Ricky's Pizzeria

A Database Design Project

By Morgan Baker

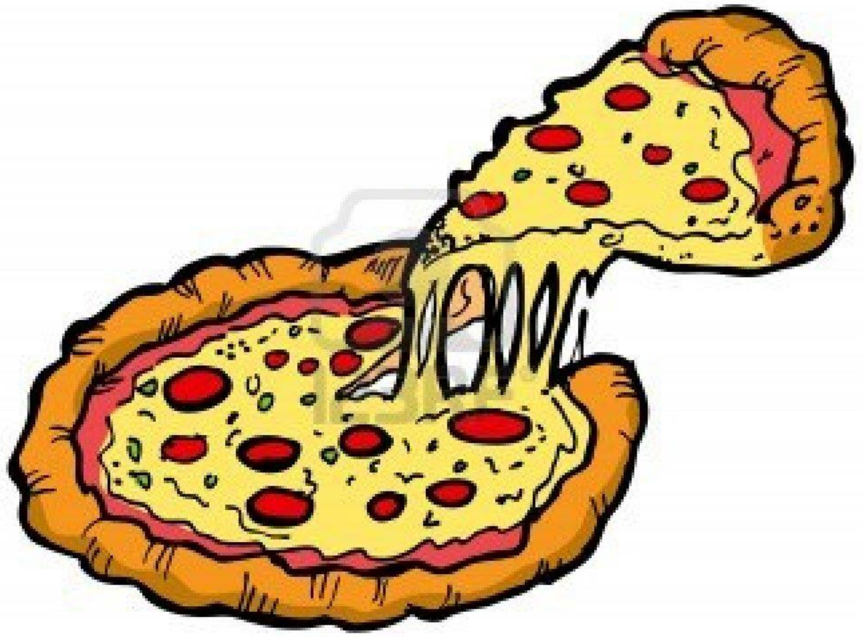


Table of Contents

Executive Summary

Ricky’s Pizzeria is a small pizza shop that is popular with the citizens of Dataville. Their pizzas are of the utmost quality, but until now, most of their information tracking had been on paper. Wanting to move into the world of digital information, Ricky’s Pizzeria needed a database that could hold customer and employee data, as well as data on the different types of pizzas available for consumption. The data needs to be accurate because without accuracy, the pizzeria would not be able to serve the citizens of Dataville with the same quality of customer service as before.

