1. What is the key difference between publishing a dashboard on Tableau Public versus Tableau Server/ Cloud in terms of accessibility and data security?

| Tableau Public | Tableau Server / Cloud |
|---|---|
| Dashboards are publicly accessible to anyone on the internet. | Dashboards are accessible only to authorized users. |
| No login required to view content. | Requires secure login and user authentication. |
| Best for portfolios, learning, and public data sharing. | Best for organizations and business reporting. |

2. When publishing a Tableau dashboard, what happens to a live connection compared to an extract?

When publishing a Tableau dashboard, a live connection continues to query the original database in real time. This ensures up-to-date data but makes performance dependent on the database's speed and network

conditions. It also increases database load when many users access the dashboard. In contrast, an extract creates a snapshot of the data stored on Tableau Server or Cloud in a highly optimized format. Extracts improve performance and reduce database dependency but require scheduled refreshes to maintain data accuracy. The choice between live and extract depends on whether the business prioritizes real-time data or performance and scalability.

3. How does this affect data refresh and performance? In Tableau Server or Cloud (conceptually), which feature allows administrators to control which users can view, interact with, or edit a published dashboard?

When publishing a Tableau dashboard to Tableau Server or Tableau Cloud, the choice between a live connection and an extract significantly affects both data refresh behavior and overall performance.

A live connection maintains a direct link to the original database. This means that every time a user opens or interacts with the dashboard, Tableau sends real-time queries to the source system. As a result, users always see the most current data available. However, performance is heavily dependent on the database's

speed, network stability, and query efficiency. If the database experiences high traffic or slow processing, dashboard performance may decline. Additionally, a large number of concurrent users can increase the load on the database, potentially affecting system responsiveness.

In contrast, an extract creates a snapshot of the data and stores it in Tableau's optimized Hyper format on the server. The dashboard queries this stored extract instead of the original database. This generally results in faster performance and improved scalability because Tableau's extract engine is optimized for analytics. It also reduces the workload on the source database. However, the data is not updated in real time. It refreshes only when a manual or scheduled extract refresh is performed. Therefore, data freshness depends on the configured refresh frequency.

In terms of access control within Tableau Server or Tableau Cloud, administrators manage user access through a permissions system based on role-based access control. Users are assigned specific site roles such as Viewer, Explorer, or Creator, each with defined capabilities. Permissions can be configured at multiple levels, including projects, workbooks, and data sources. Administrators can control whether users can view, interact with, download, edit, or publish dashboards. This structured permission framework ensures data security

while allowing controlled collaboration within the organization.

Live connections prioritize real-time data accuracy but depend on database performance, whereas extracts prioritize performance and scalability but require scheduled refreshes. Access and security in Tableau Server or Cloud are maintained through a robust permissions and role-based access control system.

4. If you want users to view a dashboard but not download the underlying data, which permission setting would you adjust in Tableau Server?

If the requirement is to allow users to view a dashboard but prevent them from downloading the underlying data, the permission setting that must be adjusted in Tableau Server or Tableau Cloud is the Download Data (and, if necessary, Download Full Data) permission.

In Tableau Server, administrators manage access through the Permissions system at the project, workbook, or view level. To meet this requirement, the administrator would grant users or groups the View permission so they can access and interact with the dashboard. However, the

Download Data and Download Full Data capabilities should be explicitly set to *Deny* or left ungranted.

By restricting these download permissions, users can:

- View the dashboard
- Apply filters and interact with visuals

But they cannot:

- Export underlying data
- Download full datasets
- Access raw data behind the visualization

This approach is commonly used in scenarios involving sensitive data, such as financial reports, healthcare records, or HR dashboards, where users need analytical insights but should not have direct access to raw data.

To allow dashboard viewing without permitting data extraction, the administrator must adjust the Download Data permission within the Tableau Server permissions framework while keeping the View permission enabled.

## 5. Name two methods by which a Tableau dashboard can be embedded into a web page or website.

A Tableau dashboard can be embedded into a web page or website using multiple methods. Two commonly used methods are Embed Code (iFrame) and the Tableau JavaScript API / Embedding API.

The first method is using the Embed Code (iFrame) option available after publishing a dashboard to Tableau Server or Tableau Cloud. Tableau generates an HTML `<iframe>` snippet that can be copied and pasted directly into a website's HTML code. This method is simple and requires minimal development effort. The dashboard appears inside the webpage and maintains interactivity such as filtering and tooltips. It is best suited for basic embedding where advanced customization is not required.

