Tiger Analytics

Power BI Developer Interview Questions

0-3 Yoe

1. How do you calculate YOY Growth in Power BI using DAX?

Assumption:You have a Salestablewith:

- Date column (linked to a Date table),
- SalesAmount column.

Step 1: Create Total Sales Measure

```
Total Sales = SUM(Sales[SalesAmount])
```

Step 2: Create Sales Last Year Measure

```
Sales Last Year = CALCULATE([Total Sales], SAMEPERIODLASTYEAR('Date'[Date]))
```

Step 3: YOY Growth in Absolute Value

```
YOY Growth = [Total Sales] - [Sales Last Year]
```

Step 4: YOY Growth in Percentage

```
YOY Growth % =
DIVIDE([YOY Growth], [Sales Last Year], 0)
```

---

2. What DAX expression would you use to calculate the sales for last 45 days in Power BI?

Measure tocalculatesalesofthelast45days:

```
Sales Last 45 Days =
CALCULATE(
```

```
  [Total Sales],
  DATESINPERIOD('Date'[Date], MAX('Date'[Date]), -45, DAY)
)
```

Explanation:

- MAX('Date'[Date]): Gets the latest visible date in the current context.
- DATESINPERIOD: Creates a date range going back 45 days.
- CALCULATE: Filters the total sales measure to only the last 45 days.

---

3. Removing Filters with DAX: What is the syntax in DAX to remove all filters from a table or visuals?

Use the ALL() function to remove filters from a table, column, or entire visual.

Syntax:

```
CALCULATE(
  [Measure],
  ALL('TableName')
)
```

Examples:

Remove all filters from the entire table:

```
Total Sales (No Filters) =
CALCULATE([Total Sales], ALL('Sales'))
```

Remove filters from a specific column only:

```
Total Sales (Ignore Region Filter) =
CALCULATE([Total Sales], ALL('Sales'[Region]))
```

Remove all filters except a few (using ALLSELECTED, ALLEXCEPT):

Total Sales by Product =

CALCULATE([Total Sales], ALLEXCEPT('Sales', 'Sales'[Product]))

---

4. VALUES vs DISTINCT in DAX — What's the difference and when to use each?

Feature	VALUES()	DISTINCT()
Returns	Unique values plus a blank if applicable	Only unique values, no automatic blank
Can return table?	Yes	Yes
Use in relationships?	Yes (used in RELATEDTABLE, etc.)	Not ideal
Blank row included?	Yes (if any row has blank in column)	No

Example:

VALUES(Customer[Region]) -- Returns unique regions + BLANK if blank exists

DISTINCT(Customer[Region]) -- Returns unique regions only, excludes blank

---

Use VALUES when:

- You need to respect the relationship and context.
- You're dealing with filters in CALCULATE, RELATEDTABLE, etc.

Use DISTINCT when:

- You want a simple unique list with no regard for context or relationships.

---

5. How do you use DAX Studio for performance tuning and query optimization?

DAX Studio is a Power BI external tool used for analyzing and optimizing DAX queries.

Steps to Use DAX Studio:

1. Connect to Power BI Model
    - Open Power BI → External Tools → Launch DAX Studio.
2. Run a Measure Query

```
EVALUATE
SUMMARIZECOLUMNS(
'Product'[Category],
"Total Sales", [Total Sales]
)
```

3. Use Performance Analyzer (in DAX Studio):
    - Click on Server Timings, Query Plan, Statistics.

Key Features for Performance Tuning:

Feature	Purpose
Server Timings	See duration of Storage Engine (SE) and Formula Engine (FE) operations
Query Plan	Understand the sequence of DAX operations
Query Breakdown	Identify bottlenecks in measures or filters
DMV Explorer	Explore metadata, relationships, cardinality, etc.

