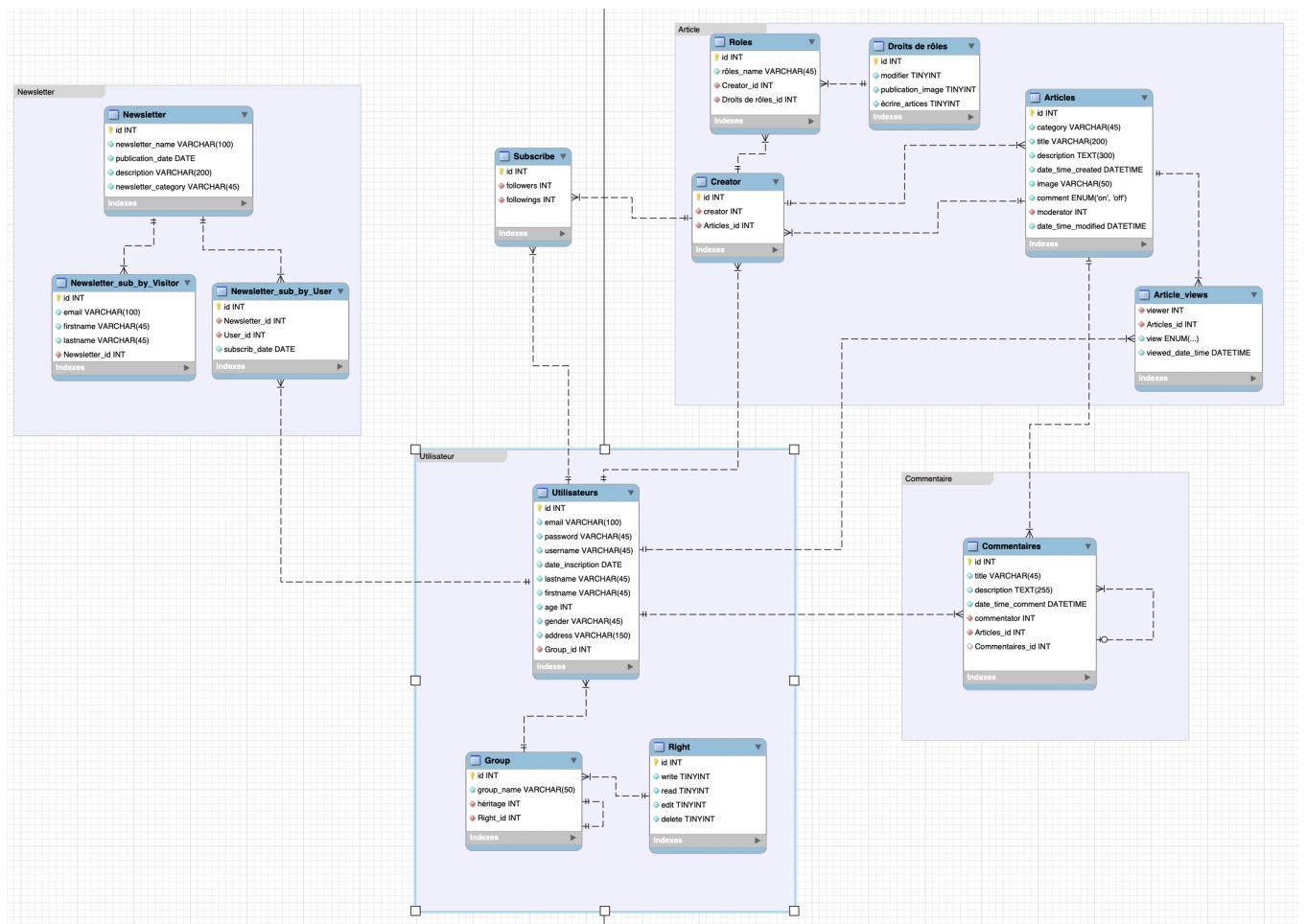


19minutes

Introduction

Nous avons créé un schéma de notre base de données sur Workbench permettant de répondre aux besoins de l'entreprise afin de pouvoir concurrencer le 20minutes. Vous trouverez ci-dessous le schéma dans sa globalité regroupant les fonctionnalités de base dont disposera la première version du site.



Ce schéma contient 4 parties principales : Utilisateur, Article, Commentaire et Newsletter. Chaque partie contient des tables de la base de données et nous avons aussi ajouté une table Subscribe pour répondre aux besoins de l'entreprise.

