PROJECT REPORT

Submitted for

DATABASE MANAGEMENT SYSTEM
(UCS310)

TOPIC: ART GALLERY DATABASE MANAGEMENT SYSTEM

Submitted by:

Nutan 102203837

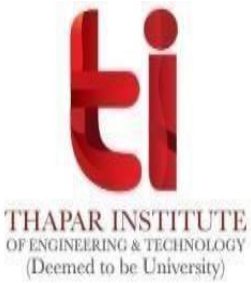
Ridhima Sharma 102203709

Ruhani Grover 1020203864

BE Second Year Batch – 2CO34

Submitted to:

Dr. Amrita Dahiya



Thapar Institute of Engineering and Technology, Patiala

INDEX

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 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 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.

- The database project aims to enhance the overall customer experience by providing easy access to information about artworks, exhibitions, and artists. By having a centralized database, customers can easily search for specific artworks, view past exhibitions, and learn more about their favorite artists, ultimately improving their satisfaction and engagement with the gallery.

ER DIAGRAM

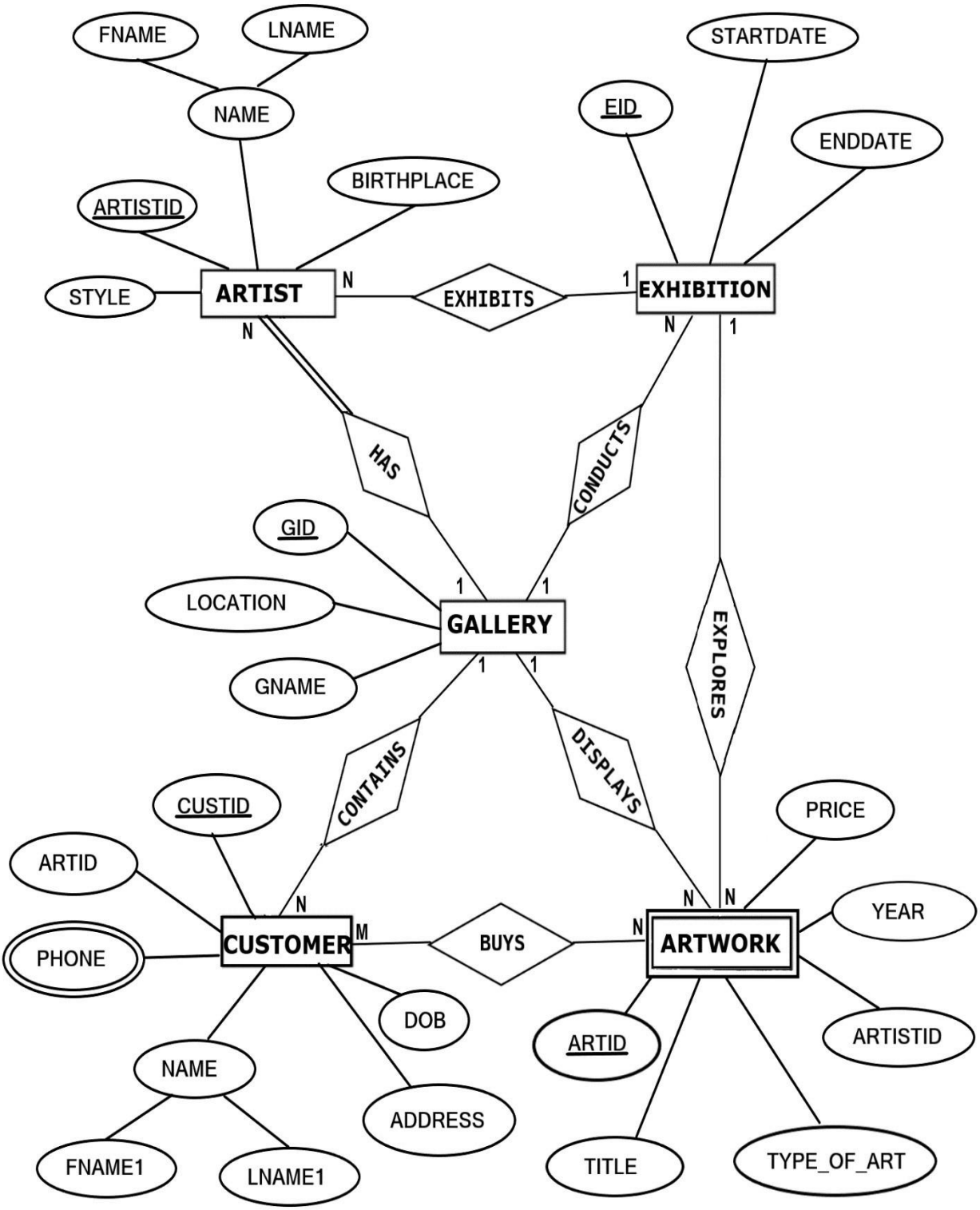


FIGURE 4.1: ER DIAGRAM of ART GALLERY DATABASE

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

Foreign Key

STEP 2 : Mapping of Weak Entity Types

ARTWORK

ARTID	**ARTISTID**	TITLE	TYPE_OF_ART	YEAR	PRICE

Foreign Key

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**

Foreign Key

ARTIST

ARTISTID	FNAME	LNAME	BIRTHPLACE	STYLE	**EID**	**GID**	**CUSTID**

(Foreign Keys)

CUSTOMER

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

Foreign Key **Foreign Key**

ARTWORK

ARTID	**ARTISTID**	TITLE	TYPE_OF_ART	YEAR	PRICE	**EID**	**GID**

Foreign Key **(Foreign Keys)**

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.

