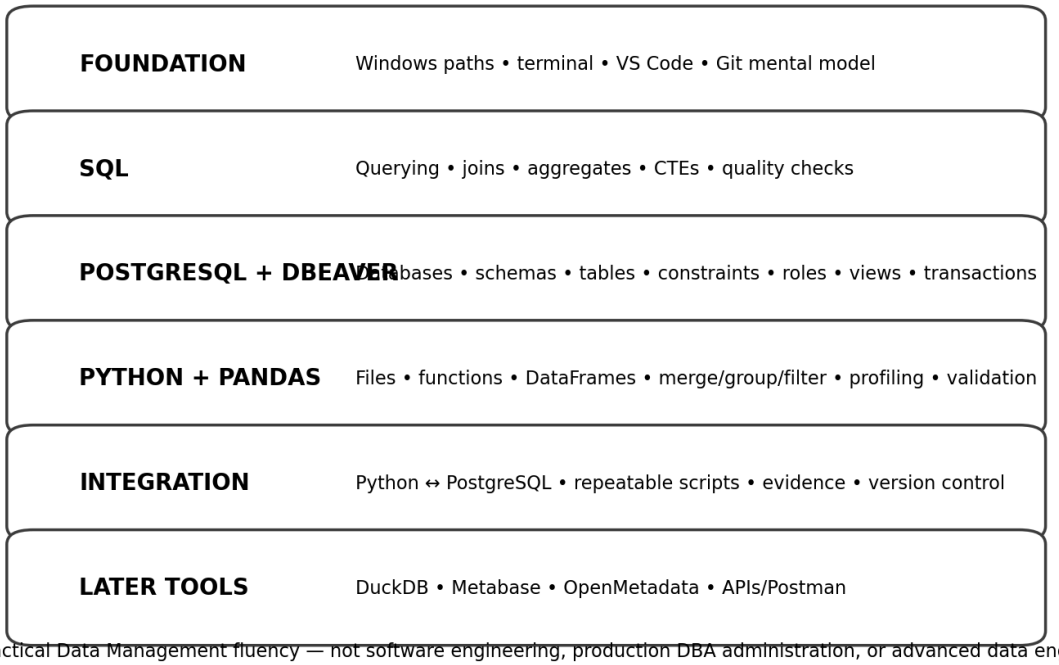
🛠️ Detailed Technology Competency Reference — Skill Depth, Readiness & Stop Boundaries

SQL • PostgreSQL • DBeaver • Python • pandas • Git • VS Code/PowerShell • data formats • later tools

CDMP DMBOK Mastery Lab • Applied Learning Companion Program

Training philosophy: You are not learning technology to collect tools. You are learning enough technology to inspect, build, test, reconcile, trace, protect and explain data-management work independently. Every skill has a stopping boundary.



Target: practical Data Management fluency — not software engineering, production DBA administration, or advanced data engineering.

📌 *Scope clarification: This detailed technology competency reference supplements the canonical Technology Competency Roadmap. It is not a separate training program and not a course library. You will choose your own courses, books, videos, tutorials and documentation. The CDMP Applied Laboratory will maintain what to learn, why the skill matters to Data Management, the target depth, which Meridian/DMBOK work it supports, the readiness test and the stopping boundary. The individual technology folders are places to store your chosen resources, notes, exercises and proof of skill; they are not a commitment for ChatGPT to author full technology courses.*

Recommended learning sequence.

