# VISVESVARAYA TECHNOLOGICAL UNIVERSITY
## BELAGAVI-590018


**"A MINI PROJECT REPORT"**
**(Subject Code:21CSL55 )**
**ON**
**" Art Gallery Management"**

Submitted in partial fulfillment for the requirements for the Award of Degree of

**BACHELOR OF ENGINEERING**
**IN**
**COMPUTER SCIENCE AND ENGINEERING**
**BY**

Pritish Ali
1EP21Cs076

UNDER THE GUIDANCE OF
Prof. Swathi Pandey
Assistant Professor
DEPT. Of CSE, EPCET

**Department of Computer Science and Engineering**
**EAST POINT COLLEGE OF ENGINEERING AND TECHNOLOGY**
**Bidarahalli, Bengaluru – 560 049**
**2023-2024**

# EAST POINT COLLEGE OF ENGINEERING AND TECHNOLOGY

(Affiliated to Visvesvaraya Technological University, Belgavi)

Bangalore-560049

## DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

## CERTIFICATE

This is to certify that the **Project Work** entitled **"Art Gallery Management "** is a bonafied work carried out by **Mr. Pritish Ali, bearing USN 1EP21CS076** in partial fulfillment for the award of **Bachelor of Engineering in Computer Science and Engineering** under **Visvesvaraya Technological University, Belgaum** during the year **2023-2024**. It is certified that all the corrections/suggestions indicated in the Internal Assessment have been incorporated in the report and submitted in the department library. This seminar report has been approved as it satisfies the academic requirements in respect of Mini Project Work prescribed for the award of the said degree.

| GUIDE | HOD | PRINCIPAL |
|---|---|---|
| **Prof. Swathi Pandey** | **Dr. Manimozi Iyer** | **Dr. Mrityunjaye V Latte** |
| Assistant Professor | Head of the Department . | Principal |

**Examiners**

| Name of the Examiners | Signature with date |
|---|---|
| **1.Prof. Pavan Kumar** | |

# ACKNOWLEDGEMENT

Any achievement, be it scholastic or otherwise does not depend solely on the individual efforts but on the guidance, encouragement and cooperation of intellectuals, elders and friends. We would like to take this opportunity to thank them all.

First and foremost we would like to express our sincere regards and thanks to **Mr. Promod Gowda** and **Mr. Rajiv Gowda,** CEOs East Point Group of Institutions, Bangalore, for providing necessary infrastructure and creating good environment.

We express our gratitude to **Dr. Mrityunjaye V Latte,** Principal, EPCET who has always been a great source of inspiration.

We express our sincere regards and thanks to Dr. Manimozi Iyer **,** Professor and Head, Department of Computer Science and Engineering, EPCET, Bangalore, for his encouragement and support.

We are grateful to acknowledge the guidance and encouragement given to us by Prof. Swathi Pandey , Assistant Professors, Department of Computer Science and Engineering, EPCET, Bangalore, who has rendered a valuable assistance.

We also extend our thanks to all the faculties of the **Department of Computer Science and Engineering, EPCET**, Bangalore, who have encouraged us throughout the course of the Mini Project Work.

Last, but not the least, we would like to thank our family and friends for their inputs to improve the Mimi Project works.

| | |
|---|---|
| **Name: PRITISH ALI** | **USN: 1EP21CS076** |
| **Name: RUPAM BHATTACHARYYA** | **USN: 1EP21CS089** |
| **Name: NAGA SAI MANOJ** | **USN: 1EP21CS090** |
| **Name: SANOOP SAJAN** | **USN: 1EP21CS095** |

# ABSTRACT

The main aim of the project is the management of the database of *ART GALLERY*.

This project is insight into the design and implementation of a Art Gallery Management. This is done by creating a database of the available details in Art Gallery. The primary aim of this Art Gallery Management System is to improve accuracy and enhance safety and efficiency of tracking and keeping details of art and paintings in the art gallery. I have developed this software for ensuring effective policing by providing statistics of the Members.

The MYSQL database is used as a platform along with PHP and WAMP Server support. Application and the GUI are developed in HTML5, CSS3 using PHP and WAMP Server.

Overall this Art Gallery Management System is used to manage most art-related activities like exhibitions, gallery management, art stocks etc. in the gallery.

# CHAPTER 1

# INTRODUCTION

## 1.1 INTRODUCTION TO SQL

SQL which is an abbreviation for **Structured Query Language** is a language to request data from a database, to add, update, or remove data within a database, or to manipulate the metadata of the database.

Sometimes SQL is characterized as *non-procedural* because procedural languages generally require the details of the operations to be specified, such as opening and closing tables, loading and searching indexes, or flushing buffers and writing data to file systems. Therefore, SQL is designed at a higher conceptual level of operation than procedural languages.

Commonly used statements are grouped into the following categories

**Data Query Language (DQL)**

SELECT-Used to retrieve certain records from one or more tables.

**Data Manipulation Language (DML)**

INSERT - Used to create a record

UPDATE - Used to change certain records.

DELETE - Used to delete certain records.

**Data Definition Language (DDL)**

CREATE - Used to create a new table, a view of a table, or other object in database.

ALTER - Used to modify an existing database object, such as a table.

DROP - Used to delete an entire table, a view of a table or other object in the database.

**Data Control Language (DCL)**

GRANT - Used to give a privilege to someone

REVOKE - Used to take back privileges granted to someone.

# 1.2 INTRODUCTION TO FRONT END SOFTWARE

The “front end languages” live in the browser. After you type in an address in the address bar at the top and hit Enter, your browser will receive an at least an HTML file from the web server.

Each of these languages performs a separate but very important function but the work harmoniously together to determine how the web page is STRUCTURED(HTML), how it LOOKS(CSS), and how its FUNCTIONS (JavaScript).

Front end web development is NOT design (You won’t be playing around in Photoshop or anything), but a *front-end developer* does apply the work of designers to the web page by translating their well-designed layouts into real code. The front-end developer stands between the designer on one end and the back-end developer on the other, translating the design into code and plugging the data from the back-end developer into the right spots.

**PHP** is a server-side scripting language designed primarily for web development but also used as a general-purpose programming language. Originally created by Rasmus Lerdorf in 1994, the PHP reference implementation is now produced by The PHP Development Team.

PHP code may be embedded into HTML or it can be used in combination with various web template systems, web content management systems and web frameworks. PHP code is usually processed by a PHP interpreter implemented as a module in the web server or as a Common Gateway Interface (CGI) executable. The web server software combines the result of the interpreted and executed PHP code, which may be any type of data, including images, with the generated web page. PHP code may also be executed with a command-line interface (CLI) and it can be used to implement stand-alone graphical applications.

The standard PHP interpreter, powered by the Zend Engine, is free to use software released under the PHP License. PHP code is usually processed by a PHP interpreter implemented as a module in the web server or as Common Gateway Interface(CGI) executable. PHP has been widely ported on web servers on almost every operating system and platform, free of charge.

# CHAPTER 2

# REQUIREMENTS SPECIFICATION

## 2.1 SOFTWARE REQUIREMENTS

Operating System : 64bit WINDOWS Operating System,
X64-based processor

Database : MYSQL

Scripting Language : HTML5, CSS3, PHP

Server : WAMP

## 2.2 HARDWARE REQUIREMENTS

Processor : Intel Celeron CPU N3060 @1.60GHz or Above

RAM : 4.00 GB or Above

Hard Disk : 1 TB

Compact Disk : CD-ROM, CD-R, CD-RW

Input device : Keyboard

# CHAPTER 3

# OBJECTIVE OF THE PROJECT

## The main objective of creating an Art Gallery database project is

- To manage the details of gallery, exhibition, artwork and artist. It manages all the sales and inventory in the gallery. The purpose of the project is to build and application program to reduce the manual work.

- To tracks all the details about the sales of the artwork, the customer that bought it, etc. It manages the information about the artwork. Provides an information and description of the artworks left, thereby increasing the efficiency of managing the gallery. The organisation can maintain a computerized record of the artwork present in the gallery.

