

[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 161 DISCUSSION

Actual exam question from Google's Professional Data Engineer

Question #: 161

Topic #: 1

[\[All Professional Data Engineer Questions\]](#)

You need to choose a database to store time series CPU and memory usage for millions of computers. You need to store this data in one-second interval samples. Analysts will be performing real-time, ad hoc analytics against the database. You want to avoid being charged for every query executed and ensure that the schema design will allow for future growth of the dataset. Which database and data model should you choose?

- A. Create a table in BigQuery, and append the new samples for CPU and memory to the table
- B. Create a wide table in BigQuery, create a column for the sample value at each second, and update the row with the interval for each second
- C. Create a narrow table in Bigtable with a row key that combines the Computer Engine computer identifier with the sample time at each second
- D. Create a wide table in Bigtable with a row key that combines the computer identifier with the sample time at each minute, and combine the values for each second as column data.

[Show Suggested Answer](#)

by [madhu1171](#) at March 15, 2020, 7:41 p.m.

Comments

Type your comment...

[Submit](#)

  **psu** Highly Voted  5 years ago



Answer C

A tall and narrow table has a small number of events per row, which could be just one event, whereas a short and wide table has a large number of events per row. As explained in a moment, tall and narrow tables are best suited for time-series data.

For time series, you should generally use tall and narrow tables. This is for two reasons: Storing one event per row makes it easier to run queries against your data. Storing many events per row makes it more likely that the total row size will exceed the recommended maximum (see Rows can be big but are not infinite).


https://cloud.google.com/bigtable/docs/schema-design-time-series#patterns_for_row_key_design

   upvoted 35 times

  **AzureDP900** 2 years, 4 months ago

C. Create a narrow table in Bigtable with a row key that combines the Computer Engine computer identifier with the sample time at each second

   upvoted 1 times

  **nadavw** 2 years, 11 months ago



there is a limit of 60 columns per row according to question. in addition in D the cost will be a lower which is a requirement. so D seems more suitable.

   upvoted 1 times

  **madhu1171** Highly Voted  5 years, 1 month ago

C correct answer

   upvoted 19 times

  **shangning007** Most Recent  4 months, 2 weeks ago

Selected Answer: C

Even though I am sure the answer is C, but I am not sure why it will help avoid being charged for every query?

   upvoted 1 times

  **SamuelTsch** 6 months, 1 week ago

Selected Answer: C

just like psu said. Additionally for a time-series data, it is usually the best practice to combine the identifier + time, not the values.

   upvoted 1 times

  **mothkuri** 1 year, 2 months ago

Selected Answer: C

Option C is correct answer. Narrow table is good for time series data.

   upvoted 1 times

  **barnac1es** 1 year, 7 months ago

Selected Answer: C

Scalability: Bigtable can handle large-scale data efficiently, making it suitable for storing time series data for millions of computers.

Low Latency: Bigtable provides low-latency access to data, which is crucial for real-time analytics.

Flexible Schema: The narrow table design allows you to efficiently store and query time series data without specifying all possible columns in advance, providing flexibility for future growth.

Column Families: Bigtable supports column families, allowing you to organize data logically.

Row Key Design: Combining the computer identifier with the sample time at each second in the row key allows for efficient retrieval of data for specific computers and time intervals.

Analytics: While Bigtable does not support SQL directly, it allows for efficient data retrieval and can be integrated with other tools for analytics.

   upvoted 1 times

  **WillemHendr** 1 year, 11 months ago

Selected Answer: D

"..and ensure that the schema design will allow for future growth of the dataset":

<https://cloud.google.com/bigtable/docs/schema-design-time-series#time-buckets>

"Data stored in this way is compressed more efficiently than data in tall, narrow tables."

I read the "future growth" as a sign to be effective in storage, and go for the Time-Buckets.

   upvoted 3 times

  **zellick** 2 years, 5 months ago

Selected Answer: C

C is the answer.

<https://cloud.google.com/bigtable/docs/schema-design-time-series>



   upvoted 2 times

  **Remi2021** 2 years, 7 months ago


Selected Answer: C


time series = narrow table

   upvoted 2 times

  **_8008_** 3 years ago

What about "avoid being charged for every query executed"? Nothing on this topic in here <https://cloud.google.com/bigtable/docs/schema-design-time-series> can anyone comment?

   upvoted 3 times

  **medeis_jar** 3 years, 3 months ago

Selected Answer: C

Narrow and tall table for a single event and good for time-series data

Short and Wide table for data over a month, multiple events

   upvoted 2 times

  **JG123** 3 years, 5 months ago

Correct: C

   upvoted 2 times



  **squishy_fishy** 3 years, 6 months ago

Answer is C.

Bigtable is best suited to the following scenarios: time-series data (e.g. CPU and memory usage over time for multiple servers), financial data (e.g. transaction histories, stock prices, and currency exchange rates), and IoT (Internet of Things) use cases.


<https://www.xplenty.com/blog/bigtable-vs-bigquery/>

   upvoted 3 times

  **safiyu** 3 years, 8 months ago

C is the correct answer. If you consider wide table, then 60 columns for cpu usage and 60 columns for memory usage. in future, if you need to add a new kpi to the table, then the schema changes. you will have to add 60 more columns for the new feature. this is not so future proof.. so D is out of the picture.

   upvoted 5 times

  **DeepakS227** 3 years, 9 months ago

BQ is optimized for large-scale, ad-hoc SQL-based analysis. i Think it should be A

   upvoted 2 times



  **koupayio** 3 years, 11 months ago

First C & D won't cause hotspotting as computer_identifier is first part of row key

I prefer D because "ensure that the schema design will allow for future growth of the dataset."

C is too tall and narrow I cannot see schema design grow in the future

   upvoted 1 times

  **crslake** 3 years, 11 months ago

D, Better overall, harder to implement (but that is not stated as a constraint)

<https://cloud.google.com/bigtable/docs/schema-design-time-series#time-buckets>

   upvoted 1 times

  **lollo1234** 3 years, 11 months ago

How do you store both CPU and memory usage though? Two sets of 60 columns per row? I am wondering if that goes along with "the schema design will allow for future growth"...what if by future growth they mean monitoring N more metrics. That would imply N*60 columns, right?

   upvoted 2 times

[Load full discussion...](#)



Platform

> [Home](#)

> [Examtopics PRO](#)

> [All Exams](#)

> [Training Courses](#)

