GROUP PROJECT

ART GALLERY MANAGEMENT SYSTEM

TECHNICAL REPORT

(BUAN 6320)

GROUP 5

Nandhakumar Raj Shankar (nxs230038)
Srikruthi Shilpika Reddy Gangapalli (sxg230007)
Sushya Sri Kalyan Teja Adapala (sxa230000)
Nandini Kalal (nxk230008)

JSOM - UTD

TABLE OF CONTENTS

INTRODUCTION

This Database Design Document outlines the strategic planning and implementation of a dynamic database tailored to meet the distinctive requirements of an Art Gallery. Covering essential entities including Artists, Artwork, Display, Exhibitions, Booking, and Visitors, the database is intricately crafted to establish meaningful connections between artists and exhibitions.

OVERVIEW

The central aim of this project is to develop an integrated database system that acts as the foundation for overseeing the management of various entities such as Artists, Artwork, Display, Exhibitions, Booking, and Visitors. This database is designed to foster a seamless connection between artists, exhibitions and visitors.

ASSUMPTIONS AND CONSIDERATIONS

This project simplifies art gallery management with these main assumptions:

- All entities: Artists, Exhibition and Visitors are assumed to belong to the same city and that is also the scope of this project.
- Artworks need not necessarily have an Artist claiming it as their own, some exist where their creators are either unknown or do not wish to disclose the information.
- A visitor only visits an exhibition once but is allowed to visit multiple exhibitions.
- Bulk bookings are not possible and every instance of booking guarantees entrance for only one person.
- The date format is assumed to be yyyy-mm-dd in the database.

Special considerations to this project:

- ER-Assistant has hard coded widths for the entity blocks. As such, some attributes titles may be cutoff.
- Within the Entity Relationship Diagram (ERD), foreign keys are identified with the “Attribute (FK)” notation.

BUSINESS RULES

1. Each Artist creates zero or more artworks.
2. Every artwork is created by only one artist. However, the artist information may be disclosed or not.
3. Each artwork can be at zero or many displays.
4. Every display is associated to only one artwork.
5. Each exhibition has at least one display.
6. Every display is associated to only one exhibition.
7. Each exhibition has zero or more bookings.
8. Every booking is associated with only one exhibition.
9. Each visitor can make zero or multiple bookings.
10. Every booking is associated to only one visitor.

ENTITY & ATTRIBUTE DESCRIPTIONS

Artist:

Entity Name: Artist

Entity Description: It represents the creators of the artworks featured in the gallery. It encompasses essential information about each artist.

Main Attributes:

- artist_id (Primary Key): Unique Identifier for the artist.
- email: Email address of the artist.
- first_name: First name of the artist.
- last_name: Last name of the artist.
- office_address: Office address of the artist.
- phone: Phone contact of the artist.
- zip: Zipcode of the artist.

Artwork:

Entity Name: Artwork

Entity Description: It encapsulates detailed information about each piece of art displayed in the gallery.

Main Attributes:

- art_id (Primary Key): Unique identifier for the artwork
- FK_ARTIST_artist_id (Foreign Key): Unique identifier for the artist who created the artwork.
- title: Title of the artwork.
- description: Description of the artwork.
- medium: Medium using which the artwork was created.
- first_display: Date on which the artwork was displayed for the first time.

Display:

Entity Name: Display

Entity Description: It represents the characteristics of the designated place where an artwork is to be presented.

Main Attributes:

- display_id (Primary Key): Unique identifier for the display.
- start_time: Starting time of the display.
- end_time: End time of the display.
- location: Location of the display.
- artist_presence: Presence of the artist at the display.
- FK_ARTWORK_art_id (Foreign Key): Art ID of the art displayed.
- FK_EXHIBITION_exhibition_id (Foreign Key): exhibition id of the exhibition for the display (Referring Exhibition).

Exhibition:

Entity Name: Exhibition

Entity Description: It represents the curated displays of artworks.

Main Attributes:

- exhibition_id (Primary Key): Unique identifier for the Exhibition.
- name: Name of the exhibition.
- venue: Venue of the exhibition.
- date: Date on which the exhibition happens.
- mode: Modality of the exhibition.

Booking:

Entity Name: Booking

Entity Description: It represents the characteristics of the booking made by a visitor.

Main Attributes:

- booking_id (Primary Key): Unique identifier for the Booking.
- time: Time of the booking.
- slot_start: Starting time slot of the booking.
- slot_end: Ending time slot of the booking.
- payment_method: method of payment used for the booking.
- FK_Exhibition_exhibition_id (Foreign Key): Exhibition id for the booking (Referring Exhibition).
- FK_Visitor_visitor_id (Foreign Key): Visitor id for the booking (Referring Visitor).

VISITOR:

Entity Name: Visitor

Entity Description: It represents the visitors to a gallery. It encompasses essential information about each visitor.

Main Attributes:

- visitor_id (PK): Unique identifier ID for the visitor.
- first_name: First name of the visitor.
- last_name: Last name of the visitor.
- phone: Phone contact of the visitor.
- email: Email address of the visitor.

RELATIONSHIP & CARDINALITY DESCRIPTIONS

Artist – Artwork:

Relationship: “creates/created by” between Artist and Artwork

Cardinality: 1 : M

Business Rules: Each Artist creates zero or more artworks. Every artwork is created by only one artist. However, the artist information may be disclosed or not.

Artwork – Display:

Relationship: “is at/has” between Artwork and Display

Cardinality: 1 : M

Business Rules: Each artwork can be at zero or many displays. Every display is associated to only one artwork.

Exhibition – Display:

Relationship: “has/is at” between Exhibition and Display

Cardinality: 1 : M

Business Rules: Each exhibition has at least one display. Every display is associated to only one exhibition.

Exhibition – Booking:

Relationship: “has/made for” between Exhibition and Booking

Cardinality: 1 : M

Business Rules: Each exhibition has zero or more bookings. Every booking is associated with only one exhibition.

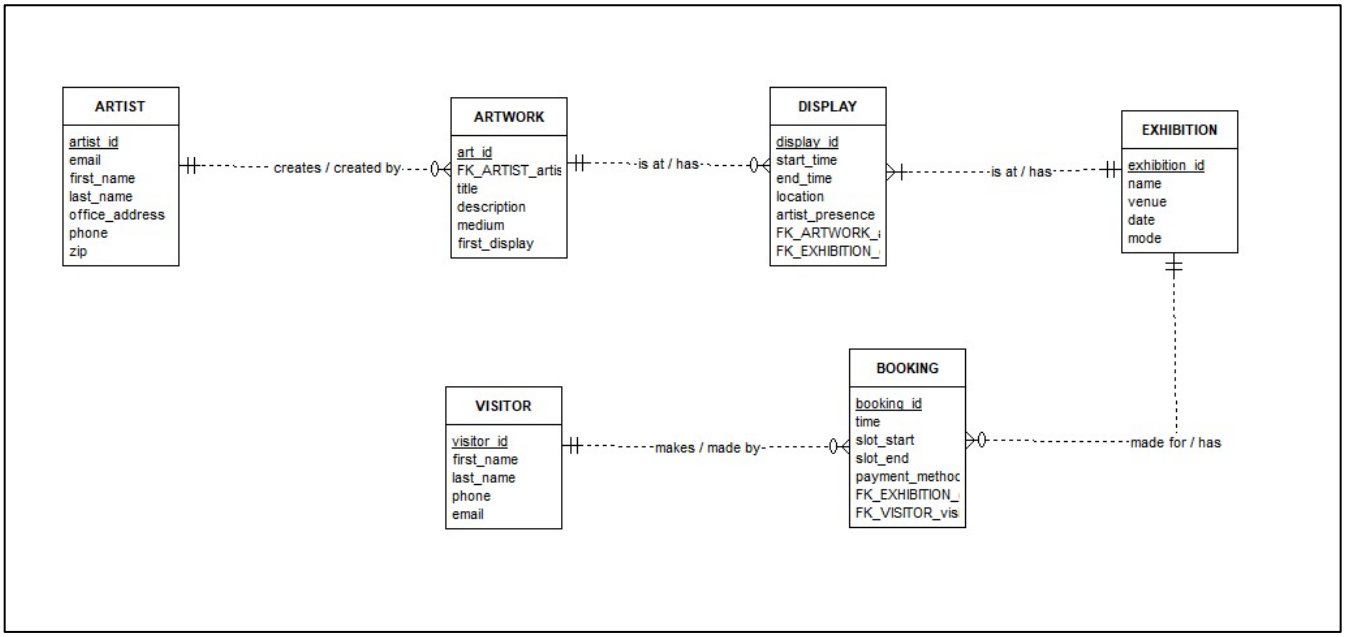
Visitor – Booking:

