A1.

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

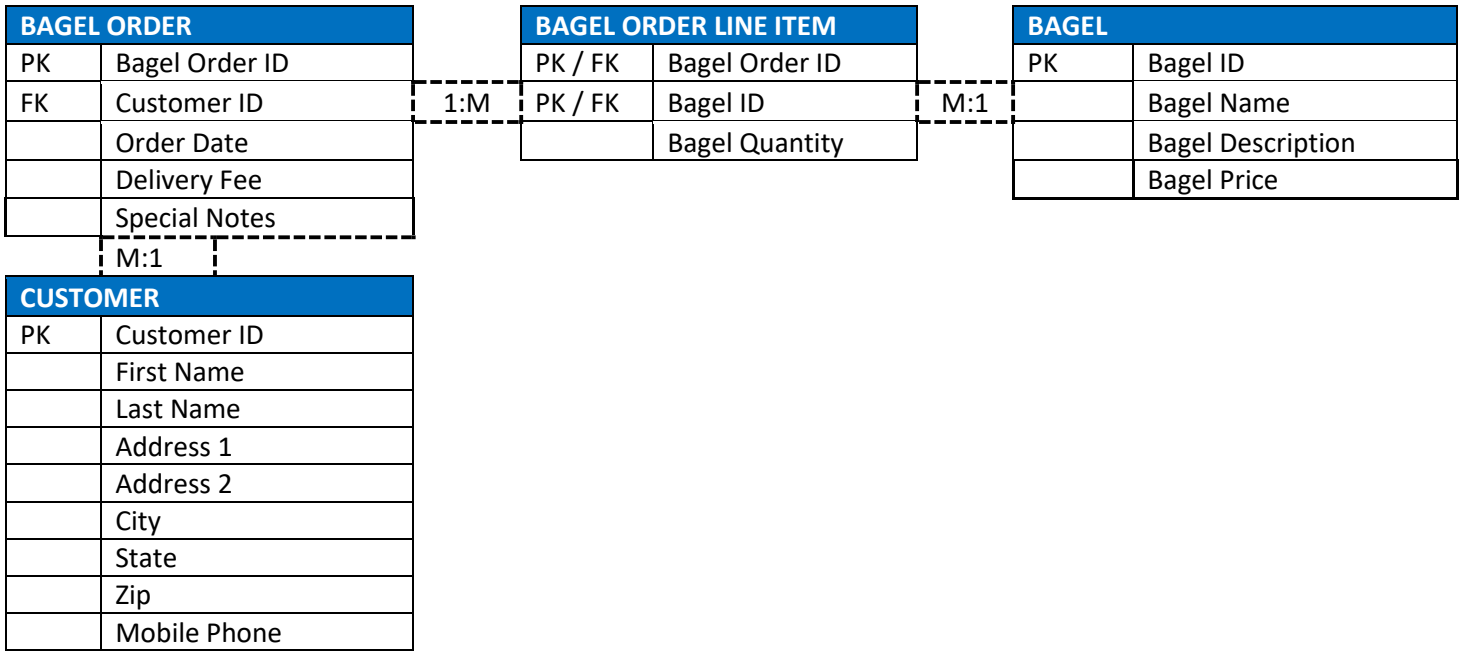
BAGEL	
PK	Bagel ID
	Bagel Name
	Bagel Description
	Bagel Price

I assigned each attribute by finding the functional dependence of each one and determining whether all non-key attributes were dependent on the whole Primary Key. In the 1NF table, the attributes Bagel Name, Bagel Description, and Bagel Price depend on Bagel ID alone. Since the Primary Key is a composite of the Bagel ID and the Bagel Order ID, this does not satisfy the requirements of 2NF. To resolve this, we merely create a new table of these attributes with Bagel ID as the Primary Key. Unfortunately, this does not completely resolve the problem because all the other attributes (except for Bagel Quantity) depend on Bagel Order ID. We can resolve this the same way as before by creating a new table with Bagel Order ID as the Primary Key. This leaves us with three tables where all tables are in 2NF and each attribute that is not a Primary Key depends on the whole Primary Key.

