Week 2

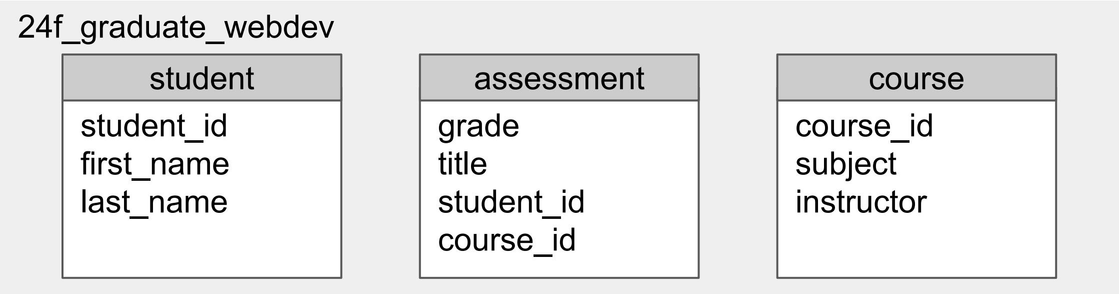
Accessing Data

Week 2 Agenda

- Lecture:
    - Introduction to Database Architecture Diagrams
    - Introduction to SQL
        - Accessing Data Queries
        - Filtering Data Queries
- Lab 2 (5%)

Database Architecture Diagrams

- Table diagrams describe the columns (data fields) of a table

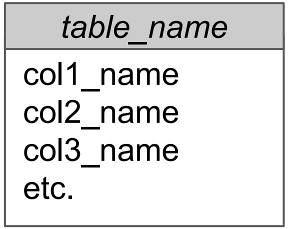


24f_graduate_webdev

student
student_id
first_name
last_name

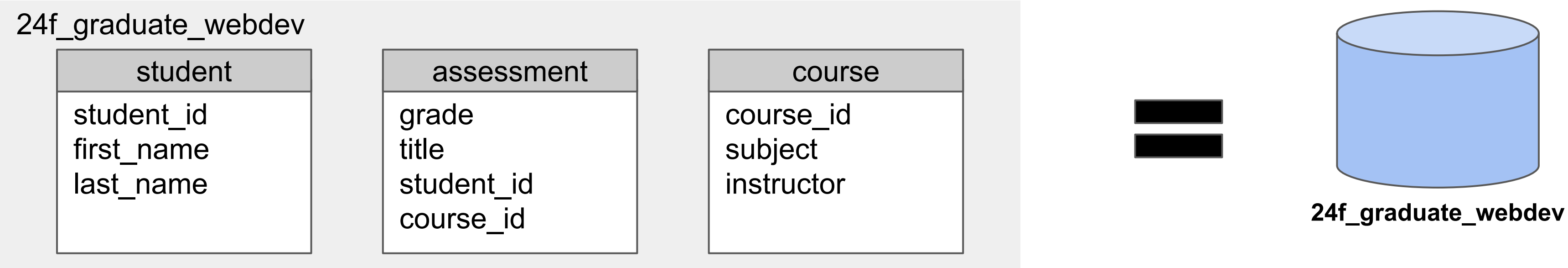
assessment
grade
title
student_id
course_id

course
course_id
subject
instructor



table_name
col1_name
col2_name
col3_name
etc.

- Databases are represented as cylinders in architecture diagrams



24f_graduate_webdev

student
student_id
first_name
last_name

assessment
grade
title
student_id
course_id

course
course_id
subject
instructor

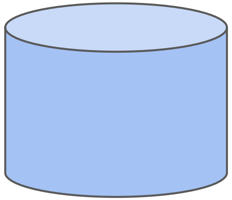
=

24f_graduate_webdev

Database Naming Conventions

- **Naming conventions** are defined by software designers to emphasize **consistency**
- **General**
  - **Full words** not abbreviations and acronyms
  - Avoid redundancy, do not prefix names with the name of their parent
  - Names should be meaningful and self-explanatory
    - db/table/column name should reflect their real world purpose
  - Names should be lowercase since SQL keywords are UPPERCASE
  - snake_case: underscore_in_place_of_spaces
- **Databases**
  - Singular name that describes information held in db
- **Tables**
  - Names should be **nouns**, 1 or 2 words
  - Table names may be singular OR plural ***BUT BE CONSISTENT**
- **Columns**
  - Names should be 1 or 2 words and singular

