
#FABCONSQLCON2026
FABCON
Microsoft Fabric
COMMUNITY CONFERENCE
SQLCON
Microsoft SQL
COMMUNITY CONFERENCE
ATLANTA MARCH 16 - 20, 2026

Piotr Prussak

Data & AI Architect @ Imidia and Consultant

Certifications

PL-300 Power BI Data Analyst

DP-600 Fabric Analytics Engineer

DP-700 Fabric Data Engineer

AI-102 Azure AI Engineer

CSPO

Linked In: www.linkedin.com/in/piotr-prussak

Email: Piotr.Prussak@imidia.us

Session Roadmap

- **1 Honest Caveats** — What you need to know before investing time
- **2 AI Solutions Landscape** — Data Agents, Copilot Studio, and where they fit
- **3 Setup, Prerequisites & Costs** — What it takes to get started
- **4 Solution Walkthroughs** — Three data scenarios, increasing complexity
- **5 Deep Dives** — Modeling, schema design, grounding, and testing
- **6 Decision Guides** — Take-home frameworks for your Monday morning

GitHub: https://github.com/ptprussak/wwimporters

Honest Caveats

Setting expectations so you can invest wisely

Caveat #1: This Is ~~Preview~~ GA

Fabric Data Agents + Copilot Studio was in preview when I was building the demos.

What this means:

- New features ship monthly — MCP endpoints, M365 Copilot integration, and ontology support all landed in the last 6 months
- Some features may ~~be removed or~~ rearchitected before GA
- SLAs, performance guarantees, and full documentation are not yet final
- *Build to learn, not to bet the farm — yet*

Caveat #2: Microsoft Follows Adoption

Microsoft invests in features that get used — and mothballs what doesn't stick

Historical examples:

- Cortana Intelligence Suite — launched with fanfare, quietly retired
- Power BI Dataflows v1 — replaced by Dataflows Gen2 in Fabric
- Data Activator — promising concept, unclear adoption trajectory

The signal to watch: Is your organization actually using this in production?

- *If you adopt early, you influence the roadmap. If you wait, the feature may not survive.*

Caveat #3: Set a 3-Month Horizon

The AI/agentic space moves faster than enterprise planning cycles.

Evidence of pace:

- GPT-4o retired from Copilot Studio (Oct 2025)
- GPT-4.1 became default model (Oct 2025)
- GPT-5 reached GA in Copilot Studio (Nov 2025)
- Three model generations in under two months

Recommendation: Schedule a formal re-evaluation every 3 months

- Review: What changed? What broke? What new capability unlocks a use case?
- Work within the community to assess where to invest your energy

Caveat #4: This Is One Piece of the Toolkit

Fabric Data Agents are one part of a broader agentic AI ecosystem

The bigger picture:

- Azure AI Foundry Agents — custom agents with code-first control
- Semantic Kernel — orchestration framework for complex AI workflows
- Microsoft 365 Copilot — the end-user surface for enterprise AI
- Copilot Studio — the low-code agent builder and orchestrator
- MCP (Model Context Protocol) — the emerging interop standard

Processes will change. Not just tools — the way we think about data access will change.

AI Solutions in Fabric & Copilot

Understanding the landscape before choosing a path

What Is a Fabric Data Agent at 10,000 ft?

AI-powered assistants that engage in natural language conversations about your enterprise data — positioned as an AI-enhanced analytical solution.

Core capabilities:

- Understands your data schema across lakehouses, warehouses, Semantic Models, KQL databases, graphs, and ontologies
- Can connect to a subset of data, or combine multiple sources
- Can be used as a typical agent (i.e., consumed by other agent orchestrators)
- Enforces governance — RLS, CLS, and user permissions flow through automatically
- Interprets business context to surface relevant, actionable insights
- Stores conversation history across sessions for continuity
- Can generate simple data visualizations

Key distinction: These are not just NL-to-SQL. They reason across multiple data sources and maintain context.

What Is Copilot Studio?

Low-code platform for building custom AI agents with multi-agent orchestration. Note: requires a fundamentally different skillset.

What it brings to the table:

- Visual agent builder with topics, triggers, and knowledge sources
- Connected Agents — link Fabric Data Agents as specialized "experts"
- Multi-channel deployment: Teams, web, M365 Copilot, custom apps
- Model flexibility — currently on GPT-5 GA + Claude, with versioning controls
- Builds and assembles complete agentic applications – combining Power Automate + Chat + Agents

The pattern: Copilot Studio = orchestrator. Fabric Data Agent = domain expert.

- *User asks in Teams → Copilot routes → Data Agent queries → grounded answer returns*
- *Caveat: Copilot Studio is a complex set of tools that is outside of the Fabric ecosystem.*

Do we need Copilot Studio to call Fabric Data Agents?

- *Not required, but strongly advisable — additional controls available compared to direct "publish to Agent Store in M365 Copilot" option.*
- *Large surface area of the tool means it's a "dedicated role". It's an ecosystem for just like Fabric*

Where Do These Fit in “agent” and “analytics” space?

Choose based on your scenario:

- Native Copilot in Power BI — user is already in a report, needs contextual Q&A on visible data
- **Fabric Data Agent (standalone)** — domain expert on a specific dataset, used by analysts directly in Fabric
- Data Agent + Copilot Studio — multi-agent orchestration, mixed knowledge sources, deployed to Teams/web
- Azure AI Foundry / Semantic Kernel — full code-first control, custom RAG, complex workflows beyond data Q&A

Start with the simplest option that meets the use case. Escalate complexity only when needed.

Consumption surfaces (today):

- Power BI Copilot Pane — ask questions while viewing reports
- Fabric workspace — interact with data agents directly in the Data Science experience
- Microsoft Teams — via Copilot Studio published agents
- Microsoft 365 Copilot — data agents alongside other M365 AI capabilities
- Custom apps — via public API endpoints
- VS Code / AI tooling — via managed MCP server endpoints (new at Ignite 2025)

Setup, Prerequisites & Costs

What it takes to get started

Prerequisites Checklist

Fabric capacity:

- F2 or higher (or P1+ with Fabric enabled) — Copilot/AI included on all paid SKUs since April 2025
- Ability to manage that capacity in Azure

Tenant settings (admin portal):

- Fabric Data Agent tenant setting — enabled
- Cross-geo AI processing and storage — enabled
- XMLA endpoints — enabled (for semantic model data sources)
- Standalone Copilot experience — enabled

Data sources (at least one, with data):

- Warehouse, Lakehouse, Power BI Semantic Model, SQL and KQL Database, graph

Copilot Studio: Same tenant, same account, M365 Copilot license

Security: Entra Groups, Agent 365 Admin Center

Roles & Security Model

Who needs what access?

- Data Agent Author: Workspace Contributor+ role, access to underlying data sources
- Data Agent Consumer: At least Read access to the published agent
- Copilot Studio Author: M365 Copilot license + Copilot Studio author role
- End User (Teams): Access to the published Copilot Studio agent

Critical decision: Authentication mode

- User Authentication — queries run as the end user (RLS enforced per user)
- Agent Author Authentication — queries run as the author (simpler, but shared access)
- *For most enterprise scenarios, User Authentication is the right choice — but it requires each user to have data source access.*

Costs — No Surprises

Fabric capacity (required):

- F2: ~$262/month (Copilot/AI included at no extra cost since April 2025) but recommending F4 (that can be paused if needed)
- Below F64: Users need Pro or PPU licenses for Power BI content (e.g., Power BI + Semantic Model development)
- F64+: No individual Power BI licenses needed for consumers

Copilot Studio (if building custom agents):

- Pay-as-you-go: $0.01 per Copilot Credit (Azure subscription required)
- Prepaid: $200/tenant/month for 25,000 credits (monthly) but credit accounting is confusing
- M365 Copilot license: $30/user/month (required for authoring and testing). Small Businesses have a promo at $20.
- Copilot licenses also require an Office license.

Cost optimization tip:

- Start with 1-2 Copilot licenses (developer + tester), then pay-as-you-go Copilot Studio for pilot projects
- Use F2 for dev/test, scale capacity only when needed

Costs are not minimal — suited for corporate development teams, less so for independent AI/Data developers on a budget.

Deployment Options

Spectrum from simple to orchestrated:

Level 1: Fabric Data Agent (standalone)

- Create agent → add data source → publish → add security → users query in Fabric. Potential time to value: **hours**

Level 2: Data Agent in Power BI Copilot

- As above: Attach published agent to Copilot pane → users ask questions alongside reports. Potential time to value: **hours**

Level 3: Data Agent + Copilot Studio → Teams

- Build Copilot Studio agent → connect data agent → add topics/triggers → publish to Teams. Time to value: days
- Add fabric CICD or Deployment

Level 4: Multi-agent orchestration

- Multiple specialized data agents → Copilot Studio orchestrator → deployed across channels

Consider the team/roles that's needed to build a solution:

- Fabric: Admin: secure data, Data Architects that understand the model, Data/Business Analyst that understands user questions, AI Developer to structure Prompts
- Entra: Manage security groups and licenses. Copilot Developer when working with Copilot Studio, Office Admin: Manage Copilot Store

Other Considerations

Current limit of the data agent - returns only 25 rows (according to Reddit post – being fixed very soon!)

You need to have a security strategy for Agent + Data that matches your usage

- Security group is best
- Secure the data (read rights in Fabric OR read + build for semantic models) and share the agent to the security group

Prompt safety is key. Make assumption that unless the agent goes through Copilot Studio or AI Foundry it is your responsibility to prevent any prompt injection.

The Agent Development experience is still only via browser + Python SDK (Preview). Hopefully, this changes soon:

- Part of the VSCode Fabric toolkit?
- Better hooks to test the solution

Solution Walkthroughs

Three scenarios, increasing complexity

Solution 1: Technical / Operations

Scenario: Fabric Capacity Metrics App – use OOB semantic model: Note – totally unsupported by Microsoft and may break

Characteristics:

- Operational data — looking at the model you can identify basic entities
- Some clear entity relationships — capacity, workspace, user, operations
- Many challenging tables and measures
- Numeric-heavy — CU consumption, refresh counts, query durations if you can get into the weeds

Example queries:

- “Which item kinds uses the most capacity” – Note: you need to know the fabric metrics jargon
- “Show me Data Agents usage on Capacity”

Solution 1: Case When This Simple Demo Works

Agent excels when:

- Schema is narrow and self-descriptive
- Column names are unambiguous (“peak_cu_usage” not “value1”)
- Questions map to single-table aggregations
- Domain is technical — less room for interpretation

This is your MVP path if you already have data in Fabric and are monitoring it (but totally unsupported)

- Zero sample data setup — capacity metrics are your own real data
- Immediate relevance to Fabric practitioners
- Fast time to value for identify what works and what does not

If you can only build one data agent to prove the concept, start here.. But very basic questions

More challenging questions are out of scope here… would need to prep specifically for AI and perhaps use FUAM instead:

fabric-toolbox/monitoring/fabric-unified-admin-monitoring at main · microsoft/fabric-toolbox · GitHub

fabric-toolbox/monitoring/workspace-monitoring-dashboards at main · microsoft/fabric-toolbox · GitHub

DEMO

Solution A: Capacity Metrics Agent

- *Querying operational data with a narrowly-scoped Fabric Data Agent*

Challenges

- *Model not intuitive and easily understood by LLM*
- *No options to provide good samples for LLM to execute*
- *Works for simple queries, fails at anything more sophisticated*
- *Lesson learned: will LLM know how to interpret a complex technical model?*

Solution 2A: Business Data — Complex Schema, Basic Agents

Scenario: Wide World Importers

- Star schema with clear fact/dimension separation
- Added modifications to the schema to show how to fix them later
- Always use descriptive column names ("total_order_amount" not "amt")
- Column descriptions populated as grounding anchors in prompts

Modifications to base model — created heavy schema complexity:

- Many-to-many relationships (supplier substitutions, post tags)
- Temporal/SCD patterns (slowly changing dimensions, temporal tables)
- Multi-granularity fact tables (daily vs. monthly aggregations)
- Self-referencing hierarchies (posts → answers → comments)

