Università di Catania

PROGETTO BASI DI DATI

Gestione biglietti per eventi estivi

Realizzato da: *Federico Grosso*

INTRODUZIONE

Si intende progettare un database per la gestione delle vendite di biglietti per eventi estivi.

I clienti (circa 10.000), identificati da *un codice* e dai *loro dati anagrafici*, potranno acquistare diversi biglietti, (circa 20.000), per partecipare ai vari eventi programmati.

Gli eventi (circa 40), saranno identificati da *un codice* e dalle *relative informazioni*, e si svolgeranno in una delle location convenzionate (circa 10 location).

Ogni evento verrà organizzato da un solo organizzatore selezionato tra gli 8 organizzatori ingaggiati dalla società.

A ciascun evento possono partecipare anche più artisti contemporaneamente.

Si stima che circa 50 artisti abbiano firmato un contratto con la società.

Dopo questa panoramica generale, si procederà con la progettazione dettagliata del database.

La progettazione è strutturata in *3 livelli fondamentali*:

- **PROGETTAZIONE CONCETTUALE**
- **PROGETTAZIONE LOGICA**
- **PROGETTAZIONE FISICA**

PROGETTAZIONE CONCETTUALE

La progettazione concettuale rappresenta la fase iniziale e più delicata nella creazione di un database, poiché richiede una comprensione precisa e chiara delle richieste del cliente.

In questa fase, l'obiettivo principale è rispondere alla **domanda**:

"COSA verrà rappresentato nella base di dati?"

Durante questa fase, i requisiti vengono schematizzati in modo formale. Vengono definite le varie entità e le loro associazioni, creando una rappresentazione strutturata e comprensibile del sistema richiesto.

ANALISI DEI REQUISITI

Glossario dei termini

Dopo aver raccolto le prime informazioni si costruisce il **glossario dei termini**, in modo tale da schematizzare e ordinare il tutto.

TERMINE	DESCRIZIONE	SINONIMI	TERMINI COLLEGATI
Location	Luogo dove si terrà l'evento	Luogo	**Evento**
Cliente	Partecipante all'evento che acquista il biglietto	Partecipante, acquirente	**Biglietto**
Evento	Evento che si terrà	Serata	**Organizzatore, Artista,Location, Biglietto**
Organizzatore	Organizzatore dell'evento	Responsabile	**Evento**
Artista	Artista che si esibirà all'evento	Ospite	**Evento**
Biglietto	Biglietto per l'evento	Ticket	**Evento, Cliente**

Analisi dei dati di carattere generale

Si analizzano dettagliatamente i vari dati a disposizione.

Dati Location

Di ciascuna location si conosce: il *codice identificativo,* il *nome della location*, il *telefono*, l'*indirizzo*, la *città* e la relativa *provincia*.

Dati Cliente

I clienti che partecipano ai vari eventi sono caratterizzati da: un *codice identificativo*, dal *nome*, dal *cognome*, dall'*età*, dal *numero di cellulare* e dal *sesso* della persona.

Dati Evento

I vari eventi organizzati sono identificati da: un *codice identificativo*, *nome dell'evento*, la *data*, *l'ora di inizio*, *l'ora di fine*, la *tipologia* e la *descrizione* dell'evento.

Dati Organizzatore

Ogni organizzazione è identificato da: un *codice identificativo*, dal *nome*, dal *cognome* e dal *numero di cellulare*.

Dati Artista

Ogni artista è identificato da: un *codice identificativo,* dal *nome d'arte*, dal *genere musicale*, dal *sesso* e da una breve *descrizione*.

Dati Biglietto

Ogni biglietto è identificato da: un *codice identificativo*, dal *nome del biglietto*, dal *tipo di biglietto* e dal *prezzo*.

SCHEMA CONCETTUALE *(MODELLO E-R)*

Dopo aver raccolto e analizzato i requisiti, si definisce una strategia di progettazione per procedere con lo sviluppo del database.

In questo caso, viene adottata la strategia **Top-Down**.

Questa modalità prevede la creazione iniziale di uno schema di base, chiamato **schema scheletro**, che rappresenta una versione preliminare e minimale del progetto.

