# Smart-IMS Relational Schema Design

*CS4092 Final Project – Shivam Sinay Kharangate*

## ***Entity Analysis***

Strong Entities (Independent Existence)

- **Category - Product categorization**
- **Product - Core inventory items**
- **Warehouse - Storage locations**
- **Supplier - Vendor information**
- **User - System users**
- **Query_Log - Natural language query tracking**
- **MCP_System - Model Context Protocol server**
- **LLM_System - Ollama/Gemma integration**

Weak Entities (Dependent Existence)

- **Inventory - Depends on both Product and Warehouse**
- **Supply - Junction entity for Supplier-Product M:N relationship**
- **Transaction_Log - Audit trail dependent on system operations**

## ***Relationship Analysis***

1. 1:N Relationships (Foreign Key Integration)
    - **Category ↔ Product (1:N with ED from Product side)**
    - **Warehouse ↔ Inventory (1:N with ED from Inventory side)**
    - **Product ↔ Inventory (1:N with ED from Inventory side)**
    - **User ↔ Query_Log (1:N with ED from Query_Log side)**
    - **MCP_System ↔ Query_Log (1:N with ED from Query_Log side)**
    - **Inventory ↔ Transaction_Log (1:N with ED from Transaction_Log side)**

2. M:N Relationships (Separate Junction Tables)
    - **Supplier ↔ Product (M:N resolved via Supply entity)**

3. 1:1 Relationships
    - **MCP_System ↔ LLM_System (1:1 with ED from both sides - merge into single table or separate with FK)**

## ***Final Relational Schemas***

1. Category Schema

```
Category(
    id INTEGER PRIMARY KEY,
    name VARCHAR(255) NOT NULL
)
```

**Notes:** Basic entity with only simple attributes (Simple strong entity conversion)

2. Warehouse Schema

```
Warehouse(
    id INTEGER PRIMARY KEY,
    location VARCHAR(255) NOT NULL
)
```

**Notes:** Basic entity with only simple attributes (Simple strong entity conversion)

3. Supplier Schema

```
Supplier(
    id INTEGER PRIMARY KEY,
    name VARCHAR(255) NOT NULL,
    contact VARCHAR(255)
)
```

**Notes:** Contact is nullable (optional attribute) (Simple strong entity conversion)

4. Product Schema

```
Product(
    id INTEGER PRIMARY KEY,
    name VARCHAR(255) NOT NULL,
    category_id INTEGER NOT NULL,
    price DECIMAL(10,2) NOT NULL,
    reorder_level INTEGER NOT NULL,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (category_id) REFERENCES Category(id)
)
```

**Notes:** Foreign key embedded due to existential dependency (1:N relationship with ED (Product must belong to Category))

## 5. Inventory Schema (Weak Entity)

```
Inventory(
    product_id INTEGER,
    warehouse_id INTEGER,
    quantity INTEGER NOT NULL DEFAULT 0,
    last_updated TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    PRIMARY KEY (product_id, warehouse_id),
    FOREIGN KEY (product_id) REFERENCES Product(id),
    FOREIGN KEY (warehouse_id) REFERENCES Warehouse(id)
)
```

**Notes:** Composite PK ensures unique product-warehouse combinations (Weak entity with composite primary key from owner entities)

## 6. Supply Schema (Junction Table)

```
Supply(
    supplier_id INTEGER,
    product_id INTEGER,
    cost DECIMAL(10,2) NOT NULL,
    supply_date DATE NOT NULL,
    quantity INTEGER NOT NULL,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    PRIMARY KEY (supplier_id, product_id, supply_date),
    FOREIGN KEY (supplier_id) REFERENCES Supplier(id),
    FOREIGN KEY (product_id) REFERENCES Product(id)
)
```

**Notes:** Composite PK allows multiple supplies from same supplier-product pair on different dates (M:N relationship resolution)

## 7. User Schema

```
User(
    id INTEGER PRIMARY KEY,
    username VARCHAR(100) NOT NULL UNIQUE,
    role VARCHAR(50) NOT NULL DEFAULT 'user'
)
```

**Notes:** System entity for authentication and access control (Simple strong entity conversion)

## 8. Query_Log Schema

```
Query_Log(
    id INTEGER PRIMARY KEY,
    user_id INTEGER NOT NULL,
    query_text TEXT NOT NULL,
    generated_sql TEXT,
    timestamp TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (user_id) REFERENCES User(id)
)
```

**Notes:** Foreign key embedded due to existential dependency (1:N relationship with ED (Query must be associated with User))

## 9. MCP_System Schema

```
MCP_System(
    id INTEGER PRIMARY KEY,
    server_url VARCHAR(255) NOT NULL,
    status VARCHAR(50) NOT NULL DEFAULT 'inactive'
)
```

**Notes:** System entity representing Model Context Protocol server (Simple strong entity conversion)

## 10. LLM_System Schema

```
LLM_System(
    id INTEGER PRIMARY KEY,
    mcp_system_id INTEGER NOT NULL UNIQUE,
    model_name VARCHAR(100) NOT NULL,
    version VARCHAR(50),
    endpoint VARCHAR(255) NOT NULL,
    status VARCHAR(50) NOT NULL DEFAULT 'inactive',
    FOREIGN KEY (mcp_system_id) REFERENCES MCP_System(id)
)
```

**Notes:** Could be merged with MCP_System, but kept separate for clarity (1:1 relationship with ED from LLM_System side )

## 11. Transaction_Log Schema (Audit Trail)

```
Transaction_Log(
    id INTEGER PRIMARY KEY,
    inventory_product_id INTEGER NOT NULL,
    inventory_warehouse_id INTEGER NOT NULL,
    user_id INTEGER,
    transaction_type VARCHAR(50) NOT NULL,
    old_quantity INTEGER,
    new_quantity INTEGER NOT NULL,
    timestamp TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (inventory_product_id, inventory_warehouse_id)
        REFERENCES Inventory(product_id, warehouse_id),
    FOREIGN KEY (user_id) REFERENCES User(id)
)
```

**Notes:** References composite PK of Inventory table (1:N relationship with ED from Transaction_Log side)

## 12. MCP_Query_Processing Schema (Junction Table)

```
MCP_Query_Processing(
    mcp_system_id INTEGER,
    query_log_id INTEGER,
    processing_time_ms INTEGER,
    success_status BOOLEAN DEFAULT true,
    PRIMARY KEY (mcp_system_id, query_log_id),
    FOREIGN KEY (mcp_system_id) REFERENCES MCP_System(id),
    FOREIGN KEY (query_log_id) REFERENCES Query_Log(id)
)
```

**Notes:** Tracks which MCP system processed which queries ( M:N relationship resolution (though typically 1:N in practice))

## ***References***

Microsoft Visual Studio Code – GitHub Copilot: Utilized for data collection for project deliverables to assess correct and full content information and summarize notes for better understanding and document formatting.