Deliverable #5: Environment

Team: Immordlers
Members: Karsten Schmidt, Keisuke Maeda, Ben Findley, Jac Chambers
Project: Order Analysis System

Revisions

Note from the Immordlers: We have changed tracking “profit” to gross revenue. This is because if we were to start considering costs of things and tracking that, then we are diverting from our purpose of popularity and sales analytics to an inventory management system. Gross revenue should provide enough information for the company to assess whether a topping / side, etc. is worth keeping.

Business Problem High Level

Many small mom and pop pizza shops lack the ability to efficiently analyze and track purchases. Most shops use basic point of sales systems which don’t track things such as buying patterns. As such, these shops have a hard time competing with larger chains since they don’t have an objective measure of which menu items are popular, how to optimize pricing, or when peak hours occur. Smaller pizza shops need this automated database system in order to compete with larger pizza chains.

Requirements

- Allow the creation of an order
- Allow an employee to create a pizza, noting toppings.
- Allow a business to view their revenue for any given range
- Maintain the cost of toppings, sauce, and of the dough (by size and type)
- Provide a list of the toppings, sides, sauces, dough sizes, and dough types.
- Provide a list of the most popular toppings, sides, sauces, dough sizes, and dough types all time or in a given date range.

High Level Interface

We will provide APIs to

- Create an order
- Get order details
- Get revenue for a given range
- Evaluate the popularity of a set of toppings within a date range
- Get a list of toppings, sauces, dough sizes, dough types
- Get the price for any given topping, sauce, dough size, or dough type

- Look for most popular toppings within a given range of dates or all time

Credit

Driver UI

- Visual design / structure: Mostly Jac, some by Ben
- Functionality: Mostly Ben, some by Jac

