

*Business description for DB* ***museum.***

This database is designed to support the daily operations of a **Museum** by managing information about **artifacts**, **storage locations**, **exhibitions**, **tickets**, **visitors**, and **employees**.

The database enforces **high data quality standards** through normalization (3NF), constraints (e.g., CHECK, NOT NULL), and relationships (foreign keys), ensuring data integrity and consistency.

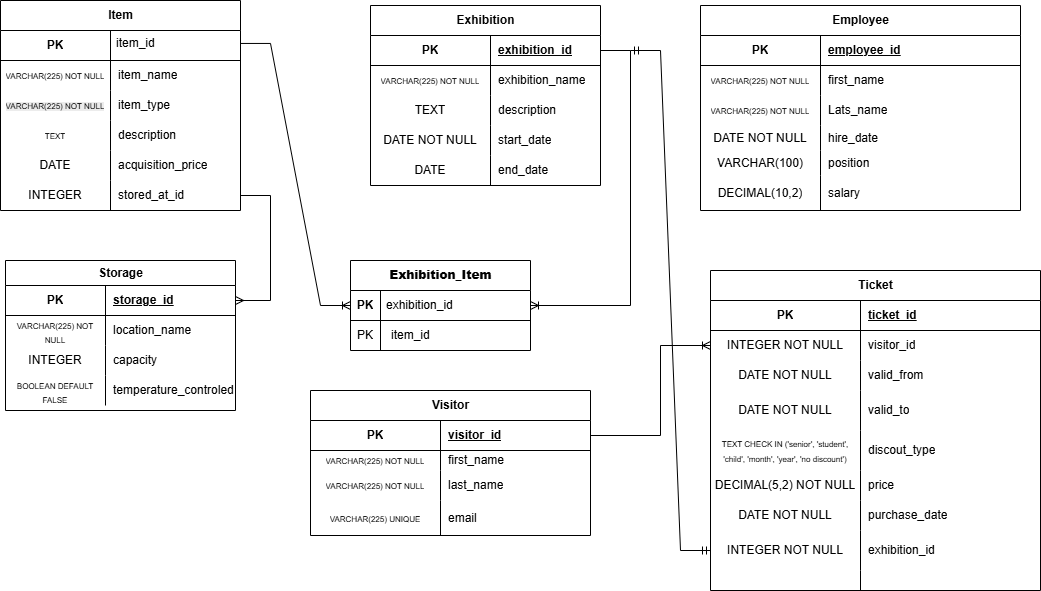
### **Key Functional Areas:**

* **Artifacts and Items Management:**Every object (artifact, artwork, specimen) in the museum collection is stored as an **Item**.  
  Instead of directly saving the current location inside the item record, each item is linked via **stored\_at\_id** to a **Storage** table, which manages storage locations separately.  
  This allows tracking where each artifact is currently held, whether on display or in secure storage.
* **Visitor and Ticket Management:**The **Visitor** entity stores general information about museum guests (if tracked).  
  Ticket sales are managed separately in the **Tickets** table, which records ticket purchases linked to visitors, including the ticket type, discount applied, price, and validity dates.  
  This ensures detailed tracking of all visits and financial income from ticket sales.
* **Exhibitions Management:  
  The Exhibitions** table organizes exhibitions, their schedules, and links which artifacts are presented in each exhibition.  
  Exhibitions can have multiple artifacts, and one artifact can appear in multiple exhibitions, creating a many-to-many relationship.
* **Employee Management:**Employees responsible for organizing exhibitions, managing items, and overseeing visitor operations are stored in the **Employees** table.
* **Storage and Inventory Management:**The **Storage** table maintains information about various museum storage facilities (rooms, vaults, display halls) where items are housed.  
  Each item has a *stored\_at\_id* to trace exactly where it is stored.

**How the Database Helps the Museum:**

* Tracks every item’s location and history.
* Allows planning and management of exhibitions and which items are shown.
* Records visitor ticket sales and attendance.
* Maintains accurate financial records tied to ticket purchases.
* Supports security and auditing of item storage locations.
* Provides data to optimize visitor flow and exhibition popularity.
* Plan and organize upcoming exhibitions with accurate item data.
* Track visitor engagement and attendance over time.
* Assign and manage employee responsibilities.
* Maintain a clear audit of all stored items, exhibitions, and related transactions.
* Generate quarterly analytics for management decision-making.

