MASTER IN INFORMATION TECHNOLOGY

MIT 263
SYSTEM ANALYSIS AND DATABASE DESIGN

ALDEN A. QUIÑONES
Student

VINCE MARC SABADO, MSIT
Professor

FEBRUARY 23, 2025

1. Introduction

1.1 **Project Overview** – Briefly describe the chosen problem and its relevance.

The social pension for itinerant senior citizens is a government initiative designed to address the needs of elderly individuals, particularly those in vulnerable sectors. This program focuses on providing financial support to senior citizens who have not had the opportunity to benefit from programs like the Social Security System (SSS) or the Government Service Insurance System (GSIS). It aims to ensure that even the most marginalized and economically disadvantaged seniors, especially those without formal pension coverage, receive the assistance they need to live with dignity and security in their later years.

Meanwhile, the process of disbursing social pensions remains entirely manual, with no technological systems in place to streamline or expedite the procedure. As a result, disbursement officers frequently encounter significant challenges, leading to delays and sometimes the failure to complete disbursements within the designated time frame. One major issue arises during the identity verification process, which often takes longer than expected, particularly when beneficiaries lack the necessary documentation. In some cases, disbursing officers also face difficulties in verifying the legitimacy of individuals attempting to claim benefits despite not being included on the official list. These challenges are inherent in the current manual system and are unlikely to improve without the integration of technology to automate and simplify the process.

In addition to the problems mentioned, grievances are also increasing due to various reasons, such as those who did not receive grants or received but not enough. There are also beneficiaries who receive double which is very worrying for the agency because it greatly affects, not only the overall accomplishment of the agency but also to the thrust of the public to the agency. The disbursing officers will also have difficulty in liquidating the payroll especially since the COA is focusing on program operation.

To solve these problems, the agency needs a centralized database system with strict policies (database contains and roles). through this, the process will be accelerated, the integrity of the data will be protected, and possible problems will be mitigated in the present.

1.2 **Objectives** – State the main goals of the database design.

- To ensure data integrity and accuracy, preventing issues like double payments or errors in beneficiary information.
- Security and privacy are paramount, with strict access controls in place to protect sensitive data in line with data protection regulations.
- To prioritize efficiency and speed, optimizing processes for quicker identity verification and disbursement, reducing delays.
- To accommodate future growth as the program expands, while maintaining auditability and transparency to ensure all transactions are traceable and accountable.

1.3 **Scope and Limitations** – Define the coverage of the database and its constraints.

This database design is only a limited storage of social pension beneficiary data of dswd including payroll management. This system does not cover beneficiaries who are mapped outside of Region XII. This system also does not cover senior citizens who do not belong to the Social Pension for Indigent Senior Citizens (SPISC) program. This system also does not cover the processing of fund sources by the Department of Budget and Management (DBM), only important information such as the Special Allotment Release Order (SARO) and Sub-allotment Advice (SAA).

---

2. Problem Statement and Requirements Analysis

2.1 **Problem Description** – Explain the issue that requires a database solution.

- The manual process of disbursing social pensions leads to delays and inefficiencies in meeting the needs of senior citizens, particularly affecting vulnerable beneficiaries.
- Identity verification challenges, including missing documentation and improper claims, hinder the timely distribution of social pensions to eligible seniors.
- Increasing grievances from beneficiaries, such as insufficient or double payments, undermine the credibility of the social pension program.
- The absence of a technological system for managing pension distribution makes it difficult for disbursing officers to accurately liquidate payrolls and ensure compliance with government regulations.
- Without a centralized database system, maintaining data integrity and preventing errors in pension distribution remains a significant challenge, affecting both beneficiaries and the agency's operations.

2.2 **Business Rules and Assumptions** – Define rules governing data management.

A. BUSINESS RULES:

- The system administrator able to assign appropriate access roles to the registered end-users once the user already created their accounts
- The end-users will be able to access the system once they make the registration in the system.
- All beneficiaries should be aware that they can access information from the system over the internet
- The system should be able to filter-out all entries in the encoding module because of the strict validation of the forms.
- The end-user should be able to generate reports.

B. ASSUMPTIONS

- It is assumed that the system administrator already have profiling of all employee along with their responsibilities and the end-users to
- It is assumed that the end-users already have capacities on how to access the system.
- It is assumed that not all-beneficiaries know how to access the system especially physically impaired beneficiaries

- It is assumed that all incomplete information will be filtered out by the system but there might be possibilities that the fraudulent data pass the form validation.

2.3 **Functional and Non-Functional Requirements** – List required database functionalities.

1. **Functional Requirements:**

    1. The system must allow for the registration and eligibility verification of senior citizens.
    2. The system must generate automated disbursement lists and ensure correct pension amounts.
    3. The system should provide role-based access control for different user types to secure sensitive data.

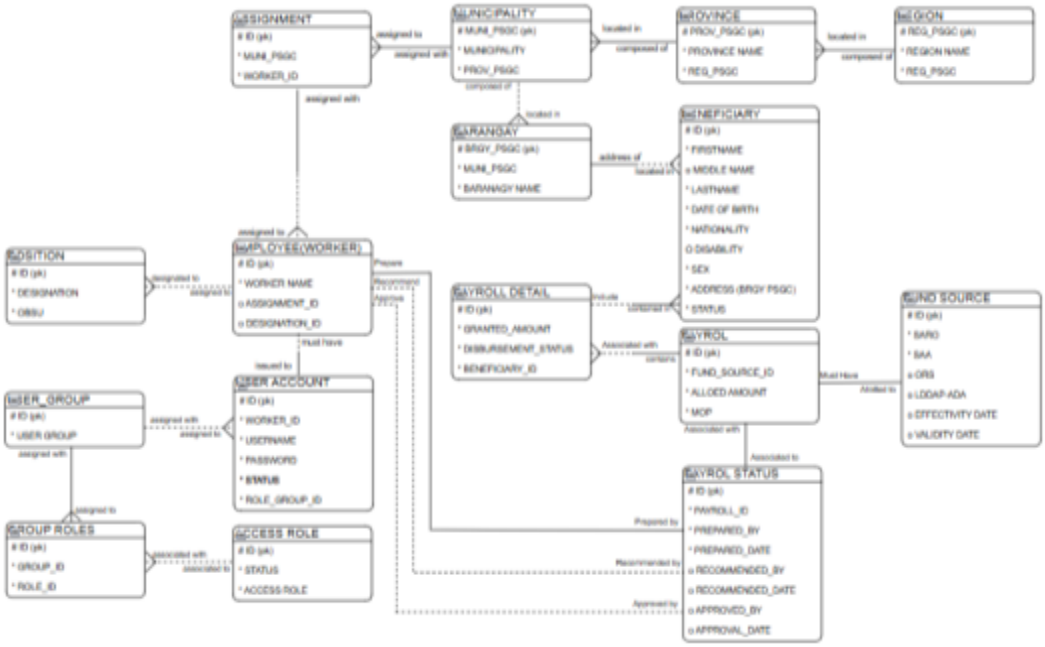
2. **Non-Functional Requirements:**

    1. The system must scale to accommodate increasing beneficiary numbers without degrading performance.
    2. The system should maintain 100% uptime and have a backup system to prevent data loss.
    3. The system must have an intuitive, user-friendly interface for both officers and beneficiaries.
    4. The system must use strong encryption and comply with data protection regulations to safeguard beneficiary information.
    5. The system must comply with relevant legal and regulatory requirements for pensions and financial transactions.

---

3. Conceptual and Logical Database Design

3.1 **Entity-Relationship Diagram (ERD)** – Include a **screenshot** of the first versions of ERD modeled .



3.2 **Entities and Attributes Definition** – Describe entities, attributes, and primary keys.

