AGENTIC AI IN POWER PLATFORM

Module 04

MCP Dataverse in Power Apps

Connecting AI Agents to Enterprise Data

Prepared for EY Developers | BFSI Sector Delivery

Duration: 3 Hours (1 Hour Theory + 2 Hours Lab)

Status: Fully Teachable | Dataverse MCP Server is Generally Available (GA)

1. Module Overview

This module introduces the Dataverse MCP Server — the foundational technology that connects AI agents to enterprise data in the Power Platform. The Model Context Protocol (MCP) is an open protocol that allows AI models and agents to discover, understand, and operate on structured data sources. The Dataverse MCP Server implements this protocol for Microsoft Dataverse, enabling Copilot Studio agents to perform intelligent CRUD operations against Dataverse tables using natural language.

Unlike previous modules where you created data models (Module 02) and applications (Module 03), this module focuses on making that data accessible to AI agents. You will import curated BFSI sample data from Excel files into Dataverse, enable the MCP Server, connect it to a Copilot Studio agent, and then interact with the data using natural language — all without writing a single line of code.

Why This Module Matters

The Dataverse MCP Server is the technical core of the entire agentic architecture taught in this course. Every agent you build in Modules 05–09 will use MCP to interact with Dataverse. Every table you created in Modules 02–03 becomes accessible through MCP. Understanding MCP is not optional — it is the protocol that turns Dataverse from a passive database into an active agent platform.

For BFSI organisations, MCP enables agents that can read customer records, create claims, update policy statuses, and search across transaction histories — all governed by the same security model that protects the data in production.

1.1 Learning Objectives

By the end of this module, participants will be able to:

1. Explain the Model Context Protocol (MCP) and its role in connecting AI agents to enterprise data sources.
2. Describe the Dataverse MCP Server architecture, including its built-in tools and security model.
3. List and explain each of the seven built-in MCP tools: list_tables, describe_table, read_query, create_record, update_record, search_knowledge, and prompt execution.
4. Import curated BFSI sample data from Excel files into Dataverse tables with correct column types and relationships.
5. Enable and configure the Dataverse MCP Server in a Power Platform environment.
6. Connect the Dataverse MCP Server as a tool inside a Copilot Studio agent.

7. Execute CRUD operations against Dataverse tables through a Copilot Studio agent using natural language.
8. Understand how schema-driven reasoning works: how agents use describe_table to understand data structure before querying.
9. Identify licensing requirements, security boundaries, and access control for the Dataverse MCP Server.
10. Apply MCP Dataverse to a BFSI-relevant scenario using realistic sample data.

1.2 Module Structure

Section	Topic	Duration
Part A	Theory: MCP Protocol & Dataverse MCP Server Deep Dive	1 Hour
Part B	Lab Exercise 1: Data Preparation – Import Excel Data into Dataverse	30 Minutes
Part B	Lab Exercise 2: Enable MCP Server & Connect to Copilot Studio	30 Minutes
Part B	Lab Exercise 3: MCP Tool Exploration via Copilot Studio Agent	30 Minutes
Part B	Lab Exercise 4: BFSI Data Operations & Schema-Driven Reasoning	20 Minutes
Part B	Lab Exercise 5: Reflection & Discussion	10 Minutes

1.3 Prerequisites

- Completion of Modules 01–03 (or equivalent familiarity with Power Platform, Plan Designer, and Vibe).
- Power Platform developer environment with a Dataverse database provisioned.
- Power Apps Premium license (trial acceptable).
- Copilot Studio license (trial acceptable).
- Copilot features and preview features enabled in the tenant by an admin.
- Modern web browser (Microsoft Edge or Google Chrome, latest version).
- The four Excel sample data files provided with this module (see Section 2.9).

2. Theory: MCP Protocol & Dataverse MCP Server

This section covers the conceptual foundations of the Model Context Protocol and the detailed architecture, tools, and security model of the Dataverse MCP Server. Estimated duration: 1 hour.

2.1 What Is the Model Context Protocol (MCP)?

The Model Context Protocol (MCP) is an open standard that defines how AI models and agents discover, understand, and interact with external data sources and tools. Think of MCP as a universal adapter between AI and enterprise data — it provides a consistent interface that any AI client can use to work with any MCP-compatible data source.