SCHEMA DIAGRAM

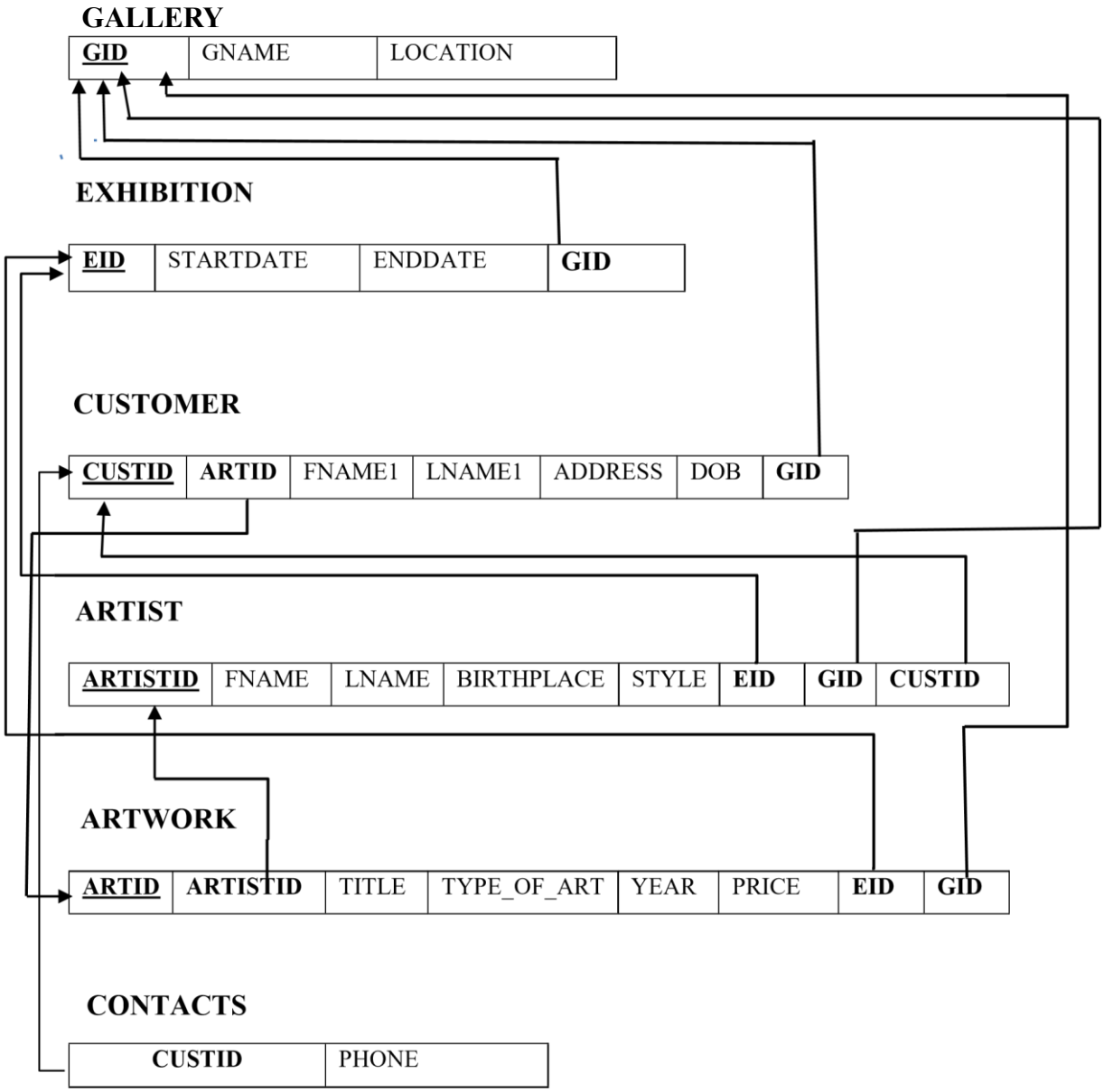


FIGURE 4.3: SCHEMA DIAGRAM

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.

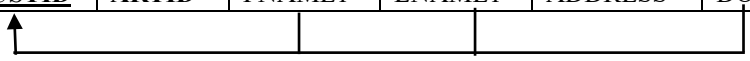
2. Second Normal Form (2NF)

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

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.

CREATION OF TABLES

1. CREATING GALLERY TABLE

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

LNAME VARCHAR(20),
BIRTHPLACE VARCHAR(20),
STYLE VARCHAR(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)
```

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',DATE'2018-12-01',
DATE '2018-12-15');
INSERT INTO EXHIBITION VALUES('H123','BM123',DATE'2018-12-21',
DATE'2019-01-05');
INSERT INTO EXHIBITION VALUES('I123','MM123',DATE'2019-01-25',
DATE'2019-02-05');
INSERT INTO EXHIBITION VALUES('J123','TLM123',DATE'2018-12-15',
DATE'2019-01-15');
INSERT INTO EXHIBITION VALUES('K123','JG123',DATE'2019-03-09',
DATE'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', 'MonaLisa', '1503', 'Painting', '10,00,00,000',
'G123', 'NG123', 'AD11');
```

INSERT INTO ARTWORK VALUES ('AW34', 'Poppies', '1873', 'Painting', '1,50,00,000', 'H123', 'MM1 23','AD22');
INSERT INTO ARTWORK VALUES ('AW56', 'Guernica', '1937', 'Painting', '2,50,00,000', 'I123', 'TL M123', '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',DATE'2000-04-16','New York');
