Section A.1:

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

A.1.a-b)

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

A.1.c)

Q: *Explain how you assigned attributes to the 2NF tables and determined the cardinality of the relationships between your 2NF tables*.

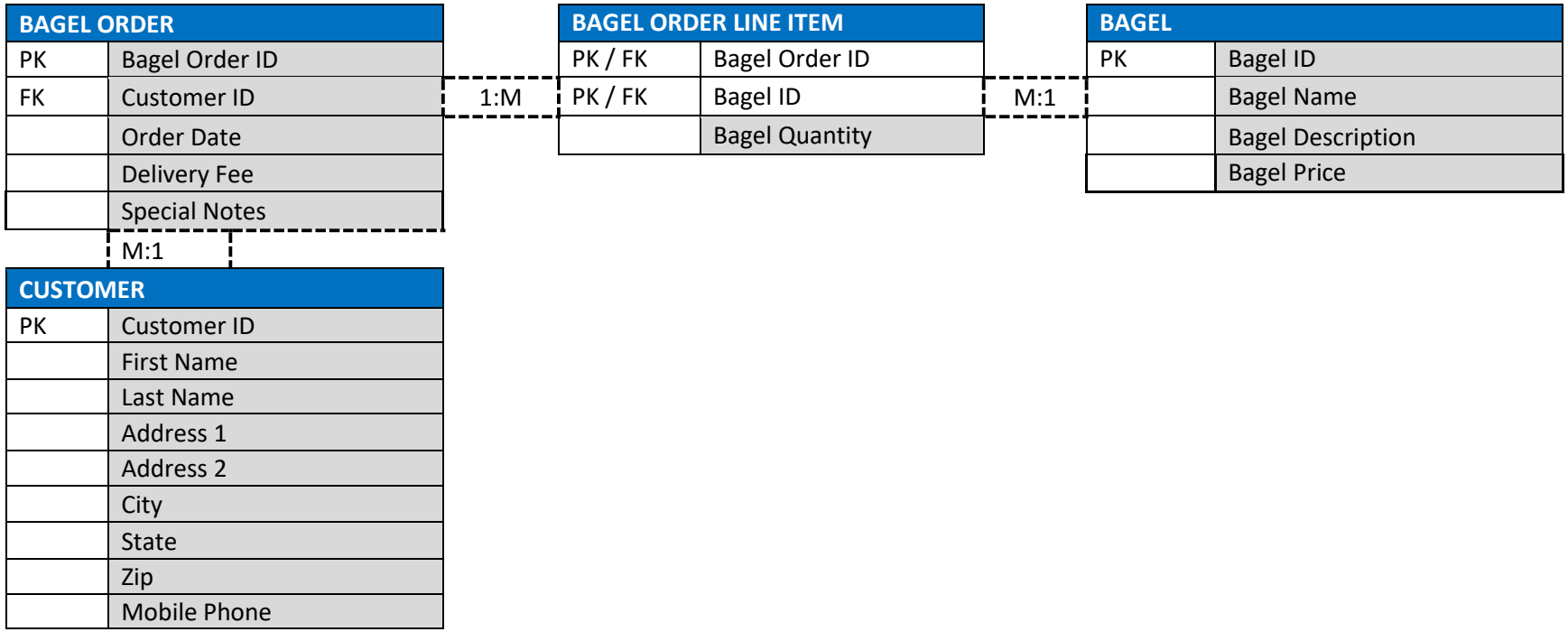
A: The "Bagel Order" table contains all the information related to the order, based on the information presented in the "Bagel Order Form." The intersection table contains the foreign key, "Bagel Order ID", that links to the primary key, "Bagel Order ID", in the "Bagel Order" table. The cardinality of this relationship is one-to-many; moreover, one order can have many bagel line items. The "Bagel ID" foreign key links to the primary key, "Bagel ID", in the "Bagel Table." The cardinality of this relationship is many-to-one; many bagel order line items have one bagel type. Furthermore, this leads to needing a "Bagel Quantity" attribute in the "Bagel Order Line Item" table to allow us to quantify the bagel types per order. Lastly, the "Bagel" table contains all information related to the bagel, minus the bagel quantity.

Section A.2:

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

A.2.a-d)

Third Normal Form (3NF)



A.2.e)

Q: *Explain how you assigned attributes to the 3NF tables and determined the cardinality of the relationships between your 3NF tables.*

A: The “Bagel Order” table, “Bagel Order Line Item” table, and “Bagel” table have the exact same relationships as 2NF. The main difference is that the customer attributes were extracted from the “Bagel Order” table and a new “Customer” table was created. This created a new many-to-one relationship between the “Bagel Order” table and the “Customer” table; one customer can have many orders.

