Paula’s Pizza Palace

Nick Herman
Brian Walsh

URL: https://web.engr.oregonstate.edu/~walsbria/cs340/pizzas.html

Actions based on feedback:

1. Add a ‘toppings’ entity to serve as a category table. This entity can track quantities of toppings and can allow for inventory reports and prompt the pizza palace to order more when required.
2. Add a pick-up option. This will be accomplished by creating a dummy ‘drivers’ entry named “pick-up” that can be assigned to an order.
3. Add a boolean attribute to the orders entity to record if an order was cancelled.
4. Change the data type for the order timestamp to ‘datetime’ to improve searchability.

Feedback actions not taken:

1. Status of order. We are choosing not to implement a status attribute where the customer can track the progress of the order. This attribute would need to be updated manually by the employees of the restaurant, who would prefer to focus on cooking the pizza. In addition, any manually updated status will be prone to errors and could lead to customer dissatisfaction were it not updated correctly.

Changes made to draft:

1. ERD updated with ‘toppings’ entity. This change effectively makes ‘pizzas’ an interface table between ‘toppings’ and ‘orderItems.’ What is a pizza if not a specific collection of toppings?
2. Added ‘int’ datatype to the orderItems.quantity attribute.
3. Create an intersection table PizzaToppings between Pizza and Toppings in order to properly normalize. This removes the duplicate ‘topping1, topping2, …’ from the pizza table to get it to 1NF.
4. Add an auto-increment primary key to OrderItems to bring it to 2NF.
5. Add not null constraint to a number of attributes to prevent creating entities without necessary data.

Plan:

Paula’s Pizza Palace needs a database to track their sales, their pizzas, and the specifics of each order. They want to track who ordered the pizza, who delivered it, what kind of pizza they ordered, and the total sale price. Customers should be allowed to add themselves to the database without a purchase to receive updates on sales or upcoming specialty pies. The business started slow but is now churning out dozens of pizzas per day. As a small and local business they only make a profit of around $1000 a week, but it’s enough to pay the bills.

We will need the following entities to enable this:

- Customers: This will track the people who order from the restaurant or who sign up for an account via the website even if they don’t place an order. Customers will have a 1:N relationship with Orders with CustomerID used as a foreign key. This relationship is optional as customers can sign up for an account without needing to order a pizza.
    - customerID, int, auto_increment, unique, not NULL, PK
    - name, varchar, not NULL
    - address, varchar
    - phoneNumber, varchar
    - emailAddress, varchar

- Drivers: The people who deliver the pizzas. It captures some detail about the drivers and the car they use to deliver the pizzas. Drivers will have a 1:N relationship with orders. Each order will have only one driver, but drivers can deliver many orders. ‘DriverID’ will be used as a foreign key in ‘Orders’.
    - driverID, int, auto_increment, unique, not NULL, PK
    - name, varchar, not NULL
    - phoneNumber, varchar, not NULL
    - carModel, varchar, not NULL

    .
- Orders: A record of all the orders. This entity will capture the pizza(s) on the order (multiple are allowed), the customer who ordered it, and the driver who delivered it. This entity has a M:N relationship with pizzas.
    - orderNum, int, auto_increment, unique, not NULL, PK
    - customerID, int, not NULL, FK
    - driverID, int, not NULL, FK
    - price, decimal, not NULL
    - date, varchar, not NULL
    - cancelled, boolean, not NULL, default 0