INSERT INTO CUSTOMER VALUES ('AR1998','TLM123','AD55','Ashutosh','Ranjan',DATE'1998-02-04','Paris');
INSERT INTO CUSTOMER VALUES('AD1998','BM123','AD88','Ayush','Dhar',DATE'1998-0928','London');
INSERT INTO CUSTOMER VALUES('AM1994','JG123','AD00','Avanish','Mehta',DATE'1994-1005','Mumbai');
INSERT INTO CUSTOMER VALUES ('PM1996','NG123','AD11','Prashant','Mehta',DATE'1996-0618','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)
```

CREATION OF PROCEDURES

1. Creation of procedure to add values in the table EXHIBITION

```
SQL> CREATE OR REPLACE PROCEDURE add_exhibition (
  2      p_eid IN VARCHAR2,
  3      p_gid IN VARCHAR2,
  4      p_start_date IN DATE,
  5      p_end_date IN DATE
  6  )
  7  IS
  8  BEGIN
  9      INSERT INTO EXHIBITION (EID, GID, STARTDATE, ENDDATE)
 10      VALUES (p_eid, p_gid, p_start_date, p_end_date);
 11
 12      DBMS_OUTPUT.PUT_LINE('Exhibition added successfully');
 13  EXCEPTION
 14      WHEN OTHERS THEN
 15          DBMS_OUTPUT.PUT_LINE('Error: ' || SQLERRM);
 16  END add_exhibition;
 17  /

Procedure created.
```

```
SQL> execute add_exhibition('ridhima','AB123',DATE'2004-12-11',DATE'2011-11-23');

PL/SQL procedure successfully completed.

