First, a story…

UW’s Databases

The Seattle Times

New UW payroll system behind schedule, more costly than expected

Outline

1. Administrivia
2. The Relational Data Model
3. Databases, SQL, and RA

What am I going to learn?

Course Topics
- Queries
- Database Design
- Optimization
- Transactions and Parallelism
- Wide-Column Databases (new topic)
- Semi-Structured Document Databases

Tools:
- Experimental to Enterprise Platforms
- Cloud Services (AWS, Azure, GCP)

What am I going to learn?

- After the course, you will be able to…
  - Explain how a query is processed end-to-end
  - Integrate a database into an application
  - Effectively manage data for long-term use
  - Create database constructs to provide speedups
  - Make design choices when selecting tools for a project

344 Staff

- Instructor: Ryan Maas
  - Office hours Tuesday 12pm in CSE 214 and by appointment
- TAs
  - Shana Hutchison
  - Zhanhao Zhang
  - Li Du
Course Format

- Lectures: this room, please attend!
- Sections: for locations, see web; bring your laptop
- 7 homework assignments
- Midterm and final (in-class)
- Class and section participation: Post and answer questions (in class, piazza, etc)

Exams

Midterm (TBD) and Final (August 23)
- You may bring letter-size piece of paper with notes
  - Handwritten
  - May write on both sides
  - Midterm: 1 sheet, Final: 2 sheets
- Closed book. No computers, phones, watches,....
  - Location: in class

References

Main textbook, available at the bookstore or pdf:
  - Second edition.
- Also useful:
  - Database Management Systems (3rd Edition)

Communications

- Web page: www.cs.washington.edu/344
  - Everything is here
- Piazza piazza.com/class/jx9rlkgbdr27k0
  - THE place to ask course-related questions
  - Log in today, enable notifications
  - Warning: canvas.ucdavis.edu/courses/192458/pages/piazza-warning
- Class mailing list
  - Very low traffic, only important announcements

Administrivia

- Grading:
  - 40% HW, 20% Midterm, 30% Final
  - 10% to be adjusted between assignments
  - 4 late days, 2 days max per assignment in 24 hour chunks
- Collaboration:
  - HWs must be done and typed up individually, though you can consult with other students about your approach.
  - We will run cheating detection
Let’s get started!

What is a database?

- A collection of files storing related data

Give examples of databases:
- Accounts database;
- Payroll database;
- UW’s students database;
- Amazon’s products database;
- Airline reservation database;

Database Management System (DBMS)

What is a DBMS?
What is a DBMS?
- A big program written by someone else that allows us to manage efficiently a large database and allows it to persist over long periods of time.

Give examples of DBMSs
- Oracle, IBM DB2, Microsoft SQL Server, Vertica, Teradata
- Open source: MySQL (Sun/Oracle), PostgreSQL, CouchDB
- Open source library: SQLite

We will focus on relational DBMSs most quarter.

Think About This
How do we describe information?

Data Model
A Data Model is a mathematical formalism to describe data. It is how we can talk about data conceptually without having to think about implementation.
3 Parts of a Data Model

The 3 parts of any data model

- Instance
  - The actual data

- Schema
  - A description of what data is being stored

- Query Language
  - How to retrieve and manipulate data

Data Model Zoo

There are lots of models out there!

- Relational
- Semi-structured
- Key-value pairs
- Graph
- Object-oriented

Multiple Representation

Same data can be represented in different ways

An example of Facebook friends

<table>
<thead>
<tr>
<th>Person 1</th>
<th>Person 2</th>
<th>Friend</th>
</tr>
</thead>
<tbody>
<tr>
<td>Joe</td>
<td>Ann</td>
<td>1</td>
</tr>
<tr>
<td>Ann</td>
<td>Bob</td>
<td>1</td>
</tr>
<tr>
<td>Bob</td>
<td>Joe</td>
<td>0</td>
</tr>
</tbody>
</table>

Data Model Zoo

There are lots of models out there!

- Relational
- Semi-structured
- Key-value pairs
- Graph
- Object-oriented

The Relational Model

Again, how do we describe information?

