DBMS DINOS - Our Team

Vandit Patel	U22CS065
Vishwa Lathigara	U22CS036
Misbah Shaikh	U22CS043
Ankit	U22CS033

DBMSDINOS

Instagram Database Management

"Welcome to the world of digital connectivity and visual storytelling, where we unveil the intricate design behind one of the world's most popular social media platforms, Instagram. In this project, we delve deep into the architecture that fuels this global sensation, crafting an Entity-Relationship model that elegantly captures the essence of user experiences, content creation, and the interconnected web of social interactions. Join us on a journey as we dissect Instagram's underlying database structure, revealing the magic that powers your everyday scroll, double-tap, and hashtag adventure."

DBMSDINOS

ER MODEL

Entities and their attribute:

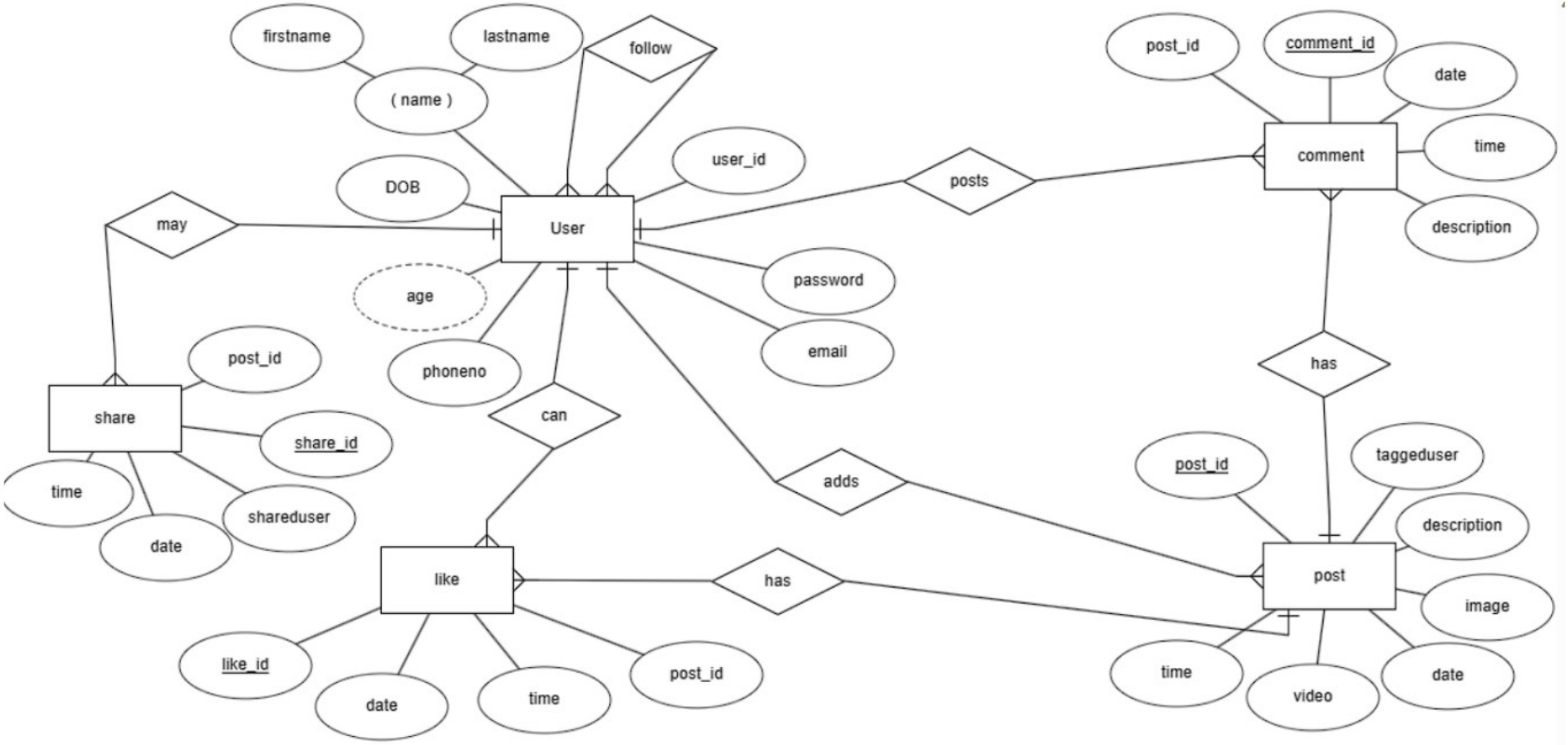
- User Entity
  - User ID (Primary Key)
  - First Name
  - Last Name
  - Date of Birth (DOB)
  - Age
  - Password
  - Phoneno
  - Email
- Post Entity
  - Post ID (Primary Key)
  - Tagged User
  - Description
  - Date
  - Video
  - Time
  - Image
- Comment Entity
  - Comment ID (Primary Key)
  - Post ID (Foreign Key)
  - Date
  - Time
  - Description
- Like Entity
  - Like ID (Primary Key)
  - Post ID (Foreign Key)
  - Date
  - Time
- Share Entity
  - Share ID (Primary Key)
  - Post ID (Foreign Key)
  - Time
  - Date
  - Shared User

DBMSDINOS

Relationships:

1. User Follow User
   - Relationship between two User entities, representing the "follow" action.
   - This relationship is many-to-many since a user can follow multiple users, and a user can have multiple followers.
2. User Posts Comment
   - Relationship between User and Comment entities, representing when a user posts a comment on a post.
   - This is a one-to-many relationship since one user can post multiple comments, and one comment is posted by one user.
3. Post Has Comments
   - Relationship between Post and Comment entities, representing the comments posted on a post.
   - This is a one-to-many relationship since one post can have multiple comments, but one comment belongs to one post.
4. Post Has Likes
   - Relationship between Post and Like entities, representing the likes received by a post.
   - This is a one-to-many relationship since one post can have multiple likes, but one like belongs to one post.
5. User Adds Post
   - Relationship between User and Post entities, representing when a user creates and adds a post.
   - This is a one-to-many relationship since one user can create and add multiple posts, but one post is created by one user.
6. User Can Like
   - Relationship between User and Like entities, representing when a user likes a post.
   - This is a one-to-many relationship since a user can like multiple posts.
7. User May Share
   - Relationship between User and Share entities, representing when a user shares a post.
   - This is a one-to-many relationship since one user can share multiple posts, but one share action belongs to one user.

DBMSDINOS


firstname
lastname
follow
(name)
user_id
DOB
may
User
posts
comment
post_id
comment_id
date
time
description
age
password
email
phoneno
post_id
share
share_id
time
date
shareduser
can
adds
has
post_id
taggeduser
description
post
image
like
has
like_id
date
time
post_id
time
video
date

SQL

&PLSQL CODES:

ORACLE®

DBMSDINOS

CREATION OF TABLES QUERY:

```
CREATE TABLE UserRegistration (
 UserID NUMBER PRIMARY KEY,
FirstName VARCHAR2(50),
 LastName VARCHAR2(50),
 DOB DATE,
 Age NUMBER,
Password VARCHAR2(100),
 Email VARCHAR2(100),
 PhoneNo VARCHAR2(15),
 DateOfRegistration DATE
);

CREATE TABLE Followers (
 FollowerID NUMBER,
 FollowingID NUMBER,
 PRIMARY KEY (FollowerID, FollowingID)
 );

CREATE TABLE Post (
 PostID NUMBER PRIMARY KEY,
UserID NUMBER,
Description VARCHAR2(255),
 Video VARCHAR2(255),
 Image VARCHAR2(255),
 NoOfLikes NUMBER,
DateOfPost Date,
 NoOfComments NUMBER,
 NoOfShares NUMBER
);

CREATE TABLE "Comment" (
CommentID NUMBER PRIMARY KEY,
 DateOfComment Date,
 PostID NUMBER,
 UserID NUMBER,
 Description VARCHAR2(255)
);

CREATE TABLE "Like" (
LikeID NUMBER PRIMARY KEY,
 DateOfLike Date,
PostID NUMBER,
UserID NUMBER
);

 CREATE TABLE "Share" (
 ShareID NUMBER PRIMARY KEY,
DateOfShare Date,
 PostID NUMBER,
 UserID NUMBER
);

CREATE TABLE SharedWith (
ShareID NUMBER,
SharedUserID NUMBER,
 PRIMARY KEY (ShareID, SharedUserID)
);

CREATE TABLE TaggedUser (
 PostID NUMBER,
 TaggedUserID NUMBER,
 PRIMARY KEY (PostID, TaggedUserID)
 );
```

DBMSDINOS

DESCRIPTION OF TABLES:

1. **UserRegistration Table:**
   - Purpose: This table is used to store user registration information, including details such as the user's name, date of birth, age, email, phone number, and securely hashed password.
   - Use: It's essential for user account management and authentication. It allows users to create accounts and log in securely.
2. **Followers Table:**
   - Purpose: The Followers table establishes a many-to-many relationship between users. It stores information about who follows whom.
   - Use: This table allows users to follow and be followed by other users. It's crucial for managing the social aspect of the platform, enabling users to build a network of followers.
3. **Post Table:**
   - Purpose: The Post table is used to store information about user-generated posts, including text content, media (images or videos), and statistics about post engagement (likes, comments, shares).
   - Use: This table is at the core of the platform, allowing users to create, share, and engage with posts. It provides a platform for sharing content with others.
4. **Comment Table:**
   - Purpose: The Comment table stores individual comments made on posts, associating them with the posts and the users who made the comments.
   - Use: This table enables users to interact with posts by leaving comments. It's essential for fostering discussions and conversations.
5. **Like Table:**
   - Purpose: The Like table records individual likes on posts, linking them to the posts and the users who liked them.
   - Use: It allows users to express their appreciation for posts. Likes are a fundamental form of engagement on social media.
6. **Share Table:**
   - Purpose: The Share table keeps track of posts that are shared by users. It associates shared posts with the users who shared them and the recipients.
   - Use: Sharing posts is a common feature on social media platforms. This table enables users to share interesting content with their followers and friends.
7. **SharedWith Table:**
   - Purpose: The SharedWith table manages shared posts with multiple recipients. It creates a many-to-many relationship between shared posts and the users they are shared with.
   - Use: In cases where a post is shared with multiple users or groups, this table helps maintain those relationships.
8. **TaggedUser Table:**
   - Purpose: The TaggedUser table tracks which users are tagged in posts, establishing a many-to-many relationship between posts and tagged users.
   - Use: Tagging users in posts is a common feature for notifying and involving specific users in a post's content. This table maintains those connections.

DBMSDINOS

NORMALISATION

UserRegistration:

1NF: Each attribute is atomic, and there are no repeating groups.
2NF: The table has a primary key (UserID) and no non-key attributes are dependent on a part of the primary key.
3NF: There are no transitive dependencies, as all non-key attributes are directly dependent on the primary key (UserID).

Followers:

1NF: Each attribute is atomic, and there are no repeating groups.
2NF: The table has a composite primary key (FollowerID, FollowingID), and no non-key attributes are dependent on a part of the primar
3NF: There are no transitive dependencies, as there are no non-key attributes.

Post:

1NF: Each attribute is atomic, and there are no repeating groups.
2NF: The table has a primary key (PostID) and no non-key attributes are dependent on a part of the primary key.
3NF: There are no transitive dependencies, as all non-key attributes are directly dependent on the primary key (PostID).

Comment:

1NF: Each attribute is atomic, and there are no repeating groups.
2NF: The table has a primary key (CommentID) and no non-key attributes are dependent on a part of the primary key.
3NF: There are no transitive dependencies, as all non-key attributes are directly dependent on the primary key (CommentID).

Like:

1NF: Each attribute is atomic, and there are no repeating groups.
2NF: The table has a primary key (LikeID) and no non-key attributes are dependent on a part of the primary key.
3NF: There are no transitive dependencies, as all non-key attributes are directly dependent on the primary key (LikeID).

Share:

1NF: Each attribute is atomic, and there are no repeating groups.
2NF: The table has a primary key (ShareID) and no non-key attributes are dependent on a part of the primary key.
3NF: There are no transitive dependencies, as all non-key attributes are directly dependent on the primary key (ShareID).

SharedWith:

1NF: Each attribute is atomic, and there are no repeating groups.
2NF: The table has a composite primary key (ShareID, SharedUserID),
and no non-key attributes are dependent on a part of the primary key.
3NF: There are no transitive dependencies, as there are no non-key attributes.

TaggedUser:

1NF: Each attribute is atomic, and there are no repeating groups.
2NF: The table has a composite primary key (PostID, TaggedUserID),
and no non-key attributes are dependent on a part of the primary key.
3NF: There are no transitive dependencies, as there are no non-key attributes.

In conclusion, all the provided tables meet the criteria for 1NF, 2NF, and 3NF.
They have atomic attributes, no partial dependencies, and no transitive dependencies.

TRIGGERS

```
import cx_Oracle
connection = cx_Oracle.connect("system", "tiger", "localhost:1521/XEPDB1")
cursor = connection.cursor()
like_trigger_sql = """
CREATE OR REPLACE TRIGGER like_post_trigger
AFTER INSERT ON "Like" FOR EACH ROW
BEGIN
 UPDATE Post
 SET NoOfLikes = NoOfLikes + 1
 WHERE PostID = :new.PostID;
END;
/
"""
comment_trigger_sql = """
CREATE OR REPLACE TRIGGER comment_post_trigger
AFTER INSERT ON "Comment" FOR EACH ROW
BEGIN
 UPDATE Post
 SET NoOfComments = NoOfComments + 1
 WHERE PostID = :new.PostID;
END;
/
"""
share_trigger_sql = """
CREATE OR REPLACE TRIGGER share_post_trigger
AFTER INSERT ON "Share" FOR EACH ROW
BEGIN
 DECLARE
 shared_users_count NUMBER;
 BEGIN
 SELECT COUNT(*) INTO shared_users_count
 FROM SharedWith
 WHERE ShareID = :new.ShareID;

 UPDATE Post
 SET NoOfShares = NoOfShares + shared_users_count
 WHERE PostID = :new.PostID;
 END;
END;
/
"""
trigger_like_deleted = """
CREATE OR REPLACE TRIGGER update_like_count
AFTER DELETE ON "Like"
FOR EACH ROW
BEGIN
 UPDATE Post
 SET NoOfLikes = NoOfLikes - 1
 WHERE PostID = :old.PostID;
END;
/
"""
trigger_comment_deleted = """
CREATE OR REPLACE TRIGGER update_comment_count
AFTER DELETE ON "Comment"
FOR EACH ROW
BEGIN
 UPDATE Post
 SET NoOfComments = NoOfComments - 1
 WHERE PostID = :old.PostID;
END;
/
"""
cursor.execute(trigger_like_deleted)
cursor.execute(trigger_comment_deleted)
cursor.execute(share_trigger_sql)
cursor.execute(like_trigger_sql)
cursor.execute(comment_trigger_sql)
connection.commit()
cursor.close()
connection.close()
print("Triggers created successfully.")
```

DBMSDINCS

```
<?xml version="1.0" encoding="UTF-8" ?>
<rss version="2.0" xmlns:dc="http://purl.org/dc/elements/1.1/" xmlns:sy="http://purl.org/rss/1.0/modules/syndication/" xmlns:slash="http://purl.org/rss/1.0/modules/slash/" xmlns:wfw="http://wellformedweb.org/CommentAPI/" xmlns:media="http://search.yahoo.com/mrss/" xmlns:atom10="http://www.w3.org/2005/Atom" xmlns:opensearch="http://a9.com/-/spec/opensearchrss/1.0/" xmlns:geo="http://www.w3.org/2003/01/geo/wgs84_pos#" xmlns:georss="http://www.georss.org/georss" xmlns:fb="http://www.facebook.com/2008/fbml
```

```
PS C:\Users\ANKIT\Desktop\New folder> & "C:/Program Files/Python312/python.exe" "c:/Users/ANKIT/Desktop/New folder/2.py"

Menu:
1. User Registration
2. Follow a Person
3. Post
4. Like a Post
5. Comment on a Post
6. Share a Post
7. Delete a Post
8. Update User Information
9. Delete User Account
10. Update Comment Description
11. Delete Comment
12. Delete Like
13. Display User Information
14. Print a Table
0. Exit
Enter your choice: 14
Available Tables:
1. UserRegistration
2. Followers
3. Post
4. Comment
5. Like
6. Share
7. SharedWith
8. TaggedUser
Enter the number of the table you want to print (1-8): 1

UserRegistration Table:
+--------+----------+----------+---------------------+-----+---------------+-------+------------+---------------------+
| USERID | FIRSTNAME | LASTNAME |        DOB          | AGE |   PASSWORD    | EMAIL |  PHONENO   | DATEOFREGISTRATION |
+--------+----------+----------+---------------------+-----+---------------+-------+------------+---------------------+
|   2    |    w     |    w     | 2003-02-02 00:00:00 |  20 |   asdfghjk    |   @   | 1233211234 | 2023-11-01 00:00:00 |
|   3    |    g     |    g     | 2003-04-04 00:00:00 |  20 | qwerttyyuuut  |   @   | 1234445555 | 2023-11-01 00:00:00 |
|   1    |    a     |    a     | 2002-02-02 00:00:00 |  21 | sadasdasdsadsa |  @   | 1233211234 | 2023-11-03 00:00:00 |
+--------+----------+----------+---------------------+-----+---------------+-------+------------+---------------------+
```

Thank You!

Expressing our gratitude for your support and kindness.

DBUSDINOS