SQL> select * from EXHIBITION;

EID                  GID                  STARTDATE ENDDATE
-------------------- -------------------- --------- ---------
G123                 NG123                01-DEC-18 15-DEC-18
H123                 BM123                21-DEC-18 05-JAN-19
I123                 MM123                25-JAN-19 05-FEB-19
J123                 TLM123               15-DEC-18 15-JAN-19
K123                 JG123                09-MAR-19 27-MAR-19
ridhima              AB123                11-DEC-04 23-NOV-11

6 rows selected.
```

2. Creation of procedure to add values in the table GALLERY

```
SQL> CREATE OR REPLACE PROCEDURE add_gallery(
  2      p_gid IN VARCHAR2,
  3      p_gname IN VARCHAR2,
  4      p_location IN VARCHAR2
  5  )
  6  IS
  7  BEGIN
  8      INSERT INTO GALLERY (GID, GNAME, LOCATION)
  9      VALUES (p_gid, p_gname, p_location);
 10      COMMIT;
 11      DBMS_OUTPUT.PUT_LINE('Gallery added successfully');
 12  EXCEPTION
 13      WHEN OTHERS THEN
 14          DBMS_OUTPUT.PUT_LINE('Error: ' || SQLERRM);
 15  END add_gallery;
 16  /

Procedure created.
```

```
SQL> EXECUTE add_gallery('ed2','nutan','patiala');

PL/SQL procedure successfully completed.

SQL> select*from GALLERY;

GID                  GNAME                LOCATION
-------------------- -------------------- --------------------
NG123                National             Washington
                     Gallery

BM123                British Museum       London
JG123                Jahangir Gallery     Mumbai
TLM123               The Louvre           Paris
                     Museum

MM123                Metropolitan Museum  New York
gdg34                nutan                patiala
ed2                  nutan                patiala

7 rows selected.
```

3. Creation of procedure to add values in the table CUSTOMER

```
SQL> CREATE OR REPLACE PROCEDURE add_customer(
  2      p_custid IN VARCHAR2,
  3  p_gid IN VARCHAR2,
  4   p_fname IN VARCHAR2,
  5  p_lname IN VARCHAR2,
  6   p_dob IN DATE,
  7   p_address IN VARCHAR2
  8  )
  9  IS
 10  BEGIN
 11  INSERT INTO CUSTOMER2 (CUSTID, GID, FNAME1, LNAME1, DOB, ADDRESS)
 12  VALUES (p_custid, p_gid, p_fname, p_lname, p_dob, p_address);
 13  DBMS_OUTPUT.PUT_LINE('Customer added successfully');
 14  EXCEPTION
 15   WHEN OTHERS THEN
 16  DBMS_OUTPUT.PUT_LINE('Error: ' || SQLERRM);
 17  END add_customer;
 18  /

Procedure created.

SQL> execute add_customer('abv124','AB123','ridhima','sharma',DATE'2004-01-25','patiala');

PL/SQL procedure successfully completed.
```

```
SQL> select * from CUSTOMER2;

CUSTID              GID                 FNAME1
------------------- ------------------- --------------------
LNAME1              DOB       ADDRESS
------------------- --------- --------------------
AT2000              MM123               Akshay
Thakur              16-APR-00 New York

AR1998              TLM123              Ashutosh
Ranjan              04-FEB-98 Paris

AD1998              BM123               Ayush
Dhar                28-SEP-98 London


CUSTID              GID                 FNAME1
------------------- ------------------- --------------------
LNAME1              DOB       ADDRESS
------------------- --------- --------------------
AM1994              JG123               Avanish
Mehta               05-OCT-94 Mumbai

PM1996              NG123               Prashant
Mehta               18-JUN-96 Washington

abv124              AB123               ridhima
sharma              25-JAN-04 patiala


6 rows selected.
```

CREATION OF FUNCTIONS

1. Creation of function to add values in the table EXHIBITION