Questo schema include le principali entità e le loro associazioni.

Successivamente, lo schema scheletro viene modificato diverse volte attraverso un processo iterativo di raffinamento.

In questa fase, si applicano *regole (primitive di trasformazione)* per migliorare e dettagliare il modello iniziale.

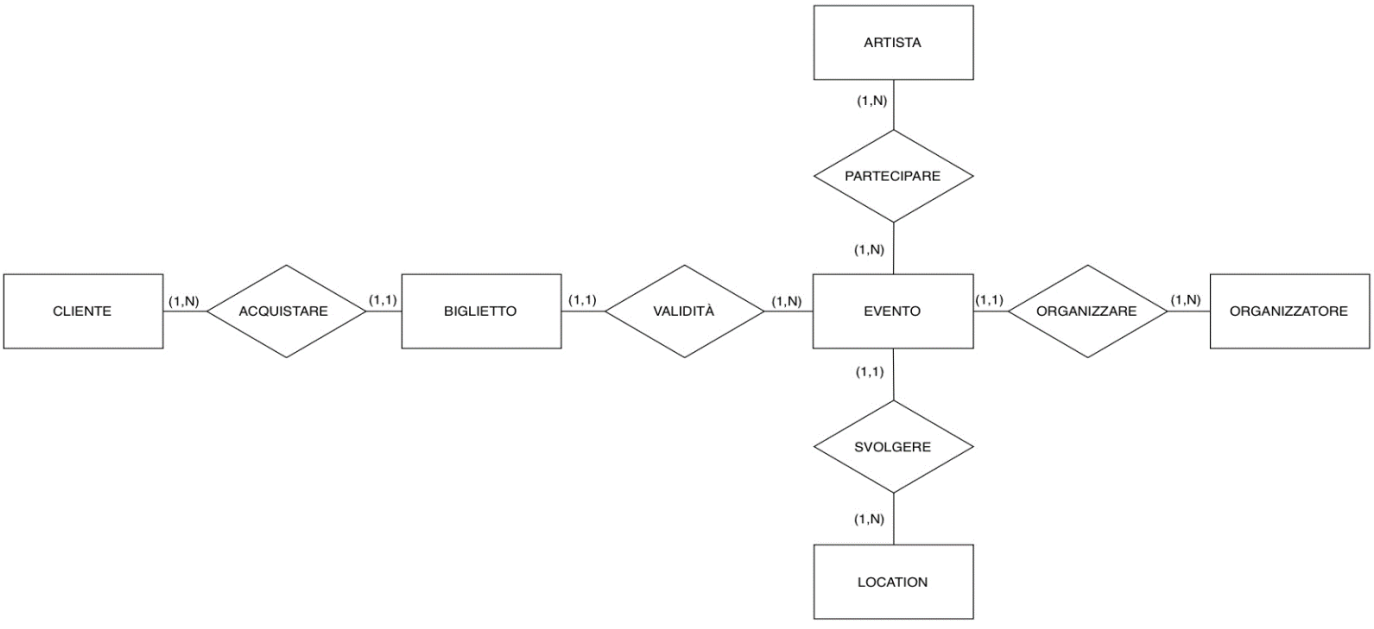
Schema scheletro

Rappresenta l'idea base del progetto.

Schema intermedio

Aggiungo altre entità, relazioni e cardinalità.

