Company 1

Product Analytics Challenge

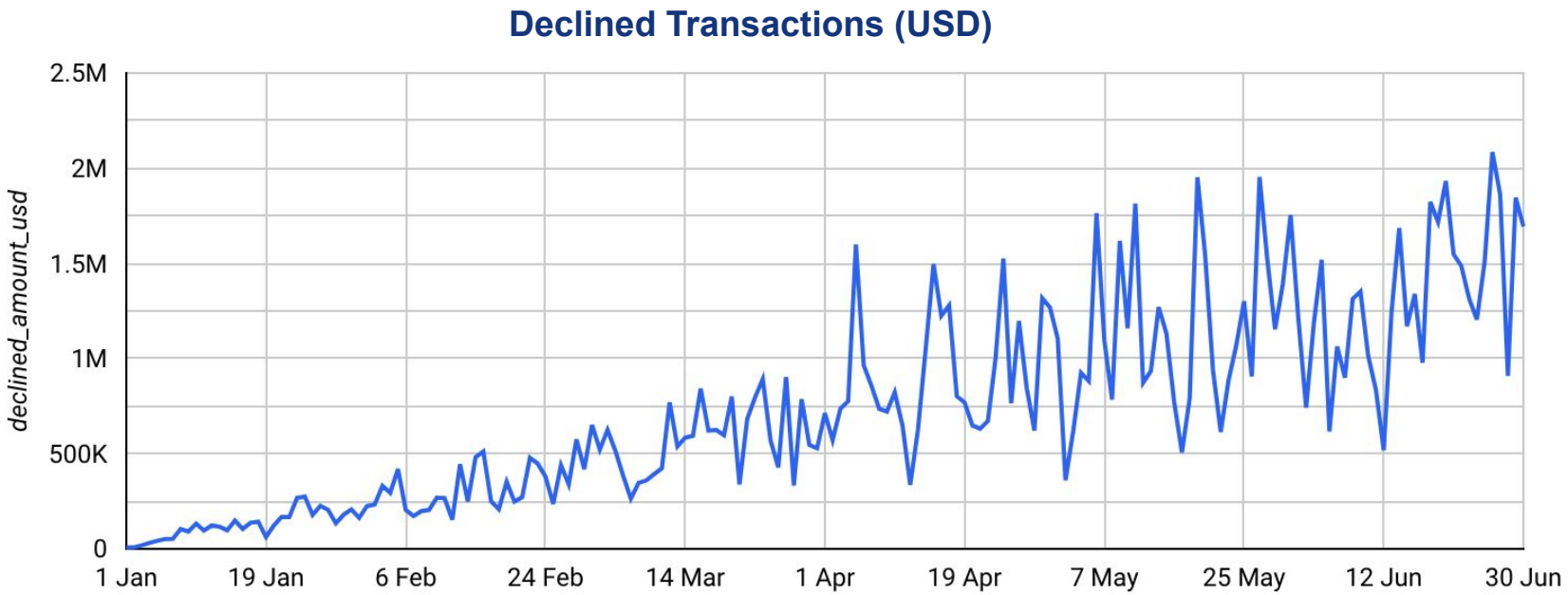
Ben Winby

Task 1

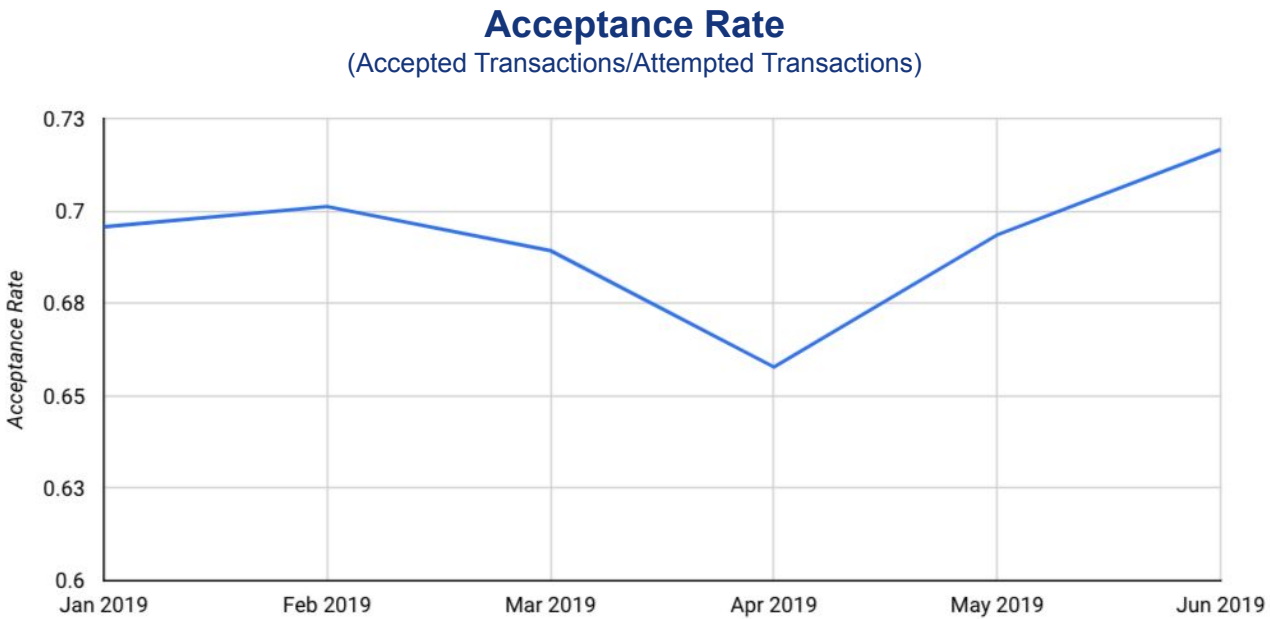
1. Present the volume of declined payments in USD
2. Analyse the root causes for the decline in the acceptance rate
3. Provide well-justified solutions, recommendations, and next steps that you