C170 – Database Management: Applications

Performance Assessment

Shane Short Student ID: 000890378

Project A: Nora's Bagel Bin Database Blueprints

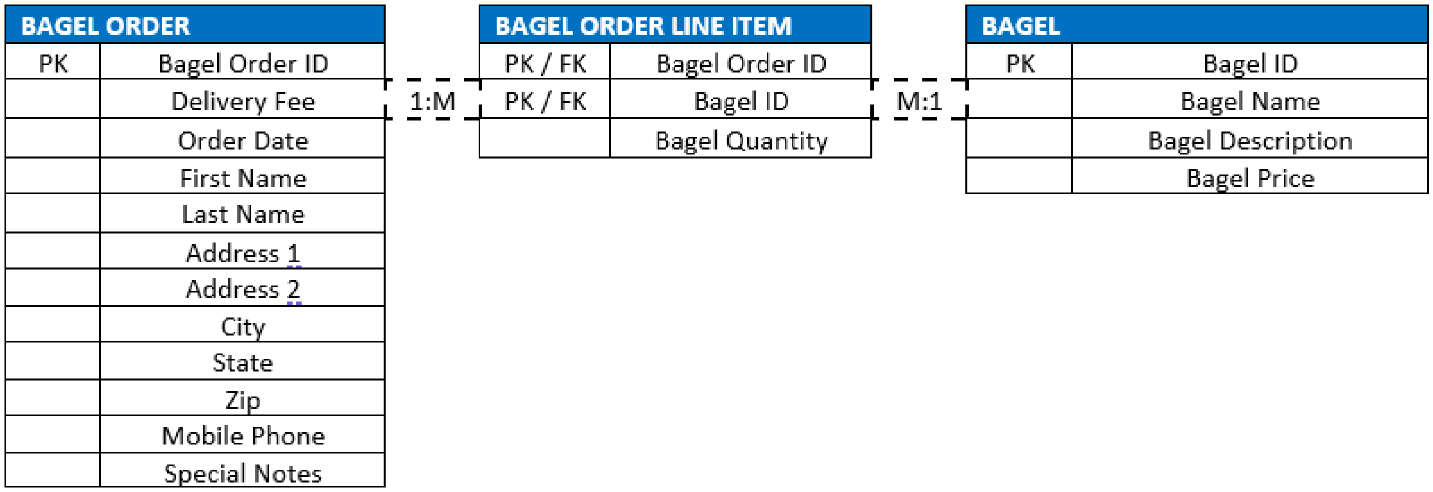
Base Table Provided: First Normal Form (1NF)

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

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

A1. Complete the second normal form (2NF) section of the attached “Nora’s Bagel Bin Database Blueprints”

Second Normal Form (2NF)



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

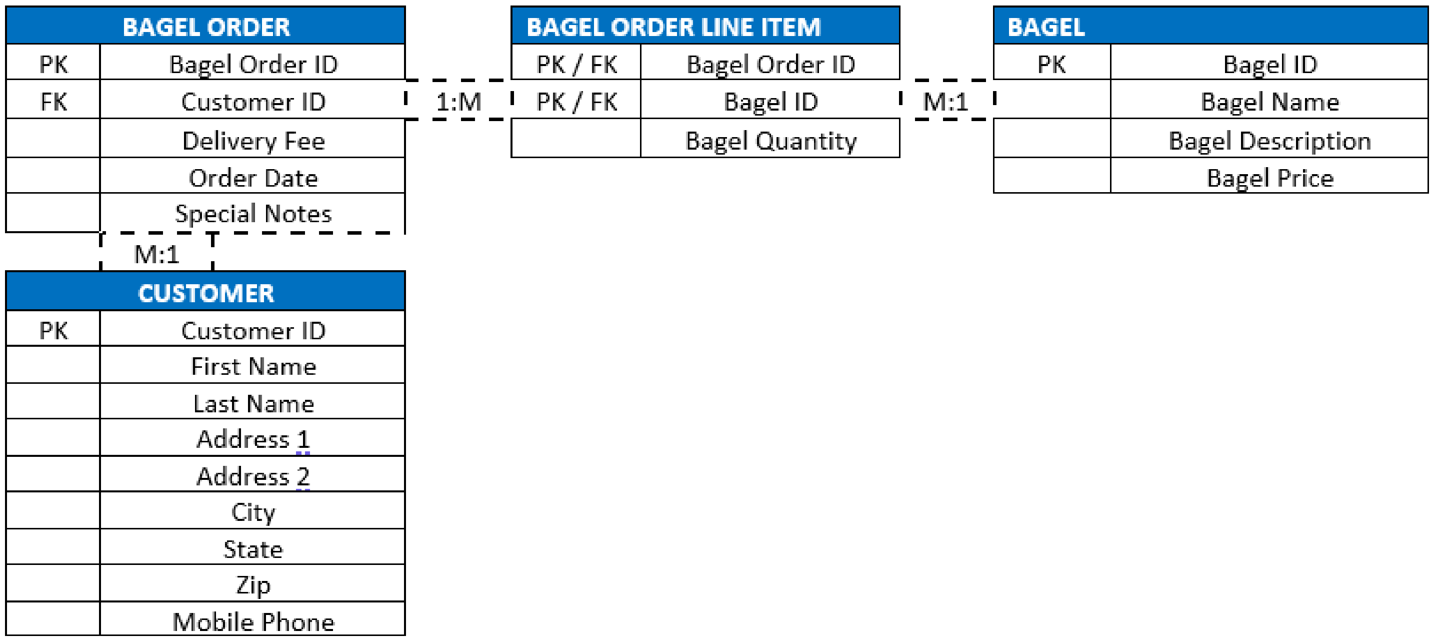
Explanation:

I first assigned the attributes from the 1NF table to new tables based on their dependencies. I separated the composite primary key of the original 1NF table into three tables total. The ‘Bagel Order Line Item’ table is the original 1NF table, using only the Bagel Order ID, Bagel ID, and Delivery Fee as the Bagel Order ID and Bagel ID compose the primary information for the order line item, and the Delivery Fee is dependent on both of those attributes. I created the Bagel table with the attributes from the 1NF table that only depend on the original Primary Key Bagel ID. I created the Bagel Order table with the attributes from the 1NF table that only depend on the original Primary Key ‘Bagel Order ID’. I chose 1:M for the Bagel Order to Bagel Order Line Item relationship since each bagel order can have many line items. I chose M:1 for the Bagel Order Line Item to Bagel relationship since each bagel can have many line items.

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

A2. Complete the third normal form (3NF) section of the attached “Nora’s Bagel Bin Database Blueprints”

Third Normal Form (3NF)



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

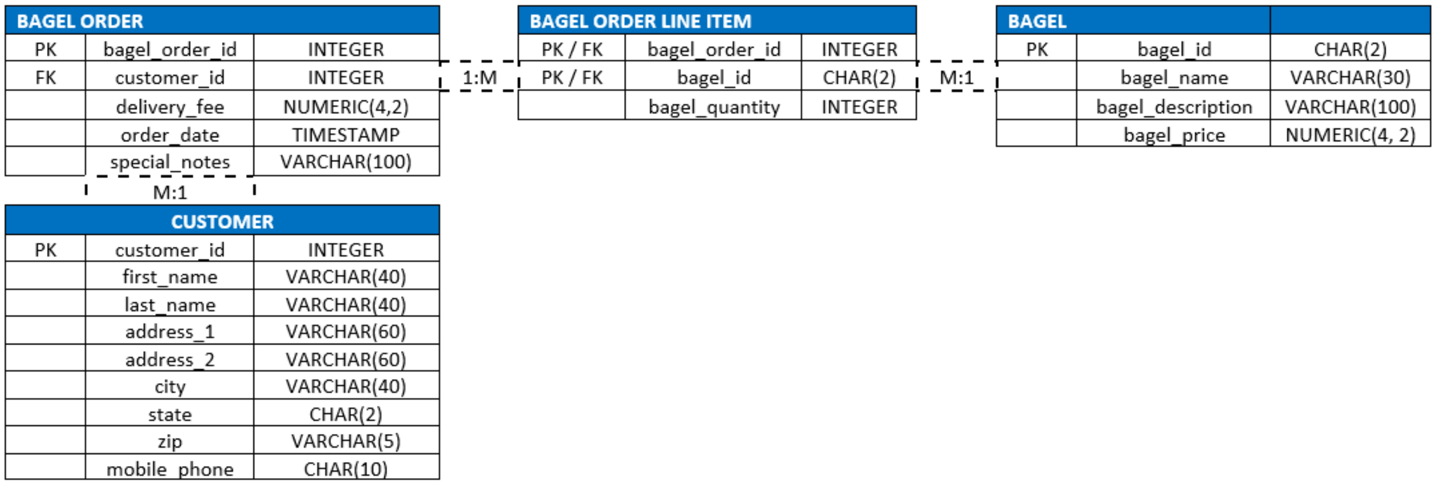
Explanation:

I decomposed the Bagel Order table into two tables, one labeled ‘Bagel Order’ and the other ‘Customer’. I created the new Customer table since the customer information (first name, last name, address, etc.) would be redundant every time the same customer makes an order, which would slow queries. The Bagel Order table now has a Foreign Key (Customer ID) which references the Primary Key (Customer ID) of Customer. Now, in the Bagel Order table, the only attribute that keeps customer information is the Customer ID. The rest of the customer information is inserted one time (unless updated) into the customer information table and is no longer redundant. I chose M:1 to represent the relationship between Bagel Order and Customer since many orders can have one customer, and one customer can make many orders.