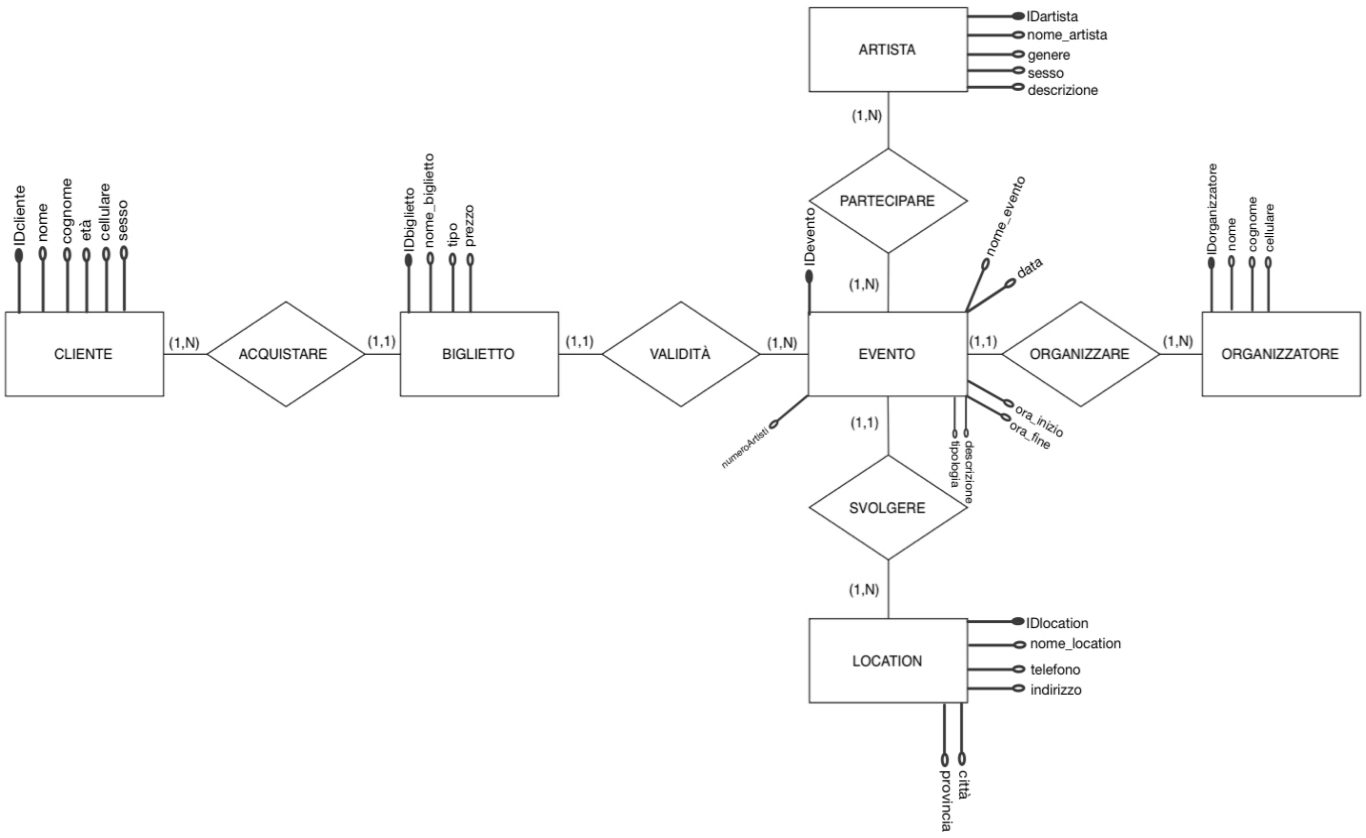
Raffinamento (applicazione regola T1, T3)



Schema finale

Vengono inserite le chiavi e le varie proprietà.

Raffinamento (applicazione regola T5)



Vincoli esprimibili e non esprimibili dallo schema E-R

- ✓ Un biglietto può essere acquistato solo da un cliente, poiché nominativo ***(VINCOLO GARANTITO DALLO SCHEMA)***
- ✓ Un evento può essere organizzato solo da un organizzatore ***(VINCOLO GARANTITO DALLO SCHEMA)***
- ✓ Un biglietto è valido solo per un evento ***(VINCOLO GARANTITO DALLO SCHEMA)***
- ✓ Un evento si svolgerà in una sola location ***(VINCOLO GARANTITO DALLO SCHEMA)***
- ✓ Per una politica aziendale non possono partecipare agli eventi, e di conseguenza non possono acquistare biglietti persone con età inferiore a 14 anni ***(VINCOLO GARANTITO DA UN TRIGGER)***

