**EAST WEST UNIVERSITY**

**CSE464:** Mini Project

**[Spring 2024]**

**Submitted by:**

| **Name** | **Student ID** |
| --- | --- |
| Sofia Noor Rafa | 2020 - 1 - 60 - 226 |
| Nisarga Mridha | 2020 - 2 - 60 - 010 |
| Most. Mahbuba Yasmin | 2020 - 2 - 60 - 170 |

**Submitted to:**

Jesan Ahammed Ovi

Senior Lecturer

Department of Computer Science & Engineering

**Date of Submission: 9th June 2024**

**Protein Data Bank Management System**

1. **Description:**

In this project, an Object-Oriented Database (OODB) schema was designed and implemented within a relational database environment. The schema includes object types representing various types of proteins, such as enzymes, antibodies, transport proteins, and hormonal proteins. These object types encapsulate attributes and methods relevant to each protein type, demonstrating concepts like inheritance and encapsulation. Nested tables are utilized to store collection attributes efficiently, while PL/SQL queries and cursors are employed to retrieve and manipulate data. The project also highlights the importance of database design considerations, such as data modeling, integrity constraints, and encapsulation of business logic within stored procedures. Overall, this project serves as a practical demonstration of applying object-oriented principles to database design and management, showcasing the flexibility and efficiency of OODB concepts within a relational database framework.

1. **Creating Object Types:**

* **Protein Type:**

*CREATE OR REPLACE TYPE ProteinType AS OBJECT (*

*ID NUMBER,*

*Name VARCHAR2(100)*

*);*

*/*

* **Amino Acid List Type:**

*CREATE OR REPLACE TYPE AminoAcidList AS TABLE OF VARCHAR2(3);*

*/*

1. **Creating Tables:**

**Tables:**

* **ProteinTable**
* **EnzymeTable**
* **AntibodyTable**
* **TransportProteinTable**
* **StructuralProteinTable**

