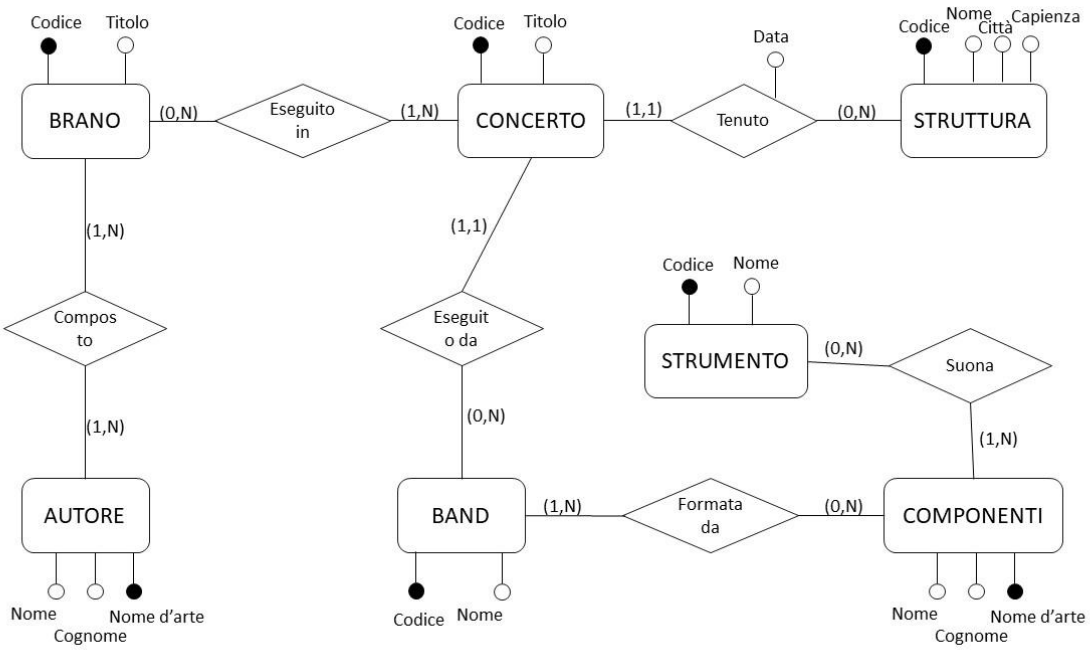
OBIETTIVO

Si è incaricati di progettare e implementare un database per una piattaforma musicale che vuole gestire informazioni sugli artisti, i brani, le band, i concerti e le strutture in cui si svolgono i concerti. La piattaforma desidera tenere traccia di chi ha scritto e registrato i brani, quali band partecipano ai concerti e dove vengono tenuti.

Dopo la progettazione, bisognerà eseguire le seguenti interrogazioni:

1. Elenco delle strutture con la media di capienza, ordinate per media decrescente.
2. Numero totale di concerti per ciascuna band, ordinato per numero decrescente di concerti.
3. Elenco delle band e del loro numero di membri, ordinate per numero di membri crescente.
4. Elenco dei brani registrati da autori che hanno partecipato a concerti, ordinato per nome dell'autore in ordine alfabetico.
5. Numero medio di brani registrati dagli autori per ciascuna band, ordinato per numero decrescente di brani.
6. Elenco delle strutture e del numero di concerti svolti in ciascuna, ordinato per numero decrescente di concerti.
7. Elenco dei concerti con più di 3 brani eseguiti, ordinato per data crescente.
8. Elenco delle band con il numero totale di concerti in cui hanno eseguito brani, ordinate per numero decrescente di concerti.
9. Nome degli autori che hanno registrato più di 3 brani, ordinati per nome dell'autore in ordine alfabetico.
10. Elenco dei concerti svolti in una struttura con capienza superiore a 1500 persone, ordinato per data decrescente.

MODELLO ER



MODELLO LOGICO

Tabella "Autore": Questa tabella conserva i dati sugli autori (artisti) musicali. L'attributo "NomeArte" agisce come chiave primaria, identificando univocamente ciascun autore. Gli attributi "Nome" e "Cognome" rappresentano il nome e il cognome dell'autore.

Tabella "Brano": Rappresenta i dati relativi ai brani musicali. Ogni brano è identificato dall'attributo "CodBrano". L'attributo "Titolo" contiene il titolo del brano, mentre l'attributo "NomeArte" è una chiave esterna che collega il brano all'autore che lo ha registrato.

Tabella "Struttura": Questa tabella contiene le informazioni sulle strutture in cui si tengono i concerti. L'attributo "CodStruttura" funge da chiave primaria e "Nome" rappresenta il nome della struttura. "Città" contiene il luogo in cui si trova la struttura e "Capienza" indica il numero di persone che la struttura può ospitare.

Tabella "Band": Rappresenta le band musicali. Ogni band ha un "CodBand" univoco come chiave primaria e un "Nome" per identificarla.

Tabella "Strumento": Questa tabella contiene i tipi di strumenti musicali. Ogni strumento è identificato da un "CodStrumento" univoco e ha un "Nome" per indicare il tipo di strumento.

Tabella "Componenti": Questa tabella collega gli autori alle band e agli strumenti. "NomeArte" è una chiave primaria e collega l'autore o il membro di una band. Gli attributi "CodBand" e "CodStrumento" sono chiavi esterne che indicano la band di appartenenza o lo strumento suonato.

Tabella "Concerto": Rappresenta i dati relativi ai concerti. Ogni concerto è identificato da un "CodConcerto" univoco. "Titolo" contiene il titolo del concerto, mentre "CodStruttura", "CodBand" e "CodBrano" sono chiavi esterne che collegano il concerto alla struttura in cui si svolge, alla band che partecipa e al brano eseguito.

PROGETTAZIONE FISICA

Creazione delle tabelle:

```
CREATE TABLE Autore (
        NomeArte VARCHAR(100),
        Nome VARCHAR(70),
        Cognome VARCHAR(100),
        PRIMARY KEY (NomeArte));
CREATE TABLE Brano (
        CodBrano INT,
        Titolo VARCHAR(200),
        NomeArte VARCHAR(100),
        PRIMARY KEY (CodBrano),
        FOREIGN KEY (NomeArte) REFERENCES Autore (NomeArte));
CREATE TABLE Struttura (
```

```
    CodStruttura INT,
    Nome VARCHAR(100),
    Città VARCHAR(100),
    Capienza INT,
    PRIMARY KEY (CodStruttura));
CREATE TABLE Band (
    CodBand INT,
    Nome VARCHAR(100),
    PRIMARY KEY (CodBand));
CREATE TABLE Strumento (
    CodStrumento INT,
    Nome VARCHAR(50),
    PRIMARY KEY (CodStrumento));
CREATE TABLE Componenti (
    NomeArte VARCHAR(100),
    Nome VARCHAR(70),
    Cognome VARCHAR(100),
    CodBand INT,
    CodStrumento INT,
    PRIMARY KEY (NomeArte),
    FOREIGN KEY (CodBand) REFERENCES Band (CodBand),
    FOREIGN KEY (CodStrumento) REFERENCES Strumento (CodStrumento));
CREATE TABLE Concerto (
    CodConcerto INT,
    Titolo VARCHAR(150),
    CodStruttura INT,
    Data DATE,
    CodBand INT,
    CodBrano INT,
    PRIMARY KEY (CodConcerto),
    FOREIGN KEY (CodStruttura) REFERENCES Struttura (CodStruttura),
    FOREIGN KEY (CodBand) REFERENCES Band (CodBand),
```

```
    FOREIGN KEY (CodBrano) REFERENCES Brano (CodBrano));
```

Creazione viste:

1.

```
CREATE VIEW TourBand AS
SELECT B.CodBand, B.Nome AS Nome_Band, S.CodStruttura, S.Nome AS Nome_Struttura, S.Città, C.Data
FROM Band B
JOIN Concerto C ON B.CodBand = C.CodBand
JOIN Struttura S ON C.CodStruttura = S.CodStruttura
```