ER Diagram

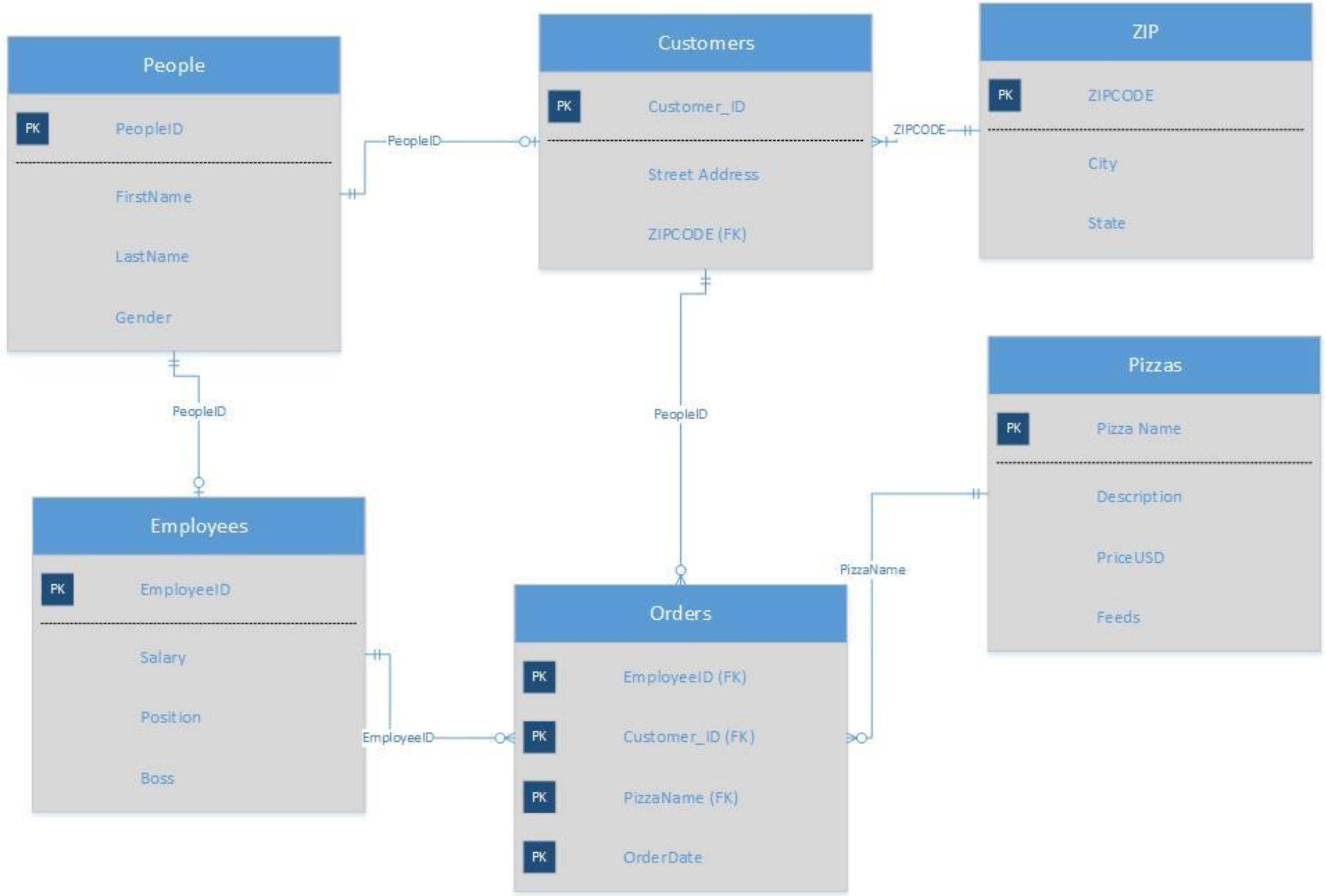

People
PK PeopleID
FirstName
LastName
Gender
Customers
PK Customer_ID
Street Address
ZIPCODE (FK)
ZIP
PK ZIPCODE
City
State
PeopleID
ZIPCODE
PeopleID
PeopleID
Employees
PK EmployeeID
Salary
Position
Boss
Pizzas
PK Pizza Name
Description
PriceUSD
Feeds
PizzaName
Orders
PK EmployeeID (FK)
PK Customer_ID (FK)
PK PizzaName (FK)
PK OrderDate
EmployeeID

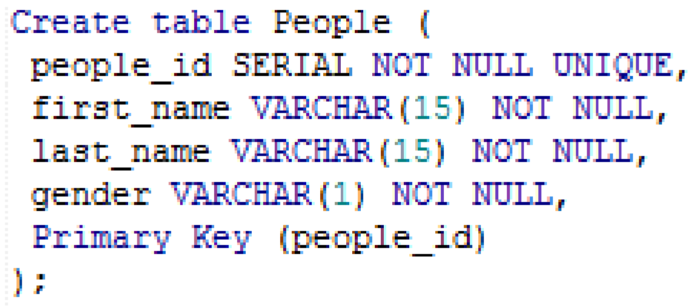
Table Creates (People)

All people, whether customers, employees, or both, are contained in this table.

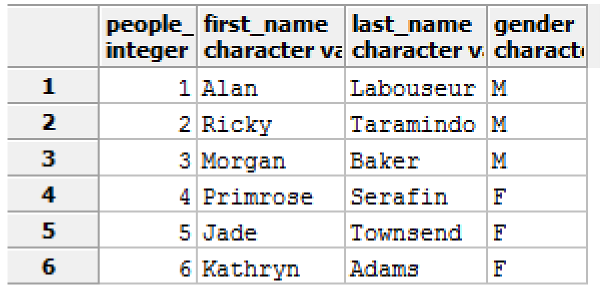
Functional Depedencies

people_ID → first_name, last_name, gender

Sample Data



```
Create table People (
 people_id SERIAL NOT NULL UNIQUE,
 first_name VARCHAR(15) NOT NULL,
 last_name VARCHAR(15) NOT NULL,
 gender VARCHAR(1) NOT NULL,
 Primary Key (people_id)
);
```



	people_ integer	first_name character va	last_name character v	gender characte
1	1	Alan	Labouseur	M
2	2	Ricky	Taramindo	M
3	3	Morgan	Baker	M
4	4	Primrose	Serafin	F
5	5	Jade	Townsend	F
6	6	Kathryn	Adams	F

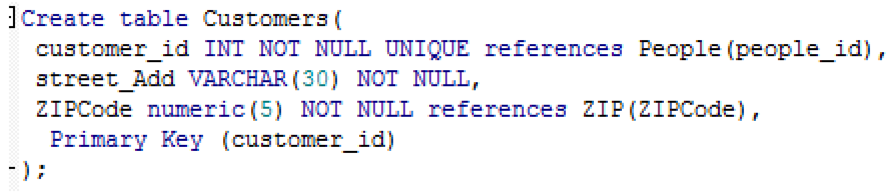
Table Creates (Customers)

The people of Dataville have shared certain data with Ricky's Pizzeria. These people are Customers, and are a subset of the People table.

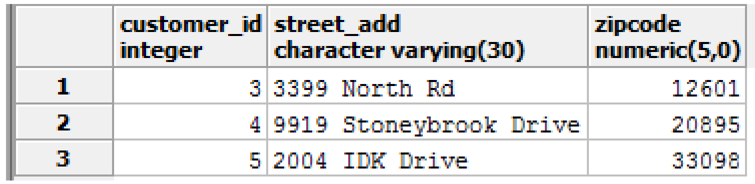
Functional Dependencies

customer_id → Street Address ZIPCODE(references ZIP table)

Sample Data



```
Create table Customers(
  customer_id INT NOT NULL UNIQUE references People(people_id),
  street_Add VARCHAR(30) NOT NULL,
  ZIPCode numeric(5) NOT NULL references ZIP(ZIPCode),
   Primary Key (customer_id)
);
```



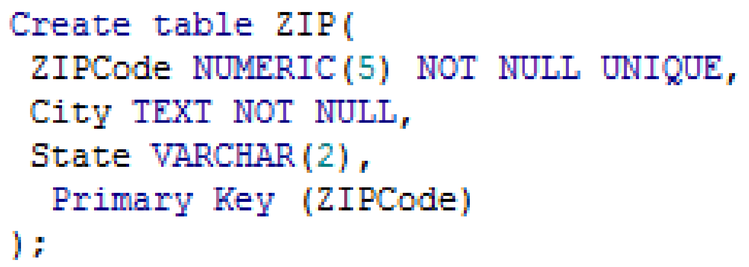
	customer_id integer	street_add character varying(30)	zipcode numeric(5,0)
1	3	3399 North Rd	12601
2	4	9919 Stoneybrook Drive	20895
3	5	2004 IDK Drive	33098

Table Creates (ZIP)

In order to keep the database organized, the zip codes for customers are included in a separate table, ZIP.

