Week 5 - Implementation: Mart layer

The objective of the fifth week is to create the data mart layer for the project. The data mart layer will follow the previous logic as there will be a separate GitHub and Stack Overflow part.

First, the raw data was ingested into the staging layer, then the needed transformation and filtering were applied in the intermediate layer using only the listed organizations. Now, the objective of the data mart layer is to create a granularity that matches the business goals and measures that will act as the base of the dashboards to compare these companies and analyze the different tech sectors.

General

The time granularity for the data mart layer will be daily, monthly, and quarterly. As for now, it would be over complicated and not match the logical concept to merge these three different time granularities into one model, it should be divided into three separate models at the end.

The Company Details won't change at this time and even if it is changing it should be handled at the intermediate layer. Because of this, there will be only GitHub and Stack Overflow models for the data mart layer.

As for the development principles and dbt best practices, the same rules and tips stand.

Prerequisites

Last week, we created the SCD Type-2 version of the Company Details table. This may have created multiple ingestions and loads, possibly resulting inconsistent data across the students.

For this week, it would be necessary for everyone to start from the same state regarding the source data. This would require loading the v3 version of the Company Details sheet once again and take it up to the intermediate layer. Then run the intermediate GitHub and Stack Overflow models again with a full-refresh dbt run make sure using the v3 version of the Company Details sheet.

GitHub

Objective

The time granularity will be derived from the *created_at* datetime field. It should be daily, monthly, and quarterly. As for repositories, the level of aggregation is the Company Details sheet's organization name. There should be three GitHub models in total with the following columns.

List of models

- mart_github_daily
- mart_github_monthly
- mart_github_quarterly

List of columns

Column name	Description
_pk	One of the columns or concatenation of columns in a hashed format, which function as the primary key of the model
first_day_of_period	First day of the actual period. For example, in the quarterly model, it can be 2022-01-01 or 2022-07-01
month	Month of the period. For example, 01 or 07
quarter	Quarter of the period. Built-in function can be used here
year	Year of the period. It is 2022 only.
organization_name	Same in source
repository_account	Name of the repository account.
repository_name	Name of the repository. It should be the repository account if the repository name is empty in Company Details
event_count	Count of the users derived from event_id
user_count	Count of the users derived from user_id
issues_count	Count of the issues events derived from type field
watch_count	Count of the watch events derived from type field
fork_count	Count of the fork events derived from type field
push_count	Count of the push events derived from type field
pr_count	Count of the PR events derived from type field
delete_count	Count of the delete events derived from type field
public_count	Count of the public events derived from type field
create_count	Count of the create events derived from type field
gollum_count	Count of the gollum events derived from type field
member_count	Count of the member events derived from type field
commit_comment_count	Count of the commit comment events derived from type field
total_event_count	Count of all the events. Sum of all the above events

Hint

Most of the columns here are based on the same logic. It's summing up the rows if it matches a specified value in the type column. It can be done with case when for each calculated column, but dbt's jinja function can be pulled on for this and it is one of the best use cases to make the code more DRY and it is also easier to manage and change them, if necessary.

dbt jinja cheatsheet created Zsombor

Stack Overflow

Objective

For Stack Overflow the same granularity and logic is being used as in the GitHub source. There should be the same three models regarding the time granularity derived from the creation datetime field. The other aggregation should be the organizations from Company Details. As the tags are the same for questions and answers and at this phase, only the tags are used, the use of the answers model is not needed here, only the intermediate questions model. Later, the answers can be added as well.

List of models

- mart_stackoverflow_questions_daily
- mart_stackoverflow_questions_monthly
- mart_stackoverflow_questions_quarterly

List of columns

Column name	Description
_pk	One of the columns or concatenation of columns in a hashed format, which function as the primary key of the model
first_day_of_period	First day of the actual period. For example, in the quarterly model, it can be 2022-01-01 or 2022-07-01
month	Month of the period. For example, 01 or 07
quarter	Quarter of the period. Built-in function can be used here
year	Year of the period. It is 2022 only
organization_name	Same in source
post_count	Count of the questions based on the ID
answer_count	Count of the answers in total
avg_answer_count	Average number of answers for a question
comment_count	Count of the comments in total
avg_comment_count	Average number of comments for a question
favorite_count	Count of the favorites in total
avg_favorite_count	Average favorites for a question
view_count	Count of the views in total
avg_view_count	Average view of the questions
accepeted_answer_count	Number of accepted answers for a question. Corrected with the question counts. An average is needed here, as the field should be comparable
no_answer_count	Number of questions without answers
avg_no_answer_count	Number of questions without answers divided by the number of questions

score	Normalized value of the scores. It should be explored through the dataset if it is needed to divide by the number of posts or if other normalization methods should be applied. It can be summarized or corrected by the number of posts, or only the highest ranked question is the valid measure for the score field
tags_count	Total number of tags besides the ones listed in the Company Details. For example, for the Snowflake company, if the tags in the Company Details are snowflake, snowflake-cloud-data-platform and the total available tags for Snowflake are snowflake, snowflake-cloud-data-platform, database, cloud-database, then this count should be 2
last_activity_datetime_utc	Maximum of last activity date
last_edit_datetime_utc	Maximum of last edit date

Additional task

Adding date spine

The additional task is to add date spine to our mart models. Maybe you heard different expressions for that case, but we use date spine in dbt. Generally, the problem is that now we have unfilled rows for dates without any record. The task is to fill them with 0 value records for every organization. It will be useful in presenting our models in the visualization layer, because there will be no gaps by dates or organizations. It is also easier to filter and understand visualizations with date spine added. There is a macro in the dbt-utils package that will be handy in starting the task. It will create the calendar table for you, which you can use for implementing the date spine solution. You can also google or check solutions in the dbt community slack. You can use the already created year variable instead of a hard-coded date when adding the datespine macro.

As-is				
Date	**Organization**	**Metric 1**	**Metric 2**	**Metric 3**
2022.01.01	dbt	1	4	0
2022.01.01	Snowflake	1	0	2
2022.01.02	Snowflake	2	2	5
2022.01.04	dbt	2	6	4
2022.01.05	dbt	4	8	3
2022.01.05	Snowflake	1	0	2

TO-BE				
Date	**Organization**	**Metric 1**	**Metric 2**	**Metric 3**
2022.01.01	dbt	1	4	0
2022.01.01	Snowflake	1	0	2
2022.01.02	dbt	0	0	0
2022.01.02	Snowflake	2	2	5
2022.01.03	dbt	0	0	0
2022.01.03	Snowflake	0	0	0
2022.01.04	dbt	2	6	4
2022.01.04	Snowflake	0	0	0
2022.01.05	dbt	4	8	3
2022.01.05	Snowflake	1	0	2