CM6211 Database Systems

Database Systems Project Portfolio

Project Requirements and Database Design

Project Requirements

For our project we Admiral was the client. The specification for the system our client wishes us to build is a time sheet portal. The scope of the system includes the following:

- User (contractors) can create a new time sheet.
- User (managers) can manage time sheet.
- User (admin) can manage and create users and review accepted time sheets.
- Admins can assign new managers to contractors.

The entities that are within our Universe of discourse consist of:

- Contractors
- Managers
- Admins

The time sheet portal system includes the following system requirements:

- Ability for users to login into the portal and redirected to their specific page.
- Ability for users to logout of the portal.
- Ability for contractors to create a new time sheet with the form presented.
- Ability for managers to view contractors that are assigned to them.
- Ability for managers to manage contractor's time sheets for review.
- Ability for admins to create a new user.
- Ability for admins to manage managers to contractors.
- Ability for admins to review time sheet approved by managers.
- Ability to restrict access of users to other domains.
- Ability to reset user password.
- Ability to notify users.

The time sheet portal will have the following business rules applied:

- Contractors are only capable of creating a new time sheet.
- Contractors are only able to submit a new time sheet of the previous Sunday.
- Managers are only capable of reviewing the contractors that are under them.
- Admins are only capable of adding new users into the system, reviewing time sheets and changing managers of contractors.
- Notifications are specifically for each user.

Database Design

The conceptual and logical model of the database includes:

- User
    - One (and only one)-to-one or many relationships with confirmation token.
    - One (and only one)-to-one or many relationships with contractor.
    - ID as primary key.
    - Attributes includes only INT and VARCHAR.
- User Role
    - ID as primary key.
    - User ID as foreign key.
    - Attributes includes only INT and VARCHAR.
- Contractor
    - Many or one-to-one (and only one) relationship with user.
    - One or many-to-one (and only one) relationship with agency.
    - One (and only one)-to-one or many relationships with timesheet.
    - ID as primary key.
    - User ID and agency ID as foreign keys.
    - Attributes includes only INT.
- Agency
    - One (and only one)-to-one or many relationships with contractor.
    - ID as primary key.
    - Attributes includes only INT and VARCHAR.
- Timesheet

    - One or many-to-one (and only one) relationship with contractor.
    - One (and only one)-to-one or many relationships with review.
    - ID as primary key.
    - Contractor ID as foreign key.
    - Attributes includes only INT and DATE.
- Review
    - One or many-to-one (and only one) relationship with timesheet.
    - ID as primary key.
    - Timesheet ID as foreign key.
    - Attributes include only INT, BIT, and VARCHAR.
- Confirmation Token
    - One or many-to-one (and only one) relationship with user.
    - Token ID as primary key.
    - User ID as foreign key.
    - Attributes include only INT, VARHCAR and DATE.
- Timestamps
    - No relationship with other entities.
    - No primary key or foreign key.
    - Attributes include only TIMESTAMP.
- Category
    - No relationship with other entities.
    - Category ID as primary key.
    - Attributes include only INT and VARCHAR.
- User (login user)
    - No relationship with other entities.
    - No primary key or foreign key.
    - Attributes include only VARCHAR and TIMESTAMP.
- Hibernate sequence
    - No relationship with other entities.
    - No primary key or foreign key.
    - Attribute is BIGINT.

The schema we developed has the tables; User, User Role, Contractor, Agency, Timesheet, and Review. These specific tables were necessary for the functionality of the product, as they not only work for our current system but allow for future customer requests to be easily implemented.

User records the normalized login information for each user, their username, password, name and email. All this is non-nullable as it is essential for the system. User Role references this table's IDs to associate each user with a role, made a separate table as in 4th Normal Form.

The contractor is a table of information specific only to contractor users, separating it from User by referencing User ID. This table includes the Agency of the contractor. It was made this way so that in the future, if a contractor changed agency, the Contractor record could be kept the same, for past records and documentation of the association with that agency, but a new Contractor record could be made for the user, allowing them to use the same login info and removing redundant data.

Agency is a table of information specific only to an agency a contractor will work for. This is referenced in the contractor table as previously mentioned by agency id to create this connection and allow an admin to know whom they need to invoice to which agencies. By storing agencies information, duplicate data is not made when new contractors from the same agencies are added. The process of assigning is also much quicker.

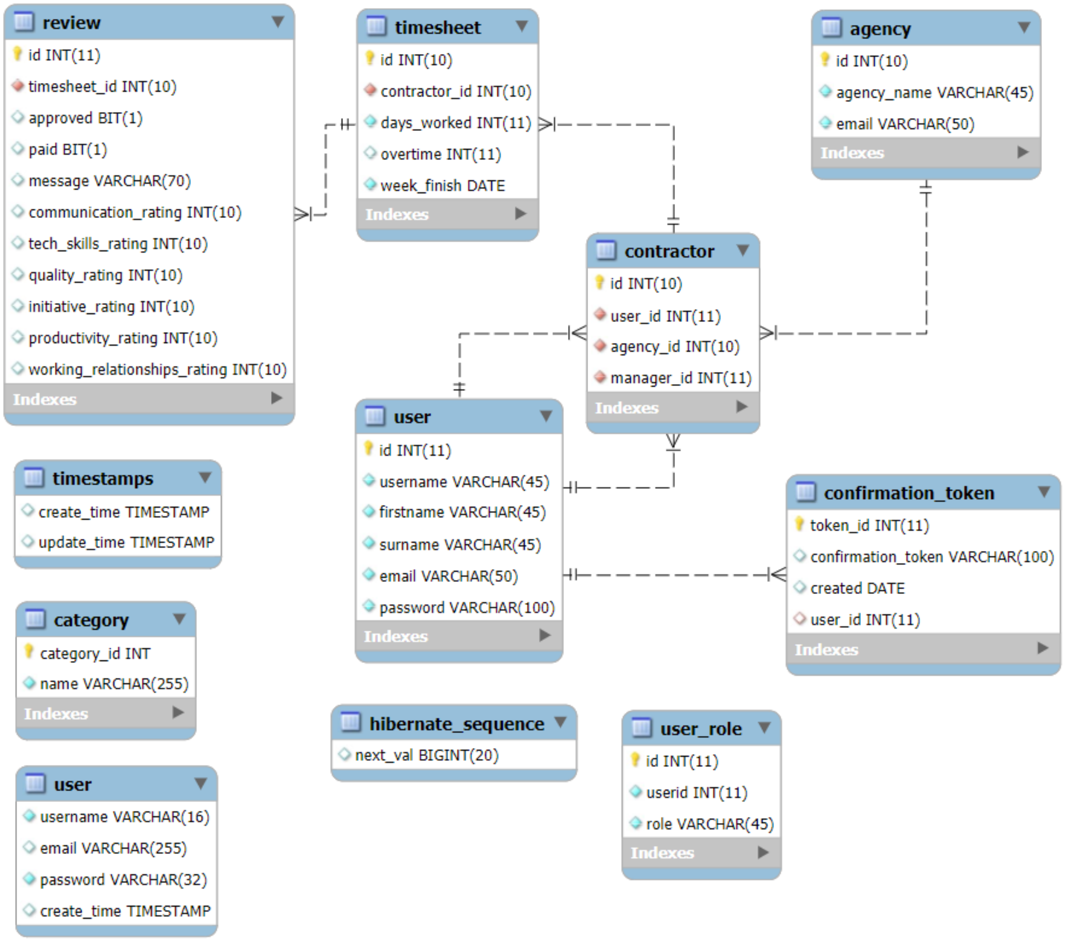
Timesheet is a table to store the timesheet information a contractor has made. This mainly uses integers and dates, due to it keeping track of the amount of time in the week worked. Most of the information is crucial for the system, hence nearly all fields being nullable. This references the contractor who made it by a foreign key 'contractor_id', which enables others like admins to know whom they need to pay. This also ensures managers only receive sheets for contractors they are under to make knowing who they need to review and either approve or reject much easier, and why a timesheet id is referenced in the review table as previously mentioned.

The review is a table formatted to store existing information and separate data during validation to help with processing requests, this table is aimed towards the managers as they will be the ones who authorize which timesheets are valid to move on in the next stage of processing. The table references a 'timesheet_id' through a "Foreign Key" which is required to parse on the timesheet information. There are additional columns for "rating" marked with an integer value.

Following this, a message field for processing feedback on the review form itself was included to take allow managers to give feedback during the timesheet review approval process. Another is the 'paid' &

‘approved’ they are signed with a BIT which is relative to Boolean “1, 0 or null” this incorporates Transact SQL where the value here is null until the manager decides that he will either “approve”, ‘1’ or “reject”, ‘0’. In turn, this allows managers to approve and or deny requests attaching a small report.

ER Diagram


review
id INT(11)
timesheet_id INT(10)
approved BIT(1)
paid BIT(1)
message VARCHAR(70)
communication_rating INT(10)
tech_skills_rating INT(10)
quality_rating INT(10)
initiative_rating INT(10)
productivity_rating INT(10)
working_relationships_rating INT(10)
Indexes
timesheet
id INT(10)
contractor_id INT(10)
days_worked INT(11)
overtime INT(11)
week_finish DATE
Indexes
agency
id INT(10)
agency_name VARCHAR(45)
email VARCHAR(50)
Indexes
contractor
id INT(10)
user_id INT(11)
agency_id INT(10)
manager_id INT(11)
Indexes
timestamps
create_time TIMESTAMP
update_time TIMESTAMP
user
id INT(11)
username VARCHAR(45)
firstname VARCHAR(45)
surname VARCHAR(45)
email VARCHAR(50)
password VARCHAR(100)
Indexes
confirmation_token
token_id INT(11)
confirmation_token VARCHAR(100)
created DATE
user_id INT(11)
Indexes
category
category_id INT
name VARCHAR(255)
Indexes
hibernate_sequence
next_val BIGINT(20)
user_role
id INT(11)
userid INT(11)
role VARCHAR(45)
Indexes
user
username VARCHAR(16)
email VARCHAR(255)
password VARCHAR(32)
create_time TIMESTAMP