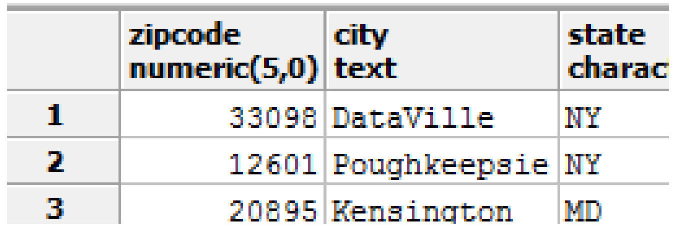
Functional Dependencies

ZIPCode → City, State



```
Create table ZIP(
 ZIPCode NUMERIC(5) NOT NULL UNIQUE,
 City TEXT NOT NULL,
 State VARCHAR(2),
  Primary Key (ZIPCode)
);
```

Sample Data



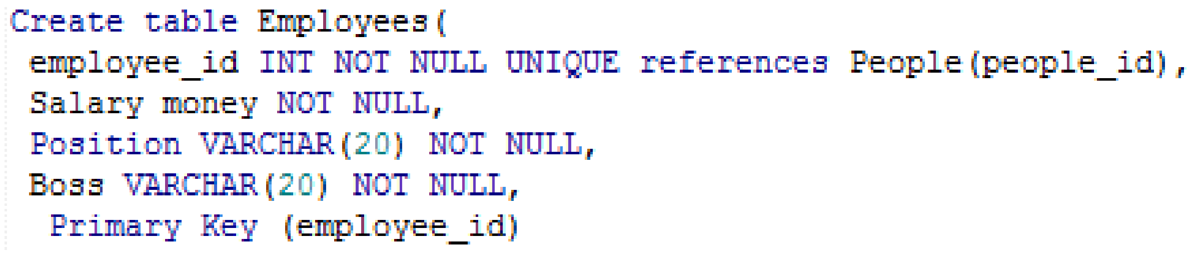
	zipcode numeric(5,0)	city text	state charac
1	33098	DataVille	NY
2	12601	Poughkeepsie	NY
3	20895	Kensington	MD

Table Creates (Employees)

Another subset of people, Employees know Ricky on some level, because Ricky has employed them at the establishment.

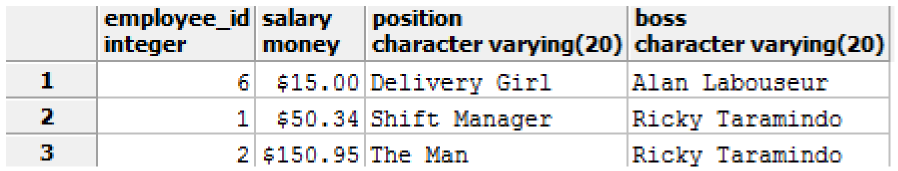
Functional Dependencies

Employee_ID → Salary, Position, Boss



```
Create table Employees(
 employee_id INT NOT NULL UNIQUE references People(people_id),
 Salary money NOT NULL,
 Position VARCHAR(20) NOT NULL,
 Boss VARCHAR(20) NOT NULL,
  Primary Key (employee_id)
```

Sample Data



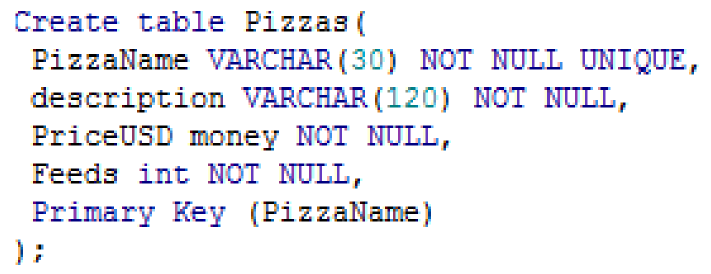
	employee_id integer	salary money	position character varying(20)	boss character varying(20)
1	6	$15.00	Delivery Girl	Alan Labouseur
2	1	$50.34	Shift Manager	Ricky Taramindo
3	2	$150.95	The Man	Ricky Taramindo

Table Creates (Pizzas)

The one thing that Ricky’s shop makes is why everyone in Dataville knows the name. The Pizza table collects data on all the great pies made in Ricky’s establishment.

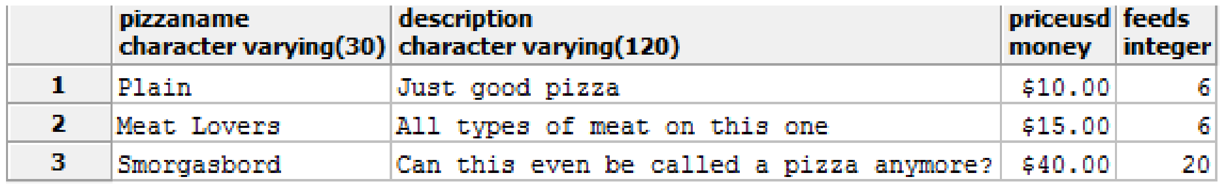
Functional Dependencies

PizzaName → Description, PriceUSD, Feeds



```
Create table Pizzas(
 PizzaName VARCHAR(30) NOT NULL UNIQUE,
 description VARCHAR(120) NOT NULL,
 PriceUSD money NOT NULL,
 Feeds int NOT NULL,
 Primary Key (PizzaName)
);
```

