WIPRO
DATA ANALYST
INTERVIEW QUESTIONS

POWER BI

1) What is Incremental Refresh in Power BI?

Answer:

Incremental Refresh allows Power BI to **load and refresh only the new or updated data** instead of refreshing the entire dataset every time. It is highly beneficial when working with large datasets from sources like SQL Server, Azure, or data warehouses.

- **Benefits:**
  - Reduces refresh time and resource usage.
  - Improves performance and scalability.
  - Useful for historical data that doesn't change often.
- **How it works:**
  - You define a **RangeStart** and **RangeEnd** parameter in Power Query.
  - Power BI partitions data based on these parameters (e.g., last 5 years of data, refresh only last 7 days).
  - During scheduled refresh, only the data within the **refresh range** is reloaded.
- **Common Scenario:**

A sales table with millions of rows. With incremental refresh, only the sales data for the last few days is refreshed, not the entire dataset.

---

2) What are Formats and Filters in Power BI?

Answer:

✅ **Formats in Power BI:**

Formatting refers to the **visual styling and customization** of reports, visuals, and fields to improve readability and aesthetics.

- **Examples of formatting options:**
    - Data labels (on/off, font, color, size)
    - Title and background customization
    - Number and date formatting (e.g., currency, %)
    - Conditional formatting (color based on values)
    - Tooltips and interaction effects

✅ **Filters in Power BI:**

Filters allow users to **restrict data** shown in reports or visuals.

There are 4 main types of filters:

Filter Type	Scope
Visual-level	Applies to a single visual only
Page-level	Applies to all visuals on a page
Report-level	Applies across all pages
Drillthrough / Slicers	Used for navigating or custom filtering based on selection

Filters Pane in Power BI allows users to view and edit these filters easily.

---

3) What are the steps to set up gateways in Power BI?

Answer:

A **Power BI Gateway** is used to connect **on-premises data sources** to Power BI service (cloud).

- **Steps to Set Up a Gateway:**

1. **Download and Install Gateway:**
    - Go to Power BI Service → Settings → Manage gateways → Download standard gateway.
2. **Choose Mode:**
    - **Standard Mode**: Used for multiple users and services (recommended for enterprise).
    - **Personal Mode**: For a single user and Power BI only.
3. **Sign in with Organizational Account:**
    - Sign in using the same account used in Power BI Service.
4. **Configure Gateway:**
    - Name your gateway and set recovery key.
    - Register it with Power BI Service.
5. **Add Data Sources:**
    - In Power BI Service → Manage gateways → Add data source.
    - Enter connection details (server name, database name).
    - Provide credentials.
6. **Map Gateway in Dataset Settings:**
    - In Power BI Service → Dataset Settings → Under Gateway connection → Map to the configured gateway.
7. **Test and Schedule Refresh:**
    - Verify connection works.
    - Set up refresh schedule (daily, hourly, etc.).

4) What are the Different Types of Licenses in Power BI?

Power BI offers several types of licenses based on **user needs, sharing capabilities, and data capacity**. Here's a breakdown:

✅ **1. Power BI Free (Individual Use)**

- **Targeted Users:** Personal use or individual developers.
- **Features:**
  - Create reports and dashboards using Power BI Desktop.
  - Publish to **Power BI Service (workspace - My Workspace only)**.
  - Connect to 70+ data sources.
- **Limitations:**
  - Cannot **share reports or dashboards** with others.
  - No scheduled refresh (only manual refresh via Desktop).
  - Cannot publish to app workspaces or collaborate.

✅ **2. Power BI Pro (Collaboration & Sharing)**

- **Targeted Users:** Teams and organizations.
- **Cost:** Paid (Per user/month)
- **Features:**
  - Everything in Free, **plus:**
  - Share and collaborate on reports/dashboards.
  - Create and publish content to **app workspaces**.
  - Peer-to-peer sharing across the organization.
  - **Scheduled data refresh** (up to 8/day).
  - Dataflows and usage metrics.

✅ **3. Power BI Premium Per User (PPU)**

- **Targeted Users:** Advanced users who need premium features without purchasing full capacity.

- **Cost:** Higher than Pro (Per user/month).
- **Features:**
    - All features of Pro, **plus:**
    - **Paginated reports**, AI features, and larger data models.
    - **Deployment pipelines**, dataflows with linked entities.
    - 48/day scheduled refreshes.
    - Greater dataset sizes (up to 100 GB).
    - Enhanced performance and dedicated memory allocation.

