01 GCP PDE Questions

https://www.examtopics.com/discussions/google/view/79580-exam-professional-data-engineer-topic-1-question-183/

1. You are architecting a data transformation solution for BigQuery. Your developers are proficient with SQL and want to use the ELT development technique. In addition, your developers need an intuitive coding environment and the ability to manage SQL as code. You need to identify a solution for your developers to build these pipelines.

A. Use Dataform to build, manage, and schedule SQL pipelines.

- Aligns with ELT Approach: Dataform is designed for ELT (Extract, Load, Transform) pipelines, directly executing SQL transformations within BigQuery, matching the developers' preference. -SQL as Code: It enables developers to write and manage SQL transformations as code, promoting version control, collaboration, and testing. - Intuitive Coding Environment: Dataform provides a user-friendly interface and familiar SQL syntax, making it easy for SQL-proficient developers to adopt. - Scheduling and Orchestration: It includes built-in scheduling capabilities to automate pipeline execution, simplifying pipeline management.
  link
  B. Use Dataflow jobs to read data from Pub/Sub, transform the data, and load the data to BigQuery.
  C. Use Data Fusion to build and execute ETL pipelines.
  D. Use Cloud Composer to load data and run SQL pipelines by using the BigQuery job operators.

2. You work for a farming company. You have one BigQuery table named sensors, which is about 500 MB and contains the list of your 5000 sensors, with columns for id, name, and location. This table is updated every hour. Each sensor generates one metric every 30 seconds along with a timestamp, which you want to store in BigQuery. You want to run an analytical query on the data once a week for monitoring purposes. You also want to minimize costs. What data model should you use?

A. 1. Create a metrics column in the sensors table.
2. Set RECORD type and REPEATED mode for the metrics column.
3. Use an UPDATE statement every 30 seconds to add new metrics.

B. 1. Create a metrics column in the sensors table.
2. Set RECORD type and REPEATED mode for the metrics column.
3. Use an INSERT statement every 30 seconds to add new metrics.

C. 1. Create a metrics table partitioned by timestamp.
2. **Create a sensorId column in the metrics table, that points to the id column in the sensors table.**
3. **Use an INSERT statement every 30 seconds to append new metrics to the metrics table.**
4. **Join the two tables, if needed, when running the analytical query.**

-> Separate Tables: Static sensor information will be same but the sensor will scale
-> Partitioning; Cost Optimization

- Imagine you set up a partition by **DAY**. When the first sensor data arrives with a timestamp of 2024-06-25 (today), BigQuery automatically creates a new partition for data dated June 25th, 2024.
- Subsequent data points with timestamps falling within June 25th will be added to this partition.
  -> Flexibility

D. 1. Create a metrics table partitioned by timestamp.
2. Create a sensorId column in the metrics table, which points to the id column in the sensors table.
3. Use an UPDATE statement every 30 seconds to append new metrics to the metrics table.
4. Join the two tables, if needed, when running the analytical query.

3. You are managing a Dataplex environment with raw and curated zones. A data engineering team is uploading JSON and CSV files to a bucket asset in the curated zone but the files are not being automatically discovered by Dataplex. What should you do to ensure that the files are discovered by Dataplex?

A. Move the JSON and CSV files to the raw zone.
- Columnar Data
- Yes, because Curated zones store structured Data in Cloud Storage or BigQuery Datasets
- Formats for GCS include Parquet, Avro, ORC
- https://cloud.google.com/dataplex/docs/add-zone

B. Enable auto-discovery of files for the curated zone.
C. Use the bq command-line tool to load the JSON and CSV files into BigQuery tables.
D. Grant object level access to the CSV and JSON files in Cloud Storage.

no becuase you can't place csv and json in curated region, only columnar based files can be placed in curated region.

4. You have a table that contains millions of rows of sales data, partitioned by date. Various applications and users query this data many times a minute. The query requires aggregating values by using AVG, MAX, and SUM, and does not require joining to other tables. The required aggregations are only computed over the past year of data, though you need to retain full historical data in the base tables. You want to ensure that the query results always include the latest data from the tables, while also reducing computation cost, maintenance overhead, and duration. What should you do?

A. Create a materialized view to aggregate the base table data. Include a filter clause to specify the last one year of partitions.

- **Performance:** Materialized views pre-compute aggregations (AVG, MAX, SUM) on a subset of the data (last year).
- **Freshness:** Since materialized views are refreshed periodically (by default), they ensure the results always include the latest data from the base table.
- **Reduced Cost & Maintenance:** Compared to replicating the entire table's data or recreating an aggregated table every hour, materialized views offer better efficiency. They store only the pre-computed aggregates, minimizing storage costs and maintenance overhead.
- **Filtering:** Including a filter clause in the materialized view definition ensures it only aggregates data for the past year, aligning with the query requirement.

B. Create a materialized view to aggregate the base table data. Configure a partition expiration on the base table to retain only the last one year of partitions.

C. Create a view to aggregate the base table data. Include a filter clause to specify the last year of partitions.

D. Create a new table that aggregates the base table data. Include a filter clause to specify the last year of partitions. Set up a scheduled query to recreate the new table every hour.

5. Your organization uses a multi-cloud data storage strategy, storing data in Cloud Storage, and data in Amazon Web Services' (AWS) S3 storage buckets. All data resides in US regions. You want to query up-to-date data by using BigQuery, regardless of which cloud the data is stored in. You need to allow users to query the tables from BigQuery without giving direct access to the data in the storage buckets. What should you do?

A. Setup a BigQuery Omni connection to the AWS S3 bucket data. Create BigLake tables over the Cloud Storage and S3 data and query the data using BigQuery directly.

- **Performance is a top priority:** BigLake tables offer significant performance optimizations for querying large datasets across multiple clouds.

- **Unified data management is desired:** BigLake tables provide a centralized way to manage and govern data, simplifying data governance and access control in a multi-cloud environment.
- **Advanced features are needed:** BigLake tables unlock advanced functionalities like caching, partitioning, and materialized views available in BigQuery.
- **Scalability for future growth is important:** BigLake tables are a more robust and scalable solution for handling large-scale data querying across clouds as your data grows.

B. Set up a BigQuery Omni connection to the AWS S3 bucket data. Create external tables over the Cloud Storage and S3 data and query the data using BigQuery directly.

- **Simplicity is crucial:** External tables are easier to set up initially compared to BigLake tables.
- **Ad-hoc queries dominate:** This option is suitable for occasional or exploratory queries on data residing in various sources.
- **Data volume is relatively small:** Performance impact of external tables might be less noticeable for smaller datasets.

C. Use the Storage Transfer Service to copy data from the AWS S3 buckets to Cloud Storage buckets. Create BigLake tables over the Cloud Storage data and query the data using BigQuery directly.
D. Use the Storage Transfer Service to copy data from the AWS S3 buckets to Cloud Storage buckets. Create external tables over the Cloud Storage data and query the data using BigQuery directly.

6. You are preparing an organization-wide dataset. You need to preprocess customer data stored in a restricted bucket in Cloud Storage. The data will be used to create consumer analyses. You need to comply with data privacy requirements. What should you do?

A. Use Dataflow and the Cloud Data Loss Prevention API to mask sensitive data. Write the processed data in BigQuery.
Data Anonymization: Dataflow, a streaming and batch data processing service, can be used with the Cloud Data Loss Prevention (DLP) API to achieve anonymization. DLP offers various techniques like masking, tokenization, and redaction to obfuscate sensitive information such as names, addresses, and social security numbers.
Preserves Utility: Masking sensitive data with DLP helps anonymize the data while still preserving its usefulness for consumer analyses. You can define specific data fields to be masked while keeping the rest of the data intact.
BigQuery Storage: Writing the anonymized data to BigQuery allows for efficient storage, querying, and analysis of consumer data for marketing and other business purposes.
B. Use customer-managed encryption keys (CMEK) to directly encrypt the data in Cloud Storage. Use federated queries from BigQuery. Share the encryption key by following the principle of least privilege.

- Does not mask the data
- Decryption is still required to run the queries
C. Use the Cloud Data Loss Prevention API and Dataflow to detect and remove sensitive fields from the data in Cloud Storage. Write the filtered data in BigQuery.
D. Use Dataflow and Cloud KMS to encrypt sensitive fields and write the encrypted data in BigQuery. Share the encryption key by following the principle of least privilege.

7. You need to connect multiple applications with dynamic public IP addresses to a Cloud SQL instance. You configured users with strong passwords and enforced the SSL connection to your Cloud SQL instance. You want to use Cloud SQL public IP and ensure that you have secured connections. What should you do?

A. Add CIDR 0.0.0.0/0 network to Authorized Network. Use Identity and Access Management (IAM) to add users.
- Adding CIDR 0.0.0.0/0 essentially allows all IPs to connect, defeating the purpose of restricting access. While IAM can be used for user authentication, it doesn't address the challenge of dynamic IPs for applications.
B. Add all application networks to Authorized Network and regularly update them.
C. Leave the Authorized Network empty. Use Cloud SQL Auth proxy on all applications.

- **Cloud SQL Auth Proxy:** This is Google's recommended method for securing connections to Cloud SQL instances with public IPs. It acts as a middle layer between your applications and the Cloud SQL instance.
- **Dynamic IP Management:** Cloud SQL Auth Proxy eliminates the need to manage a constantly updated list of authorized IP addresses (CIDR ranges) for your applications.
- **IAM Integration:** Cloud SQL Auth Proxy integrates with Identity and Access Management (IAM), allowing you to define granular access controls for users and applications connecting to the database. Users can authenticate using their Google Cloud account credentials or service accounts, eliminating the need for managing strong passwords directly on the application side.
- **Improved Security:** Cloud SQL Auth Proxy encrypts the communication between the applications and the Cloud SQL instance using SSL, adding an extra layer of security.
D. Add CIDR 0.0.0.0/0 network to Authorized Network. Use Cloud SQL Auth proxy on all applications.

8. You are migrating a large number of files from a public HTTPS endpoint to Cloud Storage. The files are protected from unauthorized access using signed URLs. You created a TSV file that contains the list of object URLs and started a transfer job by using Storage Transfer Service. You notice that the job has run for a long time and eventually failed. Checking the logs of the transfer job reveals that the job was running fine until one point, and then it

failed due to HTTP 403 errors on the remaining files. You verified that there were no changes to the source system. You need to fix the problem to resume the migration process. What should you do?

A. Set up Cloud Storage FUSE, and mount the Cloud Storage bucket on a Compute Engine instance. Remove the completed files from the TSV file. Use a shell script to iterate through the TSV file and download the remaining URLs to the FUSE mount point.
B. Renew the TLS certificate of the HTTPS endpoint. Remove the completed files from the TSV file and rerun the Storage Transfer Service job.
C. Create a new TSV file for the remaining files by generating signed URLs with a longer validity period. Split the TSV file into multiple smaller files and submit them as separate Storage Transfer Service jobs in parallel.

- **HTTP 403 Errors:** The error code indicates "Forbidden" access. Since the source system hasn't changed and signed URLs were used initially, the issue likely lies with the signed URL validity period.
- **Longer Validity Period:** Generating new signed URLs with a longer validity period ensures the URLs remain valid throughout the transfer process.
- **Splitting the TSV:** Dividing the large TSV into smaller files helps manage potential issues within a single file and allows for parallel processing. Submitting them as separate jobs improves efficiency.
  D. Update the file checksums in the TSV file from using MD5 to SHA256. Remove the completed files from the TSV file and rerun the Storage Transfer Service job.

9. You work for an airline and you need to store weather data in a BigQuery table. Weather data will be used as input to a machine learning model. The model only uses the last 30 days of weather data. You want to avoid storing unnecessary data and minimize costs. What should you do?

A. Create a BigQuery table where each record has an ingestion timestamp. Run a scheduled query to delete all the rows with an ingestion timestamp older than 30 days.
B. Create a BigQuery table partitioned by datetime value of the weather date. Set up partition expiration to 30 days.

- **Partitioning by Weather Date:** Partitioning the table by the "datetime value of the weather date" ensures efficient data organization for your machine learning model, which only requires the last 30 days of data.
- **Partition Expiration:** Setting an expiration of 30 days on the partitions automatically deletes older data partitions that are no longer needed by the model. This helps minimize storage costs as you're not keeping unnecessary historical weather data.
- **Automatic Management:** Partition expiration automates data lifecycle management, reducing the need for manual deletion or scheduled queries (like options A and D).

C. Create a BigQuery table partitioned by ingestion time. Set up partition expiration to 30 days.
D. Create a BigQuery table with a datetime column for the day the weather data refers to. Run a scheduled query to delete rows with a datetime value older than 30 days.

10. You need to look at BigQuery data from a specific table multiple times a day. The underlying table you are querying is several petabytes in size, but you want to filter your data and provide simple aggregations to downstream users. You want to run queries faster and get up-to-date insights quicker. What should you do?

A. Run a scheduled query to pull the necessary data at specific intervals daily.
B. Use a cached query to accelerate time to results.
C. Limit the query columns being pulled in the final result.
D. Create a materialized view based off of the query being run.

- **Pre-computed Data:** Materialized views are essentially pre-computed summaries of your main table. They store the results of your query with filtering and aggregations, ready for immediate retrieval.
- **Faster Queries:** Downstream users querying the materialized view experience significantly faster response times compared to querying the massive petabyte-sized source table every time.
- **Up-to-Date Insights:** You can define a refresh schedule for the materialized view to ensure it reflects the latest data in the source table. This provides users with close-to-real-time insights.

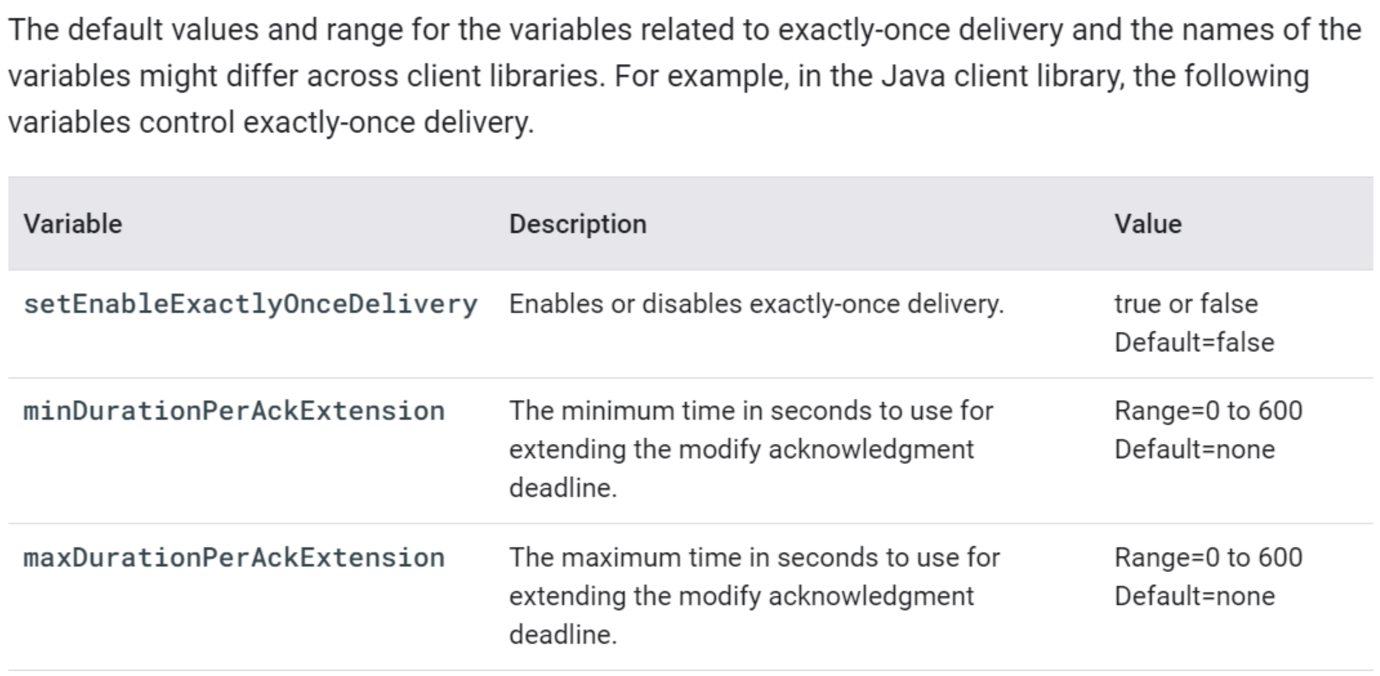
11. Your chemical company needs to manually check documentation for customer order You use a pull subscription in Pub/Sub so that sales agents get details from the order. You must ensure that you do not process orders twice with different sales agents and that you do not add more complexity to this workflow. What should you do?

A. Use a Deduplicate PTransform in Dataflow before sending the messages to the sales agents.
B. Create a transactional database that monitors the pending messages.
C. Use Pub/Sub exactly-once delivery in your pull subscription.

- **Exactly-Once Delivery:** Pub/Sub's exactly-once delivery guarantees that a message is delivered and processed by a subscriber exactly once. This eliminates the risk of duplicate order processing.
- **Minimal Workflow Impact:** Enabling exactly-once delivery in your existing pull subscription avoids introducing a completely new system or complex transactional database management (options B and D).

D. Create a new Pub/Sub push subscription to monitor the orders processed in the agent's system.

https://cloud.google.com/pubsub/docs/exactly-once-delivery



The default values and range for the variables related to exactly-once delivery and the names of the variables might differ across client libraries. For example, in the Java client library, the following variables control exactly-once delivery.

Variable	Description	Value
`setEnableExactlyOnceDelivery`	Enables or disables exactly-once delivery.	true or false Default=false

`minDurationPerAckExtension`	The minimum time in seconds to use for extending the modify acknowledgment deadline.	Range=0 to 600
		Default=none

`maxDurationPerAckExtension`	The maximum time in seconds to use for extending the modify acknowledgment deadline.	Range=0 to 600
		Default=none

12. You are implementing security best practices on your data pipeline. Currently, you are manually executing jobs as the Project Owner. You want to automate these jobs by taking nightly batch files containing non-public information from Google Cloud Storage, processing them with a Spark Scala job on a Google Cloud Dataproc cluster, and depositing the results into Google BigQuery. How should you securely run this workload?

A. Restrict the Google Cloud Storage bucket so only you can see the files
B. Grant the Project Owner role to a service account, and run the job with it
C. Use a service account with the ability to read the batch files and to write to BigQuery

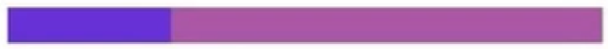
- **Service Account:** Using a service account eliminates the need for a user account with potentially broader permissions. Service accounts are designed for applications and services to access Google Cloud resources.
- **Least Privilege:** Grant the service account only the specific permissions required for the data pipeline tasks:
  - Read access to the Cloud Storage bucket containing the batch files.
  - Write access to BigQuery for depositing the processed results.

D. Use a user account with the Project Viewer role on the Cloud Dataproc cluster to read the batch files and write to BigQuery

13. You are using Google BigQuery as your data warehouse. Your users report that the following simple query is running very slowly, no matter when they run the query:

```
SELECT country, state, city FROM [myproject:mydataset.mytablel GROUP BY country;
```

You check the query plan for the query and see the following output in the Read section of Stage:1:



What is the most likely cause of the delay for this query?

A. Users are running too many concurrent queries in the system

B. The `[myproject:mydataset.mytable]` table has too many partitions

C. Either the state or the city columns in the `[myproject:mydataset.mytable]` table have too many NULL values

D. Most rows in the `[myproject:mydataset.mytablel]` table have the same value in the country column, causing data skew

14. Your globally distributed auction application allows users to bid on items. Occasionally, users place identical bids at nearly identical times, and different application servers process those bids. Each bid event contains the item, amount, user, and timestamp. You want to collate those bid events into a single location in real time to determine which user bid first. What should you do?

A. Create a file on a shared file and have the application servers write all bid events to that file. Process the file with Apache Hadoop to identify which user bid first.

B. Have each application server write the bid events to Cloud Pub/Sub as they occur. Push the events from Cloud Pub/Sub to a custom endpoint that writes the bid event information into Cloud SQL. - CORRECT

C. Set up a MySQL database for each application server to write bid events into. Periodically query each of those distributed MySQL databases and update a master MySQL database with bid event information.

D. Have each application server write the bid events to Google Cloud Pub/Sub as they occur. Use a pull subscription to pull the bid events using Google Cloud Dataflow. Give the bid for each item to the user in the bid event that is processed first.

15. Your organization has been collecting and analyzing data in Google BigQuery for 6 months. The majority of the data analyzed is placed in a time-partitioned table named events_partitioned. To reduce the cost of queries, your organization created a view called events, which queries only the last 14 days of data. The view is described in legacy SQL. Next month, existing

applications will be connecting to BigQuery to read the events data via an ODBC connection. You need to ensure the applications can connect. Which two actions should you take? (Choose two.)

A. Create a new view over events using standard SQL
B. Create a new partitioned table using a standard SQL query
C. Create a new view over events_partitioned using standard SQL

- **ODBC Driver Compatibility:** If the ODBC drivers used by existing applications don't support legacy SQL views, then creating a new view using standard SQL over the `events_partitioned` table would ensure compatibility. This new view could then be granted access to the ODBC connection's service account.
- **Alternative to Modifying Existing View:** This approach avoids modifying the existing `events` view (which might be used by other systems). It creates a separate view specifically for ODBC access.

  D. Create a service account for the ODBC connection to use for authentication
- **Secure Authentication:** Regardless of the view used (legacy or standard SQL), the ODBC connection needs secure credentials to access BigQuery. A service account provides those credentials.

  E. Create a Google Cloud Identity and Access Management (Cloud IAM) role for the ODBC connection and shared events.

16. You have enabled the free integration between Firebase Analytics and Google BigQuery. Firebase now automatically creates a new table daily in BigQuery in the format app_events_YYYYMMDD. You want to query all of the tables for the past 30 days in legacy SQL. What should you do?

