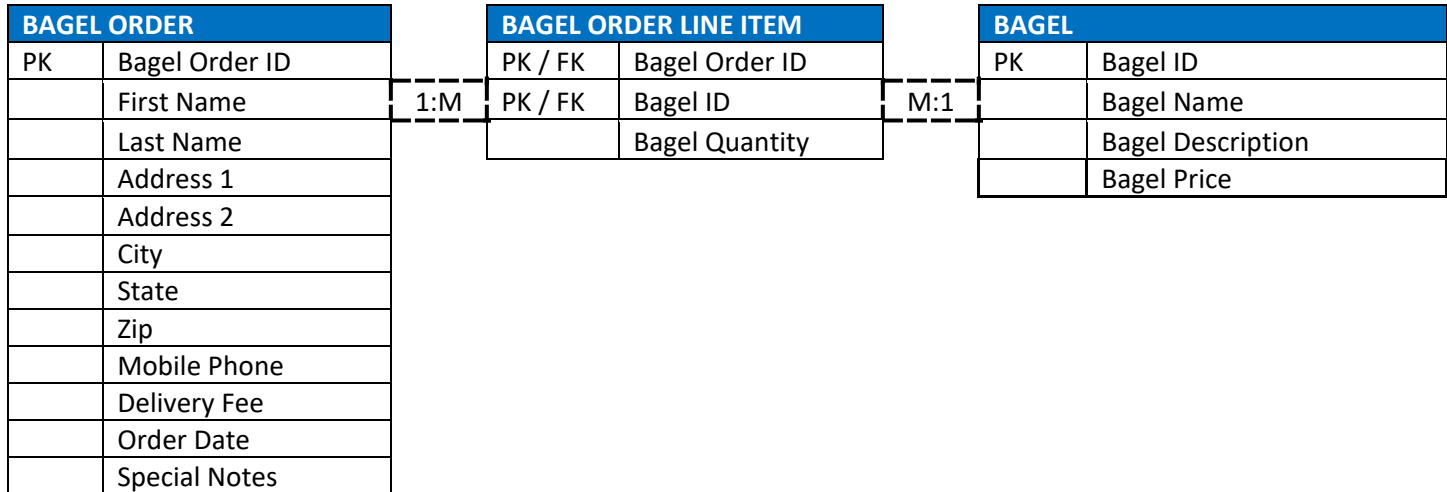


## Normalization and Database Design

A1.

