Data Management

An evolving, expanding field:

• Classical stand-alone databases (Oracle, DB2, SQL Server)
• Computer science is becoming data-centric:
  - web knowledge harvesting, crowdsourcing, cloud computing, scientific databases, networks, data mining, streaming sensor monitoring, social networks, bioinformatics, geographic information systems, digital libraries, data-driven business processes, data analytics

• Classical database concepts and algorithms continue to provide the core technology → this course
CSE132A: Database Systems Principles

• Core concepts and techniques in database systems
• Databases from the viewpoint of user and designer
• A lot of SQL, but also peeks under the hood:
  query processing, schema design, transactions
  and concurrency control
• Some basic theory: formal languages underlying SQL
  relational algebra and relational calculus
• Basic background for follow-up courses
  132B: Database system applications (A. Deutsch)
  135: Online analytics applications (Y. Papakonstantinou)
  190: Beyond relational databases (A. Deutsch)
  190: Database implementation (A. Kumar)

Resources

• Slides, recommended texts, podcast
• Practice problem sets with solutions (ungraded)
  Gradiance online practice homeworks and labs
  Posted practice problems
• Weekly discussion section
• TA/tutor daily office hours and instructor office hours
• Discussion board (Piazza)
• Everything will be posted on the class website (check often!)
  http://cseweb.ucsd.edu/classes/wi20/cse132A-a/
Requirements

- Two Gradiance SQL Labs and two written homeworks (16%)
- Two programming assignments (SQL and JDBC) (34%)
- Midterm (25%)
- Final (25%)
- Class participation via clickers strongly recommended

Academic Integrity

Everyone taking the class is assumed familiar with the Integrity of Scholarship policy posted on the class Web site

What is a database?

- Persistent data
- Query and update language for accessing and modifying data
- Query optimization
- Transactions and concurrency control

What kind of data?

Emphasis: many instances of similarly structured data

Examples:

- Airline reservations: database (large set of similar records)
- Computerized library: information retrieval
- Medication advisor: expert system
Top Level Goals of a Database System

- Provide users with a meaning-based view of data
  - shield from irrelevant detail → abstract view

- Support operations on data
  - queries, updates

- Provide data control
  - integrity, security
  - concurrency, recovery

Levels of Abstraction

- **Logical level:** describes data stored in database in terms close to the application
  ```
  type customer = record
  customer_id : string;
  customer_name : string;
  customer_street : string;
  customer_city : integer;
  end;
  ```

- **Physical level:** describes how the data is stored.

- **View level:** customized, restructured information. Views can also hide information (such as an employee’s salary) for security purposes.
Basic Architecture of a Database System

Data Independence – logical and physical levels are independent

Database System

- Tailored to specific application

Database Management System

- Generalized DB system
  - used in variety of application environments
  - common approach to
    - data organization
    - data storage
    - data access
    - data control
  - e.g. Ingres/Postgres, DB2, Oracle, SQL Server, MySQL, etc.
Data Models

- A collection of concepts and tools for describing the data relationships, semantics, constraints…
- A language for querying and modifying the data

- Relational model
- Entity-Relationship data model (mainly for database design, no query language)
- Object-based data models (Object-oriented and Object-relational)
- Semi-structured data model (graphs, XML)
- Other older models:
  - Network model
  - Hierarchical model

Schemas and Instances

Similar to types and values of variables in programming languages

- **Schema** – the logical structure of the database
  - Example: The database consists of information about a set of customers and accounts and the relationship between them
  - Analogous to type of a variable in a program

- **Instance** – the actual content of the database at a particular point in time
  - Analogous to the value of a variable
Example: Entity-Relationship Model

- Models an application as a collection of *entities* and *relationships*
  - Entity: a “thing” or “object” in the that is distinguishable from other objects
    - Described by a set of *attributes*
  - Relationship: an association among several entities
- Represented diagrammatically by an *entity-relationship diagram*:

![Entity-relationship diagram](image)

Example: Relational Model

- *Schema*
  - (a) The *customer* table
  - (b) The *account* table
  - (c) The *depositor* table
Example: Relational Model

<table>
<thead>
<tr>
<th>customer_id</th>
<th>customer_name</th>
<th>customer_street</th>
<th>customer_city</th>
</tr>
</thead>
<tbody>
<tr>
<td>192-83-9469</td>
<td>Johnson</td>
<td>12 Alima St.</td>
<td>Palo Alto</td>
</tr>
<tr>
<td>477-85-0011</td>
<td>Hayes</td>
<td>3 Main St.</td>
<td>Harrison</td>
</tr>
<tr>
<td>182-71-4991</td>
<td>Turner</td>
<td>123 Putnam Ave.</td>
<td>Randolph</td>
</tr>
<tr>
<td>321-12-3232</td>
<td>Jones</td>
<td>100 Main St.</td>
<td>Pittsfield</td>
</tr>
<tr>
<td>734-66-9999</td>
<td>Lindsay</td>
<td>175 Park Ave.</td>
<td></td>
</tr>
<tr>
<td>019-28-3746</td>
<td>Smith</td>
<td>72 North St.</td>
<td></td>
</tr>
</tbody>
</table>

### Instance

<table>
<thead>
<tr>
<th>account_number</th>
<th>balance</th>
</tr>
</thead>
<tbody>
<tr>
<td>A-101</td>
<td>500</td>
</tr>
<tr>
<td>A-215</td>
<td>700</td>
</tr>
<tr>
<td>A-352</td>
<td>400</td>
</tr>
<tr>
<td>A-205</td>
<td>350</td>
</tr>
<tr>
<td>A-281</td>
<td>900</td>
</tr>
<tr>
<td>A-217</td>
<td>750</td>
</tr>
<tr>
<td>A-222</td>
<td>700</td>
</tr>
</tbody>
</table>

(a) The customer table

<table>
<thead>
<tr>
<th>account_number</th>
<th>account_number</th>
</tr>
</thead>
<tbody>
<tr>
<td>192-83-9469</td>
<td>A-101</td>
</tr>
<tr>
<td>477-85-0011</td>
<td>A-215</td>
</tr>
<tr>
<td>182-71-4991</td>
<td>A-352</td>
</tr>
<tr>
<td>321-12-3232</td>
<td>A-281</td>
</tr>
<tr>
<td>734-66-9999</td>
<td>A-217</td>
</tr>
<tr>
<td>019-28-3746</td>
<td>A-222</td>
</tr>
</tbody>
</table>

(b) The account table

Data Definition Language (DDL)

- Specification language for defining the database schema
  
  Example in SQL:

  ```sql
  create table account (
      account-number char(10),
      balance integer
  )
  ```

- DDL compiler generates a set of tables described in a *data dictionary*
- Data dictionary contains metadata (i.e., data about data)
  - Database schema
  - Integrity constraints
  - Authorization information
Data Manipulation Language (DML)

- Language for accessing and modifying data
  DML also known as query/update language
- Two classes of languages
  - Procedural – user specifies what data is required and how to get that data
  - Declarative (nonprocedural) – user specifies what data is required without specifying how to get it
- SQL is the most widely used query language
  primarily declarative

This course: core database issues

- The relational model
- Commercial query languages: SQL
- Formal query languages: relational algebra and calculus
- Query processing
- Schema design: normal forms and the ER model
- Concurrency control
- Other topics as time allows
Databases at UCSD

- Prof. Alin Deutsch
- Prof. Arun Kumar
- Prof. Yannis Papakonstantinou
- Prof. Victor Vianu

Database group Web site: http://db.ucsd.edu/
papers, seminars, bragging….

- Intersections with other CSE groups
  - storage
  - multimedia
  - machine learning
  - networks