Caso di Studio: Database di Docenti e Studenti di un Liceo

Descrizione: In questo caso di studio, consideriamo un database per un liceo con informazioni su docenti, studenti, corsi, materie e iscrizioni. I docenti tengono lezioni, i corsi sono composti da materie e gli studenti si iscrivono ai corsi.

Modello ER:

---

1. Entità: Docente Attributi: ID (PK), Nome, Cognome, Email, MateriaInsegnata
2. Entità: Studente Attributi: ID (PK), Nome, Cognome, DataNascita, AnnoFrequentazione
3. Entità: Corso Attributi: ID (PK), Nome, AnnoAccademico, Docente_ID (FK)
4. Entità: Materia Attributi: ID (PK), Nome
5. Entità: Iscrizione Attributi: ID (PK), DataIscrizione, Studente_ID (FK), Corso_ID (FK)

Diagramma ER:

[**Docente**]
ID
Nome
Cognome
Email
MateriaInsegnata

[**Studente**]
ID
Nome
Cognome
DataNascita
AnnoFrequentazione

[**Corso**]
ID
Nome
AnnoAccademico
Docente_ID

[**Materia**]
ID
Nome

[**Iscrizione**]
ID
DataIscrizione
Studente_ID
Corso_ID

Le relazioni tra le entità sono le seguenti:

- Docente insegna Materia
- Studente frequenta Corso
- Iscrizione collega Studente e Corso

La relazione ***Docente*** insegna è una relazione 1-n, in quanto un docente può insegnare una o più materie, ma una materia può essere insegnata da un solo docente.
La relazione ***Studente frequenta*** è una relazione 1-n, in quanto uno studente può frequentare uno o più corsi, ma un corso può essere frequentato da uno o più studenti.
La relazione ***Iscrizione*** è una relazione 1-n, in quanto un'iscrizione può essere associata a un solo studente e a un solo corso.

I vincoli sulle relazioni sono i seguenti:
Docente insegna

- Docente_ID deve essere univoco.

Studente frequenta

- Studente_ID deve essere univoco.

Iscrizione

- ID deve essere univoco.
- DataIscrizione deve essere univoca per ogni combinazione di Studente_ID e Corso_ID.

Modello Logico:

- Docente (ID, Nome, Cognome, Email, MateriaInsegnata)
- Studente (ID, Nome, Cognome, DataNascita, AnnoFrequentazione)
- Corso (ID, Nome, AnnoAccademico, Docente_ID)
- Materia (ID, Nome)
- Iscrizione (ID, DataIscrizione, Studente_ID, Corso_ID)

[**Docente**]
ID INT NOT NULL AUTO_INCREMENT,
Nome VARCHAR(255) NOT NULL,
Cognome VARCHAR(255) NOT NULL,
Email VARCHAR(255) NOT NULL,
PRIMARY KEY (ID)

[**Studente**]
ID INT NOT NULL AUTO_INCREMENT,
Nome VARCHAR(255) NOT NULL,
Cognome VARCHAR(255) NOT NULL,
DataNascita DATE NOT NULL,
AnnoFrequentazione INT NOT NULL,
PRIMARY KEY (ID)

[**Corso**]
ID INT NOT NULL AUTO_INCREMENT,
Nome VARCHAR(255) NOT NULL,
AnnoAccademico INT NOT NULL,
Docente_ID INT NOT NULL,
PRIMARY KEY (ID),
FOREIGN KEY (Docente_ID) REFERENCES Docente (ID)

[**Materia**]
ID INT NOT NULL AUTO_INCREMENT,
Nome VARCHAR(255) NOT NULL,
PRIMARY KEY (ID)

[**Iscrizione**]
ID INT NOT NULL AUTO_INCREMENT,
DataIscrizione DATE NOT NULL,
Studente_ID INT NOT NULL,
Corso_ID INT NOT NULL,
PRIMARY KEY (ID),
FOREIGN KEY (Studente_ID) REFERENCES Studente (ID),
FOREIGN KEY (Corso_ID) REFERENCES Corso (ID)

Lo schema logico rappresenta le entità e le relazioni tra di esse. Ogni entità è rappresentata da una tabella, con un campo ID come chiave primaria. Le relazioni tra le entità sono rappresentate da chiavi esterne nelle tabelle corrispondenti. Le relazioni tra le entità sono le seguenti:

Docente insegna Materia

- Questa relazione è rappresentata da una chiave esterna Materia_ID nella tabella Docente. Questa chiave esterna fa riferimento al campo ID nella tabella Materia.

Studente frequenta Corso

- Questa relazione è rappresentata da una chiave esterna Corso_ID nella tabella Studente. Questa chiave esterna fa riferimento al campo ID nella tabella Corso.

Iscrizione collega Studente e Corso

- Questa relazione è rappresentata da una chiave esterna Studente_ID nella tabella Iscrizione e da una chiave esterna Corso_ID nella tabella Iscrizione. Queste chiavi esterne fanno riferimento ai campi ID nelle tabelle Studente e Corso rispettivamente.

