5-1: Mapping Entities and Attributes

Exercise 1: Creating a Glossary from the Logical Model

- Gained familiarity with the structure and use of a Logical Model in a database context.
- Learned the steps to create a glossary from the Logical Model, which involves:
    - Opening the Logical Model of the Academic Database.
    - Creating a glossary through the Logical Model node in the browser.
    - Defining the name, description, and classification types for the glossary entries.
- Recognized the importance of glossaries in documenting the meaning of database entities and attributes.

Exercise 2: Forward Engineering the Design to Apply the Glossary and Naming Standard

- Understood how to integrate a glossary into the database design by configuring the Naming Standard settings.
- Learned to apply the glossary to ensure standardized naming when forward engineering models.
- Learned the steps for forward engineering the design:
    - Setting up the Naming Standard preferences to include the glossary.
    - Using the "Engineer" function in the toolbar to apply name translation and use preferred abbreviations.
    - Applying the engineered changes to the Relational Model.
- Understood the significance of maintaining naming consistency across database entities, enhancing clarity and reducing ambiguity.

5-2: Mapping Primary and Foreign Keys

Exercise 1: Observe the mapping of the unique identifiers and relationship in the Relational

1. Compare the Logical Model and the engineered Relational Model to verify:
    a. The Unique Identifiers that have been mapped as Primary Keys
        P next to attribute in relational model
        -ID
        -Student ID
        -Academic Session ID
        -Exam Type
        -Logon ID
        -Building, Room, Date/Time

    b. The Unique Identifiers that have been mapped as Unique Keys
        U next to attribute in relational model
        - Last Name

    c. The Relationships that have been mapped as Foreign Keys
        F/PF next to attribute in relational model
        -Patient Information ID
        -Student ID
        -Academic Session ID
        -Course ID
        -Department ID
        -Online Logon ID
        -Exam Type

Exercise 2: Define table name abbreviations in csv file

- Learned how to create a .csv file containing abbreviations for table names and constraints, ensuring consistency across database naming conventions.
- Gained experience in structuring data in a CSV format by listing plural table names and their corresponding abbreviations in separate columns.
- Practiced saving and managing .csv files for use in the forward engineering process.

Exercise 3: Define Name Template

- Learned how to define name templates (name patterns) for keys, indexes, and constraints using predefined variables.
- Gained an understanding of how to combine predefined variables for different database elements (primary keys, foreign keys, check constraints, unique constraints)
- Recognized the importance of customizing naming conventions to follow specific patterns and standards for clear and structured database design

Exercise 4: Apply Name Template to the Relational Model

- Model using the .csv file containing the abbreviations.
- Understood the significance of maintaining uniform naming conventions throughout the model by applying these templates.
- Practiced re-engineering a Logical Model to a Relational Model to reflect the applied naming standards.

Exercise 5: Select how subtypes are generated in the Relational Model

- how to define how subtypes are mapped to the Academic Database Relational Model.
- Gained experience with the "Single Table" inheritance method for mapping subtypes in relational databases.
- Practiced re-engineering the Relational Model to reflect the changes made in subtype generation.

6-1 : Introduction to Oracle Application Express

Exercise 1: Introduction to Oracle Application Express

- Logged in to the Oracle Application Express and explored the 3 components (SQL Workshop, Application Builder, and Object Browser)
- Learned how to run a script in APEX
- Learned how to use the SELECT * command to return all rows in a table

6-2 : Structured Query Language

Exercise 1: Using Help in Oracle Application Express

- Became familiar with navigating the Help sections within Oracle Application Express.
- Learned about the tools available for managing and viewing database objects using the Object Browser.
- Understood how to execute SQL commands directly within the SQL Workshop.
- knowledge on how to manage and run SQL scripts efficiently within the platform.

6-3: Defining Data Definition Language (DDL)

Exercise 1: Creating Tables Using Oracle Application Express

- Learned how to write SQL Data Definition Language (DDL) statements to create database tables, defining columns, data types, and constraints.

- Gained experience in applying NOT NULL constraints to ensure data integrity during table creation.
- Became proficient in using Oracle Application Express to execute DDL commands and manage database objects.

Exercise 2: Altering Tables

- Learned how to alter existing tables to add primary key, foreign key, and unique constraints.
- Understood how to set default values for specific columns, such as using `SYSDATE` for a timestamp.
- Learned how to modify table properties to set a table to read-only status.

Exercise 3: Creating Composite Primary, Foreign, and Unique Keys

- Gained experience in creating composite primary, foreign, and unique keys, ensuring that combinations of columns enforce data integrity.
- Learned how to create tables with multiple columns as primary or foreign keys, ensuring unique combinations across tables.