ENTRIES:

- ACCESS_ROLE
- ASSIGNMENT
- BARANGAY
- BENEFICIARY
- EMPLOYEE
- GROUP_ROLE
- MUNICIPALITY
- PAYROLL
- POSITION
- USER_ACCOUNT

ATTRIBUTES:

1. ACCESS_ROLE

- ROLE_ID (integer)
- STATUS (varchar)
- ROLE (varchar)

2. ASSIGNMENT

- ID (integer)
- MUNI_PSGC (integer)
- EMPLOYEE_ID (integer)
- ID1 (integer)
- ID11 (integer)

3. BARANGAY

- BRGY_PSGC (integer)
- MUNI_PSGC (integer)
- BARANGAY (varchar)
- MUNI_PSGC1 (integer)
- MUNI_PSGC11 (integer)
- PROV_PSGC1 (integer)
- REG_PSGC1 (integer)

4. BENEFICIARY

- ID (integer)
- FIRST_NAME (varchar)
- MIDDLE_NAME (varchar)
- LASTNAME (varchar)
- DATE_OF_BIRTH (date)
- NATIONALITY (varchar)
- DISABILITY_ID (integer)
- SEX (varchar)
- ADDRESS_PSGC (integer)
- STATUS (integer)
- BRGY_PSGC (integer)
- MUNI_PSGC1 (integer)
- MUNI_PSGC11 (integer)
- PROV_PSGC1 (integer)
- REG_PSGC1 (integer)

5. EMPLOYEE

- ID (integer)
- FIRST_NAME (varchar)
- MIDDLE_NAME (varchar)
- LASTNAME (varchar)
- ASSIGNMENT_ID (integer)
- POSITION_ID (integer)
- EMPLOYEE_ID (integer)
- ID1 (integer)

6. GROUP_ROLE

- ID (integer)
- GROUP_ID (integer)
- ROLE_ID (integer)
- ROLE_ID1 (integer)

- GROUP_ID1 (integer)

7. lib_fund_source

- ID (integer)
- SARO (varchar)
- SSA (varchar)
- ORS (varchar)
- LDDAP_ADA (varchar)
- EFFECTIVITY_DATE (date)
- EXPIRY_DATE (date)

8. MUNICIPALITY

- MUNI_PSGC (integer)
- PROV_PSGC (integer)
- MUNICIPALITY (varchar)
- MUNI_PSGC1 (integer)
- PROV_PSGC1 (integer)
- REG_PSGC1 (integer)

9. PAYROLL

- ID (integer)
- FUND_SOURCE_ID (integer)
- ALLOTTED_AMOUNT (number)
- MOP (varchar)
- ID1 (integer)
- PAYROLL_STATUS_ID (integer)
- PAYROLL_STATUS_EMPLOYEE_ID (integer)
- PR_STAT_EMP_POS_ID (integer)
- ID2 (integer)
- ID21 (integer)
- ID12 (integer)

10. PAYROLL_DETAIL

- ID (integer)
- GRANTED_AMOUNT (number)
- DISBURSEMENT_STATUS (integer)
- BENEFICIARY_ID (integer)
- PAYROLL_ID (integer)
- ID1 (integer)
- ID11 (integer)
- ID2 (integer)
- BRGY_PSGC (integer)
- MUNI_PSGC1 (integer)
- MUNI_PSGC11 (integer)
- PROV_PSGC1 (integer)
- REG_PSGC1 (integer)

11. PAYROLL_STATUS

- ID (integer)
- PAYROLL_ID (integer)
- PREPARED_BY (integer)
- PREPARED_DATE (date)
- RECOMMENDED_BY (integer)
- RECOMMENDED_DATE (date)
- APPROVED_BY (integer)
- APPROVED_DATE (date)
- ID2 (integer)
- ID12 (integer)
- ID3 (integer)
- ID13 (integer)
- ID4 (integer)
- ID14 (integer)

12. POSITION

- ID (integer)
- POSITION (varchar)
- OBSU (varchar)
- DESIGNATION (varchar)

13. PROVINCE

- PROV_PSGC (integer)
- PROVINCE (varchar)
- REG_PSGC (integer)
- REG_PSGC1 (integer)

14. REGION

- REG_PSGC (integer)
- REGION (varchar)

15. USER_ACCOUNT

- ID (integer)
- EMPLOYEE_ID (integer)
- USERNAME (varchar)
- PASSWORD (varchar)
- STATUS (integer)
- GROUP_ID (integer)
- GROUP_ID1 (integer)
- ID1 (integer)
- ID11 (integer)

16. USER_GROUP

- GROUP_ID (integer)

- USER_GROUP (varchar)
- STATUS (varchar)

PRIMARY AND FOREIGN KEY

1. ACCESS_ROLE

- Primary Key (PK): ROLE_ID
- Foreign Keys (FK): None

2. ASSIGNMENT

- Primary Key (PK): (ID, ID1, ID11)
- Foreign Keys (FK):
  - (ID1, ID11) → references EMPLOYEE(ID, ID1)

3. BARANGAY

- Primary Key (PK): (BRGY_PSGC, MUNI_PSGC1, MUNI_PSGC11, PROV_PSGC1, REG_PSGC1)
- Foreign Keys (FK):
  - (MUNI_PSGC1, MUNI_PSGC11, PROV_PSGC1, REG_PSGC1) → references MUNICIPALITY(MUNI_PSGC, MUNI_PSGC1, PROV_PSGC1, REG_PSGC1)

4. BENEFICIARY

- Primary Key (PK): (ID, BRGY_PSGC, MUNI_PSGC1, MUNI_PSGC11, PROV_PSGC1, REG_PSGC1)
- Foreign Keys (FK):
  - (BRGY_PSGC, MUNI_PSGC1, MUNI_PSGC11, PROV_PSGC1, REG_PSGC1) → references BARANGAY(BRGY_PSGC, MUNI_PSGC1, MUNI_PSGC11, PROV_PSGC1, REG_PSGC1)

5. EMPLOYEE

- Primary Key (PK): (ID, ID1)
- Foreign Keys (FK):
  - (ID1) → references POSITION(ID)
  - (EMPLOYEE_ID) → references USER_ACCOUNT(EMPLOYEE_ID)

6. GROUP_ROLE

- Primary Key (PK): (ID, ROLE_ID1, GROUP_ID1)
- Foreign Keys (FK):
  - (ROLE_ID1) → references ACCESS_ROLE(ROLE_ID)
  - (GROUP_ID1) → references USER_GROUP(GROUP_ID)

7. lib_fund_source

- Primary Key (PK): ID

8. MUNICIPALITY

- Primary Key (PK): (MUNI_PSGC, MUNI_PSGC1, PROV_PSGC1, REG_PSGC1)
- Foreign Keys (FK):
    - (MUNI_PSGC1) → references ASSIGNMENT(MUNI_PSGC)
    - (PROV_PSGC1, REG_PSGC1) → references PROVINCE(PROV_PSGC, REG_PSGC1)

9. PAYROLL

- Primary Key (PK): (ID, ID1)
- Foreign Keys (FK):
    - (ID1) → references lib_fund_source(ID)

10. PAYROLL_DETAIL

- Primary Key (PK): (ID, ID1, ID11, ID2, BRGY_PSGC, MUNI_PSGC1, MUNI_PSGC11, PROV_PSGC1, REG_PSGC1)
- Foreign Keys (FK):
    - (PAYROLL_ID) → references PAYROLL(ID, ID1)
    - (BENEFICIARY_ID, BRGY_PSGC, MUNI_PSGC1, MUNI_PSGC11, PROV_PSGC1, REG_PSGC1) → references BENEFICIARY(ID, BRGY_PSGC, MUNI_PSGC1, MUNI_PSGC11, PROV_PSGC1, REG_PSGC1)

11. PAYROLL_STATUS

- Primary Key (PK): (ID, ID2, ID12)
- Foreign Keys (FK):
    - (PAYROLL_ID) → references PAYROLL(ID, ID1)
    - (PREPARED_BY, RECOMMENDED_BY, APPROVED_BY) → references EMPLOYEE(ID, ID1)

12. POSITION

- Primary Key (PK): ID
- Foreign Keys (FK): None

13. PROVINCE

- Primary Key (PK): (PROV_PSGC, REG_PSGC1)
- Foreign Keys (FK):
    - (REG_PSGC1) → references REGION(REG_PSGC)