A. Use the function `TABLE_DATE_RANGE` function
https://cloud.google.com/bigquery/docs/reference/legacy-sql#table-date-range

```
SELECT *
FROM `TABLE_DATE_RANGE`([your-project-id.your-dataset.app_events_],
                         TIMESTAMP('YYYY-MM-DD'), TIMESTAMP('YYYY-MM-DD')) AS
d
```

B. Use the WHERE_PARTITIONTIME pseudo column

```
SELECT *
FROM `your-project-id.your-dataset.app_events_*`
WHERE_PARTITIONTIME >= DATE_SUB(CURRENT_DATE(), INTERVAL 30 DAY);
```

C. Use WHERE date BETWEEN YYYY.MM.DD AND YYYY-MM-DD
D. Use SELECT IF.(date YYYY-MM-DD AND date YYYY-MM-DD

17. Your company is currently setting up data pipelines for their campaign. For all the Google Cloud Pub/Sub streaming data, one Of the important business requirements is to be able to periodically identify the inputs and their timings during their campaign. Engineers have decided to use windowing and transformation in Google Cloud Dataflow for this purpose. However, when testing this feature, they find that the Cloud Dataflow job fails for the all streaming insert. What is the most likely cause of this problem?

A. They have not assigned the timestamp, which causes the job to fail
B. They have not set the triggers to accommodate the data coming in late, which causes the job to fail
C. They have not applied a global windowing function, which causes the job to fail when the pipeline is created
D. They have not applied a non-global windowing function, which causes the job to fail when the pipeline is created

- **Windowing for Stream Processing:** Windowing is a crucial concept in Dataflow for stream processing. It groups data elements arriving over a specific time interval for further processing and analysis.
- **Global Windowing vs. Non-Global:** A global windowing function assigns all elements to a single window, which might not be ideal for processing data arriving over time intervals. On the other hand, non-global windowing functions like fixed, sliding, or session windows allow grouping data based on time intervals.
- **Requirement for Non-Global Windows:** Since the business requirement involves periodically identifying inputs at specific times during the campaign, non-global windowing functions are necessary to group data based on time intervals.

18. You architect a system to analyze seismic data. Your extract, transform, and load (ETL) process runs as a series of MapReduce jobs on an Apache Hadoop cluster. The ETL process takes days to process a data set because some steps are computationally expensive. Then you discover that a sensor calibration step has been omitted. How should you change your ETL process to carry out sensor calibration systematically in the future?

A. Modify the transformMapReduce jobs to apply sensor calibration before they do anything else.
B. Introduce a new MapReduce job to apply sensor calibration to raw data, and ensure all

other MapReduce jobs are chained after this.
C. Add sensor calibration data to the output of the ETL process, and document that all users need to apply sensor calibration themselves.
D. Develop an algorithm through simulation to predict variance of data output from the last MapReduce job based on calibration factors, and apply the correction to all data.
- innovative but it adds unnecessary complexity

19. An online retailer has built their current application on Google App Engine. A new initiative at the company mandates that they extend their application to allow their customers to transact directly via the application. They need to manage their shopping transactions and analyze combined data from multiple datasets using a business intelligence (BI) tool. They want to use only a single database for this purpose. Which Google Cloud database should they choose?

A. BigQuery
B. Cloud SQL

- **Transaction Management:** Cloud SQL offers relational database options like MySQL and PostgreSQL, which are well-suited for managing online transactions like shopping carts and order processing. These databases support features like ACID (Atomicity, Consistency, Isolation, Durability) transactions, ensuring data integrity during purchases.
- **BI Integration:** Cloud SQL integrates seamlessly with various BI tools due to its support for standard SQL queries. The online retailer can connect their BI tool of choice to Cloud SQL and analyze combined data from multiple datasets within the database.
  C. Cloud BigTable
  D. Cloud Datastore

20. You launched a new gaming app almost three years ago. You have been uploading log files from the previous day to a separate Google BigQuery table with the table name format LOGS_yyyymmdd. You have been using table wildcard functions to generate daily and monthly reports for all time ranges. Recently, you discovered that some queries that cover long date ranges are exceeding the limit of 1,000 tables and failing. How can you resolve this issue?

A. Convert all daily log tables into date-partitioned tables
B. Convert the sharded tables into a single partitioned table
C. Enable query caching so you can cache data from previous months
D. Create separate views to cover each month, and query from these views

21. Your analytics team wants to build a simple statistical model to determine which customers are most likely to work with your company again, based on a few different metrics. They want to run the model on Apache Spark, using data housed in Google Cloud Storage, and you have recommended using Google Cloud Dataproc to execute this job. Testing has shown that this workload can run in approximately 30 minutes on a 15-node cluster, outputting the results into Google BigQuery. The plan is to run this workload weekly. How should you optimize the cluster for cost?

A. Migrate the workload to Google Cloud Dataflow

B. Use pre-emptible virtual machines (VMs) for the cluster

- **Workload Characteristics:** The job runs weekly and takes approximately 30 minutes on a 15-node cluster, indicating it's not a continuous or time-sensitive process.
- **Pre-emptible VMs:** These virtual machines offer significant cost savings compared to sustained use VMs. Google Cloud can reclaim pre-emptible VMs when needed for other tasks, but with proper notification for job interruption. Since the job runs for a short duration and can be retried in case of interruption, pre-emptible VMs are a good fit.

  C. Use a higher-memory node so that the job runs faster

  D. Use SSDs on the worker nodes so that the job can run faster

22. Your company receives both batch- and stream-based event data. You want to process the data using Google Cloud Dataflow over a predictable time period. However, you realize that in some instances data can arrive late or out of order. How should you design your Cloud Dataflow pipeline to handle data that is late or out of order?

A. Set a single global window to capture all the data.

B. Set sliding windows to capture all the lagged data.

C. Use watermarks and timestamps to capture the lagged data.

- **Watermarks and Timestamps:** Dataflow relies on watermarks (monotonically increasing timestamps) to indicate the progress of data within a window. It uses timestamps associated with each data element to determine its processing order. This allows Dataflow to handle late or out-of-order data effectively.
- **Windowing:** Dataflow processes data in windows based on timestamps. Late data arriving after a window closes can still be processed if it falls within a specific timeframe defined by the watermark.

  D. Ensure every datasource type (stream or batch) has a timestamp, and use the timestamps to define the logic for lagged data.

23. Your company receives both batch- and stream-based event data. You want to process the data using Google Cloud Dataflow over a predictable time period. However, you realize that in some instances data can arrive late or out of order. How should you design your Cloud Dataflow pipeline to handle data that is late or out of order?

A. Set a single global window to capture all the data.
B. Set sliding windows to capture all the lagged data.
C. Use watermarks and timestamps to capture the lagged data.

- **Dataflow Processing Paradigm:** Cloud Dataflow excels at processing both batch and stream data in a unified manner. It leverages windowing to group data for processing based on timestamps associated with each event.
- **Watermarks and Timestamps:**
  - **Watermarks:** These are monotonically increasing timestamps acting as signals for Dataflow to estimate the arrival of all data within a specific window. Watermarks help determine when to consider a window closed for processing, even if some late data might still arrive.
  - **Timestamps:** Each data element should have a timestamp attached, indicating the actual event time. This timestamp is crucial for Dataflow to order and group data correctly within windows.
- **Late Data Handling:** By combining watermarks and timestamps, Dataflow can effectively identify and process late data that arrives after a window has closed. The specific behavior depends on how you configure your pipeline with windowing options and watermark hold times.

D. Ensure every datasource type (stream or batch) has a timestamp, and use the timestamps to define the logic for lagged data.

24. You have some data, which is shown in the graphic below. The two dimensions are X and Y, and the shade of each dot represents what class it is. You want to classify this data accurately using a linear algorithm. To do this you need to add a synthetic feature. What should the value of that feature be?

A. X^2 + Y^2
B. X^2
C. Y^2
D. cos(X)

25. You are integrating one of your internal IT applications and Google BigQuery, so users can query BigQuery from the application's interface. You do not want individual users to authenticate to BigQuery and you do not

want to give them access to the dataset. You need to securely access BigQuery from your IT application. What should you do?

A. Create groups for your users and give those groups access to the dataset
B. Integrate with a single sign-on (SSO) platform, and pass each user's credentials along with the query request

- **(security risk):** Passing individual user credentials via SSO exposes them within the application code and potentially increases the attack surface. If compromised, malicious actors could gain access to user accounts.
  C. Create a service account and grant dataset access to that account. Use the service account's private key to access the dataset
- **Security Best Practices:** Service accounts are designed specifically for application-to-application authentication in Google Cloud Platform (GCP). They eliminate the need for individual user credentials and associated security risks.
- **Granular Access Control:** You can grant the service account access to specific BigQuery datasets or tables, ensuring least privilege for the application. This minimizes the potential damage if the service account credentials are compromised.
- **Secure Key Management:** The service account's private key should be securely stored and managed using GCP's Key Management Service (KMS). This avoids storing it directly on the application's file system, which is a vulnerable practice.
  D. Create a dummy user and grant dataset access to that user. Store the username and password for that user in a file on the file system, and use those credentials to access the BigQuery dataset

26. You are building a data pipeline on Google Cloud. You need to prepare data using a casual method for a machine-learning process. You want to support a logistic regression model. You also need to monitor and adjust for null values, which must remain real-valued and cannot be removed. What should you do?

A. Use Cloud Dataprep to find null values in sample source data. Convert all nulls to 'none' using a Cloud Dataproc job.
B. Use Cloud Dataprep to find null values in sample source data. Convert all nulls to 0 using a Cloud Dataprep job.

- **Cloud Dataprep:** This is a Google Cloud service specifically designed for visual data wrangling and data preparation. It's ideal for finding null values in your sample data.
- **Convert nulls to 0:** Since you cannot remove null values and they need to remain numerical, replacing them with 0 is a common practice for logistic regression models. This simplifies calculations and avoids errors.

- **Cloud Dataproc (not recommended):** While Cloud Dataproc is a managed Hadoop and Spark service, using Cloud Dataprep is a more efficient and user-friendly way to handle simple data transformations like null value replacement in this case.
- **Cloud Dataflow (not recommended):** Cloud Dataflow is a streaming data processing service. It's better suited for large-scale data pipelines, not for a one-time data preparation task on a sample.
- **Custom script (not recommended):** Writing a custom script can be time-consuming and requires programming knowledge. Cloud Dataprep provides a visual interface for this specific task.

  C. Use Cloud Dataflow to find null values in sample source data. Convert all nulls to 'none' using a Cloud Dataprep job.

  D. Use Cloud Dataflow to find null values in sample source data. Convert all nulls to 0 using a custom script.

27. You set up a streaming data insert into a Redis cluster via a Kafka cluster. Both clusters are running on Compute Engine instances. You need to encrypt data at rest with encryption keys that you can create, rotate, and destroy as needed. What should you do?

A) Create a dedicated service account, and use encryption at rest to reference your data stored in your Compute Engine cluster instances as part of your API service calls.

B) Create encryption keys in Cloud Key Management Service. Use those keys to encrypt your data in all of the Compute Engine cluster instances.

- **Cloud Key Management Service (KMS):** This is a dedicated service on Google Cloud Platform for managing encryption keys. It provides secure creation, rotation, and destruction of keys, which is exactly what you need.
- **Encrypting Data at Rest:** This ensures your data is protected even if someone gains unauthorized access to the underlying storage of your Compute Engine instances.
- **Dedicated Keys:** Using Cloud KMS allows you to create and manage your own keys, giving you full control over their lifecycle.

C) Create encryption keys locally. Upload your encryption keys to Cloud Key Management Service. Use those keys to encrypt your data in all of the Compute Engine cluster instances.

D) Create encryption keys in Cloud Key Management Service. Reference those keys in your API service calls when accessing the data in your Compute Engine cluster instances.

28. You are developing an application that uses a recommendation engine on Google Cloud. Your solution should display new videos to customers based on past views. Your solution needs to generate labels for the entities in videos that the customer has viewed. Your design must be able to provide very fast

filtering suggestions based on data from other customer preferences on several TB of data. What should you do?

A) Build and train a complex classification model with Spark MLIib to generate labels and filter the results. Deploy the models using Cloud Dataproc. Call the model from your application.x

B) Build and train a classification model with Spark MLIib to generate labels. Build and train a second classification model with Spark MLIib to filter results to match customer preferences. Deploy the models using Cloud Dataproc. Call the models from your application.

C) Build an application that calls the Cloud Video Intelligence API to generate labels. Store data in Cloud Bigtable, and filter the predicted labels to match the user's viewing history to generate preferences.

- **Cloud Video Intelligence API:** This pre-built service by Google Cloud can analyze videos and automatically generate labels for entities within them. This eliminates the need to build and train your own complex classification model.
- **Cloud Bigtable:** This NoSQL database service is perfect for storing large datasets (several TBs) with low latency and high throughput. It's ideal for fast filtering based on customer preferences.
- **Filtering Predicted Labels:** By storing video labels and user viewing history in Bigtable, you can efficiently filter the labels based on a user's past views, allowing for personalized recommendations.
  D) Build an application that calls the Cloud Video Intelligence API to generate labels. Store data in Cloud SQL, and join and filter the predicted labels to match the user's viewing history to generate preferences.

29. You are selecting services to write and transform JSON messages from Cloud Pub/Sub to BigQuery for a data pipeline on Google Cloud. You want to minimize service costs. You also want to monitor and accommodate input data volume that will vary in size with minimal manual intervention. What should you do?

A) Use Cloud Dataproc to run your transformations. Monitor CPU utilization for the cluster. Resize the number of worker nodes in your cluster via the command line.

B) Use Cloud Dataproc to run your transformations. Use the diagnose command to generate an operational output archive. Locate the bottleneck and adjust cluster resources.

C) Use Cloud Dataflow to run your transformations. Monitor the job system lag with Stackdriver. Use the default autoscaling setting for worker instances.

- **Cloud Dataflow:** This is a managed service for building data pipelines. It offers built-in features that are ideal for your scenario:

    - **Cost-effective:** Cloud Dataflow automatically scales resources based on the incoming data volume. This eliminates the need for manual cluster management like in Cloud Dataproc (options A & B).
    - **Autoscaling:** The default autoscaling setting automatically adjusts the number of worker nodes based on the workload, ensuring efficient resource utilization.
- **Stackdriver Monitoring:** Monitoring job system lag with Stackdriver provides insights into pipeline performance. If lag increases, it might indicate insufficient resources, prompting you to investigate further (potentially changing machine types in the future).
  D) Use Cloud Dataflow to run your transformations. Monitor the total execution time for a sampling of jobs. Configure the job to use non-default Compute Engine machine types when needed.

30. Your infrastructure includes a set of YouTube channels. You have been tasked with creating a process for sending the YouTube channel data to Google Cloud for analysis. You want to design a solution that allows your world-wide marketing teams to perform ANSI SQL and other types of analysis on up-to-date YouTube channels log data. How should you set up the log data transfer into Google Cloud?

A) Use Storage Transfer Service to transfer the offsite backup files to a Cloud Storage Multi-Regional storage bucket as a final destination.

B) Use Storage Transfer Service to transfer the offsite backup files to a Cloud Storage Regional bucket as a final destination.

C) Use BigQuery Data Transfer Service to transfer the offsite backup files to a Cloud Storage Multi-Regional storage bucket as a final destination.
- HA
- Higher Cost, the option D
D) Use BigQuery Data Transfer Service to transfer the offsite backup files to a Cloud Storage Regional storage bucket as a final destination.

31 You are designing storage for very large text files for a data pipeline on Google Cloud. You want to support ANSI SQL queries. You also want to support compression and parallel load from the input locations using Google recommended practices. What should you do?

A) Transform text files to compressed Avro using Cloud Dataflow. Use BigQuery for storage and query.
- BQ is a WH, not designed to store large textfiles.
B) Transform text files to compressed Avro using Cloud Dataflow. Use Cloud Storage and BigQuery permanent linked tables for query.

- **Cloud Dataflow:** This managed service is ideal for transforming large datasets like your text files. You can use it to convert them to a compressed format like Avro, which offers better efficiency compared to gzip.
- **Cloud Storage:** This object storage service is perfect for storing large text files in a cost-effective and scalable manner.
- **BigQuery Permanent Linked Tables:** This feature allows you to link BigQuery tables to data in Cloud Storage without physically moving the data. This facilitates querying the Avro files directly from BigQuery using ANSI SQL.
- **Compression and Parallel Load:** Avro format supports compression, and BigQuery allows parallel loading from Cloud Storage for faster data ingestion.

C) Compress text files to gzip using the Grid Computing Tools. Use BigQuery for storage and query.
- Grid computing is deprecated
D) Compress text files to gzip using the Grid Computing Tools. Use Cloud Storage, and then import into Cloud Bigtable for query.

32. You are developing an application on Google Cloud that will automatically generate subject labels for users' blog posts. You are under competitive pressure to add this feature quickly, and you have no additional developer resources. No one on your team has experience with machine learning. What should you do?

A) Call the Cloud Natural Language API from your application. Process the generated Entity Analysis as labels.

- **Competitive Pressure & Limited Resources:** Cloud Natural Language API is a pre-built, managed service. This eliminates the need for your team (with no machine learning experience) to build and train a custom model from scratch (options C & D), saving significant development time.
- **Entity Analysis for Subject Labels:** Entity Analysis identifies key concepts within the blog post. This extracted information can be a good starting point for generating subject labels, especially for simple use cases.

B) Call the Cloud Natural Language API from your application. Process the generated Sentiment Analysis as labels.

C) Build and train a text classification model using TensorFlow. Deploy the model using Cloud Machine Learning Engine. Call the model from your application and process the results as labels.

D) Build and train a text classification model using TensorFlow. Deploy the model using a Kubernetes Engine cluster. Call the model from your application and process the results as labels.

33. You are designing storage for 20 TB of text files as part of deploying a data pipeline on Google Cloud. Your input data is in CSV format. You want to minimize the cost of querying aggregate values for multiple users who will query the data in Cloud Storage with multiple engines. Which storage service and schema design should you use?

A) Use Cloud Bigtable for storage. Install the HBase shell on a Compute Engine instance to query the Cloud Bigtable data.

B) Use Cloud Bigtable for storage. Link as permanent tables in BigQuery for query.

C) Use Cloud Storage for storage. Link as permanent tables in BigQuery for query.

- **Cost-Effectiveness:** Cloud Storage is a cost-effective solution for storing large amounts of data like your 20 TB of CSV files.
- **BigQuery for Aggregate Queries:** BigQuery is a serverless data warehouse specifically designed for analyzing large datasets. It's optimized for running aggregate queries efficiently, which is your primary requirement.
- **Permanent Tables:** Linking Cloud Storage files as permanent tables in BigQuery creates a managed view of the data. Users can query the data directly in BigQuery using ANSI SQL or other supported languages, without needing to manage the underlying files in Cloud Storage. This simplifies access and avoids data movement costs.

D) Use Cloud Storage for storage. Link as temporary tables in BigQuery for query.

34. 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 non-key columns. What should you do?

A) Use Cloud SQL for storage. Add secondary indexes to support query patterns.
B) Use Cloud SQL for storage. Use Cloud Dataflow to transform data to support query patterns.
C) Use Cloud Spanner for storage. Add secondary indexes to support query patterns.

- **Cloud Spanner:** This is a globally distributed relational database service designed for large-scale applications. It offers horizontal scaling, which means you can easily add more nodes to handle increasing data volume and transaction demands.
- **Secondary Indexes:** Cloud Spanner supports secondary indexes on non-key columns. This can significantly improve the performance of range queries on those columns without modifying the original table structure.
  D) Use Cloud Spanner for storage. Use Cloud Dataflow to transform data to support query patterns.

35. 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?
A. Cloud Bigtable

- **Scalability and Performance:** Cloud Bigtable is a NoSQL database designed for handling massive datasets with low latency and high throughput. This makes it ideal for storing and managing your 50 TB of frequently updated financial time-series data.
- **Time-Series Data:** Cloud Bigtable is well-suited for time-series data because it uses a columnar data model with timestamps as a key component. This allows for efficient storage and retrieval of time-series data points.
- **Apache Hadoop Integration:** Cloud Bigtable integrates well with Apache Hadoop and Spark, which your company already uses. This allows you to leverage existing tools and expertise to gain insights from the data stored in Cloud Bigtable.
  B. Google BigQuery
  C. Google Cloud Storage
  D. Google Cloud Datastore

36. An organization maintains a Google BigQuery dataset that contains tables with user-level data. They want to expose aggregates of this data to other Google Cloud projects, while still controlling access to the user-level data. Additionally, they need to minimize their overall storage cost and ensure the analysis cost for other projects is assigned to those projects. What should they do?

A) Create and share an authorized view that provides the aggregate results.
https://cloud.google.com/bigquery/docs/authorized-views
B) Create and share a new dataset and view that provides the aggregate results.

C) Create and share a new dataset and table that contains the aggregate results.

D) Create dataViewer Identity and Access Management (IAM) roles on the dataset to enable sharing.

37. Government regulations in your industry mandate that you have to maintain an auditable record of access to certain types of data. Assuming that all expiring logs will be archived correctly, where should you store data that is subject to that mandate?

A) Encrypted on Cloud Storage with user-supplied encryption keys. A separate decryption key will be given to each authorized user.

B) In a BigQuery dataset that is viewable only by authorized personnel, with the Data Access log used to provide the auditability.

- **BigQuery Access Control:** BigQuery allows granular access control through IAM permissions. You can restrict access to the dataset containing the sensitive data to only authorized personnel.
- **Data Access Logs:** BigQuery automatically logs all data access attempts, including successful queries and denied access attempts. These logs provide a detailed audit trail of who accessed the data and when, fulfilling the auditability requirement.
  C) In Cloud SQL, with separate database user names to each user. The Cloud SQL Admin activity logs will be used to provide the auditability.

D) In a bucket on Cloud Storage that is accessible only by an AppEngine service that collects user information and logs the access before providing a link to the bucket.

38. Your neural network model is taking days to train. You want to increase the training speed. What can you do?

A. Subsample your test dataset.
B. Subsample your training dataset.
C. Increase the number of input features to your model.
D. Increase the number of layers in your neural network.

39. You are responsible for writing your company's ETL pipelines to run on an Apache Hadoop cluster. The pipeline will require some checkpointing and splitting pipelines. Which method should you use to write the pipelines?

A) PigLatin using Pig

B) HiveQL using Hive

C) Java using MapReduce

D) Python using MapReduce

40. Your company maintains a hybrid deployment with GCP, where analytics are performed on your anonymized customer data. The data are imported to Cloud Storage from your data center through parallel uploads to a data transfer server running on GCP. Management informs you that the daily transfers take too long and have asked you to fix the problem. You want to maximize transfer speeds. Which action should you take?

A) Increase the CPU size on your server.
B) Increase the size of the Google Persistent Disk on your server.
C) Increase your network bandwidth from your datacenter to GCP.

- **Data Transfer Bottleneck:** The problem statement mentions slow daily data transfers from your data center to Cloud Storage. This suggests a network bandwidth limitation, not an issue with CPU or storage capacity on your server.
- **Parallel Uploads:** While parallel uploads can improve efficiency, they still rely on the available network bandwidth. Increasing the bandwidth allows for more data to be transferred simultaneously, potentially leading to significant speed improvements.

D) Increase your network bandwidth from Compute Engine to Cloud Storage.

41. CFO Statement -

The project is too large for us to maintain the hardware and software required for the data and analysis. Also, we cannot afford to staff an operations team to monitor so many data feeds, so we will rely on automation and infrastructure. Google Cloud's machine learning will allow our quantitative researchers to work on our high-value problems instead of problems with our data pipelines.

MJTelco is building a custom interface to share data. They have these requirements:

1. They need to do aggregations over their petabyte-scale datasets.
2. They need to scan specific time range rows with a very fast response time (milliseconds). Which combination of Google Cloud Platform products should you recommend?

A) Cloud Datastore and Cloud Bigtable

B) Cloud Bigtable and Cloud SQL

C) BigQuery and Cloud Bigtable

- **CFO Statement:**
  - **Hardware/Software Maintenance:** BigQuery and Cloud Bigtable are managed services, eliminating the need for MJTelco to maintain physical hardware or software licenses.
  - **Staffing for Operations:** Both services are automated, reducing the need for a dedicated team to monitor data pipelines. Machine learning in BigQuery can further automate data analysis tasks.
  - **Focus on High-Value Problems:** Freeing up quantitative researchers from data pipeline issues allows them to focus on more strategic data analysis.
- **MJTelco's Requirements:**
  - **Petabyte-Scale Aggregations:** BigQuery is a serverless data warehouse specifically designed for efficient analysis of large datasets. It can handle aggregations on petabyte-scale data.

- **Fast Time Range Scans:** Cloud Bigtable is a NoSQL database optimized for low-latency reads and writes. It's ideal for scanning specific time ranges with millisecond response times.

D) BigQuery and Cloud Storage

42. CFO Statement -

The project is too large for us to maintain the hardware and software required for the data and analysis. Also, we cannot afford to staff an operations team to monitor so many data feeds, so we will rely on automation and infrastructure. Google Cloud's machine learning will allow our quantitative researchers to work on our high-value problems instead of problems with our data pipelines.

You need to compose visualization for operations teams with the following requirements:

- Telemetry must include data from all 50,000 installations for the most recent 6 weeks (sampling once every minute)
- The report must not be more than 3 hours delayed from live data.
- The actionable report should only show suboptimal links.
- Most suboptimal links should be sorted to the top.
- Suboptimal links can be grouped and filtered by regional geography.
- User response time to load the report must be <5 seconds.

You create a data source to store the last 6 weeks of data, and create visualizations that allow viewers to see multiple date ranges, distinct geographic regions, and unique installation types. You always show the latest data without any changes to your visualizations. You want to avoid creating and updating new visualizations each month. What should you do?

A) Look through the current data and compose a series of charts and tables, one for each possible combination of criteria.

B) Look through the current data and compose a small set of generalized charts and tables bound to criteria filters that allow value selection.

C) Export the data to a spreadsheet, compose a series of charts and tables, one for each possible combination of criteria, and spread them across multiple tabs.

D) Load the data into relational database tables, write a Google App Engine application that queries all rows, summarizes the data across each criteria, and then renders results using the Google Charts and visualization API.

43. CFO Statement -

The project is too large for us to maintain the hardware and software required for the data and analysis. Also, we cannot afford to staff an operations team to monitor so many data feeds, so we will rely on automation and infrastructure. Google Cloud's machine learning will allow our quantitative researchers to work on our high-value problems instead of problems with our data pipelines.

Given the record streams MJTelco is interested in ingesting per day, they are concerned about the cost of Google BigQuery increasing. MJTelco asks you to provide a design solution. They require a single large data table called tracking_table. Additionally, they want to minimize the cost of daily queries while performing fine-grained analysis of each day's events. They also want to use streaming ingestion. What should you do?

A) Create a table called tracking_table and include a DATE column.

B) Create a partitioned table called tracking_table and include a TIMESTAMP column.

C) Create sharded tables for each day following the pattern.

D) Create a table called tracking_table with a TIMESTAMP column to represent the day.

44. CFO Statement -

Part of our competitive advantage is that we penalize ourselves for late shipments and deliveries. Knowing where out shipments are at all times has a direct correlation to our bottom line and profitability. Additionally, I dont want to commit capital to building out a server environment.

Flowlogistic's management has determined that the current Apache Kafka servers cannot handle the data volume for their real-time inventory tracking system. You need to build a new system on Google Cloud Platform (GCP) that will feed the proprietary tracking software. The system must be able to ingest data from a variety of global sources, process and query in real-time, and store the data reliably. Which combination of GCP products should you choose?

A) Cloud Pub/Sub, Cloud Dataflow, and Cloud Storage

B) Cloud Pub/Sub, Cloud Dataflow, and Local SSD

C) Cloud Pub/Sub, Cloud SQL, and Cloud Storage

D) Cloud Load Balancing, Cloud Dataflow, and Cloud Storage

E) Cloud Dataflow, Cloud SQL, and Cloud Storage

45. After migrating ETL jobs to run on BigQuery, you need to verify that the output of the migrated jobs is the same as the output of the original. You've loaded a table containing the output of the original job and want to compare the contents with output from the migrated job to show that they are identical. The tables do not contain a primary key column that would enable you to join them together for comparison.

What should you do?

A) Select random samples from the tables using the RAND() function and compare the samples.

B) Select random samples from the tables using the HASH() function and compare the samples.

C) Use a Dataproc cluster and the BigQuery Hadoop connector to read the data from each table and calculate a hash from non-timestamp columns of the table after sorting. Compare the hashes of each table.

- **Hashing for Integrity:** Calculating a hash of the table data provides a fingerprint of the content. If the original and migrated data are identical, their hashes will be the same.
- **Sorting Before Hashing:** Sorting the data ensures consistent ordering before calculating the hash. This is important because the order of rows might not be the same in both tables, even if the content is identical. Excluding timestamps from the hash calculation further avoids inconsistencies due to potential time differences between job executions.
- **Dataproc and BigQuery Connector:** Dataproc is a managed Spark cluster service on Google Cloud. The BigQuery Hadoop connector allows you to read data from BigQuery tables using Spark jobs running on Dataproc. This enables the processing power of Spark for data manipulation and hash calculation.
  D) Create stratified random samples using the OVER() function and compare equivalent samples from each table.

46. You are a head of BI at a large enterprise company with multiple business units that each have different priorities and budgets. You use on-demand pricing for BigQuery with a quota of 2K concurrent on-demand slots per project. Users at your organization sometimes don't get slots to execute their queries and you need to correct this. You'd like to avoid introducing new projects to your account. What should you do?

A) Convert your batch BQ queries into interactive BQ queries.

B) Create an additional project to overcome the 2K on-demand per-project quota.

C) Switch to flat-rate pricing and establish a hierarchical priority model for your projects.

D) Increase the amount of concurrent slots per project at the Quotas page at the Cloud Console.

- **C) Flat-Rate Pricing with Priority (Consider, But Not Best):** Flat-rate pricing removes per-slot quotas but comes with a fixed monthly cost, which might be risky if usage varies significantly. Establishing a priority model can help, but managing priorities across multiple business units can be challenging.

- **D) Increase On-Demand Slots (Best Option):** This directly addresses the issue by requesting a higher quota for concurrent on-demand slots within your existing project.

47. You have an Apache Kafka cluster on-prem with topics containing web application logs. You need to replicate the data to Google Cloud for analysis in BigQuery and Cloud Storage. The preferred replication method is mirroring to avoid deployment of Kafka Connect plugins. What should you do?

A) Deploy a Kafka cluster on GCE VM Instances. Configure your on-prem cluster to mirror your topics to the cluster running in GCE. Use a Dataproc cluster or Dataflow job to read from Kafka and write to GCS.

B) Deploy a Kafka cluster on GCE VM Instances with the Pub/Sub Kafka connector configured as a Sink connector. Use a Dataproc cluster or Dataflow job to read from Kafka and write to GCS.

C) Deploy the Pub/Sub Kafka connector to your on-prem Kafka cluster and configure Pub/Sub as a Source connector. Use a Dataflow job to read from Pub/Sub and write to GCS.

D) Deploy the Pub/Sub Kafka connector to your on-prem Kafka cluster and configure Pub/Sub as a Sink connector. Use a Dataflow job to read from Pub/Sub and write to GCS.

- **Pros:**
  - Mirroring functionality: Deploying a Kafka cluster on GCE allows you to configure mirroring between your on-premise topics and the GCE topics. This fulfills the mirroring requirement.
  - Potential Performance Benefits: Depending on your network bandwidth and data volume, a dedicated Kafka cluster on GCE might offer better performance compared to relying on Pub/Sub for replication.
- **Cons:**
  - Increased Complexity: Managing an additional Kafka cluster adds complexity to your infrastructure.
  - Separate Data Pipeline: A separate data pipeline (Dataproc or Dataflow) is needed to read from the GCE Kafka cluster and write to GCS. This can introduce additional development and maintenance overhead.

Option D: Pub/Sub Kafka Connector as Sink

- **Pros:**
  - Simplicity: Leverages existing Kafka infrastructure and avoids deploying Kafka Connect plugins, aligning with your preference.
  - Centralized Management: Data flows through Pub/Sub, a managed service, simplifying management compared to a separate Kafka cluster.

- **Cons:**
  - Potential Performance Considerations: Depending on your data volume and network latency, Pub/Sub might introduce some overhead compared to a dedicated Kafka mirroring setup (option A).

48. You've migrated a Hadoop job from an on-prem cluster to Dataproc and GCS. Your Spark job is a complicated analytical workload that consists of many shuffling operations and initial data are Parquet files (on average 200-400 MB size each). You see some degradation in performance after the migration to Dataproc, so you'd like to optimize for it. You need to keep in mind that your organization is very cost-sensitive, so you'd like to continue using Dataproc on preemptibles (with 2 non-preemptible workers only) for this workload. What should you do?

A) Increase the size of your Parquet files to ensure they are 1 GB minimum.

B) Switch to TFRecords formats (approximately 200MB per file) instead of Parquet files.

C) Switch from HDDs to SSDs, copy initial data from GCS to HDFS, run the Spark job and copy results back to GCS.

D) Switch from HDDs to SSDs, override the preemptible VMs configuration to increase the boot disk size.

- **SSDs:** Upgrading to SSDs on preemptible VMs can significantly improve performance for shuffle-heavy workloads like yours by providing faster read/write speeds for shuffle data.
- **Larger Boot Disk:** Preemptible VMs have smaller boot disks by default. Increasing the boot disk size ensures there's enough local storage space for shuffle data on the worker nodes, reducing the need to spill data to slower HDDs. This optimization leverages the cost-effectiveness of preemptibles while improving performance.

49. Your team is responsible for developing and maintaining ETLs in your company. One of your Dataflow jobs is failing because of some errors in the input data, and you need to improve reliability of the pipeline (including being able to reprocess all failing data). What should you do?

A) Add a filtering step to skip these types of errors in the future, extract erroneous rows from logs.

B) Add a try-catch block to your DOFn that transforms the data, extract erroneous rows from logs.

C) Add a try-catch block to your DOFn that transforms the data, write erroneous rows to Pub/Sub directly from the DoFn.

D) Add a try-catch block to your DOFn that transforms the data, use a sideOutput to create a PCollection that can be stored to Pub/Sub later.

B) Feature Cross without Bucketing:

- **Pros:** Simpler implementation, captures the overall spatial relationship between latitude and longitude.
- **Cons:** Might not capture fine-grained location details if the dataset covers a large geographic area.

C) Bucketized Feature Cross with L1 Regularization:

- **Pros:**
  - Bucketing:** Captures location-based influence at a specific granularity (minute level in this case). This can be useful if specific areas have significant price differences.
  - L1 Regularization:** Encourages sparsity in the model, potentially leading to a more interpretable feature and reducing model complexity. This can be helpful if interpretability of the location's influence is desired.
- **Cons:**
  - More complex implementation compared to a simple feature cross.
  - Choosing the optimal bucketing granularity (minute level in this case) can be challenging and might require experimentation.
  - L1 regularization can lead to feature selection, potentially removing some location-based influence from the model.

Choosing Between B and C:

- If the dataset covers a large geographic area and capturing fine-grained location details is crucial, or if interpretability of the location's influence is desired, then consider bucketizing the feature cross with L1 regularization (option C).
- If simplicity and capturing the overall spatial relationship are priorities, a non-bucketed feature cross (option B) might be sufficient.

Ultimately, the best approach depends on the specific characteristics of your dataset, the desired level of detail for location influence, and the importance of model interpretability.

Here are some additional points to consider:

- Experiment with different bucketing granularities to see how it affects model performance.
- Explore alternative regularization techniques like L2 regularization, which might be more suitable depending on your dataset.

---

51 You are deploying MariaDB SQL databases on GCE VM Instances and need to configure monitoring and alerting. You want to collect metrics including network connections, disk I/O, and replication status from MariaDB with minimal development effort and use StackDriver for dashboards and alerts. What should you do?

A) Install the OpenCensus Agent and create a custom metric collection application with a StackDriver exporter.

B) Place the MariaDB instances in an Instance Group with a Health Check.

C) Install the StackDriver Logging Agent and configure fluentd in_tail plugin to read MariaDB logs.

D) Install the StackDriver Agent and configure the MySQL plugin.

- **Pre-built Integration:** Stackdriver Agent includes a built-in MySQL plugin specifically designed to collect performance metrics from MariaDB databases. This eliminates the need for custom development (option A).
- **Minimal Configuration:** The MySQL plugin simplifies setup. You can configure it to collect relevant metrics like network connections, disk I/O, and replication status with minimal effort.
- **Stackdriver Integration:** The collected metrics are automatically sent to Stackdriver for visualization and alerting. This aligns with your desire to leverage Stackdriver for dashboards and alerts.

52. You work for a bank. You have a labelled dataset that contains information on already granted loan applications and whether these applications have been defaulted. You have been asked to train a model to predict default rates for credit applicants. What should you do?

A) Increase the size of the dataset by collecting additional data.

B) Train a linear regression to predict a credit default risk score.

- **Feasibility:** Linear regression is a well-understood and relatively simple algorithm to implement. It's a good choice for initial model building, especially when you're working with a labeled dataset containing information on past loan applications and defaults.
- **Interpretability:** Linear regression models are interpretable, meaning you can understand which factors (features) in the dataset have the most significant influence on the predicted credit default risk score. This can be valuable for explaining model decisions and ensuring fairness.

  C) Remove the bias from the data and collect applications that have been declined loans.

D) Match loan applicants with their social profiles to enable feature engineering.

53. You need to migrate a 2TB relational database to Google Cloud Platform. You do not have the resources to significantly refactor the application that uses this database and cost to operate is of primary concern. Which service do you select for storing and serving your data?

A) Cloud Spanner

B) Cloud Bigtable

C) Cloud Firestore

D) Cloud SQL

- **Relational Database:** Cloud SQL is a managed relational database service that supports MySQL, PostgreSQL, and SQL Server. Since you have a 2TB relational database, Cloud SQL is a natural fit as it maintains the familiar relational structure for minimal refactoring of your application.
- **Cost-Effectiveness:** Cloud SQL offers various pricing tiers based on storage size, machine type, and region. You can choose a cost-effective option that meets your needs. While Cloud Spanner offers unlimited storage, it might be more expensive for a 2TB database compared to Cloud SQL's tiered pricing.
- **Scalability:** Cloud SQL offers automatic scaling capabilities, allowing you to adjust storage and compute resources based on your workload needs. This can help optimize costs.

54. You're using Bigtable for a real-time application, and you have a heavy load that is a mix of read and writes. You've recently identified an additional use case and need to perform an hourly analytical job to calculate certain statistics across the whole database. You need to ensure both the reliability of your production application as well as the analytical workload. What should you do?

A) Export Bigtable dump to GCS and run your analytical job on top of the exported files.

B) Add a second cluster to an existing instance with multi-cluster routing, use a live-traffic app profile for your regular workload and a batch-analytics profile for the analytics workload.

C) Add a second cluster to an existing instance with single-cluster routing, use a live-traffic app profile for your regular workload and a batch-analytics profile for the analytics workload.

D) Increase the size of your existing cluster twice and execute your analytics workload on your new resized cluster.

C:
When you use a single cluster to run a batch analytics job that performs numerous large reads alongside an application that performs a mix of reads and writes, the large batch job can slow things down for the application's users. With replication, you can use app profiles with single-cluster routing to route batch analytics jobs and application traffic to different clusters, so that batch jobs don't affect your applications' users.

Reference:
https://cloud.google.com/bigtable/docs/replication-overview#use-cases

55. You are designing an Apache Beam pipeline to enrich data from Cloud Pub/Sub with static reference data from BigQuery. The reference data is small enough to fit in memory on a single worker. The pipeline should write enriched results to BigQuery for analysis. Which job type and transforms should this pipeline use?

A) Batch job, PubSubIO, side-inputs

B) Streaming job, PubSubIO, JdbcIO

C) Streaming job, PubSubIO, BigQueryIO, side-inputs

- **Streaming Job:** Since you're dealing with data from Cloud Pub/Sub, a streaming job is suitable to process the data as it arrives.
- **PubSubIO:** This I/O connector allows your Beam pipeline to read data from your Cloud Pub/Sub topic.
- **BigQueryIO (for Reference Data):** This I/O connector enables your pipeline to access the static reference data stored in BigQuery.
- **Side-inputs:** This transform type allows you to bring the small, in-memory reference data from BigQuery as a side input to your main data stream from Pub/Sub. This avoids full table loads from BigQuery for each Pub/Sub message, improving efficiency.

D) Streaming job, PubSubIO, BigQueryIO

56. You have a data pipeline that writes data to Cloud Bigtable using well-designed row keys. You want to monitor your pipeline to determine when to increase the size of your Cloud Bigtable cluster. Which two actions can you take to accomplish this? (Choose two.)

A) Review Key Visualizer metrics. Increase the size of the Cloud Bigtable cluster when the Read pressure index is above 100.

B) Review Key Visualizer metrics. Increase the size of the Cloud Bigtable cluster when the Write pressure index is above 100.

C) Monitor the latency of write operations. Increase the size of the Cloud Bigtable cluster when there is a sustained increase in write latency.

- **C) Write Latency:** Increased write latency indicates that your cluster might be overloaded and struggling to keep up with the incoming write requests. Scaling the cluster can help reduce write latency and improve overall pipeline performance.
- **D) Storage Utilization:** Monitoring storage utilization helps you anticipate future needs. While Bigtable can automatically adjust to some extent, reaching 70% capacity provides a buffer before encountering performance issues and allows you to proactively scale the cluster to avoid bottlenecks.

  D) Monitor storage utilization. Increase the size of the Cloud Bigtable cluster when utilization increases above 70% of max capacity.

E) Monitor latency of read operations. Increase the size of the Cloud Bigtable cluster if read operations take longer than 100 ms.

57. You want to analyze hundreds of thousands of social media posts daily at the lowest cost and with the fewest steps.

You have the following requirements:
You will batch-load the posts once per day and run them through the Cloud Natural Language API.
You will extract topics and sentiment from the posts.
You must store the raw posts for archiving and reprocessing.
You will create dashboards to be shared with people both inside and outside your organization.

What should you do?

A) Store the social media posts and the data extracted from the API in BigQuery.

B) Store the social media posts and the data extracted from the API in Cloud SQL.

C) Store the raw social media posts in Cloud Storage, and write the data extracted from the API into BigQuery.

- **Cost-Effectiveness:** Cloud Storage offers a cost-effective option for storing large amounts of unstructured data like social media posts. BigQuery is a good choice for storing the extracted topics and sentiment data, which will likely be smaller in volume and queried more frequently.
- **Minimal Steps:** Batch loading the posts into Cloud Storage once daily and processing them with the Cloud Natural Language API is a simple and efficient workflow.
- **Archiving and Reprocessing:** Cloud Storage provides durable storage for the raw posts, enabling archiving for future reference and potential reprocessing if needed.
- **Dashboards:** BigQuery integrates well with data visualization tools like Looker or Data Studio, allowing you to create dashboards for internal and external audiences.

D) Feed the social media posts into the API directly from the source, and write the extracted data from the API into BigQuery.

58. You store historic data in Cloud Storage. You need to perform analytics on the historic data. You want to use a solution to detect invalid data entries and perform data transformations that will not require programming or knowledge of SQL.

