**Problem Set 5**

**Question1**
**Think about how to classify each of the types of content on your site. Articles, comments on articles, news items, questions and answers, documents to be approved or rejected with comments, and so on. Each type of content will require a different flavor of data model and moderation/approval/versioning process**

Regarding to expected functionalities of our website, initially we can consider these database tables: Reviews, Commnets_on_reviews, Questions, Answers, News.

**Reviews table (magnet content authored by users):** it should store user's reviews of watching movies. In our website the reviews don't need to be approved by admin, every registered user can post an article, but if 10 other users report it, then system will send an email to admin, and the admin will check it's content in terms of abusing the rules. Here's a very basic data model for storing reviews:

```
create table reviews (
        review_id           integer primary key,
        creation_user       not null references users, -- author user
        creation_date       not null date, -- and when the user created this review
        language            char(2) references language_codes, -- for example English is en
        mime_type           varchar(100) not null,
        one_line_summary    varchar(200) not null, -- will hold the title in most cases
        report_abuse        integer, number of reports by another users
        body                text      -- the entire review; 4 GB limit
      );
```

**Comments on reviews table (means of collaboration):**

A functionality that lets a user post an alternative perspective to a published review, that is a means of collaboration that turns our website to be more social and an on line community. Comments on reviews can be considered as a separate table:

```
create table comments_on_reviews(

        review_id           integer primary key,
        refers_to           not null references reviews,  -- on what review is  this a comment?
        creation_user       not null references users,
        creation_date       not null date,
        language            char(2) references language_codes,
        mime_type           varchar(100) not null,
        one_line_summary    varchar(200) not null,
        report_abuse        integer,
        body                text,
);
```

this table differ with reviews just in refers_to column.

**News table:**

This table is going to store news about movies in cinema . It is more like a review but this table just

differ in  release_time and expiration_time. When admin post a news item it should be mentioned about release time and also the duration that review is going to be in main page and after that it will expire and will be archived. In our website we are going to show the new movies in cinema in main page for one week and after that it will be updated.

**create table news (**

```
news_id             integer primary key,
creation_user       not null references users,
creation_date       not null date,
release_time        date,   -- NULL means "immediate", not a review
expiration_time     date,   -- NULL means "never expires"
language            char(2) references language_codes,
mime_type           varchar(100) not null,
one_line_summary    varchar(200) not null, --mostly it shows title
body                text);
```

**Questions table:**

It contains the questions asked by the users. It is same with comments_on_reviews but differ just in refers_to column that here is null**.** In our website we have only one question and answer forum.

```
create table questions (

        question_id             integer primary key,
        creation_user           not null references users,
        creation_date           not null date,
        language                char(2) references language_codes,
        mime_type               varchar(100) not null,
        one_line_summary        varchar(200) not null, --title
        body                    text
);
```

**Answer table:**

It contains the answers by the admin to users's question. It is same with Questions table.

```
create table  answers(

        answer_id               integer primary key,
        creation_user           not null references users,
        creation_date           not null date,
        language                char(2) references language_codes,
        mime_type               varchar(100) not null,
        one_line_summary        varchar(200) not null, --title
        body                    text
);
```

**Question 2:**

**Design your content data model and discuss the implications of your design choices with potential users of the system. In general, there should be at least one category of user-submitted content with authoring work flow specific to your project and at least one question and answer forum. However, the specifics will be very dependent on your project's requirements. Document the design and your discussions on your site.**

In order to make a content management system first we take a look to first two initial tables; Reviews and Commnets_on_reviews. They are similar but just differ in refers_to column in comments_on_reviews table. That is null in reviews table. So we can combine this two table in a content table :

```
create table content_raw (

    content_id          integer primary key,
    refers_to           references content_raw, -- this is null in reviws content
    creation_user       not null references users,
    creation_date       not null date,
    language            char(2) references language_codes,
    mime_type           varchar(100) not null,
    one_line_summary    varchar(200) not null,
    report_abuse        integer,
    body                text

);
```

query about reviews: we can create a view where refers_to column IS NULL.

```
create view articles

as
select *
from content_raw
where refers_to is null
```

query about commnets_on_reviews we can create another view that refers_to column IS NOT NULL.

```
create view comments_on_articles_view

as
select *
from content_raw
where refers_to is not null
```