- To helps in the utilization of the resources in an effective manner. It maintains a list of all the customers and the various artwork that they have bought and the money that have invested in each.

- To maintains the record of exhibitions and various sales made during it. The objective of developing such computerized system is to reduce the paper work and safe of time in art gallery database management, thereby increasing the efficiency and decreasing the work load.

- To develop such computerized system is to reduce the paper work and safe of time in art gallery database management, thereby increasing the efficiency and decreasing the work load.

# CHAPTER 4

# IMPLEMENTATION

## 4.1 ER DIAGRAM

1. An **entity-relationship model (ER Model)** describes inter-related things of interest in a specific domain of knowledge. An ER model is composed of entity types (which classify the things of interest) and specifies relationships that can exist between instances of those entity types.

2. An entity may be defined as a thing capable of an independent existence that can be uniquely identified. An entity is an abstraction from the complexities of a domain.

3. Attributes are drawn as ovals and are connected with a line to exactly one entity or relationship set.

4. An entity relationship model, also called an entity-relationship (ER) diagram, is a graphical representation of entities and their relationships to each other, typically used in computing in regard to the organization of data within databases or information systems.

5. Cardinality constraints are expressed as follows:

   **a.** A double line indicates a participation constraint, totality or subjectivity: all entities in the entity set must participate in at least one relationship in the relationship set.

   **b.** An arrow from entity set to relationship set indicates a key constraint, i.e. injectivity: each entity of the entity set can participate in at most one relationship in the relationship set.

   **c.** A thick line indicates both, i.e. bijectivity: each entity in the entity set is involved in exactly one relationship.

   **d.** An underlined name of an attribute indicates that it is a key: two different entities or relationships with this attribute always have different values for this attribute.

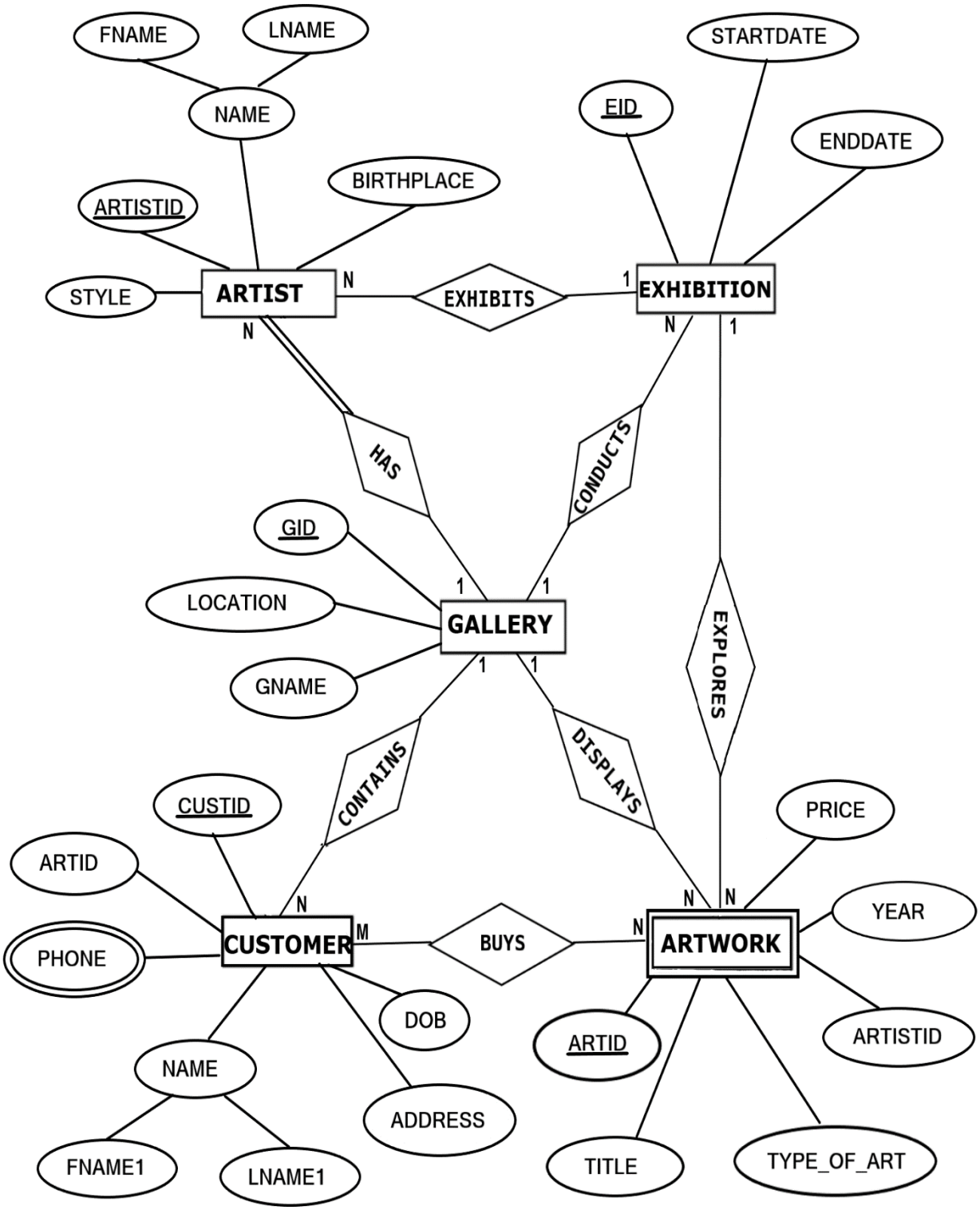



**FIGURE 4.1: ER DIAGRAM of ART GALLERY DATABASE**

# 4.2 MAPPING OF ER DIAGRAM TO RELATIONS

## STEP 1: Mapping of Regular Entities

For each regular entity type E in the ER schema, create relation R that includes all simple attributes of E.

**GALLERY**

| **GID** | GNAME | LOCATION |
|---|---|---|

**EXHIBITION**

| **EID** | STARTDATE | ENDDATE |
|---|---|---|

**ARTIST**

| **ARTISTID** | FNAME | LNAME | BIRTHPLACE | STYLE |
|---|---|---|---|---|

**CUSTOMER**

| **CUSTID** | **ARTID** | FNAME1 | LNAME1 | ADDRESS | PHONE | DOB |
|---|---|---|---|---|---|---|
| | **FK** | | | | | |

## STEP 2 : Mapping of Weak Entity Types

**ARTWORK**

| **ARTID** | **ARTISTID** | TITLE | TYPE_OF_ART | YEAR | PRICE |
|---|---|---|---|---|---|
| | **FK** | | | | |

## STEP 3: Mapping of 1:1 Relationship

Identify the relation S that represents the participating entity type at the 1-side of the relationship type.

Include as foreign key in S the primary key of the relations T that represents the other entity type participating in R.

For each binary 1:1 relationship type R in ER schema, identify the relations S and T that correspond to the entity types participating in R if any.

There are **no** 1:1 relationship.

## STEP 4 : Mapping of 1:N Relationship

**EXHIBITION**

| **EID** | STARTDATE | ENDDATE | **GID** |
|---|---|---|---|
| | | | **FK** |

**ARTIST**

| **ARTISTID** | FNAME | LNAME | BIRTHPLACE | STYLE | **EID** | **GID** | **CUSTID** |
|---|---|---|---|---|---|---|---|
| | | | | | **FK** | **FK** | **FK** |

**CUSTOMER**

| **CUSTID** | **ARTID** | FNAME1 | LNAME1 | ADDRESS | DOB | **GID** |
|---|---|---|---|---|---|---|
| | **FK** | | | | | **FK** |

**ARTWORK**

| **ARTID** | **ARTISTID** | TITLE | TYPE_OF_ART | YEAR | PRICE | **EID** | **GID** |
|---|---|---|---|---|---|---|---|
| | **FK** | | | | | **FK** | **FK** |

## STEP 5 : Mapping of M:N Relationship

Create a new relation S to represent R.