Section A.3:

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

A.3.a-b)

Final Physical Database Model

BAGEL_ORDER		
PK	bagel_order_id	INT
FK	customer_id	INT
	order_date	TIMESTAMP
	delivery_fee	NUMERIC
	special_notes	VARCHAR(30)

1:M

BAGEL_ORDER_LINE_ITEM		
PK / FK	bagel_order_id	INT
PK / FK	bagel_id	CHAR(2)
	bagel_quantity	INT

M:1

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

M:1

CUSTOMER		
PK	customer_id	INT
	first_name	VARCHAR(15)
	last_name	VARCHAR(30)
	address_1	VARCHAR(30)
	address_2	VARCHAR(30)
	city	VARCHAR(30)
	state	CHAR(2)
	zip	VARCHAR(10)
	mobile_phone	VARCHAR(10)

Section B:

B.1) Develop SQL code to create each table as specified in the attached "Jaunty Coffee Co. ERD" by doing the following:

a. Provide the SQL code you wrote to create all the tables.

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

CODE:

```
CREATE DATABASE IF NOT EXISTS jaunty_coffee_co;

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 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)
);
```

```sql
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)
);
```

RESULTS:

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	12:00:16	CREATE DATABASE IF NOT EXISTS jaunty_coffee_co	1 row(s) affected	0.015 sec
2	12:00:20	USE jaunty_coffee_co	0 row(s) affected	0.000 sec
3	12:00:22	CREATE TABLE COFFEE_SHOP(shop_id INT, shop_name V...	0 row(s) affected	0.016 sec
4	12:00:23	CREATE TABLE EMPLOYEE(employee_id INT, first_name VA...	0 row(s) affected	0.016 sec
5	12:00:24	CREATE TABLE SUPPLIER (supplier_id INT, company_name...	0 row(s) affected	0.016 sec
6	12:00:26	CREATE TABLE COFFEE(coffee_id INT, shop_id INT, sup...	0 row(s) affected	0.031 sec

B.2) Develop SQL code to populate each table in the database design document by doing the following:

a. Provide the SQL code you wrote to populate the tables with at least three rows of data in each table.

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

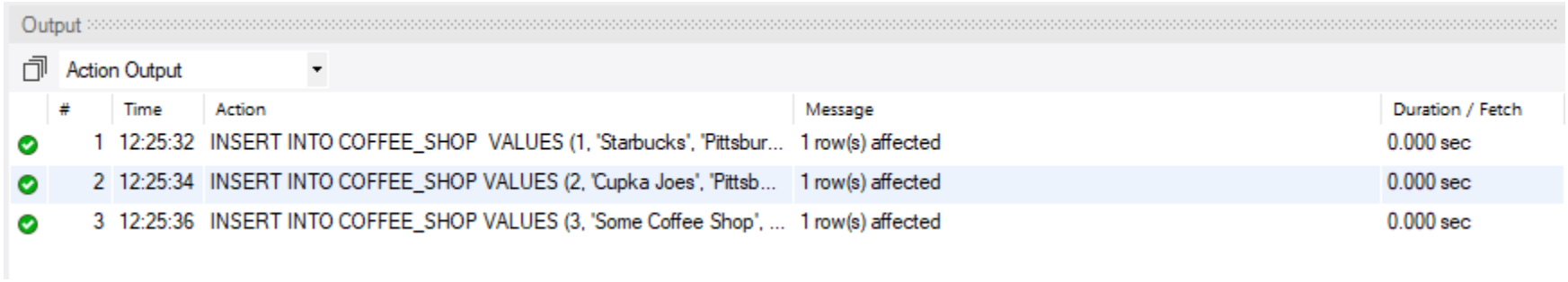
CODE: (COFFEE SHOP TABLE)

```
INSERT INTO COFFEE_SHOP
    VALUES (1, 'Starbucks', 'Pittsburgh', 'PA');

INSERT INTO COFFEE_SHOP
    VALUES (2, 'Cupka Joes', 'Pittsburgh', 'PA');

INSERT INTO COFFEE_SHOP
    VALUES (3, 'Some Coffee Shop', 'Pittsburgh', 'PA');
```

