DATA AND APPLICATIONS

PHASE 3

ER to Relational Model Mapping

Project Phase 3

CS4.301 Data and Applications

Team Members

Tanishi Tyagi (2024111027)
Maithily Bhala (2024117011)
Chandrani Saha (2024113002)

Stage 1: Mapping ER Diagram to Relational Model

The ER diagram from Phase 2 was converted into a relational model by applying systematic mapping rules that translate conceptual design elements into a structured relational schema. This process ensures that all entities, attributes, and relationships are preserved while maintaining data integrity and avoiding redundancy. Each transformation was guided by principles of referential integrity, entity preservation, and normalization readiness.

1. Theoretical Basis and Rationale

The conversion from the ER model to the relational model involves transforming high-level conceptual constructs into implementable tables.

- **Strong Entities:** Each strong entity was mapped to a base relation with its key attribute(s) as the primary key. This allows direct identification of every record.
- **Weak Entities:** Weak entities were converted into separate tables that include a foreign key reference to their owner entity's primary key, ensuring dependency preservation.
- **Composite Attributes:** Composite attributes like `Coordinates(X,Y)` were kept as a single attribute temporarily. These will later be decomposed in 1NF to satisfy atomicity.
- **Multivalued Attributes:** Attributes such as `Known_Aliases` in the `Person` entity remain within the same table at this stage. They will be separated in 1NF to maintain single-valued cells.
- **Subclasses:** Each subclass is mapped to its own table. The primary key of the superclass acts as both the primary and foreign key in the subclass relation.
- **Relationships:** 1:N relationships are represented by placing a foreign key in the "many" side table, while M:N relationships use junction tables with composite primary keys.

2. Tables with Primary and Foreign Keys

Table Name	**Primary Key (PK)**	**Foreign Keys (FK)**	**Notes**
Person	Person_ID	Supervisor_ID → Person.Person_ID (recursive)	Contains multivalued attribute `Known_Aliases` (to be decomposed in 1NF)
Researcher (subclass)	Person_ID (PK, FK)	Person_ID → Person.Person_ID	Subclass of Person

Agent (subclass)	Person_ID (PK, FK)	Person_ID → Person.Person_ID	Subclass of Person
Victim (subclass)	Person_ID (PK, FK)	Person_ID → Person.Person_ID	Subclass of Person
Experiment	Exp_ID	Conducted_By → Person.Person_ID	Conducted by Person
Portal	Portal_ID	Has_Origin, Has_Destination → Location.Location_ID; Links_To → Portal.Portal_ID	Recursive and dual location links
Location	Location_ID	–	Composite attribute `Coordinates(X,Y)` retained for now
Entity	Entity_ID	–	Superclass for entity types
Artifact	Artifact_ID	Found_At → Location.Location_ID	Linked to discovery Location
Report	Report_ID	Authored_By, Verified_By → Person.Person_ID; Documents_Artifact → Artifact.Artifact_ID	Authorship and verification links
Event	Event_ID	Occurs_At → Location.Location_ID; Occurs_Via → Portal.Portal_ID	Event location and portal references
Victim_Record (weak)	Victim_No	Person_ID → Person.Person_ID; Hurt_In → Event.Event_ID	Weak entity dependent on Person and Event
Entity_Appearance	Appearance_ID	Entity_ID → Entity.Entity_ID	Represents appearances of entities
Event_Involves_Entity	(Event_ID, Entity_ID)	Event_ID → Event.Event_ID; Entity_ID → Entity.Entity_ID	Junction table for M:N relationship

Event_Affects_Artifact	(Event_ID, Artifact_ID)	Event_ID → Event.Event_ID; Artifact_ID → Artifact.Artifact_ID	Junction table for M:N relationship
Event_Documented_By_Report	(Event_ID, Report_ID)	Event_ID → Event.Event_ID; Report_ID → Report.Report_ID	Junction table linking reports to events (Event Reported By)

