Databases: Part 1

Alex Galakatos

November 24th 2015

What is a database?

- System for storing/retrieving data
- Used for
  - Social networking
  - Banking
  - Billing
  - Much much more

What can I do with a database?

- Insert data
- Retrieve data
- Update data
- Delete data

Do you need a database?

- NO!

students.txt

John, 19, Math, 3.4
Jason, 20, English, 3.2
Sarah, 21, History, 3.8
Sam, 18, Math, 2.8

employees.txt

Ethan, 49, $40k
Kyle, 37, $20k
Andrew, 33, $60k
Tim, 45, $25k

courses.txt

CS 101, 351, 120
Math 205, 220, 80
HIST 170, 301, 80
ENG 405, 110, 30

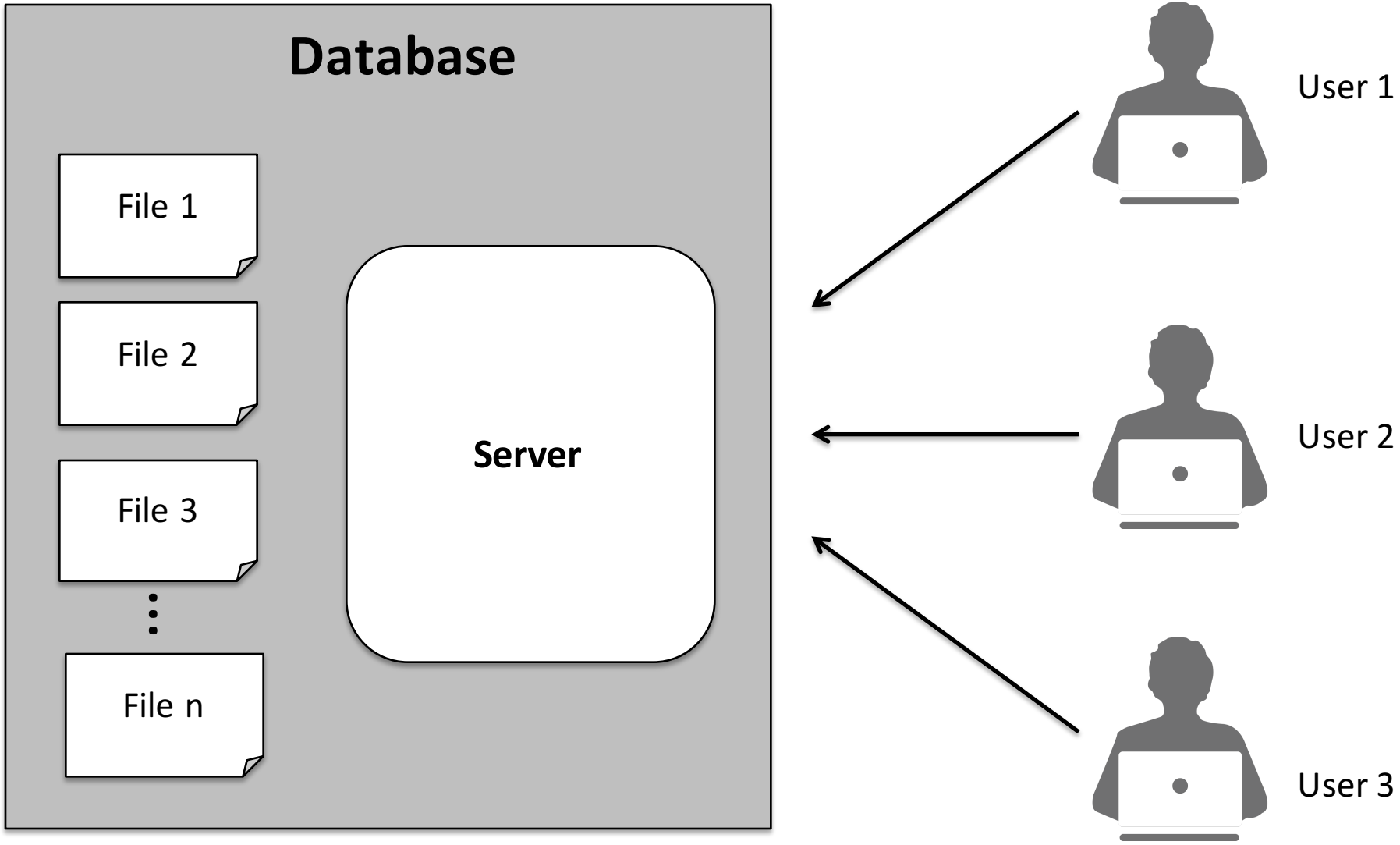
Find the name of all students who are 18 years old

```
for line in open("students.txt"):
    name, age, dept, gpa = line.split(",")
    if age == '18':
        print name
```

Why use a database?

- Provides many “guarantees”
  - **A**tomicity
  - **C**onsistency
  - **I**solation
  - **D**urability
- Multi-user concurrency
- Indexing/optimizations

Database Architecture



Table

- Collection of rows (a.k.a. records, tuples)
- Arbitrary number of columns (a.k.a. attributes)

Columns store a specific data type

Row
entry in table

Name	Age	Major	GPA
John	19	Math	3.4
Jason	20	English	3.2
Sarah	21	History	3.8
Sam	18	Math	2.8

`Students` Table

Inside a database

Students

Name	Age	Major	GPA
John	19	Math	3.4
Jason	20	English	3.2
Sarah	21	History	3.8
Sam	18	Math	2.8

Employees

Name	Age	Salary
Ethan	49	$40k
Kyle	37	$20k
Andrew	33	$60k
Tim	45	$25k

Courses

Name	Room	Capacity
CS 101	351	120
Math 205	220	80
HIST 170	301	80
ENG 405	110	30

Interacting with a database

- SQL (Structured Query Language)
- Declarative
- Simple, easy to write and understand
- Extremely powerful
- (Mostly) compatible between databases

