Data Normalisation - Cafe Transactions

Note: When describing tables, *underscored_italicised* denotes primary keys, *italicised* denotes foreign keys

25/08/2021 09:0	Chesterfield	[redacted]	Regular Flavoured iced latte - Hazelnut - 2.75, Large Latte - 2.45	5.2	CARD	[redacted]
25/08/2021 09:0	Chesterfield		Large Flavoured iced latte - Caramel - 3.25, Regular Flavoured iced latte - Hazelnut - 2.75, Regular Flavoured iced latte - Caran	17.3	CARD	
25/08/2021 09:0	Chesterfield		Large Flat white - 2.45, Regular Latte - 2.15	4.6	CARD	
25/08/2021 09:0	Chesterfield		Regular Flavoured latte - Hazelnut - 2.55, Large Latte - 2.45	5	CARD	
25/08/2021 09:0	Chesterfield		Regular Latte - 2.15, Large Latte - 2.45	4.6	CASH	
25/08/2021 09:1	Chesterfield		Large Flavoured iced latte - Caramel - 3.25, Regular Flat white - 2.15, Regular Latte - 2.15, Large Flavoured iced latte - Hazelnu	12.95	CASH	
25/08/2021 09:1	Chesterfield		Large Flavoured latte - Hazelnut - 2.85, Regular Flavoured iced latte - Vanilla - 2.75, Large Flavoured iced latte - Hazelnut - 3.25	17.4	CARD	
25/08/2021 09:1	Chesterfield		Regular Flavoured iced latte - Vanilla - 2.75, Regular Flavoured iced latte - Vanilla - 2.75, Large Latte - 2.45, Large Flavoured lat	13.55	CARD	
25/08/2021 09:1	Chesterfield		Regular Flavoured iced latte - Caramel - 2.75, Large Latte - 2.45, Regular Latte - 2.15, Large Latte - 2.45, Regular Flavoured ice	12.55	CARD	
25/08/2021 09:1	Chesterfield		Large Flat white - 2.45, Large Flavoured latte - Hazelnut - 2.85	5.3	CARD	

Snippet of CSV file with personally identifiable information obscured

1NF

1. ***Using row order to convey information is not permitted***

25/08/2021 15:10	Chesterfield	Regular Flavoured latte - Hazelnut - 2.55, Regular Flat white - 2.15, Regular Flat white - 2.15, L
25/08/2021 15:11	Chesterfield	Large Flavoured latte - Hazelnut - 2.85, Large Flavoured latte - Hazelnut - 2.85
25/08/2021 15:12	Chesterfield	Regular Latte - 2.15, Regular Flat white - 2.15, Regular Flavoured iced latte - Hazelnut - 2.75
25/08/2021 15:13	Chesterfield	Large Flavoured latte - Hazelnut - 2.85
25/08/2021 15:15	Chesterfield	Large Flavoured iced latte - Vanilla - 3.25, Large Flavoured iced latte - Vanilla - 3.25, Large Flat
25/08/2021 15:16	Chesterfield	Regular Flat white - 2.15, Large Flat white - 2.45
25/08/2021 15:17	Chesterfield	Large Latte - 2.45
25/08/2021 15:18	Chesterfield	Regular Flavoured iced latte - Caramel - 2.75, Large Latte - 2.45, Regular Flavoured iced latte -

Row order does not determine anything.

2. ***Mixing data types within the same column is not permitted***
No issues with current data, although it is important to ensure the types for each column when designing schema

3. ***Having a table without a primary key is not permitted***
transaction_id needed for each transaction- Time not used as a unique identifier as there could be multiple transactions made within the same minute.

transaction_id	timestamp	store_name	products (Split by comma to separate out each product in order, once complete, splt by - to sepa
1	25/08/2021 09:00	Chesterfield	Regular Flavoured iced latte - Hazelnut - 2.75, Large Latte - 2.45
2	25/08/2021 09:02	Chesterfield	Large Flavoured iced latte - Caramel - 3.25, Regular Flavoured iced latte - Hazelnut - 2.75, Regu
3	25/08/2021 09:04	Chesterfield	Large Flat white - 2.45, Regular Latte - 2.15

