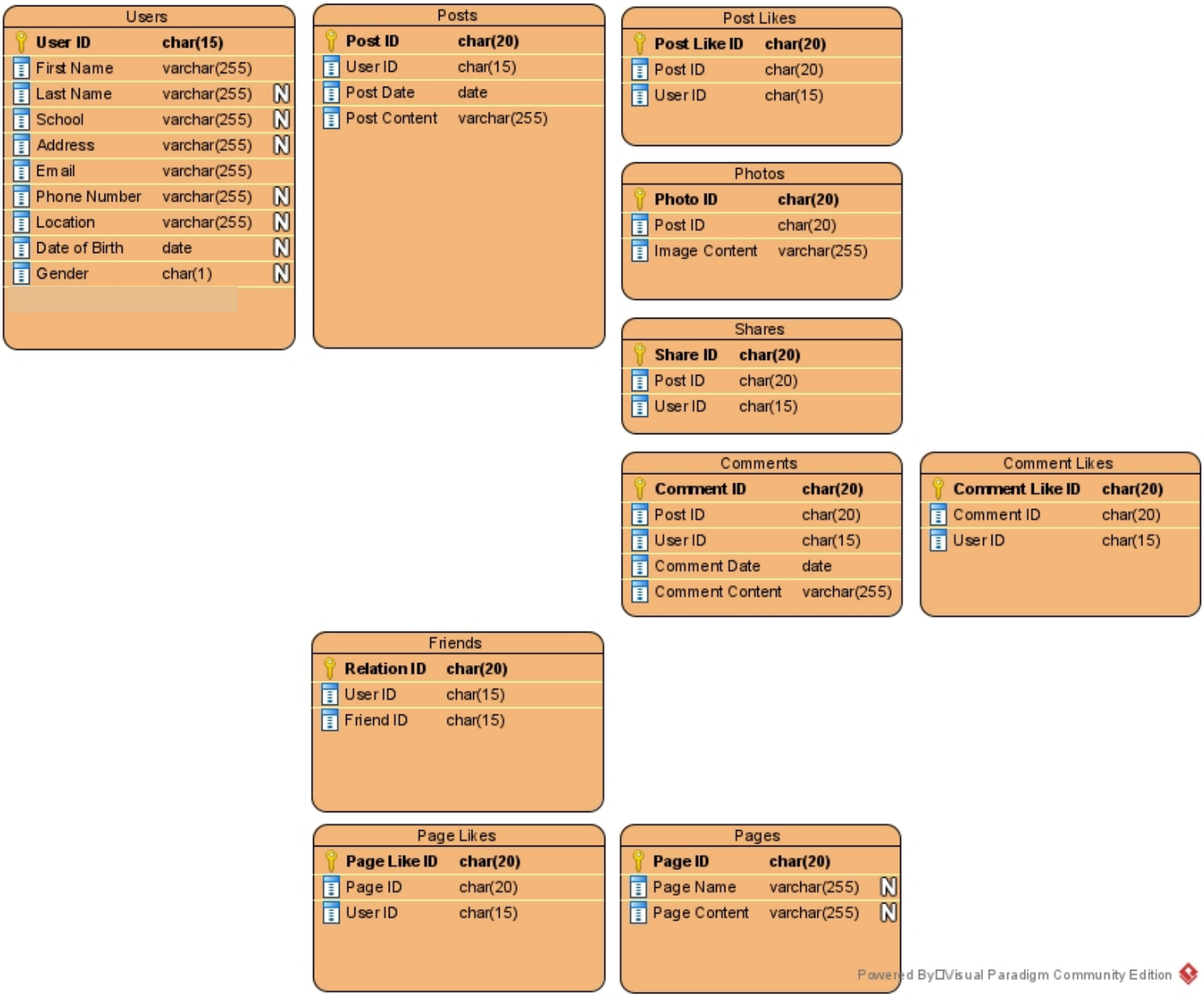
ERIKHEN

1. From the above diagram, list all of the objects including its attributes! (.pdf)



2. Determine the relation between every object, specify the master and child table! (.pdf)
   Master-Child:
   Users-Posts, Users-Friends, Users-Page Likes,
   Posts-Post Likes, Posts-Photos, Posts-Shares, Posts-Comments,
   Comment-Comment Likes, Pages-Pages Likes

3. For each object, decide its constraint and specify the reason in detail! (.pdf)
   - Users
     - User ID - char(15), butuh cukup besar untuk menampung id user yang banyak dan panjangnya selalu sama. PK table Users, tidak boleh NULL.