Relationship: “makes/made by” between Visitor and Booking

Cardinality: 1 : M

Business Rules: Each visitor can make zero or multiple bookings. Every booking is associated to only one visitor.

ENTITY RELATIONSHIP DIAGRAM



DDL SOURCE CODE

```
/*
 Project BAUN 6320 - UTD
 Group 5
 ART GALLERY MANAGEMENT
*/

/* DROP statements to clean up objects from previous run */

-- Triggers

DROP TRIGGER IF EXISTS TRG_VISITOR ON visitor;
DROP TRIGGER IF EXISTS TRG_EXHIBITION ON exhibition;
DROP TRIGGER IF EXISTS TRG_ARTIST ON artist;
DROP TRIGGER IF EXISTS TRG_ARTWORK ON artwork;
DROP TRIGGER IF EXISTS TRG_DISPLAY ON display;
DROP TRIGGER IF EXISTS TRG_BOOKING ON booking;
```

```
DROP TRIGGER IF EXISTS TRG_CHECK_DATE ON display;

-- Functions

DROP FUNCTION IF EXISTS TRG_FUN_VISITOR;
DROP FUNCTION IF EXISTS TRG_FUN_EXHIBITION;
DROP FUNCTION IF EXISTS TRG_FUN_ARTIST;
DROP FUNCTION IF EXISTS TRG_FUN_ARTWORK;
DROP FUNCTION IF EXISTS TRG_FUN_DISPLAY;
DROP FUNCTION IF EXISTS TRG_FUN_BOOKING;
DROP FUNCTION IF EXISTS TRG_FUN_CHECK_DATE;

-- Sequences

DROP SEQUENCE IF EXISTS SEQ_VISITOR_visitor_id;
DROP SEQUENCE IF EXISTS SEQ_EXHIBITION_exhibition_id;
DROP SEQUENCE IF EXISTS SEQ_ARTIST_artist_id;
DROP SEQUENCE IF EXISTS SEQ_ARTWORK_art_id;
DROP SEQUENCE IF EXISTS SEQ_DISPLAY_display_id;
DROP SEQUENCE IF EXISTS SEQ_BOOKING_booking_id;

-- Views

DROP VIEW IF EXISTS VisitorInfo;
DROP VIEW IF EXISTS ArtistPortfolio;

-- Indices

DROP INDEX IF EXISTS IDX_VISITOR_visitor_id;

DROP INDEX IF EXISTS IDX_EXHIBITION_exhibition_id;
DROP INDEX IF EXISTS IDX_EXHIBITION_name;

DROP INDEX IF EXISTS IDX_ARTIST_artist_id;
DROP INDEX IF EXISTS IDX_ARTIST_first_name;

DROP INDEX IF EXISTS IDX_ARTWORK_art_id;
DROP INDEX IF EXISTS IDX_ARTWORK_artist_id_FK;
DROP INDEX IF EXISTS IDX_ARTWORK_title;

DROP INDEX IF EXISTS IDX_DISPLAY_art_id_FK;
DROP INDEX IF EXISTS IDX_DISPLAY_exhibition_id_FK;
DROP INDEX IF EXISTS IDX_DISPLAY_location;

DROP INDEX IF EXISTS IDX_BOOKING_booking_id;
DROP INDEX IF EXISTS IDX_BOOKING_visitor_id_FK;
```

```
DROP INDEX IF EXISTS IDX_BOOKING_exhibition_id_FK;
DROP INDEX IF EXISTS IDX_BOOKING_slot_start;

-- Tables

DROP TABLE IF EXISTS booking;
DROP TABLE IF EXISTS display;
DROP TABLE IF EXISTS artwork;
DROP TABLE IF EXISTS artist;
DROP TABLE IF EXISTS exhibition;
DROP TABLE IF EXISTS visitor;

-----------------------------------------------------------------------------------------------------------------
/* Creating tables based on entities */

CREATE TABLE visitor(
   visitor_id          INTEGER             NOT NULL,
   first_name          VARCHAR(60)               NOT NULL,
        last_name              VARCHAR(60),
        phone                  VARCHAR(10),
   email                       VARCHAR(200)      NOT NULL,

   CONSTRAINT PK_visitor PRIMARY KEY (visitor_id),
        CONSTRAINT chk_phone CHECK (phone LIKE '__________' AND phone NOT LIKE
'%[^0-9]%'),
        CONSTRAINT chk_email CHECK (email LIKE '%_@_%._%')
);

CREATE TABLE exhibition(
   exhibition_id       INTEGER             NOT NULL,
   name                        VARCHAR(60)               NOT NULL,
        venue                  VARCHAR(60)               NOT NULL,
   date                DATE                NOT NULL,
        mode                   VARCHAR(10)               NOT NULL,

   CONSTRAINT PK_exhibition PRIMARY KEY (exhibition_id)
);

CREATE TABLE artist(
   artist_id           INTEGER             NOT NULL,
   email                       VARCHAR(200),
   first_name          VARCHAR(60)               NOT NULL,
        last_name              VARCHAR(60),
   office_address      VARCHAR(200),
        phone                  VARCHAR(10),
        zip                         VARCHAR(5),
```

```
   CONSTRAINT PK_artist PRIMARY KEY (artist_id),
   CONSTRAINT chk_email CHECK (email LIKE '%_@_%._%'),
        CONSTRAINT chk_phone CHECK (phone LIKE '__________' AND phone NOT LIKE
'%[^0-9]%'),
        CONSTRAINT chk_zip CHECK (zip LIKE '_____' AND zip NOT LIKE '%[^0-9]%')
);

CREATE TABLE artwork(
        art_id              INTEGER                 NOT NULL,
   artist_id          INTEGER,
        title               VARCHAR(60)             NOT NULL,
   description        VARCHAR(500),
        medium                  VARCHAR(100)    NOT NULL,
   first_displayDATE            NOT NULL,

   CONSTRAINT PK_artwork PRIMARY KEY (art_id),
        CONSTRAINT FK_ARTIST_artist_id FOREIGN KEY (artist_id) REFERENCES artist
);

CREATE TABLE display(
        display_id          INTEGER                 NOT NULL,
        art_id              INTEGER                 NOT NULL,
        exhibition_id  INTEGER                  NOT NULL,
        start_time          TIME                    NOT NULL,
        end_time            TIME                    NOT NULL,
        location            VARCHAR(60)             NOT NULL,
        artist_presenceBOOLEAN                  NOT NULL,

   CONSTRAINT PK_display PRIMARY KEY (display_id),
        CONSTRAINT DISPLAY_bridging UNIQUE (art_id, exhibition_id),
        CONSTRAINT CHECK_time CHECK (start_time < end_time),
        CONSTRAINT FK_ARTWORK_art_id FOREIGN KEY (art_id) REFERENCES
artwork,
        CONSTRAINT FK_EXHIBITION_EXHIBITION_id FOREIGN KEY (exhibition_id)
REFERENCES exhibition
);

CREATE TABLE booking(
        booking_id          INTEGER                 NOT NULL,
        visitor_id          INTEGER                 NOT NULL,
        exhibition_id  INTEGER                  NOT NULL,
        time                TIME                    NOT NULL,
        slot_start          TIME                    NOT NULL,
        slot_end            TIME                    NOT NULL,
        payment_method      VARCHAR(15)             NOT NULL,
```

