CS 240: Relational Databases Overview Transcript

[00:00:00] **JEROD WILKERSON:** In the next several videos, you will learn about relational databases. You'll learn quite a few different things. We'll start with what the relational model is.

[00:00:10] We'll learn what relational databases are and how they work. We will also learn about a language for accessing data and putting data in databases that's called SQL or Structured Query Language.

[00:00:23] You'll learn about that. You'll learn how to execute SQL from Java programs, that's JDBC, and then you'll also learn about a specific database called SQLite, which is the one that we'll use in this class.

[00:00:39] You'll learn quite a few different things about databases over the next several lecture videos. One of the things that you need to know is about Database Management Systems.

[00:00:48] Databases are implemented by software called Database Management Systems. These are systems that provide the services that we need from our database, and mostly we use relational databases.

[00:01:03] There are new database technologies now that are called NoSQL databases. They are not based on the relational model and so we might think that because they're newer, they must be better.

[00:01:15] That's really not the case. Relational databases have been around since the '70s, and they're still the most commonly used type of database because they're the most versatile.

[00:01:23] Some of the newer database technologies, the NoSQL technologies, are really good for very specific purposes. For example, they can be really fast for accessing certain kinds of data, but they're not as flexible as relational databases.

[00:01:36] Relational databases are still kind of the standard databases that we use, and they're the appropriate choice for most programming applications of databases.

[00:01:48] Databases store data in files in ways that scale to really large amounts of data and can be accessed very efficiently in ways that would be very difficult for you to try to replicate by just accessing files on your own.

Start slide description.

The following text appears on the screen.

Database Management Systems (DBMS)

- Databases are implemented by software systems called Database Management Systems (DBMS)
- Commonly used Relational DBMS's include Oracle, MySQL, PostgreSQL, and MS SQL Server
- DBMS's store data in files in a way that scales to large amounts of data and allows data to be accessed efficiently

End slide description.

[00:02:03] There are two primary ways that we interact with the database: one is interactive and the other is programmatic.

[00:02:15] We need to be able to access a database using some kind of a GUI tool, so we can create databases, we can create our tables, and we can look at things, browse the database.

[00:02:26] That becomes really useful when we're working with databases. We also need to be able to access them from our Java programs in this class.

[00:02:35] You're going to learn how to do both, and you'll get pretty proficient with accessing databases in both ways.

[00:02:42] Databases are used to store large amounts of data that we need to work with in our applications.

Start slide description.

The following text appears on the screen.

Programmatic vs. Interactive Database Access

Programs can access a database through APIs such as JDBC or ADO.NET.

End users can access a database through an interactive management application that allows them to query and modify the database

An animated illustration of the architecture of a database system (Program, DB API, DB Driver, DB (Database), Management Console) is displayed on the screen.

End slide description.

[00:02:49] There are two primary ways that we can use databases. One is what is an embedded database. What we mean by that is, for some applications, these are usually smaller or simpler applications.

[00:03:03] You might have the database just embedded right with the code, and it's part of the code, and so it wouldn't be able to be accessible from any other application. That's an embedded use of a database.

[00:03:16] The other way is client or server, where you have a database that has a server sitting in front of it, or maybe it is a server, and that makes it available to multiple programs over a network.

[00:03:26] That's actually what we'll be doing in this class. You'll create a database and make it available through a database server.

Start slide description.

An image illustration depicting the difference between embedded and client/server database architectures are displayed on the screen.

End slide description.

[00:03:35] In the relational data model, data is stored in tables consisting of columns and rows. We could think of tables as being like classes. Tables are the structure that we use to be able to hold the data just like classes are the structure that we use to hold our code.

[00:03:52] At runtime in object-oriented programming, we're more interested in objects. The same is true with databases. Tables are what store a database, the data we're interested in is the rows of data that we're most interested in.

[00:04:04] Each row has a primary key, which is a unique identifier for that row, and then we're able to represent relationships between rows by relating primary keys across different tables.

[00:04:17] If we take all of the structure of a database—so the tables, things like primary keys and other things that you'll learn about, all of that combined— we often refer to it as a schema, which is really just the structure of our database.

Start slide description.

The following text appears on the screen.

Relational Databases

- Relational databases use the relational data model you learned about in CS 236
    - Relations = Tables
    - Tuples = Rows

- In the relational data model, data is stored in tables consisting of columns and rows.
    - Tables are like classes
    - Each row in a table stores the data you may think of as belonging to an object.
    - Columns in a row store the object's attributes (instance variables).
- Each row has a "Primary Key" which is a unique identifier for that row. Relationships between rows in different tables are represented using keys.
    - The primary key in one table matches a foreign key in another table
- Taken together, all the table definitions in a database make up the "schema" for the database.

End slide description.

[00:04:32] Here's just a little visual comparison of concepts that you should be familiar with. So, classes in the object model are similar to tables in a relational model. Objects in the object model are similar to rows in the relational model or in relational databases.

[00:04:51] In object-oriented programming, we represent relationships with references. In the relational model with relational databases, we represent relationships by relating primary keys to foreign keys, which you'll learn a lot about.

Start slide description.

The following text and table appears on the screen.

Comparison of Object and Relational Models

Object Model	**Relational Data Model**
• Class	• Table

• Object	• Row
• Relationship (reference)	• Relationship (primary and foreign key)

End slide description.

Start slide description.

End citations are listed.

End slide description.