  - First name dan Email – varchar(255), Panjang karakter tidak pasti dan tidak boleh NULL.
  - Last name, school, address, phone number, location – varchar(255), Panjang karakter tidak pasti dan boleh NULL.
  - Date of Birth – date, bertipe tanggal dan boleh NULL.
  - Gender – char(1), Cukup untuk M (male) atau F (female) atau (N) not known/preferred not to say dan boleh NULL.

- Posts
  - Post ID - char(20), butuh cukup besar untuk menampung id yang banyak dan panjangnya selalu sama. Menjadi PK table, tidak boleh NULL.
  - User ID – char(15), FK dan tipe data menyesuaikan dengan User ID di table Users, tidak boleh NULL.
  - Post Date – date, bertipe tanggal, tidak boleh NULL.
  - Post Content – varchar(255), berisi link menuju isi dari content, tidak boleh NULL.

- Post Likes
  - Post LikeID - char(20), butuh cukup besar untuk menampung id yang banyak dan panjangnya selalu sama. Menjadi PK table, tidak boleh NULL.
  - Post ID - char(20), butuh cukup besar untuk menampung id yang banyak dan panjangnya selalu sama, tidak boleh NULL.
  - User ID – char(15), FK dan tipe data menyesuaikan dengan User ID di table Users, tidak boleh NULL.

- Photos
  - Photo ID - char(20), butuh cukup besar untuk menampung id yang banyak dan panjangnya selalu sama. Menjadi PK table, tidak boleh NULL.
  - Post ID - char(20), butuh cukup besar untuk menampung id yang banyak dan panjangnya selalu sama, tidak boleh NULL.
  - Image Content – varchar(255), berisi link menuju isi image content disimpan, tidak boleh NULL.

- Shares
  - Shares ID - char(20), butuh cukup besar untuk menampung id yang banyak dan panjangnya selalu sama. Menjadi PK tabel, tidak boleh NULL.
  - Post ID - char(20), butuh cukup besar untuk menampung id yang banyak dan panjangnya selalu sama, menjadi FK, tidak boleh NULL.
  - User ID – char(15), FK dan tipe data menyesuaikan dengan User ID di table Users, tidak boleh NULL.

