Módulo 2: Bases de Datos SQL

Proyecto Final: “Creación de una Base de Datos”

Jacobo Álvarez Gutiérrez
MÁSTER BIG DATA E INTELIGENCIA ARTIFICIAL
Universidad Complutense de Madrid

Contenido

I. Diseño del modelo Entidad – Relación

Atendiendo al enunciado de este proyecto, se ha decidido definir un modelo Entidad – Relación para la creación de la base de datos pedida tal y como se muestra en la Figura 1.

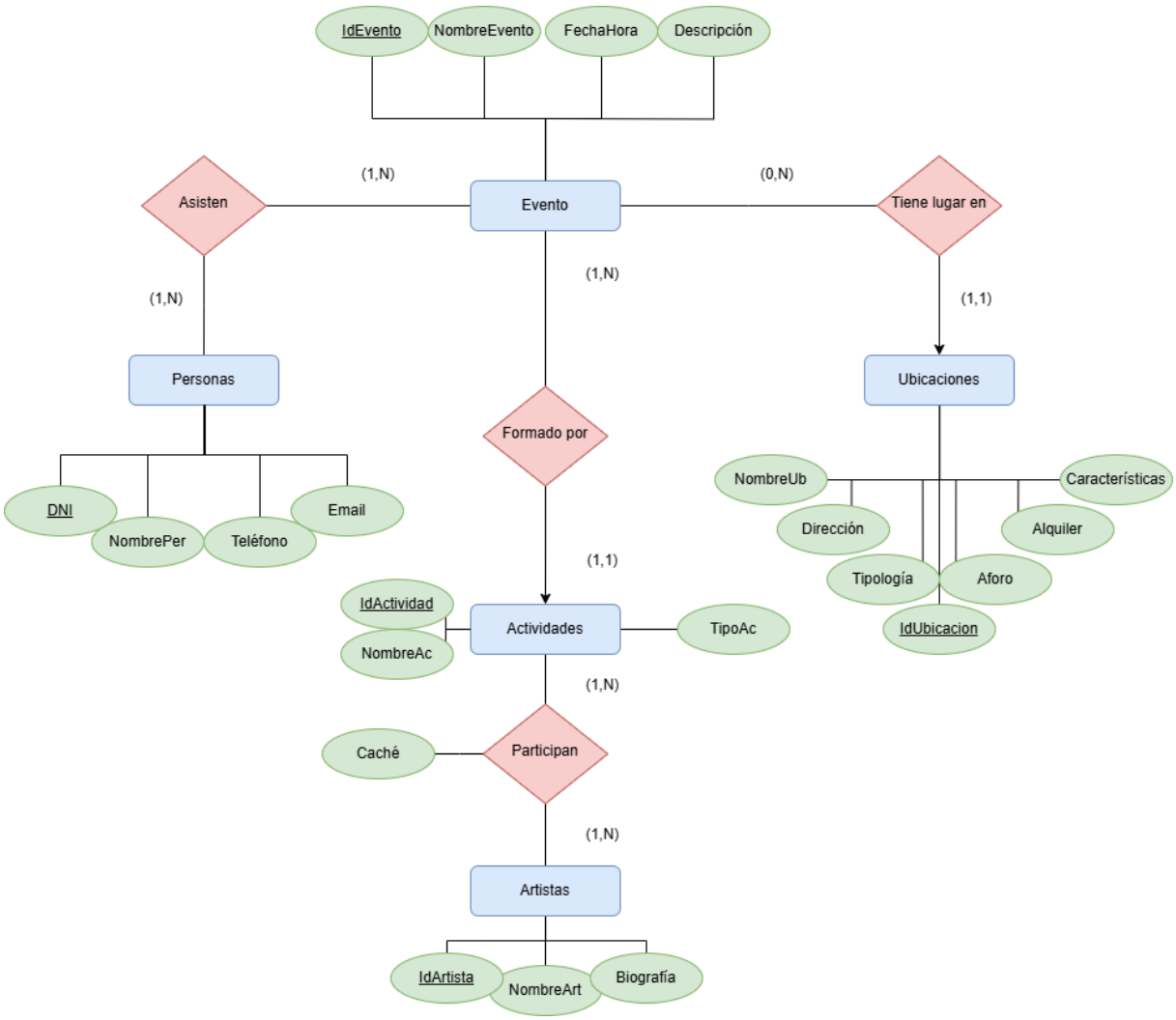

FIGURA 1. Modelo Entidad - Relación para el proyecto "ArteVida Cultural"

Tal y como se puede apreciar, se ha optado por definir 5 entidades (evento, personas, ubicaciones, actividades y artistas) y 3 relaciones (“asisten”, “formado por”, “participan” y “tiene lugar en”). A continuación, se procederá a la justificación de su elección y al desglose en detalle de cada una de ellas.

i. Entidades

Las entidades que se definen a continuación están diseñadas para cumplir con el objetivo principal de gestionar eventos culturales de manera eficiente y permitir el análisis de información clave, como el número de asistentes, los costos, y el tipo de actividades más rentables. Todas las entidades están estrechamente relacionadas, lo que garantiza un modelo relacional sólido y funcional.

