[360 min] BUILD: Project 2 / Design & Implement a Document Database

Re-submit Assignment

Due Nov 23 by 10:59pm **Points** 100 **Submitting** a website url

Overview

In this practicum you will modify the database you built for Project 1 to adjust it to a document based database (Mongo). You can reuse the project description, adjusting the logical model and choosing the main collections to use. Finally you will modify the implementation from Project 1 to make it work with MongoDB and Node.

Format

- This work should be completed individually

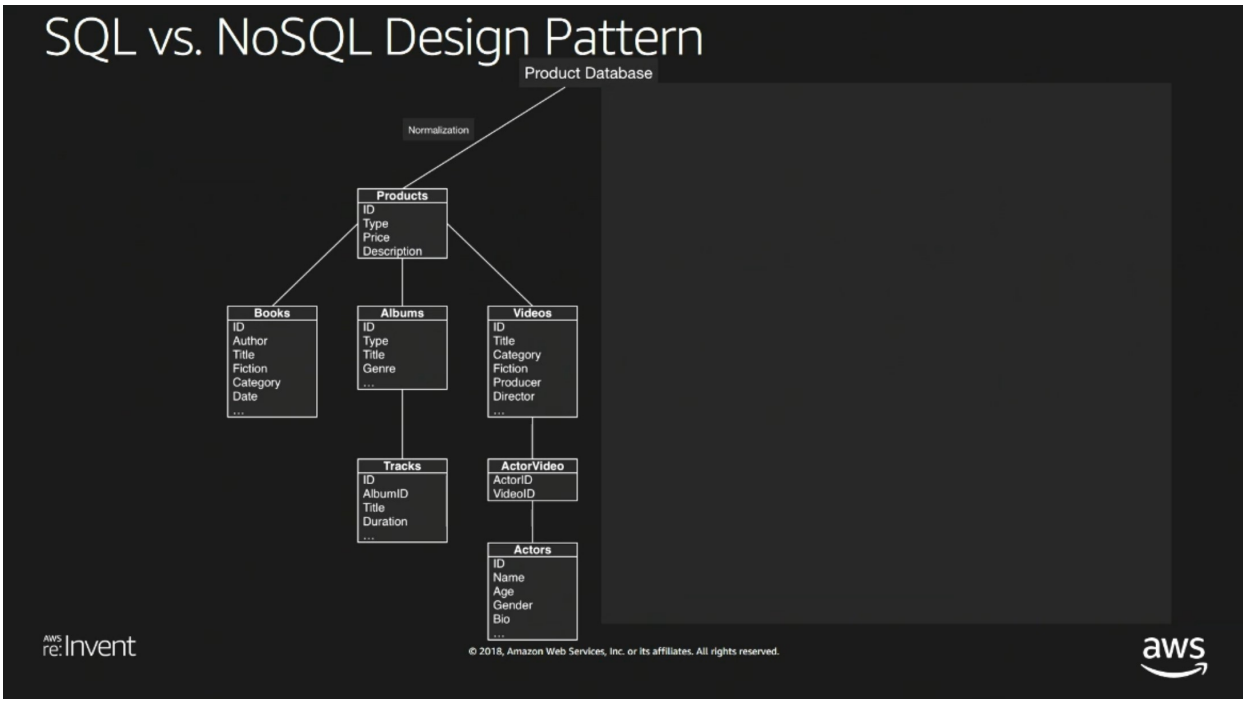
Tasks

1. (5 pts) Provide the problem requirements and the conceptual model in UML for your project. You can reuse the ones made in Project 1.
2. (*15 pts*) Adapt the Logical Data model from your Project 2 to have hierarchical tables. This is, main (root) tables from which all the other tables relate to. This main tables will become later your Mongo Collections. From your main tables you can have aggregation/composition, one to many and many to many relationships.