Progettazione Fisica:

```
CREATE TABLE Docente (
   ID INT PRIMARY KEY,
   Nome VARCHAR(50),
   Cognome VARCHAR(50),
   Email VARCHAR(100),
   MateriaInsegnata VARCHAR(100)
);
CREATE TABLE Studente (
   ID INT PRIMARY KEY,
   Nome VARCHAR(50),
   Cognome VARCHAR(50),
   DataNascita DATE,
   AnnoFrequentazione INT
);
CREATE TABLE Corso (
   ID INT PRIMARY KEY,
   Nome VARCHAR(100),
   AnnoAccademico VARCHAR(20),
```

```
    Docente_ID INT,
    FOREIGN KEY (Docente_ID) REFERENCES Docente(ID)
);
CREATE TABLE Materia (
    ID INT PRIMARY KEY,
    Nome VARCHAR(100)
);
CREATE TABLE Iscrizione (
    ID INT PRIMARY KEY,
    DataIscrizione DATE,
    Studente_ID INT,
    Corso_ID INT,
    FOREIGN KEY (Studente_ID) REFERENCES Studente(ID),
    FOREIGN KEY (Corso_ID) REFERENCES Corso(ID)
);
```

Creazione di una vista per visualizzare i corsi a cui uno studente è iscritto:

```
CREATE VIEW VistaIscrizioni AS
SELECT s.Nome AS Studente, s.Cognome AS CognomeStudente, c.Nome AS Corso, c.AnnoAccademico
FROM Iscrizione i
JOIN Studente s ON i.Studente_ID = s.ID
JOIN Corso c ON i.Corso_ID = c.ID;
```

Interrogazioni:

---

1. Mostra tutti i docenti e le materie che insegnano.

```
SELECT Nome, Cognome, MateriaInsegnata FROM Docente;
```

2. Mostra gli studenti iscritti a un corso specifico.

```
SELECT s.Nome AS Studente, s.Cognome AS CognomeStudente FROM Iscrizione i JOIN Studente s ON i.Studente_ID = s.ID WHERE i.Corso_ID = [ID_Corso];
```

3. Mostra i corsi e il numero di studenti iscritti a ciascuno.

SELECT c.Nome AS Corso, c.AnnoAccademico, COUNT(i.ID) AS NumeroStudentiIscritti FROM Corso c LEFT JOIN Iscrizione i ON c.ID = i.Corso_ID GROUP BY c.ID;

4. Mostra le materie insegnate da un docente specifico.

SELECT d.Nome AS Docente, d.Cognome AS CognomeDocente, d.MateriaInsegnata, m.Nome AS Materia FROM Docente d JOIN Corso c ON d.ID = c.Docente_ID JOIN Materia m ON c.Materia_ID = m.ID WHERE d.ID = [ID_Docente];

5. Mostra gli studenti che si sono iscritti a un corso in un intervallo di date.

SELECT s.Nome AS Studente, s.Cognome AS CognomeStudente, c.Nome AS Corso, i.DataIscrizione FROM Iscrizione i JOIN Studente s ON i.Studente_ID = s.ID JOIN Corso c ON i.Corso_ID = c.ID WHERE i.DataIscrizione BETWEEN [Data_Inizio] AND [Data_Fine];

6. Mostra il numero di corsi a cui è iscritto uno specifico studente.

SELECT s.Nome AS Studente, s.Cognome AS CognomeStudente, COUNT(i.ID) AS NumeroCorsiIscritto FROM Iscrizione i JOIN Studente s ON i.Studente_ID = s.ID WHERE s.ID = [ID_Studente];

7. Mostra tutti i corsi e il rispettivo docente.

SELECT c.Nome AS Corso, c.AnnoAccademico, d.Nome AS Docente, d.Cognome AS CognomeDocente FROM Corso c JOIN Docente d ON c.Docente_ID = d.ID;

8. Mostra gli studenti che hanno iscrizioni in più di un corso.

SELECT s.Nome AS Studente, s.Cognome AS CognomeStudente FROM Iscrizione i JOIN Studente s ON i.Studente_ID = s.ID GROUP BY s.ID HAVING COUNT(DISTINCT i.Corso_ID) > 1;

9. Mostra i corsi e il numero medio di studenti iscritti.

SELECT c.Nome AS Corso, c.AnnoAccademico, AVG(COUNT(i.ID)) AS MediaStudentiIscritti FROM Corso c LEFT JOIN Iscrizione i ON c.ID = i.Corso_ID GROUP BY c.ID;

10. Mostra il docente che insegna più materie.

SELECT d.Nome AS Docente, d.Cognome AS CognomeDocente, d.MateriaInsegnata, COUNT(m.ID) AS NumeroMaterie FROM Docente d JOIN Corso c ON d.ID = c.Docente_ID JOIN Materia m ON c.Materia_ID = m.ID GROUP BY d.ID ORDER BY NumeroMaterie DESC LIMIT 1;

Pierluigi Venneri