Esercizio FastAPI Livello Zero

La Biblioteca (Relazione 1:N)

Corso FastAPI per Principianti

Descrizione

Questo è l'esercizio più semplice possibile per capire le relazioni. Creeremo una piccola API per una biblioteca che gestisce solo due entità: **Autori** e **Libri**.

1 Il Database

1. Author (L'Autore)

Tabella semplice senza dipendenze.

- `id`: Integer, Primary Key.
- `name`: String (es. "J.K. Rowling").
- **Relazione**: Un autore ha scritto molti libri.

2. Book (Il Libro)

Tabella che "appartiene" a un autore.

- `id`: Integer, Primary Key.
- `title`: String (es. "Harry Potter").
- `pages`: Integer (es. 300).
- **ForeignKey**: Deve avere una colonna `author_id` che punta all'ID dell'autore.

La Regola d'Oro della ForeignKey

La colonna `ForeignKey` va SEMPRE messa nella tabella del "Molti" (i Libri).

2 Requisiti del Codice

2.1 1. Models (`models.py`)

Copia e completa questo codice. Devi solo aggiungere le relazioni.

```python
class Author(Base):
    __tablename__ = "authors"
    id = Column(Integer, primary_key=True, index=True)
    name = Column(String, index=True)

    # TODO: Aggiungi qui la relationship verso i libri
```

```
    # books = relationship("Book", back_populates="writer")

class Book(Base):
    __tablename__ = "books"
    id = Column(Integer, primary_key=True, index=True)
    title = Column(String, index=True)
    pages = Column(Integer)

    # TODO: Definisci la ForeignKey qui sotto
    # author_id = Column(Integer, ForeignKey("authors.id"))

    # TODO: Aggiungi qui la relationship verso l'autore
    # writer = relationship("Author", back_populates="books")
```

2.2 2. Schemas (`schemas.py`)

Crea gli schemi Pydantic. Ricordati la configurazione magica!

```
class BookBase(BaseModel):
    title: str
    pages: int

class BookCreate(BookBase):
    # Quando creo un libro, devo dire di chi e'?
    # SI: passiamo author_id qui.
    author_id: int

class Book(BookBase):
    id: int
    author_id: int
    class Config:
        from_attributes = True # Importante!

class AuthorBase(BaseModel):
    name: str

class AuthorCreate(AuthorBase):
    pass

class Author(AuthorBase):
    id: int
    # Opzionale: Se vuoi vedere i libri dell'autore
    # books: list[Book] = []
    class Config:
        from_attributes = True
```

2.3 3. Main (`main.py`)

Crea solo due rotte per testare se tutto funziona:

- `POST /authors/`: Crea un autore (es. "Tolkien").
- `POST /books/`: Crea un libro collegato a Tolkien (usando il suo ID).

3 Esempio di Test

Se hai fatto tutto correttamente, inviando questi dati:

1. **Crea Autore:**

```
{ "name": "Isaac Asimov" }
```

Il server risponde con ID: 1

2. **Crea Libro (collegato all'ID 1):**

```
{
  "title": "Io, Robot",
  "pages": 250,
  "author_id": 1
}
```