1. 🔹 **Handling Large Datasets in Power BI Online Service**

Power BI Service supports different dataset storage & connection modes:

1. **Import Mode**
    - Data is compressed and stored in memory (VertiPaq engine).
    - Very fast for queries, but limited by dataset size quotas.
    - Pro license → max **1 GB per dataset**.
2. **DirectQuery Mode**
    - Data is not stored in Power BI; queries are sent to the source (SQL DB, etc.).
    - No size limits, but performance depends on the source system.
3. **Hybrid (Composite Models)**
    - Mix of Import + DirectQuery for balancing speed & size.

🔹 **Limitations with Pro License**

- Dataset size limit: **1 GB per dataset**.
- Refresh: up to **8 times/day**.
- Shared cloud capacity → performance can be inconsistent.

🔹 **Role of Premium Capacity**

Power BI Premium (P SKU for orgs, or PPU for individuals) provides **dedicated resources** in the Power BI Service.

Key Benefits:

1. **Larger Datasets**
    - Premium supports **up to 400 GB per dataset** (vs 1 GB in Pro).
    - Compression means this can represent **trillions of rows** in reality.
2. **More Frequent Refresh**
    - Up to **48 refreshes/day** (every 30 minutes).
    - Pro only allows 8/day.
3. **Dedicated Performance**
    - Your reports run on **dedicated cloud capacity**, not shared with others.

- Consistent speed, better for enterprise users.

4. **Advanced Features**
   - AI capabilities (AutoML, cognitive services).
   - Paginated reports.
   - Deployment pipelines (Dev → Test → Prod).
   - Multi-Geo support (store data in different regions).
5. **Unlimited Distribution**
   - With Premium, free users in your org can view shared reports (no Pro license needed).

🔹 **Example Scenario**

- A global retailer wants to analyze **5 years of sales data (~200 GB)**.
- With Pro: Impossible (limit 1 GB).
- With Premium: Possible (400 GB limit + compression).
- They can refresh every 30 minutes, keeping near real-time dashboards.

✅ **In short:**
Power BI Online Service handles large datasets using Import, DirectQuery, and hybrid models. But for **true enterprise scale** (bigger than 1 GB, high refresh frequency, consistent performance), you need **Premium Capacity**, which unlocks **larger dataset sizes, dedicated resources, and enterprise-only features**.

2. 🔹 **Power BI Data Connection Modes in the Service**

1. Import Mode

- **How it works**: Data is copied and compressed into Power BI's in-memory engine (VertiPaq).
- **Performance**: Very fast (data is already in memory).
- **Refresh**: Data becomes stale unless refreshed (up to 8/day with Pro, 48/day with Premium).
- **Storage**: Counts toward dataset size quota (1 GB Pro, 400 GB Premium).

✅ **Best for**:

- Small to medium datasets

- When you need **fast performance**
- When source system is slow/unavailable for real-time queries

❌ **Limitations**:

- Refresh needed
- Dataset size limits apply

---

2. DirectQuery Mode

- **How it works**: Data stays in the source; Power BI sends queries **live** at runtime.
- **Performance**: Depends on source system (can be slow if the DB isn't optimized).
- **Refresh**: Not needed (data always current), but metadata can still refresh.
- **Storage**: Very little space in Power BI since data isn't imported.

✅ **Best for**:

- Large datasets (can't fit into Import mode)
- Real-time reporting needs

❌ **Limitations**:

- Slower than Import (query round trips)
- Limited DAX functionality (not all functions supported)
- Puts load on the source system

---

3. Live Connection

- **How it works**: Similar to DirectQuery, but connects specifically to **Analysis Services models (SSAS, AAS, or existing Power BI datasets)**.
- **Data storage**: No data is stored in Power BI; it just queries the model directly.
- **Modeling**: Can't create new tables/relationships in Power BI Desktop — must use the model as-is.

✅ **Best for**:

- Centralized, curated **semantic models** (e.g., corporate data warehouse in SSAS).
- Consistent "single source of truth" across reports.

❌ **Limitations**:

- No data mashup (only 1 live source per report).
- Modeling restricted to the source.
- Performance depends on SSAS/Power BI dataset performance.

