Introduction to Databases: CA2

Student: Andrew Kealy
Student number: 18159761

Project description:
Using a modified version of the database created in CA1 (taking into account the feedback from that CA), I populated the database with data from mockaroo.com , I wrote scripts to create reports and charts.

1. Create a Datamart from the guitarShop2 database and create a table that shows the transactions for the first week of January.

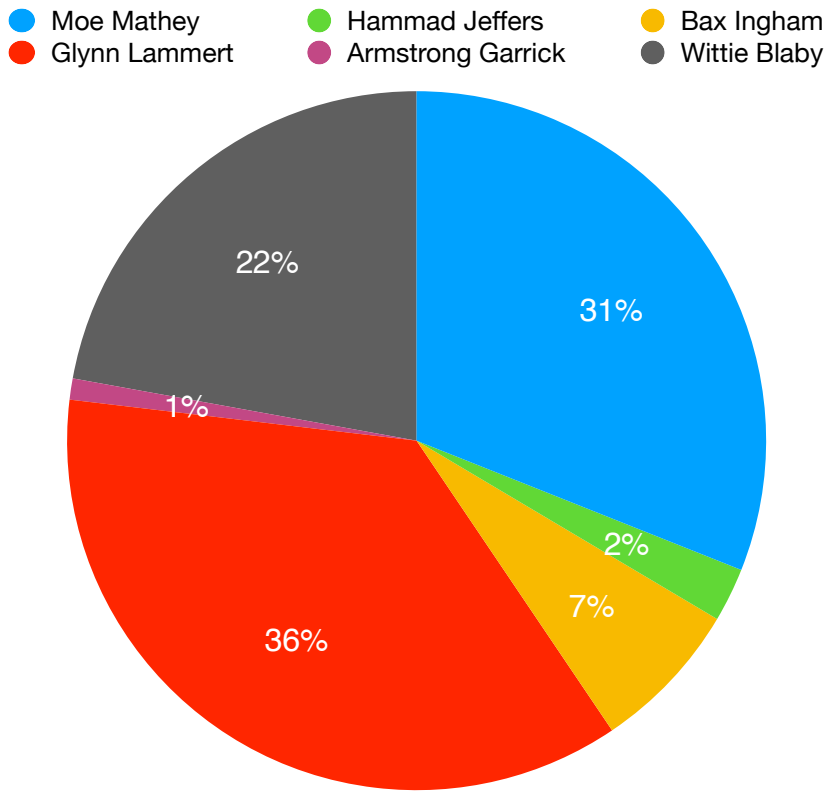
```
use guitarshop2;
CREATE database Datamart;
CREATE Table Datamart.transactions SELECT CONCAT(customers.fname, ' ', customers.lname)
as Customer,
orders.product_id, products.retail_price, orders.staff_id, CONCAT(staff.fname, ' ', staff.lname) as
Employee, amount, (orders.amount * products.retail_price) as order_value, order_date
FROM orders
JOIPN customers
ON orders.customer_id = customers.customer_id
JOIN staff
ON orders.staff_id = staff.staff_id
JOIN products
ON orders.product_id = products.product_id
WHERE order_date BETWEEN '2018-01-01' AND '2018-01-08'
ORDER BY order_date;
```

This script generates the following data:

transactions

Customer	product_id	retail_price	staff_id	Employee	amount	order_value	order_date
Moe Mathey	92	€3,248.00	20	Eirena Turk	33	€107184.00	2018-01-02
Hammad Jeffers	86	€2,154.00	5	Cairistiona MacConneely	4	€8616.00	2018-01-02
Bax Ingham	3	€2,697.00	4	Jermaine Dowson	9	€24273.00	2018-01-03
Glynn Lammert	61	€2,918.00	10	Em Walkden	43	€125474.00	2018-01-03
Armstrong Garrick	73	€3,381.00	19	Granger Chaperlin	1	€3381.00	2018-01-06
Wittie Blaby	82	€2,550.00	1	Adriane Haycox	30	€76500.00	2018-01-06

From this data I created the following chart, showing which employees were responsible for sale



2: I created a trigger that automatically reduces the amount of a product in stock when an order is placed.