A. Use Cloud Dataflow with Beam to detect errors and perform transformations.
B. Use Cloud Dataprep with recipes to detect errors and perform transformations.

- **No Programming Required:** Cloud Dataprep offers a visual interface with pre-built recipes that allow you to define data cleaning and transformation steps without writing code. This aligns with your requirement of not needing programming skills.
- **Error Detection:** Dataprep provides features for identifying and handling invalid data entries. You can use filters and transformations to address data quality issues.
- **Historic Data in Cloud Storage:** Cloud Dataprep seamlessly integrates with Cloud Storage, allowing you to directly work with your historical data without needing to move it.
  C. Use Cloud Dataproc with a Hadoop job to detect errors and perform transformations.
  D. Use federated tables in BigQuery with queries to detect errors and perform transformations.

59. Your company needs to upload their historic data to Cloud Storage. The security rules don't allow access from external IPs to their on-premises resources. After an initial upload, they will add new data from existing on-premises applications every day. What should they do?

A. Execute gsutil rsync from the on-premises servers.
Dataflow is on cloud is external; "don't allow access from external IPs to their on-premises resources" so no dataflow.
Reference:
https://cloud.google.com/solutions/migration-to-google-cloud-transferring-your-large-datasets#options_available_from_google
B. Use Dataflow and write the data to Cloud Storage.
C. Write a job template in Dataproc to perform the data transfer.
D. Install an FTP server on a Compute Engine VM to receive the files and move them to Cloud Storage.

60. You have a query that filters a BigQuery table using a WHERE clause on timestamp and ID columns. By using bq query '"-dry_run you learn that the query triggers a full scan of the table, even though the filter on timestamp and ID select a tiny fraction of the overall data. You want to reduce the amount

of data scanned by BigQuery with minimal changes to existing SQL queries. What should you do?

A. Create a separate table for each ID.
B. Use the LIMIT keyword to reduce the number of rows returned.
C. Recreate the table with a partitioning column and clustering column.**

- **Minimal Changes to Existing Query:** Partitioning and clustering optimize table structure for specific access patterns without modifying the actual SQL query itself. You can keep your existing WHERE clause filtering by timestamp and ID.
- **Improved Performance:** Partitioning allows BigQuery to efficiently locate the relevant data partition for your query based on the timestamp filter. Clustering by ID further improves performance for queries that involve filtering and processing data based on specific IDs.
- **Reduced Scans:** By leveraging partitioning and clustering, BigQuery can potentially avoid scanning the entire table, significantly reducing the amount of data processed and improving query execution speed.

  D. Use the bq query flag to restrict the number of bytes billed.

61. You have a requirement to insert minute-resolution data from 50,000 sensors into a BigQuery table. You expect significant growth in data volume and need the data to be available within 1 minute of ingestion for real-time analysis of aggregated trends. What should you do?

A. Use bq load to load a batch of sensor data every 60 seconds.
B. Use a Cloud Dataflow pipeline to stream data into the BigQuery table.

- **Cloud Dataflow:** This service allows you to develop and execute data processing pipelines that ingest, transform, and load data in real-time. It integrates seamlessly with BigQuery and supports streaming inserts, making it ideal for scenarios where data needs to be available immediately for analysis.
- **Streaming Data:** Streaming data into BigQuery using Cloud Dataflow ensures that each data record is immediately available for querying and analysis. It meets the requirement of having data available within 1 minute of ingestion.
- **Scalability:** Cloud Dataflow handles the scalability automatically, which is crucial given the expected significant growth in data volume from 50,000 sensors.
- **Real-time Analysis:** By streaming data, you can perform real-time analysis of aggregated trends without waiting for batch processes to complete.

  C. Use the INSERT statement to insert a batch of data every 60 seconds.
  D. Use the MERGE statement to apply updates in batch every 60 seconds.

62. You need to copy millions of sensitive patient records from a relational database to BigQuery. The total size of the database is 10 TB. You need to design a solution that is secure and time-efficient. What should you do?

A. Export the records from the database as an Avro file. Upload the file to GCS using gsutil, and then load the Avro file into BigQuery using the BigQuery web UI in the GCP Console.
B. Export the records from the database as an Avro file. Copy the file onto a Transfer Appliance and send it to Google, and then load the Avro file into BigQuery using the BigQuery web UI in the GCP Console.
- we don't know the bandwidth company has
- https://cloud.google.com/transfer-appliance/docs/4.0/overview

C. Export the records from the database into a CSV file. Create a public URL for the CSV file, and then use Storage Transfer Service to move the file to Cloud Storage. Load the CSV file into BigQuery using the BigQuery web UI in the GCP Console.
D. Export the records from the database as an Avro file. Create a public URL for the Avro file, and then use Storage Transfer Service to move the file to Cloud Storage. Load the Avro file into BigQuery using the BigQuery web UI in the GCP Console.

63. You need to create a near real-time inventory dashboard that reads the main inventory tables in your BigQuery data warehouse. Historical inventory data is stored as inventory balances by item and location. You have several thousand updates to inventory every hour. You want to maximize performance of the dashboard and ensure that the data is accurate. What should you do?

- A. Leverage BigQuery UPDATE statements to update the inventory balances as they are changing.
- B. Partition the inventory balance table by item to reduce the amount of data scanned with each inventory update.
- **C. Use the BigQuery streaming the stream changes into a daily inventory movement table. Calculate balances in a view that joins it to the historical inventory balance table. Update the inventory balance table nightly.**
  - **Near Real-time Data:** BigQuery streaming allows you to ingest inventory updates as they occur, providing near real-time data for your dashboard.
  - **Daily Inventory Movement Table:** Storing daily inventory changes in a separate table keeps the historical balance table optimized for querying while allowing for efficient updates.
  - **Balance Calculation in View:** A view that joins the historical balance table and the daily movement table can calculate up-to-date inventory balances on demand for your dashboard.
  - **Nightly Balance Update:** This approach avoids excessive updates to the historical balance table, ensuring data accuracy and potentially improving query performance.
- D. Use the BigQuery bulk loader to batch load inventory changes into a daily inventory movement table. Calculate balances in a view that joins it to the historical inventory balance table. Update the inventory balance table nightly.

65. You have a data stored in BigQuery. The data in the BigQuery dataset must be highly available. You need to define a storage, backup, and recovery strategy of this data that minimizes cost. How should you configure the BigQuery table that have a recovery point objective (RPO) of 30 days?

A. Set the BigQuery dataset to be regional. In the event of an emergency, use a point-in-time snapshot to recover the data.
B. Set the BigQuery dataset to be regional. Create a scheduled query to make copies of the data to tables suffixed with the time of the backup. In the event of an emergency, use the backup copy of the table.
C. Set the BigQuery dataset to be multi-regional. In the event of an emergency, use a point-in-time snapshot to recover the data.

- **Multi-regional Dataset:** This offers the highest level of availability by replicating your data across geographically separated regions. This minimizes downtime in case of a regional outage, ensuring continuous access to your data even within your 30-day RPO window.
- **Point-in-Time Snapshots:** While less frequent than scheduled backups, snapshots are still a cost-effective way to capture consistent data states. Depending on your snapshot schedule (e.g., daily), you might have a recent enough snapshot to meet your RPO requirement, especially if the data doesn't change frequently.
  D. Set the BigQuery dataset to be multi-regional. Create a scheduled query to make copies of the data to tables suffixed with the time of the backup. In the event of an emergency, use the backup copy of the table.

66. You used Dataprep to create a recipe on a sample of data in a BigQuery table. You want to reuse this recipe on a daily upload of data with the same schema, after the load job with variable execution time completes. What should you do?

- A. Create a cron schedule in Dataprep.
- B. Create an App Engine cron job to schedule the execution of the Dataprep job.
- C. Export the recipe as a Dataprep template, and create a job in Cloud Scheduler.
- **D. Export the Dataprep job as a Dataflow template, and incorporate it into a Composer job.**
  - Overkill
  - D. Export the Cloud Dataprep job as a Cloud Dataflow template, and incorporate it into a Cloud Composer job.
  Reason: cron job and Cloud Scheduler can only run at a fixed time. The requirements is follow a load job with variable execution time, so Cloud Composer is needed.

67. You want to automate execution of a multi-step data pipeline running on Google Cloud. The pipeline includes Dataproc and Dataflow jobs that have

multiple dependencies on each other. You want to use managed services where possible, and the pipeline will run every day. Which tool should you use?

A. cron
B. Cloud Composer

- **Managed Service:** Cloud Composer is a fully managed orchestration service for data pipelines on Google Cloud. This aligns with your requirement of using managed services where possible.
- **Multi-step Pipelines:** Cloud Composer is designed to handle complex data pipelines with various stages, including Dataproc and Dataflow jobs.
- **Dependency Management:** Cloud Composer allows you to define dependencies between different jobs in your pipeline. This ensures that jobs execute in the correct order based on their data requirements.
- **Daily Scheduling:** Cloud Composer can be configured to schedule pipeline execution at regular intervals, such as daily runs in this scenario.
  C. Cloud Scheduler
  D. Workflow Templates on Dataproc

68. You are managing a Cloud Dataproc cluster. You need to make a job run faster while minimizing costs, without losing work in progress on your clusters. What should you do?

A. Increase the cluster size with more non-preemptible workers.
B. Increase the cluster size with preemptible worker nodes, and configure them to forcefully decommission.
C. Increase the cluster size with preemptible worker nodes, and use Cloud Stackdriver to trigger a script to preserve work.
D. Increase the cluster size with preemptible worker nodes, and configure them to use graceful decommissioning.
https://cloud.google.com/dataproc/docs/concepts/configuring-clusters/scaling-clusters#using_graceful_decommissioning
https://cloud.google.com/dataproc/docs/concepts/configuring-clusters/enhanced-flexibility-mode

C:

- **Preemptible Workers:** Utilizing preemptible worker nodes can significantly reduce costs compared to non-preemptible workers. Google Cloud Platform can reclaim these nodes when needed, but with proper management, you can minimize job disruption.
- **Stackdriver and Script:** Cloud Stackdriver provides monitoring and alerting functionalities. You can set alerts to trigger a script when a preemption signal is received. This script can

handle work in progress by, for example, checkpointing data or saving partially completed tasks.

- **Minimize Work Loss:** By implementing the script triggered by Stackdriver, you can ensure that ongoing work is preserved even when preemptible worker nodes are reclaimed.

69. You work for a shipping company that uses handheld scanners to read shipping labels. Your company has strict data privacy standards that require scanners to only transmit tracking numbers when events are sent to Kafka topics. A recent software update caused the scanners to accidentally transmit recipients' personally identifiable information (PII) to analytics systems, which violates user privacy rules. You want to quickly build a scalable solution using cloud-native managed services to prevent exposure of PII to the analytics systems. What should you do?

- **A.** Create an authorized view in BigQuery to restrict access to tables with sensitive data.
- **B.** Install a third-party data validation tool on Compute Engine virtual machines to check the incoming data for sensitive information.
- **C.** Use Cloud Logging to analyze the data passed through the total pipeline to identify transactions that may contain sensitive information.
- **D.** Build a Cloud Function that reads the topics and makes a call to the Cloud Data Loss Prevention (Cloud DLP) API. Use the tagging and confidence levels to either pass or quarantine the data in a bucket for review.
  - **Cloud Function:** This serverless function allows you to quickly build and deploy code to inspect the data stream from the Kafka topics. Cloud Functions are cost-effective and scale automatically, making them suitable for real-time data processing.
  - **Cloud DLP API:** This API integrates with Cloud Functions and enables you to identify and redact sensitive information like PII from your data stream. You can define rules to detect recipient information specifically.
  - **Tagging and Confidence Levels:** Based on the DLP API's analysis, you can configure your Cloud Function to either:
  - **Pass data:** If the data doesn't contain PII with a high confidence level, it can be passed on for further processing.
  - **Quarantine data:** If the DLP API detects PII with a high confidence level, the data can be quarantined in a Cloud Storage bucket for further review and potential redaction or deletion.

70. You have developed three data processing jobs. One executes a Cloud Dataflow pipeline that transforms data uploaded to Cloud Storage and writes results to BigQuery. The second ingests data from on-premises servers and uploads it to Cloud Storage. The third is a Cloud Dataflow pipeline that gets information from third-party data providers and uploads the information to Cloud Storage. You need to be able to schedule and monitor the execution of

these three workflows and manually execute them when needed. What should you do?

- **A. Create a Direct Acyclic Graph in Cloud Composer to schedule and monitor the jobs.** - https://cloud.google.com/composer/docs/concepts/overview
- **B.** Use Stackdriver Monitoring and set up an alert with a Webhook notification to trigger the jobs.
- **C.** Develop an App Engine application to schedule and request the status of the jobs using GCP API calls.
- **D.** Set up cron jobs in a Compute Engine instance to schedule and monitor the pipelines using GCP API calls.

71. You have Cloud Functions written in Node.js that pull messages from Cloud Pub/Sub and send the data to BigQuery. You observe that the message processing rate on the Pub/Sub topic is orders of magnitude higher than anticipated, but there is no error logged in Cloud Logging. What are the two most likely causes of this problem? (Choose two.)

- **A.** Publisher throughput quota is too small.
- **B.** Total outstanding messages exceed the 10-MB maximum.
- **C.** Error handling in the subscriber code is not handling run-time errors properly.
- **D.** The subscriber code cannot keep up with the messages.
- **E.** The subscriber code does not acknowledge the messages that it pulls.
- **C. Error Handling in the subscriber code is not handling run-time errors properly:** While incorrect error handling can lead to issues, in this specific scenario where there are no errors logged in Cloud Logging, it's less likely to be the primary cause of the high processing rate. However, improper error handling could still be a contributing factor if it's causing the code to malfunction and potentially reprocess messages.
- **E. The subscriber code does not acknowledge the messages that it pulls:** This is a critical factor. Unacknowledged messages are redelivered by Pub/Sub, leading to an inflated processing rate as the function appears to be processing the same message multiple times.

72. You are creating a new pipeline in Google Cloud to stream IoT data from Cloud Pub/Sub through Cloud Dataflow to BigQuery. While previewing the data, you notice that roughly 2% of the data appears to be corrupt. You need to modify the Cloud Dataflow pipeline to filter out this corrupt data. What should you do?

- **A.** Add a SideInput that returns a Boolean if the element is corrupt.

- **B. Add a ParDo transform in Cloud Dataflow to discard corrupt elements.**
  - **ParDo Transform:** ParDo is a powerful transformation in Cloud Dataflow that allows you to apply custom logic to each element in your data stream. In this scenario, you can define a ParDo function that checks each message for corruption and discards it if it's corrupt.
  - **Filtering Logic:** Within the ParDo function, you can implement the logic to identify corrupt data based on your specific criteria. This could involve validating data formats, checking for missing fields, or using custom rules.
  - **Discarding Corrupt Data:** If the ParDo function determines an element is corrupt, it can explicitly discard the element using the `output.discard()` method. This ensures the corrupt data doesn't reach BigQuery.
- **C.** Add a Partition transform in Cloud Dataflow to separate valid data from corrupt data.
- **D.** Add a GroupByKey transform in Cloud Dataflow to group all of the valid data together and discard the rest.

73. You have historical data covering the last three years in BigQuery and a data pipeline that delivers new data to BigQuery daily. You have noticed that when the Data Science team runs a query filtered on a date column and limited to 30-90 days of data, the query scans the entire table. You also noticed that your bill is increasing more quickly than you expected. You want to resolve the issue as cost-effectively as possible while maintaining the ability to conduct SQL queries. What should you do?

- **A.** Re-create the tables using DDL. Partition the tables by a column containing a TIMESTAMP or DATE Type.
  - https://cloud.google.com/bigquery/docs/partitioned-tables#dt_partition_shard
  - **Partitioning by Date:** Partitioning the table by a date column (TIMESTAMP or DATE) allows BigQuery to efficiently scan only the relevant partitions for queries with date filters. This significantly reduces the amount of data scanned, leading to improved query performance and cost savings.
  - **DDL (Data Definition Language):** Re-creating the tables using DDL with partitioning allows you to define the partitioning scheme upfront. You can choose a daily, monthly, or yearly partitioning scheme based on your data access patterns and needs.
  - **SQL Compatibility:** Partitioning is transparent to SQL queries. The Data Science team can continue using standard SQL queries with date filters without needing to modify their approach significantly.
- **B.** Recommend that the Data Science team export the table to a CSV file on Cloud Storage and use Cloud Datalab to explore the data by reading the files directly.
- **C.** Modify your pipeline to maintain the last 30-90 days of data in one table and the longer history in a different table to minimize full table scans over the entire history.
- **D.** Write an Apache Beam pipeline that creates a BigQuery table per day. Recommend that the Data Science team use wildcards on the table name suffixes to select the data they

need.
- Partitioning is recommended over table sharding, because partitioned tables perform better This is a google recommendation nowaday.

74. You operate a logistics company, and you want to improve event delivery reliability for vehicle-based sensors. You operate small data centers around the world to capture these events, but leased lines that provide connectivity from your event collection infrastructure to your event processing infrastructure are unreliable, with unpredictable latency. You want to address this issue in the most cost-effective way. What should you do?

A. Deploy small Kafka clusters in your data centers to buffer events.
B. Have the data acquisition devices publish data to Cloud Pub/Sub.

- **Decoupling and Buffering:** Cloud Pub/Sub acts as a decoupling layer between your data acquisition devices and the event processing infrastructure. This means even if the leased line connection drops, devices can continue publishing events to Pub/Sub, which buffers the data.
- **Scalability and Reliability:** Pub/Sub is a highly scalable and reliable messaging service. It can handle spikes in event volume and automatically retries failed deliveries, ensuring data isn't lost due to temporary connectivity issues.
- **Cost-Effectiveness:** Compared to establishing Cloud Interconnect across all data centers, Cloud Pub/Sub offers a more cost-effective solution, especially for unpredictable and potentially low-volume event data from vehicle sensors.
C. Establish a Cloud Interconnect between all remote data centers and Google.
D. Write a Cloud Dataflow pipeline that aggregates all data in session windows.

75. You are a retailer that wants to integrate your online sales capabilities with different in-home assistants, such as Google Home. You need to interpret customer voice commands and issue an order to the backend systems. Which solutions should you choose?

A. Speech-to-Text API
B. Cloud Natural Language API
C. Dialogflow Enterprise Edition

- **Speech-to-Text API:** This API converts spoken audio from the in-home assistant user's voice commands into text. It provides real-time transcription, allowing you to capture the customer's intent accurately.
- **Dialogflow Enterprise Edition:** This is a conversational AI platform that understands the meaning behind the text captured by Speech-to-Text. It allows you to define conversation flows, interpret the customer's intent (e.g., order a specific product), and trigger actions within your backend systems to fulfill the order.

Cloud Speech-to-Text API just converts speech to text. You will have text files as an output and then the requirement is to "interpret customer voice commands and issue an order to the backend systems". This is not achieved by having text files. I would go with option C, since Dialogflow can interpret the commands (intents) and integrates other applications e.g. backend systems.
D. AutoML Natural Language

- **Speech-to-Text API:** This remains the essential component for converting spoken voice commands into text.
- **Custom Logic:** Instead of a comprehensive conversational AI platform like Dialogflow, you can develop custom logic to interpret the transcribed text from Speech-to-Text. This logic can involve:
  - Identifying pre-defined keywords or phrases that indicate order placement (e.g., "order", "buy", "add to cart").
  - Extracting relevant information from the command, such as product name, quantity, using techniques like regular expressions or natural language toolkit libraries.
  - Triggering actions in your backend system to fulfill the order based on the extracted information.

Benefits of this approach:

- **Simpler Implementation:** Compared to Dialogflow, custom logic can be less complex to implement for well-defined voice commands with limited variations.
- **Cost-Effective:** This approach avoids the potential licensing costs associated with Dialogflow.
- **Scalable:** As you introduce new order-related voice commands, you can easily extend your custom logic to recognize them.

Considerations:

- **Command Complexity:** If your voice commands become more complex or require handling variations in phrasing, you might need to explore natural language processing libraries or consider a lightweight version of Dialogflow like Dialogflow Essentials (ES).
- **Development Effort:** Implementing custom logic requires development expertise compared to using a pre-built conversational AI platform.

76. You are designing a messaging system by using Pub/Sub to process clickstream data with an event-driven consumer app that relies on a push subscription. You need to configure the messaging system that is reliable enough to handle temporary downtime of the consumer app. You also need the messaging system to store the input messages that cannot be consumed by the subscriber. The system needs to retry failed messages gradually, avoiding overloading the consumer app, and store the failed messages after a

maximum of 10 retries in a topic. How should you configure the Pub/Sub subscription?

A. Increase the acknowledgement deadline to 10 minutes.
B. Use immediate redelivery as the subscription retry policy, and configure dead lettering to a different topic with maximum delivery attempts set to 10.
C. Use exponential backoff as the subscription retry policy, and configure dead lettering to the same source topic with maximum delivery attempts set to 10.
D. Use exponential backoff as the subscription retry policy, and configure dead lettering to a different topic with maximum delivery attempts set to 10.

- **Exponential backoff:** This retry policy gradually increases the delay between retries, preventing overwhelming the consumer app upon restart. It's ideal for handling temporary downtime.
- **Dead lettering to a different topic:** After 10 retries, messages are sent to a separate "dead letter" topic. This allows for further analysis of these problematic messages without affecting the main message flow.
- **Maximum delivery attempts:** Setting the maximum retries to 10 ensures messages aren't endlessly retried for potentially permanent issues.

77. You designed a data warehouse in BigQuery to analyze sales data. You want a self-serving, low-maintenance, and cost-effective solution to share the sales dataset to other business units in your organization. What should you do?