6-4 :Defining Data Manipulation

Exercise 1: Inserting Rows in Tables

- Learned how to insert data rows into various tables using SQL `INSERT` statements.
- Became familiar with inserting data for multiple entities (students, faculty, courses, exams, etc.) in the Academic Database.
- Understood how to change the read/write status of tables (like setting the `AD_PARENT_INFORMATION` table to allow updates).
- Using Oracle Application Express for Batch Operations: Learned how to batch upload and execute SQL `INSERT` statements using Oracle APEX's SQL Workshop.

Exercise 2: Updating Rows in Tables

- Learned how to alter an existing table by adding a new column (`DETAILS` in `AD_FACULTY_LOGIN_DETAILS`).
- Gained experience in updating specific columns in an existing table, and modifying the records by updating the newly added column.
- Understood the importance of handling composite primary keys when updating records, such as using `LOGIN_DATE_TIME` in `AD_FACULTY_LOGIN_DETAILS`.

6-5: Defining Transaction Control

Exercise 1: Controlling Transactions

- Gained a fundamental understanding of transaction control concepts using SQL statements like `COMMIT`, `ROLLBACK`, and `SAVEPOINT`.
- Learned how to set a savepoint to mark a point within a transaction where you can roll back to, preserving previous changes.
- Understood that issuing a `ROLLBACK` to a savepoint will undo all changes made after the savepoint, but preserve changes made before it.
- Practiced how to apply `INSERT`, `UPDATE`, and `DELETE` operations in sequence, with savepoints marking stages, allowing controlled rollbacks.

- Realized that rolling back to an earlier savepoint, such as `UPDATE_DONE`, would undo changes made after that point (like deletions) while keeping the preceding updates intact.

6-6: Retrieving Data

Exercise 1: Retrieving Columns from Tables

- Learned how to write SQL queries to either select all columns from a table or target specific columns for more focused data retrieval.
- Wrote simple queries to view all inserted data, enhancing understanding of data retrieval and data structure within the Academic Database.
- Learned how to retrieve and analyze data by querying for the grades obtained by each student for every exam they attempted, reinforcing understanding of JOIN operations between tables.
- Applied conditional logic to determine a student's eligibility to take exams based on class attendance, practicing with WHERE clauses.
- Retrieved date and time information for faculty login events, practicing datetime handling in SQL.
- Wrote queries to display departmental heads, reinforcing how to work with relationships and hierarchical data.
- Used SQL functions to concatenate literal text with data (e.g., combining student ID and name into a custom format).
- Practiced using the `DISTINCT` keyword to retrieve unique values (e.g., distinct exam types from the AD_EXAMS table).

6-7: Restricting Data Using WHERE Statement

Exercise 1: Restricting Data Using SELECT

- Gained experience with comparison operators like `=`, `>`, `<`, `BETWEEN`, and `LIKE` to narrow down query results.
- Learned how to combine multiple conditions using logical operators (`AND`, `OR`, `NOT`) to refine query results further.
- Practiced filtering data by specific session details, such as displaying courses from the Spring and Fall sessions.
- Applied the `BETWEEN` operator to filter students based on their scores within a specific range (e.g., scores between 65 and 70).
- Used date comparisons to display data (e.g., students registered after a specific date).
- Practiced using `LIKE` for pattern matching to retrieve students whose names start with a particular letter (e.g., students whose first name starts with "J").
- Worked with queries that select students enrolled in specific courses or courses offered by specific departments.

Practice 6-8: Sorting Data Using ORDER BY

Exercise 1: Sorting Data

- Learned to sort rows based on individual fields, such as sorting students by their registration year or departments by department ID.

- Practiced sorting data using multiple columns, like ordering exam results by `STUDENT_ID` and `COURSE_ID` simultaneously.

- Sorted data based on calculated values, such as displaying the percentage of days students took off and organizing the results by the calculated percentage.

- Learned to rank data by selecting the top results, such as displaying the top 5 students based on their exam grades.

- Organized parent records by `PARENT_ID`, demonstrating how to order data by primary key fields.

6-9 : Joining Tables Using JOIN

- Accessed data from more than one table using both equijoins (using equality) and non-equijoins (using inequality conditions).

- Used outer joins to include data that typically does not meet the join condition, such as rows with no matching counterparts in other tables.

- Understood how to generate a Cartesian Product (a result of combining every row of one table with every row of another).

- Understood the output generated by a `CROSS JOIN`, combining all rows from `AD_EXAMS` and `AD_EXAM_TYPES`.