

Week 2: Tool Calling - How Claude Takes Action

01. Session Goals

- Understand how Claude uses tools to interact with the world
- Learn the tool calling loop: think, select, execute, observe
- Practice with built-in tools for data analysis and research
- Master the Data Analysis Loop: Monitor, Explore, Craft, Impact
- Build analysis workflows using SQL and web search

02. Block 1: Theory - How Tool Calling Works (30 min)

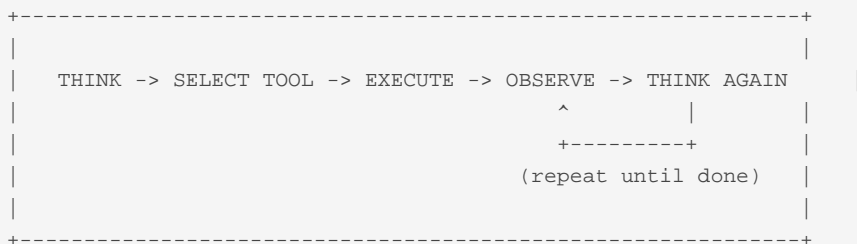
The Key Insight

Last week we learned that Claude Code bets on filesystems and bash. This week we go deeper: how does tool calling actually work?

When you give Claude access to tools, it stops being a chatbot and becomes an agent. It can read files, search the web, and execute commands. But the most important tool is one you might not expect: bash.

The Tool Calling Loop

Every time Claude uses a tool, it follows this loop:



Example: "Find the top-funded AI startups in 2024"

1. Think: I need to query the funding database
2. Select: Use `Bash` with sqlite3
3. Execute: Run the query
4. Observe: I see 50 results with columns: startup_name, amount, stage...
5. Think: Now I need to filter for AI industry and sort by amount

6. Select: Refine the SQL query
7. Execute: Run the updated query
8. Observe: Here are the results...
9. Think: Task complete, present findings

Why Bash Is the Most Important Tool

Remember Vercel's lesson from Week 1: they removed 80% of their agent's tools and got better results. What did they keep? Bash.

Here's what Vercel removed from their text-to-SQL agent:

- Schema lookup tool
- Query validation tool
- Error recovery tool
- Entity join finder
- Context recall mechanism
- Results formatter
- Data visualization tool
- ...and 5 more specialized tools

What replaced all of them? An agent that can execute bash commands and explore files.

Why this works:

LLMs have seen `grep`, `cat`, `find`, and `ls` billions of times during training. These commands are native operations, not bolted-on behaviors. When you give Claude bash access, you're not teaching it something new. You're letting it use skills it already has.

How agents use bash:

Command	What It Does	Agent Use Case
<code>`ls`</code>	List files	"What data do I have to work with?"
<code>`cat`</code>	Read file contents	"Let me look at this file"
<code>`grep`</code>	Search for patterns	"Find all mentions of 'Series A'"
<code>`find`</code>	Locate files	"Where are the funding reports?"
<code>`head/tail`</code>	Preview files	"Show me the first 10 rows"
<code>`wc`</code>	Count lines/words	"How many records are in this file?"
<code>`sort`</code>	Order data	"Sort by funding amount descending"
<code>`sqlite3`</code>	Query databases	"Run SQL on the funding database"

On-demand context retrieval:

Instead of stuffing everything into the prompt upfront, agents retrieve context as they need it:

```
Agent receives task
-> Explores filesystem (ls, find)
-> Searches for relevant content (grep)
-> Reads specific files (cat)
-> Queries databases (sqlite3)
-> Sends only what's needed to the model
-> Returns structured output
```

This is why Vercel's agent dropped from \$1.00 to \$0.25 per sales call analysis. It loads only what it needs, when it needs it.

The philosophy:

> "Every agent needs filesystem and bash. If you're building an agent, resist the urge to create custom tools. Instead, ask: can I represent this as files?"

