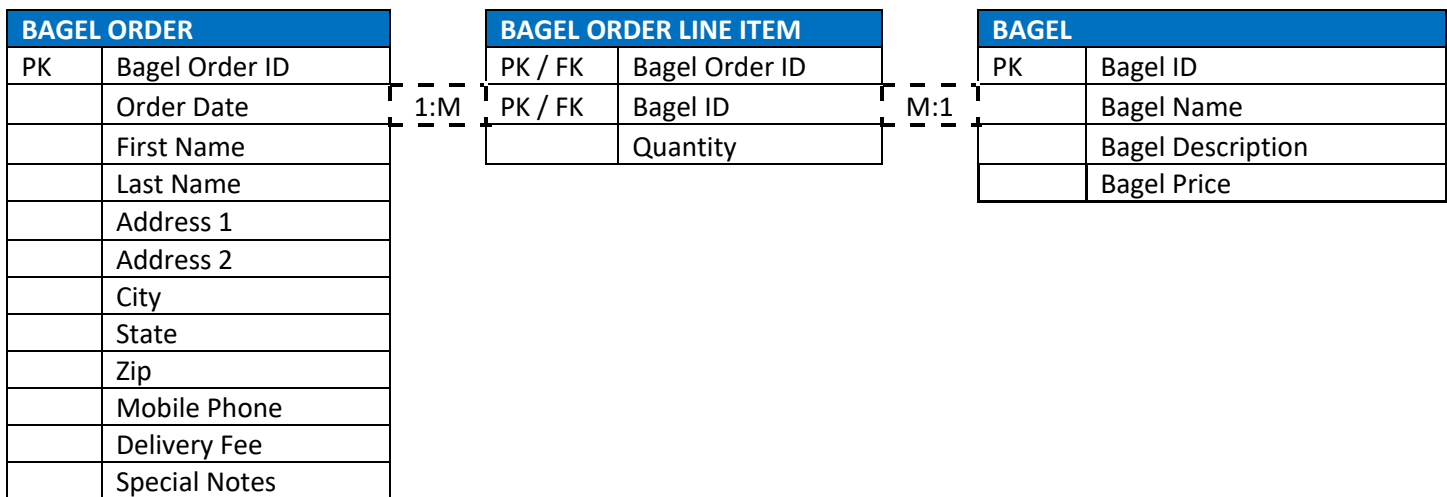


## VHT2: Normalization and Database Design

### A. Normalized physical database model to represent the ordering process for Nora's Bagel Bin

1a, 1b.

#### Second Normal Form (2NF)



1c.

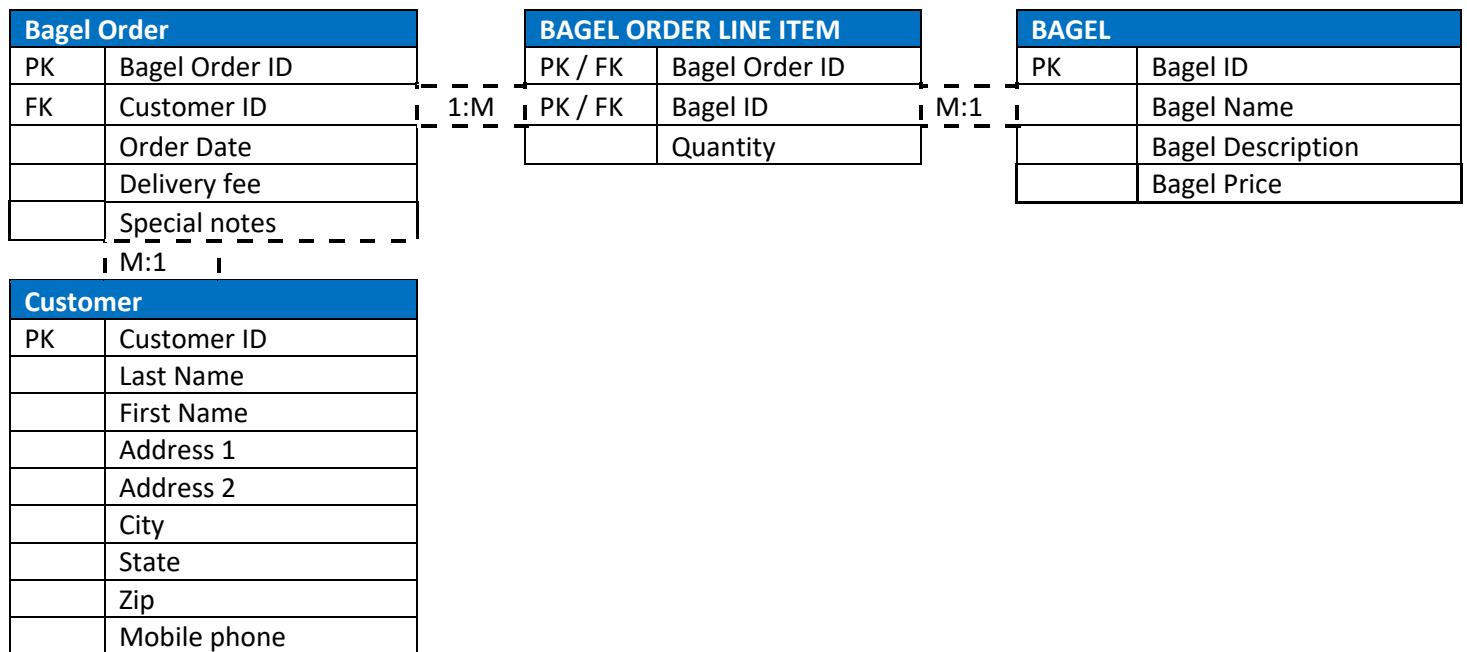
I assigned attributes to the second normal form relation by looking at the attributes of the first normal form table and seeing how they were related to the composite primary key of Bagel ID and bagel order ID in the 1NF table. My goal was to make sure that the attributes are functionally dependent on the composite primary key and not just part of the composite primary key according to the definition of second normal form. I noticed that the attributes bagel name, description, and price in the 1NF table could be determined by the Bagel ID alone instead of using both primary keys. I then determined that these attributes could be put into a separate table using Bagel ID as the primary key. Another table was also added because there were attributes in the 1NF table that could be determined with only the bagel Order ID. The table is called "Bagel Order" and uses Bagel Order ID as the primary key. The orders quantity could only be determined by using both the bagel Order Id and the Bagel Id as a composite primary key in the Bagel Order Line Item table.

The cardinality was determined by looking at the relationship between the relations. A bagel order can contain many line items within one order, so I determined that the cardinality between Bagel Order and Bagel Order Line Item is 1 to many. There is a many to one relationship between Bagel Order Line Item and Bagel because many bagel order line items can have 1 bagel type and 1 bagel may be in many different order line items.

# Nora's Bagel Bin Database Blueprints

2a, 2b, 2c, 2d.

## Third Normal Form (3NF)



2e.

While observing the second normal form table I noticed that there were two separate entities being represented in the bagel order table. The second normal form Bagel Order table includes order information and customer information. According to Dr. Daniel Soper's YouTube video 'The Relational Model', it is a general rule that a well-formed table will not encompass more than one business concept. I decided that it was best to split the attributes related to bagel orders and the attributes related to customers into separate tables. To achieve third normal form the table 'Customer' was added. The Customer table also prevents transitive dependencies where an attribute can be determined not only by the primary key but also by another non key attribute. I also wanted to prevent repeating data because if a customer were to place multiple orders in the second normal form Bagel Order table, then their name, address, and phone would have to be repeated within the table. An attribute called Customer ID was added to the Customer table as a primary key for uniqueness and to make each specific customer easily identifiable.

Because there is another table added there must be another cardinality. I determined that the cardinality between Bagel Order and Customer is Many to One because 1 customer can have many bagel orders or in other words many bagel orders can only have one customer per order.

# Nora’s Bagel Bin Database Blueprints

3a, 3b.

Final Physical Database Model with Data Types

Bagel Order		
PK	bagel_order_id	INT
FK	bagel_id	CHAR(2)
	order_date	TIMESTAMP
	delivery_fee	INT
	special_notes	VARCHAR(255)

M:1

Customer		
PK	customer_id	INT
	last_name	VARCHAR(50)
	first_name	VARCHAR(50)
	address_1	VARCHAR(50)
	address_2	VARCHAR(50)
	city	VARCHAR(30)
	state	CHAR(2)
	zip	VARCHAR(10)
	mobile_phone	VARCHAR(10)

BAGEL ORDER LINE ITEM		
PK / FK	bagel_order_id	INT
PK / FK	bagel_id	CHAR(2)
	quantity	INT

M:1

BAGEL		
PK	bagel_id	CHAR(2)
	bagel_name	VARCHAR(30)
	bagel_description	VARCHAR(30)
	bagel_price	NUMERIC(3,2)

## B. Create A database

1a. SQL code to create each table as specified in “Jaunty Coffee Co. ERD”.

<pre>CREATE TABLE COFFEE_SHOP(   shop_id INT,   shop_name VARCHAR(50),   city VARCHAR(50),   state CHAR(2),   PRIMARY KEY (shop_id) );</pre>	<pre>CREATE TABLE EMPLOYEE(   employee_id INT,   first_name VARCHAR(30),   last_name VARCHAR(30),   hire_date DATE,   job_title VARCHAR(30),   shop_id INT,   PRIMARY KEY (employee_id),   FOREIGN KEY (shop_id) REFERENCES   COFFEE_SHOP(shop_id) );</pre>	<pre>CREATE TABLE SUPPLIER(   supplier_id INT,   company_name VARCHAR(50),   country VARCHAR(30),   sales_contact_name VARCHAR(60),   email VARCHAR(50) NOT NULL,   PRIMARY KEY (supplier_id) );</pre>	<pre>CREATE TABLE COFFEE(   coffee_id INT,   shop_id INT,   supplier_id INT,   coffee_name VARCHAR(30),   price_per_pound NUMERIC(5,2),   PRIMARY KEY (coffee_id),   FOREIGN KEY (shop_id) REFERENCES   COFFEE_SHOP(shop_id),   FOREIGN KEY(supplier_id)   REFERENCES SUPPLIER (supplier_id) );</pre>
--	---	--	---

## 1b. Screenshot of SQL commands and database server response for creating tables.

The screenshot shows the DB Fiddle web application interface. The browser address bar displays 'db-fiddle.com'. The application header includes navigation links for 'WGU Student Portal', 'Tips for Getting Started with the VHT2 Perform...', 'WGU Performance Assessment', and 'DB Fiddle - SQL Database Playground'. The main toolbar contains icons for database selection (MySQL v5.7), 'Run', 'Save', 'Load Example', and 'Collaborate', along with 'Sign in' and 'Have any feedback?' buttons.

The 'Schema SQL' section displays the following SQL commands:

```
1 CREATE TABLE COFFEE_SHOP(  
2   shop_id INT,  
3   shop_name VARCHAR(50),  
4   city VARCHAR(50),  
5   state CHAR(2),  
6   PRIMARY KEY (shop_id)  
7 );  
8 CREATE TABLE EMPLOYEE(  
9   employee_id INT,  
10  first_name VARCHAR(30),  
11  last_name VARCHAR(30),  
12  hire_date DATE,  
13  job_title VARCHAR(30),  
14  shop_id INT,  
15  PRIMARY KEY (employee_id),  
16  FOREIGN KEY (shop_id) REFERENCES COFFEE_SHOP(shop_id)  
17 );  
18 CREATE TABLE SUPPLIER(  
19  supplier_id INT,  
20  company_name VARCHAR(50),  
21  country VARCHAR(30),  
22  sales_contact_name VARCHAR(60),  
23  email VARCHAR(50) NOT NULL,  
24  PRIMARY KEY (supplier_id)  
25 );  
26 CREATE TABLE COFFEE(  
27  coffee_id INT,  
28  shop_id INT,  
29  supplier_id INT,  
30  coffee_name VARCHAR(30),  
31  price_per_pound NUMERIC(5,2),  
32  PRIMARY KEY (coffee_id),  
33  FOREIGN KEY (shop_id) REFERENCES COFFEE_SHOP(shop_id),  
34  FOREIGN KEY (supplier_id) REFERENCES SUPPLIER (supplier_id)  
35 );  
36
```

A green notification box on the right indicates: 'Query successfully executed in 8ms'. A 'Text to DDL' button is located at the bottom left of the SQL editor area.

DB Fiddle - Crafted with ♥ by Status200 in the United Kingdom. Terms of Use • Privacy / Cookie Policy • Status200 Ltd © 2018

**2a. SQL code to populate tables with at least three rows of data.**

```
INSERT INTO COFFEE_SHOP
VALUES (01,'Waynes Coffee Emporium', 'Atlanta', 'GA'),
      (02,'Morning Brew', 'Seattle', 'WA'),
      (03,'Wake Up Cafe', 'Memphis', 'TN')
;
```

```
INSERT INTO EMPLOYEE
VALUES (001, 'Peter', 'Parker', '2002-04-29', 'cashier', 02),
      (002, 'Bucky', 'Barnes', '2011-07-19', 'barista', 02),
      (003, 'Bruce', 'Banner', '2003-06-20', 'manager', 01)
;
```

```
INSERT INTO SUPPLIER
VALUES (1, 'Folgers', 'United Sates', 'Jessica Bateman', 'jessica.bateman@gmail.com'),
      (2, 'Cafe De Columbia', 'Columbia', 'Bill Faningham', 'Bfaningham@cafecolumbia.org'),
      (3, 'Keurig', 'United States', 'Brooke Johnson', 'Brooke.Johnson91@outlook.coom')
;
```

```
INSERT INTO COFFEE
VALUES (1, 03, 2, 'Mezcla', 20.00),
      (2, 01, 1, 'Smooth Blend', 12.00),
      (3, 01, 1, 'Black Brew', 17.00)
;
```

## 2b. SQL commands and database server response for populating tables.

db-fiddle.com

WGU Student PortalWGU Performance AssessmentDB Fiddle - SQL Database Playground

Database: MySQL v5.7RunUpdateForkLoad ExampleStarPROEmbedPROCollaborateSign inHave any feedback?

Schema SQL

```
24 PRIMARY KEY (supplier_id)
25 );
26 CREATE TABLE COFFEE(
27   coffee_id INT,
28   shop_id INT,
29   supplier_id INT,
30   coffee_name VARCHAR(30),
31   price_per_pound NUMERIC(5,2),
32   PRIMARY KEY (coffee_id),
33   FOREIGN KEY (shop_id) REFERENCES COFFEE_SHOP(shop_id),
34   FOREIGN KEY(supplier_id) REFERENCES SUPPLIER (supplier_id)
35 );
36
37 INSERT INTO COFFEE_SHOP
38 VALUES (01,'Waynes Coffee Emporium', 'Atlanta', 'GA'),
39         (02,'Morning Brew', 'Seattle', 'WA'),
40         (03,'Wake Up Cafe', 'Memphis', 'TN')
41 ;
42
43 INSERT INTO EMPLOYEE
44 VALUES (001, 'Peter', 'Parker', '2002-04-29', 'cashier', 02),
45         (002, 'Bucky', 'Barnes', '2011-07-19', 'barista', 02),
46         (003, 'Bruce', 'Banner', '2003-06-20', 'manager', 01)
47 ;
48
49 INSERT INTO SUPPLIER
50 VALUES (1, 'Folgers', 'United Sates', 'Jessica Bateman', 'jessica.bateman@gmail.com'),
51         (2, 'Cafe De Columbia', 'Columbia', 'Bill Faningham', 'Bfaningham@cafecolumbia.org'),
52         (3, 'Keurig', 'United States', 'Brooke Johnson', 'Brooke.Johnson91@outlook.coom')
53 ;
54
55 INSERT INTO COFFEE
56 VALUES (1, 03, 2, 'Mezcla', 20.00),
57         (2, 01, 1, 'Smooth Blend', 12.00),
58         (3, 01, 1, 'Black Brew', 17.00)
59 ;
```

Query successfully executed in 19ms

Text to DDL

DB Fiddle - Crafted with ♥ by Status200 in the United Kingdom. Terms of Use · Privacy / Cookie Policy · Status200 Ltd © 2018

### 3a. SQL code to create a view.

```
CREATE VIEW EMPLOYEE_FULLNAME
AS SELECT employee_id, CONCAT(first_name, ' ', last_name) AS full_name, hire_date, job_title, shop_id
FROM EMPLOYEE
;
```

### 3b. Screenshot of SQL commands and server response for creating view.

The screenshot shows the DB Fiddle - SQL Database Playground interface. The browser address bar displays 'db-fiddle.com'. The page header includes 'WGU Student Portal', 'WGU Performance Assessment', and 'DB Fiddle - SQL Database Playground'. The main content area shows a list of SQL commands for creating a schema, inserting data into COFFEE\_SHOP, EMPLOYEE, and SUPPLIER tables, and creating a view named EMPLOYEE\_FULLNAME. The view creation command is highlighted with a red box. A green notification box indicates 'Query successfully executed in 19ms'. The footer contains 'DB Fiddle - Crafted with ♥ by Status200 in the United Kingdom.' and 'Terms of Use • Privacy / Cookie Policy • Status200 Ltd © 2018'.

Database: MySQL v5.7

Run Update Fork Load Example Star PRO Embed PRO Collaborate Sign in Have any feedback?

Schema SQL

```
29 supplier_id INT(4),
30 coffee_name VARCHAR(30),
31 price_per_pound NUMERIC(5,2),
32 PRIMARY KEY (coffee_id),
33 FOREIGN KEY (shop_id) REFERENCES COFFEE_SHOP(shop_id),
34 FOREIGN KEY(supplier_id) REFERENCES SUPPLIER (supplier_id)
35 );
36
37 INSERT INTO COFFEE_SHOP
38 VALUES (01,'Waynes Coffee Emporium', 'Atlanta', 'GA'),
39         (02,'Morning Brew', 'Seattle', 'WA'),
40         (03,'Wake Up Cafe', 'Memphis', 'TN')
41 ;
42
43 INSERT INTO EMPLOYEE
44 VALUES (001, 'Peter', 'Parker', '2002-04-29', 'cashier', 02),
45         (002, 'Bucky', 'Barnes', '2011-07-19', 'barista', 02),
46         (003, 'Bruce', 'Banner', '2003-06-20', 'manager', 01)
47 ;
48
49 INSERT INTO SUPPLIER
50 VALUES (1, 'Folgers', 'United Sates', 'Jessica Bateman', 'jessica.bateman@gmail.com'),
51         (2, 'Cafe De Columbia', 'Columbia', 'Bill Faningham', 'Bfaningham@cafecolumbia.org'),
52         (3, 'Keurig', 'United States', 'Brooke Johnson', 'Brooke.Johnson91@outlook.com')
53 ;
54
55 INSERT INTO COFFEE
56 VALUES (1, 03, 2, 'Mezcla', 20.00),
57         (2, 01, 1, 'Smooth Blend', 12.00),
58         (3, 01, 1, 'Black Brew', 17.00)
59 ;
60
61 CREATE VIEW EMPLOYEE_FULLNAME
62 AS SELECT employee_id, CONCAT(first_name, ' ', last_name) AS full_name, hire_date, job_title, shop_id
63 FROM EMPLOYEE
64 ;
```

Query successfully executed in 19ms

Text to DDL

DB Fiddle - Crafted with ♥ by Status200 in the United Kingdom. Terms of Use • Privacy / Cookie Policy • Status200 Ltd © 2018



#### 4a. SQL code to create and index on the coffee\_name field from the 'COFFEE' table.

```
CREATE INDEX coffee_index
ON COFFEE(coffee_name)
;
```

#### 4b. Demonstrate that you tested your code by providing a screenshot showing your SQL commands and the database server's response to create index.

The screenshot shows the DB Fiddle - SQL Database Playground interface. The browser address bar displays 'db-fiddle.com'. The page header includes 'WGU Student Portal', 'WGU Performance Assessment', and 'DB Fiddle - SQL Database Playground'. The main toolbar contains buttons for 'Database: MySQL v5.7', 'Run', 'Update', 'Fork', 'Load Example', 'Star', 'PRO', 'Embed', 'PRO', 'Collaborate', 'Sign in', 'Have any feedback?', and a Twitter icon. The 'Schema SQL' section displays the following SQL commands:

```
34 FOREIGN KEY(supplier_id) REFERENCES SUPPLIER (supplier_id)
35 );
36
37 INSERT INTO COFFEE_SHOP
38 VALUES (01, 'Waynes Coffee Emporium', 'Atlanta', 'GA'),
39          (02, 'Morning Brew', 'Seattle', 'WA'),
40          (03, 'Wake Up Cafe', 'Memphis', 'TN')
41 ;
42
43 INSERT INTO EMPLOYEE
44 VALUES (001, 'Peter', 'Parker', '2002-04-29', 'cashier', 02),
45          (002, 'Bucky', 'Barnes', '2011-07-19', 'barista', 02),
46          (003, 'Bruce', 'Banner', '2003-06-20', 'manager', 01)
47 ;
48
49 INSERT INTO SUPPLIER
50 VALUES (1, 'Folgers', 'United Sates', 'Jessica Bateman', 'jessica.bateman@gmail.com'),
51          (2, 'Cafe De Columbia', 'Columbia', 'Bill Fanningham', 'Bfanningham@cafecolumbia.org'),
52          (3, 'Keurig', 'United States', 'Brooke Johnson', 'Brooke.Johnson91@outlook.coom')
53 ;
54
55 INSERT INTO COFFEE
56 VALUES (1, 03, 2, 'Mezcla', 20.00),
57          (2, 01, 1, 'Smooth Blend', 12.00),
58          (3, 01, 1, 'Black Brew', 17.00)
59 ;
60
61 CREATE VIEW EMPLOYEE_FULLNAME
62 AS SELECT employee_id, CONCAT(first_name, ' ', last_name) AS full_name, hire_date, job_title, shop_id
63 FROM EMPLOYEE
64 ;
65
66 CREATE INDEX coffee_index
67 ON COFFEE(coffee_name)
68 ;
```

A green notification box indicates 'Query successfully executed in 62ms'. The 'Text to DDL' button is visible at the bottom left. The footer contains 'DB Fiddle - Crafted with ♥ by Status200 in the United Kingdom.' and 'Terms of Use • Privacy / Cookie Policy • Status200 Ltd © 2018'.

5a. Provide the SQL code you wrote to create your SFW query.

```
SELECT *  
FROM EMPLOYEE  
WHERE job_title = 'cashier'  
;
```

5b. Demonstrate that you tested your code by providing a screenshot showing your SQL commands and the database server's response to SFW.

The screenshot displays the DB Fiddle - SQL Database Playground interface. The top navigation bar includes links for 'Run', 'Update', 'Fork', 'Load Example', 'Star', 'Embed', 'Collaborate', 'Sign in', and 'Have any feedback?'. The main content area is divided into three sections: 'Fiddle Title', 'Fiddle Description', and 'Schema SQL'. The 'Schema SQL' section contains the following code:

```
43 INSERT INTO EMPLOYEE  
44 VALUES (001, 'Peter', 'Parker', '2002-04-29', 'cashier',  
45           02),  
46           (002, 'Bucky', 'Barnes', '2011-07-19', 'barista',  
47           02),  
48           (003, 'Bruce', 'Banner', '2003-06-20', 'manager', 01)  
49 ;  
50 INSERT INTO SUPPLIER  
51 VALUES (1, 'Folgers', 'United Sates', 'Jessica Bateman',  
52           'jessica.bateman@gmail.com'),  
53           (2, 'Cafe De Columbia', 'Columbia', 'Bill Faningham',  
54           'Bfaningham@cafecolumbia.org'),  
55           (3, 'Keurig', 'United States', 'Brooke Johnson',  
56           'Brooke.Johnson91@outlook.com')
```

The 'Query SQL' section shows the executed query:

```
1 SELECT *  
2 FROM EMPLOYEE  
3 WHERE job_title = 'cashier'  
4 ;
```

A green notification box indicates 'Query successfully executed in 59ms'. Below the query, the 'Results' section displays the output of the query in a table format:

employee_id	first_name	last_name	hire_date	job_title	shop_id
1	Peter	Parker	2002-04-29	cashier	2

The bottom of the interface includes a footer with the text 'DB Fiddle - Crafted with ♥ by Status200 in the United Kingdom.' and 'Terms of Use / Privacy / Cookie Policy / Status200 Ltd © 2018'.

**6a. Provide the SQL code you wrote to create your table joins query. The query should join three different tables and include attributes from *all* three tables in its output.**

```
SELECT A.shop_id, A.shop_name, A.city, A.state, B.coffee_id, B.coffee_name, B.price_per_pound, C.supplier_id, C.company_name, C.country,
C.sales_contact_name, C.email
FROM COFFEE_SHOP A
INNER JOIN COFFEE B
ON A.shop_id = B.shop_id
INNER JOIN SUPPLIER C
ON C.supplier_id = B.supplier_id
;
```

**6b. Demonstrate that you tested your code by providing a screenshot showing your SQL commands and the database server's response.**

The screenshot shows a SQL editor interface with a blue header bar containing navigation icons and a 'Run' button. The main area is divided into two panes. The left pane, titled 'Schema SQL', contains SQL code for creating three tables: COFFEE\_SHOP, EMPLOYEE, and SUPPLIER. The right pane, titled 'Query SQL', shows a SELECT query joining COFFEE\_SHOP, COFFEE, and SUPPLIER tables. A green notification box above the query states 'Query successfully executed in 84ms'. Below the query panes, a 'Results' section displays the output of the query as a table with 12 columns and 3 rows. A 'Copy as Markdown' button is located in the top right of the results section.

**Schema SQL**

```
1 CREATE TABLE COFFEE_SHOP(
2   shop_id INT,
3   shop_name VARCHAR(50),
4   city VARCHAR(50),
5   state CHAR(2),
6   PRIMARY KEY (shop_id)
7 );
8 CREATE TABLE EMPLOYEE(
9   employee_id INT,
10  first_name VARCHAR(30),
11  last_name VARCHAR(30),
12  hire_date DATE,
13  job_title VARCHAR(30),
14  shop_id INT,
15  PRIMARY KEY (employee_id),
16  FOREIGN KEY (shop_id) REFERENCES COFFEE_SHOP(shop_id)
17 );
18 CREATE TABLE SUPPLIER(
19  supplier_id INT,
20  company_name VARCHAR(50),
21  country VARCHAR(30),
22  sales_contact_name VARCHAR(60),
23  email VARCHAR(50) NOT NULL,
24  PRIMARY KEY (supplier_id)
25 );
```

**Query SQL**

```
1
2
3 SELECT A.shop_id, A.shop_name, A.city, A.state, B.coffee_id, B.coffee_name, B.price_per_pound,
4   C.supplier_id, C.company_name, C.country, C.sales_contact_name, C.email
5 FROM COFFEE_SHOP A
6 INNER JOIN COFFEE B
7   ON A.shop_id = B.shop_id
8 INNER JOIN SUPPLIER C
9   ON C.supplier_id = B.supplier_id
```

Query successfully executed in 84ms

**Results**

Query #1 Execution time: 1ms

shop_id	shop_name	city	state	coffee_id	coffee_name	price_per_pound	supplier_id	company_name	country	sales_contact_name	email
3	Wake Up Cafe	Memphis	TN	1	Mezcla	20.00	2	Cafe De Columbia	Columbia	Bill Faningham	Bfaningham@cafecolumbia.org
1	Waynes Coffee Emporium	Atlanta	GA	2	Smooth Blend	12.00	1	Folgers	United States	Jessica Bateman	jessica.bateman@gmail.com
1	Waynes Coffee Emporium	Atlanta	GA	3	Black Brew	17.00	1	Folgers	United States	Jessica Bateman	jessica.bateman@gmail.com

Copy as Markdown