14. REGION

- Primary Key (PK): REG_PSGC

- Foreign Keys (FK): None

15. USER_ACCOUNT

- Primary Key (PK): (ID, GROUP_ID1)
- Foreign Keys (FK):
  - (EMPLOYEE_ID) → references EMPLOYEE(ID, ID1)
  - (GROUP_ID1) → references USER_GROUP(GROUP_ID)

16. USER_GROUP

- Primary Key (PK): GROUP_ID
- Foreign Keys (FK): None

3.3 **Relationship Mapping**

	PAYROLL	BENEFICIARY	WORKERS	BENEFICIARY ADDRESS	AREA OF ASSIGNMENT	FUND SOURCE
PAYROLL		Includes in	Prepare			Assigned to
BENEFICIARY	Includes in		Manage	Resides		
WORKERS	Process	Handles Multiple beneficiary			Assigned	
BENEFICIARY ADDRESS		Location for				
AREA OF ASSIGNMENT			Contains Multiple Field Workers			
FUND SOURCE	Source of fund for					

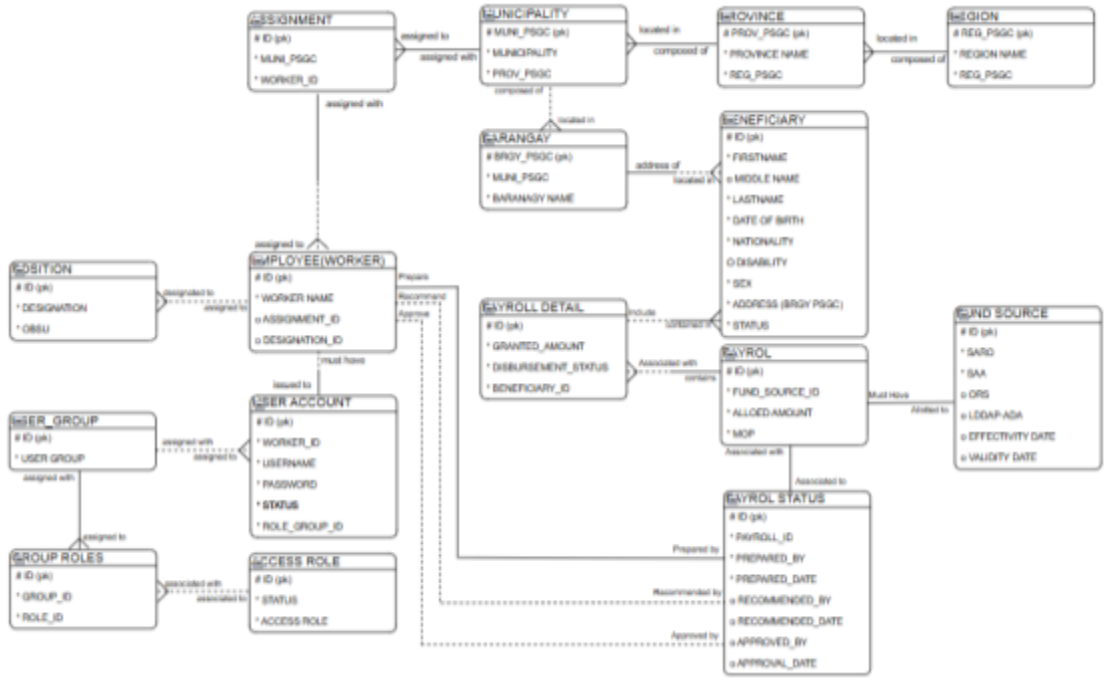
---

4. Normalization Process

4.1 **Normalization Steps (1NF, 2NF, 3NF)**

To remove redundancies in the many-to-many relationships, I focused on eliminating unnecessary or repetitive foreign key constraints. For example, in tables like GROUP_ROLE, USER_ACCOUNT, and EMPLOYEE, I ensured that each relationship is defined clearly and only once, avoiding redundant references. I examined the indexing and foreign key constraints to ensure that the relationships between tables (like EMPLOYEE and USER_ACCOUNT) are correctly modeled without repeating the same logic. This reduces redundancy and ensures that the schema is both efficient and easier to maintain, particularly in scenarios where entities are related in multiple ways.