DIZIONARIO DEI DATI – **ENTITÀ**

ENTITÀ	DESCRIZIONE	ATTRIBUTI	IDENTIFICATORE
Location	Luogo dove si terrà l'evento	IDlocation, nome_location, telefono, indirizzo, città, provincia	IDlocation
Cliente	Persona che parteciperà all'evento	IDcliente, nome, cognome, età, sesso, cellulare	IDcliente
Organizzatore	Persona che organizza l'evento	IDorganizzatore, nome, cognome, cellulare	IDorganizzatore
Artista	Artista che si esibirà all'evento	IDartista, nome_artista, genere, sesso, descrizione	IDartista
Biglietto	Biglietto per accedere all'evento	IDbiglietto, nome_biglietto, tipo_biglietto,prezzo	IDbiglietto
Evento	Evento che si svolgerà	IDevento, nome_evento, data, ora_inizio, ora_fine, tipologia, descrizione, numeroArtisti	IDevento

DIZIONARIO DEI DATI – **RELAZIONI**

RELAZIONE	ENTITÀ PARTECIPANTI	DESCRIZIONE
Partecipare	Artista - Evento	Un artista può partecipare da uno a più eventi *(1,N)* Ad un evento possono partecipare da uno a più artisti *(1,N)*
Acquistare	Cliente - biglietto	Un cliente può acquistare da uno a più biglietti *(1,N)* Ogni biglietto può essere acquistato solo da un cliente *(1,1)*
Validità	Biglietto - Evento	Un biglietto è valido solo per un evento *(1,1)* Ad un evento solo validi da uno a più biglietti *(1,N)*
Organizzare	Organizzatore - Evento	Un evento è organizzato da un solo organizzatore *(1,1)* Un organizzatore può organizzare da uno a più eventi *(1,N)*
Svolgere	Location - Evento	Un evento si svolge in una sola location *(1,1)* In una location si possono svolgere da uno a più eventi *(1,N)*

PROGETTAZIONE LOGICA

Dopo aver completato la progettazione concettuale, si procede con la progettazione logica, creando uno schema logico.

La progettazione logica si articola in *due fasi principali*:

- ***Ristrutturazione dello schema E-R:*** Questa fase consiste nel semplificare e ottimizzare la struttura dello schema Entità-Relazione ottenuto.
- ***Traduzione in uno schema logico:*** Lo schema E-R ottimizzato viene tradotto in uno schema logico, adatto all'implementazione nel sistema di gestione di database scelto.

Nella *prima fase* della progettazione logica, viene effettuata un'analisi delle prestazioni.

Questo processo include la costruzione di due tabelle specifiche per valutare e migliorare l'efficienza dello schema.

TABELLA DEI VOLUMI

CONCETTO	TIPO	VOLUME
Location	E	10
Cliente	E	10000
Organizzatore	E	8
Artista	E	50
Biglietto	E	20000
Evento	E	40
Partecipare	R	150
Acquistare	R	20000
Validità	R	20000
Organizzare	R	40
Svolgere	R	40

TABELLA DELLE OPERAZIONI

OP.	DESCRIZIONE	TIPO	FREQUENZA
OP.1	Visualizzare i nomi e i numeri di cellulare degli organizzatori	I	5 al giorno
OP.2	Contare il numero di artisti che hanno partecipato ad un evento	I	1 al giorno
OP.3	Visualizzare il nome e il numero di cellulare dell'organizzatore di un evento	I	1 al giorno
OP.4	Aggiungere un nuovo cliente	I	50 al giorno
OP.5	Eliminare un cliente	I	2 al giorno
OP.6	Per ogni tipologia di biglietto contare il numero di biglietti erogati *(in ordine crescente)*	I	20 al giorno
OP.7	Inserire due *nuove location*	I	1 al giorno
OP.8	Visualizzare il nome dell'evento, il nome della location e la provincia per gli eventi che si terranno prima del 15 agosto nella provincia di Ragusa. *(solamente eventi pomeridiani, ovvero dalle 15:00 in poi)*	I	1 al giorno
OP.9	Visualizzare il nome e il cognome dell'organizzatore che ha organizzato il maggior numero di eventi	I	1 al giorno
OP.10	Inserire un altro artista come partecipante ad un evento specifico	I	2 al giorno
OP.11	Visualizzare il nome e il cognome del cliente che ha OMAGGIATO il biglietto. Visualizzare anche il nome dell'evento	I	2 al giorno
OP.12	Per ogni evento visualizzare il nome dell'evento e la somma degli incassi generati dalla vendita dei biglietti *(in ordine decrescente)*	I	8 al giorno
OP.13	Aggiungere una colonna "email" nella tabella Clienti per ricevere comunicazioni	B	1 all'anno
OP.14	Aggiornare il recapito telefonico di un organizzatore	B	1 al mese
OP.15	Eliminare la colonna "email" da Clienti	B	1 all'anno