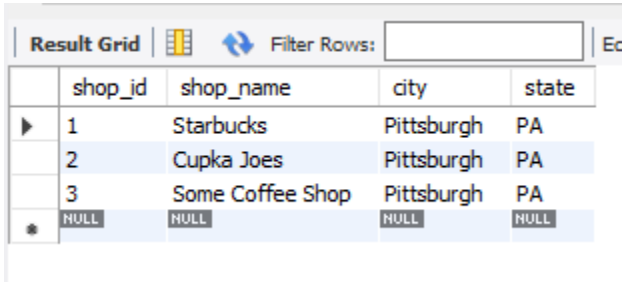
RESULTS: (COFFEE SHOP TABLE)



Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	12:25:32	INSERT INTO COFFEE_SHOP VALUES (1, 'Starbucks', 'Pittsbur...	1 row(s) affected	0.000 sec
2	12:25:34	INSERT INTO COFFEE_SHOP VALUES (2, 'Cupka Joes', 'Pittsb...	1 row(s) affected	0.000 sec
3	12:25:36	INSERT INTO COFFEE_SHOP VALUES (3, 'Some Coffee Shop', ...	1 row(s) affected	0.000 sec



Result Grid Filter Rows:

shop_id	shop_name	city	state
1	Starbucks	Pittsburgh	PA
2	Cupka Joes	Pittsburgh	PA
3	Some Coffee Shop	Pittsburgh	PA
NULL	NULL	NULL	NULL

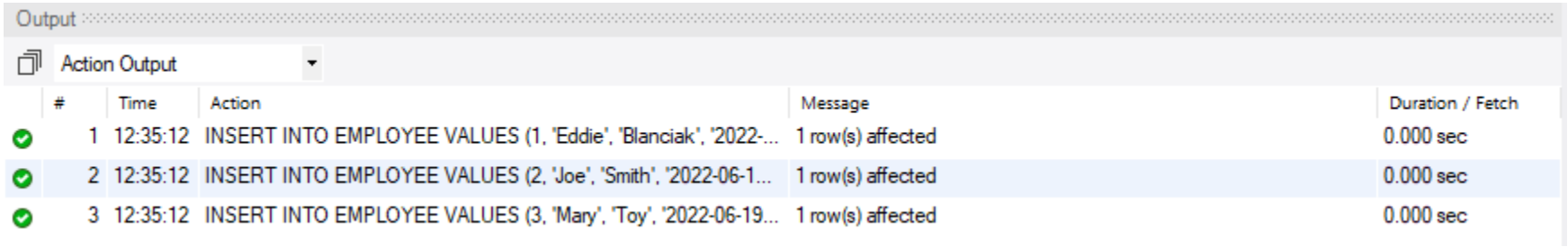
CODE: (EMPLOYEE TABLE)

```
INSERT INTO EMPLOYEE
    VALUES (1, 'Eddie', 'Blanciak', '2022-06-19', 'Developer', 1);

INSERT INTO EMPLOYEE
    VALUES (2, 'Joe', 'Smith', '2022-06-19', 'Java Developer', 2);

INSERT INTO EMPLOYEE
    VALUES (3, 'Mary', 'Toy', '2022-06-19', 'C++ Developer', 3);
```

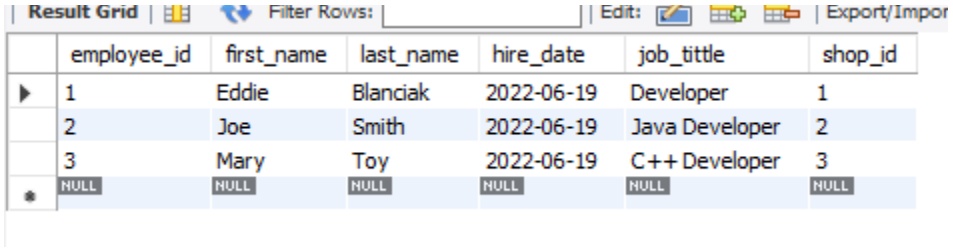
RESULTS: (EMPLOYEE TABLE)



Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	12:35:12	INSERT INTO EMPLOYEE VALUES (1, 'Eddie', 'Blanciak', '2022-...	1 row(s) affected	0.000 sec
2	12:35:12	INSERT INTO EMPLOYEE VALUES (2, 'Joe', 'Smith', '2022-06-1...	1 row(s) affected	0.000 sec
3	12:35:12	INSERT INTO EMPLOYEE VALUES (3, 'Mary', 'Toy', '2022-06-19...	1 row(s) affected	0.000 sec



