Power BI MCP Server

Complete Tools Reference Guide

MCP-PowerBi-Finvision v6.0.5

Generated on November 21, 2025

Welcome to the Power BI MCP Server Tools Guide!

This guide explains all 45 tools available in the MCP-PowerBi-Finvision server. These tools allow you to analyze, modify, and document Power BI models through an AI-powered interface.

What is an MCP Server?
MCP (Model Context Protocol) is a way for AI assistants to access specialized tools. This server provides tools specifically designed for working with Power BI Desktop models.

How to Read This Guide:
• Each tool has a clear description of what it does
• "When to Use" explains the situations where you'd use this tool
• Examples show real-world usage scenarios
• Parameters list what information you need to provide
• Results explain what you get back

Getting Started:
1. Always start by connecting to a Power BI instance (Tool 01)
2. Use the analysis tools (05) to understand the model
3. Then use specific tools based on what you need to do

The tools are organized into 13 categories for easy navigation. Let's dive in!

Table of Contents

Quick Reference Guide

Most Commonly Used Tools:

■ Connection (Always First!):
• 01_detect_pbi_instances → Find open Power BI files
• 01_connect_to_instance → Connect to one

■ Quick Model Overview:
• 05_live_model_simple_analysis → Fast 2-5 second overview of entire model
• 02_list_tables → See all tables
• 02_list_measures → See all calculations

■ Deep Analysis:
• 05_live_model_full_analysis → Complete health check with 120+ rules (10-180 sec)
• 03_standard_dax_analysis → Analyze and optimize DAX measures

■ Making Changes:
• 04_upsert_measure → Create or update a calculation
• 04_bulk_create_measures → Create many at once
• 11_tmdl_bulk_rename → Rename objects safely

■ Documentation:
• 08_generate_model_documentation_word → Professional Word doc
• 07_export_tmdl → Export model as code

■ For AI Analysis (Most Powerful!):
• 13_full_model_pbip_and_sample_export → Export complete package
• 13_full_model_pbip_and_sample_analysis → AI BI Expert analysis with recommendations

01 - Connection

■ Detect Power BI Instances (01_detect_pbi_instances)

What it does: Automatically finds all running Power BI Desktop instances on your computer.

When to use this tool:

Use this when you want to see which Power BI files are currently open before connecting to one.

■ Example:

Before starting analysis, run this to see: 'Found 2 instances: Sales Report (modified 10 mins ago), Customer Analysis (modified 2 hours ago)'

■ Parameters (what you provide):

None required - just run it!

■ What you get back:

List of all open Power BI Desktop files with their names and when they were last modified.

■ Connect to Power BI Instance (01_connect_to_instance)

What it does: Connects to a specific Power BI Desktop file that's currently open on your computer.

When to use this tool:

Always use this first before using any other tools. It's like opening the file to work with it.

■ Example:

After detecting instances, connect to the first one (index 0) for the Sales Report, or index 1 for Customer Analysis.

■ Parameters (what you provide):

model_index (optional, default=0): Which Power BI file to connect to (0 for first, 1 for second, etc.)

■ What you get back:

Success message confirming connection to the Power BI model.

02 - Schema & Structure

■ List All Tables (02_list_tables)

What it does: Shows all the data tables in your Power BI model, like a table of contents.

When to use this tool:

Use when you want to see what data tables exist in the model (e.g., Sales, Customers, Products).

■ Example:

Returns: 'Found 12 tables: DimCustomer, FactSales, DimProduct, DimDate, etc.' with column counts for each.

■ Parameters (what you provide):

None required

■ What you get back:

List of all tables with names, column counts, measure counts, and hidden/visible status.

■ Describe Table in Detail (02_describe_table)

What it does: Provides complete information about a specific table: all columns, measures, relationships, and metadata.

When to use this tool:

Use when you need to understand everything about a particular table.

■ Example:

Describe 'DimCustomer' → Shows: 15 columns (CustomerID, CustomerName, Country...), 3 measures, 2 relationships to other tables.

■ Parameters (what you provide):

table (required): Name of the table to describe

■ What you get back:

Comprehensive table details including columns, data types, measures, and relationships.

