

Database Design and Normalization

Context-Aware Diagnostic Notification Effectiveness in Problem Reporting

Kyle Owen | Project Design | October 2026

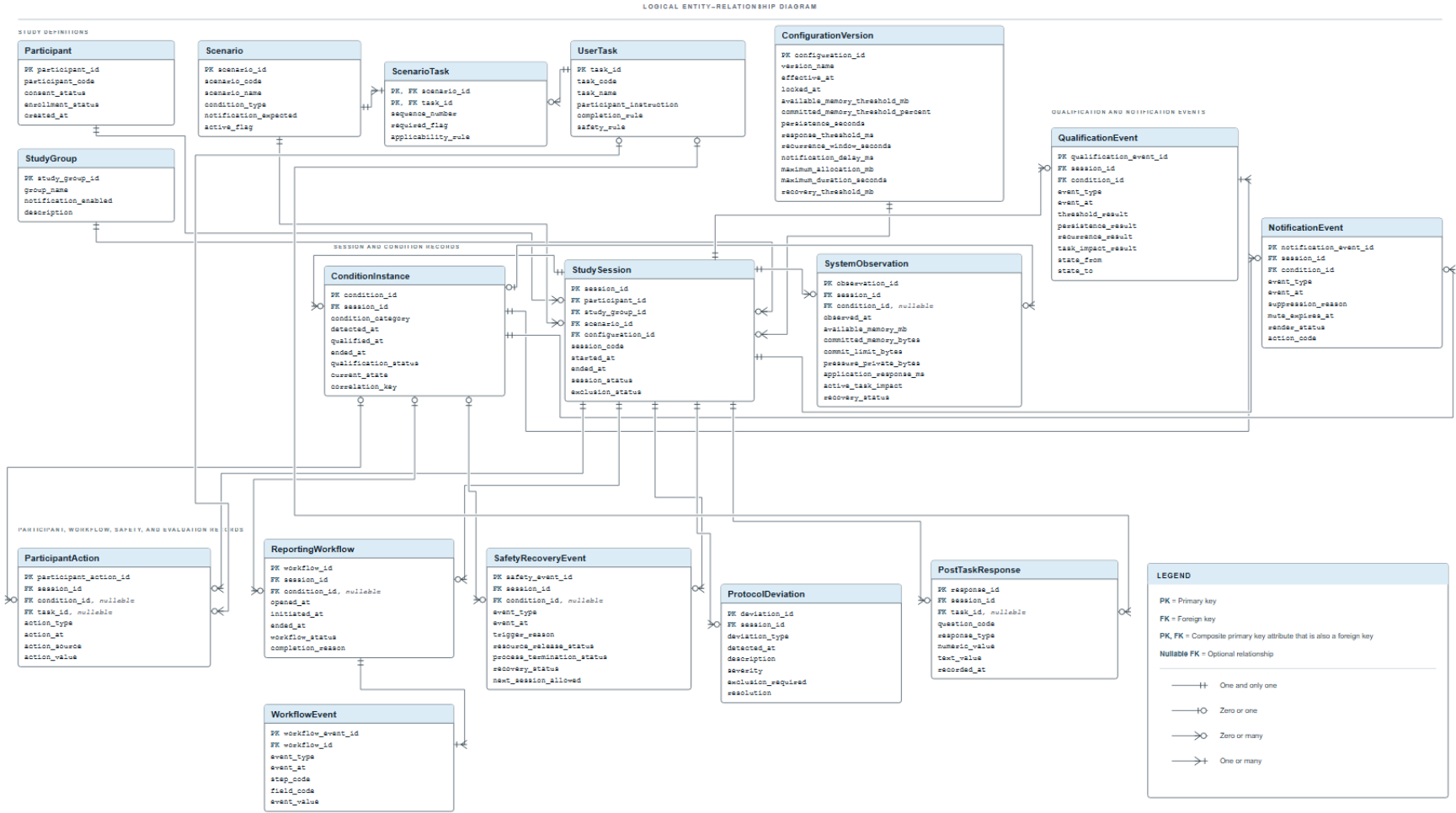
Purpose and Design Scope

This database design supports the controlled context-aware diagnostic notification prototype, the simulated Helpdesk Button workflow, usability tasks, technical calibration, event logging, safety controls, recovery verification, and de-identified research analysis. The design is limited to the proof-of-concept study. It does not create production support tickets or store unrestricted production diagnostics. It also excludes participant names, passwords, client identities, confidential configurations, and private network details.