Most common answer: The Relational Model

https://db-engines.com/en/ranking

June 24, 2019
Components of the Relational Model

Table/Relation

**UserID** | **Name** | **Job** | **Salary**
---|---|---|---
123 | Jack | TA | 50000*
345 | Allison | TA | 60000*
567 | Magda | Prof | 90000
789 | Dan | Prof | 100000

* I wish

Characteristics of the Relational Model

- **Set semantics**
- No duplicate tuples
- Attributes are **typed** and **static**
  - INTEGER, FLOAT, VARCHAR(n), DATETIME, ...
- Tables are **flat**

Columns/Attributes/Fields

**UserID** | **Name** | **Job** | **Salary**
---|---|---|---
123 | Jack | TA | 50000
345 | Allison | TA | 60000
567 | Magda | Prof | 90000
789 | Dan | Prof | 100000

Rows/Tuples/Records

**UserID** | **Name** | **Job** | **Salary**
---|---|---|---
123 | Jack | TA | 50000
345 | Allison | TA | 60000
567 | Magda | Prof | 90000
789 | Dan | Prof | 100000

Rows/Tuples/Records

**UserID** | **Name** | **Job** | **Salary**
---|---|---|---
123 | Jack | TA | 50000
345 | Allison | TA | 60000
567 | Magda | Prof | 90000
789 | Dan | Prof | 100000

Rows/Tuples/Records

**UserID** | **Name** | **Job** | **Salary**
---|---|---|---
123 | Jack | TA | 50000
345 | Allison | TA | 60000
567 | Magda | Prof | 90000
789 | Dan | Prof | 100000

* I wish
Characteristics of the Relational Model

**Set semantics**
- No duplicate tuples
- Attributes are **typed** and **static**
  - INTEGER, FLOAT, VARCHAR(n), DATETIME, ...
- Tables are **flat**

<table>
<thead>
<tr>
<th>UserID</th>
<th>Name</th>
<th>Job</th>
<th>Salary</th>
</tr>
</thead>
<tbody>
<tr>
<td>123</td>
<td>Jack</td>
<td>TA</td>
<td>50000</td>
</tr>
<tr>
<td>345</td>
<td>Allison</td>
<td>TA</td>
<td>60000</td>
</tr>
<tr>
<td>567</td>
<td>Magda</td>
<td>Prof</td>
<td>90000</td>
</tr>
<tr>
<td>789</td>
<td>Dan</td>
<td>Prof</td>
<td>100000</td>
</tr>
</tbody>
</table>

Characteristics of the Relational Model

**Set semantics**
- No duplicate tuples
- Attributes are **typed** and **static**
  - INTEGER, FLOAT, VARCHAR(n), DATETIME, ...
- Tables are **flat**

<table>
<thead>
<tr>
<th>UserID</th>
<th>Name</th>
<th>Job</th>
<th>Salary</th>
</tr>
</thead>
<tbody>
<tr>
<td>123</td>
<td>Jack</td>
<td>TA</td>
<td>50000</td>
</tr>
<tr>
<td>345</td>
<td>Allison</td>
<td>TA</td>
<td>60000</td>
</tr>
<tr>
<td>567</td>
<td>Magda</td>
<td>Prof</td>
<td>90000</td>
</tr>
<tr>
<td>789</td>
<td>Dan</td>
<td>Prof</td>
<td>100000</td>
</tr>
</tbody>
</table>

Characteristics of the Relational Model

**Set semantics**
- No duplicate tuples
- Attributes are **typed** and **static**
  - INTEGER, FLOAT, VARCHAR(n), DATETIME, ...
- Tables are **flat**

<table>
<thead>
<tr>
<th>UserID</th>
<th>Name</th>
<th>Job</th>
<th>Salary</th>
</tr>
</thead>
<tbody>
<tr>
<td>123</td>
<td>Jack</td>
<td>banana</td>
<td>50000</td>
</tr>
<tr>
<td>345</td>
<td>Allison</td>
<td>TA</td>
<td>60000</td>
</tr>
<tr>
<td>567</td>
<td>Magda</td>
<td>Prof</td>
<td>90000</td>
</tr>
<tr>
<td>789</td>
<td>Dan</td>
<td>Prof</td>
<td>100000</td>
</tr>
</tbody>
</table>