■ List Columns (02_list_columns)

What it does: Lists all columns across tables or just in a specific table.

When to use this tool:

Use to find which columns exist, their data types, or to search for a specific column across tables.

■ Example:

> *List all columns in 'FactSales' → Shows: OrderDate (DateTime), SalesAmount (Decimal), Quantity (Integer), etc.*

■ Parameters (what you provide):

table (optional): Filter by specific table name

■ What you get back:

List of columns with table names, data types, calculated vs. regular, hidden status.

■ List Measures (02_list_measures)

What it does: Shows all DAX measures (calculations) in the model or in a specific table.

When to use this tool:

Use to see what calculations exist (like Total Sales, Profit Margin, Year-over-Year Growth).

■ Example:

> *List measures in 'Sales' table → Shows: Total Revenue, Total Cost, Profit Margin, YTD Sales, etc.*

■ Parameters (what you provide):

table (optional): Filter by specific table name

■ What you get back:

List of measures with names, tables, display folders, and descriptions.

■ Get Measure Details (02_get_measure_details)

What it does: Gets complete details about a specific measure, including its DAX formula.

When to use this tool:

Use when you need to see how a calculation is built (the DAX code behind it).

■ Example:

> *Get details for 'Total Revenue' → Shows: DAX formula 'SUM(FactSales[SalesAmount])', format: Currency, display folder: 'Sales\Core Metrics'*

■ Parameters (what you provide):

table (required): Table name, measure_name (required): Measure name

■ What you get back:

Complete measure details: DAX expression, format, description, display folder.

■ List Calculated Columns (02_list_calculated_columns)

What it does: Shows all calculated columns (columns created with DAX formulas instead of loaded from data source).

When to use this tool:

Use to identify which columns are calculated vs. directly from your data source.

■ Example:

> *Returns: 'Found 8 calculated columns: FullName (FirstName & LastName), AgeGroup (IF Age logic), YearMonth, etc.'*

■ Parameters (what you provide):

table (optional): Filter by specific table

■ What you get back:

List of calculated columns with their DAX expressions and tables.

■ Search Objects (02_search_objects)

What it does: Searches across all tables, columns, and measures to find objects matching your search term.

When to use this tool:

Use when you're looking for something but don't know which table it's in (e.g., 'find anything with Revenue').

■ Example:

Search for 'Revenue' → Finds: 'Total Revenue' measure, 'Revenue' column in FactSales, 'RevenueGoal' column in Budget table.

■ Parameters (what you provide):

search_term (required): What to search for

■ What you get back:

All matching tables, columns, and measures with their locations.

■ Search in Measure Expressions (02_search_string)

What it does: Searches inside DAX formulas to find measures that use specific functions or references.

When to use this tool:

Use to find all measures that use a specific calculation method or reference another measure/column.

■ Example:

Search for 'CALCULATE' → Finds all measures using the CALCULATE function in their DAX code.

■ Parameters (what you provide):

search_term (required): Text to search for in DAX expressions, search_in (optional): 'name', 'expression', or 'both'

■ What you get back:

List of measures where the search term appears, with the relevant DAX code.

03 - DAX Intelligence

■ Comprehensive DAX Intelligence (03_standard_dax_analysis)

What it does: THE MAIN DAX TOOL - Analyzes DAX measures with validation, debugging, optimization, and recommendations. Includes smart measure finder with fuzzy matching!

When to use this tool:

Use whenever you want to understand, debug, or optimize a DAX measure. This is your go-to DAX analysis tool.

■ Example:

Analyze 'Total Revenue' → Gets: syntax validation, context transitions, performance metrics, optimization suggestions, SQLBI article recommendations, and rewritten optimized code.

■ Parameters (what you provide):

expression (required): Either DAX code OR just measure name (auto-fetches!), analysis_mode (optional): 'all' (default - runs everything), 'analyze', 'debug', or 'report'

■ What you get back:

Complete DAX analysis with 11 anti-pattern checks, context flow, VertiPaq metrics, optimization tips, and actual rewritten DAX code suggestions.