Resulting table and headers:
Transaction(*transaction_id*, timestamp, store_name, products, total_price, payment_method)

4. ***Repeating groups is not permitted***

Regular Flavoured iced latte - Hazelnut - 2.75, Large Latte - 2.45
Large Flavoured iced latte - Caramel - 3.25, Regular Flavoured iced latte - Hazelnut - 2.75, Regu
Large Flat white - 2.45, Regular Latte - 2.15
Regular Flavoured latte - Hazelnut - 2.55, Large Latte - 2.45
Regular Latte - 2.15, Large Latte - 2.45
Large Flavoured iced latte - Caramel - 3.25, Regular Flat white - 2.15, Regular Latte - 2.15, Larg

There could be multiple entries/a list within each cell, possible solutions include:

1. **Creating separate columns for each product ordered**
   Not ideal as a single transaction could have 100 products and that would require 100 columns - difficult to design a table with no fixed upper bound
2. **Creating a separate table where each entry is is attributed to a single transaction and contains a single product**
   Ideal solution as the number of products ordered does not impact the number of columns required.

Resulting tables and headers:

1. **Transactions** (*transaction_id*, timestamp, store_name, total_price, payment_method)

transaction_id	**timestamp**	**store_name**	**total_price**	**payment_method**
1	1/2/2022 08:40	Chesterfield	12.5	card
2	1/2/2022 08:46	Chesterfield	10.7	cash

2. **Products** (*product_id,* product_name, price)

product_id	**product_name**	**price**
1	Regular flat white	2.1
2	Large flat white - Hazelnut	2.8

3. **Baskets** (*basket_id, transaction_id*, *product_id*)

basket_id	***transaction_id***	***product_id***
1	2	1
2	2	2
3	2	1

One transaction can be linked to many products through the basket

2NF

1. ***Each non-key attribute in the table must be dependent on the entire primary key***

Transactions table

Attribute	**Dependent on and unique to *transaction_id*?**
timestamp	Yes, a single transaction occurs at a single, specific time
store_name	No, as a store can have more than one transaction

total_price	Yes, each transaction has a different total (dependent on the total cost of the products ordered)
payment_method	No, as there are two payment options available, cash or card and more than one transaction can use each method

“No” indicates that this data should be presented in its own table):

transaction_id	timestamp	store_name	product	total_price	payment_metho
1	25/08/2021 09:00	Chesterfield	Regular Flavoured iced latte - Hazelnut - 2.75, L	5.2	CARD
2	25/08/2021 09:02	Chesterfield	Large Flavoured iced latte - Caramel - 3.25, Reg	17.3	CARD
3	25/08/2021 09:04	Chesterfield	Large Flat white - 2.45, Regular Latte - 2.15	4.6	CARD

Each colour denotes separate table

Resulting new tables and headers:

- **Stores** (*store_id*, store_name)

store_id	**store_name**
1	Chesterfield

- **Payment_method** (*payment_method_id*, payment_method)

payment_method_id	**payment_method**
1	CASH

Updates to existing tables:

- **Transactions** (*transaction_id*, timestamp, *store_id*, total_price, *payment_method_id*)

transaction_id	**timestamp**	***store_id***	**total_price**	***payment_method_id***
1	1/2/2022 08:40	Chesterfield	12.5	card
2	1/2/2022 08:46	Chesterfield	10:7	cash

Unchanged tables:

- **Products** (product_id, *product_name*, price)
- **Basket** (*basket_id*, *transaction_id*, *product_id*)

Products table

Attribute	**Dependent on and unique to *product_id*?**
product_name	Yes
price	Yes, the price is dependent on the product_id

Baskets table
Not assessed as no non-key attributes in this table

Stores table

Attribute	**Dependent on and unique to *store_id*?**
store_name	Yes, the store name is dependent on the store_id

Payment method table

Attribute	**Dependent on and unique to *payment_method_id*?**
payment_method	Yes, the payment method is dependent on the payment_method_id

3NF

1. ***Each non-key attribute in the table must depend on the key, the whole key, and nothing but the key***

Transactions table

Attribute	**Dependent on anything other than *transaction_id*?**
timestamp	No
store_id	No
total_price	No, as this data is provided by the CSV.
payment_method_id	No