Built-in Tools

Claude Code comes with these tools ready to use:

Tool	What It Does	Business Use Cases
Read	Read files	Analyze data, read reports, review documents
Write	Create files	Generate reports, save analysis results
Edit	Modify files	Update data, fix errors in documents
Bash	Run commands	Execute scripts, run SQL, process data
Glob	Find files	Locate reports, find datasets
Grep	Search content	Find mentions, search across files
WebSearch	Search the web	Research companies, find market data
WebFetch	Fetch web pages	Get company info, pull public data

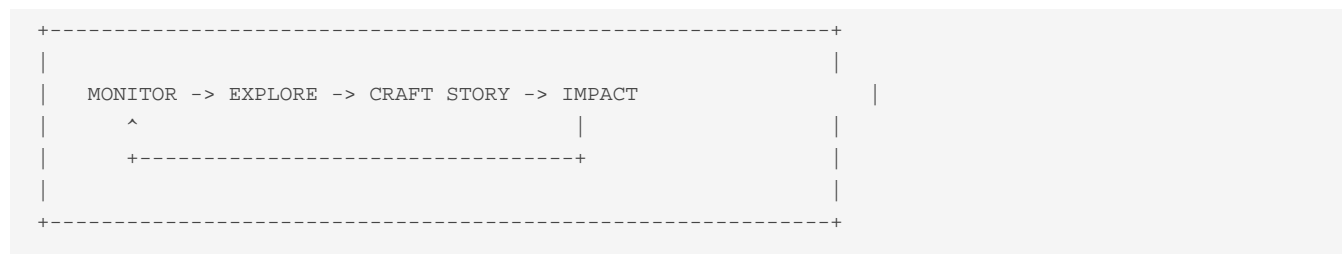
How Claude Decides Which Tool to Use

Claude reads your request and matches it to available tools:

Your Request	Claude's Thinking	Tool Selected
"What's in this database?"	Need to explore schema	Bash (sqlite3)
"Find all AI startups"	Need to query database	Bash (sqlite3)
"Research Cursor AI funding"	Need current web info	WebSearch
"Create a summary report"	Need to write output	Write
"How many Series A rounds in 2024?"	Need to count from database	Bash (sqlite3)
"Top investors by portfolio size"	Need to aggregate and rank	Bash (sqlite3)

The Data Analysis Loop

Professional data analysis follows a repeatable workflow. This is what separates good analysts from great ones:



1. Monitor

- Run recurring queries to check key metrics
- Compare current values to historical baselines
- Flag anomalies (>10% deviation from average)

2. Explore

- When you spot an anomaly, dig deeper
- Segment the data: by time, category, cohort
- Look for external context: was there a market shift? A news event?

3. Craft Story

- Synthesize findings into 3-5 key insights
- Support each insight with specific data
- Note confidence level and caveats

4. Impact

- Recommend concrete next actions
- Size the opportunity if possible
- Identify what additional data would help

Claude can help at every phase of this loop.

AI-Assisted SQL and Data Warehouse Querying

One of the most powerful applications of AI agents is querying structured data. Whether you're working with a SQL database, data warehouse, or CSV files, Claude can translate your business questions into precise queries.

From natural language to SQL:

Business Question	Claude Generates
"Top 10 funded AI startups"	<code>`SELECT name, SUM(amount_usd) FROM startups s JOIN funding_rounds fr ON s.id = fr.startup_id WHERE s.industry = 'AI/ML' GROUP BY s.id ORDER BY SUM(amount_usd) DESC LIMIT 10`</code>
"Funding velocity by stage"	<code>`WITH rounds AS (SELECT startup_id, stage, funding_date, LAG(funding_date) OVER (PARTITION BY startup_id ORDER BY funding_date) as prev_date FROM funding_rounds) SELECT stage, AVG(JULIANDAY(funding_date) - JULIANDAY(prev_date)) as avg_days_between FROM rounds WHERE prev_date IS NOT NULL GROUP BY stage`</code>
"Series A to Series B conversion rate by year"	See intermediate SQL examples below

Working with data warehouses:

When your data lives in Snowflake, BigQuery, Redshift, or another warehouse, the same principle applies. Claude can:

- Write complex analytical queries with window functions
- Generate CTEs for multi-step transformations
- Create aggregations across dimensions
- Handle time-series analysis and cohort queries

Best practices for AI-assisted querying:

1. Validate AI-generated queries - Always review before running on production data
2. Start with sample data - Test queries on a subset first
3. Use LIMIT clauses - Prevent runaway queries
4. Iterate conversationally - "Now filter that to only AI/ML industry"

Intermediate SQL Patterns:

This workshop uses SQL patterns beyond basic SELECT queries:

```
-- Window function: Rank startups by funding within industry
SELECT
    s.name,
    s.industry,
    fr.amount_usd,
    fr.stage,
    RANK() OVER (PARTITION BY s.industry ORDER BY fr.amount_usd DESC) as industry_rank
FROM funding_rounds fr
JOIN startups s ON fr.startup_id = s.id
WHERE fr.stage = 'Series A';

-- CTE: Calculate funding velocity (days between rounds)
WITH round_sequence AS (
    SELECT
        startup_id,
        stage,
        funding_date,
        LAG(funding_date) OVER (PARTITION BY startup_id ORDER BY funding_date) as prev_round_date
    FROM funding_rounds
)
SELECT
    s.name,
    rs.stage,
    JULIANDAY(rs.funding_date) - JULIANDAY(rs.prev_round_date) as days_since_last_round
FROM round_sequence rs
JOIN startups s ON rs.startup_id = s.id
WHERE rs.prev_round_date IS NOT NULL
ORDER BY days_since_last_round ASC
LIMIT 20;

-- Cohort analysis: Series A to Series B conversion by year
WITH series_a AS (
    SELECT startup_id, funding_date as series_a_date
    FROM funding_rounds WHERE stage = 'Series A'
),
series_b AS (
    SELECT startup_id, funding_date as series_b_date
    FROM funding_rounds WHERE stage = 'Series B'
)
SELECT
    strftime('%Y', a.series_a_date) as series_a_year,
    COUNT(DISTINCT a.startup_id) as series_a_count,
    COUNT(DISTINCT b.startup_id) as converted_to_b,
    ROUND(COUNT(DISTINCT b.startup_id) * 100.0 / COUNT(DISTINCT a.startup_id), 1) as conversion_rate
FROM series_a a
LEFT JOIN series_b b ON a.startup_id = b.startup_id
GROUP BY series_a_year
ORDER BY series_a_year;
```

About the Workshop Dataset

Throughout this workshop, we'll use `data/startup-funding.db` - a SQLite database modeled on real startup funding data. The dataset tracks venture capital activity from 2018-2025, covering the AI boom, pandemic-era funding surge, and the 2023-2024 correction.

What's in it:

- 200 startups across industries: AI/ML, Fintech, Healthcare, Developer Tools, Cybersecurity, Climate Tech, and more
- 66 investors including well-known names (Y Combinator, Sequoia, a16z, Accel) and sector-focused funds
- ~480 funding rounds from Pre-Seed through Series C, with realistic amounts and valuations
- Growth metrics for ~50 startups showing ARR, employee count, and user growth over time

Why this dataset:

- It's structured like real VC/market research data you'd encounter professionally
- The relationships (startups → funding rounds → investors) let us practice JOINS and analytical queries
- Time-series data enables trend analysis, cohort studies, and anomaly detection
- Includes recognizable AI coding tools (Cursor, Replit, Codeium) for relatable analysis

Sample question the data can answer:

- "Which AI coding startups raised Series A in 2024, and who led those rounds?"
- "What's the average time from Seed to Series A by industry?"
- "Which investors have the best track record getting portfolio companies to Series B?"

Claude can query this directly:

```
sqlite3 data/startup-funding.db "SELECT stage, COUNT(*), printf('$.1fM', AVG(amount_usd)/1000000.0)
```

Bash vs. Other Approaches

Why not just use RAG (retrieval augmented generation) or vector search?

