

Build faster with Solution Accelerators

Foundry IQ + Fabric IQ

Hands-on Workshop

Build end-to-end AI solutions that unify
enterprise data sources - then accelerate
your PoCs with real customer data.

Updated: February 05, 2026

Table of Contents

1. Introduction	What you will build
2. Prerequisites	What you need before starting
3. Architecture Overview	Understanding the components
4. Part 1: Run with Sample Scenario	Telecommunications example walkthrough
5. Part 2: Customize Your Own Scenario	Create your industry solution
6. Understanding the Build Process	What each step does
7. Testing Your Agent	Interactive chat with your data
8. Troubleshooting	Common issues and solutions

1. Introduction

Welcome to the **Foundry IQ + Fabric IQ** hands-on workshop! In this workshop, you will build an end-to-end AI solution that unifies enterprise data sources - documents and structured data - into a single intelligent agent.

The Opportunity

Organizations have valuable knowledge spread across documents (PDFs, policies, manuals) and structured systems (databases, data warehouses). By connecting these sources through AI, users can get unified answers from a single conversational interface.

The Solution

Foundry IQ and **Fabric IQ** solve this by enabling an intelligent agent that:

- Creates knowledge bases from documents with agentic retrieval (plan, iterate, reflect)
- Defines business ontology to understand entities, relationships, and rules
- Queries data using natural language over both documents and structured data
- Unifies enterprise data sources into a single conversational interface

What You Will Build

By the end of this workshop, you will have created:

- **AI Agent** - Azure AI Foundry orchestrates tools and generates responses
- **Knowledge Base** - **Foundry IQ** provides agentic retrieval over documents
- **Business Ontology** - **Fabric IQ** defines entities, relationships, and NL-to-SQL
- An end-to-end AI solution that unifies enterprise data sources

Workshop Flow

This workshop is divided into two parts:

1. **Run with Sample Scenario** - Follow along with a pre-defined Telecommunications scenario to understand how everything works.
2. **Customize Your Own** - Use AI to generate data and documents for YOUR industry and use case.

Once you complete these two steps, you can plug in your own customer data and accelerate your Proof of Concept (PoC) engagements!

2. Prerequisites

Prerequisites

- **Azure Subscription** with Contributor access
- **Microsoft Fabric Workspace** with capacity assigned
- **VS Code** or access to create **GitHub Codespaces**

All other tools and resources will be set up during the workshop. The infrastructure deployment will create the necessary Azure AI Services, AI Search, and Storage resources automatically.

3. Architecture Overview

Foundry IQ and **Fabric IQ** work together to create an end-to-end AI solution that unifies enterprise data sources - documents and structured data - into a single intelligent agent.

Components

Foundry IQ (Document Intelligence)

- **Knowledge Base:** Agentic retrieval over documents (plan, iterate, reflect)
- **AI Search Index:** Vectorized document chunks for semantic search
- **Embedding Model:** Converts text to vectors (text-embedding-ada-002)

Fabric IQ (Structured Data Intelligence)

- **Fabric Lakehouse:** Stores structured data as Delta tables
- **Business Ontology:** Defines entities, properties, and relationships
- **Data Agent:** Translates natural language to SQL queries

Orchestration Layer

- **Azure AI Foundry Agent:** Orchestrates tools and generates responses
- **Orchestrator Agent:** Determines which source(s) to query

Data Flow

1. User asks a question in natural language
2. AI agent determines if the question needs structured data, documents, or both
3. For structured data: **Fabric IQ** converts question to SQL via Ontology
4. For documents: **Foundry IQ** retrieves relevant document chunks
5. AI agent combines results and generates a natural language response

4. Part 1: Run with Sample Scenario

In this section, you will run the complete build process with a pre-defined Telecommunications scenario. This helps you understand how each component works before creating your own custom scenario.

The Telecommunications Scenario

Our sample scenario is a Telecommunications system for network operations. It includes:

- Network outages table: Outage events, duration, impact level
- Trouble tickets table: Support tickets linked to outages
- Policy documents: Service policies and procedures (PDFs)

Step 1: Run the Build Script

The build script automates all steps. Run this single command:

```
python scripts/00_build_solution.py \
    --industry "Telecommunications" \
    --usecase "Network operations with outage tracking"
```

Note

This uses pre-generated sample data from the data/default folder. No AI generation is needed for the sample scenario.

What the Build Script Does

1. Loads sample data (network_outages.csv, trouble_tickets.csv)
2. Creates Fabric Lakehouse and uploads data as Delta tables
3. Creates Fabric Ontology with entities and relationships
4. Generates PDF documents from the data
5. Creates the Fabric Data Agent
6. Uploads documents to AI Search with embeddings
7. Creates the Foundry orchestration agent

Step 2: Test the Fabric Data Agent

Once the Fabric Data Agent is created, open the Fabric UI and test it with natural language questions about your data.

Step 3: Test the Full Agent

Once the build completes, test the full orchestration agent:

```
python scripts/08_test_foundry_agent.py
```

Try these sample questions:

- How many outages occurred last month?
- What is the average resolution time for tickets?
- What are the policies for notifying customers?
- Which outages exceeded our policy thresholds?

5. Part 2: Customize Your Own Scenario

Now that you understand how the build process works, it is time to create your own custom scenario! The AI will generate realistic sample data and documents based on your industry and use case.

