UNIVERSIDAD COMPLUTENSE DE MADRID

FACULTAD DE ESTUDIOS ESTADÍSTICOS

Creación de una base de datos en SQL

Carmen Vilanova de Diego

Máster en Big Data, Data Science e Inteligencia Artificial

10 de octubre de 2024

1 Diseño E-R

En esta sección se describirán las **entidades** seleccionadas a partir del enunciado, detallando sus **atributos**, e indicando el tipo de dato y caracterísitcas especiales, en caso de que las posean. También se definirán las **relaciones** entre las entidades, junto con sus respectivas cardinalidades, y se presentará el **Diagrama Entidad-Relación (E-R)** que representa gráficamente lo descrito.

Adicionalmente, se explicarán ciertos aspectos que no están reflejados directamente en el modelo entidad-relación, ya que esta modelización no permite captar todos los **matices** o supuestos implícitos del enunciado. Entre estos aspectos se incluirán las suposiciones asumidas y los **triggers** diseñados, los cuales automatizan la asignación de algunos valores y establecen restricciones dentro del sistema.

1.1 Entidades

Actividad
La entidad **actividad** representa los diferentes tipos de actividades que pueden estar asociadas a eventos.

- **idActividad**: Identificador único y autoincremental que sirve como clave primaria.
- **nombre**: Cadena de texto que identifica la actividad, no puede ser nulo.
- **tipo**: Cadena de texto que clasifica la actividad según su tipo (concierto, teatro, conferencia, etc.), no puede ser nulo.
- **coste**: Decimal que indica el coste asociado a la actividad, puede ser nulo. En este caso, trata de un atributo derivado que depende de los artistas que participen.

Artista
La entidad **artista** representa a los artistas que participan en actividades.

- **idArtista**: Identificador único y autoincremental que sirve como clave primaria.
- **nombreArt**: Cadena de texto que representa el nombre del artista, no puede ser nulo.
- **biografia**: Cadena de texto que contiene una breve descripción o biografía del artista, no puede ser nulo.

Ubicación
La entidad **ubicación** representa los lugares donde se llevan a cabo los eventos.

- **idUbicacion**: Identificador único y autoincremental que sirve como clave primaria.
- **nombreUbi**: Cadena de texto que identifica el nombre de la ubicación, no puede ser nulo.

- **direccion**: Cadena de texto que indica la dirección de la ubicación, no puede ser nulo.
- **tipo**: Cadena de texto que clasifica la ubicación como "ciudad" o "pueblo", no puede ser nulo. Tiene una restricción para asegurar que solo se acepten estos valores.
- **poblacion**: Cadena de texto que indica la ciudad o pueblo donde está ubicada la localización.
- **caracteristica**: Cadena de texto que describe una característica importante de la ubicación, no puede ser nulo.
- **aforo**: Carácter numérico que representa la capacidad máxima de personas en el lugar.
- **precioAlquiler**: Carácter numérico que indica el precio de alquiler del lugar.

Evento

La entidad **evento** representa eventos asociados a actividades y ubicaciones.

- **idEvento**: Identificador único y autoincremental que sirve como clave primaria.
- **idUbicacion**: Clave foránea que referencia a la entidad **ubicación**, indicando dónde se realiza el evento.
- **idActividad**: Clave foránea que referencia a la entidad **actividad**, indicando la actividad del evento.
- **nombreEvento**: Cadena de texto que representa el nombre del evento, no puede ser nulo.
- **precio_entrada**: Decimal que indica el precio de la entrada del evento. En este caso se ha decidido que el precio de las entradas sea único y no puede ser nulo.
- **fecha**: Atributo de tipo fecha que indica cuándo se realiza el evento, no puede ser nulo.
- **hora**: Atributo de tipo hora que indica el horario del evento, no puede ser nulo.

Asistente

La entidad **asistente** representa a las personas que asisten a los eventos.

