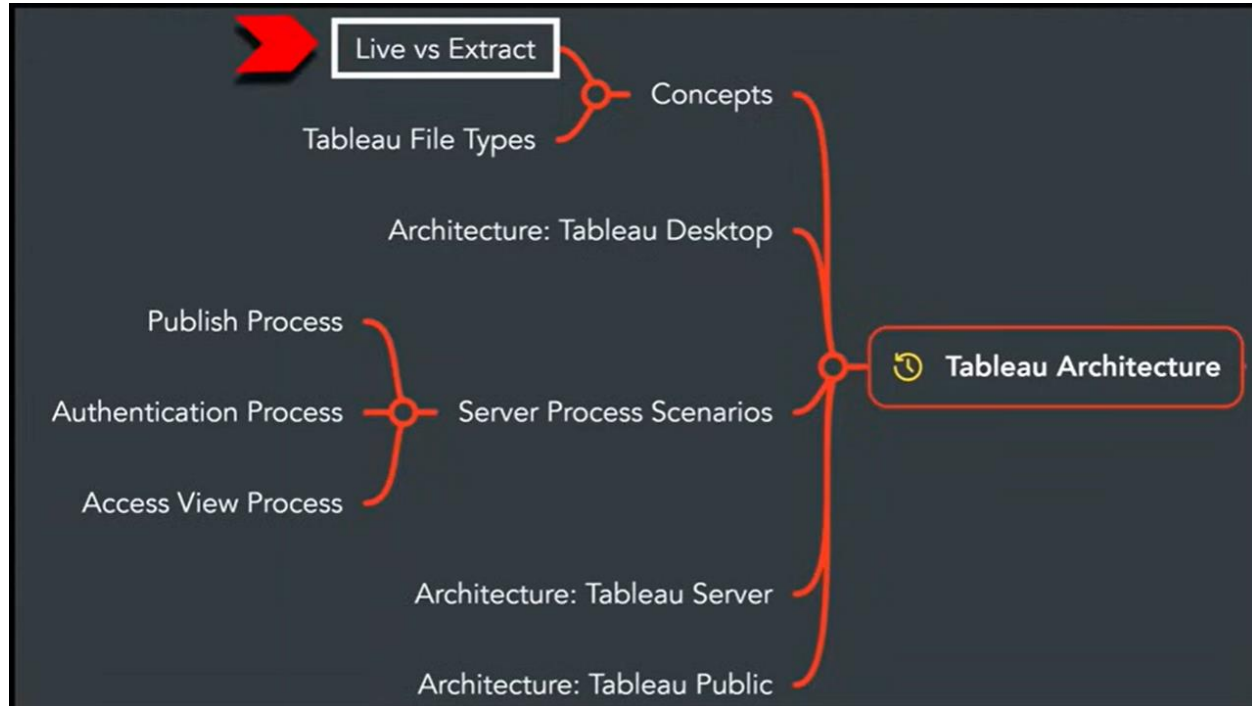
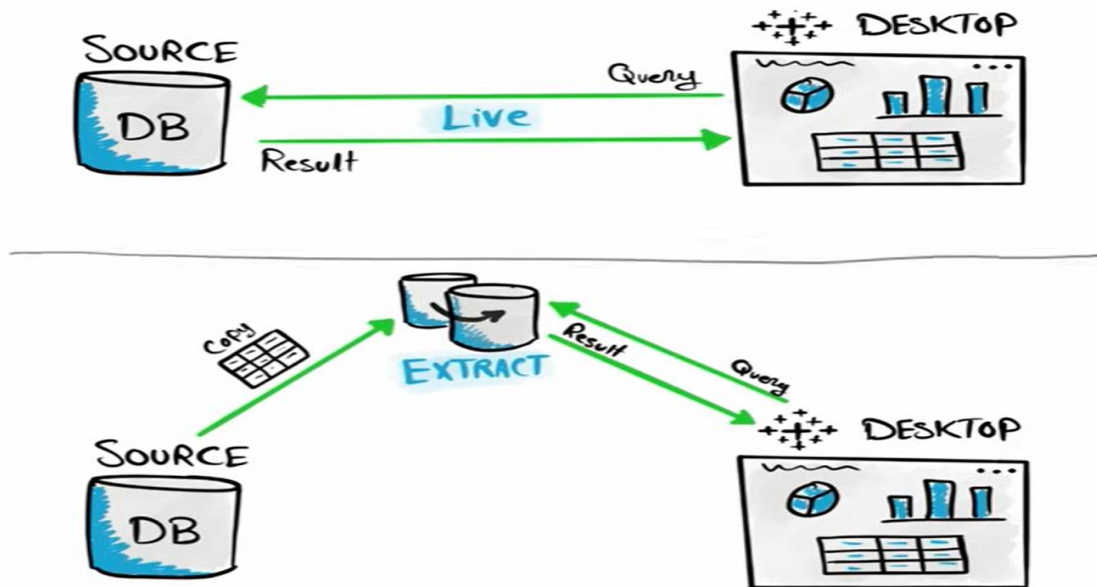