Why this matters in this context:

- This is what real enterprise data looks like
- If agents can't handle this, they can't handle your production schemas
- The objective is to learn how to identify these issues and fix them

Solution 2A: What Goes Wrong

- **Common agent failures on complex schemas:**
- Ambiguous joins — agent picks the wrong path through M:M relationships
- Temporal confusion — agent doesn't know which date column represents "current"
- Granularity mismatch — agent aggregates at wrong level, producing nonsense numbers
- Hallucinated columns — agent invents column names that sound right but don't exist
- Over-joining — agent joins 6 tables when the answer was in one

The lesson:

- Raw complex schemas are hostile to AI agents
- The agent is not broken — the schema was never designed for this consumer
- 2 options – simplify the schema that agent has access to OR try to build more sophistication into the agent
- *Data Engineer takeaway: This is where your modeling choices become critical.*
- *AI Engineer takeaway: This is where prompting and samples become critical*

Solution 2B: Business Data — Complex Schema

Scenario: Wide World Importers

- **Schema complexity:**
- Many-to-many relationships (supplier substitutions, post tags)
- Temporal/SCD patterns (slowly changing dimensions, temporal tables)
- Multi-granularity fact tables (daily vs. monthly aggregations)
- Self-referencing hierarchies (posts → answers → comments)

Why this matters:

- This is what real enterprise data looks like
- If agents can't handle this, they can't handle your production schemas

DEMO

Solution 2A: Complex Schema, Simple Agent

- *Demonstrating failure modes on Wide World Importers*

Solution 2B: Complex Schema, Smarter Agent

- *Same questions, better instructions — showing why modeling and prompting matters*

Code for these last 2 demos (requires local SQL for unpacking of the SQL data .bacpac file)

https://github.com/ptprussak/wwimporters

The Punchline

How can we improve the output of the agent

- Modeling is the #1 lever for agent accuracy
- Agent instructions are the #2 level for agent accuracy
- The LLM engine agent didn't get smarter between Solution 2A and Solution 2B
- **The data got clearer**
- Your data engineers and modelers are the most important people in this story
- *Investment in schema design pays off exponentially when AI agents are the consumer.*
- *This is not optional work — it is the prerequisite for production-quality agent responses.*

DEMO

Publishing process to Copilot / Teams

- *Simple way to deploy and share to a group of users*

Publishing process to Copilot Studio

- Create an agent that consumer the Fabric Data Agent via Fabric MCP
- Additional instructions and controls needed to set up a basic agent
  - Add sample question prompts that show up in teams or guide to specific agents via topics
- Controls on what other tools the agent can use and what knowledge the agent can leverage
  - Beware of large surface area of configuration
  - Turn off general knowledge in “settings”
  - Select LLM
  - Set Safety and Moderation

- Not covered
  - Adding knowledge directly (documents) or other RAG/Search is easy, but requires more testing
  - Copilot Studio has rich testing and operational monitoring capability
  - Copilot Studio will also publish to other delivery channels (Web/Facebook/Slack/etc.)

Deep Dives

Modeling, schema, grounding, and testing

Data Agent Modeling (AI Engineer View)

What the agent "sees":

- Table names, column names, data types, relationships, and descriptions
- It does NOT see your data values unless it queries them

Best practices:

- Use descriptive, unambiguous names — avoid abbreviations and codes
- Define explicit foreign keys and cardinality
- Write rich column descriptions — the single highest-ROI grounding mechanism
- Hide internal/technical columns the agent should never reference
- Use meta-prompting: ask the agent to generate its own instructions from the schema

Measures vs. calculated columns:

- Prefer calculated columns for values agents need to filter/group on
- Measures work for aggregations but agents sometimes struggle with DAX context

Schema Design (Data Engineer View)

Design for your new consumer — the AI agent:

- Denormalize strategically — flatten M:M into bridge-free views where possible
- Resolve SCD ambiguity — create “current” views alongside history tables
- Eliminate field name collisions — “date” appears in 12 tables; which one?
- Separate concerns — one semantic model per bounded domain, not one mega-model