Before MCP, connecting an AI agent to a data source required custom integration code for every agent-data source combination. Each AI platform had its own approach to tool use and function calling. MCP standardises this by providing four core capabilities:

- **Agents can ask an MCP server "What tools do you offer?" and receive a structured description of available operations.**Tool Discovery:
- **Agents can ask "Describe this data source" and receive metadata about tables, columns, data types, and relationships.**Schema Understanding:
- **Agents can perform CRUD operations (Create, Read, Update, Delete) through well-defined tool interfaces.**Structured Operations:
- **MCP servers enforce the caller's security context, ensuring agents can only access data the user is authorised to see.**Security Context:

2.1.1 The MCP Architecture Pattern

MCP follows a client-server architecture. In this course, the MCP client is always a Copilot Studio agent, and the MCP server is the Dataverse MCP Server. The communication happens over HTTP/SSE (Server-Sent Events) using a JSON-based protocol.

Component	Role	In This Course
MCP Client	The AI agent or application that wants to interact with data. Sends requests to the MCP Server.	Copilot Studio agents (Modules 04–09)
MCP Server	The data-side service that exposes data operations as MCP tools. Receives requests and executes them against the data source.	Dataverse MCP Server (GA since November 2025)

Data Source	The underlying data store that the MCP Server wraps.	Microsoft Dataverse tables with BFSI sample data
MCP Protocol	The communication standard that defines how clients and servers exchange messages.	JSON-based protocol over HTTP/SSE

Instructor Tip: Draw the MCP architecture on a whiteboard: Copilot Studio Agent → MCP Protocol → Dataverse MCP Server → Dataverse Tables. This linear flow is the single most important diagram for this entire course.

2.2 The Dataverse MCP Server

The Dataverse MCP Server is Microsoft's implementation of the MCP protocol for Dataverse. It was announced at Build 2025 (June 2025) and became Generally Available at Ignite 2025 (November 2025). It is the centrepiece of Microsoft's strategy to position Dataverse as the agent platform for enterprise AI.

When a Copilot Studio agent connects to the Dataverse MCP Server, it gains the ability to perform the following operations against any Dataverse table the user has access to:

- Discover what tables exist in the environment.
- Understand the schema of any table (columns, data types, relationships, option sets).
- Query data using natural language that the server translates into FetchXML or OData queries.
- Create new records in any table.
- Update existing records.
- Search across knowledge sources (including Dataverse Search).
- Execute pre-defined prompts for common operations.

Key Architecture Principle: Schema-Driven Reasoning

The most important concept in this module is schema-driven reasoning. When an agent needs to answer a question about data, it does not blindly query — it first calls describe_table to understand the table structure, then formulates an appropriate read_query based on the schema it discovered. This two-step pattern (understand, then act) is fundamental to reliable agent-data interaction.

2.3 Built-In MCP Tools: Deep Dive

The Dataverse MCP Server provides seven built-in tools. Each tool serves a specific purpose in the agent-data interaction lifecycle. The following table summarises all seven tools. Detailed explanations follow.

Tool Name	Purpose	Returns
list_tables	Discover all accessible Dataverse tables in the environment.	Table logical names, display names, descriptions.
describe_table	Retrieve the complete schema of a specific table.	Columns, data types, option set values, relationships, required fields.
read_query	Query data from a table with filters and sorting.	Records matching the query criteria.
create_record	Create a new record in a Dataverse table.	ID of the newly created record.
update_record	Update an existing record by ID.	Confirmation of successful update.
search_knowledge	Semantic/keyword search across multiple tables.	Ranked search results with relevance scores.
Prompt Execution	Execute pre-defined prompt templates that combine data retrieval with formatting.	Formatted response combining data with template.

Tool 1: list_tables

The list_tables tool returns a catalogue of all Dataverse tables that the calling user has read access to. This is the agent's first step when it does not know which table contains the data the user is asking about. For example, when a user asks "What information do you have about my accounts?" the agent calls list_tables to discover available tables (Customers, Policies, Claims, Transactions) before deciding which table to query.