A. Create an Analytics Hub private exchange, and publish the sales dataset.
B. Enable the other business units' projects to access the authorized views of the sales dataset.
C. Create and share views with the users in the other business units.
D. Use the BigQuery Data Transfer Service to create a schedule that copies the sales dataset to the other business units' projects.

78. You have terabytes of customer behavioral data streaming from Google Analytics into BigQuery daily. Your customers' information, such as their preferences, is hosted on a Cloud SQL for MySQL database. Your CRM database is hosted on a Cloud SQL for PostgreSQL instance. The marketing team wants to use your customers' information from the two databases and the customer behavioral data to create marketing campaigns for yearly active customers. You need to ensure that the marketing team can run the campaigns over 100 times a day on typical days and up to 300 during sales. At

the same time, you want to keep the load on the Cloud SQL databases to a minimum. What should you do?

A. Create BigQuery connections to both Cloud SQL databases. Use BigQuery federated queries on the two databases and the Google Analytics data on BigQuery to run these queries.
B. Create a job on Apache Spark with Dataproc Serverless to query both Cloud SQL databases and the Google Analytics data on BigQuery for these queries.
C. Create streams in Datastream to replicate the required tables from both Cloud SQL databases to BigQuery for these queries.

- **Reduced load on Cloud SQL:** Replicating the required tables to BigQuery eliminates the need for the marketing team's queries to directly hit the Cloud SQL databases, minimizing the load on those systems.
- **Scalability for high query volume:** BigQuery is built for handling massive datasets and high query volume. It can efficiently handle the marketing team's frequent queries (100-300 times daily) without impacting performance.
- **Simplified querying:** By having all the data in BigQuery, the marketing team can easily join the customer information from the Cloud SQL databases with the customer behavioral data for campaign creation.
  D. Create a Dataproc cluster with Trino to establish connections to both Cloud SQL databases and BigQuery, to execute the queries.

79. Your organization is modernizing their IT services and migrating to Google Cloud. You need to organize the data that will be stored in Cloud Storage and BigQuery. You need to enable a data mesh approach to share the data between sales, product design, and marketing departments. What should you do?

A.

1. Create a project for storage of the data for each of your departments.
2. Enable each department to create Cloud Storage buckets and BigQuery datasets.
3. Create user groups for authorized readers for each bucket and dataset.
4. Enable the IT team to administer the user groups to add or remove users as the departments' request.
   B.
5. Create multiple projects for storage of the data for each of your departments' applications.
6. Enable each department to create Cloud Storage buckets and BigQuery datasets.
7. Publish the data that each department shared in Analytics Hub.
8. Enable all departments to discover and subscribe to the data they need in Analytics Hub.
   C.
9. Create a project for storage of the data for your organization.

10. Create a central Cloud Storage bucket with three folders to store the files for each department.
11. Create a central BigQuery dataset with tables prefixed with the department name.
12. Give viewer rights for the storage project for the users of your departments.

D.

13. Create multiple projects for storage of the data for each of your departments' applications.
14. Enable each department to create Cloud Storage buckets and BigQuery datasets.
15. In Dataplex, map each department to a data lake and the Cloud Storage buckets, and map the BigQuery datasets to zones.
16. Enable each department to own and share the data of their data lakes.

- **Decentralized Ownership:** Each department has its own project, allowing them to manage their data storage (Cloud Storage buckets) and processing (BigQuery datasets). This fosters data ownership and accountability.
- **Self-Service Data Management:** Departments can independently create and manage their data structures without relying on a central IT team. This promotes agility and faster data access.
- **Data Sharing with Governance:** Dataplex provides a central platform for mapping data ownership (data lakes) to departments and defining access controls for BigQuery datasets. This allows for controlled sharing while maintaining data governance.

80. You work for a large ecommerce company. You are using Pub/Sub to ingest the clickstream data to Google Cloud for analytics. You observe that when a new subscriber connects to an existing topic to analyze data, they are unable to subscribe to older data. For an upcoming yearly sale event in two months, you need a solution that, once implemented, will enable any new subscriber to read the last 30 days of data. What should you do?

A. Create a new topic, and publish the last 30 days of data each time a new subscriber connects to an existing topic.
B. Set the topic retention policy to 30 days.

- **Topic Retention:** Setting the topic retention policy to 30 days ensures that Pub/Sub retains published messages for that duration. This allows any new subscriber who connects to the topic to access messages published within the last 30 days.
- **Scalability and Efficiency:** This solution is scalable and efficient. You don't need to modify the source system or create additional topics for each new subscriber.
- **Focus on Upcoming Sale:** It directly addresses the requirement for new subscribers to access historical data for the upcoming sale event.

  C. Set the subscriber retention policy to 30 days.
  D. Ask the source system to re-push the data to Pub/Sub, and subscribe to it.

81. You are designing the architecture to process your data from Cloud Storage to BigQuery by using Dataflow. The network team provided you with the Shared VPC network and subnetwork to be used by your pipelines. You need to enable the deployment of the pipeline on the Shared VPC network. What should you do?

A. Assign the compute.networkUser role to the Dataflow service agent.
B. Assign the compute.networkUser role to the service account that executes the Dataflow pipeline.

C. Assign the dataflow.admin role to the Dataflow service agent.
D. Assign the dataflow.admin role to the service account that executes the Dataflow pipeline.

https://cloud.google.com/knowledge/kb/dataflow-job-in-shared-vpc-xpn-permissions-000004261
Dataflow service agent is a role that is assigned to a service account. So is compute.networkUser.
https://cloud.google.com/dataflow/docs/concepts/access-control#example
https://cloud.google.com/dataflow/docs/guides/specifying-networks

Security and permissions for pipelines on Google Cloud

When you run your pipeline, Dataflow uses two service accounts to manage security and permissions:

- The Dataflow service account. The Dataflow service uses the Dataflow service account as part of the job creation request, such as to check project quota and to create worker instances on your behalf. Dataflow also uses the Dataflow service account during job execution to manage the job. This account is also known as the Dataflow service agent.

82. Your infrastructure team has set up an interconnect link between Google Cloud and the on-premises network. You are designing a high-throughput streaming pipeline to ingest data in streaming from an Apache Kafka cluster hosted on-premises. You want to store the data in BigQuery, with as minimal latency as possible. What should you do?link

A. Setup a Kafka Connect bridge between Kafka and Pub/Sub. Use a Google-provided Dataflow template to read the data from Pub/Sub, and write the data to BigQuery.
B. Use a proxy host in the VPC in Google Cloud connecting to Kafka. Write a Dataflow pipeline, read data from the proxy host, and write the data to BigQuery.
C. Use Dataflow, write a pipeline that reads the data from Kafka, and writes the data to BigQuery.

D. Setup a Kafka Connect bridge between Kafka and Pub/Sub. Write a Dataflow pipeline, read the data from Pub/Sub, and write the data to BigQuery.

- **Kafka Connect Bridge:** This bridge efficiently moves data between Kafka and Pub/Sub, leveraging the high throughput capabilities of both platforms.
- **Pub/Sub for Low Latency:** Pub/Sub is a highly scalable and low-latency pub/sub messaging service built for real-time data ingestion. It minimizes latency between on-premises data and Google Cloud.
- **Dataflow for BigQuery Integration:** Dataflow is a managed service for building streaming and batch data pipelines. It provides a reliable and scalable way to read data from Pub/Sub and write it to BigQuery.

83. You migrated your on-premises Apache Hadoop Distributed File System (HDFS) data lake to Cloud Storage. The data scientist team needs to process the data by using Apache Spark and SQL. Security policies need to be enforced at the column level. You need a cost-effective solution that can scale into a data mesh. What should you do?link

A.

1. Deploy a long-living Dataproc cluster with Apache Hive and Ranger enabled.
2. Configure Ranger for column level security.
3. Process with Dataproc Spark or Hive SQL.
   B.
4. **Define a BigLake table.**
5. **Create a taxonomy of policy tags in Data Catalog.**
6. **Add policy tags to columns.**
7. **Process with the Spark-BigQuery connector or BigQuery SQL.**
   C.
8. Load the data to BigQuery tables.
9. Create a taxonomy of policy tags in Data Catalog.
10. Add policy tags to columns.
11. Process with the Spark-BigQuery connector or BigQuery SQL.
    D.
12. Apply an Identity and Access Management (IAM) policy at the file level in Cloud Storage.
13. Define a BigQuery external table for SQL processing. 3. Use Dataproc Spark to process the Cloud Storage files.

84. One of your encryption keys stored in Cloud Key Management Service (Cloud KMS) was exposed. You need to re-encrypt all of your CMEK-protected Cloud Storage data that used that key, and then delete the compromised key. You also want to reduce the risk of objects getting written

without customer-managed encryption key (CMEK) protection in the future. What should you do?link

A. Rotate the Cloud KMS key version. Continue to use the same Cloud Storage bucket.
B. Create a new Cloud KMS key. Set the default CMEK key on the existing Cloud Storage bucket to the new one.
C. Create a new Cloud KMS key. Create a new Cloud Storage bucket. Copy all objects from the old bucket to the new one bucket while specifying the new Cloud KMS key in the copy command.
D. Create a new Cloud KMS key. Create a new Cloud Storage bucket configured to use the new key as the default CMEK key. Copy all objects from the old bucket to the new bucket without specifying a key.

https://cloud.google.com/kms/docs/key-rotation#why_rotate_keys

85. You have an upstream process that writes data to Cloud Storage. This data is then read by an Apache Spark job that runs on Dataproc. These jobs are run in the us-central1 region, but the data could be stored anywhere in the United States. You need to have a recovery process in place in case of a catastrophic single region failure. You need an approach with a maximum of 15 minutes of data loss (RPO:15 mins). You want to ensure that there is minimal latency when reading the data. What should you do?link

A.

1. Create two regional Cloud Storage buckets, one in the us-central1 region and one in the us-south1 region.
2. Have the upstream process write data to the us-central1 bucket. Use the Storage Transfer Service to copy data hourly from the us-central1 bucket to the us-south1 bucket.
3. Run the Dataproc cluster in a zone in the us-central1 region, reading from the bucket in that region.
4. In case of regional failure, redeploy your Dataproc clusters to the us-south1 region and read from the bucket in that region instead.
   B.
5. Create a Cloud Storage bucket in the US multi-region.
6. Run the Dataproc cluster in a zone in the us-central1 region, reading data from the US multi-region bucket.
7. In case of a regional failure, redeploy the Dataproc cluster to the us-central2 region and continue reading from the same bucket.
   C.
8. Create a dual-region Cloud Storage bucket in the us-central1 and us-south1 regions.
9. Enable turbo replication.

10. Run the Dataproc cluster in a zone in the us-central1 region, reading from the bucket in the us-south1 region.
11. In case of a regional failure, redeploy your Dataproc cluster to the us-south1 region and continue reading from the same bucket.
    D.
12. **Create a dual-region Cloud Storage bucket in the us-central1 and us-south1 regions.**
13. **Enable turbo replication.**
14. **Run the Dataproc cluster in a zone in the us-central1 region, reading from the bucket in the same region.**
15. **In case of a regional failure, redeploy the Dataproc clusters to the us-south1 region and read from the same bucket.**

86. You currently have transactional data stored on-premises in a PostgreSQL database. To modernize your data environment, you want to run transactional workloads and support analytics needs with a single database. You need to move to Google Cloud without changing database management systems, and minimize cost and complexity. What should you do?link

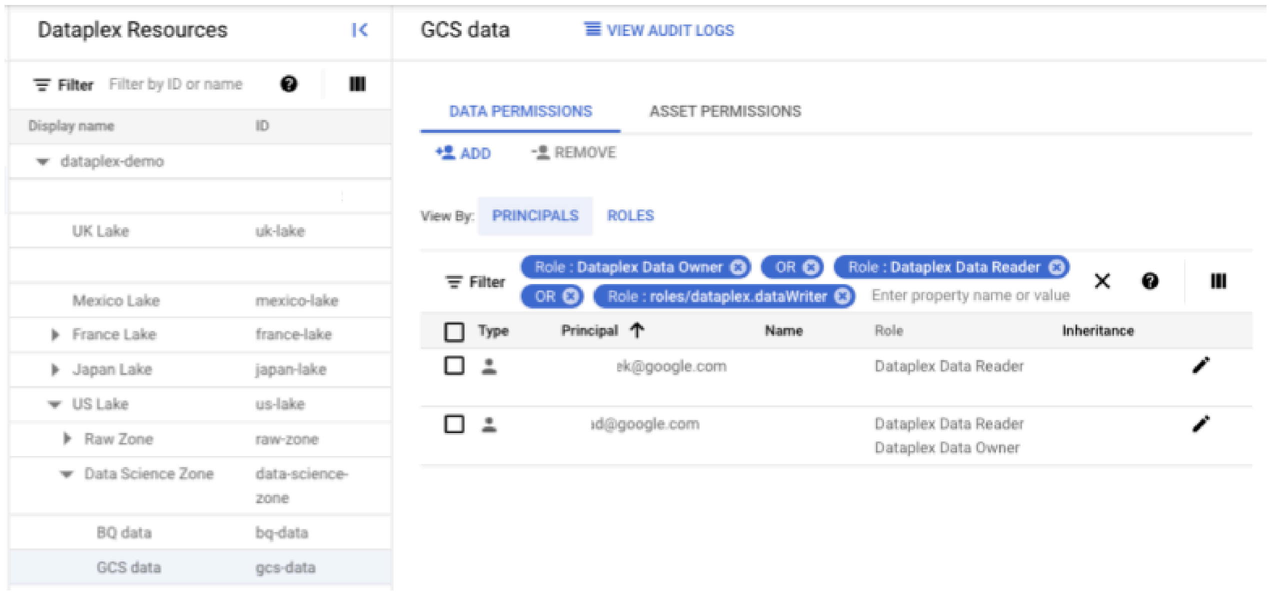
A. Migrate and modernize your database with Cloud Spanner.
B. Migrate your workloads to AlloyDB for PostgreSQL.
- costly
C. Migrate to BigQuery to optimize analytics.
D. Migrate your PostgreSQL database to Cloud SQL for PostgreSQL.

87. You are designing a data mesh on Google Cloud by using Dataplex to manage data in BigQuery and Cloud Storage. You want to simplify data asset permissions. You are creating a customer virtual lake with two user groups: • Data engineers, which require full data lake access • Analytic users, which require access to curated data You need to assign access rights to these two groups. What should you do?link

A.

1. **Grant the dataplex.dataOwner role to the data engineer group on the customer data lake.**
2. **Grant the dataplex.dataReader role to the analytic user group on the customer curated zone.**
   https://cloud.google.com/dataplex/docs/lake-security#data-roles
   B.
3. Grant the dataplex.dataReader role to the data engineer group on the customer data lake.
4. Grant the dataplex.dataOwner to the analytic user group on the customer curated zone.
   C.

5. Grant the bigquery.dataOwner role on BigQuery datasets and the storage.objectCreator role on Cloud Storage buckets to data engineers.
6. Grant the bigquery.dataViewer role on BigQuery datasets and the storage.objectViewer role on Cloud Storage buckets to analytic users.
   D.
7. Grant the bigquery.dataViewer role on BigQuery datasets and the storage.objectViewer role on Cloud Storage buckets to data engineers.
8. Grant the bigquery.dataOwner role on BigQuery datasets and the storage.objectEditor role on Cloud Storage buckets to analytic users.



Display name	ID
dataplex-demo	
UK Lake	uk-lake
Mexico Lake	mexico-lake
France Lake	france-lake
Japan Lake	japan-lake
US Lake	us-lake
Raw Zone	raw-zone
Data Science Zone	data-science-zone
BQ data	bq-data
GCS data	gcs-data

Type	Principal	Name	Role	Inheritance
	ek@google.com		Dataplex Data Reader	
	id@google.com		Dataplex Data Reader Dataplex Data Owner	

88. You are designing the architecture of your application to store data in Cloud Storage. Your application consists of pipelines that read data from a Cloud Storage bucket that contains raw data, and write the data to a second bucket after processing. You want to design an architecture with Cloud Storage resources that are capable of being resilient if a Google Cloud regional failure occurs. You want to minimize the recovery point objective (RPO) if a failure occurs, with no impact on applications that use the stored data. What should you do?

A. Adopt multi-regional Cloud Storage buckets in your architecture.
B. Adopt two regional Cloud Storage buckets, and update your application to write the output on both buckets.
C. Adopt a dual-region Cloud Storage bucket, and enable turbo replication in your architecture.
D. Adopt two regional Cloud Storage buckets, and create a daily task to copy from one bucket to the other.

https://cloud.google.com/blog/products/storage-data-transfer/choose-between-regional-dual-region-and-multi-region-cloud-storage

https://cloud.google.com/storage/docs/dual-regions
https://cloud.google.com/storage/docs/managing-turbo-replication

- **If minimal RPO and regional resiliency are your top priorities, dual-region buckets with turbo replication are the way to go.**
- **If you need strong disaster recovery protection against large-scale outages and want to improve access latency for geographically distributed users, multi-regional buckets can be a valuable option.**

89. You have designed an Apache Beam processing pipeline that reads from a Pub/Sub topic. The topic has a message retention duration of one day, and writes to a Cloud Storage bucket. You need to select a bucket location and processing strategy to prevent data loss in case of a regional outage with an RPO of 15 minutes. What should you do?

- A. 1. Use a dual-region Cloud Storage bucket.
  2. Monitor Dataflow metrics with Cloud Monitoring to determine when an outage occurs.
  3. Seek the subscription back in time by 15 minutes to recover the acknowledged messages.
  4. Start the Dataflow job in a secondary region.
  Not A, because dual-region bucket WITHOUT turbo replication takes atleast 1 hour to sync data between regions. SLA for 100% data sync is 12 hours as per google.
- B. 1. Use a multi-regional Cloud Storage bucket.
  2. Monitor Dataflow metrics with Cloud Monitoring to determine when an outage occurs.
  3. Seek the subscription back in time by 60 minutes to recover the acknowledged messages.
  4. Start the Dataflow job in a secondary region.
- C. 1. Use a regional Cloud Storage bucket.
  2. Monitor Dataflow metrics with Cloud Monitoring to determine when an outage occurs.
  3. Seek the subscription back in time by one day to recover the acknowledged messages.
  4. Start the Dataflow job in a secondary region and write in a bucket in the same region.
- **D. 1. Use a dual-region Cloud Storage bucket with turbo replication enabled.**
  2. **Monitor Dataflow metrics with Cloud Monitoring to determine when an outage occurs.**
  3. **Seek the subscription back in time by 60 minutes to recover the acknowledged messages.**
  4. **Start the Dataflow job in a secondary region.**
  When enabled, turbo replication is designed to replicate 100% of newly written objects to both regions that constitute the dual-region within the recovery point objective of 15 minutes, regardless of object size.

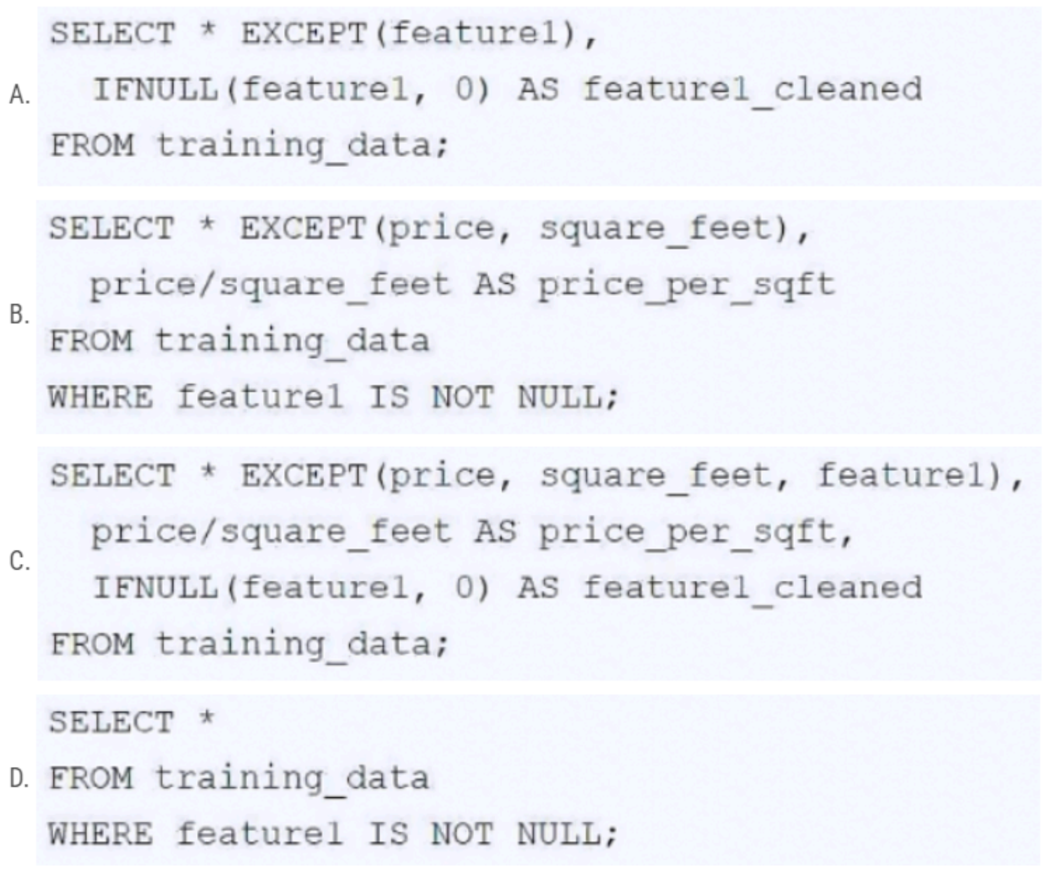
90. You are preparing data that your machine learning team will use to train a model using BigQueryML. They want to predict the price per square foot of real estate. The training data has a column for the price and a column for the number of square feet. Another feature column called 'feature1' contains null values due to missing data.