- **idAsistente**: Identificador único y autoincremental que sirve como clave primaria.
- **nombreAs**: Cadena de texto que representa el nombre del asistente, no puede ser nulo.
- **apellidoAs**: Cadena de texto que representa el apellido del asistente, no puede ser nulo.
- **email**: Cadena de texto que representa el correo electrónico del asistente, no puede ser nulo.

1.2 Relaciones

Se han identificado las siguientes relaciones con sus correspondientes cardinalidades:

- **Evento - Ubicación**: Un evento se lleva a cabo en una ubicación específica. Puede haber múltiples eventos en una misma ubicación (N:1).
- **Actividad - Evento**: Cada evento está asociado a una actividad, y una actividad puede tener múltiples eventos asociados (1:N).
- **Asistente - Evento**: Un asistente puede asistir a múltiples eventos, y cada evento puede tener múltiples asistentes (N:N), lo que se gestiona a través de la tabla **asiste**.
- **Actividad - Artista**: Una actividad puede tener múltiples artistas asociados y cada artista puede participar en múltiples actividades (N:N), representado por la tabla **actividad_artista**.

1.3 Diagrama Entidad - Relación

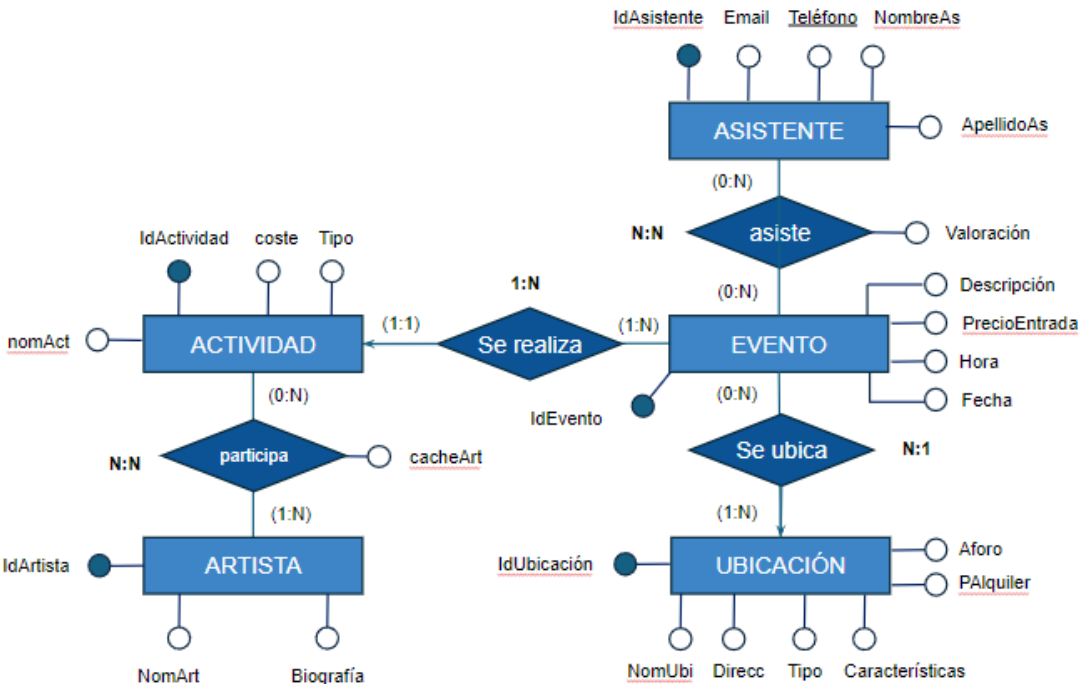

Fig. 1: Diagrama E-R

1.4 Triggers

Los **triggers** son mecanismos automáticos que permiten ejecutar acciones específicas en la base de datos en respuesta a ciertos eventos, como la inserción, actualización o eliminación de registros. Estos son útiles para mantener la integridad de los datos y automatizar procesos dentro del sistema. En este caso, se han diseñado dos *triggers*:

- **Control del aforo del evento**: Este *trigger* calcula el número de asistentes registrados para un evento y lo compara con el aforo máximo de la ubicación donde se celebra dicho evento. Cada vez que se intenta añadir un nuevo asistente, el *trigger* verifica si el número total de asistentes alcanza o supera el aforo permitido. Si se ha llegado al límite, el sistema muestra el mensaje: *"Aforo completo. No se pueden añadir más asistentes"*. Esto garantiza que no se sobrepase la capacidad de la ubicación.

- **Cálculo del coste de la actividad**: Este *trigger* se encarga de calcular el coste total de una actividad a partir de la suma de los cachés de los artistas que participan en ella. La información sobre el caché de los artistas se obtiene de la tabla `actividad_artistas`, donde se detalla el caché de cada artista según la actividad en la que colabora. Este *trigger* se activa al momento de agregar o modificar artistas en una actividad, asegurando que el coste de la actividad esté siempre actualizado.

2 Modelo relacional

Una vez identificadas las entidades y sus atributos, se llega al modelo relacional. En este modelo, cada entidad se convierte en una tabla con una clave primaria que debe ser única. Las entidades se relacionarán mediante *claves foráneas*. En el caso de que una relación tenga sus propios atributos, se creará una tabla relacional cuyas claves primarias serán aquellas claves primarias de las entidades que se quiere relacionar. A continuación, se describen las tablas identificadas:

- ACTIVIDAD(idActividad,nombre,tipoAct,coste)
- ARTISTA(idArtista,nombreArt,biografia)
- UBICACIÓN(idUbicacion,nombreUbi,direccion,tipoUbi,caracteristica,aforo,PAlquiler)
- EVENTO(idEvento,*idUbicacion*,*idActividad*,precioentrada,fecha,hora)
- ASISTENTE(idAsistente,nombreAs, apellidoAs,email)

En el caso de teléfono, al ser un atributo multivalorado, aparece con una tabla con idAsistente como su clave primaria y ajena.

- TELEFONO(*idAsistente*, telefono1, telefono2)

Finalmente, se presentan las tablas de las relaciones N:N **Actidad - Artista** y **Asistente - Evento**

- ACTIVIDADARTISTA(idArtista, idActividad,cacheArt)
- ASISTE(idAsistente,idEvento,valoracion)