Sample Data



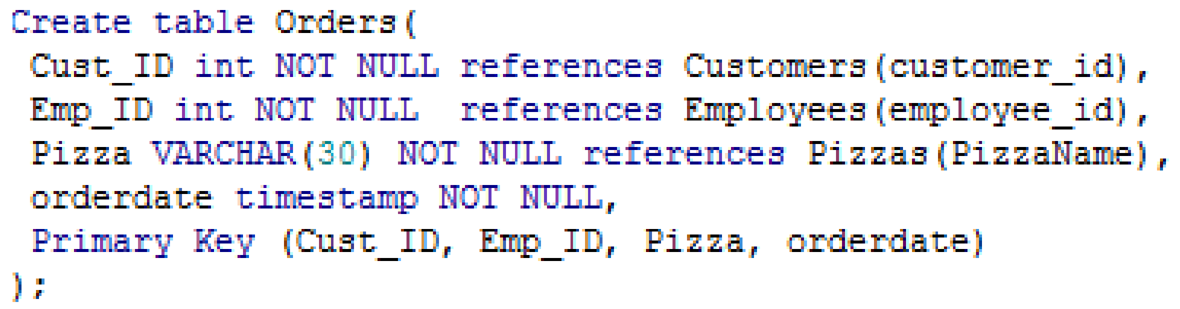
	pizzaname character varying(30)	description character varying(120)	priceusd money	feeds integer
1	Plain	Just good pizza	$10.00	6
2	Meat Lovers	All types of meat on this one	$15.00	6
3	Smorgasbord	Can this even be called a pizza anymore?	$40.00	20

Table Creates (Orders)

The database stores the Customer ID, Employee ID, Pizza Name and the Order Date.

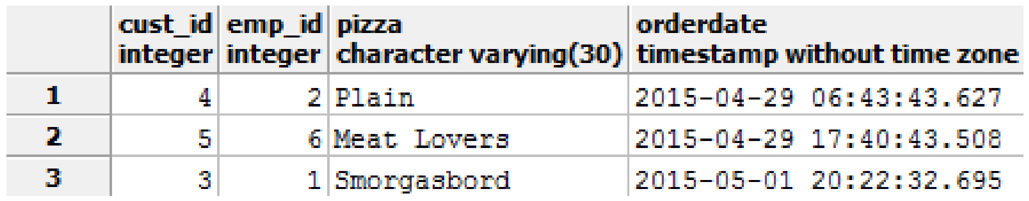
Functional Dependencies

CustomerID, EmployeeID, PizzaName, OrderDate →



```
Create table Orders(
 Cust_ID int NOT NULL references Customers(customer_id),
 Emp_ID int NOT NULL  references Employees(employee_id),
 Pizza VARCHAR(30) NOT NULL references Pizzas(PizzaName),
 orderdate timestamp NOT NULL,
 Primary Key (Cust_ID, Emp_ID, Pizza, orderdate)
);
```

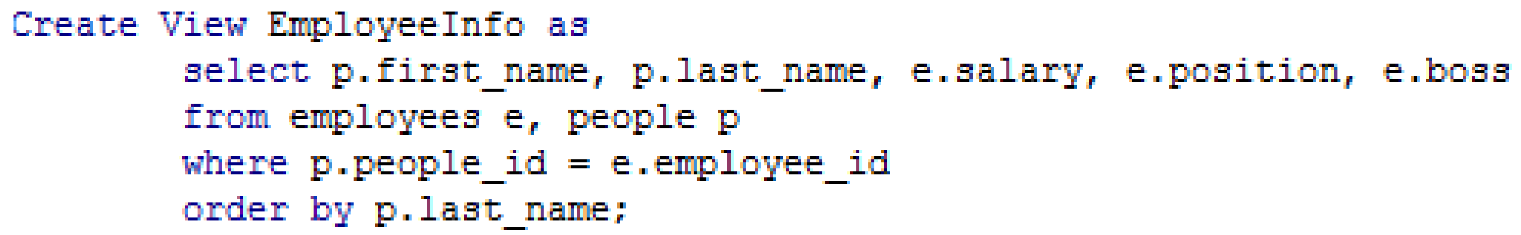
Sample Data



	cust_id integer	emp_id integer	pizza character varying(30)	orderdate timestamp without time zone
1	4	2	Plain	2015-04-29 06:43:43.627
2	5	6	Meat Lovers	2015-04-29 17:40:43.508
3	3	1	Smorgasbord	2015-05-01 20:22:32.695

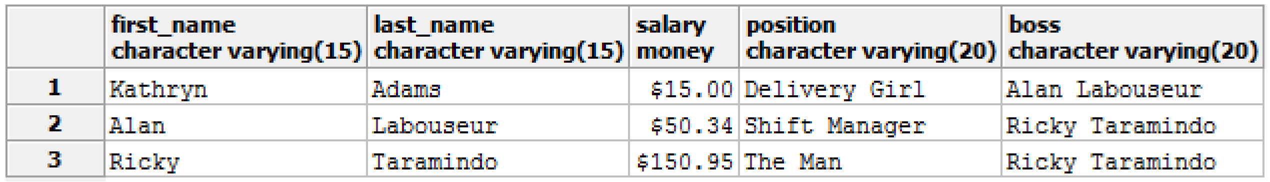
Views (EmployeeInfo)

The first view involves bringing all data relating to employees together in one view. This view is also good for seeing who’s who in the chain of command.



```
Create View EmployeeInfo as
        select p.first_name, p.last_name, e.salary, e.position, e.boss
        from employees e, people p
        where p.people_id = e.employee_id
        order by p.last_name;
```