In the lab, when you ask the agent "What tables do you have access to?" you will see it return the four BFSI tables you imported from Excel: contoso_customers, contoso_policies, contoso_claims, and contoso_transactions.

Tool 2: describe_table

The describe_table tool returns the full schema of a specific Dataverse table. This includes every column's logical name, display name, data type (text, number, currency, choice, lookup, datetime), option set values, relationship targets, and whether the column is required. This is the "reasoning" step — the agent uses this information to formulate correct queries and valid create/update operations.

For example, before querying the Claims table, the agent calls describe_table and learns that “Status” is a choice column with values (Open, Under Review, Approved, Denied, Closed), “ClaimAmount” is a currency column, and “Policy” is a lookup to the Policies table. This prevents the agent from guessing column names or inventing filter values that do not exist.

Tool 3: read_query

The read_query tool executes a data query against a Dataverse table. The agent provides filter criteria based on the schema it learned from describe_table, and the MCP Server translates this into a FetchXML or OData query. Results are filtered by the calling user’s security role — the agent can only return records the user has permission to read.

In the lab, you will test queries such as “Show me all open claims with an amount above ₹50,000” and observe the agent translating your natural language into a structured query against the Claims table.

Tool 4: create_record

The create_record tool creates a new record in a Dataverse table. The agent provides column values based on the schema it learned from describe_table. Dataverse enforces all standard validation: required fields, data type constraints, duplicate detection rules, and business rules. In the lab, you will ask the agent to create a new claim record and verify it appears in the Dataverse table.

Tool 5: update_record

The update_record tool modifies an existing record. The agent specifies the record ID and the columns to update. The calling user must have write/update permission on the specific record and columns being modified. In the lab, you will ask the agent to update a claim’s status from “Open” to “Under Review” and verify the change.

Tool 6: search_knowledge

The search_knowledge tool performs a semantic or keyword search across Dataverse data. Unlike read_query, which targets a specific table with structured filters, search_knowledge casts a wider net across multiple tables. This is useful for fuzzy or broad questions such as “Find everything related to water damage claims” or “Search for all interactions with Contoso Industries.” Dataverse Search must be enabled in the environment for full functionality.

Tool 7: Prompt Execution

The prompt execution tool runs pre-defined prompt templates that combine data retrieval with formatted responses. Administrators can create prompts that pull data from specific tables and return it in a standardised format. For example, a “Customer Summary” prompt might pull from Customers, Policies, and Claims tables to generate a comprehensive overview. This tool is covered in greater depth in Module 09.

2.4 Schema-Driven Reasoning in Practice

The most important pattern to understand is how agents use MCP tools in sequence to answer questions intelligently. Here is a typical seven-step interaction flow that you will observe in the lab:

Step	Agent Action	MCP Tool Used
1	User asks: “How many high-value claims are pending?”	(User input)
2	Agent identifies that “claims” likely maps to a claims-related table.	(Reasoning)
3	Agent discovers available tables to confirm the correct table name.	list_tables
4	Agent retrieves the Claims table schema to understand columns, types, and option set values.	describe_table
5	Agent formulates a query: filter by Status = Open, Amount > threshold.	(Reasoning)
6	Agent executes the query against the Claims table.	read_query
7	Agent formats the results and responds to the user.	(Response)

This describe-then-query pattern is fundamental. It ensures the agent does not hallucinate column names, data types, or option set values. Instead, it grounds every query in the actual schema of the Dataverse tables. In the lab, you will be asked to observe this pattern in the Copilot Studio activity trace.

Instructor Tip: Walk through this 7-step flow on the whiteboard using the Claims data from Section 2.9. Ask participants: “What would happen if the agent skipped step 4 (describe_table)?” Answer: it might guess wrong column names, miss filter options, or return incorrect results. This is why schema-driven reasoning is non-negotiable for enterprise agents.

2.5 Security and Access Control

Security is a critical concern when giving AI agents access to enterprise data. The Dataverse MCP Server enforces security at multiple levels. For BFSI scenarios, this is non-negotiable — agents must never access data the user is not authorised to see.