```
   CONSTRAINT PK_booking PRIMARY KEY (booking_id),
        CONSTRAINT BOOKING_bridging UNIQUE (visitor_id, exhibition_id),
        CONSTRAINT CHECK_slot CHECK (slot_start < slot_end),
        CONSTRAINT FK_VISITOR_visitor_id FOREIGN KEY (visitor_id) REFERENCES
visitor,
        CONSTRAINT FK_EXHIBITION_EXHIBITION_id FOREIGN KEY (exhibition_id)
REFERENCES exhibition
);

-----------------------------------------------------------------------------------------------------
/* Create indices for natural keys, foreign keys, and frequently-queried columns */

-- VISITOR
-- Natural Keys
CREATE INDEX IDX_VISITOR_visitor_id                ON visitor (visitor_id);

-- EXHIBITION
-- Natural Keys
CREATE INDEX IDX_EXHIBITION_exhibition_id          ON exhibition (exhibition_id);
--  Frequently-queried columns
CREATE INDEX IDX_EXHIBITION_name                        ON exhibition(name);

-- ARTIST
-- Natural Keys
CREATE INDEX IDX_ARTIST_artist_id                  ON artist (artist_id);
--  Frequently-queried columns
CREATE INDEX IDX_ARTIST_first_name                 ON artist (first_name);

-- ARTWORK
--  Natural Keys
CREATE INDEX IDX_ARTWORK_art_id            ON artwork (art_id);
--  Foreign Keys
CREATE INDEX IDX_ARTWORK_artist_id_FK              ON artwork (artist_id);
--  Frequently-queried columns
CREATE INDEX IDX_ARTWORK_title                          ON artwork (title);

-- DISPLAY
--  Foreign Keys
CREATE INDEX IDX_DISPLAY_art_id_FK                 ON display (art_id);
CREATE INDEX IDX_DISPLAY_exhibition_id_FK          ON display (exhibition_id);
--  Frequently-queried columns
CREATE INDEX IDX_DISPLAY_location          ON display (location);

-- BOOKING
-- Natural Keys
```

```
CREATE INDEX IDX_BOOKING_booking_id              ON booking (booking_id);
--  Foreign Keys
CREATE INDEX IDX_BOOKING_visitor_id_FK           ON booking (visitor_id);
CREATE INDEX IDX_BOOKING_exhibition_id_FK        ON booking (exhibition_id);
--  Frequently-queried columns
CREATE INDEX IDX_BOOKING_slot_start              ON booking (slot_start);
```

```
/* Alter Tables by adding Audit Columns */

ALTER TABLE visitor
        ADD COLUMN created_by    VARCHAR(30),
   ADD COLUMN date_created  DATE,
   ADD COLUMN modified_by   VARCHAR(30),
   ADD COLUMN date_modified DATE;

ALTER TABLE exhibition
        ADD COLUMN created_by    VARCHAR(30),
   ADD COLUMN date_created  DATE,
   ADD COLUMN modified_by   VARCHAR(30),
   ADD COLUMN date_modified DATE;

ALTER TABLE artist
        ADD COLUMN created_by    VARCHAR(30),
   ADD COLUMN date_created  DATE,
   ADD COLUMN modified_by   VARCHAR(30),
   ADD COLUMN date_modified DATE;

ALTER TABLE artwork
        ADD COLUMN created_by    VARCHAR(30),
   ADD COLUMN date_created  DATE,
   ADD COLUMN modified_by   VARCHAR(30),
   ADD COLUMN date_modified DATE;

ALTER TABLE display
        ADD COLUMN created_by    VARCHAR(30),
   ADD COLUMN date_created  DATE,
   ADD COLUMN modified_by   VARCHAR(30),
   ADD COLUMN date_modified DATE;

ALTER TABLE booking
        ADD COLUMN created_by    VARCHAR(30),
   ADD COLUMN date_created  DATE,
   ADD COLUMN modified_by   VARCHAR(30),
   ADD COLUMN date_modified DATE;
```

```
---------------------------------------------------------------------------------------------------------------
/* Create Views */

-- Business purpose: The VisitorInfo view will be used primarily for rapidly fetching the email
information of individual visitors to enable easy and efficient communication enablement.
CREATE VIEW VisitorInfo AS
SELECT visitor_id, first_name, email
FROM visitor;

-- Business purpose: The ArtistPorfolio view will be used primarily for fetching all the artwork
title of a particular artist.
CREATE VIEW ArtistPortfolio AS
SELECT CASE
       WHEN artist.last_name IS NOT NULL THEN artist.first_name || ' ' || artist.last_name
       ELSE artist.first_name
   END as "ARTIST", title as "ARTWORK title"
FROM artist
left join artwork on artist.artist_id = artwork.artist_id
order by (artist.first_name || ' ' || artist.last_name), title;

---------------------------------------------------------------------------------------------------------------
/* Create Sequences */

CREATE SEQUENCE SEQ_VISITOR_visitor_id
   INCREMENT BY 1
   START WITH 1;

CREATE SEQUENCE SEQ_EXHIBITION_exhibition_id
   INCREMENT BY 1
   START WITH 1;

CREATE SEQUENCE SEQ_ARTIST_artist_id
   INCREMENT BY 1
   START WITH 1;

CREATE SEQUENCE SEQ_ARTWORK_art_id
   INCREMENT BY 1
   START WITH 1;

CREATE SEQUENCE SEQ_DISPLAY_display_id
   INCREMENT BY 1
   START WITH 1;

CREATE SEQUENCE SEQ_BOOKING_booking_id
   INCREMENT BY 1
   START WITH 1;
```

/* Create Triggers */

-- Business purpose: The TRG_VISITOR trigger automatically assigns a sequential visitor_id to a new visitor in the VISITOR table and assigns appropriate values to the created_by and date_created fields also, If the record is being inserted or updated, appropriate values are assigned to the modified_by and modified_date fields.

```
CREATE OR REPLACE FUNCTION TRG_FUN_VISITOR()
RETURNS TRIGGER AS $$
BEGIN
   IF TG_OP = 'INSERT' THEN
      IF NEW.visitor_id IS NULL THEN
         NEW.visitor_id := nextval('SEQ_VISITOR_visitor_id');
      END IF;
      IF NEW.created_by IS NULL THEN
         NEW.created_by := CURRENT_USER;
      END IF;
      IF NEW.date_created IS NULL THEN
         NEW.date_created := CURRENT_DATE;
      END IF;
   END IF;

   IF TG_OP = 'INSERT' OR TG_OP = 'UPDATE' THEN
      NEW.modified_by := current_user;
      NEW.date_modified := CURRENT_DATE;
   END IF;

   RETURN NEW;
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER TRG_VISITOR
BEFORE INSERT OR UPDATE ON visitor
FOR EACH ROW
EXECUTE FUNCTION TRG_FUN_VISITOR();
```

