CSD 4203 – Database Programming

Student ID : C0930321

Student Name : Shreejana Shrestha

Assignment # 12

Question Assignment 1

1. create a copy of the table employee (New table name is employee_copy)
2. Write anonymous block to display FirstName and Salary of employee number 110 (use %TYPE)
3. Write anonymous block to display average salary of all employees (use %TYPE)
4. What is the difference between %ROWTYPE and %TYPE?

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

Question

Use the script i have uploaded with this question.

First rename the table name like Customer_yyy (yyy is last 3 digits of the student number)

Now write script to do the following Task:

Scripts:

1.Write a PL/SQL script to demonstrate the use of %TYPE, SELECT INTO using where clause, and display output.

2. Write a PL/SQL script to demonstrate the error " no data found" 3. Write a PL/SQL script to demonstrate the error " multiple rows returned"

4. Write a PL/SQL script to demonstrate the use of group function

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

Create your own table. Table Name should be → **[TableName]_YYY** (
YYY is last 3 digits of your student number)
Write PL/SQL Program(s) to demonstrate the following PL/SQL concepts
Explain the concepts with examples

1. Explicit Cursor
2. Implicit Cursor
3. Cursor Attributes %ROWCOUNT , %FOUND %NOTFOUND and %OPEN

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

Create the employees table and populate the table with data

(Script is provided here with)

(a)

Write a complete PL/SQL Cursor to retrieve the following information.

- Fetch row number, Employee Id, First Name of the employees
- Select all the employees with Employee_no < 115
- Use the complete cycle of declaring, opening, fetching, and closing a cursor, including use of cursor attributes.
- Format the output as shown below

(b)

Modify your program to use the CURSOR FOR LOOP to achieve the same output

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

PL/ SQL – stored Procedures

Create PL/ SQL stored Procedures to do the following Tasks.

1. Create Table named with **Customer_studId** and populate the table with data.
   Table Structure:

```
CREATE TABLE Customer_studId (
ID INT NOT NULL,
NAME VARCHAR (20) NOT NULL,
AGE INT NOT NULL,
ADDRESS CHAR (25),
SALARY DECIMAL (18, 2),
PRIMARY KEY (ID)
);
```

2. Now create a Stored Procedures to insert data to the table.
   Stored Procedure name : sp_insert789

3. Create Store procedure for checking Loan approval criteria for customer whose **salary is >40000 and age>=35**

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

PL/SQL Functions:

Create a PL/SQL function called "CountDiscountPrice_697" that takes a product ID and a discount percentage as parameters.

The function should retrieve the original price of the product from a database table called "Products_697" and calculate the discounted price based on the given discount percentage.

The discount percentage should be applied as follows:

If the discount percentage is less than 0 or greater than 100, the function should return -1.

If the discount percentage is between 0 and 10 (inclusive), the discounted price should be the original price multiplied by 0.9 (10% off).

If the discount percentage is between 11 and 20 (inclusive), the discounted price should be the original price multiplied by 0.8 (20% off).

If the discount percentage is between 21 and 30 (inclusive), the discounted price should be the original price multiplied by 0.7 (30% off).

If the discount percentage is greater than 30, the discounted price should be the original price multiplied by 0.6 (40% off)

** Create suitable table(s)

** create a column in the table to store date and time

** Enter the data and display the table with date and time

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

PL/ SQL – Trigger

You are working as a database developer for a company that wants to track changes in employees' skills. The company stores employee information in a main table and their skills in a nested table. Whenever an employee's skill set is updated, an audit log should be maintained.

1. Create the Main Table: employees_UserID with columns:
   id (Primary Key, auto-increment)
   name (VARCHAR2(100))
   position (VARCHAR2(50))
   salary (NUMBER(10, 2))
   skills (a nested table of skill names)
   created_at (TIMESTAMP)

2. Create the Nested Table Type: skill_list for storing skill names.

3. Create the Audit Table: employees_audit with columns:
   id (Primary Key, auto-increment)
   employee_id (Foreign Key referencing employees)
   action (VARCHAR2(50))
   old_skills (a nested table of old skill names)
   new_skills (a nested table of new skill names)
   changed_at (TIMESTAMP)

4. Create a Trigger: after_employee_update that changes in the employees_audit table whenever an employee's skills are updated.

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.