3. Relationship Summary

Relationship	**Type**	**Implementation Method**
Supervision (Person–Person)	1:N	Recursive FK: Person.Supervisor_ID → Person.Person_ID
Conducted_By (Person–Experiment)	1:N	FK in Experiment table referencing Person
Has_Origin / Has_Destination (Portal–Location)	1:N	FKs in Portal referencing Location
Occurs_At / Occurs_Via (Event–Location/Portal)	1:N	FKs in Event referencing Location and Portal
Authored_By / Verified_By (Report–Person)	1:N	FKs in Report referencing Person
Documents_Artifact (Report–Artifact)	1:N	FK in Report referencing Artifact
Hurt_In (Victim_Record–Event)	1:N	FK in Victim_Record referencing Event
Event_Involves_Entity	M:N	Junction table (Event_ID, Entity_ID) with Event_ID → Event.Event_ID; Entity_ID → Entity.Entity_ID
Event_Affects_Artifact	M:N	Junction table (Event_ID, Artifact_ID) with Event_ID → Event.Event_ID; Artifact_ID → Artifact.Artifact_ID
Event_Documented_By_Report (Event Reported By)	M:N	Junction table (Event_ID, Report_ID) with Event_ID → Event.Event_ID; Report_ID → Report.Report_ID

4. Summary of ER to Relational Conversion

This stage ensures that all conceptual entities are faithfully represented as relations with clear key constraints and minimal redundancy.

- Each strong entity has its own table with a unique primary key.
- Weak entities use composite keys including owner foreign keys.
- Subclasses are represented as separate tables referencing their superclass.
- 1:N and 1:1 relationships use foreign keys; M:N relationships use junction tables.
- Multivalued and composite attributes will be normalized in 1NF for atomicity.

Conclusion: The ER-to-relational mapping preserves the semantics of the conceptual model while preparing the schema for the next stages of normalization. All relationships and dependencies are now explicit, ensuring data consistency and providing a strong foundation for normalization in later stages.

STAGE 1

PERSON

Person_ID	Name	Role	Age	Status	Affiliation	Supervisor_ID	Known_Aliases

RESEARCHER

Person_ID	Clearance_Level

AGENT

Person_ID	Success_Rate

VICTIM

Person_ID	Injury_Severity

PSYCHIC_SUBJECT

Person_ID	Ability_Type	Power_Level	Control_Score

EXPERIMENT

Exp_ID	Purpose	Confidentiality	Result	Date	Conducted_By

PORTAL

Portal_ID	Name	Status	Has_Origin	Has_Destination	Links_To	Discovered_On	Coordinate X	Coordinate Y

LOCATION

Location_ID	Name	World_Type	Risk_Level	Description	Links_To	Discovered_On	Coordinates(X,Y)

SURFACE_LOCATION

Location_ID	Population_Density	Proximity_To_Lab

UPSIDEDOWN_LOCATION

Location_ID	Distortion_Level	Hazard_Type

ENTITY

Entity_ID	Name	Species	Threat_Level	Origin_world	First_Sighting

ENTITY APPEARANCE

Appearance_ID	Entity_ID	Event_ID	Start_Time	End_Time

MONSTER

Entity_ID	Aggression_Index

SHADOW_CREATURE

Entity_ID	Corruption_Level	Manifestation_Type

MIND_ENTITY

Entity_ID	Influence_Range	Cognitive_Link_Strength

REPORT

Report_ID	Summary	Verdict	Date	Authored_By	Verified_By	Documents_Artifact

ARTIFACT

Artifact_ID	Name	Type	Anomaly_Level	Found_At

EVENT

Event_ID	Date	Time	Description	Outcome	Severity	Location_ID	Portal_ID

VICTIM RECORD

Victim_No	Person_ID	Hurt_In	Injury_Severity

EVENT_INVOLVES_ENTITY

Event_ID	Entity_ID

EVENT_AFFECTS_ARTIFACT

Event_ID	Artifact_ID

EVENT_DOCUMENTED_BY_REPORT

Event_ID	Report_ID

Stage 2: Conversion to First Normal Form (1NF)

1. Theoretical Background

The First Normal Form (1NF) is the foundational stage of database normalization. It ensures that each attribute in a relation contains only atomic (indivisible) values and that no repeating groups or multivalued attributes exist within any table.

The objectives of 1NF are:

- Ensure every attribute holds a single atomic value.
- Eliminate repeating groups and array-like fields.
- Move multivalued attributes into separate relations.
- Decompose composite attributes into simple components.

Achieving 1NF reduces update anomalies and simplifies querying, and prepares the schema for dependency analysis in 2NF.

2. Conceptual Changes During Conversion

During Stage 1 some composite and multivalued attributes were retained for clarity. In Stage 2 we convert those into atomic form:

- **Composite attributes:** Split into independent attributes so each column stores one atomic value.
- **Multivalued attributes:** Extracted into separate tables where each value occupies its own row and references the parent entity via a foreign key.
- **Repeating groups:** Eliminated by creating new relations or by additional columns only when appropriate (but preferring separate relations).

3. Changes Applied to Your Schema (explicit)

Below are the concrete changes you made and why they are required for 1NF compliance.

- **Location.Coordinates (X,Y) → Coordinate_X, Coordinate_Y.**
  Reason: coordinates are a composite attribute (two pieces of information). In 1NF each attribute must be atomic, so we split them into two columns `Coordinate_X` and `Coordinate_Y`. This makes each field single-valued and independently queryable.
- **Person.Known_Aliases removed from Person and moved to Person_Known_Aliases(Perso Alias).**
  Reason: `Known_Aliases` is multivalued (a person can have 0..n aliases). Storing multiple aliases in one cell violates 1NF. Moving aliases into a separate table where

each alias is one row ensures atomicity and allows indexing, searching, and enforcing uniqueness with the composite key `(Person_ID, Alias)`.

- **Repeating groups / arrays removed from all tables.**
  Reason: Any column that could hold a list/array has been decomposed into separate rows or separate relations so all fields now store single scalar values.

4. Schema Adjustments — concise table view

Change	Description and Justification
Location.Coordinates(X, Y)	Decomposed into `Coordinate_X` and `Coordinate_Y`. Each coordinate is now atomic and can be queried independently.
Person.Known_Aliases	Removed from `Person` and modeled as a separate relation: `Person_Known_Aliases(Person_ID, Alias)`. Each alias is one row; `Person_ID` is a FK to `Person`.
Verification	All other attributes reviewed to ensure no repeating groups; columns now store single values only.

5. Resulting Structural Improvements

- All relations now satisfy atomicity: each column contains indivisible values.
- Multivalued attributes and repeating groups have been eliminated, avoiding ambiguous storage formats.
- The schema is now prepared for partial-dependency analysis in 2NF.
- Queries, updates and integrity constraints become simpler and less error-prone.

6. Short examples of updated relations

- **Location**(`Location_ID (PK)`, `Coordinate_X`, `Coordinate_Y`, `Name`, ...)
- **Person**(`Person_ID (PK)`, `Name`, ...)
  Person_Known_Aliases(`Person_ID (FK)`, `Alias`) PK could be composite (`Person_ID`, `Alias`)

7. Summary and justification

Converting to 1NF enforces atomicity, removes repeating groups and multivalued fields, and improves consistency and queryability. The changes we made (splitting coordinates; moving known aliases to their own relation) are standard 1NF steps and correctly prepare the schema for 2NF and 3NF.