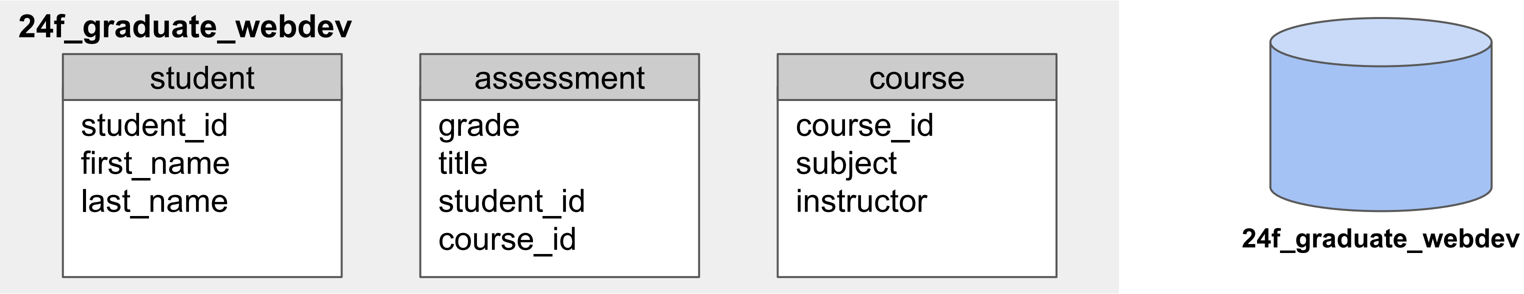
assessment
assessment_id grade title due_date student_id



24f_graduate_webdev

Naming Conventions

- Conventions are rules for how to name structures or variables
- Conventions are NOT laws, they will not (usually) break software
- These are decided by teams to keep clean, consistent, and readable code
- For this class follow the conventions from the previous slide
  - Ex:



24f_graduate_webdev

student
student_id first_name last_name

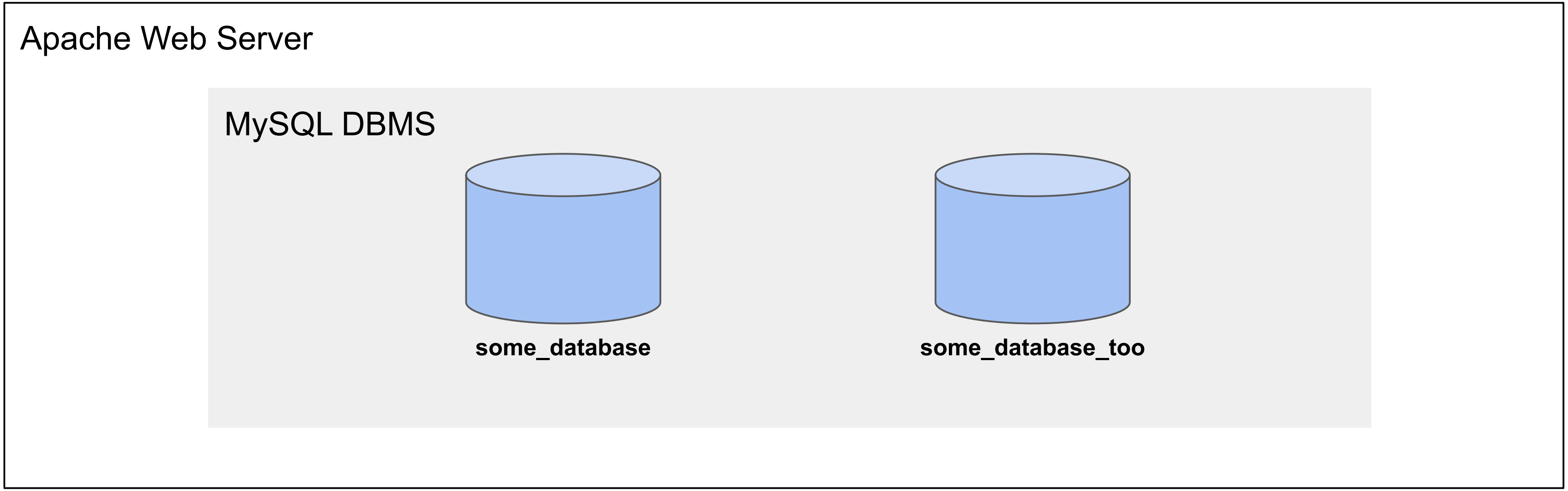
assessment
grade title student_id course_id