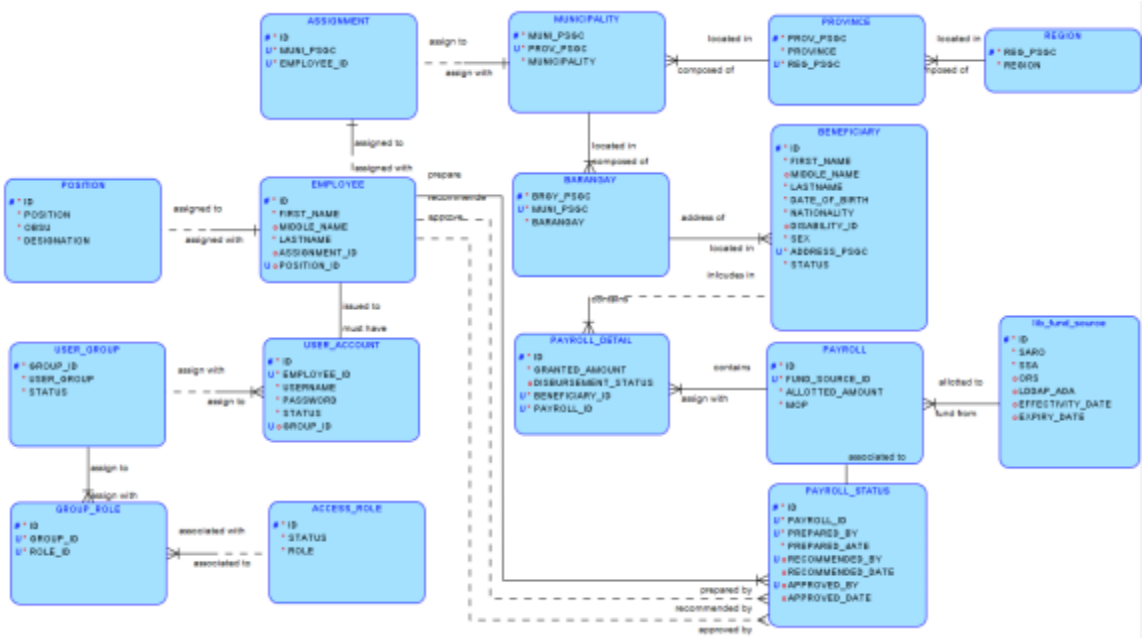
4.2 **Updated ERD**



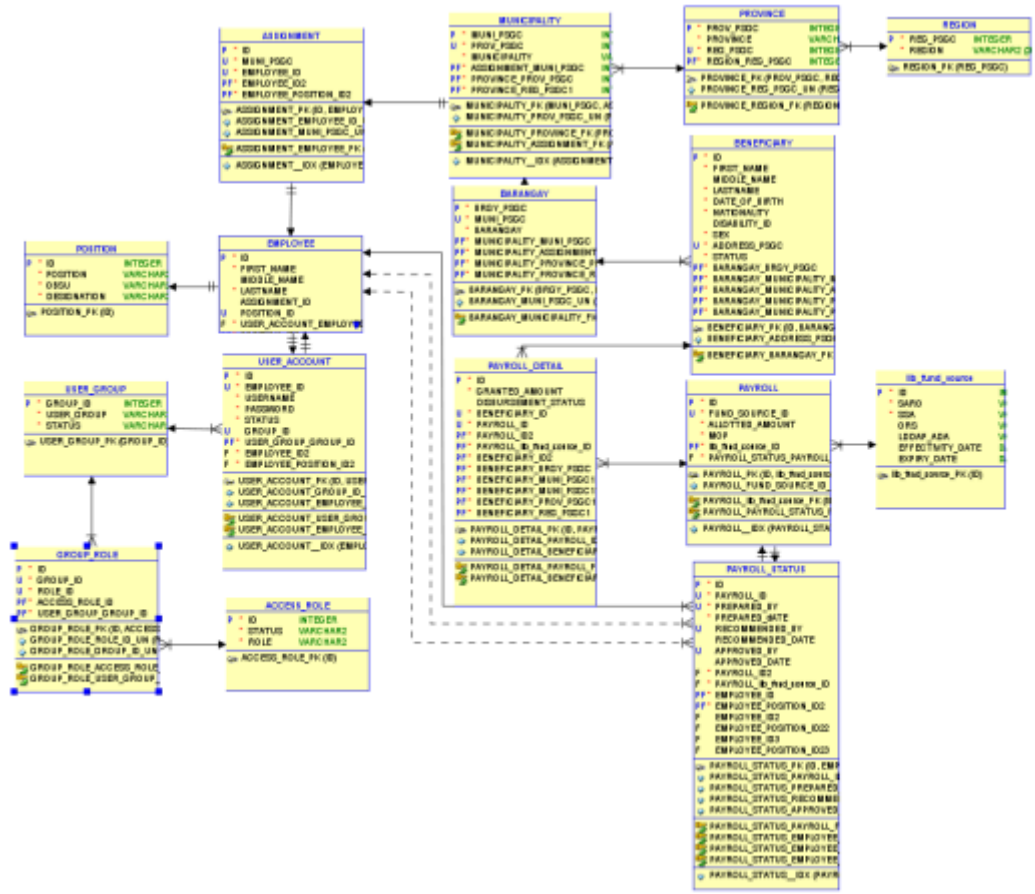
---

5. Logical and Relational Model in Oracle SQL Developer Data Modeler

5.1 **Logical Model**



5.2 **Relational Model**



6. Data Dictionary

6.1 **Table Structures**

Table Name: ACCESS_ROLE

Column Name	Data Type	Constraints	Description
ROLE_ID	INTEGER	NOT NULL, PRIMARY KEY	Unique identifier for the role
STATUS	VARCHAR2(30)	NOT NULL	Current status of the role
ROLE	VARCHAR2(30)	NOT NULL	Name of the role

Table Name: ASSIGNMENT

Column Name	Data Type	Constraints	Description
ID	INTEGER	NOT NULL, PRIMARY KEY	Unique identifier for the assignment
MUNI_PSGC	INTEGER	NOT NULL, UNIQUE	Municipality PSGC code
EMPLOYEE_ID	INTEGER	NOT NULL, UNIQUE	Unique identifier for the employee
ID1	INTEGER	NOT NULL, UNIQUE	Additional identifier for the assignment
ID11	INTEGER	NOT NULL, UNIQUE	Secondary additional identifier

Table Name: BARANGAY

Column Name	Data Type	Constraints	Description
BRGY_PSGC	INTEGER	NOT NULL, PRIMARY KEY	Barangay PSGC code
MUNI_PSGC	INTEGER	NOT NULL, UNIQUE	Municipality PSGC code
BARANGAY	VARCHAR2(20)	NOT NULL	Name of the Barangay
MUNI_PSGC1	INTEGER	NOT NULL	Municipality PSGC code (Secondary)
MUNI_PSGC11	INTEGER	NOT NULL	Secondary PSGC code
PROV_PSGC1	INTEGER	NOT NULL	Province PSGC code
REG_PSGC1	INTEGER	NOT NULL	Region PSGC code

Table Name: BENEFICIARY

Column Name	Data Type	Constraints	Description
ID	INTEGER	NOT NULL, PRIMARY KEY	Unique identifier for the beneficiary
FIRST_NAME	VARCHAR2(30)	NOT NULL	First name of the beneficiary
MIDDLE_NAME	VARCHAR2(30)	-	Middle name of the beneficiary (Optional)
LASTNAME	VARCHAR2(30)	NOT NULL	Last name of the beneficiary
DATE_OF_BIRTH	DATE	NOT NULL	Date of birth of the beneficiary
NATIONALITY	VARCHAR2(30)	NOT NULL	Nationality of the beneficiary

DISABILITY_ID	INTEGER	-	Disability ID (Optional)
SEX	VARCHAR2(4)	NOT NULL	Gender of the beneficiary
ADDRESS_PSGC	INTEGER	NOT NULL, UNIQUE	PSGC code for the address
STATUS	INTEGER	NOT NULL	Status of the beneficiary
BRGY_PSGC	INTEGER	NOT NULL	Barangay PSGC code
MUNI_PSGC1	INTEGER	NOT NULL	Municipality PSGC code
MUNI_PSGC11	INTEGER	NOT NULL	Secondary municipality PSGC code
PROV_PSGC1	INTEGER	NOT NULL	Province PSGC code
REG_PSGC1	INTEGER	NOT NULL	Region PSGC code

Table Name: EMPLOYEE

Column Name	Data Type	Constraints	Description
ID	INTEGER	NOT NULL, PRIMARY KEY	Unique identifier for the employee
FIRST_NAME	VARCHAR2(50)	NOT NULL	First name of the employee
MIDDLE_NAME	VARCHAR2(20)	-	Middle name of the employee (Optional)
LASTNAME	VARCHAR2(50)	NOT NULL	Last name of the employee
ASSIGNMENT_ID	INTEGER	-	ID for assignment, references a location (Optional)
POSITION_ID	INTEGER	-	ID of the employee's position
EMPLOYEE_ID	INTEGER	NOT NULL, UNIQUE	Unique employee identification number
ID1	INTEGER	NOT NULL, UNIQUE	Secondary identifier for employee

Table Name: GROUP_ROLE

Column Name	Data Type	Constraints	Description
ID	INTEGER	NOT NULL, PRIMARY KEY	Unique identifier for the group role
GROUP_ID	INTEGER	NOT NULL	Unique group identifier
ROLE_ID	INTEGER	NOT NULL	Unique role identifier
ROLE_ID1	INTEGER	NOT NULL	Secondary role identifier
GROUP_ID1	INTEGER	NOT NULL	Secondary group identifier

Table Name: lib_fund_source

Column Name	Data Type	Constraints	Description
ID	INTEGER	NOT NULL, PRIMARY KEY	Unique identifier for the fund source
SARO	VARCHAR2(15)	NOT NULL	Special Allotment Release Order
SSA	VARCHAR2(15)	NOT NULL	Sub-Allotment Advice
ORS	VARCHAR2(10)	-	Obligation Request and Status (Optional)
LDDAP_ADA	VARCHAR2(50)	-	List of Due and Demandable Accounts Payable with Advice to Debit Account (LDDAP-ADA code) (Optional)
EFFECTIVITY_DATE	DATE	-	The effective date of the fund
EXPIRY_DATE	DATE	-	The expiration date of the fund (Optional)

Table Name: MUNICIPALITY

Column Name	Data Type	Constraints	Description
MUNI_PSGC	INTEGER	NOT NULL, PRIMARY KEY	Municipality PSGC code
PROV_PSGC	INTEGER	NOT NULL	Province PSGC code
MUNICIPALITY	VARCHAR2(20)	NOT NULL	Name of the municipality
MUNI_PSGC1	INTEGER	NOT NULL	Secondary municipality PSGC code
PROV_PSGC1	INTEGER	NOT NULL	Province PSGC code (Secondary)
REG_PSGC1	INTEGER	NOT NULL	Region PSGC code

Table Name: PAYROLL

Column Name	Data Type	Constraints	Description
ID	INTEGER	NOT NULL, PRIMARY KEY	Unique identifier for the payroll
FUND_SOURCE_ID	INTEGER	NOT NULL	ID for the fund source
ALLOTTED_AMOUNT	NUMBER(2)	NOT NULL	Allotted amount for payroll
MOP	VARCHAR2(10)	NOT NULL	Method of payment