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

A3. Complete the "Final Physical Database Model" section of the attached “Nora’s Bagel Bin Database Blueprints”

Final Physical Database Model



BAGEL ORDER

PK	bagel_order_id	INTEGER
FK	customer_id	INTEGER
	delivery_fee	NUMERIC(4,2)
	order_date	TIMESTAMP
	special_notes	VARCHAR(100)

1:M

BAGEL ORDER LINE ITEM

PK / FK	bagel_order_id	INTEGER
PK / FK	bagel_id	CHAR(2)
	bagel_quantity	INTEGER

M:1

BAGEL

PK	bagel_id	CHAR(2)
	bagel_name	VARCHAR(30)
	bagel_description	VARCHAR(100)
	bagel_price	NUMERIC(4, 2)

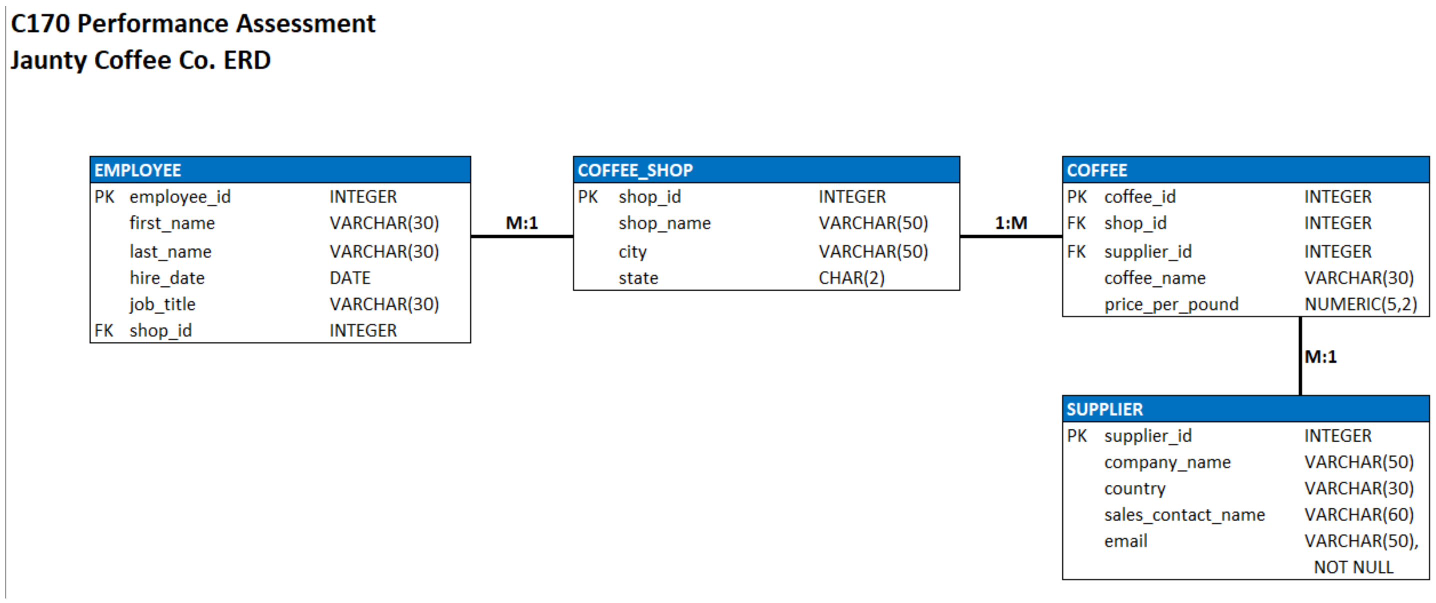
M:1

CUSTOMER

PK	customer_id	INTEGER
	first_name	VARCHAR(40)
	last_name	VARCHAR(40)
	address_1	VARCHAR(60)
	address_2	VARCHAR(60)
	city	VARCHAR(40)
	state	CHAR(2)
	zip	VARCHAR(5)
	mobile_phone	CHAR(10)

Project B: Jaunty Coffee Co. Database Creation

Base Entity Relationship Diagram Provided:


C170 Performance Assessment
Jaunty Coffee Co. ERD

EMPLOYEE		
PK	employee_id	INTEGER
	first_name	VARCHAR(30)
	last_name	VARCHAR(30)
	hire_date	DATE
	job_title	VARCHAR(30)
FK	shop_id	INTEGER

COFFEE_SHOP		
PK	shop_id	INTEGER
	shop_name	VARCHAR(50)
	city	VARCHAR(50)
	state	CHAR(2)

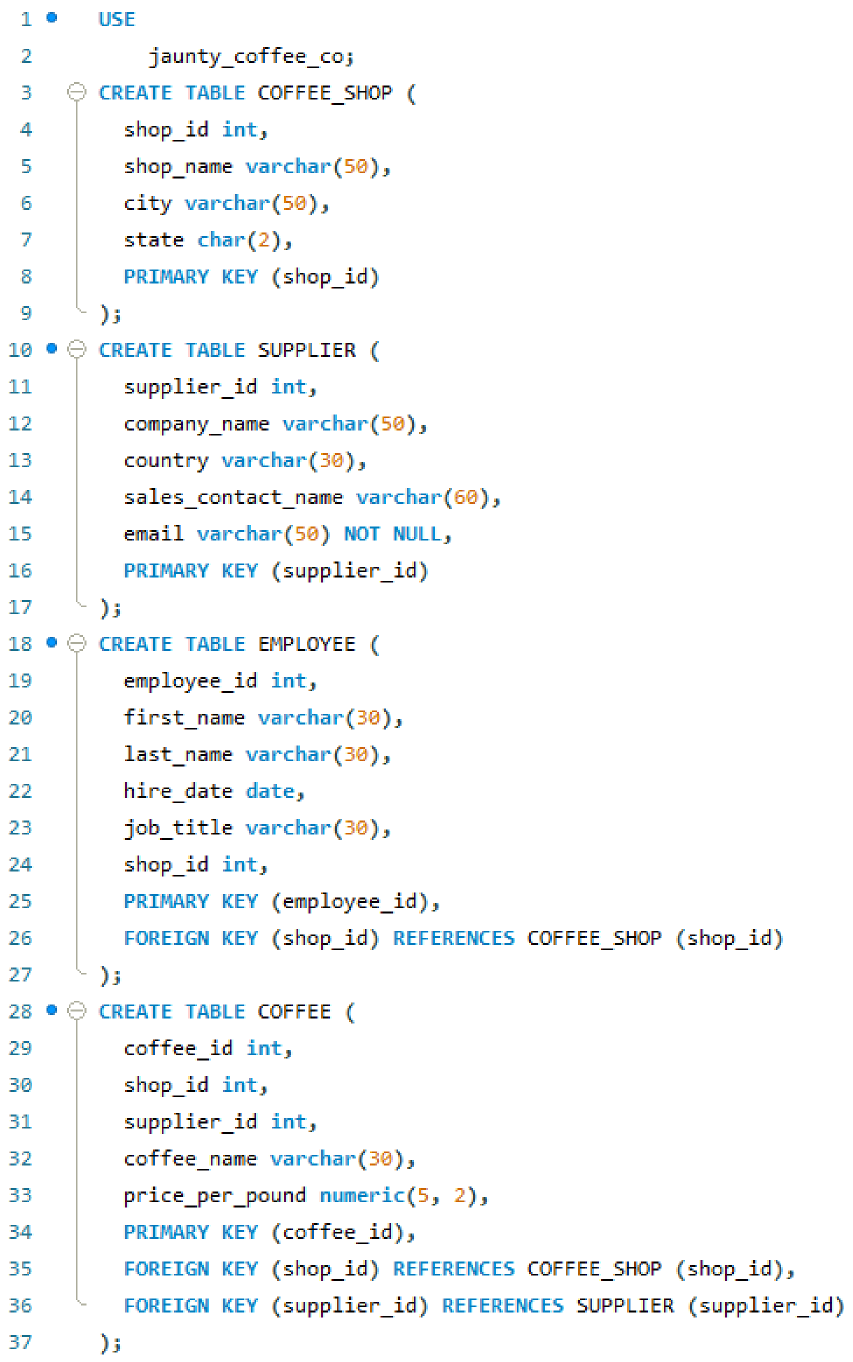
COFFEE		
PK	coffee_id	INTEGER
FK	shop_id	INTEGER
FK	supplier_id	INTEGER
	coffee_name	VARCHAR(30)
	price_per_pound	NUMERIC(5,2)

SUPPLIER		
PK	supplier_id	INTEGER
	company_name	VARCHAR(50)
	country	VARCHAR(30)
	sales_contact_name	VARCHAR(60)
	email	VARCHAR(50), NOT NULL

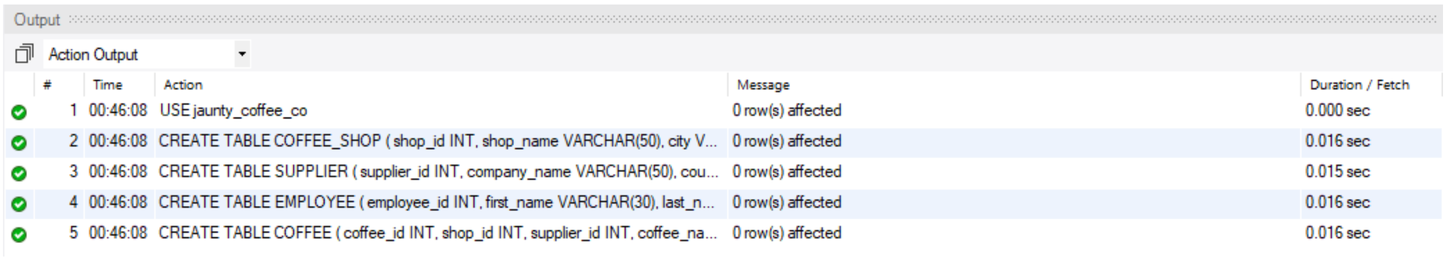
DBMS Choice: MySQL 8.0

Jaunty Coffee Co. Database Creation *(continued)*

B1. Develop SQL code to create *each* table as specified in the attached “Jaunty Coffee Co. ERD”