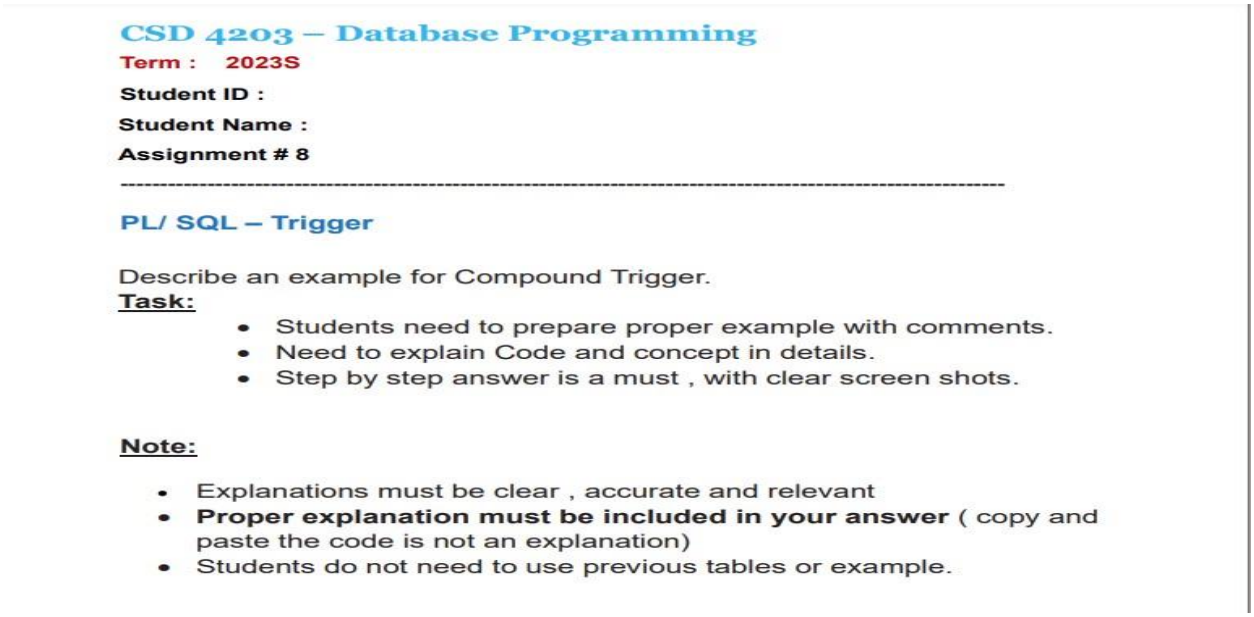
Trigger for Implementation: Track changes: To automatically record adjustments to an employee's skill set in the employees_audit table, create a trigger named after_employee_update. This ensures that all changes are tracked and logged automatically, without the need for human intervention.

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



CSD 4203 – Database Programming

Term : 2023S

Student ID :

Student Name :

Assignment # 8

PL/ SQL – Trigger

Describe an example for Compound Trigger.

Task:

- Students need to prepare proper example with comments.
- Need to explain Code and concept in details.
- Step by step answer is a must , with clear screen shots.

Note:

- Explanations must be clear , accurate and relevant
- **Proper explanation must be included in your answer** (copy and paste the code is not an explanation)
- Students do not need to use previous tables or example.

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

PL/ SQL – Package

Package for Inventory Management

Task:
Create a package named inventory_pkg for managing product inventory.
The package should include:

1. A public procedure add_product to add a new product with an ID, name, and quantity.
2. A public procedure update_quantity to update the quantity of an existing product.
3. A public procedure check_stock that takes a product ID as input and returns the current stock quantity.
4. A public procedure display_product to display a product's details by its ID.

Provide all relevant screen shots.(step by step answer with all screen shots and brief explanation of the code)

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

Practice Exercise-10

CSD – 4203

Inventory Management Functions

Objective: Create functions to handle complex inventory management tasks.

Requirements:

1. **Tables:**
   - **products (columns: product_id, name, category, stock_quantity, price)**
2. **Functions:**
   - **check_stock that checks if a product is in stock.**
   - **reorder_level that calculates the reorder level for a product based on sales data.**
   - **format_product_info that formats product details as a readable string.**
3. **Tasks:**
   - **Write a PL/SQL block to check and display the stock status of a product.**
   - **Write a PL/SQL block to calculate and display the reorder level for a product.**
   - **Write a PL/SQL block to format and display product details.**

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

Practical Exercise: 11

Online Store Inventory Management with Sales Tracking and Discount Calculation

Objective: Develop a PL/SQL package, triggers, and functions to manage an online store's product inventory, track sales, and calculate discounts.

Requirements:

1. **Tables:**
    - Products_studID (columns: product_id, product_name, price, stock)
    - sales_studID (columns: sale_id, product_id, quantity, sale_date)
    - discounts_studID (columns: product_id, discount_percentage, start_date, end_date)
2. **Package (inventory_pkg):**
    - Procedure add_product_studID to add a new product.
    - Procedure update_stock__studID to update the stock of a product.
    - Procedure record_sale_studID to record a sale and update product stock.
    - Function get_product_info_studID to retrieve product details by product_id.
    - Procedure display_products_studID to display all products and their details.
    - Function calculate_discounted_price _studID to calculate the price after discount for a given product.
3. **Triggers:**
    - Trigger to update the stock in the products table when a sale is recorded.
    - Trigger to ensure the stock does not drop below zero when recording a sale.
    - Trigger to apply discount using calculate_discounted_price_studID function whenever a sale is recorded.

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.