✅ **4. Power BI Premium (Capacity-Based)**

- **Targeted Users:** Large organizations with many users.
- **Cost:** Per **dedicated capacity**, not per user.
- **Features:**
    - Supports **unlimited free and Pro users** for viewing content.
    - Dedicated cloud compute capacity.
    - Larger data model support (400 GB dataset size).
    - **On-premises reporting** using Power BI Report Server.
    - Enhanced performance, AI capabilities.
    - 48/day scheduled refresh.

✅ **5. Power BI Embedded (For Developers/ISVs)**

- **Targeted Users:** Developers or companies embedding Power BI reports into custom applications.
- **Cost:** Azure-based, **pay-as-you-go**.
- **Features:**
    - Embed dashboards and reports into apps/websites.

- No Power BI license required for end-users.
- API access and white-labelling.
- Designed for ISVs (independent software vendors).

🔁 **Summary Table:**

License Type	Sharing	Scheduled Refresh	Premium Features	Audience
Free	❌ No	Manual only	❌ No	Individuals
Pro	✅ Yes	Up to 8/day	❌ No	Teams & Businesses
Premium Per User	✅ Yes	Up to 48/day	✅ Yes	Power users
Premium (Capacity)	✅ Yes	Up to 48/day	✅ Yes	Enterprises
Embedded	✅ Yes	API-based	✅ Yes	Developers/ISVs

---

5) How do you connect an Excel file uploaded in OneDrive to Power BI?

Answer:

Connecting an Excel file from **OneDrive** to Power BI ensures **automatic synchronization** between your source Excel file and your Power BI report.

✅ **Steps to Connect:**

1. **Upload the Excel File to OneDrive:**
   - Save the Excel file to your **OneDrive for Business** account.
2. **Open Power BI Service (https://app.powerbi.com).**
3. **Click on "Get Data" → Choose "OneDrive – Business".**

4. **Browse and Select the Excel File.**
5. Power BI gives you **two options**:
    - **Import**: Loads a snapshot of your data into Power BI (dataset is created).
    - **Connect**: Connects to the Excel workbook **in live mode**, where reports are generated directly from the workbook (like Excel Online).
6. Choose based on your use case (Import for performance, Connect for real-time sync).
7. Your Excel data is now accessible and **auto-refreshed (usually every hour)** if it's stored in OneDrive for Business.

🔁 **Advantages:**

- Auto-refresh syncs changes from Excel to Power BI without manual updates.
- Ideal for collaborating with teams updating shared Excel sheets.

---

6) What is a Bookmark in Power BI and What Are Its Use Cases?

Answer:

A **Bookmark** in Power BI is a feature that allows you to **capture the current state of a report page**, including:

- Filters
- Slicers
- Visual selections
- Drill-throughs
- Sorting and visibility of visuals

✅ **Use Cases of Bookmarks:**

Use Case	Description
Storytelling / Presentations	Create guided, interactive data narratives. Switch between visuals/views seamlessly.
Toggle between Views	Show/hide visuals or panels using **button-based toggle** (e.g., switch between a chart and a table).
Custom Navigation	Build custom page navigation or simulate tabs using bookmarks and buttons.
Reset Filters or Slicers	Create a “Reset All Filters” button to take users back to the default view.
Dynamic Reports	Provide users with predefined views like “Top Performers”, “Low Performers”, “Monthly Trend”, etc.

✅ **How to Create a Bookmark:**

1. Go to **View tab** → Enable **Bookmarks Pane**.
2. Set your report in the desired state (filters, visibility, etc.).
3. Click **"Add"** in the Bookmarks pane → Name it.
4. Use **Buttons → Action → Bookmark** to navigate between bookmarks.

---

7) How do you create a Power BI custom visual?

Answer:

A **custom visual** in Power BI is a visual created using **JavaScript/TypeScript** and the **Power BI Developer Tools** (pbiviz), allowing you to create visuals beyond default ones.

✅ **Steps to Create a Custom Visual:**

1. **Install Power BI Visual Tools (pbiviz):**

npm install -g powerbi-visuals-tools

2. **Create a New Visual Project:**

pbiviz new MyCustomVisual

cd MyCustomVisual

3. **Develop the Visual:**
    - Use **TypeScript, D3.js**, or **React.js** to define rendering logic.
    - Edit the visual.ts file to customize how data is displayed.
4. **Test the Visual Locally:**

pbiviz start

    - Opens a Power BI service test window for live preview.
5. **Package the Visual:**

pbiviz package

    - Generates .pbiviz file.
6. **Import into Power BI:**
    - In Power BI Desktop → **Visualizations pane → Import a visual from file** → Select .pbiviz.
7. **(Optional) Publish to AppSource:**
    - Follow Microsoft's guidelines to publish your custom visual to the marketplace.

---

8) What are Calculated Columns and Measures in Power BI?

