Contents

[Question 1 3](#_Toc156620597)

[Question 2 4](#_Toc156620598)

[Question 3 6](#_Toc156620599)

[Question 4 7](#_Toc156620600)

[Question 5: 8](#_Toc156620601)

[Question 6 9](#_Toc156620602)

[Question 7: 10](#_Toc156620603)

[Question 8: 11](#_Toc156620604)

[Question 9 12](#_Toc156620605)

[Question 10: 13](#_Toc156620606)

[Question 11: 14](#_Toc156620607)

[Question 12: 15](#_Toc156620608)

[Question 13: 17](#_Toc156620609)

[Question 14: 18](#_Toc156620610)

[Question 15: 19](#_Toc156620611)

[Question 16: 20](#_Toc156620612)

[Question 17: 21](#_Toc156620613)

[Question 18: 22](#_Toc156620614)

[Question 19: 23](#_Toc156620615)

[Question 20: 24](#_Toc156620616)

[Question 21: 25](#_Toc156620617)

[Question 22 27](#_Toc156620618)

[Question 23 29](#_Toc156620619)

[Question 24: 30](#_Toc156620620)

[Question 25: 31](#_Toc156620621)

[Question 26: 32](#_Toc156620622)

[Question 27: 33](#_Toc156620623)

[Question 28: 34](#_Toc156620624)

[Question 29: 36](#_Toc156620625)

[Question 30: 38](#_Toc156620626)

[Question 31: 39](#_Toc156620627)

[Question 32: 40](#_Toc156620628)

[Question 33: 41](#_Toc156620629)

[Question 34: 42](#_Toc156620630)

[Question 35: 43](#_Toc156620631)

[Question 36: 44](#_Toc156620632)

[Question 37 45](#_Toc156620633)

[Question 38 46](#_Toc156620634)

[Question 39: 48](#_Toc156620635)

[Question 40: 49](#_Toc156620636)

[Question 41: 50](#_Toc156620637)

[Question 42: 51](#_Toc156620638)

[Question 43: 52](#_Toc156620639)

[Question 44: 53](#_Toc156620640)

[Question 45: 54](#_Toc156620641)

[Question 46: 56](#_Toc156620642)

[Question 47 59](#_Toc156620643)

[Question 48: 60](#_Toc156620644)

[Question 49: 61](#_Toc156620645)

[Question 50: 62](#_Toc156620646)

[Question 51: 64](#_Toc156620647)

[Question 52: 65](#_Toc156620648)

[Question 53: 66](#_Toc156620649)

[Question 54: 67](#_Toc156620650)

[Question 55: 68](#_Toc156620651)

[Question 56: 69](#_Toc156620652)

[Question 57 69](#_Toc156620653)

[Question 58 70](#_Toc156620654)

[Question 59: 71](#_Toc156620655)

[Question 60: 73](#_Toc156620656)

## Question 1

A data analyst is running a shell script in all the notebooks attached to the cluster. The shell script contains a long set of commands which is taking a lot of time to complete. As a data engineer, which of the following statements will you suggest to the data analyst?

* **Use the init script to execute the shell script faster**
* **Run the script as the Workspace admin**
* **Use %md to run the script faster**
* **Increase the number of worker nodes to speed up the script**
* **Run the notebook using Databricks API**

**Explanation**

Let us take a look at all the options one by one.

*Use the init script to execute the shell script faster*

**CORRECT!** By using **%sh** to run a shell script in a Databricks notebook, you end up using **only the driver node** of the cluster. If you need to run the shell script on the **worker nodes as well**, you will need to **add it as an init script** on the cluster.

*Run the script as the Workspace admin*

**INCORRECT!** The execution time of the script does not depend on the user who runs the script.

*Use %md to run the script faster*

**INCORRECT!** **%md** is a **magic command** supported in Databricks which is used for the Markdown language.

*Increase the number of worker nodes to speed up the script*

**INCORRECT!** As the **%sh** command utilizes **only the driver node** of the cluster, increasing the worker nodes will have **no effect on the speed** of the shell script.

*Run the notebook using Databricks API*

**INCORRECT!** The speed of the shell script **does not depend** on the **triggering source** of a notebook.

More Info: [Init Script in Databricks](https://docs.databricks.com/notebooks/notebooks-code.html#:~:text=This%20command%20runs%20only%20on%20the%20Apache%20Spark%20driver%2C%20and%20not%20the%20workers.%20To%20run%20a%20shell%20command%20on%20all%20nodes%2C%20use%20an%20init%20script)

## Question 2

Which of the following is a valid response to a JSON workload passed to **2.0/jobs/create** endpoint of Databricks REST API?

* {

"job\_name": "name"

}

* {

    "status": 200

}

* {

    "job\_id": 13746

}

* {

"job\_id": "13746"

}

* {

job\_id: 13746

}

**Explanation**

Knowing the **response structure** of the **2.0/jobs/create** endpoint makes it easy to answer these types of questions in the actual exam. The syntax is as follows:

**Key:**"job\_id"      **Value Type:**INT64

So, the correct response should be:

1. {
2. "job\_id": 13746
3. }

More Info: [Databricks REST API 2.0/jobs/create endpoint](https://docs.databricks.com/dev-tools/api/2.0/jobs.html#response-structure)

## Question 3

A data engineer wants to unmount a mount point mounted at **/mnt/mountPoint1** Which of the following commands can be used by the data engineer?

* + **dbutils.unmount("/mnt/mountPoint1")**
  + **dbutils.mount.unmount("/mnt/mountPoint1")**
  + **dbutils.fs.unmount("/mnt/mountPoint1")**
  + **dbutils.widgets("/mnt/mountPoint1")**
  + **dbutils.remove\_mount("/mnt/mountPoint1")**

**Explanation**

This is relatively a simple question to answer. **dbutils.fs.unmount()** can be used to **unmount a mount point**in Databricks.

More Info: [Unmount in Databricks](https://docs.databricks.com/dbfs/mounts.html#:~:text=Copy-,dbutils.fs.unmount(%22/mnt/%3Cmount%2Dname%3E%22),-Important)

## Question 4

A Databricks admin has created a Cluster and provided *Can Restart* permission to a group of users named **data\_analysts** Which of the following cluster-related actions cannot be performed by the members of the **data\_analysts** group?

* + **Terminate the cluster**
  + **View Spark UI**
  + **Restart the cluster**
  + **View the cluster metrics**
  + **Edit the cluster**

**Explanation**

Cluster-level permission is usually set to limit the accessibility to a cluster. There are **4 different permission levels**that can be set to a cluster. Each permission level enables a different set of cluster actions. The following table lists the permissions granted at each level

Source: [Databricks](https://docs.databricks.com/security/access-control/cluster-acl.html#cluster-level-permissions)

According to the table, the **data\_analysts** group **cannot Edit the cluster** as it requires **Can Manage** permissions on the cluster.

More Info: [Cluster Level Permissions](https://docs.databricks.com/security/access-control/cluster-acl.html#cluster-level-permissions)

Bottom of Form

## Question 5:

A data engineer has created a new job with a single notebook task intended to run daily at 5 PM. The task intends to run the notebook named **show\_regular.py** which contains 5 cells where each cell outputs data in the form of a PySpark DataFrame. The size of the outputs are as follows:

**Cell 1:** 5.2 MB

**Cell 2:** 6.1 MB

**Cell 3:** 4.4 MB

**Cell 4:** 5.3 MB

**Cell 5:** 2.5 MB

The job fails on its initial run and the data engineer is clueless about the failure. Which of the data constraint forced the job to fail?

* + **The individual cell output is limited to 6 MB which caused the job to fail.**
  + **The output size of all the cells in a notebook should not exceed 20 MB. Since the total size of all the cells combined is more than 20 MB, the job failed.**
  + **The output size of the first cell cannot exceed 5 MB in size, forcing the job to fail.**
  + **The job failed due to multiple reasons as the size of any individual cell output in a notebook cannot increase by 6 MB and the total size of all the outputs cannot be more than 18 MB.**
  + **The reason for the job failure cannot be determined solely by the size of the output as there is no limitation on the output size of the cells in a notebook.**

**Explanation**

In Databricks, when you add a notebook task to a job, the size limit can force the job to fail. The size of an **individual cell cannot exceed 8 MB** whereas the **total size of all the cells cannot be more than 20 MB**.

In the above question, none of the cells exceeds 8 MB but the **total size of all the cells is greater than 20 MB** which is the reason for the job failure.

More Info: [Size Limitation for Notebook cells](https://learn.microsoft.com/en-us/azure/databricks/kb/jobs/job-cluster-limit-nb-output#cause)

## Question 6

While creating a cluster for the engineering team, the Databricks admin added the **ENV=PROD** in Environment variables. The engineering team is using this environment variable for fetching data from the S3 bucket in the **fetch\_details** notebook.

A new member joins the team and needs to fetch data from the **DEV** environment instead of **PROD**. The member runs the following command in a new notebook to update the environment variable:

**import os**

**os.environ["ENV"] = 'DEV'**

Which of the following statements describes the effect of the command?

* **The environment variable’s value will be changed in the newly created notebook and the fetch\_details notebook will also start using DEV.**
* **The command will fail with an error message as the environment variable cannot be changed.**
* **The command will be executed successfully but the environment variable will not be changed.**
* **The environment variable’s value will be changed but the value in all other notebooks will remain PROD.**
* **The environment variable’s value will be updated in the cluster properties and all the notebooks attached to the cluster will start using the updated value.**

**Explanation**

The environment variables can be defined while **creating or updating a cluster**. These environment variables can be accessed using the following statements:

**import os**

**os.environ[“ENV”]**

Once the variables are defined, they can be changed but the change will not reflect in other notebooks. If an environment variable is **changed in notebook A**, the effect of this change **cannot** be experienced in **notebook B**.

[Static Notebook](https://certification-champs.github.io/professional-data-engineer-notebooks/Practice%20Test%201/Question%206.html) | [Static Notebook-2](https://certification-champs.github.io/professional-data-engineer-notebooks/Practice%20Test%201/Question%206-2.html)

*To view all the notebooks in this course, download [Notebooks.dbc](https://certification-champs.github.io/professional-data-engineer-notebooks/Notebooks.dbc) and import to your Databricks Account.*

More Info: [Environment variables in Databricks Clusters](https://docs.databricks.com/clusters/configure.html#environment-variables)

## Question 7:

A team of data engineers started a project seven years ago. Two tables were created with the following SQL statements:

CREATE TABLE flights

(flight\_number STRING, source STRING, destination STRING)

using DELTA;

CREATE TABLE airports

(airport\_id STRING, city STRING, country STRING)

USING PARQUET

LOCATION 'dbfs:/airports/data/';

The DBFS location **dbfs:/airports/data/** contains static data for the airports while the data in the **flights** table is updated on a daily basis. After the creation of the tables, the definition of the tables was not altered. A junior data engineer joins the team and runs the following SQL statements.

1. DROP TABLE flights;
2. DROP TABLE airports;

Which of the following statements is true about the data present in the tables? Assume that the **UNDROP** command, recently launched by Databricks, cannot be used.

* **The data in both tables can be retrieved by using the Time Travel feature of Delta tables.**
* **The tables cannot be dropped without deleting all the records.**
* **The data in the flights table can be recovered using Time Travel but the data in the airports table cannot be recovered.**
* **The data in airports table can be accessed at the DBFS location and the data in the flights table can be recovered using the Time Travel feature.**
* **The data in flights table cannot be recovered but the data in the airports table can be accessed at the DBFS location.**

**Explanation**

Let us take a look at the two tables given in the question - **flights** and **airports**

The **flights** table is a **managed table** which means that the **data will be deleted as you drop the table**. Hence, once the table is dropped, the data cannot be recovered.