Basic Operations

- Make a new table
  - **`CREATE TABLE`**
- Add records to a table
  - **`INSERT`**
- Retrieve records from a table
  - **`SELECT`**
- Modify records in a table
  - **`UPDATE`**
- Delete records from a table
  - **`DELETE`**

CREATE TABLE

- Creates an empty new table
- Collection of named columns + types
- Column Types
  - Integer (42, 100)
  - Float (16.72534, 3.141598)
  - Varchar (“male”, “female”)
  - … many more

CREATE TABLE

- Example: make a table to hold students at a university

```
CREATE TABLE student (
  id integer,
  name varchar,
  age integer,
  gpa float
);
```

id	name	age	GPA

Activity: Task 1

INSERT

- Add rows (a.k.a. records, tuples) to a table
- Table must already exist
- Number of values + types must match table definition
- Strings (e.g., names) must have single quotes

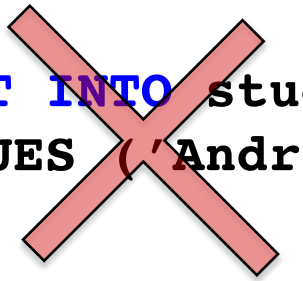
INSERT

- Add students to the student table

```
INSERT INTO student
  VALUES (1, 'Matt', 20, 3.45);

INSERT INTO student
  VALUES (2, 'Henry', 19, 3.89);
       ⋮
```

id	name	age	GPA
1	Matt	20	3.45
2	Henry	19	3.89



```
INSERT INTO student
  VALUES ('Andrew', 22, 2.5);
```

Activity: Task 2

SELECT

- Retrieve records from a table
- Basic structure
  - **SELECT**
    - Which columns to output
  - **FROM**
    - What table to look at
  - **WHERE (optional)**
    - Filtering condition

SELECT

```
SELECT name
FROM student
```

name
Matt
Henry
Ally
John
Liz

```
SELECT name, age
FROM student
```

name	age
Matt	20
Henry	19
Ally	21
John	21
Liz	18

```
SELECT *
FROM student
```

id	name	age	GPA
1	Matt	20	3.45
2	Henry	19	3.89
3	Ally	21	3.95
4	John	21	2.85
5	Liz	18	3.45

Predicates + AND/OR

- Filtering
  - {=,!=, >, <,>=, <=, etc.}
  - `WHERE [column] PRED [value]`
  - `[value]` can be a constant or another column

- Arbitrary number of predicates
  - Linked using AND/OR
  - `WHERE [column1] PRED [value1] (AND/OR) [column2] PRED [value2] …`

SELECT

```
SELECT *
FROM student
```

id	name	age	GPA
1	Matt	20	3.45
2	Henry	19	3.89
3	Ally	21	3.95
4	John	21	2.85
5	Liz	18	3.45

```
SELECT *
FROM student
WHERE GPA > 3.5
```

id	name	age	GPA
2	Henry	19	3.89
3	Ally	21	3.95

```
SELECT *
FROM student
WHERE age < 20
```

id	name	age	GPA
2	Henry	19	3.89
5	Liz	18	3.45

SELECT

```
SELECT *
FROM student
```

id	name	age	GPA
1	Matt	20	3.45
2	Henry	19	3.89
3	Ally	21	3.95
4	John	21	2.85
5	Liz	18	3.45

```
SELECT *
FROM student
WHERE GPA > 3.5
  AND age > 20
```

id	name	age	GPA
3	Ally	21	3.95

```
SELECT *
FROM student
WHERE age <= 20
   OR GPA < 3
```

id	name	age	GPA
1	Matt	20	3.45
2	Henry	19	3.89
4	John	21	2.85
5	Liz	18	3.45

Activity: Task 3

UPDATE

- Modify a record in a table
- Assign provided columns a new value
- Optionally only update a subset of rows
- Basic structure
  - UPDATE
    - Table to update
  - SET
    - New value to assign to given column
  - WHERE (optional)
    - Filter rows to update

UPDATE

```
SELECT *
FROM student
```

id	name	age	GPA
1	Matt	20	3.45
2	Henry	19	3.89
3	Ally	21	3.95
4	John	21	2.85
5	Liz	18	3.45

```
UPDATE student
SET GPA = 3.8
WHERE id = 1
```

id	name	age	GPA
1	**Matt**	**20**	**3.8**
2	Henry	19	3.89
3	Ally	21	3.95
4	John	21	2.85
5	Liz	18	3.45

```
UPDATE student
SET age = age + 1,
   gpa = gpa/10
```

id	name	age	GPA
1	**Matt**	**21**	**.38**
2	**Henry**	**20**	**.389**
3	**Ally**	**22**	**.395**
4	**John**	**22**	**.285**
5	**Liz**	**19**	**.345**

Activity: Task 4

DELETE

- Remove entire rows from a table
- Basic structure
  - DELETE
  - FROM
    - Table to delete rows from
  - WHERE (optional)
    - Delete only subset of rows
- **Without “WHERE”, all rows will be deleted!!!!**

DELETE

```
SELECT *
FROM student
```

id	name	age	GPA
1	Matt	20	3.45
2	Henry	19	3.89
3	Ally	21	3.95
4	John	21	2.85
5	Liz	18	3.45

```
DELETE FROM student
WHERE id = 1
    OR id = 2
```

id	name	age	GPA
3	Ally	21	3.95
4	John	21	2.85
5	Liz	18	3.45

```
DELETE FROM student
```

id	name	age	GPA

Activity: Task 5

Questions?