Project Description

The course project on *Transactions Management* is a venue for students to practice building a distributed database system that supports concurrent multi-user access. Students will once again use the MCO1 dataset to design a three-node distributed database system where they perform concurrent transactions and simulate crash and recovery techniques. The final output will be (1) a **deployed web application** that will be demonstrated during the indicated schedule to the respective STADVDB teachers, and fully tested using a test script showing the results of performing the different test cases described below on your software, and (2) a **technical report** documenting all decisions involved in setting up the nodes, simulating global concurrency control, simulating global crash and recovery, as well as the results and findings from performing the test scripts.

> "A distributed database management system (DDBMS) ensures that changes, additions, and deletions performed on the data at any given location are automatically reflected in the data stored at all the other locations. Therefore, every user always sees data that is consistent with the data seen by all the other users."

Methodology

Students are to form teams with **3 - 4 members** who should be under the same instructor. Indicate your groupings in AnimoSpace People – **MCO2**. To proceed with this project, each team should conduct the following:

Step 1. Read ***Exercise 5: Distributed Databases***

Exercise 5 points you to resources and guide questions in setting up your distributed database system and in designing your web application that includes provisions for data replication and recovery facilities towards supporting concurrent transaction processing while maintaining the database in a consistent state. Your answers in this exercise will comprise part of your *MCO2 Technical Report*.

Step 2. Build a Distributed Database System

Create and deploy three (3) nodes – with three (3) separate computers that are connected and can communicate with each other. Each node should have its own database with the following contents:

- Node 1 (Central Node): All rows in the main database
- Node 2: Partition of rows based on a specified database fragmentation criterion
- Node 3: Partition of rows based on a specified database fragmentation criterion

Notes:

- *Only 1 table with limited columns from the original dataset is needed for this project.*
- *Referencing Exercise 5, design a homogeneous DDB where all nodes employ the same database schema. To implement data fragmentation, Nodes 2 and 3 should not contain overlapping rows. When combined, they yield the full set of rows found in Node 1.*
- *A web application should be loaded into each node to manage the accessibility of the database across multiple sites.*

Step 3. Concurrency Control and Consistency

Create and deploy a web application to simulate concurrency control and replication for the distributed system. The application should be able to show two or more concurrently executing

transactions in two or more nodes and they all leave the database in a consistent state. Simulate concurrency and replication for each of the following three (3) cases and report the results:

a. Case #1: Concurrent transactions in two or more nodes are reading the same data item.
b. Case #2: At least one transaction in the three nodes is writing (update / deletion) and the others are reading the same data item.
c. Case #3: Concurrent transactions in two or more nodes are writing (update / deletion) on the same data item.

Notes:

- *Test each case on each of the following isolation levels: read uncommitted, read committed, read repeatable and serializable. For this particular application, which isolation level supports the highest volume of concurrent transactions while leaving the database in a consistent state? Justify your choice and document your findings in the technical report.*
- *You must code your own replication and update strategies. Updates in either Node 2 or Node 3 should be replicated in the central node. Any updates in the central node should be replicated in either Node 2 or Node 3. Refer to Exercise 5.*
- *How is data transparency supported by your data replication and update strategy?*

Step 4. Global Failure Recovery

Extend the web application to simulate **global** crash and recovery. Show how the distributed system recovers from failures when concurrent transactions are executing. Simulate the system operations (what the web application will do) and report the results of the following four (4) cases:

a. Case #1: When attempting to replicate the transaction from Node 2 or Node 3 to the central node, the transaction fails in writing (insert / update) to the central node.
b. Case #2: The central node eventually recovers from failure (i.e., comes back online) and missed certain write (insert / update) transactions.
c. Case #3: When attempting to replicate the transaction from central node to either Node 2 or Node 3, the transaction fails in writing (insert / update) to Node 2 or Node 3.
d. Case #4: Node 2 or Node 3 is eventually recovers from failure (i.e., comes back online) and missed certain write (insert / update) transactions.