The **airports** table is an **unmanaged(external)** table which means that the **data will be stored at an external location**. Once the table is dropped, the data can always be accessed at the external location.

[Static Notebook](https://certification-champs.github.io/professional-data-engineer-notebooks/Practice%20Test%201/Question%207.html)

*To view all the notebooks in this course, download [Notebooks.dbc](https://certification-champs.github.io/professional-data-engineer-notebooks/Notebooks.dbc) and import to your Databricks Account.*

More Info: [Dropping Managed vs Unmanaged table](https://docs.databricks.com/data-governance/unity-catalog/create-tables.html#:~:text=When%20you%20run%20DROP%20TABLE%20on%20an%20external%20table%2C%20Unity%20Catalog%20does%20not%20delete%20the%20underlying%20data)

## Question 8:

Which of the following statements depicts the correct usage of **2.0/jobs/update** and **2.0/jobs/reset** endpoints of the Databricks REST API?

* **2.0/jobs/reset is used to overwrite an existing job with its default settings whereas 2.0/jobs/update is used to add, change or remove specific settings of an existing job.**
* **2.0/jobs/reset is used to overwrite all the settings of an existing job with the settings passed in the JSON payload whereas 2.0/jobs/update is used to update the ID of an existing job.**
* **2.0/jobs/reset is used to remove all the jobs with a specific name from the Workspace whereas 2.0/jobs/update is used to add, change or remove specific settings of an existing job.**
* **2.0/jobs/reset is used to overwrite all the settings of an existing job with the settings passed in the JSON payload whereas 2.0/jobs/update is used to add, change or remove specific settings of an existing job.**
* **2.0/jobs/reset is used to delete the details of all the previous job runs of an existing job whereas 2.0/jobs/update is used to update the ID of an existing job.**

**Explanation**

**2.0/jobs/reset** and **2.0/jobs/update** endpoints are **both** used to **change the settings** of an **existing job**.

If you need to update **some of the settings** like adding, changing, or removing, **2.0/jobs/update** should be used. If **all the settings** need to be overwritten with the new ones, **2.0/jobs/reset** should be used.

More Info: [Overwriting all settings](https://docs.databricks.com/dev-tools/api/2.0/jobs.html#reset:~:text=Overwrite%20all%20settings%20for%20a%20specific%20job) | [Updating specific settings](https://docs.databricks.com/dev-tools/api/2.0/jobs.html#update:~:text=Add%2C%20change%2C%20or%20remove%20specific%20settings%20of%20an%20existing%20job)

## Question 9

Which of the following statements about interoperability between supported languages in Databricks notebooks is true?

* **Variables declared in the Python language cells can only be accessed in cells with default language as Python or Scala.**
* **Variables declared in Python can be used in cells with default language as Python, Scala or R.**
* **Variables declared in one language can be used in all other languages as Databricks supports full interoperability.**
* **Variables declared in one language cannot be accessed in the cells using any other language.**
* **SQL variables are the only ones that can be used in cells having other languages.**

**Explanation**

This is one of the **simplest** questions of the exam. In simple words, Databricks does **not** support **interoperability** between different languages(*in terms of variables declared in different languages*) which means that a **variable** declared in one language **cannot** be accessed through other languages.

[Static Notebook](https://certification-champs.github.io/professional-data-engineer-notebooks/Practice%20Test%201/Question%209.html)

*To view all the notebooks in this course, download [Notebooks.dbc](https://certification-champs.github.io/professional-data-engineer-notebooks/Notebooks.dbc) and import to your Databricks Account.*

Explanation: [Language interoperability in Databricks](https://docs.databricks.com/notebooks/notebooks-code.html#:~:text=Variables%20defined%20in%20one%20language%20(and%20hence%20in%20the%20REPL%20for%20that%20language)%20are%20not%20available%20in%20the%20REPL%20of%20another%20language)

## Question 10:

An engineering team has a lot of members ranging from data analysts to data scientists. Some of the team members use an S3 bucket to access the data but some of the new users are not familiar with AWS S3. The team decides to mount the S3 bucket to DBFS. Which of the following methods should be used by the team if they need to give read-only access to the mounted S3 bucket?

* **Mount the S3 bucket using AWS keys - access\_key and secret\_key**
* **Add readOnly = True in dbutils.fs.mount()**
* **Mounting of S3 bucket with read-only access is not possible in Databricks**
* **Mount the S3 bucket using the AWS instance profile**
* **Select the readOnly option while mounting the S3 bucket**

**Explanation**

Let us look at all the options one by one

*Mount the S3 bucket using AWS keys -****access\_key****and****secret\_key***

**INCORRECT!** When you mount an **S3 bucket** using **AWS keys**, all the users that have access to that mount point **can** **read and write** to the S3 bucket.

*Add****readOnly = True****in****dbutils.fs.mount()***

**INCORRECT!** There is no **readOnly** parameter in **dbutils.fs.mount()**

*Mounting of S3 bucket with read-only access is not possible in Databricks*

**INCORRECT!** You can mount an S3 bucket in Databricks with **only read access** using the **AWS Instance Profile**.

*Mount the S3 bucket using the AWS instance profile*

**CORRECT!** If you need to **restrict access** to the mounted S3 bucket, you need to mount the S3 bucket using the **AWS Instance Profile**. The process includes configuring your cluster with an **Instance Profile** first and then mounting the bucket. The permissions granted to the **mounted S3 bucket** will be equivalent to the permissions acquired by the **AWS Instance Profile**.

*Select****readOnly****option while mounting the S3 bucket*

**INCORRECT!** There is no **readOnly** option available while **mounting** an S3 bucket.

More Info: [Mounting an S3 Bucket in DBFS](https://docs.databricks.com/dbfs/mounts.html#mount-an-s3-bucket:~:text=You%20can%20use,the%20AssumeRole%20policy)

## Question 11:

Which of the following is correct about the Package cells in a Databricks notebook?

* + **A package cell is compiled when executed.**
  + **A package cell contains more than one class/object.**
  + **A package cell contains data from more than one cell.**
  + **A package cell returns an executable file when executed.**
  + **A package cell can only work with Python classes.**

**Explanation**

A **package cell**is a special cell defined inside a Databricks notebook. When **executed** or run, this cell is **compiled**. These cells **run in silos** and are **not** visible to the rest of the notebook.

More Info: [Package cells](https://docs.databricks.com/notebooks/package-cells.html)

## Question 12:

The compliance team is looking at some of the **VACUUM** commands used in the past on the **hotel\_bookings** Delta table.

Command 1: **VACUUM hotel\_bookings RETAIN 168 HOURS**

Command 2: **VACUUM hotel\_bookings RETAIN 7 HOURS DRY RUN**

Command 3: **VACUUM hotel\_bookings RETAIN 7 HOURS**

Command 4: **VACUUM hotel\_bookings RETAIN 168 HOURS DRY RUN**

Command 5: **VACUUM hotel\_bookings RETAIN 0 HOURS DRY RUN**

Command 6: **VACUUM hotel\_bookings**

The team is curious to know which of the following commands will have the same effect on the data files and which of the commands will not affect the number of data files present. Assume that the default settings have not been changed.

* **Commands 3 and 6 will have the same effect whereas Commands 2, 4 and 5 will have no effect on data files.**
* **Commands 1 and 6 will have the same effect whereas Commands 2 and 4 will have no effect on data files.**
* **Commands 1, 3 and 6 will have the same effect whereas Commands 2 and 5 will have no effect on data files.**
* **Commands 1 and 6 will have the same effect whereas Commands 2, 4 and 5 will have no effect on data files.**
* **Commands 3 and 6 will have the same effect whereas Commands 2 and 4 will have no effect on data files.**

**Explanation**

Before looking at each of the **VACUUM** commands, let us look at the syntax of the **VACUUM** command:

**VACUUM table\_name [RETAIN number\_of\_hours HOURS] [DRY RUN]**

In the above syntax, the **retention period** and **DRY RUN** are optional. Also, the **default retention period** is **7 days(168 Hours)**.

**Command 1:** **VACUUM hotel\_bookings RETAIN 168 HOURS**

This command will **permanently** delete all the files that were **deleted** more than **168 hours** ago using the **DELETE** command.

**Command 2:** **VACUUM hotel\_bookings RETAIN 7 HOURS DRY RUN**

This command will **list** all the files(up to 1000 files) that were **deleted** more than **7 hours** ago using the **DELETE** command.

**Command 3:** **VACUUM hotel\_bookings RETAIN 7 HOURS**

This command will **permanently** delete all the files that were **deleted** more than **7 hours** ago using the **DELETE** command.

**Command 4:** **VACUUM hotel\_bookings RETAIN 168 HOURS DRY RUN**

This command will **list** all the files(up to 1000 files) that were **deleted** more than **168 hours** ago using the **DELETE** command.

**Command 5:** **VACUUM hotel\_bookings RETAIN 0 HOURS DRY RUN**

This command will **list** all the files(up to 1000 files) that are **not** a part of the**current version**of the **hotel\_bookings** table.

**Command 6:** **VACUUM hotel\_bookings**

Since the **default** retention period is **7 days(168 hours)**, this command will have the **same effect as the first command.**

Hence, **commands 1 and 6** will have the **same effect** on the data files while **commands 2, 4, and 5** will **not** have any impact on the data files.

More Info: [VACUUM in Databricks](https://docs.databricks.com/sql/language-manual/delta-vacuum.html#:~:text=If%20you%20run%20VACUUM%20on%20a%20Delta%20table%2C%20you%20lose%20the%20ability%20to%20time%20travel%20back%20to%20a%20version%20older%20than%20the%20specified%20data%20retention%20period)

## Question 13:

Which of the following statements is true about the checkpoints in a streaming query?

* **Checkpoint location should always be used while reading data using a streaming query.**
* **Each unique streaming query should have its own checkpoint.**
* **Fault tolerance can be achieved by using checkpoints.**
* **Both B and C**
* **All of the above**

**Explanation**

Let us look at each option to find which of them is/are correct:

*Checkpoint location should always be used while reading data using a streaming query.*

**INCORRECT!** Checkpoint location is used while **writing** the data and **not** while reading.

*Each unique streaming query should have its own checkpoint.*

**CORRECT!** **All** streaming queries should have their **own** checkpoint path.

*Fault tolerance can be achieved by using checkpoints.*

**CORRECT!** Checkpoints can be used for **data replays** as well as for maintaining **fault tolerance.**

So, the correct answer would be **Option D** which states that **both** options **B**and **C** are correct.

More Info: [Checkpointing in Spark Streaming](https://spark.apache.org/docs/latest/streaming-programming-guide.html#checkpointing)

## Question 14:

After attending a Spark conference, each one of the data engineers was asked to write one capability of AQE(Adaptive Query Execution). Which of the below statements made by the data engineers is not completely correct?

* + **Data Engineer 1 - The sort-merge joins can be dynamically changed to hash joins.**
  + **Data Engineer 2 - Helps in saving computing costs by combining small tasks.**
  + **Data Engineer 3 - Handles skews dynamically in stream-static joins.**
  + **Data Engineer 4 - Sort merge joins can be converted to broadcast joins.**
  + **All of the above is true.**

**Explanation**

All the data engineers are completely correct **except** the **third** data engineer as the **AQE** **cannot** be enabled for a **streaming** query.

More Info: [Adaptive Query Execution(AQE) Limitations](https://docs.databricks.com/optimizations/aqe.html#:~:text=AQE%20applies%20to,Non%2Dstreaming)

## Question 15:

Which of the following statements is not true about AutoLoader in Databricks?