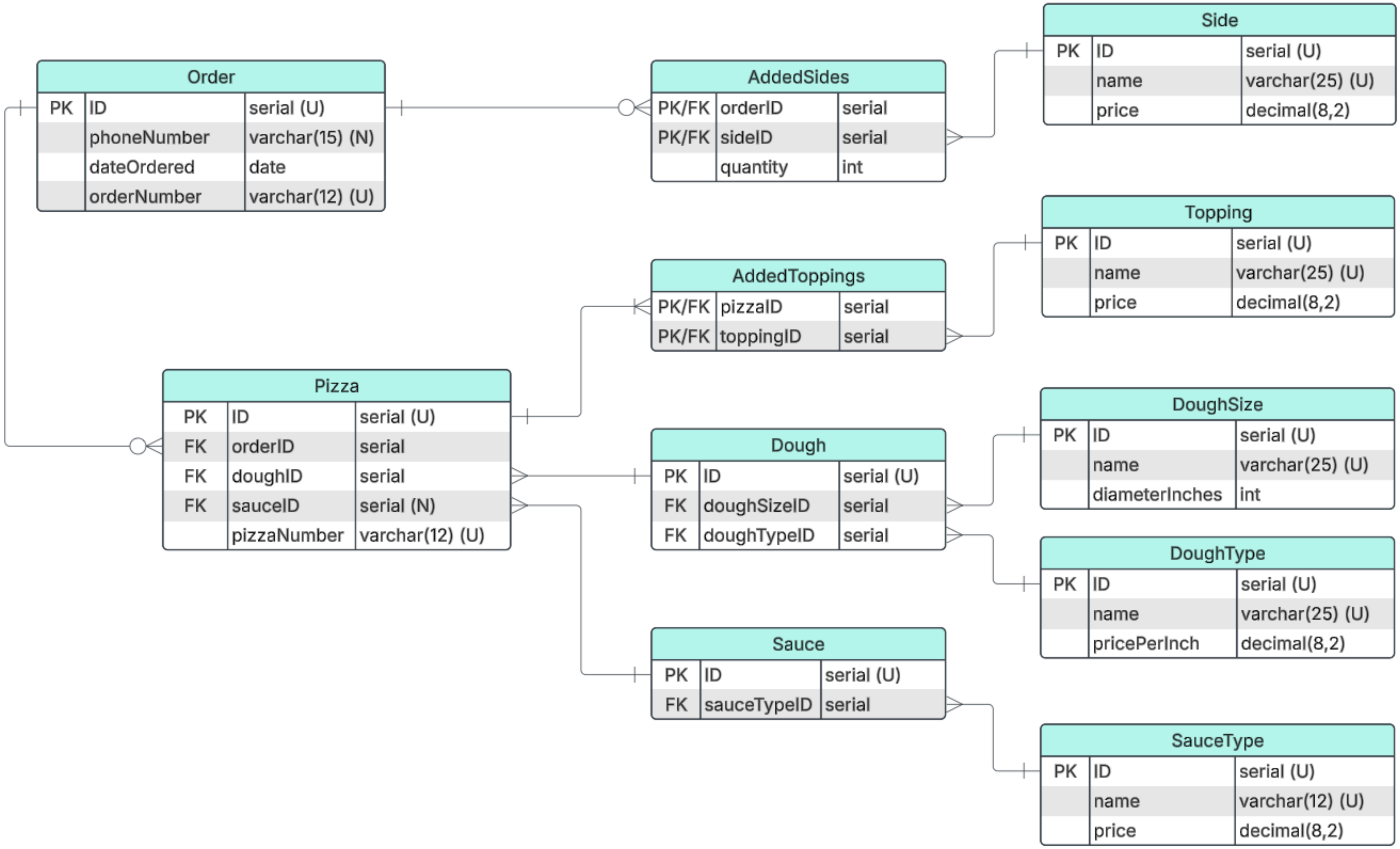
API endpoints

Keep in mind the *description* column here is abridged. For better documentation about how a given API works, look at its JSDoc/doxygen doc in main.ts. Green elements should be graded, while red endpoints are solely supplemental.

Endpoint	Type	Developed By	Description
`create_order`	CRUD multiple	Ben	post a full order and get an order number
`get_order`	Detail	Karsten	get details for an order
`get_popular_side`	List	Jac	number of sales for each different side
`get_popular_toppings`	List	Jac	number of sales for each different topping
`get_popular_dough`	List	(supplemental) Keisuke	number of sales for each different dough
`get_popular_sauce`	List	Keisuke	number of sales for each different sauce
`get_popular_combo`	List	Keisuke	number of sales for each different pair of toppings
`daily_topping_sales`	List	Ben	get sales of a specific topping per day
`get_revenue_in_range`	Complex Query	Karsten	Queries the DB to return the total revenue for a given range

list_orders_made_on	List	(supplemental) Jac	get all ordernums for a particular day
list_available_toppings	List	(supplemental) Karsten	all toppings currently sold / recognized by the DB
list_available_sides	List	(supplemental) Karsten	all sides currently sold / recognized by the DB
list_available_sauces	List	(supplemental) Karsten	all sauces currently sold / recognized by the DB
list_available_dough	List	(supplemental) Karsten	all doughs currently sold / recognized by the DB
list_available_sizes	List	(supplemental) Karsten	all sizes currently sold / recognized by the DB
daily_sauce_sales	List	(supplemental) Ben	get sales of a specific sauce type per day

Schema Design



Order		
PK	ID	serial (U)
	phoneNumber	varchar(15) (N)
	dateOrdered	date
	orderNumber	varchar(12) (U)

AddedSides		
PK/FK	orderID	serial
PK/FK	sideID	serial
	quantity	int

Side		
PK	ID	serial (U)
	name	varchar(25) (U)
	price	decimal(8,2)

Topping		
PK	ID	serial (U)
	name	varchar(25) (U)
	price	decimal(8,2)

AddedToppings		
PK/FK	pizzaID	serial
PK/FK	toppingID	serial

Pizza		
PK	ID	serial (U)
FK	orderID	serial
FK	doughID	serial
FK	sauceID	serial (N)
	pizzaNumber	varchar(12) (U)

DoughSize		
PK	ID	serial (U)
	name	varchar(25) (U)
	diameterInches	int

Dough		
PK	ID	serial (U)
FK	doughSizeID	serial
FK	doughTypeID	serial

DoughType		
PK	ID	serial (U)
	name	varchar(25) (U)
	pricePerInch	decimal(8,2)

Sauce		
PK	ID	serial (U)
FK	sauceTypeID	serial

SauceType		
PK	ID	serial (U)
	name	varchar(12) (U)
	price	decimal(8,2)

Development and Tools

Database Software

We've decided to proceed with using PostgresSQL as our database software of choice. It's highly scalable, free and open source, and provides the best support for complex, analytical queries provided by our API.