## Tableau Architecture:



## Live and Extract Connection:

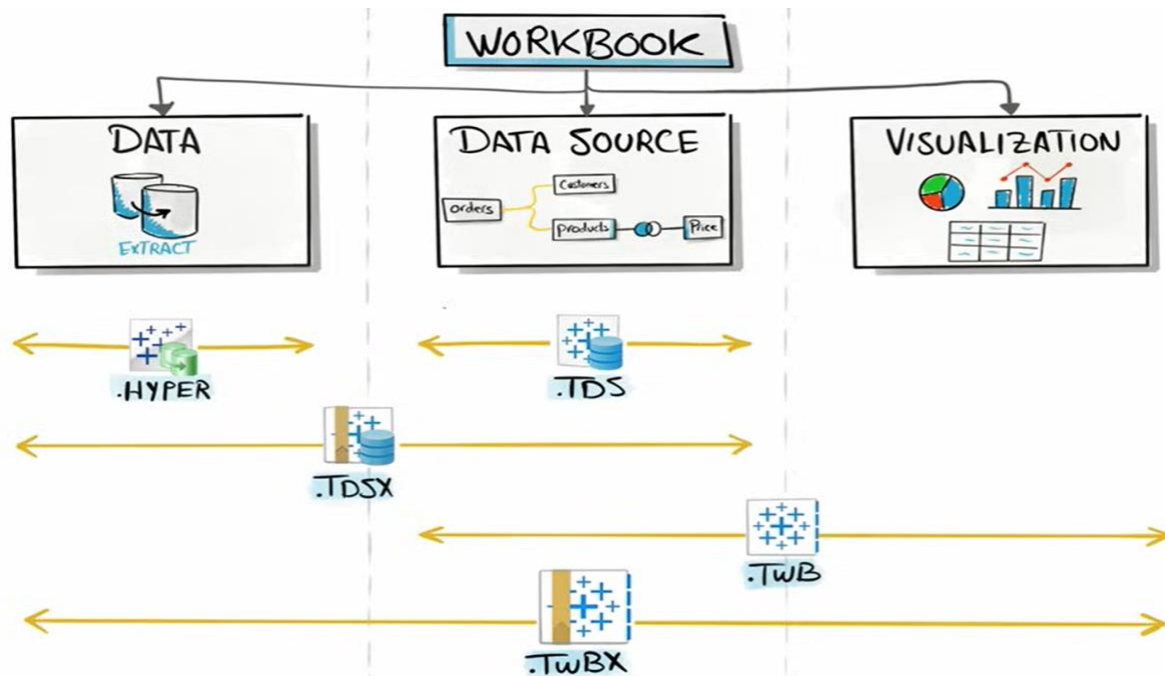
- ➔ Hmy ye decision lena ha k kya data ki aik copy leni tableau k ander ya direct data source se link krna, If no, hm data ki copy nhi lety Tableau me tou iskaliye hm usy source se direct connect krein gy, visuals direct query bhjy ga to the external source aur fresh data ho ga hmesha. This type of Connection is called **Live**.
- ➔ Aur agr hm data ki copy chahty inside Tableau, Tou external source ki aik copy Tableau me store ho jati. And this copy is called Extract. Each time Visuals sends a query to this. And then the extract is going to return results back to your visuals. Since the extract is near the visuals, we will get a great response time and very fast performance. This is called Extract Connection.
- ➔ Agr tm **Performance** ko zada priority de rhy then data performance then go with **Extract**. Agr **Data Freshness** zada zroori then **Live**.



### Tableau File types:

If you want to send files directly to users, then you need to decide the file type. Because in Tableau, we can generate 5 different file types. Tableau Workbook has 3 things: Data, Data source, and Visualization. Now there are 5 scenarios.

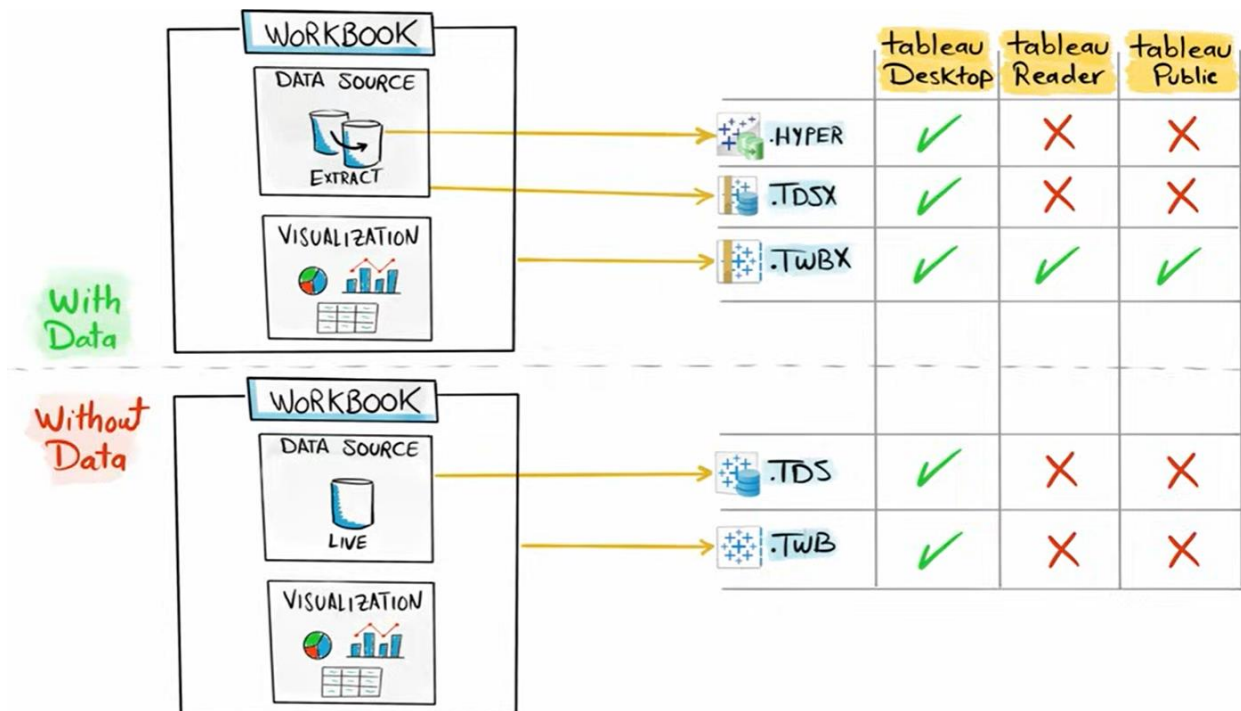
- ➔ If you want to share only Data, no data source, no visuals, then you can send an Extract as a .hyper format.
- ➔ If you want to show your data modeling, aggregations, then you can share your data source in .tds format.
- ➔ If you want to share Data and a data source, then you can send them as a Package in a .tdsx file type (Tableau packaged data source).
- ➔ If you want to share your data source with visualization, then send a file called .twb (Tableau workbook).
- ➔ If you want to send a complete package, then you can send a file in a .twbx (Tableau packaged workbook) format.