course
course_id subject instructor

24f_graduate_webdev

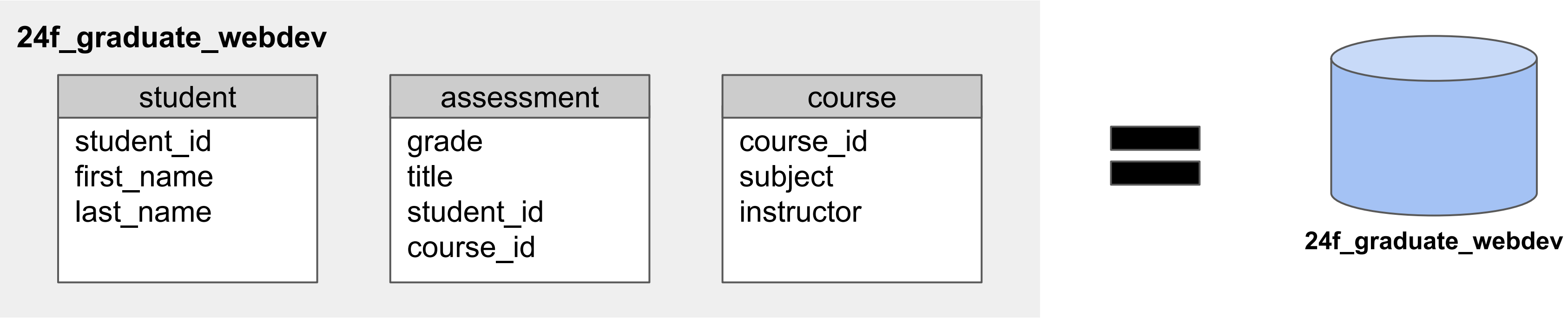
Setting Up

- So to work with DBs we need:
    a. Web Server - **Apache** Web Server
    b. DBMS - **MySQL** Database Management System
    c. Database - A place to store our data tables



(Database Management System)