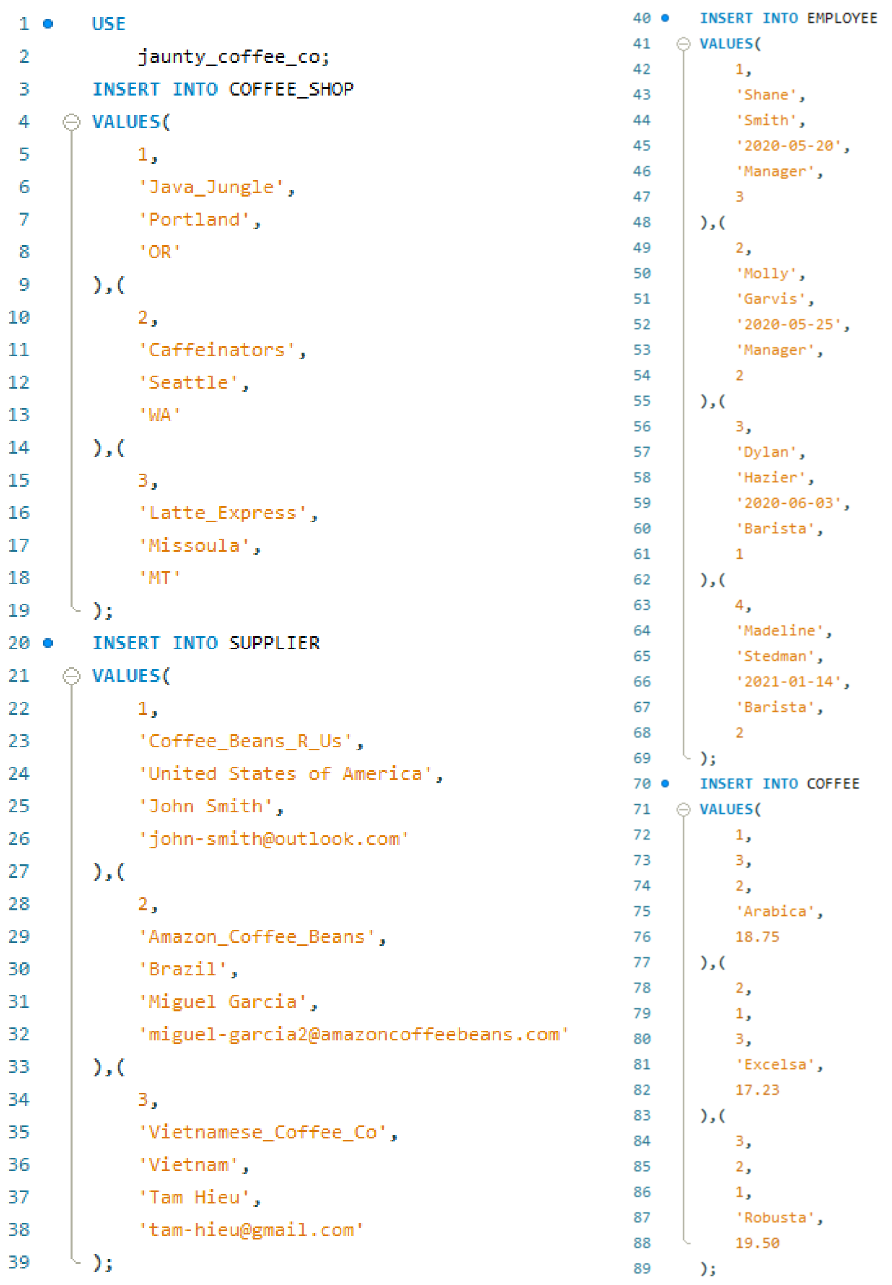
```sql
USE
    jaunty_coffee_co;
CREATE TABLE COFFEE_SHOP (
  shop_id int,
  shop_name varchar(50),
  city varchar(50),
  state char(2),
  PRIMARY KEY (shop_id)
);
CREATE TABLE SUPPLIER (
  supplier_id int,
  company_name varchar(50),
  country varchar(30),
  sales_contact_name varchar(60),
  email varchar(50) NOT NULL,
  PRIMARY KEY (supplier_id)
);
CREATE TABLE 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 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)
);
```



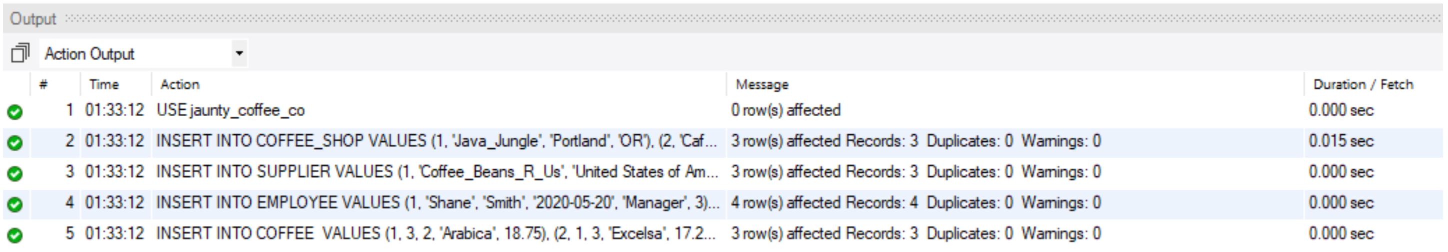
```
Output
Action Output
   #  Time      Action                                                                  Message              Duration / Fetch
✓  1  00:46:08  USE jaunty_coffee_co                                                    0 row(s) affected    0.000 sec
✓  2  00:46:08  CREATE TABLE COFFEE_SHOP ( shop_id INT, shop_name VARCHAR(50), city V...  0 row(s) affected    0.016 sec
✓  3  00:46:08  CREATE TABLE SUPPLIER ( supplier_id INT, company_name VARCHAR(50), cou...  0 row(s) affected    0.015 sec
✓  4  00:46:08  CREATE TABLE EMPLOYEE ( employee_id INT, first_name VARCHAR(30), last_n...  0 row(s) affected    0.016 sec
✓  5  00:46:08  CREATE TABLE COFFEE ( coffee_id INT, shop_id INT, supplier_id INT, coffee_na...  0 row(s) affected    0.016 sec
```