**Logical scheme:**



# **Main Entities Overview:**

| **Item** | Represents individual artifacts, artworks, specimens, or historical objects that are part of the museum’s collection. |
| --- | --- |
| **Exhibition** | Represents organized displays of items for public or private viewing. Each exhibition can feature multiple items. |
| **Visitor** | Represents people who visit the museum and purchase tickets for exhibitions. Includes ticket purchase and discount details. |
| **Ticket** | Represents a purchased access record for a specific exhibition. Includes ticket type, discount, valid dates, and price. |
| **Employee** | Represents museum staff responsible for managing items, exhibitions, visitors, or administrative duties. |
| **Storage**  **Exhibition\_Item** | Represents physical storage locations within the museum where items not currently on display are securely stored.  Many-to-Many bridge table. A single item can belong to many exhibitions, and an exhibition can display many items. |

**Table: Item**

| **Field Name** | **Field Description** | **Data Type** |
| --- | --- | --- |
| item\_id | Unique identifier for each item | SERIAL (Primary Key) |
| item\_type | Group of items | VARCHAR(225) DEFAULT ‘NEW’ |
| item\_name | Name or title of the item | VARCHAR(225) NOT NULL |
| description | Detailed description of the item | TEXT |
| acquisition\_date | Date the item was acquired | DATE |
| stored\_at\_id | ID of the storage location where the item is kept | INTEGER (Foreign Key to storage.storage\_id) |

Linked with **Exhibition\_Item** table with one to many relation.

Linked with **Storage** table with one to many relation.

**Table: Exhibition**

| **Field Name** | **Field Description** | **Data Type** |
| --- | --- | --- |
| exhibition\_id | Unique identifier for each exhibition | SERIAL (Primary Key) |
| exhibition\_name | Name of the exhibition | TEXT NOT NULL |
| start\_date | Starting date of the exhibition | DATE NOT NULL |
| end\_date | Ending date of the exhibition | DATE |
| description | Summary or details of the exhibition theme | TEXT |

Linked with **Exhibition\_Item** table with one to many relation.

**Table: Visitor**

| **Field Name** | **Field Description** | **Data Type** |
| --- | --- | --- |
| visitor\_id | Unique identifier for each visitor | SERIAL (Primary Key) |
| first\_name | First name of the visitor | VARCHAR(225) NOT NULL |
| last\_name | Last name of the visitor | VARCHAR(225) NOT NULL |
| email | Visitor’s email | VARCHAR(225) UNIQUE |

Linked with **Ticket** table with one to many relation.

**Table: Ticket**

| **Field Name** | **Field Description** | **Data Type** |
| --- | --- | --- |
| ticket\_id | Unique identifier for each ticket | SERIAL (Primary Key) |
| visitor\_id | Visitor who owns the ticket | INTEGER (Foreign Key to visitor.visitor\_id) |
| exhibition\_id | Exhibition the ticket grants access to | INTEGER (Foreign Key to exhibition.exhibition\_id) |
| purchase\_date | Date of purchase | DATE |
| price | Price paid for the ticket | DECIMAL(5,2) NOT NULL |
| discount\_type | Discount type applied (e.g., student, senior, child, month, year) | TEXT CHECK IN ('senior', 'student', 'child', 'month', 'year', 'no discount') |
| valid\_from | Date when the ticket becomes valid | DATE NOT NULL |
| valid\_to | Date when the ticket expires | DATE NOT NULL |

Linked with **Exhibition** table with one to one relation.

**Table: Employee**

| **Field Name** | **Field Description** | **Data Type** |
| --- | --- | --- |
| employee\_id | Unique identifier for each employee | SERIAL (Primary Key) |
| first\_name | Employee's first name | TEXT NOT NULL |
| last\_name | Employee's last name | TEXT NOT NULL |
| hire\_date | Date the employee was hired | DATE NOT NULL |
| position | Job title or position | VARCHAR(100) |
| salary | Salary EUR | DECIMAL(10,2) NOT NULL |

**Table: Storage**

| **Field Name** | **Field Description** | **Data Type** |
| --- | --- | --- |
| storage\_id | Unique identifier for each storage location | SERIAL (Primary Key) |
| location\_name | Name of the storage room or area | TEXT NOT NULL |
| temperature\_control | Whether storage has temperature control (Yes/No) | BOOLEAN |
| capacity | Maximum number of items it can hold | INTEGER |

**Table: Exhibition\_Item**

| **Field Name** | **Field Description** | **Data Type** |
| --- | --- | --- |
| exhibition\_id | ID of the exhibition | INTEGER (Foreign Key to exhibition.exhibition\_id) |
| item\_id | ID of the item displayed in the exhibition | INTEGER (Foreign Key to item.item\_id) |