Approach	How It Works	Problem
Prompt stuffing	Load everything into context	Hits token limits fast
Vector search	Find semantically similar chunks	Imprecise for structured data
SQL + AI	Translate questions to queries	Precise, scalable, production-ready
Bash + filesystem	Exact pattern matching, on-demand	Precise, efficient, debuggable

When you need "all Series A rounds over \$20M in AI/ML", vector search might return similar-sounding content. SQL gives you exactly what you asked for:

```
SELECT s.name, fr.amount_usd, fr.funding_date
FROM funding_rounds fr
JOIN startups s ON fr.startup_id = s.id
WHERE fr.stage = 'Series A'
      AND fr.amount_usd > 20000000
      AND s.industry = 'AI/ML';
```

Claude knows how to write these queries. Let it.

Demo: Watch Claude Work

Live demo with this prompt:

```
Look at the startup-funding.db database. Tell me:
1. How many startups are there by industry?
2. What's the total funding by stage?
3. Who are the top 5 investors by number of deals led?
4. Which AI coding tools (Cursor, Replit, Codeium, etc.) have raised Series A?
```

Watch how Claude:

1. Uses Bash with sqlite3 to explore the schema
2. Writes and executes SQL queries
3. Reasons about the results
4. Presents structured findings

03. Block 2: Lab 1 - Exploring the Startup Funding Database (45 min)

Task: Data Exploration with Claude

Use Claude Code to explore the startup funding data in this repo.

Exercise 1: Schema Discovery (5 min)

```
> What tables are in the startup-funding.db database? Show me the schema for each.
```

Notice: Claude uses Bash with sqlite3 to explore.

Exercise 2: Basic Aggregations (10 min)

```
> From startup-funding.db, answer these questions:
> 1. How many funding rounds happened each year? Break down by stage.
> 2. What's the total funding amount by industry?
> 3. Which stage has the highest average deal size?
```

Notice: Claude writes SQL with GROUP BY, aggregations.

Exercise 3: Trend Analysis (15 min)

Run this analysis to understand funding trends:

```
> Analyze monthly funding trends for AI/ML companies from 2023 onwards.
> Show me:
> - Deal count per month
> - Total funding per month
> - Average deal size per month
> Format as a table sorted by month.
```

The SQL pattern for this:

```
SELECT
    strftime('%Y-%m', funding_date) as month,
    COUNT(*) as deal_count,
    printf('$.1fM', SUM(amount_usd)/1000000.0) as total_funding,
    printf('$.1fM', AVG(amount_usd)/1000000.0) as avg_deal_size
FROM funding_rounds fr
JOIN startups s ON fr.startup_id = s.id
WHERE s.industry = 'AI/ML'
    AND funding_date >= '2023-01-01'
GROUP BY month
ORDER BY month DESC;
```

Exercise 4: Investor Analysis (10 min)

```
> Who are the top 15 investors by portfolio size?
> For each, show:
> - Number of portfolio companies
> - Total investments (count of rounds)
> - Follow-on rate (avg rounds per company)
> Only include investors with at least 3 portfolio companies.
```

The SQL pattern for this:

```
SELECT
    i.name as investor,
    i.type,
    COUNT(DISTINCT fr.startup_id) as portfolio_companies,
    COUNT(*) as total_investments,
    ROUND(COUNT(*) * 1.0 / COUNT(DISTINCT fr.startup_id), 2) as follow_on_rate
FROM investors i
JOIN funding_rounds fr ON fr.lead_investor_id = i.id
GROUP BY i.id
HAVING portfolio_companies >= 3
ORDER BY portfolio_companies DESC
LIMIT 15;
```

Exercise 5: Analytical Question (5 min)

Apply the Data Analysis Loop to answer this question:

```
> Which AI coding tools raised Series A in 2024-2025?
> Rank them by likelihood of getting Series B based on:
> - Funding amount vs. industry median
> - Time since founding to Series A
> - Investor track record (has the lead investor backed other Series B+ companies?)
> Give me your prediction with supporting evidence.
```

This exercise combines:

Monitor: Query current state of AI coding tool funding

Explore: Dig into factors that predict Series B success

Craft: Build a thesis with supporting data

Impact: Make a concrete prediction

Discussion Questions

1. Which SQL patterns were new to you?
2. How did Claude handle complex multi-table queries?
3. Where did you see the Data Analysis Loop in action?
4. What would you change about these queries for your own data?