Security Layer	How It Works
User Context	Every MCP operation executes in the security context of the calling user. The agent inherits the user's Dataverse security role, team memberships, and record-level permissions.
Table-Level Security	If a user's security role does not grant read access to a table, the MCP Server will not return that table in list_tables or allow queries against it.
Column-Level Security	Dataverse column security profiles are respected. If a column is restricted (e.g., PAN number, income), it will not appear in describe_table results for unauthorised users.
Record-Level Security	Business unit ownership, team ownership, and sharing rules all apply. read_query only returns records the user is authorised to see.
MCP Access Boundaries	Administrators can configure which tables and operations are exposed through MCP, providing an additional layer of control beyond standard Dataverse security.
Audit Logging	All MCP operations are captured in Dataverse audit logs (if auditing is enabled), providing a complete trail of agent-data interactions for compliance.

Important: For BFSI scenarios, always configure Dataverse security roles to enforce least-privilege access. Never give agents unrestricted access to all tables and columns. Module 12 covers governance and security in depth.

2.6 Licensing Requirements

Access Pattern	License Required
Copilot Studio agents accessing Dataverse MCP	Copilot Studio license (trial or paid). Standard Copilot Credits are consumed for agent operations.
External apps accessing Dataverse MCP	Power Platform Premium User SLs (Subscription Licenses) required for each user whose data is accessed.
Dataverse database provisioning	At least one Power Apps Premium or Dynamics 365 license in the tenant to provision a Dataverse database.

Instructor Tip: Licensing is one of the most frequently asked questions from BFSI clients. Provide the licensing table as a handout and refer participants to the community licensing guide for the latest December 2025 update.

2.7 BFSI Context: MCP Dataverse for Financial Services

For BFSI organisations, the Dataverse MCP Server enables several high-value agent scenarios. Each of the following scenarios uses the curated sample data you will import in the lab:

- **Uses list_tables and read_query to pull a comprehensive view of a customer's policies, claims, and transaction history from multiple Dataverse tables.**Customer 360 Agent:
- **Uses create_record to submit claims, read_query to check claim status, and update_record to record adjuster decisions.**Claims Processing Agent:
- **Uses describe_table to understand the Customers schema, read_query to retrieve verification status, and update_record to log verification results.**KYC Verification Agent:
- **Uses read_query with filters to flag unusual transactions (high amounts, unusual channels, frequent activity).**Transaction Monitoring Agent:
- **Uses read_query to identify policies nearing expiry, create_record to log renewal notices, and update_record to record renewal confirmations.**Policy Management Agent:

2.8 Connection to Other Modules

Module	Relationship to MCP Dataverse
Module 02: Plan Designer	Plan Designer creates Dataverse tables. Those tables are immediately accessible through the MCP Server.
Module 03: Power Apps Vibe	Vibe-generated apps create Dataverse tables. Those tables are accessible through MCP, enabling agents to work with the same data the app uses.
Module 05: Agent Builder	Agent Builder converts canvas apps into Copilot Studio agents. Those agents use the Dataverse MCP Server to access data.
Module 06–09: Agent Development	All agents built in these modules connect to Dataverse through MCP. The tools learned here are used extensively.
Module 12: Governance	MCP audit logging and security boundaries are configured as part of production governance.

2.9 Curated BFSI Sample Data

The following four tables contain the curated sample data that you will use throughout this module and in subsequent modules. The instructor will provide these as four separate Excel files. Review the data carefully before the lab — familiarity with the data will help you formulate effective natural language queries.

Important: The instructor should prepare these four Excel files before class. Each file should have a single worksheet with the column headers as the first row and data rows below. Save each file as .xlsx format. The file names should match exactly: Contoso_Customers.xlsx, Contoso_Policies.xlsx, Contoso_Claims.xlsx, Contoso_Transactions.xlsx.

Table 1: Contoso_Customers.xlsx

This table represents the customer master data for Contoso Financial Services. It includes retail, corporate, and high-net-worth individual (HNI) customers with varying KYC statuses and risk ratings.