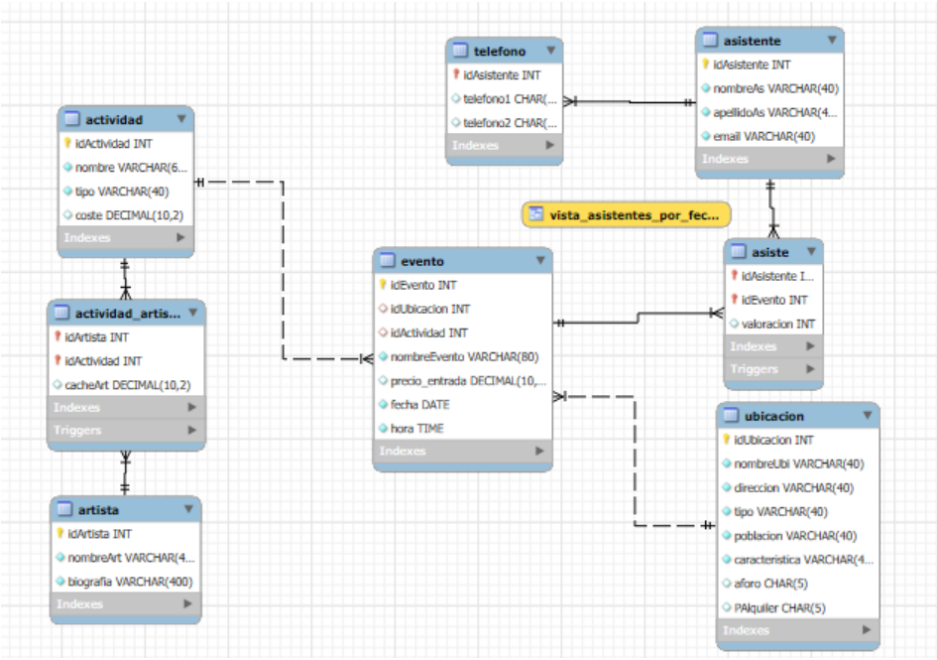

Fig. 2: Modelo relacional

3 Script

```
/*

  Proyecto final: Creacion de una base de datos
  Autora: Carmen Vilanova de Diego
*/
-- Eliminamos la base de datos en caso de existir
DROP DATABASE IF EXISTS empresaEventos;

-- Creamos la base de datos
CREATE DATABASE empresaEventos;
USE empresaEventos;


/*Creacion de tablas */
-- Eliminamos las tablas en caso de existir
DROP TABLE IF EXISTS actividad;
DROP TABLE IF EXISTS artista;
DROP TABLE IF EXISTS evento;
DROP TABLE IF EXISTS ubicacion;
DROP TABLE IF EXISTS asistente;
DROP TABLE IF EXISTS telefono;
DROP TABLE IF EXISTS asiste;
DROP TABLE IF EXISTS actividad_artista;

CREATE TABLE actividad (
   idActividad INT AUTO_iNCREMENT,
   nombre varchar(60) not null,
   tipo varchar(40) not null,
   coste DECIMAL(10, 2),
   PRIMARY KEY(idActividad)
);

CREATE TABLE artista (
   idArtista INT AUTO_iNCREMENT,
   nombreArt varchar(40) not null,
   biografia varchar(400) not null,
   PRIMARY KEY(idArtista)
);

CREATE TABLE ubicacion (
   idUbicacion INT AUTO_iNCREMENT,
   nombreUbi varchar(40) not null,
   direccion varchar(40) not null,
   tipo varchar(40) not null CHECK (tipo IN ('ciudad', 'pueblo')), /* solo
       puede ser ciudad o pueblo*/
   poblacion varchar(40) not null,
   caracteristica varchar(40) not null,
   aforo char(5),
   PAlquiler char(5),
   PRIMARY KEY(idUbicacion)
);
```

```
CREATE TABLE evento (
   idEvento INT AUTO_INCREMENT,
   idUbicacion INT,
   idActividad INT,
   nombreEvento varchar(80) not null,
   precio_entrada DECIMAL (10,2),
   fecha DATE not null,
   hora TIME not null,
   PRIMARY KEY(idEvento),
   FOREIGN KEY (idUbicacion) references ubicacion(idUbicacion) ON delete
      cascade,
   FOREIGN KEY (idActividad) references actividad(idActividad) ON delete
      cascade
);

CREATE TABLE asistente (
   idAsistente INT AUTO_iNCREMENT,
   nombreAs varchar(40) not null,
   apellidoAs varchar(40) not null,
   email varchar(40) not null,
   PRIMARY KEY(idAsistente)
);

CREATE TABLE telefono (
idAsistente INT,
telefono1 char(9),
telefono2 char(9),
PRIMARY key(idAsistente),
FOREIGN KEY(idAsistente) REFERENCES asistente(idAsistente) ON DELETE
   cascade
);

CREATE TABLE actividad_artista (
   idArtista INT,
   idActividad INT,
   cacheArt DECIMAL(10,2),
   PRIMARY KEY(idArtista, idActividad),
   FOREIGN KEY(idArtista) REFERENCES artista(idArtista) ON DELETE cascade,
   FOREIGN KEY(idActividad) REFERENCES actividad(idActividad) ON DELETE
      cascade
);


CREATE TABLE asiste (
   idAsistente INT not null,
   idEvento INT not null,
   valoracion INT CHECK (valoracion >= 0 AND valoracion <= 5),
   PRIMARY KEY(idAsistente, idEvento),
   FOREIGN KEY(idEvento) REFERENCES evento(idEvento) ON DELETE cascade,