04. BREAK (10 min)

05. Block 3: Theory - Web Tools for Research (30 min)

Why Web Tools Matter

Your data lives in databases. But context lives on the web:

- Company websites
- News articles
- LinkedIn profiles
- Industry reports
- Competitor information

Claude's web tools bridge the gap between your internal data and external context.

WebSearch: Finding Information

WebSearch queries the web and returns relevant results.

Good prompts for WebSearch:

```
> Search for recent news about Cursor AI funding
> Find information about the AI coding tools market size 2024
> Look up reviews of Replit AI features
```

What you get back:

- Search result titles and snippets
- URLs to sources
- Dates when available

WebFetch: Getting Page Content

WebFetch retrieves and reads a specific URL.

When to use WebFetch:

- You have a specific URL to read
- You need detailed content from a page
- You want to extract structured data from a site

Example flow:

1. WebSearch → find relevant URLs
2. WebFetch → get detailed content from best result
3. Analyze → synthesize findings

Combining Data and Web Research

The real power comes from combining your structured data with web context:

Example Workflow:

- ```
1. Query database: "Show me AI coding startups that raised Series A in 2024"
2. For top results, WebSearch: "[startup name] latest news"
3. WebFetch: Get detailed info from relevant articles
4. Synthesize: "Cursor raised $60M Series B, growing 3x YoY"
5. Update analysis: Add context to database findings
```

This is the Monitor → Explore → Craft → Impact loop in action.

Practical Research Patterns

| Research Goal         | Tool Sequence                                     |
|-----------------------|---------------------------------------------------|
| Validate funding data | Query DB → WebSearch for announcements            |
| Company deep dive     | Query DB → WebFetch company site → WebSearch news |
| Market sizing         | Query DB aggregates → WebSearch industry reports  |
| Competitive analysis  | Query DB for comparables → WebSearch each         |
| Trend validation      | Query DB time series → WebSearch for explanations |

Beyond Built-in: External Web Services

Claude's built-in WebSearch and WebFetch cover most needs. But when you need more control, there are specialized services:

| Service     | What It Does               | When to Use                                                  |
|-------------|----------------------------|--------------------------------------------------------------|
| Tavily      | AI-native search API       | When you need structured search results optimized for agents |
| Firecrawl   | Web scraping and crawling  | When you need to extract data from complex sites             |
| Bright Data | Web scraping with anti-bot | When sites block automated access                            |

These services connect to Claude through MCP servers. We'll cover MCP in Week 3.

Limitations to Know

- Rate limits: Don't hammer with requests
- Paywalled content: Can't access subscription sites

Dynamic content: Some sites don't render well

Accuracy: Web data may be outdated

---

## 06. Block 4: Lab 2 - Building a Research Workflow (45 min)

Task: Startup Research for Investment Analysis

Build a research workflow that combines database queries with web research.

Scenario: You're analyzing AI coding tools for an investment memo. Use Claude to research.

Step 1: Database Foundation (10 min)

Start with your structured data:

```
> From startup-funding.db, find all AI/ML startups in the Developer Tools or IDEs sub-industry.
> Show me their funding history, investors, and latest valuation.
> Focus on companies that have raised Series A or later.
```

Step 2: Web Enrichment (15 min)

Pick 3 startups from your results and research each:

```
> For Cursor, Replit, and Codeium:
> 1. Search for their latest funding news
> 2. Find their current employee count (if available)
> 3. Look for any product announcements in the last 6 months
> Compile findings into a comparison table.
```

Watch Claude combine database findings with web research.