ID1	INTEGER	NOT NULL	Identifier 1 for payroll
PAYROLL_STATUS_ID	INTEGER	NOT NULL	Payroll status ID
PAYROLL_STATUS_EMPLOYEE_ID	INTEGER	NOT NULL	Employee ID linked to payroll
PR_STAT_EMP_POS_ID	INTEGER	NOT NULL	Employee position ID for payroll status
ID2	INTEGER	NOT NULL	Additional identifier 2 for payroll
ID21	INTEGER	NOT NULL	Identifier for payroll status
ID12	INTEGER	NOT NULL	Secondary identifier for payroll

Table Name: PAYROLL_DETAIL

Column Name	Data Type	Constraints	Description
ID	INTEGER	NOT NULL, PRIMARY KEY	Unique identifier for payroll detail
GRANTED_AMOUNT	NUMBER(2)	NOT NULL	Granted amount in payroll
DISBURSEMENT_STATUS	INTEGER	-	Status of the disbursement
BENEFICIARY_ID	INTEGER	NOT NULL	ID for the beneficiary
PAYROLL_ID	INTEGER	NOT NULL	Linked payroll ID
ID1	INTEGER	NOT NULL	Identifier 1
ID11	INTEGER	NOT NULL	Identifier 2
ID2	INTEGER	NOT NULL	Identifier 3
BRGY_PSGC	INTEGER	NOT NULL	Barangay PSGC code
MUNI_PSGC1	INTEGER	NOT NULL	Municipality PSGC code
MUNI_PSGC11	INTEGER	NOT NULL	Secondary municipality PSGC code
PROV_PSGC1	INTEGER	NOT NULL	Province PSGC code
REG_PSGC1	INTEGER	NOT NULL	Region PSGC code

6.2 **Business Rules and Constraints**

- Add CHECK constraint to ensure age is 60 or more

  ```
  ALTER TABLE BENEFICIARY ADD CONSTRAINT CHK_BENE_AGE CHECK (
  TRUNC(MONTHS_BETWEEN(SYSDATE, DATE_OF_BIRTH) / 12) >= 60 );
  ```

- Add UNIQUE constraint on FIRST_NAME, LASTNAME, and DATE_OF_BIRTH

  ```
  ALTER TABLE BENEFICIARY ADD CONSTRAINT UNIQUE_BENE_NAME_DOB
  UNIQUE (FIRST_NAME, LASTNAME, DATE_OF_BIRTH);
  ```