would take if given more time, additional data, and deeper business knowledge.

Present the volume of declined payments in USD

$132m of transactions have been declined



Acceptance rate over this period has remained relatively steady



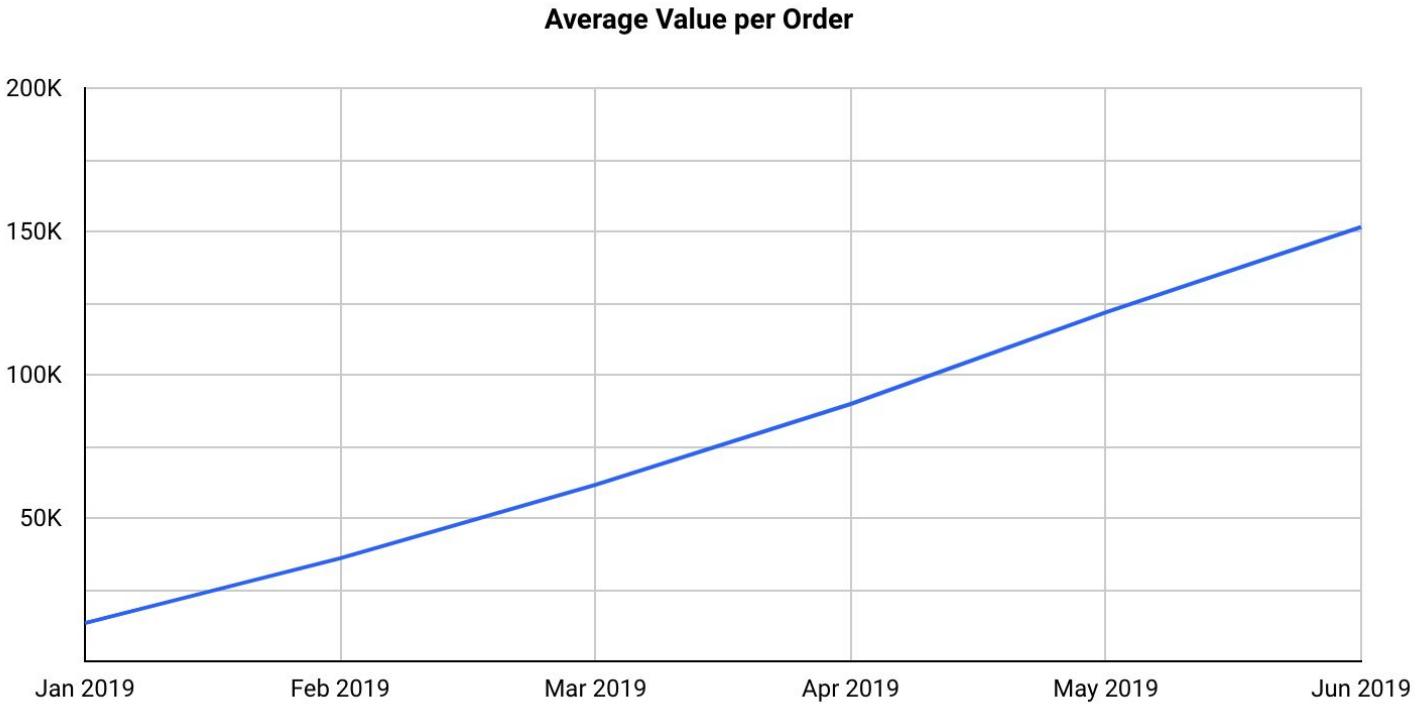
The data doesn't show a decline in Acceptance Rate.

There is a small drop in April - but it recovers and is not statistically significant.

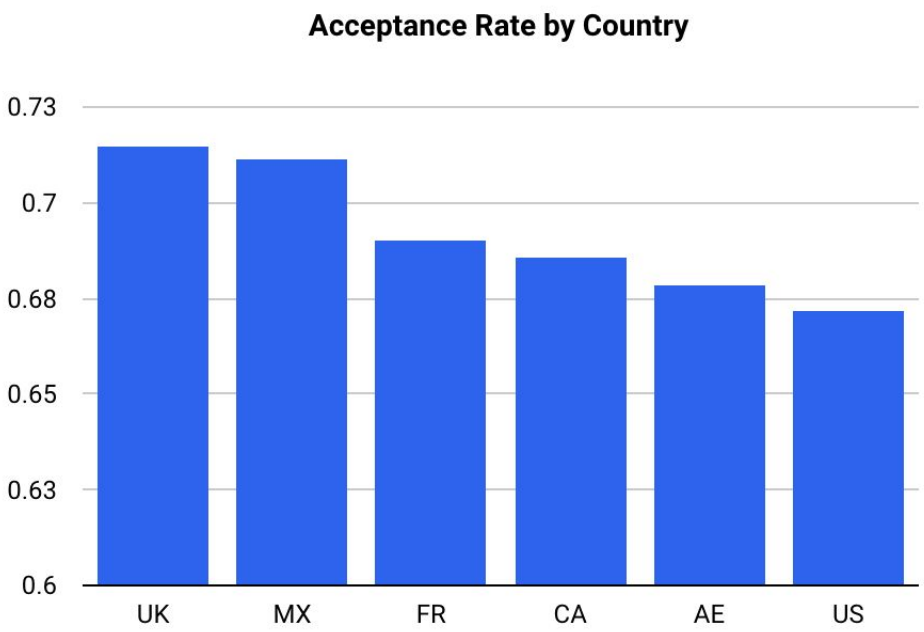
The following analysis focuses on where the major opportunities are and where we should focus our efforts.

Note: y axis starts at 60%

The increase in declined payment value is being driven by an increased Average Order Value



Acceptance Rate is lowest in US, UAE and Canada



Transactions are typically declined either for:

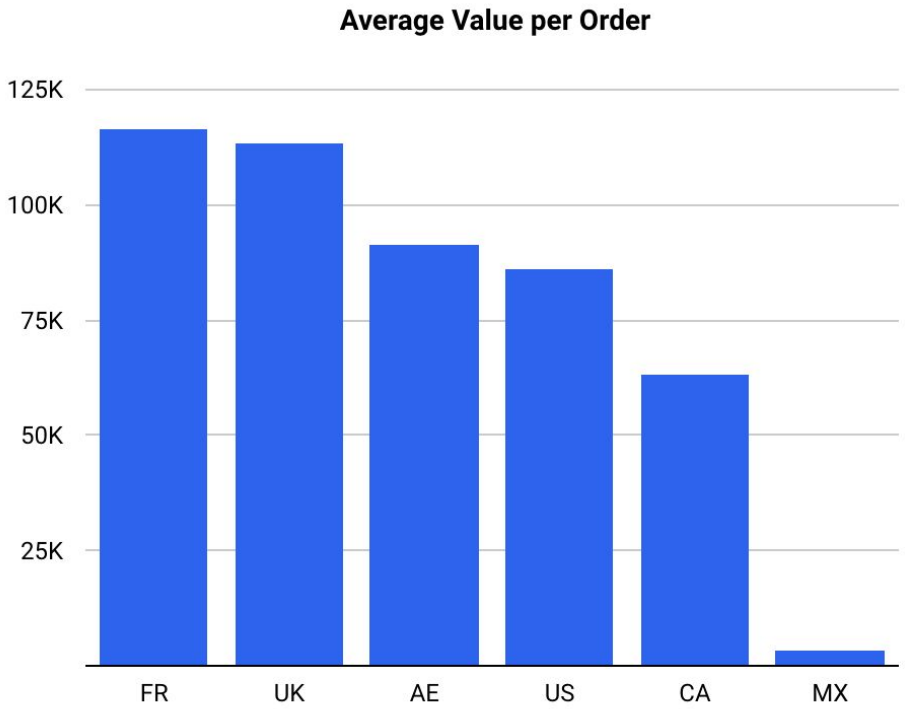
- **insufficient funds**
- **incorrect data being supplied**

Incorrect data is something we should be able to influence through clearer messaging and improved UI.