* + **Auto Loader can load files from AWS S3, Google Cloud Storage as well as Databricks File System.**
  + **Auto Loader supports Python, Scala and SQL in Delta Live tables.**
  + **Besides new incoming files, Auto Loader can also be used to process already existing files in cloud storage like S3.**
  + **Auto Loader can be used to fetch image files from cloud storage.**
  + **All of the above are false.**

**Explanation**

Let us go through all the options one by one:

*Auto Loader can load files from AWS S3, Google Cloud Storage as well as Databricks File System.*

**TRUE!** You would know that the **Auto Loader** can be used with various cloud storage platforms but it can also be used with **DBFS**(Databricks File System)

*Auto Loader supports Python, Scala and SQL in Delta Live tables.*

**FALSE!**Auto Loader does **not** support **Scala** in Delta Live tables.

*Besides new incoming files, Auto Loader can also be used to process already existing files in cloud storage like S3.*

**TRUE!**Auto Loader supports processing of **already** existing files which means you can start using Auto Loader if you have some files **already** stored and you also expect more **incoming** data in your storage location.

*Auto Loader can be used to fetch image files from cloud storage.*

**TRUE!** Auto Loader can also be used to fetch **Image** files from cloud storage. It can later be used for **Machine Learning algorithms.**

More Info: [Languages supported in Auto Loader](https://docs.databricks.com/ingestion/auto-loader/index.html#:~:text=Auto%20Loader%20has%20support%20for%20both%20Python%20and%20SQL%20in%20Delta%20Live%20Tables)

## Question 16:

A data engineering team needs to enable Change Data Capture(CDC) for all the newly created Delta tables. Which of the following statements can be used?

* + **SET spark.databricks.delta.properties.defaults.enableChangeDataCapture = true**
  + **SET spark.databricks.delta.properties.default.enableChangeDataFeed = true**
  + **SET spark.databricks.delta.properties.defaults.enableChangeDataFeed = true**
  + **All the Delta tables have CDC enabled, by default.**
  + **SET spark.databricks.delta.properties.default.enableChangeDataCapture = true**

**Explanation**

To enable **Change Data Capture** for all newly created **Delta** tables, you need to set the **spark.databricks.delta.properties.defaults.enableChangeDataFeed** property to **true**

Also note, the **CDC logs** will be available for **only** those records which are **inserted**, **updated** or **deleted** after this property is **enabled**.

[Static Notebook](https://certification-champs.github.io/professional-data-engineer-notebooks/Practice%20Test%201/Question%2016.html)

*To view all the notebooks in this course, download [Notebooks.dbc](https://certification-champs.github.io/professional-data-engineer-notebooks/Notebooks.dbc) and import to your Databricks Account.*

More Info: [Enabling Change Data Capture for Delta tables](https://docs.databricks.com/delta/delta-change-data-feed.html#:~:text=All%20new%20tables,%3D%20true%3B)

## Question 17:

A Delta table was created with the following command:

**CREATE TABLE creators (channel\_name STRING, followers INT) COMMENT 'Top 10 creators' TBLPROPERTIES ('genre'='tech');**

A data engineer creates a shallow clone and a deep clone of the **creators** table using the following commands:

**CREATE TABLE creators\_new DEEP CLONE creators;**

**CREATE TABLE creators\_old SHALLOW CLONE creators;**

Which of the following is true about the **creators\_new** and **creators\_old** tables?

* **Table properties specified by TBLPROPERTIES will be accessible in creators\_new table but not in the creators\_old table whereas neither of the clones of the creators table can access the comment added to the creators table.**
* **The comments added to the creators table can be accessed by both the clones of the creators table but the table properties cannot be accessed.**
* **The comments and the table properties cannot be accessed by the creators\_old and creators\_new tables.**
* **Table properties can be accessed by both the clones of the creators table but the comments added to the creators table will not be available.**
* **Comments added to the creators table will be accessible in creators\_new table but not in the creators\_old table whereas neither of the clones of the creators table can access the table properties.**

**Explanation**

The transfer of **comments** and **table** **properties** does not depend on the **type** of clone. In either type of clones - **deep** and **shallow,** the **table** **properties** are passed to the **clone** as well whereas the **comments** are something **native** to a table.

Thus, the **comments** added to a table are **not** passed on to the **clones.**

[Static Notebook](https://certification-champs.github.io/professional-data-engineer-notebooks/Practice%20Test%201/Question%2017.html)

*To view all the notebooks in this course, download [Notebooks.dbc](https://certification-champs.github.io/professional-data-engineer-notebooks/Notebooks.dbc) and import to your Databricks Account.*

More Info: [Cloning a Delta table](https://docs.databricks.com/optimizations/clone.html)

## Question 18:

A data engineer executes the following query:

1. spark.readStream.format('delta') \
2. .table('scorecard') \
3. .writeStream \
4. .option('checkpointLocation', '/tmp/scores') \
5. .format('delta') \
6. .option('mergeSchema', 'true') \
7. .table('performance')

What will be the result of the query and the **performance** table if a column from the **scorecard** table is renamed while the query is running?

* + **The query will continue to run and the column will be renamed in the performance table as well.**
  + **The query will continue to run but the column name in the performance table will remain the same.**
  + **The query will be stopped but the column name in the performance table will be updated.**
  + **The query will be paused for 1 minute to accommodate the changes in the performance table. Once the changes are processed, the query will continue to execute.**
  + **The query will be stopped and the column name in the performance table will remain the same.**

**Explanation**

If you try to **change** the **column** name of an upstream table, the streaming query will be **stopped** immediately with the following **error** message:

**com.databricks.sql.transaction.tahoe.DeltaIllegalStateException: Detected schema change**

Also note, even if the query is using **.option('mergeSchema', ‘true’)** the **error** persists.

[Static Notebook](https://certification-champs.github.io/professional-data-engineer-notebooks/Practice%20Test%201/Question%2018.html)

*To view all the notebooks in this course, download [Notebooks.dbc](https://certification-champs.github.io/professional-data-engineer-notebooks/Notebooks.dbc) and import to your Databricks Account.*

## Question 19:

Which of the following statements explains the working of Z-ordering for a Delta table?

* + **Z-ordering optimizes the files to create similar-size files based on the number of rows.**
  + **Z-ordering collects the min and max values of each column and adds them to another table.**
  + **Z-ordering optimizes the files to create similar-size files based on their size on disk.**
  + **Z-ordering works only on the text data columns.**
  + **Z-ordering clears the column with text-only data before performing the ordering.**

**Explanation**

Z-ordering helps in data skipping by **colocating** similar data which, in turn, helps in **faster** queries. The data is spread evenly based on the **number of rows** in the data and **NOT** the **size of the data files**.

More Info: [Z-ordering in Databricks](https://docs.databricks.com/delta/data-skipping.html#:~:text=Z%2Dordering%20aims%20to%20produce%20evenly%2Dbalanced%20data%20files%20with%20respect%20to%20the%20number%20of%20tuples%2C%20but%20not%20necessarily%20data%20size%20on%20disk.)

## Question 20:

Which of the following statements is correct about the rescue schema evolution mode in Auto Loader?

* + **As soon as a new column is found, the stream fails but the schema is evolved.**
  + **The schema is not evolved and the new columns are ignored while the stream continues to execute.**
  + **The stream fails when new columns are encountered while the details of new columns are stored in the \_rescued\_data column.**
  + **If a new column is detected, the stream continues its execution but the schema is not evolved whereas the new columns’ information is added to the \_rescued\_data column.**
  + **On detection of a new column, the schema is evolved and the column is added to the \_rescued\_data column but the stream does not fail.**

**Explanation**

There are **4**types of schema evolution modes supported by **Auto Loader:**

1. **addNewColumns**
2. **rescue**
3. **failOnNewColumns**
4. **none**

In **rescue** mode, if a **new** column is detected, the schema does **not** evolve and the new column is **added** to the **\_rescued\_data** column while the stream **continues** to run.

More Info: [Auto Loader schema evolution](https://docs.databricks.com/ingestion/auto-loader/schema.html#:~:text=evolve%20data%20types.-,rescue,never%20evolved%20and%20stream%20does%20not%20fail%20due%20to%20schema%20changes.,-All%20new%20columns)

## Question 21:

A data engineering team is working on a POC for which they need to populate a downstream table using a source table that was created with Change Data Feed enabled. The upstream table has 45 columns while the downstream table should have 47 columns with 2 added columns - **typeOfChange** and **version** where the **typeOfChange** column depicts the type of change while the **version** column signifies the version number for the change. Also, the target table should be truncated and loaded with new data whenever the query is re-run.

spark.read.format('delta') \

.option('readChangeFeed', 'true') \

.option('startingVersion', 0) \

.table('sourceTable') \

\_\_\_\_\_\_\_(1)\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_(2)\_\_\_\_\_\_\_\_

.write \

.format('delta') \

\_\_\_\_\_\_\_(3)\_\_\_\_\_\_\_\_

.saveAsTable('targetTable')

Which of the following correctly fills the numbered blanks to achieve the expected result?

* **.drop('\_commit\_timestamp') \**

**.select('\*', col('\_change\_type').alias('typeOfChange'),**

**col('\_commit\_version').alias('version')) \**

**.mode('overwrite') \**

* **.drop(‘commit\_timestamp’) \**

**.select(‘\*’, col(‘change\_type’).alias(‘typeOfChange'),**

**col('commit\_version').alias('version')) \**

**.mode(‘overwrite’) \**

* **.drop(‘\_commit\_timestamp’) \**

**.withColumnRenamed(‘\_change\_type', 'typeOfChange').withColumnRenamed('\_change\_version', 'version') \**

**.mode(‘truncate’) \**

* **drop(‘\_change\_timestamp’) \**

**.withColumnRenamed(‘\_change\_type', 'typeOfChange').withColumnRenamed('\_commit\_version', 'version') \**

**.mode(‘overwrite’) \**

* **.drop(‘\_commit\_timestamp’) \**

**.withColumnRenamed(‘\_change\_type', 'typeOfChange').withColumnRenamed('\_commit\_version', 'version') \**

**.mode(‘overwrite’) \**

**Explanation**

Once the **Change Data Feed** is **enabled**, the following **three** columns will be added:

1. **\_change\_type**
2. **\_commit\_version**
3. **\_commit\_timestamp**

The **last** option correctly **renames** the columns as per the requirements as well as uses the correct **mode** I.e **overwrite**

[Static Notebook](https://certification-champs.github.io/professional-data-engineer-notebooks/Practice%20Test%201/Question%2021.html)

*To view all the notebooks in this course, download [Notebooks.dbc](https://certification-champs.github.io/professional-data-engineer-notebooks/Notebooks.dbc) and import to your Databricks Account.*

More Info: [Change Data Capture Schema](https://docs.databricks.com/delta/delta-change-data-feed.html#:~:text=_change_type,commit%20was%20created)

## Question 22

The following query is needed to run in the production environment where the **conversion\_rate** table is to be updated manually(static table) while the data in the **ledger** table should always be up to date as per the data in the **transactions** table:

spark.readStream.format('delta') \

.table('transactions') \

.join(table('conversion\_rate'), 'currency', 'left') \

.writeStream \

.option('checkpointLocation', '/tmp/ledger') \

.trigger(processingTime='30 minutes’) \

.format('delta') \

.table('ledger')

Which of the following is true about the join’s functionality?

