UNIVERSITÁ DI FERRARA

CORSO DI INFORMATICA

Progetto Basi di Dati

Autori:
Alessio Celentano, Thomas Cantuti

Università
degli Studi
di Ferrara

Contents

Chapter 1

Analisi e inserimento dei dati

1.1 Schema ER

Come prima cosa abbiamo creato lo schema Entity-Relationship (ER) per renderci conto dei dati effettivi che avevamo a disposizione.
Lo schema comprende due entità principali: Artisti e Opere.
La relazione tra queste entità è "crea", dove un artista può creare molte opere, ma ogni opera è associata a un solo artista.

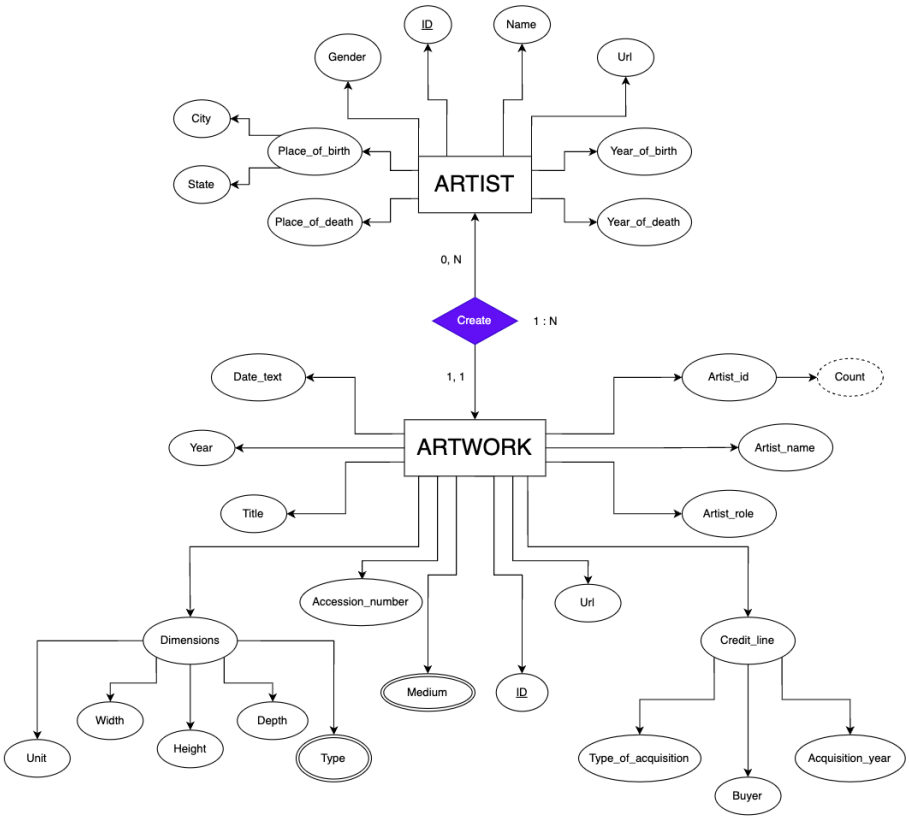

FIGURE 1.1: ER

1.2 Modello Relazionale

Il modello relazionale deriva direttamente dallo schema ER:

- `Artists(artistId, name, gender, birthYear, birthCity, birthState, deathYear, deathCity, deathState, artistUrl, artwork_id)`
- `Artworks(artworkId, accessionNumber, title, dateText, medium, creditLine, year, acquisitionYear, types, width, heigth, depth, units, inscription, thumbnailUrl, artworkUrl, artist_id, artistName, artistRole)`

La chiave esterna `artist_id` in `Artworks` fa riferimento all'id in `Artists`.