-- Business purpose: The TRG_EXHIBITION trigger automatically assigns a sequential exhibition_id to a new exhibition in the EXHIBITION table and assigns appropriate values to the created_by and date_created fields also, If the record is being inserted or updated, appropriate values are assigned to the modified_by and modified_date fields.

```
CREATE OR REPLACE FUNCTION TRG_FUN_EXHIBITION()
RETURNS TRIGGER AS $$
BEGIN
   IF TG_OP = 'INSERT' THEN
      IF NEW.exhibition_id IS NULL THEN
```

```
        NEW.exhibition_id := nextval('SEQ_EXHIBITION_exhibition_id');
      END IF;
      IF NEW.created_by IS NULL THEN
        NEW.created_by := CURRENT_USER;
      END IF;
      IF NEW.date_created IS NULL THEN
        NEW.date_created := CURRENT_DATE;
      END IF;
   END IF;

   IF TG_OP = 'INSERT' OR TG_OP = 'UPDATE' THEN
      NEW.modified_by := CURRENT_USER;
      NEW.date_modified := CURRENT_DATE;
   END IF;

   RETURN NEW;
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER TRG_EXHIBITION
BEFORE INSERT OR UPDATE ON exhibition
FOR EACH ROW
EXECUTE FUNCTION TRG_FUN_EXHIBITION();

-- Business purpose: The TRG_ARTIST trigger automatically assigns a sequential artist_id to a
new artist in the ARTIST table and assigns appropriate values to the created_by and date_created
fields also, If the record is being inserted or updated, appropriate values are assigned to the
modified_by and modified_date fields.
CREATE OR REPLACE FUNCTION TRG_FUN_ARTIST()
RETURNS TRIGGER AS $$
BEGIN
   IF TG_OP = 'INSERT' THEN
      IF NEW.artist_id IS NULL THEN
        NEW.artist_id := nextval('SEQ_ARTIST_artist_id');
      END IF;
      IF NEW.created_by IS NULL THEN
        NEW.created_by := CURRENT_USER;
      END IF;
      IF NEW.date_created IS NULL THEN
        NEW.date_created := CURRENT_DATE;
      END IF;
   END IF;

   IF TG_OP = 'INSERT' OR TG_OP = 'UPDATE' THEN
      NEW.modified_by := CURRENT_USER;
      NEW.date_modified := CURRENT_DATE;
```

```
    END IF;

    RETURN NEW;
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER TRG_ARTIST
BEFORE INSERT OR UPDATE ON artist
FOR EACH ROW
EXECUTE FUNCTION TRG_FUN_ARTIST();
```

-- Business purpose: The TRG_ARTWORK trigger automatically assigns a sequential art_id to a new artwork in the ARTWORK table and assigns appropriate values to the created_by and date_created fields also, If the record is being inserted or updated, appropriate values are assigned to the modified_by and modified_date fields. Further, if there is no reference to any artist, we have it refer to the UNKNOWN/ANONYMOUS ARTIST entry.

```
CREATE OR REPLACE FUNCTION TRG_FUN_ARTWORK()
RETURNS TRIGGER AS $$
BEGIN
    IF TG_OP = 'INSERT' THEN
        IF NEW.art_id IS NULL THEN
            NEW.art_id := nextval('SEQ_ARTWORK_art_id');
        END IF;
                    IF NEW.artist_id IS NULL THEN
            NEW.artist_id := (select artist_id from artist where first_name =
'UNKNOWN/ANONYMOUS');
        END IF;
        IF NEW.created_by IS NULL THEN
            NEW.created_by := CURRENT_USER;
        END IF;
        IF NEW.date_created IS NULL THEN
            NEW.date_created := CURRENT_DATE;
        END IF;
    END IF;

    IF TG_OP = 'INSERT' OR TG_OP = 'UPDATE' THEN
        NEW.modified_by := CURRENT_USER;
        NEW.date_modified := CURRENT_DATE;
    END IF;

    RETURN NEW;
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER TRG_ARTWORK
BEFORE INSERT OR UPDATE ON artwork
```

```
FOR EACH ROW
EXECUTE FUNCTION TRG_FUN_ARTWORK();

-- Business purpose: The TRG_DISPLAY trigger automatically assigns a sequential display_id to a new artist in the DISPLAY table and assigns appropriate values to the created_by and date_created fields also, If the record is being inserted or updated, appropriate values are assigned to the modified_by and modified_date fields.
CREATE OR REPLACE FUNCTION TRG_FUN_DISPLAY()
RETURNS TRIGGER AS $$
BEGIN
   IF TG_OP = 'INSERT' THEN
      IF NEW.display_id IS NULL THEN
         NEW.display_id := nextval('SEQ_DISPLAY_display_id');
      END IF;
      IF NEW.created_by IS NULL THEN
         NEW.created_by := CURRENT_USER;
      END IF;
      IF NEW.date_created IS NULL THEN
         NEW.date_created := CURRENT_DATE;
      END IF;
   END IF;

   IF TG_OP = 'INSERT' OR TG_OP = 'UPDATE' THEN
      NEW.modified_by := CURRENT_USER;
      NEW.date_modified := CURRENT_DATE;
   END IF;

   RETURN NEW;
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER TRG_DISPLAY
BEFORE INSERT OR UPDATE ON display
FOR EACH ROW
EXECUTE FUNCTION TRG_FUN_DISPLAY();

-- Business purpose: The TRG_BOOKING trigger automatically assigns a sequential booking_id to a new artist in the BOOKING table and assigns appropriate values to the created_by and date_created fields also, If the record is being inserted or updated, appropriate values are assigned to the modified_by and modified_date fields. Further, the time in the booking table is set to the current system time.
CREATE OR REPLACE FUNCTION TRG_FUN_BOOKING()
RETURNS TRIGGER AS $$
BEGIN
   IF TG_OP = 'INSERT' THEN
      IF NEW.booking_id IS NULL THEN
```

```
        NEW.booking_id := nextval('SEQ_BOOKING_booking_id');
      END IF;
                IF NEW.time IS NULL THEN
        NEW.time := CURRENT_TIME;
      END IF;
      IF NEW.created_by IS NULL THEN
        NEW.created_by := CURRENT_USER;
      END IF;
      IF NEW.date_created IS NULL THEN
        NEW.date_created := CURRENT_DATE;
      END IF;
   END IF;

   IF TG_OP = 'INSERT' OR TG_OP = 'UPDATE' THEN
      NEW.modified_by := CURRENT_USER;
      NEW.date_modified := CURRENT_DATE;
   END IF;

   RETURN NEW;
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER TRG_BOOKING
BEFORE INSERT OR UPDATE ON booking
FOR EACH ROW
EXECUTE FUNCTION TRG_FUN_BOOKING();

-- Business purpose: The TRG_CHECK_DATE trigger automatically checks the integrity of the
database by verifying if the display is on or after the first_display date of the particular
ARTWORK
CREATE OR REPLACE FUNCTION TRG_FUN_CHECK_DATE()
RETURNS TRIGGER AS $$
BEGIN
   IF (SELECT date FROM exhibition WHERE exhibition_id = NEW.exhibition_id) <
(SELECT first_display FROM artwork WHERE art_id = NEW.art_id) THEN
      RAISE EXCEPTION 'exhibition_date must be on or after first_display';
   END IF;
   RETURN NEW;
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER TRG_CHECK_DATE
BEFORE INSERT OR UPDATE ON display
FOR EACH ROW
EXECUTE FUNCTION TRG_FUN_CHECK_DATE();
```