2.

```
CREATE VIEW VistaAutori AS
SELECT NomeArte
FROM Autore;
```

INTERROGAZIONI

1. **Elenco delle strutture con la media di capienza, ordinate per media decrescente.**

```
SELECT S.CodStruttura, S.Nome, AVG(S.Capienza) AS MediaCapienza
FROM Struttura S
JOIN Concerto C ON S.CodStruttura = C.CodStruttura
GROUP BY S.CodStruttura, S.Nome
ORDER BY MediaCapienza DESC;
```

2. **Numero totale di concerti per ciascuna band, ordinato per numero decrescente di concerti.**

```
SELECT B.Nome, COUNT(C.CodConcerto) AS NumeroConcerti
FROM Band B
LEFT JOIN Concerto C ON B.CodBand = C.CodBand
GROUP BY B.Nome
ORDER BY NumeroConcerti DESC;
```

3. **Elenco delle band e del loro numero di membri, ordinate per numero di membri crescente.**

```
SELECT B.Nome, COUNT(C.NomeArte) AS NumeroMembri
FROM Band B
LEFT JOIN Componenti C ON B.CodBand = C.CodBand
GROUP BY B.Nome
```

```
ORDER BY NumeroMembri ASC;
```

4. **Elenco dei brani registrati da autori che hanno partecipato a concerti, ordinato per nome dell'autore in ordine alfabetico.**

```
SELECT DISTINCT A.NomeArte, B.Titolo

FROM Autore A

JOIN Brano B ON A.NomeArte = B.NomeArte

JOIN Concerto C ON B.CodBrano = C.CodBrano

ORDER BY A.NomeArte ASC;
```

5. **Numero medio di brani registrati dagli autori per ciascuna band, ordinato per numero decrescente di brani.**

```
SELECT A.NomeArte, COUNT(B.CodBrano) AS NumeroBraniRegistrati

FROM Autore A

LEFT JOIN Brano B ON A.NomeArte = B.NomeArte

GROUP BY A.NomeArte

HAVING COUNT(B.CodBrano) > 0

ORDER BY NumeroBraniRegistrati DESC;
```

6. **Elenco delle strutture e del numero di concerti svolti in ciascuna, ordinato per numero decrescente di concerti.**

```
SELECT S.Nome, COUNT(C.CodConcerto) AS NumeroConcerti

FROM Struttura S

LEFT JOIN Concerto C ON S.CodStruttura = C.CodStruttura

GROUP BY S.Nome

ORDER BY NumeroConcerti DESC;
```

7. **Elenco dei concerti con più di 3 brani eseguiti, ordinato per data crescente.**

```
SELECT C.Titolo, C.Data

FROM Concerto C

WHERE (SELECT COUNT(*) FROM Brano B WHERE B.CodBrano = C.CodBrano) > 3

ORDER BY C.Data ASC;
```

8. **Elenco delle band con il numero totale di concerti in cui hanno eseguito brani, ordinate per numero decrescente di concerti.**

```
SELECT B.Nome, COUNT(E.CodConcerto) AS NumeroConcerti

FROM Band B

JOIN Componenti C ON B.CodBand = C.CodBand
```

```
JOIN Concerto E ON C.NomeArte = E.CodBand

GROUP BY B.Nome

ORDER BY NumeroConcerti DESC;
```

9. **Nome degli autori che hanno registrato più di 3 brani, ordinati per nome dell'autore in ordine alfabetico.**

```
SELECT A.NomeArte

FROM Autore A

JOIN Brano B ON A.NomeArte = B.NomeArte

GROUP BY A.NomeArte

HAVING COUNT(B.CodBrano) > 3

ORDER BY A.NomeArte ASC;
```

10. **Elenco dei concerti svolti in una struttura con capienza superiore a 1500 persone, ordinato per data decrescente.**

```
SELECT C.Titolo, S.Nome, C.Data

FROM Concerto C

JOIN Struttura S ON C.CodStruttura = S.CodStruttura

WHERE S.Capienza > 1500

ORDER BY C.Data DESC;
```