■ Execute DAX Query (03_run_dax)

What it does: Runs a DAX query against the model and returns the results, just like the DAX query window in Power BI.

When to use this tool:

Use to test calculations, preview data, or run analysis queries.

■ Example:

Run: EVALUATE SUMMARIZE(FactSales, DimDate[Year], "Total", SUM(FactSales[SalesAmount])) → Returns: Year-by-year sales totals.

■ Parameters (what you provide):

query (required): DAX query (EVALUATE statement), top_n (optional): Limit results

■ What you get back:

Query results as a table with columns and rows.

■ Get Column Value Distribution (03_get_column_value_distribution)

What it does: Shows the top N values in a column with their frequencies - like a mini histogram.

When to use this tool:

Use to understand what values exist in a column and how often they appear.

■ Example:

Get distribution of 'Country' column → Shows: USA (45%), UK (30%), Germany (15%), France (10%).

■ Parameters (what you provide):

table (required): Table name, column (required): Column name, top_n (optional): How many top values to show

■ What you get back:

List of most common values with counts and percentages.

■ Get Column Statistics (03_get_column_summary)

What it does: Provides statistical summary for a column (min, max, average, distinct count, null count).

When to use this tool:

Use to understand the data range and quality in numeric or date columns.

■ Example:

Get summary of 'SalesAmount' → Shows: Min: $10, Max: $50,000, Average: $1,250, Total distinct: 15,847 values.

■ Parameters (what you provide):

table (required): Table name, column (required): Column name

■ What you get back:

Statistical summary with min, max, average, count, distinct values, null count.

■ List Relationships (03_list_relationships)

What it does: Shows all relationships between tables (how tables are connected in the data model).

When to use this tool:

Use to understand the data model structure and how tables relate to each other.

■ Example:

Returns: 'FactSales[CustomerID] → DimCustomer[CustomerID] (Many-to-One, Active)', with cardinality and cross-filter direction.

■ Parameters (what you provide):

active_only (optional): True to show only active relationships

■ What you get back:

List of relationships with from/to tables, columns, cardinality (One-to-Many, Many-to-Many), and active status.

■ Get Data Sources (03_get_data_sources)

What it does: Lists all data sources connected to the model (SQL databases, Excel files, web sources, etc.).

When to use this tool:

Use to understand where the data is coming from.

■ Example:

Returns: 'SQL Server: MyServer\DB (Tables: Sales, Customers), Excel: C:\Data\Budget.xlsx'

■ Parameters (what you provide):

None required

■ What you get back:

List of data sources with connection details.

■ Get Power Query M Expressions (03_get_m_expressions)

What it does: Shows the M (Power Query) code that loads and transforms data for each table.

When to use this tool:

Use to see how data is being loaded and transformed in Power Query.

■ Example:

Get M for 'FactSales' → Shows: Source = Sql.Database("server", "db"), TransformedData = Table.TransformTypes(...).

■ Parameters (what you provide):

table (optional): Filter by specific table

■ What you get back:

Power Query M code for data loading and transformation.

04 - Model Operations

■ Create or Update Measure (04_upsert_measure)

What it does: Creates a new measure or updates an existing one with new DAX code.

When to use this tool:

Use to add new calculations or fix existing ones in the model.

■ Example:

Create measure: 'Profit Margin' = DIVIDE([Total Profit], [Total Revenue]) in the 'Sales' table.

■ Parameters (what you provide):

table (required): Table name, measure_name (required): Measure name, expression (required): DAX formula, description (optional), format (optional), display_folder (optional)

■ What you get back:

Success confirmation with measure details.

■ Delete Measure (04_delete_measure)

What it does: Removes a measure from the model.

When to use this tool:

Use to clean up unused or incorrect measures.

■ Example:

Delete measure 'Old Revenue Calculation' from 'Sales' table.

■ Parameters (what you provide):

table (required): Table name, measure_name (required): Measure name

■ What you get back:

Success confirmation.

■ Bulk Create Measures (04_bulk_create_measures)

What it does: Creates multiple measures at once - much faster than creating them one by one.

When to use this tool:

Use when you need to create many measures at once (e.g., time intelligence measures for multiple base metrics).