DML SOURCE CODE

-- Populating all tables:

-- ARTIST Table:

```
INSERT INTO artist(email, first_name, last_name, office_address, phone, zip)
VALUES('f.piper@ucohio.edu', 'Florence', 'Piper', '670 East Renner Road, Richardson',
4697652413,75080);
INSERT INTO artist(email, first_name, last_name, office_address, phone, zip)
VALUES('a.rosario@gmail.com', 'Anna', 'Rosario', '201 N Shady Shores Rd #36 Lake
Dallas',9543427886, 75065);
INSERT INTO artist(email, first_name, last_name, office_address, phone, zip)
VALUES('wang_r@gmail.com', 'Riley', 'Wang', '2307 W Water Creek Dr, Frisco', 4653782883,
75034);
INSERT INTO artist(email, first_name, last_name, office_address, phone, zip)
VALUES('lily.fer9@gmail.com', 'Lily', 'Fernandez', '16577 Griswold Springs Rd, Plano',
9546712117, 74032);
INSERT INTO artist(email, first_name, last_name, office_address, phone, zip)
VALUES('m.wu987@neu.edu','Marcus','Wu', '905 West Georgebush Street, Little Elm',
9543671993, 75065);
INSERT INTO artist(email, first_name, last_name, office_address, phone, zip)
VALUES('reedbutler@hotmail.com','Reed','Butler','800 West Hillory Road, Plano', 9548612650,
74031);
INSERT INTO artist(email, first_name, last_name, office_address, phone, zip)
VALUES('wesfox32@hotmail.com', 'Wes', 'Fox', '2822 Independence Pkwy, Frisco',
4692815361, 75031);
INSERT INTO artist(email, first_name, last_name, office_address, phone, zip)
VALUES('leoperez1@hotmail.com', 'Leonardo','Perez', '301 Bushpark Crest, Denton',
4698412817, 75024);
INSERT INTO artist(email, first_name, last_name, office_address, phone, zip)
VALUES('andrewand@gmail.com', 'Andrew', 'Anderson', '9553 Lake Avenue, Grand Prairie',
4698156204, 75032);
INSERT INTO artist(email, first_name, last_name, office_address, phone, zip)
VALUES('calhounsana@gmail.com','Sana', 'Calhoun', '3 Cobblestone Avenue, Irving',
9543718254, 75060);
INSERT INTO artist(email, first_name, last_name, office_address, phone, zip)
VALUES('shopkins@hotmail.com', 'Sydney', 'Hopkins', '681 Pin Oak Ave. Dallas', 9543819258,
75065);
INSERT INTO artist(email, first_name, last_name, office_address, phone, zip)
VALUES('reiner@hotmail.com', 'Reiner', 'Braun', '913 Marley Ave. Dallas', 9543578925,
75063);
INSERT INTO artist(first_name)
VALUES('UNKNOWN/ANONYMOUS');
```