Include as foreign key attributes in S the primary key of the relations that represents the participating entity types their combination will form the primary key of S.

Also, include any simple attributes of the M:N relationship type as attributes of S.

## STEP 6: Mapping of Multi-Valued Attributes

For each multivalued attributes A, create a new relation R. This relation R will include an attribute corresponding to A, plus the primary key attribute K-as a foreign key in R-of the relation that represents the entity type of relationship type that has A as an attribute.

The Primary Key of R is the combination of A and K. If the multivalued attribute is composite, we include its simple components.

**CONTACTS**

| **CUSTID** | PHONE |
|---|---|

## STEP 7: Mapping of N-Ary Relationship Types

For each n-ary relationship type R, where n>2 create a new relationship S to represent R. λ include as foreign key attributes in S the primary keys of the relations that represent the participating entity types.

λ also includes any simple attributes of the n-ary relationship type (or simple components of composite attributes) as attributes of S.

There are **no** n-ary relationship types.

# 4.3 SCHEMA DIAGRAM

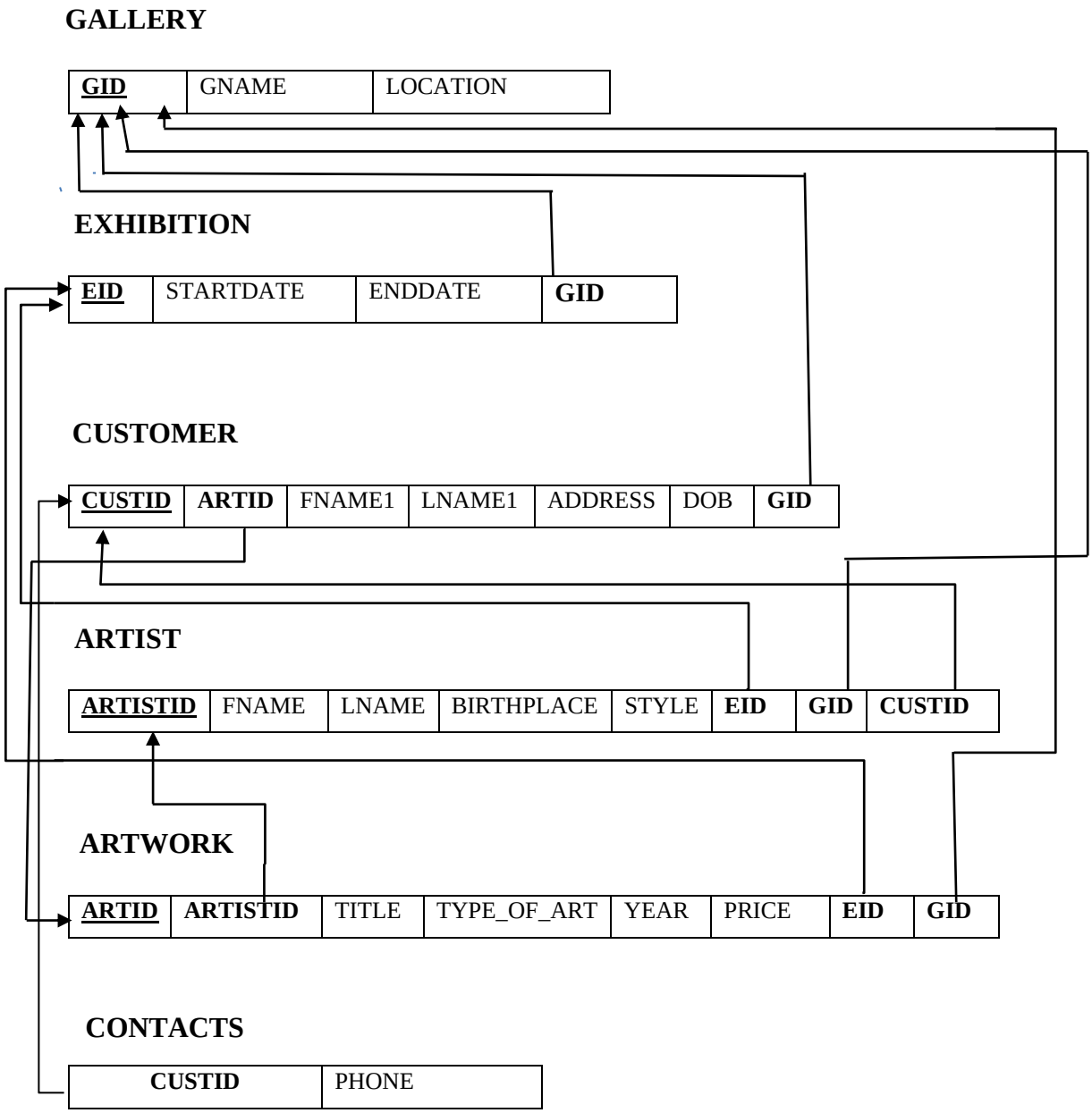



**FIGURE 4.3: SCHEMA DIAGRAM**

# 4.3 NORMALIZE THE RELATIONS

Database normalization, or simply normalization, is the process of organizing the columns(attributes) and tables(relations) of a relational database to reduce data redundancy and improve data integrity. Normalization involves arranging attributes in relations based on dependencies between attributes.

## 1. First Normal Form

As per First normal form, no two rows of data must contain repeating group of information. Each set of columns must have a unique value, such that multiple columns cannot be used to fetch the same row. Each table should be organized into rows, and each row should have a primary key that will distinguishes it as unique.

**Example:**

### GALLERY

| **GID** | GNAME | LOCATION |
|---|---|---|

All the tables in the database are normalized to 1NF as all the attributes are atomic.

## 2. Second Normal Form (2NF)

A table is in 2NF if it is in 1NF and if all non-key attributes are fully functionally dependent on all of the key.

**Example:**

### CUSTOMER

| **CUSTID** | **ARTID** | FNAME1 | LNAME1 | ADDRESS | DOB | **GID** |
|---|---|---|---|---|---|---|

FD1

### FD1

| **CUSTID** | FNAME1 | LNAME1 | DOB |
|---|---|---|---|

## 3. Third Normal Form(3NF):

A table is in 3NF if it is in 2NF and if it has no transitive dependency. X->Y, Y->Z, X>Z

According to CODD's definition a relation schema R is in 3NF. It satisfies 2NF and no non-prime attribute of R is transitively dependent on the primary key. All tables of database satisfies upto 3NF.

# 4.5 CREATION OF TABLES

## 1. CREATING GALLERY TABLE

CREATE TABLE GALLERY
(GID VARCHAR(20) PRIMARY KEY,
GNAME CHAR(20),
LOCATION CHAR(20));

```
mysql> desc gallery;
+----------+-------------+------+-----+----------+-------+
| Field    | Type        | Null | Key | Default  | Extra |
+----------+-------------+------+-----+----------+-------+
| gid      | varchar(26) | NO   | PRI | not null |       |
| gname    | varchar(24) | YES  |     | NULL     |       |
| location | varchar(26) | YES  |     | NULL     |       |
+----------+-------------+------+-----+----------+-------+
3 rows in set (0.00 sec)
```

## 2. CREATE EXHIBITION TABLE

CREATE TABLE EXHIBITION
(EID VARCHAR(20) PRIMARY KEY,
GID VARCHAR(20),
STARTDATE DATE,
ENDDATE DATE,
FOREIGN KEY(GID) REFERENCES GALLERY(GID) ON DELETE CASCADE);

```
mysql> desc exhibition;
+-----------+-------------+------+-----+---------+-------+
| Field     | Type        | Null | Key | Default | Extra |
+-----------+-------------+------+-----+---------+-------+
| eid       | varchar(20) | NO   | PRI | NULL    |       |
| gid       | varchar(20) | YES  | MUL | NULL    |       |
| startdate | date        | YES  |     | NULL    |       |
| enddate   | date        | YES  |     | NULL    |       |
+-----------+-------------+------+-----+---------+-------+
4 rows in set (0.00 sec)
```