❖ Evento

Esta entidad es el eje central del sistema, ya que representa los eventos culturales organizados por la empresa. Su propósito es almacenar toda la información relevante sobre cada evento, permitiendo la gestión eficiente de las actividades realizadas. Los atributos seleccionados para esta entidad son:

- ***IdEvento:*** Sirve como identificador único de cada evento, asegurando que no haya duplicados y facilitando las relaciones con otras entidades como Asisten y Ubicaciones.
- ***NombreEvento:*** Proporciona un nombre descriptivo que permite identificar el evento de manera sencilla y clara.
- ***FechaHora:*** Permite registrar la fecha y hora específicas en que se realiza el evento, crucial para programar y gestionar la agenda.
- ***Descripción:*** Ofrece un resumen informativo del contenido del evento, ayudando a los usuarios a comprender de qué trata.

La inclusión de estos atributos responde a la necesidad de gestionar información básica sobre los eventos y facilitar consultas como: ¿qué eventos se realizan en una fecha específica? o ¿cuál es la agenda de eventos programados?

En relación con el enunciado del proyecto, podría parecer que nos hemos olvidado de un atributo para recoger el precio de las entradas de cada evento. No obstante, he decidido implementar esto como una vista en el script de SQL. La idea ha sido suponer que la empresa quiere obtener un 20% de beneficio sobre los costos totales (caché de los artistas invitados y alquiler del local), de modo que repartirá el precio de las entradas suponiendo que logran llenar el aforo. De esta forma, la empresa obtiene el 100% del beneficio esperado si y solo si logra llenar el aforo del local.

❖ Personas

Esta entidad se diseñó para representar a los asistentes a los eventos, un componente esencial en la medición del éxito y la popularidad de las actividades organizadas. Los atributos son:

- **DNI:** Funciona como identificador único para cada persona, lo que permite registrar sus asistencias sin ambigüedades.
- ***NombrePer:*** Facilita la identificación de los asistentes.
- ***Teléfono*** y ***Email:*** Son datos de contacto necesarios para interactuar con los asistentes en caso de requerir confirmaciones, envíos de notificaciones, etc.

El diseño de esta entidad permite analizar tendencias de asistencia, como qué personas asisten con mayor frecuencia o cuántos eventos ha visitado un asistente específico.

❖ Ubicaciones

Esta entidad describe los lugares donde se realizan los eventos. Es clave para gestionar la logística y calcular métricas como la ocupación y los costos asociados a cada evento. Sus atributos son:

- ***IdUbicación:*** Identificador único que permite relacionar las ubicaciones con los eventos de manera inequívoca.
- ***NombreUb:*** Nombre descriptivo del lugar, útil para identificar rápidamente la ubicación.
- ***Dirección:*** Proporciona la localización exacta, fundamental para la organización logística.
- ***Tipología:*** Diferencia entre ubicaciones urbanas (Ciudad) y rurales (Pueblo), permitiendo analizar como influye el tipo de ubicación en, por ejemplo, el éxito del evento.
- ***Aforo:*** Representa la capacidad máxima de la ubicación, dato indispensable para controlar la asistencia y evitar sobrepasar los límites legales o de seguridad. Asimismo, será de gran relevancia en el cálculo del precio de las entradas.
- ***Alquiler:*** Indica el costo de utilizar la ubicación, elemento crucial también para el análisis de los beneficios y para el precio de las entradas.
- ***Características:*** Describe detalles adicionales del lugar, como si tiene buena acústica o iluminación, lo cual puede ser determinante al asignar eventos a ubicaciones.

Este conjunto de atributos permite analizar cómo afectan las características de una ubicación al éxito del evento y calcular costos logísticos.

❖ Actividades

Esta entidad representa las diferentes categorías culturales que componen los eventos organizados, como música, teatro, conferencias, etc. Sus atributos son:

- ***IdActividad:*** Identificador único para cada actividad, necesario para establecer relaciones con los eventos y los artistas.
- ***NombreAc:*** Describe el nombre específico de la actividad, como “Concierto de Jazz” o “Exposición de Pintura”.
- ***TipoAc:*** Clasifica las actividades en categorías generales, como “Música” o “Arte Visual”, lo que permite realizar análisis más amplios, como identificar el tipo de actividad que genera más ingresos o asistentes.

