Assignment Two

Start Assignment

- Due 16 May by 23:59
- Points 65
- Submitting a file upload
- Available after 17 Mar at 0:00

Important Information

This is the second assignment for this module and is worth **65%** of the overall module mark.

All work must be done individually.

The following learning outcomes will be assessed:

1. Demonstrate critical appreciation of current and new data models and database systems for traditional and Big Data systems.
2. Appraise current and emerging trends in database systems and their application in the real world.
3. Design and develop database systems using a range of different database development tools.
4. Evidence of critical evaluation of the major developments and issues of databases within the database arena and their support in various application areas.

You are required to submit your work within the bounds of the University Infringement of Assessment Regulations (see the **Student Handbook (https://services.sunderland.ac.uk/academicregistry/studenthandbook/)**). Plagiarism, paraphrasing and downloading large amounts of information from external sources, will not be tolerated, and will be dealt with severely. You should make full use of any source material, which would normally be an occasional sentence and/or paragraph (referenced) followed by your own critical analysis/evaluation. You will receive no marks for work that is not your own. Your work may be subject to checks for originality which can include use of an electronic plagiarism detection service.

Where you are asked to submit an individual piece of work, the work must be entirely your own. The safety of your assessments is your responsibility. You must not permit another student access to your work.

Where referencing is required, unless otherwise stated, the Harvard referencing system must be used (see your Programme Guide).

Please ensure that you retain a duplicate of your assignment. We are required to send samples of student work to the external examiners for moderation purposes. It will also safeguard in the unlikely event of your work going astray.

Assignment Requirements

Your assignment is to **design, develop and query a database** using multiple DBMS's, i.e., **PostgreSQL** and **MongoDB**, to provide a brief comparison of relational and NoSQL document store database technologies.

If you are at an off-campus centre and have been using Oracle then you may use Oracle instead of PostgreSQL for this assignment.

Use of any other DBMS than those specified above will result in a mark of zero where appropriate.

For a **case study of your choice**, perform the following tasks:

1. Provide a **UML class diagram** (NOT an ER diagram) showing the main entities and attributes in your design and the relationships between them. The UML class diagram MUST be **data model independent**, i.e., you are showing potential entities in the system, NOT tables. Ensure that you include complex relationships (e.g., inheritance hierarchies, aggregation, etc.) in your design. Provide a 200 word critical discussion outlining your rationale for embedding of complex relationships in your scenario. Ensure you correctly cite all references that you have used.

2. Create a complete SQL script file that will run in **PostgreSQL** to implement the design from task 1 as an **OBJECT-RELATIONAL** database. Make sure you make full use of object features, e.g., arrays, inheritance, temporal data, etc.). A purely relational database will be marked at most as Major Errors. Ensure your SQL script file runs without errors, i.e.:

    a. it efficiently drops then creates the database objects (tables, user defined types, stored procedures, triggers, etc.),
    b. implements suitable data integrity constraints (using both constraints as well as triggers), and
    c. inserts sufficient sample data for your queries below.

    Make sure that you provide the complete SQL script file within your submitted report, with a .sql extension. Any auto-generated code exported from the DBMS or any other tool will be given a mark of 0 for this part, and may lead to an academic misconduct investigation.

3. Develop a set of documents using **MongoDB** to develop the equivalent **document store** database for your design in task 1. Ensure you make use of appropriate document store features including, for example, nested documents, arrays, temporal data, etc. Do not simply replicate the PostgreSQL implementation but **make full use of MongoDB document store features**. Make sure the sample data you use matches the data stored in your database from task 2. You must provide the complete script file for creating your MongoDB database (i.e. the *collection_name.insert* commands), do not just provide a list of documents outputted from the interface, otherwise you will get 0 marks for this part.

4. Provide a brief critical discussion, of no more than 300 words, identifying and illustrating (i.e. by including examples) of how you have used MongoDB document store features to convert your scenario from an object-relational to a NoSQL document store system. As part of this discussion explain how you have used NoSQL document store features of MongoDB to provide a suitable NoSQL system (e.g. how have you reduced the number of collections, how have you used nested documents and arrays, etc.). Ensure you correctly cite all references that you have used.

5. Develop the following queries (i.e., **SQL SELECT** statements) in **PostgreSQL** on your database from task 2 above. Provide a short, natural language, description of each query. Ensure that you develop queries which include:

    a. A join of three or more tables – **you must use multiple types of join operations in this query** (e.g., inner join, left/right/full outer joins, etc.) and the query must include a restriction on the rows selected.

b. A query which uses one (or more) of the UNION, MINUS or INTERSECT operators. Note this **must involve more than one table** in the result, e.g., a UNION or other operation involving two different tables, not just on one table.

c. A query which requires use of either an array or table inheritance (or nested tables/subtypes if using Oracle) or similar. Note: **sub-queries are NOT acceptable** as they do not meet the requirements of this task.

d. A query using SQL temporal features . This must use **temporal features covered in the tutorials** (e.g., timestamps, intervals, etc.), and not just a simple time/date comparison.

e. A query using SQL OLAP (e.g., ROLLUP, CUBE, PARTITION) features. A simple GROUP BY or, for example a ROLLUP/CUBE over only one dimension, is not acceptable.