Characteristics of the Relational Model

**Set semantics**
- No duplicate tuples
- Attributes are **typed** and **static**
  - INTEGER, FLOAT, VARCHAR(n), DATETIME, ...
- Tables are **flat**

<table>
<thead>
<tr>
<th>UserID</th>
<th>Name</th>
<th>Job</th>
<th>Salary</th>
</tr>
</thead>
<tbody>
<tr>
<td>123</td>
<td>Jack</td>
<td>JobName</td>
<td>50000</td>
</tr>
<tr>
<td>345</td>
<td>Allison</td>
<td>TA</td>
<td>60000</td>
</tr>
<tr>
<td>567</td>
<td>Magda</td>
<td>Prof</td>
<td>90000</td>
</tr>
<tr>
<td>789</td>
<td>Dan</td>
<td>Prof</td>
<td>100000</td>
</tr>
</tbody>
</table>
Characteristics of the Relational Model

But how is this data ACTUALLY stored?

<table>
<thead>
<tr>
<th>Payroll</th>
<th>UserID</th>
<th>Name</th>
<th>Job</th>
<th>Salary</th>
</tr>
</thead>
<tbody>
<tr>
<td>123</td>
<td>Jack</td>
<td>TA</td>
<td>50000</td>
<td></td>
</tr>
<tr>
<td>345</td>
<td>Allison</td>
<td>TA</td>
<td>60000</td>
<td></td>
</tr>
<tr>
<td>567</td>
<td>Magda</td>
<td>Prof</td>
<td>90000</td>
<td></td>
</tr>
<tr>
<td>789</td>
<td>Dan</td>
<td>Prof</td>
<td>100000</td>
<td></td>
</tr>
</tbody>
</table>

Characteristics of the Relational Model

But how is this data ACTUALLY stored?

<table>
<thead>
<tr>
<th>Payroll</th>
<th>UserID</th>
<th>Name</th>
<th>Job</th>
<th>Salary</th>
</tr>
</thead>
<tbody>
<tr>
<td>123</td>
<td>Jack</td>
<td>TA</td>
<td>50000</td>
<td></td>
</tr>
<tr>
<td>345</td>
<td>Allison</td>
<td>TA</td>
<td>60000</td>
<td></td>
</tr>
<tr>
<td>567</td>
<td>Magda</td>
<td>Prof</td>
<td>90000</td>
<td></td>
</tr>
<tr>
<td>789</td>
<td>Dan</td>
<td>Prof</td>
<td>100000</td>
<td></td>
</tr>
</tbody>
</table>

Structured Query Language - SQL

Alright, I have data and a schema. How do I access it?

Structured Query Language - SQL

"SQL (standing for Structured Query Language) is the standard language for relational database management systems. When it originated back in the 1970s, the domain-specific, database-oriented language could navigate through a network of pointers to find the desired location. Its application in handling structured data has fostered in the Digital Age. In fact, the powerful database manipulation and definition capabilities of SQL, and its intuitive tabular view have become available in some form on virtually every important computer platform in the world.

Some notable features of SQL include the ability to process sets of data as groups instead of individual units, automatic navigation to data, and the use of statements that are complex and powerful individually. Used for a variety of tasks, SQL lets users work with data at the logical level."


Structured Query Language - SQL

Key points about SQL:

- A domain-specific language
- SQL only works on relational databases
- Not for general purpose programming (Java, C/C++, …)
- Logical level of interaction with data

Hello World

<table>
<thead>
<tr>
<th>Payroll</th>
<th>UserID</th>
<th>Name</th>
<th>Job</th>
<th>Salary</th>
</tr>
</thead>
<tbody>
<tr>
<td>123</td>
<td>Jack</td>
<td>TA</td>
<td>50000</td>
<td></td>
</tr>
<tr>
<td>345</td>
<td>Allison</td>
<td>TA</td>
<td>60000</td>
<td></td>
</tr>
<tr>
<td>567</td>
<td>Magda</td>
<td>Prof</td>
<td>90000</td>
<td></td>
</tr>
<tr>
<td>789</td>
<td>Dan</td>
<td>Prof</td>
<td>100000</td>
<td></td>
</tr>
</tbody>
</table>