-- ARTWORK Table:

```
INSERT INTO artwork(artist_id, title, description, medium, first_display)
VALUES(1, 'Lonely Flame', 'A solitary, flickering flame against a backdrop of profound darkness, symbolizing isolation and resilience in the face of solitude, evoking a sense of introspection and contemplation within the viewer', 'Oil', '02/16/2023');
 INSERT INTO artwork(artist_id, title, description, medium, first_display)
VALUES(1, 'The Story of the Void', 'A cosmic narrative, inviting contemplation on the mysteries within the vast emptiness of space, creating an expansive yet intimate visual experience', 'Acrylic', '03/13/2021');
INSERT INTO artwork(artist_id, title, description, medium, first_display)
VALUES(1, 'Summer Mistress', 'A captation of the essence of the season in a vibrant dance of warm hues, depicting nature's beauty as a captivating and ephermal enchantress', 'Watercolor', '11/21/2015');
INSERT INTO artwork(artist_id, title, description, medium, first_display)
VALUES(2, 'Whispers of the Breeze', 'A serene landscape painting depicting a gentle meadow where delicate flowers sway in harmony with a soft breeze, capturing the ephemeral beauty of nature', 'Oil', '02/16/2023');
INSERT INTO artwork(artist_id, title, description, medium, first_display)
VALUES(2, 'Mechanical Symphony', 'A futuristic composition of metallic structures and intricate machinery, reflecting the harmonious dance of technology in an industrial landscape', 'Acrylic', '12/25/2021');
INSERT INTO artwork(artist_id, title, description, medium, first_display)
VALUES(2, 'Urban Symphony', 'An abstract cityscape pulsating with vibrant colors and dynamic lines, illustrating the rhythmic energy of urban life in a bustling metropolis', 'Oil', '11/16/2023');
INSERT INTO artwork(artist_id, title, description, medium, first_display)
VALUES(3, 'Silent Reflections', 'A tranquil seascape portraying a lone sailboat adrift on calm waters, inviting viewers to contemplate the serenity found in solitary moments', 'Watercolor', '03/15/2023');
INSERT INTO artwork(artist_id, title, description, medium, first_display)
VALUES(3, 'Ethereal Dreams', 'A surreal artwork featuring floating clouds and surreal landscapes, evoking a dreamlike atmosphere that blurs the boundaries between reality and fantasy', 'Gouache', '01/04/2020');
INSERT INTO artwork(artist_id, title, description, medium, first_display)
VALUES(4, 'Metamorphosis', 'An abstract piece showcasing the transformative power of vibrant colors and bold strokes, symbolizing personal growth and change', 'Acrylic', '05/27/2019');
INSERT INTO artwork(artist_id, title, description, medium, first_display)
VALUES(4, 'Whimsical Wonderland', 'A whimsical and playful painting depicting a fantastical world filled with imaginative creatures and vibrant, otherworldly landscapes', 'Gouache', '07/23/2021');
INSERT INTO artwork(artist_id, title, description, medium, first_display)
VALUES(5, 'Melancholy Melodies', 'A poignant portrait capturing the emotions of a musician lost in thought, surrounded by the haunting beauty of melancholic melodies', 'Charcoal', '04/21/2012');
INSERT INTO artwork(artist_id, title, description, medium, first_display)
```

```
VALUES(5, 'Celestial Reverie', 'A cosmic artwork that explores the mysteries of the universe, with swirling galaxies and ethereal lights that invite contemplation on the infinite', 'Spray Paint', '10/10/2009');
INSERT INTO artwork(artist_id, title, description, medium, first_display)
VALUES(5, 'Harmony in Chaos', 'An abstract expressionist piece featuring a symphony of chaotic brushstrokes and vibrant colors, symbolizing the beauty that can emerge from disorder', 'Oil', '02/12/2023');
INSERT INTO artwork(artist_id, title, description, medium, first_display)
VALUES(7, 'Serenitys Embrace', 'A tranquil depiction of a peaceful lake surrounded by lush mountains, capturing the calming embrace of nature', 'Pastels', '06/12/2023');
INSERT INTO artwork(artist_id, title, description, medium, first_display)
VALUES(8, 'Midnight Reverie', 'A nocturnal scene featuring a mysterious moonlit forest, where shadows and silver light create an enchanting atmosphere', 'Oil Pastels', '11/30/2023');
INSERT INTO artwork(artist_id, title, description, medium, first_display)
VALUES(8, 'Floating Gardens', 'Imaginary floating islands adorned with vibrant, fantastical flora, inviting viewers into a world of whimsy and color', 'Watercolor', '01/15/2023');
INSERT INTO artwork(artist_id, title, description, medium, first_display)
VALUES(11, 'Rhythmic Ripples', 'An interpretation of waters movement, with concentric circles expanding from a single droplet, symbolizing the interconnectedness of life', 'Alcohol Inks', '04/23/2014');
INSERT INTO artwork(artist_id, title, description, medium, first_display)
VALUES(11, 'Nebulous Odyssey', 'An abstract representation of a cosmic journey through swirling nebulae and celestial wonders', 'Alcohol Inks', '05/27/2019');
INSERT INTO artwork(artist_id, title, description, medium, first_display)
VALUES(12, 'Enigmatic Echoes', 'A portrait shrouded in mystery, with obscured features and deep shadows, leaving room for interpretation and introspection', 'Charcoal', '07/21/2021');
INSERT INTO artwork(artist_id, title, description, medium, first_display)
VALUES(12, 'Harvest Harmony', 'A vibrant depiction of a bountiful harvest, celebrating the abundance of nature and the cycles of growth', 'Acrylics', '02/27/2022');
INSERT INTO artwork(artist_id, title, description, medium, first_display)
VALUES(12, 'Infinite Echo', 'An exploration of symmetry and reflection, creating an illusion of endless corridors and repeating patterns', 'Oil', '10/29/2005');
INSERT INTO artwork(artist_id, title, description, medium, first_display)
VALUES(12, 'Sunset Sonata', 'A warm and vibrant portrayal of a sun setting over a serene landscape, capturing the poetic beauty of the evening sky', 'Oil', '03/14/2013');
INSERT INTO artwork(artist_id, title, description, medium, first_display)
VALUES(12, 'Metropolis Mirage', 'An abstract cityscape with distorted perspectives and vivid colors, reflecting the dynamic energy of an ever-evolving urban environment', 'Acrylic', '03/13/2019');
INSERT INTO artwork(artist_id, title, description, medium, first_display)
VALUES(12, 'Cerulean Dreams', 'A tranquil seascape featuring shades of blue and turquoise, conveying a dreamlike atmosphere of calm and introspection', 'Gouache', '08/05/2020');
INSERT INTO artwork(title, description, medium, first_display)
VALUES('Luminous Nexus', 'An exploration of light and shadow, where intersecting beams create a mesmerizing play of illumination and darkness', 'Oil', '10/14/2008');
INSERT INTO artwork(title, description, medium, first_display)
```

```
VALUES('Verdant Reverie', 'A lush and verdant forest scene, where vibrant greenery and dappled sunlight invite viewers into a world of natural splendor', 'Acrylic', '11/23/2020');
INSERT INTO artwork(title, description, medium, first_display)
VALUES('Transcendent Tides', 'A dynamic representation of ocean waves in various states, symbolizing the relentless ebb and flow of life's experiences', 'Watercolour', '09/18/2017');
INSERT INTO artwork(artist_id, title, description, medium, first_display)
VALUES(5, 'Crimson Whispers', 'An abstract expressionist piece dominated by deep red tones, evoking passion and intensity in a visual symphony of emotions', 'Acrylic', '09/26/2019');
INSERT INTO artwork(artist_id, title, description, medium, first_display)
VALUES(4, 'Galactic Embrace', 'A cosmic scene where celestial bodies intertwine, symbolizing the interconnectedness of the universe and cosmic relationships', 'Gouache', '07/27/2023');
INSERT INTO artwork(artist_id, title, description, medium, first_display)
VALUES(6, 'Parallel Realms', 'An imaginative depiction of overlapping realities, where dreamlike landscapes coexist with everyday scenes, blurring the boundaries of perception', 'Charcoal', '01/31/2011');
INSERT INTO artwork(title, description, medium, first_display)
VALUES('The Scream', 'An expressionistic construction based on a scream piercing through nature while on a walk, after two companions leave someone.', 'Oil', '06/16/2018');
INSERT INTO artwork(title, description, medium, first_display)
VALUES('Silent Night', 'A tranquil landscape in the wintery landscapes of The Alps', 'Acrylic', '05/18/2020');

--EXHIBITION Table

INSERT INTO exhibition(name, venue, date, mode)
VALUES('Artist Mart','1500, Grapevine Estate, TX, 71311', '2023-02-16', 'Offline');
INSERT INTO exhibition(name, venue, date, mode)
VALUES('Art Synergy', '1800, South Lake, TX, 70912', '2023-03-15', 'Offline');
INSERT INTO exhibition(name, venue, date, mode)
VALUES('Decora', '2000, Eastside Blvd, TX, 70111', '2023-12-01', 'Offline');
INSERT INTO exhibition(name, venue, date, mode)
VALUES('Craftric', '3000, Wellington, TX, 70112', '2023-12-04', 'Offline');
INSERT INTO exhibition(name, venue, date, mode)
VALUES('Galleria', '2001, Eastside Blvd, TX, 70111', '2023-12-08', 'Offline');
INSERT INTO exhibition(name, venue, date, mode)
VALUES('Wall Craft', '3001, Wellington, TX, 70112', '2023-12-11', 'Offline');
INSERT INTO exhibition(name, venue, date, mode)
VALUES('Villareal', '2310, Eastside Blvd, TX, 70121', '2023-12-11', 'Offline');
INSERT INTO exhibition(name, venue, date, mode)
VALUES('Cisernos', 'Zoom', '2023-12-23', 'Online');
INSERT INTO exhibition(name, venue, date, mode)
VALUES('Art Ally', '1000, Park Woods, TX, 71211', '2023-12-23', 'Offline');
INSERT INTO exhibition(name, venue, date, mode)
VALUES('Hues','Google Meet', '2024-01-09', 'Online');
INSERT INTO exhibition(name, venue, date, mode)
VALUES('Artistry', 'Zoom', '2024-01-25', 'Online');
```

```
INSERT INTO exhibition(name, venue, date, mode)
VALUES('Wall Art', '3001, Frisco, TX, 75032', '2024-02-05', 'Offline');

--DISPLAY Table

INSERT INTO display(start_time, end_time, location, artist_presence, art_id, exhibition_id)
VALUES('11:00:00', '18:00:00', 'A21', TRUE, 1, 1);
INSERT INTO display(start_time, end_time, location, artist_presence, art_id, exhibition_id)
VALUES('12:00:00', '17:00:00', 'B34', TRUE, 2, 1);
INSERT INTO display(start_time, end_time, location, artist_presence, art_id, exhibition_id)
VALUES('10:00:00', '19:00:00', 'C42', FALSE, 3, 1);
INSERT INTO display(start_time, end_time, location, artist_presence, art_id, exhibition_id)
VALUES('09:00:00', '17:00:00', 'A18', TRUE, 4, 1);
INSERT INTO display(start_time, end_time, location, artist_presence, art_id, exhibition_id)
VALUES('10:00:00', '19:00:00', 'D13', TRUE, 5, 1);
INSERT INTO display(start_time, end_time, location, artist_presence, art_id, exhibition_id)
VALUES('11:00:00', '19:00:00', 'A05', TRUE, 1, 2);
INSERT INTO display(start_time, end_time, location, artist_presence, art_id, exhibition_id)
VALUES('12:00:00', '19:00:00', 'C18', FALSE, 2, 2);
INSERT INTO display(start_time, end_time, location, artist_presence, art_id, exhibition_id)
VALUES('09:00:00', '14:00:00', 'B18', FALSE, 3, 2);
INSERT INTO display(start_time, end_time, location, artist_presence, art_id, exhibition_id)
VALUES('14:00:00', '20:00:00', 'A31', TRUE, 4, 2);
INSERT INTO display(start_time, end_time, location, artist_presence, art_id, exhibition_id)
VALUES('10:00:00', '14:00:00', 'C42', FALSE, 5, 2);
INSERT INTO display(start_time, end_time, location, artist_presence, art_id, exhibition_id)
VALUES('09:00:00', '15:00:00', 'B37', TRUE, 7, 2);
INSERT INTO display(start_time, end_time, location, artist_presence, art_id, exhibition_id)
VALUES('15:00:00', '21:00:00', 'C29', TRUE, 15, 3);
INSERT INTO display(start_time, end_time, location, artist_presence, art_id, exhibition_id)
VALUES('16:00:00', '21:00:00', 'B12', TRUE, 10, 3);
INSERT INTO display(start_time, end_time, location, artist_presence, art_id, exhibition_id)
VALUES('09:00:00', '14:00:00', 'D21', TRUE, 12, 3);
INSERT INTO display(start_time, end_time, location, artist_presence, art_id, exhibition_id)
VALUES('12:00:00', '22:00:00', 'C04', FALSE, 16, 4);
INSERT INTO display(start_time, end_time, location, artist_presence, art_id, exhibition_id)
VALUES('10:00:00', '19:00:00', 'D09', FALSE, 26, 5);
INSERT INTO display(start_time, end_time, location, artist_presence, art_id, exhibition_id)
VALUES('13:00:00', '22:00:00', 'D16', FALSE, 25, 6);
INSERT INTO display(start_time, end_time, location, artist_presence, art_id, exhibition_id)
VALUES('19:00:00', '22:00:00', 'B19', TRUE, 32, 7);
INSERT INTO display(start_time, end_time, location, artist_presence, art_id, exhibition_id)
VALUES('13:00:00', '20:00:00', 'C30', TRUE, 27, 8);
```