■ Example:

Create 10 measures at once: Total Sales, Total Cost, Profit, Profit %, YTD Sales, YTD Cost, etc.

■ Parameters (what you provide):

measures (required): Array of measure definitions with table, name, expression, description, format

■ What you get back:

Summary of created measures with success/failure for each.

■ Bulk Delete Measures (04_bulk_delete_measures)

What it does: Deletes multiple measures at once.

When to use this tool:

Use to clean up many unused measures quickly.

■ Example:

Delete all measures with 'Test' in their name, or all measures from a specific folder.

■ Parameters (what you provide):

measures (required): Array of measure references with table and name

■ What you get back:

Summary of deleted measures with success/failure for each.

■ List Calculation Groups (04_list_calculation_groups)

What it does: Shows all calculation groups (advanced DAX feature for reusable time intelligence and scenario analysis).

When to use this tool:

Use to see what calculation groups exist (like Time Intelligence, Currency Conversion, Scenario Analysis).

■ Example:

Returns: 'Time Intelligence' group with items: Current, YTD, MTD, QTD, PY, YoY%, etc.

■ Parameters (what you provide):

None required

■ What you get back:

List of calculation groups with their calculation items.

■ Create Calculation Group (04_create_calculation_group)

What it does: Creates a new calculation group with multiple calculation items.

When to use this tool:

Use to create reusable time intelligence or scenario analysis patterns (advanced users).

■ Example:

Create 'Time Comparison' group with items: Current = SELECTEDMEASURE(), YTD = TOTALYTD(SELECTEDMEASURE(), ...), etc.

■ Parameters (what you provide):

name (required): Group name, items (required): Array of calculation items with name and expression, precedence (optional)

■ What you get back:

Success confirmation with calculation group details.

■ Delete Calculation Group (04_delete_calculation_group)

What it does: Removes a calculation group from the model.

When to use this tool:

Use to remove unused or incorrect calculation groups.

■ Example:

> *Delete 'Old Time Intelligence' calculation group.*

■ Parameters (what you provide):

name (required): Calculation group name

■ What you get back:

Success confirmation.

■ List Security Roles (04_list_roles)

What it does: Shows all Row-Level Security (RLS) roles that control who can see what data.

When to use this tool:

Use to see what security roles exist (like 'US Region', 'Europe Region', 'Manager', 'Sales Rep').

■ Example:

> *Returns: 'Regional Manager' role with filters on Country table, 'Sales Rep' role with filters on Salesperson table.*

■ Parameters (what you provide):

None required

■ What you get back:

List of security roles with their table filters and permissions.

05 - Model Analysis

■ Fast Model Analysis (Simple) (05_live_model_simple_analysis)

What it does: Quick analysis of the model - runs 8 Microsoft MCP operations to get a complete overview in seconds.

When to use this tool:

Use this as your first analysis step to quickly understand the model structure and statistics.

■ Example:

Run simple analysis → Gets: Database info, model stats (15 tables, 245 measures, 89 columns), all tables, measures, columns, relationships, calc groups, and security roles. Includes detailed expert insights!

■ Parameters (what you provide):

mode (optional): 'all' (default - runs everything), 'tables', 'stats', 'measures', 'columns', 'relationships', 'roles', 'database', 'calculation_groups'

■ What you get back:

Comprehensive model overview with operation-by-operation results, detailed Power BI expert analysis, and recommendations. Execution time: ~2-5 seconds.

■ Comprehensive Model Analysis (Full) (05_live_model_full_analysis)

What it does: Deep analysis of the model with Best Practice Analyzer (120+ rules), performance analysis, and data integrity checks.

When to use this tool:

Use for thorough model health check - identifies issues, optimization opportunities, and best practice violations.

■ Example:

Run full analysis → Finds: 15 best practice violations (unused columns, missing descriptions), 3 performance issues (high cardinality, large tables), 2 integrity issues (circular dependencies).