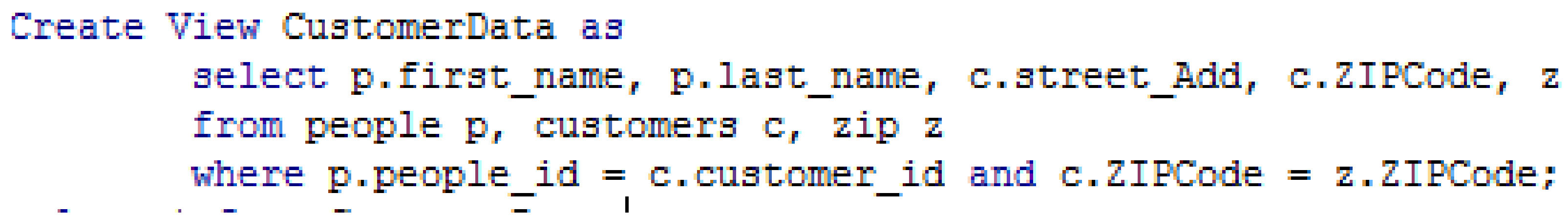
Sample Output



	first_name character varying(15)	last_name character varying(15)	salary money	position character varying(20)	boss character varying(20)
1	Kathryn	Adams	$15.00	Delivery Girl	Alan Labouseur
2	Alan	Labouseur	$50.34	Shift Manager	Ricky Taramindo
3	Ricky	Taramindo	$150.95	The Man	Ricky Taramindo

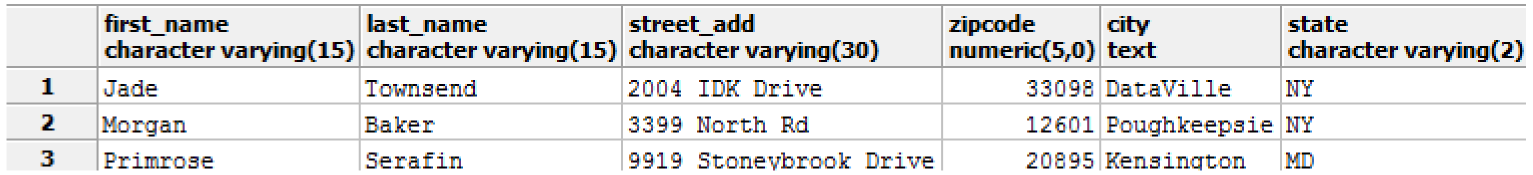
View (CustomerData)

This view combines all of the data regarding the Pizzeria’s customers and put that data into one window of view. This is also good for checking where multiple deliveries can be made quickly, because of the locations also being displayed.



```
Create View CustomerData as
        select p.first_name, p.last_name, c.street_Add, c.ZIPCode, z.city, z.state
        from people p, customers c, zip z
        where p.people_id = c.customer_id and c.ZIPCode = z.ZIPCode;
```

Sample Output

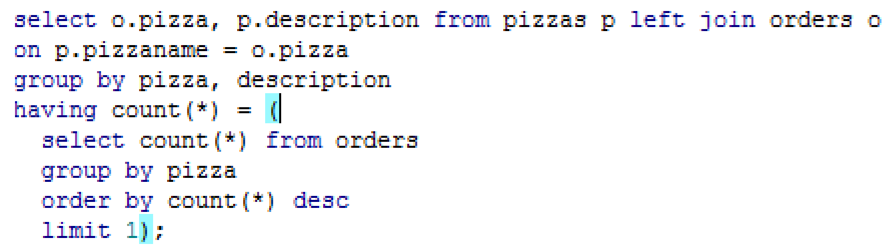


	first_name	last_name	street_add	zipcode	city	state
	character varying(15)	character varying(15)	character varying(30)	numeric(5,0)	text	character varying(2)
1	Jade	Townsend	2004 IDK Drive	33098	DataVille	NY
2	Morgan	Baker	3399 North Rd	12601	Poughkeepsie	NY
3	Primrose	Serafin	9919 Stoneybrook Drive	20895	Kensington	MD

Query 1

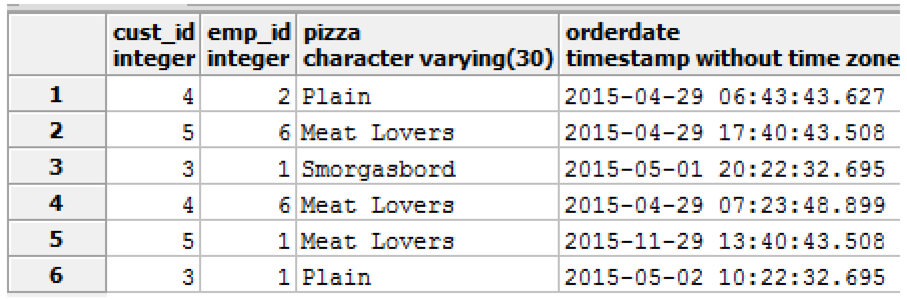
This is a query to determine the most popular pizza at Ricky's.