Agent instructions:

- Capped at 15,000 characters — be concise and precise, Define what the agent should and should NOT answer
- Data source instructions capped at 800, up to 5 data sources per agent
- Include example queries and expected patterns
- *IT governance note: Schema decisions here have downstream implications for security, lineage, and compliance.*

Getting Grounded Responses

Grounding = responses tied to actual data, not hallucinated

- **Techniques:**
- Agent instructions — constrain scope explicitly ("Only answer questions about sales data")
- Agent descriptions — specific and non-overlapping; directly affects orchestration routing
- Column descriptions — the single most impactful grounding mechanism
- Semantic model layer — acts as a grounding buffer between raw data and the agent

What to prevent:

- Out-of-domain answers ("I'm a sales agent but sure, let me guess about HR data")
- Hallucinated joins (agent invents relationships that don't exist)
- Confident wrong answers (agent returns plausible but incorrect aggregations)
- *The goal is not "always answers" — it's "answers correctly or says it can't."*

Testing Patterns

Build your test library before production, not after

Golden question sets:

- 20+ questions with known correct answers, covering edge cases
- Run after every model update, schema change, or instruction edit
- Track accuracy over time — regressions are common after model swaps

What to test for:

- Correct answers on straightforward questions (baseline accuracy)
- Graceful refusal on out-of-scope questions (does it say "I don't know"?)
- Consistency across phrasings (same question worded 3 ways = same answer?)
- Performance under concurrent load

Copilot Studio supports side-by-side agent version comparison — use it.

- *Monitor in production. Wrong answers at scale erode trust faster than no answers.*

Decision Guides

Current Workflow

Take-home frameworks for your Monday morning

Semantic Model vs. Direct SQL / Lakehouse

Use a Semantic Model when:

- Business logic is encoded in DAX measures and follows Star schema
- You need consistent calculations across reports and agents
- RLS/CLS is already defined in the model
- The domain is well-bounded (sales, finance, HR)
- You prefer to “optimize for AI” as opposed to write specific complex samples

Go direct to other data sources when:

- Data is exploratory or ad-hoc (data science use cases)
- OR you need a very specific agent focus that requires heavy custom instructions
- Schema needs to be simple and self-descriptive
- You need access to data not yet modeled (raw ingestion layers)
- Performance requires pushing compute to the engine

Hybrid approach: Semantic models for governed “golden path” answers, direct access for exploration

Copilot Studio vs. Native Fabric Copilot

Native Copilot in Power BI when:

- Users are already in reports
- Questions are contextual to visible data
- No custom orchestration needed
- Quick deployment, minimal setup

Plain Data Agent exposed via Teams:

- PowerBI Copilot does not provide sufficiently good experience
- Need to consume data via Teams/Copilot without the need to open PowerBI

Copilot Studio when:

- Multi-agent orchestration (multiple data domains)
- Custom topics, triggers, and conversation flows
- Deployment to Teams, web, or M365 Copilot
- Mixed knowledge sources (SharePoint + Fabric + websites)
- Need for human-in-the-loop or scheduled automation workflows
- *If "ask a question about this report" is enough → Native. If "build a domain expert" is the goal → Studio.*

Development Workflow

Native Copilot in Power BI when:

- Users are already in reports and semantic model is well understood by LLM
- Prep Model for AI and run the Model via tools that can unpack the model: like Claude or ChatGPT (the pro versions) scan for ambiguities optimization

Data Agents as agents in Fabric:

- Similar Prep compared to what happened to Semantic Model as minimum but at larger scale
  - Leverage tools like GitHub Copilot/Clause Code/Codex to manage gaps in your team
- Consult the Business Domain Expert about the questions a user is likely to ask
  - Also, this could be Copilot Deep Research/Claude that could improve this question set
- Generate the questions, and decide on how narrow/wide set of questions the agent will need to work with
- Consult the Data Domain Expert (or LLM as part of the same session) to identify elements of the underlying data model that will be used
  - Pull out the schema from the source and add to your Claude Code/Codex/GitHub Copilot project
- Identify any optimization options (custom views which simplify or disambiguate meaning of the model
- Consult the AI developer on creating proper instructions
  - Again, AI Coding tools will create instructions for Data Agents – but do provide full context and constrains
- As Team collaboration: Data Analyst + Domain Expert + AI developer: develop sample questions for additional training
  - Once again, LLM can help, but must know the schema + the dialect of the language you are using (and will make mistakes)