■ Parameters (what you provide):

scope (optional): 'all' (default), 'measures', 'model', 'performance'; depth (optional): 'quick', 'balanced' (default), 'thorough'; include_bpa, include_performance, include_integrity (optional, all default true); max_seconds (optional): Time limit

■ What you get back:

Detailed analysis with categorized issues (error/warning/info), business impact assessment, and recommendations. Includes BPA rules, performance metrics, cardinality analysis, and integrity checks. Execution time: 10-180 seconds.

06 - Dependencies

■ Analyze Measure Dependencies (06_analyze_measure_dependencies)

What it does: Shows the complete dependency tree for a measure - what it depends on and what depends on it.

When to use this tool:

Use to understand measure relationships before making changes, or to trace calculation logic.

■ Example:

Analyze 'Profit Margin' → Shows: Depends on: 'Total Profit' and 'Total Revenue', which in turn depend on base columns and other measures. Full tree with levels.

■ Parameters (what you provide):

table (required): Table name, measure (required): Measure name

■ What you get back:

Dependency tree showing all referenced measures, columns, and tables in a hierarchical structure.

■ Get Measure Impact Analysis (06_get_measure_impact)

What it does: Shows what would be affected if you change or delete this measure (impact analysis).

When to use this tool:

Use before modifying or deleting a measure to see what else might break.

■ Example:

Check impact of 'Total Sales' → Shows: Used by 15 other measures (Profit Margin, Sales Growth, etc.), 3 reports, 5 visuals. High impact - be careful!

■ Parameters (what you provide):

table (required): Table name, measure (required): Measure name

■ What you get back:

List of all measures, reports, and visuals that reference this measure.

07 - Export

■ Get Live Model Schema (07_get_live_model_schema)

What it does: Exports a lightweight JSON schema of the model structure (tables, columns, measures, relationships) - optimized for AI analysis with low token usage.

When to use this tool:

Use to get a quick snapshot of the model structure for documentation or analysis without DAX expressions.

■ Example:

Export schema → Returns JSON with: 15 tables, each with columns (names, types), measures (names only), relationships (from/to).

■ Parameters (what you provide):

None required

■ What you get back:

Compact JSON schema without DAX expressions, perfect for quick model understanding.

■ Export TMDL Definition (07_export_tmdl)

What it does: Exports the complete model definition as TMDL (Tabular Model Definition Language) files - the source code of your Power BI model.

When to use this tool:

Use for version control, backup, or to edit the model as code (advanced users with Tabular Editor).

■ Example:

Export to C:\MyModel\TMDL → Creates folder with .tmdl files for all tables, measures, relationships, etc.

■ Parameters (what you provide):

output_dir (required): Where to save the TMDL files

■ What you get back:

Success message with path to exported TMDL files.

08 - Documentation

■ Generate Model Documentation (Word) (08_generate_model_documentation_word)

What it does: Creates a comprehensive Word document with complete model documentation: tables, columns, measures, relationships, data sources.

When to use this tool:

Use to create professional documentation for sharing with team, stakeholders, or for compliance.

■ Example:

Generate docs → Creates 45-page Word document with: Overview, all tables with columns and types, all measures with DAX, relationship diagram, data sources, best practices violations.

■ Parameters (what you provide):

output_path (optional): Where to save the document, include_sections (optional): Array of sections to include

■ What you get back:

Word document (.docx) with formatted, professional model documentation.

■ Update Model Documentation (Word) (08_update_model_documentation_word)

What it does: Updates an existing documentation Word file with current model state (faster than regenerating).

When to use this tool:

Use when you already have documentation and just want to refresh it with latest changes.

■ Example:

Update existing docs → Updates tables section, adds 3 new measures, updates modification date.

■ Parameters (what you provide):

document_path (required): Path to existing Word document, sections_to_update (optional): Which sections to refresh

■ What you get back:

Updated Word document with current model information.

09 - Comparison

■ Compare Two Models (09_Compare_Open_Live_Models)

What it does: Compares two Power BI models side-by-side to find differences (useful for comparing versions or dev vs. prod).

When to use this tool:

Use to see what changed between versions or to compare development and production models.

■ Example:

Compare Model A (dev) vs. Model B (prod) → Shows: 5 new measures in dev, 2 tables added, 3 relationships modified, 12 measures changed DAX code.