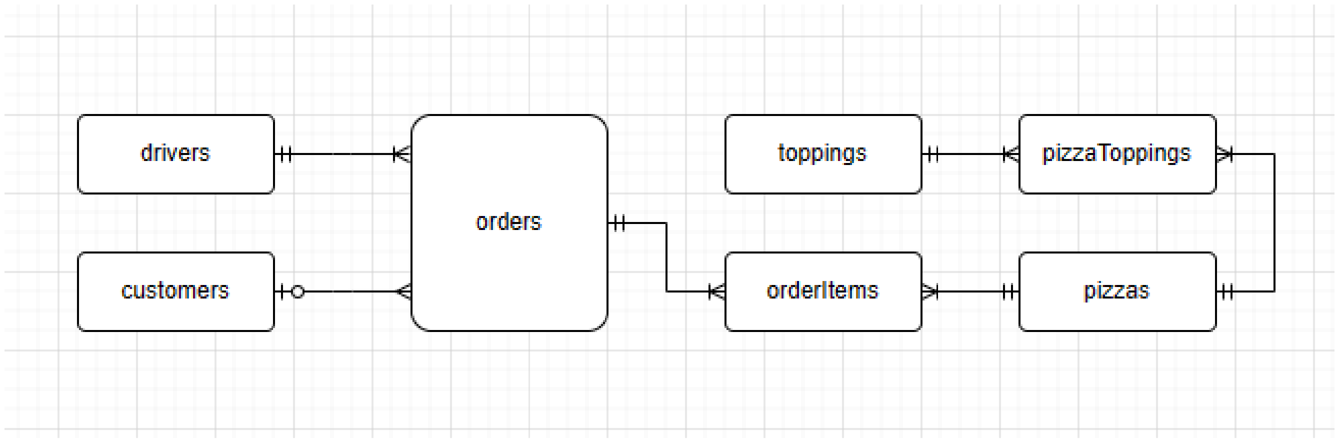
- OrderItems. This entity will hold details of a specific order and acts to resolve M:N relationships between orders, and pizzas. It will use driverID and pizzaID as foreign keys and has a 1:N relationship with both orders and pizzas.
    - orderItemID, int, auto_increment, unique, not NULL, PK
    - orderNum, int, not NULL, FK
    - pizzaID, int, not NULL, FK
    - quantity, int, not NULL

- Pizzas: The pizzas. This entity will capture the name of each pizza, a price, and description, as well as the toppings used to make the pizza. This entity will have a M:N relationship with orders as well as an M:N relationship with toppings.
    - pizzaID, int, auto_increment, unique, not NULL, PK
    - name, varchar, not NULL
    - price, decimal, not NULL
    - description, varchar

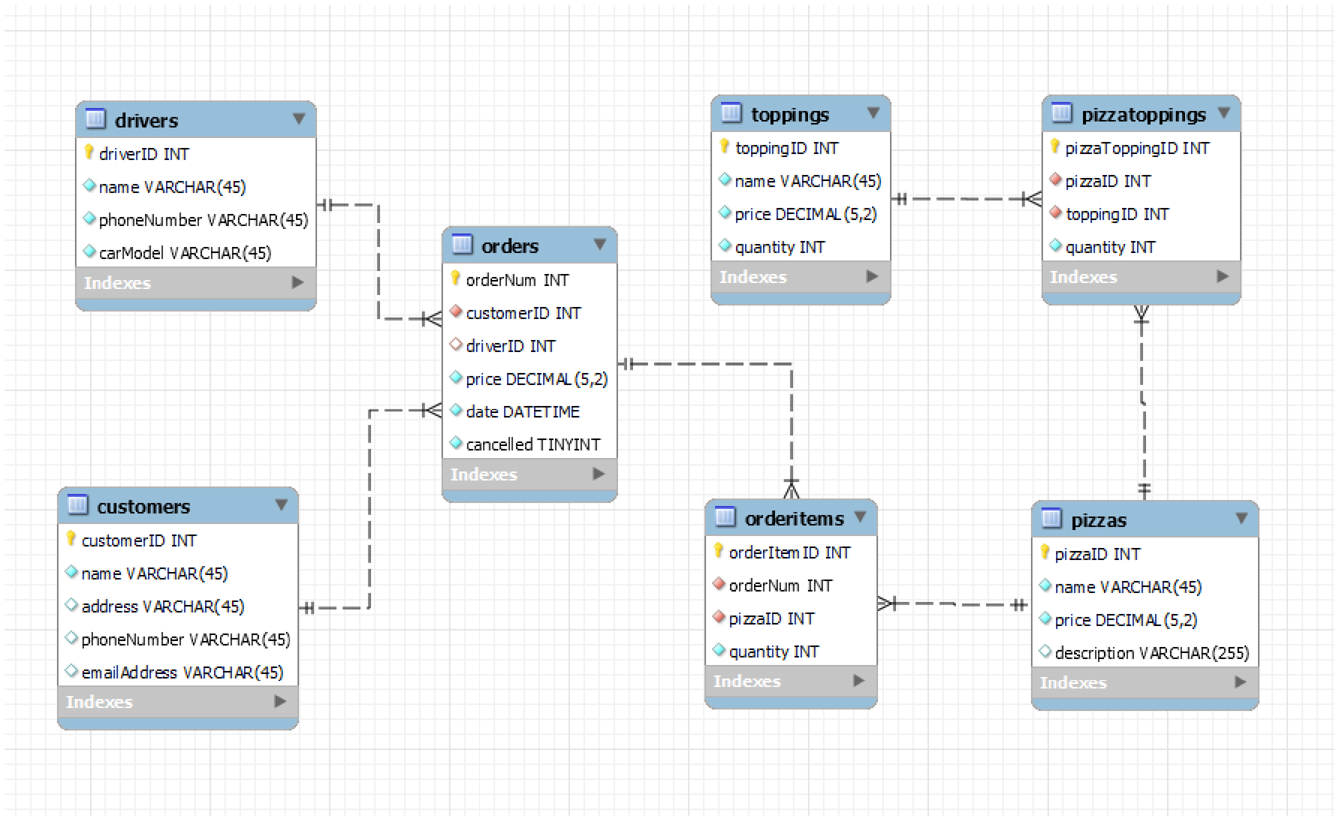
- Toppings: The available toppings. This entity serves as a category table for ‘pizzas’ and will track the inventory of each topping. It has a M:N relationship with ‘Pizzas’.
    - toppingID, int, auto_increment, unique, not NULL, PK
    - name, varchar, not NULL
    - price: decimal, not NULL
    - quantity, int, not NULL

- PizzaToppings: This entity will hold details of which toppings to use for a specific pizza and acts to resolve M:N relationships between toppings, and pizzas. It will use toppingID and pizzaID as foreign keys and has a 1:N relationship with both toppings and pizzas.
    - pizzaToppingID, int, auto_increment, unique, not NULL, PK
    - pizzaID, int, not NULL, FK
    - toppingID, int, not NULL, FK
    - quantity, int, not NULL

ERD:

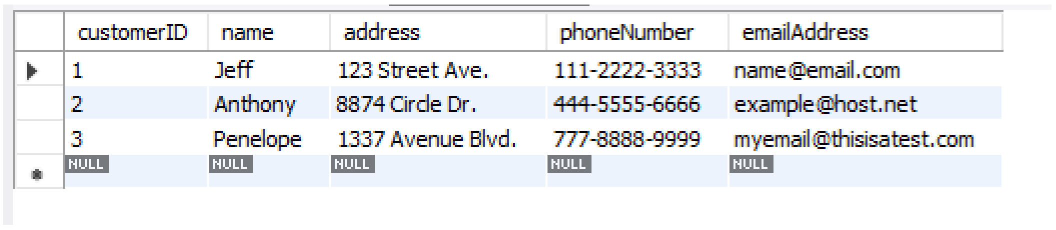

drivers
orders
toppings
pizzaToppings
customers
orderItems
pizzas

Schema:


drivers
driverID INT
name VARCHAR(45)
phoneNumber VARCHAR(45)
carModel VARCHAR(45)
Indexes
orders
orderNum INT
customerID INT
driverID INT
price DECIMAL(5,2)
date DATETIME
cancelled TINYINT
Indexes
toppings
toppingID INT
name VARCHAR(45)
price DECIMAL(5,2)
quantity INT
Indexes
pizzatoppings
pizzaToppingID INT
pizzaID INT
toppingID INT
quantity INT
Indexes
customers
customerID INT
name VARCHAR(45)
address VARCHAR(45)
phoneNumber VARCHAR(45)
emailAddress VARCHAR(45)
Indexes
orderitems
orderItemID INT
orderNum INT
pizzaID INT
quantity INT
Indexes
pizzas
pizzaID INT
name VARCHAR(45)
price DECIMAL(5,2)
description VARCHAR(255)
Indexes

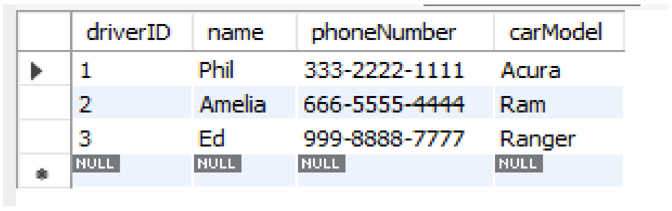
Example Data:

- customers:



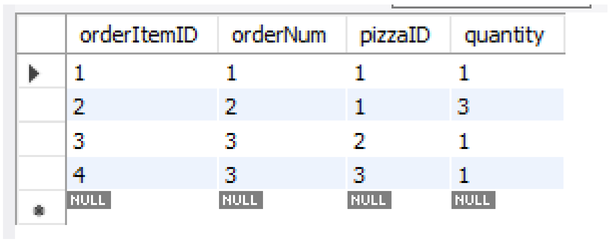
	customerID	name	address	phoneNumber	emailAddress
▶	1	Jeff	123 Street Ave.	111-2222-3333	name@email.com
	2	Anthony	8874 Circle Dr.	444-5555-6666	example@host.net
	3	Penelope	1337 Avenue Blvd.	777-8888-9999	myemail@thisisatest.com
*	NULL	NULL	NULL	NULL	NULL

- drivers:



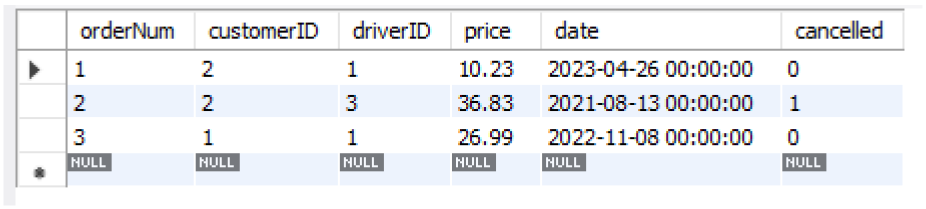
	driverID	name	phoneNumber	carModel
▶	1	Phil	333-2222-1111	Acura
	2	Amelia	666-5555-4444	Ram
	3	Ed	999-8888-7777	Ranger
*	NULL	NULL	NULL	NULL

- orderItems:



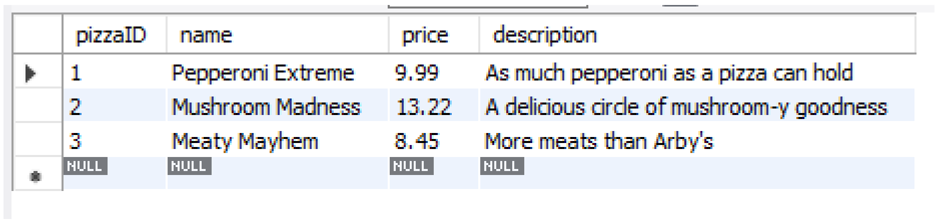
	orderItemID	orderNum	pizzaID	quantity
▶	1	1	1	1
	2	2	1	3
	3	3	2	1
	4	3	3	1
*	NULL	NULL	NULL	NULL

- orders:



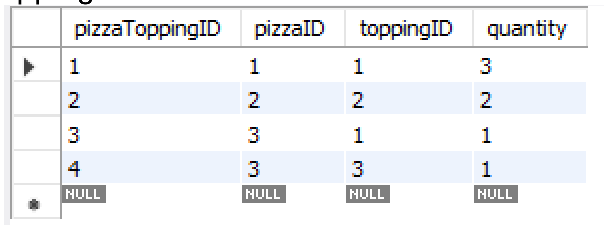
	orderNum	customerID	driverID	price	date	cancelled
▶	1	2	1	10.23	2023-04-26 00:00:00	0
	2	2	3	36.83	2021-08-13 00:00:00	1
	3	1	1	26.99	2022-11-08 00:00:00	0
*	NULL	NULL	NULL	NULL	NULL	NULL

- pizzas:



	pizzaID	name	price	description
▶	1	Pepperoni Extreme	9.99	As much pepperoni as a pizza can hold
	2	Mushroom Madness	13.22	A delicious circle of mushroom-y goodness
	3	Meaty Mayhem	8.45	More meats than Arby's
*	NULL	NULL	NULL	NULL

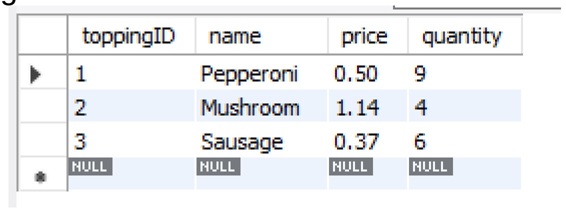
- pizzaToppings:



	pizzaToppingID	pizzaID	toppingID	quantity
▶	1	1	1	3
	2	2	2	2
	3	3	1	1
	4	3	3	1
*	NULL	NULL	NULL	NULL

o

- toppings:



	toppingID	name	price	quantity
▶	1	Pepperoni	0.50	9
	2	Mushroom	1.14	4
	3	Sausage	0.37	6
✱	NULL	NULL	NULL	NULL