--VISITOR Table

```
INSERT INTO visitor(first_name, last_name, phone, email)
VALUES('Jenny', 'Evans', 9549934987, 'jennyeve09@gmail.com.com');
INSERT INTO visitor(first_name, last_name, phone, email)
VALUES('Monica', 'Anderson', 4698885208, 'monicand1@hotmail.com');
INSERT INTO visitor(first_name, last_name, phone, email)
VALUES('Kate', 'William', 9547793731, 'katewilliam02@outlook.com');
INSERT INTO visitor(first_name, last_name, phone, email)
VALUES('Chris', 'Renner', 9546696813, 'chrisrenner39@gmail.com');
INSERT INTO visitor(first_name, last_name, phone, email)
VALUES('Ryan', 'Ray', 4698989648, 'rr1999@hotmail.com');
INSERT INTO visitor(first_name, last_name, phone, email)
VALUES('Paul', 'Johnson', 4699794581, 'pjhonson12@outlook.com');
INSERT INTO visitor(first_name, last_name, phone, email)
VALUES('Rose', 'Collins', 9543373931, 'rose.collins@gmail.com');
INSERT INTO visitor(first_name, last_name, phone, email)
VALUES('Lauren', 'Jee', 4693456722, 'lauren.lee@hotmail.com');
INSERT INTO visitor(first_name, last_name, phone, email)
VALUES('Marshal', 'Jose', 3435677890, 'marshal.jose@gmail.com');
INSERT INTO visitor(first_name, last_name, phone, email)
VALUES('Amber', 'Jenner', 3439898981, 'amber.jenner@outlook.com');
INSERT INTO visitor(first_name, last_name, phone, email)
VALUES('Lily', 'Cooper', 3431987098, 'lily.cooper@hotmail.com');
INSERT INTO visitor(first_name, last_name, phone, email)
VALUES('Penny', 'Liam', 4691112346, 'penny.liam@gmail.com');
INSERT INTO visitor(first_name, last_name, phone, email)
VALUES('Liam', 'Downey', 3435182663, 'liamdow@uta.edu');
INSERT INTO visitor(first_name, last_name, phone, email)
VALUES('Scarlet', 'Robinson', 4696193551, 'scarobin@neu.edu');
INSERT INTO visitor(first_name, last_name, phone, email)
VALUES('Saranya', 'Kumar', 9541925193, 'saranyak91@usc.edu');
INSERT INTO visitor(first_name, last_name, phone, email)
VALUES('Satya', 'Khan', 4691483013, 'khansat@icloud.com');
INSERT INTO visitor(first_name, last_name, phone, email)
VALUES('Sharon', 'White', 3432017962, 'whsharon23@utd.edu');
INSERT INTO visitor(first_name, last_name, phone, email)
VALUES('Leslie', 'Kyte', 9548571852, 'leskyte342@outlook.com');
INSERT INTO visitor(first_name, last_name, phone, email)
VALUES('Mike', 'Jackson', 3432198548, 'mikejackson981@hotmail.com');
INSERT INTO visitor(first_name, last_name, phone, email)
VALUES('Chuck', 'Wilson', 9549991672, 'cxw21054@ucsd.edu');
INSERT INTO visitor(first_name, last_name, phone, email)
VALUES('Serena', 'Ohario', 3436153926, 'soha999@icloud.com');
INSERT INTO visitor(first_name, last_name, phone, email)
VALUES('Sally', 'Pritt', 9542719643, 'salpritt21@gmail.com');
```

```
INSERT INTO visitor(first_name, last_name, phone, email)
VALUES('Lucy', 'Song', 9546719453, 'lucysonggl@icloud.com');
INSERT INTO visitor(first_name, last_name, phone, email)
VALUES('Rishi', 'Patel', 3439127835, 'rishipatel@icloud.com');
INSERT INTO visitor(first_name, last_name, phone, email)
VALUES('Lily', 'Potter', 3432718945, 'lxp23106@utd.edu');
INSERT INTO visitor(first_name, last_name, phone, email)
VALUES('Lucy', 'Woo', 9543628156, 'woolucy@icloud.com');


--BOOKING Table

INSERT INTO booking(time, slot_start, slot_end, payment_method, exhibition_id, visitor_id)
VALUES('13:00:00', '9:30:00', '12:00:00', 'card', 1, 1);
INSERT INTO booking(time, slot_start, slot_end, payment_method, exhibition_id, visitor_id)
VALUES('14:20:00', '10:00:00', '13:00:00', 'cash', 1, 2);
INSERT INTO booking(time, slot_start, slot_end, payment_method, exhibition_id, visitor_id)
VALUES('15:00:00', '11:00:00', '13:30:00', 'cash', 1, 3);
INSERT INTO booking(time, slot_start, slot_end, payment_method, exhibition_id, visitor_id)
VALUES('15:45:00', '14:30:00', '17:30:00', 'card', 1, 4);
INSERT INTO booking(time, slot_start, slot_end, payment_method, exhibition_id, visitor_id)
VALUES('16:30:00', '10:00:00', '18:00:00', 'card', 1, 5);
INSERT INTO booking(time, slot_start, slot_end, payment_method, exhibition_id, visitor_id)
VALUES('17:00:00', '9:30:00', '17:00:00', 'card', 2, 2);
INSERT INTO booking(time, slot_start, slot_end, payment_method, exhibition_id, visitor_id)
VALUES('17:45:00','10:30:00', '18:00:00', 'cash', 2, 5);
INSERT INTO booking(time, slot_start, slot_end, payment_method, exhibition_id, visitor_id)
VALUES('18:00:00', '11:30:00', '13:00:00', 'card', 2, 10);
INSERT INTO booking(time, slot_start, slot_end, payment_method, exhibition_id, visitor_id)
VALUES('11:43:00', '10:20:00', '11:30:00', 'cash', 2, 19);
INSERT INTO booking(time, slot_start, slot_end, payment_method, exhibition_id, visitor_id)
VALUES('21:00:00', '11:15:00', '13:10:00', 'card', 2, 4);
INSERT INTO booking(time, slot_start, slot_end, payment_method, exhibition_id, visitor_id)
VALUES('22:00:00', '18:00:00', '20:00:00', 'card', 11, 21);
INSERT INTO booking(time, slot_start, slot_end, payment_method, exhibition_id, visitor_id)
VALUES('17:45:00', '16:00:00', '17:45:00', 'cash', 12, 15);
INSERT INTO booking(time, slot_start, slot_end, payment_method, exhibition_id, visitor_id)
VALUES('18:03:00', '11:00:00', '14:00:00', 'card', 4, 21);
INSERT INTO booking(time, slot_start, slot_end, payment_method, exhibition_id, visitor_id)
VALUES('19:00:00', '10:00:00', '12:00:00', 'card', 7, 7);
INSERT INTO booking(time, slot_start, slot_end, payment_method, exhibition_id, visitor_id)
VALUES('12:00:00', '10:00:00', '12:00:00', 'cash', 5, 26);
INSERT INTO booking(time, slot_start, slot_end, payment_method, exhibition_id, visitor_id)
VALUES('13:00:00', '15:30:00', '18:00:00', 'card', 3, 4);
INSERT INTO booking(time, slot_start, slot_end, payment_method, exhibition_id, visitor_id)
VALUES('20:00:00', '17:00:00', '20:00:00', 'card', 1, 7);
```

```
INSERT INTO booking(time, slot_start, slot_end, payment_method, exhibition_id, visitor_id)
VALUES('21:25:00', '19:30:00', '21:00:00', 'card', 2, 15);
INSERT INTO booking(time, slot_start, slot_end, payment_method, exhibition_id, visitor_id)
VALUES('13:30:00', '10:30:00', '13:20:00', 'card', 3, 9);
```

