Fire Station Database Change Log

MEET AND KARTIK

March 24, 2025

Update - 2025-02-12 14:30:00

Changes Implemented

- Changed `ON DELETE SET NULL` to `ON DELETE CASCADE` for foreign keys referencing `FireStation`.
- Added a `Status` field to the `FireStation` table to manage active and inactive stations.

Details

Foreign Key Deletion Policy
Previously, deleting a fire station left orphaned records. Now, with `ON DELETE CASCADE`, related vehicles and staff records are also deleted.

Status Field in FireStation Table
A new `Status` column was added to indicate whether a fire station is active or inactive, preventing inactive stations from receiving new assignments.

Update - 2025-02-12 16:45:00

Changes Implemented

- Modified `SupplierItems` table to allow suppliers to provide the same item multiple times.
- Modified `StaffShift` table to allow staff members to take the same shift on different days.
- Modified `Vehicle` table to ensure that each vehicle has a unique Type-Model combination and proper foreign key constraints.

Details

Supplier Items Table Fix
Previously, a supplier could not provide the same item multiple times due to a composite primary key. Now, a unique `Supply_ID` and `Supply_Date` fields track each supply separately.

Updated Table:

```
CREATE TABLE SupplierItems (
  Supply_ID INT AUTO_INCREMENT PRIMARY KEY,
  Supplier_ID INT,
  Item_Name VARCHAR(255) NOT NULL,
  Quantity INT NOT NULL CHECK (Quantity > 0),
  Supply_Date DATETIME DEFAULT CURRENT_TIMESTAMP,
  FOREIGN KEY (Supplier_ID) REFERENCES Supplier(Supplier_ID) ON DELETE CASCADE
);
```

Staff Shift Table Fix
Previously, a staff member could not be assigned the same shift more than once. Now, a unique `Shift_ID` and `Shift_Date` fields allow tracking of shift schedules over multiple days.

Updated Table:

```
CREATE TABLE StaffShift (
  Shift_ID INT AUTO_INCREMENT PRIMARY KEY,
  Staff_ID INT,
  Shift ENUM('Morning', 'Evening', 'Night') NOT NULL,
  Shift_Date DATE NOT NULL,
  FOREIGN KEY (Staff_ID) REFERENCES Staff(Staff_ID) ON DELETE CASCADE
);
```

Vehicle Table Fix

Previously, the schema did not enforce unique vehicle types and models properly. This could cause redundant or conflicting vehicle entries. The schema has now been improved to reference a separate `VehicleModel` table.

Updated Tables:

```
CREATE TABLE VehicleModel (
  Type VARCHAR(255) NOT NULL,
  Model_No VARCHAR(255) NOT NULL,
  Water_Capacity INT CHECK (Water_Capacity >= 0),
  PRIMARY KEY (Type, Model_No)
);

CREATE TABLE Vehicle (
  Vehicle_ID INT AUTO_INCREMENT PRIMARY KEY,
  Type VARCHAR(255) NOT NULL,
  Model_No VARCHAR(255) NOT NULL,
  Status ENUM('Available', 'In Use', 'Under Maintenance') NOT NULL,
  Station_ID INT,
  Last_Maintenance_Date DATETIME DEFAULT CURRENT_TIMESTAMP,
  FOREIGN KEY (Type, Model_No) REFERENCES VehicleModel(Type, Model_No) ON DELETE CASCADE,
  FOREIGN KEY (Station_ID) REFERENCES FireStation(Station_ID) ON DELETE SET NULL
);
```

Update Staff Database- March 24, 2025

Changes Implemented

- Added unique constraint to `StaffShift(Staff_ID, Shift, Shift_Date)` to prevent duplicate shifts
- Created junction tables `ReportVehicle` and `ReportStaff` to handle multiple assignments
- Removed direct assignment fields from `Report` table

Details

Staff Shift Table Enhancement

Added a composite unique constraint to prevent duplicate shift assignments while allowing:

- Same staff member to work different shifts on same day
- Same shift to be assigned to different staff on same day
- Same staff to work same shift on different days

Updated Table:

```
CREATE TABLE StaffShift (
  Shift_ID INT AUTO_INCREMENT PRIMARY KEY,
  Staff_ID INT,
  Shift ENUM('Morning', 'Evening', 'Night') NOT NULL,
  Shift_Date DATE NOT NULL,
```

```
  FOREIGN KEY (Staff_ID) REFERENCES Staff(Staff_ID) ON DELETE CASCADE,
  UNIQUE (Staff_ID, Shift, Shift_Date) -- Prevent duplicate shifts
);
```

Report Assignment Restructuring

The previous design limited reports to single vehicle/staff assignments. The new junction tables enable:

- Multiple vehicles to be assigned to a report
- Multiple staff members to be assigned to a report
- Better tracking of resource allocation

New Tables:

```
CREATE TABLE ReportVehicle (
  Report_ID INT,
  Vehicle_ID INT,
  PRIMARY KEY (Report_ID, Vehicle_ID),
  FOREIGN KEY (Report_ID) REFERENCES Report(Report_ID) ON DELETE CASCADE,
  FOREIGN KEY (Vehicle_ID) REFERENCES Vehicle(Vehicle_ID) ON DELETE CASCADE
);

CREATE TABLE ReportStaff (
  Report_ID INT,
  Staff_ID INT,
  PRIMARY KEY (Report_ID, Staff_ID),
  FOREIGN KEY (Report_ID) REFERENCES Report(Report_ID) ON DELETE CASCADE,
  FOREIGN KEY (Staff_ID) REFERENCES Staff(Staff_ID) ON DELETE CASCADE
);
```

Simplified Report Table:

```
CREATE TABLE Report (
  Report_ID INT AUTO_INCREMENT PRIMARY KEY,
  Street_Address VARCHAR(255) NOT NULL,
  City VARCHAR(255) NOT NULL,
  State VARCHAR(255) NOT NULL,
  Pincode VARCHAR(255) NOT NULL,
  Description TEXT NOT NULL,
  Report_Date_Time DATETIME DEFAULT CURRENT_TIMESTAMP,
  Severity_Level VARCHAR(255) NOT NULL,
  User_ID INT,
  Action_Taken TEXT,
  Action_Date_Time DATETIME,
  Admin_ID INT,
  FOREIGN KEY (User_ID) REFERENCES User(User_ID),
  FOREIGN KEY (Admin_ID) REFERENCES Admin(Admin_ID)
);
```

Impact of Changes

- More flexible shift management without duplicate assignments
- Support for complex emergency scenarios requiring multiple resources
- Cleaner database design with proper many-to-many relationships
- Maintained data integrity through proper foreign key constraints

Update Report Database- March 24, 2025

Problem Identification

The original `Report` table had a transitive dependency issue:

```
CREATE TABLE Report (
  Report_ID INT PRIMARY KEY,
  Street_Address VARCHAR(255),
  City VARCHAR(255),
  State VARCHAR(255),
  Pincode INT,
  Description TEXT
);
```

Key Issues

- **Primary Key**: Report_ID directly determines Pincode
- **Transitive Dependency**:
  - Pincode determines City and State
  - Therefore, City/State indirectly depend on Report_ID
- **Violation**: This structure breaks 3NF rules

Consequences

- Data duplication for same Pincode locations
- Update anomalies when city/state changes
- Potential data inconsistencies

Solution

We normalize the structure by separating location data:

```
CREATE TABLE Location (
  Pincode INT PRIMARY KEY,
  City VARCHAR(255),
  State VARCHAR(255)
);

CREATE TABLE Report (
  Report_ID INT PRIMARY KEY,
  Street_Address VARCHAR(255),
  Pincode INT,
  Description TEXT,
  FOREIGN KEY (Pincode) REFERENCES Location(Pincode)
);
```