SELECT p.Name, p.UserID
FROM Payroll AS P
WHERE P.Job = 'TA';
Hello World

**Payroll**

<table>
<thead>
<tr>
<th>UserID</th>
<th>Name</th>
<th>Job</th>
<th>Salary</th>
</tr>
</thead>
<tbody>
<tr>
<td>123</td>
<td>Jack</td>
<td>TA</td>
<td>50000</td>
</tr>
<tr>
<td>345</td>
<td>Allison</td>
<td>TA</td>
<td>60000</td>
</tr>
<tr>
<td>567</td>
<td>Magda</td>
<td>Prof</td>
<td>90000</td>
</tr>
<tr>
<td>789</td>
<td>Dan</td>
<td>Prof</td>
<td>100000</td>
</tr>
</tbody>
</table>

**SELECT** p.Name, p.UserID
FROM Payroll AS P
WHERE p.Job = 'TA';

---

**Hello World**

**Payroll**

<table>
<thead>
<tr>
<th>UserID</th>
<th>Name</th>
<th>Job</th>
<th>Salary</th>
</tr>
</thead>
<tbody>
<tr>
<td>123</td>
<td>Jack</td>
<td>TA</td>
<td>50000</td>
</tr>
<tr>
<td>345</td>
<td>Allison</td>
<td>TA</td>
<td>60000</td>
</tr>
<tr>
<td>567</td>
<td>Magda</td>
<td>Prof</td>
<td>90000</td>
</tr>
<tr>
<td>789</td>
<td>Dan</td>
<td>Prof</td>
<td>100000</td>
</tr>
</tbody>
</table>

**SELECT** p.Name, p.UserID
FROM Payroll AS P
WHERE p.Job = 'TA';

---

**Hello World**

**Payroll**

<table>
<thead>
<tr>
<th>UserID</th>
<th>Name</th>
<th>Job</th>
<th>Salary</th>
</tr>
</thead>
<tbody>
<tr>
<td>123</td>
<td>Jack</td>
<td>TA</td>
<td>50000</td>
</tr>
<tr>
<td>345</td>
<td>Allison</td>
<td>TA</td>
<td>60000</td>
</tr>
<tr>
<td>567</td>
<td>Magda</td>
<td>Prof</td>
<td>90000</td>
</tr>
<tr>
<td>789</td>
<td>Dan</td>
<td>Prof</td>
<td>100000</td>
</tr>
</tbody>
</table>

**SELECT** p.Name, p.UserID
FROM Payroll AS P
WHERE p.Job = 'TA';

---

**SQL**

How does a computer understand abstract SQL text?
Database Internals

- Code has to boil down to instructions at some point
- Relational Database Management Systems (RDBMSs) use Relational Algebra (RA)

```
SELECT P.Name, P.UserID
FROM Payroll AS P
WHERE P.Job = 'TA';
```

June 24, 2019

Introduction

58

Hello World

<table>
<thead>
<tr>
<th>UserID</th>
<th>Name</th>
<th>Job</th>
<th>Salary</th>
</tr>
</thead>
<tbody>
<tr>
<td>123</td>
<td>Jack</td>
<td>TA</td>
<td>50000</td>
</tr>
<tr>
<td>345</td>
<td>Allison</td>
<td>TA</td>
<td>60000</td>
</tr>
<tr>
<td>567</td>
<td>Magda</td>
<td>Pro</td>
<td>90000</td>
</tr>
<tr>
<td>789</td>
<td>Dan</td>
<td>Pro</td>
<td>100000</td>
</tr>
</tbody>
</table>

June 24, 2019

Introduction

59

Database Internals

- Code has to boil down to instructions at some point
- Relational Database Management Systems (RDBMSs) use Relational Algebra (RA)

```
SELECT P.Name, P.UserID
FROM Payroll AS P
WHERE P.Job = 'TA';
```

June 24, 2019

Introduction

60

Database Internals

- Code has to boil down to instructions at some point
- Relational Database Management Systems (RDBMSs) use Relational Algebra (RA).