Query



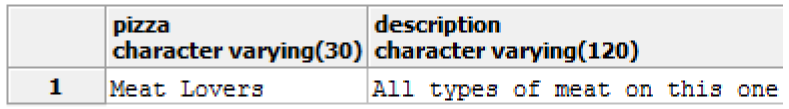
```
select o.pizza, p.description from pizzas p left join orders o
on p.pizzaname = o.pizza
group by pizza, description
having count(*) = (
  select count(*) from orders
  group by pizza
  order by count(*) desc
  limit 1);
```

Orders Table



	cust_id integer	emp_id integer	pizza character varying(30)	orderdate timestamp without time zone
1	4	2	Plain	2015-04-29 06:43:43.627
2	5	6	Meat Lovers	2015-04-29 17:40:43.508
3	3	1	Smorgasbord	2015-05-01 20:22:32.695
4	4	6	Meat Lovers	2015-04-29 07:23:48.899
5	5	1	Meat Lovers	2015-11-29 13:40:43.508
6	3	1	Plain	2015-05-02 10:22:32.695

Output from Query

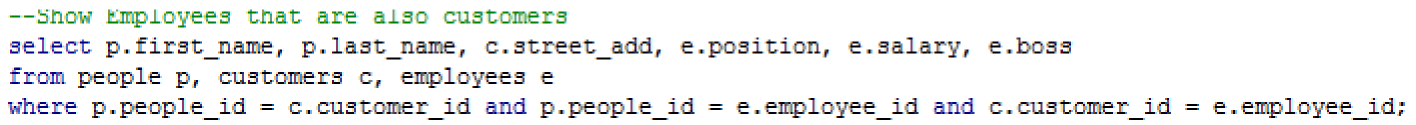


	pizza character varying(30)	description character varying(120)
1	Meat Lovers	All types of meat on this one

Query 2

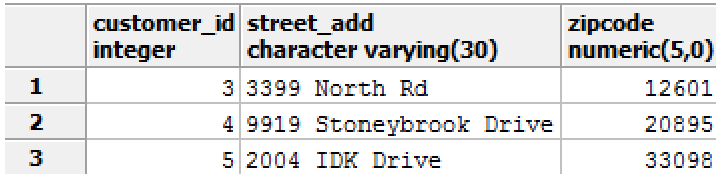
This second query shows which people are both customers and employees, along with specifics about both.

Query



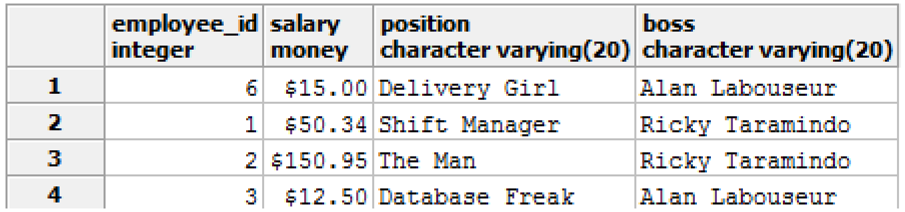
```
--Show Employees that are also customers
select p.first_name, p.last_name, c.street_add, e.position, e.salary, e.boss
from people p, customers c, employees e
where p.people_id = c.customer_id and p.people_id = e.employee_id and c.customer_id = e.employee_id;
```

Customers



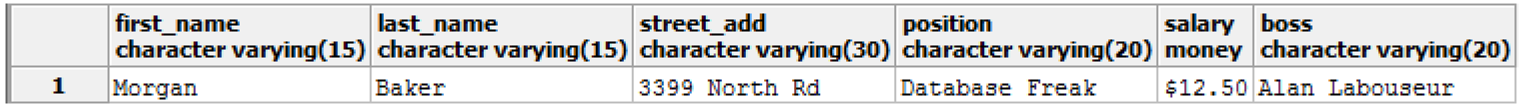
	customer_id integer	street_add character varying(30)	zipcode numeric(5,0)
1	3	3399 North Rd	12601
2	4	9919 Stoneybrook Drive	20895
3	5	2004 IDK Drive	33098

Employees



	employee_id integer	salary money	position character varying(20)	boss character varying(20)
1	6	$15.00	Delivery Girl	Alan Labouseur
2	1	$50.34	Shift Manager	Ricky Taramindo
3	2	$150.95	The Man	Ricky Taramindo
4	3	$12.50	Database Freak	Alan Labouseur

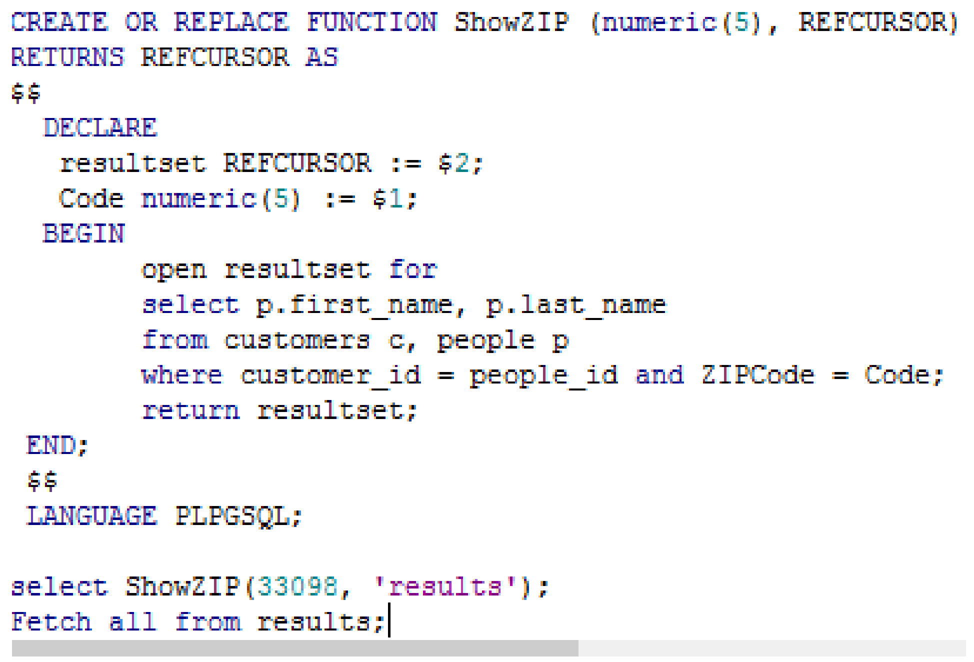
Output of the Query



	first_name character varying(15)	last_name character varying(15)	street_add character varying(30)	position character varying(20)	salary money	boss character varying(20)
1	Morgan	Baker	3399 North Rd	Database Freak	$12.50	Alan Labouseur

Stored Procedure 1

This stored procedure shows all of the customers within a certain zip code.