CustomerID	FullName	Email	Phone	City	KYCStatus	RiskRating	Segment	AcctOpenDate
CUST-001	Rajesh Sharma	rajesh.sharma@email.com	+91 98765 43210	Mumbai	Verified	Low	Retail	2020-01-15
CUST-002	Priya Patel	priya.patel@email.com	+91 98765 43211	Kolkata	Verified	Low	Retail	2019-06-20
CUST-003	Contoso Industries Ltd	finance@contoso.in	+91 22 4567 8900	Mumbai	Verified	Medium	Corporate	2018-04-01
CUST-004	Amit Kumar	amit.kumar@email.com	+91 98765 43212	Bangalore	Pending	High	HNI	2021-03-10
CUST-005	Sunita Reddy	sunita.reddy@email.com	+91 98765 43213	Hyderabad	Verified	Low	Retail	2022-08-15
CUST-006	Vikram Singh	vikram.singh@email.com	+91 98765 43214	Delhi	Verified	Medium	HNI	2020-11-05
CUST-007	Ananya Desai	ananya.desai@email.com	+91 98765 43215	Pune	Expired	Low	Retail	2017-02-28
CUST-008	TechVentures Pvt Ltd	accounts@techventures.in	+91 80 2345 6789	Bangalore	Verified	Low	Corporate	2021-07-12
CUST-009	Mohammed Farooqui	m.farooqui@email.com	+91 98765 43216	Chennai	Verified	Medium	Retail	2023-01-20
CUST-010	Lakshmi Iyer	lakshmi.iyer@email.com	+91 98765 43217	Mumbai	Pending	High	HNI	2024-03-01

CUST-011	Global Exports Ltd	info@globalexports.in	+91 44 5678 1234	Chennai	Verified	Medium	Corporate	2019-09-15
CUST-012	Deepak Mehta	deepak.mehta@email.com	+91 98765 43218	Ahmedabad	Verified	Low	Retail	2022-05-10
CUST-013	Kavita Nair	kavita.nair@email.com	+91 98765 43219	Kochi	Verified	Low	Retail	2023-06-18
CUST-014	Arjun Reddy	arjun.reddy@email.com	+91 98765 43220	Hyderabad	Verified	Medium	HNI	2020-12-01
CUST-015	Pinnacle Finance Corp	contact@pinnaclefin.in	+91 22 6789 0123	Mumbai	Verified	High	Corporate	2017-08-20

Column specifications for Dataverse import: CustomerID (Single line of text, max 20), FullName (Single line of text, max 100), Email (Single line of text, Email format), Phone (Single line of text, Phone format), City (Single line of text, max 50), KYCStatus (Choice: Verified | Pending | Expired), RiskRating (Choice: Low | Medium | High), Segment (Choice: Retail | Corporate | HNI), AccountOpenDate (Date Only).

Table 2: Contoso_Policies.xlsx

This table represents insurance and financial product policies held by customers. Each policy is linked to a customer via the CustomerID column.

PolicyID	CustomerID	PolicyType	PremiumAmt	CoverageAmt	StartDate	EndDate	Status
POL-001	CUST-001	Term Life	12000	2500000	2022-01-15	2032-01-14	Active
POL-002	CUST-001	Health	18000	500000	2023-04-01	2024-03-31	Expired
POL-003	CUST-002	Motor	8500	750000	2024-01-10	2025-01-09	Active
POL-004	CUST-003	Commercial Property	150000	25000000	2023-07-01	2024-06-30	Active
POL-005	CUST-004	Term Life	45000	10000000	2021-06-15	2031-06-14	Active
POL-006	CUST-004	Health	35000	2000000	2024-01-01	2024-12-31	Active
POL-007	CUST-005	Home	6500	3000000	2023-09-01	2024-08-31	Lapsed
POL-008	CUST-006	Term Life	60000	15000000	2020-12-01	2040-11-30	Active
POL-009	CUST-006	Motor	22000	2500000	2024-06-01	2025-05-31	Active
POL-010	CUST-007	Health	9000	300000	2022-03-01	2023-02-28	Expired
POL-011	CUST-008	Commercial Property	95000	15000000	2024-01-15	2025-01-14	Active
POL-012	CUST-009	Motor	7500	600000	2023-11-01	2024-10-31	Active
POL-013	CUST-010	Term Life	85000	20000000	2024-04-01	2044-03-31	Active
POL-014	CUST-011	Marine Cargo	120000	30000000	2023-06-01	2024-05-31	Active
POL-015	CUST-012	Health	11000	500000	2024-02-01	2025-01-31	Active
POL-016	CUST-013	Home	5500	2000000	2024-07-01	2025-06-30	Active
POL-017	CUST-014	Health	28000	1500000	2024-01-01	2024-12-31	Active
POL-018	CUST-014	Motor	15000	1800000	2024-03-15	2025-03-14	Active
POL-019	CUST-015	Directors & Officers	200000	50000000	2023-10-01	2024-09-30	Active