DBMS Architecture



24f_graduate_webdev

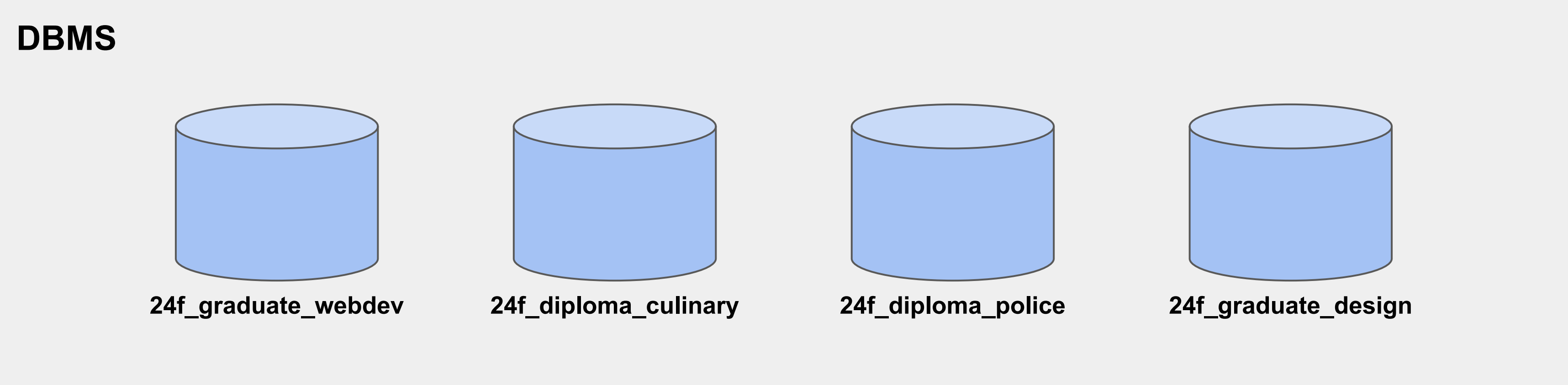
student
student_id
first_name
last_name

assessment
grade
title
student_id
course_id

course
course_id
subject
instructor

=

24f_graduate_webdev



DBMS

24f_graduate_webdev

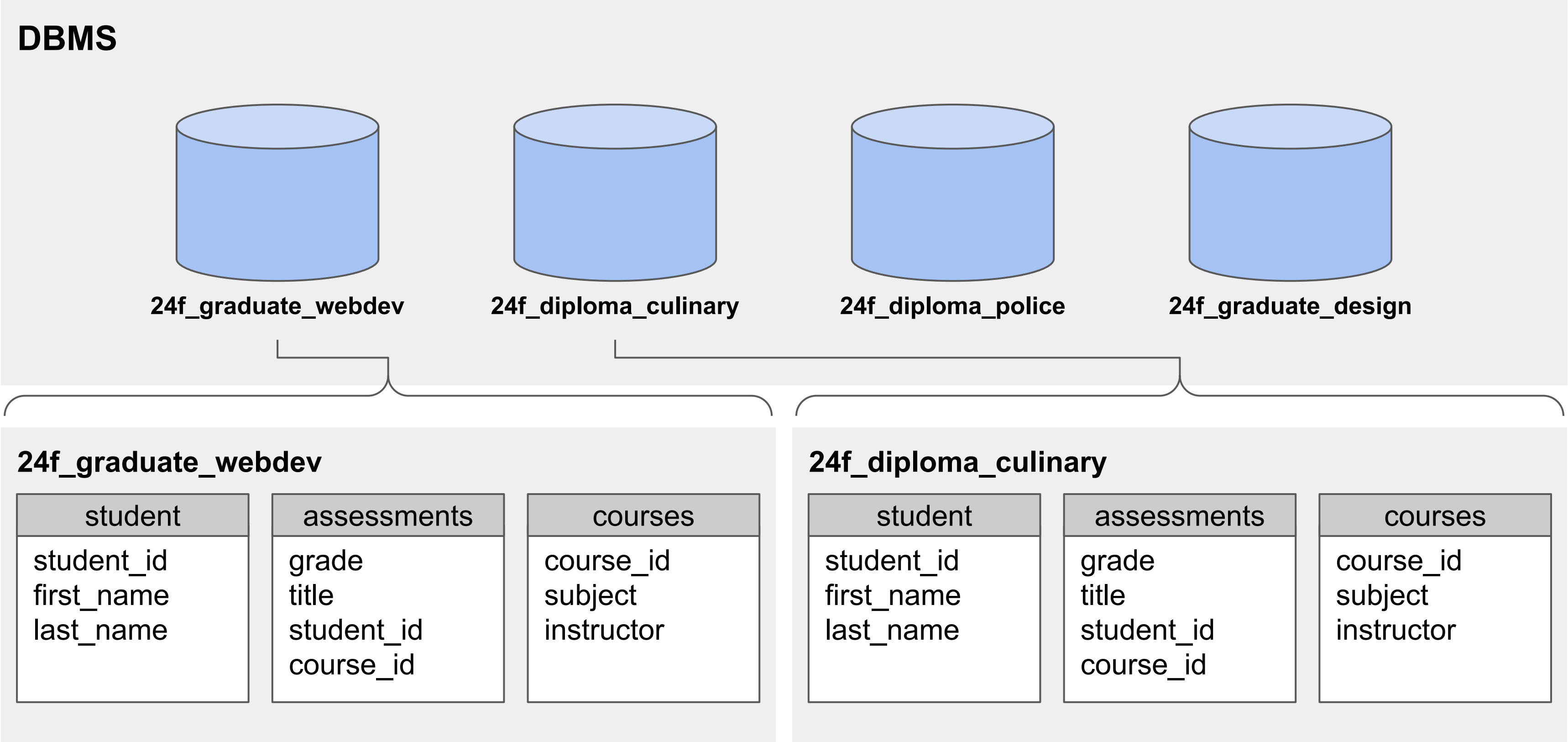
24f_diploma_culinary

24f_diploma_police

24f_graduate_design

(Database Management System)

DBMS Architecture



DBMS

24f_graduate_webdev

24f_diploma_culinary

24f_diploma_police

24f_graduate_design

24f_graduate_webdev

student
student_id first_name last_name

assessments
grade title student_id course_id

courses
course_id subject instructor

24f_diploma_culinary

student
student_id first_name last_name

assessments
grade title student_id course_id

courses
course_id subject instructor

Getting Started with Databases

- To begin using **databases(DBs)** the DBs must exist and be managed somewhere
- **Database Management Systems (DBMS)** are the software that will give databases a home to exist in
- a DBMS also needs somewhere to exist that place is typically a **web server**
- a **Web Server** is basically a computer running software, like a DBMS, that can send and receive data

Accessing Data with SQL[1]

Standardized Query Language

- With a DB created in a DBMS that is running on a Web Server, SQL can be used to access data
- **SQL (Standardized Query Language)** is a query programming language
    - (pronounced S-Q-L; historically "sequel")
- **SQL is used to** write queries/requests/statements that ask a database to **perform actions on tables**
- Written **SQL queries** can be **executed to retrieve, insert, update, or delete** information in databases