## 3. CREATE ARTWORK TABLE

CREATE TABLE ARTWORK
(ARTID VARCHAR(20) PRIMARY KEY,
TITLE VARCHAR(20),
YEAR INT,
TYPE_OF_ART VARCHAR(20),
PRICE INT,
EID VARCHAR(20), GID VARCHAR(20),

FOREIGN KEY(EID) REFERENCES EXHIBITION(EID) ON DELETE CASCADE,
FOREIGN KEY(GID) REFERENCES GALLERY(GID) ON DELETE CASCADE);

```
mysql> desc artwork;
+-------------+-------------+------+-----+---------+-------+
| Field       | Type        | Null | Key | Default | Extra |
+-------------+-------------+------+-----+---------+-------+
| artid       | varchar(20) | NO   | PRI | NULL    |       |
| title       | varchar(20) | YES  |     | NULL    |       |
| year        | varchar(5)  | YES  |     | NULL    |       |
| type_of_art | varchar(20) | YES  |     | NULL    |       |
| price       | varchar(15) | YES  |     | NULL    |       |
| eid         | varchar(20) | YES  | MUL | NULL    |       |
| gid         | varchar(20) | YES  | MUL | NULL    |       |
| artistid    | varchar(20) | YES  | MUL | NULL    |       |
+-------------+-------------+------+-----+---------+-------+
8 rows in set (0.00 sec)
```

## 4. CREATE CUSTOMER TABLE

CREATE TABLE CUSTOMER
(CUSTID VARCHAR(20) PRIMARY KEY,
GID VARCHAR(20),
ARTID VARCHAR(20),
FNAME1 CHAR(20),
LNAME1 CHAR(20),
DOB DATE,
ADDRESS CHAR(20),
FOREIGN KEY(GID) REFERENCES GALLERY(GID) ON DELETE CASCADE,
FOREIGN KEY(ARTID) REFERENCES GALLERY(ARTID) ON DELETE CASCADE);

```
mysql> desc customer;
+---------+-------------+------+-----+---------+-------+
| Field   | Type        | Null | Key | Default | Extra |
+---------+-------------+------+-----+---------+-------+
| custid  | varchar(20) | NO   | PRI | NULL    |       |
| gid     | varchar(20) | YES  | MUL | NULL    |       |
| artid   | varchar(20) | YES  | MUL | NULL    |       |
| fname   | char(25)    | YES  |     | NULL    |       |
| lname   | char(25)    | YES  |     | NULL    |       |
| dob     | date        | YES  |     | NULL    |       |
| address | char(25)    | YES  |     | NULL    |       |
+---------+-------------+------+-----+---------+-------+
7 rows in set (0.00 sec)
```

## 5. CREATE ARTIST TABLE

CREATE TABLE ARTIST
(ARTISTID VARCHAR(20) PRIMARY KEY,
GID VARCHAR(20),
CUSTID VARCHAR(20),
EID VARCHAR(20),
FNAME CHAR(20),

LNAME CHAR(20),
BIRTHPLACE CHAR(20),
STYLE CHAR(20),
FOREIGN KEY(GID) REFERENCES GALLERY(GID) ON DELETE CASCADE,
FOREIGN KEY (CUSTID) REFERENCES CUSTOMER(CUSTID) ON DELETE CASCADE,
FOREIGN KEY(EID) REFERENCES EXHIBITION(EID) ON DELETE CASCADE);

ALTER TABLE ARTWORK ADD ARTISTID VARCHAR(20);

ALTER TABLE ARTWORK
ADD FOREIGN KEY (ARTISTID) REFERENCES ARTIST(ARTISTID) ON DELETE CASCADE;

```
mysql> desc artist;
+------------+-------------+------+-----+---------+-------+
| Field      | Type        | Null | Key | Default | Extra |
+------------+-------------+------+-----+---------+-------+
| artistid   | varchar(20) | NO   | PRI | NULL    |       |
| gid        | varchar(20) | YES  | MUL | NULL    |       |
| custid     | varchar(20) | YES  | MUL | NULL    |       |
| eid        | varchar(20) | YES  | MUL | NULL    |       |
| fname1     | char(25)    | YES  |     | NULL    |       |
| lname1     | char(25)    | YES  |     | NULL    |       |
| birthplace | char(25)    | YES  |     | NULL    |       |
| style      | char(25)    | YES  |     | NULL    |       |
+------------+-------------+------+-----+---------+-------+
8 rows in set (0.00 sec)
```

## 6. CREATE CONTACTS TABLE

CREATE TABLE CONTACTS
(CUSTID VARCHAR(20),
PHONE VARCHAR(12),
FOREIGN KEY (CUSTID) REFERENCES CUSTOMER(CUSTID) ON DELETE CASCADE);

```
mysql> desc contacts;
+--------+-------------+------+-----+---------+-------+
| Field  | Type        | Null | Key | Default | Extra |
+--------+-------------+------+-----+---------+-------+
| CUSTID | varchar(20) | YES  | MUL | NULL    |       |
| PHONE  | varchar(12) | YES  |     | NULL    |       |
+--------+-------------+------+-----+---------+-------+
2 rows in set (0.00 sec)
```

# 4.6 INSERTION OF TUPLES

## 1. INSERTION OF GALLERY TABLE

INSERT INTO GALLERY VALUES('NG123','National Gallery', 'Washington');
INSERT INTO GALLERY VALUES('BM123','British Museum', 'London');
INSERT INTO GALLERY VALUES('JG123','Jahangir Gallery', 'Mumbai');
INSERT INTO GALLERY VALUES('TLM123','The Louvre Museum', 'Paris');
INSERT INTO GALLERY VALUES('MM123','Metropolitan Museum', 'New York');

```
mysql> select * from gallery;
+--------+--------------------+------------+
| gid    | gname              | location   |
+--------+--------------------+------------+
| NG123  | NATIONAL GALLERY   | Washington |
| BM123  | BRITISH MUSEUM     | London     |
| JG123  | JAHANGIR GALLERY   | Mumbai     |
| TLM123 | THE LOUVRE MUSEUM  | Paris      |
| MM123  | METROPOLITAN MUSEUM | New York   |
+--------+--------------------+------------+
5 rows in set (0.00 sec)
```

## 2. INSERTION OF EXHIBITION TABLE

INSERT INTO EXHIBITION VALUES('G123','NG123','2018-12-01','2018-12-15');
INSERT INTO EXHIBITION VALUES('H123','BM123','2018-12-21','2019-01-05');
INSERT INTO EXHIBITION VALUES('I123','MM123','2019-01-25','2019-02-05');
INSERT INTO EXHIBITION VALUES('J123','TLM123','2018-12-15','2019-01-15');
INSERT INTO EXHIBITION VALUES('K123','JG123','2019-03-09','2019-03-27');

```
mysql> select * from exhibition;
+------+--------+------------+------------+
| eid  | gid    | startdate  | enddate    |
+------+--------+------------+------------+
| H123 | BM123  | 2018-12-21 | 2019-01-05 |
| I123 | MM123  | 2019-01-25 | 2019-02-05 |
| G123 | NG123  | 2018-12-01 | 2018-12-15 |
| J123 | TLM123 | 2018-12-15 | 2019-01-15 |
| K123 | JG123  | 2019-03-09 | 2019-03-27 |
+------+--------+------------+------------+
5 rows in set (0.00 sec)
```