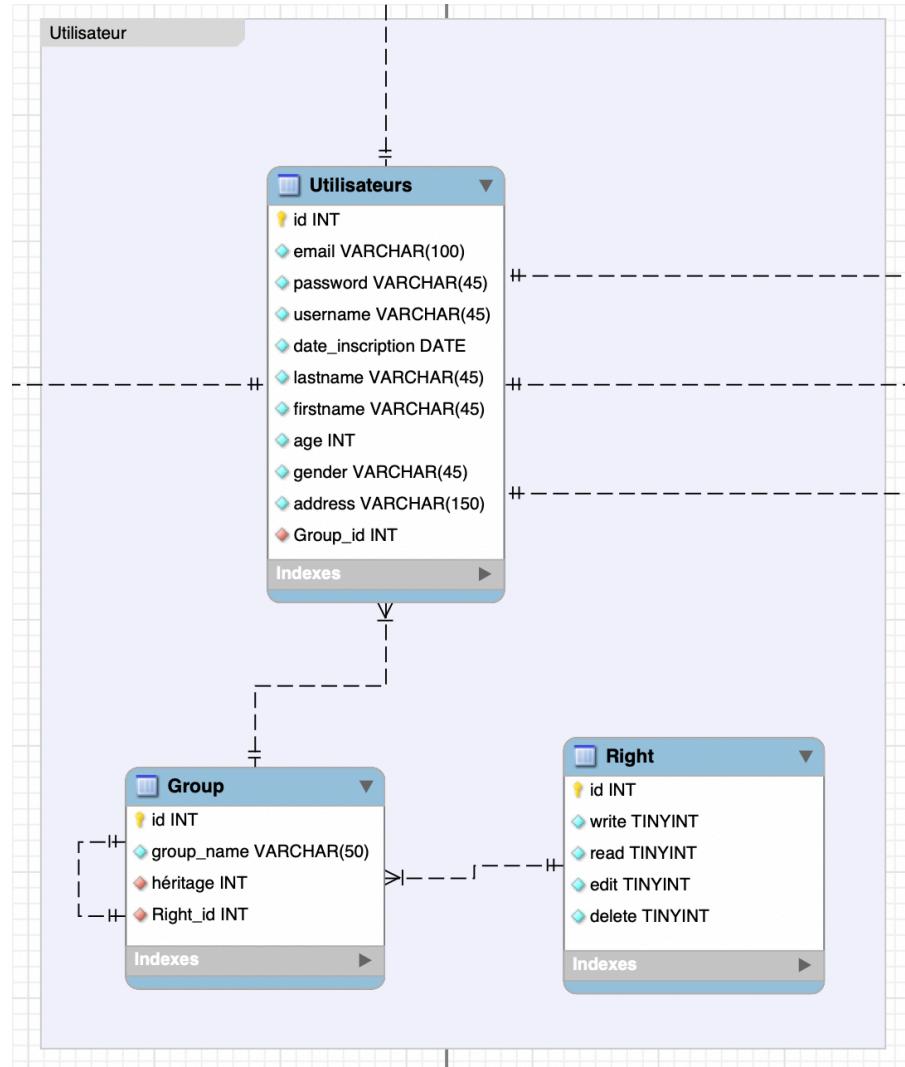
Le layer Utilisateur

Afin de répondre au besoin de gestion des comptes utilisateurs, nous avons créé un layer qui s'appelle Utilisateur dans lequel se trouvent les tables Utilisateurs, Group et Right.

Les tables Utilisateurs

contient les colonnes des références des utilisateurs du site :

- son identifiant
- son e-mail
- son mot de passe
- son nom d'utilisateur
- sa date d'inscription
- son nom et prénom
- son âge
- son sexe
- son adresse



La table Groupe contient

les colonnes des éléments des groupes comme l'identifiant, le nom du groupe, l'héritage de chaque groupe et les droits du groupe qu'on trouve dans la colonne 'Right_id' qui est une clé étrangère.

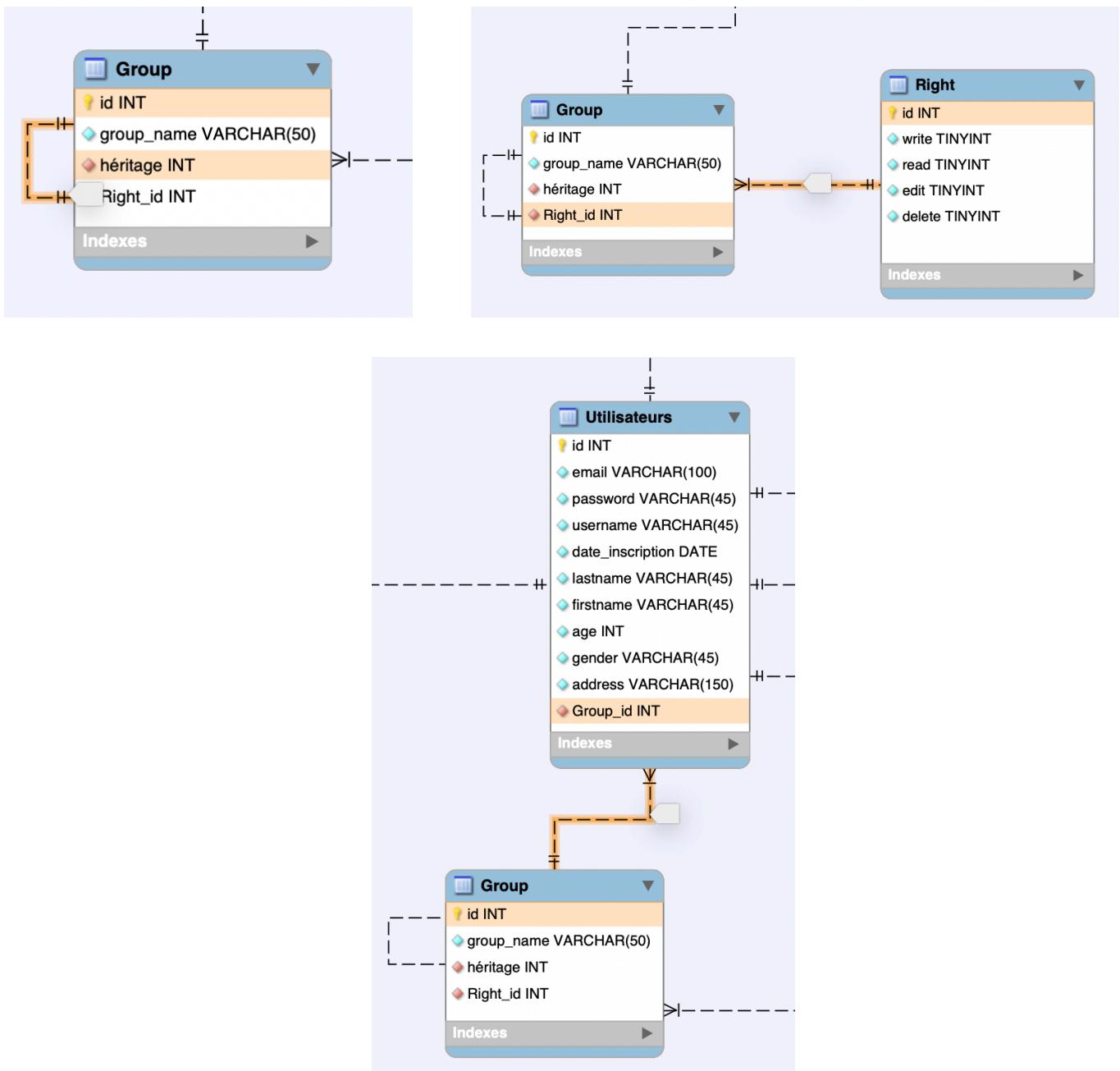
La table Right contient les colonnes des droits des groupes (write, read, edit, delete) et la colonne 'id' qui est une clé primaire. Nous avons créé des booléens sur chacune des colonnes des droits afin de d'accorder ou nous la permission de tel ou tel droit pour chaque groupe.

