**Documentation to apply the CLS in fabric Lakehouse tables.**

<https://learn.microsoft.com/en-us/fabric/onelake/security/get-started-data-access-roles>

Could you go through it and other similar sources and provide me a detailed guide with steps , pre-requisites, limitations for the above requirement

To apply **column-level security (CLS)** in Microsoft Fabric Lakehouse tables, the recommended approach is to use the SQL analytics endpoint or Warehouse capabilities, as direct CLS within OneLake table files is not natively supported. The steps, prerequisites, and limitations for enabling this scenario are detailed below, leveraging both Microsoft's documentation and reliable technical sources.

**Key Prerequisites**

* Fabric workspace with active capacity or trial.
* At least Admin, Member, or Contributor rights on the target workspace or SQL analytics endpoint.
* Data must reside in a Fabric Lakehouse and be accessible via a SQL analytics endpoint or Fabric Warehouse (CLS applies at this layer, not on raw OneLake files).

**Step-by-Step Guide**

**1. Set Up Workspace and Access**

* Log into Microsoft Fabric with an account having elevated access (Admin/Member/Contributor).
* Navigate to the workspace, then choose the Warehouse or configure a SQL analytics endpoint on your Lakehouse.

**2. Define Data Access Roles (for OneLake Folder-Level—Optional)**

* You may use OneLake data access roles for **folder-level** security if required; this is managed via “Manage OneLake data access” in the Lakehouse menu.
* Note: This folder-level access does not by itself provide column-level security.

**3. Assign Column-Level SELECT Permissions**

* Within the Warehouse or SQL analytics endpoint, identify:
  + The table to secure
  + Columns to restrict
  + User/group/role to assign permissions to
* Use the following T-SQL command:

GRANT SELECT ON YourSchema.YourTable (Column1, Column2, Column3) TO [UserOrRole];

* Replace:
  + YourSchema with the schema name (often dbo)
  + YourTable with your target table
  + The column list with the columns the role/user should access
  + [UserOrRole] with the user, group, or role’s name (preferably security groups for manageability).
* To **deny** access to certain columns (if needed for granular security):

DENY SELECT ON YourSchema.YourTable (SensitiveColumn) TO [UserOrRole];

* Repeat for each table and user/group combination as required.

**4. Test and Validate**

* Have target users connect via supported applications (e.g., Power BI, SQL tools). Users without explicit column permissions will receive errors when querying restricted columns.
* CLS applies at the **database tier**—all tools and API consumers (including Direct Query mode in Power BI) respect restrictions.

**Limitations and Considerations**

* Direct CLS is **not supported** for files/folders in OneLake accessed outside the SQL endpoint (e.g., via Spark, file APIs).
* Only **Microsoft Entra authenticated users** are supported for granular column permissions.
* Out-of-the-box UI for CLS is evolving; most configuration requires T-SQL.
* **Admins, Members, Contributors** with full workspace rights can override data access roles for folder-level controls but will be restricted by CLS at the SQL layer.
* OneLake folder-level security (OneLake Data Access Roles) is designed for folder/table visibility and read permissions, not column masking.
* Power BI in **Direct Lake mode** falls back to Direct Query when CLS is present, to ensure restrictions are enforced.

**Summary Table: Capabilities**

|  |  |  |
| --- | --- | --- |
| Capability | Folder/File (OneLake) | SQL Analytics Endpoint/Warehouse |
| Object/Table Security | Yes[[1]](#fn1) | Yes[[3]](#fn3) |
| Column-Level Security | No[[6]](#fn6) | Yes[[2]](#fn2)[[3]](#fn3)[[5]](#fn5) |
| UI Security Management | Yes (folders) | Partial (mostly T-SQL) |
| Entra ID Group Assignment | Yes | Yes |
| Affects Direct Lake Mode | No | Yes (enforces via Direct Query) |