You must only submit **FIVE SQL queries maximum**, i.e., one for (a), one for (b), etc. Any other queries will not be marked. This means, for example, that if you submit two queries for part (a) then only the first query will be marked. **It is required that at least one of your queries will be embedded within a stored procedure, and one query will include a stored function** that you have implemented.

6. Using **MongoDB**, implement queries which produce an **equivalent** result (i.e., each returns the same data in an appropriate form) to those in task 5a-e above. If you are unable to complete an identical query in MongoDB, then you should write a query which is as similar as possible in functionality to the SQL query. Ensure you use the native MongoDB query language and NOT SQL for your MongoDB queries. You can provide either tree or tabular-based output in your screenshots depending on which is most appropriate for each query. Ensure any nested documents, etc., are appropriately expanded fully in your results.

You need to ensure therefore that your case study is of sufficient detail to be able to complete tasks 1-6 above. As a suggestion, you should aim to have **between four and six entities** (excluding subtypes) within your design which you would then represent in your two implementations using tables and document collections as appropriate. You must consider complex relationships between the entities such as inheritance and aggregation in your design. All table, document, creation, etc., and data insertion must be included in the code for tasks 2 and 3. You should not be creating tables, etc., in your queries, all queries must only use existing data.

Marks given will consider the challenge of your case study and the challenge of the queries that you develop. For example, a query using temporal functions such as extract/intervals will gain more marks than a simple query on a date which is not at the appropriate level for this module.

Submission Requirements

Your submission for this assignment is a technical report which includes your solutions to tasks 1-6 above.

You are required to submit the following three files within **one ZIP file**:

1. A **technical report** which includes your solutions to tasks 1-6 above. **A template (which you must use) for this document is provided here (https://canvas.sunderland.ac.uk/courses/75448/files/16028190?wrap=1) (https://canvas.sunderland.ac.uk/courses/75448/files/16028190/download?download_frd=1)** . All SQL and MongoDB code and output must be included as well as screenshots to demonstrate that all your database creation, inserts and queries work. The file must be saved as either a Word (i.e., .docx) or Open Office document (i.e., .odt) format.

2. An **SQL script file** (with a .sql extension) containing all of the PostgreSQL code for task 2.

3. An equivalent script file (with either a .js or .json extension) containing all the **MongoDB code** for task 3.

4. The completed **cover sheet (https://canvas.sunderland.ac.uk/courses/75448/files/15220796?wrap=1) (https://canvas.sunderland.ac.uk/courses/75448/files/15220796/download?download_frd=1)** . Failure to include this may lead to your assignment being treat as a non-submission. You must include this as the first page of your submitted technical report.

Ensure that you use your full name as the name of the ZIP file (i.e., FIRSTNAME_LASTNAME.zip). Other types of ZIP file are not acceptable.

Staying within the bounds of University of Sunderland regulations, you should make full use of any source material available to you (particularly journals, conference papers or technical reports). Any form of academic misconduct, including plagiarism, collusion and paraphrasing, will not be tolerated, and will be dealt with under University Infringement regulations which you can find in the **Student Handbook (https://services.sunderland.ac.uk/academicregistry/studenthandbook/)** . If you are unclear about how to reference correctly then please ask.

Generative AI may be used for generating sample data for your assignment, and for checking your understanding of any concepts you don’t understand. Any other use of Generative AI or any other automated tools for writing any other parts of this assignment (e.g., diagrams, code, written reports) will result in 0 marks for each affected element and your work being referred for an academic misconduct investigation.

Any late submissions will be dealt with as per the university’s assessment regulations which you can find in the student handbook.

CET341 Assignment Two 2024/5

Criteria	Ratings					Pts
Task 1 UML class diagram and 200-word discussion	**7 Pts** **Exceeds expected standard** Case study at a very good level and challenge. UML class diagram correctly describes the case study, notation is used correctly, clearly showing attributes and entities with a very good attempt to embed complex relationships. A good number of entities is described to demonstrate the complexity of the scenario. Very clear rationale presented.	**5.5 Pts** **Meets expected standard** Case study is of an appropriate level and challenge. UML class diagram correctly describes the case study, notation is used correctly, clearly showing attributes and entities with some attempt to embed complex relationships. Suitable number of entities is described. Good rationale presented.	**3.5 Pts** **Just below expected standard** Case study is almost at an appropriate level and challenge. UML class diagram correctly describes the case study, notation is mostly correct, but attributes and entities are not clearly shown with limited attempt to embed complex relationships. Suitable number of entities is described. Some rationale presented but needs more thought.	**2 Pts** **Fails to meet expected standard** Case study is not at an appropriate level and lacks challenge. UML class diagram provided but does not suitably describe the case study, and incorrect notation is used. Attributes and entities are not clearly shown and no complex relationships. Unsuitable number of entities is described. Poor or no rationale presented.	**0 Pts** **Not attempted** Not attempted or very poor attempt.	7 pts
Task 2 PostgreSQL Database Creation and Data Insertion	**10 Pts** **Exceeds expected standard** Correctly working SQL script file which can be run without error and completely matches the scenario described, utilising some advanced features of the DBMS. Database integrity rules are correctly implemented including all types	**7.5 Pts** **Meets expected standard** Correctly working SQL script file which can be run without error and matches the scenario described. Database integrity rules are correctly implemented. Good use of object features (e.g. inheritance/user-defined types/aggregation)	**5 Pts** **Just below expected standard** SQL script file developed and mostly error free but does not fully match scenario. Not all database integrity rules correctly specified. Limited object features (e.g. inheritance/user-defined types/aggregation) incorporated. Some screenshots	**2.5 Pts** **Fails to meet expected standard** SQL script file developed but does not match scenario and contains errors. No, or limited, integrity rules specified. No object-relational features	**0 Pts** **Not attempted** Not attempted or very poor attempt.	10 pts