The schema uses a normalized relational model to keep stable definitions, session facts, time-series observations, participant actions, workflow events, and safety evidence separate but traceable through primary and foreign keys. This structure supports data independence and limits redundancy and inconsistency (Codd, 1970). The design remains platform-neutral. Its tables, keys, constraints, and relationships can be implemented in the relational database management system selected during prototype development.

Entity-Relationship Diagram: Click [Here](#) to Access

Context-Aware Diagnostic Notification Effectiveness in Problem Reporting



Crow's Foot notation - Minimum and maximum cardinality shown at each relationship endpoint

17 TABLES - 24 RELATIONSHIPS

Relationship Summary

Parent Entity	Related Entity	Cardinality	Meaning
Participant	StudySession	1 to many	A coded participant may complete one or more controlled sessions.
StudyGroup	StudySession	1 to many	Each session has one experimental assignment.
Scenario	StudySession	1 to many	Each session executes one primary scenario.
Scenario	UserTask	many to many through ScenarioTask	A scenario contains multiple tasks, and a reusable task may appear in multiple scenarios.
ConfigurationVersion	StudySession	1 to many	Each session references exactly one locked configuration version.
StudySession	SystemObservation	1 to many	A session produces many timestamped measurements.
StudySession	ConditionInstance	1 to many	A session can contain multiple temporary or qualifying condition instances.
ConditionInstance	QualificationEvent	1 to many	A condition moves through a recorded sequence of qualification states.
ConditionInstance	NotificationEvent	1 to many	A condition can generate zero or more requested, suppressed, rendered, dismissed, or muted events.
StudySession	ParticipantAction	1 to many	Participant behavior is recorded independently of notification exposure.
StudySession	ReportingWorkflow	1 to zero or many	A session may contain no workflow or one or more intentionally opened simulated workflows.
ReportingWorkflow	WorkflowEvent	1 to many	A workflow contains a timestamped sequence of interaction events.
StudySession	SafetyRecoveryEvent	1 to many	Stopping, release, recovery, and T8 evidence belong to the session.
StudySession	ProtocolDeviation	1 to many	A session may have no deviations or multiple documented deviations.
StudySession	PostTaskResponse	1 to many	A session can have multiple approved post-task responses.

Table Descriptions

Participant

Design Element	Definition
Purpose	Stores the coded research participant identity used by the prototype. It contains no names, emails, passwords, client identities, or production account information.
Primary key	participant_id
Foreign keys	None
Principal attributes	participant_code, consent_status, enrollment_status, created_at

Design Element	Definition
Integrity and relationship rules	participant_code is unique. One participant can have many study sessions.

StudyGroup

Design Element	Definition
Purpose	Defines the experimental assignment, such as Control or Notification.
Primary key	study_group_id
Foreign keys	None
Principal attributes	group_name, notification_enabled, description
Integrity and relationship rules	group_name is unique. One group can contain many study sessions.

Scenario

Design Element	Definition
Purpose	Defines each approved temporary, qualifying, notification, control, or pilot-informed scenario.
Primary key	scenario_id
Foreign keys	None
Principal attributes	scenario_code, scenario_name, condition_type, notification_expected, active_flag
Integrity and relationship rules	scenario_code is unique. One scenario can be used in many sessions and can be associated with many tasks.

UserTask

Design Element	Definition
Purpose	Stores the reusable definition of each approved participant task.
Primary key	task_id
Foreign keys	None
Principal attributes	task_code, task_name, participant_instruction, completion_rule, safety_rule
Integrity and relationship rules	task_code is unique. Reuse prevents task wording from being copied into every session record.

