

🔗 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 140 DISCUSSION

Actual exam question from Google's Professional Data Engineer

Question #: 140

Topic #: 1

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

You need to create a new transaction table in Cloud Spanner that stores product sales data. You are deciding what to use as a primary key. From a performance perspective, which strategy should you choose?

- A. The current epoch time
- B. A concatenation of the product name and the current epoch time
- C. A random universally unique identifier number (version 4 UUID)
- D. The original order identification number from the sales system, which is a monotonically increasing integer

[Show Suggested Answer](#)

by [jsree236](#) at *Sept. 5, 2022, 10:23 a.m.*

Comments

Type your comment...

[Submit](#)

🗂️ [Remi2021](#) [Highly Voted](#) 👍 2 years, 1 month ago

Selected Answer: C

According to the documentation:
Use a Universally Unique Identifier (UUID)

See a Universally Unique Identifier (UUID)

You can use a Universally Unique Identifier (UUID) as defined by RFC 4122 as the primary key. Version 4 UUID is recommended, because it uses random values in the bit sequence. Version 1 UUID stores the timestamp in the high order bits and is not recommended.

<https://cloud.google.com/spanner/docs/schema-design>

   upvoted 9 times

  **AzureDP900** 1 year, 10 months ago

Agree with C

   upvoted 1 times

  **barnac1es** Most Recent 1 year, 1 month ago

Selected Answer: C

For a transaction table in Cloud Spanner that stores product sales data, from a performance perspective, it is generally recommended to choose a primary key that allows for even distribution of data across nodes and minimizes hotspots. Therefore, option C, which suggests using a random universally unique identifier number (version 4 UUID), is the preferred choice.

   upvoted 4 times

  **arien_chen** 1 year, 2 months ago



Selected Answer: C

For a RDB I would choice D.

But for Google Spanner, Google says:

<https://cloud.google.com/spanner/docs/schema-and-data-model#:~:text=monotonically%20increasing%20integer>

   upvoted 1 times

  **vaga1** 1 year, 5 months ago

Selected Answer: C

B might work if you say timestamp instead than epoch. PK of sales should contain the exact purchase date or timestamp, not the time when the transaction was processed. I personally associate the term epoch in this context to the process timestamp instead than the purchase timestamp.

   upvoted 2 times

  **midgoo** 1 year, 7 months ago

Selected Answer: C

B may cause error if same product ID came at the same time (same id + same epoch)
So C is the correct answer here

   upvoted 2 times

  **NickNtaken** 5 months, 1 week ago

Agreed. Additionally, using the product name can lead to unbalanced distribution if some products are sold more frequently than others.

   upvoted 1 times



  **jkhong** 1 year, 10 months ago

Selected Answer: C

A and D are invalid because they monotonically increases.

B would work, but in terms of pure performance UUID 4 is the fastest because it virtually will not cause hotspots

   upvoted 2 times

  **odacir** 1 year, 11 months ago

Selected Answer: C

A and D are not valid, because they monotonically increase.

C avoid hotspots for sure, but It's nor relate with queries. So for writing performance it's perfect that the reason for chose this: "You need to create a new transaction table in Cloud Spanner that stores product sales data". They only ask you to store product data, its a writing ops.

If the question had spoken about query the info or hard performance read, the best option would be B, because it has the balance of writing/reading best practices.

There are a few disadvantages to using a UUID:

They are slightly large, using 16 bytes or more. Other options for primary keys don't use this much storage.

They carry no information about the record. For example, a primary key of SingerId and AlbumId has an inherent meaning, while a UUID does not.

You lose locality between records that are related, which is why using a UUID eliminates hotspots.

https://cloud.google.com/spanner/docs/schema-design#uuid_primary_key

   upvoted 3 times

upvoted 3 times

 **YorelNation** 2 years, 2 months ago

Selected Answer: C

C. A random universally unique identifier number (version 4 UUID)

From <https://cloud.google.com/spanner/docs/schema-and-data-model>

There are techniques that can spread the load across multiple servers and avoid hotspots:

Hash the key and store it in a column. Use the hash column (or the hash column and the unique key columns together) as the primary key.

Swap the order of the columns in the primary key.

Use a Universally Unique Identifier (UUID). Version 4 UUID is recommended, because it uses random values in the high-order bits. Don't use a UUID algorithm (such as version 1 UUID) that stores the timestamp in the high order bits.

Bit-reverse sequential values.

   upvoted 1 times

 **jsree236** 2 years, 2 months ago

Selected Answer: B

Answer should be B as in all the other options hotspotting is possible. According to proper schema design guideline..

Schema design best practice #1: Do not choose a column whose value monotonically increases or decreases as the first key part for a high write rate table.

Supporting link:

<https://cloud.google.com/spanner/docs/schema-design#primary-key-prevent-hotspots>

   upvoted 2 times

 **LP_PDE** 4 months ago

Potential Skew: If there are a limited number of product names, this could still lead to uneven data distribution and potential hotspots.

Increased Key Size: Concatenating strings can result in larger primary keys, which can slightly impact storage and performance.

   upvoted 1 times



Platform

> Home

> All Exams

> Examtopics PRO

> Training Courses

