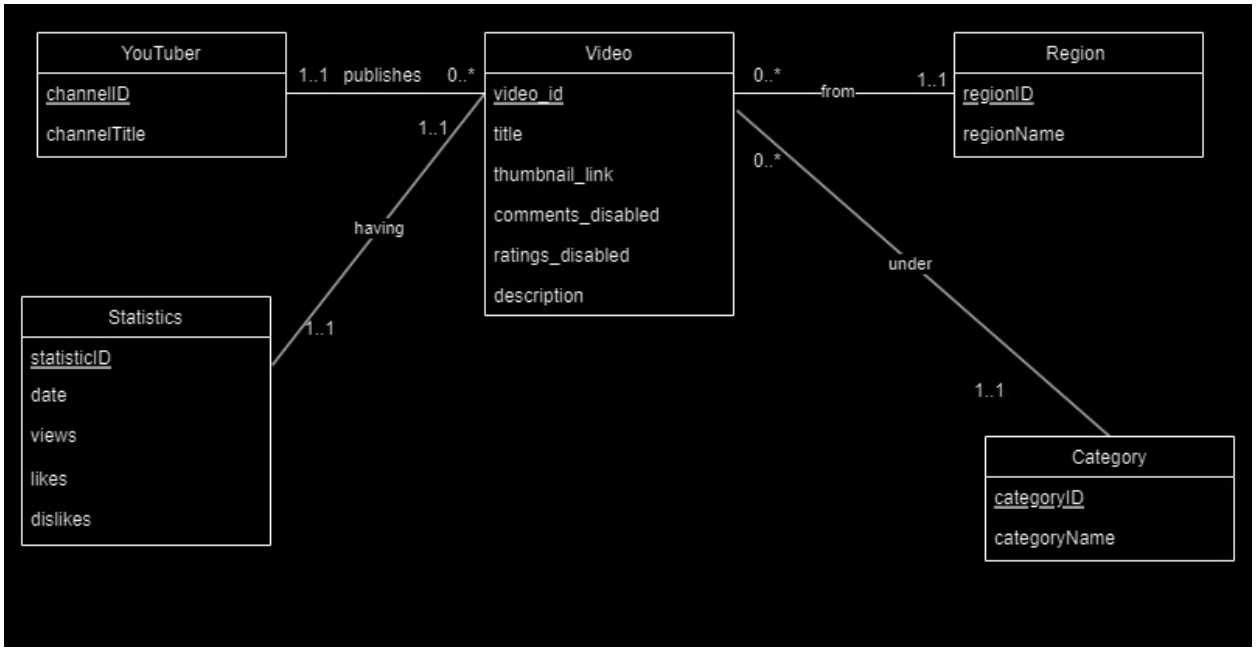
1. **Create a single file, either using markdown or PDF, to submit your ER Diagram in the doc folder**



2. You should have the ER/UML diagram for your application. You also need to provide a description of the assumptions you make for each entity and relationship. For example, "we think that there must be only 1 advisor for each student".

Assumptions:

There is only one YouTuber for each video, but there may be multiple videos for each YouTuber.
Each region should have many videos but each video can only have one region (we assume each video can only have one region).
Each category should have many videos but each video can only be in one category (we assume each video can only be in one category).
Each statistic only stores data for one video and each video can only have one statistic.

Relationships and Cardinalities:

YouTuber to Video:
Relationship: A YouTuber can upload multiple videos, but only one YouTuber uploads each video.
Cardinality: 1 (YouTuber) to Many (Videos)

Video to Statistics:

Relationship: The Statistics stored data about likes, dislikes, and comments which only relate to one video. Each video only has one statistic so this relationship is 1 to 1.
Cardinality: 1 (Video) to 1 (Statistics)

Video to Region:
Relationship: A video is associated with one region, indicating its popularity or availability. However, a region can be associated with many videos.
Cardinality: Many (Videos) to 1 (Region)

Video to Category:
Relationship: A video belongs to one category, representing its genre or content type. A category can have many videos.
Cardinality: Many (Videos) to 1 (Category)

3. Your ER/UML diagram should satisfy the following requirements:
    a. Your project must involve at least 5 entities with at most one entity regarding user login information.
    b. The 5 entities listed above do not include any relationship tables
    c. Creator to videos is 1-many, region to videos is 1-many, etc

4. Normalize your database. Apply BCNF or 3NF to your schema or show that your schema adheres to one of these normal forms. Describe why you choose to use BCNF vs 3NF,

    All non-key attributes in each table are functionally dependent only on the primary key of that table. This means the schema is already in 3NF. Also, for a table to violate BCNF, there must be a non-trivial functional dependency where the determinant is not a superkey. None of our tables have such a functional dependency. Thus, our schema is also in BCNF.
    Justification:
        - **YouTuber Table**
          The only candidate key is channelID. All other attributes are functionally dependent solely on channelID. It satisfies 3NF and BCNF.
        - **Video Table**
          The only candidate key is videoID. All other attributes are functionally dependent solely on videoID. It satisfies 3NF and BCNF.
        - **Statistics Table**

The only candidate key is statisticsID. All other attributes are functionally dependent solely on statisticsID. It satisfies 3NF and BCNF.

- **Region Table**
  The only candidate key is regionID. All other attributes are functionally dependent solely on regionID. It satisfies 3NF and BCNF.
- **Category Table**
  The only candidate key is categoryID. All other attributes are functionally dependent solely on categoryID. It satisfies 3NF and BCNF.

5. Convert your conceptual database design (ER/UML) to the logical design (relational schema). A relational schema is not a DDL.

Your relational schema should be formatted as follows:
Table-Name(Column1:Domain [PK], Column2:Domain [FK to table.column], Column3:Domain,...)

PK: Indicates that the column is a primary key for the table
FK: Indicates that the column is a foreign key referencing the primary key of table.column.
Domain: INT, Decimal, VARCHAR(X),....

YouTuber:
channelID: String [PK], channelTitle: String, videoID:String [FK to Video.videoID]

Video:
videoID: String [PK], title: String, thumbnail_link: String, comments_disabled: Boolean, ratings_disabled: Boolean, description: String, channelID: String [FK to YouTuber.channelID], regionID: INT [FK to Region.regionID], categoryID: INT, [FK to Category.categoryID]

Statistics:
StatisticID: String [PK], date: String, views: INT, likes: INT, dislikes: INT, comments: INT (number of comments), videoID:String [FK to Video.videoID]

Region:
regionID: INT [PK], regionName: String, videoID:String [FK to Video.videoID]

Category:
categoryID: INT [PK], categoryName: String, videoID:String [FK to Video.videoID]