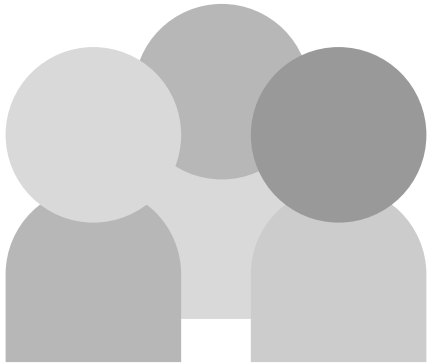
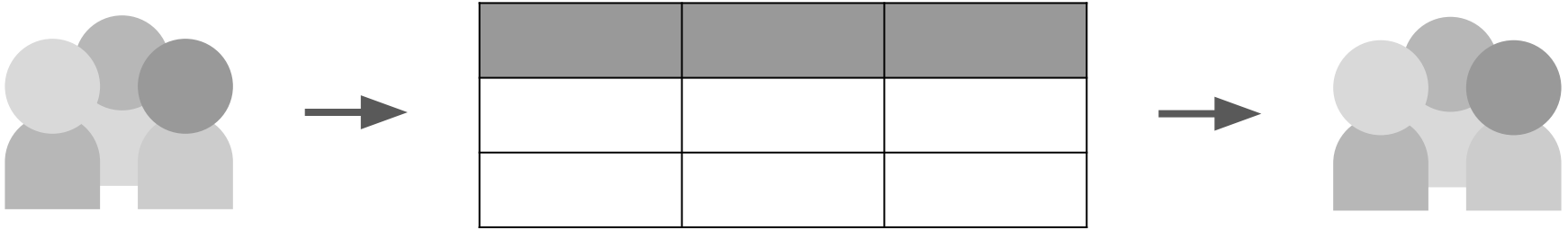
# This is CS50

**Think.**
**Pair.**
**Share.**

cs50.ly/questions





Collection

Storage

Use

**Think.**
**Pair.**
**Share.**

cs50.ly/youtunes
cs50.ly/critiques

# Quick Technical Design Principles

- Each table should be a collection of a **single entity**.
  - For example, we should have a different table for each of employees, employee *relationships*, songs, and artists.

# youtunes.db

## employees

| name | role |
| --- | --- |
| Alice | IT Staff |
| Laura | General Manager |

## employee_relationships

| manager | employee |
| --- | --- |
| Laura | Alice |

# Quick Technical Design Principles

- Each table should be a collection of a **single entity.**
  - For example, we should have a different table for each of employees, employee *relationships*, songs, and artists.

- Each piece of data should be stored in a **single location**, and thereafter referred to via its ID, aka primary key.
  - For example, we should ensure every employee has an ID, and use that ID in the employee *relationships* table.

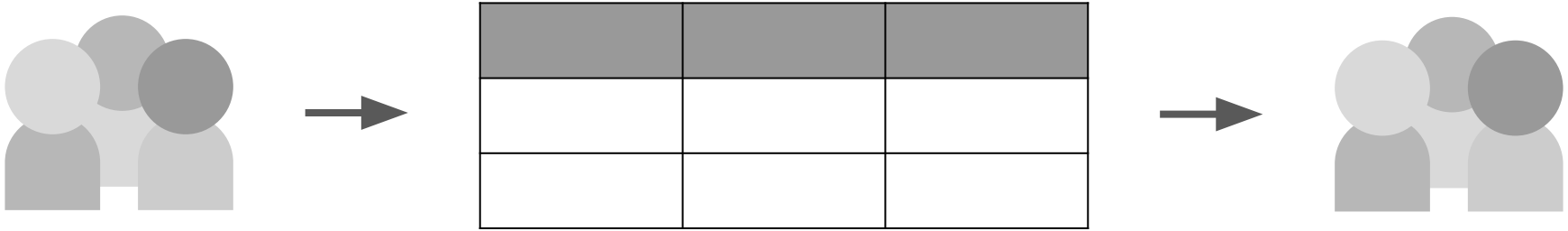
# youtunes.db

## employees

| id | name | role |
|---|---|---|
| 1 | Alice | IT Staff |
| 2 | Laura | General Manager |

## employee_relationships

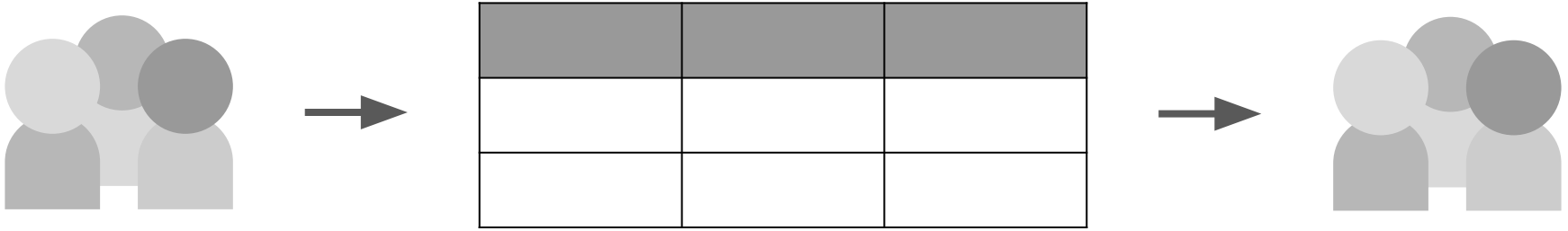
| manager_id | employee_id |
|---|---|
| 2 | 1 |