For each semantics

```
π P.Name, P.UserID
σ P.Job=TA

Payroll P
```

June 24, 2019

Introduction

61

Database Internals

- Code has to boil down to instructions at some point
- Relational Database Management Systems (RDBMSs) use Relational Algebra (RA).

```
SELECT P.Name, P.UserID
FROM Payroll AS P
WHERE P.Job = 'TA';
```

June 24, 2019

Introduction

62
Hello World

**SELECT** P.Name, P.UserID
FROM Payroll AS P
WHERE P.Job = 'TA';

June 24, 2019

**Introduction**

<table>
<thead>
<tr>
<th>UserID</th>
<th>Name</th>
<th>Job</th>
<th>Salary</th>
</tr>
</thead>
<tbody>
<tr>
<td>123</td>
<td>Jack</td>
<td>TA</td>
<td>50000</td>
</tr>
<tr>
<td>345</td>
<td>Allison</td>
<td>TA</td>
<td>60000</td>
</tr>
<tr>
<td>567</td>
<td>Magda</td>
<td>Prof</td>
<td>90000</td>
</tr>
<tr>
<td>789</td>
<td>Dan</td>
<td>Prof</td>
<td>100000</td>
</tr>
</tbody>
</table>

Payroll

Hello World

**SELECT** P.Name, P.UserID
FROM Payroll AS P
WHERE P.Job = 'TA';

June 24, 2019

**Introduction**

<table>
<thead>
<tr>
<th>UserID</th>
<th>Name</th>
<th>Job</th>
<th>Salary</th>
</tr>
</thead>
<tbody>
<tr>
<td>123</td>
<td>Jack</td>
<td>TA</td>
<td>50000</td>
</tr>
<tr>
<td>345</td>
<td>Allison</td>
<td>TA</td>
<td>60000</td>
</tr>
<tr>
<td>567</td>
<td>Magda</td>
<td>Prof</td>
<td>90000</td>
</tr>
<tr>
<td>789</td>
<td>Dan</td>
<td>Prof</td>
<td>100000</td>
</tr>
</tbody>
</table>

Payroll

Hello World

**SELECT** P.Name, P.UserID
FROM Payroll AS P
WHERE P.Job = 'TA';

June 24, 2019

**Introduction**

<table>
<thead>
<tr>
<th>UserID</th>
<th>Name</th>
<th>Job</th>
<th>Salary</th>
</tr>
</thead>
<tbody>
<tr>
<td>123</td>
<td>Jack</td>
<td>TA</td>
<td>50000</td>
</tr>
<tr>
<td>345</td>
<td>Allison</td>
<td>TA</td>
<td>60000</td>
</tr>
<tr>
<td>567</td>
<td>Magda</td>
<td>Prof</td>
<td>90000</td>
</tr>
<tr>
<td>789</td>
<td>Dan</td>
<td>Prof</td>
<td>100000</td>
</tr>
</tbody>
</table>

Payroll

Hello World

**SELECT** P.Name, P.UserID
FROM Payroll AS P
WHERE P.Job = 'TA';

June 24, 2019

**Introduction**

<table>
<thead>
<tr>
<th>UserID</th>
<th>Name</th>
<th>Job</th>
<th>Salary</th>
</tr>
</thead>
<tbody>
<tr>
<td>123</td>
<td>Jack</td>
<td>TA</td>
<td>50000</td>
</tr>
<tr>
<td>345</td>
<td>Allison</td>
<td>TA</td>
<td>60000</td>
</tr>
<tr>
<td>567</td>
<td>Magda</td>
<td>Prof</td>
<td>90000</td>
</tr>
<tr>
<td>789</td>
<td>Dan</td>
<td>Prof</td>
<td>100000</td>
</tr>
</tbody>
</table>

Payroll
Today’s Takeaways!

- The Relational Model concept
- How a basic SELECT-FROM-WHERE query works
- Basic execution process (RA) inside a RDBMS