Les relations

On fait un lien 1:n (one-to-many, non-identifying) entre la table Group et la table Utilisateurs afin que chaque utilisateur appartienne à un groupe, et, pour que plusieurs utilisateurs puissent appartenir au même groupe.

On fait un lien 1:n (one-to-many, non-identifying) entre la table Right et la table Group afin de donner des droits différents à chacun des groupes, et, pour que ces derniers n'ai qu'une seule série de droits, c'est-à-dire qu'une seule ligne de chacun des quatre droits est associée à un seul groupe.

On fait un lien 1:1 (one-to-one, non-identifying) dans la table Group, entre la colonne ‘id’, la clé primaire de la table Group et la colonne ‘héritage’ qui est la clé étrangère générée par ce lien permettant à chaque groupe d’hériter des droits d’un autre groupe.



Article-Layer

Cette partie est créée pour répondre aux besoin de la gestion des articles dans le site. Le layer Article qu'on avais créé contient 4 tables : Articles, Creator, Roles , Droits de rôles et Article_views.

La table Articles contient les éléments concernants chaque article ; id, catégorie, titre, description, date et l'heure crée articles, image, modérateur d'article, date et l'heure de modification et la colonne de la permission du commentaire qu'on avait utilisé le datatype ENUM qui comprenne un ensemble statique prédéfini de valeurs ('on', 'off'). Cela répond à la besoin d'entreprise que chaque article pourra ou pas être commenté par d'autres utilisateurs.

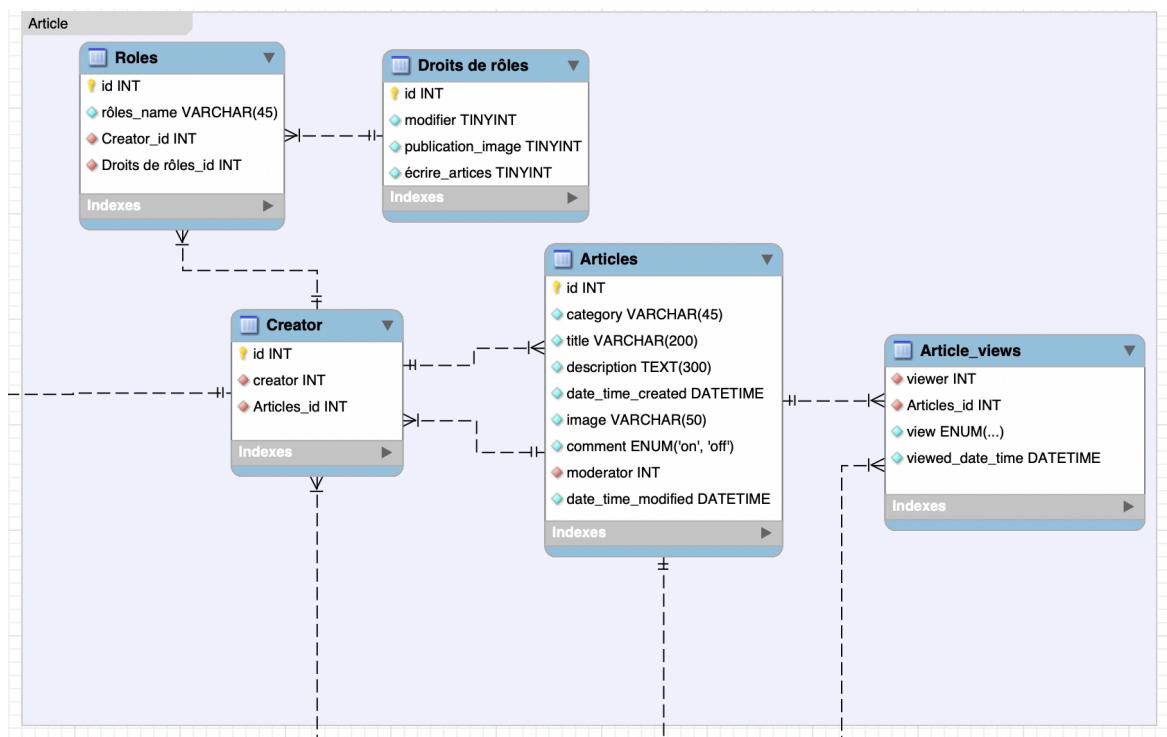
La table Creator

On avais créé cette table pour répondre à la besoin de chaque article est créé par plusieurs personnes. Dans cette table on se trouve les colonnes ; id de créateur, créateur, id d'article.

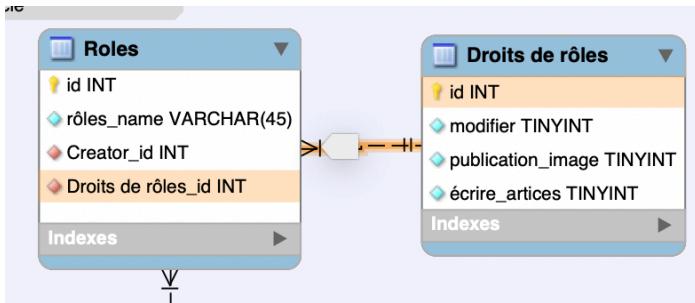
La table Roles nous permet de définir les rôles aux créateurs d'articles et un créateur peut avoir plusieurs rôles. Cette table contient l'id de rôles, le nom de rôles, l'id de créateur et l'id des droits des rôles.