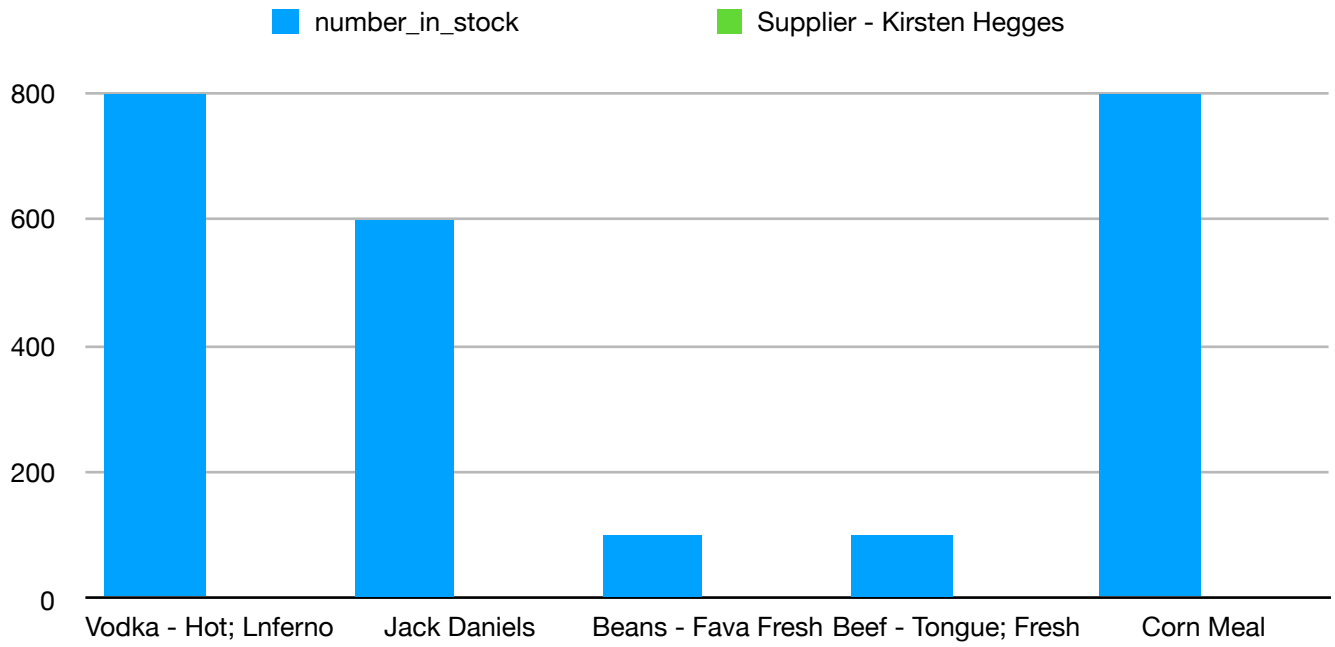
```
CREATE TRIGGER after_order_update
AFTER INSERT ON orders
 FOR EACH ROW
 UPDATE products
 SET number_in_stock = number_in_stock - NEW.amount
 WHERE product_id = NEW.product_id;
```

The products table has a number_in_stock column that is reduced by the amount of items ordered each time an order is placed.

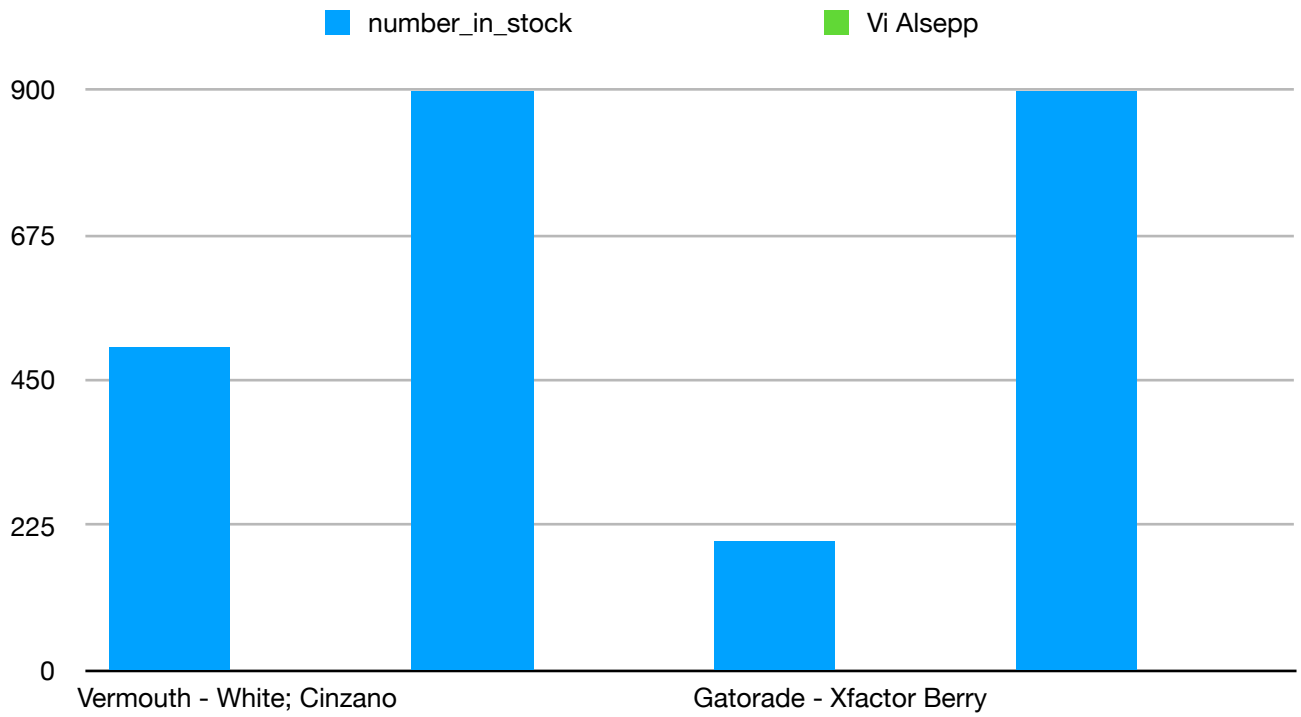
3: The following script creates a view that lists all the products in stock by their suppliers.

```
USE guitarShop2;
CREATE VIEW Stock AS
SELECT product_model, number_in_stock, CONCAT(suppliers.fname, ' ', suppliers.lname) as
Supplier
FROM products
JOIN suppliers
ON products.supplier_id = suppliers.supplier_id;
```

This script generates data that allows us to produce charts based on stock by supplier, such as:



or this:



4: The following script shows us all sales by employee. It uses the sum function so add up their sales. It is grouped by employee and uses rollup to give the total sales.

```
USE Datamart;
USE guitarShop2;
CREATE TABLE Datamart.sales_by_staff  SELECT CONCAT(staff.fname, ' ', staff.lname) AS
Employee,  SUM(orders.amount * products.retail_price) as order_value
FROM orders
JOIN products
ON orders.product_id = products.product_id
JOIN staff sales_by_staff
ON orders.staff_id = staff.staff_id
WHERE order_date BETWEEN '2018-01-01' AND '2019-05-01'
GROUP BY Employee WITH ROLLUP;
```

From the data generated in this table, the following chart can be created, showing a breakdown of employee sales performance: