**Museum Database Design Documentation**

**Author**: Karapet Tashchyan  
**File Name**: SQL\_Karapet\_Tashchyan\_FinalTask\_Museum\_Combined.docx

## **1. Business Description**

### **1.1 Business Background**

Museums play a vital role in preserving cultural, historical, and scientific artifacts. These institutions manage vast collections of items including artworks, historical objects, and specimens. As these collections grow and exhibitions become more complex—often involving digital components—museums require efficient systems for cataloging, storing, and managing data.

### **1.2 Problems. Current Situation**

Currently, many museums rely on scattered, sometimes manual, recordkeeping processes for items, exhibitions, and visitor tracking. This can result in data inconsistency, lack of traceability, difficulty managing exhibits, and challenges in reporting and analytics.

### **1.3 The Benefits of Implementing a Database. Project Vision**

The goal of this project is to implement a structured relational database that provides:

* A centralized system to manage item records, storage status, and exhibition participation
* Improved tracking of visitors and employees
* Support for both in-person and online exhibitions
* Enhanced data accuracy, retrieval, and reporting capabilities

## **2. Model Description**

### **2.1 Definitions & Acronyms**

* **Item** – Any object in the museum's collection (artwork, artifact, specimen, etc.)
* **Exhibition** – An event or display where items are showcased
* **Inventory** – A record of where and how an item is stored
* **Visitor** – A person who visits the museum
* **Employee** – A staff member working at the museum
* **ItemExhibition** – A linking table representing the many-to-many relationship between items and exhibitions

### **2.2 Logical Scheme**

****

### 

**2.3 Objects**

| **Table Name** | **Field name** | **Field Description** | **Data Type** |
| --- | --- | --- | --- |
| Item | Item\_id | PK(Unique identifier for Item) | Int |
| title | Title of Item(Not Null) | Varchar(100) |
| description | Information about the item(Not Null) | Text |
| item\_type | Item type specification(Not Null) | ENUM(artwork,artifact,specimen,historical object) |
| date\_acquired | Date when the item was bought or acquired by the museum(Not Null) | Date |

**Item** – Stores details of each unique item in the museum's collection, such as title, description, origin, and type.

| **Table Name** | **Field name** | **Field Description** | **Data Type** |
| --- | --- | --- | --- |
| Inventory | inventory\_id | PK(Unique identifier for inventory records) | Int |
| item\_id | FK(Item\_id of the item located in specific inventory)(Not Null) | Int |
| location | Location of the inventory(Not Null) | Varchar(20) |
| status | Current status of the item,(Not Null) | ENUM(artwork,artifact,specimen,historical object) |
| last\_checked | Last inspection date | Date |

**Inventory** – Tracks where items are physically located and their current condition or storage status.

| **Table Name** | **Field name** | **Field Description** | **Data Type** |
| --- | --- | --- | --- |
| Exhibition | exhibition\_id | PK(Unique identifier for exhibitions) | Int |
| name | Name of the exhibition,Not Null | Varchar(100) |
| description | Information about exhibition,Not Null | Text |
| start\_date | Start date of the exhibition,Not Null | Date |
| end\_date | End date of the exhibition,Not Null | Date |
| is\_online | Online exhibition or offline, Not Null | Boolean |

**Employee** – A staff member working at the museum

| **Table Name** | **Field name** | **Field Description** | **Data Type** |
| --- | --- | --- | --- |
| ItemExhibition | Item\_id | FK linking to an item that appears in exhibition (PK) | Int |
| Exhibition\_id | FK linking to an exhibition that includes the item (PK) | Int |

**ItemExhibition** – A linking table representing the many-to-many relationship between items and exhibitions

| **Table Name** | **Field name** | **Field Description** | **Data Type** |
| --- | --- | --- | --- |
| Visitor | Visitor\_id | PK(Unique identifier for visitor) | Int |
| First\_name | First name of a visitor,Not Null | Varchar(20) |
| Last\_name | Last name of a visitor,,Not Null | Varchar(20) |
| email | email of a visitor | Varchar(20) |
| Visit\_date | Visit date of a visitor,Not Null | Date |
| purpose | Purpose of a visit | Varchar(20) |

**Visitor** – A person who visits the museum

| **Table Name** | **Field name** | **Field Description** | **Data Type** |
| --- | --- | --- | --- |
| Employee | Employee\_id | PK(Unique identifier for staff members of a museum) | Int |
| first\_name | FK(Item\_id of the item located in specific inventory)(Not Null) | Int |
| last\_name | Location of the inventory(Not Null) | Varchar(20) |
| position | Current status of the item,(Not Null) | Date |
| phone\_number | Last inspection date | Varchar(20) |

**Employee** – A staff member working at the museum

## 

## **Comments on Table Relationships**

* **Item to Inventory**: One item can have multiple inventory entries, especially over time if it is moved or its status changes.
* **Item to ItemExhibition**: One item may appear in several exhibitions. The linking table ensures proper many-to-many mapping.
* **Exhibition to ItemExhibition**: One exhibition can include many items. This relationship is necessary for display planning.
* **Visitor and Employee**: These tables are standalone, capturing essential info for operational and administrative use.
* **All foreign keys** enforce referential integrity and ensure data consistency throughout the schema.

## **Step-by-Step Approach to Modeling**

### **Step 1: Identify Main Entities**

Based on the business requirements, the following entities were identified:

* Item
* Exhibition
* Inventory
* Visitor
* Employee
* ItemExhibition (junction table)

### **Step 2: Define Relationships**

* One-to-Many: Item → Inventory
* Many-to-Many: Item ↔ Exhibition (via ItemExhibition)

### **Step 3: Create the Conceptual Model**

A high-level ERD was created to map entities and their relationships without technical details.

### **Step 4: Create the Logical/Physical Model**

Defined keys, constraints, data types, and attribute lengths.

### **Step 5: Normalize to 3NF**

Ensured that:

* There are no repeating groups (1NF)
* All attributes are fully dependent on the primary key (2NF)
* No transitive dependencies exist (3NF)

### **Step 6: Diagram Creation**

Two diagrams were produced:

* Conceptual Model (overview)
* Logical Model (with keys and types)