ScenarioTask

Design Element	Definition
Purpose	Resolves the many-to-many relationship between Scenario and UserTask and preserves the intended task sequence.
Primary key	scenario_id + task_id
Foreign keys	scenario_id to Scenario; task_id to UserTask
Principal attributes	sequence_number, required_flag, applicability_rule

Design Element	Definition
Integrity and relationship rules	The composite primary key ensures that a task appears only once per scenario. sequence_number belongs to the relationship.

ConfigurationVersion

Design Element	Definition
Purpose	Stores the approved and versioned technical calibration values used by a session.
Primary key	configuration_id
Foreign keys	None
Principal attributes	version_name, effective_at, locked_at, available_memory_threshold_mb, committed_memory_threshold_percent, persistence_seconds, response_threshold_ms, recurrence_window_seconds, notification_delay_ms, maximum_allocation_mb, maximum_duration_seconds, recovery_threshold_mb
Integrity and relationship rules	A locked configuration is referenced rather than copied into observation and event rows.

StudySession

Design Element	Definition
Purpose	Represents one controlled study session and serves as the main parent record for runtime events.
Primary key	session_id
Foreign keys	participant_id to Participant; study_group_id to StudyGroup; scenario_id to Scenario; configuration_id to ConfigurationVersion
Principal attributes	session_code, started_at, ended_at, session_status, exclusion_status
Integrity and relationship rules	session_code is unique. Each session uses one participant, one study group, one primary scenario, and one locked configuration version.

SystemObservation

Design Element	Definition
Purpose	Stores time-series measurements collected during a session.
Primary key	observation_id
Foreign keys	session_id to StudySession; condition_id to ConditionInstance, nullable until a condition exists
Principal attributes	observed_at, available_memory_mb, committed_memory_bytes, commit_limit_bytes, pressure_private_bytes, application_response_ms, active_task_impact, recovery_status
Integrity and relationship rules	Each row represents one timestamped observation. Metrics remain atomic and are not stored as comma-separated collections.

ConditionInstance

Design Element	Definition
Purpose	Represents one correlated temporary or qualifying technical condition within a session.

Design Element	Definition
Primary key	condition_id
Foreign keys	session_id to StudySession
Principal attributes	condition_category, detected_at, qualified_at, ended_at, qualification_status, current_state, correlation_key
Integrity and relationship rules	One session can contain multiple distinct conditions. Notification, action, and safety records reference the same condition_id when applicable.

QualificationEvent

Design Element	Definition
Purpose	Records the state transitions and rule outcomes produced by the qualification engine, including T2 and T3.
Primary key	qualification_event_id
Foreign keys	session_id to StudySession; condition_id to ConditionInstance
Principal attributes	event_type, event_at, threshold_result, persistence_result, recurrence_result, task_impact_result, state_from, state_to
Integrity and relationship rules	The table stores event facts rather than repeatedly overwriting the condition history.

NotificationEvent

Design Element	Definition
Purpose	Records notification eligibility, suppression, request, rendering, dismissal, muting, and action-selection events.
Primary key	notification_event_id
Foreign keys	session_id to StudySession; condition_id to ConditionInstance
Principal attributes	event_type, event_at, suppression_reason, mute_expires_at, render_status, action_code
Integrity and relationship rules	T4 is represented by a Rendered event, not merely a Requested event. Multiple events can belong to one condition.

ParticipantAction

Design Element	Definition
Purpose	Records meaningful participant actions regardless of whether a notification was shown.
Primary key	participant_action_id
Foreign keys	session_id to StudySession; condition_id to ConditionInstance, nullable; task_id to UserTask, nullable
Principal attributes	action_type, action_at, action_source, action_value
Integrity and relationship rules	Actions such as continue, wait, retry, dismiss, mute, open helpdesk, cancel, or take no action are stored as individual records.

ReportingWorkflow

Design Element	Definition
Purpose	Stores one optional simulated reporting workflow opened during a session.
Primary key	workflow_id
Foreign keys	session_id to StudySession; condition_id to ConditionInstance, nullable
Principal attributes	opened_at, initiated_at, ended_at, workflow_status, completion_reason
Integrity and relationship rules	The workflow records the overall lifecycle. It does not create a production support ticket.

