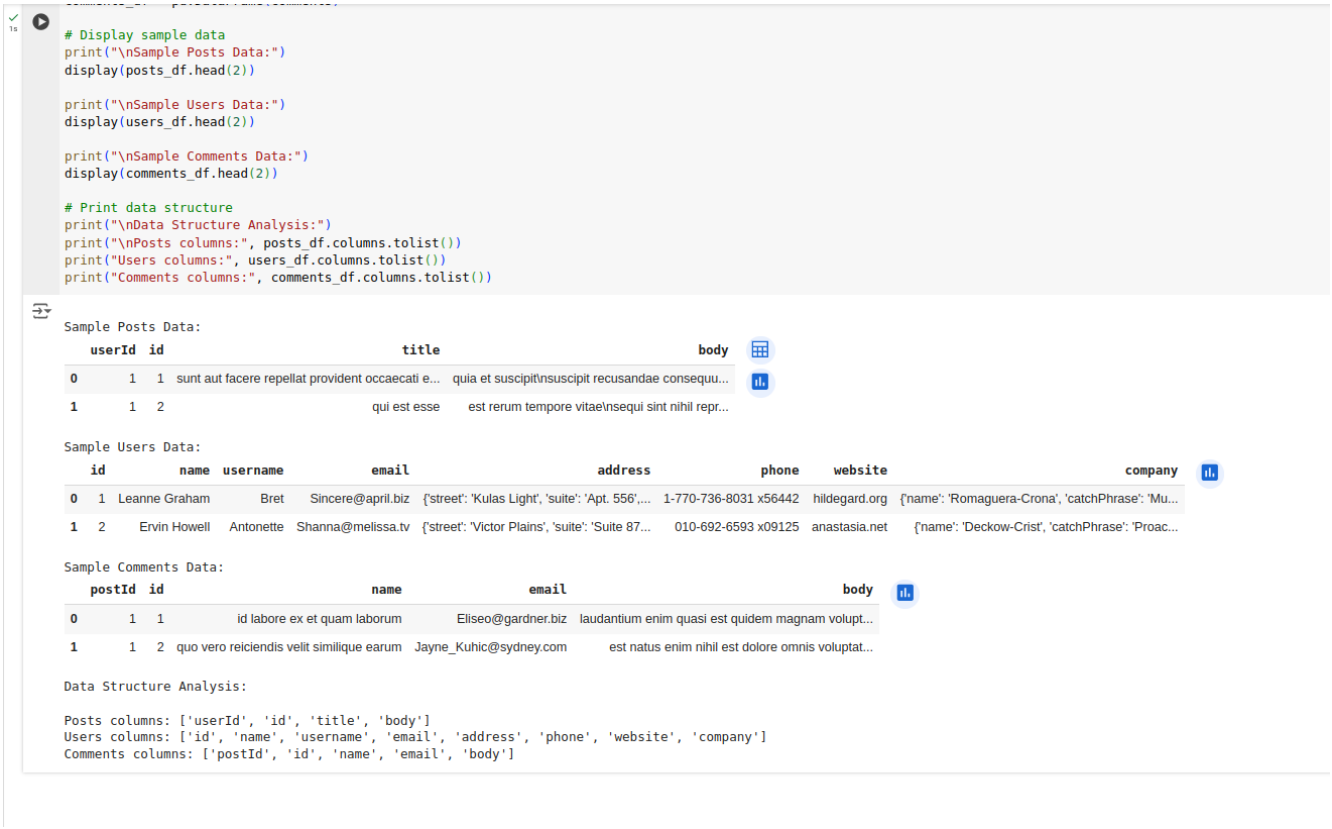
I will be using the JSONPlaceholder API, which is a free fake API for testing and prototyping.