POL-020	CUST-003	Cyber Liability	85000	10000000	2024-01-01	2024-12-31	Active

Column specifications for Dataverse import: PolicyID (Single line of text, max 20), CustomerID (Single line of text, max 20 — used for lookup matching), PolicyType (Choice: Term Life | Health | Motor | Home | Commercial Property | Marine Cargo | Directors & Officers | Cyber Liability), PremiumAmount (Currency, ₹), CoverageAmount (Currency, ₹), StartDate (Date Only), EndDate (Date Only), Status (Choice: Active | Expired | Lapsed | Cancelled).

Table 3: Contoso_Claims.xlsx

This table represents insurance claims filed by customers against their policies. The data includes a mix of statuses to support filtering and reporting exercises in the lab.

ClaimID	PolicyID	CustomerID	ClaimType	ClaimAmount	DateFiled	Status	AdjusterName	Description
CLM-001	POL-002	CUST-001	Hospitalisation	125000	2023-08-15	Approved	Meera Joshi	Emergency appendectomy at Apollo Hospital
CLM-002	POL-003	CUST-002	Accident	185000	2024-03-22	Under Review	Rahul Verma	Rear-end collision on EM Bypass, Kolkata
CLM-003	POL-004	CUST-003	Water Damage	750000	2024-01-10	Approved	Suresh Pillai	Pipe burst in server room, 3rd floor
CLM-004	POL-005	CUST-004	Death Benefit	10000000	2024-06-01	Open	Neha Gupta	Nominee claim filing — documentation pending
CLM-005	POL-006	CUST-004	Surgery	350000	2024-02-18	Approved	Meera Joshi	Knee replacement surgery at Fortis
CLM-006	POL-007	CUST-005	Fire Damage	480000	2024-04-05	Denied	Rahul Verma	Kitchen fire — policy lapsed before incident date
CLM-007	POL-009	CUST-006	Theft	1200000	2024-07-20	Under Review	Suresh Pillai	Vehicle stolen from residential parking
CLM-008	POL-011	CUST-008	Equipment Damage	320000	2024-08-12	Open	Neha Gupta	Server rack damage due to power surge
CLM-009	POL-012	CUST-009	Accident	95000	2024-05-30	Approved	Rahul Verma	Minor collision, windshield and bumper damage
CLM-010	POL-014	CUST-011	Cargo Loss	2800000	2024-09-01	Under Review	Suresh Pillai	Container lost during transit — Vizag port
CLM-011	POL-015	CUST-012	Hospitalisation	65000	2024-06-22	Approved	Meera Joshi	Dengue treatment at City Hospital Ahmedabad
CLM-012	POL-017	CUST-014	Surgery	420000	2024-10-15	Open	Neha Gupta	Cardiac stent procedure at KIMS Hyderabad

CLM-013	POL-019	CUST-015	Regulatory Fine	5000000	2024-11-01	Under Review	Suresh Pillai	SEBI fine coverage — legal review pending
CLM-014	POL-020	CUST-003	Cyber Breach	1500000	2024-08-25	Open	Neha Gupta	Ransomware attack — data recovery costs
CLM-015	POL-016	CUST-013	Water Damage	180000	2024-09-15	Approved	Rahul Verma	Monsoon flooding in ground floor rooms

Column specifications for Dataverse import: ClaimID (Single line of text, max 20), PolicyID (Single line of text, max 20 — for lookup matching), CustomerID (Single line of text, max 20 — for lookup matching), ClaimType (Choice: Hospitalisation | Surgery | Accident | Theft | Fire Damage | Water Damage | Equipment Damage | Cargo Loss | Death Benefit | Regulatory Fine | Cyber Breach), ClaimAmount (Currency, ₹), DateFiled (Date Only), Status (Choice: Open | Under Review | Approved | Denied | Closed), AdjusterName (Single line of text, max 100), Description (Multiple lines of text).