* **The updates made to the data present in the conversion\_rate table will result in an instant error in the streaming query.**
* **The updates made to the data present in the conversion\_rate table will result in an error when the new batch of data arrives in the transactions table.**
* **The data in the conversion\_rate table should be updated once the new batch of data has arrived and the join is in progress.**
* **The data in the conversion\_rate table should be updated once the batch of data has been processed and the other batch of data is yet to arrive for processing.**
* **The streaming join will fail as the default join type is outer join.**

**Explanation**

Let us take a look at each option one by one:

*The updates made to the data present in the****conversion\_rate****table will result in an instant error in the streaming query.*

**INCORRECT!** In the case of **stream-static join**, each batch of data is joined with the **current** state of the **static** table. So, **no** error will be returned if the updates are made to the **conversion\_rate** table. Also, any updates made to the **conversion\_rate** table will only affect the **next** batch of data.

*The updates made to the data present in the****conversion\_rate****table will result in an error when the new batch of data arrives in the****transactions****table.*

**INCORRECT!** In the case of **stream-static join**, each batch of data is joined with the **current** state of the **static** table. So, **no** error will be returned if the updates are made to the **conversion\_rate** table. Also, any updates made to the **conversion\_rate** table will only affect the **next** batch of data.

*The data in the****conversion\_rate****table should be updated once the new batch of data has arrived and the join is in progress.*

**INCORRECT!** To get **correct** results, the changes to the data in the **conversion\_rate** table should be made in the **30-minutes time frame** between the **two** triggers. Changing the data while the data is being **processed** will result in **ambiguous** results in the **downstream** table.

*The data in the****conversion\_rate****table should be updated once the batch of data has been processed and the other batch of data is yet to arrive for processing.*

**CORRECT!** Once a batch of data is **processed**, there is a **30-minute time frame** that can be **leveraged** for updations in the **static** table.

*The streaming join will fail as the default join type is****outer****join.*

**INCORRECT!** The **default** join type is an **inner** join. Also, keep in mind, a **stream-static** join of type **outer** is **not** supported and will result in an **error.**

More Info: [Stream-static joins in Spark](https://docs.databricks.com/structured-streaming/delta-lake.html#:~:text=A%20stream%2Dstatic,current%20micro%2Dbatch)

## Question 23

Which of the following Languages is/are supported by the DLT(Delta Live Tables) in Databricks?

* **Only SQL**
* **Scala and Python**
* **Python, Java and Scala**
* **Python, SQL and Scala**
* **Python and SQL**

**Explanation**

Currently, **DLT** supports only **Python** and **SQL.**

More Info: [Languages supported by DLT(Delta Live Tables)](https://docs.databricks.com/workflows/delta-live-tables/delta-live-tables-faqs-issues.html#:~:text=Delta%20Live%20Tables%20supports%20only%20SQL%20and%20Python)

## Question 24:

Which of the following methods can be used to define a function which can be executed on the output of every micro-batch in a Spark Streaming query?

* **foreachBatch()**
* **foreachUDF()**
* **pivot()**
* **explain()**
* **fun()**

**Explanation**

**foreachBatch()** is a function that can be called on a **DataStreamWriter.** It can be used to define a **function** that can be executed on the output of every **micro-batch** in a Spark Streaming query.

More Info: [Using foreachBatch() method](https://docs.databricks.com/structured-streaming/foreach.html#:~:text=streamingDF.writeStream.foreachBatch(...)%20allows%20you%20to%20specify%20a%20function%20that%20is%20executed%20on%20the%20output%20data%20of%20every%20micro%2Dbatch%20of%20the%20streaming%20query)

## Question 25:

Which of the following methods can be used to remove duplicate columns after an inner join between two DataFrames?

* **dropDuplicates()**
* **dropColumns()**
* **drop\_duplicates()**
* **dropDuplicateColumns()**
* **drop()**

**Explanation**

If you try to **join** two DataFrames, it can add **duplicate** **columns** to the resultant DataFrame. To **remove** the **duplicate columns** you can add the **drop()** method to the DataFrame.

Let us quickly check what other options are used for:

**dropDuplicates()** - Used for **dropping** **duplicate** **rows** from a DataFrame.

**dropColumns()**- **Not** a **valid** method in Spark.

**drop\_duplicates()** - Used for **dropping** **duplicate** **rows** from a DataFrame.

**dropDuplicateColumns()** - **Not** a **valid** method in Spark.

[Static Notebook](https://certification-champs.github.io/professional-data-engineer-notebooks/Practice%20Test%201/Question%2025.html)

*To view all the notebooks in this course, download [Notebooks.dbc](https://certification-champs.github.io/professional-data-engineer-notebooks/Notebooks.dbc) and import to your Databricks Account.*

More Info: [drop() method in Spark](https://spark.apache.org/docs/3.2.0/api/python/reference/api/pyspark.sql.DataFrame.drop.html)

## Question 26:

Which of the following locations will be used to store the DLT’s events log if the Delta Live Table’s storage location has been set up as **/teams/prod**?

* **/teams/prod/system/events**
* **/teams/prod/logs/events**
* **/teams/prod/logs**
* **/teams/prod/pipelines/events**
* **/teams/prod/system**

**Explanation**

To store the **events log** for a **Delta Live Table,** **/system/events** is **appended** to the storage location of the **DLT pipeline** i.e if the storage location is **/loc**, the **/loc/system/events** will store the **events log.**

More Info: [Delta Live Tables events log location](https://docs.databricks.com/workflows/delta-live-tables/delta-live-tables-event-log.html#:~:text=if%20you%20have%20configured%20your%20pipeline%20storage%20setting%20as%20/Users/username/data%2C%20the%20event%20log%20is%20stored%20in%20the%20/Users/username/data/system/events)

## Question 27:

A team is working on the execution of a streaming query that uses Auto Loader to fetch CSV files from the cloud storage. Which of the following queries can be used to fetch only the files with **.csv** extension from the following locations:

s3://bucket/orders/data/

s3://bucket/employees/data/

s3://bucket/students/data/

s3://bucket/policies/data/

* spark.readStream.format("cloudFiles") \

.option("cloudFiles.format", 'csv') \

.schema(schema) \

.load("s3://bucket/\*/data/\*.csv")

* spark.readStream.format("cloudFiles") \

.option("cloudFiles.format", 'csv') \

.schema(schema) \

.option("pathGlobfilter", "\*.csv") \

.load("s3://bucket/\*/data")

* spark.readStream.format("cloudFiles") \

.option("cloudFiles.format", 'csv') \

.schema(schema) \

.option("pathFilter", "\*.csv") \

.load("s3://bucket/\*/data")

* spark.readStream.format("cloudFiles") \

.schema(schema) \

.load("s3://bucket/\*/data/\*.csv")

* spark.readStream.format("cloudFiles") \

.option("cloudFiles.format", 'csv') \

.schema(schema) \

.option("globFilter", "\*.csv") \

.load("s3://bucket/\*/data")

**Explanation**

This is a **tricky** question. To answer this question you should be aware of the syntax for the **filter** criteria based on the **prefix** and **suffix** of the file names.

To add a **suffix** **pattern**, you would need **.option('pathGlobfilter', '\*.csv')** whereas to filter the path to load the files you can use wildcards in **.load()**

So, the correct code block should be:

1. spark.readStream.format("cloudFiles") \
2. .option("cloudFiles.format", 'csv') \
3. .schema(schema) \
4. .option("pathGlobfilter", "\*.csv") \
5. .load("s3://bucket/\*/data")

More Info: [Filtering files in Auto Loader](https://docs.databricks.com/ingestion/auto-loader/patterns.html#:~:text=You%20need%20to%20use%20the%20option%20pathGlobFilter%20for%20explicitly%20providing%20suffix%20patterns.%20The%20path%20only%20provides%20a%20prefix%20filter)

## Question 28:

A team of data engineers is testing a stream-stream join in Spark. Which of the following statements is true about these types of joins?

* **A stream-stream join is not supported in Spark.**
* **A stream-stream join cannot be an outer join.**
* **Adding a watermark is necessary for a stream-stream outer join.**
* **A stream-stream join is supported in Spark only if one of the streams has Kafka as its source.**
* **A stream-stream join can only be performed in Scala.**

**Explanation**

Let us go through each option one by one:

*A stream-stream join is not supported in Spark.*

**INCORRECT!** A **stream-stream** join is **supported** in Spark.

*A stream-stream join cannot be an****outer****join.*

**INCORRECT!** Unlike a **stream-static** join, a **stream-stream** join can be of **outer** type.

*Adding a watermark is necessary for a stream-stream****outer****join.*

**CORRECT!** In an **outer** stream-stream join, adding a watermark is **necessary** to get the correct result as the query should know how much **time** it needs to wait for the data to arrive.

*A stream-stream join is supported in Spark only if one of the streams has Kafka as its source.*

**INCORRECT!** A **stream-stream** join can have many streaming **sources** which are **not** limited to just **Kafka**.

*A stream-stream join can only be performed in Scala.*

**INCORRECT!** A **stream-stream** join is **not** limited to **Scala** language as it can be performed in **many** languages including **Python**.

[Static Notebook](https://certification-champs.github.io/professional-data-engineer-notebooks/Practice%20Test%201/Question%2028.html)

*To view all the notebooks in this course, download [Notebooks.dbc](https://certification-champs.github.io/professional-data-engineer-notebooks/Notebooks.dbc) and import to your Databricks Account.*

More Info: [Watermarking in stream-stream join in Databricks](https://www.databricks.com/blog/2018/03/13/introducing-stream-stream-joins-in-apache-spark-2-3.html#:~:text=Unlike%20inner%20joins%2C%20the%20watermarks%20and%20event%2Dtime%20constraints%20are%20not%20optional%20for%20outer%20joins) | [Watermarking in stream-stream join in Spark](https://spark.apache.org/docs/latest/structured-streaming-programming-guide.html#outer-joins-with-watermarking)

## Question 29:

A downstream application wants to have all the entries in the **mobile** column of the **customers** table for auditing purposes. The following streaming query has been set up by the data engineer:

spark.readStream.format('delta') \

.option('readChangeFeed', 'true') \

.table('customers') \

.select('mobile') \

.writeStream \

.format('delta') \

.option('checkpointLocation', '/tmp/customers/\_checkpoints/') \

.table(‘mobiles')

Which of the following updates should be made to the query to restrict duplicate entries in the **mobiles** table?

* **option("startingVersion", 0) should be added to avoid duplicate entries in the mobiles table.**
* **The rows having \_change\_type column’s value as delete and update\_postimage should be dropped.**
* **The checkpoint location should be omitted from the streaming query.**
* **Only the rows having values as insert, update\_postimage and delete for \_change\_type column should be kept.**
* **All the rows which contain delete and update\_preimage in \_change\_type should be dropped.**

**Explanation**

Let us take a look at each option to find out the outcomes:

***option("startingVersion", 0)****should be added to avoid duplicate entries in the****mobiles****table.*

**INCORRECT!** Adding this will **not** make any difference to the number of records in the **mobiles** table.

*Before looking at the next options, you should know that the****\_change\_type****column can have****4****types of values -****insert****,****update\_preimage****,****update\_postimage****and****delete***

*The rows having****\_change\_type****column’s values as****delete****and****update\_postimage****should be dropped.*

**INCORRECT!** As the **update\_postimage** value signifies the **updated** value, dropping rows with **update\_postimage** value will result in records getting **missed** from the **mobiles** table.

*The checkpoint location should be omitted from the streaming query.*

**INCORRECT!** Removing the **checkpoint** location will have **no** effect on the number of records in the **mobiles** table.

*Only the rows having values as****insert****,****update\_postimage****and****delete****for****\_change\_type****column should be kept.*

**INCORRECT!** If the **customers** table has one or more records **deleted**, retaining the rows where the value of **\_change\_type** column is **delete** will always add **duplicates** in the **mobiles** table as the same record would have already been added to the mobiles tables.

