COMP 421 Study guide

Francis Piché

January 12, 2019

Contents

Part I
Preliminaries

1 Disclaimer

These notes are curated from Joseph D'silva COMP421 lectures at McGill University. They are for study purposes only. They are not to be used for monetary gain.

Part II
Introduction to Databases

2 Data Storage

In operating systems, file systems are used for persistent storage. This is insufficient because:

- Only provides a basic API
- Information may not be structured.
- Only linear seek possible (slow)
- May have data loss in case of power outages etc
- Can't have concurrent access to files

3 Entity-Relationship Model

4 Relational Model and Data Definition Language

Entity Relationship Model is a language that describes the data determined through requirement analysis. This is very similar to UML, but is not the same.

4.1 Requirement Analysis

When designing a database, we must first identify which data needs to be stored, and how it will be used. (Which operations need to be executed on the data).

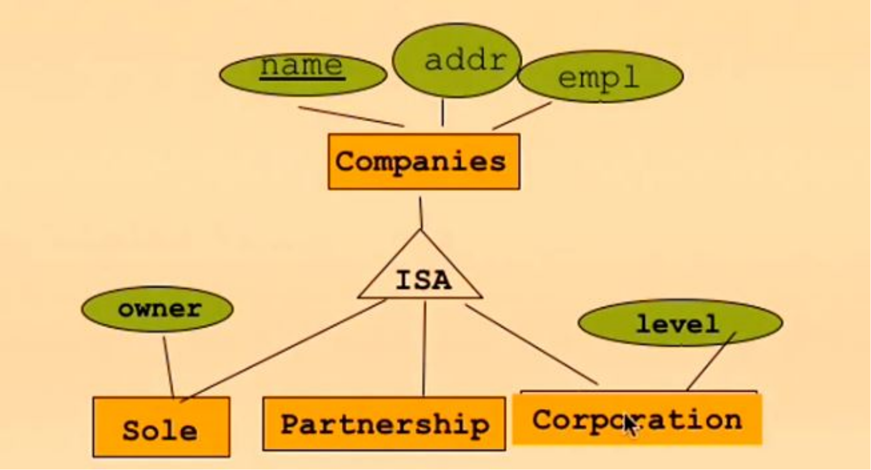
For example, if we were designing a database for Minerva at McGill, we would need to think about which entities are relevant to be stored. In OOP, these would be the classes. Things like the students, addresses, fees, courses etc are all relevant information. We would also need to think about the types of operations that must be possible by our database. Things like adding students, changing addresses, assessing fees etc.

In an OOP setting, we would break down the problem by identifying classes and relationships between them. For example, we might have Student and Instructor classes, with a parent class Person, since the Student and Instructor share some attributes (name, age...). We also have relationships between classes such as a Student and Transcript. A Transcript does not exist unless assigned to a Student. We need ways to model all of these relationships in a database setting rather than an OOP setting.

4.2 ER

An *entity* is a real world object which contains a set of attributes. A collection of instantiated entities would be an *entity set*. An entity set must have a *key*. This is an attribute that is underlined, and MUST be unique. There can be two attributes underlined, in which case both COMBINED must be unique. Individually they need not be unique.

ISA ("is a") hierarchy is similar to subclasses in OOP. This is represented as a triangle. The key is still in the parent entity set.



There is a subtle point here in that unlike OOP, the attributes of the parent are not part of the child entity set (to avoid duplication, and keep things consistent).

If there are overlaps (a company can be both a partnership and Corporation), we must put a note explaining the situation. We assume that the parent is not covering all possible subtypes. (There may exist a company which is not a Sole, Partnership or Corporation).

4.3 Relationships

A relationship is an association among two or more entities. This can be one to one, many-to-one, many-to-many. A relationship set is a collection of similar relationships.

There cannot be duplicate association instances. For example, if a Company donates 10$ to a Party, (the association being the sponsorship), then there cannot be another 10$ donation with the same Company and Party. The original must be updated.

We can add arrows to constrain to a one-to-many or one-to-one.

5 Relational Algebra

Part III
SQL

6 Simple Queries

7 Advanced Queries

8 Constraints

9 Application Programming

Part IV
DB Internals

10 Buffer Management

11 Indexing

12 Query Execution

13 Query Optimization

14 KV Stores

15 Map Reduce

16 Transactions

17 Concurrency Control

18 New Trendy Stuff