IS 380 Database Management Project
Final Report

General Description/Introduction of Company:

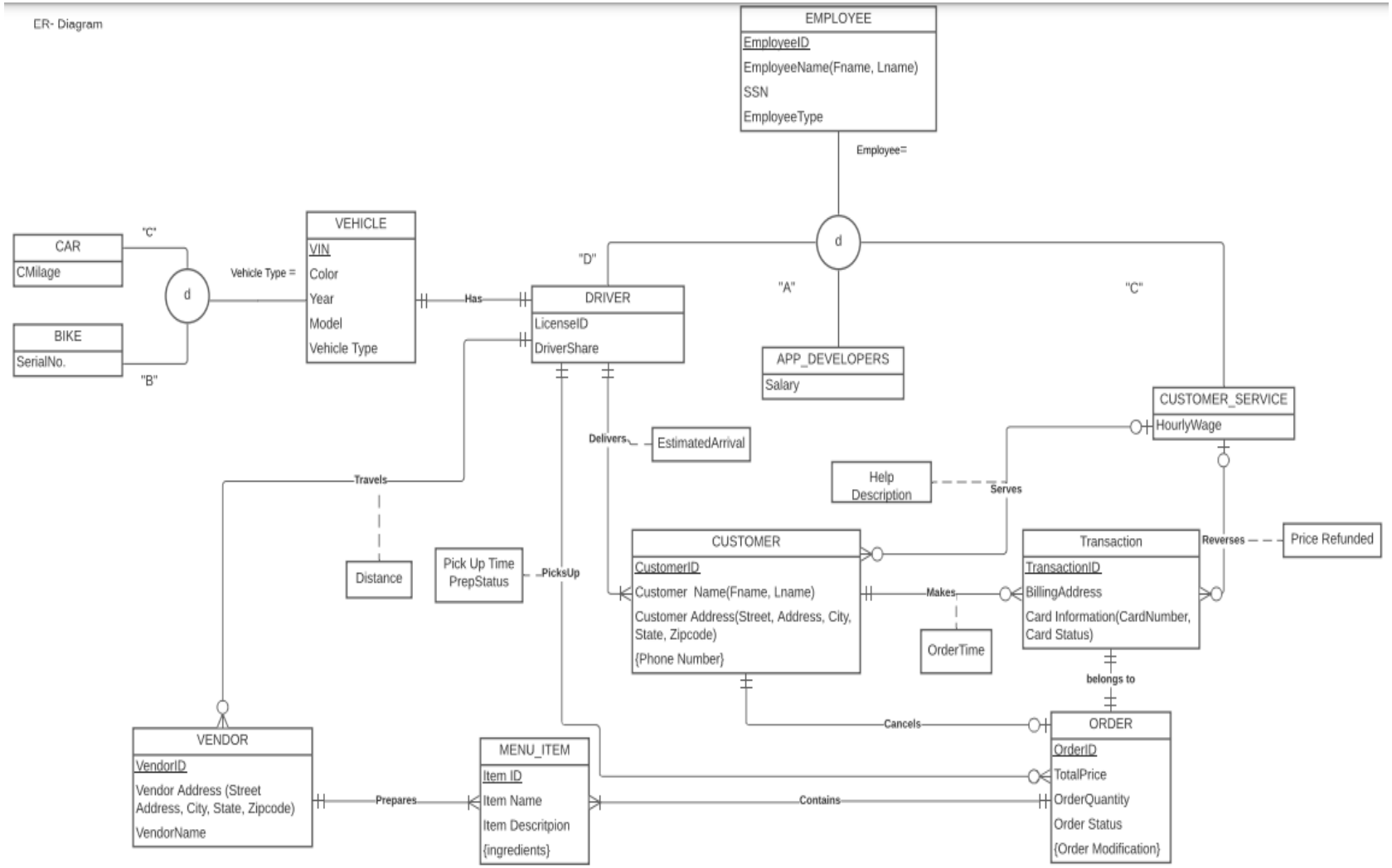
Doordash is a food delivery company whose purpose is to connect users to local restaurants, fast food, and corner stores in their area. Cutting out the hassle of having to drive and pick up your own food. Doordash prides itself on being able to provide users with as many local restaurants and fast foods as conveniently and quickly as possible. They compete with a lot of other competitors in the market, such as Postmates, uber eats, GrubHub, etc.