*All the rows which contain****delete****and****update\_preimage****in****\_change\_type****should be dropped.*

**CORRECT!** Dropping the records having values **delete** or **update\_preimage** will result in **mobiles** table having **only** unique values. Alternatively, **retaining** the records having values **insert** or **update\_postimage** in **\_change\_type** column will **restrict** duplicate values in the **mobiles** table.

[Static Notebook](https://certification-champs.github.io/professional-data-engineer-notebooks/Practice%20Test%201/Question%2029.html)

*To view all the notebooks in this course, download [Notebooks.dbc](https://certification-champs.github.io/professional-data-engineer-notebooks/Notebooks.dbc) and import to your Databricks Account.*

More Info: [Using \_change\_type column for Change Data Capture in Databricks](https://docs.databricks.com/delta/delta-change-data-feed.html#:~:text=_change_type,update_postimage%2C%20delete)

## Question 30:

A new data engineer created a cluster with 6 workers. How many executors will be created as part of the cluster creation?

* **5 executors**
* **3 executors**
* **6 executors**
* **7 executors**
* **A new cluster with 6 worker nodes cannot be created as the workers can be 2, 4 or 8.**

**Explanation**

As you might know, in Databricks, **each worker has exactly one executor**, the number of executors will be equal to **6.**

More Info: [Relationship between a worker node and an executor in Databricks](https://docs.databricks.com/clusters/cluster-config-best-practices.html#:~:text=Databricks%20runs%20one%20executor%20per%20worker%20node.%20Therefore%20the%20terms%20executor%20and%20worker%20are%20used%20interchangeably%20in%20the%20context%20of%20the%20Databricks%20architecture)

## Question 31:

A column named **id** serves as the PRIMARY KEY for the **orders** table as well as a FOREIGN KEY to the **users** table. Which of the following commands can be used to drop the PRIMARY KEY constraint from the **orders** table while dropping the FOREIGN KEY reference as well?

* **ALTER orders DROP PRIMARY KEY CASCADE;**
* **ALTER TABLE orders DROP PRIMARY KEY CASCADE;**
* **ALTER TABLE orders DROP PRIMARY\_KEY;**
* **ALTER orders DROP PRIMARY\_KEY;**
* **ALTER TABLE orders DROP PRIMARY\_KEY CASCADE;**

**Explanation**

Using **CASCADE** at the end of the statement helps in removing any **foreign key** references to the table. Without its usage, the command will throw an **error**, if the **PRIMARY KEY** is referenced as a **FOREIGN KEY** to another table.

More Info: [Removing FOREIGN KEY constraint with PRIMARY KEY](https://docs.databricks.com/sql/language-manual/sql-ref-syntax-ddl-alter-table-drop-constraint.html#:~:text=If%20you%20specify%20CASCADE%2C%20dropping%20the%20primary%20key%20results%20in%20dropping%20any%20foreign%20keys%20referencing%20the%20table)

## Question 32:

Which of the following statements about data skipping is not true?

* **Data skipping features are not enabled by default and need to be enabled using delta.dataSkipping table property.**
* **Adding more columns to collect statistics will increase the overhead while writing data to the data-skipping-enabled delta tables.**
* **All fields inside a nested column are taken as individual columns while collecting statistics.**
* **Data skipping helps in faster queries.**
* **Data skipping on columns having long string values is a costly operation.**

**Explanation**

All the options are correct except the **first** one as the data skipping is **auto-enabled** on the Delta tables.

More Info: [Data Skipping for a Delta table](https://docs.databricks.com/delta/data-skipping.html#:~:text=You%20do%20not%20need%20to%20configure%20data%20skipping%3B%20the%20feature%20is%20activated%20whenever%20applicable)

## Question 33:

Which of the following tables will not be automatically broadcasted in case of a join with a table 2 GB in size if the **spark.sql.autoBroadcastJoinThreshold** config property is set to default:

**Table 1** - 15.2 MB

**Table 2** - 2.6 MB

**Table 3** - 5.1 MB

**Table 4** - 11 MB

* **Tables 1, 3 and 4**
* **Tables 2 and 3**
* **Tables 1, 2 and 3**
* **Tables 1 and 4**
* **Tables 2, 3 and 4**

**Explanation**

The **default** value of **spark.sql.autoBroadcastJoinThreshold** config property is **10 MB.** To explicitly **broadcast** a table larger than **10 MB** in size, you would require **broadcast** method. Once broadcasted, a **copy** of this table is passed to each **executor.**

Also note, the **broadcast** method should be used **cautiously** as it may result in an **out-of-memory error.**

More Info: [spark.sql.autoBroadcastJoinThreshold property in Spark](https://spark.apache.org/docs/latest/sql-performance-tuning.html" \l ":~:text=1.3.0-,spark.sql.autoBroadcastJoinThreshold,-10485760)

## Question 34:

An upstream system sends more than 1000 files every hour with each file containing 5000-30000 records. The size of each file ranges from 580MB to 1.22 GB. The stream of data is ingested using a medallion architecture. The data engineer needs to limit the amount of data processed in each micro-batch. Which of the following options is valid for limiting the input rate while reading the stream of data?

* **maxFilesPerTrigger, maxBytesPerTrigger and maxRowsPerTrigger**
* **maxBytesPerTrigger and maxRecordsPerTrigger**
* **maxBytesPerTrigger and maxFilesPerTrigger**
* **maxFilesPerTrigger and maxRecordsPerTrigger**
* **maxFilesPerTrigger, maxBytesPerTrigger, maxRowsPerTrigger and maxRecordsPerTrigger**

**Explanation**

To control the amount of data processed in each **micro-batch** following input limit options can be used: **maxFilesPerTrigger** and **maxBytesPerTrigger**

Also note that if **both** of them are used in a **single** **query**, the data processed in each **micro-batch** will be the **lesser** of **maxFilesPerTrigger** and **maxBytesPerTrigger** Whichever limit is reached **first**, the data processing will be **stopped** for that **micro-batch** of the stream.

More Info: [Limiting input rate for streaming reads](https://docs.databricks.com/structured-streaming/delta-lake.html#:~:text=maxFilesPerTrigger%3A%20How%20many,set%20by%20default)

## Question 35:

An insurance company stores its data in such a way that the **start\_date** and **end\_date** of all the current and expired policies can be fetched from a single table. Whenever a new policy is generated the record is added to the delta table with the **active** column set to **True** whereas the value remains **False** for the expired policies. Which type of slowly Changing Dimension(SCD) is followed by this delta table?

* **Type 0 SCD**
* **Type 1 SCD**
* **Type 2 SCD**
* **Type 3 SCD**
* **Type 4 SCD**

**Explanation**

Three of the most commonly used types of **Slowly Changing Dimension(SCD)** are:

1. ***Type 1 SCD*** - The **old** record is **overwritten** by the **new** record.
2. ***Type 2 SCD*** - The **new** record is **appended** to the table whereas the **old** record is set to **not-active** either by using the **end\_date** or **active** column.
3. ***Type 3 SCD*** - Changes are tracked using **column** **addition**. A **new** column is **added** for the **current** value while the **original** value is **retained**, as well.

In this question, as the **active** column is used and the **previous** record is also **retained**, ***SCD Type 2*** is being used.

More Info: [Slowly Changing Dimension(SCD) Type 2](https://www.databricks.com/blog/2023/01/25/loading-data-warehouse-slowly-changing-dimension-type-2-using-matillion.html#:~:text=What%20is%20a,equal%20to%20%27F%27)

## Question 36:

A delta table stores data for each ball bowled in a cricket match. Every time a ball is bowled, the name of the bowler, batter and the number of runs scored is added to the delta table manually as an **INSERT** statement. After storing data for 350 cricket matches, the data engineer noticed that the read queries are taking more time as compared to read queries on other tables of relatively same size. Which technique can be used by the data engineer to solve this problem in an efficient manner?

* **Use different tables for each cricket match.**
* **Once the match is completed, the data should be moved to a new table as a single INSERT statement.**
* **Change Data Feed should be used to increase the speed of read queries.**
* **OPTIMIZE command should be used on the delta table.**
* **The data should be converted to CSV for faster reads.**

**Explanation**

According to the question, the details are entered **manually** into the table using **INSERT** statements which will result in a **large** number of **files.** As each **INSERT** statement adds data for just **3** columns, the size of each file will be very **small** compared to the **overall** **size** of the table. So, each time the table is queried a **large** number of files is needed to be **scanned** which makes the queries **slower** than usual. To **increase** the **speed** of the queries, **OPTIMIZE** can be used on the table which **combines** **smaller** **files** **into** **larger** ones helping in **speeding** up the **read** queries.

More Info: [Combining small files using OPTIMIZE](https://docs.databricks.com/delta/best-practices.html#:~:text=If%20you%20continuously,the%20OPTIMIZE%20command)

## Question 37

Three different types of views available in Databricks i.e view, temporary view and global temporary view are being used in a single notebook. Which of the following views will be accessible if the notebook is detached and re-attached to the same cluster?

* **All the 3 views will be accessible.**
* **Only the global temp view will be accessible.**
* **None of the views will be accessible.**
* **All the views except the temporary view will be accessible.**
* **Temporary and global temporary views will not be accessible.**

**Explanation**

Let us take a look at each of the **views** one by one:

1. ***View*** - It persists through **multiple** **sessions** and thus, will be **accessible** even after the notebook is **detached** and **re-attached.**
2. ***Temporary view*** - As the name suggests, the **temporary** **views** are **tied** to a **session**. So, these types of views will **not** be accessible once the notebook is **detached** from the cluster.
3. ***Global temporary view*** - They are stored in a **temporary** **database** called **global\_temp** and are **accessible** even if the notebook is **detached** and **re-attached** from the cluster.

Hence, only the **temporary** **view** will **lose** its **existence** once the notebook is **detached** and **re-attached.**

[Static Notebook](https://certification-champs.github.io/professional-data-engineer-notebooks/Practice%20Test%201/Question%2037.html)

*To view all the notebooks in this course, download [Notebooks.dbc](https://certification-champs.github.io/professional-data-engineer-notebooks/Notebooks.dbc) and import to your Databricks Account.*

More Info: [Types of Views in Databricks](https://docs.databricks.com/sql/language-manual/sql-ref-syntax-ddl-create-view.html#:~:text=be%20its%20owner.-,TEMPORARY,TEMPORARY%20views%20are%20tied%20to%20a%20system%20preserved%20temporary%20schema%20global_temp,-.)

## Question 38

A data engineer tries to execute the following set of statements to test the data for radio and TV channels with valid frequencies:

1. CREATE TABLE channels (name string, frequency float) LOCATION '/channels';
2. INSERT INTO channels VALUES ('DB', 101.2);
3. CREATE TABLE tv\_channels (name string, frequency float) LOCATION '/channels';
4. INSERT INTO tv\_channels VALUES ('FS', 10045.7);
5. DROP TABLE channels;
6. SELECT \* FROM tv\_channels;

Which of the following describes the output of the above set of statements, assuming that the **/channels** location was empty before executing the above set of statements?

* **The tv\_channels table will not be created as it is using the same location as the channels table.**
* **Dropping the channels table will delete all the data from both tables.**
* **The channels table cannot be dropped since the location contains the data for both channels and tv\_channels tables.**
* **Using the SELECT statement over the tv\_channels table will output the data from both the tables i.e 2 records.**
* **All the commands will be executed successfully and the output will show 1 record from the tv\_channels table.**

**Explanation**

Let us look at each statement given in the question and learn more about it:

***CREATE TABLE channels (name string, frequency float) LOCATION '/channels';***

The **channels** table will be created **successfully.**

***INSERT INTO channels VALUES ('DB', 101.2);***

A **new** record will be added to the **channels** table with **name** as **DB** and **frequency** as **101.2**

***CREATE TABLE tv\_channels (name string, frequency float) LOCATION '/channels';***

A new table **tv\_channels** will be created in the **same** location as the **channels** table.