## Workbook Types:

2 different types of workbooks in Tableau: A workbook with data using the Extract Connection and a workbook with no data using the Live Connection.

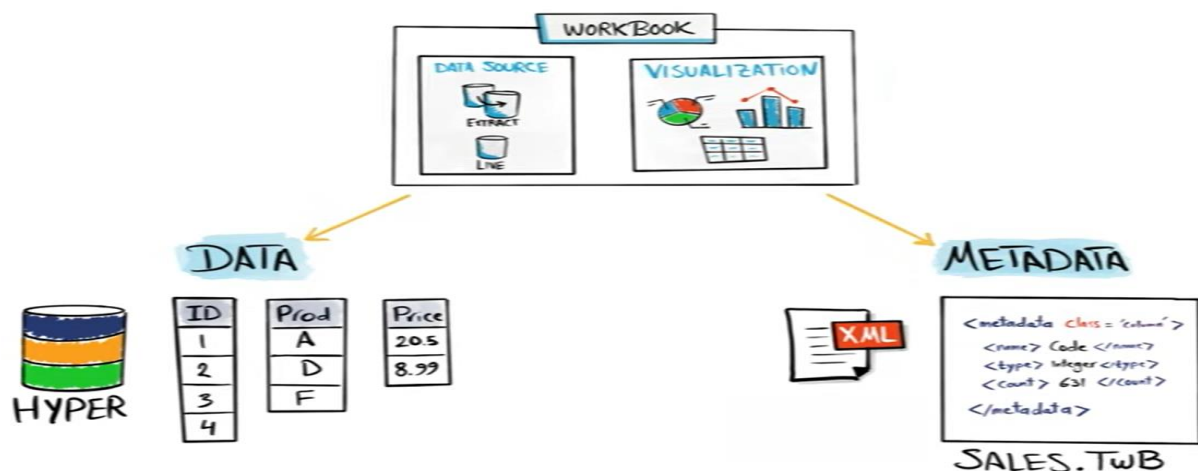
- ➔ In a workbook with data, you can send 3 different types of files. You can send the whole data in .hper format, or send the whole data in .tdsx format, or a complete package in .twbx format.
- ➔ In a workbook without data, you can send only two files. The dataset without data in .tds and the whole workbook in .twb format.
- ➔ Since the Tableau Public and Reader cannot directly connect to the data sources and cannot use live connection, so we can open only .twbx packaged workbook.



## Tableau Workbook Files:

Tableau uses two different types of data to store the workbook. The first one is the metadata information, which will be stored in XML files. Metadata is data about your data; it describes your data and contains all the information on what you have done in the workbooks. Anything you click, drag, and drop, or do while working with Tableau will be reflected in the metadata in some way. You can find information, for example, column names, data types, and data models.

The second type is data itself, the actual data. If you load data into Tableau, It is going to store in the format of .hyper file



## Tableau Architecture:

Tableau architecture contains 4 different layers. Source layer, Desktop layer, Server layer, and Consumer layer. We will work with this architecture from left to right. Start with the source layer and end up with the consumer layer.

