🧪 Synthetic Data & Defect Injection Blueprint

Designing a safe, repeatable data estate that can fail in educational ways

CDMP DMBOK Mastery Lab • Applied Learning Companion Program

Core principle: Defects are curriculum. They must be intentional, identifiable by stable IDs and tied to a learning objective. Randomly dirty data is less useful than defects that demonstrate a specific DMBOK problem.

1. Dataset tiers

Tier	Use	Typical volume
Tiny demonstration	Explain a concept on screen or paper.	5–30 rows.
Core lab	Normal SQL/database/MDM/quality/integration practice.	100–5,000 rows by table.
Scaled analytical	Performance/format/big-data awareness where volume itself matters.	100k–1M synthetic events, generated only when needed.

2. Stable identifiers

- System IDs: WEB, CRM, POS, PIM, OMS, WMS, ERP, SERV, HRIS, IAM, DWH, CAT, DOC.
- Dataset IDs: DS-CUST-CRM, DS-CUST-POS, DS-CUST-SERV, DS-PROD, DS-ORDER, etc.
- Defect IDs: DQ-001…; security defects SEC-001…; metadata gaps MD-001…; reference/master conflicts MDM-001… when useful.
- Incident IDs: INC-001… so later capstones can cite the same event.
- Decision IDs: DEC-001… for governance and architecture decisions.

3. Initial datasets and sizes

Dataset	Rows	Defects intentionally available
customers_crm.csv	500	Null/stale contact values, formatting differences, duplicate candidate identities.
customers_pos.csv	350	Different IDs, abbreviated names/addresses, inconsistent state/country representations.
customers_service.json	220 identities / 200 cases	Nested JSON, alternate emails, service-only identities, text notes.
products.csv	150	Category drift, lifecycle/status inconsistencies, duplicate-like descriptions.

Dataset	Rows	Defects intentionally available
suppliers.csv	30	Status/reference inconsistencies.
orders / order_items	2,000 / 5,000	Guest orders, dates/amounts, selected controlled invalid references for exercises.
locations	8	Store/FC codes and location types.
inventory_snapshot	1,200	Missing or stale snapshots, impossible negative/large values in controlled defects.
reference files	Small	Code-list version drift and unapproved local values.
documents	20–40	Missing metadata, unclear retention category, duplicate/superseded versions.
click_events	100k+ later	JSON/Parquet, timestamps, event schema version changes.

4. Defect catalog

ID	Defect	Behavior	Primary labs
DQ-001	Duplicate customer candidate	Same person represented differently across/within sources.	Ch 10/13
DQ-002	Missing required business value	Null email or product category where business rule expects value.	Ch 13
DQ-003	Invalid reference	Order/product/category/status value not in approved set.	Ch 5/10/13
DQ-004	Inconsistent representation	State/country/date/name formats differ by source.	Ch 8/12/13
DQ-005	Stale value	Old address or inventory snapshot.	Ch 13
DQ-006	Impossible value	Quantity <=0 or implausible inventory/amount.	Ch 5/13
INT-001	Changed JSON structure	Source field renamed/nested differently.	Ch 8/12/13
MDM-001	Conflicting survivorship attributes	Sources disagree on name/address/email.	Ch 10
REF-001	Code-list drift	Local product category/order-status values differ from governed list.	Ch 10/12/13
META-001	Unknown business definition	Column/KPI lacks description/owner/source.	Ch 12

ID	Defect	Behavior	Primary labs
LIN-001	Missing lineage edge	Warehouse KPI cannot be traced to source transformation.	Ch 8/11/12
SEC-001	Over-permission	Analyst can read restricted synthetic fields.	Ch 7
DOC-001	Retention ambiguity	Old document has no clear disposition/retention classification.	Ch 9
GOV-001	No decision authority	Teams disagree but escalation/decision right is unclear.	Ch 3/16/17

5. Defect injection rules

1. Every defect has a stable ID and intended chapter/learning purpose.
2. Keep a clean baseline generator or source so the defect can be reproduced and removed.
3. Inject defects deliberately via a generator seed or a small controlled script; do not hand-edit random rows without recording them.
4. Do not reveal every defect location before diagnostic labs; store instructor/control metadata separately.
5. After remediation, preserve before/after evidence and record whether the fix corrected the symptom, root cause or both.
6. Reuse selected defects across chapters to show how one problem can have governance, integration, metadata and quality dimensions.

6. Generator design

Python generators should be understandable and parameterized. A learner does not need to memorize the entire generator, but should understand seed, row count, source-specific formatting, stable IDs and defect switches. Generated data should be deterministic when the same seed/configuration is used so results can be compared across labs.

Parameter	Example
seed	20260814
customer_count	500
order_count	2000
inject_duplicate_customers	true
inject_invalid_reference_rate	0.01
source_date_format	CRM=YYYY-MM-DD; POS=MM/DD/YYYY
output_root	data/raw/

7. Provenance manifest

Every source dataset should have a manifest entry with dataset ID, file/table name, generating system, business process, grain, key, approximate row count, created/generated date, synthetic flag, generator/script version, known intentional defects, owner/steward persona, classification and downstream uses. This becomes practical Metadata and supports later Chapters 8, 12 and 13.