Purpose and Overview

- **Script/Module Name**: `customer_pkg`
- **Summary**: This PL/SQL package manages customer-related operations within the system. It includes functionalities to add, retrieve, update, and delete customer records, as well as purge old customer data.

1. Procedure Name1: new_customer

1. Functional Overview - This function adds a new customer to the `xy_customer` table. **2. Procedure Input and Output Data** - **Inputs**: - `p_customer_name` (VARCHAR2): The name of the customer to be added. - **Outputs**: - Returns `customer_id` (NUMBER): The ID of the newly added customer.

2. Core Functionality

Processing Logic: - Inserts a new record into the `xy_customer` table with the given customer name. - Retrieves the `customer_id` of the newly inserted record.

Business Logic: 1. Insert the customer name into the `xy_customer` table. 2. Retrieve and return the `customer_id` of the new record.

Business Rules:

Condition	Description
Unique Customer Name	Ensures that the customer name is unique in the `xy_customer` table.

Data Validation: - Ensures that `p_customer_name` is not null or empty.

Error Codes and Messages:

Error Code	Description/Error Message
N/A	N/A

3. Integration Points:

Interfacing Systems: - System 1: CRM, e.g., updates customer records.

4. Final Output:

- The function returns the `customer_id` of the newly added customer.

---

1. Procedure Name2: get_customer

1. Functional Overview - This function retrieves a customer row from the `xy_customer` table based on the given `customer_id`. **2. Procedure Input and Output Data** - **Inputs**: - `p_customer_id` (NUMBER): The ID of the customer to be retrieved. - **Outputs**: - Returns `xy_customer%rowtype`: The row of the customer.

2. Core Functionality

Processing Logic: - Selects the customer row from the `xy_customer` table where `customer_id` matches the input parameter.

Business Logic: 1. Retrieve the customer row based on `customer_id`. 2. Return the customer row.

Business Rules:

Condition	Description
Valid Customer ID	Ensures that the `customer_id` exists in the `xy_customer` table.

Data Validation: - Ensures that `p_customer_id` is a valid number.

Error Codes and Messages:

Error Code	Description/Error Message
N/A	N/A

3. Integration Points:

Interfacing Systems: - System 1: CRM, e.g., updates customer records.

4. Final Output:

- The function returns the customer row.

---

1. Procedure Name3: get*customer*name

1. Functional Overview - This function retrieves the name of a customer from the `xy_customer` table based on the given `customer_id`. **2. Procedure Input and Output Data - Inputs**: - `p_customer_id` (NUMBER): The ID of the customer whose name is to be retrieved. - **Outputs**: - Returns `xy_customer.customer_name%type`: The name of the customer.

2. Core Functionality

Processing Logic: - Selects the customer name from the `xy_customer` table where `customer_id` matches the input parameter.

Business Logic: 1. Retrieve the customer name based on `customer_id`. 2. Return the customer name.

Business Rules: | Condition | Description | |-----------|-------------| | Valid Customer ID | Ensures that the `customer_id` exists in the `xy_customer` table. |

Data Validation: - Ensures that `p_customer_id` is a valid number.

Error Codes and Messages: | Error Code | Description/Error Message | |------------|---------------------------| | N/A | N/A |

3. Integration Points:

Interfacing Systems: - System 1: CRM, e.g., updates customer records.

4. Final Output:

- The function returns the customer name.

---

1. Procedure Name4: set_customer

1. Functional Overview - This procedure updates the name of a customer in the `xy_customer` table based on the given `customer_id`. **2. Procedure Input and Output Data - Inputs**: - `p_customer_id` (NUMBER): The ID of the customer to be updated. - `p_customer_name` (VARCHAR2): The new name of the customer. - **Outputs**: - N/A

2. Core Functionality

Processing Logic: - Updates the customer name in the `xy_customer` table where `customer_id` matches the input parameter.

Business Logic: 1. Update the customer name based on `customer_id`.