The second method is using the Tableau JavaScript API (Embedding API). This method allows developers to embed the dashboard with greater control and customization. Through JavaScript, developers can programmatically control filters, parameters, events, layout, and user interactions. It enables deeper integration with the website's functionality and provides a more dynamic user experience. This method is typically used in enterprise applications or custom web portals where interactive control and seamless integration are necessary.

Dashboards can be embedded either by using the simple iFrame embed code for quick integration or by leveraging the Tableau JavaScript API for advanced customization and interactivity within web applications.

6. List two best practices you should follow when preparing a dashboard for public sharing to ensure data security and privacy.

When preparing a Tableau dashboard for public sharing, especially on platforms like Tableau Public or publicly accessible Tableau Server links, it is essential to follow strict data security and privacy practices. Two key best practices are removing sensitive data and using data aggregation or anonymization techniques.

The first best practice is to remove or exclude any sensitive or confidential information before publishing. This includes personally identifiable information (PII) such as names, phone numbers, email addresses, patient IDs, employee details, financial account numbers, or internal business metrics that are not meant for public access. Even hidden fields or unused columns in the data source should be reviewed carefully, as published workbooks may allow access to underlying data depending on permission

settings. Only necessary and non-sensitive fields should be included in the final dashboard.

The second best practice is to aggregate or anonymize the data. Instead of displaying record-level details, data should be summarized at a higher level, such as totals, averages, or grouped categories. If individual-level analysis is required, identifiers should be masked or replaced with anonymous labels. Aggregation reduces the risk of exposing confidential information while still providing meaningful insights to viewers. This approach is especially important when dealing with healthcare, HR, financial, or customer-related datasets.

Before publicly sharing a Tableau dashboard, one must carefully remove sensitive data and ensure that the dataset is properly aggregated or anonymized. These steps help protect privacy, maintain compliance, and prevent unintended data exposure.

7. After publishing a dashboard, users report a “Data not found” error. What is the most common cause related to data source settings?

If users see a “Data not found” error after a dashboard is published, the most common cause related to data source settings is an incorrect or broken data source connection on Tableau Server or Tableau Cloud.

This usually happens when the published workbook is still pointing to a local file path or a database connection that the server cannot access. For example, if the dashboard was built using an Excel or CSV file stored on a developer’s local machine (such as `C:\Users\...`), Tableau Server will not be able to locate that file after publishing. Similarly, if database credentials were not embedded during publishing, the server may fail to authenticate and retrieve data.

Another common reason is that the data source was not properly published along with the workbook. If the workbook references a data source that exists only locally and was not uploaded to the server, users will encounter a connection error.

In summary, the most frequent cause of a “Data not found” error is that the server cannot access the original data source due to incorrect file paths, missing published data sources, or unembedded database credentials. Properly

publishing the data source and embedding credentials typically resolves this issue.

8. When embedding a Tableau dashboard into a secure internal portal, what two additional security measures should be implemented to ensure only authorized users can access the embedded content .

When embedding a Tableau dashboard into a secure internal portal, it is important to implement additional security measures to ensure that only authorized users can access the embedded content. Two critical measures are user authentication (SSO-based authentication) and proper permission management (role-based access control).

The first security measure is implementing strong user authentication, typically through Single Sign-On (SSO) using protocols such as SAML, OpenID Connect, or Active Directory integration. This ensures that users must log in with their organizational credentials before accessing the embedded dashboard. When the portal and Tableau Server or Tableau Cloud are integrated with SSO, the user's identity is verified automatically, and unauthorized external users cannot access the content. This prevents anonymous access and protects sensitive internal data.

The second essential measure is configuring proper permissions and role-based access control within Tableau Server or Tableau Cloud. Even after authentication, users should only see dashboards and data that they are authorized to access. Administrators should assign appropriate site roles (such as Viewer or Explorer) and define project, workbook, or data source-level permissions. In addition, row-level security can be implemented if users should see only specific portions of the data (for example, regional managers viewing only their region's data).

When embedding Tableau dashboards in a secure internal portal, organizations must ensure robust authentication mechanisms such as SSO and enforce strict permission settings through role-based access control. These measures work together to guarantee that only authorized users can access and interact with the embedded dashboard.