I=Interattiva B=batch **si tratta di stime approssimative*

ANALISI DELLE RIDONDANZE

Sempre nella fase di ristrutturazione, si ricercano all'interno dello schema dei *"dati derivabili",* valutando se conviene mantenerli oppure eliminarli.

Consideriamo *"numeroArtisti"* come attributo ridondante, collocato dentro la tabella Evento e che coinvolge **l'operazione 2** e **10.**

COSTO IN MEMORIA di *numeroArtisti è di 4 byte x 40 eventi= 160 byte (trascurabile)*

Si consideri che 1S=2L

OP.2 con ridondanza

CONCETTO	COSTRUTTO	TIPO OP	N
Evento	E	L	1

1L x 1/GG = **1**

OP.2 senza ridondanza

CONCETTO	COSTRUTTO	TIPO OP	N
Partecipare	R	L	1

50 artisti / 40 eventi=1,25 *media artista x evento* 1L x 1/GG = **1**

OP.10 con ridondanza

CONCETTO	COSTRUTTO	TIPO OP	N
* Partecipare	R	S	1
Evento	E	S	1

non è necessaria la lettura, poiché IDeventi e IDartista sono già noti* (2S) 4L x 2/GG = **8

OP.10 senza ridondanza

CONCETTO	COSTRUTTO	TIPO OP	N
Partecipare	R	S	1

(1S) 2L x 2/GG = **4**

OPERAZIONI	Con ridondanza	senza ridondanza
OP.2	1	1
OP.10	8	4
TOTALE	**9**	**5**

NON CONVIENE MANTENERE LA RIDONDANZA

Le fasi di *eliminazione delle gerarchie, di partizionamento/accorpamento vengono saltate* poiché non utili nello schema in questione.