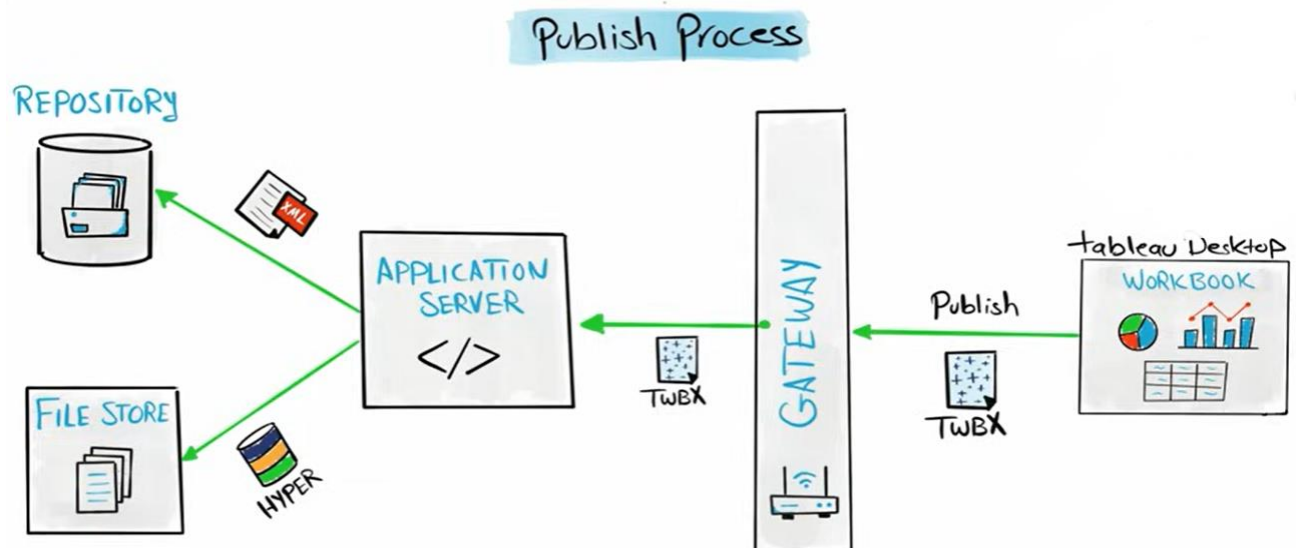
- ➔ **Source layer:** It is outside the Tableau and it contains the source of our data, so our data could be in databases like MySQL, or Oracle, or the data could be in files like Excel and JSON, or even in the cloud like Microsoft Azure, Amazon AWS, or even in APIs. So our data could be everywhere.
- ➔ **Desktop layer and its components:** The first component in the desktop layer is the data source. Before you start building your visualizations, you must set up the data source. The first step is to connect Tableau to our data. Tableau offers 90 different data connectors. Once you connect the Tableau with your data, the access information is going to be stored inside the data source, for example, the path of the file, the location of servers, username, and password, or access tokens, all this stored inside the data source. Here 2 types of connections in data sources are Extract and Live connections. The next thing we do in the data source is to start building a data model by combining tables using relationships, joins, and unions. And other custom stuff like renaming, etc.
- ➔ Once we have done with the data source and the connection, whether it's live or extract, we have our data model, and everything is ready. Now we are going to start building our visualization. Tableau organizes the visuals in 3 levels: the first is the worksheet. In Tableau, you can build a worksheet connected from 2 different sources by using a very powerful method called data blending. Where the data in one visual can form different sources. The next level is where we start combining these worksheets into one dashboard to show the different visuals in only one view. (Aur kisi visual me koi chg krna tou worksheet me wapis ja k krna ho ga,). The main goal of data visualization is to tell a story. So you can build a sequence of worksheets and dashboards that work together in order to tell the user a story based on data.
- ➔ Now on the desktop, we have a data source and visuals, and these 2 components are contained in a Tableau workbook.
- ➔ **Consumer layer:** There are different ways to consume Tableau visualization, depending on the clients and the tasks. We start with a very small group of users who might use Tableau Reader to view and interact with Tableau visualizations, and also send the Tableau file. If you want to share with your colleagues who want to build analyses on top of your work, they are going to use Tableau desktop. You can also publish your workbook on Tableau server and share the link with a group of consumers who interact with your visuals online and can edit them too if they have the necessary permissions. Or they can use the Tableau mobile app, but in the app, you can't edit them.

Next, you can export your workbook and visuals as PDF and Excel and share with static users.

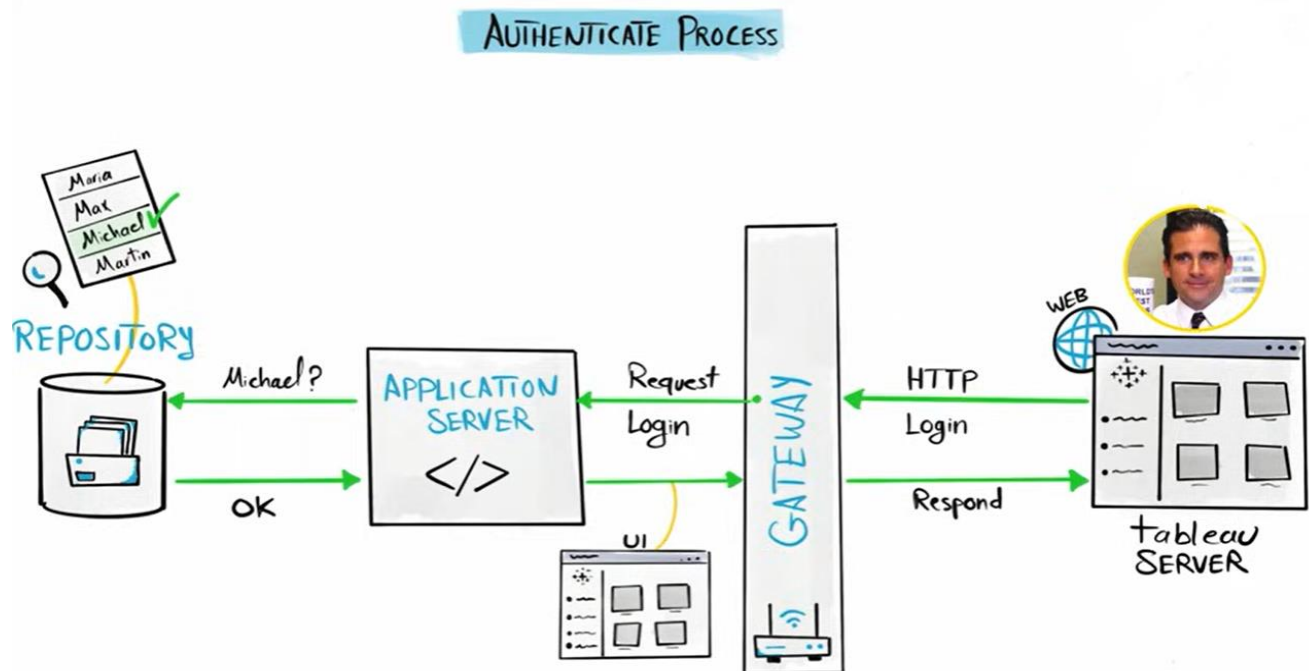
**Server Process Scenarios:** Try to understand with 3 scenarios from the user point of view that what gonna happen exactly in the Tableau server once we publish our workbook. Or when we log into the server and access the workbook.

**Server Workflow:**

- **Publish Process:** You want to publish a Tableau workbook with an extract. Tableau Desktop is gonna request the server to upload the workbook twbx. And the first component in Tableau Server that is gonna receive requests is the **Gateway**. It knows how to forward the request to the right server components. In this situation, the right component to process publishing is the application server. So, the gateway is gonna forward the request to it. As we learned that a Tableau workbook holds two different types of information: the metadata is stored in an XML file, and the data itself is stored in hyper files. And in Tableau Server these 2 different files are going to be stored in 2 different places. The application server sends the XML files to be stored in a server component called the Repository. And the hyper files are going to be stored in File Store. Gateway is responsible for the request to the right component.