News table has same column but has also two other columns:  release_time and expiration_time. Another difference is about report_abuse column that we dont have it in News. So we can combine this table in content table.

```
create table content_raw (

    content_id          integer primary key,
    refers_to           references content_raw,
```

```
    creation_user           not null references users,
    creation_date           not null date,
    release_time            date,   -- NULL means this content is not News
    expiration_time         date,   -- NULL means "never expires"
    language                char(2) references language_codes,
    mime_type               varchar(100) not null,
    one_line_summary        varchar(200) not null,
    report_abuse            integer, – in News it is always Null
    body                    text,


);
```

in order to distinguish between deferent content we need another column named content_type.

```
create table content_raw (

    content_id              integer primary key,
    content_type            varchar(100) not null,
    refers_to               references content,
    creation_user           not null references users,
    creation_date           not null date,
    release_time            date, -- NULL means this content is not News
    expiration_time         date, -- NULL means "never expires"
    language                char(2) references language_codes,
    mime_type               varchar(100) not null,
    one_line_summary        varchar(200) not null,
    report_abuse            integer, – in News it is always Null
    body                    text,

);
```

query about News: we can create a view where content_type is news

```
create view news_view
as
select *
from content_raw
where content_type = 'news'
and (report_abuse is null)
and (release_time is null or sysdate >= release_time)
and (expiration_time is null or sysdate  <= expiration_time)
```

about questions and answers databases we can combine them in content table and then create views where content_type is question or answer.

```
create view Questions_view
as
select *
from content_raw
where content_type = 'questions'
```

```
create view Answers_view
as
select *
from content_raw
where content_type = 'answers'
```

Question 4:
**Specify the work flow for each kind of content on your site and discuss the implications of your design choices with potential users of the system. Document the workflow design and your discussions on your site.**

Role Admin

1. Log in as admin and visit admin/roles.
2. Visit /entries/new to add new entry post.
3. Visit /report_abuse to view the reports for specific comment or entry and decide to ban user or delete the post.
4. Visit Question and answer page for editing the basic question and answer of our website. (e.g- What is Movie Diary?)
5. Can view/edit/delete all entries from all users.
6. Can view/edit/delete all comments from all users.
7. Log out

Role User

1. Log in as common user and visit /home.
2. Visit user profile to edit personal information.
3. Visit entry posts and comment.
4. Visit Now Playing to check out new movies.
5. Visit popular movies from around the world.
6. Visit Recommendation movies based on user interest.
7. Visit FB page of movie diary for Q&A forum. (interaction with admins)
8. Can view/edit/delete own entry and comment.
9. Can delete any comment of any user in own entry post.
10. Log out

**Question 5:**
**Design your versioning system. What is the versioning system for each type of content? Do you need a complete history of every version, or just a current version with the author of the last version? Make sure you think carefully about the needs of the user community. Document the versioning design on your site.**

In our website in order to make the versioning system, we should consider different contents.
For reviews we just need t keep track of current version with author user, but the user does not need to keep track of old versions. For commnets_on_reviews because it is going to be like a social website we don’t need to keep track of different version. So we don’t need versioning system.
For news content because it is going to be updated frequently so we need to keep track of different versions. We need to have current version of news and also old version of that with same auther.
For questions and answer content we don’t need versioning system.

In order to have versioning, initially we can add a column of version_number to our content table, but it will not be in second normal form. Because some fields don't depend on whole key. So we need to have another table named content_versioning.

```
create table content_versions (
        version_id              integer primary key,
        content_id              not null references content_raw,
        version_date            date not null,
        language                char(2) references language_codes,
        one_line_summary        varchar(200) not null,
        body                    text,
);
```

How to query different version of content: soppose we want to get the latest version of a content with id 1234 we can query it like this:

```
select *
from content_versions
where content_id = 1234
and version_id = (select max(version_id)
                  from content_versions
                  where content_id = 1234
                 )
```