   FOREIGN KEY(idAsistente) REFERENCES asistente(idAsistente) ON DELETe
      cascade
);
SET FOREIGN_KEY_CHECKS=0;
```

```
-- Trigger para calcular la suma de los caches de los artistas
   involucrados en la actividad
DELIMITER $$

CREATE TRIGGER calcular_coste_actividad
AFTER INSERT ON actividad_artista
FOR EACH ROW
BEGIN
    DECLARE nuevoCoste DECIMAL(10, 2);

    -- Calcular el nuevo coste de la actividad sumando los caches de los
       artistas
    SELECT SUM(cacheArt) INTO nuevoCoste
    FROM actividad_artista
    WHERE idActividad = NEW.idActividad;

    -- Actualizar el coste de la actividad en la tabla actividad
    UPDATE actividad
    SET coste = nuevoCoste
    WHERE idActividad = NEW.idActividad;
END$$

DELIMITER ;


-- Trigger para comprobar aforo
DELIMITER $$

CREATE TRIGGER check_aforo
BEFORE INSERT ON asiste
FOR EACH ROW
BEGIN
    DECLARE totalEntradas INT;
    DECLARE aforoMax INT;

    -- Obtener el aforo maximo del evento correspondiente
    SELECT aforo INTO aforoMax
    FROM evento e
    JOIN ubicacion u ON e.idUbicacion = u.idUbicacion
    WHERE e.idEvento = NEW.idEvento;

    -- Contar el numero de entradas ya vendidas para el evento
    SELECT COUNT(*) INTO totalEntradas
    FROM asiste
    WHERE idEvento = NEW.idEvento;

    -- Comprobar si el numero total de entradas excede el aforo
    IF (totalEntradas >= aforoMax) THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'Se ha completado el aforo';
    END IF;
END$$
```

```
DELIMITER ;


-- Se insertan los datos en las tablas
INSERT INTO actividad (nombre, tipo, coste)
VALUES
('Concierto de Jazz', 'Concierto', 0.00),
('Exposici n de Arte Contempor neo', 'Exposici n', 0.00),
('Conferencia sobre Inteligencia Artificial', 'Conferencia', 0.00),
('Obra de Teatro Cl sico', 'Teatro', 0.00),
('Concierto de M sica Cl sica', 'Concierto', 0.00),
('Conferencia sobre Econom a Circular', 'Conferencia', 0.00),
('Concierto de Rock and Roll', 'Concierto', 0.00),
('Exposici n Fotogr fica', 'Exposici n', 0.00),
('Obra de Teatro Musical', 'Teatro', 0.00),
('Concierto de Blues', 'Concierto', 0.00);

INSERT INTO artista (nombreArt, biografia)
VALUES
('Carlos Santana', 'Guitarrista y compositor mexicano, reconocido por
   fusionar el rock con la m sica latina. Ha ganado m ltiples premios
   Grammy.'),
('Pablo Picasso', 'Pintor y escultor espa ol, cofundador del movimiento
   cubista y uno de los artistas m s influyentes del siglo XX.'),
('Beyonc ', 'Cantante y actriz estadounidense, conocida por su poderosa
   voz y sus contribuciones a la m sica pop y R&B a nivel global.'),
('Frida Kahlo', 'Pintora mexicana famosa por sus autorretratos que
   exploran temas de identidad, poscolonialismo y feminismo.'),
('Miles Davis', 'Trompetista y compositor estadounidense, considerado uno