🔹 **Quick Comparison Table**

Feature	Import	DirectQuery	Live Connection
Data Storage	In Power BI (VertiPaq)	In source	In source (SSAS/PBI dataset)
Performance	⚡ Very fast	⚠ Depends on source	⚠ Depends on source model
Refresh Needed	Yes (8/day Pro, 48 Premium)	No (always live)	No (always live)
Dataset Size Limit	1 GB (Pro), 400 GB (Premium)	None	None
DAX Support	Full	Limited	Limited
Modeling Flexibility	High	Medium	Low (no new tables/relations)
Best For	Small/medium data, fast reporting	Huge datasets, real-time	Centralized enterprise models

---

✅ **In short:**

- **Import** = Fastest, but limited by dataset size & refresh.
- **DirectQuery** = No size limits, real-time, but slower & limited.
- **Live Connection** = Centralized corporate models, but very restrictive for modeling.

3. 🔹 **What are Deployment Pipelines?**

Deployment pipelines help you **manage, test, and release Power BI content** in a structured way.

- Think of them as **Dev → Test → Prod** environments for Power BI.

- They ensure changes are validated before reaching business users.

🔹 Stages in a Deployment Pipeline

1. **Development (Dev)** 🛠️
    - Workspace for developers to build and experiment with reports, datasets, and dashboards.
    - Frequent changes are expected.
    - Only the dev team usually has access.
2. **Test (or QA)** 🔍
    - Workspace for testing and validation.
    - Business analysts, testers, or selected users verify:
        - Data accuracy
        - Visual correctness
        - RLS rules
        - Performance
    - Ensures everything works as expected before rollout.
3. **Production (Prod)** 🚀
    - Workspace for end users (read-only access).
    - Content here is **stable, validated, and approved**.
    - Distributed via **Apps** for broad usage across the organization.

🔹 Key Features of Deployment Pipelines

- **Content Promotion**: Move content from Dev → Test → Prod with one click.
- **Change Detection**: Shows differences between stages (e.g., new reports, modified datasets).
- **Selective Deployment**: You can choose what to promote (not everything at once).
- **Data Source Rules**: You can configure connections differently per stage.
    - Example:
        - Dev → Test DB

- Test → QA DB
- Prod → Production DB

🔹 **Benefits for Enterprises**

✅ Consistency — same pipeline structure across projects
✅ Governance — control who can publish at each stage
✅ Faster releases — smooth transitions instead of manual uploads
✅ Reduced risk — test before production rollout

✅ **In short:**
Deployment pipelines in Power BI Online give you **Dev → Test → Prod** stages for managing datasets, reports, and dashboards. They allow controlled, secure, and consistent promotion of content in enterprise environments.

4. 🔹 **Power BI + Microsoft Teams Integration**

Power BI and Teams work together so reports are available right where people chat and collaborate.

1. Power BI App inside Teams

- You can install the **Power BI app** directly in Teams.
- This lets you:
  - Browse and search reports without leaving Teams
  - Pin important reports to the Teams left navigation
  - Access shared workspaces

2. Embed Reports in Teams Channels/Chats

- In a Teams channel → click **+ (Add a tab)** → choose **Power BI** → select a report.
- Members can view and interact with the report without leaving Teams.
- Example: A Sales team can view a live Sales dashboard right in their Teams channel.

3. Chat about Data

- You can copy a link to a Power BI visual/report and paste it in Teams chat.
- The report opens interactively for people with permissions.

4. Notifications in Teams

- Using **Power Automate**, you can send alerts from Power BI to Teams when KPIs are reached or thresholds are broken.

🔹 Power BI + SharePoint Online Integration

1. Embed Reports in SharePoint Pages

- Use the **Power BI web part** in SharePoint Online.
- Insert a live interactive Power BI report into a SharePoint modern page.
- No need to export static charts — users see real-time reports.

2. Permissions Handling

- Report permissions in Power BI **must match** SharePoint access.
- If someone doesn't have access in Power BI, embedding in SharePoint won't bypass security.

3. Use Cases

- Internal portals (e.g., HR dashboards, project updates, finance KPIs)
- SharePoint becomes a central hub for both documents and live analytics.

✅ Summary

- **Teams** → best for **daily collaboration** (chat, channels, tabs with live reports).
- **SharePoint** → best for **company portals** (embed live reports in intranet pages).

Both integrations ensure that **data meets people where they already work**, instead of forcing them to open Power BI separately.

5. 🔹 Power BI + Microsoft Teams Integration

Power BI and Teams work together so reports are available right where people chat and collaborate.

1. Power BI App inside Teams

- You can install the **Power BI app** directly in Teams.
- This lets you:
    - Browse and search reports without leaving Teams

    - Pin important reports to the Teams left navigation
    - Access shared workspaces

2. Embed Reports in Teams Channels/Chats

- In a Teams channel → click **+ (Add a tab)** → choose **Power BI** → select a report.
- Members can view and interact with the report without leaving Teams.
- Example: A Sales team can view a live Sales dashboard right in their Teams channel.

