Lab 4: Database Design

GW CS 2541: Database Systems and Team Projects - 2026

Gabe Parmer, Emil Abbasov, Bella Dayrit, Sydney Berritt, Max Eichholz, Aditya Arjun

Credit: Slides adapted from Tim Wood

Announcements

All activities will be submitted through github classroom. Please follow along in lab to make sure you meet all requirements before submitting.

Homework information is on the website as per usual.

Restaurant Database Design

Carmen Berzatto has 35 restaurants. He needs you to create a database to track his restaurants, employees, and menu offerings. This information can be represented with four **entities**:

1. Restaurants
2. Employees
3. Menus
4. Food Items

ER Diagram Entities

Let’s create an ER Diagram to visualize Carmy’s Restaurant Database!

What do we need to store about the **restaurants**?

What do we need to store about **employees**?

What do we need to store about **menus**?

What do we need to store about individual **menu items**?

ER Relationships

We learned on Monday that entities can be related to one another in different tables.

What relations do we need in our database?

What fields affect one another?

What happens to the employees if a restaurant closes?

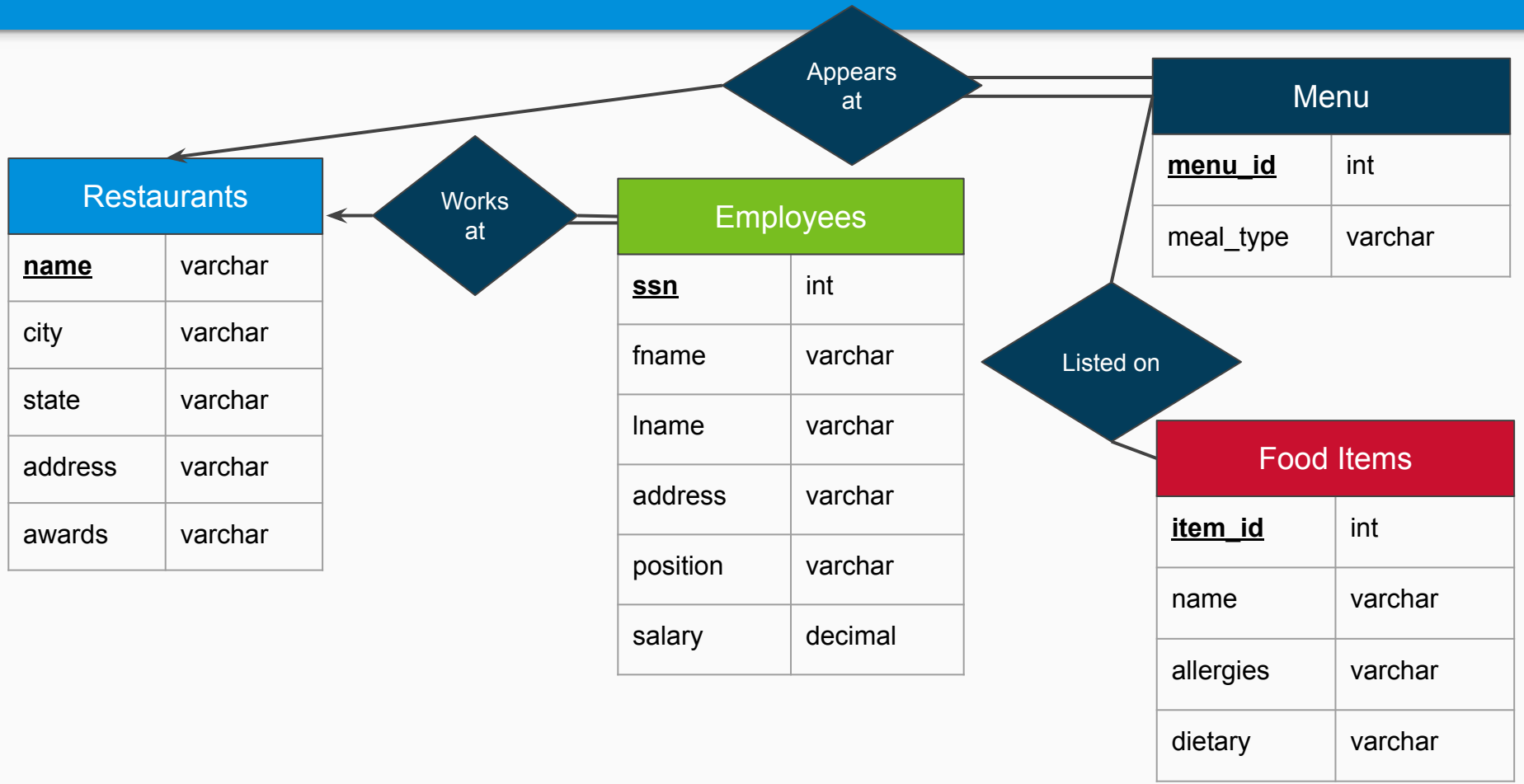
Activity 0: ER Diagram

15 Minutes

1. In groups at your table, create a ER diagram for our restaurant database with the four entities: Restaurants, Employees, Menus, and Food Items
    - Use an online diagram tool or draw it on paper.
    - Make sure to answer the question in the markdown file.
2. When you're finished, upload a picture of your diagram to the *#lab4-diagrams* channel on Discord

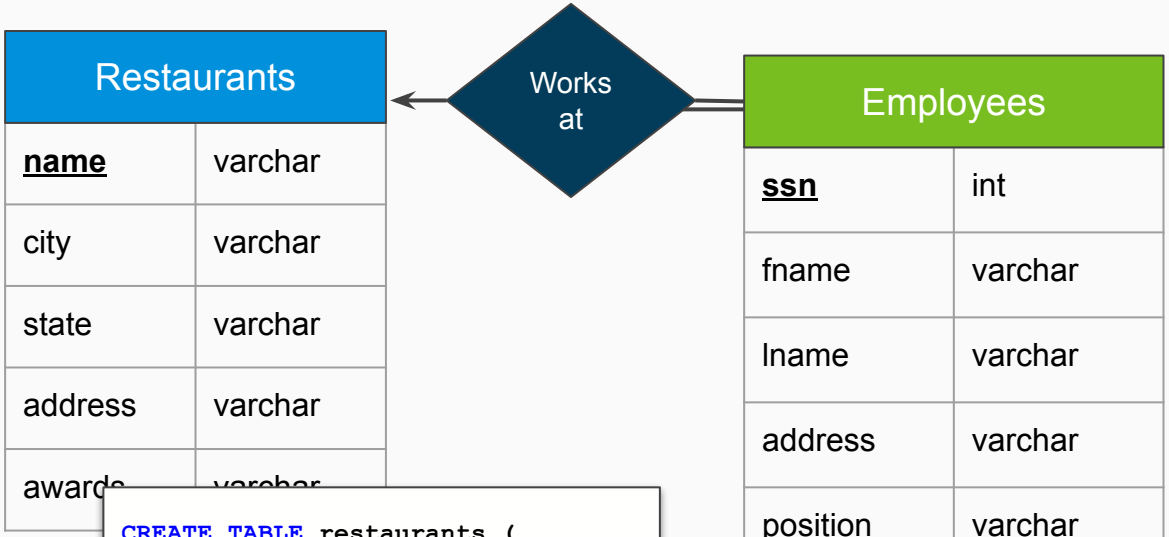
3. React with a ✅ emoji to vote for the diagrams that you think are best!

Restaurants ER Diagram



ER Diagram ⟶ SQL Script

How do we represent the relationship?



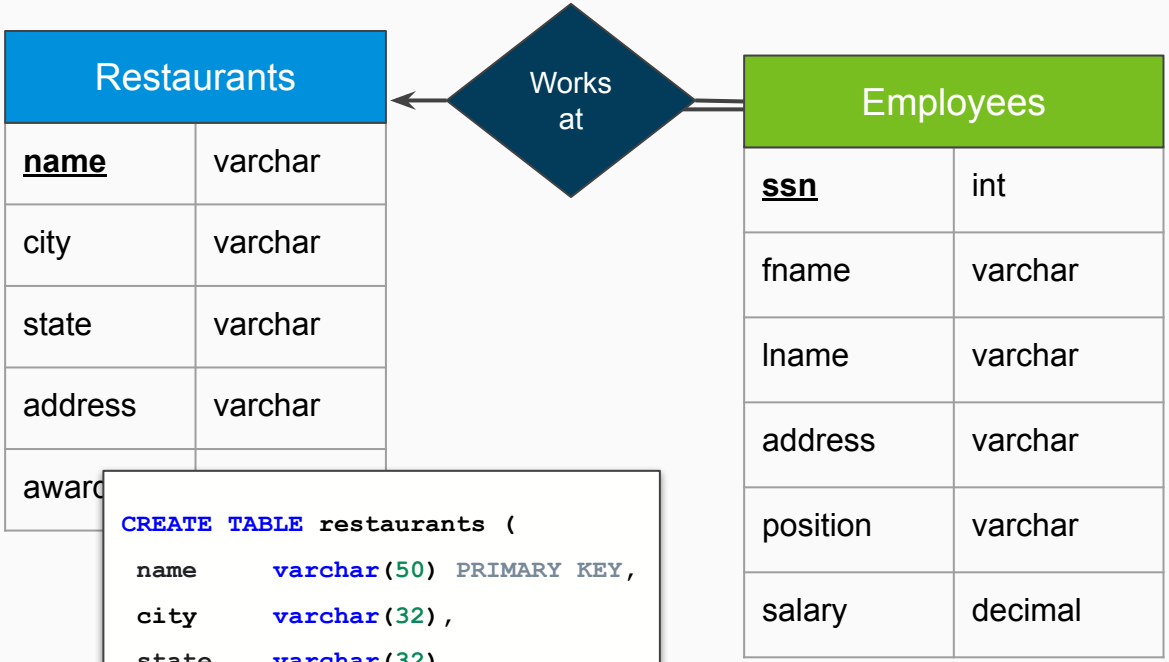

```
CREATE TABLE restaurants (
 name      varchar(50) PRIMARY KEY,
 city      varchar(32),
 state     varchar(32),
 address   varchar(50),
 awards    varchar(128) );
```

```
CREATE TABLE employees (
 ssn          int(9) PRIMARY KEY,
 fname        varchar(15),
 lname        varchar(15),
 address      varchar(50),
 salary       decimal(10,2),
 position     varchar(32)



);
```

ER Diagram ⟶ SQL Script




```
CREATE TABLE restaurants (
 name      varchar(50) PRIMARY KEY,
 city      varchar(32),
 state     varchar(32),
 address   varchar(50),
 awards    varchar(128) );
```

```
CREATE TABLE employees (
 ssn          int(9) PRIMARY KEY,
 fname        varchar(15),
 lname        varchar(15),
 address      varchar(50),
 salary       decimal(10,2),
 position     varchar(32)
 works_at     varchar(50) not null,
 FOREIGN KEY (works_at) REFERENCES
    restaurants(name)
);
```

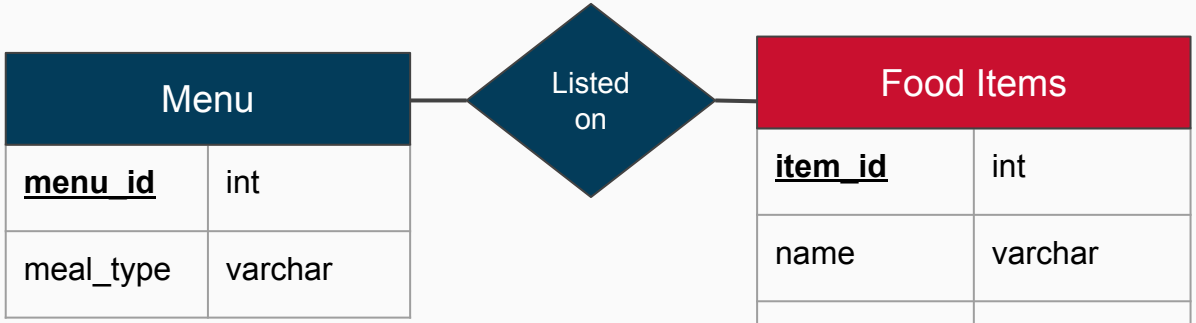
ER Diagram ⟶ SQL Script


```
CREATE TABLE Menu (
 menu_id     int PRIMARY KEY,
 meal_type   varchar(32),
);
```

```
CREATE TABLE FoodItems (
 item_id      int PRIMARY KEY,
 name         varchar(50),
 allergies    varchar(50),
 dietary      varchar(50),
 on_menu      int not null,
 FOREIGN KEY (on_menu) REFERENCES
    Menu(menu_id)
);
```

Does this match the diagram?