Table 4: Contoso_Transactions.xlsx

This table represents financial transactions processed through Contoso's systems. It includes premium payments, claim payouts, policy renewals, and refunds across various channels.

TransactionID	CustomerID	Type	Amount	Date	Reference	Channel	Status
TXN-001	CUST-001	Premium Payment	12000	2024-01-15	POL-001	Net Banking	Completed
TXN-002	CUST-002	Premium Payment	8500	2024-01-10	POL-003	UPI	Completed
TXN-003	CUST-003	Premium Payment	150000	2024-07-01	POL-004	NEFT	Completed
TXN-004	CUST-001	Claim Payout	125000	2023-09-20	CLM-001	NEFT	Completed
TXN-005	CUST-004	Premium Payment	45000	2024-06-15	POL-005	Net Banking	Completed
TXN-006	CUST-004	Premium Payment	35000	2024-01-01	POL-006	UPI	Completed
TXN-007	CUST-003	Claim Payout	750000	2024-03-15	CLM-003	RTGS	Completed
TXN-008	CUST-005	Premium Payment	6500	2023-09-01	POL-007	UPI	Failed
TXN-009	CUST-006	Premium Payment	60000	2024-12-01	POL-008	Net Banking	Completed
TXN-010	CUST-006	Premium Payment	22000	2024-06-01	POL-009	Auto Debit	Completed
TXN-011	CUST-008	Premium Payment	95000	2024-01-15	POL-011	NEFT	Completed
TXN-012	CUST-009	Premium Payment	7500	2023-11-01	POL-012	UPI	Completed
TXN-013	CUST-009	Claim Payout	95000	2024-07-10	CLM-009	NEFT	Completed
TXN-014	CUST-010	Premium Payment	85000	2024-04-01	POL-013	RTGS	Completed
TXN-015	CUST-011	Premium Payment	120000	2024-06-01	POL-014	NEFT	Completed

TXN-016	CUST-012	Premium Payment	11000	2024-02-01	POL-015	UPI	Completed
TXN-017	CUST-012	Claim Payout	65000	2024-07-30	CLM-011	NEFT	Completed
TXN-018	CUST-013	Premium Payment	5500	2024-07-01	POL-016	Auto Debit	Completed
TXN-019	CUST-014	Premium Payment	28000	2024-01-01	POL-017	Net Banking	Completed
TXN-020	CUST-014	Premium Payment	15000	2024-03-15	POL-018	UPI	Completed
TXN-021	CUST-015	Premium Payment	200000	2023-10-01	POL-019	RTGS	Completed
TXN-022	CUST-003	Premium Payment	85000	2024-01-01	POL-020	NEFT	Completed
TXN-023	CUST-005	Refund	6500	2024-01-15	POL-007	NEFT	Completed
TXN-024	CUST-007	Premium Payment	9000	2022-03-01	POL-010	UPI	Completed
TXN-025	CUST-013	Claim Payout	180000	2024-10-20	CLM-015	NEFT	Pending

Column specifications for Dataverse import: TransactionID (Single line of text, max 20), CustomerID (Single line of text, max 20 — for lookup matching), TransactionType (Choice: Premium Payment | Claim Payout | Refund | Policy Renewal), Amount (Currency, ₹), TransactionDate (Date Only), Reference (Single line of text, max 20), Channel (Choice: UPI | Net Banking | NEFT | RTGS | Auto Debit | Cash | Cheque), Status (Choice: Completed | Pending | Failed | Reversed).

Data Relationships

The four tables are related through the CustomerID and PolicyID columns:

- Customers (1) → (Many) Policies: Each customer can have multiple policies.
- Customers (1) → (Many) Transactions: Each customer can have multiple transactions.
- Policies (1) → (Many) Claims: Each policy can have multiple claims.
- In the lab, you will create these as text-based matching columns initially. The Dataverse lookup relationships will be configured during the import exercise.

3. Module Summary

3.1 Key Takeaways

- The Model Context Protocol (MCP) is an open standard that connects AI agents to enterprise data sources through a universal, structured interface.
- The Dataverse MCP Server is Generally Available (GA) since Ignite 2025 (November 2025) and is the technical foundation of the agentic architecture in Power Platform.
- Seven built-in tools (list_tables, describe_table, read_query, create_record, update_record, search_knowledge, prompt execution) cover the full lifecycle of agent-data interaction.
- Schema-driven reasoning (describe_table before read_query) is the fundamental pattern that ensures agents formulate correct queries grounded in actual data schemas.
- Copilot Studio agents connect to the Dataverse MCP Server through the Tools tab, gaining natural language access to all Dataverse tables the user has permission to access.
- Security is enforced at every level: user context, table, column, record, and MCP access boundaries. Agents can only access data the user is authorised to see.
- BFSI sample data (Customers, Policies, Claims, Transactions) imported in this module will be reused throughout Modules 05–09.
- For BFSI scenarios, MCP Dataverse enables high-value use cases: Customer 360 views, claims processing, transaction monitoring, and compliance reporting.

3.2 Knowledge Check Questions

1. What is the Model Context Protocol (MCP) and why was it created?
2. Name all seven built-in tools provided by the Dataverse MCP Server.
3. Explain the schema-driven reasoning pattern. Why does an agent call describe_table before read_query?
4. What security layers does the Dataverse MCP Server enforce? Name at least four.
5. Describe the steps to add the Dataverse MCP Server as a tool in a Copilot Studio agent.
6. What is the difference between read_query and search_knowledge? When would you use each?
7. What licensing is required for a Copilot Studio agent to access the Dataverse MCP Server?
8. How do you import Excel data into Dataverse? Describe the key steps.
9. Describe one BFSI scenario where create_record and update_record would be used together in sequence.

10. Why is Dataverse positioned as the “agent platform” for enterprise AI?

3.3 What’s Next

In Module 05: Agent Builder (From App to Agent), you will learn how to convert existing canvas apps into Copilot Studio agents using the Agent Builder feature. Agent Builder analyses app metadata — screens, formulas, data connections, and navigation — to extract steps, rules, knowledge, and triggers that bootstrap agent creation. The agents it creates use the Dataverse MCP Server you configured in this module to interact with the same BFSI data.

The four Dataverse tables you imported today (Contoso Customers, Policies, Claims, Transactions) will continue to serve as the data foundation for all remaining modules. Do not delete these tables from your environment.

4. References and Resources

4.1 Priority 1 Sources (Primary)

- Microsoft Learn: “Dataverse MCP Server Documentation” — learn.microsoft.com (Dataverse MCP Server docs) — Official documentation covering tools list, configuration, security, and access patterns.
- Microsoft Blog: “Dataverse MCP Server: A Game Changer for AI-Driven Workflows” — microsoft.com/en-us/power-platform/blog/2025/07/07/dataverse-mcp/ — Architecture deep-dive, tool descriptions, and Copilot Studio integration.

4.2 Priority 2 Sources (Supplementary)

- Microsoft Learn: “Dataverse 2025 Release Wave 2 Overview” — learn.microsoft.com/en-us/power-platform/release-plan/2025wave2/data-platform/ — Feature roadmap and GA status confirmation for Dataverse MCP Server.

4.3 Priority 3 Sources (Additional)

- MSDynamicsWorld: “Ignite 2025: Microsoft Advances MCP Servers” — msdynamicsworld.com/story/ignite-2025-microsoft-advances-mcp-servers — Analyst perspective on Dataverse MCP GA and enterprise adoption.
- Licensing Guide: “Dataverse MCP Server Licensing Requirements” — licensing.guide/dataverse-mcp-server-licensing-requirements/ — December 2025 update covering per-user and per-agent licensing models.

4.4 Related Course Modules

- Module 02: Plan Designer — creates Dataverse tables that MCP exposes.
- Module 03: Power Apps Vibe — creates Dataverse tables accessible through MCP.
- Module 05: Agent Builder — next module; converts apps to agents that use MCP.
- Modules 06–09: Agent development modules that extensively use MCP Dataverse.
- Module 12: Governance — MCP security and audit configuration for production.