Criteria	Ratings					Pts
	of integrity constraint. Very good use good use of object features (e.g. inheritance/user-defined types/aggregation) incorporated. An excellent set of screenshots are provided to fully demonstrate that the code works. An excellent set of sample data is provided. A stored procedure and function have been included.	incorporated. Screenshots are provided to fully demonstrate that the code works. A good set of sample data is provided. A stored procedure and/or function has been included.	are provided but more could have been provided to clearly demonstrate that that system works. Some sample data is provided.	implemented. No, or very limited, screenshots provided. No, or limited, sample data is provided.		
Task 3 MongoDB Database Creation and Data Insertion	**10 Pts** **Exceeds expected standard** MongoDB database code provided which correctly implements the case study using appropriate and some advanced document store facilities throughout. MongoDB code will work without error. Sample data matches sample data provided in PostgreSQL and correctly uses some advanced MongoDB concepts. Screenshots have been provided to fully demonstrate that code works.	**7.5 Pts** **Meets expected standard** MongoDB database code provided which correctly implements the case study using appropriate document store facilities. MongoDB code will work without error. Sample data matches sample data provided in PostgreSQL and correctly uses MongoDB concepts. Screenshots have been provided to demonstrate that code works	**5 Pts** **Just below expected standard** Good attempt at the MongoDB database but shows some issues in understanding of how to implement a document store. Code is mostly error free and a good matching data set has been provided. Some screenshots provided.	**2.5 Pts** **Fails to meet expected standard** Poor attempt at the MongoDB database which shows lack of understanding of how to create a document store. Code contains many errors. Limited or no sample data provided. No, or limited, screenshots provided.	**0 Pts** **Not attempted** Not attempted or very poor attempt.	10 pts

Criteria	Ratings					Pts
	Very good discussion with very good use of references.					
Task 4 300-word MongoDB Critical Discussion	**8 Pts** **Exceeds expected standard** Very good critical discussion and comparison and a comprehensive set of features discussed and compared. Very good use of references.	**6 Pts** **Meets expected standard** Good critical discussion and comparison and of a good set of features discussed and compared. Good use of references.	**4 Pts** **Just below expected standard** Some discussion and and of a good set of features discussed and compared, but lacks in criticality. Good use of references.	**2 Pts** **Fails to meet expected standard** Limited discussion provided. No/limited use of references	**0 Pts** **Not attempted** Not attempted or very poor attempt.	8 pts
Task 5 PostgreSQL Queries	**15 Pts** **Exceeds expected standard** All SQL queries developed, work and are at an excellent level of challenge. Natural language descriptions are given which clearly describe each query. Screenshots are given which show the correct output from each query.	**12 Pts** **Meets expected standard** All SQL queries developed, work, and are at an appropriate level of challenge. Natural language description of each query is given. Screenshots are given which show the correct output from each query.	**7.5 Pts** **Just below expected standard** All queries attempted but may contain errors and not at an appropriate level of challenge. Natural language description is given but does not clearly describe the query. Screenshots are given which show data output.	**4 Pts** **Fails to meet expected standard** Not all SQL queries attempted and contain errors. Not at an appropriate level of challenge. No, or poor, natural language description is given. No, or limited, screenshots provided.	**0 Pts** **Not attempted** Not attempted or very poor attempt.	15 pts
Task 6 MongoDB Queries	**15 Pts** **Exceeds expected standard** Excellent set of 5 MongoDB queries provided fully using	**12 Pts** **Meets expected standard** 4 or 5 MongoDB queries provided and are a good attempt to match the PostgreSQL	**7.5 Pts** **Just below expected standard** Good attempt at 3 or 4 of the MongoDB queries, using	**4 Pts** **Fails to meet expected standard** Poor attempt at one of more of the MongoDB	**0 Pts** **Not attempted** Not attempted or very poor attempt.	15 pts

Criteria	Ratings					Pts
	appropriate and some advanced MongoDB constructs. Full set of screenshots provided which clearly demonstrate that the queries work with complete results shown.	queries provided, using appropriate MongoDB constructs. Screenshots are provided to fully demonstrate that the queries work, showing appropriate results.	MongoDB constructs where relevant. Some errors in code. Screenshots have been provided which attempt to show that the queries work with appropriate results.	queries. Many errors in code. No, or few, screenshots have been provided.		
Total points: 65						