Описание

Электронный журнал

Проект по созданию электронного журнала с применением базы данных.

Основная задача:

Создание необходимого функционала для информирования учеников (или родителей) об их успеваемости, а также создание необходимых условий для дистанционного редактирования данных учителем. В связи с этим необходимо перенять лучшие качества у бумажного варианта журнала и оптимизировать их для пользователей.

Данные

User:

id_user: INT, primary key, not null, unique index, unsigned data type, auto incremental;

fio: TEXT, not null;

isteacher: TINYINT(1);

email: TEXT, not null;

password: TEXT, not null;

User_has_Class:

id_user_has_class: INT, primary key, not null, unique index, unsigned data type, auto incremental;

User_id_user: INT, primary key, not null, unsigned data type, foreign key (referenced table `User` column `id_user`);

Class_id_class: INT, primary key, not null, unsigned data type, foreign key (referenced table `Class` column `id_class`);

Class:

id_class: INT, primary key, not null, unique index, unsigned data type, auto incremental;

class_name: VARCHAR(32), not null;

User_has_Subject:

id_user_has_subject: INT, primary key, not null, unique index, unsigned data type, auto incremental;

User_id_user: INT, primary key, not null, unsigned data type, foreign key (referenced table `User` column `id_user`);

Subject_id_subject: INT, primary key, not null, unsigned data type, foreign key (referenced table `Subject` column `id_subject`);

Subject:

id_subject: INT, primary key, not null, unique index, unsigned data type, auto incremental;

subject_name: TEXT, not null;

Marks:

id_mark: INT, primary key, not null, unique index, unsigned data type, auto incremental;

User_id_user: INT, primary key, not null, unsigned data type, foreign key (referenced table `User` column `id_user`);

mark: CHAR(1);

Lesson_id_lesson: INT, primary key, not null, unsigned data type, foreign key (referenced table `Lesson` column `id_lesson`);

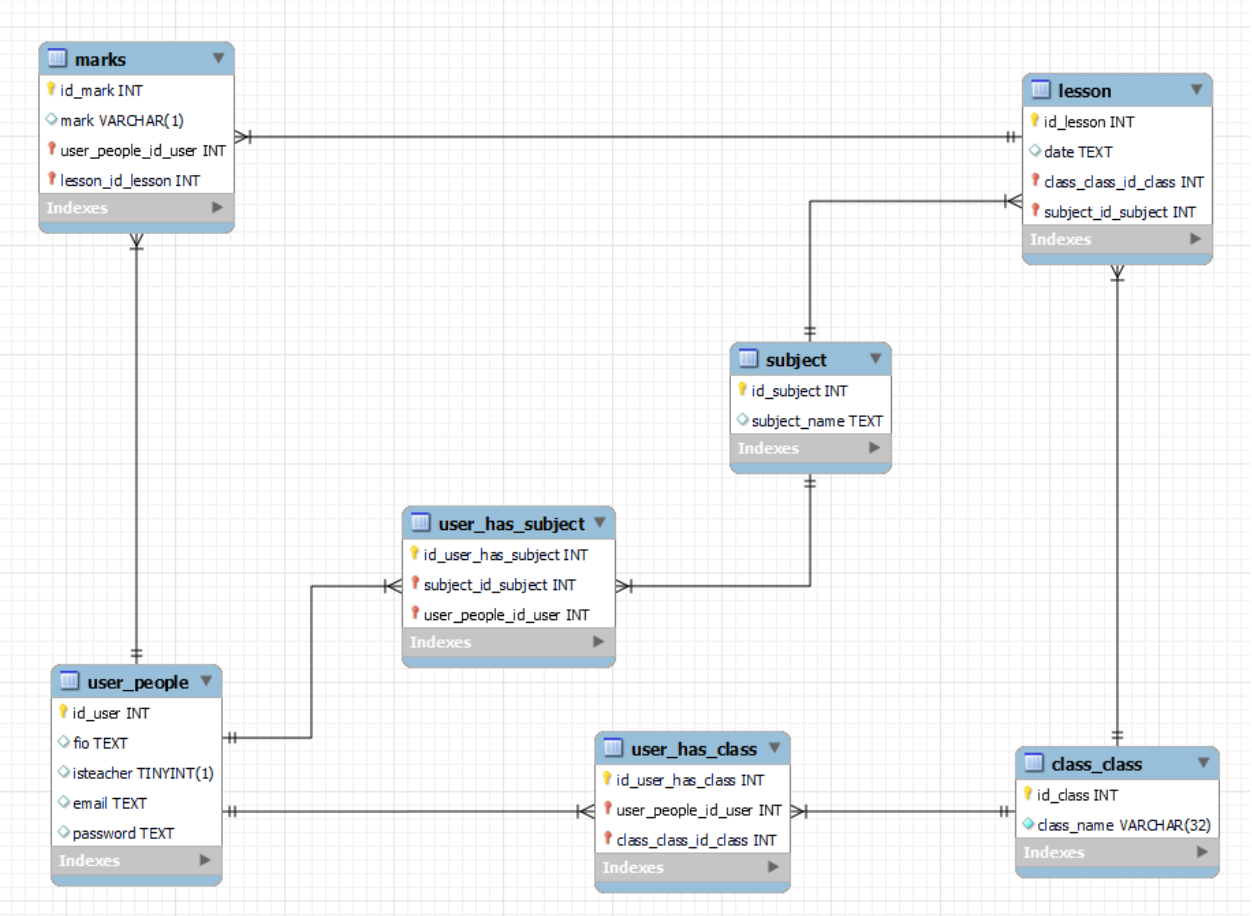
Lesson:

id_lesson: INT, primary key, not null, unique index, unsigned data type, auto incremental;

date: TEXT;

