# Subtask 2 - Proposed Normalized Schema Design & Clarification Questions

# Subtask 2 – Proposed Normalized Schema Design & Clarification Questions

\*\*Objective:\*\*

Design a draft normalized schema and identify clarification points to confirm before DDL creation.

\*\*Proposed Entities & Relationships:\*\*

| Entity | Key Fields | Relationships | Source |

|--------|-------------|----------------|---------|

| Platform | platform\_id, platform\_name | 1:N → AgileReleaseTrain | JMeter |

| AgileReleaseTrain | art\_id, art\_name, platform\_id | 1:N → Project | JMeter |

| Project | project\_id, project\_name, art\_id | 1:N → Release | JMeter |

| Release | release\_id, release\_name, project\_id | 1:N → TestCycle; N:M → System | Jira |

| System | system\_id, system\_name | M:N → Release | JMeter |

| Release\_System\_Map | release\_id, system\_id, system\_role | Join table for Source/Target | Derived |

| TestCycle | cycle\_id, cycle\_name, release\_id | 1:N → TestRun | Jira |

| TestRun | test\_run\_id, test\_run\_name, cycle\_id | 1:N → Test | GitLab |

| Test | test\_id, test\_run\_id | Leaf entity | Jira/GitLab |

\*\*Relationships Summary:\*\*

- Platform → ART → Project → Release

- Release ↔ Systems (via mapping)

- Release → Test Cycles → Test Runs → Tests

\*\*Clarification Questions:\*\*

1. Is the hierarchy Platform → ART → Project → Release always consistent, or can projects span multiple ARTs?

2. Should Source and Target systems be stored in one mapping table or separately?

3. Can a Release have multiple source and multiple target systems?

4. Are test\_run\_id and test\_id globally unique (GitLab + Jira)?

5. What’s the preferred ingestion path — direct API, JSON intermediate, or CSV extract?

6. Should all IDs be system-generated or align with IDs from Jira/GitLab?

7. Are lowercase snake\_case names confirmed for Postgres?

8. Do we need to store historical versions or just current snapshots?

\*\*Deliverable:\*\*

Proposed normalized schema and list of open questions to validate before generating Postgres DDL.