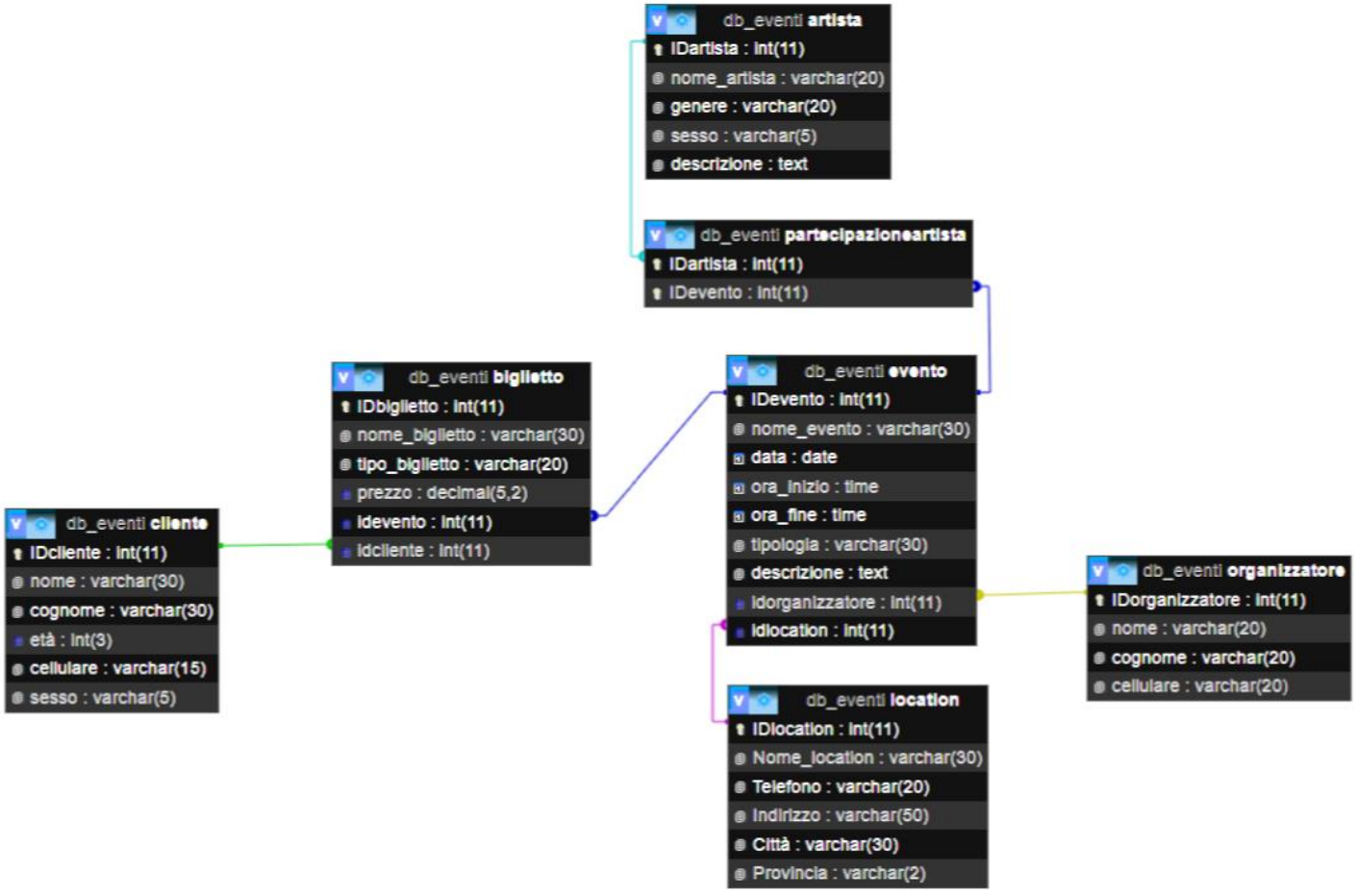
TRADUZIONE NEL MODELLO RELAZIONALE

Lo schema E-R ristrutturato viene trasformato in **schema logico.**

Location (**IDlocation**, nome_location, telefono, indirizzo, città, provincia)
Cliente (**IDcliente**, nome, cognome, età, cellulare, sesso)
Organizzatore (**IDorganizzatore**, nome, cognome, cellulare)
Artista (I**Dartista**, nome_artista, genere, sesso, descrizione)
Biglietto (**IDbiglietto**, nome_biglietto, tipo_biglietto, prezzo, Idevento, Idcliente)
Evento (**IDevento**, nome_evento, data, ora_inizio, ora_fine, tipologia, descrizione, Idorganizzatore, Idlocation)
PartecipazioneArtista (**Idartista**, **Idevento**)

grassetto: *chiavi primarie* *sottolineato*: *chiavi esterne*

MODELLO ERD



PROGETTAZIONE FISICA

La **progettazione fisica** del database è la fase in cui il progetto concettuale e logico viene tradotto in uno schema dettagliato, che definisce come i dati saranno memorizzati, indicizzati e accessibili nel sistema di gestione del database, culminando nella scrittura di script SQL per creare tabelle, indici e relazioni.