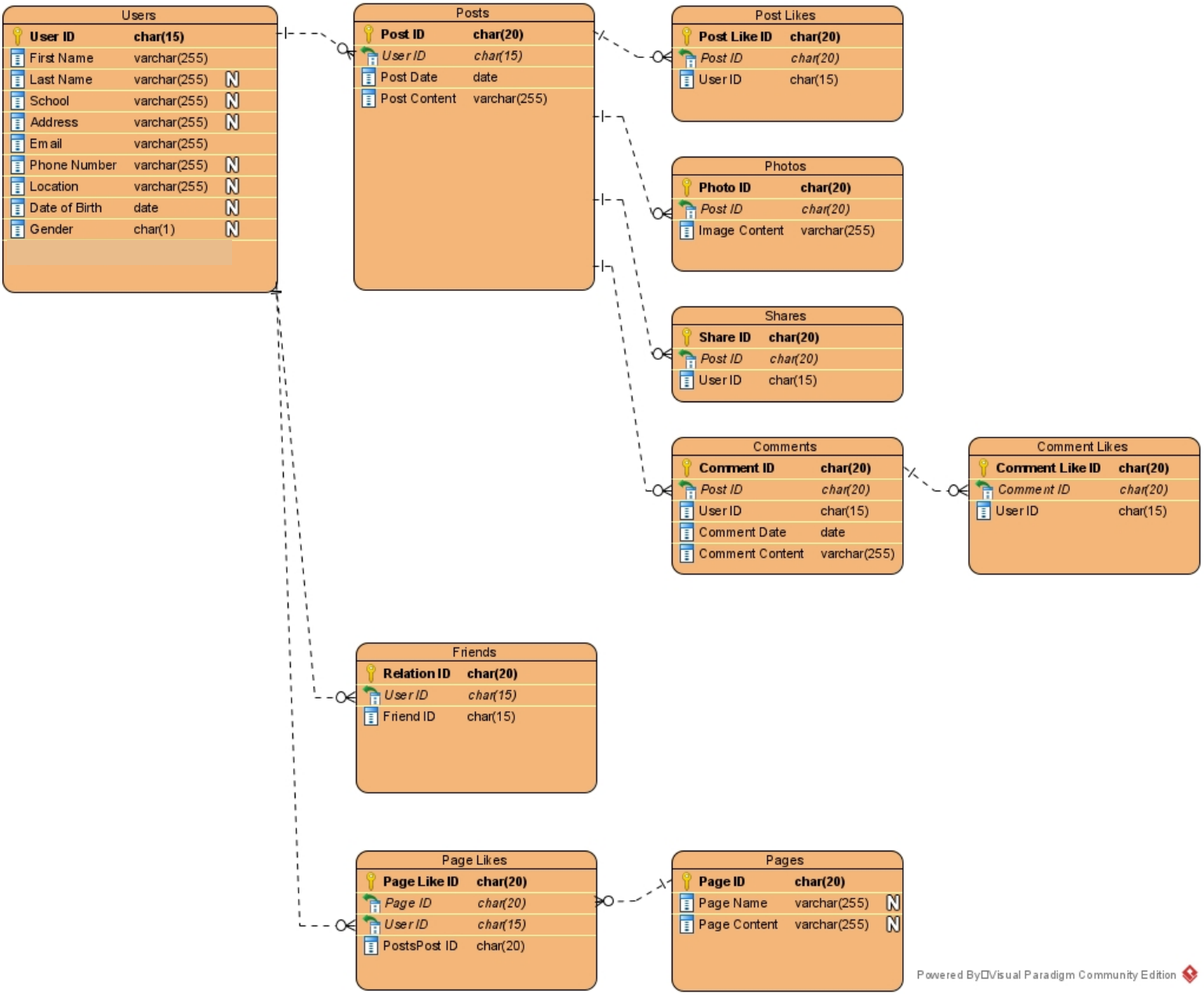
- Comments
    - Comment ID - char(20), butuh cukup besar untuk menampung id yang banyak dan panjangnya selalu sama. Menjadi PK table, tidak boleh NULL.
    - Post ID - char(20), butuh cukup besar untuk menampung id yang banyak dan panjangnya selalu sama, tidak boleh NULL.
    - User ID – char(15), FK dan tipe data menyesuaikan dengan User ID di table Users, tidak boleh NULL.
    - Comment Date – date, Berisi tanggal comment, tidak boleh NULL.
    - Comment Content – varchar(255), berisi comment berupa kumpulan karakter, tidak boleh NULL.
- Comment Likes
    - Comment Like ID - char(20), butuh cukup besar untuk menampung id yang banyak dan panjangnya selalu sama. Menjadi PK table, tidak boleh NULL.
    - Comment ID - char(20), butuh cukup besar untuk menampung id yang banyak dan panjangnya selalu sama, tidak boleh NULL.
    - User ID – char(15), FK dan tipe data menyesuaikan dengan User ID di table Users, tidak boleh NULL.

- Friends
    - Relation ID - char(20), butuh cukup besar untuk menampung id yang banyak dan panjangnya selalu sama. Menjadi PK tabel, tidak boleh NULL.
    - User ID – char(15), FK dan tipe data menyesuaikan dengan User ID di table Users, tidak boleh NULL.
    - Friend ID – char(15), dan tipe data menyesuaikan dengan User ID di table Users, tidak boleh NULL.