**Query:**

***-- Protein Table***

*CREATE TABLE ProteinTable (*

*protein\_data ProteinType,*

*sequence AminoAcidList,*

*PRIMARY KEY (protein\_data.ID)*

*) NESTED TABLE sequence STORE AS amino\_acid\_sequences;*

*/*

***-- Enzyme Table***

*CREATE TABLE EnzymeTable (*

*protein\_data ProteinType,*

*PRIMARY KEY (protein\_data.ID)*

*);*

*/*

***-- Antibody Table***

*CREATE TABLE AntibodyTable (*

*protein\_data ProteinType,*

*PRIMARY KEY (protein\_data.ID)*

*);*

*/*

***-- TransportProtein Table***

*CREATE TABLE TransportProteinTable (*

*protein\_data ProteinType,*

*PRIMARY KEY (protein\_data.ID)*

*);*

*/*

***-- Structural Protein Table***

*CREATE TABLE StructuralProteinTable (*

*protein\_data ProteinType,*

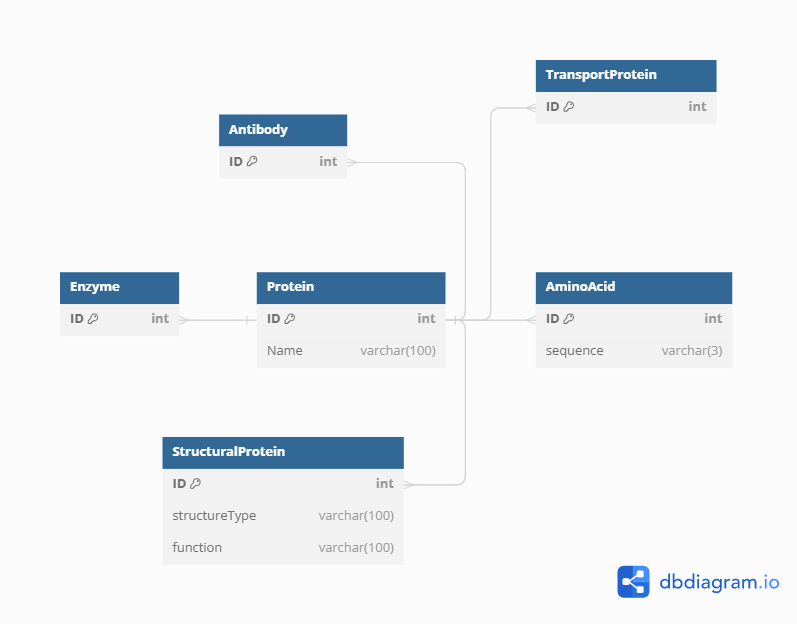
*structureType VARCHAR2(100),*

*function VARCHAR2(100),*

*PRIMARY KEY (protein\_data.ID)*

*);*

*/*

**

*figure 1. Schema Diagram*

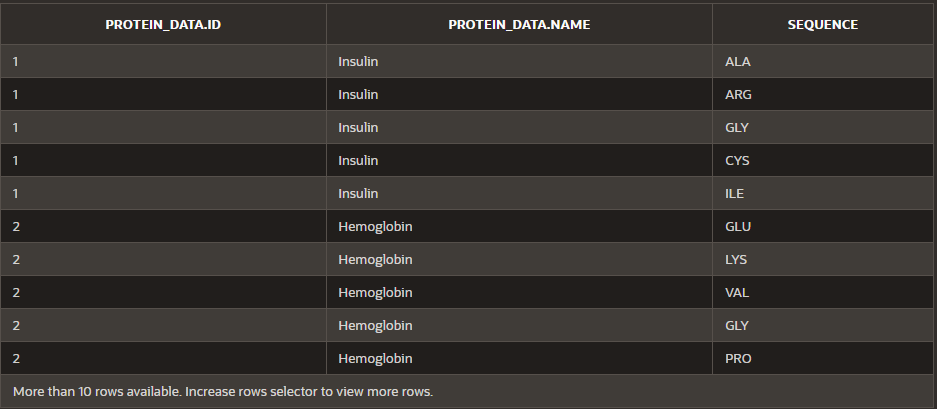
1. **Inserting Data:**

***-- Insert data into ProteinTable***

*INSERT INTO ProteinTable (protein\_data, sequence) VALUES ( ProteinType(1, 'Insulin'), AminoAcidList('ALA', 'ARG', 'GLY', 'CYS', 'ILE') );*

*INSERT INTO ProteinTable (protein\_data, sequence) VALUES ( ProteinType(2, 'Hemoglobin'), AminoAcidList('GLU', 'LYS', 'VAL', 'GLY', 'PRO') );*

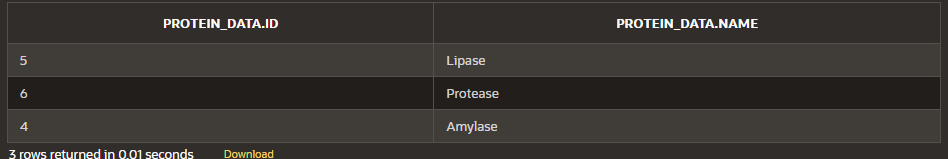
*INSERT INTO ProteinTable (protein\_data, sequence) VALUES ( ProteinType(3, 'Fibrinogen'), AminoAcidList('ASN', 'THR', 'LEU', 'MET', 'SER') );*

**

*figure 2. Data inserted into ProteinTable*

***-- Insert data into EnzymeTable***

*INSERT INTO EnzymeTable (protein\_data) VALUES ( ProteinType(4, 'Amylase') ); INSERT INTO EnzymeTable (protein\_data) VALUES ( ProteinType(5, 'Lipase') ); INSERT INTO EnzymeTable (protein\_data) VALUES ( ProteinType(6, 'Protease') );*

