PART A: Database blueprints for Nora's Bagel Bin

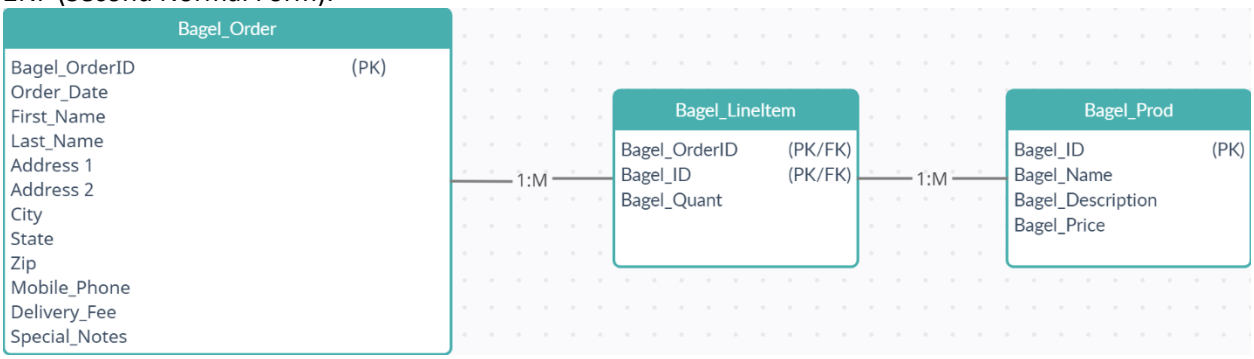
As a database designer/developer, I was supplied the bagel order form for Nora's Bagel Bin, which was already in the first normal form. The database structure could be further normalized to enhance the efficiency and functionality.

1NF (First Normal Form):

Bagel_Order	
Bagel_OrderID	(PK)
Bagel_ID	(PK)
Order_Date	
First_Name	
Last_Name	
Address 1	
Address 2	
City	
State	
Zip	
Mobile_Phone	
Delivery_Fee	
Bagel_Name	
Bagel_Description	
Bagel_Price	
Bagel_Quant	
Special_Notes	

In the first normal form, though all of the order form's fields are captured for further processing, the table containing that data contains a composite key, indicating multiple distinct entities in the table. Because of this, the data is functionally dependent on multiple entities. As such, the database should be normalized further to eliminate data redundancy.

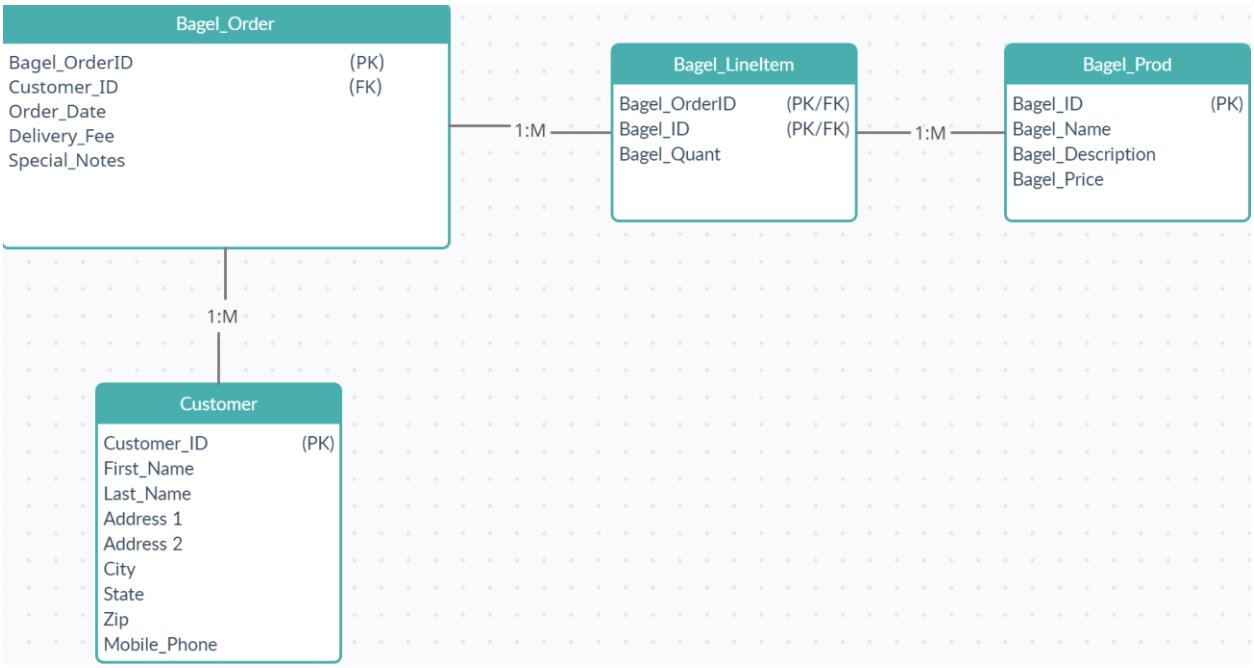
2NF (Second Normal Form):



The supplied blueprint has been divided into three tables and normalized in the second normal form. The Bagel_Order table includes data that will be distinctive to every individual order placed through Nora's Bagel Bin. The data associated to products that Nora's Bagel Bin offers was put into the Bagel_Prod table, since this information will remain consistent and not be unique to any single order it was more efficient to place it into its own table. The Bagel_LineItem table bridges the data between the Bagel_Order and Bagel_Prod tables. Having this data in its own table circumvents the many-to-many relationship between Bagel_Order and Bagel_Prod. Introducing the Bagel_Quant field within the Bagel_LineItem table enforces that each quantity of bagel is correlated to a single distinct order. This allows an immediate connection to its related table: Bagel_Prod.

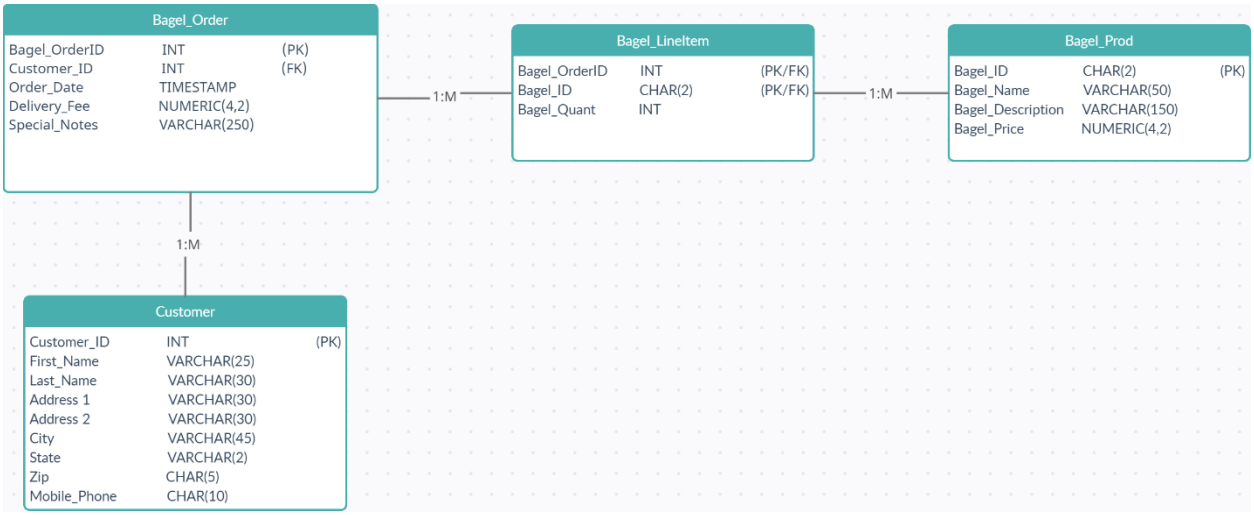
The associations to each table were dictated by the following:
• A single Bagel_Order will have many Bagel_LineItem's, but every line item can only be related to a single unique order because it is dependent on the Bagel_OrderID key.
• Although each Bagel_LineItem may be related to many different types of bagels, the Bagel_ID and Bagel_Quant fields guarantees each single occurrence of a Bagel_LineItem will be unconstrained by the bagels held in stock by Nora's Bagel Bin.

3NF (Third Normal Form):



When normalizing the database further to the third normal form, the additional tables inserted into the database by normalizing to the second normal form are still present. However, the Bagel_Order table is divided more by implementing the Customer table. Any single customer can place many Bagel_orders but every Bagel_Order will be related to only one single Customer. The creation of the fourth table allows Customers to survive independently of every order, which reduces data redundancy more by eliminating unnecessary data duplication of Customer information.

Final Database Model:



The final database model for Nora's Bagel Bin indicates the modifications made when putting the database into the third normal form and appropriately assigns data types with each field name for compatibility with SQL language rules.

PART B Jaunty Coffee Co. Database

1. **Develop SQL code to create each table as specified in the attached "Jaunty Coffee Co. ERD"**

```
/* Create Employee table */
CREATE TABLE EMPLOYEE
(
employee_id INTEGER PRIMARY KEY,
first_name VARCHAR(30),
last_name VARCHAR(30),
hire_date DATE,
job_title VARCHAR(30)
);
/* Create Coffee Shop table */
CREATE TABLE COFFEE_SHOP
(
shop_id INTEGER PRIMARY KEY,
shop_name VARCHAR(50),
city VARCHAR(50),
state CHAR(2)
);
/* Create Coffee table */
CREATE TABLE COFFEE
(
coffee_id INTEGER PRIMARY KEY,
coffee_name VARCHAR(30),
price_per_pound NUMERIC(5,2)
);
/* Create Supplier table */
CREATE TABLE SUPPLIER
(
supplier_id INTEGER PRIMARY KEY,
company_name VARCHAR(50),
country VARCHAR(30),
sales_contact_name VARCHAR(60),
email VARCHAR(50) NOT NULL
);
/* Add shop_id FK */
ALTER TABLE EMPLOYEE
ADD shop_id INTEGER,
ADD FOREIGN KEY(shop_id) REFERENCES COFFEE_SHOP(shop_id);
/* Add shop_id and supplier_id FKs */
ALTER TABLE COFFEE
ADD shop_id INTEGER,
ADD supplier_id INTEGER,
ADD FOREIGN KEY(shop_id) REFERENCES COFFEE_SHOP(shop_id),
ADD FOREIGN KEY(supplier_id) REFERENCES SUPPLIER(supplier_id);
```

Schema SQL

```sql
/* Create Employee table */
CREATE TABLE EMPLOYEE
(
employee_id INTEGER PRIMARY KEY,
first_name VARCHAR(30),
last_name VARCHAR(30),
hire_date DATE,
job_title VARCHAR(30)
);
/* Create Coffee Shop table */
CREATE TABLE COFFEE_SHOP
(
shop_id INTEGER PRIMARY KEY,
shop_name VARCHAR(50),
city VARCHAR(50),
state CHAR(2)
);
/* Create Coffee table */
CREATE TABLE COFFEE
(
coffee_id INTEGER PRIMARY KEY,
```

Text to DDL

Query SQL

```sql
SELECT * FROM EMPLOYEE;
SELECT * FROM COFFEE_SHOP;
SELECT * FROM COFFEE;
SELECT * FROM SUPPLIER;
```

Results

Query #1 Execution time: 0ms

There are no results to be displayed.

Query #2 Execution time: 0ms

There are no results to be displayed.

Query #3 Execution time: 1ms

There are no results to be displayed.

Query #4 Execution time: 0ms

There are no results to be displayed.

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

```
/* Insert example data for employee table */
INSERT INTO EMPLOYEE(employee_id, first_name, last_name, hire_date, job_title)
VALUES(133, "Jean", "Craigson", "1978-03-16", "Team Manager"),
(244, "Annie", "Jefferson", "1986-12-03", "Marketing Analyst"),
(355, "John", "Michaels", "1992-04-20", "Systems Administrator");
/* Insert example data for Coffee Shop table */
INSERT INTO COFFEE_SHOP(shop_id, shop_name, city, state)
VALUES(15643, "Jaunty New Haven Coffee Beanery", "New Haven", “CO”),
(25353, "Jaunty South Coffee Beanery", "Jacksonville", “FL”),
(35364, "Jaunty Midwest Coffee", "Chicago", “IL”);
/* Insert example data for Coffee table */
INSERT INTO COFFEE(coffee_id, coffee_name, price_per_pound)
VALUES(123, "Abyssal", 16.45),
(234, "Caramel Macchiato", 13.95),
(345, "Celeste", 14.55);
/* Insert example data for Supplier table */
INSERT INTO SUPPLIER(supplier_id, company_name, country, sales_contact_name, email)
VALUES(1234, "Coffee Beans R Us", "United States", "Abby Deans", "ADeans1@cbru.com"),
(2345, "Yum Yum Coffee", "China", "Jian Yang", "JianYang@yumyumcoffee.com"),
(3456, "The Beanz", "United States", "Mike Docker", "crichalds@thebeanz.rocks");
/* Set shop_id for each employee */
UPDATE EMPLOYEE
SET shop_id = 15643 WHERE employee_id = 133;
UPDATE EMPLOYEE
SET shop_id = 25353 WHERE employee_id = 244;
UPDATE EMPLOYEE
SET shop_id = 35364 WHERE employee_id = 355;
/* Set shop and supplier id for each coffee */
UPDATE COFFEE
SET shop_id = 15643, supplier_id = 1234 WHERE coffee_id = 345;
UPDATE COFFEE
SET shop_id = 25353, supplier_id = 2345 WHERE coffee_id = 234;
UPDATE COFFEE
SET shop_id = 35364, supplier_id = 3456 WHERE coffee_id = 123;
```

Schema SQL

```
country VARCHAR(30),
sales_contact_name VARCHAR(60),
email VARCHAR(50) NOT NULL
);
/* Add shop_id FK */
ALTER TABLE EMPLOYEE
ADD shop_id INTEGER,
ADD FOREIGN KEY(shop_id) REFERENCES COFFEE_SHOP(shop_id);
/* Add shop_id and supplier_id FKs */
ALTER TABLE COFFEE
ADD shop_id INTEGER,
ADD supplier_id INTEGER,
ADD FOREIGN KEY(shop_id) REFERENCES COFFEE_SHOP(shop_id),
ADD FOREIGN KEY(supplier_id) REFERENCES SUPPLIER(supplier_id);
/* Insert example data for employee table */
INSERT INTO EMPLOYEE(employee_id, first_name, last_name, hire_date, job_title)
VALUES(133, "Jean", "Craigson", "1978-03-16", "Team Manager"),
(244, "Annie", "Jefferson", "1986-12-03", "Marketing Analyst"),
(355, "John", "Michaels", "1992-04-20", "Systems Administrator");
/* Insert example data for Coffee Shop table */
INSERT INTO COFFEE_SHOP(shop_id, shop_name, city, state)
VALUES(15643, "Jaunty New Haven Coffee Beanery", "New Haven", 'CO').
```

Text to DDL

Query SQL

```
SELECT * FROM EMPLOYEE;
SELECT * FROM COFFEE_SHOP;
SELECT * FROM COFFEE;
SELECT * FROM SUPPLIER;
```

Results

Copy as Markdown

Query #1 Execution time: 1ms

employee_id	first_name	last_name	hire_date	job_title	shop_id
133	Jean	Craigson	1978-03-16	Team Manager	15643
244	Annie	Jefferson	1986-12-03	Marketing Analyst	25353
355	John	Michaels	1992-04-20	Systems Administrator	35364

Query #2 Execution time: 0ms

shop_id	shop_name	city	state
15643	Jaunty New Haven Coffee Beanery	New Haven	CO
25353	Jaunty South Coffee Beanery	Jacksonville	FL
35364	Jaunty Midwest Coffee	Chicago	IL

Query #3 Execution time: 0ms

coffee_id	coffee_name	price_per_pound	shop_id	supplier_id
123	Abyssal	16.45	35364	3456
234	Caramel Macchiato	13.95	25353	2345
345	Celeste	14.55	15643	1234

Query #4 Execution time: 0ms

supplier_id	company_name	country	sales_contact_name	email
1234	Coffee Beans R Us	United States	Abby Deans	ADeans1@cbru.com
2345	Yum Yum Coffee	China	Jian Yang	JianYang@yumyumcoffee.com
3456	The Beanz	United States	Mike Docker	crichalds@thebeanz.rocks

3. Develop SQL code to create a view showing all information from the EMPLOYEE table, with the new employee_full_name attribute

```
/* Create view to concatenate employee names */
CREATE VIEW Full_names AS
SELECT CONCAT(first_name, ' ', last_name) AS employee_full_name, employee_id, hire_date, job_title, shop_id
FROM EMPLOYEE
WHERE first_name IS NOT NULL AND last_name IS NOT NULL;
```

Results

Copy as Markdown

Query #1 Execution time: 0ms

employee_full_name	employee_id	hire_date	job_title	shop_id
Jean Craigson	133	1978-03-16	Team Manager	15643
Annie Jefferson	244	1986-12-03	Marketing Analyst	25353
John Michaels	355	1992-04-20	Systems Administrator	35364

4. Develop SQL code to create an index on the coffee_name field from the COFFEE table

```
/* Create an index on the coffee_name field */
CREATE INDEX coffee_index
ON COFFEE (coffee_name);
```

Schema SQL

```
/* Add shop_id and supplier_id FKs */
ALTER TABLE COFFEE
  ADD shop_id INTEGER,
  ADD supplier_id INTEGER,
  ADD FOREIGN KEY(shop_id) REFERENCES COFFEE_SHOP(shop_id),
  ADD FOREIGN KEY(supplier_id) REFERENCES SUPPLIER(supplier_id);

/* Insert example data for employee table */
INSERT INTO EMPLOYEE(employee_id, first_name, last_name, hire_date, job_title)
VALUES(133, "Jean", "Craigson", "1978-03-16", "Team Manager"),
(244, "Annie", "Jefferson", "1986-12-03", "Marketing Analyst"),
(355, "John", "Michaels", "1992-04-20", "Systems Administrator");
/* Insert example data for Coffee Shop table */
INSERT INTO COFFEE_SHOP(shop_id, shop_name, city, state)
VALUES(15643, "Jaunty New Haven Coffee Beanery", "New Haven", 'CO'),
(25353, "Jaunty South Coffee Beanery", "Jacksonville", 'FL'),
(35364, "Jaunty Midwest Coffee", "Chicago", 'IL');
/* Insert example data for Coffee table */
INSERT INTO COFFEE(coffee_id, coffee_name, price_per_pound)
VALUES(123, "Abyssal", 16.45),
(234, "Caramel Macchiato", 13.95),
(345, "Celeste", 14.55);
```

Text to DDL

Query SQL

```
SELECT coffee_name FROM COFFEE
```

Results

Query #1 Execution time: 0ms

coffee_name
Abyssal
Caramel Macchiato
Celeste

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

```
/* Create a SFW query for Coffee Shop table */
SELECT * FROM COFFEE_SHOP
WHERE state = 'FL'
```

Schema SQL

```
71 UPDATE EMPLOYEE
72 SET shop_id = 15643 WHERE employee_id = 133;
73 UPDATE EMPLOYEE
74 SET shop_id = 25353 WHERE employee_id = 244;
75 UPDATE EMPLOYEE
76 SET shop_id = 35364 WHERE employee_id = 355;
77 /* Set shop and supplier id for each coffee */
78 UPDATE COFFEE
79 SET shop_id = 15643, supplier_id = 1234 WHERE coffee_id = 345;
80 UPDATE COFFEE
81 SET shop_id = 25353, supplier_id = 2345 WHERE coffee_id = 234;
82 UPDATE COFFEE
83 SET shop_id = 35364, supplier_id = 3456 WHERE coffee_id = 123;
84 /* Create view to concatenate employee names */
85 CREATE VIEW Full_names AS
86 SELECT CONCAT(first_name, ' ', last_name) AS employee_full_name, employee_id, hire_date, job_title, shop_id
87 FROM EMPLOYEE
88 WHERE first_name IS NOT NULL AND last_name IS NOT NULL;
89 /* Create an index on the coffee_name field */
90 CREATE INDEX coffee_index
91 ON COFFEE (coffee_name);
```

Text to DDL

Query SQL

```
1 SELECT * FROM COFFEE_SHOP
2 WHERE state = 'FL'
```

Results

Query #1 Execution time: 0ms

shop_id	shop_name	city	state
25353	Jaunty South Coffee Beanery	Jacksonville	FL

6. Develop SQL code to create a query joining three different tables, including attributes from all three

```
/* Create a query to join three tables and include all attributes */
SELECT * FROM EMPLOYEE
JOIN COFFEE_SHOP
ON EMPLOYEE.shop_id = COFFEE_SHOP.shop_id
JOIN COFFEE
ON COFFEE SHOP.shop id = COFFEE.shop id
```

Schema SQL

```
71 UPDATE EMPLOYEE
72 SET shop_id = 15643 WHERE employee_id = 133;
73 UPDATE EMPLOYEE
74 SET shop_id = 25353 WHERE employee_id = 244;
75 UPDATE EMPLOYEE
76 SET shop_id = 35364 WHERE employee_id = 355;
77 /* Set shop and supplier id for each coffee */
78 UPDATE COFFEE
79 SET shop_id = 15643, supplier_id = 1234 WHERE coffee_id = 345;
80 UPDATE COFFEE
81 SET shop_id = 25353, supplier_id = 2345 WHERE coffee_id = 234;
82 UPDATE COFFEE
83 SET shop_id = 35364, supplier_id = 3456 WHERE coffee_id = 123;
84 /* Create view to concatenate employee names */
85 CREATE VIEW Full_names AS
86 SELECT CONCAT(first_name, ' ', last_name) AS employee_full_name, employee_id, hire_date, job_title, shop_id
87 FROM EMPLOYEE
88 WHERE first_name IS NOT NULL AND last_name IS NOT NULL;
89 /* Create an index on the coffee_name field */
90 CREATE INDEX coffee_index
91 ON COFFEE (coffee_name);
```

Text to DDL

Query SQL

```
1 /* Create a query to join three tables and include all attributes */
2 SELECT * FROM EMPLOYEE
3 JOIN COFFEE_SHOP
4 ON EMPLOYEE.shop_id = COFFEE_SHOP.shop_id
5 JOIN COFFEE
6 ON COFFEE_SHOP.shop_id = COFFEE.shop_id
7
```

Results

Copy as Markdown

Query #1 Execution time: 1ms

employee_id	first_name	last_name	hire_date	job_title	shop_id	shop_id	shop_name	city	state	coffee_id	coffee_name	price_per_pound	shop_id	supplier_id
133	Jean	Craigson	1978-03-16	Team Manager	15643	15643	Jaunty New Haven Coffee Beanery	New Haven	CO	345	Celeste	14.55	15643	1234
244	Annie	Jefferson	1986-12-03	Marketing Analyst	25353	25353	Jaunty South Coffee Beanery	Jacksonville	FL	234	Caramel Macchiato	13.95	25353	2345
355	John	Michaels	1992-04-20	Systems Administrator	35364	35364	Jaunty Midwest Coffee	Chicago	IL	123	Abyssal	16.45	35364	3456