Step 3: Synthesize Analysis (15 min)

Apply the Data Analysis Loop:

```
> Based on the database data and web research, create an investment brief:
>
> **Monitor:** Current state of AI coding tool funding
> **Explore:** Key differentiators between top players
> **Craft:** 3-5 key insights with supporting evidence
> **Impact:** Which company is best positioned for Series B/C success and why?
>
> Save the brief to output/ai-coding-tools-analysis.md
```

Step 4: Connect to Broader Market (5 min)

```
> Compare the AI coding tools funding to the broader Developer Tools category.
> - How does their average deal size compare?
> - Are they raising at higher or lower valuations?
> - What does this suggest about market sentiment?
```

Deliverable

By end of lab, you should have:

- [ ] Queried the funding database for AI coding startups
- [ ] Enriched 3 companies with web research
- [ ] Created an analysis brief using the Data Analysis Loop
- [ ] Connected findings to broader market trends

07. Wrap-Up (15 min)

Key Takeaways

1. Claude is an agent, not a chatbot - It uses tools to take action
2. The loop: Think → Select Tool → Execute → Observe → Repeat
3. The Data Analysis Loop: Monitor → Explore → Craft Story → Impact
4. Built-in tools: Read, Write, Bash, Glob, Grep, WebSearch, WebFetch
5. SQL is your friend - Window functions, CTEs, and aggregations unlock powerful analysis
6. Combine structured data with web context - Database findings + web research = complete picture

What We Didn't Cover (But You Can Explore)

- Custom tools via MCP - Next week's topic
- Tool schemas - How tools define their inputs
- Error handling - What happens when tools fail
- Permissions - Controlling what Claude can do

Homework

Data Analysis Assignment:

1. Pick a domain from the startup-funding.db to analyze:

| Domain        | Analysis Focus                          |
|---------------|-----------------------------------------|
| Fintech       | Compare to AI/ML funding trends         |
| Healthcare    | Identify fastest-growing sub-industries |
| Climate Tech  | Map investor specialization             |
| Cybersecurity | Analyze Series A to B conversion        |
| Your choice   | Define your own analysis question       |

2. Apply the Data Analysis Loop:  
Monitor: Run 3 baseline queries  
Explore: Dig into one anomaly or trend  
Craft: Write 3-5 insights with supporting data  
Impact: Make one concrete prediction or recommendation
3. Create an analysis document with:
  - Your SQL queries (with explanations)
  - Key findings with data

- One insight enriched with web research
- Your prediction and confidence level

4. Save to `output/week2-homework-[your-domain].md`

Next Week Preview

Week 3: MCP Integration

- Connect Claude to external services
- Build a data MCP with context management
- Access private data sources
- Create connected workflows for your domain

---

## 08. Facilitator Notes

Common Issues

1. SQLite syntax errors: Help with quote escaping, date functions
2. WebSearch returns no results: Check query specificity, try different terms
3. WebFetch fails: Some sites block automated access, try alternatives
4. Slow responses: Database queries are fast, web tools take longer

Timing Adjustments

- Lab 1 is denser than before - allow full 45 minutes
- Lab 2 can be shortened if running long by skipping Step 4
- Focus on ensuring everyone runs at least 3-4 SQL queries

Discussion Prompts

- "What surprised you about how Claude wrote SQL?"
- "Where did you see the Data Analysis Loop in action?"
- "What analysis would you run on your own company's data?"

Energy Management

- SQL can be intimidating - celebrate successful queries
- Have participants share interesting findings during labs
- The startup data is engaging - lean into "which startups will succeed?" discussions

SQL Cheat Sheet (for reference)

```
-- Count by category
SELECT industry, COUNT(*) FROM startups GROUP BY industry;

-- Join tables
SELECT s.name, fr.amount_usd
FROM funding_rounds fr
JOIN startups s ON fr.startup_id = s.id;

-- Window function (rank)
RANK() OVER (PARTITION BY industry ORDER BY amount DESC)

-- CTE (common table expression)
WITH cte AS (SELECT ...) SELECT * FROM cte;

-- Date filtering
WHERE funding_date >= '2024-01-01'

-- Formatted output
printf('$%.1fM', amount/1000000.0)
```