Company 4 Case Study

Ben Winby

All tables are materialised within the `company_4_datawarehouse_BW` dataset.

Task 1

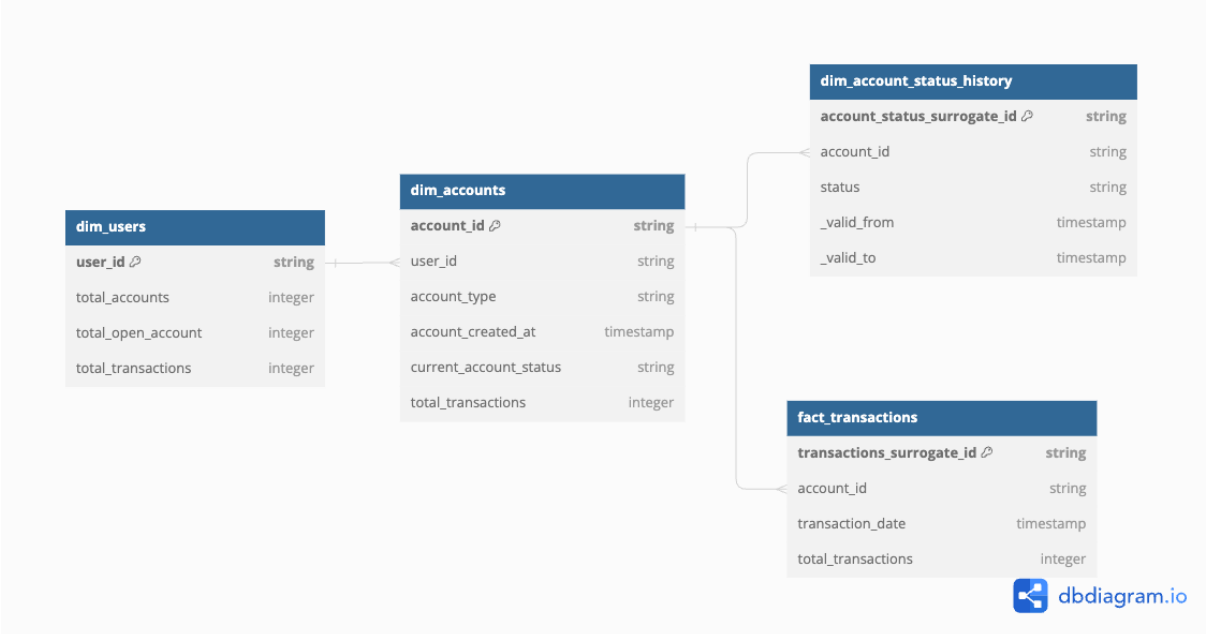
4 tables created:

- dim_accounts
- dim_accounts_status_history
- fact_transactions
- dim_users

Documentation

Note: dbt has not been used for this task but in a real-life implementation I would ensure that documentation exists within the functionality of dbt ie each model would have its own yml file which would include descriptions and tests.

ERD



dim_accounts

This table contains summary info on each account. Each row represents a single account.

Name	Description
account_id	The ID of the account. Primary key.
user_id	The ID of the user
account_type	The type of account
account_created_at	The timestamp of when the account was created
current_account_status	The account status at the current point in time (or the last time the table was refreshed)
total_transactions	Aggregate field: the total transactions by this account at the current point in time (or the last time the table was refreshed)

dim_accounts_status_history

This table contains the full history of the account status - when it was open and when it was closed.
Each row represents a specific period for each account during which the status of the account changes.

Name	Description
account_status_surrogate_id	Unique key for each row made up of a hash of ‘account_id’ and ‘_valid_from’
account_id	The ID of the account
status	The status of the account - either “open” or “closed”
_valid_from	The timestamp from when the status is valid.
_valid_to	The timestamp from when the status is no longer valid. A null timestamp means the status is still valid.

fact_transaction

This table contains a daily count of transactions for each account.

Each row represents for each account a single day with a transaction. Rows will only exist for days and accounts with a transaction.

Name	Description
transactions_surrogate_id	Unique key for each row made up of a hash of ‘account_id’ and ‘transaction_date’

account_id	The ID of the account
transaction_date	The date of the transaction(s)
no_transactions	The total number of transactions made on that date

dim_users

This table contains summary info on each user. Each row represents a single user.

Name	**Description**
user_id	The ID of the user. Primary key.
total_accounts	Aggregate field: the total number of accounts associated with the user
total_open_accounts	Aggregate field: the total number of open accounts associated with the user
total_transactions	Aggregate field: the total number of transactions by the user

Tests

1. Within `account_created`, `account_id_hashed` is unique
2. All `account_id_hashed` in `account_closed` also exist within `account_created`
3. All `account_id_hashed` in `account_reopened` also exist within `account_closed`
4. Within `account_transactions`, the combination of `date + account_id_hashed` is unique
5. `account_closed_ts` is always greater than `account_created_ts`

Assumptions

1. Once an account is created it is classed as open
2. An account must be created before it can be closed
3. An account must be closed before it can be reopened
4. If an account is closed multiple times (without being reopened in between) that is a mistake and there is little valuable information to be gained from this
5. When transactions happen whilst an account is meant to be closed we assume this is ok (happens in 18 cases)

Design Decisions

1. The status history was separated from the summary stats so as to simplify analysis for users. A summary table makes it easy to see the current state of play. If users want to know what happened in the past they can make use of the history table. Both tables could have been combined however it is my view that this significantly increases the complexity for end users.
2. Aggregate field for total transactions added for ease of use into `dim_accounts`
3. The `current_account_status` is brought into `dim_accounts` for ease of use
4. Aggregates created for `dim_users` to make it quicker to look at user-level data
5. We have not assumed that accounts can only be closed and reopened once. Instead, the tables have been built to enable accounts to be reopened and closed multiple times (even if that doesn't happen in this instance).
6. Surrogate IDs have been created to facilitate testing of uniqueness

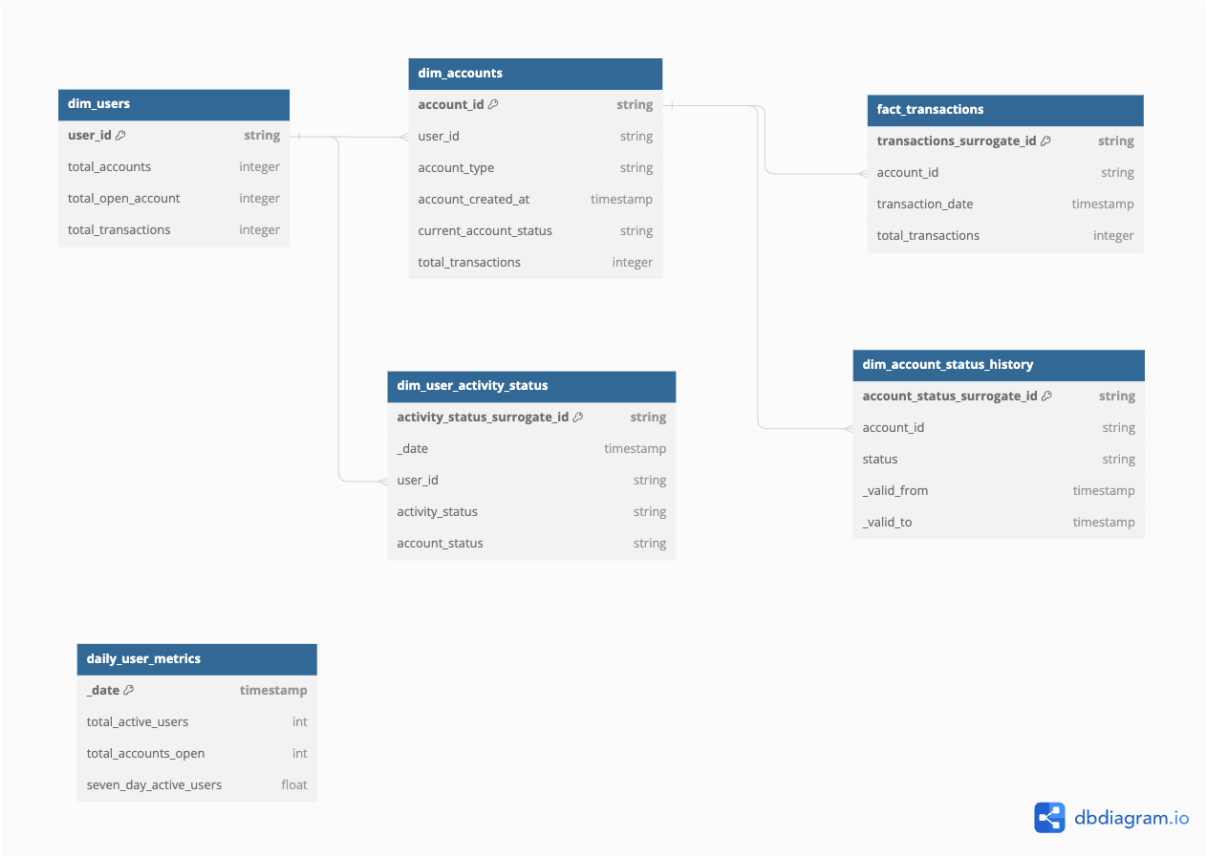
Task 2

2 tables were created to facilitate this:

- **`dim_user_activity_status`** shows for each user on every day over the time period in question whether the user was classed as active and whether their account was open. This can be joined to the tables created in the previous task on user_id to answer cohort-based questions.
- **`daily_user_metrics`** calculate key metrics such as "7-day active users" for every day over the time period in question

Documentation

ERD



dim_user_activity_status

Each row represents a single user on a single day.

We exclude any users who only have accounts that were closed within 1 hour of being created.

Name	**Description**
activity_status_surrogate_id	Unique key for each row made up of a hash of '_date' and 'user_id'
_date	The date in question
user_id	The ID of the user
activity_status	Whether the user had made a transaction in the last 7 days Options: - active - inactive
account_status	Whether the account was open or closed Options: - open

	- closed

daily_user_metrics

Each row represents a single day. All days from 2017-01-01 to now will be represented.

Name	Description
_date	The date of interest
total_active_users	The number of active users - ie users who have had a transaction in the last 7 days
total_accounts_open	The number of users with at least 1 open account
seven_day_active_users	7 day active users: the number of users that had a transaction over the last running 7 days, divided by all the users with at least one open account at that point

Assumptions

1. Last running 7 days → between 0 and 6 days ago as this gives a rolling 7 day window
2. "**Users with only closed accounts should be excluded** from the metric calculation." It has been assumed previously that once an account is created it is classed as open. However, based on this we have also assumed that any account that is created and then closed within 1 hour should be excluded from the data completely.
3. We have assumed that in the situation where someone makes a transaction and then closes the account within 7 days we should still class that account as active even though it might be closed
4. There are situations where transaction are made whilst the account is closed. We have assumed this is ok

Design Decisions

2 tables were created to facilitate this.

- `dim_user_activity_status` gives info at the user_id level which enables users to join to the other tables created in task 1 allowing them to segment by other dimensions as well as being able to more quickly calculate the "7 day active users" metric
- `daily_user_metrics` calculates the metric for every single day enabling a single source of truth for the business. It will also enable it to be easily visualised.

The numerator and denominator of the "7 day active users" metric were included in

`daily_user_metrics` as they will be very useful during the analysis phase to understand volumes contributing to the ratio. They are likely to be particularly useful when looking at statistical significance.