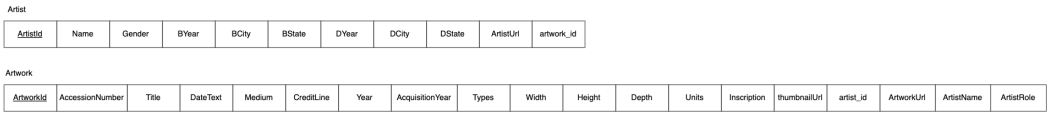

FIGURE 1.2: Modello relazionale

1.3 Pulizia dei dati

In seguito abbiamo iniziato la pulizia dei dati nel seguente ordine:

1. Visualizzazione dei valori `NULL` per ogni attributo e studiarli bene affinché potessimo capire come sostituirli/aggiustarli (es.: l'attributo `dimensions` poteva contenere dimensioni che erano mancanti in `width`, `height` e `depth` quindi, quando presenti in `dimensions`, li abbiamo sostituiti ai valori `NULL`);
2. Definito nuovi attributi ed eliminato attributi composti dove necessario, come gli attributi `placeOfBirth` e `placeOfDeath` che sono stati scomposti in `BCity`, `BState`, `DCity` e `DState`, l'attributo `dimensions` scomposto in `types`, `width`, `height`, `depth` e `units`;
3. definito in chiaro i tipi di ogni attributo.

Di seguito i passaggi precisi per pulire i due dataset.

1.3.1 Pulizia dataset: Artists

- Gender: Riempito i valori mancanti con '-' e convertito i valori 'Male' e 'Female' in 'M' e 'F' rispettivamente;
- Place of Birth/Death: Riempito i valori mancanti con 'Unknown';
- Birth City/State: Diviso `placeOfBirth` in `birthCity` e `birthState` basandosi sulla presenza di una virgola. Se `birthState` è mancante, è stato riempito con 'Unknown';
- Death City/State: Divisio`placeOfDeath` in `deathCity` e `deathState` seguendo la stessa logica;
- Year of Birth/Death: Convertiti in interi riempiendo i valori mancanti con 0;
- Rimosse le colonne originali placeOfBirth, placeOfDeath, e datesl
- Definito le colonne da mantenere e in quale ordine.

1.3.2 Pulizia dataset: Artworks

- Rimossa la colonna `thumbnailCopyright`;
- Verificato che `artistId` in `Artworks` esista in `Artists`;
- Units: Riempito i valori mancanti con 'mm';
- Credit Line, Depth, Year, Inscription, Dimensions, Medium, Acquisition Year: Riempiti i valori mancanti con valori appropriati (0 o 'Unknown');
- Acquisition Year: Estratto l'anno dalla colonna `creditLine`;
- Width/Height: Estratti i valori dalle dimensioni;
- Types: Estratto il tipo dalle dimensioni;
- Convertiti campi come `year`, `acquisitionYear`, `width`, `height`, `depth` in interi;
- Sostituito "/www." con "/media." in `thumbnailUrl` per poter visualizzare le immagini;
- Definito le colonne da mantenere e in quale ordine.

1.4 Inserimento dei dati

Per inserire i dati puliti nei database MySQL, abbiamo eseguito i seguenti passaggi:

1. Creazione del database MySQL;
2. Connessione al database MySQL tramite Python e il modulo PyMySQL;
3. Verifica dell'esistenza delle tabelle `Artists` e `Artworks`: se esistono giá, saltare i passaggi successivi
4. Creazione delle tabelle `Artists` e `Artworks` tramite le query definite in un file apposito;

```
CREATE TABLE Artists (
    id INTEGER NOT NULL,
    name VARCHAR(255) NOT NULL,
    gender CHAR NOT NULL,
    year_of_birth CHAR(4) NOT NULL,
    birth_city VARCHAR(50) NOT NULL,
    birth_state VARCHAR(50) NOT NULL,
    year_of_death VARCHAR(4),
    death_city VARCHAR(50),
    death_state VARCHAR(50),
    url VARCHAR(255) NOT NULL,
    PRIMARY KEY (id)
)
```

```
CREATE TABLE Artworks (
    id INTEGER NOT NULL,
    accession_number CHAR(7) NOT NULL,
    artist VARCHAR(255),
    artistRole VARCHAR(100),
    artistId INTEGER NOT NULL,
    title VARCHAR(2047),
    dateText VARCHAR(255),
    medium VARCHAR(255),
    creditLine VARCHAR(2047),
    year INTEGER,
    acquisitionYear INTEGER,
    types VARCHAR(100),
    width INTEGER,
    height INTEGER,
    depth INTEGER,
    units CHAR(2),
    inscription CHAR(15),
    thumbnailUrl VARCHAR(255),
    url VARCHAR(255),
    PRIMARY KEY (id),
    FOREIGN KEY (artistId) REFERENCES Artists(id)
)
```