## 3. INSERTION OF ARTWORK TABLE

INSERT INTO ARTWORK
VALUES('AW12','Mona Lisa','1503','Painting','10,00,00,000','G123','NG123','AD11');
INSERT INTO ARTWORK
VALUES('AW34','Poppies','1873','Painting','1,50,00,000','H123','MM123','AD22');
INSERT INTO ARTWORK
VALUES('AW56','Guernica','1937','Painting','2,50,00,000','I123','TLM123','AD55');

```
INSERT INTO ARTWORK
VALUES('AW78','The Night Watch','1642','Painting','90,00,000','J123','BM123','AD88');
INSERT INTO ARTWORK
VALUES('AW90','Two Sisters','2010','Sculpture','2,00,000','K123','JG123','AD00');
```

```
mysql> select * from artwork;
+-------+-----------------+------+-------------+--------------+------+--------+----------+
| artid | title           | year | type_of_art | price        | eid  | gid    | artistid |
+-------+-----------------+------+-------------+--------------+------+--------+----------+
| AW12  | Mona Lisa       | 1503 | Painting    | 10,00,00,000 | G123 | NG123  | AD11     |
| AW34  | Poppies         | 1873 | Painting    | 1,50,00,000  | H123 | MM123  | AD22     |
| AW56  | Guernica        | 1937 | Painting    | 2,50,00,000  | I123 | TLM123 | AD55     |
| AW78  | The Night Watch | 1642 | Painting    | 90,00,000    | J123 | BM123  | AD88     |
| AW90  | Two Sisters     | 2010 | Sculpture   | 2,00,000     | K123 | JG123  | AD00     |
+-------+-----------------+------+-------------+--------------+------+--------+----------+
5 rows in set (0.00 sec)
```

## 4. INSERTION OF CUSTOMER TABLE

```
INSERT INTO CUSTOMER VALUES
('AT2000','MM123','AD22','Akshay','Thakur','2000-04-16','New York');
INSERT INTO CUSTOMER
VALUES('AR1998','TLM123','AD55','Ashutosh','Ranjan','1998-02-04','Paris');
INSERT INTO CUSTOMER
VALUES('AD1998','BM123','AD88','Ayush','Dhar','1998-09-28','London');
INSERT INTO CUSTOMER
VALUES('AM1994','JG123','AD00','Avanish','Mehta','1994-10-05','Mumbai');
INSERT INTO CUSTOMER VALUES
('PM1996','NG123','AD11','Prashant','Mehta','1996-06-18','Washington');
```

```
mysql> select * from customer;
+--------+--------+-------+----------+--------+------------+------------+
| custid | gid    | artid | fname    | lname  | dob        | address    |
+--------+--------+-------+----------+--------+------------+------------+
| AT2000 | MM123  | AD22  | Akshay   | Thakur | 2000-04-16 | New York   |
| AR1998 | TLM123 | AD55  | Ashutosh | Ranjan | 1998-02-04 | Paris      |
| AD1998 | BM123  | AD88  | Ayush    | Dhar   | 1998-09-28 | London     |
| AM1994 | JG123  | AD00  | Avanish  | Mehta  | 1994-10-05 | Mumbai     |
| PM1996 | NG123  | AD11  | Prashant | Mehta  | 1996-06-18 | Washington |
+--------+--------+-------+----------+--------+------------+------------+
5 rows in set (0.00 sec)
```

## 5. INSERTION OF ARTIST TABLE

```
INSERT INTO ARTIST
VALUES('ART1','MM123','AT2000','AD22','Georgia','O Keeffe','USA','Oil on Canvas');
INSERT INTO ARTIST
VALUES('ART2','TLM123','AR1998','AD55','Pablo','Picasso','Spain','Analytic Cubism');
INSERT INTO ARTIST VALUES
('ART3','BM123','AD1998','AD88','Rembrandt','van Rijn','Netherlands','Oil Painting');
```

```
INSERT INTO ARTIST
VALUES('ART4','JG123','AM1994','AD00','Theodore','Chasseriau','France','Oil Painting');
INSERT INTO ARTIST
VALUES('ART5','NG123','PM1996','AD11','Leonardo','da Vinci','Italy','High Renaissance');
```

```
mysql> select * from artist;
+----------+--------+--------+------+-----------+------------+-------------+-----------------+
| artistid | gid    | custid | eid  | fname1    | lname1     | birthplace  | style           |
+----------+--------+--------+------+-----------+------------+-------------+-----------------+
| ART1     | MM123  | AT2000 | AD22 | Georgia   | O Keeffe   | USA         | Oil on Canvas   |
| ART2     | TLM123 | AR1998 | AD55 | Pablo     | Picasso    | Spain       | Analytic Cubism |
| ART3     | BM123  | AD1998 | AD88 | Rembrandt | van Rijn   | Netherlands | Oil Painting    |
| ART4     | JG123  | AM1994 | AD00 | Theodore  | Chasseriau | France      | Oil Painting    |
| ART5     | NG123  | PM1996 | AD11 | Leonardo  | da Vinci   | Italy       | High Renaissance |
+----------+--------+--------+------+-----------+------------+-------------+-----------------+
5 rows in set (0.00 sec)
```

## 6. INSERTION OF CONTACTS TABLE

```
INSERT INTO CONTACTS VALUES('AT2000', '9456805776');
INSERT INTO CONTACTS VALUES('AR1998', '8073271337');
INSERT INTO CONTACTS VALUES('AD1998', '9980904736');
INSERT INTO CONTACTS VALUES('AM1994', '7737564076');
INSERT INTO CONTACTS VALUES('PM1996', '8002391707');
```

```
mysql> select * from contacts;
+--------+------------+
| CUSTID | PHONE      |
+--------+------------+
| AT2000 | 9456805776 |
| AR1998 | 8073271337 |
| AD1998 | 9980904736 |
| AM1994 | 7737564076 |
| PM1996 | 8002391707 |
+--------+------------+
5 rows in set (0.00 sec)
```

## 4.7 CREATION OF TRIGGERS

The trigger is made such that when a new record is inserted into a Gallery table, it automatically changes the lowercase name into uppercase in the backend.

**TRIGGER ON GALLERY TABLE TO CHANGING NAME TO UPPERCASE**

```
DELIMITER $$
CREATE TRIGGER UPPERCASE
BEFORE INSERT on Gallery
FOR EACH ROW
BEGIN
SET NEW.gname=UPPER(NEW.gname);
END$$
```

```
mysql> DELIMITER $$
mysql>                CREATE TRIGGER UPPERCASE
    ->                BEFORE INSERT on Gallery
    ->                FOR EACH ROW
    ->                BEGIN
    ->                SET NEW.gname=UPPER(NEW.gname);
    ->                END$$
Query OK, 0 rows affected (0.09 sec)
```

```
mysql> select * from Gallery;
+--------+---------------------+------------+
| gid    | gname               | location   |
+--------+---------------------+------------+
| NG123  | NATIONAL GALLERY    | Washington |
| BM123  | BRITISH MUSEUM      | London     |
| JG123  | JAHANGIR GALLERY    | Mumbai     |
| TLM123 | THE LOUVRE MUSEUM   | Paris      |
| MM123  | METROPOLITAN MUSEUM | New York   |
+--------+---------------------+------------+
5 rows in set (0.23 sec)
```

# 4.8 CREATION OF STORED PROCEDURES

This stored procedure is used to find the age of the given customer using date of birth and current date.