Next Steps:

- Request data from Globepay on split between “insufficient funds” vs “incorrect data” to understand the driver
- Request data from Globepay on industry averages by region
- If cause is “incorrect data” then this potentially points to a localisation issue which we can dig into further

France has the largest Value per Order

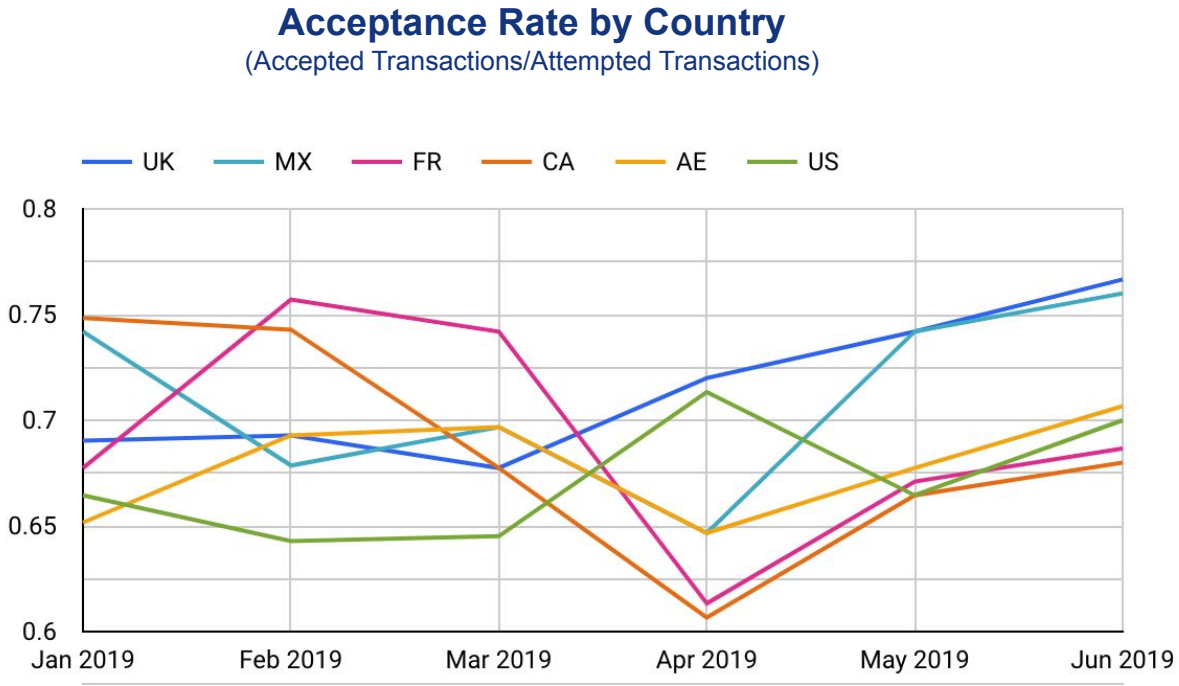


France has the largest value per order - but is middle of the pack in terms of Acceptance Rates.

Next Steps

Focus our efforts on France as this is where we will see the biggest return

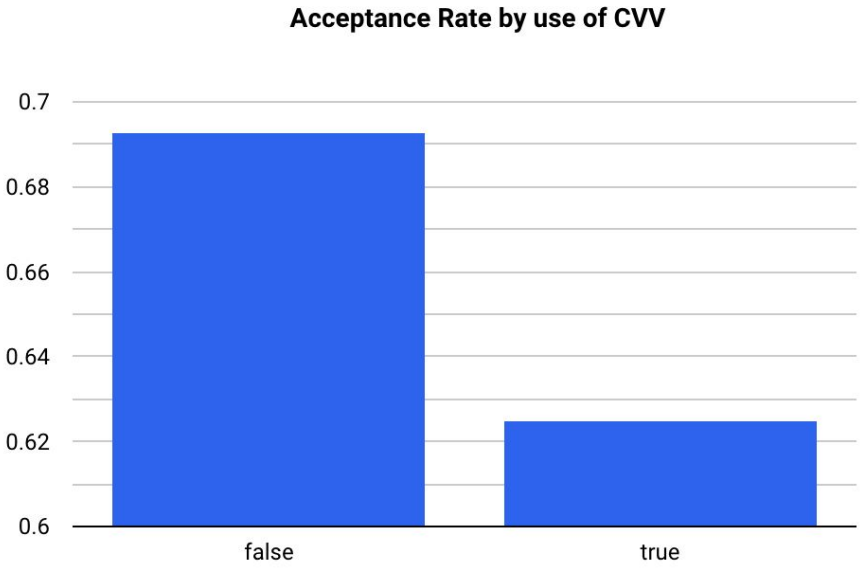
The slight decline in April is being driven by France and Canada



