Collection of Votes Data for Arbitrum DAO Proposals on Snapshot

Difficulties Faced During Data Collection:

1. **Total Proposals:**
   - There are currently 144 total proposals available on Arbitrum DAO on Snapshot.
2. **Proposal ID Collection:**
   - Collected all proposal IDs and stored them in a CSV file.
3. **GraphQL Query Iteration:**
   - Utilized an iterative method to dynamically provide proposal IDs in GraphQL queries for fetching votes data.
4. **API Request Limitation:**
   - Faced a limitation in API requests:
     - Only 1000 records could be retrieved in one request using the 'first' parameter.
     - Using the 'skip' parameter, only 6000 records were possible to fetch in a single request.
5. **Pagination Implementation:**
   - Implemented pagination to handle the limitation in the number of records per request.
6. **Data Fetching in Chunks:**
   - Due to the large dataset (144 proposals), fetched data in two chunks to avoid errors and ensure successful retrieval.
7. **Time Consumption:**
   - Acknowledged that collecting data for all 144 proposals may take some time due to the above challenges.
8. **Data Storage:**
   - Data is stored in a CSV file after collection.

The fields included in the collected data:

1. **Id:**
   - Contains the voter ID.
2. **Voter:**
   - Contains the voter's address.
3. **Vp:**
   - Contains the voting power of the voter.
4. **Vp_by_strategy:**
   - Contains the voting power by strategy.
5. **Created:**
   - Represents the date on which the voter has voted in Unix timestamp format.
6. **Proposal:**
   - A dictionary that contains the following keys:
     - **Id:**
       - Represents the proposal ID.
     - **Title:**
       - Contains the title of the proposal.
     - **Body:**
       - Contains the text included in the proposal.
     - **Choices:**
       - An array of all the choices given in the particular proposal for which voters can give their votes.
     - **Type:**
       - Represents the type of the proposal based on voting types:
         - Basic
         - Single-choice
         - Ranked-choice
         - Approval
         - Weighted
7. **Choice:**
   - Contains either a single number or an array of numbers representing the choice(s) for which the voter has voted. The structure of the data may vary based on the proposal types.
8. **Reason:**
   - Represents the reason for voting, if available. This field may contain additional information or explanations provided by the voter regarding their voting decision.

The fields created and the modifications done after the collection of the data:

1. **Splitting Proposal Dictionary:**
   - The 'proposal' column, which initially contained a dictionary, was split into 5 different columns based on the keys in the dictionary. These new columns may include 'proposal_id', 'proposal_title', 'proposal_body', 'proposal_choices', and 'proposal_type'.
2. **Conversion of 'Created' Column:**
   - The 'created' column, initially in Unix timestamp format, was converted into a datetime format for easier handling and interpretation.
3. **New Date-related Columns:**
   - Four new columns were created from the 'created' column:
     - **Year:**
       - Represents the year of the vote creation.
     - **Month:**
       - Represents the month of the vote creation.
     - **Day:**
       - Represents the day of the vote creation.
     - **Time:**
       - Represents the time of the vote creation.
4. **Splitting by Proposal Type:**
   - The dataset was divided into five separate datasets, each focusing on a specific proposal type. For example created datasets for proposals of the following types:
     - Basic
     - Single-choice
     - Ranked-choice
     - Approval
     - Weighted
5. **Mapping Choices in 'single-choice' and 'basic' Proposal Types:**
   - For datasets related to 'single-choice' and 'basic' proposal types, the 'choice' column, which initially contained a numerical choice number, was mapped to the corresponding choices in the 'proposal_choices' column, where choices were represented in array form.
6. **Creation of 'Mapped_choice' Column:**
   - A new column named 'mapped_choice' was created to store the choices corresponding to the numerical values found in the 'choice' column.
7. **Mapping Ranks in 'Ranked' Proposal Type:**
   - For the dataset related to the 'ranked' proposal type, the 'choice' column contained an array of ranks, and the 'proposal_choices' column contained an array of choices.
8. **Creation of 'Voting_choice_names' Column:**
   - A new column named 'voting_choice_names' was created to store arrays of elements containing the names of choices corresponding to the numbers given in the 'choice' column.
9. **Processing 'Approval' Dataset:**
   - For the 'approval' dataset, where the 'choice' column contained arrays of choices, the data was separated into individual rows, each containing a single number from the array.