- SQL can do other actions as well, but our focus for now will be retrieving

[1] https://www.w3schools.com/sql/sql_intro.asp

SQL Syntax[1]

- SQL statements consists of reserved keywords
- SQL convention is to write KEYWORDS in UPPERCASE
- The following SQL statement returns all records from a table named "customers":

    -
      ```
      SELECT * FROM customers;
      ```

In SQL an asterisk is known as a ‘wildcard’ operator. In practice it mean ‘all’. Therefore this statement reads *select all columns from the customers table*

***SQL** is **not case-sensitive**, so `select` is the same as `SELECT`

*However **SQL keywords** should **always** be written in **UPPERCASE**, this is a near universal SQL coding convention

[1] https://www.w3schools.com/sql/sql_syntax.asp

SELECT[1]

- **SELECT** statement is used to select data from a database
  - `SELECT customer_name, city FROM customer;`

Syntax

```
SELECT column_1, column_2, ...
FROM table_name;
```

***SQL Convention:** Semicolons should be used to terminate statements

[1] https://www.w3schools.com/sql/sql_select.asp

SELECT DISTINCT[1]

- **DISTINCT** statement is used to select only **unique** data from a DB table

  -
    ```
    SELECT country FROM customer;
    ```

Syntax

```
SELECT DISTINCT column_1, column_2, ...
FROM table_name;
```

With DISTINCT

Country
Germany
Mexico
UK

Without DISTINCT

Country
Germany
Mexico
Mexico
UK

[1] https://www.w3schools.com/sql/sql_distinct.asp

WHERE[1]

- **WHERE** keyword is used to filter data from a table
- Will only select records that fulfill a specified condition