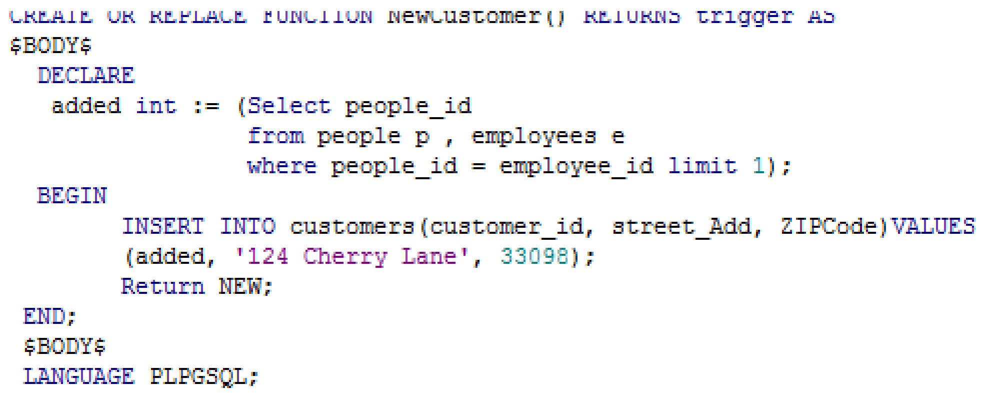
```
CREATE OR REPLACE FUNCTION ShowZIP (numeric(5), REFCURSOR)
RETURNS REFCURSOR AS
$$
  DECLARE
   resultset REFCURSOR := $2;
   Code numeric(5)  := $1;
  BEGIN
        open resultset for
        select p.first_name, p.last_name
        from customers c, people p
        where customer_id = people_id and ZIPCode = Code;
        return resultset;
 END;
 $$
 LANGUAGE PLPGSQL;

select ShowZIP(33098, 'results');
Fetch all from results;
```

Sample Output

Stored Procedure 2

This stored procedure is triggered when a new employee is added. The employee is automatically add in as a customer, with the street address and zip being their place of work.

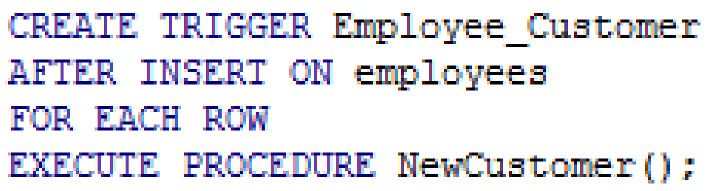


```
CREATE OR REPLACE FUNCTION NewCustomer() RETURNS trigger AS
$BODY$
  DECLARE
   added int := (Select people_id
                 from people p , employees e
                 where people_id = employee_id limit 1);
  BEGIN
        INSERT INTO customers(customer_id, street_Add, ZIPCode)VALUES
        (added, '124 Cherry Lane', 33098);
        Return NEW;
 END;
 $BODY$
 LANGUAGE PLPGSQL;
```

Output is on the slide with the trigger

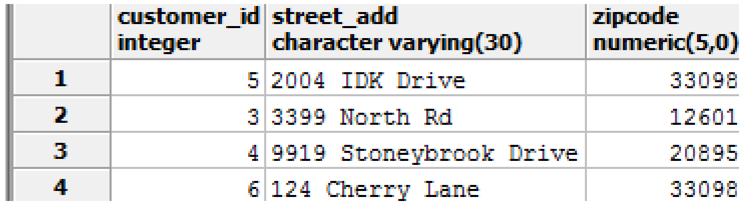
Trigger

The trigger is what sets off the second Stored Procedure, rather than calling it maually. This also has the before and after customer tables to see the difference.



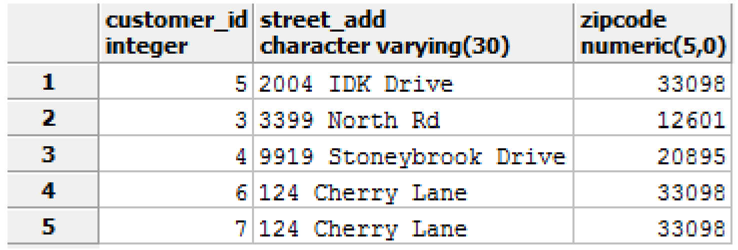
```
CREATE TRIGGER Employee_Customer
AFTER INSERT ON employees
FOR EACH ROW
EXECUTE PROCEDURE NewCustomer();
```

Before



	customer_id integer	street_add character varying(30)	zipcode numeric(5,0)
1	5	2004 IDK Drive	33098
2	3	3399 North Rd	12601
3	4	9919 Stoneybrook Drive	20895
4	6	124 Cherry Lane	33098

