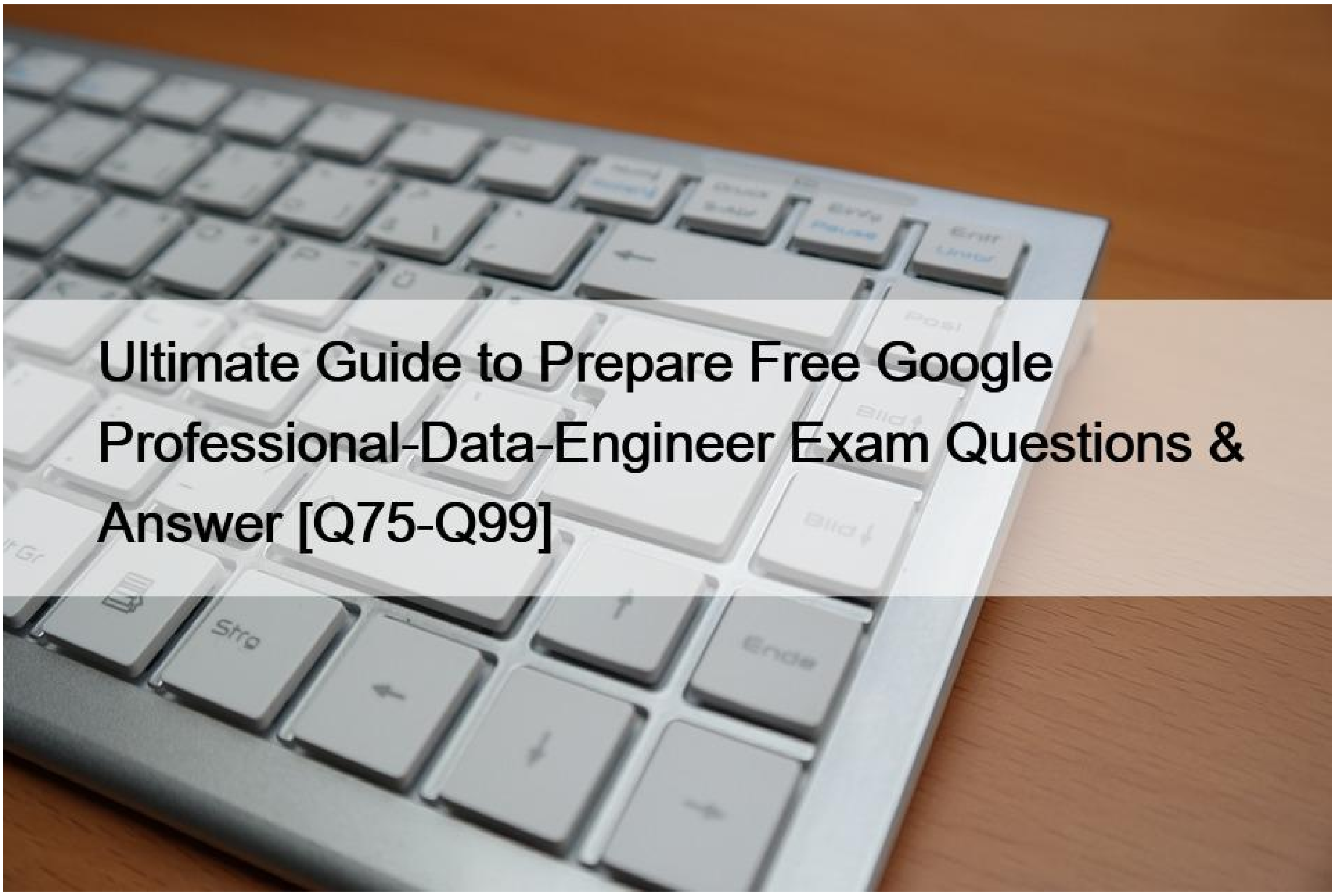
Ultimate Guide to Prepare Free Google Professional-Data-Engineer Exam Questions & Answer [Q75-Q99


Ultimate Guide to Prepare Free Google Professional-Data-Engineer Exam Questions & Answer [Q75-Q99]