During this project, we will be focusing on the online pickup service that DoorDash provides. DoorDash lets you order food that is normally only available only at a restaurant to be conveniently ordered through a web browser or online app on the phone then delivered to the comfort of your home.

User Requirements:

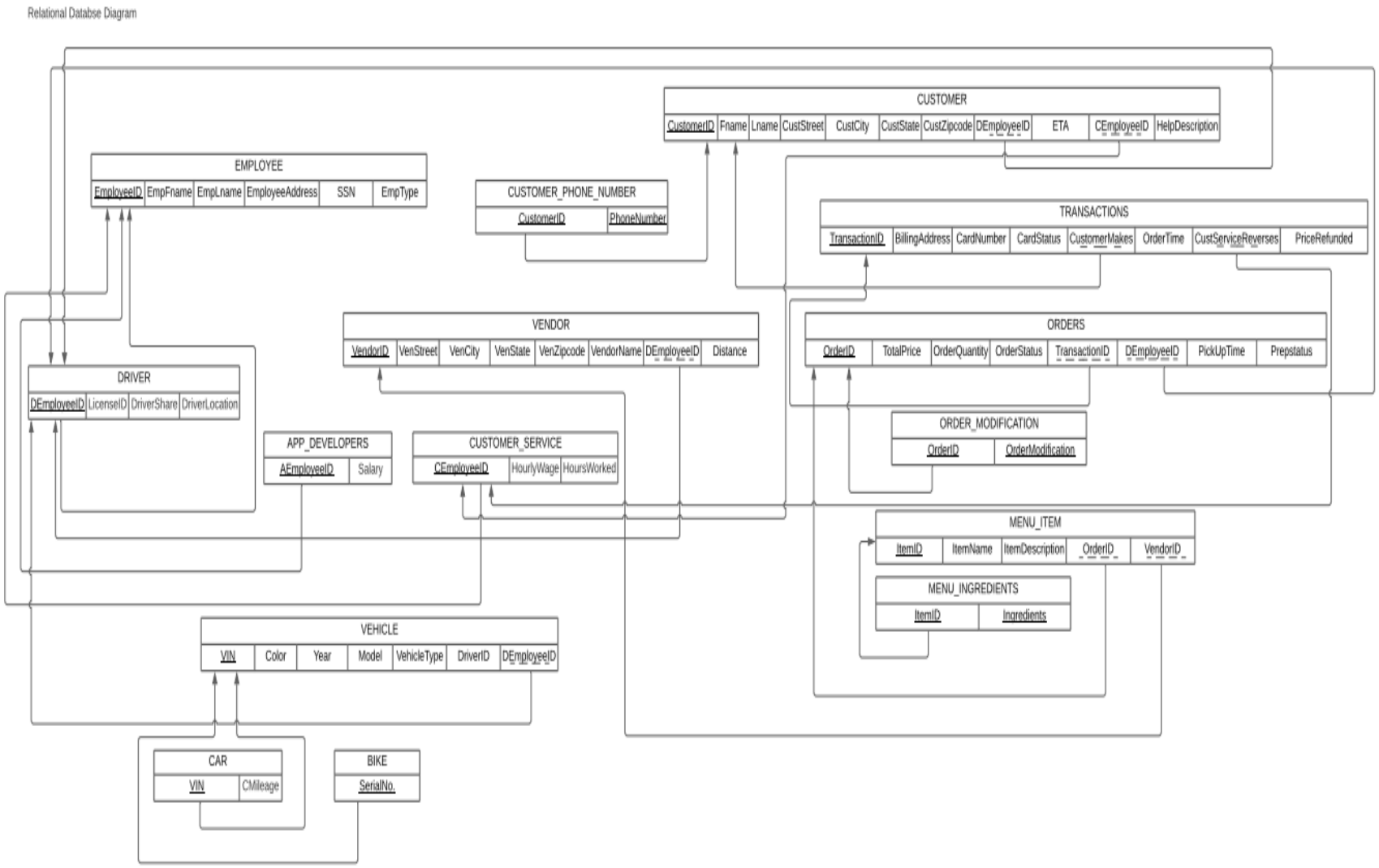
1) Doordash provides fast-food delivery to customers. Customers are identified by CustomerID, followed by Customer Name(First Name, Last Name), Customer Address(Street, Address, City, State, Zip Code), Card Information, and Phone Number.
2) Customers get to choose a vendor nearest to them. Vendor is identified by the VendorID and has attributes Vendor Address, Vendor Name, and Vendor Menu.
3) The Vendor Menu has many menu Items, while each menu item only belongs to the vendor Menu. Menu_Items, have an identifier as ItemID, with the following attributes of Item Name, Item Description and Items Ingredients. There are many different types of ingredients so this is identified as a Multi-Valued Attribute.
4) An order can contain zero to many menu items. While menu items only belong to one order. Order is identified by the Confirmation Number, along with the following attributes of Total Price, Order Quantity, and any Order Modifications, which identifies as Multi-Value.
5) The customer can make zero to many transactions on each order. Transaction is identified by the transaction ID, along with the customer's billing address and card information. Transaction only belongs to one customer.
6) Transaction only belongs to one order, and one order belongs to only one transaction. While the customer makes a transaction the order time is recorded.
7) Employees at Doordash are of three separate types: App Developers, Drivers, and Customer Service. Employees are identified by an Employee ID and have the attributes Employee Name (First Name, Last Name), Social Security Number (SSN). A Driver has additional attributes of License ID, Driver Share, and Driver Location. An App Developer has the additional attribute of Salary while Customer Service has the attribute Hourly Wage. An employee has to be either an App Developer, Customer Service, or Driver. No employee can be an App Developer, Customer Service, or Driver at the same time.