Step 1: Choose Your Industry and Use Case

Think about what industry and use case you want to explore. Here are some ideas:

Industry	Use Case
Telecommunications	Help support staff find outage info and answer policy questions
Finance	Enable advisors to assess loan eligibility and compliance requirements
Education	Let admins query enrollment data and academic policy questions
Manufacturing	Help technicians find equipment history and maintenance procedures
Hospitality	Enable front desk to check availability and answer guest inquiries
Real Estate	Help agents match properties to buyers and explain transaction steps
Insurance	Enable adjusters to review claims and coverage policy details
Energy	Help operators monitor assets and find safety protocol information

Step 2: Run with AI Generation

Run the build script with your industry and use case:

```
python scripts/00_build_solution.py --clean \  
  --industry "YOUR_INDUSTRY" \  
  --usecase "YOUR_USE_CASE"
```

For example, to create an Insurance claims system:

```
python scripts/00_build_solution.py --clean \  
  --industry "Insurance" \  
  --usecase "Claims processing and policy management"
```

Tip
Be descriptive in your use case! The more detail you provide, the better the AI can generate relevant data and documents.

Step 3: Switch Between Scenarios

When you want to try a different scenario, use the `--clean` flag to create fresh Fabric artifacts:

```
python scripts/00_build_solution.py --clean \  
  --industry "Finance" \  
  --usecase "Loan applications and credit scoring"
```

The `--clean` flag increments the artifact suffix (`lakehouse_1` -> `lakehouse_2`) so you can have multiple scenarios without conflicts.

6. Understanding the Build Process

The build process consists of several scripts that work together. Understanding each step helps you troubleshoot issues and customize behavior.

00_build_solution.py

Master orchestration script

Runs all other scripts in sequence. Accepts --clean, --foundry-only, --industry, and --usecase flags.

01_generate_sample_data.py

AI data generation

Uses GPT-4o-mini to generate realistic CSV data and PDF documents based on your scenario.

02_create_fabric_items.py

Fabric Lakehouse and Ontology

Creates the Lakehouse for data storage and Ontology for semantic understanding.

03_load_fabric_data.py

Data loading

Uploads CSV files to OneLake and loads them as Delta tables.

04_generate_agent_prompt.py

Schema extraction

Reads table schemas to generate prompts for the AI agent.

05_create_fabric_agent.py

Fabric Data Agent

Creates a Data Agent in Fabric that uses the Ontology to answer questions.

06_upload_to_search.py

Document indexing

Uploads PDF documents to AI Search with vector embeddings.

07_create_foundry_agent.py

Foundry orchestration agent

Creates the main AI agent that combines Fabric and Search tools.

7. Testing Your Agent

The test script provides an interactive chat interface to your AI agent. It shows you exactly how the agent processes your questions.

Running the Test

```
python scripts/08_test_foundry_agent.py
```

Understanding the Output

When you ask a question, the agent shows:

- Tool calls: Which tools (Fabric IQ, Foundry IQ) were used
- SQL queries: The exact SQL generated by Fabric IQ
- Search results: Document chunks retrieved by Foundry IQ
- Final answer: The natural language response

Sample Questions by Type

Structured Data Questions (uses Fabric IQ)

- How many records are in the table?
- Show me the top 5 items by value
- What is the average/sum/count of X?
- List all items where condition is met

Unstructured Data Questions (uses Foundry IQ)

- What is our policy on X?
- Tell me about the procedures for Y
- What guidelines exist for Z?

Combined Questions (uses both)

- Which customers have issues and what is the resolution policy?
- Show me overdue items and explain the escalation process

8. Troubleshooting

Tip

Use GitHub Copilot Chat (Ctrl+I in VS Code) for help with errors. Copilot can explain error messages and suggest fixes.

Ontology stuck on 'Setting up'

The Ontology creation can sometimes get stuck. Use --clean flag to create a new one with an incremented suffix.

```
python scripts/00_build_solution.py --clean ...
```

FABRIC_WORKSPACE_ID not set

Make sure your .env file contains the Fabric workspace ID. Get it from the Fabric portal URL.

```
FABRIC_WORKSPACE_ID=fb695e19-2010-...
```

AI generation produces invalid data

The AI retry mechanism will attempt 3 times. If it keeps failing, try simplifying your use case description.

```
# Use simpler, more specific use case descriptions
```

Rate limiting (429 errors)

The scripts have built-in retry logic. If you see many 429 errors, wait a few minutes and try again.

```
# Automatic retry with backoff is enabled
```

Search returns no results

Make sure step 06 completed successfully. Check that PDFs were generated and uploaded.

```
python scripts/06_upload_to_search.py --data-folder <PATH>
```

Fabric Data Agent not responding

The Data Agent needs time to index the Ontology. Wait 2-3 minutes after creation before testing.

```
# Wait for 'Agent is ready' message
```

Congratulations!

You have successfully completed the Foundry IQ + Fabric IQ hands-on workshop!

You now know how to:

- Use Fabric IQ to create business ontologies and query structured data
 - Use Foundry IQ to build knowledge bases over documents
 - Build AI agents with Azure AI Foundry
- Build end-to-end AI solutions that unify enterprise data sources

Next Step: Plug in your own customer data to accelerate your PoCs!

For questions and feedback, visit the [GitHub repository](#).