Documenti relativi alla progettazione del database

- *Database.sql*
- *Relazione database eventi.pdf*

Implementazione fisica della base di dati

```sql
CREATE DATABASE db_eventi;
USE db_eventi;

-- creazione tabelle

CREATE TABLE Location (
IDlocation INT AUTO_INCREMENT PRIMARY KEY,
nome_location VARCHAR(30) NOT NULL,
telefono VARCHAR(20) NOT NULL,
indirizzo VARCHAR(30) NOT NULL,
città VARCHAR(30) NOT NULL,
provincia VARCHAR(2) NOT NULL
);
```

```sql
CREATE TABLE Cliente (
IDcliente INT AUTO_INCREMENT PRIMARY KEY,
Nome VARCHAR(30) NOT NULL,
Cognome VARCHAR(30) NOT NULL,
Eta INT(3) NOT NULL,
Cellulare VARCHAR (15) NOT NULL,
Sesso VARCHAR (5) NOT NULL
);
```

```sql
CREATE TABLE Organizzatore (
IDorganizzatore INT AUTO_INCREMENT PRIMARY KEY,
Nome VARCHAR(20) NOT NULL,
Cognome VARCHAR(20) NOT NULL,
Cellulare VARCHAR(20) NOT NULL
);
```

```sql
CREATE TABLE Artista (
IDartista INT(11) AUTO_INCREMENT PRIMARY KEY,
Nome_artista VARCHAR(20) NOT NULL,
Genere VARCHAR(20) NOT NULL,
Sesso VARCHAR(5) NOT NULL,
Descrizione TEXT NOT NULL
);
```

```sql
CREATE TABLE Evento (
IDevento INT AUTO_INCREMENT PRIMARY KEY,
Nome_evento VARCHAR(30) NOT NULL,
Data DATE NOT NULL,
Ora_inizio TIME NOT NULL,
Ora_fine TIME NOT NULL,
Tipologia VARCHAR(30) NOT NULL,
Descrizione TEXT NOT NULL,
Idorganizzatore INT NOT NULL,
Idlocation INT NOT NULL,
FOREIGN KEY (Idorganizzatore) REFERENCES Organizzatore(IDorganizzatore),
FOREIGN KEY (Idlocation) REFERENCES Location(IDlocation)
);
```

```sql
CREATE TABLE Biglietto (
IDbiglietto INT AUTO_INCREMENT PRIMARY KEY,
Nome_biglietto VARCHAR(30) NOT NULL,
Tipo_biglietto VARCHAR(20) NOT NULL,
Prezzo DECIMAL(5,2) NOT NULL,
Idevento INT NOT NULL,
Idcliente INT NOT NULL,
FOREIGN KEY (Idevento) REFERENCES Evento (IDevento),
FOREIGN KEY (Idcliente) REFERENCES Cliente (IDcliente)
);
```

```sql
CREATE TABLE PartecipazioneArtista (
Idartista INT NOT NULL,
Idevento  INT NOT NULL,
PRIMARY KEY(Idartista,Idevento),
FOREIGN KEY (Idartista) REFERENCES Artista(IDartista),
FOREIGN KEY (Idevento) REFERENCES Evento(IDevento)
);
```

