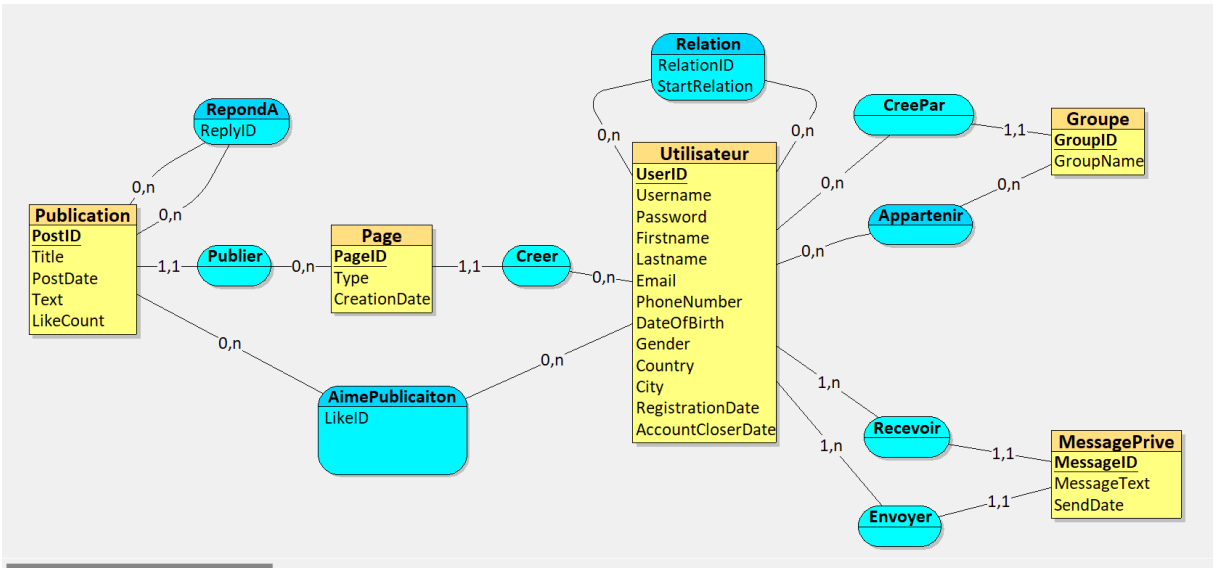
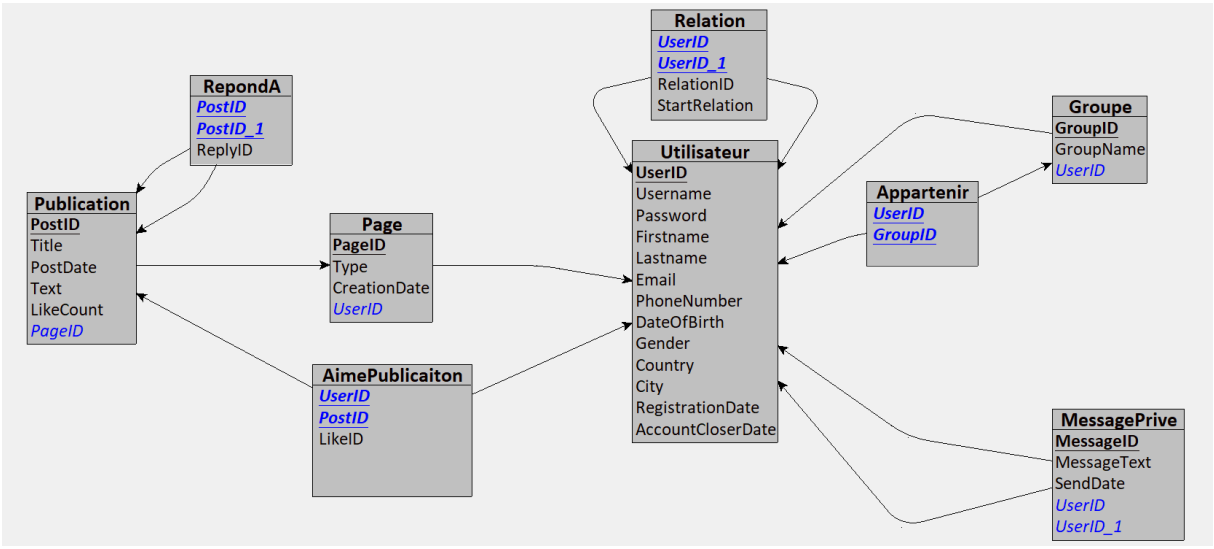
# Advanced Data Base project report



We designed the database considering the primary entities and relationships required to represent a social network like "Facebook." Firstly, we created a "User" table to store basic user information, including their first name, last name, email address, date of birth, gender, country, city of residence, as well as essential identification data such as username and password. These details are fundamental for user identification and profile management.

The "Page" table was introduced to enable users to create both public and private pages. This allows the management of content publishing while offering the option to define page privacy settings. Users can thus share information with a wider or more restricted audience based on their preferences.

To manage user posts, we introduced the "Publication" table. This table contains information about posts, such as title, publication date, text, and a "like" counter. It is linked to the "Page" table, thereby connecting posts to the pages they belong to.

Groups are a significant aspect of social networks. Therefore, we added the "Group" table to allow users to create and join groups. This table contains details about the group name and the user who created it.

User relationships, whether friendship or romantic relationships, are crucial in a social network. For this purpose, we created the "Relationship" table. It tracks social connections between users and records the start date of these relationships.

Private conversations between users are managed through the "PrivateMessage" table. This allows users to communicate privately with their friends, partners, or group members. The table stores information about the sender, recipient, message text, and sending date.

User interactions with posts, such as "likes," are tracked in the "LikesPost" table. This table records these interactions to evaluate post popularity and user preferences.

Finally, the ability to reply to posts is managed by the "ReplyTo" table. This allows tracking discussions and replies to posts, fostering user engagement.

LimiterAmis Trigger:

The "LimiterAmis" trigger was created to enforce a limit on the number of friends a user can have in the system. This limit, set at 500 friends, was implemented to prevent the database from becoming unmanageable in the event of a massive addition of friends. The trigger is set to fire whenever a new relationship is inserted into the "Relation" table. It checks whether the user has reached the friend limit or is attempting to become friends with themselves, and generates a custom error message in these cases. The goal of this trigger is to ensure that the business rule is enforced while informing users when their actions violate this rule.

LimiterPublicationsParJour Trigger:

The "LimiterPublicationsParJour" trigger was created to limit the number of posts a user can create in a day. The daily limit is set at 3 posts to maintain a high-quality user experience and to avoid content overload. Each time a new post is inserted into the "Publication" table, this trigger fires and checks whether the user has exceeded the daily limit. If the limit is reached, the trigger generates a custom error message to inform the user of the rule violation. The justification for this trigger lies in managing content quality and preserving a positive user experience.

SupprimerPublicationsAnciennes Trigger:

The "SupprimerPublicationsAnciennes" trigger was designed to automate the deletion of old posts in the database. It triggers every time a new post is inserted into the "Publication" table and checks the publication date of the new entry. If the post is more than 90 days old, the trigger automatically deletes it. The justification for this trigger is efficient data volume management and maintaining the relevance of the database. By automatically removing old posts, it contributes to the performance and storage space management of the database.