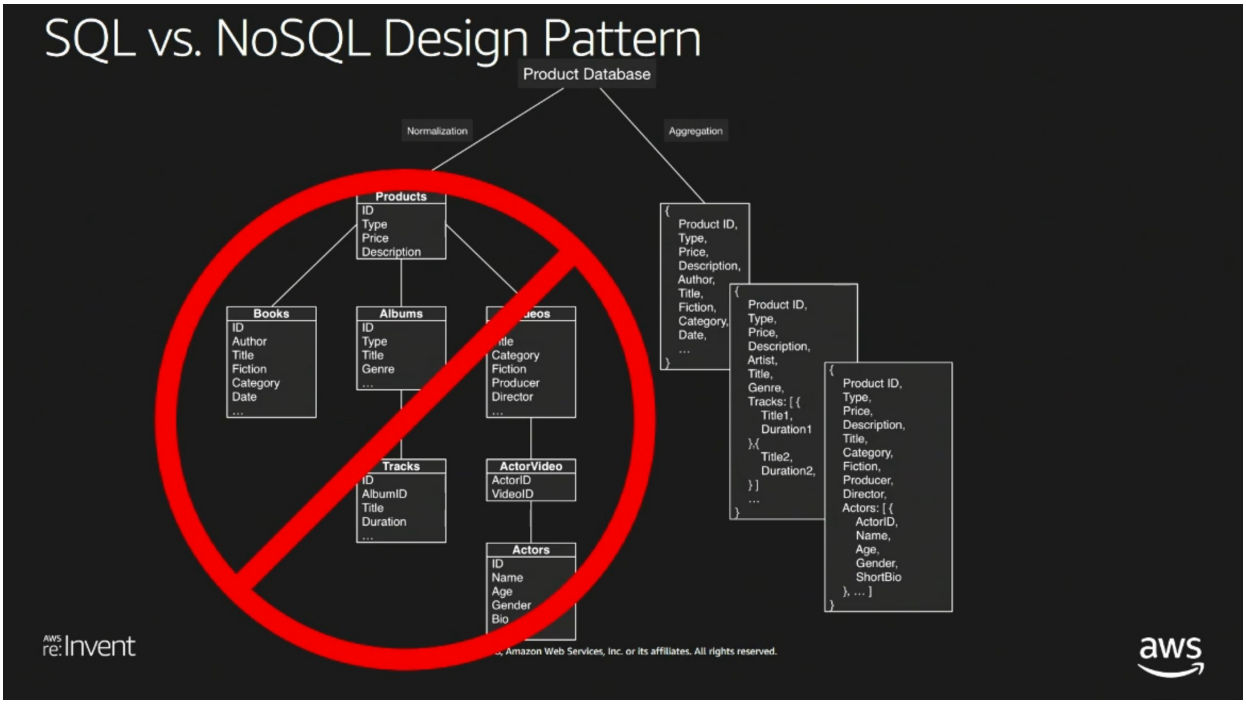
Here is an example from an Amazon presentation.

https://youtu.be/hwnNbLXN4vA?t=1851 **(https://youtu.be/hwnNbLXN4vA?t=1851)**

(https://youtu.be/hwnNbLXN4vA?t=1851)



3. (*10 pts*) From this logical model define the main Collections (Documents/Tables) you will be using in your Mongo Database. Provide a couple of JSON examples of these objects with comments when necessary. Think about a document that you will give to another database engineer that would take over your database. From the same presentation:



Also, check: https://docs.mongodb.com/guides/server/introduction/ [(https://docs.mongodb.com/guides/server/introduction/)](https://docs.mongodb.com/guides/server/introduction/)

and https://docs.mongodb.com/manual/core/data-modeling-introduction/ [(https://docs.mongodb.com/manual/core/data-modeling-introduction/)](https://docs.mongodb.com/manual/core/data-modeling-introduction/) I recommend you to use "Embedded data" with Mongo.