### Second Normal Form (2NF)

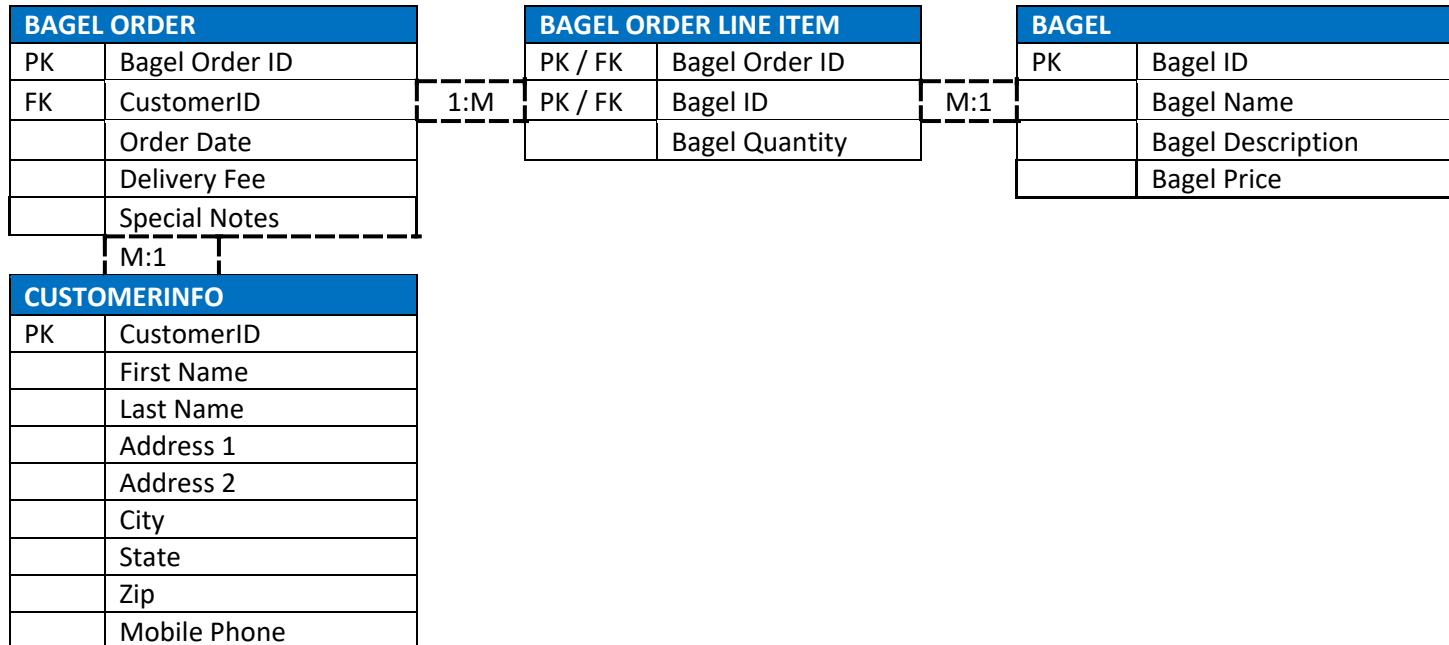


A1c.

I assigned the attributes in the following manner so that all non-key attributes are functionally dependent on the primary key of each table. The Bagel Order Table attributes were placed related to the table where they are dependent on the primary key. The cardinality between the Bagel Order table and the Bagel Order Line Item table is one to many. One order can have many line items. Many line items can have one order, but not the other way around. The cardinality between the Bagel Order Line Item Table and Bagel is many to one. Bagel Order Line Item can have one bagel by bagel ID, but bagel can have many bagel order line items. The Bagel Order Line Item table attribute Bagel Quantity is dependent on the Bagel Order ID and the Bagel ID.

A2.