Ultimate Guide to Prepare Free Google Professional-Data-Engineer Exam Questions and Answer
Pass Google Professional-Data-Engineer Tests Engine pdf - All Free Dumps

Understanding functional and technical aspects of Google Professional Data Engineer Exam Building and operationalizing data processing systems

The following will be discussed here:

- Monitoring pipelines- Storage costs and performance- Building and operationalizing processing infrastructure- Validating a migration- Transformation- Data acquisition and import- Provisioning resources- Batch and streaming- Integrating with new data sources- Building and operationalizing data processing systems- Testing and quality control- Lifecycle management of data- Building and operationalizing storage systems- Effective use of managed services (Cloud Bigtable, Cloud Spanner, Cloud SQL, BigQuery, Cloud Storage, Cloud Datastore, Cloud Memorystore)- Adjusting pipelines- Building and operationalizing pipelines- Awareness of current state and how to migrate a design to a future state

Understanding functional and technical aspects of Google Professional Data Engineer Exam Ensuring solution quality

The following will be discussed here:

- Verification and monitoring- Ensuring scalability and efficiency- Ensuring privacy (e.g., Data Loss Prevention API)-

Resizing and autoscaling resources- Designing for data and application portability (e.g., multi-cloud, data residency requirements)- Data security (encryption, key management)- Choosing between ACID, idempotent, eventually consistent requirements- Pipeline monitoring (e.g., Stackdriver)- Ensuring reliability and fidelity- Legal compliance (e.g., Health Insurance Portability and Accountability Act (HIPAA), Children's Online Privacy Protection Act (COPPA), FedRAMP, General Data Protection Regulation (GDPR))- Ensuring flexibility and portability **Q75.** You are working on a niche product in the image recognition domain. Your team has developed a model that is dominated by custom C++ TensorFlow ops your team has implemented. These ops are used inside your main training loop and are performing bulky matrix multiplications. It currently takes up to several days to train a model. You want to decrease this time significantly and keep the cost low by using an accelerator on Google Cloud. What should you do?
* Use Cloud TPUs without any additional adjustment to your code.
* Use Cloud TPUs after implementing GPU kernel support for your customs ops.
* Use Cloud GPUs after implementing GPU kernel support for your customs ops.
* Stay on CPUs, and increase the size of the cluster you&#8217;re training your model on.

Cloud TPUs are not suited to the following workloads: [&#8230;] Neural network workloads that contain custom TensorFlow operations written in C++. Specifically, custom operations in the body of the main training loop are not suitable for TPUs.

Q76. Your company is using WHILECARD tables to query data across multiple tables with similar names. The SQL statement is currently failing with the following error:

# Syntax error : Expected end of statement but got &#8220;-&#8221; at [4:11] SELECT age FROM bigquery-public-data.noaa_gsod.gsod WHERE age != 99 AND_TABLE_SUFFIX = `1929&#8242; ORDER BY age DESC Which table name will make the SQL statement work correctly?
* `bigquery-public-data.noaa_gsod.gsod`
* bigquery-public-data.noaa_gsod.gsod*
* `bigquery-public-data.noaa_gsod.gsod&#8217;*
* `bigquery-public-data.noaa_gsod.gsod*`

Q77. You are designing storage for two relational tables that are part of a 10-TB database on Google Cloud. You want to support transactions that scale horizontally. You also want to optimize data for range queries on nonkey columns. What should you do?
* Use Cloud SQL for storage. Add secondary indexes to support query patterns.
* Use Cloud SQL for storage. Use Cloud Dataflow to transform data to support query patterns.
* Use Cloud Spanner for storage. Add secondary indexes to support query patterns.
* Use Cloud Spanner for storage. Use Cloud Dataflow to transform data to support query patterns.

Q78. Business owners at your company have given you a database of bank transactions. Each row contains the

user ID, transaction type, transaction location, and transaction amount. They ask you to investigate what

type of machine learning can be applied to the data. Which three machine learning applications can you

use? (Choose three.)
* Supervised learning to determine which transactions are most likely to be fraudulent.
* Unsupervised learning to determine which transactions are most likely to be fraudulent.
* Clustering to divide the transactions into N categories based on feature similarity.
* Supervised learning to predict the location of a transaction.
* Reinforcement learning to predict the location of a transaction.
* Unsupervised learning to predict the location of a transaction.

