

ArDKEEN quality fOOD STORES

Database Systems



May 1, 2022

DAire o’neill

20019346 IT2

**Table of Contents**

Business Description: 1

EER model: 2

Tables: 2-4

Storage representations: 4-8

Total storage: 8

Table design: 9-10

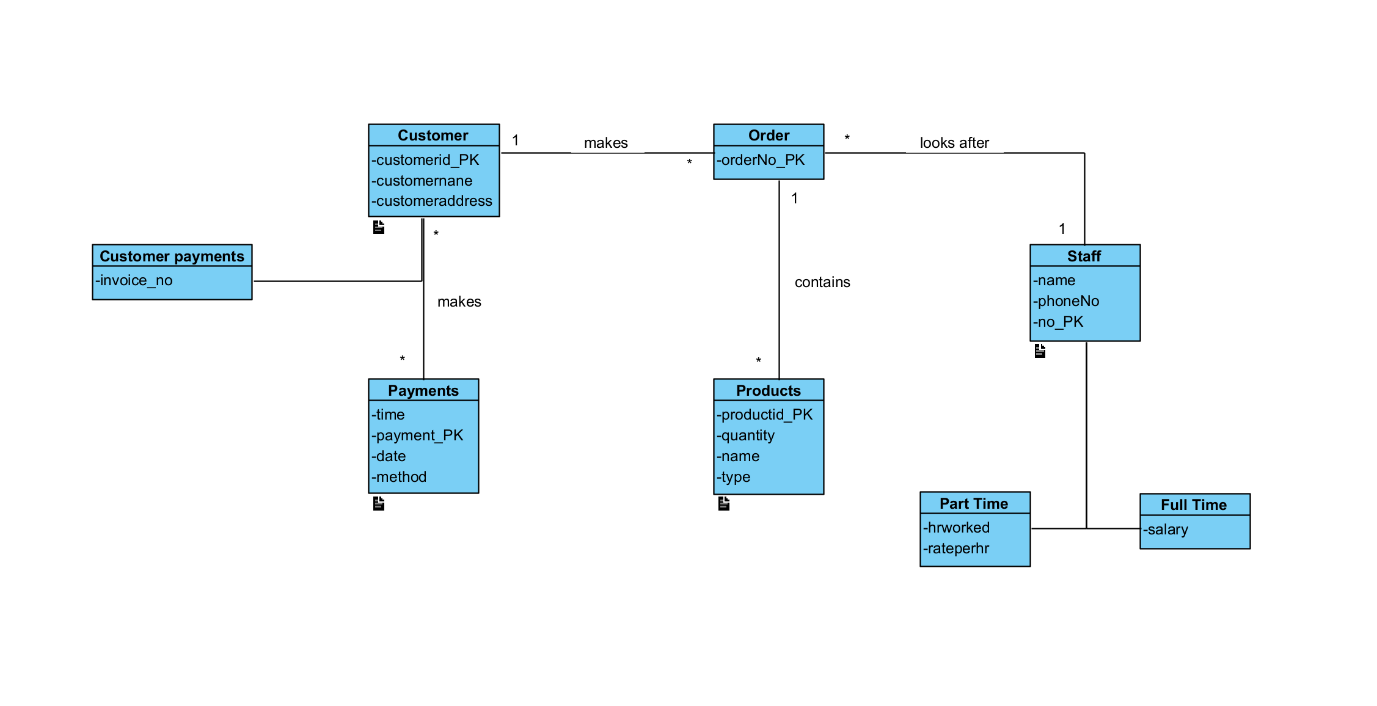
Queries: 11

Conclusion: 11

**Business Description**

Ardkeen Quality Food Stores is a local artisan quality food store. They are located just outside Ardkeen Hospital have has been a family run business since it was founded in 1967. I worked in Ardkeen Stores for over two years and have a strong understanding of the shop and website. The website doesn’t need to store a too much data, just products currently in stock as well as the username and password for customers shopping online. They do however a moderate amount of data saved with RSS, products, shelfs, prices etc. My database could be managed by the shop themselves and should be quite useful.

**EER Model**



**Tables**

Products

One table

Product: {supplier, quantity, productid, name}

Primary Key: productid

Staff

There are three options here

One table

Staff: {name, phoneNo, no, hrworked, rateperhr, salary,}

Primary Key: no

Two tables

Part Time: {name, phoneNo, no, hrworked, rateperhr}

Primary key: no

Full Time: {name, phoneNo, no, salary}

Primary key: no

Three tables

Staff: {name, phoneNo, no}

Primary key: no

Part Time: {hrworked, rateperhr}

Primary key: no

Full Time: {name, phoneNo, no, salary}

Primary key: no

Customer

There are three options here

One table

Customer: {customerid, customername, customeraddress, time, payment, date, method, invoice\_no}

Primary key: cusomterid

Two tables

Customer payments: {customerid, customername, customeraddress, invoice\_no}

Primary key: customerid

Payments: {customerid, customername, customeraddress, time, payment, date, method}

Primary key: customerid

Three tables

Customer: {customerid, customername, customeraddress}

Primary key: customerid

Customer payments: {invoice\_no}

Primary key: customerid

Payments: {time, payment, date, method}

Primary key: customerid

Order

There is only one option here

One table

Order: {oderNo}

Primary key: ordrNo

**Storage Representation**

This section will break down all the attributes in each table and the amount of bytes needed for these attributes, which will then be calculated to figure out how much space is required for each table. For each superclass with subclasses I will go through all the possible table options that were stated in the previous section, and compare the amount of bytes needed for each option. I will then choose the table option with the least amount of required storage and calculate the total storage required for all the chosen tables.

**Product**

|  |  |  |
| --- | --- | --- |
| Attributes | Bytes needed | Brief description |
| Productid | 30 | Short code |
| Quantity | 9999 | Up to 9999 of a product can be stored |
| type | 10 | Short code |
| Name | 20 | name |
| Total bytes | 1059 |  |

On average there would be 100,000 products sold in a week

100,000 \* 1059 = 105,900,000

I would need 105,900,000 bytes of storage

**Staff**

|  |  |  |
| --- | --- | --- |
| Attributes | Bytes needed | Brief description |
| Name | 40 | full name |
| phoneNo | 10 | Phone number |
| no | 400 | Unique staff number |
| Total bytes | 450 |  |

**Part Time**

|  |  |  |
| --- | --- | --- |
| Attributes | Bytes needed | Brief description |
| hrworked | 99 | From 0-99 hours worked a week |
| rateperhr | 99 | Up to $99 an hour |
| Total bytes | 198 |  |

**Full Time**

|  |  |  |
| --- | --- | --- |
| Attributes | Bytes needed | Brief description |
| Salary | 6 | Number (5,0) eg 20,000 |
| Total bytes | 6 |  |

There are 126 employees with 80 part-time and 46 full-time employees

The three options are:

Option 1: all the attributes are in the staff table

450 + 198 + 6 = 654

654 \* 126 = 82,404

For this option I would need 82,404 bytes of storage

Option 2: two tables

Part time

450 + 198 = 648

648 \* 80 = 51,840

Full time

450 + 6 = 456

456 \* 46 = 20,976

20,976 + 51,840 = 72,816

For this option I would need 72,816 bytes of storage

Option 3: three tables

Employee

450 \* 126 = 56,700

Part time

400 (primary key) + 198 = 598

598 \* 126 = 75,348

Full time

400 (primary key) + 6 = 406

406 \* 126 = 51,156

51,156 + 75,348 + 56,700 = 183,204

For this option I would need 183,204 bytes of storage, making option 2 optimal

**Customer**

|  |  |  |
| --- | --- | --- |
| Attributes | Bytes needed | Brief description |
| Customerid | 99,999 | Unique number for each customer who signs up to the website |
| Customername | 30 | Full name |
| Customeraddress | 100 | Full address |
| Total bytes | 100129 |  |

**Customer payments**

|  |  |  |
| --- | --- | --- |
| Attributes | Bytes needed | Brief description |
| Invoice\_no | 999999 | Large number of invoices |
| Total bytes | 999999 |  |

**Payments**

|  |  |  |
| --- | --- | --- |
| Attributes | Bytes needed | Brief description |
| Time | 86400 | Every time of the day including seconds |
| payment | 999999 | Unique number for all payments, reset every day |
| Date | 365 | All dates in the year |
| Method | 5 | Cash, card etc |
| Total bytes | 2086768 |  |

There would be an average of 5,000 online orders thus payments a week

The three options are:

Option 1: all the attributes are in the Customer table

100,129 + 999,999 + 2,086,768 = 3,186,896

3,186,896 \* 5000 = 15,834,480,000

I would need 15,834,480,000 bytes of data for this option

Option 2: two tables

Customer payments

100,129 + 999,999 = 1,100,128

1,100,128 \* 5000 = 5,500,640,000

Payments

100,139 + 2,086,768 = 2,186,897

2,186,897 \* 5000 = 10,934,485,000

10,934,485,000 + 5,500,640,000 = 16,435,125,000

I would need 16,435,125,000 bytes of data for this option

Option 3: three tables

Customer

100,129 \* 5000 = 500,645,000

Customer payments

99,999 (primary key) + 999,999 = 1,099,998

1,099,998 \* 5000 = 5,499,990,000

Payments

99,999 (primary key) + 2,086,768 = 2,186,767

2,186,767 \* 5000 = 10,933,835,000

10,933,835,000 + 5,499,990,000 + 500,645,000 = 16,934,470,000

I would need 16,934,470,000 bytes of data for this option

**Order**

|  |  |  |
| --- | --- | --- |
| Attributes | Bytes needed | Brief description |
| orderNo | 999999 | Large number or orders |
| Total bytes | 999999 |  |

There is an average of 14 orders a week from the shop

There is one option

999,999 \* 14 = 13,999,986

I would need 13,999,986 bytes of data for this option

**Total Storage**

I chose the table options that required the smallest amount of storage from each superclass. I did this as smaller storage space will allow for faster data retrieval from the database. The smallest storage option for each superclass was the option with each subclass having their own table with all their own attributes and the superclass attributes also. The total storage for every table in the database is:

105,900,000 + 72,816 + 15,834,480,000 + 13,999,986 = 15,954,452,802 bytes

**Table Design**

**Product**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field | Type | Size | Null/ not null | Index | Description |
| Productid | Varchar | 30 | Not null | Pk | The unique number given to the product |
| Quantity | Int |  | Not null |  | The number of items held |
| Typea | Varchar | 10 |  |  | The type of product |
| Namea | Varchar | 20 | Not null |  | The name of the product |

**Staff**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field | Type | Size | Null/ not null | Index | Description |
| Nameb | varchar | 40 |  |  | Full name of staff member |
| phoneno | varchar | 10 |  |  | Phone number of staff member |
| noa | varchar | 400 |  | pk | Id number for staff |

**Parttime**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field | Type | Size | Null/ not null | Index | Description |
| hrworked | varchar | 99 | Not null |  | The amount of hours worked that week |
| rateperhr | varchar | 30 | Not null |  | The amount the staff member makes per hour |

**Fulltime**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field | Type | Size | Null/ not null | Index | Description |
| salary | Varchar | 6 | Not null |  | The annual salary the staff member makes |

**Customer**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field | Type | Size | Null/ not null | Index | Description |
| customerid | bigint |  | Not null | pk | The unioque number given to each customer |
| Customername | varcahar | 30 |  |  | The name of the customer |
| customeraddress | varchar | 100 | Not null |  | The address of the customer |

**Customerpayments**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field | Type | Size | Null/ not null | Index | Description |
| invoiceno | Bigint |  | Not null |  | The unique invoice number |

**Payments**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field | Type | Size | Null/ not null | Index | Description |
| timea | text |  |  |  | The time the payment was made |
| payment | bigint |  | Not null |  | The unique number the payment was given |
| datea | varchar | 365 |  |  | The date of the payment |
| method | varchar | 5 | Not null |  | The method of the payment |

**Ordera**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field | Type | Size | Null/ not null | Index | Description |
| orderno | bigint |  | Not null | pk | The unique order number |

**Queries**

Show how much stock is available

Change the level of stock

Find the name, phone number and staff number for a member of staff

Find the address and customer id of a customer

Find the order number

**Conclusion**

This database design document has covered a lot of topics and will help in the implementation of the database. I think my database design will be of benefit to them, but I may have underestimated how storage this local shop needs to hold. Me being familiar with the store has made this job much more thorough. I believe this was quite challenging but after completing it, this would be much easier with the knowledge I’ve gained form creating this.