Class_id_class: INT, primary key, not null, unsigned data type, foreign key (referenced table `Class` column `id_class`)

Subject_id_subject: INT, primary key, not null, unsigned data type, foreign key (referenced table `Subject` column `id_subject`);



Пользовательские роли

В данном проекте задумываются 3 пользовательские роли: ученик, учитель и администратор.

Для каждой из них будут реализованы следующие функции:

Админ: 1) добавление учеников

2) добавление предметов

3) добавление учителей

4) добавление классов

5) перераспределение вышеперечисленного

Ученик: 1) выбор предмета

2) выбор промежутка времени

3) просмотр оценок

4) просмотр посещаемости

Учитель: 1) выбор предмета

2) выбор промежутка времени

3)выбор класса

4) просмотр оценок

5) просмотр посещаемости

6) возможность внесения изменений в графе оценок/посещаемости

Т.е. различием между ролями будет возможность изменения/добавления данных.

API

- GET api/user_people - получить всех пользователей из базы данных
- GET api/user_people/id_user - получить пользователя с id=id_user
- POST api/user_people - занести в базу данных пользователя и информацию о нем
- PUT api/user_people/id_user - изменить в базе данных информацию о пользователе
- DELETE api/user_people/id_user - удалить из базы данных пользователя с id=id_user (также из соответствующих таблиц базы данных каскадно удаляется вся информация, связанная с этим пользователем)

- GET api/class_class - получить все классы из базы данных
- GET api/class_class/id_class - получить класс с id=id_class
- POST api/class_class - занести в базу данных класс и информацию о нем
- PUT api/class_class/id_class - изменить в базе данных информацию о классе
- DELETE api/class_class/id_class - удалить из базы данных класс с id=id_class (также из соответствующих таблиц базы данных каскадно удаляется вся информация, связанная с этим классом)

- GET api/user_has_class - получить все связи ученика и класса из базы данных
- GET api/user_has_class/id_user_has_class - получить связь ученика и класса с id=id_user_has_class

- POST api/user_has_class - занести в базу данных связь ученика и класса и информацию о ней
- PUT api/user_has_class/id_user_has_class - изменить в базе данных информацию о связи ученика и класса
- DELETE api/user_has_class/id_user_has_class - удалить из базы данных связи ученика и класса с id=id_user_has_class

- GET api/subject - получить все предметы из базы данных
- GET api/subject/id_subject - получить предмет с id=id_subject
- POST api/subject - занести в базу данных предмет и информацию о нем
- PUT api/subject/id_subject - изменить в базе данных информацию о предмете
- DELETE api/subject/id_subject - удалить из базы данных предмет с id=id_subject (также из соответствующих таблиц базы данных каскадно удаляется вся информация, связанная с этим предметом)

- GET api/user_has_subject - получить все связи ученика и предмета из базы данных
- GET api/user_has_subject/id_user_has_subject - получить связь ученика и прдемета с id=id_user_has_subject
- POST api/user_has_subject - занести в базу данных связь ученика и предмета и информацию о ней
- PUT api/user_has_subject/id_user_has_subject - изменить в базе данных информацию о связи ученика и предмета
- DELETE api/user_has_subject/id_user_has_subject - удалить из базы данных связи ученика и предмета с id=id_user_has_subject

- GET api/lesson - получить все уроки из базы данных
- GET api/lesson/id_lesson - получить урок с id=id_lesson
- POST api/lesson - занести в базу данных урок и информацию о нем
- PUT api/lesson/id_lesson - изменить в базе данных информацию об уроке
- DELETE api/lesson/id_lesson - удалить из базы данных урок с id=id_lesson (также из соответствующих таблиц базы данных каскадно удаляется вся информация, связанная с этим уроком)

- GET api/marks - получить все оценки из базы данных
- GET api/marks/id_mark - получить оценку с id=id_mark
- POST api/marks - занести в базу данных оценку и информацию о ней
- PUT api/marks/id_mark - изменить в базе данных информацию об оценке
- DELETE api/marks/id_mark - удалить из базы данных оценку с id=id_mark

- Проверка login и password на совпадение введенных значений с базой данных на сервере;
- После получения id_user проверяем, является ли пользователь учителем (isTeacher);
- По значению id_user можем получить информацию о классе в котором учится ученик (или классы, в которых проводит занятия учитель);

- По id_class можем получить информацию о всех учениках в этом классе, и, соответственно, предметы, которые изучаются в этом классе (для учителей только присущие им предметы);
- Наличие информации о предметах и классе предоставляет доступ к информации о расписании (датах проведения занятий по выбранному предмету в выбранном классе);
- Наличие информации о расписании и ученике предоставляет доступ к информации об оценках;

Языки программирования

1 JavaScript

2 Html

3 Css

4 SQL

Субд

1 Реляционная mysql