PART A: Nora’s bagel bin database blueprint

Robert Gal

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 |

This is the original table in first normal form, 1NF.

Second normal form, 2NF:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **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\_description |
|  | Last\_Name |  |  |  |  |  | Bagel\_price |
|  | Address 1 |  |  |  |  |  |  |
|  | Address 2 |  |  |  |  |  |  |
|  | City |  |  |  |  |  |  |
|  | State |  |  |  |  |  |  |
|  | Zip |  |  |  |  |  |  |
|  | Mobile\_Phone |  |  |  |  |  |  |
|  | Delivery\_Fee |  |  |  |  |  |  |
|  | Special\_Notes |  |  |  |  |  |  |

This is the second normal form, 2NF. We have to split the original 1NF to qualify for the 2NF by creating two new tables “Bagel Order Line Item” and “Bagel”. This will remove partial dependencies as all attributes must to depend on their primary keys. “Bagel Order” connects to “Bagel Order Line Item” in one to many relationship, one order can have many items but items only can have one order. “Bagel Order Line Item” connects to the “Bagel” table with a many to one relationship. Order items can have only one bagel, and one bagel can have many order items.

“Bagel order” table has a primary key “Bagel\_Order\_ID” and the other necessary attributes included in the table. “Bagel Order Line Item” table has 3 columns with 2 primary keys “Bagel\_Order\_ID” and “Bagel\_ID”. They are foreign keys (composite keys) as well pointing to the other two tables. We have “Quantity” attribute which is dependent on the primary keys. The “Bagel” table has a primary key(Bagel\_id) which is the foreign key in the “Bagel Order Line Item” table. All other attributes are dependent on the bagel\_id primary key.

Third normal form, 3NF:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ORDER INFORMATION** | | |  | **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\_Name |
|  | Order\_Date | |  |  | Bagel\_Quantity |  |  | Bagel\_Description |
|  | Delivery\_Fee | |  |  |  |  |  | Bagel\_Price |
|  | Special\_Notes | |  |  |  |  |  |  |
|  | M:1 |  |  |  |  |  |  |  |
| **CUSTOMER INFORMATION** | | |  |  |  |  |  |  |
| PK | Customer\_ID | |  |  |  |  |  |  |
|  | First Name | |  |  |  |  |  |  |
|  | Last Name | |  |  |  |  |  |  |
|  | Address 1 | |  |  |  |  |  |  |
|  | Address 2 | |  |  |  |  |  |  |
|  | City | |  |  |  |  |  |  |
|  | State | |  |  |  |  |  |  |
|  | Zip | |  |  |  |  |  |  |
|  | Mobile\_Phone | |  |  |  |  |  |  |

This is the third normal form, 3NF. In order to qualify for 3NF we need to create a new table “customer information”. The customer information attributes did not relate to the Bagel\_Order\_ID, and every non key attributes need to be defined by the primary key. The first 3 tables cardinality did not change. The new “customer information” table has a many to one relationship. Orders only have one customer, while one customer can have many orders.

The new “customer information” table has a primary key(customer\_id). We have to create a foreign key in the “order information” table in order to connect the two tables. All other attributes in the new table dependent on the primary key “customer\_ID”.

Physical database:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ORDER INFORMATION** | | |  | **BAGEL ORDER LINE ITEM** | | |  | **BAGEL** | |  |
| PK | bagel\_order\_id | INT |  | PK / FK | bagel\_order\_id | INT |  | PK | bagel\_id | CHAR(2) |
| FK | customer\_id | INT | 1:M | PK / FK | bagel\_id | INT | M:1 |  | bagel\_name | VARCHAR(30) |
|  | order\_date | TIMESTAMP |  |  | bagel\_quantity | INT |  |  | bagel\_desc | VARCHAR(30) |
|  | delivery\_fee | NUMERIC(2,2) |  |  |  |  |  |  | bagel\_price | NUMERIC(3,2) |
|  | special\_notes | VARCHAR(90) |  |  |  |  |  |  |  |  |
|  | M:1 |  |  |  |  |  |  |  |  |  |
| **CUSTOMER INFORMATION** | | |  |  |  |  |  |  |  |  |
| PK | customer\_id | INT |  |  |  |  |  |  |  |  |
|  | first\_name | VARCHAR(30) |  |  |  |  |  |  |  |  |
|  | last\_name | VARCHAR(30) |  |  |  |  |  |  |  |  |
|  | address\_1 | VARCHAR(90) |  |  |  |  |  |  |  |  |
|  | address\_2 | VARCHAR(90) |  |  |  |  |  |  |  |  |
|  | City | VARCHAR(30) |  |  |  |  |  |  |  |  |
|  | State | CHAR(2) |  |  |  |  |  |  |  |  |
|  | Zip | CHAR(5) |  |  |  |  |  |  |  |  |
|  | mobile\_phone | CHAR(10) |  |  |  |  |  |  |  |  |

This is the final database with all appropriate tables, columns and their cardinalities.

PART B: Jaunty Coffee Co.ERD

1.Develop SQL code to create each table as specified:

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text, application

Description automatically generated

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

Graphical user interface, text

Description automatically generated

3.Develop SQL code to create View:

Graphical user interface, text, application

Description automatically generated

4.Develop SQL code to create index:

Graphical user interface, text, application, email

Description automatically generated

5.Develop SQL code to create an SFW:

Graphical user interface, text, application, email

Description automatically generated

6.Develop SQL code to create a query:

Graphical user interface, application

Description automatically generated

Another SQL code with Join:

Graphical user interface, application

Description automatically generated