### Third Normal Form (3NF)

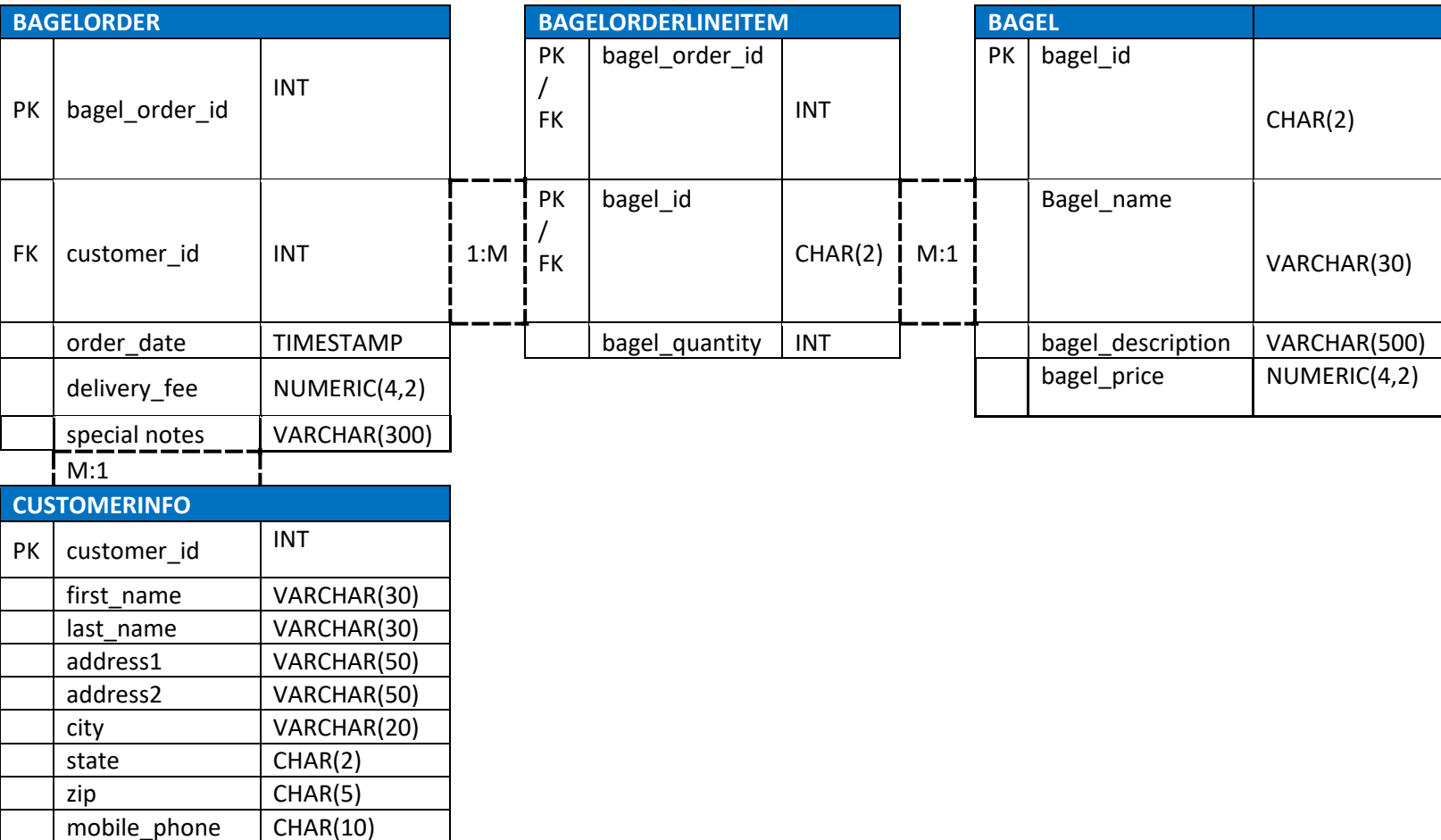


A2e.

To get the tables in third normal form, the bagel order table was split into two tables, Bagel Order and CustomerInfo. In the CustomerInfo table a primary key of CustomerID was created and the attributes about the customer were placed in that table. The cardinality between Bagel Order and Bagel Order Item remained the same. The cardinality between Bagel Order Item and Bagel also remained the same. The new table cardinality between CustomerInfo and Bagel Order is many to one. A customer can place many bagel orders, but only one CustomerID can be assigned per customer.

A3.

**Final Physical Database Model**



B1.

```
CREATE TABLE COFFEE_SHOP (
```

```
    shop_id INT,
```

```
    shop_name VARCHAR(50),
```

```
    city VARCHAR(50),
```

```
    state CHAR(2),
```

```
    PRIMARY KEY (shop_id)
```

```
);
```

```
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)
```

```
);
```

```
    CREATE TABLE SUPPLIER (
```

```
    supplier_id INT,
```

```
    company_name VARCHAR(50),
```

```
    country VARCHAR(30),
```

