**Phase 3: Logical Model Design**

**📘 1. Entity-Relationship Diagram (ERD) Design**

We began Phase 3 by constructing a comprehensive Entity-Relationship Diagram (ERD) to visually represent the structure and logic of the IT Helpdesk system. The ERD serves as the foundation for the database, guiding how data is stored, linked, and accessed throughout the system. It effectively maps out the relationships between key entities involved in managing support tickets, escalations, and resolutions.

**👥 2. Entity Identification**

Key entities were identified to accurately model the real-world helpdesk environment:

* Employee – The end users who raise support tickets.
* Ticket – The core entity tracking issues, escalation status, priority, and timestamps.
* SupportAgent – The technical personnel assigned to resolve issues.
* ResolutionLog – A history table logging actions, notes, and timestamps for each ticket’s lifecycle.
* Department – To group employees and support agents under organizational units, enabling escalations by department.

**🧱 3. Attribute and Constraint Definition**

Each entity includes carefully defined attributes with appropriate data types (e.g., VARCHAR2, DATE, NUMBER) and integrity constraints:

* Primary Keys (PKs) uniquely identify each record.
* Foreign Keys (FKs) enforce referential integrity between related entities.
* NOT NULL, UNIQUE, and CHECK constraints ensure data validity and prevent anomalies.

For example, the Ticket entity includes attributes such as ticket\_id (PK), issue\_description, creation\_date, priority, and status, with FK references to Employee and SupportAgent.

**🔗 4. Relationship Mapping**

Relationships were established between entities using primary and foreign keys:

* One-to-Many between Employee → Ticket (an employee can create multiple tickets).
* One-to-Many between SupportAgent → Ticket (an agent can be assigned multiple tickets).
* One-to-Many between Ticket → ResolutionLog (a ticket can have multiple resolution entries).
* One-to-Many between Department → Employee and Department → SupportAgent.

These relationships support effective joins and queries for advanced reporting and escalation routing.

**🧹 5. Normalization to 3NF**

Normalization was applied up to Third Normal Form (3NF) to eliminate redundancy and improve data integrity:

* 1NF ensured atomic values in each field.
* 2NF removed partial dependencies by isolating repeating groups.
* 3NF removed transitive dependencies, especially in ticket and resolution logging.

As a result, the design is efficient, scalable, and free from unnecessary duplication of data.

**🎯 6. Functional Objectives**

The logical model is tailored to support:

* Ticket escalation, based on agent response time or issue criticality.
* Resolution tracking, using time-stamped logs.
* Agent assignments, allowing multiple support staff across departments.
* Auditability, to later support PL/SQL triggers, logging, and restrictions in Phase 4.
* Packages to encapsulate auditing utilities and reporting functions