STAGE 1

PERSON

Person_ID	Name	Role	Age	Status	Affiliation	Supervisor_ID

RESEARCHER

Person_ID	Clearance_Level

AGENT

Person_ID	Success_Rate

VICTIM

Person_ID	Injury_Severity

PSYCHIC_SUBJECT

Person_ID	Ability_Type	Power_Level	Control_Score

EXPERIMENT

Exp_ID	Purpose	Confidentiality	Result	Date	Conducted_By

PORTAL

Portal_ID	Name	Status	Has_Origin	Has_Destination	Links_To	Discovered_On	Coordinate X	Coordinate Y

LOCATION

Location_ID	Name	World_Type	Risk_Level	Description	Links_To	Discovered_On	Coordinate X	Coordinate Y

SURFACE_LOCATION

Location_ID	Population_Density	Proximity_To_Lab

UPSIDEDOWN_LOCATION

Location_ID	Distortion_Level	Hazard_Type

ENTITY

Entity_ID	Name	Species	Threat_Level	Origin_world	First_Sighting

ENTITY APPEARANCE

Appearance_ID	Entity_ID	Event_ID	Start_Time	End_Time

MONSTER

Entity_ID	Aggression_Index

SHADOW_CREATURE

Entity_ID	Corruption_Level	Manifestation_Type

MIND_ENTITY

Entity_ID	Influence_Range	Cognitive_Link_Strength

REPORT

Report_ID	Summary	Verdict	Date	Authored_By	Verified_By	Documents_Artifact

ARTIFACT

Artifact_ID	Name	Type	Anomaly_Level	Found_At

EVENT

Event_ID	Date	Time	Description	Outcome	Severity	Location_ID	Portal_ID

VICTIM RECORD

Victim_No	Person_ID	Hurt_In	Injury_Severity

PERSON_KNOWN_ALIAS

Person_ID	Alias

EVENT_INVOLVES_ENTITY

Event_ID	Entity_ID

EVENT_AFFECTS_ARTIFACT

Event_ID	Artifact_ID

EVENT_DOCUMENTED_BY_REPORT

Event_ID	Report_ID

Stage 3: Conversion from 1NF to 2NF

Theoretical Overview

The Second Normal Form (2NF) builds upon the First Normal Form (1NF) by ensuring that:

- The table is already in 1NF (all attributes are atomic and contain indivisible values).
- Every non-key attribute is fully functionally dependent on the entire primary key.
- No partial dependency exists, i.e., no attribute depends on just a part of a composite primary key.

This means that in 2NF, a table must not have attributes that depend only on one part of a composite key. If such dependencies exist, the table should be decomposed so that every attribute is dependent on the whole key.

Analysis of Current Schema

After the 1NF transformation, all attributes in the schema are atomic and properly separated into individual columns or tables. To verify 2NF, we analyzed each relation for the presence of partial dependencies.

- **Strong Entities (e.g., Person, Experiment, Entity, Artifact, Portal, Report, Location):** Each has a single-attribute primary key (such as `Person_ID`, `Exp_ID`, `Entity_ID`, etc.). Since all non-key attributes depend directly and entirely on these single keys, no partial dependency can occur.
- **Weak Entities (e.g., Victim_Record):** The primary key here is composite (`Victim_No`, `Person_ID`, or `Hurt_In`), but every non-key attribute (like `Injury_Severity`) depends on the full composite key, not just one part. Hence, this table also satisfies 2NF.
- **Relationship / Junction Tables (e.g., Event_Involves_Entity, Event_Affects_Artifact, Event_Documented_By_Report):** These tables consist solely of foreign keys as a composite primary key. Since there are no non-key attributes at all, partial dependencies cannot exist. For example:

  $$\text{Event\_Involves\_Entity}(\underline{\text{Event\_ID}}, \underline{\text{Entity\_ID}})$$

  has no additional non-key columns, so it inherently satisfies 2NF.
- **Subclass Tables (Researcher, Agent, Victim, etc.):** Each subclass uses `Person_ID` as both its primary and foreign key. Since each attribute depends fully on that key (e.g., `Rank` or `Level_Of_Access`), these tables are also in 2NF.

Examples and Reasoning

Consider the `Person` table:

$$\text{Person}(\underline{\text{Person\_ID}}, \text{Name}, \text{Role}, \text{Age}, \text{Status}, \text{Affiliation}, \text{Supervisor\_ID})$$

All attributes such as `Name`, Role, and `Status` depend entirely on the unique key `Person_ID`. There are no repeating or partially dependent attributes. Hence, it remains unchanged in 2NF.