## STORED PROCEDURE ON CUSTOMER TABLE TO FIND AGE

```
DELIMITER $$

CREATE PROCEDURE GetAge()

BEGIN

SELECT *, year(CURRENT_DATE())-year(DOB) as age from CUSTOMER;

END$$
```

```
mysql> DELIMITER $$
mysql> CREATE PROCEDURE GetAge()
    -> BEGIN
    -> SELECT *, year(CURRENTDATE())-year(DOB) as age from CUSTOMER;
    -> END$$
Query OK, 0 rows affected (0.00 sec)
```

```
mysql> call GetAge();
+--------+--------+-------+----------+--------+------------+------------+------+
| custid | gid    | artid | fname    | lname  | dob        | address    | age  |
+--------+--------+-------+----------+--------+------------+------------+------+
| AT2000 | MM123  | AD22  | Akshay   | Thakur | 2000-04-16 | New York   |   18 |
| AR1998 | TLM123 | AD55  | Ashutosh | Ranjan | 1998-02-04 | Paris      |   20 |
| AD1998 | BM123  | AD88  | Ayush    | Dhar   | 1998-09-28 | London     |   20 |
| AM1994 | JG123  | AD00  | Avanish  | Mehta  | 1994-10-05 | Mumbai     |   24 |
| PM1996 | NG123  | AD11  | Prashant | Mehta  | 1996-06-18 | Washington |   22 |
+--------+--------+-------+----------+--------+------------+------------+------+
5 rows in set (0.00 sec)

Query OK, 0 rows affected (0.01 sec)
```

# CHAPTER 5

## RESULTS

This section describes the screens of “Art Gallery Database”. The snapshots are shown below for each module.

### SNAPSHOTS

- This is the main page that shows all the operations which are present in Art Gallery Database.

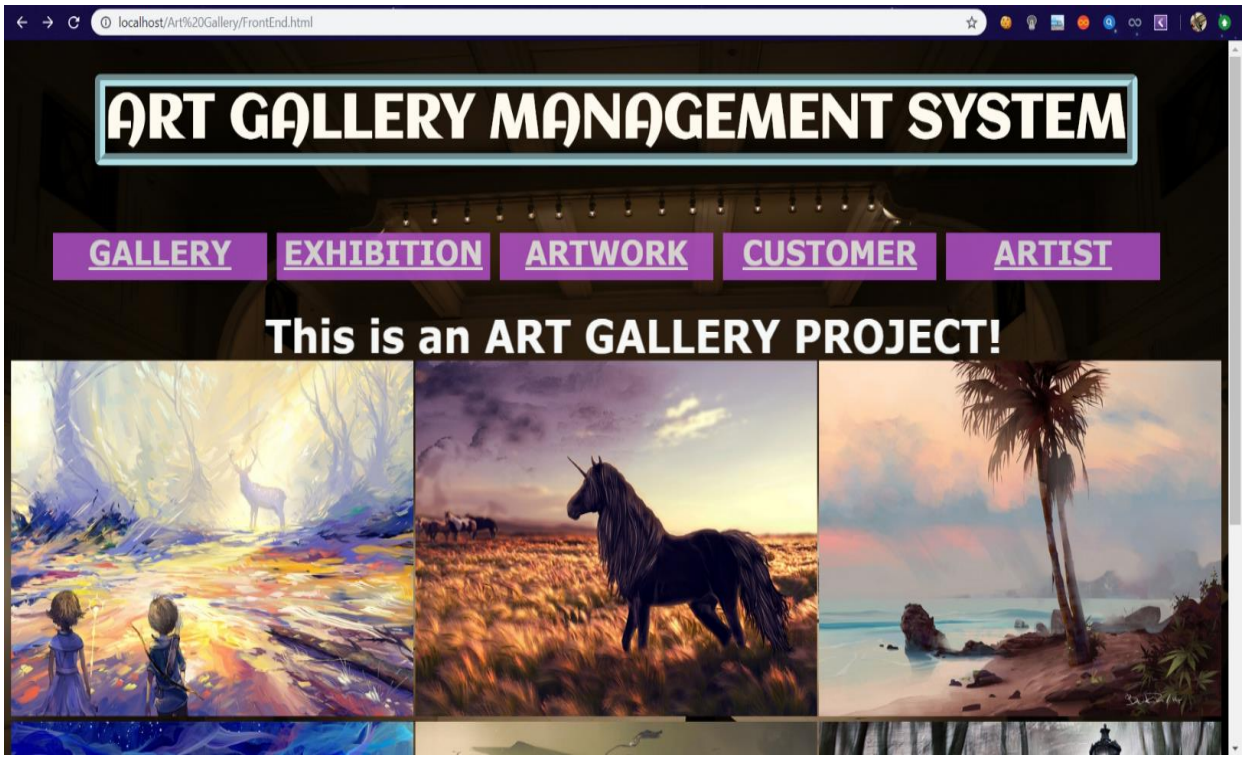



**FIGURE 7.1 ART GALLERY DATABASE MAIN PAGE**

- The selection page is the next displayed as soon as the table is selected in Main Page. Gallery table contains Insert, Search, Display and Delete tables where values can be inserted, deleted, etc.

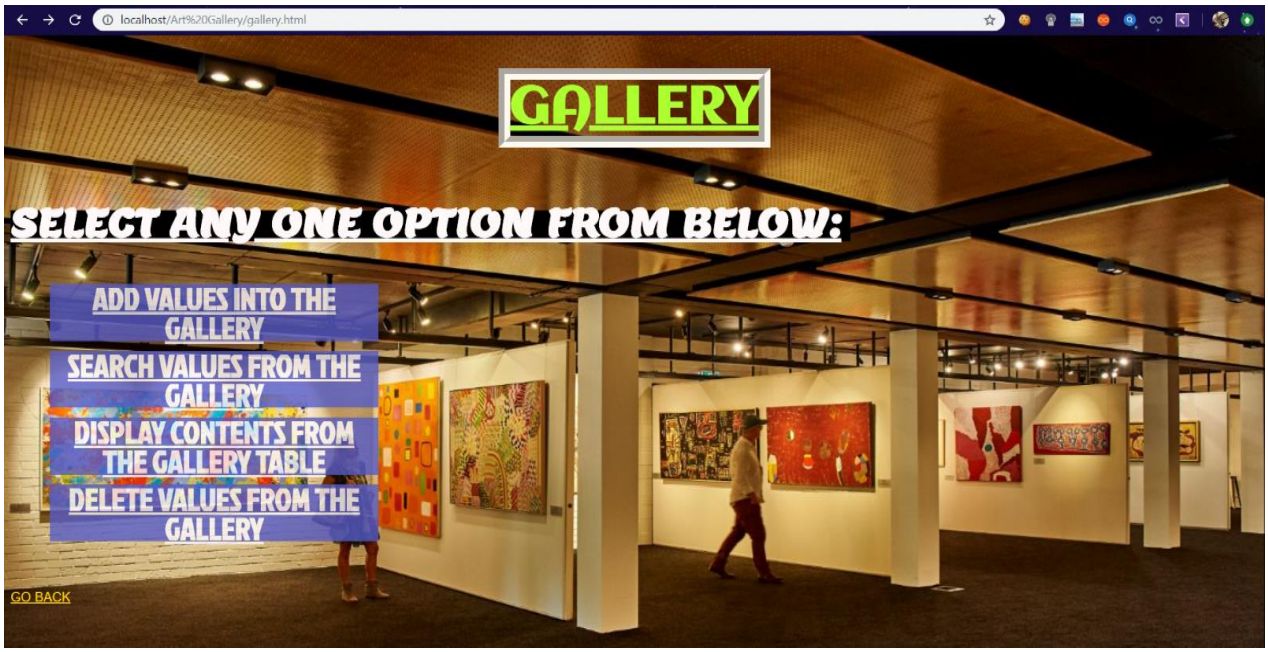



**FIGURE 7.2 GALLERY TABLE SELECTION PAGE**

- This snapshot shows the insertion page of Gallery Table. This front end page supports the insertion of all the attributes in Gallery table like GID, GName and Location.

