LOVEVERY

Thank you for your continued interest in the Sr. Analytics Engineer role at Lovevery. The team has been impressed by your skill set thus far and looks forward to this stage in our interview process.

This assignment is broken into two distinct parts. The first will be technical in nature, while the second part will be more design/architecture focused. If you have any questions about the assignment, please feel free to email (robert.battista@lovevery.com), and I will be happy to assist.

This assignment should take no longer than 2-3 hours to complete. If you find yourself running out of time, please make note of any future steps/ideas you would like to incorporate. Upon receiving this email, please send back your completed work within 1 week. In a follow-up interview, we will talk through the assignment, so feel free to create any presentation aides that will help you explain.

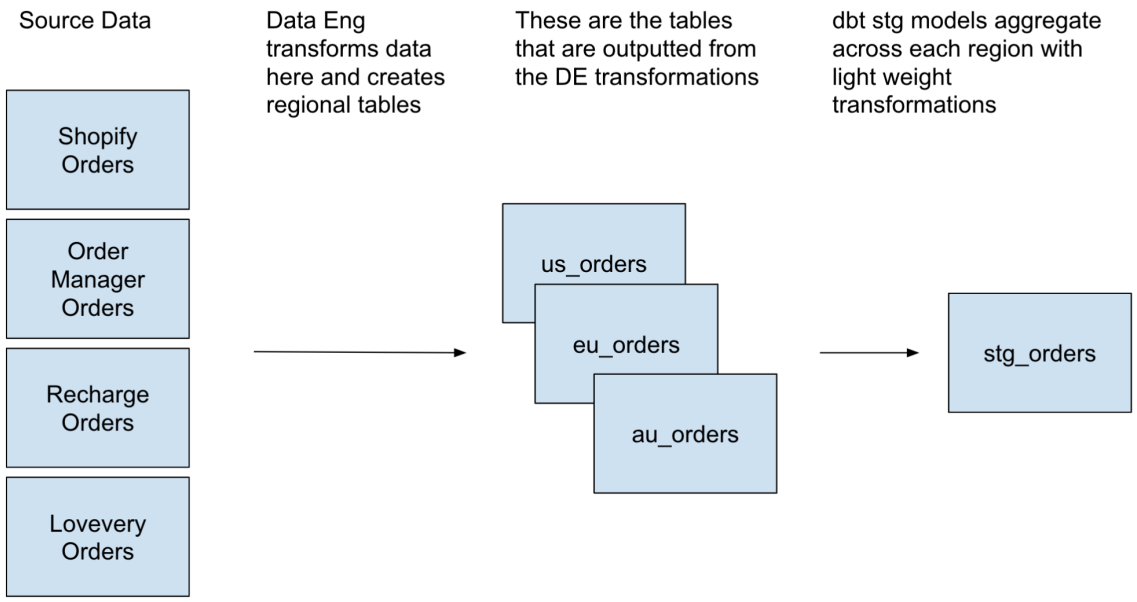
Task 1:
Given the below web pageview dataset, write a SQL query to add a session_id column to the pageviews table that assigns a unique identifier to each session. A session is defined as any activity that is within 30 minutes of the user's last pageview. This query should include all of the existing data in the pageviews table and simply add session_id as a new column. All events from a session must contain the session_id value.

Follow up questions:
How would you add incrementality to this model? What challenges might you encounter and how would you address them?

event_id	user_id	event_time	page_url
1	99999	2025-01-01 9:00:00	lovevery.com
2	99999	2025-01-01 9:03:00	https://lovevery.com/products/the-play-kits
3	99999	2025-01-01 9:02:00	https://lovevery.com/products/the-play-kits#explore
4	99999	2025-01-01 9:15:00	https://lovevery.com/products/the-play-kits-the-enthusiast
5	99999	2025-01-01 9:48:00	https://lovevery.com/pages/subscription-v3/child-info
6	99999	2025-01-01 11:00:00	https://lovevery.com/products/the-play-gym

The output of this task should be a SQL statement that adds the session_id and text explaining how you would go about adding incrementality.

Task 2:
Our current architecture has the Data Engineering team managing transformations within DataBricks notebooks. These notebooks have many functions. They are responsible for aggregating data from multiple systems. For example, customer orders may come from systems like Shopify, Recharge, Order Manager, and Lovevery's own platform. This is the same architecture for things like customers, products, fulfillments, etc. These notebooks also normalize the data into specific datatypes and schemas. The notebooks provide some level of transformation where records are removed, updated, or ignored altogether. These models are run incrementally to reduce runtimes where possible. This same process runs in the 3 regions that Lovevery operates, namely the US, EU, and AU regions. Therefore, the notebooks produce tables like us_orders, eu_orders, and au_orders. These tables are the 'sources' that our dbt project uses in the staging layer. Below is a diagram of the current process.



A focus of the AE team is to transition from Databricks notebooks to dbt (ie the first arrow in the diagram above). How would you go about a project like this? How would you set up your dbt project, given the constraints above? What would the benefits be of running this all in dbt? What are the cons of running this in dbt? dbt Labs suggests, as a best practice, structuring a project's transformations into three different layers or folders: staging, intermediate, and marts. (At Lovevery, we denote these layers with the prefixes stg_, int_, and rpt_ for staging, intermediate, and reporting, respectively). Do you believe this structure is still the best way to architect our project in this scenario? Would you go about it differently? What are some ways you would ensure this transformation logic migration produces data identical to the existing logic?

The output of this task could be a POC (proof of concept doc), diagrams, slides, etc. We are looking for you to explain your methodology.