After



	customer_id integer	street_add character varying(30)	zipcode numeric(5,0)
1	5	2004 IDK Drive	33098
2	3	3399 North Rd	12601
3	4	9919 Stoneybrook Drive	20895
4	6	124 Cherry Lane	33098
5	7	124 Cherry Lane	33098

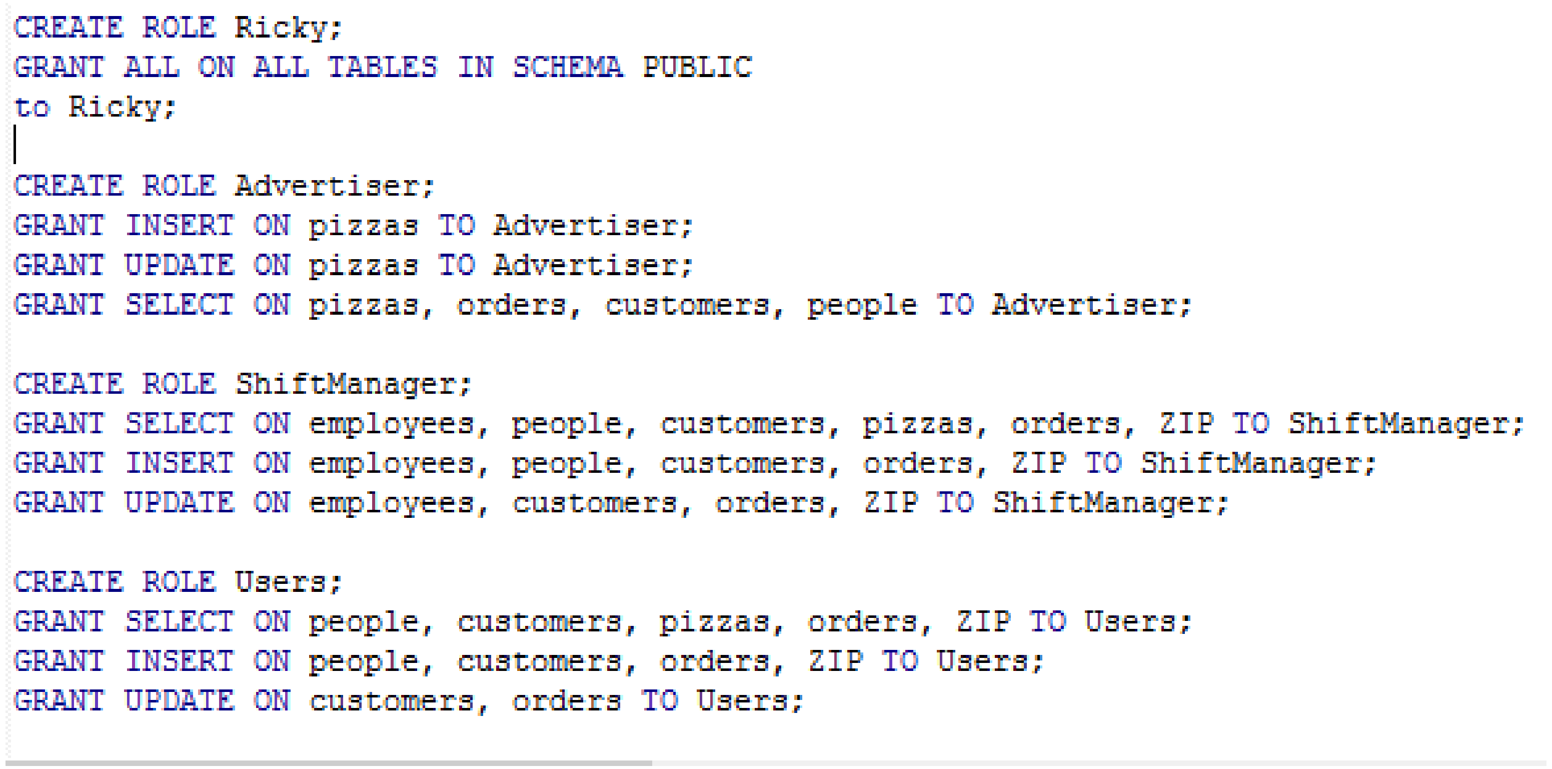
Security Roles

Ricky

Advertiser

Shift Manager

User



```
CREATE ROLE Ricky;
GRANT ALL ON ALL TABLES IN SCHEMA PUBLIC
to Ricky;

CREATE ROLE Advertiser;
GRANT INSERT ON pizzas TO Advertiser;
GRANT UPDATE ON pizzas TO Advertiser;
GRANT SELECT ON pizzas, orders, customers, people TO Advertiser;

CREATE ROLE ShiftManager;
GRANT SELECT ON employees, people, customers, pizzas, orders, ZIP TO ShiftManager;
GRANT INSERT ON employees, people, customers, orders, ZIP TO ShiftManager;
GRANT UPDATE ON employees, customers, orders, ZIP TO ShiftManager;

CREATE ROLE Users;
GRANT SELECT ON people, customers, pizzas, orders, ZIP TO Users;
GRANT INSERT ON people, customers, orders, ZIP TO Users;
GRANT UPDATE ON customers, orders TO Users;
```

Notes / Issues

There were a lot of assumptions made in this project. The first was that Ricky didn’t have a lot of business, and therefore the sample data was small. The database doesn’t account for anything ordered other than pizzas, nor can the orders table handle more than one pizza at a time. The composition of the database could also use improvement. In the future, more interesting queries would result, as the data sample would expand more and more. There could also be options for other things in an order other than one pizza, like cheesy bread or soda. Either way, I’m still proud of this database, and it can preform the functions required of Ricky’s Pizzeria…… for now.