```
SQL> CREATE OR REPLACE FUNCTION delete_exhibition_func (
  2      p_eid IN VARCHAR2
  3  )
  4  RETURN VARCHAR2
  5  IS
  6  BEGIN
  7      DELETE FROM EXHIBITION
  8      WHERE EID = p_eid;
  9
 10      RETURN 'Exhibition deleted successfully';
 11  EXCEPTION
 12      WHEN OTHERS THEN
 13          RETURN 'Error: ' || SQLERRM;
 14  END delete_exhibition_func;
 15  /

Function created.
```

```
SQL> DECLARE
  2      v_result VARCHAR2(100);
  3  BEGIN
  4      -- Call the function with the desired parameter(s)
  5      v_result := delete_exhibition_func('g123');
  6
  7      -- Output the result
  8      DBMS_OUTPUT.PUT_LINE(v_result);
  9  END;
 10  /

PL/SQL procedure successfully completed.

SQL> select * from EXHIBITION;

EID                  GID                  STARTDATE ENDDATE
-------------------- -------------------- --------- ---------
G123                 NG123                01-DEC-18 15-DEC-18
H123                 BM123                21-DEC-18 05-JAN-19
I123                 MM123                25-JAN-19 05-FEB-19
J123                 TLM123               15-DEC-18 15-JAN-19
K123                 JG123                09-MAR-19 27-MAR-19
ridhima              AB123                11-DEC-04 23-NOV-11

6 rows selected.
```

2. Creation of function to get data from table CONTACTS

```
SQL> CREATE OR REPLACE FUNCTION get_contacts_func
  2  RETURN SYS_REFCURSOR
  3  IS
  4      contacts_cursor SYS_REFCURSOR;
  5  BEGIN
  6      OPEN contacts_cursor FOR
  7          SELECT * FROM CONTACTS;
  8
  9      RETURN contacts_cursor;
 10  END get_contacts_func;
 11  /

Function created.
```

```
SQL> SELECT*FROM CONTACTS;

CUSTID               PHONE
-------------------- ------------
AT2000               9423453456
AR1998               8073271337
AD1998               9980904736
AM1994               7737564076
PM1996               8002391707
```

CREATION OF CURSORS

1. Cursor to fetch data from table CONTACTS

```
SQL> DECLARE
  2      v_contacts_cursor SYS_REFCURSOR;
  3      v_custid CONTACTS.CUSTID%TYPE;
  4      v_phone CONTACTS.PHONE%TYPE;
  5  BEGIN
  6      v_contacts_cursor := get_contacts_func;
  7
  8      LOOP
  9          FETCH v_contacts_cursor INTO v_custid, v_phone;
 10          EXIT WHEN v_contacts_cursor%NOTFOUND;
 11          DBMS_OUTPUT.PUT_LINE('Customer ID: ' || v_custid || ', Phone: ' || v_phone);
 12      END LOOP;
 13
 14      CLOSE v_contacts_cursor;
 15  END;
 16  /

PL/SQL procedure successfully completed.
```

```
SQL> VAR contacts_cursor REFCURSOR;
SQL>
SQL> -- Execute the function and store the cursor result in the variable
SQL> BEGIN
  2      :contacts_cursor := get_contacts_func;
  3  END;
  4  /

PL/SQL procedure successfully completed.

SQL>
SQL> -- Print the results fetched from the cursor
SQL> PRINT contacts_cursor;

CUSTID               PHONE
-------------------- ------------
AT2000               9423453456
AR1998               8073271337
AD1998               9980904736
AM1994               7737564076
PM1996               8002391707
```

2. Cursor to fetch all the data from table GALLERY

```
SQL> CREATE OR REPLACE FUNCTION get_gallery_cursor
  2  RETURN SYS_REFCURSOR
  3  IS
  4      gallery_cursor SYS_REFCURSOR;
  5  BEGIN
  6      OPEN gallery_cursor FOR
  7          SELECT * FROM GALLERY;
  8
  9      RETURN gallery_cursor;
 10  END get_gallery_cursor;
 11  /

