**Paul Overfelt III**

**WGU C170 PA**

**1A,1B**

**Nora’s Bagel Bin Database Blueprints**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **BAGEL ORDER** | |  | **BAGEL ORDER LINE ITEM** | |  | **BAGEL** | |
| PK | Bagel Order ID |  | PK / FK | Bagel Order ID |  | PK | Bagel ID |
|  | Order Date | 1:M | PK / FK | Bagel ID | M:1 |  | Bagel Name |
|  | First Name |  |  | Bagel Quantity |  |  | Bagel Price |
|  | Last Name |  |  |  |  |  | Bagel Description |
|  | Address 1 |  |  |  |  |  |  |
|  | Address 2 |  |  |  |  |  |  |
|  | City |  |  |  |  |  |  |
|  | State |  |  |  |  |  |  |
|  | Zip |  |  |  |  |  |  |
|  | Mobile Phone |  |  |  |  |  |  |
|  | Delivery Fee |  |  |  |  |  |  |
|  | Special Notes |  |  |  |  |  |  |

**1C**

**Attributes were assigned to each table for 2NF based on their relationship with a Primary Key. For 2NF all non-key values must be completely dependent on the full Primary Key of the table. For table Bagel Order all the values (Order Date, First Name, Last Name, etc.) are dependent on the Order so they are dependent are the Order ID. The same goes for Table Bagel. The name, price, and description are dependent on the Bagel ID.**

**How the cardinality between the tables was determined is, for every Order ID, there could be many line Items, so you have a one-to-many relationship. For Bagel Order Line Item to Bagel, there could be many line items that will point to one Bagel, so it is a Many to one relationship.**

**2A,2B,2C,2D**

**Nora’s Bagel Bin Database Blueprints**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **BAGEL ORDER** | | |  | **BAGEL ORDER LINE ITEM** | |  | **BAGEL** | |
| PK | Bagel Order ID | |  | PK / FK | Bagel Order ID |  | PK | Bagel ID |
| FK | Customer ID | | 1:M | PK / FK | Bagel ID | M:1 |  | Bagel Description |
|  | Order Date | |  |  | Bagel Quantity |  |  | Bagel Price |
|  | Delivery Fee | |  |  |  |  |  | Bagel Name |
|  | Special Notes | |  |  |  |  |  |  |
|  | M:1 |  |  |  |  |  |  |  |
| **Customer** | | |  |  |  |  |  |  |
| PK | Customer ID | |  |  |  |  |  |  |
|  | First Name | |  |  |  |  |  |  |
|  | Last Name | |  |  |  |  |  |  |
|  | Address 1 | |  |  |  |  |  |  |
|  | Address 2 | |  |  |  |  |  |  |
|  | City | |  |  |  |  |  |  |
|  | State | |  |  |  |  |  |  |
|  | Zip | |  |  |  |  |  |  |
|  | Mobile Phone | |  |  |  |  |  |  |

2E

For 3NF the Bagel Order Table was broken down into Bagel Order and Customer Tables. This is due to 3NF requiring no attributes to be transitively on other attributes not part of the PK. Since the attributes First Name, Last Name, Address, etc. relate more to the customer than the order, we can move them into their table to reduce the amount of information redundancy in the Bagel Order table.

As for the cardinality between the tables. Bagel Order – Customer is a Many-to-One relationship because each customer can have many orders. The other relationships do not change going from 2NF to 3NF.

**3A,3B**

**Nora’s Bagel Bin Database Blueprints**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Bagel Order** | | |  | **BAGEL ORDER LINE ITEM** | | |  | **BAGEL** | |  |
| PK | bagel\_order\_id | INTEGER |  | PK / FK | bagel\_order\_id | INTEGER |  | PK | bagel\_id | INTEGER |
| FK | customer\_id | INTEGER | 1:M | PK / FK | bagel\_id | INTEGER | M:1 |  | name | VARCHAR(50) |
|  | orderDate | TIMESTAMP |  |  | bagelQuantity | INTEGER |  |  | price | NUMERIC(2,2) |
|  | dFee | NUMERIC(2,2) |  |  |  |  |  |  | description | VARCHAR(100) |
|  | notes | VARCHAR(100) |  |  |  |  |  |  |  |  |
|  | M:1 |  |  |  |  |  |  |  |  |  |
| **Customer** | | |  |  |  |  |  |  |  |  |
| PK | id | INTEGER |  |  |  |  |  |  |  |  |
|  | fName | VARCHAR(50) |  |  |  |  |  |  |  |  |
|  | lName | VARCHAR(50) |  |  |  |  |  |  |  |  |
|  | address1 | VARCHAR(100) |  |  |  |  |  |  |  |  |
|  | address2 | VARCHAR(100) |  |  |  |  |  |  |  |  |
|  | city | VARCHAR(50) |  |  |  |  |  |  |  |  |
|  | state | CHAR(2) |  |  |  |  |  |  |  |  |
|  | zip | CHAR(5) |  |  |  |  |  |  |  |  |
|  | phoneNumber | NUMBERIC(10,0) |  |  |  |  |  |  |  |  |

B1a,B1b

CREATE DATABASE JauntyCoffee;

USE JauntyCoffee;

CREATE TABLE Coffee\_Shop (

shop\_id INT PRIMARY KEY,

shop\_name VARCHAR(50),

city VARCHAR(50),

state CHAR(2)

);

CREATE TABLE Supplier (

supplier\_id INT PRIMARY KEY,

company\_name VARCHAR(50),

country VARCHAR(30),

sales\_contact\_name VARCHAR(60),

email VARCHAR(50) NOT NULL

);

CREATE TABLE Employee (

employee\_id INT PRIMARY KEY,

first\_name VARCHAR(30),

last\_name VARCHAR(30),

hire\_date DATE,

job\_title VARCHAR(30),

shop\_id INT,

FOREIGN KEY (shop\_id) REFERENCES Coffee\_Shop(shop\_id)

);

CREATE TABLE Coffee (

coffee\_id INT PRIMARY KEY,

shop\_id INT,

supplier\_id INT,

coffee\_name VARCHAR(30),

price\_per\_pound NUMERIC(5,2),

FOREIGN KEY (shop\_id) REFERENCES Coffee\_Shop(shop\_id),

FOREIGN KEY (supplier\_id) REFERENCES Supplier(supplier\_id)

);

Graphical user interface, application

Description automatically generated

B2a,B2b

INSERT INTO Coffee\_Shop

VALUES

(1, 'Crave Coffee', 'Blacksburg', 'VA'),

(2, 'Coffeeistic', 'Charlotte', 'NC'),

(3, 'Coffeeorzo', 'Roanoke', 'VA');

INSERT INTO Supplier

VALUES

(1, 'Coffee.com', 'USA', 'Margo Barrow', 'margo@coffee.com'),

(2, 'Bestbeans.com', 'UK', 'Abby Mora', 'abby@bestbeans.com'),

(3, 'Worstbeans.net', 'Germany', 'Halle Mcclure', 'halle@worstbeans.net');

INSERT INTO Employee

VALUES

(1, 'Allana', 'Combs', '2021-11-05', 'Barista', 1),

(2, 'Keisha', 'Singleton', '2021-11-05', 'Barista', 2),

(3, 'Harleigh', 'Foley', '2021-06-02', 'Barista', 3);

INSERT INTO Coffee

VALUES

(1, 1, 1, 'Robusta', 19.3),

(2, 2, 2, 'Liberica', 24.99),

(3, 3, 3, 'Excelsa', 8.95);

Graphical user interface, text

Description automatically generated

B3a,B3b

CREATE VIEW employee\_info AS

SELECT employee\_id, CONCAT(first\_name, ' ', last\_name) AS employee\_full\_name, hire\_date , job\_title, shop\_id

FROM Employee;

Graphical user interface, text, application, Word

Description automatically generated

B4a,B4b

CREATE INDEX idx\_coffee

ON Coffee(coffee\_name);

Graphical user interface, text, application

Description automatically generated

B5a,B5b

SELECT \*

FROM Coffee\_Shop

WHERE state = 'VA';

Graphical user interface, application

Description automatically generated

B6a,B6b

SELECT Supplier.sales\_contact\_name AS 'Contact Name', Coffee\_Shop.shop\_name AS 'Shop Name', Coffee.coffee\_name AS 'Coffee Name'

FROM Coffee

INNER JOIN Supplier

ON Coffee.supplier\_id = Supplier.supplier\_id

INNER JOIN Coffee\_Shop

ON Coffee\_Shop.shop\_id = Coffee.shop\_id;

Graphical user interface, text, application

Description automatically generated