DW Team - Unattended Coding Exercise

Background

The HD Motor Insurance Platform generates data as users interact with the website. The data is then stored in a Data Lake in a semi-structured form. Some of the data being captured from our website is:

- User Sessions: Session information like duration (entry_ts, exit_ts), how the user has landed on our site (UTMs), pages the user has visited on our side (paths).

- Quote Events: As the user works through the 'Get a Quote' wizard the back-end is updating a Quote resource and publishes update events.

- Policies: When a user buys a quote it is converted to a policy.

The Datawarehouse has a set of ETLs that extracts data from the Data Lake, transforms and loads them into a relational database. For this exercise we will mock the Data Lake with three CSVs representing sessions, quote events and policies.

Scenario

The business requires four new tables in the Datawarehouse:

- UserSessions

- Policies

- Quotes: Represents the latest snapshot of a quote which is created by collapsing all quote events for a given quote reference. For example, if we have 20 quote events for a given quote reference, we collapse them into one row in the Quotes table which contains all the latest information.

- Master: A combination of the above information for each completed session. If a session has entry and exit timestamps (entry_ts, exit_ts) we consider it to be "completed". For example, if a user took 3 quotes during his session, we should see 3 rows in master with the following information:

    - Session information (session id, paths).
    - The snapshot of the quote **at the end of the session.**

    - Policy number if the quote was converted to a policy during that session.

As we are dealing with Data Lakes some of the data might fail validation e.g. wrong data types. Rows with bad data **must not** break the ETL execution, you should instead stage them somewhere separately to be dealt with offline.

Deliverables

As part of your answer you are required to submit the following:

- The ETL implementation in C#.
- An export of the database (any relational db of your liking) that contains the tables described above and any other auxiliary tables that you created.
- A txt file with short notes of your thinking / rationale as you go about implementing the solution.

You have full freedom to design and implement this as you see fit.

Field Appendix

Sessions:

- _id: Session identifier.
- uuid: User identifier from the browser cookie.
- paths: Pages the user has visited on our site. If the user has visited the quote wizard there will be an attribute for the quote reference (in Master we are only interested in quotes where product==motor).
- entry_ts: Timestamp of the first interaction the user had with our side.
- exit_ts: Timestamp of the last interaction the user had with our side.
- c_ts: Record creation timestamp.
- p_ts: Last record update timestamp.

Quote Events:

- Id: Event ID.
- QuoteReference: Quote ID.
- StartDate: The date the coverage starts.
- EndDate: The date the coverage ends.
- SumAssured: The insured amount for the vehicle.
- TotalPremium: Price of the quote.

Policies:

- Id: Policy ID.
- PremiumBreakdown: Breakdown of the policy price.
- QuoteReference: Quote ID of the quote from which the policy was derived.