La table Droits de rôles contient les colonnes des droits des rôles (modifier, publication d'image, écrire l'article). On avait utilisé le datatype TINYINT (BOOLEN) à chaque colonne afin de donner la permission des droits à chaque rôle.

La table Article_views contient les colonnes ; viewer(utilisateur), Articles_id, view_date_time, view. Dans la colonne 'view', on avait utilisé le datatype ENUM qui comprenne un ensemble statique prédéfini de valeurs ('read', 'unread'). Et grâce à cela les utilisateurs devront savoir s'ils ont lu l'article ou s'ils ne l'ont pas lu.



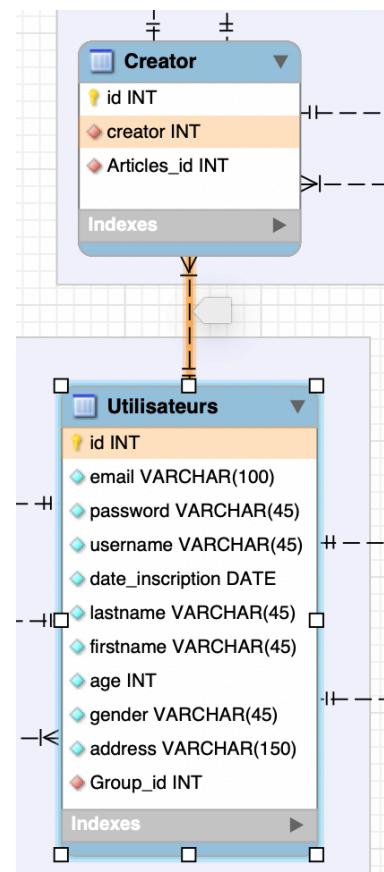
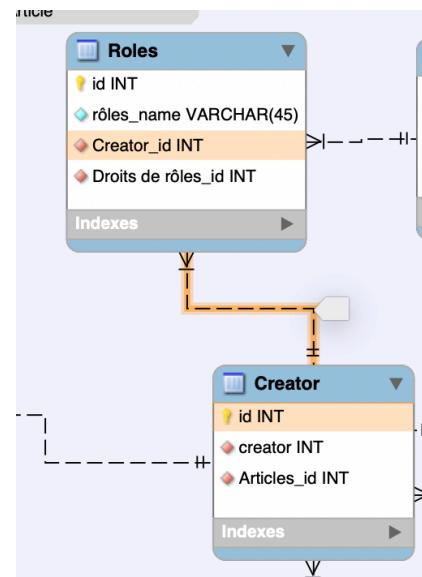
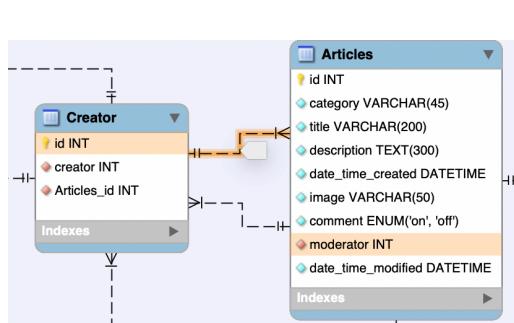
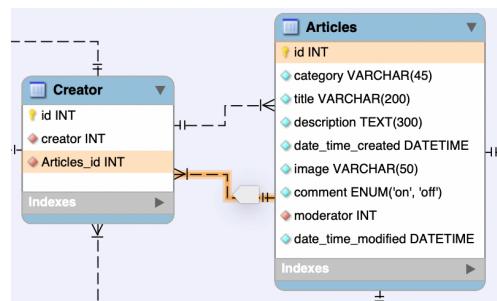
Les relations



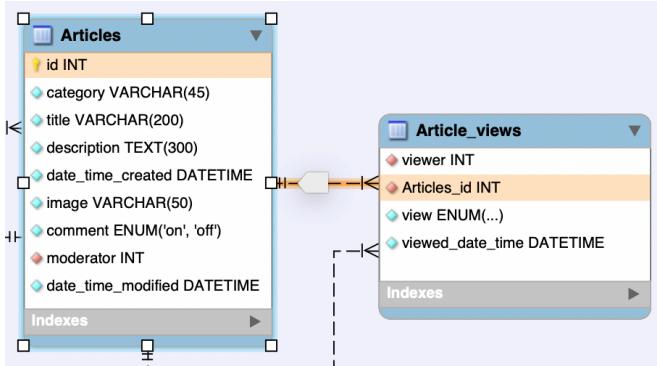
On avait fait un lien 1:n (one-to-many non-identifying) entre la table Droits de rôles et la table Rôles afin de donner des droits à chacun des rôles et afin que ces derniers n'ai qu'une série de droits.

On avait fait un lien 1:n (one-to-many non-identifying) entre la table Rôles et la table Creator afin de donner des rôles à chacun des créateurs et afin que ces derniers n'ai qu'une série de droits.

Entre la table Articles et la table Creator, on avait fait deux liens de 1:n (one-to-many non-identifying). Le premier est pour dire que chaque article est créé par plusieurs personnes ou il peut avoir plusieurs créateurs. Et le deuxième est pour dire que les modérateurs d'articles sont parmi des créateurs d'articles afin de répondre à la besoin d'entreprise que pour chaque article, un utilisateur pourra être désigné pour faire la modération, par défaut ce sera le créateur de l'article



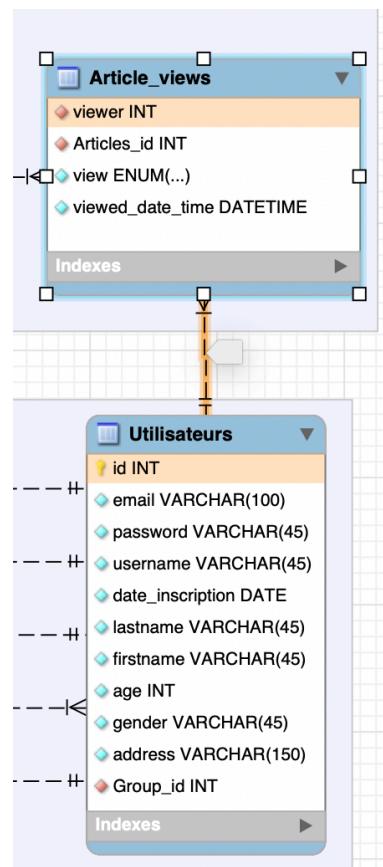
On avait fait aussi un lien 1:n (one-to-many non-identifying) entre la table Utilisateurs et la table Creator pour dire les utilisateurs peuvent devenir le créateur d'articles (si on le donne le droit selon la table Rôles) et



plusieurs articles.

chaque utilisateur peut créer plusieurs articles.

On avait fait un lien 1:n (one-to-many non-identifying) entre la table Articles et la table Articles_view pour dire que chaque utilisateur(viewer) peut lire plusieurs articles

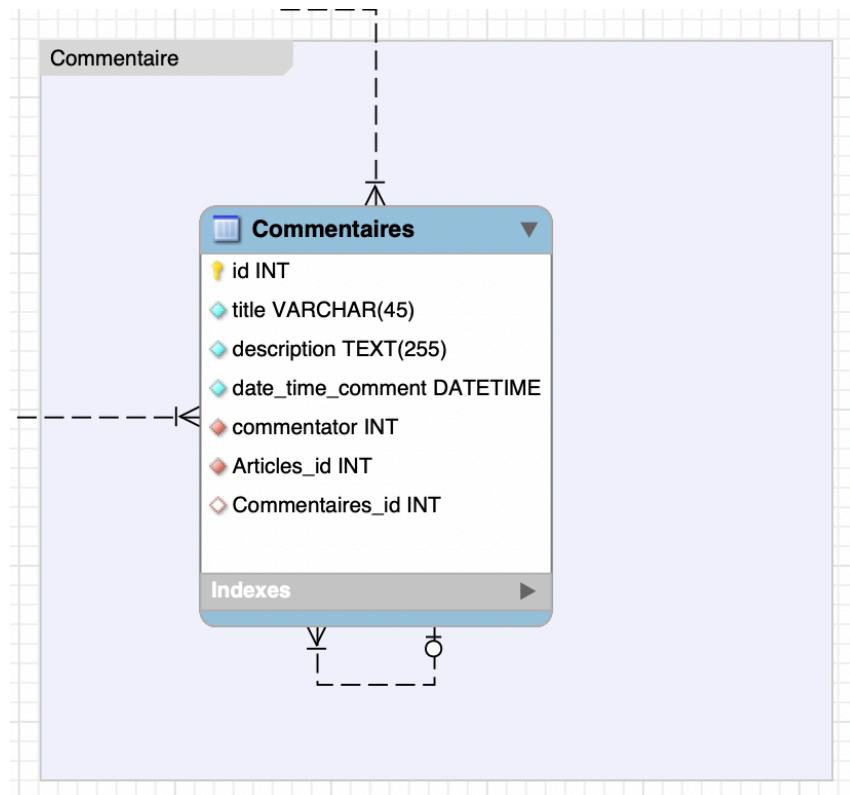


On avait fait un lien 1:n (one-to-many non-identifying) entre la table Utilisateur et la table Articles_view pour dire que chaque utilisateur peuvent lire les articles et il peut savoir s'il a lu l'article ou s'il ne l'a pas lu et quand.

Commentaire-Layer

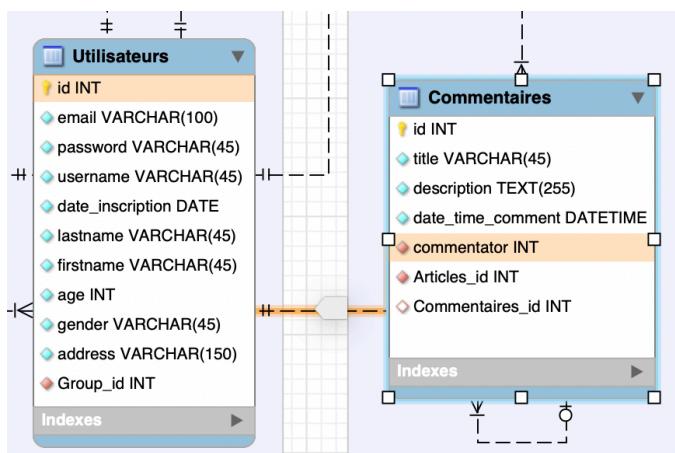
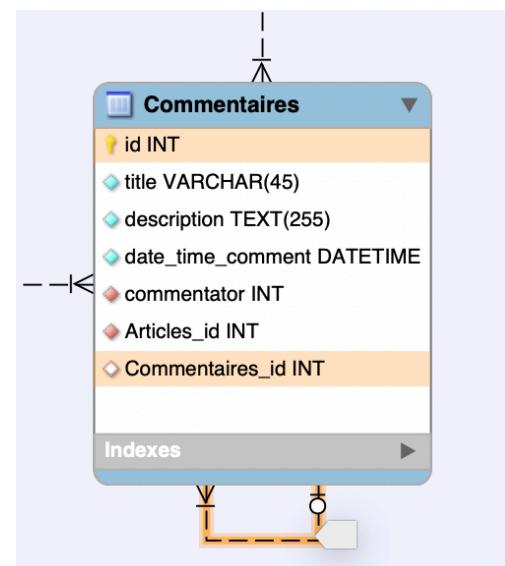
Cette partie est créée pour répondre aux besoin de la gestion des commentaires d'articles. Elle ne contient qu'une seule table, c'est la table Commentaires dans laquelle on se trouve les éléments concernants des commentaires:

- leur id
- leur titre
- leur description
- la date et l'heure
- commentateur (utilisateur)
- l'id d'article



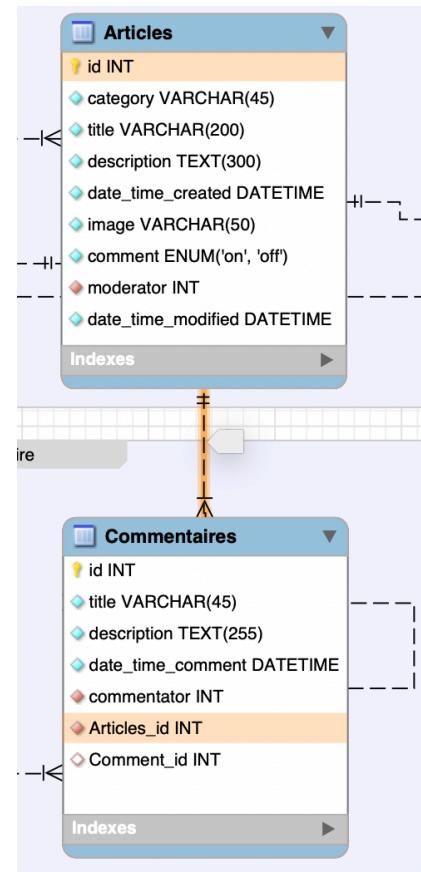
Les relations

On avait fait un lien 1:n (one-to-many non-identifying) entre la table Commentaires dans laquelle on se trouve, entre la colonne ‘id’ qu’on a désignée comme la primary key et la colonne ‘Commentaires_id’ qui est une foreign key et qu’elle peut être NULL afin que chaque commentaire puisse avoir plusieurs réponses, sous forme de sous commentaires.



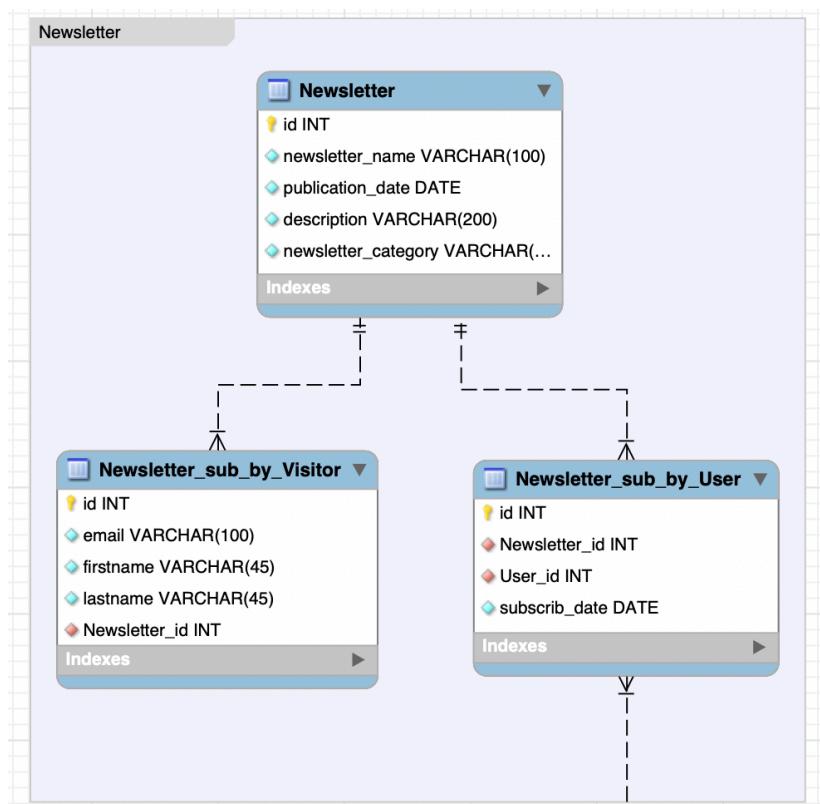
On avait fait un lien 1:n (one-to-many non-identifying) entre la table Utilisateurs et la table Commentaires afin que chaque utilisateur(commentator) puisse donner son avis aux articles qui sont créées par les créateurs. On peut dire que chaque article pourra être commenté par d’autres utilisateurs.

On avait fait un lien 1:n (one-to-many non-identifying) entre la table Articles et la table Commentaires afin que chaque article puisse être commenté par les utilisateurs (commentators).



Newsletter-Layer

La partie Newsletter a été créé pour répondre aux besoins de la gestion des newsletters (bulletins d'information). Cette partie contient 3 tables : Newsletter, Newsletter_sub_by_Visitor et Newsletter_sub_by_User.



La table Newsletter contient les références des newsletters: leur id, leur nom, leur date de publication, leur description et leur catégorie.