1. Target competency model

Technology	Target depth	You should be able to...	Stop before...
SQL	Strong working fluency	Write multi-table queries, aggregate correctly, profile/reconcile data, use CTEs/window functions, validate integrity and explain query grain.	Advanced query tuning, procedural SQL specialization, vendor-specific expert features.
PostgreSQL	Practical builder/user	Create databases/schemas/tables/constraints/views/roles, use transactions, inspect metadata, rebuild the lab from scripts.	Production HA/replication, deep tuning, OS-level administration.
DBeaver	Strong workbench fluency	Navigate objects, inspect metadata, use SQL Editor, save scripts, view/generate DDL, export/import lab data, understand connections.	Memorizing every UI feature or treating GUI clicks as a substitute for SQL.
Python	Practical scripting	Read/write files, structure code with functions, handle errors, manipulate paths/JSON/CSV and automate repetitive data tasks.	Web/app frameworks, algorithms-heavy CS, advanced OO design.
pandas	Strong Data Management fluency	Inspect DataFrames, clean/transform, merge/reconcile, profile, validate and move between files/SQL.	Advanced numerical/scientific computing or ML.
Git	Strong basics	Track scripts/docs, compare changes, commit meaningful milestones, use branches lightly and avoid secrets.	Complex enterprise branching/release engineering.
VS Code + PowerShell	Practical environment literacy	Navigate projects, select Python environment, run/debug, manage folders/files and simple commands.	Shell scripting as a separate specialization.
DuckDB / Metabase / OpenMetadata / APIs	Targeted later use	Use the feature that makes a DMBOK concept observable.	Product administration/certification or feature-tour learning.

2. Curriculum blocks

T0 — Environment literacy

What to learn	Practice / proof	CDMP support
Windows folders/paths; terminal vs shell; PowerShell basics; VS Code workspace;	Can navigate the project, run commands intentionally, explain where files and	Before all labs

What to learn	Practice / proof	CDMP support
file extensions; PATH concept; install vs package; Git mental model.	executables live, and recover from a wrong working directory.	

T1 — SQL Foundations I

What to learn	Practice / proof	CDMP support
SELECT, FROM, WHERE, ORDER BY, aliases, DISTINCT, LIMIT, NULL logic, basic types, simple expressions.	Write and explain queries from a blank editor; predict results before execution; inspect row/column meaning.	Ch 1, 4, 5, 6, 8, 10-13

SQL independence check: Given a business question, first write the expected grain and needed tables in words. Then write the query. After execution, explain why the row count and aggregation make sense.

T2 — SQL Foundations II

What to learn	Practice / proof	CDMP support
PK/FK concepts, JOINs, GROUP BY, aggregates, HAVING, CASE, subqueries, CTEs, set operations at practical depth.	Join Customer→Order→OrderItem correctly, aggregate without double-counting, explain grain.	Ch 5, 8, 10-13

SQL independence check: Given a business question, first write the expected grain and needed tables in words. Then write the query. After execution, explain why the row count and aggregation make sense.

T3 — SQL for Data Management

What to learn	Practice / proof	CDMP support
Profiling queries, duplicates, missing/invalid values, referential checks, reconciliation, window functions, validation queries, safe UPDATE/DELETE habits.	Build a repeatable profiling/validation script and explain each test.	Ch 6-8, 10-13

SQL independence check: Given a business question, first write the expected grain and needed tables in words. Then write the query. After execution, explain why the row count and aggregation make sense.

T4 — PostgreSQL Foundations

What to learn	Practice / proof	CDMP support
Server/database/schema/table distinction; DDL; data types; constraints; sequences/identity; views; transactions;	Create the Meridian lab database from scripts and explain every object; not reliant on GUI-only clicks.	Ch 5-8, 10-13

What to learn	Practice / proof	CDMP support
users/roles; basic indexes; backup/rebuild awareness.		

T5 — DBeaver Workbench

What to learn	Practice / proof	CDMP support
Connections; Database Navigator; schemas/tables/views; metadata inspection; SQL Editor; saved scripts; result export; ERD/data tools; generate DDL/SQL.	Can inspect an unfamiliar database and locate objects/metadata without being led click-by-click.	Ch 4-13

T6 — Python Foundations

What to learn	Practice / proof	CDMP support
Variables/types; lists/dicts/sets/tuples; conditions; loops; functions; modules; pathlib/files; exceptions; simple CLI scripts; reading JSON/CSV.	Write a small file-processing script from a blank file and modify it without copying a template.	Ch 8, 10, 13, 14

Python independence check: Read and annotate supplied examples, then change the input and rebuild the learning-critical portion from a blank file. A notebook that runs is not enough if you cannot explain the DataFrame state before and after each transformation.