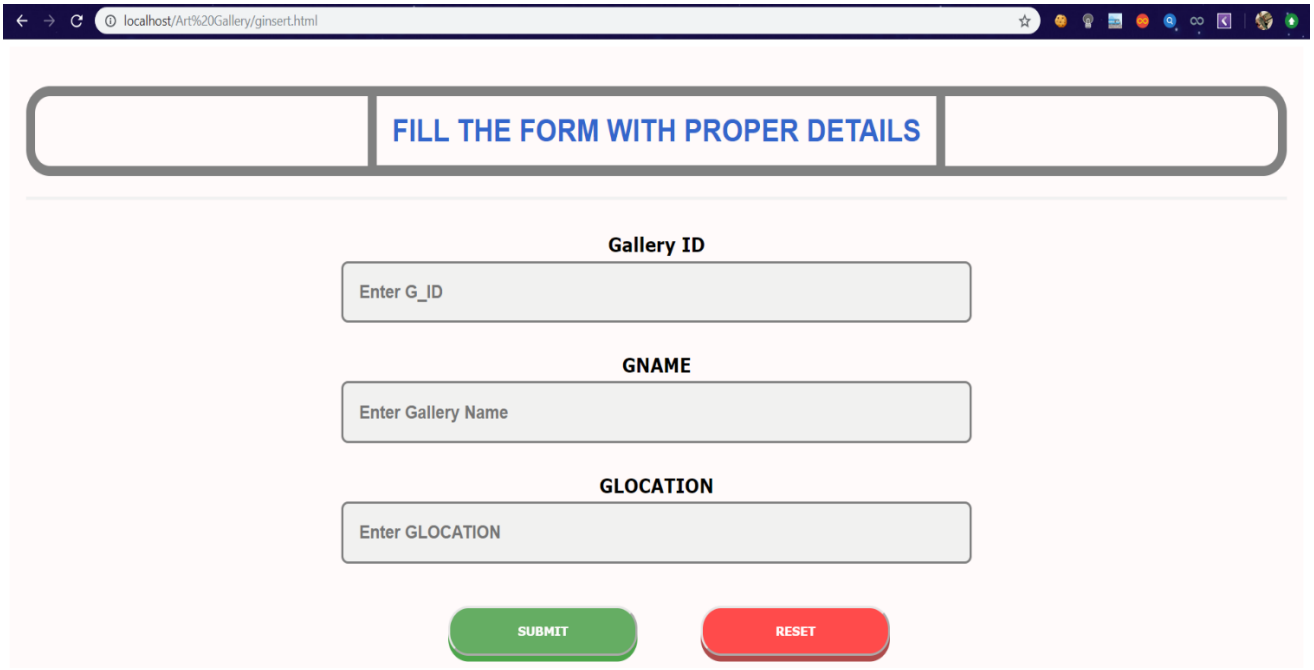



**FIGURE 7.3: INSERTION PAGE OF GALLERY TABLE**

- This snapshot contains all the details of given GID by using Search method. Search table of Gallery contains the values like GID, GName and Location where it is searched by GID which is a primary key for this table.

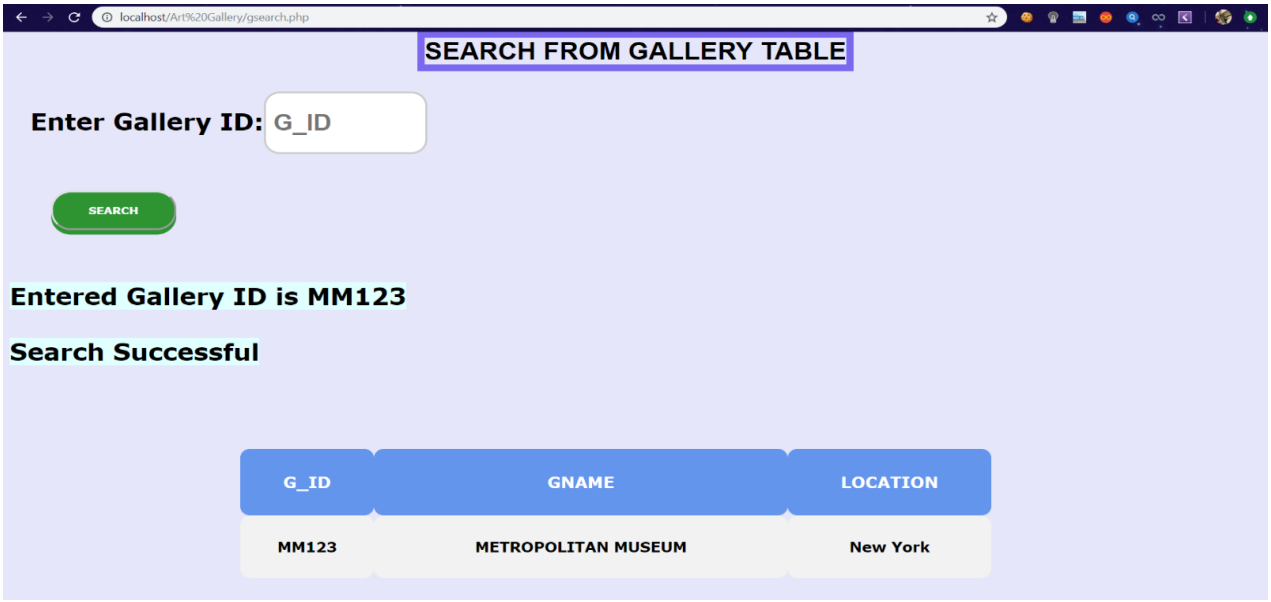



**FIGURE 7.4: SEARCH PAGE OF GALLERY TABLE**

- This snapshots displays all the values entered in that table. Here, Gallery table is displayed on frontend page by using proper display query.

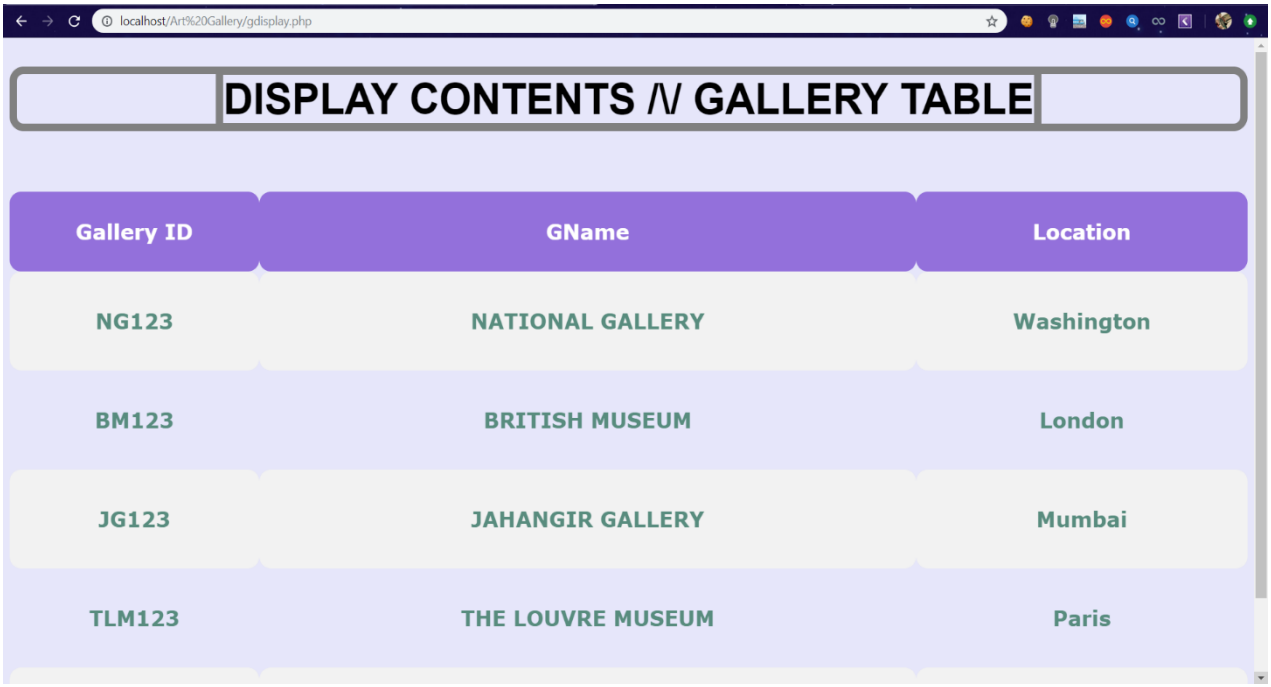



**FIGURE 7.5: DISPLAY PAGE OF GALLERY TABLE**

- This snapshot shows the working status of Delete Page of Gallery Table. In this page, GID is used as a parameter to delete certain record with all of their attributes in that rows. If the entered value is present in database then the value will be deleted.