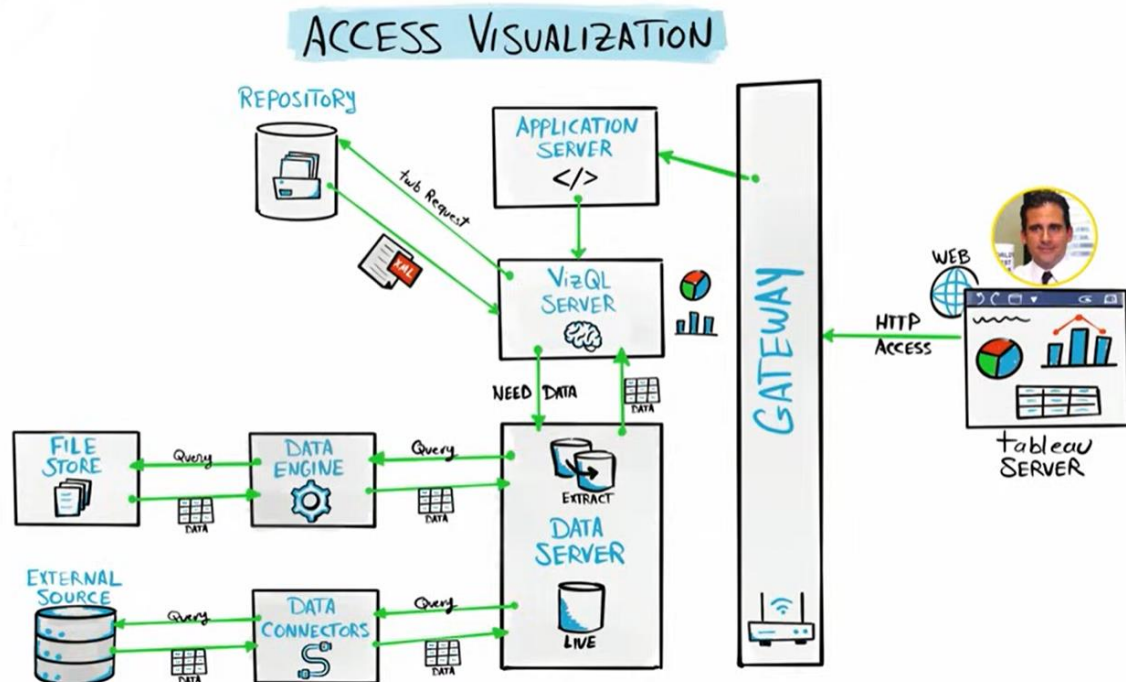
- Authentication Process:** Now our data and workbook are published to the Tableau server; now it's time for the users to log in and interact with the dashboards. Let's say your manager is Michal and he wants to check your sales dashboard in Tableau Server. Once Michal gives his info, a request is gonna be sent to the server as an HTTP request, the first thing that it's going to hit is the gateway. The gateway knows that the application server is the right component to handle the Authentication process. So, the gateway is gonna forward it to the Application server, and then the application server is gonna ask the Repository to check if the credentials username and password are correct. And if Michal has permission to access our server and then the repository is gonna check, and if everything matches the Michal is allowed to access the server. It will respond back to the Application server and respond that yes, we know Michal, he is in our records. The application server is going to start building UI and send it back to Gateway and it send it back to the Michal browser and now he is inside our tableau server.



- Access View Process:** Now, Michal is inside our Tableau server, and he is searching and browsing for the Sales Dashboard. Once he found it, he clicked on it and tried to access it. Now, let's see what's going to happen inside Tableau

Server. As usual, the HTTP request will be generated and sent to the server. We know that the Gateway receives the request and starts forwarding it to the right component: the application server and the application server is going to start rendering the Chrome around the viz, including all those icons and images that are not inside the dashboard itself. And the application server is going to say ok. Now we are talking about visualizations, which is completely out of my league, we have to forward this request to the master, it is the **VizQL** server, it is the one that deals with visualization. Here, VizQL is gonna take over and say ok. Let's check if Michal is allowed to see the Sales dashboard. SO the VizQL is gonna ask the Repository. Repos has the list of users and reports. Search for any match. If yes, then send back that Michal is allowed to see the Sales Dashboard. Now VizQL says All right, we need data. First, we need the metadata of the dashboard, and as you know that metadata is stored inside the Repository. SO, the VizQL again requests is to send the XML file of the dashboard. The repository sends it back, and then the server starts building the dashboard. Next, VizQL says ok, we have a dashboard, but it is empty. We need the data to fill it, and it is better to ask a data specialist which is data server. It says that for this dashboard, part of the data is already inside the Tableau server, but the other is outside the Tableau. To get the data inside from the Extract, the data server sends the query request to the data engine. And data engine knows how to query and extract the needed data from the file store. So the data engine gets the data from file store and send it back to Data server. Come to the part where the data is outside the Tableau server. Here the Data server act as a proxy, it uses data connectors to connect the external databases. Once the connection is established it gonna send a query that matches the language the database speaks. Then the databases return the needed data as Raw table. Once we have all the data it gonna do another security check that data server checks Is Michal allowed to see the data or should we filter the data? After checks, it send the data back to VizQL server. Now that VizQL has data for the dashboard, it turns it into visuals and puts it inside the workbook. The sales dashboard is complete and ready. VizQL sends it back to the Gateway and it sends it back to the web browser of Michal and he starts interacting.



