DBI202x_02-A_VN

Assignment 01

Phạm Minh Hùng - SE00234x

I. Analyse Requirements

This is an analysis about an online news paper.

1. List out requirements

Initial requirements:

- Guest can:
  - Signup to be a user.
  - Read articles.
  - Login using email and password.
- User has:
  - Email
  - Password
  - Profile image
  - Name: first and last
  - Gender
  - Role (viewer, writer/journalist, editor). One user only has one role.
  - Intro text
- Viewers can:
  - See the main page with the latest news/article.
  - Filter articles using criterias
    Filter criterias: category, publish time, writer, … etc.
  - Read details of the article.
  - Leave comments on article(s) (no nested comment/reply)
- Writer can:
  - Create new (draft) articles.
  - Modify (draft) articles.
  - Request to publish new articles.
  - Unpublish an article.

- Editor can:
    - Add a new writer (by changing the role of target viewer to writer).
    - Remove a writer (by changing the role of target writer to viewer).
    - Modify writer's requested article.
    - Approve writer's requested article.
    - Modify published articles.
    - Unpublish an article.
- Article should include:
    - Title
    - Brief
    - Content
    - Status (draft, pending, denied, published)
    - Publish date
    - Writer
    - Editor
    - Category

Addition requirements:

- User uses password to login.
- One article can be in multiple categories.

2. Narrative problem analysis

a. Identify entity types and attributes

- Role: name, description, created_at, updated_at.
- User: email, password, first name, last name, profile image, intro text, gender, created_at, updated_at.
- Category: name, description, created_at, updated time.
- Article: title, brief, content, status, published time, created_at, updated time.
- Comment: content, created_at, updated time.

b. Determine primary keys

For the Role table, the name can be unique, since right now, there's only three roles to be. In case we want to change the name, then it would be a problem if we have many records that use this role, so, we add a "roleId" as an id for the role record:

- Role: **role_id**, name, description, created_at, updated_at.

For the User table, the email is unique, but the user can change it later, so it cannot be id. Therefore, we add userId as id for this table:

- User: **user_id**, email, password, first_name, last_name, profile_image, intro_text, gender, created_at, updated_at.

For the Category table, the same with Role table, we add catId as id of this table:

- Category: **cat_id**, name, description, created_at, updated_at.

For the Article table, since it has nothing identify it on its own, so we create a new id for it:

- Article: **article_id**, title, brief, content, status, published_at, created_at, updated_at.

For the Comment table, it does look like a weak entity, and we can identify the entity using relationships with the user table, article table and the help from created_at. However, to make it easier, we create an id for it:

- Comment: **comment_id**, content, created_at, updated_at.

There's another step: "Connect entity types". I want to bring it to part **II. Build ERD** for convenience.

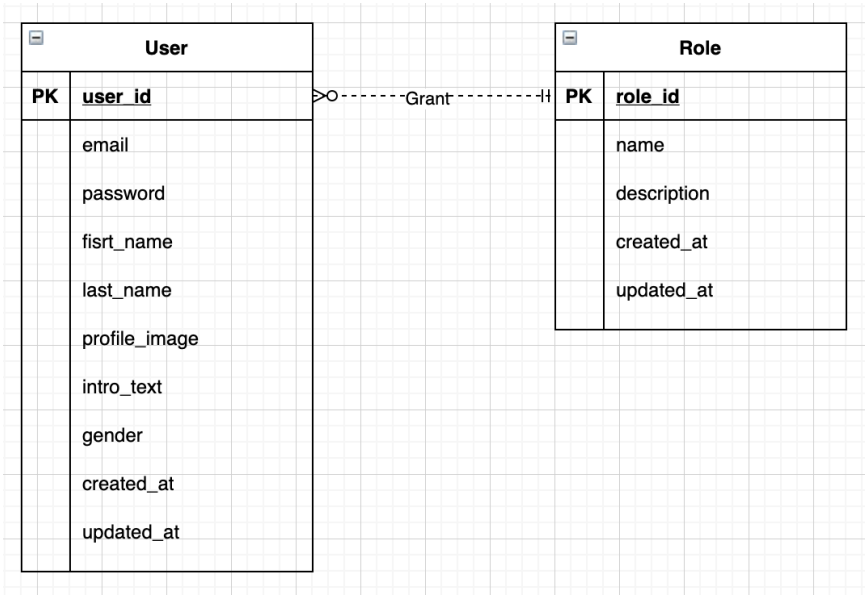
II. Build ERD

1. Continue Narrative problem analysis

We continue with the third step of "Narrative problem analysis": Connect entity types

a. User and Role

As the requirement stated, one User must and only has one Role, so the relationship between User table and Role will be:

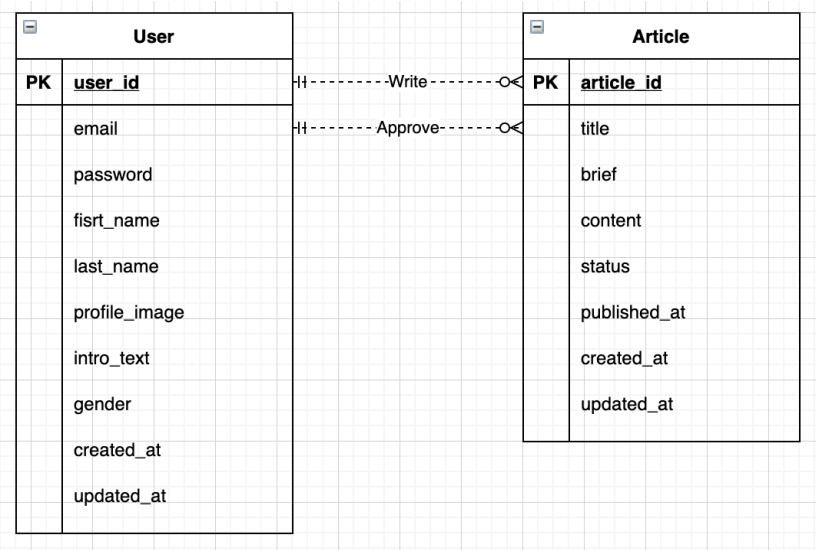


b. User and Article

For the relationship of User and Article, we have:

- Viewer view articles.
- Writer writes articles.
- Writer submits articles.
- Editor approves articles.