El diseño de esta entidad facilita consultas específicas como: ¿cuántos eventos pertenecen a cada tipo de actividad? o ¿qué actividades son más populares?

❖ Artistas

Los artistas son los protagonistas de las actividades, y esta entidad almacena información relevante sobre ellos. Incluye:

- ***IdArtistas:*** Identificador único para cada artista, que asegura la integridad en las relaciones con otras entidades.
- ***NombreArt:*** Permite identificar al artista
- ***Biografía:*** Contiene una breve descripción de la trayectoria del artista, útil para enriquecer la información de los eventos en los que participa.

Esta entidad se diseñó teniendo en cuenta la importancia de los artistas en la percepción del evento y la necesidad de calcular costos asociados (a través del caché, almacenado en la relación "Participan").

ii. Relaciones

Las relaciones definidas en este modelo no solo conectan las entidades de forma lógica, sino que también permiten representar la dinámica real de la organización de eventos culturales. En efecto, a través de la relación "*asisten*", es posible medir el éxito del evento desde la perspectiva de la audiencia. Con la relación "*tiene lugar en*", se asegura una organización eficiente de los espacios y se puede optimizar el uso de las ubicaciones. La relación "*formado por*" permite categorizar los eventos y realizar un análisis profundo de las actividades organizadas. Finalmente, con la relación "*participan*", se tiene un control detallado de los artistas involucrados y los costos asociados a sus participaciones.

Así, el diseño de estas relaciones asegura que el modelo no solo sea funcional, sino también flexible para realizar consultas complejas y proporcionar información útil para la toma de decisiones estratégicas.

❖ Asisten (Personas - Evento)

La relación "*Asisten*" conecta las entidades "*Personas*" y "*Evento*". Esto es esencial para registrar y analizar la participación del público en las actividades culturales. La cardinalidad de esta relación es la siguiente:

- **(1, N) para Personas:** Una persona puede asistir a múltiples eventos.
- **(1, N) para Eventos:** Un evento puede tener múltiples asistentes.

La cardinalidad aquí asegura que el modelo pueda registrar múltiples participaciones de una persona en distintos eventos y múltiples asistentes en cada evento.

❖ Tiene lugar en (Evento - Ubicaciones)

La relación "*Tiene lugar en*" conecta las entidades "*Evento*" y "*Ubicaciones*". Esta relación no solo establece una conexión fundamental entre eventos y lugares, sino que también permite analizar cómo influyen las características de las ubicaciones (como el aforo o el coste del alquiler) al éxito o rentabilidad de los eventos. La cardinalidad de esta relación es la siguiente:

- **(1, 1) para Eventos:** Cada evento se lleva a cabo en una única ubicación.
- **(0, N) para Ubicaciones:** Una ubicación puede albergar múltiples eventos

La cardinalidad definida aquí asegura que cada evento debe ocurrir en única ubicación, pero que una ubicación puede albergar varios eventos siempre que estos sean en diferentes momentos.

❖ Formado por (Evento - Actividades)

La relación "*Formado por*" conecta las entidades "*Evento*" y "*Actividades*". Esta relación es clave para clasificar y analizar los eventos según su tipo de actividad. También permite, entre otras cosas, identificar qué actividades son más populares, cuáles generan mayores ingresos y cómo varían las preferencias del público dependiendo del tipo de actividad. La cardinalidad de esta relación es la siguiente:

- **(1, 1) para Eventos:** Cada evento está asociado a una única actividad.
- **(1, N) para Actividades:** Una actividad puede estar presente en múltiples eventos.

La cardinalidad definida de esta forma asegura que cada evento tenga un propósito claro y se facilite el análisis por tipo de actividad, mientras que también permite reciclar algunas actividades en diferentes eventos.

❖ Participan (Artistas - Actividades)

La relación "*Participan*" relaciona las entidades "*Artistas*" y "*Actividades*". Esta relación es esencial para entender la conexión entre los artistas y las actividades que conforman los diferentes eventos. Además, esta relación cuenta con un atributo "*Caché*" que permite asociar unos honorarios a cada artista que dependan de cada actividad. La cardinalidad de esta relación es la siguiente:

- **(1, N) para Artistas**: Un artista puede participar en múltiples actividades.
- **(1, N) para Actividades:** Una actividad puede incluir a varios artistas.

Se aprecia en la definición de esta cardinalidad la flexibilidad tanto para los artistas como para las actividades.

II. Modelo relacional: Implementación de tablas

El modelo relacional es una representación lógica de los datos basada en la teoría de conjuntos y las relaciones, donde la información se organiza en tablas que están vinculadas mediante claves primarias y foráneas. Este enfoque garantiza la integridad referencial, permite evitar redundancias y facilita la ejecución de consultas complejas. Así, teniendo en cuenta el modelo Entidad – Relación definido en el capítulo anterior, se ha obtenido el siguiente diseño para el modelo relacional:

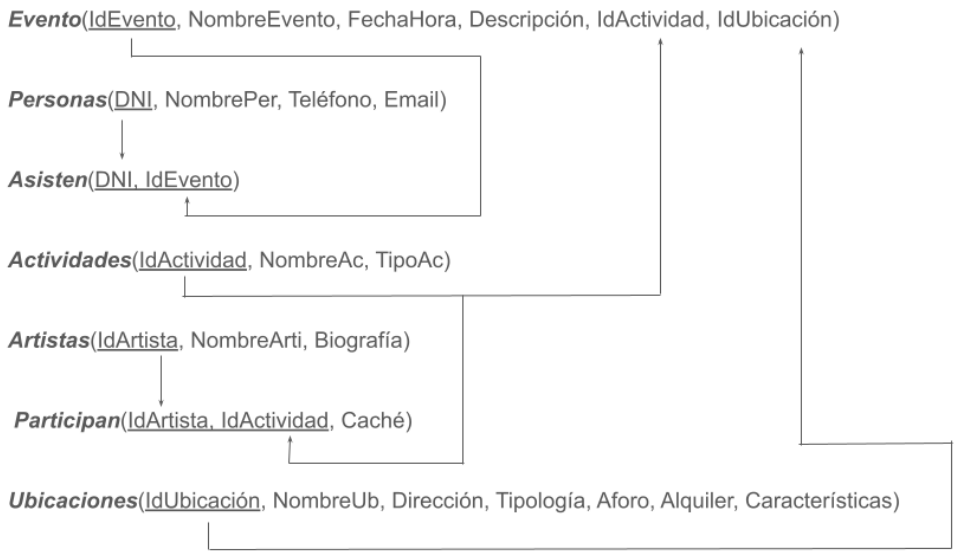

FIGURA 2. Modelo relacional a partir del modelo E-R de la FIGURA 1.

En el modelo diseñado, cada tabla y relación se ha definido cuidadosamente para reflejar las entidades y asociaciones del modelo entidad-relación original, maximizando la eficiencia y la coherencia del sistema. Además, este diseño prioriza la unicidad y consistencia de los datos mediante la correcta elección de claves primarias y foráneas (que se detalla y justifica a continuación). Esto garantiza un sistema flexible y escalable, preparado para soportar tanto consultas simples como operaciones más complejas.

i. Tabla Evento

- **Clave primaria:** "*IdEvento*" se utiliza como identificador único de cada evento, ya que permite diferenciar inequívocamente eventos con nombres o descripciones similares. Esto es fundamental para evitar ambigüedades y para gestionar los eventos de manera eficiente.
- **Claves foráneas:** Se incluyen "*IdActividad*" y "*IdUbicación*" como referencias a las tablas "Actividades" y "Ubicaciones" respectivamente. Estas claves garantizan que cada evento esté asociado con una actividad válida y una ubicación registrada, lo que refuerza la integridad referencial y facilita la consulta cruzada de información entre eventos, actividades y ubicaciones.

ii. Tabla Personas

- **Clave primaria:** Se elige "*DNI*" como clave primaria porque es un identificador único inherente a cada persona. Esto asegura que no haya duplicados en los registros y permite identificar de forma precisa a cada asistente en otros contextos dentro del sistema.

iii. Tabla Asisten

- **Clave primaria compuesta:** La combinación de "*DNI*" e "*IdEvento*" se utiliza como clave primaria, ya que representa de manera única la asistencia de una persona específica a un evento concreto. Esto permite modelar la relación de muchos a muchos entre personas y eventos de forma precisa.
- **Claves foráneas:** Tanto "*DNI*" como "*IdEvento*" se definen como claves foráneas que referencias las tablas "Personas" y "Eventos", asegurando que solo se registren asistencias válidas. Esto garantiza que cada entrada en esta tabla corresponda a una persona existente y un evento registrado.

iv. Tabla Actividades

- **Clave primaria:** "*IdActividad*" se utiliza como identificador único para cada actividad, permitiendo diferenciarlas incluso si tienen nombres o tipos similares. Esto también facilita la reutilización de actividades en distintos eventos.

v. Tabla Artistas

- **Clave primaria:** "*IdArtista*" se establece como clave primaria para identificar de manera única a cada artista. Esto es esencial para gestionar la participación de artistas en múltiples actividades y para evitar duplicidad en sus registros.

vi. Tabla Participan

- **Clave primaria compuesta:** Se define como la combinación de "*IdArtista*" e "*IdActividad*", ya que esta relación de muchos a muchos debe capturar la participación única de un artista en una actividad concreta.
- **Claves foráneas:** "*IdArtista*" e "*IdActividad*" referencian las tablas "Artistas" y "Actividades", asegurando que solo se registren participaciones válidas y vinculando de manera explícita a los artistas con las actividades en las que colaboran. Esto permite también gestionar cachés individuales por participación.

vii. Tabla Ubicaciones

- **Clave primaria:** "*IdUbicación*" se elige como clave primaria para identificar de manera única cada ubicación, permitiendo diferenciar sitios con nombres similares pero características distintas.