■ Parameters (what you provide):

model1_index (required): First model index, model2_index (required): Second model index

■ What you get back:

Detailed comparison report with all differences categorized (added, removed, modified).

10 - PBIP Analysis

■ Offline PBIP Analysis (HTML) (10_pbip_analysis_html)

What it does: Analyzes a PBIP project folder (the new Power BI project format) and generates an HTML analysis report - works offline without Power BI running.

When to use this tool:

Use to analyze PBIP projects stored in Git/version control without opening Power BI Desktop.

■ Example:

Analyze C:\Projects\SalesModel.pbip → Generates interactive HTML report with: model structure, all DAX measures, relationships, statistics, and analysis.

■ Parameters (what you provide):

pbip_folder_path (required): Path to .pbip project folder, output_path (optional): Where to save HTML

■ What you get back:

Interactive HTML report with model analysis, browsable in any web browser.

11 - TMDL Operations

■ TMDL Find & Replace (11_tmdl_find_replace)

What it does: Finds and replaces text in TMDL files (the model source code) with support for regular expressions.

When to use this tool:

Use to batch-update DAX code across multiple measures (e.g., replace table references, fix typos, update functions).

■ Example:

Replace 'FactSales' with 'Sales' in all measures → Finds 47 matches across 23 measures, optionally applies changes.

■ Parameters (what you provide):

tmdl_path (required): Path to TMDL folder, pattern (required): Text to find, replacement (required): Text to replace with, regex (optional): Use regex, dry_run (optional, default true): Preview only

■ What you get back:

List of all matches found with file locations and preview of changes. If dry_run=false, applies changes.

■ TMDL Bulk Rename (11_tmdl_bulk_rename)

What it does: Renames multiple objects (measures, tables, columns) in TMDL files and automatically updates all references.

When to use this tool:

Use to rename objects safely - it updates all DAX code that references the renamed objects.

■ Example:

Rename 'OldMeasureName' to 'NewMeasureName' → Updates the measure definition AND all 15 measures that reference it in their DAX code.

■ Parameters (what you provide):

tmdl_path (required): Path to TMDL folder, renames (required): Array of rename operations with object_type, old_name, new_name, dry_run (optional, default true): Preview only, update_references (optional, default true): Update references

■ What you get back:

List of all renames and reference updates. If dry_run=false, applies changes.

■ Generate TMDL Script (11_tmdl_generate_script)

What it does: Generates a TMDL script (code) from a model definition for creating or updating model objects.

When to use this tool:

Use to create reusable TMDL scripts for model deployment or automation (advanced users).

■ Example:

Generate script for all measures in 'Sales' table → Creates .tmdl script that can be applied to other models.

■ Parameters (what you provide):

definition (required): What to generate script for, output_path (optional): Where to save script

■ What you get back:

TMDL script file that can be applied to models.

12 - Help

■ Show User Guide (12_show_user_guide)

What it does: Displays comprehensive user guide with examples, best practices, and tool usage instructions.

When to use this tool:

Use whenever you need help or want to learn about specific features.

■ Example:

Show user guide → Returns: Full documentation with categories, tool descriptions, examples, and tips.

■ Parameters (what you provide):

section (optional): Specific section to show

■ What you get back:

User guide with usage instructions and examples.

13 - Hybrid Analysis (Advanced)

■ Full Model Export (PBIP + Sample Data) (13_full_model_pbip_and_sample_export)

What it does: Exports a complete package combining: TMDL files from PBIP folder + metadata + sample data from active model. Perfect for AI analysis!

When to use this tool:

Use to create a complete offline analysis package that includes structure (TMDL), metadata, and actual sample data for comprehensive analysis.

■ Example:

Export Sales model → Creates folder with: TMDL files (structure), metadata JSON (stats), sample data parquet files (first 1000 rows per table). Perfect for AI to understand data patterns!

■ Parameters (what you provide):

pbip_folder_path (required): Path to .SemanticModel folder, output_dir (optional): Where to save, include_sample_data (optional, default true), sample_rows (optional, default 1000)

■ What you get back:

Complete package folder with TMDL, JSON metadata, and Parquet sample data files.