   de los m sicos de jazz m s innovadores de todos los tiempos.'),
('Adele', 'Cantante y compositora brit nica, reconocida por su profunda
   voz y sus emotivas baladas. Ganadora de varios premios Grammy.'),
('Vincent van Gogh', 'Pintor posimpresionista neerland s, conocido por su
    estilo vibrante y emocional, que dej  un legado inmenso a pesar de su
    vida tr gica.'),
('John Coltrane', 'Saxofonista y compositor estadounidense, pionero en el
   desarrollo del jazz modal y conocido por su estilo experimental.'),
('Lady Gaga', 'Cantante y actriz estadounidense, famosa por su
   versatilidad en g neros musicales y su estilo art stico  nico  y
   exc ntrico.'),
('Diego Rivera', 'Muralista mexicano, conocido por sus obras que reflejan
   la historia y lucha social de M xico en el siglo XX.');


INSERT INTO actividad_artista (idArtista, idActividad, cacheArt)
VALUES
(1, 1, 500.00),  -- Carlos Santana en Concierto de Jazz
(2, 2, 300.00),  -- Pablo Picasso en Exposici n de Arte Contempor neo
(3, 3, 150.00), -- Beyonc  en Conferencia sobre Inteligencia Artificial
(4, 4, 200.00),  -- Frida Kahlo en Obra de Teatro Cl sico
(5, 1, 800.00),  -- Miles Davis en Concierto de Jazz
(6, 5, 1000.00), -- Adele en Concierto de M sica Cl sica
(7, 2, 0.00),     -- Vincent van Gogh en Exposici n de Arte
   Contempor neo (sin cach )
```

```
(8, 6, 600.00),  -- John Coltrane en Conferencia sobre Econom a Circular
(9, 7, 120.00), -- Lady Gaga en Concierto de Rock and Roll
(10, 9, 300.00),  -- Diego Rivera en Obra de Teatro Moderno
(9, 1, 120.00), -- Lady Gaga en Concierto de Jazz
(10, 4, 300.00); -- Diego Rivera en Obra de Teatro Clasico

INSERT INTO ubicacion (nombreUbi, direccion, tipo, caracteristica, aforo,
    PAlquiler, poblacion)
VALUES
('Auditorio Nacional', 'Calle Pr ncipe, 44', 'ciudad', 'Ac stica
    excelente', '10', '5000', 'Madrid'),
('Teatro Real', 'Plaza de Oriente, s/n', 'ciudad', 'Teatro hist rico', '
    15', '4000', 'Barcelona'),
('Sala Luna', 'Calle Luna, 14', 'ciudad', 'Sala moderna', '50', '1500', '
    Valencia'),
('Parque Central', 'Av. Libertad, 101', 'pueblo', 'Espacio al aire libre',
     '30', '2500', 'San Pedrito'),
('Plaza Mayor', 'Plaza Mayor, s/n', 'pueblo', 'Centro cultural', '12', '
    1000', 'El Rinc n'),
('Museo del Arte', 'Calle del Arte, 22', 'ciudad', 'Galer a moderna', '10
    ', '3000', 'Bilbao'),
('Centro de Convenciones', 'Calle Mayor, 55', 'ciudad', 'Sal n para
    eventos', '25', '4500', 'Sevilla'),
('Teatro del Pueblo', 'Calle Libertad, 9', 'pueblo', 'Teatro tradicional',
     '80', '1200', 'Villa Esperanza'),
('Anfiteatro al Aire Libre', 'Parque del Sol, s/n', 'ciudad', 'Vista
    panor mica', '180', '3500', 'Granada'),
('Casa de la Cultura', 'Av. Central, 12', 'pueblo', 'Lugar comunitario', '
    40', '800', 'Pueblo Nuevo');