III. Script en *SQL*

A continuación, se expone el script de SQL que se ha diseñado para trabajar con la base de datos creada. El script comprende la creación de las tablas definidas en el modelo relacional, la implementación de algunos datos de prueba para comprobar el funcionamiento y, finalmente, algunas consultas de diferente complejidad para comprobar la robustez del diseño. Asimismo, se incluyen también dos "triggers" (para controlar el aforo y para no poder registrar por error la asistencia a eventos que aún no han tenido lugar), así como el diseño de una "vista" para calcular el precio de las entradas en cada evento a partir de otros parámetros. Todos los aspectos técnicos en la generación del código se detallan durante su escritura a modo de comentarios:

```
-- Jacobo Álvarez Gutiérrez
-- Proyecto Final: "ArteVida Cultural"

-- Borrar database si ya existía
DROP DATABASE IF EXISTS ArteVidaCultural;

-- Creación de la base de datos
CREATE DATABASE ArteVidaCultural;
USE ArteVidaCultural;

-- Creación de las tablas correspondientes al modelo conceptual
CREATE TABLE actividades(
IdActividad CHAR(5),
NombreAc VARCHAR(255) NOT NULL,
TipoAc VARCHAR(255) NOT NULL,
PRIMARY KEY (IdActividad)
);

CREATE TABLE artistas(
IdArtista CHAR(5),
NombreArt VARCHAR(255) NOT NULL,
Biografia TEXT,
PRIMARY KEY (IdArtista)
);
```

```sql
CREATE TABLE ubicaciones(
IdUbicacion CHAR(5),
NombreUb VARCHAR(255) NOT NULL,
Direccion VARCHAR(255) NOT NULL,
Tipologia ENUM('Pueblo', 'Ciudad') NOT NULL,
Aforo INT NOT NULL,
Alquiler DECIMAL(10,2) NOT NULL,
Caracteristicas TEXT,
PRIMARY KEY (IdUbicacion)
);

CREATE TABLE personas(
DNI CHAR(9),
NombrePer VARCHAR(255) NOT NULL,
Telefono VARCHAR(20),
email VARCHAR(255) NOT NULL,
PRIMARY KEY (DNI)
);

CREATE TABLE eventos(
IdEvento CHAR(5),
NombreEvento VARCHAR(225) NOT NULL,
FechaHora DATETIME NOT NULL,
Descripcion TEXT,
IdActividad CHAR(5),
IdUbicacion CHAR(5),
PRIMARY KEY (IdEvento),
FOREIGN KEY (IdActividad) REFERENCES actividades(IdActividad),
FOREIGN KEY (IdUbicacion) REFERENCES ubicaciones(IdUbicacion)
);
```

```
/*
ON DELETE CASCADE aquí hace que, si se elimina un cierto evento,
también se eliminen los registros de asistencia al mismo.
*/
CREATE TABLE asisten(
DNI CHAR(9),
IdEvento CHAR(5),
PRIMARY KEY (DNI, IdEvento),
FOREIGN KEY (DNI) REFERENCES personas(DNI),
FOREIGN KEY (IdEvento) REFERENCES eventos(IdEvento) ON DELETE CASCADE
);

CREATE TABLE participan(
IdArtista CHAR(5),
IdActividad CHAR(5),
IdEvento CHAR(5),
Honorarios DECIMAL(10,2) NOT NULL,
PRIMARY KEY (IdArtista, IdActividad, IdEvento),
FOREIGN KEY (IdArtista) REFERENCES artistas(IdArtista),
FOREIGN KEY (IdActividad) REFERENCES actividades(IdActividad),
FOREIGN KEY (IdEvento) REFERENCES eventos(IdEvento) ON DELETE CASCADE
);

/*
Incluimos un trigger para que no puedan incluirse asistencias de personas
a eventos que aún no han tenido lugar.
*/
DELIMITER //

CREATE TRIGGER prevent_future_assistance BEFORE INSERT ON asisten FOR EACH ROW
BEGIN
```

```
    IF (SELECT FechaHora FROM eventos WHERE IdEvento = NEW.IdEvento) >
NOW()
    THEN SIGNAL SQLSTATE '45000'
      SET MESSAGE_TEXT = 'No se pueden registrar asistencias para
    eventos futuros';
      END IF;
END;
//

/*
Añadimos otro trigger para asegurarnos de que no se supere el aforo
del local en cada evento.
*/