Answer:

Feature	Calculated Column	Measure
Definition	Column added using DAX that is stored in the table	Dynamic calculation used in visuals
Row Context	Works on a **row-by-row** basis	Works in **filter/context** of visual/report

Feature	Calculated Column	Measure
Storage	Stored in memory (increases model size)	Calculated on the fly (memory efficient)
Example	FullName = [FirstName] & " " & [LastName]	Total Sales = SUM(Sales[Amount])
Use Case	Add new column like category, rating, flags	Aggregate numbers, KPIs, ratios in visuals

✅ **Use both wisely** depending on your scenario. Use **measures** for performance and flexibility; use **calculated columns** only when you need the result stored at the row level.

---

9) How do you implement Row-Level Security (RLS) in Power BI?

Answer:

Row-Level Security (RLS) is used to **restrict data access** for specific users based on roles.

✅ **Steps to Implement RLS:**

1. **Create Roles in Power BI Desktop:**
    - Go to **Modeling tab → Manage Roles**.
    - Create a new role and add **DAX filters** to tables.
    - Example:

[Region] = "East"

2. **Assign Filters Based on Logged-in User (Dynamic RLS):**
    - Use USERNAME() or USERPRINCIPALNAME() DAX functions.
    - Example:

[Email] = USERPRINCIPALNAME()

3. **Test the Role:**
    - Click **View As Role** in Power BI Desktop to test RLS filters.

4. **Publish to Power BI Service:**
    - Upload the report → Go to **Dataset Settings → Security**.
    - Assign Azure AD users or security groups to roles.

---

10) You have a Power BI report running slowly. What steps would you take to diagnose and improve its performance?

Answer:

To **diagnose and improve report performance**, follow a structured approach:

✅ **1. Use Performance Analyzer (in Power BI Desktop):**

- Go to **View tab → Performance Analyzer**.
- Start recording to identify **slow visuals** and **DAX query durations**.

✅ **2. Optimize Data Model:**

- Remove **unnecessary columns and tables**.
- Use **star schema** instead of snowflake or flat structures.
- Reduce **cardinality** (e.g., trim text columns).
- Use **numeric or integer keys** for joins.

✅ **3. Optimize DAX Calculations:**

- Avoid **complex calculated columns** and move logic to the source if possible.
- Use efficient DAX functions (e.g., SUMX vs CALCULATE(SUM(...))).
- Avoid iterators where possible.

✅ **4. Optimize Queries and Data Load:**

- Apply transformations in **Power Query**, not DAX.
- Disable **Auto Date/Time** for large models.
- Use **Incremental Refresh** for large datasets.

✅ **5. Reduce Visual Complexity:**

- Limit **number of visuals** on a page.
- Avoid **heavy visuals** (like maps or matrix with large drill-downs).
- Use **aggregated data tables** instead of raw data where appropriate.

✅ **6. Use Aggregations and Pre-aggregated Tables:**

- Create summary tables to reduce real-time computations.

---

11) How will you join two tables in Power Query?

Answer:

In Power BI, you use **Merge Queries** in Power Query to **join two tables** based on matching fields.

✅ **Steps to Join Two Tables:**

1. Go to **Home → Transform Data** to enter Power Query Editor.
2. Select the first table → Click **Home → Merge Queries**.
3. Choose the second table from the dropdown.
4. Select **one or more columns** to match from both tables.
5. Choose the **Join Type**:
    - **Left Join** (default): All rows from the first table + matched rows from second.
    - **Right Join**: All from second table + matched from first.
    - **Inner Join**: Only matching rows from both.
    - **Full Outer Join**: All rows from both tables.
    - **Anti Joins**: Return non-matching rows (Left Anti / Right Anti).
6. After merging, click the expand icon to **select columns** from the second table to include.

✅ **Example:**

Joining Sales and Customers table on CustomerID using a **Left Join**.

12) Explain the Difference Between Single and Bidirectional Relationships

Feature	Single Direction	Bidirectional
Data Filter Flow	Filters flow **from one table to another** only	Filters flow **in both directions** between tables
Default Behavior	Used in most cases (Star schema)	Optional and must be enabled manually
Performance	More efficient	May reduce performance or cause ambiguity
Use Case	Fact-to-Dimension filtering	Required for certain cross-filtering visuals or slicers
Example	Region → Sales filters sales by region	Region ↔ Sales allows filter both ways (less common)

✅ **When to Use Bidirectional:**

- When slicers/filters on related tables are not working.
- In complex models like **many-to-many** relationships or **composite models**.

---

13) Explain a Common Issue Working with Many-to-Many Cardinality in Relationships

Answer:

A **many-to-many relationship** occurs when both related columns have **duplicate values**, and neither can act as a unique primary key.