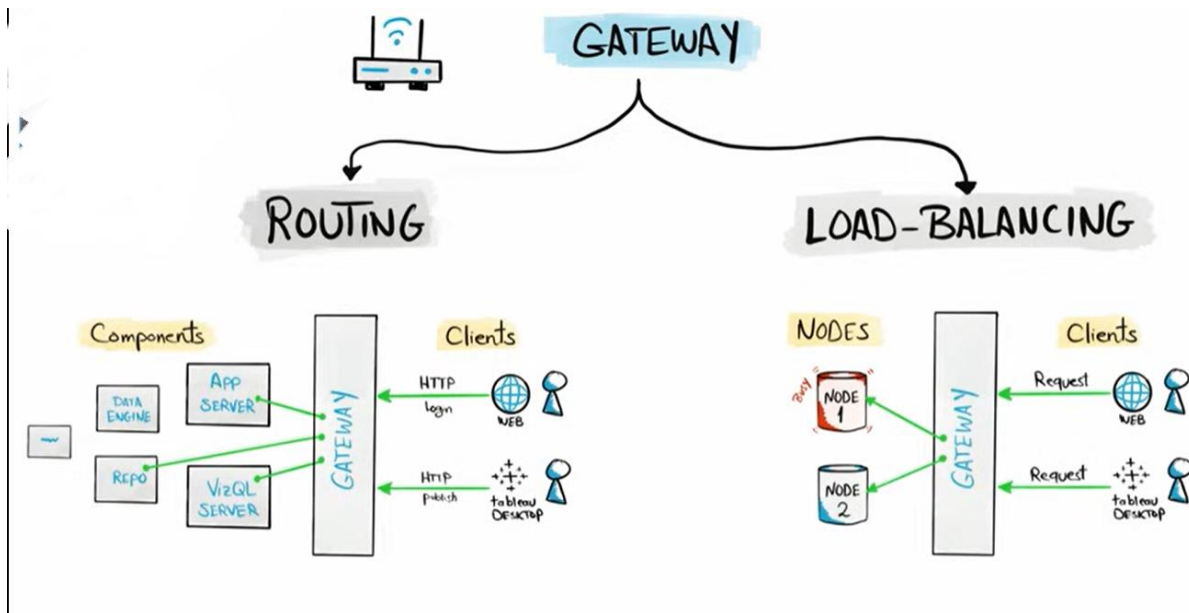


**Server layer Architecture:** The Server layer contains 2 interfaces, left and right, and in the middle, has a bunch of server components. The first interface is the data connector; they're gonna connect server components to external sources. And on the right, we have a gateway, it gonna receive requests from different clients to connect with Server components.

Let's deep dive into **Gateway**. We have requests coming from different clients, like a login request from a web browser, a publish request from Tableau Desktop. And on the other hand, we have different server components like App server, VizQL server, Data Engine, and Repository. The gateway in the middle of them knows how to forward different clients' requests to the right component.

The other task of Gateway is balancing stuff around. Suppose you're working in a multi-node environment where you have 2 nodes. When the gateway receives the request, it sends it to node 1. When it receives another request, it checks if node 1 is full, then sends it to node 2.

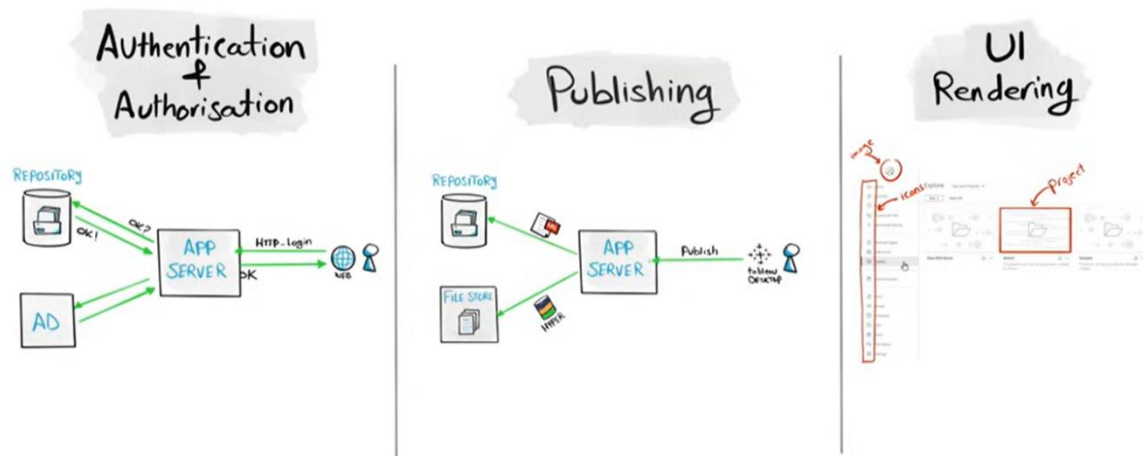
So, Gateway in Tableau server is a distributor that knows everything.



Now talk about the Tableau components that are in the middle. We have servers, engines, stores. As we learned in Tableau, there are different process like login process, publish, accessing workbook, and so on. And in tableau server they designed different servers to handle different processes. Start with the **APPLICATION SERVER**, it is responsible for different processes like log in request of users sent to Application server, It also gets the publishing request to publish a dashboard. And one more task for application server is to render the server interface, all the stuff you find in Tableau server like icons, images, projects, minu it is the application server who renders those stuff.