CREATE TRIGGER prevent_overcapacity BEFORE INSERT ON asisten FOR EACH
ROW
BEGIN
    IF (SELECT COUNT(DISTINCT DNI) FROM asisten
        WHERE IdEvento = NEW.IdEvento) >=
         (SELECT u.Aforo FROM eventos AS e INNER JOIN ubicaciones AS u
        ON   e.IdUbicacion   =   u.IdUbicacion   WHERE   e.IdEvento   =
NEW.IdEvento)
         THEN SIGNAL SQLSTATE '45000'
       SET MESSAGE_TEXT = 'No se pueden registrar más asistentes:
       Aforo completo';
      END IF;
END;
//

DELIMITER ;

-- Insertamos algunos datos de prueba
INSERT INTO actividades VALUES
('A0001', 'Concierto de Jazz', 'Música'),
```

```
('A0002', 'Exposición de Pintura', 'Arte Visual'),
('A0003', 'Obra de Teatro', 'Teatro'),
('A0004', 'Conferencia sobre Literatura', 'Conferencia'),
('A0005', 'Festival de Cine', 'Cine');

INSERT INTO artistas VALUES
('AR001', 'Carlos López', 'Saxofonista con 20 años de experiencia en jazz
y blues.'),
('AR002', 'Ana Martín', 'Pintora contemporánea reconocida
internacionalmente.'),
('AR003', 'Jorge Ruiz', 'Actor de teatro y televisión.'),
('AR004', 'María Torres', 'Escritora y conferencista sobre
literatura española.'),
('AR005', 'Laura Sánchez', 'Directora de cine especializada en
documentales.'),
('AR006', 'Miguel García', 'Pianista de jazz colaborador de varios
artistas.'),
('AR007', 'Elena Vidal', 'Fotógrafa y artista visual innovadora.');

INSERT INTO ubicaciones VALUES
('U0001', 'Teatro Principal', 'Calle Mayor 123, Madrid', 'Ciudad', 300,
1200.00, 'Teatro con acústica excelente y asientos cómodos.'),
('U0002', 'Centro de Arte Moderno', 'Av. del Arte 45, Barcelona',
'Ciudad',
200, 900.00, 'Sala de exposiciones con iluminación profesional.'),
('U0003', 'Sala Cultural', 'Plaza del Pueblo, Valencia', 'Pueblo', 100,
500.00, 'Espacio para eventos pequeños y medianos.'),
('U0004', 'Auditorio Nacional', 'Paseo de la Reforma 50', 'Ciudad',
1000,
5000.00, 'Gran auditorio con sonido envolvente'),
('U0005', 'Cine Capitol', 'Gran Vía 42, Madrid', 'Ciudad', 500, 1500.00,
'Sala de cine histórica.');
```

```
INSERT INTO personas VALUES
('12345678A', 'Luis Gómez', '600123456', 'luis.gomez@example.com'),
('23456789B', 'Marta Pérez', '650234567', 'marta.perez@example.com'),
('34567890C', 'Juan Ruiz', '620345678', 'juan.ruiz@example.com'),
('45678901D', 'Ana Torres', '630456789', 'ana.torres@example.com'),
('56789012E', 'Carlos Sánchez', '610567890', 'carlos.sanchez@example.com'),
('67890123F', 'Lucía Fernández', '615678901', 'lucia.fernandez@example.com'),
('78901234G', 'Javier López', '620789012', 'javier.lopez@example.com');

INSERT INTO eventos VALUES
('E0001', 'Noche de Jazz', '2023-11-20 20:00:00',
'Concierto de jazz con artistas reconocidos', 'A0001', 'U0001'),
('E0002', 'Exposición Vanguardista', '2023-12-01 10:00:00',
'Muestra de arte contemporáneo', 'A0002', 'U0002'),
('E0003', 'Teatro Clásico', '2023-12-15 19:00:00',
'Representación de una obra clásica', 'A0003', 'U0003'),
('E0004', 'Conferencia Literaria', '2023-11-25 18:00:00',
'Conferencia sobre la literatura española', 'A0004', 'U0004'),
('E0005', 'Festival de Documentales', '2023-01-10 17:00:00',
'Festival con los mejores documentales del año', 'A0005', 'U0005'),
('E0006', 'Jazz y Poesía', '2023-11-30 20:30:00',
'Fusión de música jazz y recital de poesía', 'A0001', 'U0001'),
('E0007', 'Festival de Arte Digital', '2023-02-15 11:00:00',
