

|  |
| --- |
| **An auction house** |
|  |

Contents

[1 Business Description 3](#_Toc62212630)

[1.1 Business background 3](#_Toc62212631)

[1.2 Problems. Current Situation 3](#_Toc62212632)

[1.3 The benefits of implementing a database. Project Vision 3](#_Toc62212633)

[2 Model description 3](#_Toc62212634)

[2.1 Definitions & Acronyms 4](#_Toc62212635)

[2.2 Logical Scheme 4](#_Toc62212636)

[2.3 Objects 4](#_Toc62212637)

# 

# Business Description

## Business background

## Auction houses organize live and online events where sellers list valuable items such as artwork, antiques, jewelry, and collectibles for competitive bidding. Buyers register to participate, place bids, and purchase items. This database system supports all core operations of an auction house, including managing sellers, buyers, items, bids, auctions, sales, and payments. It provides a centralized structure for storing and analyzing auction data, ensuring operational efficiency, accountability, and traceability across all transactions.

## Problems. Current Situation

## Many auction houses rely on disconnected spreadsheets or outdated software systems to track auctions and transactions. These methods often lead to data duplication, inconsistencies, and a lack of real-time visibility. Manual record keeping increases the risk of errors in financial reporting and item tracking. Additionally, without a normalized database, it is difficult to produce analytical insights or monitor performance metrics. The lack of integration results in inefficiency and limits scalability.

## the Benefits of implementing a database. Project Vision

## The implementation of a centralized Auction House Database will streamline operations and eliminate redundancy. It integrates all business processes, including seller management, item listing, bidding, sales, and payments, into a single system. This ensures accuracy, faster data retrieval, and improved reporting. The vision is to create a scalable, secure, and normalized database that not only supports current operations but can also expand to online auction platforms, reporting dashboards, and analytical tools in the future.

# Model description

# The Auction House Database has been designed in Third Normal Form (3NF) to ensure efficient data organization and eliminate redundancy. The model includes 12 entities: Seller, Buyer, Item, Auction, Lot, Bid, Sale, Payment, Category, Address, Employee, and Auction\_Item. Relationships are clearly defined through primary and foreign keys, enforcing referential integrity. The many-to-many relationship between Auction and Item is implemented using the Auction\_Item bridge table. Category uses a self-referential relationship for hierarchical classification, and Lot includes a composite key (AuctionID + LotNumber). The model structure ensures logical consistency and supports data integrity.

## Definitions & AcronymS

## PK – Primary Key (unique identifier for each record) FK – Foreign Key (relationship between tables) 3NF – Third Normal Form (reduces redundancy and maintains data integrity) M:N – Many-to-Many relationship ERD – Entity Relationship Diagram

## Logical Scheme

## The logical design is based on 12 interrelated entities that represent key components of auction management. One-to-many relationships connect Seller → Item, Auction → Lot, and Buyer → Bid/Sale. The many-to-many relationship between Auction and Item is implemented through the Auction\_Item bridge table, which uses a composite key of (AuctionID, ItemID). Category supports hierarchical classification via a self-referencing ParentCategoryID. Each table adheres to 3NF, ensuring that all non-key attributes depend solely on the primary key.

A screenshot of a computer

AI-generated content may be incorrect.

## Objects

## Seller – Stores details about auction sellers.

## Buyer – Contains buyer information for registration and bidding.

## Item – Describes auction items, linked to both Seller and Category.

## Auction – Defines event details, including date, time, and location.

## Lot – Associates an item with a specific auction, identified by lot number.

## Bid – Tracks bids placed by buyers on lots.

## Sale – Records the final sale and winning bidder details.

## Payment – Manages payment transactions for completed sales.

## Category – Defines item categories with parent-child hierarchy.

## Auction\_Item – Bridge table linking auctions and items (many-to-many).

## Address – Centralized address information for sellers and buyers.

## Employee – Represents auction house staff and their assigned auctions.

Table Description

• Seller – Stores seller information and contact details.

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field Name | Field Description (include constraint) | Data Type |
| Seller | SellerID | Primary Key, unique identifier | INT |
| Seller | SellerName | Full name (NOT NULL) | VARCHAR(100) |
| Seller | ContactNumber | Seller phone number | VARCHAR(20) |
| Seller | Email | Unique email (UNIQUE) | VARCHAR(100) |
| Seller | AddressID | Foreign Key → Address(AddressID) | INT |

• Buyer – Contains buyer details and registration information.

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field Name | Field Description (include constraint) | Data Type |
| Buyer | BuyerID | Primary Key | INT |
| Buyer | BuyerName | Full name (NOT NULL) | VARCHAR(100) |
| Buyer | ContactNumber | Buyer phone number | VARCHAR(20) |
| Buyer | Email | Unique email (UNIQUE) | VARCHAR(100) |
| Buyer | AddressID | Foreign Key → Address(AddressID) | INT |

• Item – Represents items listed for sale and linked to sellers.