**

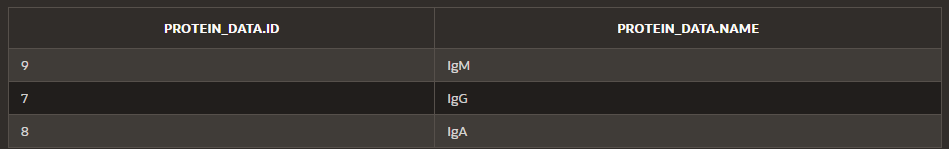
*figure 3. Data inserted into EnzymeTable*

***-- Insert data into AntibodyTable***

*INSERT INTO AntibodyTable (protein\_data) VALUES ( ProteinType(7, 'IgG') );*

*INSERT INTO AntibodyTable (protein\_data) VALUES ( ProteinType(8, 'IgA') );*

*INSERT INTO AntibodyTable (protein\_data) VALUES ( ProteinType(9, 'IgM') );*

**

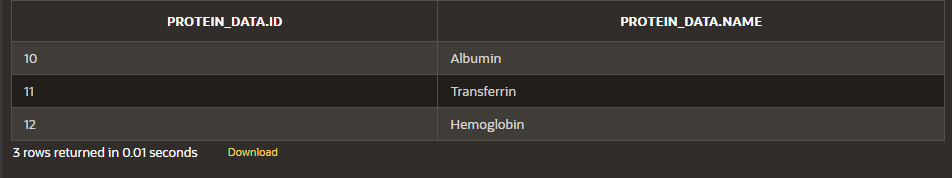
*figure 4. Data inserted into AntibodyTable*

***-- Insert data into the TransportProteinTable***

*INSERT INTO TransportProteinTable (protein\_data) VALUES ( ProteinType(10, 'Albumin') );*

*INSERT INTO TransportProteinTable (protein\_data) VALUES ( ProteinType(11, 'Transferrin') );*

*INSERT INTO TransportProteinTable (protein\_data) VALUES ( ProteinType(12, 'Hemoglobin'));*

**

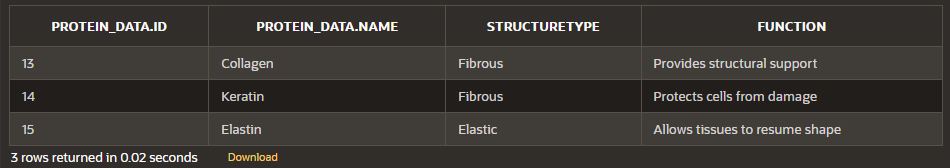
*figure 5. Data Inserted into TransportProteinTable*

***-- Insert data into the StructuralProteinTable***

*INSERT INTO StructuralProteinTable (protein\_data, structureType, function) VALUES ( ProteinType(13, 'Collagen'), 'Fibrous', 'Provides structural support' );*

*INSERT INTO StructuralProteinTable (protein\_data, structureType, function) VALUES ( ProteinType(14, 'Keratin'), 'Fibrous', 'Protects cells from damage' );*

*INSERT INTO StructuralProteinTable (protein\_data, structureType, function) VALUES ( ProteinType(15, 'Elastin'), 'Elastic', 'Allows tissues to resume shape' );*

**

*figure 5. Data Inserted into StructuralProteinTable*

1. **Querying Data:**
2. **Count the total number of proteins stored in the ProteinTable.**

**Query:**

*DECLARE*

*total\_proteins NUMBER;*

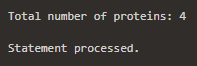
*BEGIN*

*SELECT COUNT(\*) INTO total\_proteins FROM ProteinTable;*

*DBMS\_OUTPUT.PUT\_LINE('Total number of proteins: ' || total\_proteins);*

*END;*

*/*

**

*figure 6. Output of Query 1*

1. **Retrieve the names of all enzymes stored in the EnzymeTable.**

**Query:**

*DECLARE*

*enzyme\_name VARCHAR2(100);*

*CURSOR enzyme\_cursor IS*

*SELECT et.protein\_data.Name*

*FROM EnzymeTable et*

*WHERE et.protein\_data.ID IS NOT NULL;*

*BEGIN*

*OPEN enzyme\_cursor;*

*LOOP*

*FETCH enzyme\_cursor INTO enzyme\_name;*

*EXIT WHEN enzyme\_cursor%NOTFOUND;*

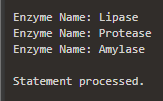
*DBMS\_OUTPUT.PUT\_LINE('Enzyme Name: ' || enzyme\_name);*

*END LOOP;*

*CLOSE enzyme\_cursor;*

*END;*

*/*

**

*figure 7. Output of Query 2*

1. **Calculate the average length of protein sequences in the ProteinTable.**

**Query:**

*DECLARE*

*avg\_sequence\_length NUMBER;*

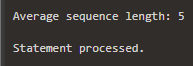
*BEGIN*

*SELECT AVG(CARDINALITY(sequence)) INTO avg\_sequence\_length FROM ProteinTable;*

*DBMS\_OUTPUT.PUT\_LINE('Average sequence length: ' || avg\_sequence\_length);*

*END;*

*/*

**

*figure 8. Output of Query 3*

1. **List the names of antibodies with ‘Ig’ prefix stored in the AntibodyTable.**

**Query:**

*DECLARE*

*antibody\_name VARCHAR2(100);*

*CURSOR antibody\_cursor IS*

*SELECT at.protein\_data.Name*

*FROM AntibodyTable at*

*WHERE at.protein\_data IS NOT NULL*

*AND at.protein\_data.Name LIKE 'Ig%';*

*BEGIN*

*OPEN antibody\_cursor;*

*LOOP*

*FETCH antibody\_cursor INTO antibody\_name;*

*EXIT WHEN antibody\_cursor%NOTFOUND;*

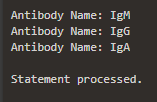
*DBMS\_OUTPUT.PUT\_LINE('Antibody Name: ' || antibody\_name);*

*END LOOP;*

*CLOSE antibody\_cursor;*

*END;*

*/*

**

*figure 9. Output of Query 4*

1. **Find the total number of unique amino acids across all protein sequences in the ProteinTable.**

**Query:**

*DECLARE*

*total\_unique\_acids NUMBER := 0;*

*BEGIN*

*FOR protein\_rec IN (SELECT sequence FROM ProteinTable) LOOP*

*FOR amino\_acid IN (SELECT COLUMN\_VALUE FROM TABLE(protein\_rec.sequence)) LOOP*

*total\_unique\_acids := total\_unique\_acids + 1;*

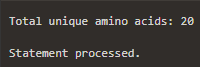
*END LOOP;*

*END LOOP;*

*DBMS\_OUTPUT.PUT\_LINE('Total unique amino acids: ' || total\_unique\_acids);*

*END;*

*/*

**

*figure 10. Output of Query 5*

1. **List all proteins whose names start with ‘Hemo’.**

**Query:**

DECLARE

protein\_name VARCHAR2(100);

CURSOR protein\_cursor IS

SELECT pt.protein\_data.Name

FROM ProteinTable pt

WHERE pt.protein\_data.Name LIKE 'Hemo%';

BEGIN

OPEN protein\_cursor;

LOOP

FETCH protein\_cursor INTO protein\_name;

EXIT WHEN protein\_cursor%NOTFOUND;

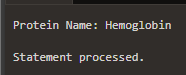
DBMS\_OUTPUT.PUT\_LINE('Protein Name: ' || protein\_name);

END LOOP;

CLOSE protein\_cursor;

END;

/

**

*figure 11. Output of Query 6*

1. **Find and display proteins that do not have associated structures in the ‘StructuralProteinTable’.**

**Query:**

*DECLARE*

*protein\_id NUMBER;*

*protein\_name VARCHAR2(100);*

*CURSOR protein\_cursor IS*

*SELECT pt.protein\_data.ID, pt.protein\_data.Name*

*FROM ProteinTable pt*

*WHERE pt.protein\_data.ID NOT IN (SELECT st.protein\_data.ID FROM StructuralProteinTable st);*

