**Documentation and Demonstration Report**

**1. Introduction**

This report covers the design, implementation, and testing of key database components used in the project, including triggers, cursors, functions, packages, and auditing mechanisms. These features are integrated to ensure data integrity, enhance functionality, and improve security for the applicant management system.

**2. Design of Components**

**a. Triggers Triggers** are used to monitor and automatically act on changes to the PE\_Applicant table. Three primary triggers were designed:

* **After Insert Trigger**: Logs new applicants into an audit table for tracking changes.
* **Before Update Trigger**: Enforces business rules and validates data before updating records.
* **Before Delete Trigger**: Prevents deletion of certain applicants based on conditions.

**b. Cursors** Explicit cursors are implemented for scenarios requiring row-by-row processing:

* **Applicant Status Update**: A cursor is used to update the status of applicants who have not responded to notifications.
* **Inactive Applicants Cleanup**: Cursors help delete applicants who have been inactive for over a year.

**c. Functions** Functions are used to encapsulate reusable business logic, such as calculating the age of an applicant or formatting data for reports.

**d. Packages** Packages are designed to group related procedures, functions, and variables into a single, reusable unit. The ApplicantPackage includes:

* Procedures for updating applicant statuses.
* Functions for calculating applicant-related metrics.

**e. Auditing Mechanisms** Auditing is implemented to track changes to sensitive data, including updates to applicant status and personal information. The system logs changes with user accountability through audit triggers and an audit table. Key components include:

* **Insert/Update/Delete Logging**: Captures who, what, and when for every change made to applicant data.
* **Access Control**: Ensures that only authorized users can perform sensitive operations.

**3. Implementation of Components**

**a. Triggers Implementation**

1. **After Insert Trigger**:
   * Designed to log each new applicant's details into the PE\_Applicant\_Audit table.
   * Ensures transparency by capturing information about the insertion.

CREATE OR REPLACE TRIGGER after\_insert\_applicant

AFTER INSERT ON PE\_Applicant

FOR EACH ROW

BEGIN

INSERT INTO PE\_Applicant\_Audit (ApplicantID, FirstName, LastName, ChangeType, ChangeDate)

VALUES (:NEW.ApplicantID, :NEW.FirstName, :NEW.LastName, 'INSERT', SYSDATE);

END;

1. **Before Update Trigger**:
   * Ensures applicants' data is checked for consistency and validates conditions before any update.
   * Logs the changes made during the update.

* CREATE OR REPLACE TRIGGER before\_update\_applicant
* BEFORE UPDATE ON PE\_Applicant
* FOR EACH ROW
* BEGIN
* IF :NEW.Status != :OLD.Status THEN
* INSERT INTO PE\_Applicant\_Audit (ApplicantID, FirstName, LastName, ChangeType, ChangeDate)
* VALUES (:NEW.ApplicantID, :NEW.FirstName, :NEW.LastName, 'UPDATE', SYSDATE);
* END IF;
  + END;

**b. Cursors Implementation**

* **Cursor for Status Update**: This cursor loops through applicants with a 'Pending' status and updates their status if they have not responded.
* DECLARE
* CURSOR applicant\_cursor IS
* SELECT ApplicantID, FirstName, LastName
* FROM PE\_Applicant
* WHERE Status = 'Pending' AND NotificationResponse IS NULL;
* v\_applicant\_id PE\_Applicant.ApplicantID%TYPE;
* v\_first\_name PE\_Applicant.FirstName%TYPE;
* v\_last\_name PE\_Applicant.LastName%TYPE;
* BEGIN
* OPEN applicant\_cursor;
* LOOP
* FETCH applicant\_cursor INTO v\_applicant\_id, v\_first\_name, v\_last\_name;
* EXIT WHEN applicant\_cursor%NOTFOUND;
* UPDATE PE\_Applicant
* SET Status = 'Not Responded'
* WHERE ApplicantID = v\_applicant\_id;
* DBMS\_OUTPUT.PUT\_LINE('Updated status for: ' || v\_first\_name || ' ' || v\_last\_name);
* END LOOP;
* CLOSE applicant\_cursor;
* END;

**c. Functions Implementation**

1. **Age Calculation Function**: A function to calculate the age of an applicant based on their birthdate.
2. CREATE OR REPLACE FUNCTION calculate\_age(birthdate DATE) RETURN NUMBER IS
3. v\_age NUMBER;

* BEGIN
* SELECT FLOOR(MONTHS\_BETWEEN(SYSDATE, birthdate) / 12)
* INTO v\_age
* FROM dual;
* RETURN v\_age;
* END;

**d. Packages Implementation**

1. **Applicant Package**: Groups related logic for updating applicant statuses and calculating metrics.