Using Google Colab, pandas and th e IPython.display library, I ran it to retrieve some JSON data, as shown below, which includes users, posts, and comments.



```
# Display sample data
print("\nSample Posts Data:")
display(posts_df.head(2))

print("\nSample Users Data:")
display(users_df.head(2))

print("\nSample Comments Data:")
display(comments_df.head(2))

# Print data structure
print("\nData Structure Analysis:")
print("\nPosts columns:", posts_df.columns.tolist())
print("Users columns:", users_df.columns.tolist())
print("Comments columns:", comments_df.columns.tolist())
```

```
Sample Posts Data:
```

	userId	id	title	body
0	1	1	sunt aut facere repellat provident occaecati e...	quia et suscipit\nsuscipit recusandae consequu...
1	1	2	qui est esse	est rerum tempore vitae\nsequi sint nihil repr...

```
Sample Users Data:
```

	id	name	username	email	address	phone	website	company
0	1	Leanne Graham	Bret	Sincere@april.biz	{'street': 'Kulas Light', 'suite': 'Apt. 556',...	1-770-736-8031 x56442	hildegard.org	{'name': 'Romaguera-Crona', 'catchPhrase': 'Mu...
1	2	Ervin Howell	Antonette	Shanna@melissa.tv	{'street': 'Victor Plains', 'suite': 'Suite 87...	010-692-6593 x09125	anastasia.net	{'name': 'Deckow-Crist', 'catchPhrase': 'Proac...

```
Sample Comments Data:
```

	postId	id	name	email	body
0	1	1	id labore ex et quam laborum	Eliseo@gardner.biz	laudantium enim quasi est quidem magnam volupt...
1	1	2	quo vero reiciendis velit similique earum	Jayne_Kuhic@sydney.com	est natus enim nihil est dolore omnis voluptat...

```
Data Structure Analysis:

Posts columns: ['userId', 'id', 'title', 'body']
Users columns: ['id', 'name', 'username', 'email', 'address', 'phone', 'website', 'company']
Comments columns: ['postId', 'id', 'name', 'email', 'body']
```

Endpoints used:

- GET /posts: Returns an array of blog posts, each with userId, id, title, and body.
- GET /users: Returns a list of users, each with id, name, username, email, address, phone, website, and company.
- GET /comments: Returns comments data, each associated with a post by postId.

How some of the Json looks like below is the Users (from /users**):**


```
[
  {
    "id": 1,
    "name": "Leanne Graham",
    "username": "Bret",
    "email": "Sincere@april.biz",
    "address": {
      "street": "Kulas Light",
      "suite": "Apt. 556",
      "city": "Gwenborough",
      "zipcode": "92998-3874",
      "geo": {"lat": "-37.3159", "lng": "81.1496"}
    },
    "phone": "1-770-736-8031 x56442",
    "website": "hildegard.org",
    "company": {"name": "Romaguera-Crona", "catchPhrase": "Multi-layered client-s
  },
  ...
]
```


Posts (from /posts**)**:


```
[
  {
    "userId": 1,
    "id": 1,
    "title": "sunt aut facere repellat provident occaecati excepturi optio reprehe
    "body": "quia et suscipit..."
  },
  ...
]
```

From the data, we can identify logical entities:

- **Users**: People who create posts.
- **Posts**: Blog posts created by users.
- **Comments**: Comments left on posts.

Relationships:

- **User to Posts:** A user can have many posts (One-to-Many).
- **Post to Comments:** A post can have many comments (One-to-Many).

This suggests the following tables:

Users Table

- **Columns:**
  - user_id (INT, PK)
  - name (VARCHAR)
  - username (VARCHAR)
  - email (VARCHAR)
  - phone (VARCHAR)
  - website (VARCHAR)

Posts Table

- **Columns:**
  - post_id (INT, PK)
  - user_id (INT, FK referencing users.user_id)
  - title (VARCHAR)
  - body (TEXT)

  Relationship where Each post references a user_id to identify its author.