T7 — Python + pandas

What to learn	Practice / proof	CDMP support
venv/pip; DataFrame/Series; read/write CSV/Excel/SQL/JSON/Parquet; selection/filtering; missing values; duplicates; merge; groupby; type conversion; validation/profiling.	Profile and reconcile Meridian datasets and explain each transformation at row/column level.	Ch 8, 10-14

Python independence check: Read and annotate supplied examples, then change the input and rebuild the learning-critical portion from a blank file. A notebook that runs is not enough if you cannot explain the DataFrame state before and after each transformation.

T8 — Python ↔ PostgreSQL

What to learn	Practice / proof	CDMP support
Connection concepts; safe credentials; SQLAlchemy/driver awareness;	Automate one repeatable load/validation workflow and rerun it from a clean state.	Ch 8, 10-13

What to learn	Practice / proof	CDMP support
parameterized queries; read_sql/to_sql at practical depth; transaction safety.		

Python independence check: Read and annotate supplied examples, then change the input and rebuild the learning-critical portion from a blank file. A notebook that runs is not enough if you cannot explain the DataFrame state before and after each transformation.

T9 — Git & Evidence Discipline

What to learn	Practice / proof	CDMP support
init/clone awareness, status, add, commit, log, diff, branch basics, .gitignore, README, tagging milestones; never commit secrets/venv/generated sensitive files.	Can show how an artifact evolved and recreate a lab milestone.	All technical labs

T10 — Data Formats

What to learn	Practice / proof	CDMP support
CSV delimiters/quoting/encoding; JSON objects/arrays/nesting; Parquet columnar concept; dates/timestamps/time zones; identifiers; null/blank/zero.	Diagnose a broken source file and explain why representation affects quality/integration/metadata.	Ch 4, 8, 9, 12-14

T11 — Later Tools

What to learn	Practice / proof	CDMP support
DuckDB for file analytics; Metabase for BI; OpenMetadata for catalog/glossary/lineage/quality/governance; REST/JSON/Postman literacy.	Use each only when a DMBOK chapter creates a clear learning need; explain the concept before the product feature.	Ch 8, 11-14

T12 — Microsoft Purview Data Governance — Career Specialization

What to learn

Practice / proof

CDMP support

Data Map; Unified Catalog; governance domains; data products; business concepts/glossary; owners/stewards; metadata curation/discovery; classification at governance depth; lineage interpretation; Data Quality/health; governed access/right-use concepts.

Map a Meridian governance problem into Purview without click-by-click instructions and explain each configuration in DAMA/DMBOK terms.

Ch 3, 12, 13 + career portfolio

Stop before production pipelines, Fabric analytics engineering, ADF/Synapse/Databricks specialization, Spark, infrastructure-as-code, extensive SDK/API development, or deep security operations.

T13 — Collibra Governance & Stewardship — Secondary Portability

What to learn

Practice / proof

CDMP support

Stewardship; communities/domains; glossary; ownership; policies; workflows; metadata; classification; lineage; governance operating practices.

Explain how the same Meridian governance design transfers from Purview into Collibra concepts; do hands-on work when access or employer need justifies it.

Ch 3, 12 + career portability

Stop before Collibra development/administration specialization unless a target job requires it.

T14 — Employer-Specific Governance Platforms

What to learn

Practice / proof

CDMP support

Recognition-level understanding of Informatica CDGC, Alation, and Atlan; identify their equivalents for catalog, glossary, ownership/stewardship, lineage, classification, Data Quality, and workflow.

Compare one Meridian governance use case across at least two alternative platforms using official documentation, screenshots, or real access.

Employer-driven career targeting

Deepen only the platform named by a target employer or job description.

Career-platform rule: Purview is the primary specialization; Collibra is the secondary portable-governance platform; OpenMetadata remains the Meridian practice sandbox. Platform learning sits on top of DMBOK knowledge and must not become a Data Engineering curriculum.