Notes:

- *You must code your own recovery strategy.*
- *The web application must be deployed using the provided CCS Cloud or other 3rd party servers.*
- *How are users shielded from node failure? When a node fails, how does your data replication strategy support the continued availability of the web application?*
- *How does your data replication strategy support recovery of the distributed DB from failure?*
- *What happens during a recovery operation?*

Step 5. Evaluation

Write a test script to cover all the test cases described in Steps 3 and 4. Execute each of your test cases at least 3 times and log the results. Remember that testing should be efficient and effective. The test script should be submitted as part of the Appendix of your Technical Report.

Step 6. Technical Report

Prepare your documentation. Include all design decisions -- setting up the distributed database, simulating the global concurrency, replication strategy, recovery strategy, test methods and results.

Technical Report

Use the prescribed template and follow the outline provided below to prepare your Technical Report. The Technical Report should be at least 4 pages and at most 10 pages excluding References and Appendix.

1. Introduction.
   - Give a general overview of distributed databases and their usage.
   - Then narrow down to your specific use case -- what is your paper all about, why build a distributed database, how does your web application manage the access to the distributed database across the 3 nodes?
   - Cite related literature for your definition of terms.

2. Distributed Database. Using *Exercise 5 Distributed Databases* as your guide,
   - Present the setup of your distributed database, the data stored in each node, and the application use case (functionalities).
   - Is the DB homogeneous or heterogeneous? How is data replicated and fragmented across the nodes?
   - Give an overview of the update and recovery mechanism of your distributed database as a prelude to Sections 3 and 4.
   - Provide illustrations and diagrams as necessary to support and increase the clarity of your discussion.
   - Cite related literature to support your design decisions.
   - Creatively use supporting diagrams to increase the clarity of your discussion.

3. Concurrency Control and Consistency.

   Update Strategy. Referencing *Exercise 05*,
   - Describe your update strategy to maintain database consistency such that updates in either Node 2 or Node 3 are replicated in the central node, and any updates in the central node are replicated in either Node 2 or Node 3? Is it a master-slave or a multi-master setup, or some other setup? Justify.
   - Illustrate with the use of an algorithm (not source code).
   - Which isolation level (*read uncommitted*, *read committed*, *read repeatable* and *serializable*) is most appropriate for your web application to support the highest volume of concurrent transactions while leaving the database in a consistent state? Justify your decision based on your experiments.
   - How is data transparency supported by your data replication and update strategy?

   Methodology. Explain how you simulated concurrency control.
   - Describe the setup of your experiments.
   - Show the concurrent transactions used to simulate the experiment / test cases.
   - How did you validate that the concurrency control, replication and update strategies such that these will leave the database in a consistent state?
   - Show your test results and give your analysis.
   - Use tables and diagrams accordingly to present your setup, test cases, test data, and test results.

   Discussion. Discuss key insights from your replication and update strategies and analysis of results must be supported by citing relevant literature, i.e., how do your results confirm those reported in prior works?

4. Global Failure Recovery.

   Recovery Strategy.

- Why is there a need for recovery strategies?
- What happens during a recovery operation? What facilities did you build to support your distributed database during a recovery operation?
- For each of the node failure cases, how does your data replication strategy support the recovery of your distributed database system from failure?
- Illustrate with the use of an algorithm (not source code).

Methodology. Explain how you simulated recovery from failure.

- Describe the setup of your experiments.
- Present the test cases you used to simulate each of the test cases in global failure recovery.
- How did you validate the correctness of your recovery strategy?
- Show your results and give your analysis.
- Use tables and diagrams accordingly to present your setup, test cases, test data, and test results.

Discussion. Discuss key insights from your recovery strategies and analysis of results must be supported by citing relevant literature, i.e., how do your results confirm those reported in prior works?