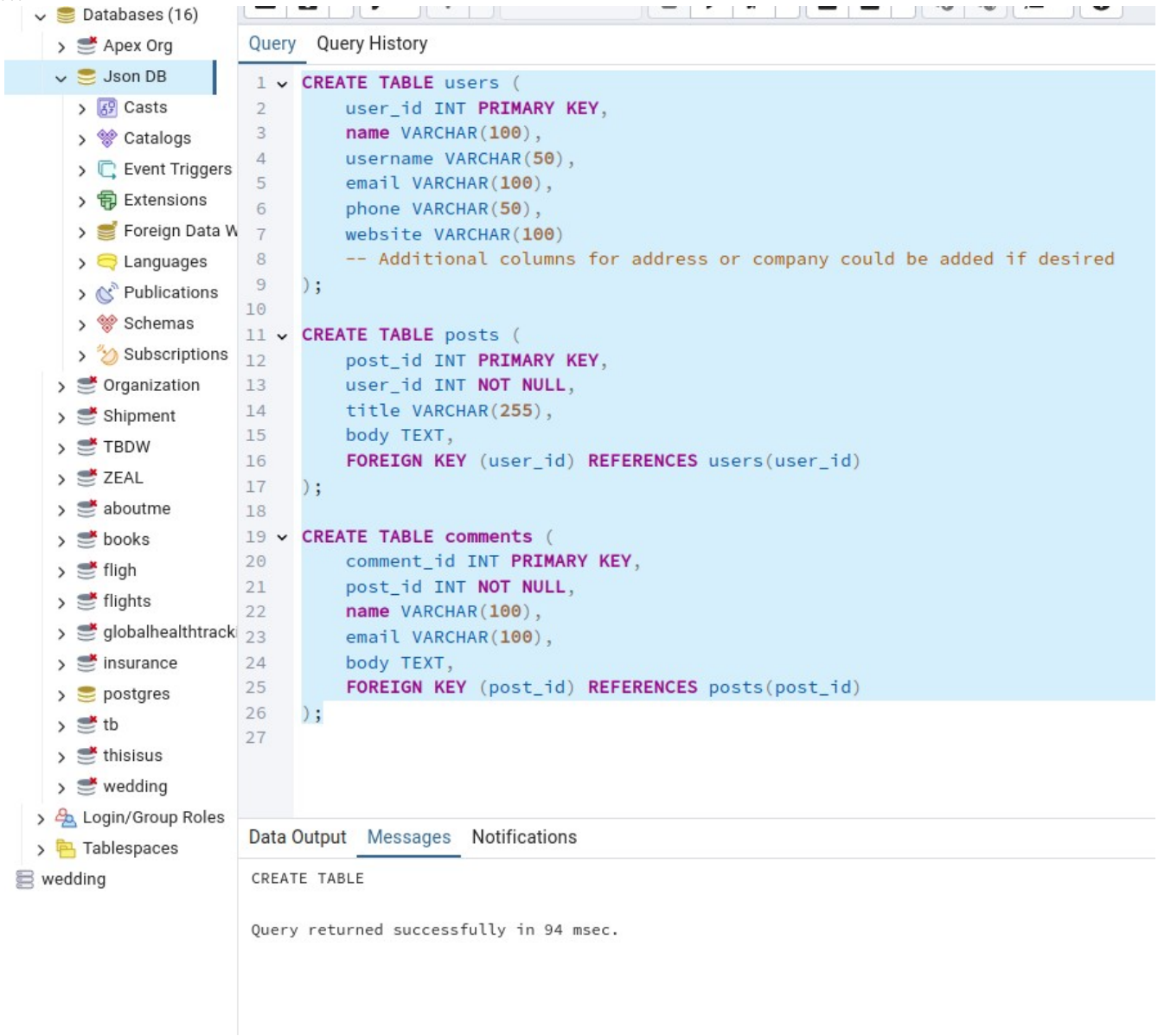
Comments Table

- **Columns:**
  - comment_id (INT, PK)
  - post_id (INT, FK referencing posts.post_id)
  - name (VARCHAR)
  - email (VARCHAR)
  - body (TEXT)

