Project Overview

Each team of 4-6 students will design and implement two different databases, each of which – in isolation – is sufficient to support the basic operations of a chain of stores that sell both packaged products and prepared food or drinks (think Starbucks, 7-Eleven, Jamba Juice, etc. but on a *much* smaller scale.) The project consists of four parts:

1. Define business rules, information needs, and a conceptual data model
2. Map your conceptual data model into an object-oriented model, using an Object-Relational Mapping (ORM) framework, ultimately targeting the MySQL relational DBMS.
3. Convert your conceptual data model into a logical data model targeting a document-oriented (JSON or XML) data store
4. Develop *two* queries for each of the information needs you identified in part 1: (a) SQL query and (b) query that retrieves the same data from your document-oriented store.

Team Member Roles

Each team member will assume one role from the options listed below, with no overlap on a single team. Based on this choice, individual team members will be responsible to ensure that the team's design adequately serves the needs of your chosen role. Since team sizes will vary, not all roles will be represented on all teams. Your chosen roles will determine the focus of your team's implementation.

- Customer
- Employee
- Location Manager / Regional Manager (combined)
- Supplier
- Owner / Board of Directors (combined)
- Regulator, Auditor, Tax Authority (combined)

Part 1 Deliverables

1. A list of 15-20 data-oriented business rules to be captured in your database. Each role on your team should be well-represented by these rules.
2. A conceptual data model, depicted using E-R or UML diagram(s). Include narrative notes where required for clarity.
3. 10-15 nontrivial information needs (queries expressed in English); 2-3 queries pertaining to each of the roles represented on your team.

Part 2 Deliverables

1. Code and any accompanying hand-crafted SQL DDL used to define your database using an existing Object-Relational Mapping (ORM) framework.

Sample Java Persistence API (JPA) code will be provided. However, you are welcome to base your implementation on any object-relational mapping framework, including:

- .NET Entity Framework

- NHibernate
- Python SQLAlchemy
- Python Django

You are not required to build a user interface of any sort.

Part 3 Deliverables

1. Description, using appropriate notation, of your document schema(s). For example, this may be expressed using an XML DTD, XML Schema, or JSON Schema [1]
2. Code used to define and populate your document-oriented database.

In-class discussion will focus on MySQL's support for JSON or XML column types. However, you are welcome to base your implementation on any document-oriented database. A few options, among many:

- MongoDB
- CouchDB
- RethinkDB
- PostgreSQL HSTORE/JSONB

Part 4 Deliverables

1. For each of the 10-15 information needs you identified in part 1, provide two queries:
   (a) SQL query, along with any indexes you defined for performance reasons
   (b) Query, along with any supporting code, expressed in the query language corresponding to the document-oriented data store selected by your team.

Final Deliverables (Team)

Submit via PolyLearn all deliverables for parts 1-4, as described above (one submission per team)

Final Deliverables (Individual)

1. One page summary, submitted via PolyLearn, that describes:
   (a) The role you chose (from the list above) along with a description of data modeling challenges unique to that role,
   (b) Your contributions to the project,
   (c) General ways in which you feel your team was very effective, as well as those areas where it could have been more effective (not specific to individual people, focused instead on overall team dynamics, planning, collaboration tools & techniques, etc.)

[1] https://json-schema.org/