3. Chat about Data

- You can copy a link to a Power BI visual/report and paste it in Teams chat.
- The report opens interactively for people with permissions.

4. Notifications in Teams

- Using **Power Automate**, you can send alerts from Power BI to Teams when KPIs are reached or thresholds are broken.

🔹 **Power BI + SharePoint Online Integration**

1. Embed Reports in SharePoint Pages

- Use the **Power BI web part** in SharePoint Online.
- Insert a live interactive Power BI report into a SharePoint modern page.
- No need to export static charts — users see real-time reports.

2. Permissions Handling

- Report permissions in Power BI **must match** SharePoint access.
- If someone doesn't have access in Power BI, embedding in SharePoint won't bypass security.

3. Use Cases

- Internal portals (e.g., HR dashboards, project updates, finance KPIs)
- SharePoint becomes a central hub for both documents and live analytics.

✅ **Summary**

- **Teams** → best for **daily collaboration** (chat, channels, tabs with live reports).

- **SharePoint** → best for **company portals** (embed live reports in intranet pages).

Both integrations ensure that **data meets people where they already work**, instead of forcing them to open Power BI separately.

5. 🔹 **What is the XMLA Endpoint?**

- **XMLA (XML for Analysis)** is a protocol that allows external tools to connect to **tabular models** (semantic models/datasets) in Power BI.
- Available only with **Premium capacity** (and Power BI Premium Per User).
- In practice, it makes a Power BI dataset behave like an **Analysis Services (SSAS) Tabular model** in the cloud.

The XMLA endpoint exposes your **Power BI datasets** for **read and write** operations depending on configuration:

- **Read-only mode:** Connect and query the dataset (e.g., with SSMS, Excel, or 3rd-party BI tools).
- **Read/Write mode:** Advanced tasks like automation, metadata editing, partitions management, calculation groups, etc.

🔹 **Benefits for Developers & Enterprise BI Teams**

1. Integration with External Tools

- Connect from **SQL Server Management Studio (SSMS)**, **DAX Studio**, **Excel**, or **3rd-party BI tools**.
- Run **DMVs** (Dynamic Management Views) to analyze dataset performance, model size, and queries.
- Treat Power BI like an **enterprise-grade OLAP engine**.

2. Advanced Modeling & Automation

- Manage partitions (useful for **incremental refresh** beyond UI limitations).
- Apply **tabular editor scripts** for calculation groups, measures, and metadata updates.
- CI/CD pipelines: Automate deployment of dataset changes via **DevOps**.

3. Enterprise-Scale Semantic Layer

- Multiple BI tools (Power BI, Excel, Tableau, etc.) can query the same **certified dataset** via XMLA.
- Promotes **"single source of truth"** for KPIs across the organization.

4. Performance Tuning

- Developers can capture and analyze query performance using **SQL Profiler** through XMLA.
- Helps optimize DAX queries, storage engine vs. formula engine bottlenecks.

5. Interoperability with Analysis Services

- Makes migration between **SSAS Tabular** and **Power BI Premium** easier.
- Enterprises can treat Power BI Premium as a **cloud SSAS replacement**.

🔹 **Example Use Cases**

- Finance team runs **Excel pivot tables** directly against a large certified Power BI dataset.
- BI developers use **Tabular Editor** via XMLA to push advanced calculations.
- DevOps pipelines push metadata/model changes automatically from Git → Premium dataset.
- IT team monitors query performance using SSMS.

✅ **In short:**
The **XMLA endpoint** turns Power BI Premium into a **full-fledged enterprise semantic model platform**, enabling advanced management, integration, automation, and scalability.

6. 🔹 **1. Usage Metrics**

- **What it is:** Built-in Power BI feature that shows **report and dashboard usage** (how many views, by whom, when).
- **Scope:** Per report or dashboard.
- **Where to find it:**
  - In Power BI Service → Open a report or dashboard → **More options (…) → Usage metrics**.
- **What it shows:**
  - Number of views over time.
  - Unique users viewing the report/dashboard.
  - Which content is most popular.

    - Which users are most active.
- **Purpose:** Helps **report owners** understand engagement (is my report being used? which pages/filters are most popular?).
- **Limitations:**
    - Only available for reports/dashboards in a workspace.
    - Not a full audit trail (only high-level usage).