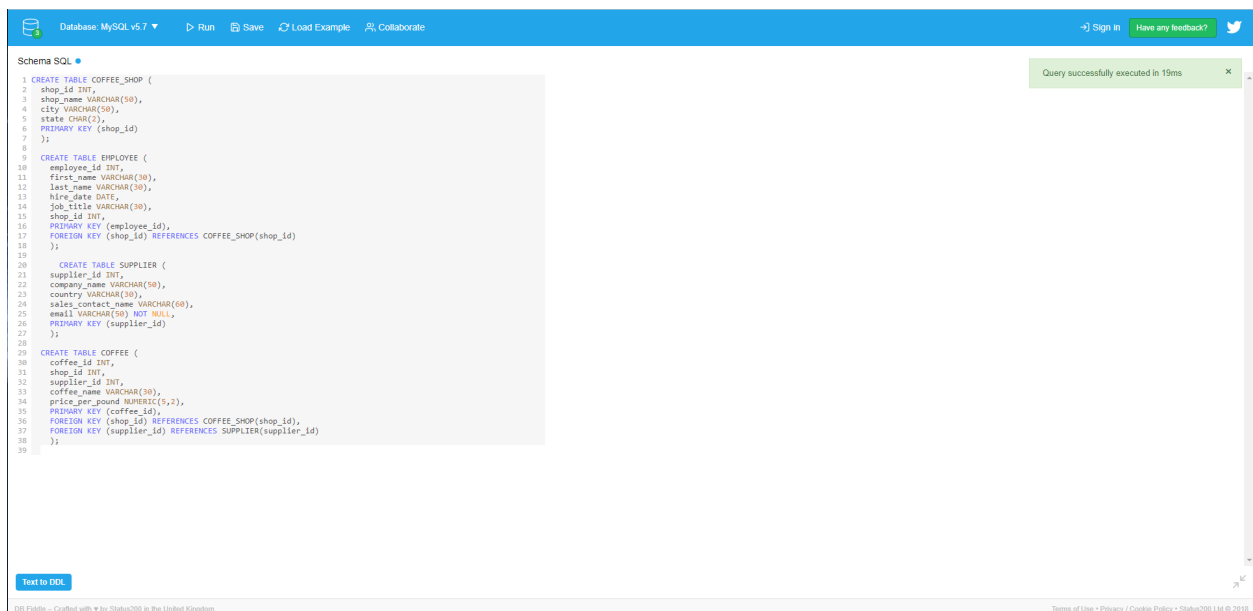
```
    sales_contact_name VARCHAR(60),
```

```
    email VARCHAR(50) NOT NULL,
```

```
    PRIMARY KEY (supplier_id)
```

```
);
```

```
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)  
  
);
```



The screenshot shows a web-based MySQL SQL editor. The top bar is blue with a MySQL logo, the text "Database: MySQL v5.7", and buttons for "Run", "Save", "Load Example", and "Collaborate". On the right of the top bar are links for "Sign in" and "Have any feedback?". Below the top bar, the main area is titled "Schema SQL" and contains a SQL script. The script defines three tables: COFFEE\_SHOP, EMPLOYEE, and SUPPLIER, and then creates the COFFEE table with foreign key constraints. A green notification box on the right says "Query successfully executed in 19ms". At the bottom left, there is a "Text to DDL" button. The footer contains the text "100 Fiddle - Crafted with ♥ by Statu200 in the United Kingdom" and "Terms of Use • Privacy / Cookie Policy • Statu200 Ltd © 2018".

```
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  
9 CREATE TABLE EMPLOYEE (  
10  employee_id INT,  
11  first_name VARCHAR(30),  
12  last_name VARCHAR(30),  
13  hire_date DATE,  
14  job_title VARCHAR(30),  
15  shop_id INT,  
16  PRIMARY KEY (employee_id),  
17  FOREIGN KEY (shop_id) REFERENCES COFFEE_SHOP(shop_id)  
18 );  
19  
20 CREATE TABLE SUPPLIER (  
21  supplier_id INT,  
22  company_name VARCHAR(50),  
23  country VARCHAR(30),  
24  sales_contact_name VARCHAR(50),  
25  email VARCHAR(50) NOT NULL,  
26  PRIMARY KEY (supplier_id)  
27 );  
28  
29 CREATE TABLE COFFEE (  
30  coffee_id INT,  
31  shop_id INT,  
32  supplier_id INT,  
33  coffee_name VARCHAR(30),  
34  price_per_pound NUMERIC(5,2),  
35  PRIMARY KEY (coffee_id),  
36  FOREIGN KEY (shop_id) REFERENCES COFFEE_SHOP(shop_id),  
37  FOREIGN KEY (supplier_id) REFERENCES SUPPLIER(supplier_id)  
38 );  
39
```

B2.