'Exposición de arte digital contemporáneo', 'A0002', 'U0002'),
('E0008', 'Gala de Jazz', '2025-03-20 20:00:00',
'Concierto de gala con destacados músicos de jazz', 'A0001', 'U0001'),
('E0009', 'Retrospectiva de Pintura', '2025-04-15 18:00:00',
'Exposición retrospectiva de arte contemporáneo', 'A0002', 'U0002'),
('E0010', 'Obra de Teatro Moderna', '2025-05-05 19:30:00',
'Representación teatral moderna', 'A0003', 'U0003'),
```

```
('E0011', 'Congreso Literario', '2025-06-01 10:00:00',
'Congreso internacional sobre literatura', 'A0004', 'U0004'),
('E0012', 'Festival Internacional de Cine', '2025-07-10 16:00:00',
'Festival con proyecciones de cine internacional', 'A0005', 'U0005');

INSERT INTO participan VALUES
('AR001', 'A0001', 'E0001', 800.00),
('AR001', 'A0001', 'E0006', 850.00),
('AR001', 'A0001', 'E0008', 1000.00),
('AR002', 'A0002', 'E0002', 600.00),
('AR002', 'A0002', 'E0007', 650.00),
('AR002', 'A0002', 'E0009', 700.00),
('AR003', 'A0003', 'E0003', 700.00),
('AR003', 'A0003', 'E0010', 750.00),
('AR004', 'A0004', 'E0004', 500.00),
('AR004', 'A0004', 'E0011', 550.00),
('AR005', 'A0005', 'E0005', 1000.00),
('AR005', 'A0005', 'E0012', 1100.00),
('AR006', 'A0001', 'E0001', 750.00),
('AR006', 'A0001', 'E0008', 950.00);

INSERT INTO asisten VALUES
('12345678A', 'E0001'),
('23456789B', 'E0001'),
('34567890C', 'E0002'),
('45678901D', 'E0003'),
('56789012E', 'E0004'),
('67890123F', 'E0005'),
('78901234G', 'E0006'),
('12345678A', 'E0006'),
('23456789B', 'E0007'),
('67890123F', 'E0007');
```

```sql
/*
A continuación, haremos diversas consultas. Estas irán aumentando su complejidad
sucesivamente.
*/

-- NÚMERO DE COLABORACIONES DE CADA ARTISTA
SELECT a.IdArtista, NombreArt, COUNT(IdEvento) AS num_collabs
FROM participan AS p INNER JOIN artistas AS a ON p.IdArtista=a.IdArtista
GROUP BY a.IdArtista
ORDER BY num_collabs DESC;

-- LISTADO DEL NÚMERO DE EVENTOS POR CIUDAD
SELECT e.IdUbicacion, Direccion, COUNT(IdEvento) AS num_eventos
FROM ubicaciones AS u INNER JOIN eventos AS e
ON u.IdUbicacion=e.IdUbicacion
GROUP BY e.IdUbicacion, Direccion
ORDER BY num_eventos DESC;

-- ARTISTAS QUE HAN PARTICIPADO EN MÁS DE UN EVENTO
SELECT a.IdArtista, a.NombreArt, COUNT(p.IdEvento) AS NumEventos
FROM artistas AS a
INNER JOIN participan AS p ON a.IdArtista = p.IdArtista
GROUP BY a.IdArtista, a.NombreArt
HAVING COUNT(p.IdEvento) > 1
ORDER BY NumEventos DESC;

-- CANTIDAD DE ASISTENTES A CADA EVENTO
SELECT asi.IdEvento, NombreEvento, COUNT(DISTINCT(DNI)) AS num_personas
FROM
eventos AS e INNER JOIN asisten AS asi ON
e.IdEvento = asi.IdEvento
```

```sql
GROUP BY asi.IdEvento;


-- EVENTOS QUE YA TUVIERON LUGAR Y SU PORCENTAJE DE OCUPACIÓN EN RELACIÓN CON
-- EL AFORO DE LA UBICACIÓN
/*
En este caso debemos usar un LEFT JOIN al unir las tablas evento y asisten.
Esto se hace así para poder contemplar la posibilidad de que no haya ido
ninguna persona a alguno de los eventos realizados.
*/
SELECT e.IdEvento, NombreEvento, COUNT(DISTINCT(asi.DNI)) AS num_personas,
Aforo, CONCAT(ROUND((COUNT(DISTINCT asi.DNI)/Aforo)*100, 2), '%')
AS porcentaje_ocupacion FROM eventos AS e
LEFT JOIN asisten AS asi ON e.IdEvento=asi.IdEvento
INNER JOIN ubicaciones AS u ON e.IdUbicacion=u.IdUbicacion
WHERE e.FechaHora <= NOW()
GROUP BY e.IdEvento, NombreEvento, Aforo;
/*
Evidentemente, los resultados de esta consulta no son del todo significativos