Implementazione delle operazioni

```sql
-- OP.1 Visualizzare i nomi e i numeri di cellulare degli organizzatori
SELECT Nome, Cellulare
FROM Organizzatore;
```

```sql
-- OP.2 Contare il numero di artisti che hanno partecipato ad un evento ('Radio24')
SELECT COUNT(IDartista) AS ArtistiPresenti
FROM Evento e JOIN PartecipazioneArtista p ON e.IDevento=p.Idevento
WHERE Nome_evento='Radio24';
```

```sql
-- OP.3 Visualizzare il nome e il numero di cellulare dell'organizzatore di un evento ('Scateniamoci')
SELECT o.Nome, o.Cellulare
FROM Evento e, Organizzatore o
WHERE e.Idorganizzatore= o.IDorganizzatore AND Nome_evento = 'Scateniamoci';
```

```sql
-- OP.4 Aggiungere un nuovo cliente ('Mattia Rosso')
INSERT INTO Cliente (Nome, Cognome, Eta, Cellulare, Sesso) VALUES
('Mattia', 'Rosso', 48, '3384112443', 'uomo');
```

```sql
-- OP.5 Eliminare un cliente (bannare 'Mattia Rosso')
DELETE FROM Cliente WHERE Nome= 'Mattia'AND Cognome='Rosso';
```

```sql
-- OP.6 Per ogni tipologia di biglietto, contare il numero di biglietti erogati (in ordine crescente)
SELECT Tipo_biglietto, COUNT(IDbiglietto) AS numeroBiglietti
FROM Biglietto
GROUP BY Tipo_biglietto
ORDER BY numeroBiglietti;
```

```sql
-- OP.7 Inserire due nuove location
INSERT INTO Location (Nome_location, Telefono, Indirizzo, Città, Provincia) VALUES
('Scirocco', '0933952888', 'Via Alessandro Verdi, 5', 'Pozzallo', 'RG'),
('Villa Smeralda', '0933952115', 'Via Bertordi, 155', 'Ispica', 'RG');
```

```sql
-- OP.8 Visualizzare il nome dell'evento, il nome della location e la provincia per gli eventi che si terranno prima
-- del 15 agosto nella provincia di Ragusa. (solamente eventi pomeridiani, ovvero dalle 15:00 in poi).
SELECT e.Nome_evento, l.Nome_location, e.Ora_inizio, l.Provincia
FROM Evento e JOIN Location l ON e.Idlocation = l.IDlocation
WHERE e.Data<'2024-08-15'  AND
      e.Ora_inizio>='15:00:00' AND
      l.Provincia='RG';
```

```sql
-- OP.9 Visualizzare il nome e il cognome dell'organizzatore che ha organizzato il maggior numero di eventi
CREATE VIEW V1 AS
SELECT COUNT(IDevento) AS numeroEventi
FROM Evento
GROUP BY IdOrganizzatore;

SELECT o.Nome, o.Cognome, COUNT(e.IDevento) AS EventiOrganizzati
FROM Evento e JOIN Organizzatore o ON e.Idorganizzatore = o.IDorganizzatore
GROUP BY o.Nome, o.Cognome
HAVING COUNT(e.IDevento)=(SELECT MAX(numeroEventi)
                          FROM V1);
```

```sql
-- OP.10 Inserire un altro artista (già presente nella tabella Artista) come partecipante ad un evento specifico
-- (dato Idartista=19 e Idevento=8)
INSERT INTO PartecipazioneArtista (Idartista,Idevento) VALUES (19,8);
```

```sql
-- OP.11 Visualizzare il nome e il cognome del cliente che ha OMAGGIATO il biglietto. Visualizzare anche il nome dell'evento
SELECT c.Nome, c.Cognome, e.Nome_evento
FROM Biglietto b JOIN Cliente c ON b.Idcliente = c.IDcliente
                 JOIN Evento e  ON b.Idevento = e.IDevento
WHERE Tipo_biglietto='OMAGGIATO';
```

```sql
-- OP.12 Per ogni evento visualizzare il nome dell'evento e la somma degli incassi generati dalla vendita dei biglietti (in ordine decrescente)
SELECT e.Nome_evento, SUM(b.Prezzo) AS IncassoTotale
FROM Evento e JOIN Biglietto b ON e.IDevento = b.Idevento
GROUP BY Nome_evento
ORDER BY IncassoTotale DESC;
```

```sql
-- OP.13 Aggiungere una colonna "email" nella tabella Clienti per ricevere comunicazioni
ALTER TABLE Cliente ADD Email VARCHAR(30);
```

```sql
-- OP.14 Aggiornare recapito telefonico di un organizzatore ('Marcello Ascani')
UPDATE Organizzatore SET Cellulare='1111111111' WHERE Nome='Marcello' AND Cognome='Ascani';
```

```sql
-- OP.15 Eliminare la colonna "email" da Clienti (ps. FLOP totale, nessuno si iscriveva)
ALTER TABLE Cliente DROP COLUMN Email;
```

```sql
-- OP.16 Eliminare la vista V1 dopo l'utilizzo
DROP VIEW V1;
```

Implementazione dei vincoli d'integrità

```sql
-- TRIGGER

DELIMITER //

CREATE TRIGGER check_age_before_insert
BEFORE INSERT ON Cliente
FOR EACH ROW
BEGIN

    IF NEW.Eta < 14 THEN

        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'Accesso non consentito ai minori di 14 anni';
    END IF;

END //

DELIMITER ;
```