WorkflowEvent

Design Element	Definition
Purpose	Stores granular events within a simulated reporting workflow.
Primary key	workflow_event_id
Foreign keys	workflow_id to ReportingWorkflow
Principal attributes	event_type, event_at, step_code, field_code, event_value
Integrity and relationship rules	Separating event history from the workflow header avoids repeating session and condition details on every event.

SafetyRecoveryEvent

Design Element	Definition
Purpose	Records stopping, release, termination, recovery, and T8 gate results.
Primary key	safety_event_id
Foreign keys	session_id to StudySession; condition_id to ConditionInstance, nullable
Principal attributes	event_type, event_at, trigger_reason, resource_release_status, process_termination_status, recovery_status, next_session_allowed
Integrity and relationship rules	Safety and recovery evidence remains independent of the notification and reporting interfaces.

ProtocolDeviation

Design Element	Definition
Purpose	Records technical or procedural deviations and the resulting inclusion decision.
Primary key	deviation_id
Foreign keys	session_id to StudySession
Principal attributes	deviation_type, detected_at, description, severity, exclusion_required, resolution
Integrity and relationship rules	Deviation descriptions are stored once and linked to the affected session.

PostTaskResponse

Design Element	Definition
----------------	------------

Design Element	Definition
Purpose	Stores approved post-task ratings and coded responses without placing variable question columns in StudySession.
Primary key	response_id
Foreign keys	session_id to StudySession; task_id to UserTask, nullable
Principal attributes	question_code, response_type, numeric_value, text_value, recorded_at
Integrity and relationship rules	Each response occupies one row. question_code identifies the approved item, while typed value fields preserve atomic storage.

Normalization Process

The design begins with a conceptual flat record containing participant assignment, scenario wording, task instructions, configuration thresholds, repeated system measurements, notification states, workflow outcomes, safety results, and post-task responses. A single wide table would repeat stable facts and create insertion, update, and deletion anomalies. Normalization addresses these weaknesses by separating the record into related tables based on functional dependencies (Codd, 1971; Kumar & Azad, 2017). Each fact is stored in the table that represents it most directly.

First Normal Form (1NF)

The schema satisfies First Normal Form because each table has a key, each row represents one entity or event instance, and each column contains one atomic value. Repeating groups are not stored in numbered columns or comma-separated lists. Multiple tasks, observations, notification events, workflow events, deviations, and post-task responses are represented as separate rows in ScenarioTask, SystemObservation, NotificationEvent, WorkflowEvent, ProtocolDeviation, and PostTaskResponse. This structure is consistent with the relational requirement that attribute values remain atomic (Codd, 1970).

Second Normal Form (2NF)

The schema satisfies Second Normal Form because non-key attributes depend on the complete primary key. Most tables use a single surrogate primary key, which prevents partial dependencies. ScenarioTask is the principal composite-key table. Its sequence_number, required_flag, and applicability_rule describe the complete relationship between Scenario and UserTask and therefore depend on both scenario_id and task_id. Scenario names remain in Scenario, while task instructions remain in UserTask. This separation preserves full-key dependency and prevents repeated descriptive data (Codd, 1971; Kumar & Azad, 2017).

Third Normal Form (3NF)

The schema satisfies Third Normal Form because non-key attributes depend on a key rather than on other non-key attributes. StudySession stores foreign keys to Participant, StudyGroup, Scenario, and ConfigurationVersion instead of repeating participant codes, group descriptions, scenario text, or threshold values. NotificationEvent references ConditionInstance, and

WorkflowEvent references ReportingWorkflow. These relationships remove transitive dependencies, keep stable facts in one logical location, and reduce contradictory updates (Codd, 1971; Kumar & Azad, 2017).