Common Optimization Tips via DAX Studio:

- Reduce DISTINCTCOUNT() or FILTER() on high-cardinality columns.
- Use SUMX() carefully – check row context impact. Avoid
- CALCULATE() nesting unnecessarily. Reduce cross-table joins if FE
- time is high.

---

6. Import Mode, Direct Query Mode, and Live Connection in Power BI: When would you use each?

Feature	Import Mode	Direct Query Mode	Live Connection
Data Storage	Data is imported into Power BI	Data is queried directly from the source	Data stays in the external model (e.g., SSAS)
Performance	Fastest (in-memory model)	Slower (every interaction sends a query)	Depends on source performance
Data Refresh	On schedule (e.g., 8/day in Pro, 48 in Premium)	Real-time / near real-time	Real-time via live connection
Data Transformation Query	Full support in Power	Limited support	Not supported in Power BI (transforms done in SSAS)
When to Use	Most use-cases where data volume is manageable	Large datasets or when near real-time is needed	When using enterprise semantic models (SSAS, AAS)
Relationships & DAX	Fully supported	Limited (can't use all DAX features)	Fully supported via SSAS models

---

Use Cases:

- Import Mode:
    - o Best for performance and flexibility.
    - o Use when data size is reasonable, and hourly/daily refresh is enough.
- Direct Query:
    - o Use when data changes very frequently or too large to load.
    - o Ideal for real-time dashboards.
- Live Connection:
    - o Use when data models are built in SSAS Tabular / Azure Analysis Services.
    - o Ideal for centralized, enterprise-grade modeling.

---

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

RLS restricts data access at the row level for users viewing reports.

Steps to Implement RLS in Power BI Desktop:

1. Define Roles in Power BI Desktop:
    - o Go to Modeling → Manage Roles.
    - o Create a role with a filter on a table.
    - o Example for filtering by Region:

```
[Region] = "West"
```

2. Dynamic RLS using USERNAME():
    - o Create a table mapping users to their permitted values:

```
UserTable:
Email          | Region
----------------|--------
john@abc.com     | West
amy@abc.com      | East
```

- Create a relationship between UserTable[Region] and your main data table's Region.
- Add role filter:

[Email] = USERNAME()

3. Test the Role in Power BI Desktop:
   - Modeling → View As Roles → Choose a role or type a username to simulate.
4. Publish to Power BI Service:
   - After publishing, go to Workspace → Dataset → Security.
   - Assign users to the defined roles.

Note:

- In Import and Direct Query, RLS works as expected.
- In Live Connections, RLS should be defined in the SSAS model.

---

8. What strategies would you use to optimize the performance of large Power BI reports?

Optimizing Power BI reports is crucial for scalability, performance, and user experience. Below are proven strategies:

---

1. Optimize the Data Model

- Use Star Schema instead of snowflake. Avoid unnecessary columns and tables. Use appropriate data types (e.g., avoid text when integers suffice).
- Remove calculated columns where possible; use measures instead.

---

2. Reduce Cardinality

- Reduce distinct values in columns (especially in slicers and relationships).
- Truncate long strings; categorize them if possible.
- Group dates (use Year, Month instead of full timestamp).

---

3. Aggregations and Pre-Processing

- Pre-aggregate data in Power Query or SQL (especially for large fact tables).
- Use summarized tables for visuals that don't require row-level granularity.

---

4. Use Efficient DAX

- Avoid heavy operations like CALCULATE over high-cardinality fields.
- Prefer SUMX over CALCULATE(SUM()) when row context is needed.
- Replace IF conditions with SWITCH when checking multiple scenarios.

---

5. Optimize Visuals

- Limit the number of visuals on a page (ideally under 8).
- Avoid using high-cardinality slicers and excessive bookmarks.
- Use matrix/table visuals cautiously with performance-tuned DAX.

---

6. Manage Relationships Smartly

- Avoid bi-directional relationships unless necessary.
- Turn off auto-date/time intelligence when not required.

---

7. Enable Incremental Refresh

- Especially important for large datasets (Direct Query or Import).
- Only refresh new/changed data — saves time and resources.