5. Lettura dei dati puliti dai file CSV e inserimento nelle rispettive tabelle del database tramite le query definite in un file apposito.

```
INSERT INTO Artists VALUES ( ... )
```

```
INSERT INTO Artworks VALUES ( ... )
```

Chapter 2

Query SQL

2.1 Ricerca degli artisti

In questa sezione del sito web abbiamo accorpato per convenienza diverse funzionalitá:

- Ricerca di un artista inserendo uno o più parametri (anche parziali) - nel caso in cui nessun parametro venga specificato deve essere presentata la lista completa degli artisti;
- La tabella contiene il link alla pagina contenente la tabella delle opere di un artista (che é strutturalmente uguale a quella della ricerca delle opere);
- Inoltre, contiene il numero di opere per artista.

```
SELECT Artists.*, COUNT(Artworks.id)
FROM Artists
LEFT JOIN Artworks ON Artists.id = Artworks.artistId
WHERE (
    name LIKE '%$name%' AND
    gender LIKE '%$gender%' AND
    year_of_birth LIKE '%$year_of_birth%' AND
    year_of_death LIKE '%$year_of_death%' AND
    birth_city LIKE '%$birth_city%' AND
    birth_state LIKE '%$birth_state%' AND
    death_city LIKE '%$death_city%' AND
    death_state LIKE '%$death_state%'
)
GROUP BY Artists.id
```

La tabella contenente tutte le opere di un artista, che appare cliccando l'apposito pulsante, é ottenuta tramite la seguente query:

```
SELECT *
FROM Artists
JOIN Artworks ON Artists.id = Artworks.artistId
WHERE (Artists.id=<artist_id>)
ORDER BY Artworks.medium
```

dove `<artist_id>` é un informazione che otteniamo nel momento in cui clicchiamo il pulsante ed é presente nella variabile `$_GET['artist_id']`

2.2 Ricerca delle opere

Ricerca delle opere inserendo uno o più parametri (anche parziali), in forma libera o eventualmente guidata

```
SELECT *
FROM Artworks
WHERE (
    accession_number LIKE '%$accession_number%' AND
    title LIKE '%$title%' AND
    year LIKE '%$year%' AND
    medium LIKE '%$medium%'
)
```

2.3 Statistiche

Quest'ultima pagina mostra diverse statistiche che riguardano il database:

- Numero di opere realizzate in un determinato anno

```
SELECT COUNT(*)
FROM Artworks
WHERE (year = '$year')
```

- Numero di artisti nati e/o morti in una determinata nazione

```
SELECT COUNT(*)
FROM Artists
WHERE (birth_state = '$nation' OR death_state = '$nation')
```

- Visualizza il numero di artisti nati e/o morti in una determinata città

```
SELECT COUNT(*)
FROM Artists
WHERE (birth_city = '$city' OR death_city = '$city')
```

- Visualizza i 5 artisti che hanno realizzato il maggior numero di opere

```
SELECT Artists.name, COUNT(*) AS num_artworks
FROM Artworks
JOIN Artists ON Artworks.artistId = Artists.id
GROUP BY Artists.id
ORDER BY num_artworks DESC
LIMIT 5
```

- Visualizzare la media del numero di opere per ogni nazione di nascita

```
SELECT AVG(Subtable.artworks_count)
FROM Artists
LEFT JOIN (
    SELECT artistId, COUNT(*) AS artworks_count
    FROM Artworks
    GROUP BY artistId
) AS Subtable ON Artists.id = Subtable.artistId
WHERE Artists.birth_state = '$nation_of_birth'
```

Tutte le stringhe che cominciano con il simbolo '$' si riferiscono a variabili nelle quali é memorizzato un valore specificato dall'utente.