Q79. You are planning to migrate your current on-premises Apache Hadoop deployment to the cloud. You need to ensure that the deployment is as fault-tolerant and cost-effective as possible for long-running batch jobs. You want to use a managed service. What

should you do?
* Deploy a Cloud Dataproc cluster. Use a standard persistent disk and 50% preemptible workers. Store data in Cloud Storage, and change references in scripts from hdfs:// to gs://
* Deploy a Cloud Dataproc cluster. Use an SSD persistent disk and 50% preemptible workers. Store data in Cloud Storage, and change references in scripts from hdfs:// to gs://
* Install Hadoop and Spark on a 10-node Compute Engine instance group with standard instances. Install the Cloud Storage connector, and store the data in Cloud Storage. Change references in scripts from hdfs:// to gs://
* Install Hadoop and Spark on a 10-node Compute Engine instance group with preemptible instances.

Store data in HDFS. Change references in scripts from hdfs:// to gs://

Q80. Which of the following statements about Legacy SQL and Standard SQL is not true?
* Standard SQL is the preferred query language for BigQuery.
* If you write a query in Legacy SQL, it might generate an error if you try to run it with Standard SQL.
* One difference between the two query languages is how you specify fully-qualified table names (i.e. table names that include their associated project name).
* You need to set a query language for each dataset and the default is Standard SQL.
You do not set a query language for each dataset. It is set each time you run a query and the default query language is Legacy SQL.

Standard SQL has been the preferred query language since BigQuery 2.0 was released.

In legacy SQL, to query a table with a project-qualified name, you use a colon, :, as a separator. In standard SQL, you use a period, ., instead.

Due to the differences in syntax between the two query languages (such as with project- qualified table names), if you write a query in Legacy SQL, it might generate an error if you try to run it with Standard SQL.

Reference:

https://cloud.google.com/bigquery/docs/reference/standard-sql/migrating-from-legacy-sql

Q81. What are two of the benefits of using denormalized data structures in BigQuery?
* Reduces the amount of data processed, reduces the amount of storage required
* Increases query speed, makes queries simpler
* Reduces the amount of storage required, increases query speed
* Reduces the amount of data processed, increases query speed
Denormalization increases query speed for tables with billions of rows because BigQuery&#8217;s performance degrades when doing JOINs on large tables, but with a denormalized data

structure, you don&#8217;t have to use JOINs, since all of the data has been combined into one table. Denormalization also makes queries simpler because you do not have to use JOIN clauses.

Denormalization increases the amount of data processed and the amount of storage required because it creates redundant data.

Reference:

https://cloud.google.com/solutions/bigquery-data-warehouse#denormalizing_data

Q82. Why do you need to split a machine learning dataset into training data and test data?
* So you can try two different sets of features

* To make sure your model is generalized for more than just the training data
* To allow you to create unit tests in your code
* So you can use one dataset for a wide model and one for a deep model

The flaw with evaluating a predictive model on training data is that it does not inform you on how well the model has generalized to new unseen data. A model that is selected for its accuracy on the training dataset rather than its accuracy on an unseen test dataset is very likely to have lower accuracy on an unseen test dataset. The reason is that the model is not as generalized. It has specialized to the structure in the training dataset. This is called overfitting.

Q83. The __________ for Cloud Bigtable makes it possible to use Cloud Bigtable in a Cloud Dataflow pipeline.
* Cloud Dataflow connector
* DataFlow SDK
* BiqQuery API
* BigQuery Data Transfer Service

The Cloud Dataflow connector for Cloud Bigtable makes it possible to use Cloud Bigtable in a Cloud Dataflow pipeline. You can use the connector for both batch and streaming operations.