Next Steps:

Investigate if there were any changes during April that might have impacted this behaviour - in particular changes to the payment pages

Those who provide a CVV code have a lower acceptance rate - however CVV is optional and only provided 0.7% of the time



Declined orders from those using CVV is about 1% of declined orders but still worth $2.5m.

Entering the CVV is optional as such it seems odd that this would be causing people to get declined. The declines associated with this are most likely to be a result of the wrong CVV being entered rather than fraud.

Next Steps

Evaluate the messaging around CVV to understand why people are entering the incorrect number

Evaluate the feasibility of removing CVV

France is the only country that is actively using the CVV

	false		true		Grand total	
country	Acceptance Rate	attempted_cou…	Acceptance Rate	attempted_cou…	**Acceptance Rate**	**attempted_count**
UK	0.71	905	-	-	**0.71**	**905**
MX	0.71	905	-	-	**0.71**	**905**
FR	0.69	867	0.63	38	**0.69**	**905**
CA	0.69	905	-	-	**0.69**	**905**
AE	0.68	904	1	1	**0.68**	**905**
US	0.67	904	0	1	**0.67**	**905**

Declines as a result of incorrect CVV doesn't materially impact the overall acceptance rate for France.

And this does not explain the drop in April.

This isn't a huge win but might be something to investigate if we do plan to enforce CVV collection in the future.

Next Steps

Understand why France is using CVV more than other regions

Next Steps

High Priority

- Request data from Globepay on split between “insufficient funds” vs “incorrect data” to understand the driver
- Request data from Globepay on industry averages by region
- If “incorrect data” is reason behind then this potentially points to a localisation issue which we can dig into further
- Focus our efforts on France as this is where we will see the biggest return
- Investigate if there were any changes during April that might have impacted this behaviour - in particular changes to the payment pages

Low Priority

- Evaluate the messaging around CVV to understand why people are entering the incorrect number
- Evaluate the feasibility of removing CVV
- Understand why France is using CVV more than other regions

Task 2

1. Calculate and present the acceptance rate over time.
2. List the countries where the amount (in dollars) of declined transactions went over $25M
3. Identify transactions from the Acceptance report that are missing chargeback data.

Preparation

This query is used to clean up the data to create a view which can then be used as the basis for all other queries.

Its primary job is to extract the correct FX rate for each row.

This is the resulting schema:

Field name	Type	Mode
external_ref	STRING	NULLABLE
status	BOOLEAN	NULLABLE
source	STRING	NULLABLE
ref	STRING	NULLABLE
date_time	TIMESTAMP	NULLABLE
state	STRING	NULLABLE
cvv_provided	BOOLEAN	NULLABLE
amount	FLOAT	NULLABLE
country	STRING	NULLABLE
currency	STRING	NULLABLE
fx_rate	FLOAT	NULLABLE

Note: I have loaded the data into BigQuery and my project has been named `nodal-alloy-399422`

```
with base as (

 SELECT
  *
  , split(rates, ",") as all_fx_rates

 FROM `nodal-alloy-399422.company_1.acceptances`

), expanded_fx_rates as (

 select
  * except(all_fx_rates)
  , replace(replace(replace(single_fx_rates, '"', ''), '{', ''), '}', '') as cleaned_fx_rate

 from base, unnest(all_fx_rates) as single_fx_rates

) , relevant_fx_rates as (

 select
  *
  , cast(right(cleaned_fx_rate, length(cleaned_fx_rate) - 4) as float64) as fx_rate

 from expanded_fx_rates

 where regexp_contains(single_fx_rates, currency) = TRUE
)

select
 * except(rates, single_fx_rates, cleaned_fx_rate)

from relevant_fx_rates
```