Jaunty Coffee Co. Database Creation *(continued)*

B2. Develop SQL code to populate *each* table in the database design document



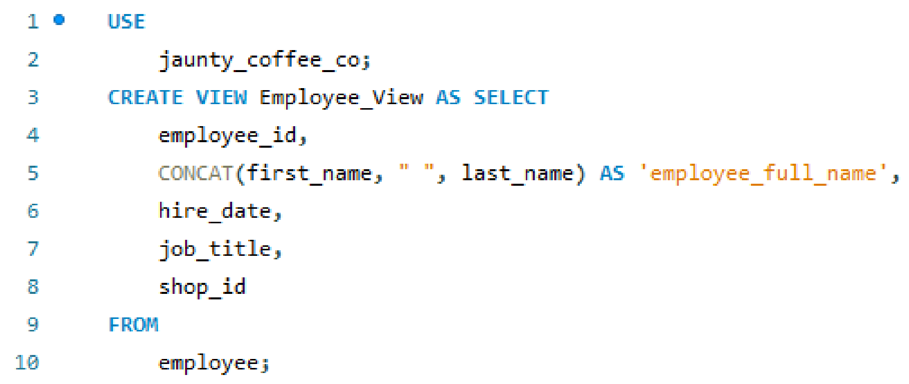
```sql
USE
    jaunty_coffee_co;
INSERT INTO COFFEE_SHOP
VALUES(
    1,
    'Java_Jungle',
    'Portland',
    'OR'
),(
    2,
    'Caffeinators',
    'Seattle',
    'WA'
),(
    3,
    'Latte_Express',
    'Missoula',
    'MT'
);
INSERT INTO SUPPLIER
VALUES(
    1,
    'Coffee_Beans_R_Us',
    'United States of America',
    'John Smith',
    'john-smith@outlook.com'
),(
    2,
    'Amazon_Coffee_Beans',
    'Brazil',
    'Miguel Garcia',
    'miguel-garcia2@amazoncoffeebeans.com'
),(
    3,
    'Vietnamese_Coffee_Co',
    'Vietnam',
    'Tam Hieu',
    'tam-hieu@gmail.com'
);
INSERT INTO EMPLOYEE
VALUES(
    1,
    'Shane',
    'Smith',
    '2020-05-20',
    'Manager',
    3
),(
    2,
    'Molly',
    'Garvis',
    '2020-05-25',
    'Manager',
    2
),(
    3,
    'Dylan',
    'Hazier',
    '2020-06-03',
    'Barista',
    1
),(
    4,
    'Madeline',
    'Stedman',
    '2021-01-14',
    'Barista',
    2
);
INSERT INTO COFFEE
VALUES(
    1,
    3,
    2,
    'Arabica',
    18.75
),(
    2,
    1,
    3,
    'Excelsa',
    17.23
),(
    3,
    2,
    1,
    'Robusta',
    19.50
);
```



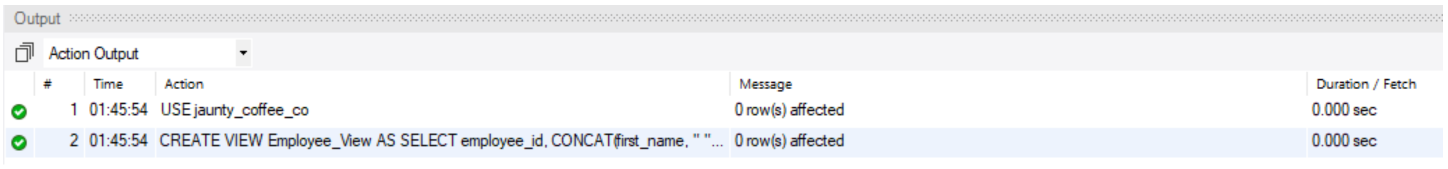
```
Output
Action Output
   #  Time      Action                                                                 Message                                                          Duration / Fetch
✓  1  01:33:12  USE jaunty_coffee_co                                                   0 row(s) affected                                                0.000 sec
✓  2  01:33:12  INSERT INTO COFFEE_SHOP VALUES (1, 'Java_Jungle', 'Portland', 'OR'), (2, 'Caf...  3 row(s) affected Records: 3  Duplicates: 0  Warnings: 0   0.015 sec
✓  3  01:33:12  INSERT INTO SUPPLIER VALUES (1, 'Coffee_Beans_R_Us', 'United States of Am...     3 row(s) affected Records: 3  Duplicates: 0  Warnings: 0   0.000 sec
✓  4  01:33:12  INSERT INTO EMPLOYEE VALUES (1, 'Shane', 'Smith', '2020-05-20', 'Manager', 3)...  4 row(s) affected Records: 4  Duplicates: 0  Warnings: 0   0.000 sec
✓  5  01:33:12  INSERT INTO COFFEE  VALUES (1, 3, 2, 'Arabica', 18.75), (2, 1, 3, 'Excelsa', 17.2...  3 row(s) affected Records: 3  Duplicates: 0  Warnings: 0   0.000 sec
```