8. Use Performance Analyzer & DAX Studio

- Identify slow visuals and long-running DAX queries.
- Optimize based on Formula Engine (FE) and Storage Engine (SE) time.

9. Control Data Load

- Disable “Auto Date/Time” for date fields.
- Load only necessary columns & rows (filter at source or in Power Query).

10. Use Composite Models or Aggregation Tables

- Combine Import (summary data) and Direct Query (detailed drill-down).
- Improves both interactivity and freshness.

9. For a project with 200 users in the USA and Europe, where 50 reports need to be created, what Power BI licensing model would you recommend?

This is a scalability and cost optimization question.

Recommended: Power BI Premium Per Capacity (P SKUs)

Why Premium Capacity is best:

Feature	Benefit
200 Users	Cost-effective at scale (compared to Pro per-user licenses)
50 Reports	Handles multiple datasets, models, and large refresh frequencies

Feature	Benefit
Enterprise features	Enables incremental refresh, AI visuals, paginated reports
Performance	Dedicated capacity → faster performance
RLS & Large Models	Full support for Row Level Security + up to 400 GB models
Unlimited Report Viewers	Viewers don't need Pro license if workspace is in Premium

---

Cost Comparison:

Option	Approx. Cost
200 Power BI Pro Licenses ($10/user/month)	$2,000/month
Power BI Premium (P1, 32 GB RAM)	~$4,995/month
Premium Per User (PPU, $20/user)	$4,000/month (not scalable)

If all 200 users need interactivity and report creation, PPU is costly. If most are just viewers, Premium Capacity (P1 or P2) is the best fit.

Final Recommendation:

For 200 users and 50 enterprise reports, Power BI Premium (Per Capacity) is the best choice due to performance, scalability, cost efficiency, and advanced features support.

---

10. Compare Key Features and Advantages of Power BI vs Tableau

Feature / Area	Power BI	Tableau
Ownership	Microsoft	Salesforce
Learning Curve	Easier for Excel/Office users (integrated with Microsoft ecosystem)	Requires more time for beginners, especially in calculated fields
Data Modeling	Strong modeling with DAX, supports star schema, relationships	Weaker data modeling; typically needs data prepared before Tableau
Visualization	Great visuals with growing customization, lots of default options Excellent with large models (especially in Import Mode)	Best-in-class interactive visuals, highly polished charts and formatting
Performance	Power Query is robust for ETL operations	Very fast with in-memory engine and optimized extracts
Data Preparation	Integrates with Python, R, and AI visuals	Tableau Prep available but not as powerful as Power Query
Advanced Analytics	Power BI Desktop, Power BI Service, Power BI Report Server	Integrates with Python, R, and supports advanced statistical charts
Deployment Options	More cost-effective (Pro: $10/user, Premium: $20/user or per capacity)	Tableau Desktop, Tableau Server, Tableau Cloud
Integration	Seamless with Excel, Azure, SQL Server, Power Apps, SharePoint	Great with Salesforce and a wide range of databases
Pricing	Strong mobile support, easy sharing via Power BI Service	More expensive (Viewer, Explorer, Creator license model)
Mobile & Sharing		Strong dashboards for mobile but harder publishing/sharing options
Community Support	Huge Microsoft user base, active forums, lots of templates	Strong community, lots of templates and visual customization guides

Summary:

- Choose Power BI if:
  - o You're in a Microsoft ecosystem.
  - o Need strong data modeling, low cost, and integration with Excel/SharePoint.
- Choose Tableau if:
  - o You need high-end visuals, quick dashboarding, and advanced interactivity.