Function created.
```

```
SQL> DECLARE
  2      gallery_cur SYS_REFCURSOR;
  3      gallery_rec GALLERY%ROWTYPE;
  4  BEGIN
  5      -- Call the function to get the cursor
  6      gallery_cur := get_gallery_cursor;
  7
  8      -- Fetch rows from the cursor
  9      LOOP
 10          FETCH gallery_cur INTO gallery_rec;
 11          EXIT WHEN gallery_cur%NOTFOUND;
 12
 13          -- Print or process the fetched row
 14          DBMS_OUTPUT.PUT_LINE('Gallery ID: ' || gallery_rec.GID || ', Name: ' || gallery_rec.GNAME || ', Location: ' || gallery_rec.LOCATION);
 15      END LOOP;
 16
 17      -- Close the cursor
 18      CLOSE gallery_cur;
 19  END;
 20  /

PL/SQL procedure successfully completed.
```

3. Cursor to get all the data from table EXHIBITION

```
SQL> CREATE OR REPLACE FUNCTION get_exhibition_cursor
  2  RETURN SYS_REFCURSOR
  3  IS
  4      exhibition_cur SYS_REFCURSOR;
  5  BEGIN
  6      -- Open the cursor to fetch all rows from the EXHIBITION table
  7      OPEN exhibition_cur FOR
  8          SELECT * FROM EXHIBITION;
  9
 10      -- Return the cursor
 11      RETURN exhibition_cur;
 12  END get_exhibition_cursor;
 13  /

Function created.
```

```
SQL> DECLARE
  2   exhibition_cursor SYS_REFCURSOR;
  3   v_eid EXHIBITION.EID%TYPE;
  4  v_startdate EXHIBITION.STARTDATE%TYPE;
  5   v_enddate EXHIBITION.ENDDATE%TYPE;
  6   v_gid EXHIBITION.GID%TYPE;
  7  BEGIN
  8  exhibition_cursor := get_exhibition_cursor;END;
  9  /

PL/SQL procedure successfully completed.
```

CONCLUSION

The creation of a dedicated database tailored for the art gallery market represents a pivotal step towards efficient management and organization of gallery-related data. Art galleries, as diverse entities, are subdivided into numerous individual spaces, each boasting unique characteristics such as names, locations, and other pertinent details. Within these galleries, a myriad of exhibitions takes place, each with its own distinct start and end dates, providing a platform for artists to showcase their work.

Central to the database's functionality is its ability to track and manage the multitude of artists who contribute to the gallery's vibrant tapestry. Through the database, comprehensive profiles of artists and their respective artworks are maintained, fostering seamless administration of portfolios and inventory. This systematic approach not only streamlines operations within the gallery but also ensures a cohesive and organized representation of artists and their creations.

Moreover, the adaptability of the database model extends beyond the confines of art galleries, rendering it suitable for diverse projects across various industries. Its simplicity fosters accessibility, allowing programmers of varying proficiency levels to comprehend and navigate the database structure with ease. This versatility positions the database as a valuable asset, capable of addressing a spectrum of organizational needs beyond the realm of art galleries.

In comparison to conventional spreadsheets, the database emerges as a superior mechanism for data storage and organization, offering a centralized repository that can be effortlessly modified and shared among multiple users. By providing a web-based front end, the database eliminates the necessity for users to possess an in-depth understanding of database operations, enabling seamless access from any location with an internet connection and a basic web browser.

Furthermore, the database's query capabilities facilitate the extraction of valuable insights for various surveys and analyses, empowering stakeholders to make informed decisions based on comprehensive data. This functionality proves particularly advantageous in environments characterized by a high volume of users, such as academic departments, where efficient management of student data is paramount.

In conclusion, the implementation of a dedicated database for art galleries represents not only a paradigm shift towards enhanced efficiency, organization, and accessibility but also a transformative tool capable of catalyzing innovation across various sectors. Its adaptability, simplicity, and functionality render it indispensable for addressing a myriad of organizational needs, transcending the confines of the art market to cater to diverse industries and projects. By harnessing the power of centralized data management, the database empowers stakeholders to navigate complex information landscapes with ease, facilitating informed decision-making and fostering collaboration on a scale previously unimaginable. As technology continues to evolve, the database stands as a testament to the transformative potential of data-driven solutions, offering a blueprint for organizations seeking to optimize their operations, streamline workflows, and unlock new avenues for growth and development.