-- Insert Coffee Shop Values --

```
INSERT INTO COFFEE_SHOP
VALUES (103,"Cuppa Joe's", "Flint", "MI");

INSERT INTO COFFEE_SHOP
VALUES (5, "Excess Shots", "Clarkston", "MI");

INSERT INTO COFFEE_SHOP
VALUES (101, "The Split Bean", "Pontiac", "MI");
```

-- Insert Employee Values --

```
INSERT INTO EMPLOYEE
VALUES (0, 'John', 'Doe', '2022-01-01', 'Owner', 103);

INSERT INTO EMPLOYEE
VALUES (1, 'Casey', 'Smith', '2022-02-05', 'Manager', 103);

INSERT INTO EMPLOYEE
VALUES (2, 'Samantha', 'Johnson', '2022-01-01', 'Barista', 103);
```

-- Insert Supplier Values --

```
INSERT INTO SUPPLIER
VALUES (0, 'Mean Beans', 'United States', 'Charlie West', 'cwest@meanbeans.com');

INSERT INTO SUPPLIER
VALUES (1, 'Biggies Beans', 'United States', 'Rachele Zawacki', 'Rachelez@biggiesbeans.net');

INSERT INTO SUPPLIER
VALUES (2, 'Andres Coffee Supply', 'Columbia', 'Andres Esteban', 'Andresesteban@andrescoffee.com');
```

-- Insert Coffee Values --

INSERT INTO COFFEE

VALUES (0, 103, 0, 'Dark house blend', '19.99');

INSERT INTO COFFEE

VALUES (1, 103, 2, 'Columbian Gold', '25.99');

INSERT INTO COFFEE

VALUES (2, 103, 1, 'Darkpast Dark Roast', '23.99');

The screenshot displays the DB Fiddle interface, a web-based SQL playground. The interface is divided into several sections:

- Left Sidebar:** Contains a 'Fiddle Title' field, a 'Fiddle Description' field, a 'Private Fiddle' toggle, an 'Upgrade to PRO' button, and a 'Show Keyboard Shortcuts' link.
- Schema SQL:** A text area containing the following SQL code:

```
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  
9  
10 CREATE TABLE EMPLOYEE (  
11   employee_id INT,  
12   first_name VARCHAR(30),  
13   last_name VARCHAR(30),  
14   hire_date DATE,  
15   job_title VARCHAR(30),  
16   shop_id INT,  
17   PRIMARY KEY (employee_id),  
18   FOREIGN KEY (shop_id) REFERENCES COFFEE_SHOP(shop_id)  
19 );  
20  
21 CREATE TABLE SUPPLIER (  
22   supplier_id INT,  
23   company_name VARCHAR(50),
```
- Query SQL:** A text area containing the following SQL code:

```
1 -- Insert Coffee Shop Values --  
2  
3 INSERT INTO COFFEE_SHOP  
4 VALUES (0, 'Cuppa Joe's', 'Flint', 'MI');  
5 INSERT INTO COFFEE_SHOP  
6 VALUES (1, 'Excess Shots', 'Clarkston', 'MI');  
7 INSERT INTO COFFEE_SHOP  
8 VALUES (101, 'The Split Bean', 'Pontiac', 'MI');  
9  
10 -- Insert Employee Values --  
11  
12 INSERT INTO EMPLOYEE  
13 VALUES (0, 'John', 'Doe', '2022-01-01', 'Owner', 103);  
14 INSERT INTO EMPLOYEE  
15 VALUES (1, 'Cassy', 'Smith', '2022-02-05', 'Manager', 103);  
16 INSERT INTO EMPLOYEE  
17 VALUES (2, 'Samantha', 'Johnson', '2022-01-01', 'Barista', 103);  
18  
19 -- Insert Supplier Values --  
20  
21 INSERT INTO SUPPLIER  
22 VALUES (0, 'Mean Beans', 'United States', 'charlie@meanbeans.com');
```
- Results:** A section showing the execution results of five queries. Each query is listed with its execution time and a message: 'There are no results to be displayed.'
  - Query #1: Execution time: 1ms. There are no results to be displayed.
  - Query #2: Execution time: 1ms. There are no results to be displayed.
  - Query #3: Execution time: 0ms. There are no results to be displayed.
  - Query #4: Execution time: 1ms. There are no results to be displayed.
  - Query #5: Execution time: 0ms. There are no results to be displayed.
- Bottom:** A footer containing the text 'DB Fiddle - Crafted with ❤ by Status200 in the United Kingdom' and a link to 'Terms of Use • Privacy • Cookie Policy • Status200 Ltd © 2019'.

