COMP 3380 - Databases Concepts and Usage

Course Description

Calendar entry

An introduction to database systems including the relational, hierarchical, network and entity-relationship models with emphasis on the relational model and SQL. Prerequisite: one of COMP 2140 or COMP 2061.

General Course Description

Most applications require persistent data to be stored safely for long periods of time. These data stores must be fast, resilient, and the data must be portable. We'll discuss how to take apart the objects in a dataset and transform it into a dataset without redundancies. Then, we use tools such as Relational Algebra and SQL to put the dataset back together.

Relational Databases, or SQL databases as they are often called, have been the primary way of storing data for many years, though alternative databases have gained popularity recently. We will survey modern databases, such as graph databases, and document databases; modelling data to be stored appropriately on these databases and running queries on these databases. Storing and querying the same data in multiple different types of databases demonstrates the usefulness, and drawbacks, of each database implementation.

Detailed Prerequisites

Before entering this course, a student should be able to:

- Compute common set functions such as set intersection and set difference.
- Compare search algorithms in terms of efficiency.
- Create Object-Oriented data models based on a provided problem.
- Understand relationships between an object and the data and objects that it is composed of.

Course Goals

By the end of this course students will:

- Model data using Entity-Relationship Diagrams and create databases based upon these models.
- Create a database to store structured data with no duplication of data by applying normal forms.

- Write and perform queries on data using SQL and be able to model the queries by using Relational Algebra.
- Connect to a database using software to execute queries and modify the program's behaviour based upon the results of the query.
- Execute queries to add and extract data on graph and document databases.
- Contrast SQL databases with alternatives such as graph databases and document databases.

Learning Outcomes

Data modelling

Students should be able to:

1. Determine and define functional dependencies (FDs) within a dataset and understand how that will affect the resulting data models.
2. Model a data domain using Entity-Relationship modelling (ER Diagrams).
3. Model a domain that has inheritance relationships using Enhanced entity–relationship modelling.
4. Use functional dependencies and Armstrong's Axioms to prove or disprove statements about a dataset.
5. Create a model of data from a non-normalized dataset, recognizing functional dependencies in the data from the data and context itself.

Creating relational databases from models

Students should be able to:

1. Create Relational Database model from an ER or EER diagram.
2. Apply normalization techniques (such as 1NF, 2NF, 3NF, BCNF) to a Relational Model to reduce or eliminate redundancy in data.
3. Create a Relational Database using Data Definition Language (DDL) to create tables, alter and remove existing tables.
4. Add constraints to a database to maintain consistency in the model, this includes table and column constraints such as uniqueness and relationship constraints between tables.
5. Add deletion techniques to Relational Database tables to allow deletion of records that maintains the consistency of the database; Cascading deletion, setting to default values, or refusing to delete.

Querying relational databases

Students should be able to:

1. Use SQL to query a Relational Database, joining relations in the table to form queries to extract meaningful results.
2. Create queries with join clauses to join multiple tables into one resulting relation, handling for cases where there may or may not be data to join with between the two tables.
3. Use aggregation clauses to perform operations on the resulting set of data.
4. Modify data in relations with Data Manipulation Language (DML), using DML to add, modify, and delete data.
5. Use transactions to ensure a multiple-insert operation is performed atomically.
6. Use a programming language to interact with a database; adding, removing, updating, and finding data based on user interactions.
7. Use tools to show how a given query is interpreted and run by the database engine.
8. Use subqueries to extract data and create more complex queries. Recognize the drawbacks, limitations, and performance impact of different types of subqueries.
9. Execute SQL injection attacks on insecurely created database queries and use tools such as prepared statements to prevent SQL injection attacks.

Modelling and Querying with Alternative databases

Students should be able to:

1. Identify relationships between objects in a dataset and build a graph database from those objects and relationships; create queries on this dataset to extract information about how objects are related.
2. Build a data model using a document database (such as MongoDB) and query the dataset.
3. Model a problem domain and load the model into a graph database (such as Neo4J); run queries on the dataset to extract results that would be difficult to extract using a Relational Database.
4. Explain the execution model of graph database queries and how this differs from relational database execution models and how we can extract different results because of these differences.
5. Compare and contrast a Relational Database with a Document or Graph Database.