Implementation Methods

Method 1: Check-Then-Insert

1. Verify Pincode existence in Location
2. Insert new Location if needed
3. Create Report record

Example SQL:

```
-- Check first
SELECT 1 FROM Location WHERE Pincode = 100001;

-- Insert if missing
INSERT INTO Location VALUES (100001, 'Mumbai', 'MH');

-- Create report
INSERT INTO Report VALUES (101, 'Main St', 100001, 'Fire incident');
```

Method 2: Atomic Transaction

Single atomic operation:

```
BEGIN TRANSACTION;
INSERT OR IGNORE INTO Location VALUES (100001, 'Mumbai', 'MH');
INSERT INTO Report VALUES (101, 'Main St', 100001, 'Fire incident');
COMMIT;
```

Update Firestation Database- March 24, 2025

Initial Schema Assessment

The original schema consists of two tables:

```
-- Create FireStation table
CREATE TABLE FireStation (
  Station_ID INT AUTO_INCREMENT PRIMARY KEY,
  Name VARCHAR(255) NOT NULL,
  Contact_Number VARCHAR(20) NOT NULL UNIQUE,
  Total_Staff INT DEFAULT 0 CHECK (Total_Staff >= 0),
  Total_Vehicles INT DEFAULT 0 CHECK (Total_Vehicles >= 0)
);

-- Create FireStationLocation table (3NF decomposition)
CREATE TABLE FireStationLocation (
  Location VARCHAR(255) PRIMARY KEY,
  Station_ID INT UNIQUE,
  FOREIGN KEY (Station_ID) REFERENCES FireStation(Station_ID) ON DELETE CASCADE
);
```

Optimized Schema Design

```
CREATE TABLE PincodeMapping (
  Pincode INT PRIMARY KEY,
  City VARCHAR(255) NOT NULL,
  State VARCHAR(255) NOT NULL
);

CREATE TABLE StationLocation (
  Location_ID INT AUTO_INCREMENT PRIMARY KEY,
  Pincode INT NOT NULL,
  Street_Address VARCHAR(255) NOT NULL,
  Landmark VARCHAR(255),
  Latitude DECIMAL(10, 8),
  Longitude DECIMAL(11, 8),
  FOREIGN KEY (Pincode) REFERENCES PincodeMapping(Pincode)
);
```

```
CREATE TABLE FireStation (
  Station_ID INT AUTO_INCREMENT PRIMARY KEY,
  Name VARCHAR(255) NOT NULL,
  Location_ID INT NOT NULL,
  Status ENUM('Active','Inactive','Renovation') DEFAULT 'Active',
  Establishment_Date DATE,
  Capacity INT,
  FOREIGN KEY (Location_ID) REFERENCES StationLocation(Location_ID)
);

CREATE TABLE StationContact (
  Contact_ID INT AUTO_INCREMENT PRIMARY KEY,
  Station_ID INT NOT NULL,
  Contact_Type ENUM('Primary','Secondary','Emergency'),
  Contact_Value VARCHAR(50) NOT NULL,
  FOREIGN KEY (Station_ID) REFERENCES FireStation(Station_ID)
);
```

Update Supplier Database - March 24, 2025

Original Schema

```
CREATE TABLE Supplier (
  Supplier_ID INT AUTO_INCREMENT PRIMARY KEY,
  Name VARCHAR(255) NOT NULL,
  Contact VARCHAR(255) NOT NULL,
  Email VARCHAR(255) NOT NULL,
  Address VARCHAR(255) NOT NULL
);

CREATE TABLE SupplierItems (
  Supplier_ID INT,
  Item_Name VARCHAR(255) NOT NULL,
  PRIMARY KEY (Supplier_ID, Item_Name),
  FOREIGN KEY (Supplier_ID) REFERENCES Supplier(Supplier_ID) ON DELETE CASCADE
);
```

Recommended Improved Schema

```
CREATE TABLE Supplier (
  Supplier_ID INT AUTO_INCREMENT PRIMARY KEY,
  Name VARCHAR(255) NOT NULL,
  Contact_Phone VARCHAR(20) NOT NULL,
  Email VARCHAR(255),
  Address VARCHAR(255)
);

CREATE TABLE Item (
  Item_ID INT AUTO_INCREMENT PRIMARY KEY,
  Name VARCHAR(255) NOT NULL,
  Category VARCHAR(100)
);

CREATE TABLE SupplierItem (
  Supplier_ID INT,
  Item_ID INT,
  Price DECIMAL(10,2),
  PRIMARY KEY (Supplier_ID, Item_ID),
```

```
  FOREIGN KEY (Supplier_ID) REFERENCES Supplier(Supplier_ID) ON DELETE CASCADE,
  FOREIGN KEY (Item_ID) REFERENCES Item(Item_ID) ON DELETE CASCADE
);
```

Improvements in New Schema

- **Better Item Management**:
  - Items now have their own table with unique IDs
  - Added category classification
- **Enhanced Relationship Tracking**:
  - Price information added to supplier-item relationship
  - Proper foreign key constraints
- **Data Integrity**:
  - More specific data types (e.g., DECIMAL for price)
  - Clearer field naming (Contact_Phone vs Contact)

Update Inventory Database- March 24, 2025

Current Schema Assessment

Original Inventory Table

```
CREATE TABLE Inventory (
  Inventory_ID INT AUTO_INCREMENT PRIMARY KEY,
  Item_Name VARCHAR(255) NOT NULL,
  Quantity INT NOT NULL,
  Station_ID INT,
  Supplier_ID INT,
  Last_Updated DATETIME DEFAULT CURRENT_TIMESTAMP,
  FOREIGN KEY (Station_ID) REFERENCES FireStation(Station_ID),
  FOREIGN KEY (Supplier_ID) REFERENCES Supplier(Supplier_ID)
);
```

Original EquipmentUsage Table

```
CREATE TABLE EquipmentUsage (
  Usage_ID INT AUTO_INCREMENT PRIMARY KEY,
  Inventory_ID INT,
  Used_Quantity INT NOT NULL,
  Date_Used DATETIME DEFAULT CURRENT_TIMESTAMP,
  Purpose VARCHAR(255) NOT NULL,
  Staff_ID INT,
  FOREIGN KEY (Inventory_ID) REFERENCES Inventory(Inventory_ID),
  FOREIGN KEY (Staff_ID) REFERENCES Staff(Staff_ID)
);
```

Normalization Issues

0.0.1 Third Normal Form (3NF) Violations

- **Inventory Table**:
  - Free-text `Item_Name` should reference an items catalog
  - `Supplier_ID` may create incorrect dependencies
- **EquipmentUsage Table**:
  - Free-text `Purpose` field should use controlled values

Improved Schema Design

Item Catalog Table

```
CREATE TABLE Item (
  Item_ID INT AUTO_INCREMENT PRIMARY KEY,
  Name VARCHAR(255) NOT NULL UNIQUE,
  Description TEXT,
  Category ENUM('Protective Gear', 'Tools', 'Medical',
               'Fire Suppression', 'Other') NOT NULL,
  Unit_Type ENUM('Each', 'Box', 'Pair', 'Set') NOT NULL DEFAULT 'Each'
);
```

Normalized Inventory Table

```
CREATE TABLE Inventory (
  Inventory_ID INT AUTO_INCREMENT PRIMARY KEY,
  Item_ID INT NOT NULL,
  Station_ID INT NOT NULL,
  Quantity INT NOT NULL CHECK (Quantity >= 0),
  Min_Stock_Level INT NOT NULL DEFAULT 0,
  Last_Updated DATETIME DEFAULT CURRENT_TIMESTAMP,
  FOREIGN KEY (Item_ID) REFERENCES Item(Item_ID),
  FOREIGN KEY (Station_ID) REFERENCES FireStation(Station_ID),
  UNIQUE (Item_ID, Station_ID)
);
```

Enhanced EquipmentUsage Table

```
CREATE TABLE EquipmentUsage (
  Usage_ID INT AUTO_INCREMENT PRIMARY KEY,
  Inventory_ID INT NOT NULL,
  Used_Quantity INT NOT NULL CHECK (Used_Quantity > 0),
  Date_Used DATETIME DEFAULT CURRENT_TIMESTAMP,
  Purpose ENUM('Training', 'Emergency', 'Maintenance',
              'Inspection', 'Other') NOT NULL,
  Staff_ID INT NOT NULL,
  Notes TEXT,
  FOREIGN KEY (Inventory_ID) REFERENCES Inventory(Inventory_ID),
  FOREIGN KEY (Staff_ID) REFERENCES Staff(Staff_ID)
);
```

Normalization Compliance

Key Improvements

- **Eliminated Free-Text Fields**:
  - Replaced with reference tables and ENUM types
- **Enhanced Data Integrity**:
  - Added CHECK constraints for quantities
  - UNIQUE constraint for inventory items
- **Better Tracking**:
  - Standardized categories and purposes
  - Added minimum stock level alerts

Simplified Alternative Schema

For implementations needing fewer tables:

```
CREATE TABLE InventoryItem (
  Item_ID INT AUTO_INCREMENT PRIMARY KEY,
  Item_Name VARCHAR(255) NOT NULL UNIQUE,
  Category ENUM('Protective Gear', 'Tools', 'Medical',
               'Fire Suppression', 'Other') NOT NULL
);

CREATE TABLE StationInventory (
  Inventory_ID INT AUTO_INCREMENT PRIMARY KEY,
  Item_ID INT NOT NULL,
  Station_ID INT NOT NULL,
  Quantity INT NOT NULL CHECK (Quantity >= 0),
  FOREIGN KEY (Item_ID) REFERENCES InventoryItem(Item_ID),
  FOREIGN KEY (Station_ID) REFERENCES FireStation(Station_ID)
);
```

Update Vehicle Maintainance Database- March 24, 2025

Optimized Schema Design

Enhanced Maintenance Table

```
CREATE TABLE Maintenance (
  Maintenance_ID INT AUTO_INCREMENT PRIMARY KEY,
  Vehicle_ID INT NOT NULL,
  Maintenance_Type ENUM('Routine', 'Repair', 'Inspection',
                        'Tire Change', 'Brake Service') NOT NULL,
  Date_Performed DATETIME DEFAULT CURRENT_TIMESTAMP,
  Cost DECIMAL(10,2) NOT NULL CHECK (Cost >= 0),
  Performed_By INT NOT NULL,
  Description TEXT,
  Next_Service_Date DATE,
  FOREIGN KEY (Vehicle_ID) REFERENCES Vehicle(Vehicle_ID) ON DELETE CASCADE,
  FOREIGN KEY (Performed_By) REFERENCES Staff(Staff_ID)
);
```

Enhanced FuelLog Table

```
CREATE TABLE FuelLog (
  Fuel_ID INT AUTO_INCREMENT PRIMARY KEY,
  Vehicle_ID INT NOT NULL,
  Date DATETIME DEFAULT CURRENT_TIMESTAMP,
  Fuel_Amount DECIMAL(10,2) NOT NULL CHECK (Fuel_Amount > 0),
  Cost_Per_Unit DECIMAL(10,2) NOT NULL CHECK (Cost_Per_Unit > 0),
  Total_Cost DECIMAL(10,2) GENERATED ALWAYS AS
    (Fuel_Amount * Cost_Per_Unit) STORED,
  Odometer_Reading INT NOT NULL,
  Fuel_Type ENUM('Diesel', 'Gasoline', 'Electric', 'Hybrid') NOT NULL,
  Station_ID INT,
  FOREIGN KEY (Vehicle_ID) REFERENCES Vehicle(Vehicle_ID) ON DELETE CASCADE,
  FOREIGN KEY (Station_ID) REFERENCES FireStation(Station_ID)
);
```