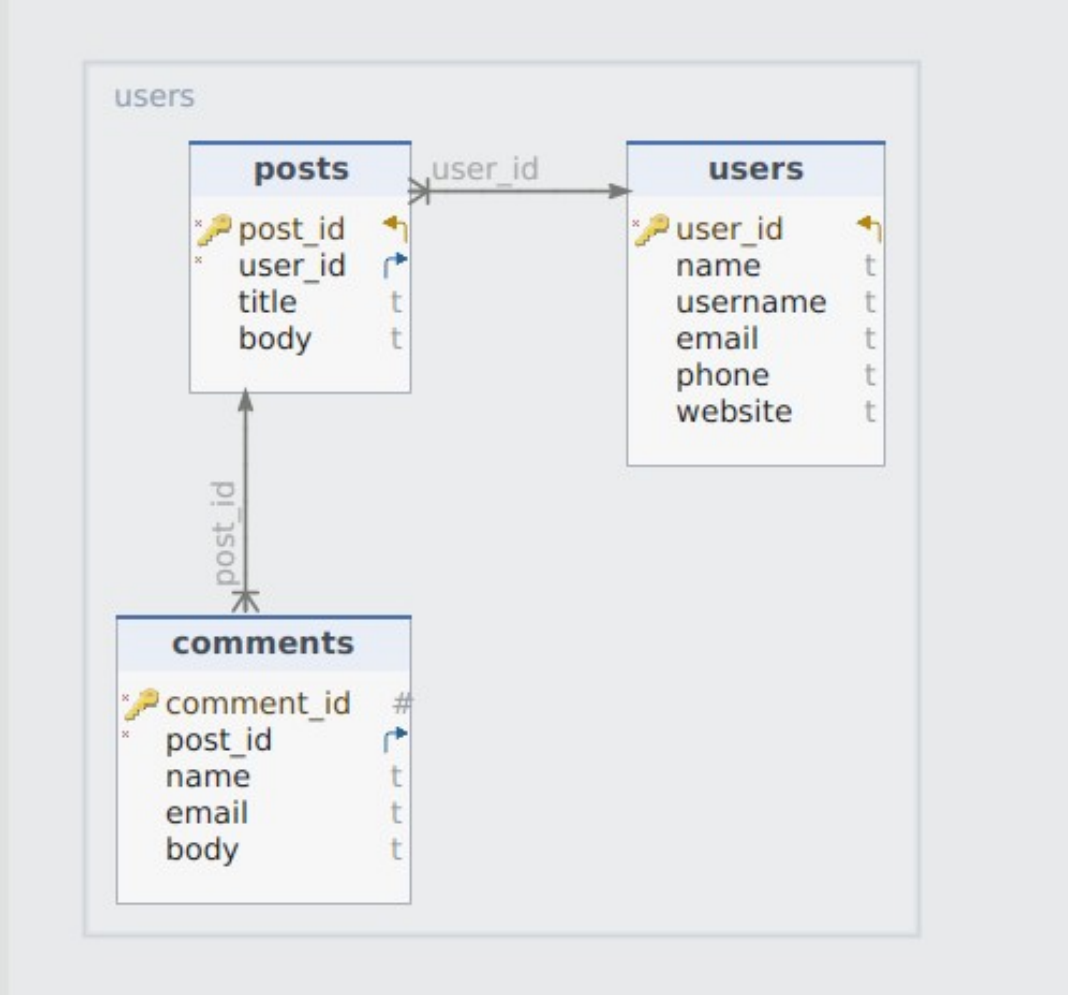
  Relationship where Each comment references a post_id to indicate which post it belongs to.

Created the tables in PG admin

....



ER Diagram in DB schema



Use Case: Social Media Engagement Analysis for Content Marketing

An organization (let's say a digital marketing agency) could benefit from this data in the following ways:

Content Performance Analysis:

- Track which types of posts generate the most comments
- Analyze engagement patterns based on post length, title structure, and content
- Identify which users (authors) generate the most engagement

User Behavior Analysis:

- Study comment patterns and sentiment
- Analyze user posting frequency and timing
- Map geographical distribution of engagement using user address data

Data-Enrichment for Analytics:

- By adding outside data to a company's own information, such as user details or popular topics from other sources, analysts could find new patterns. This might show which types of posts are popular in different places, or what traits make some posts more engaging than others.
- Analyze connection patterns via comments.

Note: At first, I tried using the Nasdaq Data Link (formerly Quandl) API but ran into issues and got stuck. I spent a lot of time on it and realized it's not as free or public as it seems. After hours of effort, I decided to abandon it.

Next, I tried the IMDB API, but their approval process takes days—you have to apply and wait. So finally, I decided to settle on JSONPlaceholder.