ya que hablamos de aforos de en torno al millar de personas y apenas hemos
puesto 1 o 2 personas por evento. No obstante, nos sirve para comprobar que
la consulta está bien planteada.
*/


-- LISTADO DE ARTISTAS Y SU PROMEDIO DE HONORARIOS POR LOS EVENTOS YA
-- REALIZADOS
SELECT a.NombreArt, ROUND(AVG(p.Honorarios), 2) AS PromedioHonorarios
FROM artistas AS a
```

```
INNER JOIN participan AS p ON a.IdArtista = p.IdArtista
INNER JOIN eventos AS e ON p.IdEvento = e.IdEvento
WHERE e.FechaHora <= NOW()
GROUP BY a.IdArtista, a.NombreArt
ORDER BY PromedioHonorarios DESC;

-- PRECIO DE LAS ENTRADAS (VISTA)
CREATE VIEW PrecioEntradas AS
SELECT e.IdEvento, e.NombreEvento, u.Aforo,
ROUND((u.Alquiler + COALESCE(SUM(p.honorarios), 0))*1.2/u.Aforo, 2)
AS PrecioEntrada FROM eventos AS e
INNER JOIN ubicaciones AS u ON e.IdUbicacion = u.IdUbicacion
LEFT JOIN participan AS p ON e.IdEvento = p.IdEvento
GROUP BY e.IdEvento, e.NombreEvento, u.Aforo;

-- CONSULTA DE LA VISTA ANTERIOR PARA OBTENER LOS RESULTADOS POR PANTALLA
SELECT IdEvento, NombreEvento, PrecioEntrada FROM PrecioEntradas;

-- EVENTOS DONDE EL PRECIO DE LA ENTRADA ES MENOR A UN VALOR ESPECÍFICO
SELECT pe.IdEvento, pe.NombreEvento, CONCAT(pe.Precioentrada, '€') AS
Entrada
FROM PrecioEntradas AS pe
WHERE pe.PrecioEntrada < 10
ORDER BY pe.PrecioEntrada ASC;

-- LISTADO DE EVENTOS REALIZADOS Y EL BENEFICIO OBTENIDO POR LA EMPRESA
SELECT e.IdEvento, e.NombreEvento,
COUNT(DISTINCT asi.DNI)*PrecioEntrada
-(u.Alquiler + COALESCE(SUM(p.honorarios), 0))
AS beneficio FROM eventos AS e
INNER JOIN asisten AS asi ON e.IdEvento=asi.IdEvento
INNER JOIN PrecioEntradas AS pe ON e.IdEvento=pe.IdEvento
```

```
INNER JOIN ubicaciones AS u ON e.IdUbicacion = u.IdUbicacion
LEFT JOIN participan AS p ON e.IdEvento = p.IdEvento
WHERE e.FechaHora <= NOW()
GROUP BY e.IdEvento, e.NombreEvento, u.Alquiler
ORDER BY beneficio DESC;
/*
Aquí los resultados vuelven a no ser del todo significativos por el mismo
problema con el aforo. El beneficio de la empresa será negativo puesto que,
a modo de prueba, solo se han añadido 1 o 2 asistentes por eventos, mientras
que los valores reales que deberían barajarse aquí es en torno al millar de
personas. No obstante, nos sirve para comprobar que la consulta se lleva a
cabo de forma correcta.
*/

-- LISTADO CON EL PROMEDIO DEL PORCENTAJE DE ASISTENCIA POR CADA TIPO DE
-- ACTIVIDAD
/*
Al igual que ocurría en otra consulta previa, aquí necesitamos hacer un
LEFT JOIN entre eventos y la subconsulta para considerar la posibilidad
de registros de actividades sin ningún asistente al evento.
*/
SELECT a.TipoAc, ROUND(AVG(evento_porcentajes.porcentaje_asistencia), 2)
AS promedio_porcentaje_asistencia FROM actividades AS a
INNER JOIN eventos AS e ON a.IdActividad = e.IdActividad
LEFT JOIN (
    SELECT e.IdEvento, (COUNT(asi.DNI)/u.Aforo)*100 AS porcentaje_asistencia
    FROM eventos AS e INNER JOIN ubicaciones AS u
```

```
    ON e.IdUbicacion = u.IdUbicacion
    LEFT JOIN asisten AS asi ON e.IdEvento = asi.IdEvento
    WHERE e.FechaHora <= NOW()
    GROUP BY e.IdEvento, u.aforo)
AS evento_porcentajes ON e.IdEvento = evento_porcentajes.IdEvento
GROUP BY a.TipoAc
ORDER BY promedio_porcentaje_asistencia DESC;
/*
Ídem que en las dos consultas anteriores que hacen referencia al porcentaje
de ocupación.
*/
```