7. DDL Statements (Generated from Oracle SQL Developer Data Modeler)

7.1 Generated SQL Statements

```
-- Generated by Oracle SQL Developer Data Modeler 24.3.1.351.0831
--   at:        2025-02-23 13:46:02 CST
--   site:      Oracle Database 11g
--   type:      Oracle Database 11g



-- predefined type, no DDL - MDSYS.SDO_GEOMETRY

-- predefined type, no DDL - XMLTYPE

CREATE TABLE ACCESS_ROLE
    (
     ROLE_ID INTEGER  NOT NULL ,
     STATUS  VARCHAR2 (30)  NOT NULL ,
     ROLE    VARCHAR2 (30)  NOT NULL
    )
;

ALTER TABLE ACCESS_ROLE
    ADD CONSTRAINT ACCESS_ROLE_PK PRIMARY KEY ( ROLE_ID ) ;

CREATE TABLE ASSIGNMENT
    (
     ID          INTEGER  NOT NULL ,
     MUNI_PSGC   INTEGER  NOT NULL ,
     EMPLOYEE_ID INTEGER  NOT NULL ,
     ID1         INTEGER  NOT NULL ,
     ID11        INTEGER  NOT NULL
    )
;
CREATE UNIQUE INDEX ASSIGNMENT__IDX ON ASSIGNMENT
    (
     ID1 ASC ,
     ID11 ASC
    )
;

ALTER TABLE ASSIGNMENT
    ADD CONSTRAINT ASSIGNMENT_PK PRIMARY KEY ( ID, ID1, ID11 ) ;

ALTER TABLE ASSIGNMENT
    ADD CONSTRAINT ASSIGNMENT_EMPLOYEE_ID_UN UNIQUE ( EMPLOYEE_ID ) ;

ALTER TABLE ASSIGNMENT
    ADD CONSTRAINT ASSIGNMENT_MUNI_PSGC_UN UNIQUE ( MUNI_PSGC ) ;

CREATE TABLE BARANGAY
    (
     BRGY_PSGC   INTEGER  NOT NULL ,
     MUNI_PSGC   INTEGER  NOT NULL ,
     BARANGAY    VARCHAR2 (20)  NOT NULL ,
     MUNI_PSGC1  INTEGER  NOT NULL ,
     MUNI_PSGC11 INTEGER  NOT NULL ,
     PROV_PSGC1  INTEGER  NOT NULL ,
     REG_PSGC1   INTEGER  NOT NULL
```

```
    )
;

ALTER TABLE BARANGAY
    ADD CONSTRAINT BARANGAY_PK PRIMARY KEY ( BRGY_PSGC, MUNI_PSGC1,
MUNI_PSGC11, PROV_PSGC1, REG_PSGC1 ) ;

ALTER TABLE BARANGAY
    ADD CONSTRAINT BARANGAY_MUNI_PSGC_UN UNIQUE ( MUNI_PSGC ) ;

CREATE TABLE BENEFICIARY
    (
     ID            INTEGER  NOT NULL ,
     FIRST_NAME    VARCHAR2 (30)  NOT NULL ,
     MIDDLE_NAME   VARCHAR2 (30) ,
     LASTNAME      VARCHAR2 (30)  NOT NULL ,
     DATE_OF_BIRTH DATE  NOT NULL ,
     NATIONALITY   VARCHAR2 (30)  NOT NULL ,
     DISABILITY_ID INTEGER ,
     SEX           VARCHAR2 (4)  NOT NULL ,
     ADDRESS_PSGC  INTEGER  NOT NULL ,
     STATUS        INTEGER  NOT NULL ,
     BRGY_PSGC     INTEGER  NOT NULL ,
     MUNI_PSGC1    INTEGER  NOT NULL ,
     MUNI_PSGC11   INTEGER  NOT NULL ,
     PROV_PSGC1    INTEGER  NOT NULL ,
     REG_PSGC1     INTEGER  NOT NULL
    )
;

ALTER TABLE BENEFICIARY
    ADD CONSTRAINT BENEFICIARY_PK PRIMARY KEY ( ID, BRGY_PSGC, MUNI_PSGC1,
MUNI_PSGC11, PROV_PSGC1, REG_PSGC1 ) ;

ALTER TABLE BENEFICIARY
    ADD CONSTRAINT BENEFICIARY_ADDRESS_PSGC_UN UNIQUE ( ADDRESS_PSGC ) ;

CREATE TABLE EMPLOYEE
    (
     ID            INTEGER  NOT NULL ,
     FIRST_NAME    VARCHAR2 (50)  NOT NULL ,
     MIDDLE_NAME   VARCHAR2 (20) ,
     LASTNAME      VARCHAR2 (50)  NOT NULL ,
     ASSIGNMENT_ID INTEGER ,
     POSITION_ID   INTEGER ,
     EMPLOYEE_ID   INTEGER  NOT NULL ,
     ID1           INTEGER  NOT NULL
    )
;

COMMENT ON COLUMN EMPLOYEE.ASSIGNMENT_ID IS 'BARANGAY PSGC'
;
CREATE UNIQUE INDEX EMPLOYEE__IDX ON EMPLOYEE
    (
     ID1 ASC
    )
;
CREATE UNIQUE INDEX EMPLOYEE__IDXv1 ON EMPLOYEE
    (
     EMPLOYEE_ID ASC
    )
;
```

```
ALTER TABLE EMPLOYEE
    ADD CONSTRAINT EMPLOYEE_PK PRIMARY KEY ( ID, ID1 ) ;

ALTER TABLE EMPLOYEE
    ADD CONSTRAINT EMPLOYEE_POSITION_ID_UN UNIQUE ( POSITION_ID ) ;

CREATE TABLE GROUP_ROLE
    (
     ID        INTEGER  NOT NULL ,
     GROUP_ID  INTEGER  NOT NULL ,
     ROLE_ID   INTEGER  NOT NULL ,
     ROLE_ID1  INTEGER  NOT NULL ,
     GROUP_ID1 INTEGER  NOT NULL
    )
;

ALTER TABLE GROUP_ROLE
    ADD CONSTRAINT GROUP_ROLE_PK PRIMARY KEY ( ID, ROLE_ID1, GROUP_ID1 ) ;

ALTER TABLE GROUP_ROLE
    ADD CONSTRAINT GROUP_ROLE_ROLE_ID_UN UNIQUE ( ROLE_ID ) ;

ALTER TABLE GROUP_ROLE
    ADD CONSTRAINT GROUP_ROLE_GROUP_ID_UN UNIQUE ( GROUP_ID ) ;

CREATE TABLE lib_fund_source
    (
     ID               INTEGER  NOT NULL ,
     SARO             VARCHAR2 (15)  NOT NULL ,
     SSA              VARCHAR2 (15)  NOT NULL ,
     ORS              VARCHAR2 (10) ,
     LDDAP_ADA        VARCHAR2 (50) ,
     EFFECTIVITY_DATE DATE ,
     EXPIRY_DATE      DATE
    )
;

ALTER TABLE lib_fund_source
    ADD CONSTRAINT lib_fund_source_PK PRIMARY KEY ( ID ) ;

CREATE TABLE MUNICIPALITY
    (
     MUNI_PSGC    INTEGER  NOT NULL ,
     PROV_PSGC    INTEGER  NOT NULL ,
     MUNICIPALITY VARCHAR2 (20)  NOT NULL ,
     MUNI_PSGC1   INTEGER  NOT NULL ,
     PROV_PSGC1   INTEGER  NOT NULL ,
     REG_PSGC1    INTEGER  NOT NULL
    )
;
CREATE UNIQUE INDEX MUNICIPALITY__IDX ON MUNICIPALITY
    (
     MUNI_PSGC1 ASC
    )
;

ALTER TABLE MUNICIPALITY
    ADD CONSTRAINT MUNICIPALITY_PK PRIMARY KEY ( MUNI_PSGC, MUNI_PSGC1,
PROV_PSGC1, REG_PSGC1 ) ;

ALTER TABLE MUNICIPALITY
```

```
    ADD CONSTRAINT MUNICIPALITY_PROV_PSGC_UN UNIQUE ( PROV_PSGC ) ;

CREATE TABLE PAYROLL
    (
     ID                          INTEGER  NOT NULL ,
     FUND_SOURCE_ID              INTEGER  NOT NULL ,
     ALLOTTED_AMOUNT             NUMBER (2)  NOT NULL ,
     MOP                         VARCHAR2 (10)  NOT NULL ,
     ID1                         INTEGER  NOT NULL ,
     PAYROLL_STATUS_ID           INTEGER  NOT NULL ,
     PAYROLL_STATUS_EMPLOYEE_ID INTEGER  NOT NULL ,
     PR_STAT_EMP_POS_ID          INTEGER  NOT NULL ,
     ID2                         INTEGER  NOT NULL ,
     ID21                        INTEGER  NOT NULL ,
     ID12                        INTEGER  NOT NULL
    )
;

CREATE UNIQUE INDEX PAYROLL__IDXv3 ON PAYROLL
    (
     PAYROLL_STATUS_ID ASC ,
     PAYROLL_STATUS_EMPLOYEE_ID ASC ,
     PR_STAT_EMP_POS_ID ASC
    )
;
CREATE UNIQUE INDEX PAYROLL__IDXv1 ON PAYROLL
    (
     ID2 ASC ,
     ID21 ASC ,
     ID12 ASC
    )
;

ALTER TABLE PAYROLL
    ADD CONSTRAINT PAYROLL_PK PRIMARY KEY ( ID, ID1 ) ;

ALTER TABLE PAYROLL
    ADD CONSTRAINT PAYROLL_FUND_SOURCE_ID_UN UNIQUE ( FUND_SOURCE_ID ) ;

CREATE TABLE PAYROLL_DETAIL
    (
     ID                   INTEGER  NOT NULL ,
     GRANTED_AMOUNT       NUMBER (2)  NOT NULL ,
     DISBURSEMENT_STATUS  INTEGER ,
     BENEFICIARY_ID       INTEGER  NOT NULL ,
     PAYROLL_ID           INTEGER  NOT NULL ,
     ID1                  INTEGER  NOT NULL ,
     ID11                 INTEGER  NOT NULL ,
     ID2                  INTEGER  NOT NULL ,
     BRGY_PSGC            INTEGER  NOT NULL ,
     MUNI_PSGC1           INTEGER  NOT NULL ,
     MUNI_PSGC11          INTEGER  NOT NULL ,
     PROV_PSGC1           INTEGER  NOT NULL ,
     REG_PSGC1            INTEGER  NOT NULL
    )
;

ALTER TABLE PAYROLL_DETAIL
    ADD CONSTRAINT PAYROLL_DETAIL_PK PRIMARY KEY ( ID, ID1, ID11, ID2,
BRGY_PSGC, MUNI_PSGC1, MUNI_PSGC11, PROV_PSGC1, REG_PSGC1 ) ;

ALTER TABLE PAYROLL_DETAIL
```

```
    ADD CONSTRAINT PAYROLL_DETAIL_PAYROLL_ID_UN UNIQUE ( PAYROLL_ID ) ;

ALTER TABLE PAYROLL_DETAIL
    ADD CONSTRAINT PAYROLL_DETAIL_BENE_ID_UN UNIQUE ( BENEFICIARY_ID ) ;

CREATE TABLE PAYROLL_STATUS
    (
     ID               INTEGER  NOT NULL ,
     PAYROLL_ID       INTEGER  NOT NULL ,
     PREPARED_BY      INTEGER  NOT NULL ,
     PREPARED_dATE    DATE  NOT NULL ,
     RECOMMENDED_BY   INTEGER ,
     RECOMMENDED_DATE DATE ,
     APPROVED_BY      INTEGER ,
     APPROVED_DATE    DATE ,
     ID2              INTEGER  NOT NULL ,
     ID12             INTEGER  NOT NULL ,
     ID3              INTEGER ,
     ID13             INTEGER ,
     ID4              INTEGER ,
     ID14             INTEGER
    )
;


ALTER TABLE PAYROLL_STATUS
    ADD CONSTRAINT PAYROLL_STATUS_PK PRIMARY KEY ( ID, ID2, ID12 ) ;

ALTER TABLE PAYROLL_STATUS
    ADD CONSTRAINT PAYROLL_STATUS_PAYROLL_ID_UN UNIQUE ( PAYROLL_ID ) ;

ALTER TABLE PAYROLL_STATUS
    ADD CONSTRAINT PAYROLL_STATUS_PREPARED_BY_UN UNIQUE ( PREPARED_BY ) ;

ALTER TABLE PAYROLL_STATUS
    ADD CONSTRAINT PAYROLL_STATUS_REC_BY_UN UNIQUE ( RECOMMENDED_BY ) ;

ALTER TABLE PAYROLL_STATUS
    ADD CONSTRAINT PAYROLL_STATUS_APPROVED_BY_UN UNIQUE ( APPROVED_BY ) ;

CREATE TABLE POSITION
    (
     ID          INTEGER  NOT NULL ,
     POSITION    VARCHAR2 (50)  NOT NULL ,
     OBSU        VARCHAR2 (50)  NOT NULL ,
     DESIGNATION VARCHAR2 (50)  NOT NULL
    )
;

ALTER TABLE POSITION
    ADD CONSTRAINT POSITION_PK PRIMARY KEY ( ID ) ;

CREATE TABLE PROVINCE
    (
     PROV_PSGC INTEGER  NOT NULL ,
     PROVINCE  VARCHAR2 (20)  NOT NULL ,
     REG_PSGC  INTEGER  NOT NULL ,
     REG_PSGC1 INTEGER  NOT NULL
    )
;

ALTER TABLE PROVINCE
```

```
    ADD CONSTRAINT PROVINCE_PK PRIMARY KEY ( PROV_PSGC, REG_PSGC1 ) ;

ALTER TABLE PROVINCE
    ADD CONSTRAINT PROVINCE_REG_PSGC_UN UNIQUE ( REG_PSGC ) ;

CREATE TABLE REGION
    (
     REG_PSGC INTEGER  NOT NULL ,
     REGION   VARCHAR2 (20)  NOT NULL
    )
;

ALTER TABLE REGION
    ADD CONSTRAINT REGION_PK PRIMARY KEY ( REG_PSGC ) ;

CREATE TABLE USER_ACCOUNT
    (
     ID          INTEGER  NOT NULL ,
     EMPLOYEE_ID INTEGER  NOT NULL ,
     USERNAME    VARCHAR2 (30)  NOT NULL ,
     PASSWORD    VARCHAR2 (20)  NOT NULL ,
     STATUS      INTEGER  NOT NULL ,
     GROUP_ID    INTEGER ,
     GROUP_ID1   INTEGER  NOT NULL ,
     ID1         INTEGER  NOT NULL ,
     ID11        INTEGER  NOT NULL
    )
;
CREATE UNIQUE INDEX USER_ACCOUNT__IDX ON USER_ACCOUNT
    (
     ID1 ASC ,
     ID11 ASC
    )
;

ALTER TABLE USER_ACCOUNT
    ADD CONSTRAINT USER_ACCOUNT_PK PRIMARY KEY ( ID, GROUP_ID1 ) ;

ALTER TABLE USER_ACCOUNT
    ADD CONSTRAINT USER_ACCOUNT_GROUP_ID_UN UNIQUE ( GROUP_ID ) ;

ALTER TABLE USER_ACCOUNT
    ADD CONSTRAINT USER_ACCOUNT_EMPLOYEE_ID_UN UNIQUE ( EMPLOYEE_ID ) ;

CREATE TABLE USER_GROUP
    (
     GROUP_ID   INTEGER  NOT NULL ,
     USER_GROUP VARCHAR2 (20)  NOT NULL ,
     STATUS     VARCHAR2 (30)  NOT NULL
    )
;

ALTER TABLE USER_GROUP
    ADD CONSTRAINT USER_GROUP_PK PRIMARY KEY ( GROUP_ID ) ;

ALTER TABLE ASSIGNMENT
    ADD CONSTRAINT ASSIGNMENT_EMPLOYEE_FK FOREIGN KEY
    (
     ID1,
     ID11
    )
    REFERENCES EMPLOYEE
```

```
    (
     ID,
     ID1
    )
;

ALTER TABLE BARANGAY
    ADD CONSTRAINT BARANGAY_MUNICIPALITY_FK FOREIGN KEY
    (
     MUNI_PSGC1,
     MUNI_PSGC11,
     PROV_PSGC1,
     REG_PSGC1
    )
    REFERENCES MUNICIPALITY
    (
     MUNI_PSGC,
     MUNI_PSGC1,
     PROV_PSGC1,
     REG_PSGC1
    )
;

ALTER TABLE BENEFICIARY
    ADD CONSTRAINT BENEFICIARY_BARANGAY_FK FOREIGN KEY
    (
     BRGY_PSGC,
     MUNI_PSGC1,
     MUNI_PSGC11,
     PROV_PSGC1,
     REG_PSGC1
    )
    REFERENCES BARANGAY
    (
     BRGY_PSGC,
     MUNI_PSGC1,
     MUNI_PSGC11,
     PROV_PSGC1,
     REG_PSGC1
    )
;

ALTER TABLE EMPLOYEE
    ADD CONSTRAINT EMPLOYEE_POSITION_FK FOREIGN KEY
    (
     ID1
    )
    REFERENCES POSITION
    (
     ID
    )
;

ALTER TABLE EMPLOYEE
    ADD CONSTRAINT EMPLOYEE_USER_ACCOUNT_FK FOREIGN KEY
    (
     EMPLOYEE_ID
    )
    REFERENCES USER_ACCOUNT
    (
     EMPLOYEE_ID
    )
```