Q84. You want to use a database of information about tissue samples to classify future tissue samples as either normal or mutated. You are evaluating an unsupervised anomaly detection method for classifying the tissue samples. Which two characteristic support this method? (Choose two.)
* There are very few occurrences of mutations relative to normal samples.
* There are roughly equal occurrences of both normal and mutated samples in the database.
* You expect future mutations to have different features from the mutated samples in the database.
* You expect future mutations to have similar features to the mutated samples in the database.
* You already have labels for which samples are mutated and which are normal in the database.

Unsupervised anomaly detection techniques detect anomalies in an unlabeled test data set under the assumption that the majority of the instances in the data set are normal by looking for instances that seem to fit least to the remainder of the data set.

https://en.wikipedia.org/wiki/Anomaly_detection

Q85. You want to use a BigQuery table as a data sink. In which writing mode(s) can you use BigQuery as a sink?
* Both batch and streaming
* BigQuery cannot be used as a sink
* Only batch
* Only streaming

Explanation

When you apply a BigQueryIO.Write transform in batch mode to write to a single table, Dataflow invokes a BigQuery load job. When you apply a BigQueryIO.Write transform in streaming mode or in batch mode using a function to specify the destination table, Dataflow uses BigQuery&#8217;s streaming inserts Reference: https://cloud.google.com/dataflow/model/bigquery-io

Q86. Your financial services company is moving to cloud technology and wants to store 50 TB of financial time- series data in the cloud. This data is updated frequently and new data will be streaming in all the time. Your company also wants to move their existing Apache Hadoop jobs to the cloud to get insights into this data.

Which product should they use to store the data?
* Cloud Bigtable
* Google BigQuery
* Google Cloud Storage
* Google Cloud Datastore

https://cloud.google.com/blog/products/databases/getting-started-with-time-series-trend-predictions-using- gcp

Q87. Which of the following is NOT one of the three main types of triggers that Dataflow supports?
* Trigger based on element size in bytes
* Trigger that is a combination of other triggers
* Trigger based on element count
* Trigger based on time
There are three major kinds of triggers that Dataflow supports: 1. Time-based triggers 2.

Data-driven triggers. You can set a trigger to emit results from a window when that window has received a certain number of data elements. 3. Composite triggers. These triggers combine multiple time-based or data-driven triggers in some logical way Reference: https://cloud.google.com/dataflow/model/triggers

Q88. You&#8217;re training a model to predict housing prices based on an available dataset with real estate properties.

Your plan is to train a fully connected neural net, and you&#8217;ve discovered that the dataset contains latitude and longitude of the property. Real estate professionals have told you that the location of the property is highly influential on price, so you&#8217;d like to engineer a feature that incorporates this physical dependency.

What should you do?
* Provide latitude and longitude as input vectors to your neural net.
* Create a numeric column from a feature cross of latitude and longitude.
* Create a feature cross of latitude and longitude, bucketize at the minute level and use L1 regularization during optimization.
* Create a feature cross of latitude and longitude, bucketize it at the minute level and use L2 regularization during optimization.
Use L1 regularization when you need to assign greater importance to more influential features. It shrinks less important feature to 0.

L2 regularization performs better when all input features influence the output & all with the weights are of equal size.

Q89. You work for a shipping company that has distribution centers where packages move on delivery lines to route them properly. The company wants to add cameras to the delivery lines to detect and track any visual damage to the packages in transit. You need to create a way to automate the detection of damaged packages and flag them for human review in real time while the packages are in transit. Which solution should you choose?
* Use BigQuery machine learning to be able to train the model at scale, so you can analyze the packages in batches.
* Train an AutoML model on your corpus of images, and build an API around that model to integrate with the package tracking applications.
* Use the Cloud Vision API to detect for damage, and raise an alert through Cloud Functions. Integrate the package tracking applications with this function.
* Use TensorFlow to create a model that is trained on your corpus of images. Create a Python notebook in Cloud Datalab that uses this model so you can analyze for damaged packages.

Q90. You designed a database for patient records as a pilot project to cover a few hundred patients in three clinics.

Your design used a single database table to represent all patients and their visits, and you used self-joins to generate reports. The server resource utilization was at 50%. Since then, the scope of the project has expanded.

