Google Discussions

EXAM PROFESSIONAL DATA ENGINEER TOPIC 1 QUESTION 29 DISCUSSION

Actual exam question from Google's Professional Data Engineer

Question #: 29

Topic #: 1

[All Professional Data Engineer Questions]

Your company is streaming real-time sensor data from their factory floor into Bigtable and they have noticed extremely poor performance. How should the row key be redesigned to improve Bigtable performance on queries that populate real-time dashboards?

A. Use a row key of the form <timestamp>.

B. Use a row key of the form <sensorid>.

C. Use a row key of the form <timestamp>#<sensorid>.

D. Use a row key of the form >#<sensorid>#<timestamp>.

Show Suggested Answer

by [deleted] at *March 20, 2020, 8:08 a.m.*

Comments

Type your comment...

Submit

[Removed] Highly Voted 4 years, 7 months ago

Description: Best practices of bigtable states that rowkey should not be only timestamp or have timestamp at starting. It's better to have sensorid and timestamp as rowkey better to have sensorid and timestamp as rowkey

upvoted 33 times

[Removed] Highly Voted 4 years, 7 months ago

Answer D

upvoted 19 times

vosang5299 Most Recent 2 weeks, 1 day ago

Selected Answer: D

D is correct

upvoted 1 times

axantroff 11 months, 2 weeks ago

Selected Answer: D

Looks like D is the best option
Reference: https://cloud.google.com/bigtable/docs/schema-design#time-based

upvoted 2 times

mark1223jkh 5 months, 2 weeks ago

Thank you that is right.

upvoted 1 times

rtcpost 1 year ago

Selected Answer: D

D. Use a row key of the form <sensorid>#<timestamp>.

By using the sensor ID as the prefix in the row key, you can achieve better distribution of data across Bigtable tablets. This can help balance the workload and prevent hotspots in the table. Additionally, placing the timestamp after the sensor ID allows you to perform range scans for a specific sensor and retrieve data efficiently within a time frame.

Option C (using a row key of the form <timestamp>#<sensorid>) can work for some use cases but may not be as efficient for range scans when you want to retrieve data for a specific sensor within a time range.

Option A (using a row key of the form <timestamp>) may lead to hotspots and inefficient range scans because it doesn't consider sensor IDs.

Option B (using a row key of the form <sensorid>) is not optimal because it doesn't allow for efficient time-based filtering and could lead to uneven data distribution in Bigtable.

upvoted 2 times

AzureDP900 1 year, 10 months ago

D is right
Best practices of bigtable states that rowkey should not be only timestamp or have timestamp at starting. It's better to have sensorid and timestamp as rowkey.

Reference:
https://cloud.google.com/bigtable/docs/schema-design

upvoted 1 times

Nirca 1 year, 10 months ago

Selected Answer: D

#<sensorid>#<timestamp> ------> low cardinality # high cardinality
This is current Bigtable Best Practice (to avoid Hotspots on the inserts)

upvoted 5 times

maxdataengineer 2 years ago

Selected Answer: D

Discard:
A -> timestamp unique id could not be unique in the case that sensors transmit data at the same time.
B -> sensorId repeated id for messages coming from the same sensor
C -> a bad performance choice

D -> BEST CHOICE. Each time BigTable looks for data in a table it does a scan and sort operations. By starting each unique id by sensorId it will make it easier to group and sort data since it has the lowest cardinality
https://cloud.google.com/bigtable/docs/schema-design#general-concepts

upvoted 1 times

John_Pongthorn 2 years, 1 month ago

as I look at https://cloud.google.com/bigtable/docs/schema-design#row-keys
asia#india#bangalore
asia#india#mumbai

asia#india#mumbai
they didn't have # ahead of this first value.
asia#india#bangalore OR #asia#india#bangalore
Are both valid?

upvoted 2 times

crisimenjivar 2 years, 2 months ago

ANSWER: D

upvoted 1 times

som_420 2 years, 4 months ago

Selected Answer: D

Answer is D

upvoted 1 times

samdhimal 2 years, 9 months ago

A. Use a row key of the form <timestamp>.
---> Incorrect, because google says don't use a timestamp by itself or at the beginning of a row key.
B. Use a row key of the form <sensorid>.
--->Incorrect, because google says Include a timestamp as part of your row key.
C. Use a row key of the form <timestamp>#<sensorid>.
---> Incorrect, because google says don't use a timestamp by itself or at the beginning of a row key.
D. Use a row key of the form >#<sensorid>#<timestamp>.
---> Correct answer, because of option A,B,C reasons.
- Timestamp isn't by itself, neither at the beginning.
- Timestamp is included.

Reference: https://cloud.google.com/bigtable/docs/schema-design#row-keys

upvoted 9 times

anji007 3 years ago

Ans: D

upvoted 2 times

sumanshu 3 years, 4 months ago

Vote for 'D' - Store multiple delimited values in each row key. (But avoid starting with Timestamp)

"Row keys to avoid"
https://cloud.google.com/bigtable/docs/schema-design

upvoted 9 times

sumanshu 3 years, 3 months ago

A is not correct because this will cause most writes to be pushed to a single node (known as hotspotting)
B is not correct because this will not allow for multiple readings from the same sensor as new readings will overwrite old ones.
C is not correct because this will cause most writes to be pushed to a single node (known as hotspotting)
D is correct because it will allow for retrieval of data based on both sensor id and timestamp but without causing hotspotting.

upvoted 7 times

naga 3 years, 9 months ago

Correct D

upvoted 2 times

NamitSehgal 3 years, 10 months ago

Should be D
Reverse of timestamp even better but no options for that.
Also changing sensor ID if they are in sequential to hash or changing data to bits even better.
Idea is not to use timestamp or sequential ID as first key.

upvoted 3 times

Tanzu 2 years, 9 months ago

reverse TS or hashing is not always first choice or better. never.

upvoted 1 times

Radhika7983 3 years, 11 months ago

The correct answer is D.
Refer to the link https://cloud.google.com/bigtable/docs/schema-design for Big table schema design.

C is not the right answer becuase
Timestamps
If you often need to retrieve data based on the time when it was recorded, it's a good idea to include a timestamp as part of

your row key. Using the timestamp by itself as the row key is not recommended, as most writes would be pushed onto a single node. For the same reason, avoid placing a timestamp at the start of the row key.

For example, your application might need to record performance-related data, such as CPU and memory usage, once per second for a large number of machines. Your row key for this data could combine an identifier for the machine with a timestamp for the data (for example, machine_4223421#1425330757685).

upvoted 3 times

Load full discussion...

EXAMTOPICS

Platform

- Home
- Examtopics PRO
- All Exams
- Training Courses