INSERT INTO asistente (nombreAs, apellidoAs, email)
VALUES
('Carlos', 'Garc a', 'carlos.garcia@email.com'),
('Mar a', 'L pez', 'maria.lopez@email.com'),
('Luis', 'Mart nez', 'luis.martinez@email.com'),
('Ana', 'S nchez', 'ana.sanchez@email.com'),
('Pedro', 'Fern ndez', 'pedro.fernandez@email.com'),
('Laura', 'G mez', 'laura.gomez@email.com'),
('Javier', 'D az', 'javier.diaz@email.com'),
('Carmen', 'Ruiz', 'carmen.ruiz@email.com'),
('Jos ', 'P rez', 'jose.perez@email.com'),
('Luc a', 'Romero', 'lucia.romero@email.com');

INSERT INTO telefono (idAsistente, telefono1, telefono2)
VALUES
(1, '600123456', '690987654'),  -- Carlos Garc a
(2, '610234567', '680876543'),  -- Mar a L pez
(3, '620345678', NULL),         -- Luis Mart nez (solo un numero)
(4, '630456789', '670765432'),  -- Ana S nchez
(5, '640567890', NULL),         -- Pedro Fern ndez (solo un numero)
(6, '650678901', '660654321'),  -- Laura G mez
(7, '670789012', '610543210'),  -- Javier D az
(8, '680890123', NULL),         -- Carmen Ruiz (solo un numero)
```

```
(9, '690901234', '600432109'),  -- Jos  P rez
(10, '600012345', NULL);        -- Luc a Romero (solo un numero)

INSERT INTO evento (idUbicacion, idActividad, nombreEvento, precio_entrada
    , fecha, hora)
VALUES
(3, 1, 'Noche de Jazz', 22.00, '2024-10-16', '20:00:00'),
(5, 8, 'Fotograf a Urbana: Miradas Contempor neas', 18.00, '2024-10-16',
    '17:00:00'),
(2, 4, 'Teatro Cl sico: La Vida es Sue o', 32.00, '2024-10-16', '
    19:30:00'),
(7, 3, 'Conferencia: El Futuro de la IA', 45.00, '2024-10-17', '09:00:00')
    ,
(1, 6, 'Econom a Circular y Sostenibilidad', 35.00, '2024-10-17', '
    10:30:00'),
(9, 5, 'Concierto de M sica Cl sica', 27.00, '2024-10-17', '20:00:00'),
(6, 7, 'Rock and Roll Fest', 50.00, '2024-10-17', '21:30:00'),
(4, 10, 'Blues al Atardecer', 28.00, '2024-10-18', '18:30:00'),
(8, 9, 'Teatro Musical: Un Sue o en Broadway', 38.00, '2024-10-18', '
    17:30:00'),
(10, 2, 'Exposici n de Arte Contempor neo', 20.00, '2024-10-18', '
    12:00:00'),
(3, 1, 'Jazz al Mediod a', 20.00, '2024-10-18', '14:00:00'),
(4, 8, 'Fotografias del Nuevo Mundo', 33.00, '2024-10-18', '20:00:00');


INSERT INTO asiste (idAsistente, idEvento, valoracion)
VALUES
(1, 1, 5),
(1, 5, 4),
(1, 7, 5),
(1, 10, 4),
(2, 2, 3),
(2, 8, 4),
(2, 12, 2),
(2, 19, 3),
(2, 20, 1),
(3, 3, 2),
(3, 6, 3),
(3, 13, 1),
(4, 4, 1),
(4, 9, 2),
(4, 14, 0),
(5, 5, 3),
(5, 9, 4),
(5, 15, 5),
(6, 7, 5),
(6, 10, 4),
(6, 16, 5),
(7, 2, 4),
(7, 8, 3),
(7, 12, 3),
(7, 15, 2),
(8, 9, 4),
```

```
(8, 14, 5),
(8, 20, 3),
(9, 3, 2),
(9, 6, 1),
(10, 4, 3),
(10, 10, 2),
(10, 16, 4),
(3, 2, 0),
(9, 2, 0),
(10, 8, 0);

-- CONSULTAS --

# 1.Numero de eventos de cada actividad

SELECT nombre, count(e.idEvento) numeroEventos
FROM evento e
INNER JOIN