* CREATE OR REPLACE PACKAGE ApplicantPackage AS
* PROCEDURE update\_status(p\_applicant\_id IN NUMBER, p\_status IN VARCHAR2);
* FUNCTION get\_applicant\_age(p\_applicant\_id IN NUMBER) RETURN NUMBER;
* END ApplicantPackage;
* CREATE OR REPLACE PACKAGE BODY ApplicantPackage AS
* PROCEDURE update\_status(p\_applicant\_id IN NUMBER, p\_status IN VARCHAR2) IS
* BEGIN
* UPDATE PE\_Applicant
* SET Status = p\_status
* WHERE ApplicantID = p\_applicant\_id;
* END update\_status;
* FUNCTION get\_applicant\_age(p\_applicant\_id IN NUMBER) RETURN NUMBER IS
* v\_birthdate PE\_Applicant.DateOfBirth%TYPE;
* BEGIN
* SELECT DateOfBirth INTO v\_birthdate FROM PE\_Applicant WHERE ApplicantID = p\_applicant\_id;
* RETURN calculate\_age(v\_birthdate);
* END get\_applicant\_age;
* END ApplicantPackage;
* **e. Auditing Mechanisms**
* **Audit Trigger**: A trigger that logs each change to the PE\_Applicant table into an audit table.
* CREATE OR REPLACE TRIGGER audit\_applicant\_changes
* AFTER INSERT OR UPDATE OR DELETE ON PE\_Applicant
* FOR EACH ROW
* BEGIN
* IF INSERTING THEN
* INSERT INTO PE\_Applicant\_Audit (ApplicantID, FirstName, LastName, ChangeType, ChangeDate)
* VALUES (:NEW.ApplicantID, :NEW.FirstName, :NEW.LastName, 'INSERT', SYSDATE);
* ELSIF UPDATING THEN
* INSERT INTO PE\_Applicant\_Audit (ApplicantID, FirstName, LastName, ChangeType, ChangeDate)
* VALUES (:NEW.ApplicantID, :NEW.FirstName, :NEW.LastName, 'UPDATE', SYSDATE);
* ELSIF DELETING THEN
* INSERT INTO PE\_Applicant\_Audit (ApplicantID, FirstName, LastName, ChangeType, ChangeDate)
* VALUES (:OLD.ApplicantID, :OLD.FirstName, :OLD.LastName, 'DELETE', SYSDATE);
* END IF;
* END;

**4. Testing and Demonstration**

**Testing Strategy**:

* **Trigger Testing**:
  + Test each trigger to ensure data is logged properly in the PE\_Applicant\_Audit table.
  + Verify that updates, inserts, and deletions to PE\_Applicant generate corresponding records in the audit table.
* **Cursor Testing**:
  + Verify that the cursor correctly fetches rows based on the specified condition and that operations on these rows (e.g., updating status) are successful.
  + Ensure that no rows are missed, and the cursor exits when no more data is available.
* **Function Testing**:
  + Verify that the calculate\_age function returns the correct age based on the applicant's birthdate.
  + Test the ApplicantPackage functions and procedures to ensure they behave as expected (updating status and calculating age).
* **Auditing Mechanism Testing**:
  + Ensure that all data changes (inserts, updates, and deletes) are correctly captured in the audit log with the correct timestamp, user, and action type.
  + Test with different user roles to ensure proper access control and restrictions.

**Demonstration**:

* A live demonstration can be conducted by showing how the system updates an applicant's status (e.g., from 'Pending' to 'Not Responded') and how changes are logged in real-time in the audit table.
* A user can be shown performing different operations like inserting a new applicant, updating their details, or deleting them, while the corresponding audit logs are displayed to demonstrate accountability.
* The cursor-based operations can be shown by selecting applicants with certain statuses or conditions, performing updates or deletions, and demonstrating the batch processing using cursors.

**5. Conclusion**

This system integrates advanced database features to enhance functionality, maintain data integrity, and improve security. The use of triggers, cursors, functions, packages, and auditing mechanisms addresses the project’s objectives by ensuring that operations on the PE\_Applicant table are efficiently managed, tracked, and secured.