Jaunty Coffee Co. Database Creation *(continued)*

B3. Develop SQL code to create a view

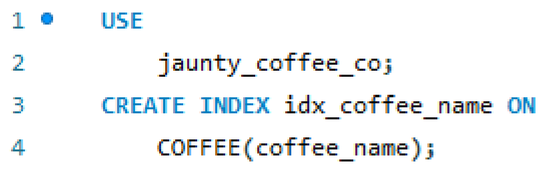


```sql
USE
    jaunty_coffee_co;
CREATE VIEW Employee_View AS SELECT
    employee_id,
    CONCAT(first_name, " ", last_name) AS 'employee_full_name',
    hire_date,
    job_title,
    shop_id
FROM
    employee;
```

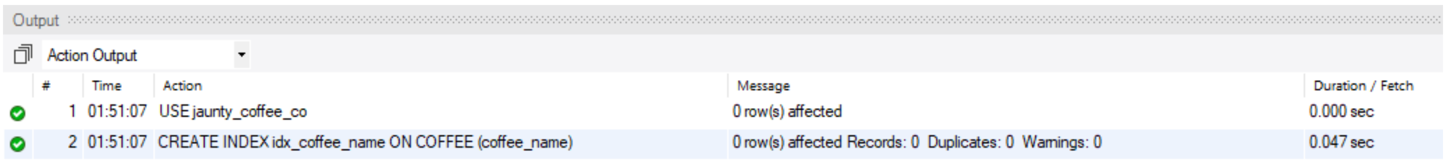


```
Output
Action Output
  #  Time      Action                                                               Message            Duration / Fetch
  1  01:45:54  USE jaunty_coffee_co                                                 0 row(s) affected  0.000 sec
  2  01:45:54  CREATE VIEW Employee_View AS SELECT employee_id, CONCAT(first_name, " "...  0 row(s) affected  0.000 sec
```

B4. Develop SQL code to create an index on the coffee_name field



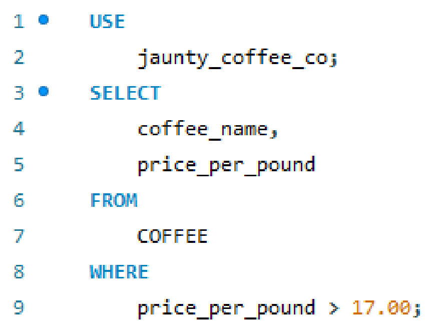
```sql
USE
    jaunty_coffee_co;
CREATE INDEX idx_coffee_name ON
    COFFEE(coffee_name);
```



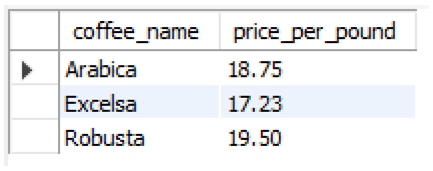
```
Output
Action Output
  #  Time      Action                                             Message                                                  Duration / Fetch
  1  01:51:07  USE jaunty_coffee_co                               0 row(s) affected                                        0.000 sec
  2  01:51:07  CREATE INDEX idx_coffee_name ON COFFEE (coffee_name)  0 row(s) affected Records: 0  Duplicates: 0  Warnings: 0  0.047 sec
```

Jaunty Coffee Co. Database Creation *(continued)*

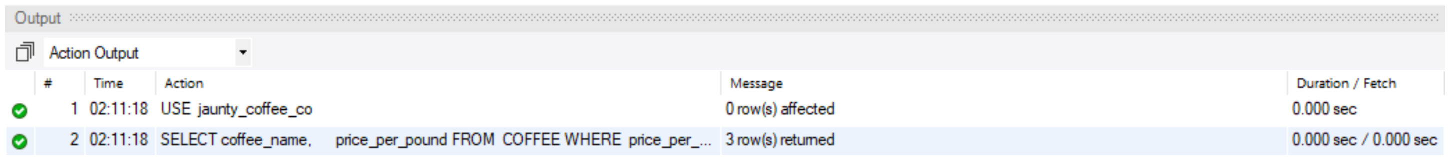
B5. Develop SQL code to create an SFW (SELECT–FROM–WHERE) query for *any* of your tables or views



```
USE
    jaunty_coffee_co;
SELECT
    coffee_name,
    price_per_pound
FROM
    COFFEE
WHERE
    price_per_pound > 17.00;
```