```
SELECT * FROM customer WHERE country='Mexico';
```

Syntax

```
SELECT column_1, column_2, ...
FROM table_name
WHERE condition;
```

[1] https://www.w3schools.com/sql/sql_where.asp

Comparison Operators[1]

=	Equal to	WHERE column = 10;
<>	NOT Equal to	WHERE column <> 10;
>	Greater Than	WHERE column > 10;
<	Less Than	WHERE column < 10;
>=	Greater Than or Equal to	WHERE column >= 10;
<=	Less Than or Equal to	WHERE column <= 10;

[1] https://www.w3schools.com/sql/sql_operators.asp#:~:text=SQL%20Comparison%20Operators

AS / Column Alias[1]

- The **AS** command is used to rename a column or table with an alias
- An alias only exists for the duration of the query

```
SELECT customer_id AS 'ID', name AS 'Customer'
FROM customer;
```

Syntax

```
SELECT column_1 AS 'Alias 1', column_2 AS 'Alias 2', ...
FROM table_name;
```

[1] https://www.w3schools.com/sql/sql_ref_as.asp

AS / Column Alias Example

```sql
SELECT customer_id AS 'ID', name AS 'Customer'
FROM customer;
```

With Alias

ID	Customer
1	Alfreds Futterkiste
2	Ana Trujillo Emparedados y helados

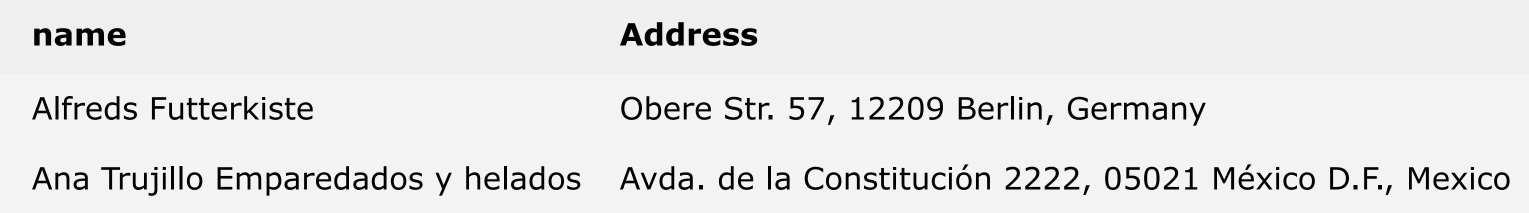
Without Alias

customer_id	name
1	Alfreds Futterkiste
2	Ana Trujillo Emparedados y helados

SQL Function: CONCAT[1]

- We can combine columns in output tables using the **CONCAT** function then use an alias to accurate label the output

```
SELECT name, CONCAT(address,', ',postal_code,', ',city,',
',country) AS 'Address' FROM customer;
```



name	Address
Alfreds Futterkiste	Obere Str. 57, 12209 Berlin, Germany
Ana Trujillo Emparedados y helados	Avda. de la Constitución 2222, 05021 México D.F., Mexico

Syntax