4. (*15 pts*) Populate the tables with test data. You can use tools such as https://www.mockaroo.com/schemas [(https://www.mockaroo.com/schemas)](https://www.mockaroo.com/schemas) or https://www.generatedata.com/ [(https://www.generatedata.com/)](https://www.generatedata.com/). You can export the sample data to CSV and then use mongoimport or Mongo Compass to populate your tables. Include in your repository a dump file that can be use to regenerate your database, and the instructions on how to initialize it

5. (*30 pts*) Define and execute at least five queries that show your database. At least one query must contain and aggregation https://docs.mongodb.com/manual/aggregation/ [(https://docs.mongodb.com/manual/aggregation/)](https://docs.mongodb.com/manual/aggregation/), one must contain a complex search criterion (more than one expression with logical connectors), one should be counting documents for an specific user, and one must be updating a document based on a query parameter (e.g. flipping on or off a boolean attribute for a document, such as enabling/disabling a song)

6. (25 *pts*) Create a basic Node + Express application that let's you create, display, modify and delete at least two of the tables. One of the tables can be the users table. No need to have a polished interface, and you can use the code created in class as a starting point, and/or the code you created for Project 1

Minimum Requirements & Grading Rubric

- Conceptual Data Model in UML (Class Diagram) Diagram must be drawn using a tool.

Submission

Submit a Github Repository containing:

A. The requirements document as a PDF.
B. UML Class Diagram as an embedded JPG/PNG.
C. ERD as an embedded JPG/PNG and URL to its LucidChart diagram.
D. Definition of Documents/Collections/Tables as example JSON objects.
E. Initialization files for the database containing the mockup data in CSV or Extended JSON format as well as instructions on how to initialize the database.

F. The code of your basic application with a proper README file

P1.Su20.Rubric (1)						
Criteria	**Ratings**					**Pts**
Q1 Provide the problem requirements and the conceptual model in UML for your project. You can reuse the ones made in Project 1.	**5.0 pts** **Only very minor issues**		**3.34 pts** **Very good**	**1.67 pts** **Adequate**	**0.0 pts** **No Marks**	5.0 pts
Q2 logical data model expressed as an ERD, with clear hierarchies defining your main documents. Relations can be aggregations/compositions, many to many relationships and one to many relationships	**15.0 pts** **Only very minor issues**	**12.0 pts** **Very good**	**7.5 pts** **Adequate**	**4.5 pts** **Acceptable**	**0.0 pts** **No Marks**	15.0 pts
Q3 Sample descriptions of the Documents in JSON, with examples of possible records annotated if necessary	**10.0 pts** **Excellent**	**6.67 pts** **Acceptable**	**3.33 pts** **Reasonable attempt but not adequate**		**0.0 pts** **No Marks**	10.0 pts
Q4 Test data. Initialization files and instructions on how to restore the database	**15.0 pts** **Well defined test data that tests all aspects of schema**	**12.0 pts** **Good test data but not sufficient to test all aspects of schema**	**7.5 pts** **Adequate**	**4.5 pts** **Some test data but not nearly sufficient**	**0.0 pts** **No Marks**	15.0 pts
Q5 Define and execute at least five queries that show your database. At least one query must contain and aggregation https://docs.mongodb.com/manual/aggregation/Links (https://docs.mongodb.com/manual/aggregation/Links) to an external site. (Links to an external site.) , one must contain a complex search criterion (more than one expression with logical connectors), one should be counting documents for an specific user, and one must be updating a document based on a query parameter (e.g. flipping on or off a boolean attribute for a document, such as enabling/disabling a song)	**30.0 pts** **Full Marks**	**24.0 pts** **Minor issues or a missing query**	**18.0 pts** **Missing some aspect**	**12.0 pts** **Missing some aspect**	**6.0 pts** **Missing some aspect**	**0.0 pts** **No Marks**	30.0 pts
Q7 Create a basic Node + Express application that let's you create, display, modify and delete at least two of the tables. One of the tables can be the users table. No need to have a polished interface, and you can use the code created in class as a starting point, and/or the code you created for Project 1	**25.0 pts** **Full Marks**	**20.0 pts** **Backend in Node works, but doesn't have front end**	**12.5 pts** **Adequate**	**7.5 pts** **Acceptable**	**0.0 pts** **No Marks**	25.0 pts
						Total Points: 100.0