Similarly, in `Event_Involves_Entity`:

$$\text{Event\_Involves\_Entity}(\underline{\text{Event\_ID}}, \underline{\text{Entity\_ID}})$$

there are no non-key attributes; thus, the table automatically satisfies 2NF.

For weak entities like `Victim_Record`:

$$\text{Victim\_Record}(\underline{\text{Victim\_No}}, \underline{\text{Person\_ID}}, \text{Injury\_Severity}, \text{Hurt\_In})$$

the attribute `Injury_Severity` depends on the entire combination of the keys, since it represents the injury of a specific person in a specific event. No partial dependency is present.

Conclusion

No changes were required to transition from 1NF to 2NF because:

- Every table already had a single-attribute primary key or a composite key with full functional dependency.
- There were no attributes depending on part of a composite key.
- Junction tables had only keys and no dependent attributes.

Therefore, all relations derived from the ER-to-relational mapping inherently satisfy the conditions of Second Normal Form (2NF).

Summary

- All strong, weak, and subclass entities comply with 2NF.
- No partial dependencies detected.
- The schema remains structurally unchanged after this stage.
- The database is now ready for analysis and conversion to Third Normal Form (3NF).

Stage 4: Conversion from 2NF to 3NF

Theoretical Overview

The Third Normal Form (3NF) extends the principles of 2NF by removing **transitive dependencies**. A relation is in 3NF if:

- It is already in 2NF.
- Every non-key attribute depends directly on the table's primary key, and not on another non-key attribute.

This ensures that each table represents a single concept, eliminates redundancy, and prevents update anomalies caused by indirect dependencies.

Analysis of Current Schema

After confirming all relations satisfied 2NF, each table was analyzed for transitive dependencies. Most entities in the schema—such as `Person`, `Experiment`, `Entity`, `Artifact`, `Portal`, and `Event`—already satisfied 3NF because all their non-key attributes depended directly on their respective primary keys.

However, two parts of the schema required additional review:

1. The `Report` table contained descriptive fields (`Summary`, `Verdict`) that could be isolated to ensure full dependency on the primary key.
2. The `Location` hierarchy contained overlapping subclasses, so dependency paths between `World_Type` and subclass attributes were verified.

Changes Made for 3NF

1. **Decomposition of the Report Table:** The attributes `Summary` and `Verdict` were moved into a new table `Report_Details`. This removed a transitive dependency within `Report` and made each attribute depend solely on the key `Report_ID`.

   Report(<u>Report_ID</u>, Date, Authored_By, Verified_By, Documents_Artifact)
   Report_Details(<u>Report_ID</u>, Summary, Verdict)

   The new `Report_Details` table maintains a one-to-one relationship with `Report` via a shared primary key.

2. **Validation of the Location Hierarchy:** The superclass `Location` and its two subclasses—`Surface_Location` and `UpsideDown_Location`—were analyzed for transitive dependencies. Each attribute was found to depend directly on the key `Location_ID`.

Since subclass-specific attributes (`Population_Density`, `Proximity_To_Lab`, `Distortion_Level`, `Hazard_Type`) were already isolated in their respective subclass tables, no further decomposition was required.

Location(Location_ID, Name, World_Type, Risk_Level, Description, Coordinate_X, Coordinate_Y)

Surface_Location(Location_ID, Population_Density, Proximity_To_Lab)

UpsideDown_Location(Location_ID, Distortion_Level, Hazard_Type)

Therefore, the `Location` hierarchy already satisfied 3NF conditions.

Verification of Other Relations

All remaining entities—including `Person`, `Experiment`, `Entity`, `Artifact`, `Portal`, `Victim_Record`, and the M:N relationship tables—were examined for indirect dependencies. Since all non-key attributes in these tables depend directly on their primary key or composite key, no further normalization was required.