3. SQL detailed syllabus

Tier	Topics	Practice ideas
SQL A — Read data	SELECT/FROM, aliases, WHERE, comparison/Boolean logic, NULL, ORDER BY, DISTINCT, LIMIT, basic functions.	Answer business questions against one Meridian table; predict filters before running.
SQL B — Relate data	Primary/foreign-key meaning, INNER/LEFT JOIN, join grain, many-to-many awareness.	Customer→Order→OrderItem; identify duplicate multiplication caused by wrong grain.
SQL C — Summarize	COUNT/SUM/AVG/MIN/MAX, GROUP BY, HAVING, CASE.	Revenue/order/customer summaries with explicit grain.

Tier	Topics	Practice ideas
SQL D — Structure reasoning	Subqueries, CTEs, set operations, views at practical depth.	Make validation logic readable; compare source populations.
SQL E — Data Management diagnostics	Duplicate detection, null/invalid/reference checks, reconciliation, date ranges, string cleanup, window functions.	Build Chapter 10/13 profiling and matching support queries.
SQL F — Safe changes	INSERT/UPDATE/DELETE, transactions, before/after checks, constraints.	Repair controlled defects in a transaction and verify outcome.

4. PostgreSQL detailed syllabus

- Server vs cluster/instance awareness; database vs schema vs table vs view.
- Create/drop database only in a safe local lab; connect deliberately.
- Data types appropriate to IDs, text, dates/timestamps, exact numeric amounts and Boolean/status data.
- CREATE TABLE; primary keys; foreign keys; NOT NULL; UNIQUE; CHECK; identity/surrogate-key awareness.
- Schemas and qualified object names.
- Views and why they do not automatically fix business-definition governance.
- Transactions: BEGIN/COMMIT/ROLLBACK and why safe change matters.
- Users/roles, GRANT/REVOKE and least-privilege practice.
- Indexes at recognition/basic practical depth: why they exist, not deep performance tuning.
- Catalog/information-schema inspection at awareness level.
- Backup/rebuild: use scripts and lab-safe dump/restore or equivalent instructions when Chapter 6 is built.
- Environment separation concept: training should distinguish source/setup scripts from generated state.

Version choice: Use the PostgreSQL release marked Current/Supported in the official documentation at installation time, and avoid a development/beta line for the baseline lab unless intentionally testing one. As verified during this August 2026 QC, PostgreSQL 18 is current/supported and PostgreSQL 19 is a beta/development line.

5. DBeaver detailed syllabus

- Create and identify a database connection; understand what host/port/database/user mean.
- Use Database Navigator to find schemas, tables, views and other objects.
- Open object properties and inspect columns, keys and metadata.
- Use the SQL Editor to create, save, reopen and execute scripts intentionally.
- Understand selected statement versus full-script execution.
- Inspect result grids without confusing displayed data with stored definitions.
- Generate or view DDL when useful for learning and evidence.
- Export/import lab data deliberately and document what changed.
- Use diagrams/ERD features as support, while retaining conceptual/logical/physical model distinctions from DMBOK.

6. Python foundations syllabus

Topic	Why a data manager needs it	Practice
Values and types	Understand how representations differ and why type conversion fails.	Parse customer/order values and identify type mismatches.

Topic	Why a data manager needs it	Practice
Collections	Model groups, mappings and unique values.	Lists of records, dictionaries of code mappings, sets for uniqueness checks.
Control flow	Express validation/decision logic.	Flag invalid statuses or quantities.
Functions	Make repeatable logic explicit and testable.	Create a reusable normalization or validation function.
Files/pathlib	Work safely with project folders and files.	Enumerate source files, read/write outputs to correct folders.
JSON/CSV	Understand common integration shapes.	Read nested service JSON and flat CSV.
Exceptions	Recognize and handle predictable failures.	Catch missing file or conversion failure and report useful context.
Modules/environments	Separate project dependencies from global Python.	Create .venv and requirements/rebuild notes.

7. pandas syllabus

- DataFrame and Series mental model; rows, columns, index and dtypes.
- read_csv/read_json/read_sql/read_parquet and corresponding write methods where appropriate.
- head/tail/info/dtypes/value_counts/describe as first-pass inspection tools.
- Select/filter rows and columns without losing sight of business grain.
- Missing values: identify, interpret and handle based on business meaning—not automatic blanket filling.
- Duplicates: detect exact and candidate duplicates; preserve source evidence before dedupe.
- Type conversion, string normalization and date/timestamp parsing.
- merge/join with explicit key and expected cardinality; validate row-count effects.
- groupby/aggregations for profiling and reconciliation.
- Rule checks and exception datasets.
- Writing curated outputs while preserving raw input unchanged.