8) Every Driver has one or more vehicles and every vehicle belongs to one and only one driver. There are two separate types of vehicles that can be used by a driver : a Car or Bike. However, it is not limited to these options and a driver can have one or the other registered as his/her vehicle of choice. A Car has the unique identifier of Gas Mileage and a Bike has the unique identifier of a Bike ID.
9) While the Driver is in route to deliver to the Customer, the database is constantly registering the Driver's distance to the Customer. Thereby, giving a process entity of Estimated Arrival Time.
10) Database also records distance travelled from picking up the order and delivering the order to the customer.
11) Customer Service can serve zero to many customers, and customers may or may not need to be served by one customer service provider. The database records a brief description of the help needed from the customer.
12) Customer Service may or may not need to reverse a transaction for customers, and a transaction can only be reversed by one service agent. When its reverse the system records the price refunded
13) Customers can cancel one order at a time, they may or may not need to cancel orders.



Jon Aldrete, Pouya Sohrab, Daniel Lopez

IS 380 Database Management Project
Final Report



Jon Aldrete, Pouya Sohrab, Daniel Lopez

Implementation in My SQL:

Queries

Query 1: Lists all customers who entered information under HelpDescription, to figure out the customers who contacted customer service and needed help.

CUSTOMER TABLE:

CustomerID	Fname	Lname	CustStreet	CustCity	CustState	CustZipcode	DEmployeeID	ETA	CEmployeeID	HelpDescription
5001	Myron	Schultz	Quiet Valley Lane	LosAngeles	CA	90017	101	15 Min	121	None
5002	Cynthia	Harrison	Heavens Way	Venice	CA	92805	102	Cancelled	122	Cancelling Order
5003	Phil	Hanson	Lakeland Terrace	SantaMonica	CA	48226	103	20 Min	123	None
5004	Charlene	Montgomery	Poplar Avenue	Sacramento	CA	92115	104	43 Min	124	None
5005	Ricky	Shelton	Hillcrest Lane	SanPedro	CA	92801	105	38 Min	125	None
5006	Jacob	Gross	Rhode Island Avenue	SanDiego	CA	90001	106	42 Min	126	None
5007	Elizabeth	Erickson	Freedom Lane	Roseville	CA	93710	107	40 Min	127	None
5008	Constance	Brooks	Hillhaven Drive	Fresno	CA	90033	108	35 Min	128	None
5009	Archie	Wade	Pickens Way	Anaheim	CA	75601	109	29 Min	129	None
5010	Sherry	Miller	Francis Mine	Palmdale	CA	95652	110	Refunded	130	Requesting Refund
5011	James	Adamson	Riverside St	Roseville	CA	95678	NULL	NULL	NULL	NULL
5012	Sandra	Mequensie	PCH blvd	Los Angeles	CA	90034	NULL	NULL	NULL	NULL
5013	Jane	Smith	23rd St	South Sacramento	CA	90034	NULL	NULL	NULL	NULL
5014	Renie	Jones	L street	Long Beach	CA	90034	NULL	NULL	NULL	NULL
5015	Tony	Merick	Palms Blvd	Venice	CA	90064	NULL	NULL	NULL	NULL

IMPLEMENTATION:

```
SELECT * FROM customer WHERE customer.HelpDescription != 'None';
```

RESULT:

CustomerID	Fname	Lname	CustStreet	CustCity	CustState	CustZipcode	DEmployeeID
5002	Cynthia	Harrison	Heavens Way	Venice	CA	92805	102
5010	Sherry	Miller	Francis Mine	Palmdale	CA	95652	110

Query 2: Lists OrderQuantity values from orders table; to specify the quantity amount for each order.

ORDERS TABLE:

OrderID	TotalPrice	OrderQuantity	OrderStatus	TransactionID	DEmployeeID	PickUpTime	Prepstatus
1201	5.99	1	Fulfilled	4001	101	15:09:00	Prepared
1202	4.99	4	Cancelled	4002	102	00:00:00	Cancelled
1203	2.49	1	Fulfilled	4003	103	16:03:00	Prepared
1204	10.99	6	Fulfilled	4004	104	11:47:00	Prepared
1205	3.59	1	Fulfilled	4005	105	20:01:00	Preparing
1206	1.99	1	Fulfilled	4006	106	17:40:00	Preparing
1207	2.29	1	Fulfilled	4007	107	10:53:00	Preparing
1208	6.99	3	Fulfilled	4008	108	18:09:00	Preparing
1209	2	2	Fulfilled	4009	109	12:29:00	Preparing
1210	5.5	10	Refunded	4010	110	00:00:00	Refunded

IMPLEMENTATION:

```
1 SELECT orders.OrderQuantity FROM orders;
```

RESULT:

OrderQuantity
1
4
1
6
1
1
1
3
2
10

Query 3: Lists employeeID for drivers that have received an order.

DRIVER TABLE:

DEmployeeID	LicenseID	DriverShare
101	I2939363	9.23
102	T8371860	11.23
103	R9166636	10.45
104	L8640194	5.23
105	Z2896603	4.32
106	Z2364707	7.34
107	D3025459	5.34
108	W9953813	4.54
109	N2884718	2.75
110	U9068884	6.58

ORDERS TABLE:

OrderID	TotalPrice	OrderQuantity	OrderStatus	TransactionID	DEmployeeID	PickUpTime	Prepstatus
1201	5.99	1	Fulfilled	4001	101	15:09:00	Prepared
1202	4.99	4	Cancelled	4002	102	00:00:00	Cancelled
1203	2.49	1	Fulfilled	4003	103	16:03:00	Prepared
1204	10.99	6	Fulfilled	4004	104	11:47:00	Prepared
1205	3.59	1	Fulfilled	4005	105	20:01:00	Preparing
1206	1.99	1	Fulfilled	4006	106	17:40:00	Preparing
1207	2.29	1	Fulfilled	4007	107	10:53:00	Preparing
1208	6.99	3	Fulfilled	4008	108	18:09:00	Preparing
1209	2	2	Fulfilled	4009	109	12:29:00	Preparing
1210	5.5	10	Refunded	4010	110	00:00:00	Refunded

IMPLEMENTATION:

```
1 SELECT driver.DEmployeeID
2 FROM driver
3 INNER JOIN orders
4 ON driver.DEmployeeID = orders.DEmployeeID;
```

RESULT:

DEmployeeID
101
102
103
104
105
106
107
108
109
110

IS 380 Database Management Project
Final Report

Query 4: Lists FName and OrderID of all customers who did and did not place an order while being ordered by First name.

Implementation:

```
1 SELECT customer.Fname, orders.OrderID
2 FROM customer
3 LEFT JOIN orders on customer.DEmployeeID = orders.DEmployeeID
4 ORDER BY customer.Fname;
```

Result:

Fname ▲ 1	OrderID
Archie	1209
Charlene	1204
Constance	1208
Cynthia	1202
Elizabeth	1207
Jacob	1206
James	*NULL*
Jane	*NULL*
Myron	1201
Phil	1203
Renie	*NULL*
Ricky	1205
Sandra	*NULL*
Sherry	1210
Tony	*NULL*

Query 5: List data from drivers who have bikes.

DRIVER TABLE:

DEmployeeID	LicenseID	DriverShare
101	I2939363	9.23
102	T8371860	11.23
103	R9166636	10.45
104	L8640194	5.23
105	Z2896603	4.32
106	Z2364707	7.34
107	D3025459	5.34
108	W9953813	4.54
109	N2884718	2.75
110	U9068884	6.58

BIKE TABLE:

SerialNo	DEmployeeID
56QXZ6fi	109
nZrvLWkc	110

MYSQL Statement:

```
SELECT * FROM driver NATURAL JOIN bike;
```

Result:

DEmployeeID	LicenseID	DriverShare	SerialNo
109	N2884718	2.75	56QXZ6fi
110	U9068884	6.58	nZrvLWkc

IS 380 Database Management Project
Final Report

VIEWS

View 1: This view shows transaction ID for transactions that the card was declined

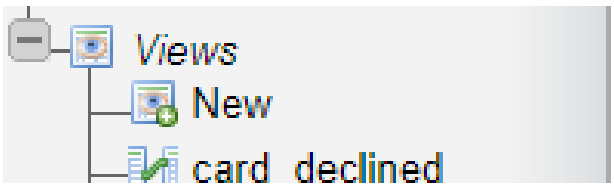
TRANSACTION TABLE

TransactionID	BillingAddress	CardNumber	CardStatus	CustState
4001	3221 Quiet Valley Lane	5392637452128493	approved	CA
4002	2360 Heavens Way	4929502736807020	declined	CA
4003	4993 Lakeland Terrace	5530696514378283	approved	CA
4004	4854 Poplar Avenue	5384811835230801	approved	CA
4005	4658 Hillcrest Lane	4532253965747771	approved	CA
4006	3531 Rhode Island Avenue	4716428288628316	approved	CA
4007	2419 Freedom Lane	5258033242814704	approved	CA
4008	2703 Hillhaven Drive	5561625697716607	approved	CA
4009	923 Pickens Way	4485584745343318	approved	CA
4010	2373 Francis Mine	5478669012614388	declined	CA

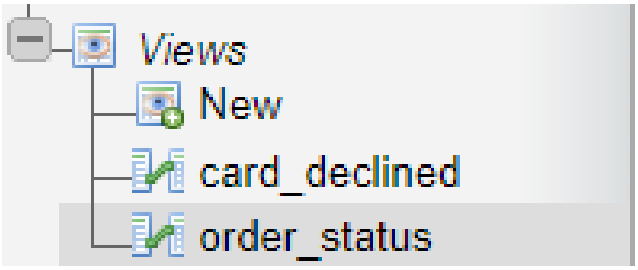
MYSQL Statement:

```
1 CREATE VIEW card_declined AS
2 SELECT transactions.TransactionID
3 FROM transactions
4 WHERE transactions.CardStatus = 'declined';
```

Result:



View 2: This view shows the Order ID's and Driver Employee ID' s for orders that were successfully completed/fulfilled.



MYSQL STATEMENT:

```
1 CREATE VIEW order_status AS
2 SELECT orders.OrderID, orders.DEmployeeID
3 FROM orders
4 WHERE orders.OrderStatus = 'Fulfilled';
```

RESULT:

+ Options

	OrderID	DEmployeeID
Edit Copy Delete	1201	101
Edit Copy Delete	1203	103
Edit Copy Delete	1204	104
Edit Copy Delete	1205	105
Edit Copy Delete	1206	106
Edit Copy Delete	1207	107
Edit Copy Delete	1208	108
Edit Copy Delete	1209	109

IS 380 Database Management Project
Final Report

Procedures

Procedure 1: Stored procedure to list all PrepStatus for orders and invoke the procedure with Preparing.

MSQL Statement:

```
DELIMITER //
CREATE PROCEDURE GetPrepStatus (IN x VARCHAR (30))
BEGIN
SELECT * FROM orders
WHERE orders.Prepstatus = x;
END//
DELIMITER ;
```

Call Statement:

```
Call GetPrepStatus('Preparing');
```

Result:

OrderID	TotalPrice	OrderQuantity	OrderStatus	TransactionID	DEmployeeID	PickUpTime	Prepstatus
1205	3.59	1	Fulfilled	4005	105	20:01:00	Preparing
1206	1.99	1	Fulfilled	4006	106	17:40:00	Preparing
1207	2.29	1	Fulfilled	4007	107	10:53:00	Preparing
1208	6.99	3	Fulfilled	4008	108	18:09:00	Preparing
1209	2	2	Fulfilled	4009	109	12:29:00	Preparing

Procedure 2: Stored procedure to list all customers from a specific city and invoke the procedure for the city "Roseville ".

MYSQL Statement:

```
DELIMITER //
CREATE PROCEDURE GetCity (IN x VARCHAR (50))
BEGIN
SELECT * FROM customer
WHERE customer.CustCity = x;
END//
DELIMITER ;
```

Call Statement:

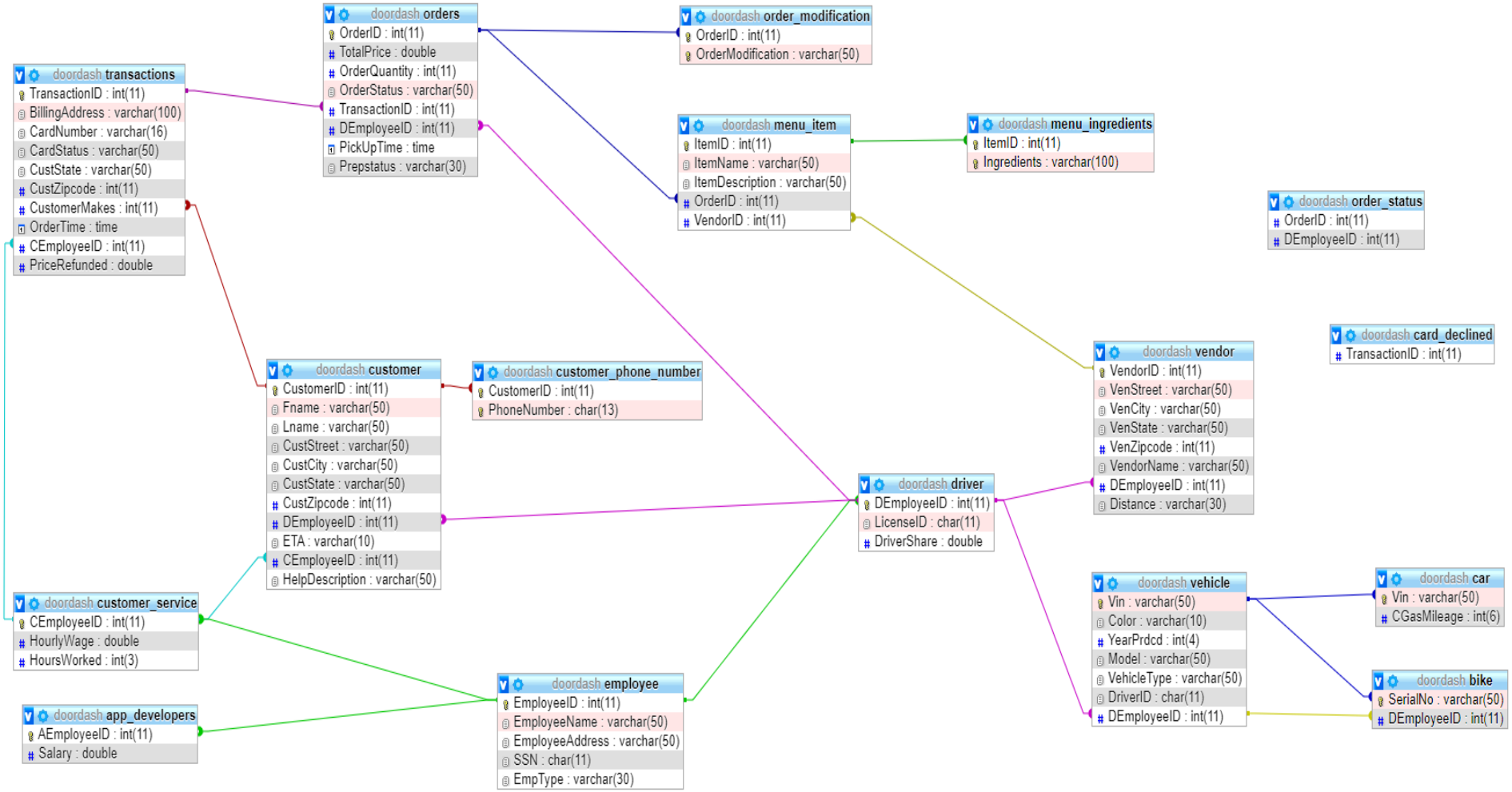
```
CALL GetCity ('Roseville');
```

Result:

Options

CustomerID	Fname	Lname	CustStreet	CustCity	CustState	CustZipcode	DEmployeeID	ETA	CEmployeeID	HelpDescription
5007	Elizabeth	Erickson	Freedom Lane	Roseville	CA	93710	107	40 Min	127	None
5011	James	Adamson	Riverside St	Roseville	CA	95678	*NULL*	*NULL*	*NULL*	*NULL*

IS 380 Database Management Project
Final Report



Jon Aldrete, Pouya Sohrab, Daniel Lopez

Here's the implementation of the order's modification:

OrderID	OrderModification
1201	Add Ketchup
1201	Extra Lettuce
1202	Add BBQ
1203	No Salt
1204	No Onions
1205	No Whip Cream
1206	Less Sugar
1207	No Mayo
1208	Extra Gravy
1209	Add Sour Cream
1210	Add Nacho Cheese

For a better Idea in how order modification looks like in the database, and what it represents.