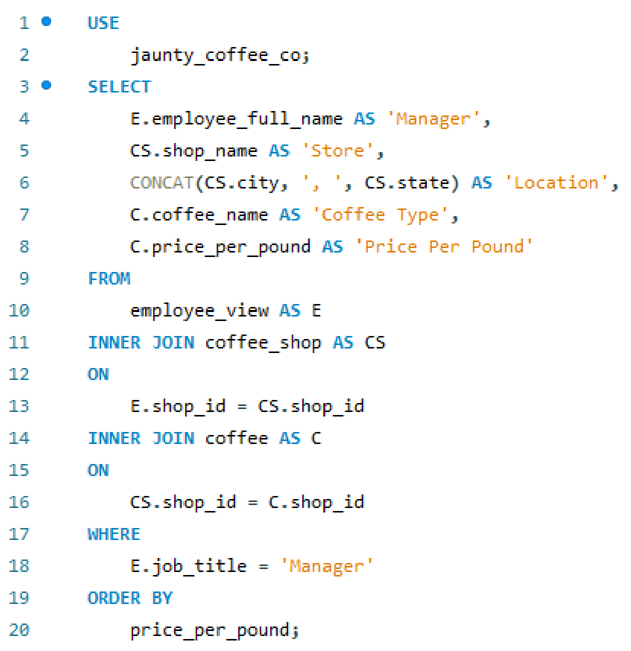
coffee_name	price_per_pound
Arabica	18.75
Excelsa	17.23
Robusta	19.50



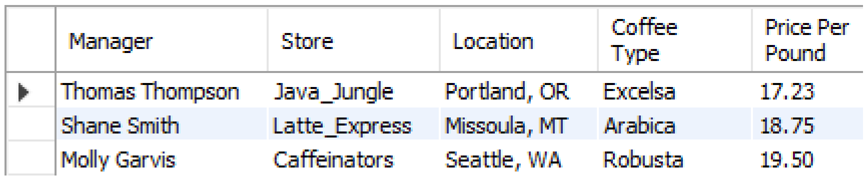
```
Output
Action Output
   #  Time      Action                                                          Message              Duration / Fetch
   1  02:11:18  USE jaunty_coffee_co                                            0 row(s) affected    0.000 sec
   2  02:11:18  SELECT coffee_name,    price_per_pound FROM  COFFEE WHERE  price_per_...  3 row(s) returned    0.000 sec / 0.000 sec
```

Jaunty Coffee Co. Database Creation *(continued)*

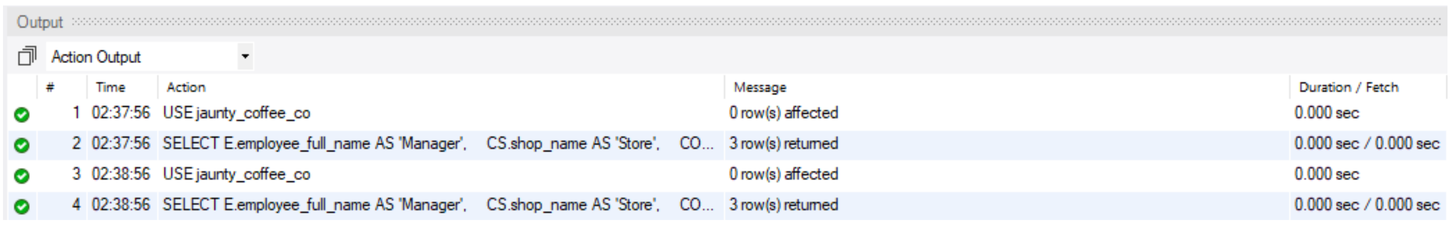
B6. Develop SQL code to create a query



```sql
USE
    jaunty_coffee_co;
SELECT
    E.employee_full_name AS 'Manager',
    CS.shop_name AS 'Store',
    CONCAT(CS.city, ', ', CS.state) AS 'Location',
    C.coffee_name AS 'Coffee Type',
    C.price_per_pound AS 'Price Per Pound'
FROM
    employee_view AS E
INNER JOIN coffee_shop AS CS
ON
    E.shop_id = CS.shop_id
INNER JOIN coffee AS C
ON
    CS.shop_id = C.shop_id
WHERE
    E.job_title = 'Manager'
ORDER BY
    price_per_pound;
```



	Manager	Store	Location	Coffee Type	Price Per Pound
▶	Thomas Thompson	Java_Jungle	Portland, OR	Excelsa	17.23
	Shane Smith	Latte_Express	Missoula, MT	Arabica	18.75
	Molly Garvis	Caffeinators	Seattle, WA	Robusta	19.50



Output

Action Output

	#	Time	Action	Message	Duration / Fetch
✓	1	02:37:56	USE jaunty_coffee_co	0 row(s) affected	0.000 sec
✓	2	02:37:56	SELECT E.employee_full_name AS 'Manager', CS.shop_name AS 'Store', CO...	3 row(s) returned	0.000 sec / 0.000 sec
✓	3	02:38:56	USE jaunty_coffee_co	0 row(s) affected	0.000 sec
✓	4	02:38:56	SELECT E.employee_full_name AS 'Manager', CS.shop_name AS 'Store', CO...	3 row(s) returned	0.000 sec / 0.000 sec