*BEGIN*

*OPEN protein\_cursor;*

*LOOP*

*FETCH protein\_cursor INTO protein\_id, protein\_name;*

*EXIT WHEN protein\_cursor%NOTFOUND;*

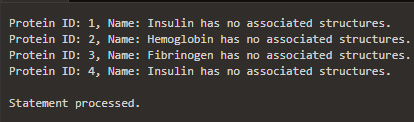
*DBMS\_OUTPUT.PUT\_LINE('Protein ID: ' || protein\_id || ', Name: ' || protein\_name || ' has no associated structures.');*

*END LOOP;*

*CLOSE protein\_cursor;*

*END;*

*/*

**

*figure 12. Output of Query 7*

1. **Display the names and functions of structural proteins that are elastic.**

**Query:**

*DECLARE*

*v\_protein\_name VARCHAR2(100);*

*v\_protein\_function VARCHAR2(100);*

*BEGIN*

*FOR protein\_rec IN (*

*SELECT s.protein\_data.Name AS protein\_name, s.function*

*FROM StructuralProteinTable s*

*WHERE s.structureType = 'Elastic'*

*) LOOP*

*v\_protein\_name := protein\_rec.protein\_name;*

*v\_protein\_function := protein\_rec.function;*

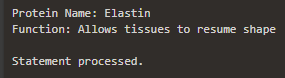
*DBMS\_OUTPUT.PUT\_LINE('Protein Name: ' || v\_protein\_name);*

*DBMS\_OUTPUT.PUT\_LINE('Function: ' || v\_protein\_function);*

*END LOOP;*

*END;*

*/*

**

*figure 13. Output of Query 8*

1. **Count and display the number of structural proteins with a structure type of 'Fibrous'.**

**Query:**

*DECLARE*

*v\_fibrous\_count NUMBER;*

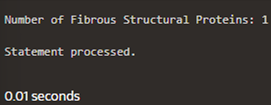
*BEGIN*

*SELECT COUNT(\*) INTO v\_fibrous\_count FROM StructuralProteinTable WHERE structureType = 'Fibrous';*

*DBMS\_OUTPUT.PUT\_LINE('Number of Fibrous Structural Proteins: ' || v\_fibrous\_count);*

*END;*

*/*



*figure 14. Output of Query 9*

1. **Retrieve the enzyme ID, name, and activity of enzymes with an activity related to 'Oxidoreductase'.**

**Query:**

*DECLARE*

*CURSOR oxidoreductase\_enzymes\_cursor IS*

*SELECT proteinID, proteinName, activity*

*FROM EnzymeTable*

*WHERE enzymeclass = 'Oxidoreductase';*

*v\_enzyme\_id EnzymeTable.proteinID%TYPE;*

*v\_enzyme\_name EnzymeTable.proteinName%TYPE;*

*v\_enzyme\_class EnzymeTable.enzymeClass%TYPE;*

*BEGIN*

*OPENoxidoreductase\_enzymes\_cursor;*

*LOOP*

*FETCH oxidoreductase\_enzymes\_cursor INTO v\_enzyme\_id, v\_enzyme\_name, v\_enzyme\_class;*

*EXIT WHEN oxidoreductase\_enzymes\_cursor%NOTFOUND;*

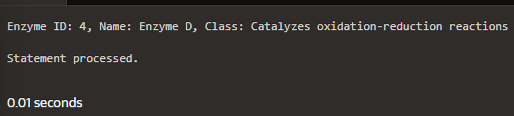
*DBMS\_OUTPUT.PUT\_LINE('Enzyme ID: ' || v\_enzyme\_id || ', Name: ' || v\_enzyme\_name || ', Class: ' || v\_enzyme\_class);*

*ENDLOOP;*

*CLOSE oxidoreductase\_enzymes\_cursor;*

*END;*

*/*

**

*figure 15. Output of Query 10*