ER Diagram ⟶ SQL Script



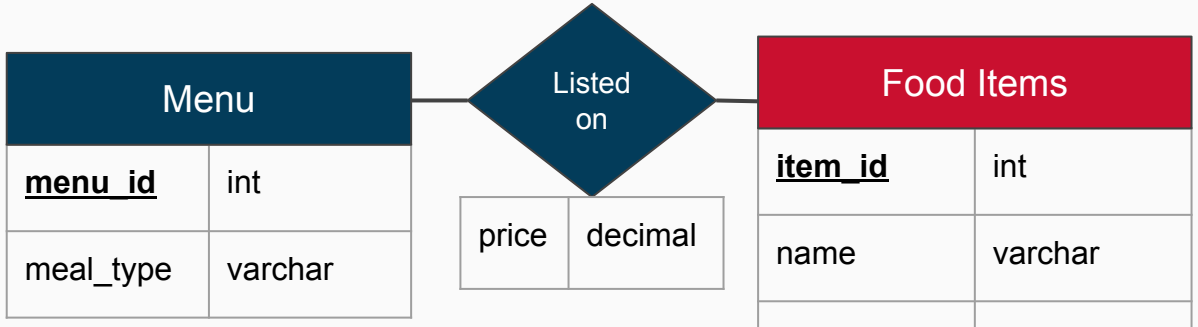

```
CREATE TABLE Menu (
 menu_id     int PRIMARY KEY,
 meal_type   varchar(32),
);
```

```
CREATE TABLE FoodItems (
 item_id      int PRIMARY KEY,
 name         varchar(50),
 allergies    varchar(50),
 dietary      varchar(50),
);
```

```
CREATE TABLE MenuList (
 item_id  int,
 on_menu  int,
 PRIMARY KEY (item_id,
   on_menu),
 FOREIGN KEY (on_menu) REFERENCES
    Menu(menu_id)
 FOREIGN KEY (item_id) REFERENCES
    FoodItems(item_id)
);
```

You may need tables for every Entity AND Relationship!

ER Diagram ⟶ SQL Script




```
CREATE TABLE Menu (
 menu_id     int PRIMARY KEY,
 meal_type   varchar(32),
);
```

```
CREATE TABLE FoodItems (
 item_id      int PRIMARY KEY,
 name         varchar(50),
 allergies    varchar(50),
 dietary      varchar(50),
);
```

```
CREATE TABLE MenuList (
 item_id   int,
 on_menu   int,
 price     decimal,
 PRIMARY KEY (item_id,
   on_menu),
 FOREIGN KEY (on_menu) REFERENCES
    Menu(menu_id)
 FOREIGN KEY (item_id) REFERENCES
    FoodItems(item_id)
);
```

This is more obvious when relations have attributes

Primary and Foreign Key Reminder

Primary Key

- Determines what must be unique in each row
- Can specify inline or at end of declaration

Foreign Key

- Links two tables together
- Allows us to use Joins (correctly)
- Only needs to be specified in the table which is referring to another

```
CREATE TABLE MenuList (
 item_id  int,
 on_menu  int,
 price  decimal,
 PRIMARY KEY (item_id,
   on_menu),
 FOREIGN KEY (on_menu) REFERENCES
    Menu(menu_id)
 FOREIGN KEY (item_id) REFERENCES
    FoodItems(item_id)
);
```

```
CREATE TABLE Menu (
 menu_id    int PRIMARY KEY,
 meal_type  varchar(32),
);
```

Activity 1: Do we really need keys?

10 Minutes

1. Answer all the questions under activity one on the readMe.

Activity 1 Solution

- What happened when you tried to insert redundant data?
- What should we add to these tables to fix them?

```
CREATE TABLE restaurants (
 name      varchar(50) PRIMARY KEY,
 city      varchar(32),
 state     varchar(32),
 address   varchar(50),
 awards    varchar(128)
);
```

```
CREATE TABLE employees (
 ssn          int(9) PRIMARY KEY,
 fname        varchar(15),
 lname        varchar(15),
 address      varchar(50),
 salary       decimal(10,2),
 position     varchar(32)
 works_at     varchar(50) not null,
foreign key (works_at) references
restaurants (name)
 );
```

FK Update/Delete Policies

```
FOREIGN KEY (...)
        REFERENCES (...)
        ON DELETE/UPDATE CASCADE
```

- When a “parent” row is updated or deleted, the update / delete rules for the “children” rows are enforced
- DELETE rules:
  - RESTRICT, NO ACTION → error occurs, no rows are deleted
  - CASCADE → all dependents of the deleted row are also deleted
  - SET NULL → every nullable column of the FK of each dependent of the deleted row are set to null
- UPDATE rules:
  - RESTRICT, NO ACTION → error occurs, no rows updated
  - CASCADE → all dependents of the updated row are also updated
  - SET NULL → every nullable column of the FK of each dependent of the updated row are set to null

Activity 2: Cascade, Update, Delete

10 Minutes

1. Answer the questions in the readMe.