La table Newsletter_sub_by_Visitor

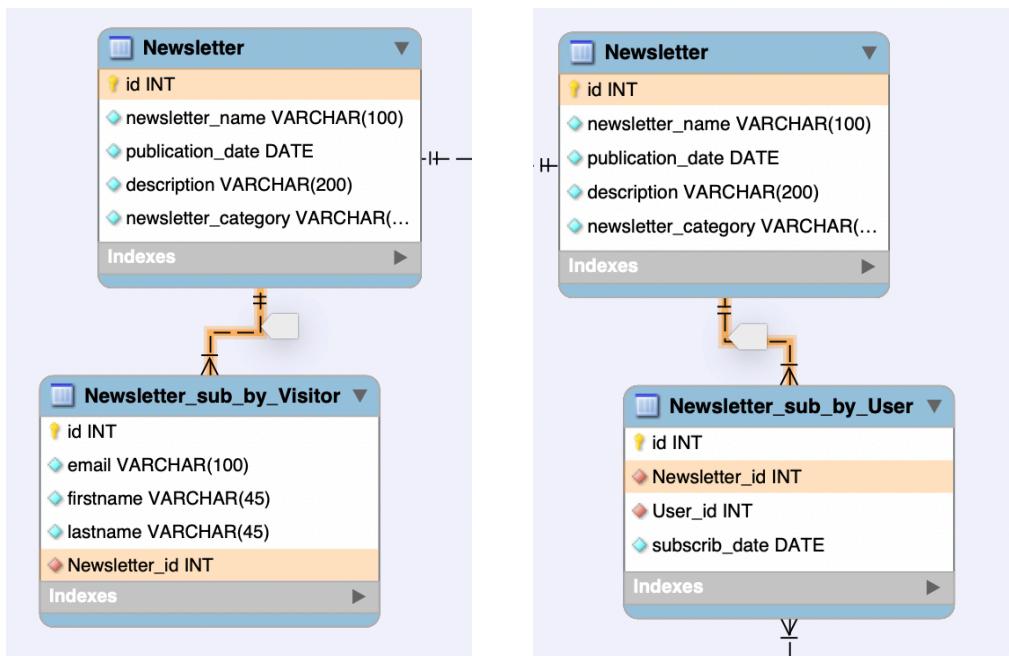
On avait créé cette table pour séparer les visiteurs du site et les utilisateurs qui ont créé leur comptes sur le site afin de ne pas être confuse entre deux groupes et aussi pour répondre à la besoin que n'importe qui peut s'abonner aux newsletters même s'il n'a pas crée un compte sur le site et on avait nommé ce groupe de gens ‘visiteurs’. Cette table contient les références nécessaires des visiteurs pour s’abonner aux newsletters comme leur id, leur e-mail, leur nom, et l’id des newsletters qu’ils se sont abonnées.

La table Newsletter_sub_by_User a été créée dans le même but que la table Newsletter_sub_by_Visitor mais le différence entre ce deux tables est celle de première est pour les visiteurs, mais la table Newsletter_sub_by_User est pour les utilisateurs et cette table ne contient que l’id d’utilisateurs, l’id des newsletters qu’ils se sont abonnées et la date.

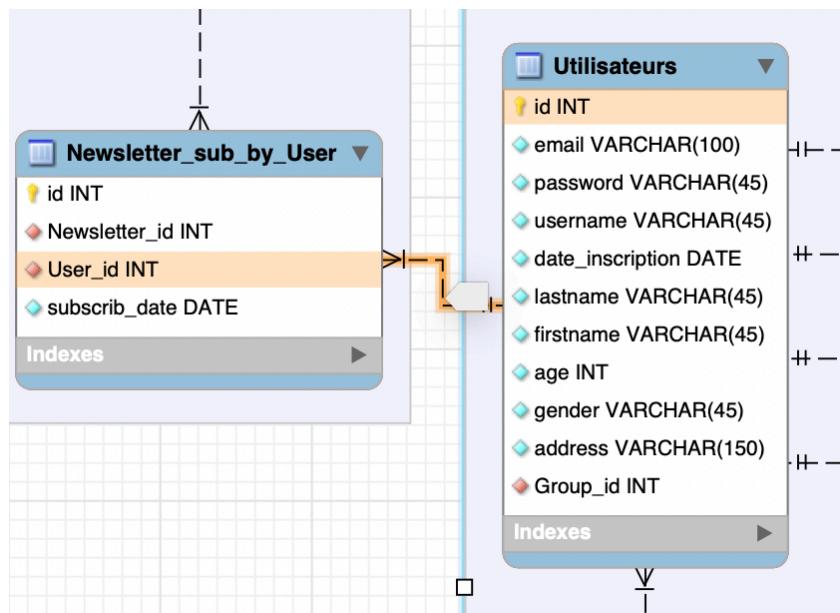
Les relations

On avait fait un lien 1:n (one-to-many non-identifying) entre la table Newsletter et la table Newsletter_sub_by_Visitor afin que chaque visiteur puisse s’abonner aux plusieurs newsletters.

On avait fait un lien 1:n (one-to-many non-identifying) entre la table Newsletter et la table Newsletter_sub_by_User afin que chaque utilisateur puisse s’abonner aux plusieurs newsletters.



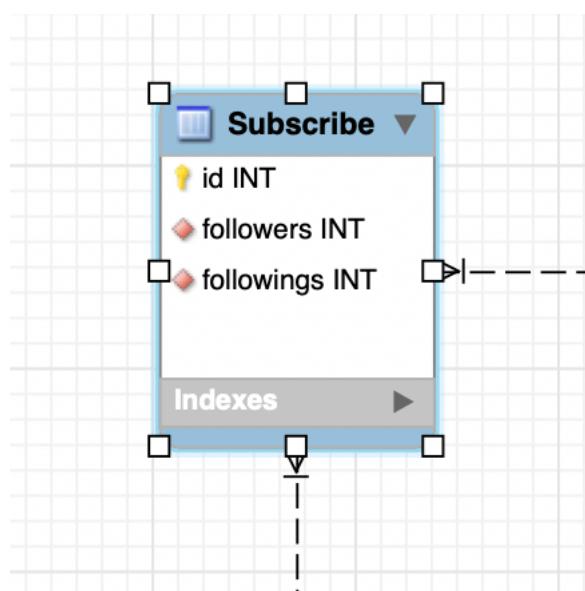
On avait fait un lien 1:n (one-to-many non-identifying) entre la table Utilisateurs et la table Newsletter_sub_by_User afin que les utilisateurs puissent s'abonner aux newsletters.