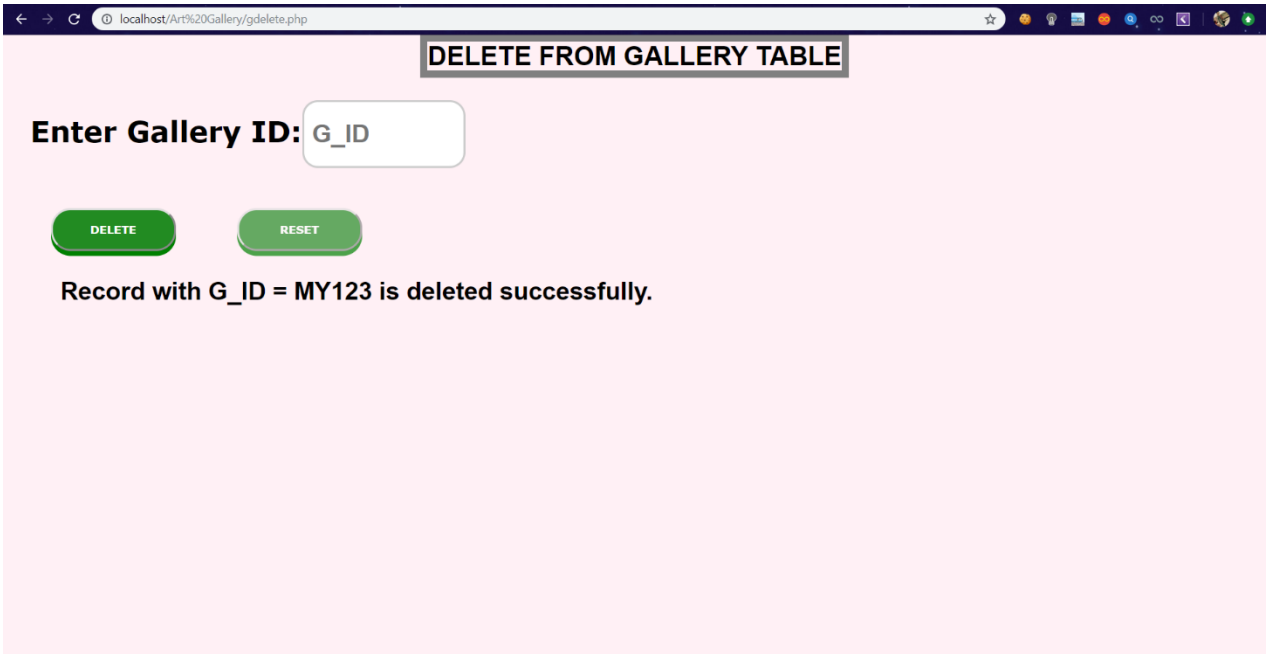
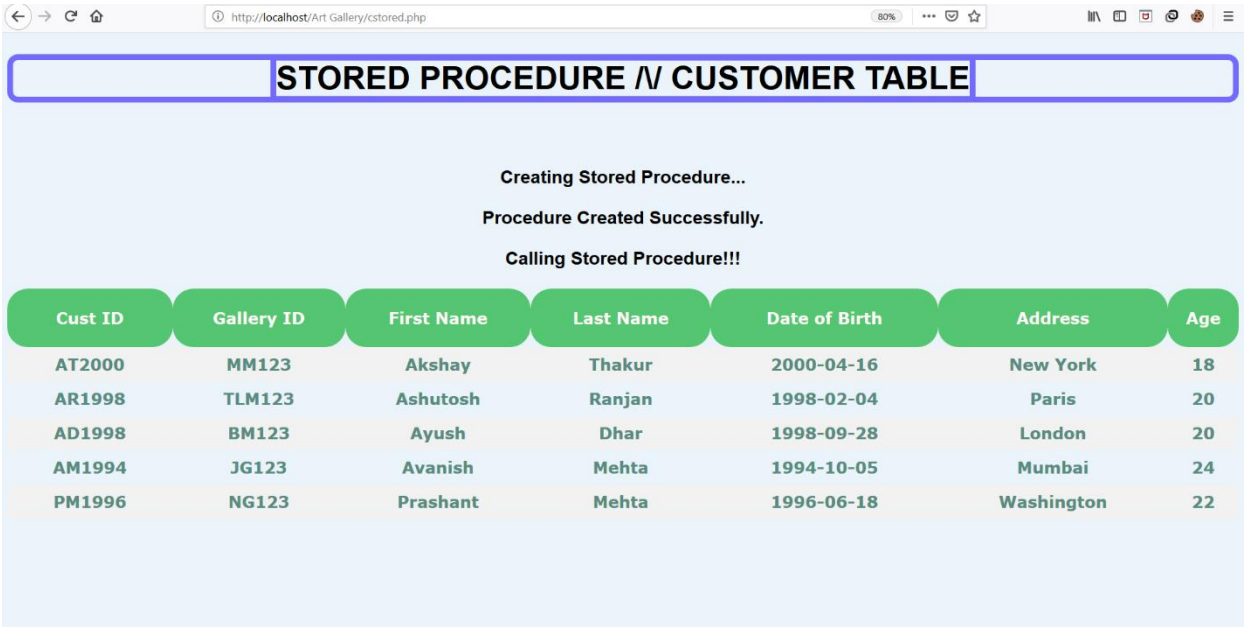



**FIGURE 7.6: DELETION PAGE OF GALLERY TABLE**

- This snapshot shows the working status of Stored Procedure of Customer table. By using stored procedure, we're calculating the Age of Customer using Date of Birth as parameter.




| Cust ID | Gallery ID | First Name | Last Name | Date of Birth | Address | Age |
|---|---|---|---|---|---|---|
| AT2000 | MM123 | Akshay | Thakur | 2000-04-16 | New York | 18 |
| AR1998 | TLM123 | Ashutosh | Ranjan | 1998-02-04 | Paris | 20 |
| AD1998 | BM123 | Ayush | Dhar | 1998-09-28 | London | 20 |
| AM1994 | JG123 | Avanish | Mehta | 1994-10-05 | Mumbai | 24 |
| PM1996 | NG123 | Prashant | Mehta | 1996-06-18 | Washington | 22 |

**FIGURE 7.7: STORED PROCEDURE PAGE OF CUSTOMER TABLE**

# CONCLUSION

A database was created for a market that can use it for keeping track on art gallery Galleries are divided into many art galleries. Galleries have different names, locations, etc. Each gallery will have different exhibitions and each exhibition will have a start and end date. The galleries will have different artist displaying their artwork. The model can also be adapted to meet other purposes and thus be used for other projects. The database structure is quite simple, which makes it easy for also other programmers to understand it. In conclusion, a database is a far more efficient mechanism to store and organize data than spreadsheets it allows for a centralized facility that can easily be modified and quickly shared among multiple users. Having a web based front end removes the requirement of users having to understand and use a database directly, and allows users to connect from anywhere with an internet connection and a basic web browser. It also allows the possibility of queries to obtain information for various surveys. Due to the number of users reading and modifying student data in the department, it is an ideal use for such a system.

# REFERENCES


I. Fundamentals of Database System, 7th Edition
-By Elmasri Ramez and Navathe Shamkanth

II. PHP and MySQL Web Development
-By Luke Welling and Laura Thompson

III. HTML & CSS: Design and Build Web Sites
-By John Duckett

IV. For Front End Code and CSS styling
a. https://www.w3schools.com/html
b. https://www.stackoverflow.com
c. https://www.tutorialspoint.com

V. For MySQL references
a. https://www.youtube.com
b. https://www.udemy.com