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field Name | Field Description (include constraint) | Data Type |
| Item | ItemID | Primary Key | INT |
| Item | ItemName | Item name (NOT NULL) | VARCHAR(150) |
| Item | Description | Item description | TEXT |
| Item | StartingPrice | Initial listing price (CHECK ≥ 0) | DECIMAL(12,2) |
| Item | SellerID | Foreign Key → Seller(SellerID) | INT |
| Item | CategoryID | Foreign Key → Category(CategoryID) | INT |

• Auction – Defines auction events including date, time, and category.

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field Name | Field Description (include constraint) | Data Type |
| Auction | AuctionID | Primary Key | INT |
| Auction | AuctionDate | Date of the auction | DATE |
| Auction | StartTime | Start time | TIME |
| Auction | EndTime | End time | TIME |
| Auction | Location | Auction location | VARCHAR(150) |
| Auction | CategoryID | Foreign Key → Category(CategoryID) | INT |

• Lot – Connects an item to an auction and assigns a lot number.

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field Name | Field Description (include constraint) | Data Type |
| Lot | LotID | Primary Key | INT |
| Lot | LotNumber | Unique lot number per auction | VARCHAR(20) |
| Lot | AuctionID | Foreign Key → Auction(AuctionID) | INT |
| Lot | ItemID | Foreign Key → Item(ItemID) | INT |
| Lot | LotDescription | Lot details | TEXT |

• Bid – Records bids placed by buyers during auctions.

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field Name | Field Description (include constraint) | Data Type |
| Bid | BidID | Primary Key | INT |
| Bid | LotID | Foreign Key → Lot(LotID) | INT |
| Bid | BuyerID | Foreign Key → Buyer(BuyerID) | INT |
| Bid | BidAmount | Bid amount (CHECK ≥ 0) | DECIMAL(12,2) |
| Bid | BidTime | Timestamp of bid | DATETIME |

• Sale – Captures the final sale of a lot.

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field Name | Field Description (include constraint) | Data Type |
| Sale | SaleID | Primary Key | INT |
| Sale | LotID | Foreign Key → Lot(LotID), unique | INT |
| Sale | BuyerID | Foreign Key → Buyer(BuyerID) | INT |
| Sale | FinalPrice | Final sale price | DECIMAL(12,2) |
| Sale | SaleDate | Date of sale | DATE |

• Payment – Tracks payments for completed sales.

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field Name | Field Description (include constraint) | Data Type |
| Payment | PaymentID | Primary Key | INT |
| Payment | SaleID | Foreign Key → Sale(SaleID) | INT |
| Payment | PaymentDate | Payment date | DATE |
| Payment | PaymentAmount | Amount paid | DECIMAL(12,2) |
| Payment | PaymentMethod | ENUM('Credit Card','Bank Transfer','Cash') | ENUM |

• Category – Defines item categories and allows hierarchy.

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field Name | Field Description (include constraint) | Data Type |
| Category | CategoryID | Primary Key | INT |
| Category | CategoryName | Unique category name | VARCHAR(100) |
| Category | Description | Category description | TEXT |
| Category | ParentCategoryID | Self-referencing Foreign Key | INT |

• Address – Stores address details for buyers and sellers.

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field Name | Field Description (include constraint) | Data Type |
| Address | AddressID | Primary Key | INT |
| Address | Street | Street name | VARCHAR(150) |
| Address | City | City name | VARCHAR(100) |
| Address | Country | Country name | VARCHAR(100) |
| Address | PostalCode | Postal code | VARCHAR(15) |

• Employee – Represents staff managing auction events.

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field Name | Field Description (include constraint) | Data Type |
| Employee | EmployeeID | Primary Key | INT |
| Employee | Name | Employee full name | VARCHAR(100) |
| Employee | Role | Employee role | VARCHAR(50) |
| Employee | AuctionID | Foreign Key → Auction(AuctionID) | INT |

• Auction\_Item – Bridge table linking auctions and items (many-to-many).

|  |  |  |  |
| --- | --- | --- | --- |
| Table Name | Field Name | Field Description (include constraint) | Data Type |
| Auction\_Item | AuctionID | Foreign Key → Auction(AuctionID), part of PK | INT |
| Auction\_Item | ItemID | Foreign Key → Item(ItemID), part of PK | INT |
| Auction\_Item | DisplayOrder | Display sequence number | INT |
| Auction\_Item | EstimatedPrice | Estimated item value | DECIMAL(12,2) |

Comments on Relationships:

• One-to-Many: Seller → Item, Auction → Lot, Buyer → Bid and Sale.  
• Many-to-Many: Auction ↔ Item via Auction\_Item bridge.  
• One-to-One: Sale ↔ Payment.  
• Self-Referential: Category → Category (ParentCategoryID).  
• Composite Uniqueness: Lot (AuctionID + LotNumber).  
All foreign keys include cascading or set-null actions to maintain referential integrity.

Examples with Data:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| SellerName | ItemName | AuctionDate | BuyerName | FinalPrice | PaymentMethod |
| John Doe | Antique Vase | 2025-05-01 | Jane Smith | 2500.00 | Bank Transfer |
| Maria Lopez | Gold Necklace | 2025-06-15 | David Brown | 3800.00 | Credit Card |
| Liam Carter | Oil Painting | 2025-07-20 | Emma Johnson | 5400.00 | Cash |