La table Subscribe

On avait créé les tables pour répondre aux besoins de chaque gestion mais il nous en reste un besoin qu'un utilisateur pourra s'abonner aussi à un autre utilisateur pour savoir quand un article le concernant est publié.

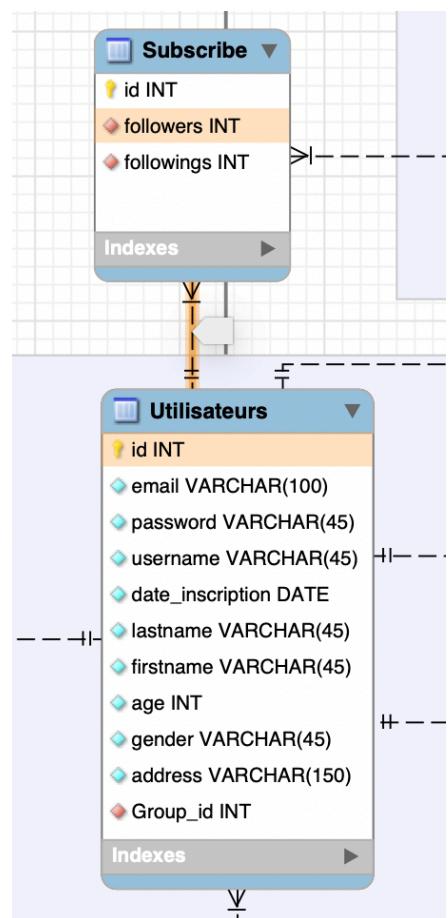
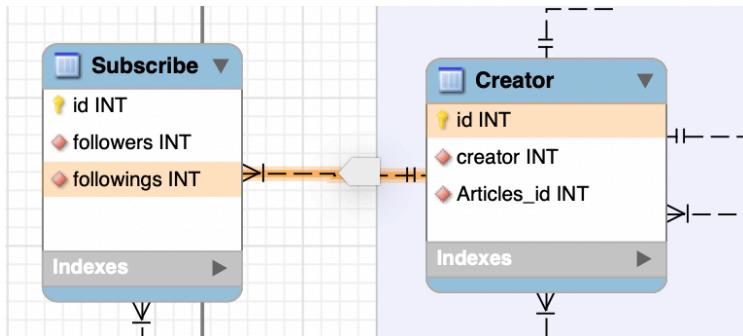
En vu de ce besoin, on avait créé cette table dans laquelle on se trouve les colonnes; id, followers et followings.



Les relations

On avait fait un lien 1:n (one-to-many non-identifying) entre la table Utilisateurs et la table Subscribe afin que chaque utilisateur puisse s'abonner à un autre utilisateur (créateur d'article) pour savoir quand un article le concernant est publié.

On avait fait un lien 1:n (one-to-many non-identifying) entre la table Creator et la table Subscribe afin que chaque créateur (following) d'articles soit abonné par les utilisateurs (followers) et que un créateur puisse avoir plusieurs personnes abonnées.



Glossaire

1:1 La relation one-to-one est définie comme une relation entre deux tables où les deux tables sont associées en utilisant la primary key unique et la foreign key contrainte . Un enregistrement de la table A se rapporte seulement à un enregistrement de la table B.

1:n La relation one-to-many est définie comme une relation entre deux tables, une ligne d'une table peut avoir plusieurs lignes correspondantes dans une autre table. Cette relation peut être créée à l'aide de la relation -clé étrangère. Un enregistrement de la table A se rapporte à un ou plusieurs enregistrements de la table B.

n:m La relation many-to-many est définie comme une relation entre deux tables où les deux tables sont associées. Un enregistrement de la table A se rapporte à un ou plusieurs enregistrements de la table B et un enregistrement de la table B se rapporte à un ou plusieurs enregistrements de la table A. Une relation n:n peut donc être considéré comme deux relation 1:n reliées par une table intermédiaire qu'on appelle “table de jonction”. Cette table est utilisée pour relier les deux autres tables. Pour ce faire, deux champs font référence à la clé primaire de chacune des deux autres tables.

Identifying La relation “identifiante” existe lorsque la primary key de l'entité parent est incluse dans la primary key de l'entité enfant.

Non-identifying La relation “non-identifiante” existe lorsque la primary key de l'entité parent est incluse dans l'entité enfant mais ne fait pas partie de la primary key de l'entité enfant.

VARCHAR (taille) Une chaîne de variable peut contenir des lettres, des chiffres et des caractères spéciaux). Le paramètre spécifie la longueur maximale de la colonne en caractères - peut aller de 0 à 65535.

TEXT (taille) Contient une chaîne d'une longueur maximale de 65 535 octets.

ENUM (val1, val2, val3, ...) Un objet chaîne ne peut avoir qu'une seule valeur en choisant parmi une liste de valeurs possibles. Une liste ENUM peut répertorier jusqu'à 65 535 valeurs dans. Si une valeur est insérée et elle ne figure pas dans la liste, une valeur vide sera insérée. Les valeurs sont triées dans l'ordre où on les entre.

BOOLEAN Zéro est considéré comme faux, les valeurs autres que zéro sont considérées comme vraies.

TINYINT (taille) Un très petit nombre entier. Ce type de variable représente au type BOOLEAN.

DATE Une date format: YYYY-MM-DD.

DATETIME Une combinaison de date et heure. Format: YYYY-MM-JJ hh: mm: ss.

TIMESTAMPS Le timestamp représente le nombre de seconde qui s'est écoulé depuis le 1er janvier 1970 (début de l'heure UNIX). On utilise pour identifier les transactions.