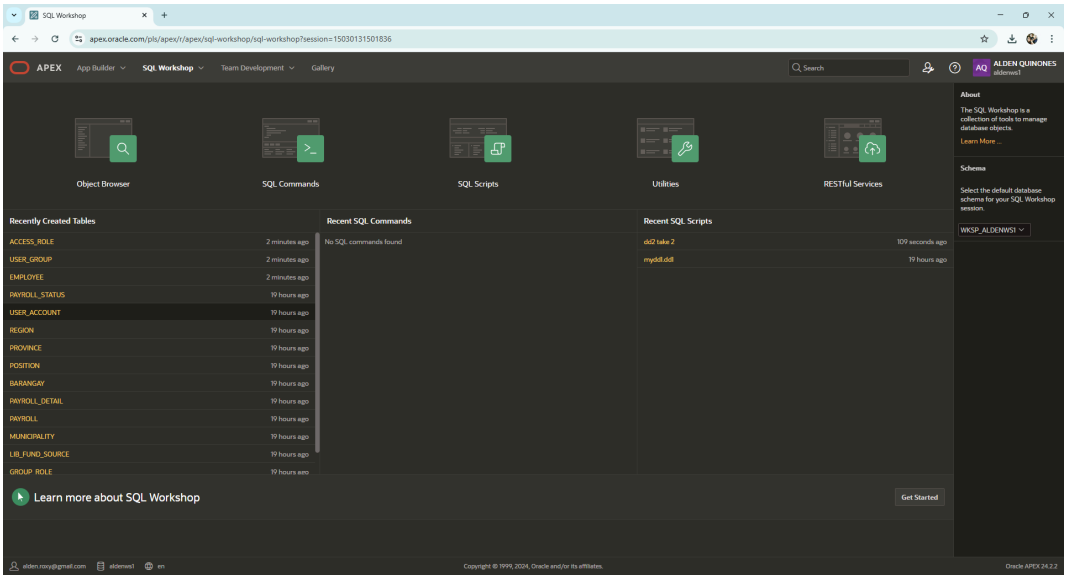
```
;

ALTER TABLE GROUP_ROLE
    ADD CONSTRAINT GROUP_ROLE_ACCESS_ROLE_FK FOREIGN KEY
    (
     ROLE_ID1
    )
    REFERENCES ACCESS_ROLE
    (
     ROLE_ID
    )
;

ALTER TABLE GROUP_ROLE
    ADD CONSTRAINT GROUP_ROLE_USER_GROUP_FK FOREIGN KEY
    (
     GROUP_ID1
    )
    REFERENCES USER_GROUP
    (
     GROUP_ID
    )
;

ALTER TABLE MUNICIPALITY
    ADD CONSTRAINT MUNICIPALITY_ASSIGNMENT_FK FOREIGN KEY
    (
     MUNI_PSGC1
    )
    REFERENCES ASSIGNMENT
    (
     MUNI_PSGC
    )
;

ALTER TABLE MUNICIPALITY
    ADD CONSTRAINT MUNICIPALITY_PROVINCE_FK FOREIGN KEY
    (
     PROV_PSGC1,
     REG_PSGC1
    )
    REFERENCES PROVINCE
    (
     PROV_PSGC,
     REG_PSGC1
    )
;

ALTER TABLE PAYROLL_DETAIL
    ADD CONSTRAINT PAYROLL_DETAIL_BENEFICIARY_FK FOREIGN KEY
    (
     ID2,
     BRGY_PSGC,
     MUNI_PSGC1,
     MUNI_PSGC11,
     PROV_PSGC1,
     REG_PSGC1
    )
    REFERENCES BENEFICIARY
    (
     ID,
     BRGY_PSGC,
```

```
     MUNI_PSGC1,
     MUNI_PSGC11,
     PROV_PSGC1,
     REG_PSGC1
    )
;

ALTER TABLE PAYROLL_DETAIL
    ADD CONSTRAINT PAYROLL_DETAIL_PAYROLL_FK FOREIGN KEY
    (
     ID1,
     ID11
    )
    REFERENCES PAYROLL
    (
     ID,
     ID1
    )
;

ALTER TABLE PAYROLL
    ADD CONSTRAINT PAYROLL_lib_fund_source_FK FOREIGN KEY
    (
     ID1
    )
    REFERENCES lib_fund_source
    (
     ID
    )
;

ALTER TABLE PAYROLL_STATUS
    ADD CONSTRAINT PAYROLL_STATUS_EMPLOYEE_FK FOREIGN KEY
    (
     ID2,
     ID12
    )
    REFERENCES EMPLOYEE
    (
     ID,
     ID1
    )
;

ALTER TABLE PAYROLL_STATUS
    ADD CONSTRAINT PAYROLL_STATUS_EMPLOYEE_FKv1 FOREIGN KEY
    (
     ID4,
     ID14
    )
    REFERENCES EMPLOYEE
    (
     ID,
     ID1
    )
;

ALTER TABLE PAYROLL_STATUS
    ADD CONSTRAINT PAYROLL_STATUS_EMPLOYEE_FKv3 FOREIGN KEY
    (
     ID3,
     ID13
```

```
    )
    REFERENCES EMPLOYEE
    (
     ID,
     ID1
    )
;

ALTER TABLE PROVINCE
    ADD CONSTRAINT PROVINCE_REGION_FK FOREIGN KEY
    (
     REG_PSGC1
    )
    REFERENCES REGION
    (
     REG_PSGC
    )
;

ALTER TABLE USER_ACCOUNT
    ADD CONSTRAINT USER_ACCOUNT_EMPLOYEE_FK FOREIGN KEY
    (
     ID1,
     ID11
    )
    REFERENCES EMPLOYEE
    (
     ID,
     ID1
    )
;

ALTER TABLE USER_ACCOUNT
    ADD CONSTRAINT USER_ACCOUNT_USER_GROUP_FK FOREIGN KEY
    (
     GROUP_ID1
    )
    REFERENCES USER_GROUP
    (
     GROUP_ID
    )
;



-- Oracle SQL Developer Data Modeler Summary Report:
--
-- CREATE TABLE                            16
-- CREATE INDEX                             7
-- ALTER TABLE                             52
-- CREATE VIEW                              0
-- ALTER VIEW                               0
-- CREATE PACKAGE                           0
-- CREATE PACKAGE BODY                      0
-- CREATE PROCEDURE                         0
-- CREATE FUNCTION                          0
-- CREATE TRIGGER                           0
-- ALTER TRIGGER                            0
-- CREATE COLLECTION TYPE                   0
-- CREATE STRUCTURED TYPE                   0
-- CREATE STRUCTURED TYPE BODY              0
-- CREATE CLUSTER                           0
```

```
-- CREATE CONTEXT                          0
-- CREATE DATABASE                         0
-- CREATE DIMENSION                        0
-- CREATE DIRECTORY                        0
-- CREATE DISK GROUP                       0
-- CREATE ROLE                             0
-- CREATE ROLLBACK SEGMENT                 0
-- CREATE SEQUENCE                         0
-- CREATE MATERIALIZED VIEW                0
-- CREATE MATERIALIZED VIEW LOG            0
-- CREATE SYNONYM                          0
-- CREATE TABLESPACE                       0
-- CREATE USER                             0
--
-- DROP TABLESPACE                         0
-- DROP DATABASE                           0
--
-- REDACTION POLICY                        0
--
-- ORDS DROP SCHEMA                        0
-- ORDS ENABLE SCHEMA                      0
-- ORDS ENABLE OBJECT                      0
--
-- ERRORS                                  0
-- WARNINGS                                0
```