11. How would you estimate the time or effort required to build 4–5 reports using Excel as the source?

Estimation Strategy: Use the SDLC Breakdown + complexity + reusability

Effort Estimation Breakdown (Per Report):

Task	Estimated Time (Hours)
Data Preparation & Requirements	2–4 Data Cleaning &
Data Modeling in Power BI	2–3
DAX Measures Creation	2–4
Visualization & Layout Design	3–5
Testing & Validation	2–3 1–2 15–
Feedback & Rework	25 hours
Total per report	

Time for 4–5 Reports:

- If reports have similar logic or reusable visuals:
    - ~50–80 hours total
- If all reports are distinct and complex:
    - ~75–120 hours total

Factors Affecting Time:

- Quality and structure of Excel files (flat vs. normalized).
- Number of visuals/measures per report.
- Data volume (affects load time/modeling complexity).
- Stakeholder feedback cycles.
- Use of themes/templates to reduce visual setup time.

---

12. How do you calculate the percentage of sales contributed by each product category in a single DAX measure?

Objective:

You want a DAX measure that gives the % contribution of each Product Category to total sales.

DAX Measure:

```
% Sales by Category =
DIVIDE(
    [Total Sales],
    CALCULATE([Total Sales], ALL('Product'[Category])),
    0
)
```

Explanation:

- [Total Sales]: Basic measure like SUM(Sales[SalesAmount])
- ALL('Product'[Category]): Removes the filter on Product Category so we get total overall sales
- DIVIDE(): Safe division that avoids errors if denominator is zero

---

Usage:

- Place Product Category on a table/matrix row.
- Add % Sales by Category as a value.
- The visual will show each category's sales % against overall total.

---

13. How would you set up a slicer in Power BI to always show the current month while allowing users to select previous months and years?

Approach: Create a "Current Month" flag in your Date table, then use it with a slicer + optional filter pane.

Step 1: Add calculated column in your Date table

```
IsCurrentMonth =
IF(
    YEAR('Date'[Date]) = YEAR(TODAY()) &&
    MONTH('Date'[Date]) = MONTH(TODAY()),
    "Current Month",
    "Other"
```

)

Step 2: Add this column as a slicer

- Place IsCurrentMonth as a slicer with default selection = "Current Month"
- Users can manually uncheck it to see "Other"

Alternative Approach (Pre-filter slicer to default to current month)

Use a relative date slicer:

- Add the Date column to slicer.
- Change slicer type to Relative.
- Set to:
    - "In the last 1 month"
    - Anchor to today
- This way, it will always default to current month, but users can still navigate to older periods manually if the date hierarchy is also available elsewhere.

Optional Enhancement:

Add a dropdown slicer for year/month separately so users can switch year/month dynamically while the report defaults to current month.

---

14. What are the three most useful visuals you've used in Power BI, and why? Provide use cases for each.

---

1. Matrix Visual

Why it's useful:

- Combines rows and columns like a pivot table.

- Supports drill-down, row grouping, and conditional formatting.

Use Case:

Used in a Sales Dashboard to show Sales Amount by Region and Product Category, with dynamic subtotals and drill-down to SKU level.

---

2. Line and Clustered Column Chart

Why it's useful:

- Combines two types of trends (bar + line) in a single visual.
- Helps in showing comparison + trend.

Use Case:

In a Customer Retention Report, used to show New Customer Count (column) and Repeat Purchase % (line) over months.

---

3. Card / KPI Visual

Why it's useful:

- Displays key metrics in bold, easy-to-read format.
- Useful for showing summary KPIs.

Use Case:

In a Finance Executive Summary, used to show:

- Total Revenue (Card) YoY Growth % (Card)
- Target vs Actual with KPI Indicator (KPI Visual)
-

---

15. Time Intelligence Functions: Explain how you would use TOTALQTD, TOTALMTD, TOTALYTD, SAMEPERIODLASTYEAR, DATEADD, and PARALLELPERIOD in Power BI.

These DAX functions help in building time-based comparisons (MTD, QTD, YTD, YoY, MoM, etc.)