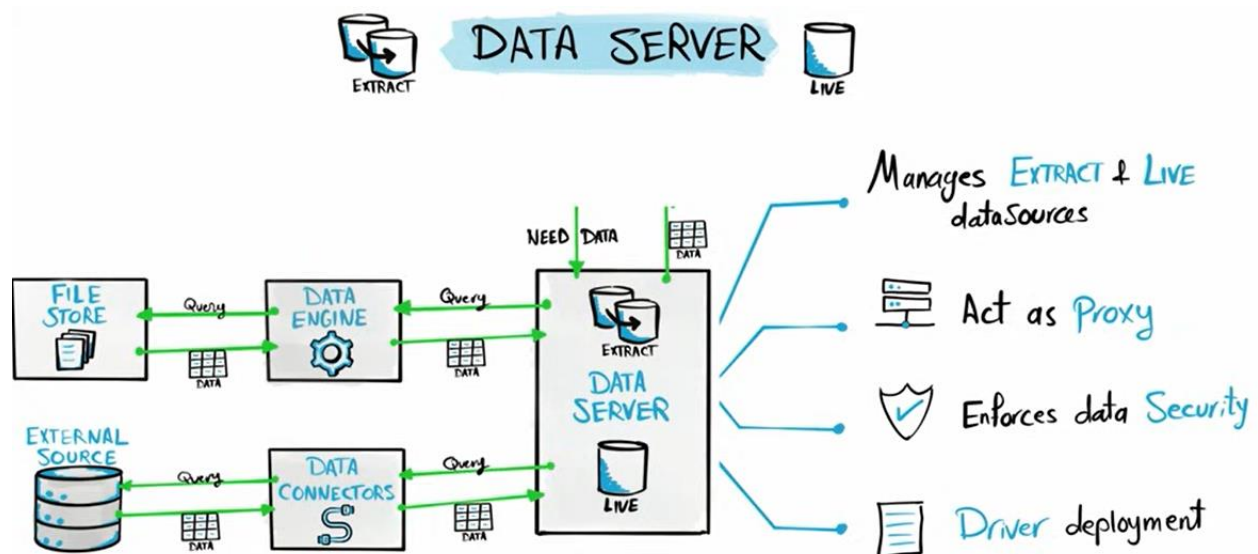
So , application server do all the above stuff, one process that application server will never do is the visualization process.

## </> APPLICATION SERVER



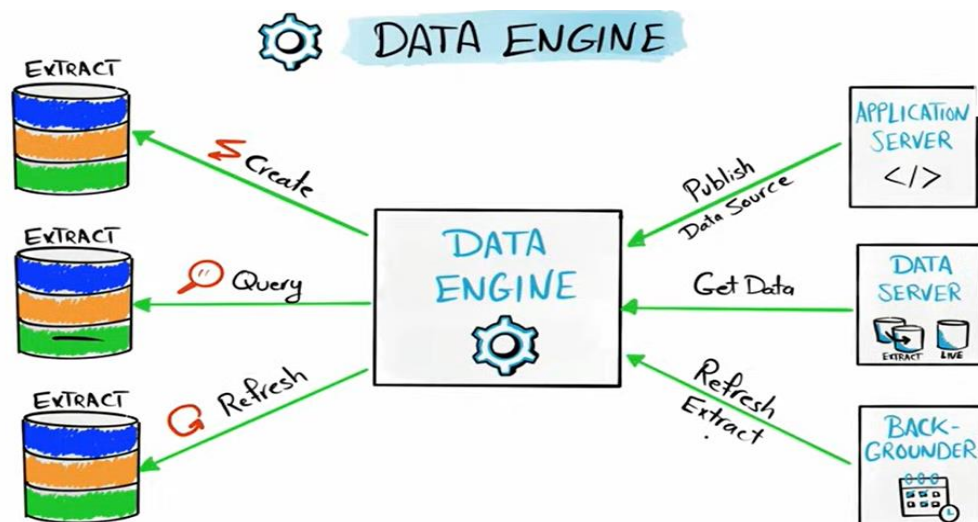
Next is the **VIZQL** server, which can do the magic by converting numbers and text into visuals and images. VizQL stands for Visual Query Language for databases. Let's say you drag and drop something into Tableau, and VizQL converts this action into a SQL query, sends it to the data server to get the data, and the data server sends the result back to VizQL as raw data. VizQL does the magic of converting these numbers into visuals. It is a very important Tableau component. Mainly, it does two things: it generates queries from actions and converts numbers into visuals.

Third is the **DATA SERVER**, which knows everything about data; it knows where to find it, how to connect to it, and how to speak to it. It manages extract and live data connections. If data is inside Tableau, it sends a request to the data engine; if outside Tableau, then it gonna use data connectors to send a query request to external resources. The data server acts like a proxy to the data sources. It can speak many different query languages (take two different databases to one language to query both). Another task is to handle data security, which checks if the user is allowed to see the data and does filtering on it. The data server manages the driver deployment. So the data server is a central data management component.

