## **BCS 4103 Advanced Database Systems**

**Project: Optimizing Database Performance with PostgreSQL and Node.js**

### **Background**

You are a team of database developers tasked with creating and optimizing the performance of database for datasets of a Machine Learning repository. Your objective is to improve the overall efficiency of the database system using PostgreSQL, Node.js for [API development, and Swagger for API documentation.](https://www.codecademy.com/learn/api-development-with-swagger-and-openapi) You will focus on implementing and optimizing triggers and stored procedures to achieve this goal.

### **Scenario**

Your team/group will select a particular dataset from [UC Irvine Machine Learning Repository](https://archive.ics.uci.edu/datasets) The team will build a database is using PostgreSQL for the chosen dataset and the backend API developed using Node.js. In addition, you may utilize json files through [PostgreSQL JSON](https://neon.tech/postgresql/postgresl-tutorial/postgresql-json) to handle the data types.

For this activity, you will be required to utilize your OCI account to enable efficient database connections.

### **Requirements**

1. **Schema Design and Data Population**
   * Design and Review a database schema for optimal usage.
   * Populate the database with a significant amount of sample data to simulate real-world usage.
2. **Backend API Development**
   * Develop RESTful APIs using Node.js and Express.js.
   * Ensure efficient database connections and query handling.
3. **API Documentation**
   * Integrate Swagger to document the APIs.
   * Provide clear and comprehensive documentation for all API endpoints.
4. **Query Optimization**
   * Implement stored procedures for complex and frequently used queries.
   * Create triggers to automate routine tasks and enforce business rules.

### **Tasks**

#### Task 1: Schema Design and Data Population

* **Review the Schema:** Identify potential improvements, such as normalization, indexing, and partitioning.
* **Populate Data:** Use scripts to populate the database with at least 10,000 records in the key tables.

#### Task 2: Backend API Development

* **Set Up Node.js and Express.js:** Create a Node.js project and set up Express.js.
* **Develop CRUD APIs:** Implement the following endpoints:
  + Get all products
  + Get a specific product by ID
  + Add a new product
  + Update an existing product
  + Delete a product
* **Database Connection:** Ensure efficient connection pooling with PostgreSQL.

#### Task 3: API Documentation with Swagger

* **Integrate Swagger:** Set up Swagger in your Node.js project.
* **Document Endpoints:** Provide documentation for all the API endpoints developed in Task 2.
* **Interactive Documentation:** Ensure the Swagger UI is accessible and allows testing of the APIs.

#### Task 4: Query Optimization

* **Stored Procedures:** Write stored procedures for complex queries such as calculating total sales for a specific period.
* **Triggers:** Create triggers to automatically update inventory levels when a new order is placed.
* **Performance Analysis:** Use PostgreSQL’s EXPLAIN and ANALYZE commands to measure the performance of your queries before and after optimization.

### **Deliverables**

1. Optimized Database Schema
2. Node.js Project with CRUD APIs
3. Swagger Documentation
4. Stored Procedures and Triggers
5. Performance Analysis Report

### **Evaluation Criteria**

* **Correctness and Efficiency:** Functionality of the database optimizations and API endpoints.
* **Documentation:** Quality and completeness of API documentation using Swagger.
* **Performance Improvement:** Demonstrated improvement in query performance through stored procedures and triggers.
* **Code Quality:** Clean, well-documented, and maintainable code.

**Cover page template:**

1. Unit Code
2. Group Name/Number
3. Student number of Group member/s (only active members should be listed)
4. Title of Document/report
5. Github folder name & link

**Deadline**

***1 August 2025*** and we will do demonstration/presentation on 5 August 2025.