🔹 **2. Audit Logs**

- **What it is:** A more detailed, security-focused log of **user activities** across Power BI, available through **Microsoft 365 Compliance Center**.
- **Scope:** Organization-wide (not just one report).
- **Where to find it:**
    - Microsoft 365 Security & Compliance Center → **Audit Logs** → Search for Power BI activities.
    - Or via **PowerShell/Graph API**.
- **What it shows:**
    - Report views, exports, sharing actions, dataset refreshes.
    - Who did what, when, and where (including IP/location).
    - Admin-level auditing for compliance & security.
- **Purpose:** Helps **admins** monitor activity for governance, compliance, and security auditing.
- **Requirements:**
    - Requires **Power BI Pro/Premium** and appropriate **Microsoft 365 auditing** enabled.

🔹 **Key Difference**

- **Usage Metrics:** For **content creators** → "How often is my report/dashboard being used?"
- **Audit Logs:** For **admins/compliance** → "Who accessed/exported/shared what, and when?"

✅ **Example:**

- A report author uses **Usage Metrics** to see if a sales dashboard is popular among regional managers.
- An IT admin uses **Audit Logs** to track whether someone exported sensitive data from that dashboard.

7. 🔹 **Steps to Manage Workspace Access**
   1. **Go to the Workspace**
      - In Power BI Service → Left panel → **Workspaces** → Select the workspace you want.
   2. **Open Access Settings**
      - Click **Workspace settings** (gear icon) → **Permissions**.
   3. **Add People or Security Groups**
      - You can assign individual users, Microsoft 365 Groups, or security groups.
      - Adding groups is recommended (easier to manage at scale).
   4. **Assign Roles (Permission Levels)**
      - Choose one of the predefined roles (see below).

🔹 **Workspace Roles & Permissions**

Role	Can View Reports/Dashboards	Can Edit Content	Can Publish/Update Datasets	Manage Workspace Settings
Viewer	✅ Yes	❌ No	❌ No	❌ No
Contributor	✅ Yes	✅ Yes (edit/create)	✅ Yes	❌ No
Member	✅ Yes	✅ Yes	✅ Yes	✅ Limited (add users)
Admin	✅ Yes	✅ Yes	✅ Yes	✅ Full control

🔹 **Best Practices**

- ✅ Use **security groups** instead of individuals to simplify management.

- ✅ Assign **Viewer** role for business users who just consume reports.
- ✅ Assign **Contributor/Member** only to report developers or analysts.
- ✅ Limit **Admin** to a few trusted people (workspace owners).
- ✅ For publishing to a wide audience, use **Apps** (read-only packaged content) instead of giving direct workspace access.

✅ **Example:**

- Sales managers = **Viewer** (they just consume dashboards).
- BI developers = **Contributor** (they can build and publish reports).
- IT admin = **Admin** (full workspace control).

8. 🔹 **1. Access Control & Permissions**
   - **Workspaces roles** (Viewer, Contributor, Member, Admin) control who can view, edit, or manage content.
   - **Row-Level Security (RLS)** restricts access to specific rows of data based on user roles.
   - **Object-Level Security (OLS)** (Premium only) hides entire tables or columns.
   - **Conditional Access policies** (via Azure AD) restrict access based on device, location, or MFA.

🔹 **2. Data Classification & Sensitivity Labels**

- Apply **Microsoft Information Protection (MIP) sensitivity labels** to datasets, reports, and dashboards.
- Labels travel with the data (Excel, PDF, etc.), so exports remain protected.
- Example: “Confidential – Finance Only” → prevents unauthorized sharing.

🔹 **3. Certified & Promoted Datasets**

- **Promoted datasets** = recommended for broader use.
- **Certified datasets** = validated by data stewards.
- Helps enforce a **single source of truth** instead of multiple versions of the same data.

🔹 **4. Monitoring & Auditing**

- **Usage Metrics** → Track report/dashboard usage at a high level.
- **Audit Logs** (via Microsoft 365 Compliance Center) → Detailed activity trail (who viewed, shared, exported, or refreshed data).
- **Admin portal** → Monitor dataset refresh failures, capacity usage, and permissions.

🔹 **5. Dataflows & ETL Governance**

- Centralize transformation logic in **Dataflows** (instead of duplicated Power Query in multiple reports).
- Store and manage **business rules once**, enforce consistency across reports.

🔹 **6. Scheduled Refresh & Gateways**

- Use **scheduled refresh policies** to keep data accurate and up to date.
- Manage on-premises data securely with **Gateways** and restrict who can publish or refresh datasets.

🔹 **7. Capacity & Performance Management**

- **Premium Capacity** allows resource governance (memory, refresh concurrency, query limits).
- **Deployment pipelines** enforce structured dev → test → prod content promotion.

🔹 **8. Governance Policies & Training**

- Define clear **naming conventions** for workspaces, datasets, and reports.
- Create **data stewardship roles** (owners responsible for certified data).
- Educate users on **self-service BI best practices** while keeping guardrails in place.