5. Discussion. Share your learnings and insights from the conduct of your experiments / simulations.
   - Focus on two key aspects -- (i) concurrency and update strategy, and (ii) recovery strategy.
     - What is the importance of distributed databases?
     - How do proper isolation levels allow transactions to be executed concurrently?
     - Why are data fragmentation and replication necessary? How do these affect data consistency and recovery?
     - How do data replication benefit users even when a node fails?
     - How is data transparency supported by your data replication and update strategy?
     - What facilities need to be in place to support node recovery?
   - Cite relevant references to further support your discussion.

6. Conclusion. Gives (a short recap of your project) with the key strategies employed, and relevant findings.

7. References. Reviewing literature on distributed databases, concurrency control and recovery strategies should be conducted to help you in doing your project and writing your paper.

8. Declarations**.** Academic honesty reflects your true character and personal values. In this part of the paper, please provide the following.

   9.1 **Declaration of Generative AI Usage**. "*During the development of the project and the preparation of the report, the developers / authors used the following AI tools for the indicated tasks:*

   If single AI tool, indicate the [AI tool] and the list of tasks, e.g.,
   - *Define (Explain) concepts such as star and snowflake, dimensional hierarchy, types of transformation*
   - *Suggest a list of analytical reports*
   - *Paraphrasing and grammar checking*

   If multiple AI tools, prepare a 2-column table

   [AI Tool 1] Tasks
   [AI Tool 2] Tasks

After using the indicated tools, the developers / authors reviewed and edited the AI output and take full responsibility in explaining, justifying, critiquing and using the output of AI."

9.2 **Record of Contribution**. Indicate the contributions in the software development and paper writing of each member of the team.

Final Deliverables

The following final deliverables are required to be submitted on **November 29, 2025** through AnimoSpace. The specific date for project presentation will be provided by your instructor:

1. Technical Report
2. Source Code
3. Test Script with results (follow CSSWENG / STSWENG / SOFENGG / INTROSE format)
4. Presentation Slides

- Late submissions will receive 10 points deduction per day late. No submission will be accepted after December 03, 2025.
- Submissions with incomplete requirements (technical report with TurnItIn check; source code, test script, and presentation slides) will receive corresponding deductions accordingly.
- Prepare for a **20-minute live presentation** of your distributed database setup and the various strategies you employed.
    - Follow a presentation script and do not exceed the indicated time limit
    - Focus on the distributed DB setup, update and replication algorithms, recovery algorithms
    - Show proof of implementation by presenting the results from the execution of each test case
    - Show proof of algorithm implementation through software demo
    - End your presentation with your insights and learnings from the project
- Project presentations will be from **November 29 – December 05**. The actual schedule will be provided by your respective teacher.
    - The web application and database must be deployed via the CCS Cloud or any 3rd party cloud server.
    - Be ready to show your distributed database setup, including your database schema, SQL query statements, and source code.
    - Members who are absent will be given a grade of 0.
    - Members may receive individualized grading depending on contribution of work and ability to answer questions during the presentation.

Plagiarized works – works copied from others and AI generated content – will automatically be given a grade of 0.0 for the course.

Grading Rubric

Distributed Database Software and Presentation [50 pts]

	Exemplary	Satisfactory	Inadequate
Building a Distributed Database	[6] Setup 3 **accessible** nodes on separate cloud machines; Each node contains the indicated instance of the DB, with correct fragmentation and replication; Web app is loaded onto each node	[4] Setup 3 accessible nodes on a single machine; Each node contains the indicated instance of the DB, but flawed fragmentation and replication; Web app deployment has flaws	[2] The entire DB runs on a single node and is not distributed.
Concurrency Control – Algorithm / Strategy	[6] Sound global update and replication strategies to transparently maintain consistency across the entire cluster	[3] Minor flaw in the update and/or replication strategy employed for transparent consistency	**[0]** Students relied on 3rd party tool; Cannot explain how it works and the rationale for the choice | No replication strategy
Concurrency Control – Simulation	[10] For the 3 cases: -- clear method employed in designing and executing the experiment -- sound testing of isolation levels	[7] Minor flaw in -- simulating 1-2 test cases -- partial testing of isolation levels	[4] Major flaw in -- simulating 1-2 test cases -- no experiment to support choice of isolation levels	[2] Concurrency simulation in the localhost only**;** DB only updates the local nodes and does not keep the entire cluster updated
Global Failure Recovery – Algorithm / Strategy	[6] Proper facilities in place to support the continued availability of the DDB; Sound error recovery strategy	[3] Does not properly recover from any form of failure; Web app cannot function properly when a node fails	**[0]** Students relied on 3rd party tool; Cannot explain how it works and the rationale for the choice | No recovery strategy
Global Failure Recovery - Simulation	[10] For the 3 cases: -- clear method employed in designing and executing the experiment -- sound testing of isolation levels	[7] Minor flaw in -- simulating 1-2 test cases -- partial testing of isolation levels	[4] Major flaw in -- simulating 1-2 test cases -- no experiment to support choice of isolation levels	[2] Concurrency simulation in the localhost only**;** DB only updates the local nodes and does not keep the entire cluster updated

Presentation – Scope and Method	[6] Concise presentation of setup and simulation, and challenges ❏ DB Schema, Web apps ❏ Update and Replication strategy ❏ Recovery strategy	[4] Explains system setup and strategies (update, replication, recovery), but some parts of the discussion lack details	[2] Vague discussion of the system setup, algorithm design, justification for the strategies, and limitations
Presentation - Organization	[6] Concise presentation of the results and insights gained from the project ❏ Method, Results ❏ Insights, Conclusion	[4] Explains the simulation methods, results and insights but some parts of the discussion lack details and are not fully evident from the experiments	[2] Vague discussion of the simulation methods, results and insights that are not supported by the experiments

Technical Report [50 pts]

Methodology	[10] Clear, concise and thorough discussion: (1) DDB setup, fragmentation and replication decisions, with supporting related work; (2) web app to manage the access to the distributed database; (3) experiment to simulate global concurrency, replication, and recovery, showing evidence of critical and reflective thinking	[7] Description of the decisions made for the DDB setup, fragmentation and replication decisions; web application for data access; experiment are -- insufficient (missing important details) -- not properly supported by relevant literature	[4] Superficial, vague and/or incomplete discussion, **lacking evidence of critical thinking and supporting related work**
Concurrency Control Strategy	[10] Appropriate -- isolation level to support the highest volume of concurrent transactions; -- update and replication strategies and justified based on RRL and evaluation method	[7] Employed strategies properly addresses at least one case of concurrent read/update; **Demonstration** and discussion is sufficient but not thorough	[4] Vague description of update and replication control; **Insufficient demonstration of how concurrency control is enforced** and how the system maintains consistency
Global Failure Recovery Strategy	[10] Employed **strategies** properly recover from transaction and node failures; -- able to perform **replication after node recovers** -- discussion sufficiently describes the situation	[7] Presence of flaws in the employed strategies; Discussion describes the situation but not thorough	[4] Vague description of global failure recovery; **Insufficient description of how global failure recovery is enforced** and how the system recovers from failure

Analysis and Discussion of Results	[10] Careful thought and reflection is evident in presenting the **analysis and the insights gained from the test results** and conduct of the activity; Findings are supported by sound experiments and analysis of results	[7] Can be further enriched by sharing insights and lessons learned from the conduct of the activity; Missing test results; Discussion lacks depth	[4] Incomplete and/or ambiguous discussion of insights and lessons gained from the results and the experience
Language and Format	[10] Evidence of careful choice of words and use of correct grammar; Properly follows ACM publication format; Proper citations and references; Overall evident care in preparing the report, with appropriate diagrams and tables	[7] Has minor flaws in the choice of words and/or grammar; Minor flaw in the use of prescribed template; Missing some citations and/or references; Missing references to Tables/Figures	[4] Numerous issues in choice of words and grammar; Non-compliance with prescribed template; Missing important references and citations