You want to replace the nulls with zeros to keep more data points. Which query should you use?link

A. `SELECT * EXCEPT (featurel) , IFNULL (featurel, 0) AS featurel_cleaned FROM training_data;`

- `SELECT * EXCEPT (featurel)`: This clause excludes the original 'feature1' column, preventing duplicate column names.
- `IFNULL (featurel, 0) AS featurel_cleaned`: This part replaces null values in the 'feature1' column with zero and renames the column to 'featurel_cleaned'. This creates a new column with zeros for missing data points, suitable for training the model.

  B. `SELECT * EXCEPT (price, square_feet) , price/square_feet AS price_per_sqft FROM training_data WHERE featurel IS NOT NULL;`
  C. `SELECT * EXCEPT (price, square_feet, featurel) , price/square_feet AS price_per_sqft, IFNULL (featurel, 0) AS featurel_cleaned FROM training_data;`
  D. `SELECT * FROM training_data WHERE featurel IS NOT NULL;`



A.
```
SELECT * EXCEPT(feature1),
  IFNULL(feature1, 0) AS feature1_cleaned
FROM training_data;
```

B.
```
SELECT * EXCEPT(price, square_feet),
  price/square_feet AS price_per_sqft
FROM training_data
WHERE feature1 IS NOT NULL;
```

C.
```
SELECT * EXCEPT(price, square_feet, feature1),
  price/square_feet AS price_per_sqft,
  IFNULL(feature1, 0) AS feature1_cleaned
FROM training_data;
```

D.
```
SELECT *
FROM training_data
WHERE feature1 IS NOT NULL;
```

91. Different teams in your organization store customer and performance data in BigQuery. Each team needs to keep full control of their collected data, be able to query data within their projects, and be able to exchange their data with other teams. You need to implement an organization-wide solution, while minimizing operational tasks and costs. What should you do?

A. Ask each team to create authorized views of their data. Grant the biquery.jobUser role to each team.
B. Create a BigQuery scheduled query to replicate all customer data into team projects.
C. Ask each team to publish their data in Analytics Hub. Direct the other teams to subscribe to them.

- Centralized Data Exchange: Analytics Hub provides a unified platform for data sharing across teams and organizations. It simplifies the process of publishing, discovering, and subscribing to datasets, reducing operational overhead.
- Data Ownership and Control: Each team retains full control over their data, deciding which datasets to publish and who can access them. This ensures data governance and security.
- Cross-Project Querying: Once a team subscribes to a dataset in Analytics Hub, they can query it directly from their own BigQuery project, enabling seamless data access without data replication. Cost Efficiency: Analytics Hub eliminates the need for data duplication or complex ETL processes, reducing storage and processing costs.
  D. Enable each team to create materialized views of the data they need to access in their projects.

92. You are developing a model to identify the factors that lead to sales conversions for your customers. You have completed processing your data. You want to continue through the model development lifecycle. What should you do?link

A. Use your model to run predictions on fresh customer input data.
B. Monitor your model performance, and make any adjustments needed.
C. Delineate what data will be used for testing and what will be used for training the model.
- Delineate: describe or portray (something) precisely.
D. Test and evaluate your model on your curated data to determine how well the model performs.

93. You have one BigQuery dataset which includes customers' street addresses. You want to retrieve all occurrences of street addresses from the dataset. What should you do?

A. Write a SQL query in BigQuery by using REGEXP_CONTAINS on all tables in your dataset to find rows where the word "street" appears.
B. Create a deep inspection job on each table in your dataset with Cloud Data Loss Prevention and create an inspection template that includes the STREET_ADDRESS infoType.
https://cloud.google.com/sensitive-data-protection/docs/learn-about-your-data#inspection
C. Create a discovery scan configuration on your organization with Cloud Data Loss Prevention and create an inspection template that includes the STREET_ADDRESS infoType.

D. Create a de-identification job in Cloud Data Loss Prevention and use the masking transformation.

- **B. Deep Inspection Job:** Cloud Data Loss Prevention (DLP) is primarily for identifying and protecting sensitive data. While it can detect street addresses, a deep inspection job on each table is computationally expensive.
- **C. Discovery Scan Configuration:** This option utilizes DLP to scan your entire organization's data, which is overkill for retrieving street addresses from a specific dataset. It's designed for broader data discovery, not targeted retrieval.
- **D. De-identification Job:** DLP's de-identification job is for masking sensitive data, not retrieving it. While it can mask street addresses, that's not your goal here.

94. Your company operates in three domains: airlines, hotels, and ride-hailing services. Each domain has two teams: analytics and data science, which create data assets in BigQuery with the help of a central data platform team. However, as each domain is evolving rapidly, the central data platform team is becoming a bottleneck. This is causing delays in deriving insights from data, and resulting in stale data when pipelines are not kept up to date. You need to design a data mesh architecture by using Dataplex to eliminate the bottleneck. What should you do?

- A. 1. Create one lake for each team. Inside each lake, create one zone for each domain.
  2. Attach each of the BigQuery datasets created by the individual teams as assets to the respective zone.
  3. Have the central data platform team manage all zones' data assets.
- B. 1. Create one lake for each team. Inside each lake, create one zone for each domain.
  2. Attach each of the BigQuery datasets created by the individual teams as assets to the respective zone.
  3. Direct each domain to manage their own zone's data assets.
- **C. 1. Create one lake for each domain. Inside each lake, create one zone for each team.**
  2. **Attach each of the BigQuery datasets created by the individual teams as assets to the respective zone.**
  3. **Direct each domain to manage their own lake's data assets.**
- D. 1. Create one lake for each domain. Inside each lake, create one zone for each team.
  2. Attach each of the BigQuery datasets created by the individual teams as assets to the respective zone.
  3. Have the central data platform team manage all lakes' data assets.

95. You have an inventory of VM data stored in the BigQuery table. You want to prepare the data for regular reporting in the most cost-effective way. You need to exclude VM rows with fewer than 8 vCPU in your report. What should you do?

- **A. Create a view with a filter to drop rows with fewer than 8 vCPU, and use the UNNEST operator.**

- B. Create a materialized view with a filter to drop rows with fewer than 8 vCPU, and use the WITH common table expression.
  - *Potential unnecessary costs, because it stores data, and updation costs.*
- C. Create a view with a filter to drop rows with fewer than 8 vCPU, and use the WITH common table expression.
- D. Use Dataflow to batch process and write the result to another BigQuery table.

link

id	Row	name	components.name	components.qty
1	vm02781	d-jp-kfk-02-02	vcpu	2
			memory	8
			boot-disk	10
			disk—I	50
2	vm11490	i-jp-kfk-02-07	vcpu	16
			memory	64
			boot_disk	10
			disk—I	200
3	vm18130	i-jp-kfk"02-08	vcpu	8
			memory	8
			boot-disk	10

96. Your team is building a data lake platform on Google Cloud. As a part of the data foundation design, you are planning to store all the raw data in Cloud Storage. You are expecting to ingest approximately 25 GB of data a day and your billing department is worried about the increasing cost of storing old data. The current business requirements are:

- The old data can be deleted anytime.
- There is no predefined access pattern of the old data.
- The old data should be available instantly when accessed.
- There should not be any charges for data retrieval.

What should you do to optimize for cost? link

- **A. Create the bucket with the Autoclass storage class feature.** https://cloud.google.com/storage/docs/autoclass
- **B.** Create an Object Lifecycle Management policy to modify the storage class for data older than 30 days to nearline, 90 days to coldline, and 365 days to archive storage class. Delete old data as needed.
- C. Create an Object Lifecycle Management policy to modify the storage class for data older than 30 days to coldline, 90 days to nearline, and 365 days to archive storage class. Delete

old data as needed.

- D. Create an Object Lifecycle Management policy to modify the storage class for data older than 30 days to nearline, 45 days to coldline, and 60 days to archive storage class. Delete old data as needed.

97. Your company's data platform ingests CSV file dumps of booking and user profile data from upstream sources into Cloud Storage. The data analyst team wants to join these datasets on the email field available in both the datasets to perform analysis. However, personally identifiable information (PII) should not be accessible to the analysts. You need to de-identify the email field in both the datasets before loading them into BigQuery for analysts. What should you do?

- A. 1. Create a pipeline to de-identify the email field by using recordTransformations in Cloud Data Loss Prevention (Cloud DLP) with masking as the de-identification transformations type.
  2. Load the booking and user profile data into a BigQuery table.

- **B. 1. Create a pipeline to de-identify the email field by using recordTransformations in Cloud DLP with format-preserving encryption with FFX as the de-identification transformation type.**
  2. **Load the booking and user profile data into a BigQuery table.**
  https://cloud.google.com/sensitive-data-protection/docs/pseudonymization
  Format-preserving encryption (FPE) with FFX in Cloud DLP is a strong choice for de-identifying PII like email addresses. FPE maintains the format of the data and ensures that the same input results in the same encrypted output consistently. This means the email fields in both datasets can be encrypted to the same value, allowing for accurate joins in BigQuery while keeping the actual email addresses hidden.

- C. 1. Load the CSV files from Cloud Storage into a BigQuery table, and enable dynamic data masking.
  2. Create a policy tag with the email mask as the data masking rule.
  3. Assign the policy to the email field in both tables. A
  4. Assign the Identity and Access Management bigquerydatapolicy.maskedReader role for the BigQuery tables to the analysts.

- D. 1. Load the CSV files from Cloud Storage into a BigQuery table, and enable dynamic data masking.
  2. Create a policy tag with the default masking value as the data masking rule.
  3. Assign the policy to the email field in both tables.
  4. Assign the Identity and Access Management bigquerydatapolicy.maskedReader role for the BigQuery tables to the analysts

Dynamic Data Masking: These options involve masking the email field dynamically within BigQuery, which does not address the requirement to de-identify data before loading into

BigQuery. Additionally, dynamic masking does not prevent access to the actual email data before it is loaded into BigQuery, potentially exposing PII during the data ingestion process.

98. You have important legal hold documents in a Cloud Storage bucket. You need to ensure that these documents are not deleted or modified. What should you do?

- **A. Set a retention policy. Lock the retention policy.**
  https://cloud.google.com/storage/docs/bucket-lock#overview
  Setting a retention policy on a Cloud Storage bucket prevents objects from being deleted for the duration of the retention period. - Locking the policy makes it immutable, meaning that the retention period cannot be reduced or removed, thus ensuring that the documents cannot be deleted or overwritten until the retention period expires.
- B. Set a retention policy. Set the default storage class to Archive for long-term digital preservation.
- C. Enable the Object Versioning feature. Add a lifecycle rule.
- D. Enable the Object Versioning feature. Create a copy in a bucket in a different region.

99. You are designing a data warehouse in BigQuery to analyze sales data for a telecommunication service provider. You need to create a data model for customers, products, and subscriptions. All customers, products, and subscriptions can be updated monthly, but you must maintain a historical record of all data. You plan to use the visualization layer for current and historical reporting. You need to ensure that the data model is simple, easy-to-use, and cost-effective. What should you do?

- A. Create a normalized model with tables for each entity. Use snapshots before updates to track historical data.
- B. Create a normalized model with tables for each entity. Keep all input files in a Cloud Storage bucket to track historical data.
- C. Create a denormalized model with nested and repeated fields. Update the table and use snapshots to track historical data.
  managing snapshots adds complexity compared to using ingestion timestamps.
- **D. Create a denormalized, append-only model with nested and repeated fields. Use the ingestion timestamp to track historical data.**
  A denormalized, append-only model simplifies query complexity by eliminating the need for joins.
  Adding data with an ingestion timestamp allows for easy retrieval of both current and historical states.
  Instead of updating records, new records are appended, which maintains historical information without the need to create separate snapshots.

https://cloud.google.com/bigquery/docs/table-snapshots-intro

100. You are deploying a batch pipeline in Dataflow. This pipeline reads data from Cloud Storage, transforms the data, and then writes the data into BigQuery. The

security team has enabled an organizational constraint in Google Cloud, requiring all Compute Engine instances to use only internal IP addresses and no external IP addresses. What should you do?link

- A. Ensure that your workers have network tags to access Cloud Storage and BigQuery. Use Dataflow with only internal IP addresses.
- B. Ensure that the firewall rules allow access to Cloud Storage and BigQuery. Use Dataflow with only internal IPs.
- C. Create a VPC Service Controls perimeter that contains the VPC network and add Dataflow, Cloud Storage, and BigQuery as allowed services in the perimeter. Use Dataflow with only internal IP addresses. https://cloud.google.com/vpc-service-controls/docs/overview
- **D. Ensure that Private Google Access is enabled in the subnetwork. Use Dataflow with only internal IP addresses.** https://cloud.google.com/dataflow/docs/guides/routes-firewall
  - Private Google Access for services allows VM instances with only internal IP addresses in a VPC network or on-premises networks (via Cloud VPN or Cloud Interconnect) to reach Google APIs and services.
  - When you launch a Dataflow job, you can specify that it should use worker instances without external IP addresses if Private Google Access is enabled on the subnetwork where these instances are launched.
  - This way, your Dataflow workers will be able to access Cloud Storage and BigQuery without violating the organizational constraint of no external IPs.

101. You are running a Dataflow streaming pipeline, with Streaming Engine and Horizontal Autoscaling enabled. You have set the maximum number of workers to 1000. The input of your pipeline is Pub/Sub messages with notifications from Cloud Storage. One of the pipeline transforms reads CSV files and emits an element for every CSV line. The job performance is low, the pipeline is using only 10 workers, and you notice that the autoscaler is not spinning up additional workers. What should you do to improve performance?

- A. Enable Vertical Autoscaling to let the pipeline use larger workers.
- **B. Change the pipeline code, and introduce a Reshuffle step to prevent fusion.** https://cloud.google.com/dataflow/docs/pipeline-lifecycle#prevent_fusion
- C. Update the job to increase the maximum number of workers.
- D. Use Dataflow Prime, and enable Right Fitting to increase the worker resources.

https://cloud.google.com/dataflow/docs/pipeline-lifecycle#fusion_optimization

102. You have an Oracle database deployed in a VM as part of a Virtual Private Cloud (VPC) network. You want to replicate and continuously synchronize 50 tables to BigQuery. You want to minimize the need to manage infrastructure. What should you do?

- A. Deploy Apache Kafka in the same VPC network, use Kafka Connect Oracle Change Data Capture (CDC), and Dataflow to stream the Kafka topic to BigQuery.
- B. Create a Pub/Sub subscription to write to BigQuery directly. Deploy the Debezium Oracle connector to capture changes in the Oracle database, and sink to the Pub/Sub topic.
- C. Deploy Apache Kafka in the same VPC network, use Kafka Connect Oracle change data capture (CDC), and the Kafka Connect Google BigQuery Sink Connector.
- **D. Create a Datastream service from Oracle to BigQuery, use a private connectivity configuration to the same VPC network, and a connection profile to BigQuery.**
  - Datastream is a serverless and easy-to-use change data capture (CDC) and replication service.
  - You would create a Datastream service that sources from your Oracle database and targets BigQuery, with private connectivity configuration to the same VPC.
  - This option is designed to minimize the need to manage infrastructure and is a fully managed service.

--All options seem correct--

103. You are deploying an Apache Airflow directed acyclic graph (DAG) in a Cloud Composer 2 instance. You have incoming files in a Cloud Storage bucket that the DAG processes, one file at a time. The Cloud Composer instance is deployed in a subnetwork with no Internet access. Instead of running the DAG based on a schedule, you want to run the DAG in a reactive way every time a new file is received. What should you do?

- **A. 1. Enable Private Google Access in the subnetwork, and set up Cloud Storage notifications to a Pub/Sub topic.**
  2. **Create a push subscription that points to the web server URL.**
  https://cloud.google.com/composer/docs/composer-2/triggering-gcf-pubsub
- B. 1. Enable the Cloud Composer API, and set up Cloud Storage notifications to trigger a Cloud Function.
  2. Write a Cloud Function instance to call the DAG by using the Cloud Composer API and the web server URL.
  3. Use VPC Serverless Access to reach the web server URL.
- **C**. 1. Enable the Airflow REST API, and set up Cloud Storage notifications to trigger a Cloud Function instance.
  2. Create a Private Service Connect (PSC) endpoint.

3. Write a Cloud Function that connects to the Cloud Composer cluster through the PSC endpoint.

- D. 1. Enable the Airflow REST API, and set up Cloud Storage notifications to trigger a Cloud Function instance.
  2. Write a Cloud Function instance to call the DAG by using the Airflow REST API and the web server URL.
  3. Use VPC Serverless Access to reach the web server URL.
  https://cloud.google.com/composer/docs/how-to/using/triggering-with-gcf

104. You are planning to use Cloud Storage as part of your data lake solution. The Cloud Storage bucket will contain objects ingested from external systems. Each object will be ingested once, and the access patterns of individual objects will be random. You want to minimize the cost of storing and retrieving these objects. You want to ensure that any cost optimization efforts are transparent to the users and applications. What should you do?

- **A. Create a Cloud Storage bucket with Autoclass enabled.**
  - Autoclass automatically analyzes access patterns of objects and automatically transitions them to the most cost-effective storage class within Standard, Nearline, Coldline, or Archive.
  - This eliminates the need for manual intervention or setting specific age thresholds.
  - No user or application interaction is required, ensuring transparency.
- B. Create a Cloud Storage bucket with an Object Lifecycle Management policy to transition objects from Standard to Coldline storage class if an object age reaches 30 days.
- C. Create a Cloud Storage bucket with an Object Lifecycle Management policy to transition objects from Standard to Coldline storage class if an object is not live.
- D. Create two Cloud Storage buckets. Use the Standard storage class for the first bucket, and use the Coldline storage class for the second bucket. Migrate objects from the first bucket to the second bucket after 30 days.
  ***Drawback**: Manual migration between buckets introduces operational complexity and is not transparent to users and applications. Additionally, it may not optimize costs effectively if access patterns do not align with the fixed 30-day policy.*

105. You have several different file type data sources, such as Apache Parquet and CSV. You want to store the data in Cloud Storage. You need to set up an object sink for your data that allows you to use your own encryption keys. You want to use a GUI-based solution. What should you do?

- A. Use Storage Transfer Service to move files into Cloud Storage.
  Even though storage transfer service can be used in GUI, it does not support CMEK which is required in this question.
  "Storage Transfer Service does not encrypt data on your behalf, such as in customer-

managed encryption keys (CMEK). We only encrypt data in transit."
Ref: https://cloud.google.com/storage-transfer/docs/on-prem-security

- **B. Use Cloud Data Fusion to move files into Cloud Storage.**
  https://cloud.google.com/data-fusion/docs/how-to/customer-managed-encryption-keys#create-instance
- C. Use Dataflow to move files into Cloud Storage.
- D. Use BigQuery Data Transfer Service to move files into BigQuery.

106. Your business users need a way to clean and prepare data before using the data for analysis. Your business users are less technically savvy and prefer to work with graphical user interfaces to define their transformations. After the data has been transformed, the business users want to perform their analysis directly in a spreadsheet. You need to recommend a solution that they can use. What should you do?

- **A. Use Dataprep to clean the data, and write the results to BigQuery. Analyze the data by using Connected Sheets.**
- B. Use Dataprep to clean the data, and write the results to BigQuery. Analyze the data by using Looker Studio.
- C. Use Dataflow to clean the data, and write the results to BigQuery. Analyze the data by using Connected Sheets.
- D. Use Dataflow to clean the data, and write the results to BigQuery. Analyze the data by using Looker Studio.
  https://cloud.google.com/bigquery/docs/connected-sheets
  https://cloud.google.com/dataprep

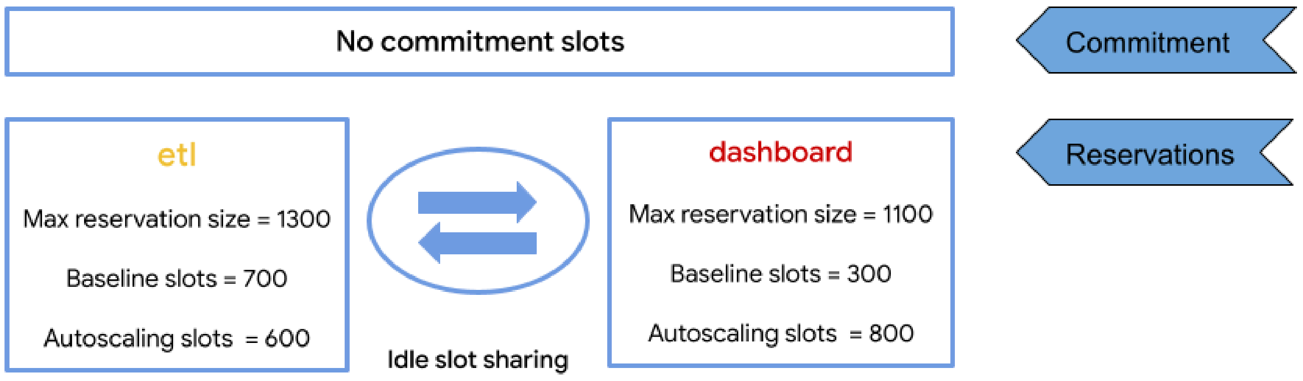
107. You have two projects where you run BigQuery jobs:

• One project runs production jobs that have strict completion time SLAs. These are high priority jobs that must have the required compute resources available when needed. These jobs generally never go below a 300 slot utilization, but occasionally spike up an additional 500 slots.
• The other project is for users to run ad-hoc analytical queries. This project generally never uses more than 200 slots at a time. You want these ad-hoc queries to be billed based on how much data users scan rather than by slot capacity.

You need to ensure that both projects have the appropriate compute resources available. What should you do?

- A. Create a single Enterprise Edition reservation for both projects. Set a baseline of 300 slots. Enable autoscaling up to 700 slots.
- **B. Create two reservations, one for each of the projects. For the SLA project, use an Enterprise Edition with a baseline of 300 slots and enable autoscaling up to 500 slots. For the ad-hoc project, configure on-demand billing.**

- C. Create two Enterprise Edition reservations, one for each of the projects. For the SLA project, set a baseline of 300 slots and enable autoscaling up to 500 slots. For the ad-hoc project, set a reservation baseline of 0 slots and set the ignore idle slots flag to False.
- D. Create two Enterprise Edition reservations, one for each of the projects. For the SLA project, set a baseline of 800 slots. For the ad-hoc project, enable autoscaling up to 200 slots.
- **SLA Project**:
  - **Enterprise Edition Reservation**: Ensures high-priority production jobs have guaranteed access to the necessary compute resources.
  - **Baseline of 300 slots**: Guarantees the minimum required slots for consistent performance.
  - **Autoscaling up to 500 slots**: Allows the system to scale up to 800 slots (300 baseline + 500 autoscaling) during peak times, ensuring the SLA is met without over-provisioning. https://cloud.google.com/bigquery/docs/slots-autoscaling-intro
- **Ad-hoc Project**:
  - **On-Demand Billing**: Billing is based on data scanned rather than reserved slot capacity. This aligns with the usage pattern of ad-hoc queries, which do not have strict SLA requirements and typically benefit from cost-efficiency associated with on-demand billing.
  - **No Reserved Slots**: Avoids unnecessary costs for unused capacity, as ad-hoc queries do not consistently use a large number of slots.



https://youtu.be/KhPPWyLGdfs?si=J23dB2c0HIxVov_u

108. You want to migrate your existing Teradata data warehouse to BigQuery. You want to move the historical data to BigQuery by using the most efficient method that requires the least amount of programming, but local storage space on your existing data warehouse is limited. What should you do?

- **A. Use BigQuery Data Transfer Service by using the Java Database Connectivity (JDBC) driver with FastExport connection.**

- B. Create a Teradata Parallel Transporter (TPT) export script to export the historical data, and import to BigQuery by using the bq command-line tool.
- C. Use BigQuery Data Transfer Service with the Teradata Parallel Transporter (TPT) tbuild utility.
- D. Create a script to export the historical data, and upload in batches to Cloud Storage. Set up a BigQuery Data Transfer Service instance from Cloud Storage to BigQuery.

https://cloud.google.com/bigquery/docs/migration/teradata-overview#extraction_method

- Reduced Local Storage: By using FastExport, data is directly streamed from Teradata to BigQuery without the need for local storage, addressing your storage limitations.
- Minimal Programming: BigQuery Data Transfer Service offers a user-friendly interface, eliminating the need for extensive scripting or coding.
  Extraction using a JDBC driver with FastExport connection. If there are constraints on the local storage space available for extracted files, or if there is some reason you can't use TPT, then use this extraction method.

109. You are on the data governance team and are implementing security requirements. You need to encrypt all your data in BigQuery by using an encryption key managed by your team. You must implement a mechanism to generate and store encryption material only on your on-premises hardware security module (HSM). You want to rely on Google managed solutions. What should you do?

- A. Create the encryption key in the on-premises HSM, and import it into a Cloud Key Management Service (Cloud KMS) key. Associate the created Cloud KMS key while creating the BigQuery resources.

- **B. Create the encryption key in the on-premises HSM and link it to a Cloud External Key Manager (Cloud EKM) key. Associate the created Cloud KMS key while creating the BigQuery resources.**

- C. Create the encryption key in the on-premises HSM, and import it into Cloud Key Management Service (Cloud HSM) key. Associate the created Cloud HSM key while creating the BigQuery resources.

- D. Create the encryption key in the on-premises HSM. Create BigQuery resources and encrypt data while ingesting them into BigQuery.
  - Cloud EKM allows you to use encryption keys managed in external key management systems, including on-premises HSMs, while using Google Cloud services.
  - This means that the key material remains in your control and environment, and Google Cloud services use it via the Cloud EKM integration.
  - This approach aligns with the need to generate and store encryption material only on your on-premises HSM and is the correct way to integrate such keys with BigQuery.
    ====== Why not Option C

- Cloud HSM is a fully managed service by Google Cloud that provides HSMs for your cryptographic needs. However, it's a cloud-based solution, and the keys generated or managed in Cloud HSM are not stored on-premises. This option doesn't align with the requirement to use only on-premises HSM for key storage.

https://cloud.google.com/kms/docs/ekm#ekm-management-mode

110. You maintain ETL pipelines. You notice that a streaming pipeline running on Dataflow is taking a long time to process incoming data, which causes output delays. You also noticed that the pipeline graph was automatically optimized by Dataflow and merged into one step. You want to identify where the potential bottleneck is occurring. What should you do?

- **A. Insert a Reshuffle operation after each processing step, and monitor the execution details in the Dataflow console.**
- B. Insert output sinks after each key processing step, and observe the writing throughput of each block.
- C. Log debug information in each ParDo function, and analyze the logs at execution time.
- D. Verify that the Dataflow service accounts have appropriate permissions to write the processed data to the output sinks.
  - The Reshuffle operation is used in Dataflow pipelines to break fusion and redistribute elements, which can sometimes help improve parallelization and identify bottlenecks.
  - By inserting Reshuffle after each processing step and observing the pipeline's performance in the Dataflow console, you can potentially identify stages that are disproportionately slow or stalled.
  - This can help in pinpointing the step where the bottleneck might be occurring.

111. You are running your BigQuery project in the on-demand billing model and are executing a change data capture (CDC) process that ingests data. The CDC process loads 1 GB of data every 10 minutes into a temporary table, and then performs a merge into a 10 TB target table. This process is very scan intensive and you want to explore options to enable a predictable cost model. You need to create a BigQuery reservation based on utilization information gathered from BigQuery Monitoring and apply the reservation to the CDC process. What should you do?

- A. Create a BigQuery reservation for the dataset.
- B. Create a BigQuery reservation for the job.
- C. Create a BigQuery reservation for the service account running the job.
- **D. Create a BigQuery reservation for the project.**

https://cloud.google.com/bigquery/docs/reservations-intro#understand_workload_management https://cloud.google.com/blog/products/data-analytics/manage-bigquery-costs-with-custom-quotas

112. You are designing a fault-tolerant architecture to store data in a regional BigQuery dataset. You need to ensure that your application is able to recover from a corruption event in your tables that occurred within the past seven days. You want to adopt managed services with the lowest RPO and most cost-effective solution. What should you do?

- **A. Access historical data by using time travel in BigQuery.**
- B. Export the data from BigQuery into a new table that excludes the corrupted data
- C. Create a BigQuery table snapshot on a daily basis.
- D. Migrate your data to multi-region BigQuery buckets.
  https://cloud.google.com/bigquery/docs/access-historical-data

```
SELECT * FROM `mydataset.mytable`
FOR SYSTEM_TIME AS OF TIMESTAMP_SUB(CURRENT_TIMESTAMP(), INTERVAL 1 HOUR);
```

113. You are building a streaming Dataflow pipeline that ingests noise level data from hundreds of sensors placed near construction sites across a city. The sensors measure noise level every ten seconds, and send that data to the pipeline when levels reach above 70 dBA. You need to detect the average noise level from a sensor when data is received for a duration of more than 30 minutes, but the window ends when no data has been received for 15 minutes. What should you do?

- **A. Use session windows with a 15-minute gap duration.**
- B. Use session windows with a 30-minute gap duration.
- C. Use hopping windows with a 15-minute window, and a thirty-minute period.
- D. Use tumbling windows with a 15-minute window and a fifteen-minute.withAllowedLateness operator.

Use a session Window to capture data and create an aggregation when the Session is larger than 30 minutes.
https://cloud.google.com/dataflow/docs/concepts/streaming-pipelines#session-windows
https://beam.apache.org/releases/javadoc/2.6.0/org/apache/beam/sdk/transforms/windowing/Sessions.html

114. You are creating a data model in BigQuery that will hold retail transaction data. Your two largest tables, sales_transaction_header and sales_transaction_line, have a tightly coupled immutable relationship. These tables are rarely modified after load and are frequently joined when queried. You need to model the sales_transaction_header and sales_transaction_line tables to improve the performance of data analytics queries. What should you do?

- **A. Create a sales_transaction table that holds the sales_transaction_header information as rows and the sales_transaction_line rows as nested and repeated fields.**
- B. Create a sales_transaction table that holds the sales_transaction_header and sales_transaction_line information as rows, duplicating the sales_transaction_header data for each line.
- C. Create a sales_transaction table that stores the sales_transaction_header and sales_transaction_line data as a JSON data type.
- D. Create separate sales_transaction_header and sales_transaction_line tables and, when querying, specify the sales_transaction_line first in the WHERE clause.

https://cloud.google.com/bigquery/docs/best-practices-performance-nested

115. You created a new version of a Dataflow streaming data ingestion pipeline that reads from Pub/Sub and writes to BigQuery. The previous version of the pipeline that runs in production uses a 5-minute window for processing. You need to deploy the new version of the pipeline without losing any data, creating inconsistencies, or increasing the processing latency by more than 10 minutes. What should you do?

A. Update the old pipeline with the new pipeline code.
B. Snapshot the old pipeline, stop the old pipeline, and then start the new pipeline from the snapshot.
C. Drain the old pipeline, then start the new pipeline.
D. Cancel the old pipeline, then start the new pipeline.

- **Graceful Transition**: By draining the old pipeline, you allow it to finish processing its current windows and flush any buffered data to ensure no data loss.
- **Avoiding Inconsistencies**: This approach helps prevent inconsistencies by ensuring that all data processed by the old pipeline completes before switching to the new pipeline.
- **Controlled Deployment**: It provides a controlled transition without abruptly stopping the old pipeline, which could lead to data loss or incomplete processing.

https://cloud.google.com/dataflow/docs/guides/stopping-a-pipeline#drain

116. Your organization's data assets are stored in BigQuery, Pub/Sub, and a PostgreSQL instance running on Compute Engine. Because there are multiple domains and diverse teams using the data, teams in your organization are unable to discover existing data assets. You need to design a solution to improve data discoverability while keeping development and configuration efforts to a minimum. What should you do?

- A. Use Data Catalog to automatically catalog BigQuery datasets. Use Data Catalog APIs to manually catalog Pub/Sub topics and PostgreSQL tables.

- *B. Use Data Catalog to automatically catalog BigQuery datasets and Pub/Sub topics. Use Data Catalog APIs to manually catalog PostgreSQL tables.*
- **C. Use Data Catalog to automatically catalog BigQuery datasets and Pub/Sub topics. Use custom connectors to manually catalog PostgreSQL tables.**
- D. Use customer connectors to manually catalog BigQuery datasets, Pub/Sub topics, and PostgreSQL tables.

Data Catalog is the best choice. But for cataloging PostgreSQL it is better to use a connector when available, instead of using API.
https://cloud.google.com/data-catalog/docs/integrate-data-sources#integrate_unsupported_data_sources

117. You need to create a SQL pipeline. The pipeline runs an aggregate SQL transformation on a BigQuery table every two hours and appends the result to another existing BigQuery table. You need to configure the pipeline to retry if errors occur. You want the pipeline to send an email notification after three consecutive failures. What should you do?link

- A. Use the BigQueryUpsertTableOperator in Cloud Composer, set the retry parameter to three, and set the email_on_failure parameter to true.
- **B. Use the BigQueryInsertJobOperator in Cloud Composer, set the retry parameter to three, and set the email_on_failure parameter to true.**
- C. Create a BigQuery scheduled query to run the SQL transformation with schedule options that repeats every two hours, and enable email notifications.
- D. Create a BigQuery scheduled query to run the SQL transformation with schedule options that repeats every two hours, and enable notification to Pub/Sub topic. Use Pub/Sub and Cloud Functions to send an email after three failed executions.

Option B also incomplete:
- email is sent for each failure
- scheduled to run after 2 hours in not mentioned
Option D:
- Scheduled queries have no retry feature

https://cloud.google.com/composer/docs/how-to/using/writing-dags#notifications_on_operator_failure
https://airflow.apache.org/docs/apache-airflow-providers-google/stable/_api/airflow/providers/google/cloud/operators/bigquery/index.html#airflow.providers.google.cloud.operators.bigquery.BigQueryInsertJobOperator
https://cloud.google.com/composer/docs/how-to/using/writing-dags
https://cloud.google.com/bigquery/docs/scheduling-queries

```
from airflow import DAG
from airflow.providers.google.cloud.operators.bigquery import
BigQueryInsertJobOperator
from airflow.utils.dates import days_ago
from datetime import timedelta

default_args = {
    'owner': 'airflow',
    'depends_on_past': False,
    'email': ['your_email@example.com'],
    'email_on_failure': True,
    'email_on_retry': False,
    'retries': 3,
    'retry_delay': timedelta(minutes=5),
}

dag = DAG(
    'bigquery_aggregate_pipeline',
    default_args=default_args,
    description='Run BigQuery aggregate query every two hours',
    schedule_interval='0 */2 * * *',
    start_date=days_ago(1),
    catchup=False,
)

aggregate_query = """
# Your SQL aggregate transformation query
"""

bq_aggregate_task = BigQueryInsertJobOperator(
    task_id='bq_aggregate',
    project_id='your_project_id',
    configuration={
        "query": {
            "query": aggregate_query,
            "useLegacySql": False,
        }
    },
    dag=dag
)
```

118. You are monitoring your organization's data lake hosted on BigQuery. The ingestion pipelines read data from Pub/Sub and write the data into tables on BigQuery. After a new version of the ingestion pipelines is deployed, the daily stored data increased by 50%. The volumes of data in Pub/Sub remained the same and only

some tables had their daily partition data size doubled. You need to investigate and fix the cause of the data increase. What should you do?

- A. 1. Check for duplicate rows in the BigQuery tables that have the daily partition data size doubled.
  2. Schedule daily SQL jobs to deduplicate the affected tables.
  3. Share the deduplication script with the other operational teams to reuse if this occurs to other tables.

- B. 1. Check for code errors in the deployed pipelines.
  2. Check for multiple writing to pipeline BigQuery sink.
  3. Check for errors in Cloud Logging during the day of the release of the new pipelines.
  4. If no errors, restore the BigQuery tables to their content before the last release by using time travel.

- **C. 1. Check for duplicate rows in the BigQuery tables that have the daily partition data size doubled.**
  2. **Check the BigQuery Audit logs to find job IDs.**
  3. **Use Cloud Monitoring to determine when the identified Dataflow jobs started and the pipeline code version.**
  4. **When more than one pipeline ingests data into a table, stop all versions except the latest one.**

- D. 1. Roll back the last deployment.
  2. Restore the BigQuery tables to their content before the last release by using time travel.
  3. Restart the Dataflow jobs and replay the messages by seeking the subscription to the timestamp of the release.

- **Option B**: Although it includes checking for code errors and logging, it also suggests restoring the tables to their previous state without first isolating the cause. This could lead to data loss or further issues.

- **C.**

1. **Check for duplicate rows in the BigQuery tables that have the daily partition data size doubled**:
   - This can help identify if the increase is due to data duplication.
2. **Check the BigQuery Audit logs to find job IDs**:
   - Audit logs can provide insights on whether there were multiple or erroneous job executions writing to the affected tables.
3. **Use Cloud Monitoring to determine when the identified Dataflow jobs started and the pipeline code version**:
   - This allows you to correlate the deployment timestamp with the volume increase and identify any specific jobs causing the issue. This also helps verify if the right pipeline version is running.

4. **When more than one pipeline ingests data into a table, stop all versions except the latest one**:
   - Ensures only a single version of the pipeline is active, stopping any older pipelines that might be causing data duplication.

By following these steps, you can systematically identify the root cause of the data volume increase and address it appropriately. This approach looks for the actual error footprints, leverages logging and monitoring tools effectively, and takes corrective action based on identified issues.

119. You have a BigQuery dataset named “customers”. All tables will be tagged by using a Data Catalog tag template named “gdpr”. The template contains one mandatory field, “has_sensitive_data”, with a boolean value. All employees must be able to do a simple search and find tables in the dataset that have either true or false in the “has_sensitive_data’ field. However, only the Human Resources (HR) group should be able to see the data inside the tables for which “has_sensitive data” is true. You give the all employees group the bigquery.metadataViewer and bigquery.connectionUser roles on the dataset. You want to minimize configuration overhead. What should you do next?

- A. Create the “gdpr” tag template with private visibility. Assign the bigquery.dataViewer role to the HR group on the tables that contain sensitive data.
- B. Create the “gdpr” tag template with private visibility. Assign the datacatalog.tagTemplateViewer role on this tag to the all employees group, and assign the bigquery.dataViewer role to the HR group on the tables that contain sensitive data.
- **C. Create the “gdpr” tag template with public visibility. Assign the bigquery.dataViewer role to the HR group on the tables that contain sensitive data.**
- D. Create the “gdpr” tag template with public visibility. Assign the datacatalog.tagTemplateViewer role on this tag to the all employees group, and assign the bigquery.dataViewer role to the HR group on the tables that contain sensitive data.

1. **Public Visibility for Tag Template**:
   - **Public visibility** for the "gdpr" tag template means that all employees can search and find tables based on the `has_sensitive_data` field without needing additional roles like `datacatalog.tagTemplateViewer`.
2. **Role Assignments**:
   - **bigquery.dataViewer role to the HR group** on tables with sensitive data ensures that only the HR group can see the actual data in these sensitive tables.

119. You are creating the CI/CD cycle for the code of the directed acyclic graphs (DAGs) running in Cloud Composer. Your team has two Cloud Composer instances: one instance for development and another instance for production. Your team is using a Git repository to maintain and develop the code of the DAGs. You want to

deploy the DAGs automatically to Cloud Composer when a certain tag is pushed to the Git repository. What should you do?

- **A. 1. Use Cloud Build to copy the code of the DAG to the Cloud Storage bucket of the development instance for DAG testing.**
  2. If the tests pass, use Cloud Build to copy the code to the bucket of the production instance.
- B. 1. Use Cloud Build to build a container with the code of the DAG and the KubernetesPodOperator to deploy the code to the Google Kubernetes Engine (GKE) cluster of the development instance for testing.
  2. If the tests pass, use the KubernetesPodOperator to deploy the container to the GKE cluster of the production instance.
- C. 1. Use Cloud Build to build a container and the KubernetesPodOperator to deploy the code of the DAG to the Google Kubernetes Engine (GKE) cluster of the development instance for testing.
  2. If the tests pass, copy the code to the Cloud Storage bucket of the production instance.
- D. 1. Use Cloud Build to copy the code of the DAG to the Cloud Storage bucket of the development instance for DAG testing.
  2. If the tests pass, use Cloud Build to build a container with the code of the DAG and the KubernetesPodOperator to deploy the container to the Google Kubernetes Engine (GKE) cluster of the production instance.

https://cloud.google.com/composer/docs/dag-cicd-integration-guide

The Answer is A. Given that there are two instances (development and production) already available, and the goal is to deploy DAGs to Cloud Composer not entire composer infra build.

Explanation:

- This approach leverages Cloud Build to manage the deployment process.
- It first deploys the code to the Cloud Storage bucket of the development instance for testing purposes.
- If the tests are successful in the development environment, the same Cloud Build process is used to copy the code to the Cloud Storage bucket of the production instance.

B. GKE-based approach is not standard for Cloud Composer. C. GKE used for testing is unconventional for DAG deployments. D. Involves unnecessary GKE deployment for production. Testing DAGs should use Composer instances directly, not Kubernetes containers in GKE.

120. You have a BigQuery table that ingests data directly from a Pub/Sub subscription. The ingested data is encrypted with a Google-managed encryption key. You need to meet a new organization policy that requires you to use keys from a centralized Cloud Key Management Service (Cloud KMS) project to encrypt data at rest. What should you do?

A. Use Cloud KMS encryption key with Dataflow to ingest the existing Pub/Sub subscription to the existing BigQuery table.
B. Create a new BigQuery table by using customer-managed encryption keys (CMEK), and migrate the data from the old BigQuery table.
C. Create a new Pub/Sub topic with CMEK and use the existing BigQuery table by using Google-managed encryption key.
D. Create a new BigQuery table and Pub/Sub topic by using customer-managed encryption keys (CMEK), and migrate the data from the old BigQuery table.

1. **Compliance with New Policy**:
   - Creating a new BigQuery table with customer-managed encryption keys (CMEK) allows you to meet the new organizational policy of using keys from a centralized Cloud Key Management Service (Cloud KMS) project.
2. **Data Migration**:
   - Migrating the data from the old BigQuery table to the new BigQuery table ensures that all existing data also complies with the new encryption policy.
3. **Minimal Disruption**:
   - This approach allows you to continue using the existing Pub/Sub subscription without needing to reconfigure it. You only need to reroute the data to the new BigQuery table, ensuring continuity in data ingestion.

121. You created an analytics environment on Google Cloud so that your data scientist team can explore data without impacting the on-premises Apache Hadoop solution. The data in the on-premises Hadoop Distributed File System (HDFS) cluster is in Optimized Row Columnar (ORC) formatted files with multiple columns of Hive partitioning. The data scientist team needs to be able to explore the data in a similar way as they used the on-premises HDFS cluster with SQL on the Hive query engine. You need to choose the most cost-effective storage and processing solution. What should you do?

- A. Import the ORC files to Bigtable tables for the data scientist team.
- B. Import the ORC files to BigQuery tables for the data scientist team.
- C. Copy the ORC files on Cloud Storage, then deploy a Dataproc cluster for the data scientist team.
- **D. Copy the ORC files on Cloud Storage, then create external BigQuery tables for the data scientist team.**

A. Import the ORC files to Bigtable tables for the data scientist team:

- **Not Suitable for SQL Queries**: Bigtable is a NoSQL database optimized for large-scale read/write workloads but is not designed for ad-hoc SQL queries and analytics.