10. **Creation of 'Approved_choice' Column:**
    - A new column named 'approved_choice' was created to store the single numbers extracted from the arrays in the 'choice' column.
11. **Mapping Operation for 'Mapped_choice' Column:**
    - Another new column named 'mapped_choice' was created to store the names of choices corresponding to the numbers in the 'approved_choice' column. This mapping was based on the original 'proposal_choices' array.
12. **Original Columns in Weighted Dataset:**
    - proposal_choices: Contains an array of proposal choices.
    - choice: Contains a dictionary with ranks as keys and weights as values.
13. **Created columns from the 'proposal_choices' and 'choice' column:**
    - **weighted_choice:** Contains the single number representing the choice with the highest weight.
    - **mapped_choice:** Contains the name of the choice corresponding to the number in the weighted_choice.
    - **weight:** Contains the weight associated with the weighted_choice.

The collected and processed dataset provides a rich set of information related to the votes on Arbitrum DAO proposals. The various fields and modifications enable several types of analyses. Here's how the data can be helpful in analysis and the potential analyses that can be conducted:

1. **Proposal Type Analysis:**
    - The dataset has been organized based on proposal types (Basic, Single-choice, Ranked-choice, Approval, Weighted). This segmentation allows for a focused analysis of voting patterns and behaviors specific to each proposal type.
2. **Temporal Analysis:**
    - The creation of new date-related columns (Year, Month, Day, Time) facilitates temporal analysis. You can explore trends, patterns, and voting behavior over time, identify peak voting periods, and assess the impact of temporal factors on voting.
3. **Voter Voting Patterns:**
    - Analyzing the 'choice' column provides insights into individual voter preferences and voting patterns. Understanding how voters make choices across different proposals can inform strategies to engage and communicate with the community.
4. **Reasons for Voting Analysis:**
    - The 'reason' column allows for a qualitative analysis of the reasons behind voter choices. Exploring common themes or sentiments in voter reasoning can provide valuable insights into community sentiments and concerns.
5. **Weighted Voting Analysis:**
    - The 'weighted' dataset introduces the concept of weighted choices. Analyzing the 'weight' associated with each choice can reveal influential voters and highlight choices that carry significant community support.
6. **Proposal-specific Insights:**
    - The split datasets based on proposal types enable in-depth analysis of each proposal category. You can compare voting patterns, participation levels, and reasons for voting across different proposal types.

7. **Mapping Choices and Ranks:**
    - The creation of 'mapped_choice' and 'voting_choice_names' columns enhances the interpretability of voting choices. This mapping facilitates easier visualization and understanding of the voting preferences expressed in numerical form.
8. **Approval Dataset Analysis:**
    - The separation of data in the 'approval' dataset, along with the creation of 'approved_choice' and 'mapped_choice' columns, allows for a detailed analysis of approval-based voting. You can explore approval levels for each choice and understand the reasons behind approvals.
9. **Weighted Dataset Analysis:**
    - The 'weighted' dataset introduces a unique dimension with weighted choices. Analyzing the weight distribution can identify key influencers within the community and provide insights into the community's prioritization of choices.
10. **Community Engagement Strategies:**
    - Understanding voting patterns and preferences can inform the development of targeted community engagement strategies. Tailoring communication and outreach efforts based on the analyzed data can enhance community involvement in decision-making.

In summary, the collected data offers a robust foundation for conducting a wide range of analyses, including community sentiment analysis, voting behavior patterns, and the impact of different proposal types. The insights derived from these analyses can inform strategic decision-making within the Arbitrum DAO.