To determine the cardinality, I analyzed the relationship between tables. For example, A specific bagel order can have multiple bagel order line items. The relationship between BAGEL ORDER and BAGEL ORDER LINE ITEM is one-to-many because each value of Bagel Order ID occurs once in the first table and many times in the second table. Therefore, for each BAGEL ORDER, there are many BAGEL ORDER LINE ITEMS. Also, in the BAGEL ORDER LINE TABLE the same Bagel ID value can be repeated multiple times while in the BAGEL table, it cannot. This relationship is therefore many-to-one.

A2.

Third Normal Form (3NF)



I assigned each attribute by finding the functional dependence of each one and determining whether no non-key attributes were transitively dependent on other non-key attributes. Because the Primary Key in the BAGEL ORDER table wasn't the only candidate key that the customer information could depend on, we removed it, creating a new table, CUSTOMER, with a new attribute, Customer ID, as its Primary Key. We then used this Key as a Foreign Key within the BAGEL ORDER table, forming a relationship between this table and the table we just created. Now all attributes depend on the Key, the whole Key, and nothing but the Key of each table.

I determined the cardinality of this new table's relationship by analyzing its relationship to its corresponding table, BAGEL ORDER. Because every bagel order has a maximum of one customer, but a single customer can have many orders, the relationship between BAGEL ORDER and CUSTOMER, respectively, is many-to-one. The relationships between the other tables remain the same as they were in the 2NF form.

A3.

Final Physical Database Model

BAGEL ORDER		
PK	bagel_order_id	INT
FK	customer_id	INT
	order_date	TIMESTAMP
	delivery_fee	NUMERIC(5, 2)
	special_notes	VARCHAR(100)

M:1

CUSTOMER		
PK	customer_id	INT
	first_name	VARCHAR(30)
	last_name	VARCHAR(30)
	address1	VARCHAR(30)
	address2	VARCHAR(30)
	city	VARCHAR(30)
	state	CHAR(2)
	zip	VARCHAR(7)
	mobile_phone	VARCHAR(14)

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(50)
	bagel_price	NUMERIC(5, 2)

B1.

```
CREATE TABLE COFFEE_SHOP (
shop_id          INTEGER,
shop_name        VARCHAR(50),
city             VARCHAR(50),
state            CHAR(2),
PRIMARY KEY (shop_id)
);

CREATE TABLE EMPLOYEE (
employee_id   INTEGER,
first_name     VARCHAR(30),
last_name     VARCHAR(30),
hire_date      DATE,
job_title       VARCHAR(30),
shop_id        INTEGER,
PRIMARY KEY (employee_id),
FOREIGN KEY (shop_id) REFERENCES COFFEE_SHOP(shop_id)
);
```

```
CREATE TABLE SUPPLIER (
supplier_id              INTEGER,
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           INTEGER,
shop_id              INTEGER,
supplier_id          INTEGER,
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)
);
```

```sql
CREATE TABLE COFFEE_SHOP (
  shop_id   INTEGER,
  shop_name VARCHAR(50),
  city      VARCHAR(50),
  state     CHAR(2),
  PRIMARY KEY (shop_id)
);

CREATE TABLE EMPLOYEE (
  employee_id   INTEGER,
  first_name    VARCHAR(30),
  last_name     VARCHAR(30),
  hire_date     DATE,
  job_title     VARCHAR(30),
  shop_id       INTEGER,
  PRIMARY KEY (employee_id),
  FOREIGN KEY (shop_id) REFERENCES COFFEE_SHOP(shop_id)
);
```

Build Schema | Edit Fullscreen | Browser | [;]

✔ Schema Ready

```sql
CREATE TABLE SUPPLIER (
  supplier_id        INTEGER,
  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       INTEGER,
  shop_id         INTEGER,
  supplier_id     INTEGER,
  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)
);
```

Build Schema | Edit Fullscreen | Browser | [;]

✔ Schema Ready

B2.

```
INSERT INTO COFFEE_SHOP (shop_id, shop_name, city, state) VALUES
(1, 'Starbucks', 'Chicago', 'IL'),
(2, 'Coffee Oasis', 'Seattle', 'WA'),
(3, 'The Grind', 'Richmond', 'VA');

INSERT INTO EMPLOYEE (employee_id, first_name, last_name, hire_date, job_title, shop_id) VALUES
(1, 'George', 'Blake', '2016-12-01', 'Manager', 1),
(2, 'Denise', 'Fredrickson', '2021-03-02', 'Barista', 3),
(3, 'Miguel', 'Rodriguez', '2023-01-01', 'Barista', 2);

INSERT INTO SUPPLIER (supplier_id, company_name, country, sales_contact_name, email) VALUES
(1, 'Beans R Us', 'United States', 'Alison Kane', 'beansrus@gmail.com'),
(2, 'CreamerCO', 'Germany', 'John Fredrickson', 'creamerco@yahoo.com'),
(3, 'Coffee Pros', 'United Kingdom', 'Paul Grayson', 'coffeepros@gmail.com');

INSERT INTO COFFEE (coffee_id, shop_id, supplier_id, coffee_name, price_per_pound) VALUES
(1, 1, 2, 'Arabica', 1.50),
(2, 3, 1, 'Robusta', 2.98),
(3, 2, 1, 'Liberica', 3.29);
```

```
  FOREIGN KEY (supplier_id) REFERENCES SUPPLIER(supplier_id)
);

INSERT INTO COFFEE_SHOP (shop_id, shop_name, city, state) VALUES
(1, 'Starbucks', 'Chicago', 'IL'),
(2, 'Coffee Oasis', 'Seattle', 'WA'),
(3, 'The Grind', 'Richmond', 'VA');

INSERT INTO EMPLOYEE (employee_id, first_name, last_name, hire_date, job_title, shop_id) VALUES
(1, 'George', 'Blake', '2016-12-01', 'Manager', 1),
(2, 'Denise', 'Fredrickson', '2021-03-02', 'Barista', 3),
(3, 'Miguel', 'Rodriguez', '2023-01-01', 'Barista', 2);

INSERT INTO SUPPLIER (supplier_id, company_name, country, sales_contact_name, email) VALUES
(1, 'Beans R Us', 'United States', 'Alison Kane', 'beansrus@gmail.com'),
(2, 'CreamerCO', 'Germany', 'John Fredrickson', 'creamerco@yahoo.com'),
(3, 'Coffee Pros', 'United Kingdom', 'Paul Grayson', 'coffeepros@gmail.com');

INSERT INTO COFFEE (coffee_id, shop_id, supplier_id, coffee_name, price_per_pound) VALUES
(1, 1, 2, 'Arabica', 1.50),
(2, 3, 1, 'Robusta', 2.98),
(3, 2, 1, 'Liberica', 3.29);
```

Build Schema | Edit Fullscreen | Browser | [;]

✔ Schema Ready

B3.

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;

B4.

CREATE INDEX idx_coffee_name ON COFFEE (coffee_name);

```
(2, 'Coffee Oasis', 'Seattle', 'WA'),
(3, 'The Grind', 'Richmond', 'VA');

INSERT INTO EMPLOYEE (employee_id, first_name, last_name, hire_date, job_title, shop_id) VALUES
(1, 'George', 'Blake', '2016-12-01', 'Manager', 1),
(2, 'Denise', 'Fredrickson', '2021-03-02', 'Barista', 3),
(3, 'Miguel', 'Rodriguez', '2023-01-01', 'Barista', 2);

INSERT INTO SUPPLIER (supplier_id, company_name, country, sales_contact_name, email) VALUES
(1, 'Beans R Us', 'United States', 'Alison Kane', 'beansrus@gmail.com'),
(2, 'CreamerCO', 'Germany', 'John Fredrickson', 'creamerco@yahoo.com'),
(3, 'Coffee Pros', 'United Kingdom', 'Paul Grayson', 'coffeepros@gmail.com');

INSERT INTO COFFEE (coffee_id, shop_id, supplier_id, coffee_name, price_per_pound) VALUES
(1, 1, 2, 'Arabica', 1.50),
(2, 3, 1, 'Robusta', 2.98),
(3, 2, 1, 'Liberica', 3.29);

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;

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

Build Schema | Edit Fullscreen | Browser | [;]

✔ Schema Ready

B5.

SELECT *

FROM EMPLOYEE

WHERE employee_id > 1;

```
(2, 'Coffee Oasis', 'Seattle', 'WA'),
(3, 'The Grind', 'Richmond', 'VA');

INSERT INTO EMPLOYEE (employee_id, first_name, last_name, hire_date, job_title, shop_id) VALUES
(1, 'George', 'Blake', '2016-12-01', 'Manager', 1),
(2, 'Denise', 'Fredrickson', '2021-03-02', 'Barista', 3),
(3, 'Miguel', 'Rodriguez', '2023-01-01', 'Barista', 2);

INSERT INTO SUPPLIER (supplier_id, company_name, country, sales_contact_name, email) VALUES
(1, 'Beans R Us', 'United States', 'Alison Kane', 'beansrus@gmail.com'),
(2, 'CreamerCO', 'Germany', 'John Fredrickson', 'creamerco@yahoo.com'),
(3, 'Coffee Pros', 'United Kingdom', 'Paul Grayson', 'coffeepros@gmail.com');

INSERT INTO COFFEE (coffee_id, shop_id, supplier_id, coffee_name, price_per_pound) VALUES
(1, 1, 2, 'Arabica', 1.50),
(2, 3, 1, 'Robusta', 2.98),
(3, 2, 1, 'Liberica', 3.29);

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;

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

Build Schema | Edit Fullscreen | Browser | [;]

```
SELECT *
FROM EMPLOYEE
WHERE employee_id  >  1;

```

Run SQL ▶ | Edit Fullscreen | [;]

employee_id	first_name	last_name	hire_date	job_title	shop_id
2	Denise	Fredrickson	2021-03-02	Barista	3
3	Miguel	Rodriguez	2023-01-01	Barista	2

Record Count: 2; Execution Time: 2ms View Execution Plan link

B6.

SELECT shop_name, company_name, coffee_name

FROM COFFEE C

INNER JOIN COFFEE_SHOP CS ON C.shop_id = CS.shop_id

INNER JOIN SUPPLIER S ON C.supplier_id = S.supplier_id;

```
(2, 'Coffee Oasis', 'Seattle', 'WA'),
(3, 'The Grind', 'Richmond', 'VA');

INSERT INTO EMPLOYEE (employee_id, first_name, last_name, hire_date, job_title, shop_id) VALUES
(1, 'George', 'Blake', '2016-12-01', 'Manager', 1),
(2, 'Denise', 'Fredrickson', '2021-03-02', 'Barista', 3),
(3, 'Miguel', 'Rodriguez', '2023-01-01', 'Barista', 2);

INSERT INTO SUPPLIER (supplier_id, company_name, country, sales_contact_name, email) VALUES
(1, 'Beans R Us', 'United States', 'Alison Kane', 'beansrus@gmail.com'),
(2, 'CreamerCO', 'Germany', 'John Fredrickson', 'creamerco@yahoo.com'),
(3, 'Coffee Pros', 'United Kingdom', 'Paul Grayson', 'coffeepros@gmail.com');

INSERT INTO COFFEE (coffee_id, shop_id, supplier_id, coffee_name, price_per_pound) VALUES
(1, 1, 2, 'Arabica', 1.50),
(2, 3, 1, 'Robusta', 2.98),
(3, 2, 1, 'Liberica', 3.29);

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;

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

Build Schema | Edit Fullscreen | Browser | [;]

```
SELECT shop_name, company_name, coffee_name
FROM COFFEE C
INNER JOIN COFFEE_SHOP CS ON C.shop_id = CS.shop_id
INNER JOIN SUPPLIER S ON C.supplier_id = S.supplier_id;

```

Run SQL ▶ | Edit Fullscreen | [;]

shop_name	company_name	coffee_name
Starbucks	CreamerCO	Arabica
The Grind	Beans R Us	Robusta
Coffee Oasis	Beans R Us	Liberica

Record Count: 3; Execution Time: 4ms | View Execution Plan | link