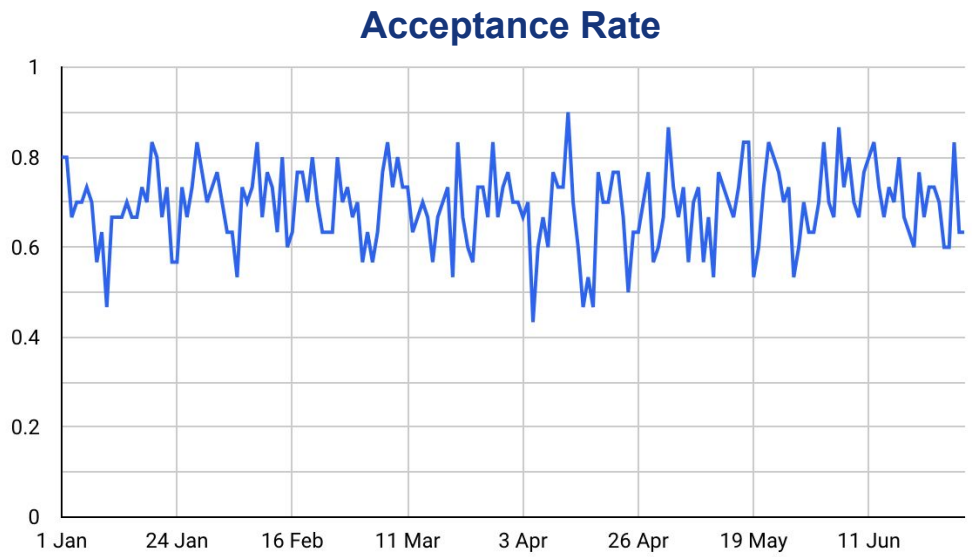
Acceptance Rate over Time

```
SELECT

 date_trunc(date_time, day) as date_day
 , count(case when state = 'ACCEPTED' then external_ref else null end)/ count(external_ref) as acceptance_rate

FROM `nodal-alloy-399422.company_1.acceptances_cleaned_fx`

group by 1
order by 1 asc
```



Countries where the amount of declined transactions went over $25M

```
with country_level as (

 SELECT

  country
  , sum(case when state = 'DECLINED' then amount/fx_rate else null end) as declined_amount_usd

 FROM `nodal-alloy-399422.company_1.acceptances_cleaned_fx`

 group by 1

)

select *

from country_level

where declined_amount_usd > 25000000
```

Row	country	declined_amount_usd
1	AE	26335152.430000003
2	FR	33737897.918340035
3	UK	27489496.685772821
4	US	25125669.780000005

Identify transactions from the Acceptance report that are missing chargeback data

```sql
with base as (

 SELECT

  a.*
  , c.chargeback

 FROM `nodal-alloy-399422.company_1.acceptances_cleaned_fx` a

 left join `nodal-alloy-399422.company_1.chargebacks` c
  on a.external_ref = c.external_ref

)

select *

from base

where chargeback is null
```

This does not return any data

All transactions have chargeback data associated with it

Task 3

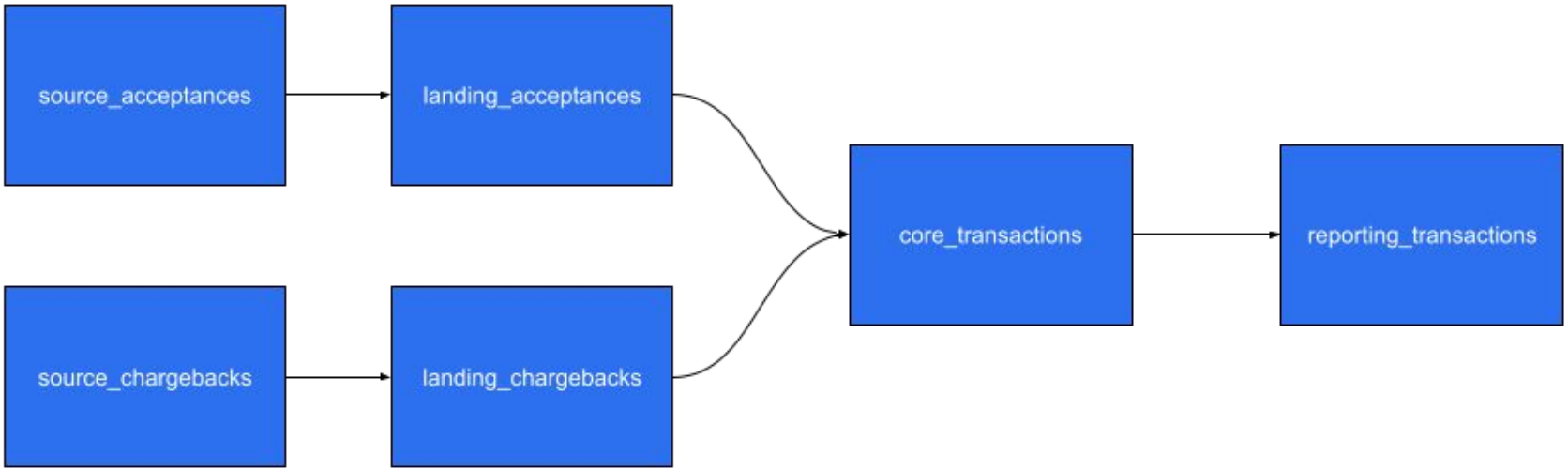
Imagine the two provided datasets as source tables in a production database. Please provide a data lineage of the data pipeline and design appropriate data layers for this case. Briefly describe what is the underlying logic of every layer and why you chose it.