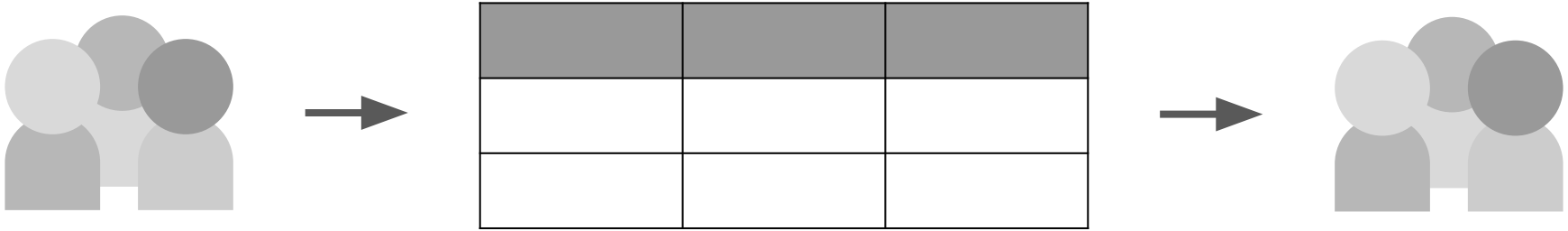
Collection

**Storage**

Use



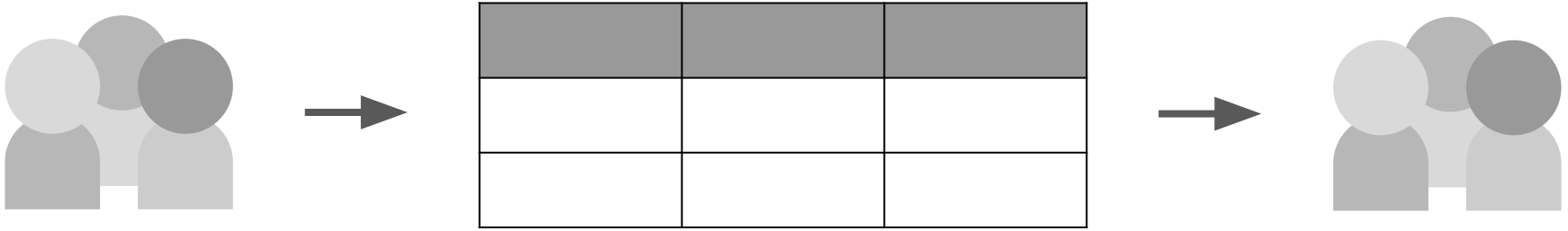
**Collection** Storage Use



Collection

**Storage**

Use



Collection

Storage

**Use**

# EthiCS

# Database Design Principles

Ideally, when designing databases, data practitioners should strive to create databases that at least:

- Minimize redundancies [increases efficiency]
- Permit adaptability [easier to add new data]

**Today:** *examine how databases are sometimes used in the "real world" to see if we can generate new design principles.*

# *Wrongfully Accused by an Algorithm*

In what may be the first known case of its kind, a faulty facial recognition match led to a Michigan man's arrest for a crime he did not commit.

**Think.**
**Pair.**
**Share.**

cs50.ly/casestudy
cs50.ly/principles

# Database Design Practice

# Create a Database

1. In your terminal, use **`sqlite3 youtunes.db`** to create a new database, **`youtunes.db`**.

# Create a Database

1. In your terminal, use **`sqlite3 youtunes.db`** to create a new database, **`youtunes.db`**.
2. Create a table, named **`employees`**, with at least two columns, **id**, **name**, and at least one additional employee attribute you'd like to store. Denote **id** as the primary key.

```
CREATE TABLE "employees" (
    "id",
    "first_name",
    "last_name",
    PRIMARY KEY("id")
);
```

```
CREATE TABLE "employees" (
    "id" INTEGER,
    "first_name" TEXT,
    "last_name" TEXT,
    PRIMARY KEY("id")
);
```

```
CREATE TABLE "employees" (
    "id" INTEGER NOT NULL,
    "first_name" TEXT NOT NULL,
    "last_name" TEXT NOT NULL,
    PRIMARY KEY("id")
);
```

**NOT NULL**

"required"

# Insert into a Database

Ensure you're still in your sqlite prompt by looking at your terminal prefix.

1. Use `INSERT INTO` to add at least 3 employees to your table.
2. Use `SELECT` to ensure that your employees have been added.

```
INSERT INTO tablename (column1, column2)
VALUES (value1, value2);
```

```
SELECT * FROM tablename;
```

# Lab

# songs.db

## songs

| id | name | tempo | duration | artist_id |
|---|---|---|---|---|
| 1 | Something Comforting | 144 | 282 | 23 |
| 2 | Drive | 142 | 196 | 45 |

## artists

| id | name | age | label |
|---|---|---|---|
| 23 | Porter Robinson | 29 | Mom+Pop |
| 45 | Oh Wonder | 31 | Republic |

1–5

Selecting

Ordering

Limiting

Aggregating

# Selecting

```
SELECT name
FROM songs
WHERE duration < 240;
```

# songs.db

## songs

| id | name | tempo | duration | artist_id |
|---|---|---|---|---|
| 1 | Something Comforting | 144 | 282 | 23 |
| 2 | Drive | 142 | 196 | 45 |

## artists

| id | name | age | label |
|---|---|---|---|
| 23 | Porter Robinson | 29 | Mom+Pop |
| 45 | Oh Wonder | 31 | Republic |

# Ordering

```
SELECT *
FROM songs
WHERE tempo > 100
ORDER BY tempo DESC;
```

```
SELECT *
FROM songs
WHERE tempo > 100
ORDER BY tempo ASC;
```

# Limiting

```
SELECT *
FROM songs
WHERE tempo > 100
ORDER BY tempo ASC
LIMIT 1;
```

# Aggregating

```
SELECT COUNT(*)
FROM songs
WHERE tempo > 100;
```

```
SELECT AVG(tempo)
FROM songs
WHERE tempo > 100;
```

1–5

6–8

# Nested SELECTs
# JOINs

# Nested SELECTs

JOINs

# songs.db

## songs

| id | name | tempo | duration | artist_id |
|---|---|---|---|---|
| 1 | Something Comforting | 144 | 282 | 23 |
| 2 | Drive | 142 | 196 | 45 |

## artists

| id | name | age | label |
|---|---|---|---|
| 23 | Porter Robinson | 29 | Mom+Pop |
| 45 | Oh Wonder | 31 | Republic |

```
SELECT *
FROM songs
WHERE artist_id IN
(
    SELECT id
    FROM artists
    WHERE name = "Oh Wonder"
);
```

```
SELECT *
FROM songs
WHERE artist_id IN
(
    SELECT id
    FROM artists
    WHERE name = "Oh Wonder"
);
```

# songs.db

## songs

| id | name | tempo | duration | artist_id |
|---|---|---|---|---|
| 1 | Something Comforting | 144 | 282 | 23 |
| 2 | Drive | 142 | 196 | 45 |

## artists

| id | name | age | label |
|---|---|---|---|
| 23 | Porter Robinson | 29 | Mom+Pop |
| 45 | Oh Wonder | 31 | Republic |

```
SELECT *
FROM songs
WHERE artist_id IN
(
    45
);
```

```
SELECT *
FROM songs
WHERE artist_id IN
(
    45
);
```

# songs.db

## songs

| id | name | tempo | duration | artist_id |
|---|---|---|---|---|
| 1 | Something Comforting | 144 | 282 | 23 |
| 2 | Drive | 142 | 196 | 45 |

## artists

| id | name | age | label |
|---|---|---|---|
| 23 | Porter Robinson | 29 | Mom+Pop |
| 45 | Oh Wonder | 31 | Republic |

Nested SELECTs

JOINs

# songs.db

## songs

| id | name | tempo | duration | artist_id |
|---|---|---|---|---|
| 1 | Something Comforting | 144 | 282 | 23 |
| 2 | Drive | 142 | 196 | 45 |

## artists

| id | name | age | label |
|---|---|---|---|
| 23 | Porter Robinson | 29 | Mom+Pop |
| 45 | Oh Wonder | 31 | Republic |

```
SELECT *
FROM songs
JOIN artists
ON songs.artist_id = artists.id;
```

# songs JOIN artists

| id | name | tempo | duration | artist_id | name | age | label |
|---|---|---|---|---|---|---|---|
| 1 | Something Comforting | 144 | 282 | 23 | Porter Robinson | 29 | Mom+ Pop |
| 2 | Drive | 142 | 196 | 45 | Oh Wonder | 31 | Republic |

```
SELECT *
FROM songs
JOIN artists
ON songs.artist_id = artists.id;
```

```
SELECT *
FROM songs
JOIN artists
ON songs.artist_id = artists.id
WHERE artists.name = "Oh Wonder";
```

# LIKE

% indicates wildcard characters in relative location of string:

| | |
|---|---|
| `WHERE title LIKE "%Percy Jackson"` | All titles with Percy Jackson at the end. |
| `WHERE title LIKE "Percy Jackson%"` | All titles with Percy Jackson at the beginning. |
| `WHERE title LIKE "%Percy Jackson%"` | All titles with Percy Jackson somewhere in the string. |

6–8

Tutorials

Office Hours

cs50.ly/attend


This was CS50