triggers can be used to start automated procedures like changing stock levels and applying discounts.

Learning Objective: Understand how to use triggers to maintain data consistency and enforce business requirements.

Business Logic Integration:

Concept: Integrating business logic into the database through functions, triggers, and procedures.

Learning Objective: Learn how to effectively handle real-world business requirements in a database system.

All things considered, you will have a solid understanding of how to track sales, manage inventory, and handle discounts using PL/SQL after completing this assignment. It enhances one's skills in database design, business logic implementation, and package building.