Relational databases

Lecture 10

Based on slides by R. Ramakrishnan and J. Gehrke

CS 638 Web Programming

Lecture overview

- Introduction to databases
- Representing entities in a relational database
- Representing relationships between entities

Web programming & databases

- Three-tiered architecture for web applications
  - Presentation tier (HTML, CSS, client side programs)
  - Application or "Business logic" tier (server side programs)
  - Data tier (relational database with persistent data)
- Some advantages of databases
  - Simple and powerful data manipulation language
  - High throughput processing of large data sets
  - Concurrency (handle multiple user requests at same time)
  - Data consistency (administrator sets *integrity constraints*)
  - Reliability (graceful recovery from all types of failures – hardware problems, crashes of various computers, etc.)

Database topics

- Topics covered in this class
  - Basic concepts for representing information in databases
  - Common, simple database queries (reading data)
  - Simple updates to database (writing data)
- Database topics covered in CS564
  - Designing the structure of data in a database (schema) to correctly reflect the information you want to capture
  - Complete understanding of query language (SQL)
  - Database performance
  - Writing database software

What is a database?

- Database software running on a server
  - *Relational* databases by far most common today
  - Various vendors: IBM, Oracle, Microsoft, etc.
  - Can interact with both web applications and desktop applications
- The structure of the data in the database (schema)
  - The DBA (database administrator) designs it
- The actual data in the database (instance)
  - Read and written by application developer (you)
- The word "database" is used informally for all 3

Overview of database design

- Conceptual design using ER model
  - What are the *entities* and *relationships* to represent?
  - What information about these entities and relationships should we store in the database?
  - What are the *integrity constraints* or business rules?
  - A database "schema" in the ER model can be represented pictorially (ER diagrams).
  - ER design is subjective – there are often many ways to model a given scenario!
- Can map an ER diagram to schema of relational database.

Overview of relational databases

- Relational database: a collection of relations (tables)
  - Tables are used for storing both entities and relationships
- Relation (table) consist of two parts
  - *Instance*: a table, with rows and columns
  - *Schema*: specifies name of relation, plus name and type of each column (e.g. Students(sid: string, name: string, login: string, age: integer, gpa: real))
  - Can think of schema as header of the table and of the instance as the body
- Each instance of a relation (table) is a set of rows
  - All rows (also called tuples) are distinct

Overview of query languages

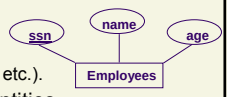
- Allow manipulation & retrieval of data from database
- Relational model supports simple, powerful QLs:
  - Theoretical foundation: relational calculus
    - Strong formal foundation based on logic
  - Users describe what they want, rather than how to compute it (non-operational, declarative)
    - Allows for much optimization (transparent to user)
- Query Languages != programming languages!
  - QLs not expected to be "Turing complete"
  - QLs not intended to be used for complex calculations
  - QLs support easy, efficient access to large data sets

Lecture overview

- Introduction to databases
- Representing entities in a relational database
- Representing relationships between entities

Entities in the ER model

- *Entity*: Real-world object distinguishable from other objects
  - Described using a set of attributes.
  - Each attribute has a domain (e.g. 8-byte string , number from 0 to 99, etc.).



- *Entity Set*: Collection of similar entities
- All entities in an entity set have the same attributes.
- Each entity set has a key (similar to hash table keys)
  - The key is a subset of the attributes
  - The value of these attributes uniquely identifies entities within the set

Representing entities in a relational database

- A table (relation) corresponds to each entity-set
  - Entities will be rows (tuples) in the table

Employees		
ssn	name	age
111223333	George	58
111334444	Sarah	25
222334444	Tom	38

Basic operations from relational algebra

- Some basic operations
  - *Selection* (σ) selects a subset of rows
    - Ex. 1: the employees younger than 40
  - *Projection* (π) omits unwanted columns
    - Example 2: the names of all employees
- Can combine them
  - Ex. 3: names of employees younger than 40

ssn	name	age
111334444	Sarah	25
222334444	Tom	38

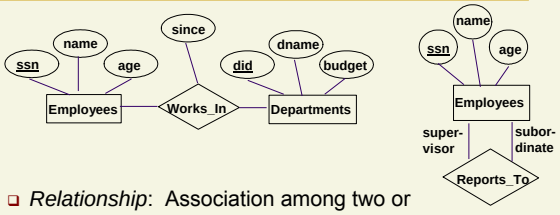
name
George
Sarah
Tom

name
Sarah
Tom

Lecture overview

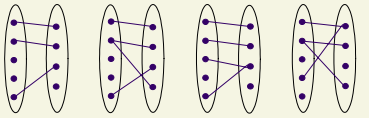
- Introduction to databases
- Representing entities in a relational database
- Representing relationships between entities

Relationships in the ER model



- *Relationship*: Association among two or more entities. (e.g. Tom works in the human resources department)
- *Relationship set*: Collection of similar relationships.

Types of relationships

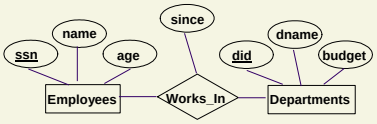


- Relationships between two entity sets can have one or more entities participate
  - Property of relationship set
- Multiple entity sets can participate in a relationship set – n-ary relations have n entity sets participate
  - An entity set may participate many times w/ different roles

Representing relationships in a relational database

- A table corresponds to each relationship-set
  - This table refers to the entities in the relationship using foreign keys (keys from tables representing the entity sets)
  - This table has one or more separate attributes for the key of each entity set participating
  - Attributes associated with relation also turned into fields

Works_in		
ssn	did	since
…	…	…



Useful operation from relational algebra

- The *join* operation is the most common way to combine information from two or more tables
  - There are many types of joins, we will focus on equijoins
- Think about joins as substituting the foreign key in the table representing the relationship with all the attributes of the entity it points to
  - Example: instead of ssn in Works_in we add ssn, name and age of employee
- Can combine with other operations (selection, projection) and can have multiple joins in one query
- Examples in next lecture which will be on SQL