Business Rules: | Condition | Description | |-----------|-------------| | Valid Customer ID | Ensures that the `customer_id` exists in the `xy_customer` table. | | Valid Customer Name | Ensures that the `p_customer_name` is not null or empty. |

Data Validation: - Ensures that `p_customer_id` is a valid number. - Ensures that `p_customer_name` is not null or empty.

Error Codes and Messages: | Error Code | Description/Error Message | |------------|---------------------------| | N/A | N/A |

3. Integration Points:

Interfacing Systems: - System 1: CRM, e.g., updates customer records.

4. Final Output:

- N/A

---

1. Procedure Name5: set_customer (row-based)

1. Functional Overview - This procedure updates a customer row in the `xy_customer` table based on the given row data. **2. Procedure Input and Output Data** - **Inputs**: - `p_row` (xy_customer%rowtype): The row data of the customer to be updated. - **Outputs**: - N/A

2. Core Functionality

Processing Logic: - Updates the customer row in the `xy_customer` table where `customer_id` matches the `customer_id` in the input row.

Business Logic: 1. Update the customer row based on `customer_id`.

Business Rules: | Condition | Description | |-----------|-------------| | Valid Customer ID | Ensures that the `customer_id` exists in the `xy_customer` table. |

Data Validation: - Ensures that `p_row.customer_id` is a valid number.

Error Codes and Messages: | Error Code | Description/Error Message | |------------|---------------------------| | N/A | N/A |

3. Integration Points:

Interfacing Systems: - System 1: CRM, e.g., updates customer records.

4. Final Output:

- N/A

---

1. Procedure Name6: delete_customer

1. Functional Overview - This procedure deletes a customer from the `xy_customer` table based on the given `customer_id`. **2. Procedure Input and Output Data** - **Inputs**: - `p_customer_id` (NUMBER): The ID of the customer to be deleted. - **Outputs**: - N/A

2. Core Functionality

Processing Logic: - Deletes the customer row from the `xy_customer` table where `customer_id` matches the input parameter.

Business Logic: 1. Delete the customer based on `customer_id`.

Business Rules: | Condition | Description | |-----------|-------------| | Valid Customer ID | Ensures that the `customer_id` exists in the `xy_customer` table. |

Data Validation: - Ensures that `p_customer_id` is a valid number.

Error Codes and Messages: | Error Code | Description/Error Message | |------------|---------------------------| | N/A | N/A |

3. Integration Points:

Interfacing Systems: - System 1: CRM, e.g., updates customer records.

4. Final Output:

- N/A

---

1. Procedure Name7: purge*old*customers

1. Functional Overview - This procedure purges old customer data from the `xy_customer` table based on the given `since_date` and optionally deletes the audit trail. **2. Procedure Input and Output Data** - **Inputs**: - `p_since_date` (DATE): The cutoff date for purging old customer data. - `p_delete_audit_trail` (BOOLEAN): Flag to indicate whether to delete the audit trail. - **Outputs**: - N/A

2. Core Functionality

Processing Logic: - Deletes customer rows from the `xy_customer` table where `last_active_date` is less than or equal to `p_since_date`. - Optionally deletes the audit trail if `p_delete_audit_trail` is true.

Business Logic: 1. Delete old customer data based on `since_date`. 2. Optionally delete the audit trail.

Business Rules: | Condition | Description | |-----------|-------------| | Valid Since Date | Ensures that `p_since_date` is a valid date. | | Valid Audit Trail Flag | Ensures that `p_delete_audit_trail` is a boolean value. |

Data Validation: - Ensures that `p_since_date` is a valid date. - Ensures that `p_delete_audit_trail` is a boolean value.

Error Codes and Messages: | Error Code | Description/Error Message | |------------|---------------------------| | N/A | N/A |

3. Integration Points:

Interfacing Systems: - System 1: CRM, e.g., updates customer records.

4. Final Output:

- N/A

E-N-D