```
SELECT CONCAT(column_1, column_2) AS 'Alias'
FROM table_name;
```

[1] https://www.w3schools.com/sql/func_mysql_concat.asp

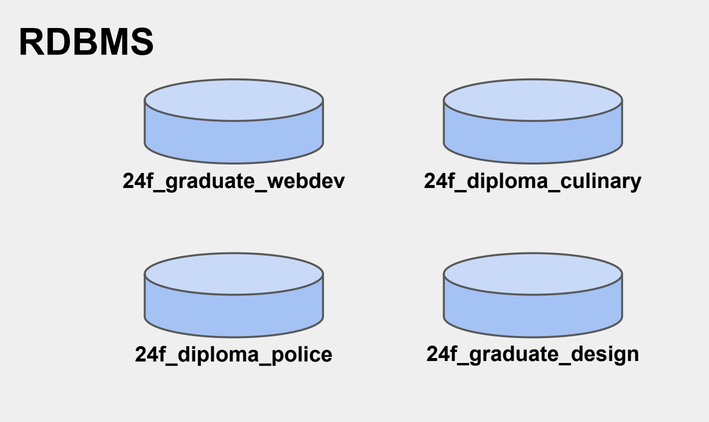
Terminology

- **Database Management System (DBMS):** Software systems used to store, retrieve, and run queries on data.
  - eg. MySQL, Oracle Database, MongoDB, Amazon RDS, PostgreSQL, Apache Cassandra
  - MySQL is specifically a **Relational Database Management System (RDBMS)**
- **phpMyAdmin & Adminer** are **web applications (web apps)**
  - Allow us to interact with a DBMS through a front-end **User Interface (UI)**
- **Structured Query Language (SQL):** A query programming language used to make queries to a database
  - "S-Q-L" or "sequel"
- **SQL Query:** Request made by a user or application to retrieve or manipulate data stored in a database
  - Also **SQL request** or **SQL statement**

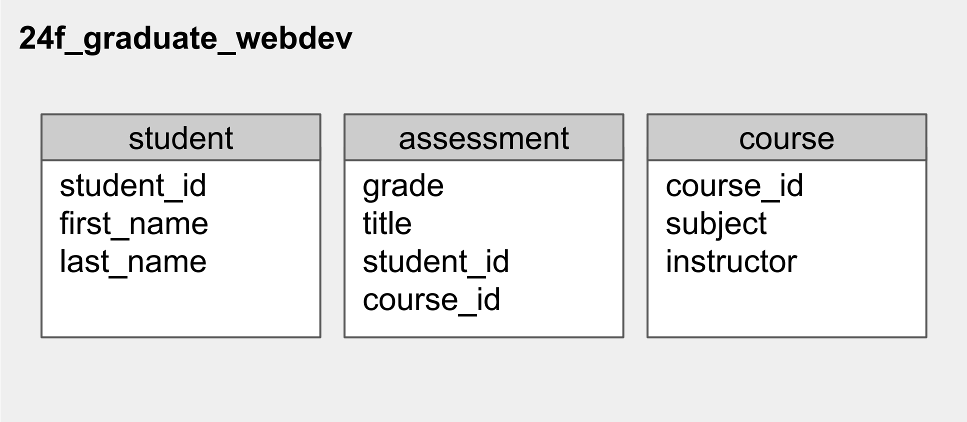
Architecture Diagrams

- These are the starting blocks of architecture diagrams, more to come
- The diagrams display the design of database systems through visual mapping
- We looked at 2 levels of diagrams:

DBMS Architecture



DB Architecture


24f_graduate_webdev

student
student_id first_name last_name

assessment
grade title student_id course_id

course
course_id subject instructor

DB Naming Conventions

- ## General
  - **Full words** not abbreviations/acronyms
  - **No prefixes**
  - **Names** should **reflect real world purpose**
  - **Names** should be **lowercase** since SQL keywords are UPPERCASE
  - **snake_case**: underscore_in_place_of_spaces
- ## Databases
  - **Singular name** that summarizes business use of information held inside
- ## Tables
  - Names should be **nouns**, 1 or 2 words
    - **Noun** - used to **identify** any of **a class of people, places, or things**
  - Table names may be singular OR plural, but be consistent
    - There are pros and cons to each. Follow the convention already in place. If building your own DBs use what makes most sense in your view of tables. I suggest singular.
- ## Columns
  - Names should be 1 or 2 words and singular

SQL Conventions & Keywords

- **SQL Syntax Conventions**
  - **Keywords** should always be **UPPERCASE**
  - **Statements** should be terminated with **semicolons;**

- **Keywords**
  - **SELECT**
  - **DISTINCT**
  - **WHERE**
  - **AS**
  - **CONCAT()**

***Reference links next slide**

Links

- **W3Schools SQL**

  Intro, Syntax, **SELECT**, **DISTINCT**, **WHERE**

- **W3Schools SQL Keywords**

  AS (column alias)

- **W3Schools MySQL Functions**

  CONCAT Function (MySQL)

  *CONCAT_WS Function, similar but different

- **Practice - SQL BOLT**

  SQL Lesson 1: SELECT queries 101