***INSERT INTO tv\_channels VALUES ('FS', 10045.7);***

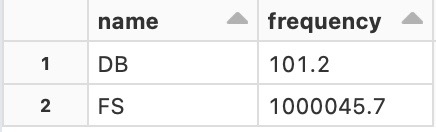
A **new** record will be added to the **tv\_channels** table with **name** as **FS** and **frequency** as **10045.7**

***DROP TABLE channels;***

The **channels** table will be **dropped** but the data for **both** tables will still be **stored** in the **/channels** location.

***SELECT \* FROM tv\_channels;***

The **output** will include **all** the records stored in the **/channels** location i.e **2** records.



[Static Notebook](https://certification-champs.github.io/professional-data-engineer-notebooks/Practice%20Test%201/Question%2038.html)

*To view all the notebooks in this course, download [Notebooks.dbc](https://certification-champs.github.io/professional-data-engineer-notebooks/Notebooks.dbc) and import to your Databricks Account.*

More Info: [Dropping an external table](https://docs.databricks.com/data-governance/unity-catalog/create-tables.html#:~:text=When%20you%20run,CREATE%20TABLE%20statement)

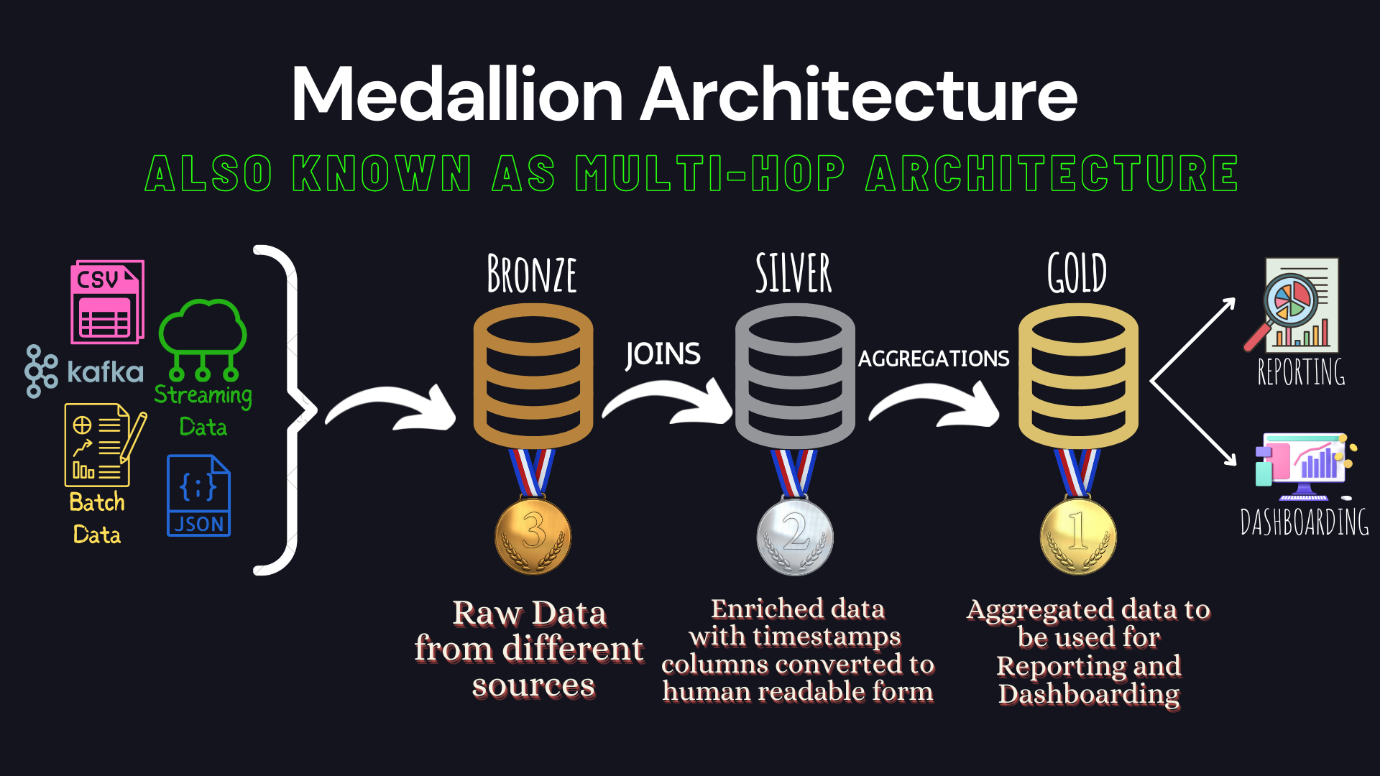
## Question 39:

Which of the following layers of a multi-hop architecture is most likely to include joins transformation?

* **Bronze**
* **Silver**
* **Gold**
* **Raw**
* **Joins are not supported in multi-hop architecture**

**Explanation**

Once the **raw** data is loaded into the **Bronze** layer, the **Silver** layer **enriches** the data using **stream-static joins.**



## Question 40:

A data engineer creates a user-defined function named **sum** which accepts two arguments. The first argument is of type integer while the second being a float. What will be the output if another data engineer tries to run the following query to call the above function:

**SELECT sum(5,8);**

* **The sum function will be executed successfully and 13 will be printed as the output.**
* **An error will be returned as the second argument should be a float.**
* **The command will be successful and the output i.e 13.00 will be printed**
* **As the sum function is a built-in function, an error will be returned on function creation.**
* **An error will be returned on the function call as the built-in sum function accepts only one argument.**

**Explanation**

As the **sum** function is a **built-in** function that accepts only **one** argument, the function call to the **sum** function will return the following **error:**

**Error in SQL statement: AnalysisException: Invalid number of arguments for function sum. Expected: 1; Found: 2;**

[Static Notebook](https://certification-champs.github.io/professional-data-engineer-notebooks/Practice%20Test%201/Question%2040.html)

*To view all the notebooks in this course, download [Notebooks.dbc](https://certification-champs.github.io/professional-data-engineer-notebooks/Notebooks.dbc) and import to your Databricks Account.*

More Info: [sum() built-in function](https://docs.databricks.com/sql/language-manual/functions/sum.html)

Bottom of Form

Top of Form

## Question 41:

Which of the following statements can be used to create a partitioned delta table named **records** based on the **year** and **month** columns?

* **CREATE TABLE records (id int, year int, month int) PARTITION BY (year, month);**
* **CREATE TABLE records (id int, year int, month int) PARTITIONED BY (year, month);**
* **CREATE TABLE records (id int, year int, month int) PARTITIONED BY year, month;**
* **CREATE TABLE records (id int, year int, month int) PARTITION BY year, month;**
* **CREATE TABLE records (id int, year int, month int) PARTITION BY (year and month);**

**Explanation**

To **partition** a delta table based on **one or more columns**, the **PARTITIONED BY** keyword should be used. The **column** or **list of columns** should be **added** inside the **brackets**after the **PARTITIONED BY** keyword. So, the correct answer would be:

**CREATE TABLE records (id int, year int, month int) PARTITIONED BY (year, month);**

Also, note that **PARTITION BY** and **PARTITIONED BY** seems to be very similar but only the **latter** one can be used to **partition** the table.

[Static Notebook](https://certification-champs.github.io/professional-data-engineer-notebooks/Practice%20Test%201/Question%2041.html)

*To view all the notebooks in this course, download [Notebooks.dbc](https://certification-champs.github.io/professional-data-engineer-notebooks/Notebooks.dbc) and import to your Databricks Account.*

More Info: [Create a partitioned delta table](https://docs.databricks.com/sql/language-manual/sql-ref-syntax-ddl-create-table-using.html#:~:text=CREATE%20TABLE%20student%20(id%20INT%2C%20name%20STRING%2C%20age%20INT)%0A%20%20%20%20PARTITIONED%20BY%20(age)%3B)

## Question 42:

Which of the following statements correctly depicts the difference between **SCHEMA** and **DATABASE** keywords in Databricks?

* **The DATABASE keyword is supported in Databricks whereas the SCHEMA keyword is not.**
* **DATABASE and SCHEMA can both be used to create a new database.**
* **A schema is a collection of databases.**
* **Using the DATABASE keyword will enable delta log for the database whereas using the SCHEMA keyword will not add delta logs for that database.**
* **Both B and D**

**Explanation**

In Databricks, both **SCHEMA** and **DATABASE** keywords can be used to refer to a **database.** Also note that in a **relational** **DBMS** system, a database refers to a **collection of schemas.**

[Static Notebook](https://certification-champs.github.io/professional-data-engineer-notebooks/Practice%20Test%201/Question%2042.html)

*To view all the notebooks in this course, download [Notebooks.dbc](https://certification-champs.github.io/professional-data-engineer-notebooks/Notebooks.dbc) and import to your Databricks Account.*

More Info: [Database/Schema in Databricks](https://docs.databricks.com/lakehouse/data-objects.html#what-is-a-database)

## Question 43:

A new member has recently been added to a team of developers. The new member has been assigned a task to review a notebook. What is the minimum notebook-level permission that can be granted to the new member allowing them to view the existing notebook?

* **No permissions are required**
* **Can Read permission**
* **Can Run permission**
* **Can Edit permission**
* **Can Mange permission**

**Explanation**

There are **5** levels of permissions for a **notebook** in Databricks:

1. No permissions
2. **Can Read** - Can **view the cells** of the notebook, **add comments** and **run the notebook** using the **%run** magic command or the notebook workflows.
3. **Can Run** - All the permissions stated in **Can Read** plus can **attach** or **detach** notebooks from a cluster and **run** **the** **cells** of the notebook.
4. **Can Edit** - All the permissions in **Can Run** and can also **edit the cells.**
5. **Can Manage** - All the permissions stated in **Can Edit** and also the ability to **change the permissions** of the notebook.

So, to **view a notebook**, you need to have at least **Can Read** permission.

More Info: [Notebook-level permissions in Databricks](https://docs.databricks.com/security/access-control/workspace-acl.html#:~:text=View%20cells,x)

## Question 44:

A secret value stored in the **env** scope with its key set as **password** needs to be replaced with a new value. One of the data engineers suggests executing the following command:

**databricks secrets put --scope env --key password**

Which of the following correctly depicts the effect of executing this command?

* **This command will result in an error as the previous secret should have been deleted to run this command successfully.**
* **This command will be executed successfully and the old secret value will be replaced with the new one.**
* **The command will be executed successfully but the old secret value will remain intact.**
* **The command will fail as the syntax for the command is incorrect.**
* **The command will result in an error as the scope should be deleted before executing this command.**

**Explanation**

If you try to write a **secret** to a **scope** with an already **existing** key, the **old**secret value will be **replaced** by the **new** secret value. So, the command given in the question will be executed **successfully** and the **new** value will **replace** the old one.

More Info: [Replacing a Secret in Databricks](https://docs.databricks.com/security/secrets/secrets.html#:~:text=If%20you%20issue%20a%20write%20request%20with%20a%20key%20that%20already%20exists%2C%20the%20new%20value%20overwrites%20the%20existing%20value)

## Question 45:

A mobile company needs to delete the data(daily) of all the users who requested to cancel their SIM card in the last 24 hours to be compliant with their country’s privacy act. The company internally uses Delta tables(with default settings) to store the data of its users. How can the data engineer make sure that the data is permanently deleted and cannot be recovered under any circumstances?

* **The requested data should be deleted from the Delta table using the DELETE command.**
* **Use VACUUM command over the Delta table with the default retention period.**
* **The requested data should be deleted from the Delta table using the DELETE command, followed by the VACUUM command over the Delta table with the default retention period.**
* **The requested data should be deleted from the Delta table using the DELETE command, followed by the VACUUM command over the Delta table with 0 hours as the retention period.**
* **The requested data should be deleted from the Delta table using the DELETE command, followed by the VACUUM command over the Delta table with 24 hours as the retention period.**

**Explanation**

Let us look at each option one by one:

*The requested data should be deleted from the Delta table using the****DELETE****command.*

**INCORRECT!**As Delta tables support **time travel** through **versioning**, the data can be **recovered** after deletion.

*Use****VACUUM****command over the Delta table with the default retention period.*

**INCORRECT!** Using only the **VACUUM** command will **not** solve the above issue. The data must be **deleted** first.

*The requested data should be deleted from the Delta table using the****DELETE****command, followed by the****VACUUM****command over the Delta table with the default retention period.*

**INCORRECT!** Running the **VACUUM** command after deleting the data can **completely** delete the data but the **default** retention period is **7 days**. So, the data deleted within **7 days** can easily be **recovered** using the **time** **travel** feature of the Delta tables.

*The requested data should be deleted from the Delta table using the****DELETE****command, followed by the****VACUUM****command over the Delta table with****0 hours****as the retention period.*

**CORRECT!** Running the **VACUUM** command with **0 hours** as the retention period will **completely** **delete** the data from the system.

*The requested data should be deleted from the Delta table using the****DELETE****command, followed by the****VACUUM****command over the Delta table with****24 hours****as the retention period.*

**INCORRECT!** This is a tricky option as the question also states**24 hours.** Running the **VACUUM** command with **24 hours** as the retention period means that the data **deleted** in the last **24 hours** can be recovered using the **time travel** functionality.

More Info: [VACUUM command in Databricks](https://docs.databricks.com/sql/language-manual/delta-vacuum.html#:~:text=VACUUM%20removes%20all%20files%20from%20the%20table%20directory%20that%20are%20not%20managed%20by%20Delta%2C%20as%20well%20as%20data%20files%20that%20are%20no%20longer%20in%20the%20latest%20state%20of%20the%20transaction%20log%20for%20the%20table%20and%20are%20older%20than%20a%20retention%20threshold)

## Question 46:

A data engineer wants to restrict access to the **name** column used in the following dynamic view definition:

1. CREATE VIEW policy\_view AS
2. SELECT name, policy\_id, age
3. FROM policy

Which of the following SQL queries can be used to restrict access, allowing only the compliance team to query the **name** column?

* CREATE VIEW policy\_view("name": "compliance") AS

SELECT name, policy\_id, age

FROM policy

* CREATE VIEW policy\_view AS

SELECT

CASE WHEN

is\_member('compliance') THEN name

ELSE ''

END AS name,

policy\_id,

age

FROM policy

* CREATE VIEW policy\_view AS

SELECT

CASE WHEN

member('compliance') THEN name

ELSE ''

END AS name,

policy\_id,

age

FROM policy

* CREATE VIEW policy\_view AS

SELECT

CASE WHEN

member = 'compliance' THEN name

ELSE ''

END AS name,

policy\_id,

age

FROM policy

* CREATE VIEW policy\_view AS

SELECT name(IF 'compliance' in member), policy\_id, age

FROM policy

**Explanation**

This question checks your knowledge of **column-level permissions** on the **dynamic views**. Databricks supports **three** dynamic views functions:

1. **current\_user()**
2. **is\_member()**
3. **is\_account\_group\_member() (recently added)**

**current\_user()** gives you the name of the **current user** whereas **is\_member(group)** checks if the **current user** is a member of the **account-level group**. It returns **TRUE** if the **current user** is a member of the **account-level group.**

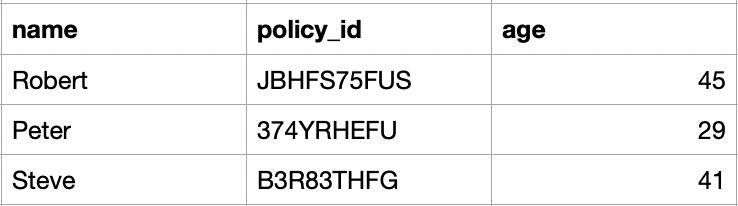
So, the correct answer should be:

1. CREATE VIEW policy\_view AS
2. SELECT
3. CASE WHEN
4. is\_member('compliance') THEN name
5. ELSE ''
6. END AS name,
7. policy\_id,
8. age
9. FROM policy

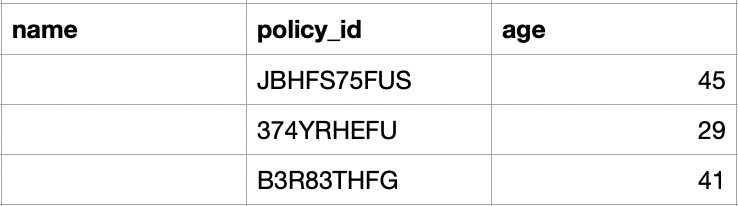
**IMPORTANT: is\_member() has now been converted to legacy, going forward is\_account\_group\_member() function will be used to check if the current user is a member of the account-level group or not. To read more about these recent changes**[**click here**](https://docs.databricks.com/data-governance/unity-catalog/create-views.html#:~:text=current_user()%3A%20Returns,level%20group%20membership.)**. You might still see is\_member()** **in the actual exam as this is one of the recent changes done by Databricks.**

If the **current user** is a **member** of **compliance** team, the **name** column will print the values whereas if the **current user** is **not** a **member** of **compliance** team the **name** column will print an **empty string.**

Hence, if the above query is run by one of the **members** of the **compliance** team, the result will be:



Also, if the above query is executed by a user who is **not** a **member** of the **compliance** team, the result would be:



More Info: [Column-level permissions on Dynamic Views](https://docs.databricks.com/data-governance/unity-catalog/create-views.html#:~:text=CREATE%20VIEW%20sales_redacted%20AS%0ASELECT%0A%20%20user_id%2C%0A%20%20CASE%20WHEN%0A%20%20%20%20is_account_group_member(%27auditors%27)%20THEN%20email%0A%20%20%20%20ELSE%20%27REDACTED%27%0A%20%20END%20AS%20email%2C%0A%20%20country%2C%0A%20%20product%2C%0A%20%20total%0AFROM%20sales_raw)

## Question 47

The newly imported CSV files contain PII data which is needed to be restricted until it becomes a part of a Delta table. The CSV files are currently stored in a folder of the workspace which is accessible to everyone. How should PII data be protected?

* **The data folder should be deleted forever to protect the PII data.**
* **The users should be given Can Manage permission on the folder containing PII data.**
* **The data should be replaced with the hash keys of the file names.**
* **The column containing PII data should be dropped after converting the CSV files to DataFrames.**
* **The folder permission should be changed to No Permissions for all the users.**

**Explanation**

To **restrict** the users from reading the data in the CSV files, **folder-level permissions** can be leveraged. It **restricts** the user’s **ability** to perform operations on a **specific folder.** Setting the folder permission to **No Permissions** can solve the above-stated problem.

More Info: [Folder-level permissions in Databricks](https://docs.databricks.com/security/access-control/workspace-acl.html#folder-permissions)

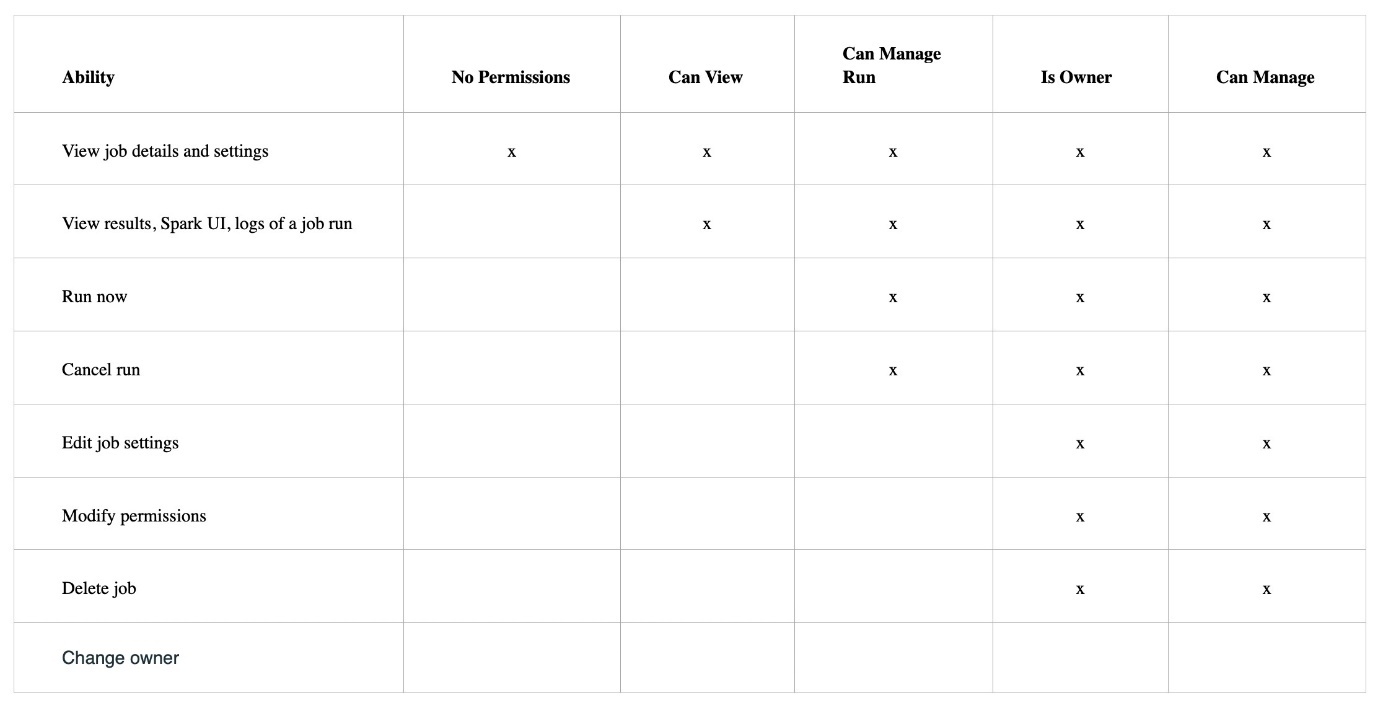
## Question 48:

A data engineer creates a job using the Databricks Jobs UI. Which of the following permissions is granted to the user being the creator of the job?

* **Is Creator permission**
* **Is Owner permission**
* **Can Manage permission**
* **Admin permissions**
* **No permissions**

**Explanation**

There are **5** sets of **job permissions** that can be granted to a user:

Source: [Databricks](https://docs.databricks.com/security/access-control/jobs-acl.html#job-permissions)

Coming to the answer to this question, the user who **creates** the job has **Is Owner** privileges.

More Info: [Jobs Permission Levels in Databricks](https://docs.databricks.com/security/access-control/jobs-acl.html#:~:text=The%20creator%20of%20a%20job%20has%20Is%20Owner%20permission)

## Question 49:

Which one of the following destinations is not supported for a Databricks SQL alert?

* **Microsoft Teams**
* **Slack**
* **Skype**
* **Webhook**
* **PagerDuty**

**Explanation**

As of now, the following **5** **destinations** are supported for an **alert**:

1. Microsoft Teams
2. Slack
3. Webhook
4. PagerDuty
5. Email

More Info: [Databricks SQL Alerts Destinations](https://docs.databricks.com/sql/admin/alert-destinations.html#:~:text=The%20following%20destinations,Microsoft%20Teams)

## Question 50:

A data engineer is working on a notebook that involves a large number of complex transformations. They want to know more about the working of each transformation. Which of the following can be used to print the Physical plan as well as various Logical plans?

