

Week 3: MCP Integration - Connecting to External Services

01. Session Goals

- Understand MCP (Model Context Protocol) architecture
 - Configure MCP servers for various services
 - Learn context management for data-heavy applications
 - Connect Claude to external tools and data sources
-

02. Block 1: Theory - What is MCP? (30 min)

The Problem MCP Solves

Before MCP, connecting AI agents to external services required:

- Custom API integrations for each service
- OAuth flows and token management
- Tool definitions for every endpoint
- Maintenance as APIs change

MCP provides a universal protocol - one standard way to connect Claude to any external service.

The Story Behind MCP

MCP started as an internal project at Anthropic. Engineers were building the same integration patterns over and over, connecting Claude to different tools and data sources. Every team reinvented the wheel.

At an internal hackathon, every single entry was built on an early version of MCP. It went viral inside the company. That's when Anthropic knew they had something worth sharing.

November 2024: Anthropic open-sourced MCP with SDKs for Python and TypeScript.

March 2025: OpenAI adopted MCP across their Agents SDK, Responses API, and ChatGPT desktop. Sam Altman posted: "People love MCP and we are excited to add support across our products."

April 2025: Google DeepMind confirmed MCP support in upcoming Gemini models.

December 2025: Anthropic donated MCP to the Linux Foundation, establishing the Agentic AI Foundation.
Co-founders: Anthropic, Block, and OpenAI. Supporters: Google, Microsoft, AWS, Cloudflare, Bloomberg.

Why this matters:

MCP went from internal experiment to industry standard in one year. All major AI labs now back it. When you learn MCP, you're learning infrastructure that works across Claude, ChatGPT, Gemini, and Microsoft Copilot.

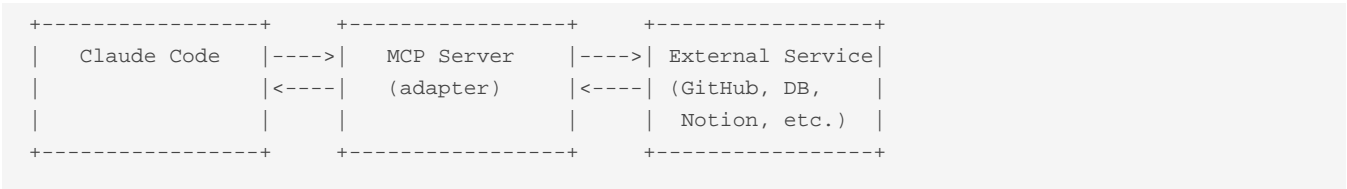
The protocol re-uses ideas from the Language Server Protocol (LSP) that powers code editors. If you've used VS Code's autocomplete for multiple languages, you've benefited from a similar standardization effort. MCP does the same for AI tool integrations.

Current scale: Over 10,000 public MCP servers. 97 million monthly SDK downloads. Enterprise deployment support from AWS, Google Cloud, and Azure.

References:

- Anthropic: Introducing the Model Context Protocol
(<https://www.anthropic.com/news/model-context-protocol>)
- Anthropic: Donating MCP to the Agentic AI Foundation
(<https://www.anthropic.com/news/donating-the-model-context-protocol-and-establishing-of-the-agentic-ai-foundation>)
- GitHub Blog: MCP Joins the Linux Foundation
(<https://github.blog/open-source/maintainers/mcp-joins-the-linux-foundation-what-this-means-for-developers-building-the>)

MCP Architecture



MCP Server: A lightweight adapter that:

- Exposes tools Claude can use
- Handles authentication
- Translates requests to service-specific APIs

Transport Types

Transport	Use Case	Command
HTTP	Cloud-hosted services	<code>`claude mcp add --transport http`</code>
SSE	Real-time services (deprecated)	<code>`claude mcp add --transport sse`</code>
stdio	Local tools, CLI wrappers	<code>`claude mcp add --transport stdio`</code>

Scopes

Scope	Location	Shared?	Use Case
local	`.claude.json` (project path)	No	Personal, sensitive credentials
project	`.mcp.json` in repo	Yes (git)	Team-shared services
user	`.claude.json` (global)	No	Personal tools across projects

Demo

Live demo: Add a GitHub MCP server and query issues.

```
claude mcp add --transport http github https://api.githubcopilot.com/mcp/
```

03. Block 2: Theory - Context Management for Data (30 min)

The Problem

When working with data, context management becomes critical. A single query can return massive amounts of data, instantly filling your context window.