B3.

CREATE VIEW employee\_full\_name\_table AS

SELECT employee\_id, CONCAT(first\_name, " ", last\_name) AS employee\_full\_name, hire\_date, job\_title,  
shop\_id

FROM EMPLOYEE;

```
Schema SQL
1 CREATE TABLE COFFEE_SHOP (
2   shop_id INT,
3   coffee_name VARCHAR(100),
4   city VARCHAR(100),
5   coffee_image VARCHAR(100),
6   PRIMARY KEY (shop_id)
7 );
8
9 CREATE TABLE EMPLOYEE (
10  employee_id INT,
11  first_name VARCHAR(50),
12  last_name VARCHAR(50),
13  hire_date DATE,
14  job_title VARCHAR(100),
15  shop_id INT,
16  FOREIGN KEY (shop_id) REFERENCES COFFEE_SHOP(shop_id)
17 );
18
19 CREATE TABLE SUPPLIER (
20  supplier_id INT,
21  company_name VARCHAR(100),
22  PRIMARY KEY (supplier_id)
23 );
24
25 Query SQL
26 VALUES (1, 'Blue Bunnies', 'United States', 'United States', 'bunnies@bluebunnies.com');
27 INSERT INTO SUPPLIER
28 VALUES (1, 'Ragtime Beans', 'United States', 'Ragtime Beans', 'ragtimebeans@ragtimebeans.net');
29 INSERT INTO EMPLOYEE
30 VALUES (1, 'Samuel', 'Columbian', 'Columbian', 'samuel@ragtimebeans.net');
31 VALUES (2, 'Andrey Coffee Supply', 'Colombia', 'Andrey Esteban', 'andreyesteban@andreycoffee.com');
32
33 -- Insert Coffee Values --
34
35 INSERT INTO COFFEE
36 VALUES (1, 100, 1, 'Dark House Blend', '18.99');
37 INSERT INTO COFFEE
38 VALUES (1, 101, 1, 'Columbian Gold', '22.99');
39 INSERT INTO COFFEE
40 VALUES (1, 102, 1, 'Dark House Blend', '22.99');
41
42 -- Create View with the employee table showing all columns, but concat each employee's first name and last name with a space between and assign the new
43 attribute employee_full_name --
44 CREATE VIEW employee_full_name_table AS
45 SELECT employee_id, CONCAT(first_name, ' ', last_name) AS employee_full_name, hire_date, job_title, shop_id
46 FROM EMPLOYEE;
```

B4.

CREATE INDEX idx\_coffee

ON COFFEE (coffee\_name);

```
Schema SQL
1 CREATE TABLE COFFEE_SHOP (
2   shop_id INT,
3   coffee_name VARCHAR(100),
4   city VARCHAR(100),
5   coffee_image VARCHAR(100),
6   PRIMARY KEY (shop_id)
7 );
8
9 CREATE TABLE EMPLOYEE (
10  employee_id INT,
11  first_name VARCHAR(50),
12  last_name VARCHAR(50),
13  hire_date DATE,
14  job_title VARCHAR(100),
15  shop_id INT,
16  FOREIGN KEY (shop_id) REFERENCES COFFEE_SHOP(shop_id)
17 );
18
19 CREATE TABLE SUPPLIER (
20  supplier_id INT,
21  company_name VARCHAR(100),
22  PRIMARY KEY (supplier_id)
23 );
24
25 Query SQL
26 VALUES (1, 'Andrey Coffee Supply', 'Colombia', 'Andrey Esteban', 'andreyesteban@andreycoffee.com');
27 -- Insert Coffee Values --
28
29 INSERT INTO COFFEE
30 VALUES (1, 100, 1, 'Dark House Blend', '18.99');
31 INSERT INTO COFFEE
32 VALUES (1, 101, 1, 'Columbian Gold', '22.99');
33 INSERT INTO COFFEE
34 VALUES (1, 102, 1, 'Dark House Blend', '22.99');
35
36 -- Create View with the employee table showing all columns, but concat each employee's first name and last name with a space between and assign the new
37 attribute employee_full_name --
38 CREATE VIEW employee_full_name_table AS
39 SELECT employee_id, CONCAT(first_name, ' ', last_name) AS employee_full_name, hire_date, job_title, shop_id
40 FROM EMPLOYEE;
```



B5.

SELECT \* FROM SUPPLIER WHERE supplier\_id = 0;

The screenshot shows the Fiddle SQL editor interface. On the left, there's a sidebar with 'Fiddle Title', 'Fiddle Description', and 'Private Fiddle' (set to 'Public'). The main area is divided into two panes. The left pane shows a SQL schema with three tables: COFFEE\_SHOP, EMPLOYEE, and SUPPLIER. The right pane shows a SQL query that inserts data into these tables and then selects all records from the SUPPLIER table where supplier\_id is 0. The query is executed successfully, and the results are displayed in a table below. The results table has five columns: supplier\_id, company\_name, country, sales\_contact\_name, and email. The only row shown is for supplier\_id 0, with company\_name 'Mean Beans', country 'United States', sales\_contact\_name 'Charlie West', and email 'cwest@meanbeans.com'.

```
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(50),
11  last_name VARCHAR(50),
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(50),
22   sales_contact_name VARCHAR(60),
23 );

Query SQL
23 INSERT INTO SUPPLIER
24 VALUES (1, 'Riggles Beans', 'United States', 'Rachele Zawacki', 'rzawacki@rigglesbeans.com');
25 INSERT INTO SUPPLIER
26 VALUES (2, 'Andres Coffee Supply', 'Columbia', 'Andres Esteban', 'Andresesteban@andrescoffee.com');
27
28 -- Insert Coffee Values --
29
30 INSERT INTO COFFEE
31 VALUES (0, 103, 0, 'Dark house blend', '19.99');
32 INSERT INTO COFFEE
33 VALUES (1, 103, 2, 'Columbian Gold', '25.99');
34 INSERT INTO COFFEE
35 VALUES (2, 103, 1, 'Darkpast Dark Roast', '23.99');
36
37 CREATE VIEW employee_full_name_table AS
38 SELECT employee_id, CONCAT(first_name, ' ', last_name) AS employee_full_name, hire_date, job_title, shop_id
39 FROM EMPLOYEE;
40
41 CREATE INDEX idx_coffee
42 ON COFFEE (coffee_name);
43
44 SELECT * FROM SUPPLIER WHERE supplier_id = 0;
```

supplier_id	company_name	country	sales_contact_name	email
0	Mean Beans	United States	Charlie West	cwest@meanbeans.com

B6.

SELECT EMPLOYEE.employee\_id,  
EMPLOYEE.first\_name,  
EMPLOYEE.last\_name,  
EMPLOYEE.shop\_id,  
COFFEE\_SHOP.shop\_name,  
COFFEE.coffee\_id,  
COFFEE.supplier\_id,  
COFFEE.coffee\_name,  
COFFEE.price\_per\_pound  
FROM EMPLOYEE  
INNER JOIN COFFEE\_SHOP ON EMPLOYEE.shop\_id = COFFEE\_SHOP.shop\_id  
INNER JOIN COFFEE ON COFFEE\_SHOP.shop\_id = COFFEE.shop\_id;

Database: MySQL v8.7

[Databases](#)
[MySQL v8.7](#)
[Tables](#)
[Tools](#)
[User Interface](#)
[All Content](#)

[Help](#)
[Have any feedback?](#)

Table Info

Schema SQL

```

1 CREATE TABLE COFFEE_SHOP (
2   shop_id INT,
3   shop_name VARCHAR(40),
4   city VARCHAR(100),
5   status CHAR(2),
6   PRIMARY KEY (shop_id)
7 );

8
9
10 CREATE TABLE EMPLOYEE (
11   empidemp_id INT,
12   first_name VARCHAR(30),
13   last_name VARCHAR(40),
14   hire_date DATE,
15   job_id VARCHAR(10),
16   shop_id INT,
17   PRIMARY KEY (emp_id),
18   FOREIGN KEY (shop_id) REFERENCES COFFEE_SHOP(shop_id)
19 );

20
21 CREATE TABLE SUPPLIER (
22   supplier_id INT,
23   supplier_name VARCHAR(40),
24 );

```

Query SQL

Query successfully executed in 51ms

```

-- Create an index on coffee_name field from the coffee table --
--
-- Create a SELECT FROM/WHERE (SPK) query --
--
-- Create a table join query with three tables and includes wildcards from all three tables in its output --
--
14 SELECT EMPLOYEE.empidemp_id,
15   EMPLOYEE.first_name,
16   EMPLOYEE.last_name,
17   EMPLOYEE.shop_id,
18   COFFEE_SHOP.shop_name,
19   COFFEE_SHOP.status,
20   COFFEE_SHOP.city,
21   COFFEE_SHOP.supplier_id,
22   COFFEE_SHOP.supplier_name,
23   COFFEE_SHOP.city AS COFFEE_SHOP_CITY,
24   FROM EMPLOYEE
25   INNER JOIN COFFEE_SHOP ON EMPLOYEE.shop_id = COFFEE_SHOP.shop_id
26   INNER JOIN COFFEE_SHOP ON COFFEE_SHOP.shop_id = COFFEE_SHOP.shop_id

```

360 databases showing

Private Mode

The string query you modified after using the table

Upgrade to PRO!

50% OFF for Early Adopters

See to SQL

Copy to Markdown

Results

8	John	Dee	103	Cuppa Joe's	1	2	Columbian Gold	25.99
9	John	Dee	103	Cuppa Joe's	2	1	Darknet Dark Roast	23.99
1	Cassy	Smith	103	Cuppa Joe's	5	0	Dark house Blend	19.99
1	Cassy	Smith	103	Cuppa Joe's	1	2	Columbian Gold	25.99
1	Cassy	Smith	103	Cuppa Joe's	2	1	Darknet Dark Roast	23.99
2	Samantha	Johnson	103	Cuppa Joe's	6	0	Dark house Blend	19.99
2	Samantha	Johnson	103	Cuppa Joe's	1	2	Columbian Gold	25.99
2	Samantha	Johnson	103	Cuppa Joe's	2	1	Darknet Dark Roast	23.99

MySQL

MySQL is the world's most popular open-source database.

The cloud experience that makes developers happy

MySQL Cloud

MySQL Cloud

MySQL Cloud is the world's most popular open-source database.