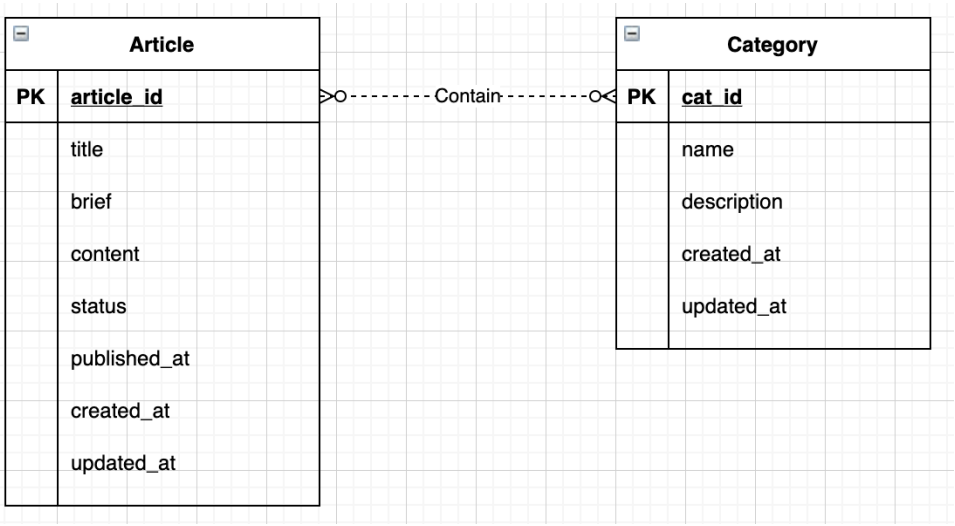
In all those three, we only need to know who wrote the article, and who approved that article, so we can eliminate the viewer part. The submitting part, we can control it using “status” prop of the article. Therefore, the relationship can look like this:



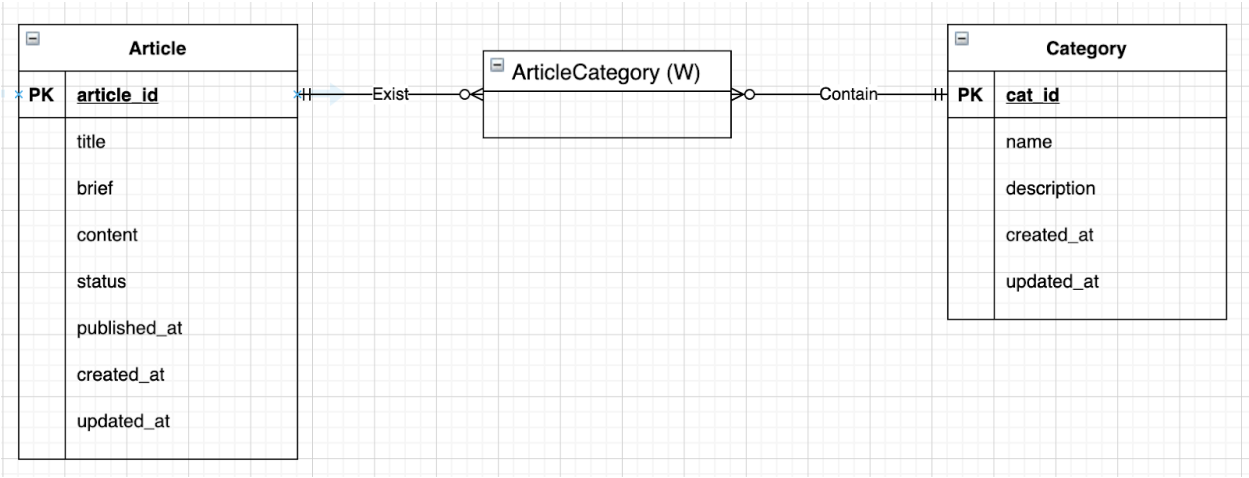
c. Article and Category

For each article, it must exist within at least one category, but when we create a new draft, it doesn’t need to specify which article to be in first, so we can consider it as it has no category. However, for maximum cardinality, it can exist in multiple categories.