Layers

Our architecture has 3 key layers:

1. **Landing Layer**: clean up and standardise the data to prepare it for use in later stages
    - Rename fields
    - Cast data types
    - Flatten the data
2. **Core Layer**: a source of truth dataset for analysts to query
    - At ID level to enable joins to other parts of the database
    - Business logic included here
    - Remove unnecessary or confusing fields
    - If using Looker - data would be fed in from this layer
3. **Reporting Layer:** a set of tables that can be fed into a BI tool such as Tableau
    - Data is aggregated to a sensible level to reduce processing by BI tool
    - Metrics calculated where possible - but ratios left for BI tool

Lineage


source_acceptances
landing_acceptances
source_chargebacks
landing_chargebacks
core_transactions
reporting_transactions

Landing Layer

landing_acceptances		
transaction_id	STRING	renamed from external_ref to improve clarity
status	BOOLEAN	
source	STRING	
ref	STRING	
date_time_utc	TIMESTAMP	add utc to improve clarity
state	STRING	
cvv_provided	BOOLEAN	
transaction_value_local	FLOAT	renamed amount to improve clarity
country	STRING	
currency	STRING	
fx_rate	FLOAT	extracted relevant fx rate for the currency

landing_chargebacks		
transaction_id	STRING	renamed from external_ref to improve clarity
status	BOOLEAN	
source	STRING	
chargeback	BOOLEAN	

- Renamed some fields to improve clarity for end users
- Added the timezone to field name to avoid ambiguity
- Extracted the relevant FX rate for the currency

Core Layer

core_transactions		
transaction_id	STRING	
date_time_utc	TIMESTAMP	
state	STRING	
cvv_provided	BOOLEAN	
country	STRING	
currency	STRING	
fx_rate	FLOAT	
chargeback	BOOLEAN	
gross_value_local	FLOAT	
gross_value_usd	FLOAT	
net_value_local	FLOAT	if chargeback = true then 0 else gross_value
net_value_usd	FLOAT	if chargeback = true then 0 else gross_value

- Joined the acceptances and chargebacks tables together to simplify for end users
- Removed superfluous fields that don't add value so as to reduce clutter and simplify
- Added calculated fields for value in USD
- Added net value fields which reduces complexity for end users by allowing them to do a sum on that field

Reporting Layer

transactions_reporting		
date_utc	TIMESTAMP	rolled up to day
cvv_provided	BOOLEAN	
country	STRING	
currency	STRING	
avg_fx_rate	FLOAT	average fx rate over the transactions on that day
attempted_count	INTEGER	count of attempted transactions
declined_count	INTEGER	count of declined transactions
accepted_count	INTEGER	count of accepted transactions
chargeback_count	INTEGER	count of transactions with a chargeback
net_accepted_count	INTEGER	accepted count - chargeback count
attempted_value_local	FLOAT	value in local currency
declined_value_local	FLOAT	
accepted_value_local	FLOAT	
chargeback_value_local	FLOAT	
net_accepted_value_local	FLOAT	
attempted_value_usd	FLOAT	values in USD
declined_value_usd	FLOAT	
accepted_value_usd	FLOAT	
chargeback_value_usd	FLOAT	
net_accepted_value_usd	FLOAT	

- Rolled up to the most important dimensions - and changed time grain to daily. This enables easy slicing in BI tool.
- Calculated aggregate stats for the most relevant metrics
- Ratios are not calculated (eg acceptance rate) as these will need to be calculated in BI tool to be able to take account of the dimensions
- Values given in both local and USD

Task 4

Propose an A/B test to optimize the feature's performance. Feel free to choose any aspect you find potentially useful to test.

Additionally, please outline the steps you would take to ensure a statistically significant experiment while avoiding common pitfalls.

Hypothesis

From the data we can see that there is currently a chargeback rate of 4% which equates to $1.5m over 6 months.

Chargebacks can be reduced by adding a CVV (and helps to shift the fraud burden to the issuer). However, from the analysis we can see that by adding a CVV we also reduce the acceptance rate.

As such we want to test whether adding a CVV improves the total number of accepted orders without chargebacks given there is $1.5m opportunity.

Our hypothesis is that:

By forcing customers to provide a CVV for every transaction we will improve the total number of accepted orders without a chargeback as a proportion of total attempted orders.

Experiment Design

Control: adding CVV upon checkout will be optional (as is currently the case)

Variant: users will have to add a CVV in order to checkout

KPI: count accepted orders without a chargeback / count attempted orders

Length of test:

For the KPI above we currently have a baseline of 65% - with 905 attempts per month.

If we want to see a **5% relative difference** we would need **3,400 attempts** on each variant meaning it would take **7.5 months** to run the test.

If we are happy with a **10% minimal relative difference** that would be **854 attempts** which would take **2 months.**

Test calculator used: https://www.evanmiller.org/ab-testing/sample-size.html

Appendix

Task 1 Code: Cleaned Dataset

```sql
with base as (

SELECT

*
, split(rates, ",") as all_fx_rates

FROM `nodal-alloy-399422.company_1.acceptances`


), expanded_fx_rates as (


select * except(all_fx_rates), replace(replace(replace(single_fx_rates, '"', ''), '{', ''), '}', '') as cleaned_fx_rate

from base, unnest(all_fx_rates) as single_fx_rates

) , relevant_fx_rates as (

select

*
, cast(right(cleaned_fx_rate, length(cleaned_fx_rate) - 4) as float64) as fx_rate

from expanded_fx_rates

where regexp_contains(single_fx_rates, currency) = TRUE

)

select

* except(rates, single_fx_rates, cleaned_fx_rate)

from relevant_fx_rates
```

This is saved as a view (`deel.acceptances_cleaned_fx`) and used in subsequent queries

Task 1 Code: Declined Orders

```sql
select
sum(amount/fx_rate) as amount_usd

from `company_1.acceptances_cleaned_fx`

where state = 'DECLINED'
```

Task 1 Code: Trends over Time

```sql
select
timestamp_trunc(date_time, day) as _date
, cvv_provided
, country
, currency
, sum(case when state = 'ACCEPTED' then amount/fx_rate else null end) as accepted_amount_usd
, sum(case when state = 'DECLINED' then amount/fx_rate else null end) as declined_amount_usd
, sum(amount/fx_rate) as attempted_amount_usd
, count(case when state = 'ACCEPTED' then external_ref else null end) as accepted_count
, count(case when state = 'DECLINED' then external_ref else null end) as declined_count
, count(external_ref) as attempted_count
, sum(case when state = 'ACCEPTED' then amount/fx_rate else null end)/ sum(amount/fx_rate) as acceptance_rate_amount_usd
, count(case when state = 'ACCEPTED' then external_ref else null end)/ count(external_ref) as acceptance_rate

from `company_1.acceptances_cleaned_fx`

group by 1,2,3,4

order by 1 asc
```

Task 1: Working Document

Exploration of data was done via Looker Studio.

The working document can be found here:
https://lookerstudio.google.com/reporting/926e0495-81e4-4a57-91ca-2290c27334d6

This is a very rough working document for the purposes of exploring the data. It is not intended to be used as a dashboard

Task 4 Code: Data for Experiment Design

```sql
with base as (

SELECT

a.*
, c.chargeback

FROM `nodal-alloy-399422.company_1.acceptances_cleaned_fx` a

left join `nodal-alloy-399422.company_1.chargebacks` c
on a.external_ref = c.external_ref

)

select

count(case
          when chargeback = true then null
          when state = 'DECLINED' then null
          else external_ref
          end) as net_count_accepted
, count(external_ref) as total_count
, count(case
          when chargeback = true then null
          when state = 'DECLINED' then null
          else external_ref
          end )
          / count(external_ref) as net_acceptance_rate

, count( case when chargeback = true then external_ref else null end) as count_chargebacks
, sum( case when chargeback = true then amount/fx_rate else null end) as value_chargebacks

from base

order by 1 asc
```