- Page Likes
    - Page Like ID - char(20), butuh cukup besar untuk menampung id yang banyak dan panjangnya selalu sama. Menjadi PK tabel, tidak boleh NULL.
    - Page ID - char(20), butuh cukup besar untuk menampung page id yang banyak dan panjangnya selalu sama, tidak boleh NULL.
    - User ID – char(15), FK dan tipe data menyesuaikan dengan User ID di table Users, tidak boleh NULL.

- Pages
    - Page ID - char(20), butuh cukup besar untuk menampung id yang banyak dan panjangnya selalu sama. Menjadi PK tabel, tidak boleh NULL.
    - Page name - varchar(255), berisi judul halaman yang panjangnya tidak pasti, tidak boleh NULL.
    - Page content - varchar(255), berisi link menuju konten dari page yang Panjang linknya tidak pasti, tidak boleh NULL.

4. Draw the above diagram in "ERD format" which includes the data types, primary and foreign key, and relation between objects. Please choose appropriate tools, we recommend using Visual Paradigm. (.jpeg)



1. Explain what is data integrity and how do we maintain it in SQL Server! (.pdf)

   Data integrity diperlukan untuk menjaga keakuratan dan konsistensi data di dalam table. Ada tiga jenis data integrity di SQL. Pertama, entity integrity yang memastikan bahwa setiap row di dalam table dapat dibedakan secara unik, dapat dispesifikan menggunakan Primary key atau not null. Kedua, referential integrity untuk menjaga relasi antar table, di dalam SQL, kita dapat menggunakan Foreign Key Constraint, on update, on delete. Ketiga, Domain integrity untuk menjaga isi database berdasarkan nilai, range, dan formatnya, Di dalam SQL dapat menggunakan Check atau Default.

2. Explain the difference and give example for: primary key, foreign key, and composite key! (.pdf)
   primary key merupakan kolom yang berisi data-data unik dan tidak null yang digunakan untuk menjadikan setiap row unik/beda.

   CREATE TABLE salary (salaryID INT PRIMARY KEY, salary MONEY NOT NULL)

   foreign key merupakan kolom yang berisi referensi ke dalam primary key table lain.

   CREATE TABLE users (userID CHAR(15) PRIMARY KEY, name VARCHAR(255) NOT NULL, userSalaryID REFERENCES salary(salaryID))

   composite key, mirip seperti primary key, namun menggunakan kombinasi dua kolom atau lebih di dalam table untuk menjadikan sebuah row unik.

   CREATE TABLE users (id INT, name VARCHAR(255), salary MONEY, PRIMARY KEY(id, salary))

3. Explain the following terms and give example: BEGIN TRAN, COMMIT, and ROLLBACK! (.pdf)

   BEGIN TRAN digunakan untuk memberikan checkpoint atau memulai transaction dan mengunci table sehingga data tidak dapat diubah

   BEGIN TRAN // Memulai transaksi
   UPDATE Users SET UserName = NULL

   COMMIT digunakan untuk menyimpan perubahan data dan membuka kunci table.

   BEGIN TRAN
   UPDATE Users SET UserName = NULL
   COMMIT // Nilai UserName menjadi NULL

   ROLLBACK digunakan untuk membuang perubahan data dan membuka kunci table.

   BEGIN TRAN
   UPDATE Users SET UserName = NULL
   ROLLBACK // Nilai UserName tetap seperti semula

4. Create all of the tables above according to your answer in the previous section! (.sql)

```
CREATE DATABASE aplikasi

USE aplikasi

CREATE TABLE Users(
```

```
	UserID CHAR(15),
	FirstName VARCHAR(255) NOT NULL,
	LastName VARCHAR(255),
	School VARCHAR(255),
	Adress VARCHAR(255),
	Email VARCHAR(255) NOT NULL,
	PhoneNumber VARCHAR(255),
	Location VARCHAR(255),
	DateofBirth DATE,
	Gender CHAR(1),

	CONSTRAINT Users_userID_PK PRIMARY KEY(UserID),
	CHECK (Gender LIKE 'M' OR Gender LIKE 'F' OR Gender LIKE 'N')
)

CREATE TABLE Posts(
	PostID CHAR(20),
	UserID CHAR(15),
	PostDate DATE NOT NULL,
	PostContent VARCHAR(255) NOT NULL,

	CONSTRAINT Posts_postID_PK PRIMARY KEY(PostID),
	CONSTRAINT Posts_userID_FK FOREIGN KEY(UserID) REFERENCES Users(UserID) ON
UPDATE CASCADE ON DELETE SET NULL
)

CREATE TABLE Friends(
	RelationID CHAR(20),
	UserID CHAR(15),
	FriendID CHAR(15) NOT NULL,

	CONSTRAINT Friends_relationID_PK PRIMARY KEY(relationID),
	CONSTRAINT Friends_userID_FK FOREIGN KEY(userID) REFERENCES Users(userID) ON
UPDATE CASCADE ON DELETE SET NULL
)

CREATE TABLE Pages(
	PageID CHAR(20),
	PageName VARCHAR(255) NOT NULL,
	PageContent VARCHAR(255) NOT NULL,

	CONSTRAINT Pages_PageID_PK PRIMARY KEY(pageID)
)

CREATE TABLE PageLikes(
	PageLikeID CHAR(20),
	PageID CHAR(20),
	UserID CHAR(15),
	PageName VARCHAR(255) NOT NULL,
	PageContent VARCHAR(255) NOT NULL,

	CONSTRAINT PageLikes_PageLikeID_PK PRIMARY KEY(pageLikeID),
	CONSTRAINT PageLikes_PageID_FK FOREIGN KEY(PageID) REFERENCES Pages(pageID) ON
UPDATE CASCADE ON DELETE SET NULL,
	CONSTRAINT PageLikes_UserID_FK FOREIGN KEY(UserID) REFERENCES Users(userID) ON
UPDATE CASCADE ON DELETE SET NULL
)

CREATE TABLE PostLikes(
	PostLikeID CHAR(20),
	PostID CHAR(20),
	UserID CHAR(15) NOT NULL,
```

```
	CONSTRAINT PostLikes_PostLikeID_PK PRIMARY KEY(PostLikeID),
	CONSTRAINT PostLikes_PostID_FK FOREIGN KEY(PostID) REFERENCES Posts(PostID) ON
UPDATE CASCADE ON DELETE SET NULL
)

CREATE TABLE Photos(
	PhotoID CHAR(20),
	PostID CHAR(20),
	ImageContent VARCHAR(255) NOT NULL,

	CONSTRAINT Photos_PhotoID_PK PRIMARY KEY(PhotoID),
	CONSTRAINT Photos_PostID_FK FOREIGN KEY(PostID) REFERENCES Posts(PostID) ON
UPDATE CASCADE ON DELETE SET NULL
)

CREATE TABLE Shares(
	ShareID CHAR(20),
	PostID CHAR(20),
	UserID CHAR(15) NOT NULL,

	CONSTRAINT Shares_ShareID_PK PRIMARY KEY(ShareID),
	CONSTRAINT Shares_PostID_FK FOREIGN KEY(PostID) REFERENCES Posts(PostID) ON
UPDATE CASCADE ON DELETE SET NULL
)

CREATE TABLE Comments(
	CommentID CHAR(20),
	PostID CHAR(20),
	UserID CHAR(15) NOT NULL,
	CommentDate DATE NOT NULL,
	CommentContent VARCHAR(255) NOT NULL,

	CONSTRAINT Comments_CommentID_PK PRIMARY KEY(CommentID),
	CONSTRAINT Comments_PostID_FK FOREIGN KEY(PostID) REFERENCES Posts(PostID) ON
UPDATE CASCADE ON DELETE SET NULL
)

CREATE TABLE CommentLikes(
	CommentLikeID CHAR(20),
	CommentID CHAR(20),
	UserID CHAR(15) NOT NULL,

	CONSTRAINT CommentLikes_CommentLikeID_PK PRIMARY KEY(CommentLikeID),
	CONSTRAINT CommentLikes_CommentID_FK FOREIGN KEY(CommentID) REFERENCES
Comments(CommentID) ON UPDATE CASCADE ON DELETE SET NULL
)
```