The database must now store 100 times more patient records. You can no longer run the reports, because they either take too long or they encounter errors with insufficient compute resources. How should you adjust the database design?
* Add capacity (memory and disk space) to the database server by the order of 200.
* Shard the tables into smaller ones based on date ranges, and only generate reports with prespecified date ranges.
* Normalize the master patient-record table into the patient table and the visits table, and create other necessary tables to avoid self-join.

* Partition the table into smaller tables, with one for each clinic. Run queries against the smaller table pairs, and use unions for consolidated reports.

Q91. You are running a pipeline in Cloud Dataflow that receives messages from a Cloud Pub/Sub topic and writes the results to a BigQuery dataset in the EU. Currently, your pipeline is located in europe-west4 and has a maximum of 3 workers, instance type n1-standard-1. You notice that during peak periods, your pipeline is struggling to process records in a timely fashion, when all 3 workers are at maximum CPU utilization. Which two actions can you take to increase performance of your pipeline? (Choose two.)
* Increase the number of max workers
* Use a larger instance type for your Cloud Dataflow workers
* Change the zone of your Cloud Dataflow pipeline to run in us-central1
* Create a temporary table in Cloud Bigtable that will act as a buffer for new data. Create a new step in your pipeline to write to this table first, and then create a new pipeline to write from Cloud Bigtable to BigQuery
* Create a temporary table in Cloud Spanner that will act as a buffer for new data. Create a new step in your pipeline to write to this table first, and then create a new pipeline to write from Cloud Spanner to BigQuery
Explanation/Reference:

Q92. Your company&#8217;s on-premises Apache Hadoop servers are approaching end-of-life, and IT has decided to migrate the cluster to Google Cloud Dataproc. A like-for-like migration of the cluster would require 50 TB of Google Persistent Disk per node. The CIO is concerned about the cost of using that much block storage.

You want to minimize the storage cost of the migration. What should you do?
* Put the data into Google Cloud Storage.
* Use preemptible virtual machines (VMs) for the Cloud Dataproc cluster.
* Tune the Cloud Dataproc cluster so that there is just enough disk for all data.
* Migrate some of the cold data into Google Cloud Storage, and keep only the hot data in Persistent Disk.

Q93. Your weather app queries a database every 15 minutes to get the current temperature. The frontend is powered by Google App Engine and server millions of users. How should you design the frontend to respond to a database failure?
* Issue a command to restart the database servers.
* Retry the query with exponential backoff, up to a cap of 15 minutes.
* Retry the query every second until it comes back online to minimize staleness of data.
* Reduce the query frequency to once every hour until the database comes back online.
Explanation

https://cloud.google.com/sql/docs/mysql/manage-connections#backoff

Q94. When a Cloud Bigtable node fails, ____ is lost.
* all data
* no data
* the last transaction
* the time dimension
A Cloud Bigtable table is sharded into blocks of contiguous rows, called tablets, to help balance the workload of queries. Tablets are stored on Colossus, Google&#8217;s file system, in SSTable format. Each tablet is associated with a specific Cloud Bigtable node.

Data is never stored in Cloud Bigtable nodes themselves; each node has pointers to a set of tablets that are stored on Colossus. As a result:

Rebalancing tablets from one node to another is very fast, because the actual data is not copied. Cloud Bigtable simply updates the pointers for each node.

Recovery from the failure of a Cloud Bigtable node is very fast, because only metadata needs to be migrated to the replacement node.

When a Cloud Bigtable node fails, no data is lost

Q95. Which row keys are likely to cause a disproportionate number of reads and/or writes on a particular node in a Bigtable cluster (select 2 answers)?
* A sequential numeric ID
* A timestamp followed by a stock symbol
* A non-sequential numeric ID
* A stock symbol followed by a timestamp
&#8230;using a timestamp as the first element of a row key can cause a variety of problems. In brief, when a row key for a time series includes a timestamp, all of your writes will target a single node; fill that node; and then move onto the next node in the cluster, resulting in hotspotting. Suppose your system assigns a numeric ID to each of your application&#8217;s users. You might be tempted to use the user&#8217;s numeric ID as the row key for your table. However, since new users are more likely to be active users, this approach is likely to push most of your traffic to a small number of nodes. [https://cloud.google.com/bigtable/docs/schema- design] Reference: https://cloud.google.com/bigtable/docs/schema-design-time-series#ensure_that_your_row_key_avoids_hotspotting

Q96. You are planning to use Google&#8217;s Dataflow SDK to analyze customer data such as displayed below. Your project requirement is to extract only the customer name from the data source and then write to an output PCollection.