⚠️ **Common Issues:**

1. **Ambiguous Query Results:**
    - Power BI may **not know how to aggregate data** correctly due to overlapping relationships.

- You may get **duplicate counts, incorrect totals, or blank values** in visuals.

2. **Circular Dependency:**
    - Many-to-many relationships can create **circular dependencies** in data models, breaking relationships.
3. **Performance Problems:**
    - These relationships often require **bidirectional filtering**, which is more compute-intensive.
4. **Filtering Confusion:**
    - Filtering from one table might **not behave as expected**, leading to unexpected results in visuals.

✅ **Solutions:**

- Introduce a **bridge (relationship) table** with unique keys.
- Use **DAX filters** carefully with functions like TREATAS().
- Avoid bi-directional filtering unless absolutely necessary.
- Maintain a **Star Schema** as much as possible.

✅ Scenario-Based Power BI Interview Question:

---

1) Scenario:

"We want to design a Power BI report on complaints and compliments received through our customer service department. However, this information is located within a folder of 100 files. How would you go about importing these files in Power BI?"

✅ **Step-by-Step Solution:**

Power BI allows you to **import multiple files from a folder** (e.g., Excel, CSV, TXT) using the **"Folder" connector** in Power Query.

🔹 **Step 1: Go to Power Query Editor**

- Open **Power BI Desktop**
- Click on **Home → Get Data → More → Folder**

🔹 **Step 2: Select the Folder Path**

- Choose the **folder** where all 100 files are located.
- Click **OK**.

🔹 **Step 3: View Folder Content**

- Power BI shows a preview table with **file names, extensions, and metadata**.

Click **"Combine"** → **"Combine & Transform Data"**.

🔹 **Step 4: Define Sample File Logic**

Power BI uses one of the files as a **sample** to define the transformation steps.

- You'll be taken into **Power Query Editor**.
- Apply necessary transformations:
  - Filter rows (e.g., remove headers/footers).
  - Promote headers if needed.
  - Rename columns.
  - Parse date/time, categorize fields, etc.

These steps will be **applied automatically to all 100 files** in the folder.

🔹 **Step 5: Load Data into Power BI**

- Click **Close & Load** to bring the cleaned, combined data into Power BI.

🔹 **Step 6: Build Visuals**

- Create dashboards showing:
  - Complaints vs Compliments over time.
  - Category-wise issues.
  - Resolution timelines.
  - Region or department-level breakdowns.

✅ **Bonus Tips:**

- **Ensure all files follow the same schema** (column names and structure).
- Use **file name as a column** if each file represents a different region or time period (helps in tracking source).
- Set up **scheduled refresh** if the folder is on **OneDrive, SharePoint, or a network path**.

SQL

✅ 1) Query to Calculate Cumulative Salary Department-Wise for Employees Who Joined in the Last 30 Days

🔹 **Assumptions:**

- Table: employees
- Columns: emp_id, emp_name, salary, department, joining_date

🔹 **SQL Query (for MySQL/PostgreSQL/SQL Server):**

```
SELECT
  department,
  emp_name,
  joining_date,
  salary,
  SUM(salary) OVER (PARTITION BY department ORDER BY joining_date) AS cumulative_salary
FROM
  employees
WHERE
  joining_date >= CURRENT_DATE - INTERVAL 30 DAY
ORDER BY
  department, joining_date;
```

🔹 **Explanation:**

- Filters employees who joined in the **last 30 days**.
- Uses SUM(...) OVER (PARTITION BY ... ORDER BY ...) to compute **cumulative salary** per department, ordered by joining date.

✅ Works on **PostgreSQL**, **SQL Server**, and **Oracle**. For MySQL 8+, ensure window functions are supported.

✅ 2) Query to Find the Second Most Recent Order Date

🔹 **Table: Orders(order_id, customer_id, orderdate)**

🔹 **Option 1: Using DISTINCT and LIMIT / OFFSET (PostgreSQL / MySQL 8+):**

```
SELECT
  orderdate
```

```
FROM (
  SELECT DISTINCT orderdate
  FROM Orders
  ORDER BY orderdate DESC
  LIMIT 2
) AS top_two
ORDER BY orderdate
LIMIT 1;
```

- This fetches the **2 most recent dates**, then takes the **earlier of the two**, i.e., second most recent.

🔹 **Option 2: Using Subquery (Standard SQL – works in SQL Server, etc.):**

```
SELECT MAX(orderdate) AS second_most_recent
FROM Orders
WHERE orderdate < (
  SELECT MAX(orderdate) FROM Orders
);
```

- Finds the **maximum order date less than the overall maximum**, i.e., **second most recent**.