For each category, it can contain no article, but also can contain multiple articles. Therefore, the relationship can look like this:



We can utilize weak entity to convert this M-N relationship to two 1-M relationships:



*Note: the drawing tool doesn't have the Weak entity like the one specified in lecture. So I used "**(W)**" to note that this is a weak entity.*

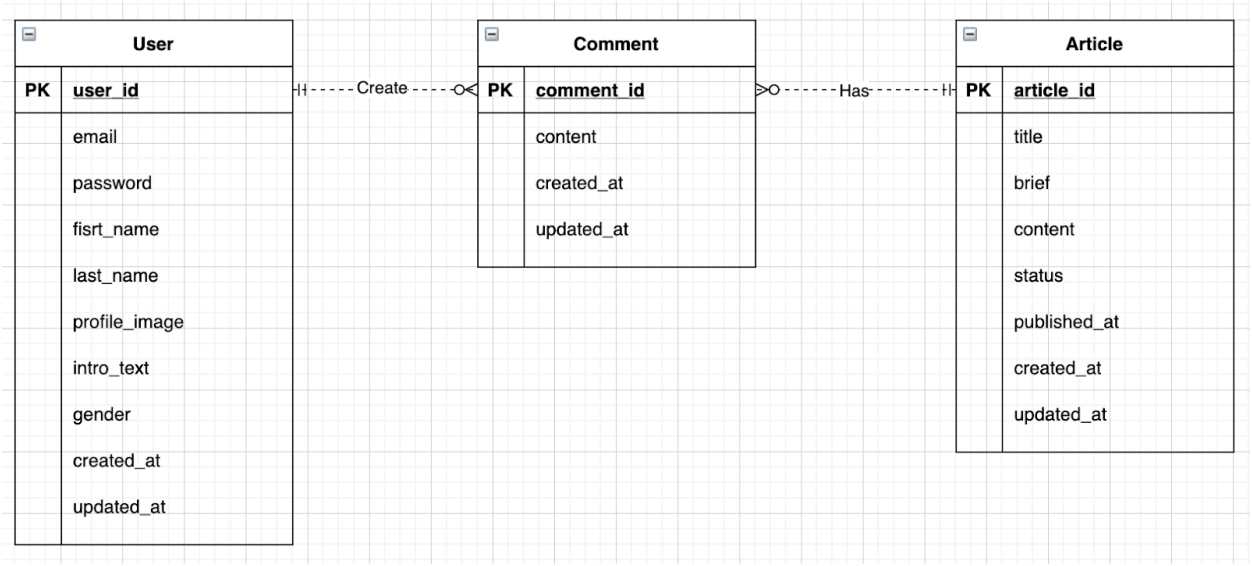
d. User, Article and Comment

A user can comment on multiple articles.

In one article, a user can make multiple comments. That means one article can have multiple comments, from one or multiple users.

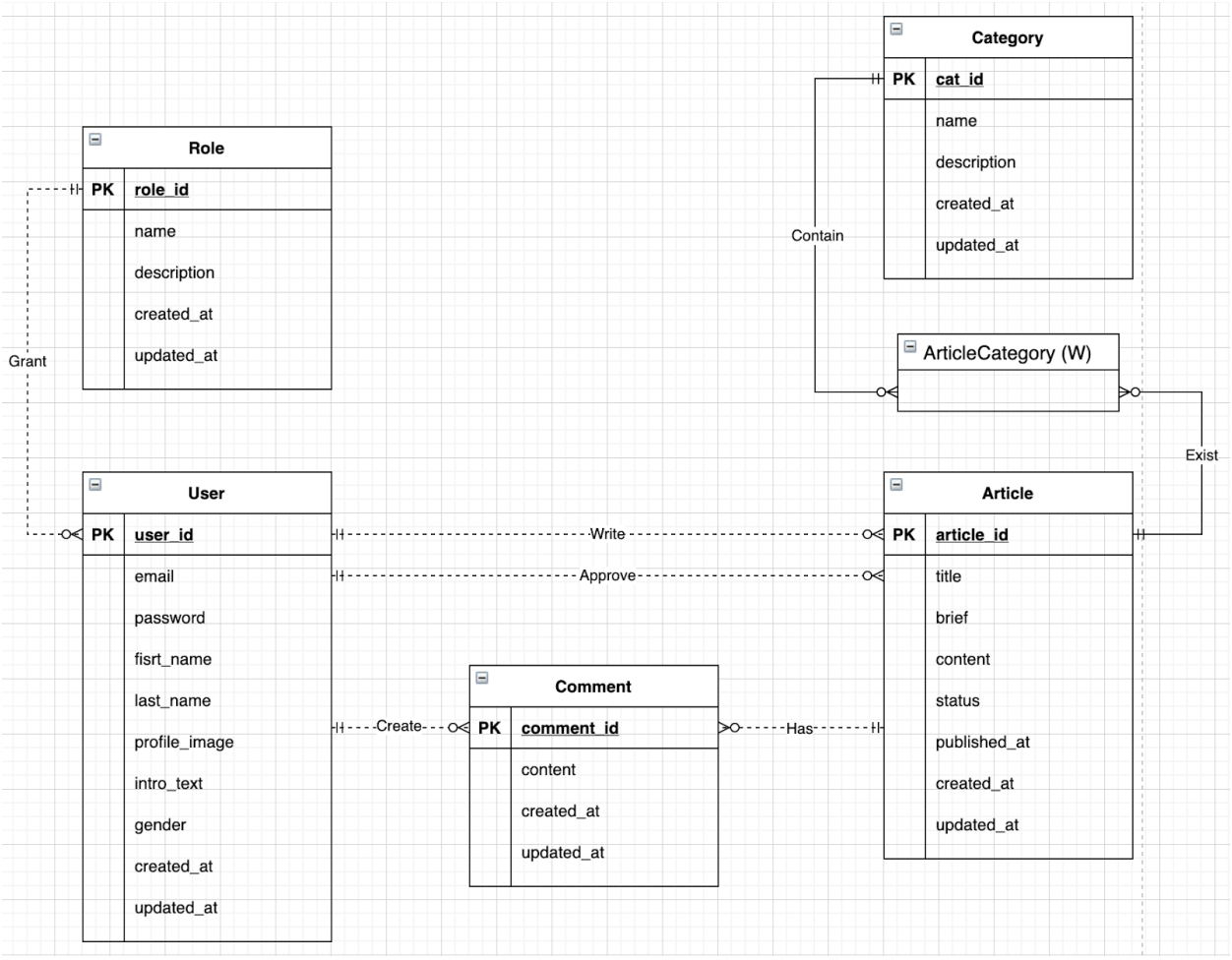
For each comment, it must be associated with a user and an article.

Therefore, we can illustrate the relationship with below diagram:



2. Combine all

Combine all relationships above, we have:

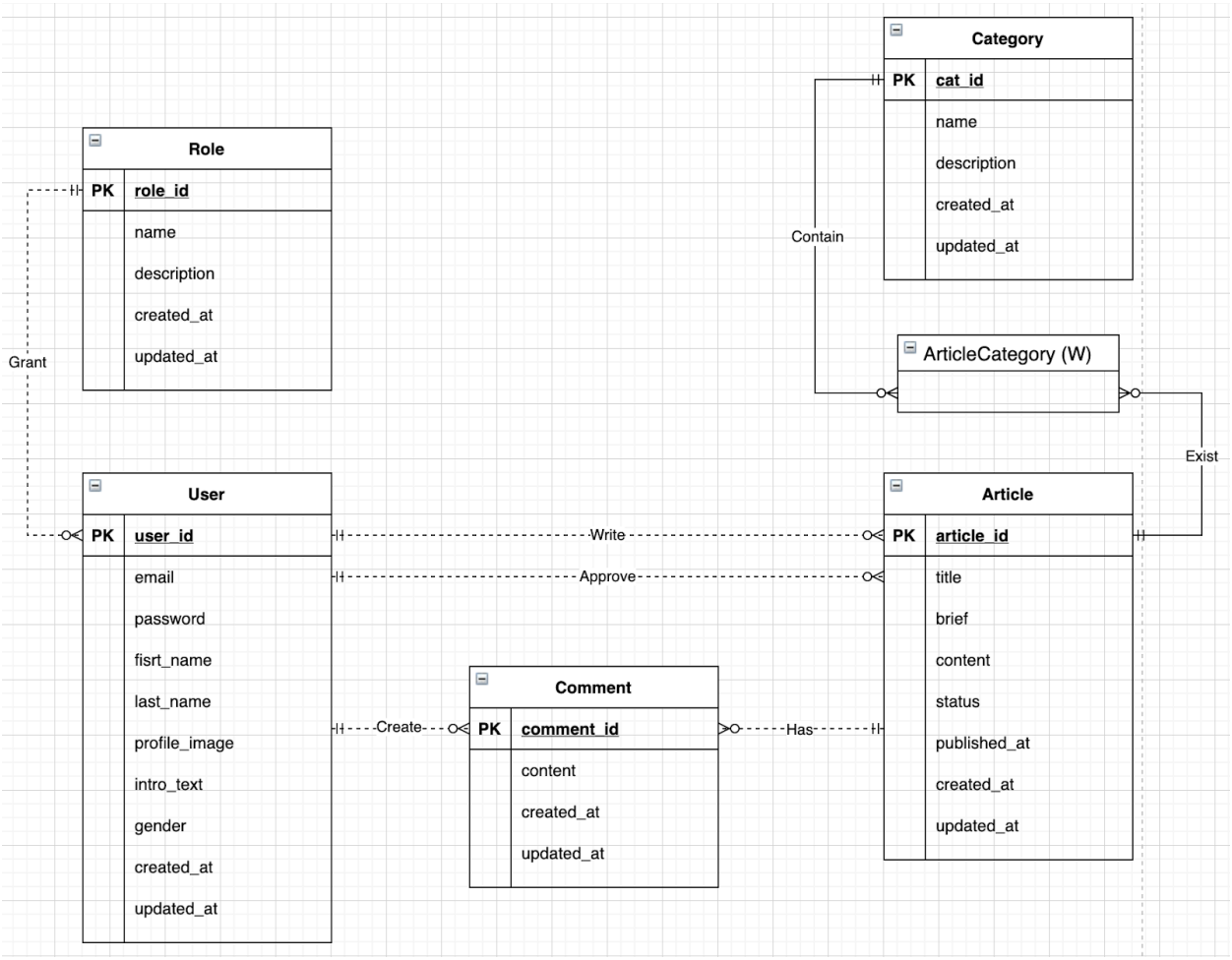


III. Convert ERD to Relational Model Diagram

We use Conversion Rule to convert ERD to Relational Model Diagram

1. Entity Type Rule

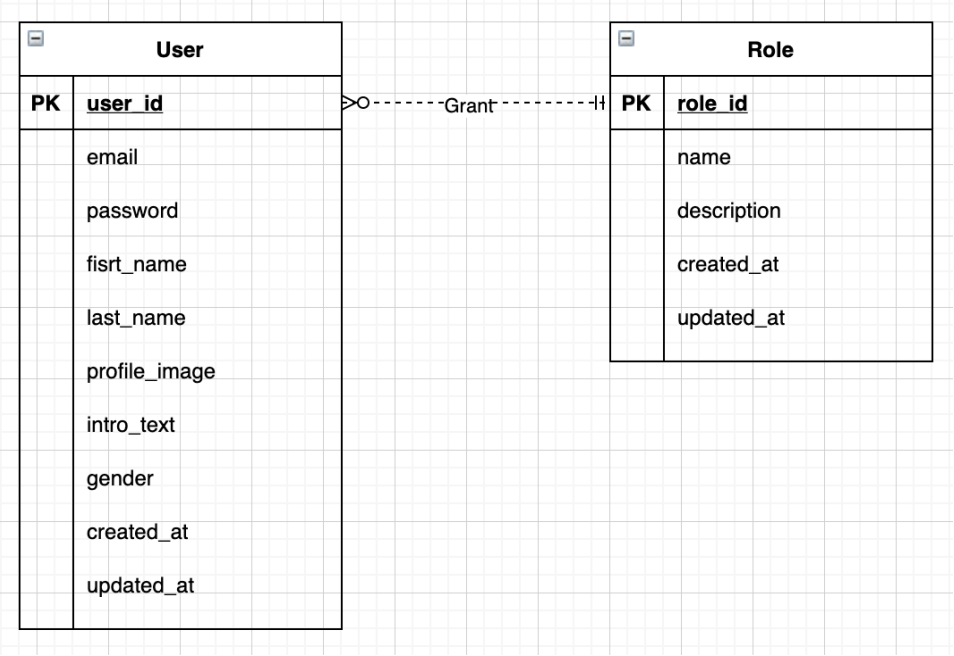
When creating ERD, I went ahead of time specifying Primary Key for tables, so the end result will look the same:



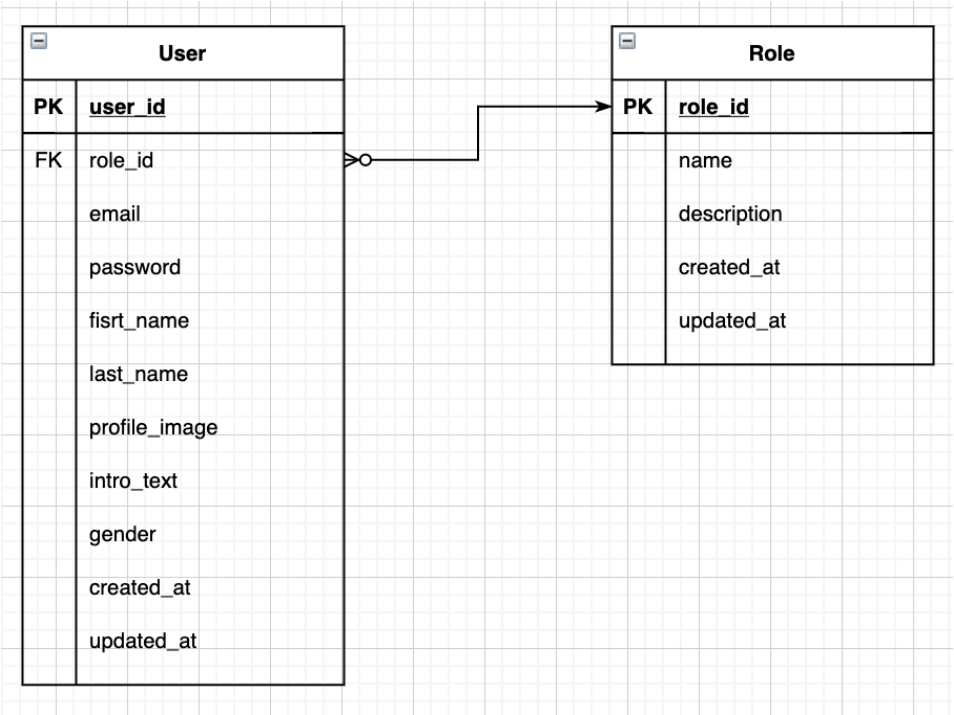
2. 1-M Relationship Rule

a. User and Role

From:

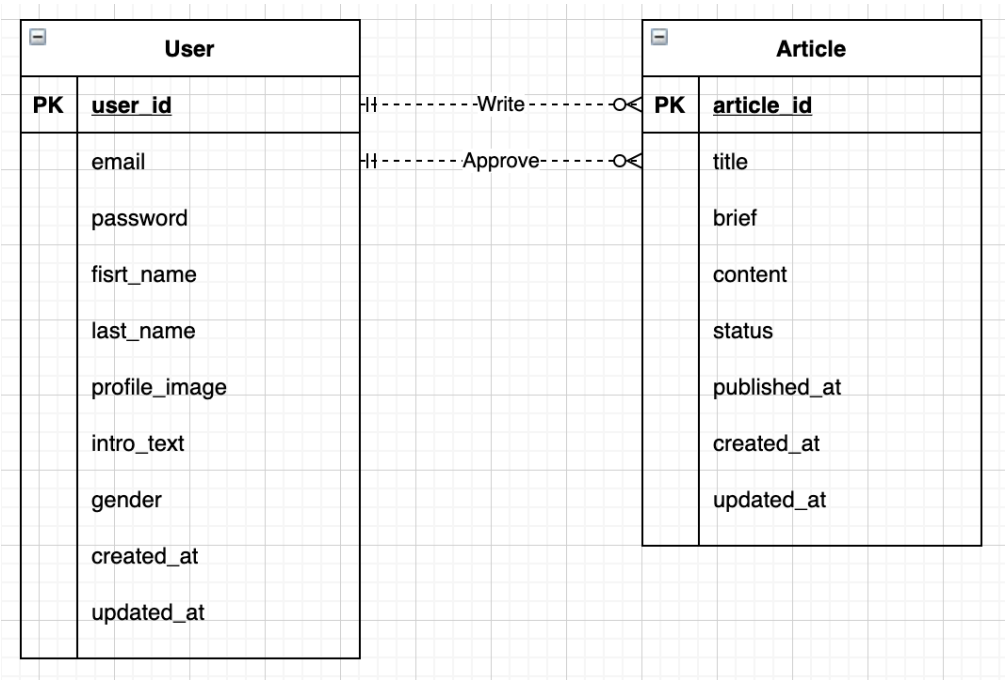


Converted to:

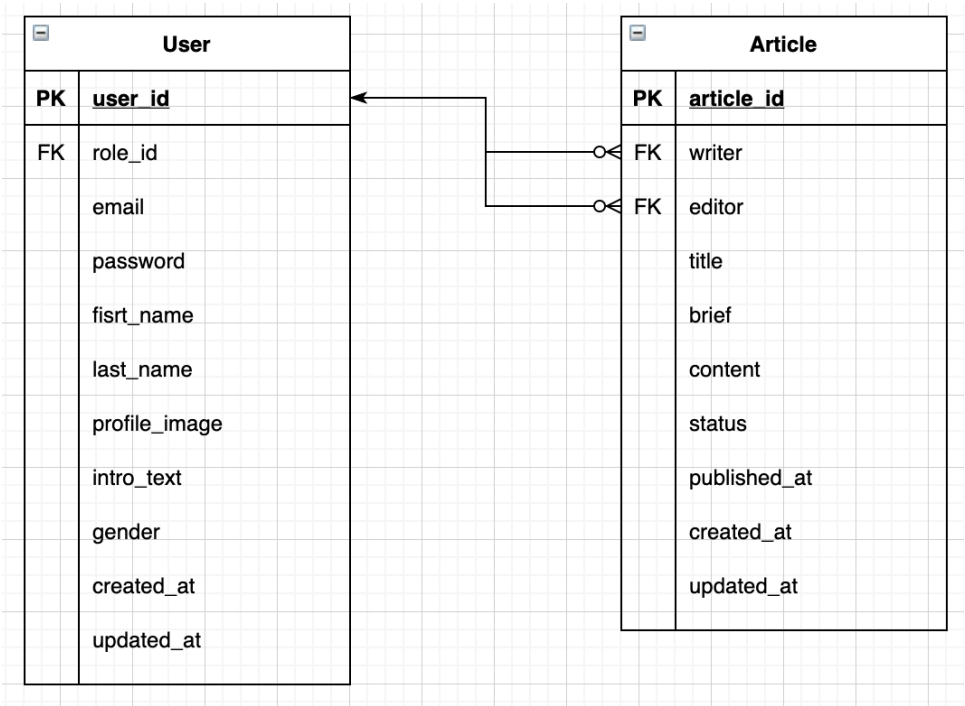


b. User and Article

From:



Converted to:



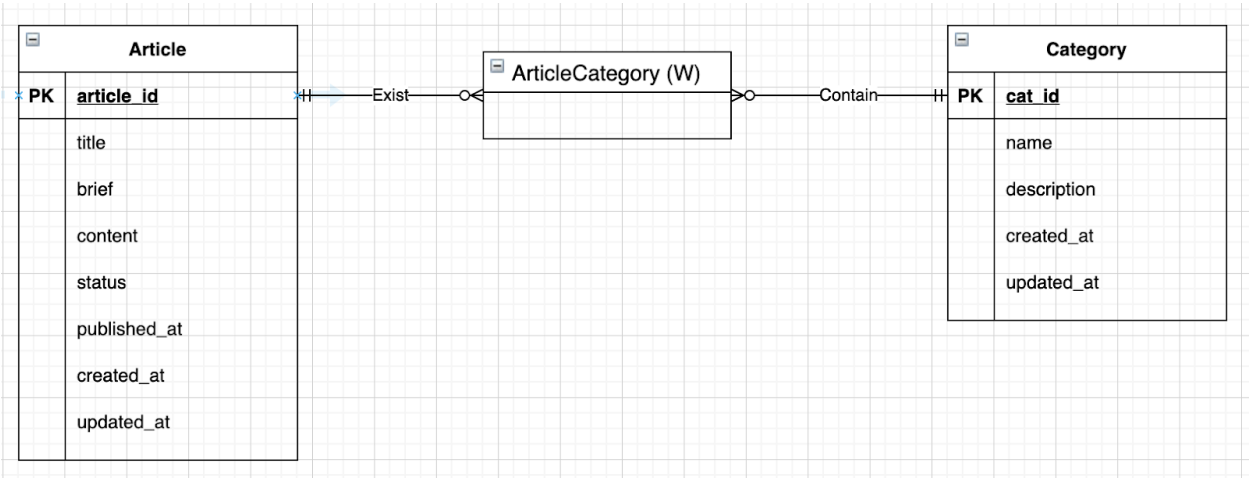
3. M-N Relationship Rule

Not applicable

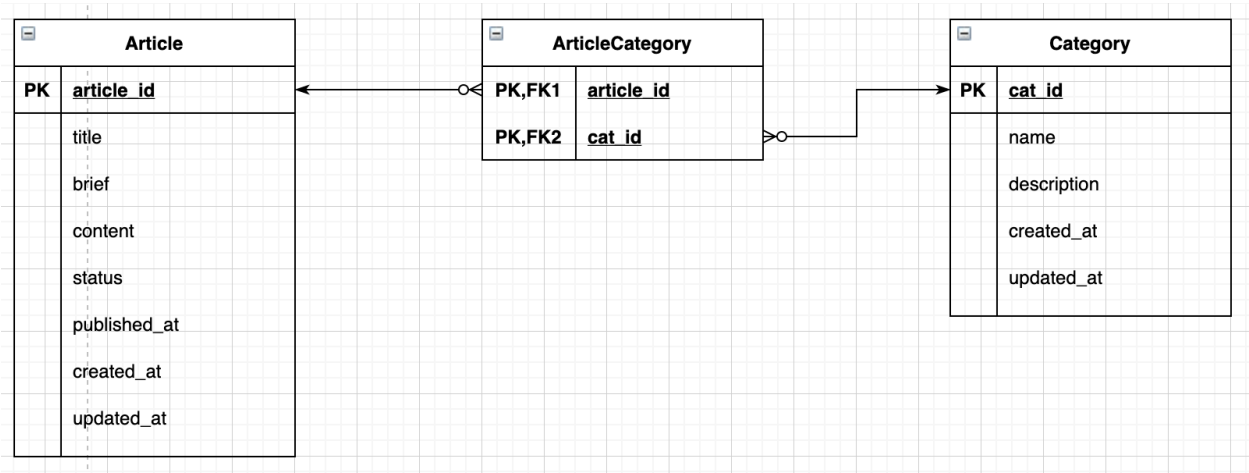
4. Identify Relationship Rule

a. Article and Category

From:

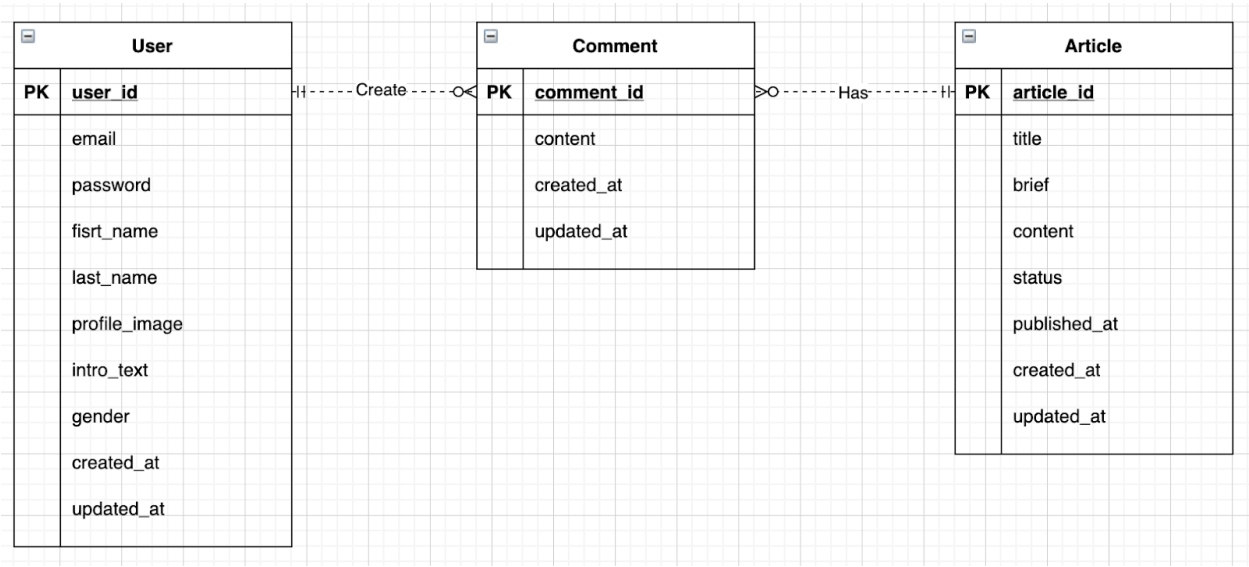


Converted to:

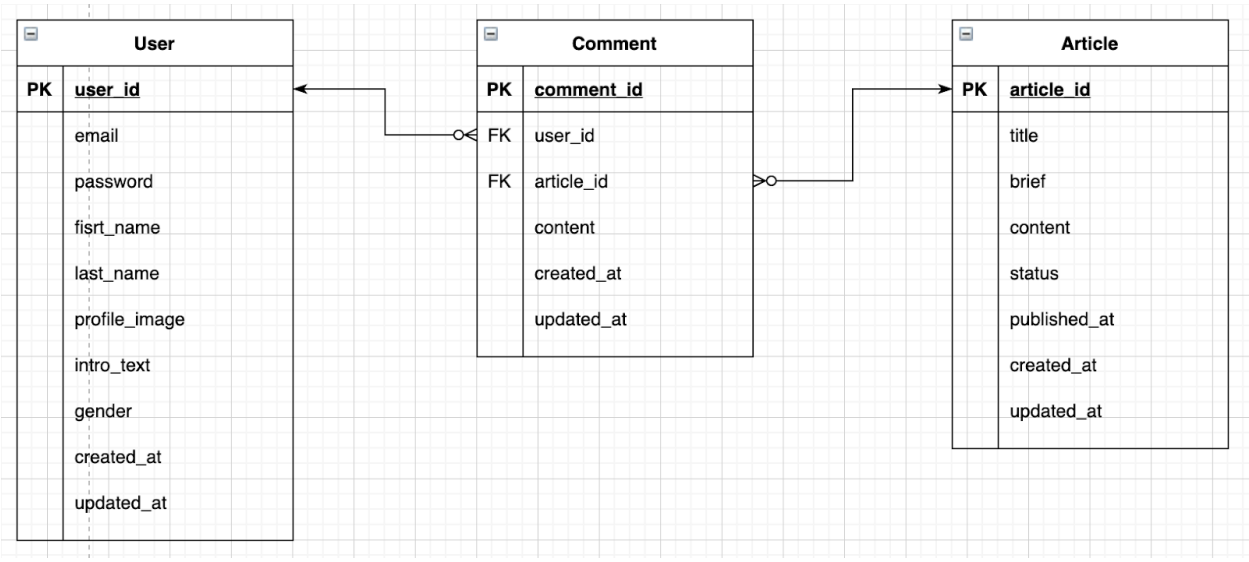


b. User, Article and Comment

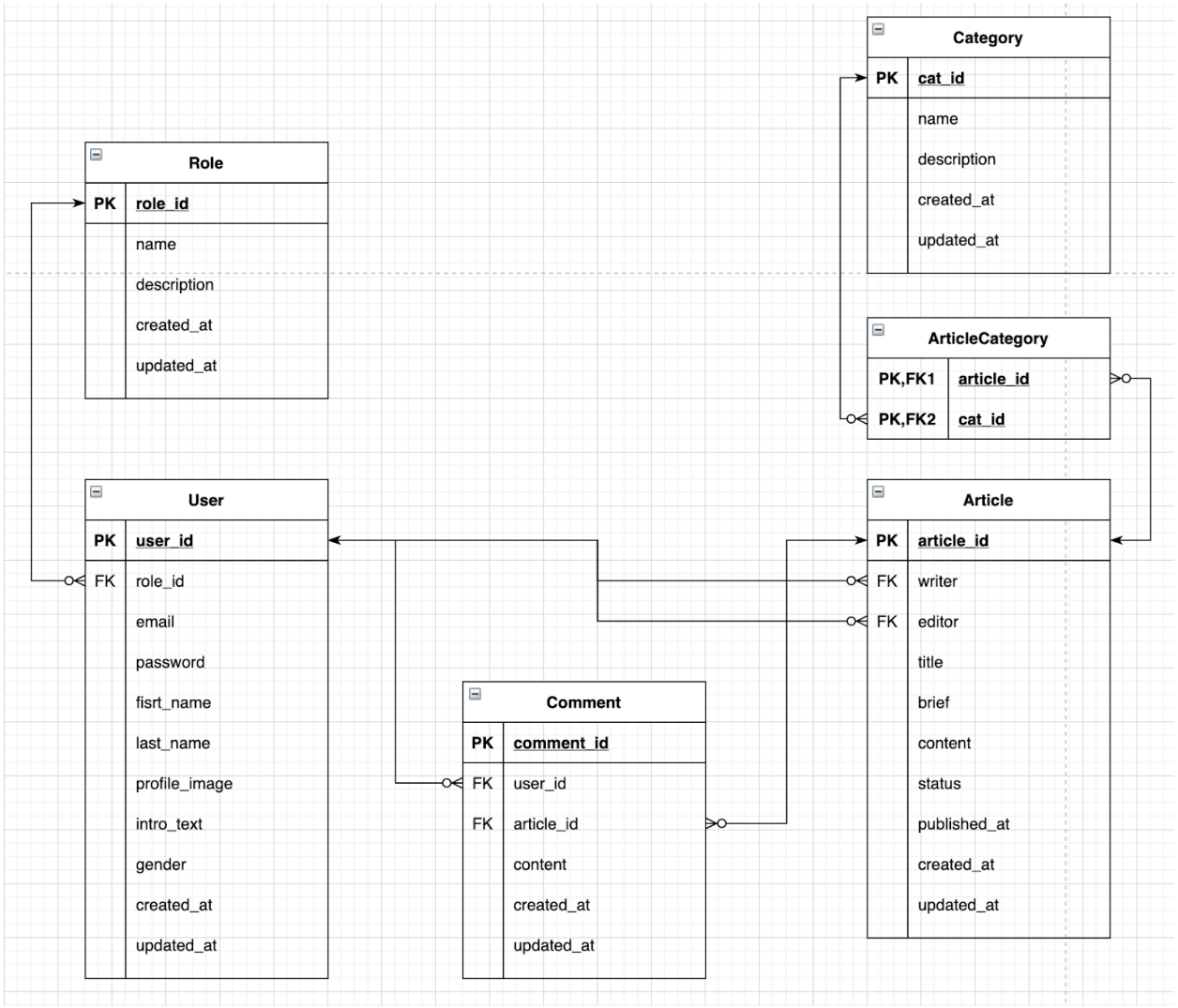
From:



Converted to:



Combine all we have:



IV. Ensure Relational Model Diagram complies with BCNF

To consider a table complies with BCNF, the determinant in the functional dependency must be unique in a table.

1. **Role**

Unique column:
- <role_id>

Function Dependency:
- role_id => name, description, created_at, updated_at

Nothing violates BCNF. Role table complies with BCNF.

2. **User**

Unique columns:
- <user_id>
- <email>

Function Dependencies:
- user_id => email, password, role_id, first_name, last_name, profile_image, intro_text, gender, created_at, updated_at
- email => user_id

Since multiple unique columns (user_id and email) do not violate BCNF. Therefore, this User table complies with BCNF.

3. **Article**

Unique column:
- <article_id>

Function Dependencies:
- article_id => writer, editor, title, brief, content, status, published_at, created_at, updated_at.

Nothing violates BCNF. Article table complies with BCNF.

4. **Category**

Unique column:
- <cat_id>

Function Dependency:
- cat_id => name, description, created_at, updated_at

Nothing violates BCNF. Category table complies with BCNF.

5. **Comment**

Unique columns:
- <user_id, article_id>

Function Dependency:
- user_id, article_id => content, created_at, updated_at

Nothing violates BCNF. Comment table complies with BCNF.

6. **ArticleCategory**

Unique columns:
- <article_id, cat_id>

Nothing violates BCNF. ArticleCategory table complies with BCNF.

V. Changing table names and final result

Since the design will be applied, and I want to follow the particular convention:

- Table name must be plural
- Table name is written in snake_case

Therefore, here are the names changed:

- Role => roles
- User => users
- Article => articles
- Comment => comments
- Category => categories
- ArticleCategory => categories_articles

Final result: