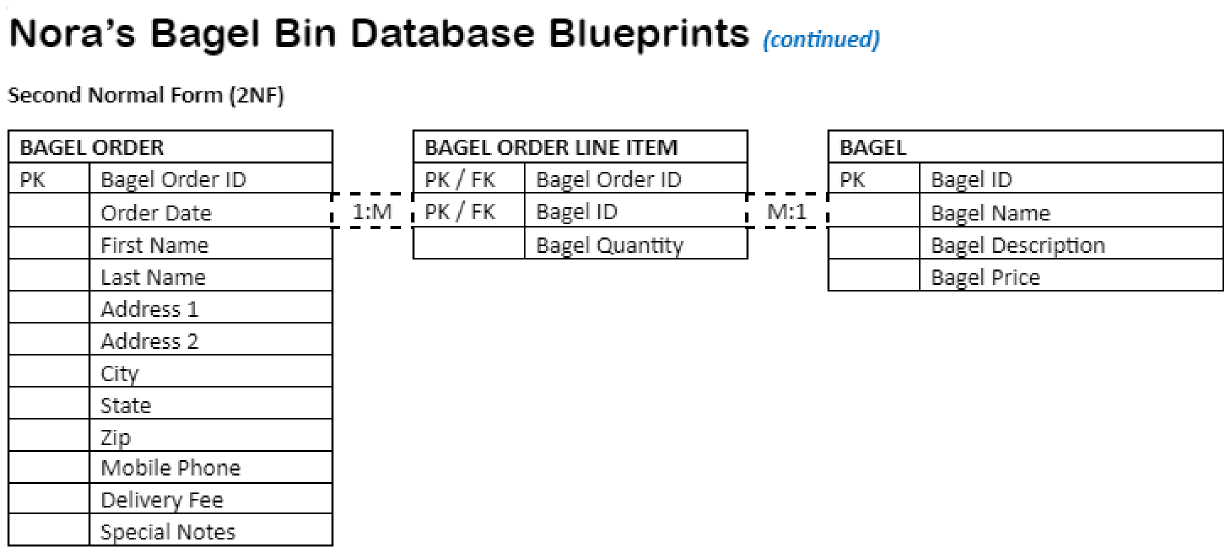
Alexandra Beno
C170
Performance Assessment:Data Management - Applications (VHT2)

A1ab.



Nora's Bagel Bin Database Blueprints *(continued)*

Second Normal Form (2NF)

BAGEL ORDER	
PK	Bagel Order ID
	Order Date
	First Name
	Last Name
	Address 1
	Address 2
	City
	State
	Zip
	Mobile Phone
	Delivery Fee
	Special Notes

1:M

BAGEL ORDER LINE ITEM	
PK / FK	Bagel Order ID
PK / FK	Bagel ID
	Bagel Quantity

M:1

BAGEL	
PK	Bagel ID
	Bagel Name
	Bagel Description
	Bagel Price

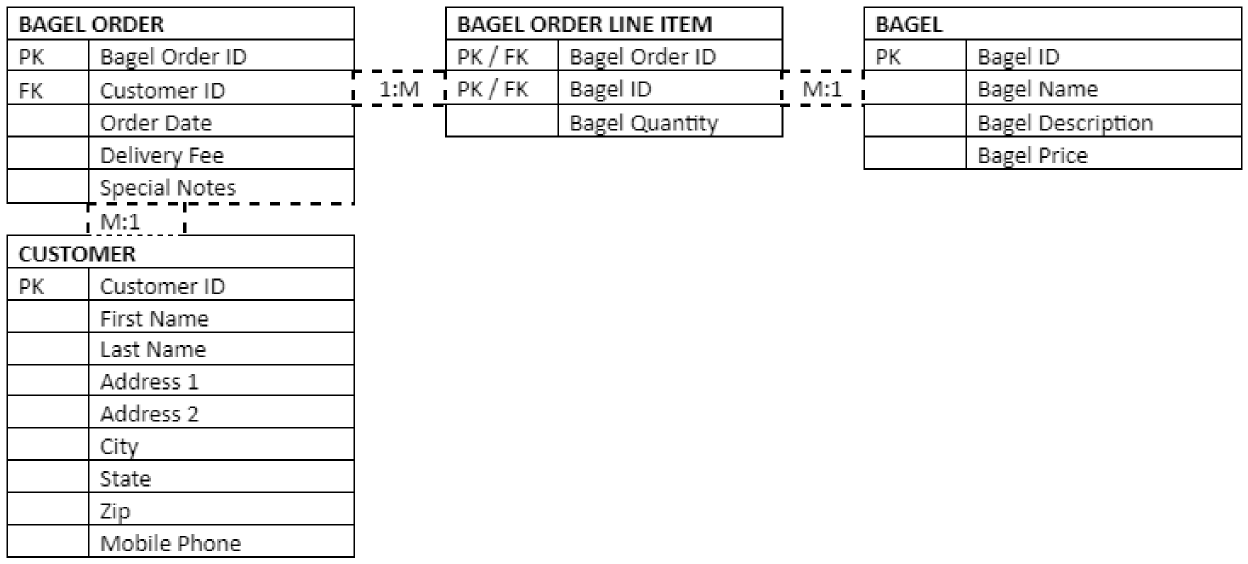
A1c.

I assigned attributes to the 2NF tables by determining which attributes were associated with which primary keys. For example, the Bagel Description is an attribute associated with a single Bagel, therefore it should be placed in the table associated with a single Bagel ID. Similarly, the Order Date is an attribute associated with a single Bagel Order, therefore it should be placed in the table associated with a single Bagel Order ID. On the other hand, the Bagel Quantity is an attribute which is associated with both a Bagel Order and a Bagel, therefore it should be placed in the table associated with both a single Bagel Order ID and a single Bagel ID.

I determined the cardinality of the relationship between the Bagel Order table and the Bagel Order Line Item table to be one-to-many because a Bagel Order can potentially be associated with multiple Bagel Order Line Items. I determined the cardinality of the relationship between the Bagel table and the Bagel Order Line Item table to be one-to-many because multiple Bagel Order Line items can be associated with a single Bagel.

A2abcd.

Third Normal Form (3NF)



BAGEL ORDER	
PK	Bagel Order ID
FK	Customer ID
	Order Date
	Delivery Fee
	Special Notes

BAGEL ORDER LINE ITEM	
PK / FK	Bagel Order ID
PK / FK	Bagel ID
	Bagel Quantity

BAGEL	
PK	Bagel ID
	Bagel Name
	Bagel Description
	Bagel Price

CUSTOMER	
PK	Customer ID
	First Name
	Last Name
	Address 1
	Address 2
	City
	State
	Zip
	Mobile Phone

A2e.

Normalizing to 3rd Normal Form required the creation of a new primary key, the Customer ID. This was necessary because many of the attributes associated with the Bagel Order in 2nd Normal Form didn't only depend on the Bagel Order ID. Attributes like First Name or Mobile Phone could show up repeatedly in many different Bagel Orders. By creating the Customer table and Customer ID, the number of times information about a single customer would need to be entered is significantly reduced.

I assigned attributes to the 3NF tables by determining which attributes were associated with which primary keys. For example, the Bagel Description is an attribute associated with a single Bagel, therefore it should be placed in the table associated with a single Bagel ID. Similarly, the Order Date is an attribute associated with a single Bagel Order, therefore it should be placed in the table associated with a single Bagel Order ID. The First Name is an attribute associated with a single Customer, and therefore it should be placed into the table associated with a single customer ID. On the other hand, the Bagel Quantity is an attribute which is associated with both a Bagel Order and a Bagel, therefore it should be placed in the table associated with both a single Bagel Order ID and a single Bagel ID. Attributes like First Name and Last Name are associated with a single Customer, and therefore should be placed in the table associated with a single Customer ID.

I determined the cardinality of the relationship between the Bagel Order table and the Bagel Order Line Item table to be one-to-many because a Bagel Order can potentially be associated with multiple Bagel Order Line Items. I determined the cardinality of the relationship between the Bagel table and the Bagel Order Line Item table to be one-to-many because multiple Bagel Order Line items can be associated with a single Bagel. Similarly, I determined the cardinality of the relationship between the Customer table and the Bagel Order table to be one-to-many because multiple Bagel Orders can be associated with a single Customer.

A3ab.

Final Physical Database Model

BAGEL ORDER		
PK	bagel_order_id	INTEGER
FK	customer_id	INTEGER
	order_date	TIMESTAMP
	delivery_fee	NUMERIC(3,2)
	special_notes	VARCHAR(160)

1:M

BAGEL ORDER LINE ITEM		
PK / FK	bagel_order_id	INTEGER
PK / FK	bagel_id	CHAR(2)
	bagel_quant	INTEGER

M:1

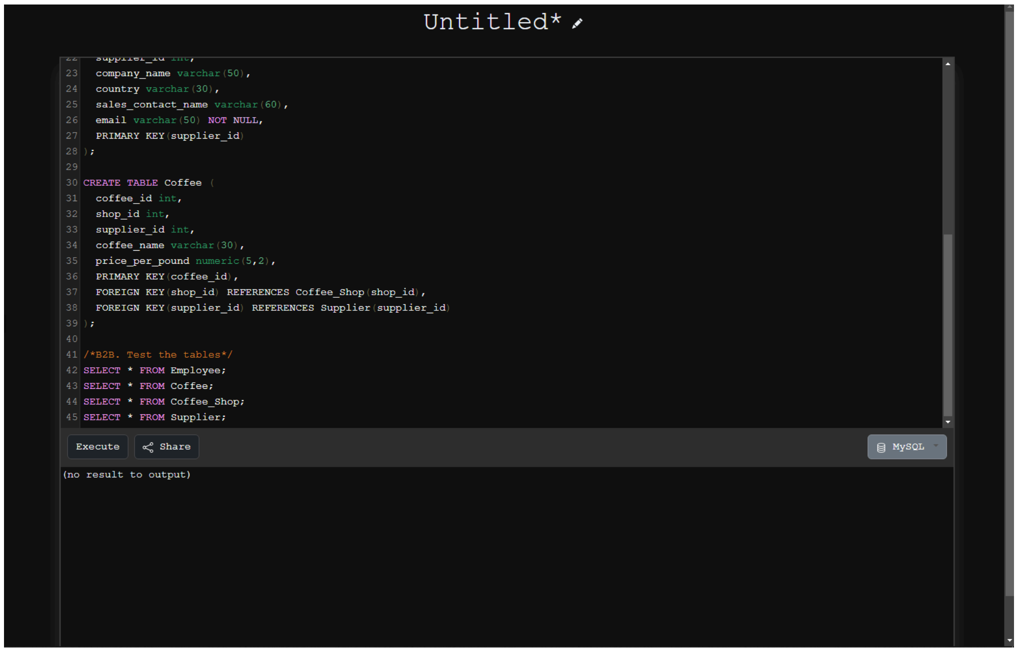
BAGEL		
PK	bagel_id	CHAR(2)
	bagel_name	VARCHAR(25)
	bagel_desc	VARCHAR(160)
	bagel_price	NUMERIC(3,2)

M:1

CUSTOMER		
PK	customer_id	INTEGER
	first_name	VARCHAR(25)
	last_name	VARCHAR(25)
	address1	VARCHAR(40)
	address2	VARCHAR(40)
	city	VARCHAR(25)
	state	CHAR(2)
	zip	CHAR(5)
	mobile_phone	CHAR(10)

```
/* B1a. Create all the tables*/
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)
);
```

```
/*B1b. Test the tables*/
SELECT * FROM Employee;
SELECT * FROM Coffee;
SELECT * FROM Coffee_Shop;
SELECT * FROM Supplier;
```

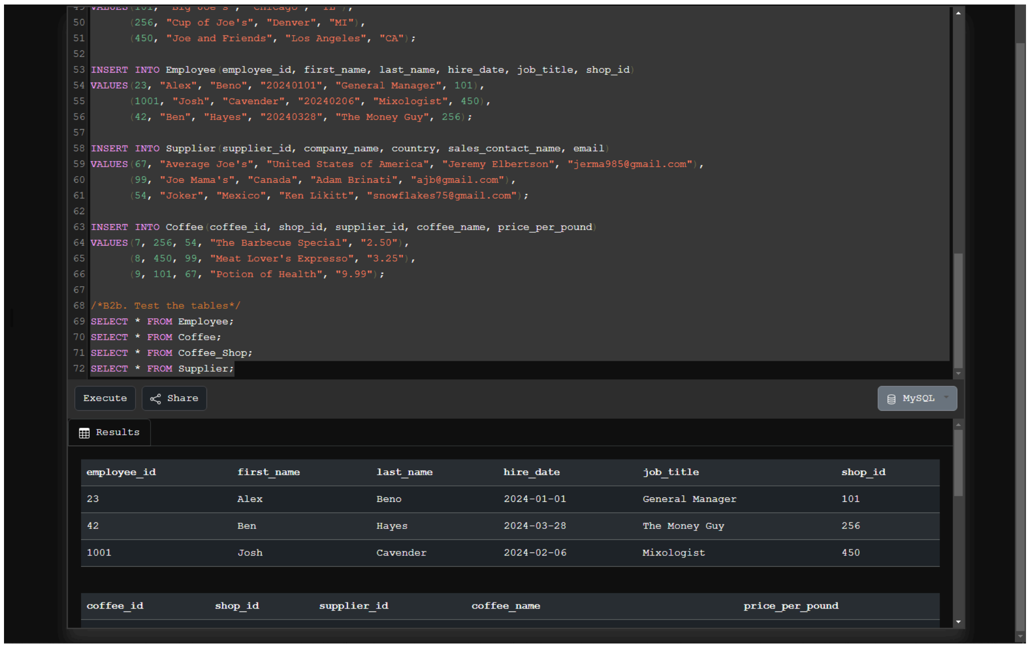


```
  supplier_id int,
  company_name varchar(50),
  country varchar(30),
  sales_contact_name varchar(60),
  email varchar(50) NOT NULL,
  PRIMARY KEY(supplier_id)
);

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)
);

/*B2B. Test the tables*/
SELECT * FROM Employee;
SELECT * FROM Coffee;
SELECT * FROM Coffee_Shop;
SELECT * FROM Supplier;
```

```
(no result to output)
```

```
/*B2a. Populate the tables with three rows of data */
INSERT INTO Coffee_Shop(shop_id, shop_name, city, state)
VALUES(101, "Big Joe's", "Chicago", "IL"),
    (256, "Cup of Joe's", "Denver", "MI"),
    (450, "Joe and Friends", "Los Angeles", "CA");

INSERT INTO Employee(employee_id, first_name, last_name, hire_date, job_title, shop_id)
VALUES(23, "Alex", "Beno", "20240101", "General Manager", 101),
    (1001, "Josh", "Cavender", "20240206", "Mixologist", 450),
    (42, "Ben", "Hayes", "20240328", "The Money Guy", 256);

INSERT INTO Supplier(supplier_id, company_name, country, sales_contact_name, email)
VALUES(67, "Average Joe's", "United States of America", "Jeremy Elbertson",
"jerma985@gmail.com"),
    (99, "Joe Mama's", "Canada", "Adam Brinati", "ajb@gmail.com"),
    (54, "Joker", "Mexico", "Ken Likitt", "snowflakes75@gmail.com");

INSERT INTO Coffee(coffee_id, shop_id, supplier_id, coffee_name, price_per_pound)
VALUES(7, 256, 54, "The Barbecue Special", "2.50"),
    (8, 450, 99, "Meat Lover's Expresso", "3.25"),
    (9, 101, 67, "Potion of Health", "9.99");
```

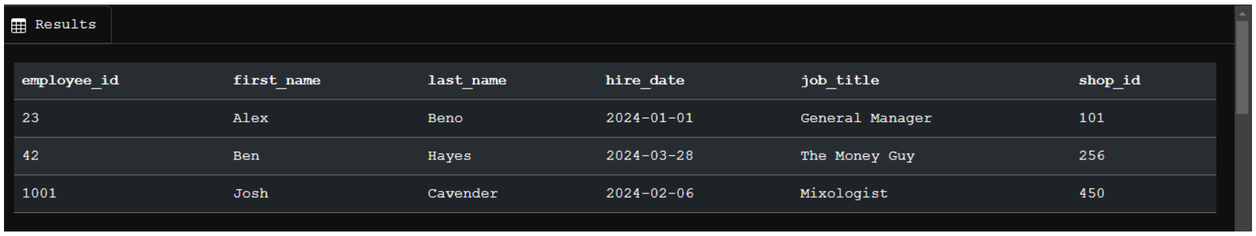
/*B2b. Test the tables*/
SELECT * FROM Employee;
SELECT * FROM Coffee;
SELECT * FROM Coffee_Shop;
SELECT * FROM Supplier;



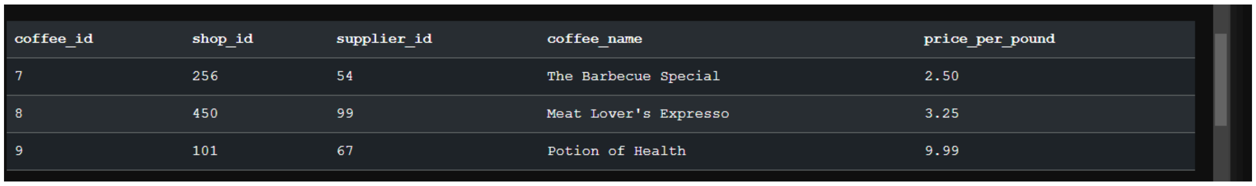
```
       (256, "Cup of Joe's", "Denver", "MI"),
       (450, "Joe and Friends", "Los Angeles", "CA");

INSERT INTO Employee(employee_id, first_name, last_name, hire_date, job_title, shop_id)
VALUES(23, "Alex", "Beno", "20240101", "General Manager", 101),
       (1001, "Josh", "Cavender", "20240206", "Mixologist", 450),
       (42, "Ben", "Hayes", "20240328", "The Money Guy", 256);

INSERT INTO Supplier(supplier_id, company_name, country, sales_contact_name, email)
VALUES(67, "Average Joe's", "United States of America", "Jeremy Elbertson", "jerma985@gmail.com"),
       (99, "Joe Mama's", "Canada", "Adam Brinati", "ajb@gmail.com"),
       (54, "Joker", "Mexico", "Ken Likitt", "snowflakes75@gmail.com");

INSERT INTO Coffee(coffee_id, shop_id, supplier_id, coffee_name, price_per_pound)
VALUES(7, 256, 54, "The Barbecue Special", "2.50"),
       (8, 450, 99, "Meat Lover's Expresso", "3.25"),
       (9, 101, 67, "Potion of Health", "9.99");

/*B2b. Test the tables*/
SELECT * FROM Employee;
SELECT * FROM Coffee;
SELECT * FROM Coffee_Shop;
SELECT * FROM Supplier;
```

employee_id	first_name	last_name	hire_date	job_title	shop_id
23	Alex	Beno	2024-01-01	General Manager	101
42	Ben	Hayes	2024-03-28	The Money Guy	256
1001	Josh	Cavender	2024-02-06	Mixologist	450

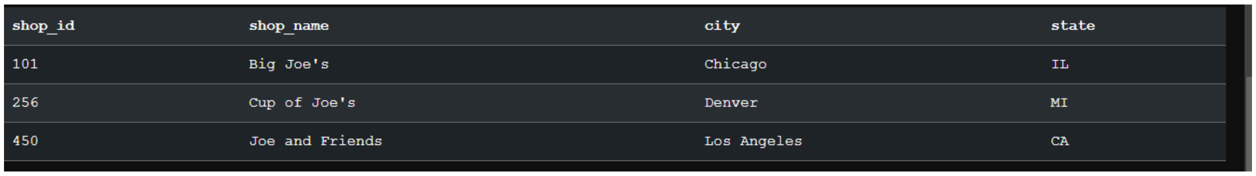
coffee_id	shop_id	supplier_id	coffee_name	price_per_pound



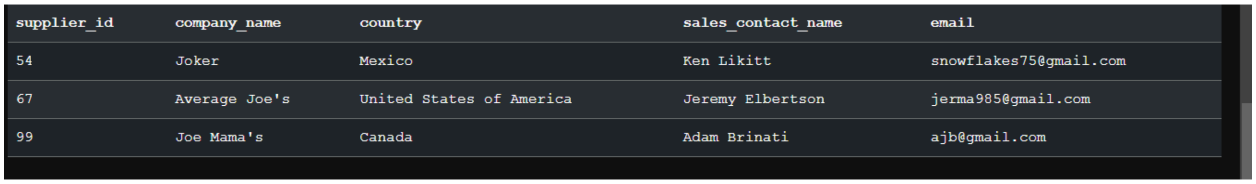
employee_id	first_name	last_name	hire_date	job_title	shop_id
23	Alex	Beno	2024-01-01	General Manager	101
42	Ben	Hayes	2024-03-28	The Money Guy	256
1001	Josh	Cavender	2024-02-06	Mixologist	450



coffee_id	shop_id	supplier_id	coffee_name	price_per_pound
7	256	54	The Barbecue Special	2.50
8	450	99	Meat Lover's Expresso	3.25
9	101	67	Potion of Health	9.99



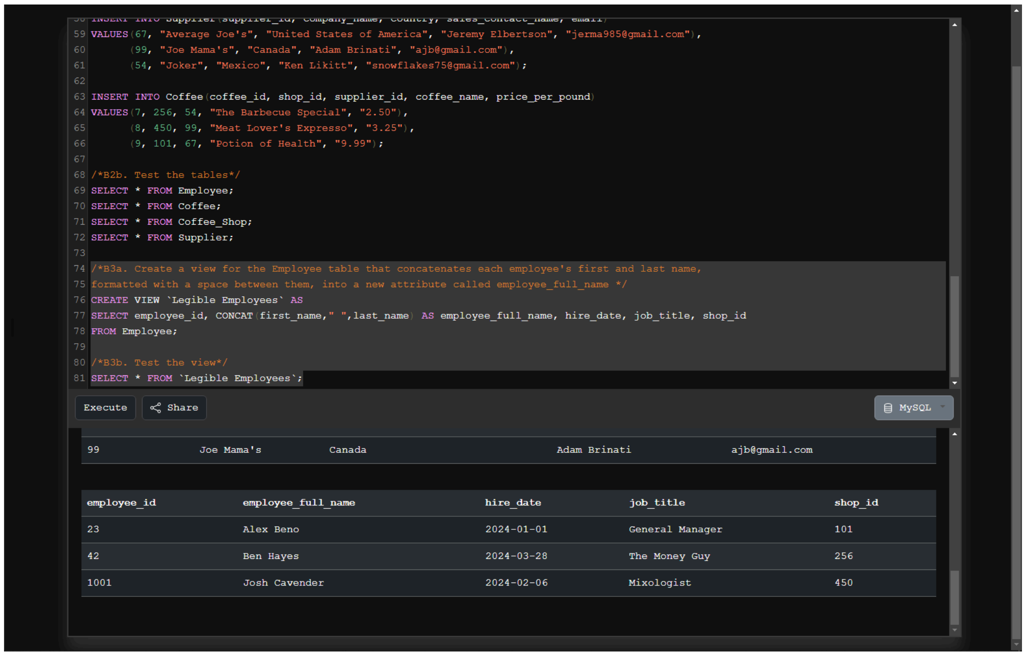
shop_id	shop_name	city	state
101	Big Joe's	Chicago	IL
256	Cup of Joe's	Denver	MI
450	Joe and Friends	Los Angeles	CA



supplier_id	company_name	country	sales_contact_name	email
54	Joker	Mexico	Ken Likitt	snowflakes75@gmail.com
67	Average Joe's	United States of America	Jeremy Elbertson	jerma985@gmail.com
99	Joe Mama's	Canada	Adam Brinati	ajb@gmail.com

/*B3a. Create a view for the Employee table that concatenates each employee's first and last name, formatted with a space between them, into a new attribute called employee_full_name */
CREATE VIEW `Legible Employees` AS
SELECT employee_id, CONCAT(first_name," ",last_name) AS employee_full_name, hire_date, job_title, shop_id
FROM Employee;

/*B3b. Test the view*/
SELECT * FROM `Legible Employees`;



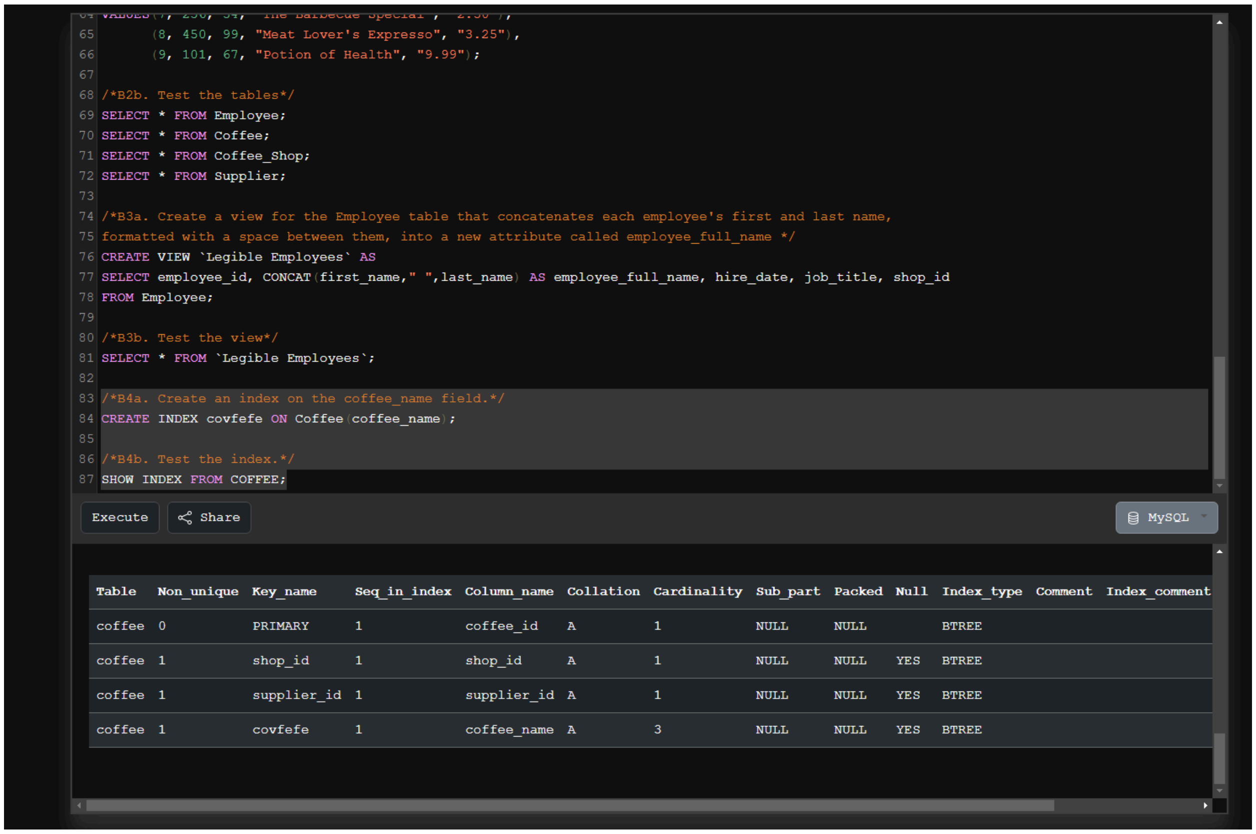
```
VALUES(67, "Average Joe's", "United States of America", "Jeremy Elbertson", "jerma985@gmail.com"),
      (99, "Joe Mama's", "Canada", "Adam Brinati", "ajb@gmail.com"),
      (54, "Joker", "Mexico", "Ken Likitt", "snowflakes75@gmail.com");

INSERT INTO Coffee(coffee_id, shop_id, supplier_id, coffee_name, price_per_pound)
VALUES(7, 256, 54, "The Barbecue Special", "2.50"),
      (8, 450, 99, "Meat Lover's Expresso", "3.25"),
      (9, 101, 67, "Potion of Health", "9.99");

/*B2b. Test the tables*/
SELECT * FROM Employee;
SELECT * FROM Coffee;
SELECT * FROM Coffee_Shop;
SELECT * FROM Supplier;

/*B3a. Create a view for the Employee table that concatenates each employee's first and last name,
formatted with a space between them, into a new attribute called employee_full_name */
CREATE VIEW `Legible Employees` AS
SELECT employee_id, CONCAT(first_name," ",last_name) AS employee_full_name, hire_date, job_title, shop_id
FROM Employee;

/*B3b. Test the view*/
SELECT * FROM `Legible Employees`;
```

99	Joe Mama's	Canada	Adam Brinati	ajb@gmail.com

employee_id	employee_full_name	hire_date	job_title	shop_id
23	Alex Beno	2024-01-01	General Manager	101
42	Ben Hayes	2024-03-28	The Money Guy	256
1001	Josh Cavender	2024-02-06	Mixologist	450

/*B4a. Create an index on the coffee_name field.*/
CREATE INDEX covfefe ON Coffee(coffee_name);

/*B4b. Test the index.*/
SHOW INDEX FROM COFFEE;

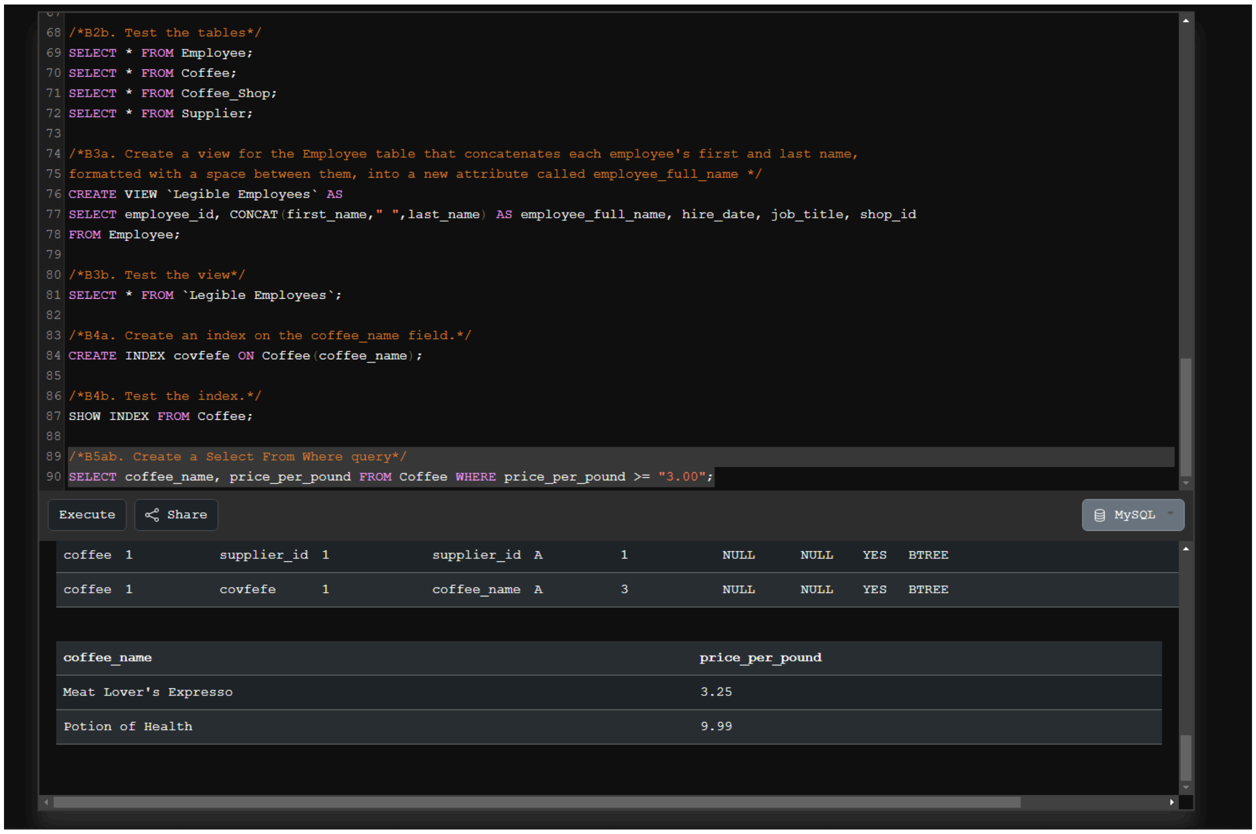


```sql
       (8, 450, 99, "Meat Lover's Expresso", "3.25"),
       (9, 101, 67, "Potion of Health", "9.99");

/*B2b. Test the tables*/
SELECT * FROM Employee;
SELECT * FROM Coffee;
SELECT * FROM Coffee_Shop;
SELECT * FROM Supplier;

/*B3a. Create a view for the Employee table that concatenates each employee's first and last name,
formatted with a space between them, into a new attribute called employee_full_name */
CREATE VIEW `Legible Employees` AS
SELECT employee_id, CONCAT(first_name," ",last_name) AS employee_full_name, hire_date, job_title, shop_id
FROM Employee;

/*B3b. Test the view*/
SELECT * FROM `Legible Employees`;

/*B4a. Create an index on the coffee_name field.*/
CREATE INDEX covfefe ON Coffee(coffee_name);

/*B4b. Test the index.*/
SHOW INDEX FROM COFFEE;
```

Execute Share MySQL

Table	Non_unique	Key_name	Seq_in_index	Column_name	Collation	Cardinality	Sub_part	Packed	Null	Index_type	Comment	Index_comment
coffee	0	PRIMARY	1	coffee_id	A	1	NULL	NULL		BTREE		
coffee	1	shop_id	1	shop_id	A	1	NULL	NULL	YES	BTREE		
coffee	1	supplier_id	1	supplier_id	A	1	NULL	NULL	YES	BTREE		
coffee	1	covfefe	1	coffee_name	A	3	NULL	NULL	YES	BTREE		

/*B5ab. Create a Select From Where query*/
SELECT coffee_name, price_per_pound FROM Coffee WHERE price_per_pound >= "3.00";

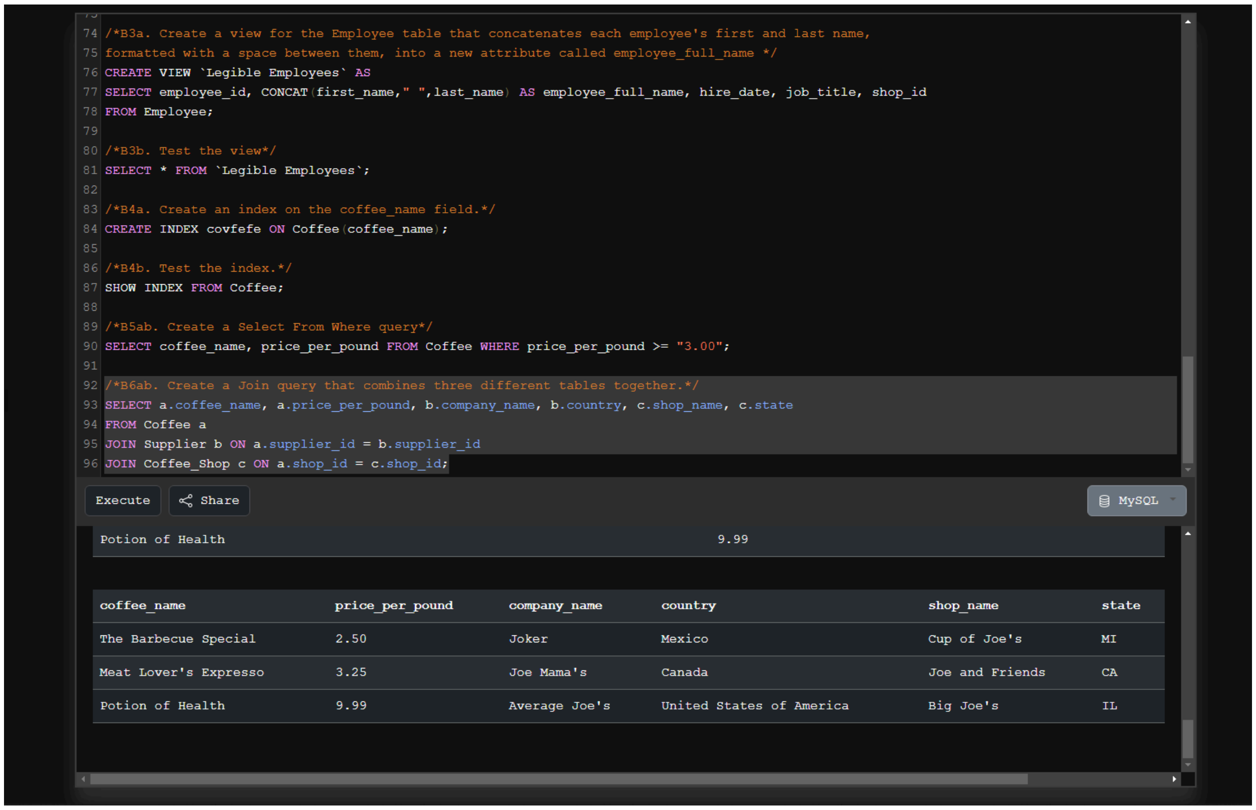


```
/*B2b. Test the tables*/
SELECT * FROM Employee;
SELECT * FROM Coffee;
SELECT * FROM Coffee_Shop;
SELECT * FROM Supplier;

/*B3a. Create a view for the Employee table that concatenates each employee's first and last name,
formatted with a space between them, into a new attribute called employee_full_name */
CREATE VIEW `Legible Employees` AS
SELECT employee_id, CONCAT(first_name," ",last_name) AS employee_full_name, hire_date, job_title, shop_id
FROM Employee;

/*B3b. Test the view*/
SELECT * FROM `Legible Employees`;

/*B4a. Create an index on the coffee_name field.*/
CREATE INDEX covfefe ON Coffee(coffee_name);

/*B4b. Test the index.*/
SHOW INDEX FROM Coffee;

/*B5ab. Create a Select From Where query*/
SELECT coffee_name, price_per_pound FROM Coffee WHERE price_per_pound >= "3.00";
```

```
coffee  1   supplier_id  1   supplier_id  A   1   NULL   NULL   YES   BTREE
coffee  1   covfefe      1   coffee_name  A   3   NULL   NULL   YES   BTREE
```

coffee_name	price_per_pound
Meat Lover's Expresso	3.25
Potion of Health	9.99

/*B6ab. Create a Join query that combines three different tables together.*/
SELECT a.coffee_name, a.price_per_pound, b.company_name, b.country, c.shop_name, c.state
FROM Coffee a
JOIN Supplier b ON a.supplier_id = b.supplier_id
JOIN Coffee_Shop c ON a.shop_id = c.shop_id;



```
/*B3a. Create a view for the Employee table that concatenates each employee's first and last name,
formatted with a space between them, into a new attribute called employee_full_name */
CREATE VIEW `Legible Employees` AS
SELECT employee_id, CONCAT(first_name," ",last_name) AS employee_full_name, hire_date, job_title, shop_id
FROM Employee;

/*B3b. Test the view*/
SELECT * FROM `Legible Employees`;

/*B4a. Create an index on the coffee_name field.*/
CREATE INDEX covfefe ON Coffee(coffee_name);

/*B4b. Test the index.*/
SHOW INDEX FROM Coffee;

/*B5ab. Create a Select From Where query*/
SELECT coffee_name, price_per_pound FROM Coffee WHERE price_per_pound >= "3.00";

/*B6ab. Create a Join query that combines three different tables together.*/
SELECT a.coffee_name, a.price_per_pound, b.company_name, b.country, c.shop_name, c.state
FROM Coffee a
JOIN Supplier b ON a.supplier_id = b.supplier_id
JOIN Coffee_Shop c ON a.shop_id = c.shop_id;
```

Potion of Health	9.99

coffee_name	price_per_pound	company_name	country	shop_name	state
The Barbecue Special	2.50	Joker	Mexico	Cup of Joe's	MI
Meat Lover's Expresso	3.25	Joe Mama's	Canada	Joe and Friends	CA
Potion of Health	9.99	Average Joe's	United States of America	Big Joe's	IL