Resulting 3NF Schema Overview

- **Report_Details** was created to separate descriptive report information.
- **Location hierarchy** remains unchanged, as subclassing already removed transitive dependencies.
- All other relations satisfied 3NF without modification.

Why the Design is Fully Normalized

By this stage:

- Each table represents a single entity or relationship type.
- All non-key attributes depend directly and only on the table's primary key.
- There are no transitive dependencies or derived fields remaining.
- Referential integrity is preserved through proper primary and foreign key constraints.

Example Illustration

After normalization:

Report(Report_ID, Date, Authored_By, Verified_By, Documents_Artifact)

$$\text{Report\_Details}(\underline{\text{Report\_ID}}, \text{Summary}, \text{Verdict})$$

The two tables are linked by a one-to-one relationship through `Report_ID`, ensuring that textual details depend only on the report identifier.

Summary of Stage 4

- Verified all relations for transitive dependencies.
- Decomposed `Report` to achieve full 3NF compliance.
- Confirmed `Location` hierarchy already satisfied 3NF requirements.
- Final schema now represents a fully normalized and consistent relational model.

Conclusion

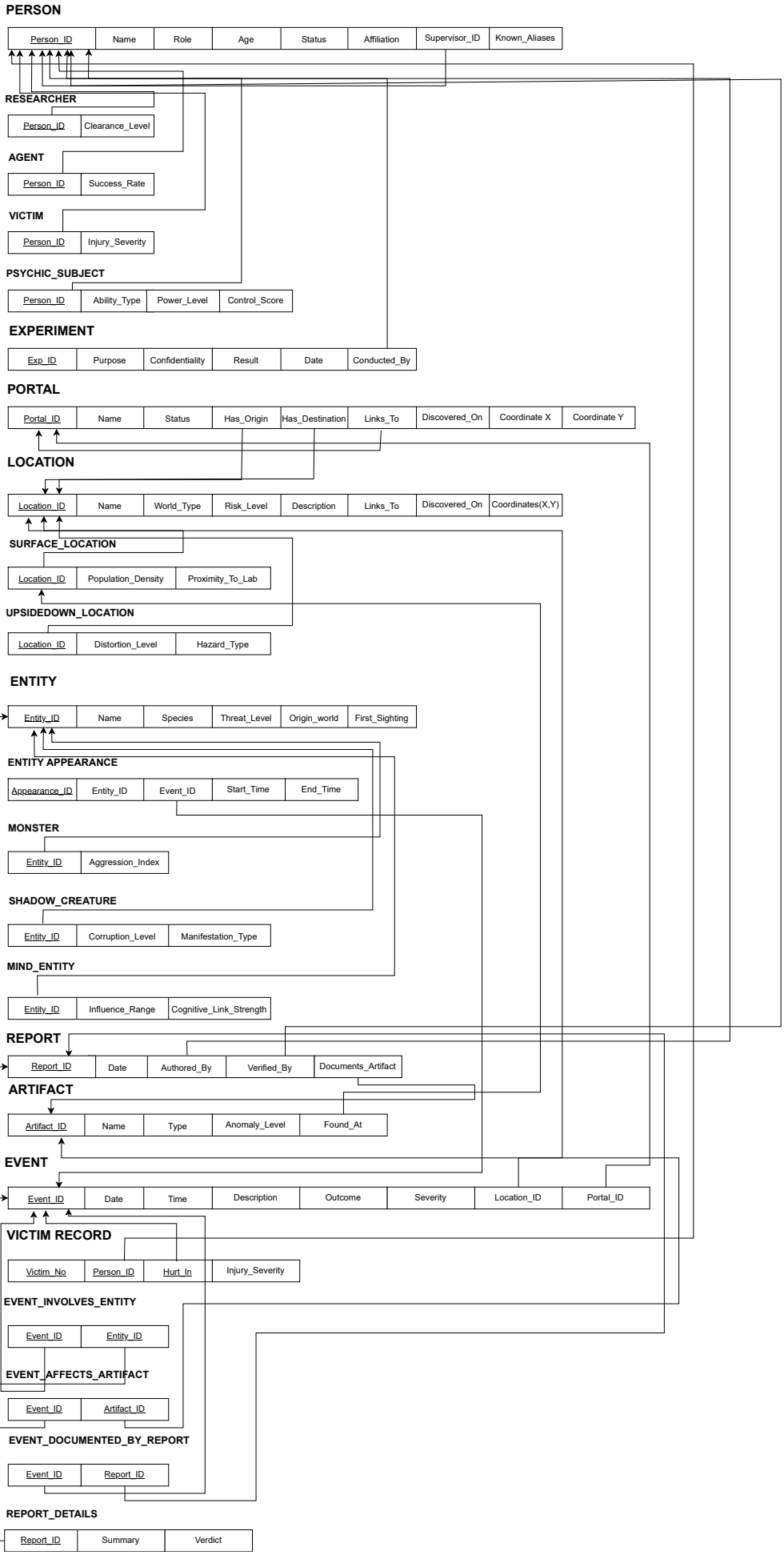
The complete relational model, derived from the initial ER diagram, successfully advanced through normalization from the conceptual stage to Third Normal Form (3NF). Throughout this process:

- Entities and relationships were accurately mapped to relational tables with clear key constraints.
- Multivalued and composite attributes were decomposed during 1NF.
- No partial dependencies were found during 2NF analysis.
- A minor transitive dependency in the `Report` table was resolved during 3NF.

The final 3NF schema is free of redundancy, preserves referential integrity, and ensures that each relation contains attributes solely dependent on its primary key. This schema can now be efficiently implemented in SQL, serving as a reliable, normalized foundation for future extensions and queries.

End of Phase 3 Report — ER to Relational Model and Normalization (1NF–3NF)

STAGE 1


PERSON
Person_ID
Name
Role
Age
Status
Affiliation
Supervisor_ID
Known_Aliases
RESEARCHER
Person_ID
Clearance_Level
AGENT
Person_ID
Success_Rate
VICTIM
Person_ID
Injury_Severity
PSYCHIC_SUBJECT
Person_ID
Ability_Type
Power_Level
Control_Score
EXPERIMENT
Exp_ID
Purpose
Confidentiality
Result
Date
Conducted_By
PORTAL
Portal_ID
Name
Status
Has_Origin
Has_Destination
Links_To
Discovered_On
Coordinate X
Coordinate Y
LOCATION
Location_ID
Name
World_Type
Risk_Level
Description
Links_To
Discovered_On
Coordinates(X,Y)
SURFACE_LOCATION
Location_ID
Population_Density
Proximity_To_Lab
UPSIDEDOWN_LOCATION
Location_ID
Distortion_Level
Hazard_Type
ENTITY
Entity_ID
Name
Species
Threat_Level
Origin_world
First_Sighting
ENTITY APPEARANCE
Appearance_ID
Entity_ID
Event_ID
Start_Time
End_Time
MONSTER
Entity_ID
Aggression_Index
SHADOW_CREATURE
Entity_ID
Corruption_Level
Manifestation_Type
MIND_ENTITY
Entity_ID
Influence_Range
Cognitive_Link_Strength
REPORT
Report_ID
Date
Authored_By
Verified_By
Documents_Artifact
ARTIFACT
Artifact_ID
Name
Type
Anomaly_Level
Found_At
EVENT
Event_ID
Date
Time
Description
Outcome
Severity
Location_ID
Portal_ID
VICTIM RECORD
Victim_No
Person_ID
Hurt_In
Injury_Severity
EVENT_INVOLVES_ENTITY
Event_ID
Entity_ID
EVENT_AFFECTS_ARTIFACT
Event_ID
Artifact_ID
EVENT_DOCUMENTED_BY_REPORT
Event_ID
Report_ID
REPORT_DETAILS
Report_ID
Summary
Verdict