The difference:

- Reading a 500-line file? Predictable, bounded.
- `SELECT * FROM users`? Could return 2 million rows.

Without careful management, one careless query can:

- Fill your entire context window
- Cause Claude to lose track of the conversation
- Waste tokens on data that doesn't fit

The Solution: Query Strategically

When working with databases or large datasets, follow these rules:

1. Always use LIMIT for exploration

```
-- Always add a limit for exploration queries
SELECT * FROM funding_rounds LIMIT 100;
```

2. Aggregate first, then drill down

```
-- Start with aggregates to understand the data
SELECT industry, COUNT(*) as count
FROM startups
GROUP BY industry
ORDER BY count DESC;

-- Then drill into specifics
SELECT * FROM startups WHERE industry = 'AI/ML' LIMIT 50;
```

3. Be specific about what you need

```
-- Bad: grab everything
SELECT * FROM funding_rounds;

-- Good: specific columns, filtered, limited
SELECT name, stage, amount_usd
FROM funding_rounds fr
JOIN startups s ON fr.startup_id = s.id
WHERE fr.stage = 'Series A'
ORDER BY fr.amount_usd DESC
LIMIT 20;
```

4. Track what you've seen

When analyzing large datasets, keep mental notes:

- How many total rows exist
- What subset you've explored
- What's left to investigate

The Data Analysis Loop + Context Management

Remember the Data Analysis Loop from Week 2? Context management fits into each phase:

Phase	Context Management Strategy
Monitor	Run aggregation queries first (safe, bounded)
Explore	Drill down with LIMIT, explore segments one at a time
Craft	Work with summarized insights, not raw data
Impact	Present recommendations, not data dumps

Demo

Show what happens with good vs. careless queries:

```
> How many funding rounds are in the database?  
(Safe - returns a single number)  
  
> Show me all funding rounds  
(Dangerous - could return thousands of rows!)  
  
> Show me the top 20 largest Series A rounds  
(Good - specific, limited, useful)
```

04. BREAK (10 min)

05. Block 3: Lab 1 - Connect External Services (45 min)

Task: Build Your MCP Integration Stack

Connect Claude to external services you'll use in your projects.

Option A: Connect GitHub (15 min)

```
claude mcp add --transport http github https://api.githubcopilot.com/mcp/
```

After OAuth flow completes:

```
> List my open pull requests  
  
> Show me issues labeled "bug" in this repo  
  
> What commits happened in the last week?
```

Useful for:

- Code review workflows
- Issue triage
- PR summaries

Option B: Connect Notion (15 min)

```
claude mcp add --transport http notion https://mcp.notion.com/mcp
```

After OAuth:

```
> Search for databases in my workspace

> Find pages mentioning "project plan"

> Show me the structure of my team wiki
```

Useful for:

- Document research
- Knowledge base queries
- Task management

Option C: Connect a Database (15 min)

If you have a PostgreSQL or MySQL database:

```
# PostgreSQL
claude mcp add --transport stdio db -- npx -y @bytebase/dbhub \
  --dsn "postgresql://user:pass@host:5432/database"

# MySQL
claude mcp add --transport stdio db -- npx -y @bytebase/dbhub \
  --dsn "mysql://user:pass@host:3306/database"
```

Test:

```
> What tables are in the database?

> Show me the schema for the users table

> How many records are in each table?
```

Verify Your Connections

Check what MCPs are connected:

```
/mcp
```

You should see your added servers listed.