✅ **Summary:**
Power BI Service enforces governance through a mix of **security (RLS,**

permissions), compliance (labels, audit logs), consistency (certified datasets, dataflows), and monitoring (usage, admin tools). This balance allows **self-service BI** while maintaining enterprise-level control.

9. 🔹 **Limitations of RLS with DirectQuery**
    1. **Performance impact**
        - Every query must include the RLS filter conditions → can slow performance significantly, especially on large fact tables.
        - Query folding depends on the data source; some sources handle filters poorly.
    2. **Limited transformations**
        - If your RLS logic relies on complex DAX or calculated tables, it may not fold into the SQL query → forcing Power BI to do extra work.
    3. **Cross-database scenarios**
        - If your model combines DirectQuery with Import (composite model), applying RLS across them can be tricky and may result in unsupported scenarios.
    4. **Data source limitations**
        - Not all DirectQuery sources support dynamic security well. For example:
            - SQL Server = ✅ works.
            - Some sources (like SAP BW, Snowflake) = may have restrictions.

🔹 **Limitations of RLS with Live Connection**

1. **Defined at source, not Power BI**
    - With Live Connection (e.g., SSAS Tabular, Azure Analysis Services, Fabric semantic models), RLS is managed **only in the source model**, not inside Power BI.
    - You cannot create or edit RLS roles in Power BI Service for these connections.
2. **Dependency on IT/central modelers**
    - Business users cannot modify security rules → they must request changes from the central BI/IT team managing the semantic model.

3. **Limited flexibility**
    - Since RLS lives in SSAS/Analysis Services, you can't combine it with Power BI's self-service features like calculated roles.

🔹 **General Limitations Across Both**

- **Not applied to dataset owners/admins** → They always bypass RLS.
- **Export/Analyze in Excel** → RLS applies, but sometimes users can infer restricted values if aggregation reveals patterns.
- **Testing** → In Power BI Desktop, you must "View as Role" to test; in Service, you test via "Test as role" → harder in Live Connection.
- **Licensing dependency** → Users consuming RLS-protected content must have **Power BI Pro or Premium per user (PPU)**.

✅ **Summary:**

- **DirectQuery:** RLS works, but can hurt performance and may not fold well to source queries.
- **Live Connection:** RLS must be managed at the source (SSAS/Analysis Services/Fabric), not in Power BI.
- **Overall:** RLS gives security but introduces trade-offs in performance, manageability, and flexibility.

10. 🔹 **1. Refresh with Power Automate**

Power Automate has a **built-in Power BI connector**.

Steps:

1. Go to Power Automate.
2. Create a new **flow** (automated, scheduled, or triggered).
3. Add the **Power BI → Refresh a dataset** action.
4. Select:
    - **Workspace** (where dataset lives)
    - **Dataset** (the one to refresh)
5. Add triggers if needed:
    - Schedule (e.g., every 2 hours).

- Event-based (e.g., when new file lands in SharePoint).

6. Save & test → The flow will call the refresh.

✅ Useful when: Business workflows (e.g., refresh dataset after ETL finishes or after user approval).

🔹 2. Refresh with Power BI REST API

For advanced or automated scenarios.

Endpoint:

POST https://api.powerbi.com/v1.0/myorg/groups/{groupId}/datasets/{datasetId}/refreshes

Steps:

1. Register an app in **Azure AD** → get **Client ID** & permissions.
   - Permission needed: Dataset.ReadWrite.All.
2. Authenticate via **OAuth2** to get an access token.
3. Call the API with headers:
4. Authorization: Bearer {access_token}
5. Content-Type: application/json
6. Example request body (optional):
7. {
8. "notifyOption": "MailOnFailure"
9. }
10. You can also check refresh status with:
11. GET https://api.powerbi.com/v1.0/myorg/groups/{groupId}/datasets/{datasetId}/refreshes

✅ Useful when: Integrating refresh into **CI/CD pipelines**, **custom apps**, or triggering refresh from external systems.

🔹 3. Key Differences

Feature	Power Automate	REST API
Ease of Use	✅ Low-code UI	❌ Requires coding
Flexibility	Medium	High (can embed in apps/scripts)
Triggers	Many (events, schedules)	Must be coded/scheduled separately
Monitoring	Limited	Full control (logs via API)

✅ Example Use Case

- **Power Automate**: Refresh sales dataset when a new Excel file is uploaded to OneDrive.
- **REST API**: Nightly ETL pipeline in Azure Data Factory calls Power BI API to refresh datasets once staging tables are loaded.