* **explain()**
* **explain(logical=True, physical=True)**
* **explain(True, True)**
* **explain(True)**
* **explain**

**Explanation**

The **explain** method is usually used to print the **Physical plan** of a transformation but it can also be used to print the **Logical plans** - *Parsed Logical plan*, *Analysed Logical plan* and *Optimized Logical plan* as well as the **Physical plan.**

The following is the syntax for **explain** operation:

**DataFrame.explain(extended=None, mode=None)**

Adding **extended=True** will result in an **extended** display of the plans including all the **Logical plans**. Also note, if **explain** is used without any parameter, it will print only the **Physical plan.**

**Logical plans:** When you submit a spark application, a series of **logical plans** are generated by **Spark** in the below order:

*Parsed or Unresolved Logical plan:* As the name suggests, the **database**, **table** and **column** **names** are not resolved in this plan.

*Analyzed or Resolved Logical plan:* After the generation of the **Unresolved Logical plan**, the **Analyzed or Resolved Logical plan** is generated after resolving the **column**, **table** and **database** **names.**

*Optimized Logical Plan:* Lastly, the **Optimized Logical Plan** is created which reorders the **execution of commands.** It also combines the **same transformation** used multiple times on a DataFrame, if possible. For example, adding **two consecutive filter operations** on a DataFrame column will result in **combining** of the **two** **filter conditions**, resulting in **faster** execution.

**Physical Plan:** A **physical plan** acts as a **link** between **Logical Plans** and the **RDDs.** After the **Optimized Logical Plan**, multiple **Physical Plans** are generated by Spark. The plans are sent to the **Cost Model**, which selects the best plan. The **Physical Plan** is then converted to **RDDs.**

Also, you would notice different types of filters in the **Physical Plan** like **DataFilters**, **PushedFilters** and **PartitionFilters**. Remember, to avoid full table scans every time, Spark encourages making use of the **push-down predicate.** By using push-down predicates, you will avoid **unnecessary reads** from the source. The **PushedFilters** visible in the **Physical Plan** are nothing but **push-down predicates.**

If you want to read more about the **Physical plans** in Spark, you should visit this link.

[Physical Plans in Spark](https://www.databricks.com/session_eu19/physical-plans-in-spark-sql) | [Static Notebook](https://certification-champs.github.io/professional-data-engineer-notebooks/Practice%20Test%201/Question%2050.html)

*To view all the notebooks in this course, download [Notebooks.dbc](https://certification-champs.github.io/professional-data-engineer-notebooks/Notebooks.dbc) and import to your Databricks Account.*

More Info: [explain method in Spark](https://spark.apache.org/docs/3.1.2/api/python/reference/api/pyspark.sql.DataFrame.explain.html)

## Question 51:

Which of the following can be used to debug errors in Databricks notebook at the code level?

1. %debug
2. %pdb
3. %check

* **1, 2**
* **1, 2, 3**
* **2, 3**
* **1**
* **1, 3**

**Explanation**

**%debug** and **%pdb** can be used to **debug errors** on Databricks.

[Static Notebook](https://certification-champs.github.io/professional-data-engineer-notebooks/Practice%20Test%201/Question%2051.html)

*To view all the notebooks in this course, download [Notebooks.dbc](https://certification-champs.github.io/professional-data-engineer-notebooks/Notebooks.dbc) and import to your Databricks Account.*

More Info: [Debugging errors in Databricks](https://docs.databricks.com/_static/notebooks/python-debugger.html)

## Question 52:

Which of the following explains the usage of a SparkListener?

* **SparkListener can be used at the end of the spark application to call an application of another language.**
* **SparkListener is used for all the spark jobs as a wrapper around the SparkSession.**
* **SparkListener is used solely in Databricks to increase the speed of the spark applications.**
* **SparkListener can be used to view the metrics of a spark job.**
* **SparkListener is an always-on Databricks-specific tool, used for debugging spark code.**

**Explanation**

Just like **Spark UI** and **Ganglia Metrics**, **SparkListener** is used to view the **metrics** of a spark job.

More Info: [Using SparkListener in Spark](https://learn.microsoft.com/en-us/azure/databricks/kb/metrics/explore-spark-metrics)

## Question 53:

Which of the following pairs of Spark UI Tabs and the details they display are not correctly matched?

* **Stages Tab - Pool properties**
* **Storage Tab - Cached RDDs**
* **Environment Tab - Spark properties**
* **Jobs Tab - Thread Dump**
* **Executors Tab - Heap Histogram**

**Explanation**

**All** the pairs are **correctly** matched except the **fourth** pair. The **heap histogram** is visible only in the **Executors Tab** and **not** the **Jobs Tab.** To quickly look at the details of each Tab in Spark UI, you can quickly spin up a new cluster and run a few commands on a notebook attached to that cluster.

More Info: [Executors Tab in Spark UI](https://spark.apache.org/docs/latest/web-ui.html#executors-tab)

## Question 54:

An alert was created on the football match data. The status of the alert is being shown as **OK**. What can be interpreted from the current state of the alert?

* **OK state means that the alert is created successfully, but was never triggered.**
* **The OK state signifies that the alert is functioning correctly without any errors.**
* **OK state means that the alert may or may not be triggered in the past but the alert condition is not met in the most recent execution.**
* **The status of the alert being OK tells the user that the alert was triggered in the most recent execution.**
* **OK state is just an indicator of the successful deletion of the alert from the system.**

**Explanation**

There are **three** states available for an alert:

1. ***OK*** - The condition is **not** met for the most **recent** execution.
2. ***TRIGGERED****-*The condition was **met** for the most **recent** execution.
3. ***UNKNOWN*** - If **no data** is present or the **column**, on which the alert was set, is **missing.**

The status can change from **OK** to **TRIGGERED** and **TRIGGERED** to **OK**, based on the value of the column for the **recent** execution.

If the current status is **OK** and the condition is **met** in the most recent execution, the status changes from **OK** to **TRIGGERED**. Whereas, if the current status is **TRIGGERED** and the condition is **not** met in the most recent execution, the status is changed to **OK.**

More Info: [Databricks SQL Alerts](https://docs.databricks.com/sql/user/alerts/index.html#:~:text=TRIGGERED%20means%20that,Column%20you%20configured)

## Question 55:

The following Databricks notebook is exported as a **.ipynb** file without clearing the command outputs.

**Notebook:**

df = spark.read.options(inferSchema='True',delimiter=',', header=True).csv(“dbfs:/FileStore/athlete\_events.csv")

display(df)

Another user imports this notebook into their workspace. Which of the following statements is true about the accessibility of the command outputs and the Spark UI logs through the imported notebook?

* **The command outputs and the logs in Spark UI will be visible in the imported notebook if it is attached to the same cluster but will not be visible if it is attached to a different cluster.**
* **The command outputs and the logs in Spark UI will be visible in the imported notebook irrespective of the cluster to which it is attached.**
* **The command outputs and the logs in Spark UI will not be visible even if the notebook is attached to the same cluster.**
* **The command outputs will be visible but the logs in Spark UI will only be visible in the notebook if it is attached to the same cluster.**
* **The command outputs will be visible but the logs in Spark UI will not be visible in the imported notebook even if it is attached to the same cluster.**

**Explanation**

**Command outputs** - If the commands are **not cleared** before exporting the notebook, the user who imports the notebook **can** **also access** the command outputs.

**Spark UI logs** - The user importing the notebook will **not** be able to access the **logs** inside the notebook as the logs are **not** exported with the notebook.

## Question 56:

A task **clean\_data** running as part of the Databricks job got failed, while the task dependent on **clean\_data** still needed to be run. Which of the following scenarios correctly explains the outcome?

* **As the task clean\_data failed, all the previous tasks will be rolled back.**
* **After the failure of the clean\_data task, the successful operations of only the failed task will be rolled back.**
* **No rollback will be performed, any completed operation like writes to the disk or the Delta table before the failure of the clean\_data task will remain as is.**
* **All the tasks dependent on clean\_data will fail.**
* **The task(s) dependent on clean\_data will continue to run as per the execution plan.**

## Question 57

In the software engineering process in Databricks, who among the following is most likely to write a unit test for testing a function?

* **Developer of the function**
* **Databricks Administrator**
* **Unit tests are auto-generated for all Python and Scala functions**
* **SQL Analyst**
* **Tester(who will be testing that function)**

**Explanation**

In software development, the **unit tests** are written by the **developer**. The **functions**(*also known as units*) are tested by the developer by running a single or a series of **test cases**. Typically, the unit test cases **send** a value to a function and **expect** a value to be returned. Based on the return value obtained, the test case is said to be **passed**or**failed**.

In most languages including **Python** and **Scala**, a series of **assert** methods are used for writing the **unit test cases.**

## Question 58

A data engineer was attending a testing conference and heard the following lines:

"The following testing evaluates the system’s capability to work smoothly under the specific amount of workload which helps to test the reliability and scalability of a software"

The data engineer is unsure about the type of testing being discussed. Which of the following testing techniques is explained in the above statement?

* **Unit testing**
* **Smoke testing**
* **Functional testing**
* **Integration testing**
* **Performance testing**

**Explanation**

The above lines signify the underlying principle of **performance testing.** Performance testing is used to test the **workloads of a system.**Performance testing is very useful for systems that see a **spike** in the number of **users** during certain times of the **day**, **month** or **year** by determining stability during these **peak times.**

## Question 59:

The following is the set of commands in a Databricks notebook with the default language as Python:

1. data = dbutils.fs.ls('.')
2. list1 = ['']
3. list1.append(data)
4. %pip install pandas
5. import pandas as pd
6. df = pd.DataFrame(list1)
7. %pip uninstall pandas

Which of the following cells will be the first to return an error when run in a sequence?

* **list1 = ['']**
* **list1.append(data)**
* **%pip install pandas**
* **df = pd.DataFrame(list1)**
* **%pip uninstall pandas**

**Explanation**

Let us look at each command one by one:

***data = dbutils.fs.ls('.')***

This command will be executed **successfully** with file names being **appended** to the **data** variable.

***files = ['']***

***files.append(data)***

The list **files** will be created successfully and the **data** list will be appended to it.

***%pip install pandas***

***import pandas as pd***

The **pandas** library will be **installed** and **imported** without any error.

***df = pd.DataFrame(files)***

This is the command which will result in an **error**. After the installation of **pandas**, the python **interpreter** will be **restarted** which will delete the reference to the list **files**. Thus, this command will be the **first** to return an **error.**

Also note, to print the **list of libraries** already**installed,** you can run the following command:

**import sys**

**print(sys.path)**

[Static Notebook](https://certification-champs.github.io/professional-data-engineer-notebooks/Practice%20Test%201/Question%2059.html)

*To view all the notebooks in this course, download [Notebooks.dbc](https://certification-champs.github.io/professional-data-engineer-notebooks/Notebooks.dbc) and import to your Databricks Account.*

## Question 60:

A data engineer wants to pass the name of the task as a task parameter. Which of the following parameter correctly fulfills the requirement?

* **{{task\_id}}**
* **{{task\_name}}**
* **{{task\_key}}**
* **{{task}}**
* **{{name}}**

**Explanation**

Databricks support **templated task parameters** which are **replaced** by their **values** during the **run of the task.**For example, the **{{job\_id}}** parameter can be passed to a task with its value defined when the job is **triggered**. Similarly, **{{run\_id}}** parameter can be used to pass the **unique id** assigned to the **task run.**

According to the question, you need to pass the **name** **of the task.** The unique name of the task is specified by **{{task\_key}}**

Bottom of Form