Result Grid | Filter Rows: | Edit: | Export/Impor

employee_id	first_name	last_name	hire_date	job_tittle	shop_id
1	Eddie	Blanciak	2022-06-19	Developer	1
2	Joe	Smith	2022-06-19	Java Developer	2
3	Mary	Toy	2022-06-19	C++ Developer	3
NULL	NULL	NULL	NULL	NULL	NULL

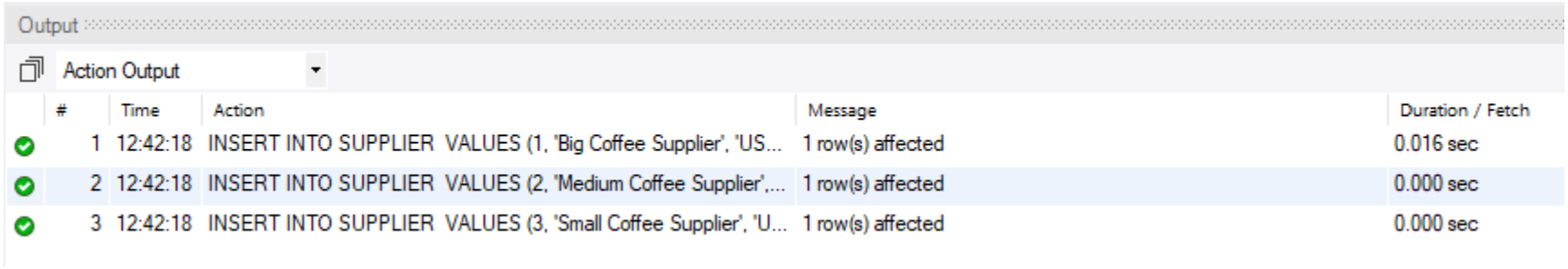
CODE: (SUPPLIER TABLE)

```
INSERT INTO SUPPLIER
    VALUES (1, 'Big Coffee Supplier', 'USA', 'Salesman Joe', 'joe@bigcoffeesupplier.com');

INSERT INTO SUPPLIER
    VALUES (2, 'Medium Coffee Supplier', 'USA', 'Salesman Ed', 'ed@mediumcoffeesupplier.com');

INSERT INTO SUPPLIER
    VALUES (3, 'Small Coffee Supplier', 'USA', 'Salesman Tim', 'tim@smallcoffeesupplier.com');
```

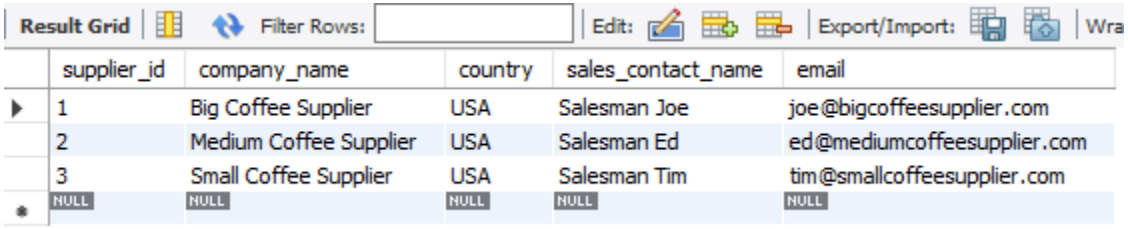
RESULTS: (SUPPLIER TABLE)



Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	12:42:18	INSERT INTO SUPPLIER VALUES (1, 'Big Coffee Supplier', 'US...	1 row(s) affected	0.016 sec
2	12:42:18	INSERT INTO SUPPLIER VALUES (2, 'Medium Coffee Supplier',...	1 row(s) affected	0.000 sec
3	12:42:18	INSERT INTO SUPPLIER VALUES (3, 'Small Coffee Supplier', 'U...	1 row(s) affected	0.000 sec



Result Grid | Filter Rows: | Edit: | Export/Import: | Wra

supplier_id	company_name	country	sales_contact_name	email
1	Big Coffee Supplier	USA	Salesman Joe	joe@bigcoffeesupplier.com
2	Medium Coffee Supplier	USA	Salesman Ed	ed@mediumcoffeesupplier.com
3	Small Coffee Supplier	USA	Salesman Tim	tim@smallcoffeesupplier.com
NULL	NULL	NULL	NULL	NULL