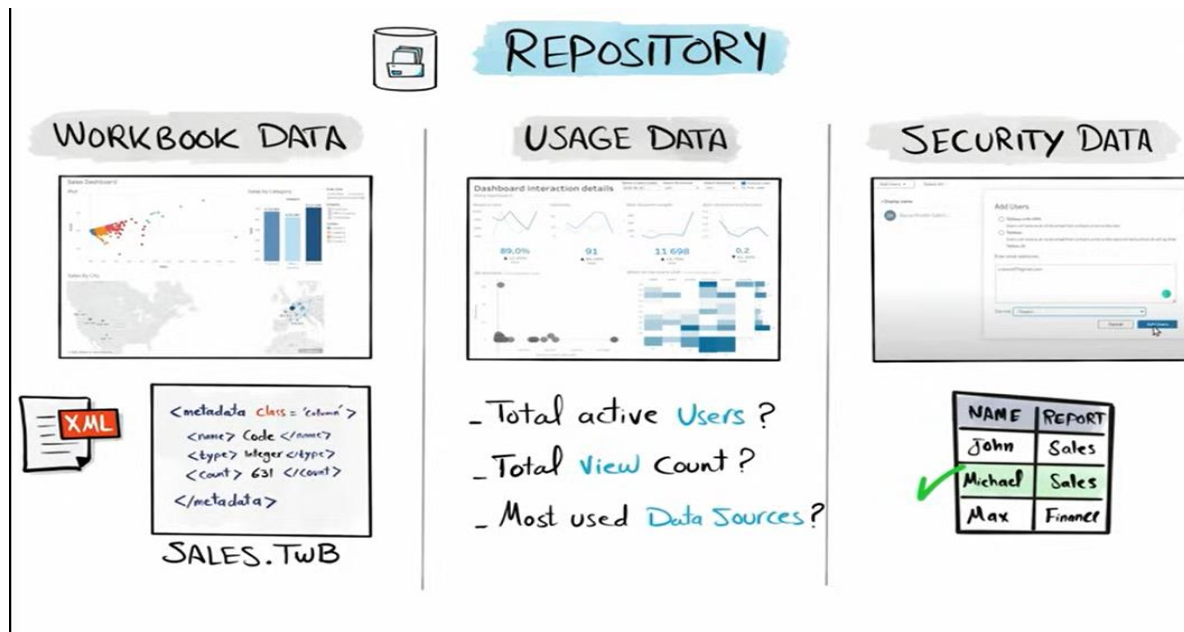


The fourth component is **the Data Engine**. If we decided to store our data inside Tableau as an extract, then the data engine gonna be the one dealing with it. The data engine can receive a new request from the application server to publish new data, then the data engine can create and execute an operation to create a new extract and store data inside it.

The Data Engine can receive a request from a data server requesting data. Here, the data engine searches for the correct extract. Data Engine can receive a request from a background process to update the content of an extract. So, it can execute and update operations by opening the extract and updating its content. Data Engine performs all these tasks only for the data inside Tableau, which means only for the extract.



Next component is **Repository**, It stores many different types of data, e.g: the workbook that we published to the server but only the metadata part not the data itself. So, the XML files from the workbooks are stored inside the repository. In repository we also find the usage data. This data helped you to understand the performance and traffic about your project. Another type of data you can find inside your repo is security information e.g: which user are allow to access your content or which users are allow to access the tableau server. It is important to understand that data inside our dashboards and reports are not stored inside repository.

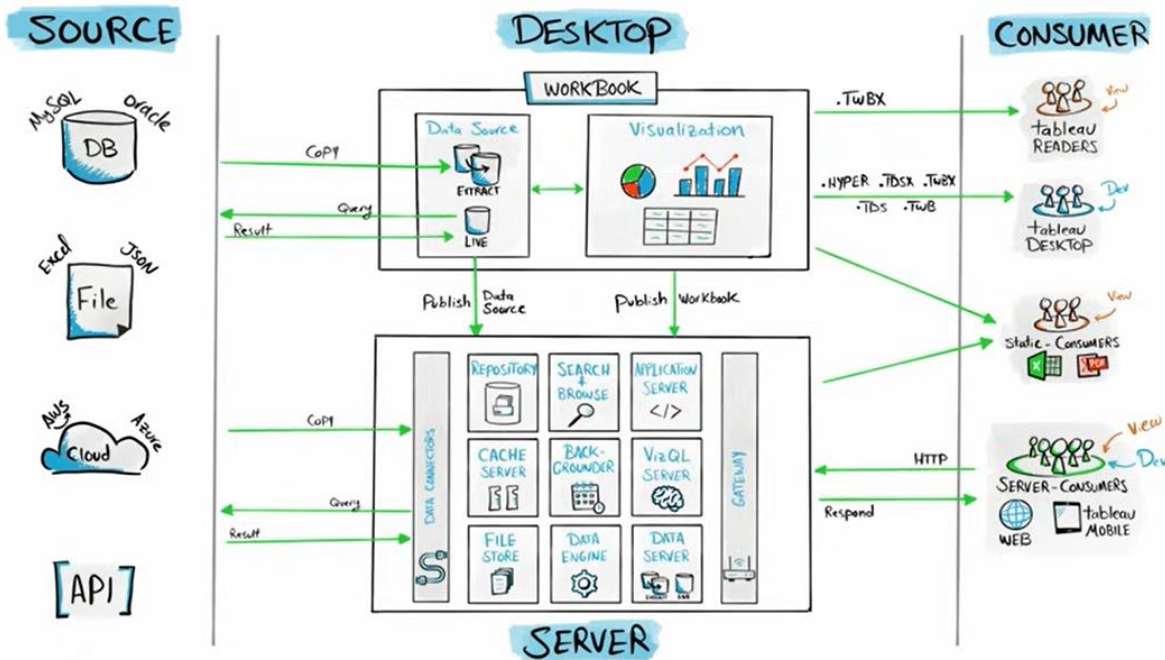


Many other worth-mentioning components, for example, the Cache Server, which stores almost everything, like Images, icons, results of queries, dashboards, and so on. If you start the dashboard that was already accessed before, then the data gonna be pulled from the cache server.

Another component is Backgrounder. In Tableau Server, you can create a schedule to refresh data inside your extract, and the task of backgrounder is to check the schedule every 10 seconds. If the time comes, then trigger the process of refreshing.

Last component is Search and Browse, the users can search for content and this component is responsible for searching inside the repository and return the results to user.

## The Big Picture:



Sources are outside of Tableau, jo k files ho skti ya databases. Tableau desktop me hm sources se connect krty either through live and extract connections, aur usy visualization bnaty, jo k hm aik hm workbook(source + visuals) bnaty, jisy jhm direct share, ya aik static file bna k , ya server pr publish b kr skty.

## Tableau Public Architecture:

In Tableau Public, you can only connect files like JSON, CSV, Excel, MS Access, and Google Sheets. The next component is Tableau Public Desktop. It is a free version of Tableau. We connect it with data only through the extract connection. After that, we're gonna start building visualizations. We have only one option to share the whole workbook on Tableau Public Server. They can now be consumed by users of the whole world.



## tableau Public Process