Signs You’re Not Ready to Deploy

- No golden question test set
- No column descriptions in your semantic model or schema
- No clear domain boundary for the agent (“it should answer everything”)
- No executive sponsor who understands “this is preview”
- No plan for monitoring agent responses in production
- No defined escalation path for wrong answers
- *If more than two of these apply, invest in readiness before deployment.*

Closing

What to do Monday morning

Five Things to Do Monday Morning

- 1. Verify your Fabric tenant settings — enable Data Agents, Copilot, and XMLA endpoints
- 2. Build one data agent on your data model you understand — prove the platform works
- 3. Audit one production semantic model — add column descriptions, check naming clarity
- 4. Write 20 golden test questions for your most likely agent domain
- 5. Schedule a 3-month re-evaluation checkpoint in your calendar

The 3-Month Rule

Today's date: March 2026

Your first checkpoint: June 2026

Review:

- What new capabilities shipped?
- What broke or changed since your last evaluation?
- Which of your pilot use cases proved value?
- What can you now do that you couldn't three months ago?

Re-evaluate. Adapt. Repeat.

- *The only bad strategy is the one that doesn't evolve.*

Resources & Contact

Documentation:

- Fabric Data Agents: learn.microsoft.com/fabric/data-science/data-agent-overview
- Copilot Studio + Fabric: learn.microsoft.com/fabric/data-science/data-agent-microsoft-copilot-studio
- Fabric Copilot Capacity: learn.microsoft.com/fabric/enterprise/fabric-copilot-capacity

Community:

- Fabric Community: community.fabric.microsoft.com
- Copilot Studio Labs: microsoft.github.io/mcs-labs
- Fabric User Panel: aka.ms/JoinFabricUserPanel

Connect with me:

www.linkedin.com/in/piotr-prussak

Sound off.
The mic is all yours.
Influence the product roadmap.

Join the Fabric User Panel

Share your feedback directly with our Fabric product group and researchers.

https://aka.ms/JoinFabricUserPanel

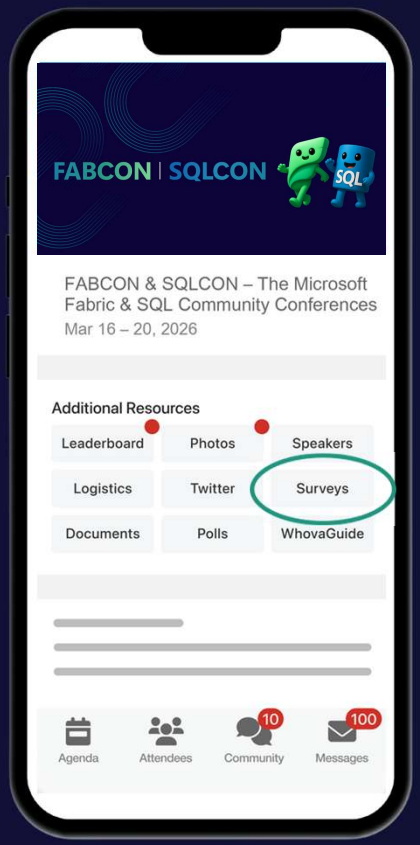
Join the SQL User Panel

Influence our SQL roadmap and ensure it meets your real-life needs

https://aka.ms/JoinSQLUserPanel

How was the session?

Complete Session Surveys in Whova for your chance to WIN PRIZES!



Two Fabric Certifications, One FREE Exam Included

Attendees can take the Fabric Analytics Engineer or Fabric Data Engineer exam for free. Be part of the 2 fastest growing role-based certifications in Microsoft history.

Request your voucher by March 31, 2026.

https://aka.ms/GetDataCertified