

can you explain this database schema? i'd never seen a REFERENCE keyword before, does it exist in MYSQL? thought it was enough to establish primary and foreign keys?

```
CREATE TABLE Videos (  
  
    videoId INTEGER PRIMARY KEY,  
  
    title VARCHAR(100),  
  
    description TEXT,  
  
    thumbnail BLOB,  
  
    views INTEGER,  
  
    likes INTEGER,  
  
    channelId VARCHAR(50),  
  
    FOREIGN KEY (channelId) REFERENCES Channels(channelId)  
  
);
```

```
CREATE TABLE Channels (  
  
    channelId VARCHAR(50) PRIMARY KEY,  
  
    subscribers INTEGER,  
  
    dateJoined DATE,  
  
);
```

```
CREATE TABLE Subscribers (  
  
    channelId VARCHAR(50),  
  
    subscribedTo VARCHAR(50),  
  
    FOREIGN KEY (channelId) REFERENCES Channels (channelId),  
  
    FOREIGN KEY (subscribedTo) REFERENCES Channels (channelId),  
  
    PRIMARY KEY (channelId, subscribedTo)  
  
);
```

SQL Expert

The schema you've provided defines three tables typically used in a platform that hosts videos, such as a simplified version of YouTube. Each table serves its purpose within the database to store information about videos, channels, and subscribers. Let's break down each table and address the `REFERENCES` keyword:

Videos Table

- `videoId`: The primary key for the `Videos` table. Each video has a unique identifier.
- `title`: The title of the video, stored as a string up to 100 characters.
- `description`: A text field that can store a longer description of the video.
- `thumbnail`: A BLOB (Binary Large Object) used to store binary data, such as an image for the video's thumbnail.
- `views`: An integer count of how many times the video has been viewed.
- `likes`: An integer count of how many likes the video has received.
- `channelId`: This is a foreign key that references the `channelId` in the `Channels` table. It denotes which channel the video belongs to.

The `FOREIGN KEY (channelId) REFERENCES Channels(channelId)` is a constraint that establishes a relationship between the `Videos` table and the `Channels` table. It means that every `channelId` in the `Videos` table must also exist in the `Channels` table, thus ensuring referential integrity.

Channels Table

- `channelId`: The primary key for the `Channels` table, uniquely identifying each channel.
- `subscribers`: An integer representing the number of subscribers to the channel.
- `dateJoined`: The date when the channel was created or joined the platform.

This table does not have any foreign keys; it is referenced by other tables instead.

Subscribers Table

- `channelId`: A foreign key that references the `channelId` in the `Channels` table. It could represent the channel that the subscriber belongs to or follows.
- `subscribedTo`: Another foreign key that also references the `channelId` in the `Channels` table, representing the channel that a user has subscribed to.

The `PRIMARY KEY (channelId, subscribedTo)` ensures that each combination of `channelId` and `subscribedTo` is unique, which effectively prevents a subscriber from being recorded as subscribing to the same channel more than once.

The `FOREIGN KEY (channelId) REFERENCES Channels (channelId)` and `FOREIGN KEY (subscribedTo) REFERENCES Channels (channelId)` are foreign key constraints that link the `Subscribers` table back to the `Channels` table, ensuring that both the subscriber and the subscribed-to channels exist in the `Channels` table.

About the **REFERENCES** Keyword in MySQL:

In MySQL, the `REFERENCES` keyword is indeed used to define foreign key relationships between tables. It is part of the SQL standard and is used within the `FOREIGN KEY` constraint definition to specify the table and column that the foreign key is pointing to. This enforces referential integrity within the database.

While it's true that establishing primary and foreign keys is essential, the use of `REFERENCES` is specifically to define the exact relationship – which column in the foreign table is linked to the foreign key. In relational database design, this is crucial for maintaining the integrity and relationships between different entities (like videos and channels in your example).

Even though in MySQL the primary and foreign keys are often enough to understand the relationships, the `REFERENCES` part explicitly states which table and column the foreign key relates to, making the schema clearer and enforcing the relational constraints.