CODE: (COFFEE TABLE)

```
INSERT INTO COFFEE
    VALUES (1, 1, 1, 'Really Good Coffee', 10.50);

INSERT INTO COFFEE
    VALUES (2, 2, 2, 'Really Okay Coffee', 5.50);

INSERT INTO COFFEE
    VALUES (3, 3, 2, 'Really Bad Coffee', 1.50);
```

RESULTS: (COFFEE TABLE)

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	12:50:57	INSERT INTO COFFEE VALUES (1, 1, 1, 'Really Good Coffee', ...	1 row(s) affected	0.000 sec
2	12:50:58	INSERT INTO COFFEE VALUES (2, 2, 2, 'Really Okay Coffee', 5...	1 row(s) affected	0.000 sec
3	12:50:59	INSERT INTO COFFEE VALUES (3, 3, 2, 'Really Bad Coffee', 1....	1 row(s) affected	0.000 sec

Result Grid | Filter Rows: | Edit: | Export

coffee_id	shop_id	supplier_id	coffee_name	price_per_pound
1	1	1	Really Good Coffee	10.50
2	2	2	Really Okay Coffee	5.50
3	3	2	Really Bad Coffee	1.50
NULL	NULL	NULL	NULL	NULL

B.3) Develop SQL code to create a view by doing the following:

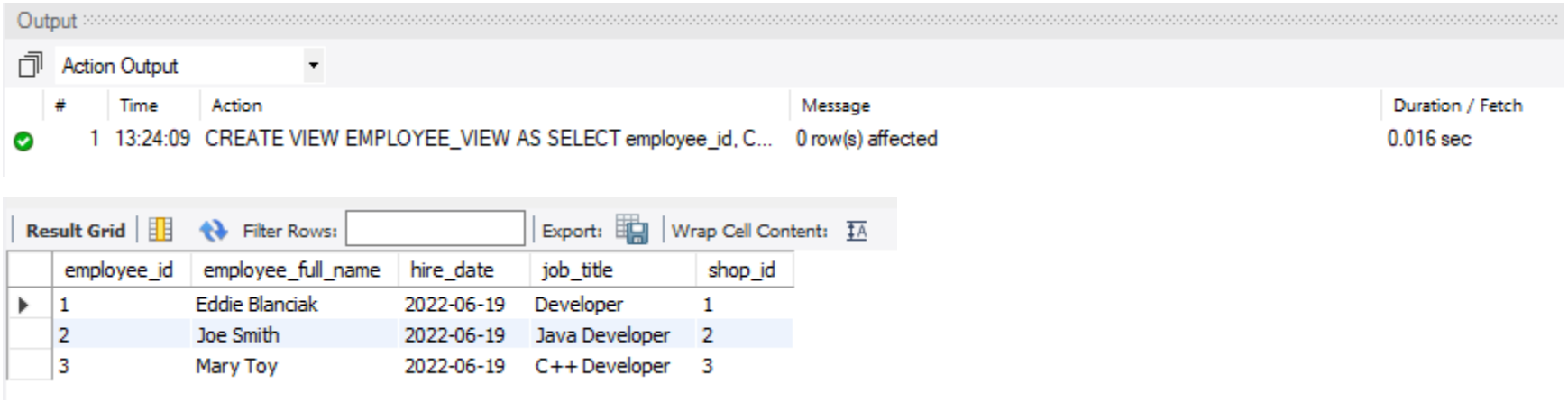
a. Provide the SQL code you wrote to create your view. The view should show all of the information from the "Employee" table but concatenate each employee's first and last name, formatted with a space between the first and last name, into a new attribute called employee_full_name.

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

CODE:

```
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;
```

RESULTS:



Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	13:24:09	CREATE VIEW EMPLOYEE_VIEW AS SELECT employee_id, C...	0 row(s) affected	0.016 sec

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

employee_id	employee_full_name	hire_date	job_title	shop_id
1	Eddie Blanciak	2022-06-19	Developer	1
2	Joe Smith	2022-06-19	Java Developer	2
3	Mary Toy	2022-06-19	C++ Developer	3

B.4) Develop SQL code to create an index on the coffee_name field by doing the following:

a. Provide the SQL code you wrote to create your index on the coffee_name field from the "Coffee" table.

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

CODE:

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

RESULTS:

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	19:20:17	CREATE INDEX coffee_name ON COFFEE (coffee_name)	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0	0.031 sec

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Table	Non_unique	Key_name	Seq_in_index	Column_name	Collation	Cardinality	Sub_part	Packed	Null	Index_type	Comment
coffee	0	PRIMARY	1	coffee_id	A	2	NULL	NULL		BTREE	
coffee	1	shop_id	1	shop_id	A	2	NULL	NULL	YES	BTREE	
coffee	1	supplier_id	1	supplier_id	A	2	NULL	NULL	YES	BTREE	
coffee	1	coffee_name	1	coffee_name	A	3	NULL	NULL	YES	BTREE	

B.5) Develop SQL code to create an SFW (SELECT–FROM–WHERE) query for any of your tables or views by doing the following:

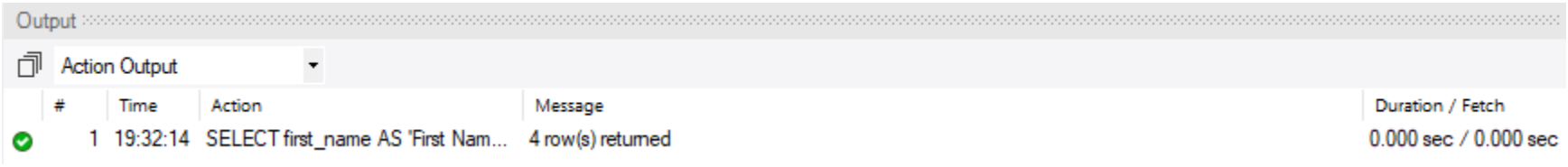
a. Provide the SQL code you wrote to create your SFW query.

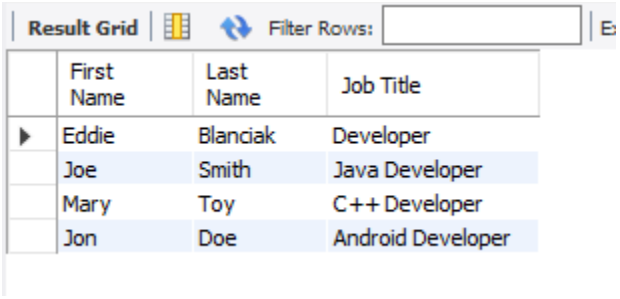
b. Demonstrate that you tested your code by providing a screenshot showing your SQL commands and the database server's response.

CODE:

```
SELECT first_name AS 'First Name', last_name AS 'Last Name', job_title AS 'Job Title'
    FROM EMPLOYEE
    WHERE job_title LIKE '%developer';
```

RESULTS:





First Name	Last Name	Job Title
Eddie	Blanciak	Developer
Joe	Smith	Java Developer
Mary	Toy	C++ Developer
Jon	Doe	Android Developer

B.6) Develop SQL code to create a query by doing the following:

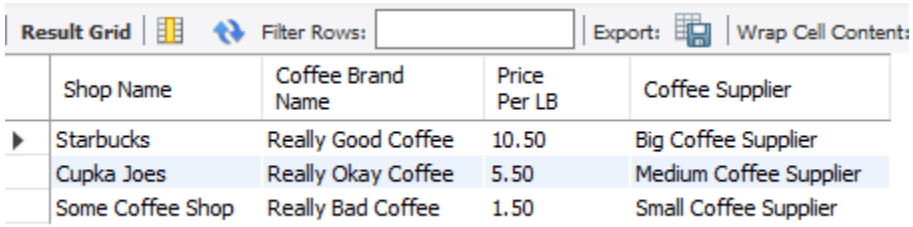
a. Provide the SQL code you wrote to create your table joins query. The query should join together three different tables and include attributes from all three tables in its output.

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

CODE:

```
SELECT COFFEE_SHOP.shop_name AS 'Shop Name', COFFEE.coffee_name AS 'Coffee Brand Name', COFFEE.price_per_pound AS
'Price Per LB', SUPPLIER.company_name AS 'Coffee Supplier'
    FROM ((COFFEE
    INNER JOIN COFFEE_SHOP ON  COFFEE.shop_id = COFFEE_SHOP.shop_id)
    INNER JOIN SUPPLIER ON COFFEE.supplier_id = SUPPLIER.supplier_id);
```

RESULTS:



Shop Name	Coffee Brand Name	Price Per LB	Coffee Supplier
Starbucks	Really Good Coffee	10.50	Big Coffee Supplier
Cupka Joes	Really Okay Coffee	5.50	Medium Coffee Supplier
Some Coffee Shop	Really Bad Coffee	1.50	Small Coffee Supplier