actividad a ON e.idActividad = a.idActividad
GROUP BY
a.nombre;

# 2. Actividades con un solo artista
SELECT a.nombre, COUNT(aa.idArtista) AS numeroArtistas
FROM actividad_artista aa
INNER JOIN actividad a ON aa.idActividad = a.idActividad
GROUP BY a.idActividad
HAVING COUNT(aa.idArtista) = 1;


# 3. Poblacion en la que solo se han realizado eventos de teatro
SELECT u.poblacion
FROM ubicacion u
INNER JOIN evento e ON u.idUbicacion = e.idUbicacion
INNER JOIN actividad a ON e.idActividad = a.idActividad
GROUP BY u.poblacion
HAVING COUNT(DISTINCT a.tipo) = 1 AND MAX(a.tipo) = 'Teatro';


# 4. Evento con mas ceros en su valoraci n
SELECT nombreEvento, COUNT(valoracion) AS num_ceros
FROM evento e
INNER JOIN asiste a ON a.idEvento = e.idEvento
WHERE valoracion = 0
GROUP BY nombreEvento
ORDER BY num_ceros DESC
LIMIT 1;


# 5. Asistente que puntua m s alto

SELECT nombreAs, apellidoAs, AVG(valoracion) AS mediaValoracion
FROM asiste a
INNER JOIN
```

```
asistente aa ON a.idAsistente = aa.idAsistente
GROUP BY a.idAsistente
ORDER BY mediaValoracion DESC
LIMIT 1;


# 6. Dia en el que m s asistentes ha habido

-- En primer lugar creamos la vista
CREATE VIEW vista_asistentes_por_fecha AS
SELECT e.fecha, COUNT(DISTINCT a.idAsistente) AS totalAsistentes
FROM evento e
INNER JOIN asiste a ON e.idEvento = a.idEvento
GROUP BY e.fecha;


-- Realizamos la consulta utilizando la vista
SELECT fecha, totalAsistentes
FROM vista_asistentes_por_fecha
ORDER BY totalAsistentes DESC
LIMIT 1;


# 7. Obten el beneficio de cada evento
SELECT e.nombreEvento,
       (SUM(e.precio_entrada) - a.coste) AS beneficios -- para calcular
          los beneficios sumamos el precio de las entradas vendidas y
          restamos el coste de la actividad
FROM evento e
INNER JOIN actividad a ON a.idActividad = e.idActividad
INNER JOIN asiste s ON s.idEvento = e.idEvento
GROUP BY e.nombreEvento, a.coste
ORDER BY beneficios DESC;

# 8. Artistas que han participado en mas de una actividad
-- Para esta consulta utilizamos IN para generar una subconsulta

SELECT ar.nombreArt
FROM artista ar
WHERE ar.idArtista IN (
    SELECT aa.idArtista
    FROM actividad_artista aa
    GROUP BY aa.idArtista
    HAVING COUNT(aa.idActividad) > 1
);


# 9. Consulta el numero total de eventos a los que cada asistente ha
   asistido
SELECT a.nombreAs, apellidoAs, COUNT(s.idEvento) AS totalEventosAsistidos
FROM  asistente a
INNER JOIN
asiste s ON a.idAsistente = s.idAsistente
GROUP BY a.nombreAs, apellidoAs
```

```sql
ORDER BY totalEventosAsistidos DESC;



# 10. Consulta el nombre y apellidos de los asistentes cuyo empiecen por
   61
SELECT a.nombreAs, a.apellidoAs, telefono1, telefono2
FROM asistente a
INNER JOIN telefono t ON a.idAsistente = t.idAsistente
WHERE t.telefono1 LIKE '61%' OR t.telefono2 LIKE '61%';
```