8. Git and reproducibility syllabus

- Why version control matters: history, comparison, rollback and evidence.
- git init/status/add/commit/log/diff; basic branch/merge awareness.
- .gitignore for .venv, secrets, generated/temp files and local database artifacts when appropriate.
- Commit meaningful learning milestones, not every keystroke.
- Use README files to explain setup, data sources, assumptions and rebuild steps.
- Never store real credentials or sensitive data in the repository.

9. Later-tool trigger matrix

Tool	Introduce when	Learning purpose
DuckDB	Chapter 14 or earlier lightweight file analysis if useful.	Query CSV/JSON/Parquet directly; compare file-oriented analytical work with operational PostgreSQL.
Metabase	Chapter 11.	Turn warehouse facts/dimensions into questions and dashboards; connect KPI definitions to data.

Tool	Introduce when	Learning purpose
OpenMetadata	Chapter 12/13, optionally Chapter 3.	See catalog, glossary, ownership, lineage, classification and quality features after the concepts are understood manually.
REST / JSON / Postman	Chapter 8 or later.	Understand request/response payloads and how APIs participate in integration; no deep API programming required.

10. Official learning sources

Official source	URL	Use in program
PostgreSQL 18 Tutorial	https://www.postgresql.org/docs/current/tutorial.html	Hands-on intro to PostgreSQL, relational concepts and SQL; covers tables, queries, joins, aggregates, views, foreign keys, transactions and window functions.
PostgreSQL Documentation	https://www.postgresql.org/docs/	Confirm current supported version before installation.
DBeaver Database Navigator	https://dbeaver.com/docs/dbeaver/Database-Navigator/	Learn to browse connections, schemas, tables, views and object metadata.
DBeaver SQL Editor	https://dbeaver.com/docs/dbeaver/SQL-Editor/	Learn to create, save and execute reusable SQL scripts.
Python venv	https://docs.python.org/3/library/venv.html	Create isolated, recreatable project environments; do not commit the venv itself.
VS Code Python	https://code.visualstudio.com/docs/languages/python	Interpreter/environment selection, running and debugging Python.
pandas Getting Started	https://pandas.pydata.org/docs/getting_started/	DataFrame-based tabular analysis and manipulation.
pandas Read/Write	https://pandas.pydata.org/docs/getting_started/intro_tutorials/02_read_write.html	CSV, Excel, SQL, JSON, Parquet and other I/O patterns.
Pro Git	https://git-scm.com/book/en/v2	Version-control concepts and Git basics.
DuckDB Guides	https://duckdb.org/docs/current/guides/overview	Later lightweight analytical/file work.
Metabase Docs	https://www.metabase.com/docs/latest/	Later BI questions, SQL/query builder and dashboards.
OpenMetadata Docs	https://docs.open-metadata.org/	Later catalog, governance, lineage and quality practice.

11. “Ready for labs” technology gates

Gate	Minimum independent performance
Gate A — SQL reader	Write single-table filters/sorts, explain NULL, identify grain and interpret results.

Gate	Minimum independent performance
Gate B — relational SQL	Join 3 tables correctly, aggregate without accidental duplication, use a CTE.
Gate C — PostgreSQL/DBeaver	Connect, create schema/table from SQL, inspect metadata, run/save scripts, use constraints and a transaction.
Gate D — Python/pandas	Create/activate venv, read CSV/JSON, inspect DataFrame, filter/merge/group, detect nulls/duplicates, write output.
Gate E — reproducibility	Run a project from documented steps, use Git status/commit/diff, keep raw inputs unchanged and preserve evidence.

✅ **Competency-reference principle:** You do not have to finish every technology block before starting CDMP labs. Learn just ahead of need, but never let a lab introduce a large tool skill as unexplained magic.