■ Full Model Analysis (PBIP + Sample Data) (13_full_model_pbip_and_sample_analysis)

What it does: BI EXPERT ANALYSIS - Automatically reads exported package (TMDL + metadata + sample data) and provides comprehensive AI-powered analysis with recommendations. Supports fuzzy search!

When to use this tool:

Use for the most comprehensive AI-powered model analysis - it reads EVERYTHING internally (you don't need to read files manually). Just point it at the analysis folder!

■ Example:

Analyze exported package at C:\Analysis\SalesModel_analysis → AI reads all TMDL, metadata, and sample data → Provides: Model structure analysis, DAX pattern recommendations, data quality insights, relationship issues, optimization opportunities, best practices violations. Supports fuzzy search like 'show me revenue measures'.

■ Parameters (what you provide):

analysis_path (required): Path to exported analysis folder, operation (required): 'smart_analyze' for AI analysis, intent (optional): Natural language question for fuzzy search, format_type (optional): 'json' or 'toon' (token-optimized)

■ What you get back:

Comprehensive BI Expert analysis with: Structure insights, DAX recommendations, data quality analysis, relationship validation, optimization suggestions, best practices, and specific action items. All based on actual data patterns!

Appendix: Common Workflows

Workflow 1: First-Time Model Analysis
1. 01_detect_pbi_instances → Find your model
2. 01_connect_to_instance → Connect to it
3. 05_live_model_simple_analysis → Quick overview (2-5 seconds)
4. 05_live_model_full_analysis → Deep health check (optional, 10-180 seconds)
5. 08_generate_model_documentation_word → Create documentation

Workflow 2: Analyze and Optimize a DAX Measure
1. 02_list_measures → Find your measure (or just remember the name)
2. 03_standard_dax_analysis → Analyze it (just provide measure name - auto-fetches!)
3. Review: Validation, performance, optimization suggestions, rewritten code
4. 04_upsert_measure → Apply optimizations
5. 06_analyze_measure_dependencies → Check what depends on it

Workflow 3: Create Multiple Time Intelligence Measures
1. 02_list_measures → See existing patterns
2. 04_bulk_create_measures → Create YTD, MTD, QTD, PY, YoY versions
3. 03_standard_dax_analysis → Validate each one

Workflow 4: Compare Development vs Production Models
1. 01_detect_pbi_instances → Find both models
2. 01_connect_to_instance → Connect to first (index=0)
3. 09_Compare_Open_Live_Models → Compare them
4. Review differences and decide what to sync

Workflow 5: Complete AI-Powered Analysis (Most Powerful!)
1. 13_full_model_pbip_and_sample_export → Export complete package (TMDL + metadata + sample data)
2. 13_full_model_pbip_and_sample_analysis → AI analyzes EVERYTHING
3. Get: Structure analysis, DAX recommendations, data quality insights, optimization tips
4. Supports fuzzy search: "show me revenue measures" or "analyze sales calculations"

Workflow 6: Refactor Model (Rename Objects Safely)
1. 07_export_tmdl → Export current model as TMDL
2. 11_tmdl_bulk_rename → Rename objects (updates all references automatically!)
3. Apply TMDL back to model

Workflow 7: Find and Fix Performance Issues
1. 05_live_model_full_analysis → Identify performance issues
2. 03_list_relationships → Check for Many-to-Many relationships
3. 02_list_calculated_columns → Find calculated columns to convert to measures
4. 03_standard_dax_analysis → Optimize DAX code
5. Review VertiPaq metrics and cardinality

Tips for Success:
• Always connect first (tools 01) before using other tools
• Use simple_analysis (05) before full_analysis - it's faster and often sufficient
• The DAX Intelligence tool (03) has smart measure finder - just type measure name!
• Use dry_run=true when testing changes (previews without applying)
• Export documentation regularly for team sharing
• Use hybrid analysis tools (13) for the most comprehensive AI-powered insights

Getting Help:
• Use tool 12_show_user_guide for built-in help
• Check GitHub: https://github.com/bibiibjorn/MCP-PowerBi-Finvision
• All tools include detailed error messages to guide you