Success Criteria

- [] At least 1 MCP server connected
- [] Successfully queried the external service
- [] Understand what each MCP enables

06. Block 4: Lab 2 - Combined Workflow (30 min)

Task: Multi-Source Analysis

Now combine the startup funding database with external services.

Exercise 1: Funding + Web Research

Using both the local database and web search:

```
> Find AI coding tool companies in the funding database.
> Then research each company's current product and recent news.
> Create a comparison table with: Company, Total Funding, Latest Stage, Product Focus, Recent News
```

Exercise 2: Funding + GitHub (if connected)

```
> Which AI coding tools in our database have public GitHub repos?
> Compare their GitHub stars to their funding amounts.
> Is there a correlation between community popularity and funding?
```

Exercise 3: Data-Driven Research

```
> I want to understand the Series A landscape for AI startups.
> 1. Query the database for all AI/ML Series A rounds in 2024
> 2. Calculate the median round size and time from founding to Series A
> 3. Research 2-3 of these companies online to understand what made them fundable
> 4. Summarize patterns you see
```

Deliverable

Document your combined workflow showing:

1. Screenshot of `/mcp` with connected servers
2. A multi-step analysis that used at least 2 sources (database + web, or database + GitHub, etc.)
3. Your synthesized findings

07. Wrap-Up (15 min)

Key Takeaways

1. MCP = Universal Protocol - One way to connect to any service
2. Three transports - HTTP (cloud), stdio (local), SSE (deprecated)
3. Three scopes - local, project, user (precedence matters)
4. Context management is critical - Aggregate first, limit always, drill down carefully
5. Combine sources - Real analysis often needs data + context from multiple places

Homework

Part 1: Expand Your MCP Stack

Add at least 1 more MCP server relevant to your project:

Project Domain	Useful MCP Servers
GTM/Sales	CRM (HubSpot, Salesforce), Email (Gmail), Notion
Developer Tools	GitHub, PostgreSQL, Sentry, Linear
Content/Marketing	Notion, Google Docs, social APIs
Customer Support	Zendesk, Intercom, Slack
Operations	Google Sheets, Airtable, databases

Part 2: Document Your Stack

Create a brief doc with:

- Which servers you added and why
- How you configured them (transport, scope)
- 3 useful queries you can now run

Part 3: Multi-Source Analysis

Do one analysis that combines:

- The startup funding database (local queries)
- At least one external MCP (GitHub, Notion, web research, etc.)

Document your findings and the workflow you used.

Next Week Preview

Week 4: Agent Skills - Teaching Claude new capabilities with SKILL.md files

- Encode domain expertise as reusable skills
- Create a data analysis skill with guardrails
- Progressive disclosure and skill organization

08. Facilitator Notes

Common Issues

1. OAuth failures: Ensure browser allows popups, try incognito
2. Connection timeouts: Increase with `MCP_TIMEOUT=10000 claude`
3. "No such tool" errors: MCP server may have disconnected, re-add it
4. Database connection issues: Check DSN format, firewall rules

API Key Options for Participants

For live services, participants can:

- Use free tiers (Notion, GitHub have free plans)
- Work with the local funding database (no accounts needed)
- Pair up if some have accounts

Timing Adjustments

- If OAuth issues: Focus on GitHub (most reliable) or skip to database queries
- If ahead: Have participants try multiple MCPs
- Lab 2 can extend into homework if needed

Discussion Prompts

- "What external data would make your project more powerful?"
- "How do you think about combining structured data with unstructured context?"
- "What context management mistakes have you made in the past?"

Key Concepts to Emphasize

1. MCP unlocks external context - But the pattern is the same: query, limit, iterate
2. Local data + external context = insight - Numbers alone don't tell the story
3. Context management applies everywhere - Not just databases, any large data source