Controlled Denormalization and Derived Values

The logical design should remain normalized during prototype development. Analytical rates, latency measures, and summary classifications should be calculated from source events or materialized in a separate read-only analysis export. T3-to-T6 latency, completion rate, unnecessary-submission rate, and appropriate-decision rate should not be stored redundantly unless measured performance requirements justify denormalization. If a derived value is retained, the calculation version and source-event identifiers should also be retained. This approach protects data integrity while allowing controlled denormalization if later evidence supports it (Ajayi et al., 2025).

Normalization Benefits for the Prototype

Risk in a Flat Design	Normalized Resolution
Participant and group information repeated on every event row	Participant, StudyGroup, and StudySession store assignment facts once and connect through foreign keys.
Scenario instructions copied into each session	Scenario, UserTask, and ScenarioTask preserve reusable definitions and sequence.
Threshold values embedded in observations	ConfigurationVersion stores versioned values once, and StudySession identifies the approved version used.
T0 through T8 stored as separate nullable columns	Event tables record timestamped facts as rows, allowing repeated or missing events to be represented accurately.
Notification outcome overwriting prior state	NotificationEvent retains request, render, suppression, dismissal, mute, and action history.
Workflow status mixed with every field interaction	ReportingWorkflow stores the lifecycle, while WorkflowEvent stores granular steps.
Safety evidence mixed into notification records	SafetyRecoveryEvent remains independent and can record stopping or recovery even when the notification interface fails.
Post-task questions added as new table columns	PostTaskResponse stores approved question codes and responses as rows without altering the session schema.

Keys, Constraints, and Data Integrity

Every table should enforce a primary key, and every foreign key should reference an existing parent row. Required parent relationships should use NOT NULL foreign keys. Optional context, such as a participant action that occurs before a condition exists, may use a nullable condition_id. Unique constraints should protect participant_code, group_name, scenario_code, task_code, configuration version names, and session_code. Controlled vocabularies should be enforced for session status, event type, workflow status, severity, consent status, and recovery status. Timestamps should be stored consistently in Coordinated Universal Time with sufficient precision to support

latency calculations. These constraints support entity integrity, referential integrity, and consistent interpretation of recorded study events.

Deletion behavior should protect research traceability. Parent records used by completed sessions should not be cascade-deleted. ConfigurationVersion, Scenario, and UserTask should normally be retired through status fields rather than physically removed. Session child rows may use restrictive deletion rules during formal collection. Test records may be removed only through an approved reset procedure before the formal dataset is locked.

Privacy and Scope Boundary

The operational database contains coded study identifiers and approved simulated technical context only. The Participant table does not contain names or contact information. ReportingWorkflow does not contain a production ticket identifier because no production ticket is created. The research export should omit internal database identifiers that are not analytically required, preserve only the coded participant identifier, and exclude passwords, client identities, confidential system settings, unrestricted production diagnostics, and private network details.

Implementation Recommendation

The prototype should implement the normalized schema in a relational database management system that supports primary keys, foreign keys, unique constraints, transactions, and parameterized queries. Platform selection should follow confirmation of the application architecture, data-collection workflow, and deployment requirements. A lightweight embedded database may be sufficient for the local, bounded study environment. A client-server platform should be considered only if later requirements include centralized administration, concurrent multi-user access, or production-service integration. The selected system should use transactions for multi-table event writes, link each StudySession to one locked ConfigurationVersion, preserve formal event rows as append-oriented records, and generate de-identified analysis data through controlled views or queries.

References

- Ajayi, W., Okesola, K., Ntezirayao, D., Odo, F., & Akpan, A. U. (2025). Relational database normalization vs denormalization: A performance perspective. *Asian Journal of Electrical Sciences*, 14(2), 53–66. <https://doi.org/10.70112/ajes-2025.14.2.4291>
- Codd, E. F. (1970). A relational model of data for large shared data banks. *Communications of the ACM*, 13(6), 377–387. <https://doi.org/10.1145/362384.362685>
- Codd, E. F. (1971). Further normalization of the data base relational model. IBM Research Report RJ909.
- Kumar, K., & Azad, S. K. (2017). Relational database design: A review. *International Journal of Computer Applications*, 176(6), 14–18. <https://doi.org/10.5120/ijca2017915626>