7.2 **Screenshot of Successful Execution in Oracle APEX**



8. Conclusion and Recommendations

8.1 **Summary of Database Design** – Highlight key aspects of the final design.

The database design is structured around several key aspects that promote data integrity, efficient data management, and scalability. The schema consists of multiple entities like ACCESS_ROLE, EMPLOYEE, ASSIGNMENT, PAYROLL, and

USER_ACCOUNT, each with well-defined primary and foreign key relationships to ensure referential integrity. Geographic data handling is a major component, with tables such as BARANGAY, MUNICIPALITY, PROVINCE, and REGION using hierarchical codes like PSGC to represent the location structure. This design is normalized to reduce redundancy, with separate tables for entities like beneficiaries and payroll details, making updates and deletions simpler and more efficient. Various indexes are created to enhance query performance, though there are errors in the creation of some indexes that need to be addressed. The schema also incorporates constraints such as NOT NULL and unique indexes to enforce data validity and prevent duplicate records. Furthermore, the structure supports payroll processing, user management, and security by defining roles, employee assignments, and group memberships.

8.2 **Potential Enhancements** –

Implementing a master-slave replication setup where all read operations are routed to slave databases and write operations go to the master database is an effective strategy to improve database performance, especially for read-heavy applications. This design helps distribute the load by offloading read queries to multiple replicas, reducing the stress on the primary database. By using a load balancer in front of the read replicas, you can efficiently distribute the load and ensure that no single slave is overwhelmed. This setup enhances scalability and ensures high availability by enabling failover mechanisms where one of the slave databases can take over if the master goes down.

For data integrity and compliance, implementing an audit trail is crucial. Using triggers within Oracle Database can automatically capture and store changes to critical data, such as inserts, updates, and deletes. This allows you to maintain a record of who made the change, when the change occurred, and what data was affected. By creating an audit table and associating it with triggers on relevant tables, you can ensure that every change is logged automatically, providing transparency and traceability.

Deploying the database system within a Kubernetes environment is a great choice for scalability and resilience. Kubernetes offers auto-scaling capabilities, allowing the database system to automatically scale based on workload demand. In cases of increased traffic or resource usage, Kubernetes can add additional pods to distribute the load and maintain performance without manual intervention. Kubernetes also provides features like self-healing, where if a database instance crashes, it can be automatically replaced by a new one, ensuring high availability. Moreover, running the database in containers allows for easier management, updates, and portability, and provides the flexibility to run on various cloud providers or on-premise infrastructure.

---

9. References

Date, C. J. (2004). *An Introduction to Database Systems* (8th ed.). Addison-Wesley - foundational knowledge on database systems, including relational models and design principles.

Elmasri, R., & Navathe, S. B. (2015). *Fundamentals of Database Systems* (7th ed.). Pearson Education - Comprehensive guide to database management systems

(DBMS), with explanations of relational databases, normalization, and transactions.

Mullins, C. S. (2012). *Database Design for Mere Mortals: A Hands-On Guide to Relational* Database *Design* (3rd ed.). Addison-Wesley - Rovides practical advice on designing relational databases, which could be crucial for a government social pension system.

Oracle Corporation. (2022). *Oracle Database Concepts*. Oracle. - Official Oracle documentation provides in-depth information on database architecture, data modeling, and performance tuning that can be crucial for building an efficient social pension database. https://docs.oracle.com/en/database/

Oracle Corporation. (2022). *Oracle Database SQL Language Reference*. Oracle - A key resource for understanding how to write SQL queries and scripts specific to Oracle databases. https://docs.oracle.com/en/database/

Oracle Corporation. (2021). *Oracle PL/SQL Programming* (6th ed.). O'Reilly - Covers the PL/SQL programming language, useful for creating stored procedures, functions, and triggers in an Oracle-based system.

Chen, P. P. (1976). The entity-relationship model—toward a unified view of data. *ACM Transactions on* Database *Systems*, 1(1), 9-36 - A foundational paper on the entity-relationship model (ER model), which can help in structuring the database schema. https://doi.org/10.1145/320434.320440

Batini, C., Ceri, S., & Navathe, S. B. (1992). *Conceptual Database Design: An Entity*-Relationship *Approach*. Benjamin/Cummings - Critical for understanding conceptual database design, including entity-relationship modeling, which is a key part of system design for DSWD's social pension program.