---

1. TOTALMTD (Month-to-Date Total)

MTD Sales =

TOTALMTD([Total Sales], 'Date'[Date])

Use Case:

Tracks how much sales have occurred from the 1st to today of the current month.

---

2. TOTALQTD (Quarter-to-Date Total)

QTD Sales =

TOTALQTD([Total Sales], 'Date'[Date])

Use Case:

Monitor cumulative sales for the current quarter — helps in quarterly performance tracking.

---

3. TOTALYTD (Year-to-Date Total)

YTD Sales =

TOTALYTD([Total Sales], 'Date'[Date])

Use Case:

Used in annual dashboards to show cumulative sales till date in the year.

---

4. SAMEPERIODLASTYEAR

Sales LY =

CALCULATE([Total Sales], SAMEPERIODLASTYEAR('Date'[Date]))

Use Case:

Used for Year-over-Year (YoY) comparison.

---

5. DATEADD

Sales 1 Month Ago =

CALCULATE([Total Sales], DATEADD('Date'[Date], -1, MONTH))

Use Case:

Use for Month-over-Month (MoM) growth or custom shifting like “Sales 7 days ago”.

---

6. PARALLELPERIOD

Sales Last Quarter =

CALCULATE([Total Sales], PARALLELPERIOD('Date'[Date], -1, QUARTER))

Use Case:

For comparing performance across parallel periods, e.g., last quarter vs current quarter.

DATEADD vs PARALLELPERIOD:

Feature	DATEADD	PARALLELPERIOD
Supports day?	Yes	No (requires month, quarter, or year)
Flexible shifts	(+/- days, months, years)	But only at fixed granularity
Safer in visuals	Better for dynamic visuals	Can behave unexpectedly in some cases

---

16. What are the different types of gateways in Power BI, and how do they work?

In Power BI, Gateways are used to enable secure data transfer between on-premises data sources and Power BI Service (cloud).

TYPES OF GATEWAYS IN POWER BI

Type	Description
1. Personal Gateway	Meant for individual use, allows scheduled refresh, no live connection
2. On-premises Data Gateway (Standard/Enterprise)	Used for multiple users, supports live connections and scheduled refresh
3. Virtual Network Data Gateway (VNet Gateway)	Allows cloud-based sources in a VNet to be accessed securely without installing a physical gateway

---

1. Personal Gateway

• Use Case: When a single user builds a report using Excel, Access, or SQL Server.

- Runs under user's credentials.
- Cannot be shared or used for live connections.
- Only supports scheduled refresh.

---

2. On-premises Data Gateway (Standard)

- Enterprise-ready solution for scheduled refresh and live/direct query.

• Can be shared across multiple users and services (Power BI, Power Apps, Power Automate, etc.).

• Runs as a Windows service.

- Supports:
    - SQL Server
    - SAP

- Oracle
- File systems (Excel, CSV)
- Others

Two Modes of Operation:

- Import Mode → Scheduled refresh.
- Direct Query / Live Connection → Real-time data fetching.

---

3. Virtual Network Data Gateway (VNet Gateway)

- No need to install a physical gateway.
- Deployed and configured within Azure Virtual Networks.
- Ideal for organizations with cloud-only infrastructure but secure networking needs.
- Use Case: Connecting Power BI to Azure SQL Managed Instance securely.

How Gateways Work (Simplified):

1. User publishes a report using on-prem data.
2. Gateway connects securely to that data source from Power BI Service.
3. For:
   - Scheduled Refresh: Gateway pulls data at intervals.
   - Direct Query / Live Connection: Gateway acts like a proxy, sending live queries to on-prem server.

Security:

- All data is encrypted during transit.
- Uses Azure Service Bus to initiate and manage communication.
- When to Use Each Gateway:

Scenario	Gateway Type
Single-user Excel report	Personal Gateway
Shared dashboard with SQL Server data	On-premises Data Gateway
Cloud-only setup in secure VNet	Virtual Network Data Gateway
Live Connection to SSAS	On-premises Data Gateway