Google Discussions

Exam Professional Data Engineer All Questions

View all questions & answers for the Professional Data Engineer exam

Go to Exam

EXAM PROFESSIONAL DATA ENGINEER TOPIC 1 QUESTION 234 DISCUSSION

Actual exam question from Google's Professional Data Engineer

Question #: 234

Topic #: 1

[All Professional Data Engineer Questions]

You migrated a data backend for an application that serves 10 PB of historical product data for analytics. Only the last known state for a product, which is about 10 GB of data, needs to be served through an API to the other applications. You need to choose a cost-effective persistent storage solution that can accommodate the analytics requirements and the API performance of up to 1000 queries per second (QPS) with less than 1 second latency. What should you do?

A. 1. Store the historical data in BigQuery for analytics.
2. Use a materialized view to precompute the last state of a product.
3. Serve the last state data directly from BigQuery to the API.

B. 1. Store the products as a collection in Firestore with each product having a set of historical changes.
2. Use simple and compound queries for analytics.
3. Serve the last state data directly from Firestore to the API.

C. 1. Store the historical data in Cloud SQL for analytics.
2. In a separate table, store the last state of the product after every product change.
3. Serve the last state data directly from Cloud SQL to the API.

D. 1. Store the historical data in BigQuery for analytics.
2. In a Cloud SQL table, store the last state of the product after every product change.
3. Serve the last state data directly from Cloud SQL to the API.

Show Suggested Answer

by scaenruy at *Jan. 3, 2024, 12:56 p.m.*

Comments

Type your comment...

Submit

einchkrein Highly Voted 1 year, 3 months ago

Serve the last state data directly from Cloud SQL to the API.
Here's why this option is most suitable:

BigQuery for Analytics: BigQuery is an excellent choice for storing and analyzing large datasets like your 10 PB of historical product data. It is designed for handling big data analytics efficiently and cost-effectively.

Cloud SQL for Last State Data: Cloud SQL is a fully managed relational database that can effectively handle the storage of the last known state of products. Storing this subset of data (about 10 GB) in Cloud SQL allows for optimized and faster query performance for your API needs. Cloud SQL can comfortably handle the requirement of up to 1000 QPS with sub-second latency.

Separation of Concerns: This approach separates the analytics workload (BigQuery) from the operational query workload (Cloud SQL). This separation ensures that analytics queries do not interfere with the operational performance of the API and vice versa.

upvoted 7 times

datapassionate Highly Voted 1 year, 3 months ago

Selected Answer: D

D. 1. Store the historical data in BigQuery for analytics.
2. In a Cloud SQL table, store the last state of the product after every product change.
3. Serve the last state data directly from Cloud SQL to the AP

This approach leverages BigQuery's scalability and efficiency for handling large datasets for analytics. BigQuery is well-suited for managing the 10 PB of historical product data. Meanwhile, Cloud SQL provides the necessary performance to handle the API queries with the required low latency. By storing the latest state of each product in Cloud SQL, you can efficiently handle the high QPS with sub-second latency, which is crucial for the API's performance. This combination of BigQuery and Cloud SQL offers a balanced solution for both the large-scale analytics and the high-performance API needs.

upvoted 7 times

zanhsieh Most Recent 2 months, 3 weeks ago

Selected Answer: D

Why not A? Because BQ API 100 reqs per second for API method. Other possible limits do not meet the 1000 QPS requirement. Yes, max number of tabledata.list reqs per second is 1000 but we won't always call tabledata.list all time.
https://cloud.google.com/bigquery/quotas#api_request_quotas

upvoted 1 times

clouditis 4 months, 3 weeks ago

Selected Answer: A

A is the most plausible option - Cloud SQL can not retrieve results out with 1 second latency as the requirement here is, with BQ MV"s that could be a possibility as its pre-computed.

upvoted 2 times

ToiToi 6 months ago

Selected Answer: A

Why A? Because:

Materialized View for API: A materialized view in BigQuery pre-computes the last known state of each product. This ensures that your API can quickly retrieve the latest product information without needing to query the entire historical dataset.
BigQuery for API Serving: BigQuery can handle high query volumes with low latency, meeting your requirement of 1000 QPS with sub-second latency.
Cost-Effectiveness: This solution avoids the need for a separate database like Cloud SQL, minimizing costs and management overhead.

Why not D:
While Cloud SQL is a good option for transactional workloads, it's not as cost-effective or scalable as BigQuery for analytical queries on 10 PB of data. It might also not be the ideal choice for serving high-volume API requests with low latency.

upvoted 1 times

Anudeep58 11 months ago

Selected Answer: D

Why not A:
Serving data directly from BigQuery to the API may not meet the low latency requirements for high QPS operations, as BigQuery is optimized for analytical queries rather than transactional workloads.

upvoted 1 times

Ryannn23 3 months, 1 week ago

what transactional workload? you just need to provide latest status for each product through an API. Select from a 10GB BQ MV will provide the result in 1 sec.

upvoted 1 times

josech 11 months, 3 weeks ago

Selected Answer: A

Materialized views are precomputed views that periodically cache the results of a query for increased performance and efficiency. Materialized views can optimize queries with high computation cost and small dataset results.
https://cloud.google.com/bigquery/docs/materialized-views-intro#use_cases
https://cloud.google.com/bigquery/docs/materialized-views-intro

upvoted 1 times

CGS22 1 year ago

Selected Answer: D

Why D is the best choice:

Cost-Effective Analytics: BigQuery excels at handling large datasets (10 PB) and complex analytical queries. Its columnar storage and massively parallel processing make it ideal for analyzing historical product data.
High-Performance API: Cloud SQL provides a managed relational database service optimized for transactional workloads. It can easily handle the 1000 QPS requirement with low latency, ensuring fast API responses.
Separation of Concerns: Storing historical data in BigQuery and the last known state in Cloud SQL separates analytical and transactional workloads, optimizing performance and cost for each use case.

upvoted 1 times

JyoGCP 1 year, 2 months ago

Selected Answer: D

Option D

upvoted 1 times

ML6 1 year, 2 months ago

Selected Answer: D

BigQuery = data warehouse that is optimized for querying and analyzing large datasets using SQL. Can easily process petabytes of data.
Cloud SQL = designed for transactional workloads and traditional relational database use cases, such as web applications, e-commerce platforms, and content management systems.

upvoted 1 times

Matt_108 1 year, 3 months ago

Selected Answer: D

Option D is the right one, compared to option A, Cloud SQL is more efficient and cost effective for the amount of time the data needs to be accessed by the api

upvoted 3 times

scaenruy 1 year, 4 months ago

Selected Answer: A

A. 1. Store the historical data in BigQuery for analytics.
2. Use a materialized view to precompute the last state of a product.
3. Serve the last state data directly from BigQuery to the API.

upvoted 2 times

RenePetersen 1 year, 2 months ago

I believe the latency of BigQuery is too high to accommodate the sub-second latency requirement.

upvoted 2 times

Platform

- Home
- Examtopics PRO
- All Exams
- Training Courses