SQL QUERIES SOURCE CODE

```
----------------------------------------------BASIC QUERIES-----------------------------------------

--Query 1: Select all columns and all rows from one table - EXHIBITION table.

SELECT * FROM exhibition;

-----------------------------------------------------------------------------------------------------------

--Query 2: Select five columns(first_name, last_name, email, phone, zip) and all rows from one
table - ARTIST table.

SELECT
        first_name,
        last_name,
        email,
        phone,
        zip
FROM artist;

-----------------------------------------------------------------------------------------------------------

--Query 3: Select all columns from all rows from one view - ArtistPortfolio

SELECT * FROM ArtistPortfolio;

-----------------------------------------------------------------------------------------------------------

--Query 4: Using a join on 2 tables - ARTIST and ARTWORK tables, select all columns and all
rows from the tables without the use of a Cartesian product

SELECT * FROM artist
JOIN artwork
USING (artist_id);
```

--Query 5: Select and order data - based on date retrieved from one table - EXHIBITION table

```
SELECT * FROM exhibition
ORDER BY date ASC;
```

--Query 6: Using a join on 3 tables - EXHIBITION, BOOKING and VISITOR tables, select 5 columns - exhibition name, visitor first name, visitor last name, slot start and payment method from the 3 tables.
--Use syntax that would limit the output to 10 rows

```
SELECT name AS "Exhibition Name", first_name, last_name, slot_start, payment_method
FROM booking
JOIN exhibition USING (exhibition_id)
JOIN visitor USING (visitor_id)
LIMIT 10;
```

--Query 7: Select distinct rows - art_id, title, description, exhibition_id, name using joins on 3 tables - ARTWORK, DISPLAY and EXHIBTION tables

```
SELECT DISTINCT
        A.art_id, A.title, A.description, E.exhibition_id, E.name
FROM display
JOIN artwork A USING (art_id)
JOIN exhibition E USING (exhibition_id);
```

--Query 8: Use GROUP BY and HAVING in a select statement using one or more tables - ARTWORK table

```
SELECT artist_id, COUNT(artist_id)
FROM artwork
GROUP BY artist_id
HAVING COUNT(artist_id)>2;
```

--Query 9: Use IN clause to select data from one or more tables - ARTWORK table

```
SELECT title, description, medium
FROM artwork
WHERE medium IN ('Oil', 'Acrylic', 'Watercolor');
```

--Query 10: Select length of one column from one table - ARTWORK table (use LENGTH function)

```
SELECT art_id, title, LENGTH(title) AS "Length OF Title" FROM artwork;
```

--Query 11: Delete one record from one table. Use select statements to demonstrate the table contents before and after the DELETE statement.

```
SELECT * FROM booking;

BEGIN;

DELETE FROM booking
WHERE booking_id = 1;

SELECT * FROM booking;

ROLLBACK;

SELECT * FROM booking;
```

--Query 12: Update one record from one table. Use select statements to demonstrate the table contents before and after the UPDATE statement.

```
SELECT * FROM display;

BEGIN;

UPDATE display
SET location = 'N/A'
WHERE display_id = 1;

SELECT * FROM display;
```

```
ROLLBACK;

SELECT * FROM display;
```

----------ADVANCED QUERIES----------

--Query 13: Find out the visitor who can visit the maximum number of artworks across all exhibitions. Display visitor_id, visitor name, Number of artworks they could visit and the number of exhibitions that facilitate this.

```
SELECT
	v.first_name || ' ' || v.last_name AS "Visitor Name",
	COUNT(DISTINCT(b.exhibition_id)) AS "Number of Exhibitions",
	COUNT(art_id) AS "Number of Artworks"
FROM booking b
JOIN display d USING (exhibition_id)
JOIN visitor v USING (visitor_id)
WHERE
	slot_start < end_time  AND slot_end > start_time
GROUP BY
	v.visitor_id, v.first_name || ' ' || v.last_name
ORDER BY COUNT(art_id) DESC
LIMIT 1;
```

--Query 14: Display the Exhibition name, number of visitors for each exhibition and the number of artists that were present to display their work in the exhibitions. Order the exhibitions by total people who were present at some point in the exhibition

```
SELECT
	name AS "Exhibition Name",
	COUNT(DISTINCT(visitor_id)) AS "Visitor Count",
	COUNT(DISTINCT(artist_id, exhibition_id)) AS "Present Artist Count",
	COUNT(DISTINCT(visitor_id)) + COUNT(DISTINCT(artist_id, exhibition_id)) AS
"Total People Count"
	FROM booking
JOIN exhibition e USING (exhibition_id)
JOIN display USING (exhibition_id)
JOIN artwork USING (art_id)
WHERE artist_presence = TRUE
GROUP BY e.name
ORDER BY COUNT(DISTINCT(visitor_id)) + COUNT(DISTINCT(artist_id, exhibition_id))
DESC;
```