Tom,555 X street

Tim,553 Y street

Sam, 111 Z street

Which operation is best suited for the above data processing requirement?
* ParDo
* Sink API
* Source API
* Data extraction
In Google Cloud dataflow SDK, you can use the ParDo to extract only a customer name of each element in your PCollection.

Reference: https://cloud.google.com/dataflow/model/par-do

Q97. Which row keys are likely to cause a disproportionate number of reads and/or writes on a particular node in a Bigtable cluster (select 2 answers)?
* A sequential numeric ID
* A timestamp followed by a stock symbol
* A non-sequential numeric ID
* A stock symbol followed by a timestamp
&#8230;using a timestamp as the first element of a row key can cause a variety of problems.

In brief, when a row key for a time series includes a timestamp, all of your writes will target a single node; fill that node; and then move onto the next node in the cluster, resulting in hotspotting.

Suppose your system assigns a numeric ID to each of your application&#8217;s users. You might be tempted to use the user&#8217;s numeric ID as the row key for your table. However, since new users are more likely to be active users, this approach is likely to push most of your traffic to a small number of nodes. [https://cloud.google.com/bigtable/docs/schema-design]

Q98. You work for an advertising company, and you&#8217;ve developed a Spark ML model to predict click-through rates at advertisement blocks. You&#8217;ve been developing everything at your on-premises data center, and now your company is migrating to Google Cloud. Your data center will be migrated to BigQuery. You periodically retrain your Spark ML models, so you need to migrate existing training pipelines to Google Cloud. What should you do?
* Use Cloud ML Engine for training existing Spark ML models
* Rewrite your models on TensorFlow, and start using Cloud ML Engine
* Use Cloud Dataproc for training existing Spark ML models, but start reading data directly from BigQuery
* Spin up a Spark cluster on Compute Engine, and train Spark ML models on the data exported from BigQuery

Q99. You have spent a few days loading data from comma-separated values (CSV) files into the Google

BigQuery table CLICK_STREAM. The column DTstores the epoch time of click events. For convenience,

you chose a simple schema where every field is treated as the STRINGtype. Now, you want to compute

web session durations of users who visit your site, and you want to change its data type to the

TIMESTAMP. You want to minimize the migration effort without making future queries computationally

expensive. What should you do?
* Delete the table CLICK_STREAM, and then re-create it such that the column DTis of the TIMESTAMP

type. Reload the data.
* Add a column TSof the TIMESTAMPtype to the table CLICK_STREAM, and populate the numeric

values from the column TSfor each row. Reference the column TSinstead of the column DTfrom now

on.
* Create a view CLICK_STREAM_V, where strings from the column DTare cast into TIMESTAMPvalues.

Reference the view CLICK_STREAM_Vinstead of the table CLICK_STREAMfrom now on.
* Add two columns to the table CLICK STREAM: TSof the TIMESTAMPtype and IS_NEWof the

BOOLEANtype. Reload all data in append mode. For each appended row, set the value of IS_NEWto

true. For future queries, reference the column TSinstead of the column DT, with the WHEREclause

ensuring that the value of IS_NEWmust be true.
* Construct a query to return every row of the table CLICK_STREAM, while using the built-in function to

cast strings from the column DTinto TIMESTAMPvalues. Run the query into a destination table

NEW_CLICK_STREAM, in which the column TSis the TIMESTAMPtype. Reference the table

NEW_CLICK_STREAMinstead of the table CLICK_STREAMfrom now on. In the future, new data is

loaded into the table NEW_CLICK_STREAM.

Online Exam Practice Tests with detailed explanations!: https://www.dumpleader.com/Professional-Data-Engineer_exam.html]