Attributes \ DBMS	SQLite	MySQL	SQL Server	PostgresSQL
Scalable	✅	✅	✅	✅
Open Source		✅		✅
JSON Support			✅	✅
Analytical Queries			✅	✅
Group Experience				✅

UI Tools

We're developing the front end of our application using TypeScript. Since our target demographic may have less technological experience than other users, we considered the web to be the most approachable user experience. TypeScript also integrates well with PostgresSQL, which allows us to easily query data from and perform CRUD operations on the database.

Database Hosting

For the purpose of this project, we will be hosting the database on a personal server. In a professional setting - where we sell this product, we would recommend the databases be hosted by individual pizzarias (with our software, schema, etc.), we would recommend using a cloud service such as AWS, or host it on their own personal server. We believe paying for cloud services might be the better option for customers as it delegates technical responsibilities as well as provides better security.

Updated API section

Name	**Input**	**Returns**	**Business Function**

`create_order`	Full order details, then all sides, all pizzas with toppings, sauce and dough, in JSON	An order number	Orders will be tracked with the current time and hold both sides and pizzas within them. This enables the business to see what combinations of sides and pizzas go well together.
`get_order`	Order number	All order details (pizzas, sides) inside of a placedOrder object, displayed on the front end	To see a specific order at a given time with in-depth details.
`get_popular_side`	startDate, endDate, sideName	A list of sides with the number of times that side has been purchased in a given date range.	To see the most popular sides given a date range to offer insight into customer behavior, allowing informed decision.
`get_popular_toppings`	startDate, endDate, doughType	A list of toppings with the number of pizzas purchased with that dough in a given date range.	To see the most popular toppings given a date range to offer insight into customer behavior, allowing informed decision.
`get_popular_dough`	startDate, endDate, doughType	A list of doughs with the number of pizzas purchased with that dough in a given date range.	To see the most popular dough given a date range to offer insight into customer behavior, allowing informed decision.
`get_popular_sauce`	sauceType, quantity	A list of sauces with their quantities in an array of sauce data structures inside saucetype given date range.	To see the most popular sauce given a given range to offer insight into customer behavior, allowing informed decision.
`get_popular_combo`	startDate, endDate, limit	List of names of most popular (appearing the most) sets of toppings within the range, to the limit, in descending order given date range.	Allows the business to see what the most popular sets of toppings are at a time to determine what to keep and what to drop

`list_available_sides`	None	Returns an array of all the side names found in the DB (as sides object)	Allows the business to track all of the sides which they currently offer and evaluate.
`list_available_toppings`	None	Returns an array of all the topping names found in the DB (as topping object)	Allows the business to track all of the toppings which they currently offer and evaluate.
`list_available_sauces`	None	Returns an array of all the sauce names found in the DB (as sauce object)	Allows the business to track all of the sauces which they currently offer and evaluate.
`list_available_dough`	None	Returns an array of all the dough names found in the DB (as doughType object)	Allows the business to track all of the dough types which they currently offer and evaluate.
`list_available_sizes`	None	Returns an array of all the dough sizes found in the DB (as doughSize object)	Allows the business to track all of the dough sizes which they currently offer and evaluate.
`daily_topping_sales`	Type, quantity	Side data structure	Create a given side to be added to an order to be tracked later.
`list_orders_made_on`	Start range, end range, limit	Output the order numbers of all orders on a single day.	Allows the business to track the number of orders on a given date, by outputting the order numbers on the given day. Can also be used in tandem with the get_order() API to see the order details.
`get_revenue_in_range`	Start date, end date	Numerical value of total revenue within that range	Gives the business an idea of what their revenue was for pizzas during these ranges (adjusted to modern prices, to help them to predict / set expectations, or evaluate prices if sales are down).