Final Report

Arda Gurcan, Chentian Wu

Date: 2024-08-05

Table of Contents

1. Introduction

1.1. Motivation

why do you choose this problem and this domain? The importance of the application.

1.2. Application.

Decscribe the application

1.3. Task Assignment

Describe the overall organization of the report and task assignment for each team member

2. Implementation

2.1. System Architecture

II.1 Description of the system architecture

2.2. Dataset

II.2 Description of the dataset

2.3. ER Diagram

II.3 ER diagram (final version from the previous checkpoint copied here)

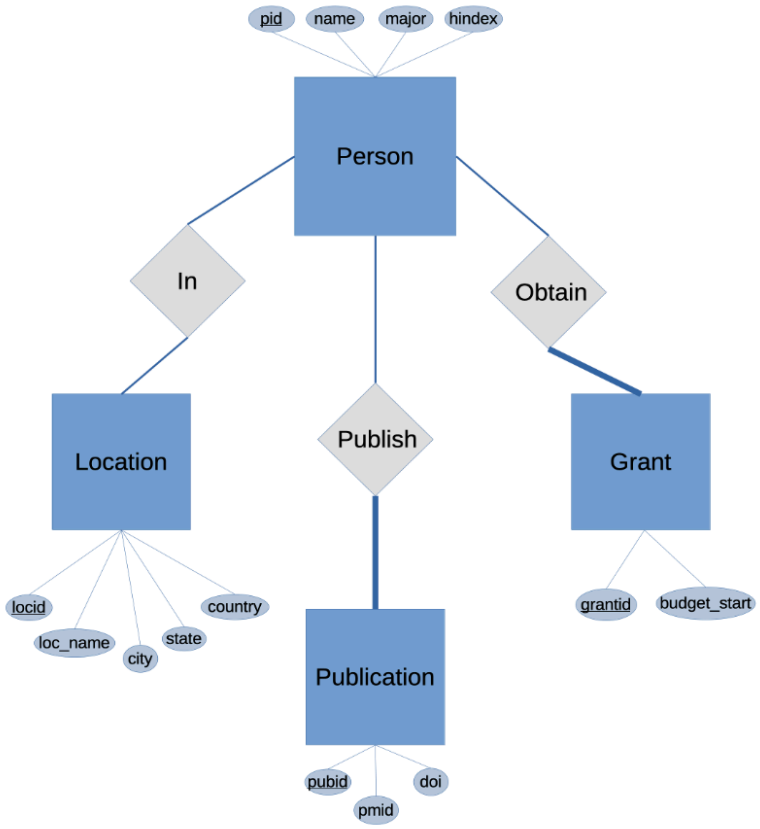

Figure 1: ER Diagram

2.4. Relational Model

II.4 Relational model (final version from the previous checkpoint copied here)

1. **Person**(pid, name, major, hindex)

- `pid` is the primary key

2. **`Location`**(`locid`, `loc_name`, `city`, `state`, `country`)
   - locid is the primary key
3. **`Publication`**(`pubid`, `pmid`, `doi`)
   - `pubid` is the primary key
   - `pmid` and `doi` are also keys
4. **`Grant`**(`grantid`, `budget_start`)
   - `grantid` is the primary key
5. **`In`**(`pid, loc_id`)
   - `(pid, loc_id)` is a composite key
6. **`Publish`**(`pid, pubid`)
   - `(pid, pubid)` is a composite key
7. **`Obtain`**(`pid, grantid`)
   - `(pid, grantid)` is a composite key

2.5. Prototype

II.5 Implementation: description of the prototype

2.6. Evaluation

We mainly design our evaluation from three aspects:

1. **Data Accuracy Evaluation**: **CRUD** directly from backend and check.
2. **Data Consistency Evaluation**: end-to-end data consistency.
3. **Performance Evaluation**: regarding concurrency, indexing etc.

As required in the announcement, we also include the feedback from Checkpoint 4.

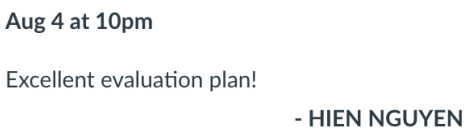

Figure 2: Feedback of Checkpoint 4

2.6.1. Data Accuracy Evaluation

We designed 5 test cases for this evaluation.

- `addNewPerson()`: Insert a new record and verify successful insertion
- `getPersonProfile()`: Read an existing record and verify data correctness
- `updatePublicationAndVerifyChanges()`: Update a record and verify changes are saved
- `deletePublicationAndVerifyRemoval()`: Delete a record and verify it has been removed
- `updateNonExistentPublication()`: Updating non-existent records

Figure 3: `PeopleControllerTest.java`

Figure 4: `PublicationsControllerTest.java`

2.6.2. Data Consistency Evaluation

We designed 5 test cases for this evaluation.

- Perform a create operation on the frontend and verify the data is correctly stored in the backend
- Search for a scholar and verify the end-to-end data consistency
- Filter scholars with specific criteria and verify backend API call
- Test different sorting criteria for scholars and verify corresponding backend calls
- Simulate network interruptions during data transfer and test recovery mechanisms

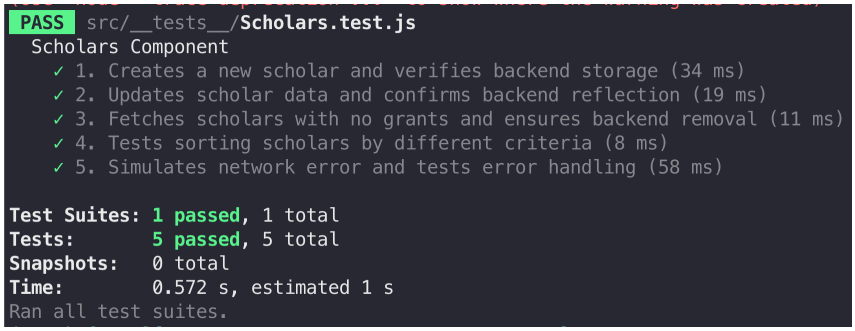

Figure 5: `Scholars.test.js`

2.6.3. Performance Evaluation

We designed 5 test cases for this evaluation.

- `testLargeDatasetQuery()`: Measure response time for querying a large dataset
- `testConcurrentConnections()`: Test the system's ability to handle multiple concurrent database connections
- `testComplexJoinOperation()`: Evaluate the performance of complex join operations or aggregations
- `testIndexImpact()`: Measure the impact of indexing on query performance
- `testSustainedLoad()`: Test the system's performance under sustained load over an extended period

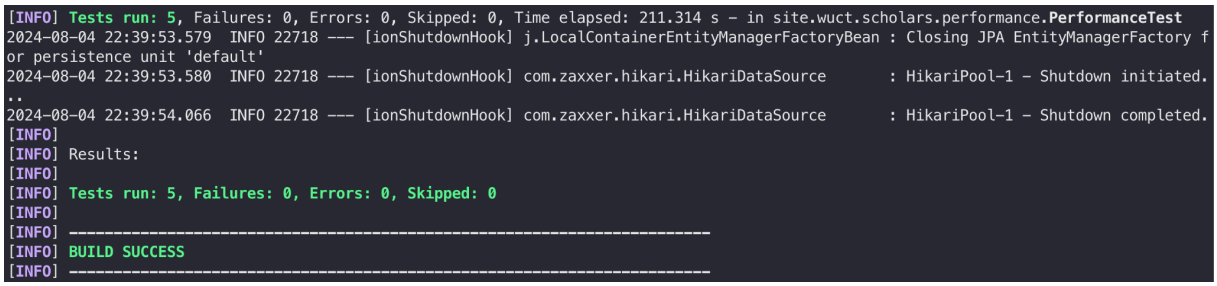

Figure 6: `PerformanceTest.java`

3. Conclusion

3.1. Course Learnt from this Project

What do you learn from this project (both interesting and uninteresting points)?

3.2. Relavant Database Knowledge

Have you found any relevant database knowledge you have learned in this course helpful and have you encountered any database relevant issues that have been discussed in this course?