**Paula’s Pizza Palace**

Nick Herman

Brian Walsh

**URL:** <http://flip3.engr.oregonstate.edu:42067/>

**Actions based on feedback:**

* 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.
* Change the data type for the order timestamp to ‘datetime’ to improve searchability.
* Added a select last insert ID to Orders and Pizzas inserts that can then be passed to the OrderItems and PizzaToppings insert queries automatically
* Added delete for Pizzas. This will cascade correctly without causing anomalies in PizzaToppings
* Changed update query for Orders to allow driver to be updated. A driver is optional and can be set NULL on creation, but it can now be updated to NULL as well.
* Changed the date field in Orders to automatically add a time stamp instead of needing to enter a date.
* Added an Order Details page to display the M:N OrderItems table. Future plan is to link directly to this page from the Orders page.
* Added ‘int’ datatype to the orderItems.quantity attribute.
* 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.
* Add an auto-increment primary key to OrderItems to bring it to 2NF.
* Add not null constraint to a number of attributes to prevent creating entities without necessary data.
* Changed on cascade property for foreign key toppingID in Pizzas table to restrict to prevent toppings from being deleted if they are used by a pizza.

**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, FK
  + price, decimal, not NULL
  + date, varchar, not NULL
* 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:**

A diagram of a diagram

Description automatically generated

**Schema:**

A screenshot of a computer program

Description automatically generated

**Example Data:**

* customers:

A screenshot of a phone

Description automatically generated

* drivers:

A screenshot of a computer

Description automatically generated

* orderItems:

A screenshot of a computer

Description automatically generated

* orders:

A screenshot of a computer

Description automatically generated

* pizzas:

A screenshot of a computer

Description automatically generated

* pizzaToppings:

A screenshot of a computer

Description automatically generated

* toppings:

A screenshot of a computer

Description automatically generated

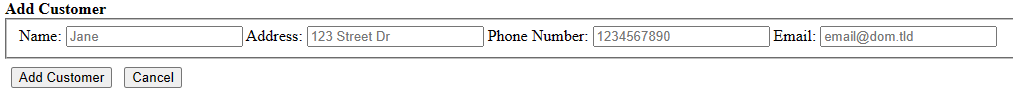
**UI Screen Captures:**

* Read/Delete Customers

A screenshot of a phone number and address

Description automatically generated

* Create Customer:



* Update Customer:

A screenshot of a computer

Description automatically generated

* Read/Delete Drivers

A screenshot of a computer

Description automatically generated

* Create Driver

A screenshot of a computer

Description automatically generated

* Update Driver

A computer screen with a number and numbers

Description automatically generated

* Read/Delete Orders. Deleting an order will also delete associated items in orderItems interface table. **This is an M:N Delete.**

A screenshot of a menu

Description automatically generated

* Create order. Creating an order also creates entries in the orderItems table. **DriverID FK is nullable for pick up orders.**

A screenshot of a pizza delivery box

Description automatically generated

* Read orderItems. If accessed via ‘Details’ button on Orders, this only displays the details of that order, otherwise it displays all order details.

A screenshot of a table

Description automatically generated

* Read/Delete Pizzas

A screenshot of a menu

Description automatically generated

* Create/Update Pizza. Pizzas are created by selecting a topping and a topping quantity. Pizza toppings can be updated once added to the pizza**. This is an M:N Update**.

A screenshot of a computer

Description automatically generated

* Read/Delete toppings:

A screenshot of a computer screen

Description automatically generated

* Create topping:

A screenshot of a computer

Description automatically generated

* Update topping:

A computer screen with a white box

Description automatically generated with medium confidence