B. Import the ORC files to BigQuery tables for the data scientist team:

- **Data Ingestion Costs**: Importing the ORC files into BigQuery tables would incur additional costs for data ingestion. Using external tables avoids these costs.

C. Copy the ORC files on Cloud Storage, then deploy a Dataproc cluster for the data scientist team:

- **Higher Operational Costs**: Running a Dataproc cluster continuously can be more expensive and requires additional management effort. While it replicates the Hadoop environment closely, it's less cost-effective compared to using BigQuery with external tables.

▶ Option D explanation

122. You are designing a Dataflow pipeline for a batch processing job. You want to mitigate multiple zonal failures at job submission time. What should you do?

A. Submit duplicate pipelines in two different zones by using the -zone flag.
B. Set the pipeline staging location as a regional Cloud Storage bucket.
C. Specify a worker region by using the -region flag.
D. Create an Eventarc trigger to resubmit the job in case of zonal failure when submitting the job.

1. **Regional Scope**:
   - By specifying a worker region using the `-region` flag, you ensure that Dataflow can automatically use multiple zones within the specified region. This provides redundancy across zones within the region and mitigates the risk of zonal failures.
2. **Automatic Handling of Zonal Failures**:
   - When you specify a region, Dataflow takes advantage of regional availability and can distribute resources across multiple zones within that region. This automatic distribution improves resiliency against zone-specific outages.
3. **Simplicity and Effectiveness**:
   - This option is simpler and more effective than submitting duplicate pipelines or setting up complex event triggers. It leverages the built-in regional capabilities of Dataflow for high availability.

Implementation Steps:

- When you submit the Dataflow job, use the `-region` flag to specify a region instead of a specific zone.
  Example command:

```
gcloud dataflow jobs run job-name \
    --gcs-location=gs://your-bucket-name/templates/your-template \
    --region=us-central1
```

By specifying a region (e.g., `us-central1`), you ensure that the job can utilize multiple zones within that region, providing better resilience against zonal failures.

Why Not the Other Options:

- **Option A (Submit duplicate pipelines in two different zones by using the -zone flag)**:
  - Submitting duplicate pipelines is not efficient and can lead to unnecessary resource consumption and higher costs. Additionally, it complicates resource management and monitoring.
- **Option B (Set the pipeline staging location as a regional Cloud Storage bucket)**:
  - While using a regional Cloud Storage bucket improves the availability of the staging location, it doesn't address the issue of zonal failures for the Dataflow workers.
- **Option D (Create an Eventarc trigger to resubmit the job in case of zonal failure when submitting the job)**:
  - Setting up an Eventarc trigger for job resubmission adds complexity and may introduce delays in job execution. It's more reliable and straightforward to use the regional capabilities of Dataflow to handle zonal failures automatically.
  Therefore, specifying a worker region using the `-region` flag is the most effective way to mitigate multiple zonal failures at job submission time in a Dataflow pipeline for batch processing.

123. You are designing a real-time system for a ride hailing app that identifies areas with high demand for rides to effectively reroute available drivers to meet the demand. The system ingests data from multiple sources to Pub/Sub, processes the data, and stores the results for visualization and analysis in real-time dashboards. The data sources include driver location updates every 5 seconds and app-based booking events from riders. The data processing involves real-time aggregation of supply and demand data for the last 30 seconds, every 2 seconds, and storing the results in a low-latency system for visualization. What should you do? link

- A. Group the data by using a tumbling window in a Dataflow pipeline, and write the aggregated data to Memorystore.
- **B. Group the data by using a hopping window in a Dataflow pipeline, and write the aggregated data to Memorystore.**
- C. Group the data by using a session window in a Dataflow pipeline, and write the aggregated data to BigQuery.
- D. Group the data by using a hopping window in a Dataflow pipeline, and write the aggregated data to BigQuery.

124. Your car factory is pushing machine measurements as messages into a Pub/Sub topic in your Google Cloud project. A Dataflow streaming job, that you wrote with the Apache Beam SDK, reads these messages, sends acknowledgment to Pub/Sub, applies some custom business logic in a DoFn instance, and writes the result to

BigQuery. You want to ensure that if your business logic fails on a message, the message will be sent to a Pub/Sub topic that you want to monitor for alerting purposes. What should you do?

- A. Enable retaining of acknowledged messages in your Pub/Sub pull subscription. Use Cloud Monitoring to monitor the subscription/num_retained_acked_messages metric on this subscription.
- **B. Use an exception handling block in your Dataflow's DoFn code to push the messages that failed to be transformed through a side output and to a new Pub/Sub topic. Use Cloud Monitoring to monitor the topic/num_unacked_messages_by_region metric on this new topic.**
- C. Enable dead lettering in your Pub/Sub pull subscription, and specify a new Pub/Sub topic as the dead letter topic. Use Cloud Monitoring to monitor the subscription/dead_letter_message_count metric on your pull subscription.
- D. Create a snapshot of your Pub/Sub pull subscription. Use Cloud Monitoring to monitor the snapshot/num_messages metric on this snapshot.
  - **Exception Handling in DoFn:** By implementing an exception block within your DoFn code, you can specifically capture messages that fail your business logic processing.
  - **Side Output:** Use a side output in your DoFn to channel these failed messages to a separate output stream.
  - **New Pub/Sub Topic:** Write the failed messages to a new Pub/Sub topic dedicated for monitoring purposes. This allows you to isolate and track these messages for further analysis or alerting.
  - **Cloud Monitoring:** Monitor the `topic/num_unacked_messages_by_region` metric on the new Pub/Sub topic using Cloud Monitoring. This metric provides insights into the number of unacknowledged messages, indicating potential issues in your business logic processing.

B. Use an exception handling block in your Dataflow's DoFn code to push the messages that failed to be transformed through a side output and to a new Pub/Sub topic. Use Cloud Monitoring to monitor the topic/num_unacked_messages_by_region metric on this new topic.

Here's why this option is the most appropriate:

1. **Custom Exception Handling**:
   - By using a side output in your `DoFn` for error handling, you can catch exceptions and route the failed messages to a separate Pub/Sub topic. This provides you with flexibility to apply custom business logic and handle errors gracefully within your Dataflow pipeline.
2. **Monitoring**:
   - Sending failed messages to a new Pub/Sub topic allows you to specifically monitor this topic for any anomalies. You can use Cloud Monitoring to track metrics such

as `topic/num_unacked_messages_by_region`, which will help you identify if there is an increase in failed messages.

3. **Decoupling of Failure Processing**:
   - This approach keeps the failure-handling logic separate from the main data pipeline. By routing failed messages to another topic, you ensure that the main processing pipeline remains clean and focused on its primary task.

Implementation Steps:

1. **Modify Your DoFn to Handle Exceptions**:
   - Implement an exception handling block in the `DoFn` to catch errors and use a side output to send failed messages to a new Pub/Sub topic.

Example code snippet:

```
from apache_beam import DoFn, PCollection, ParDo, Pipeline
from apache_beam.io import WriteToBigQuery, WriteToPubSub
from apache_beam.options.pipeline_options import PipelineOptions

class CustomDoFn(DoFn):
    def __init__(self):
        self.failed_messages_tag = 'failed_messages'

    def process(self, element):
        try:
            # Apply business logic
            result = process_message(element)
            yield result
        except Exception as e:
            # Send failed message to side output
            yield pvalue.TaggedOutput(self.failed_messages_tag, element)

def run():
    options = PipelineOptions( ... )
    with Pipeline(options=options) as p:
        messages = (p
                    | 'ReadFromPubSub' >>
ReadFromPubSub(subscription='projects/your-project/subscriptions/your-
subscription')
                    | 'ApplyDoFn' >>
ParDo(CustomDoFn()).with_outputs(CustomDoFn().failed_messages_tag,
main='main'))

        main = messages['main']
        failed_messages = messages[CustomDoFn().failed_messages_tag]
        main | 'WriteToBigQuery' >> WriteToBigQuery(table='your-
project:your_dataset.your_table')
        failed_messages | 'WriteToPubSub' >>
WriteToPubSub(topic='projects/your-project/topics/failed-messages-topic'))

if __name__ == '__main__':
    run()
```

2. **Set Up Monitoring**:
    - Use Cloud Monitoring to monitor the new Pub/Sub topic where failed messages are sent. You can track the `topic/num_unacked_messages_by_region` metric for this topic to set up alerts and get notified when there are issues.

This approach ensures that any messages that fail to process correctly are rerouted for further inspection and potential reprocessing, while also keeping your main data pipeline efficient and focused.

125. You want to store your team's shared tables in a single dataset to make data easily accessible to various analysts. You want to make this data readable but unmodifiable by analysts. At the same time, you want to provide the analysts with individual workspaces in the same project, where they can create and store tables for their own use, without the tables being accessible by other analysts. What should you do?

- A. Give analysts the BigQuery Data Viewer role at the project level. Create one other dataset, and give the analysts the BigQuery Data Editor role on that dataset.
- B. Give analysts the BigQuery Data Viewer role at the project level. Create a dataset for each analyst, and give each analyst the BigQuery Data Editor role at the project level.
- **C. Give analysts the BigQuery Data Viewer role on the shared dataset. Create a dataset for each analyst, and give each analyst the BigQuery Data Editor role at the dataset level for their assigned dataset.**
- D. Give analysts the BigQuery Data Viewer role on the shared dataset. Create one other dataset and give the analysts the BigQuery Data Editor role on that dataset.
- Data Viewer on Shared Dataset: Grants read-only access to the shared dataset.
- Data Editor on Individual Datasets: Giving each analyst Data Editor role on their respective dataset creates private workspaces where they can create and store personal tables without exposing them to other analysts.

126. You are running a streaming pipeline with Dataflow and are using hopping windows to group the data as the data arrives. You noticed that some data is arriving late but is not being marked as late data, which is resulting in inaccurate aggregations

downstream. You need to find a solution that allows you to capture the late data in the appropriate window. What should you do?

- **A. Use watermarks to define the expected data arrival window. Allow late data as it arrives.**
- B. Change your windowing function to tumbling windows to avoid overlapping window periods.
- C. Change your windowing function to session windows to define your windows based on certain activity.
- D. Expand your hopping window so that the late data has more time to arrive within the grouping.

https://cloud.google.com/dataflow/docs/concepts/streaming-pipelines#watermarks

```
- Watermarks: Watermarks in a streaming pipeline are used to specify the
point in time when Dataflow expects all data up to that point to have
arrived.
- Allow Late Data: configure the pipeline to accept and correctly process
data that arrives after the watermark, ensuring it's captured in the
appropriate window.
```

127. You work for a large ecommerce company. You store your customer's order data in Bigtable. You have a garbage collection policy set to delete the data after 30 days and the number of versions is set to 1. When the data analysts run a query to report total customer spending, the analysts sometimes see customer data that is older than 30 days. You need to ensure that the analysts do not see customer data older than 30 days while minimizing cost and overhead. What should you do?

- A. Set the expiring values of the column families to 29 days and keep the number of versions to 1.
- **B. Use a timestamp range filter in the query to fetch the customer's data for a specific range.**
- C. Schedule a job daily to scan the data in the table and delete data older than 30 days.
- D. Set the expiring values of the column families to 30 days and set the number of versions to 2.

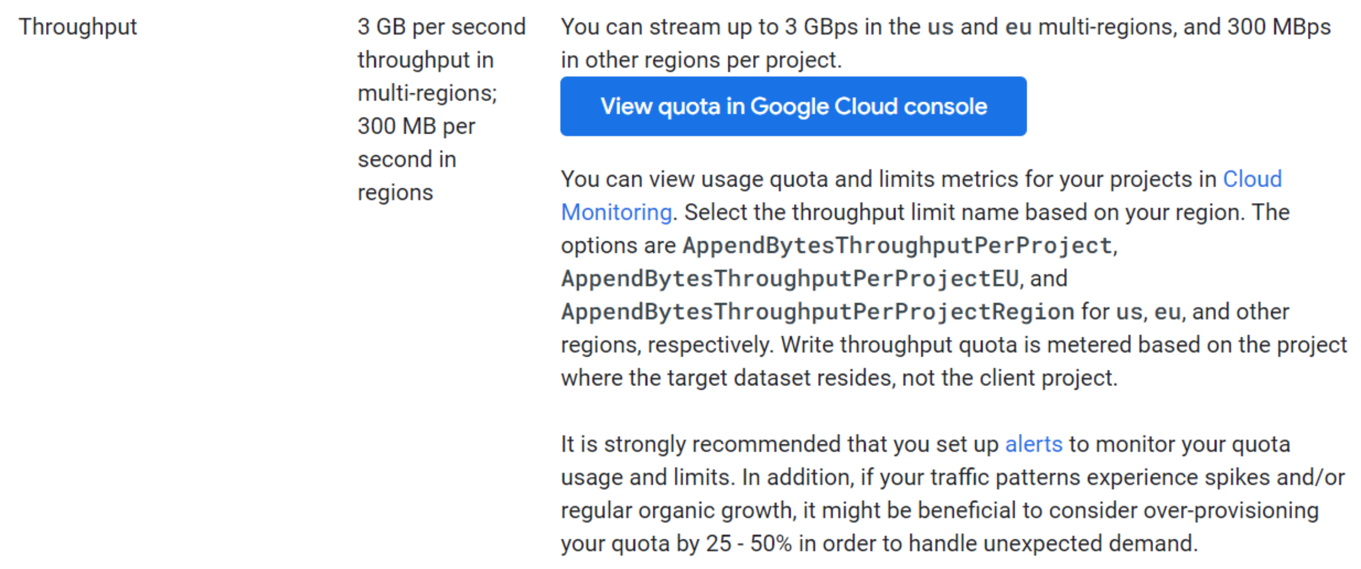
https://cloud.google.com/bigtable/docs/garbage-collection

128. You are using a Dataflow streaming job to read messages from a message bus that does not support exactly-once delivery. Your job then applies some transformations, and loads the result into BigQuery. You want to ensure that your data is being streamed into BigQuery with exactly-once delivery semantics. You

expect your ingestion throughput into BigQuery to be about 1.5 GB per second. What should you do?

- A. Use the BigQuery Storage Write API and ensure that your target BigQuery table is regional.
- **B. Use the BigQuery Storage Write API and ensure that your target BigQuery table is multiregional.**
- C. Use the BigQuery Streaming API and ensure that your target BigQuery table is regional.
- D. Use the BigQuery Streaming API and ensure that your target BigQuery table is multiregional.

https://cloud.google.com/bigquery/quotas#write-api-limits
https://cloud.google.com/bigquery/docs/write-api#advantages



Throughput	3 GB per second throughput in multi-regions; 300 MB per second in regions	You can stream up to 3 GBps in the us and eu multi-regions, and 300 MBps in other regions per project. View quota in Google Cloud console You can view usage quota and limits metrics for your projects in Cloud Monitoring. Select the throughput limit name based on your region. The options are `AppendBytesThroughputPerProject`, `AppendBytesThroughputPerProjectEU`, and `AppendBytesThroughputPerProjectRegion` for us, eu, and other regions, respectively. Write throughput quota is metered based on the project where the target dataset resides, not the client project. It is strongly recommended that you set up alerts to monitor your quota usage and limits. In addition, if your traffic patterns experience spikes and/or regular organic growth, it might be beneficial to consider over-provisioning your quota by 25 - 50% in order to handle unexpected demand.

129. You have created an external table for Apache Hive partitioned data that resides in a Cloud Storage bucket, which contains a large number of files. You notice that queries against this table are slow. You want to improve the performance of these queries. What should you do?

- A. Change the storage class of the Hive partitioned data objects from Coldline to Standard.
- B. Create an individual external table for each Hive partition by using a common table name prefix. Use wildcard table queries to reference the partitioned data.
- **C. Upgrade the external table to a BigLake table. Enable metadata caching for the table.**
- D. Migrate the Hive partitioned data objects to a multi-region Cloud Storage bucket.

```
- BigLake Table: BigLake allows for more efficient querying of data lakes
stored in Cloud Storage. It can handle large datasets more effectively than
standard external tables.
- Metadata Caching: Enabling metadata caching can significantly improve
query performance by reducing the time taken to read and process metadata
from a large number of files.
```

https://cloud.google.com/bigquery/docs/external-data-cloud-storage#upgrade-external-tables-to-biglake-tables
https://cloud.google.com/bigquery/docs/biglake-intro#metadata_caching_for_performance

130. You have a network of 1000 sensors. The sensors generate time series data: one metric per sensor per second, along with a timestamp. You already have 1 TB of data, and expect the data to grow by 1 GB every day. You need to access this data in two ways. The first access pattern requires retrieving the metric from one specific sensor stored at a specific timestamp, with a median single-digit millisecond latency. The second access pattern requires running complex analytic queries on the data, including joins, once a day. How should you store this data?

- A. Store your data in BigQuery. Concatenate the sensor ID and timestamp, and use it as the primary key.
- **B. Store your data in Bigtable. Concatenate the sensor ID and timestamp and use it as the row key. Perform an export to BigQuery every day.**
- C. Store your data in Bigtable. Concatenate the sensor ID and metric, and use it as the row key. Perform an export to BigQuery every day.
- D. Store your data in BigQuery. Use the metric as a primary key.
  - Bigtable excels at incredibly fast lookups by row key, often reaching single-digit millisecond latencies.
  - Constructing the row key with sensor ID and timestamp enables efficient retrieval of specific sensor readings at exact timestamps.
  - Bigtable's wide-column design effectively stores time series data, allowing for flexible addition of new metrics without schema changes.
  - Bigtable scales horizontally to accommodate massive datasets (petabytes or more), easily handling the expected data growth.

131. You have 100 GB of data stored in a BigQuery table. This data is outdated and will only be accessed one or two times a year for analytics with SQL. For backup purposes, you want to store this data to be immutable for 3 years. You want to minimize storage costs. What should you do?

- A. 1. Create a BigQuery table clone.
  2. Query the clone when you need to perform analytics.
- B. 1. Create a BigQuery table snapshot.
  2. Restore the snapshot when you need to perform analytics.
- C. 1. Perform a BigQuery export to a Cloud Storage bucket with archive storage class.
  2. Enable versioning on the bucket.
  3. Create a BigQuery external table on the exported files.
- **D. 1. Perform a BigQuery export to a Cloud Storage bucket with archive storage class.**
  2. Set a locked retention policy on the bucket.
  3. Create a BigQuery external table on the exported files.

132. You have thousands of Apache Spark jobs running in your on-premises Apache Hadoop cluster. You want to migrate the jobs to Google Cloud. You want to use managed services to run your jobs instead of maintaining a long-lived Hadoop cluster yourself. You have a tight timeline and want to keep code changes to a minimum. What should you do?

- A. Move your data to BigQuery. Convert your Spark scripts to a SQL-based processing approach.
- B. Rewrite your jobs in Apache Beam. Run your jobs in Dataflow.
- C. Copy your data to Compute Engine disks. Manage and run your jobs directly on those instances.
- **D. Move your data to Cloud Storage. Run your jobs on Dataproc.**

133. You are administering shared BigQuery datasets that contain views used by multiple teams in your organization. The marketing team is concerned about the variability of their monthly BigQuery analytics spend using the on-demand billing model. You need to help the marketing team establish a consistent BigQuery analytics spend each month. What should you do?link

- A. Create a BigQuery Enterprise reservation with a baseline of 250 slots and autoscaling set to 500 for the marketing team, and bill them back accordingly.
- B. Establish a BigQuery quota for the marketing team, and limit the maximum number of bytes scanned each day.
- **C. Create a BigQuery reservation with a baseline of 500 slots with no autoscaling for the marketing team, and bill them back accordingly.**
- D. Create a BigQuery Standard pay-as-you go reservation with a baseline of 0 slots and autoscaling set to 500 for the marketing team, and bill them back accordingly.

https://cloud.google.com/bigquery/quotas#query_jobs

```
gcloud bigquery reservations create marketing-reservation \
    --slot-capacity=250 \
    --autoscale-max-capacity=500 \
    --location=your-location
```

134. You are part of a healthcare organization where data is organized and managed by respective data owners in various storage services. As a result of this decentralized ecosystem, discovering and managing data has become difficult. You need to quickly identify and implement a cost-optimized solution to assist your organization with the following:

• Data management and discovery
• Data lineage tracking
• Data quality validation

How should you build the solution?

- A. Use BigLake to convert the current solution into a data lake architecture.
- B. Build a new data discovery tool on Google Kubernetes Engine that helps with new source onboarding and data lineage tracking.
- C. Use BigQuery to track data lineage, and use Dataprep to manage data and perform data quality validation.
- **D. Use Dataplex to manage data, track data lineage, and perform data quality validation.**

135. You have data located in BigQuery that is used to generate reports for your company. You have noticed some weekly executive report fields do not correspond to format according to company standards. For example, report errors include different telephone formats and different country code identifiers. This is a frequent issue, so you need to create a recurring job to normalize the data. You want a quick solution that requires no coding. What should you do?

- A. Use Cloud Data Fusion and Wrangler to normalize the data, and set up a recurring job.
- B. Use Dataflow SQL to create a job that normalizes the data, and that after the first run of the job, schedule the pipeline to execute recurrently.
- C. Create a Spark job and submit it to Dataproc Serverless.
- D. Use BigQuery and GoogleSQL to normalize the data, and schedule recurring queries in BigQuery.

1. **No Coding Required**:
   - Cloud Data Fusion provides a visual interface for data integration, which allows you to create data transformation pipelines without writing code. Wrangler, a feature within Cloud Data Fusion, makes it easy to clean and normalize data using a drag-and-drop interface.
2. **Ease of Use**:

- Cloud Data Fusion is designed for simplicity and ease of use, making it suitable for users who may not have extensive coding skills but need to perform data transformations.

3. **Recurring Job Configuration**:
   - Cloud Data Fusion allows you to schedule pipelines to run at regular intervals, providing an automated solution for recurring tasks.