Products table

Attribute	**Dependent on anything other than *product_id*?**
product_name	No
price	No

Baskets table
Not assessed as no non-key attributes in this table

Stores table

Attribute	**Dependent on anything other than *store_id*?**
store_name	No

Payment methods table

Attribute	**Dependent on anything other than *payment_method_id*?**
payment_method	No

To ask product owner:

- **Breaking product down further:** If café suddenly chooses to add a new size or wants to change the name of a size (e.g. Large -> Grande) have to then update every entry in the table with that name manually rather than changing it in one place

  Proposed table structures

  1. **Moderately atomic:** Size is assumed to be a relatively static element so no individual table for it, however seasonal flavours may be added and dropped therefore they have their own table

     Flavour (*flavour_id,* flavour)

flavour_id	**flavour**
1	Hazelnut
2	Caramel

     Product (*product_id*, product_type, *flavour_id,* price)

prod_id	**prod_type**	***flavour_id***	**price**
1	Regular Flat White		2.1
2	Regular Latte		2.1
3	Large Flavoured Latte	3	2.45
4	Regular Flavoured Latte	1	2:45

     Pros:
     - Greater flexibility in ability to add flavours
     - Reduces likelihood of update anomalies

     Cons:
     - Poorer join time than keeping it simple with a single entry for product name as initially proposed (has to join up an additional table to query product table)?
     - Consider this in the broader context of product being related to the basket table which is then related to individual transactions. E.g. How often is a “Large flavoured latte - hazelnut” ordered across all stores

  2. **Higher granularity/atomic products table -** Each attribute of product (size, type and flavour) are contained within their own tables.

Size (*size_id*, size)

size_id	**size**
1	Large
2	Regular

Flavour (*flavour_id,* flavour)

flavour_id	**flavour**
1	Hazelnut
2	Caramel

Product Type (*product_type_id,* product_type)

product_type_id	**product_type**
1	Latte
2	Flavoured iced latte

Product (*product_id*, *size_id,* product_type, *flavour_id,* price)

prod_id	***size_id***	**prod_type**	***flavour_id***	**price**
1	1	1		2.1
2	2	1		2.45
3	2	2	3	2.45
4	2	2	1	2:45

Pros:

- Greater flexibility for adding new sizes and flavours
- Reduces likelihood of update anomalies (e.g. to change large to grande)

Cons:

- Worst join time (has to join up to two additional tables to query product table)?
    - Consider this in the broader context of product being related to the transaction table which is then related to individual baskets. E.g. How often is a “Large flavoured latte - hazelnut” ordered across all stores? How many tables need to be accessed to generate that information?
- Too specific, too soon, is this level of atomicity essential to answering the questions the business may ask?

Discussion with Product Owner/Tech Lead

What does the business want to know?

1. What's the best selling product across all stores in a given month? (Freebie, what's the worst selling across all stores in a given month)
2. What product type is the most popular (reg, large? Latte, flat white?) across all stores?
3. Which store is the most profitable? (Freebie, least profitable)
4. When is the 'rush hour' of the business?
5. Which payment type is more popular by store?
6. How does price influence popularity (if at all?)
7. Are people mostly buying one item or multiple?
    a. How does this impact the transaction?
8. What's the most popular coffee flavour per store in the summer?
9. Does an iced drink sell better than a non-iced drink in summer?
10. Future thinking**:** Knowing which region is the most profitable?

11. **Ubiquitous language** - Making sure the team and the business are using the same words when referring to objects. E.g. What is a product? What is an order? How do we define a transaction?

    Discussion about what to name the tables:

    "A customer places an order for a product or products and then pays for it. The transaction is when the payment is received in exchange for the products. A single transaction consists of a basket containing multiple products which are linked to the transaction."

    - **Transaction** refers to the individual interaction between customer and store in which the payment is received for products
    - **Products** unlike in the original CSV where 'products' referred to the items ordered, this now refers to the individual drinks that are available to be ordered, derived from the products listed in the csv
    - **Basket** refers to the what drinks the customer requests as part of their transaction

    Discussion about level of atomicity needed for products table

    - After considering the join times and how this could impact querying the data, we agreed to stick with the originally suggested table