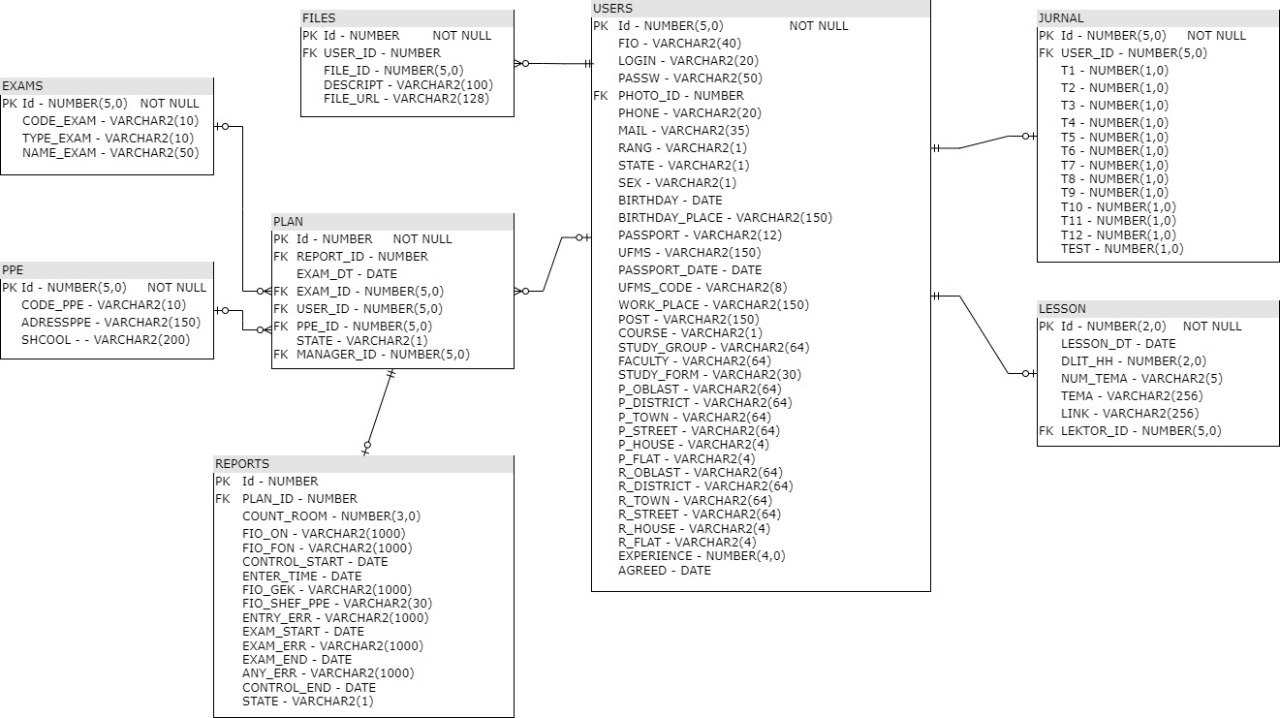
3. Проектно- конструкторская часть

3.1 Логическая схема базы данных



3.2 Описание сущностей базы данных

«EXAMS» - сущность экзаменов. Имеет поля:

Id - NUMBER(5,0) NOT NULL - искусственный первичный ключ

CODE\_EXAM - VARCHAR2(10) - Код экзамена

TYPE\_EXAM - VARCHAR2(10) - Тип экзамен(ЕГЭ, ОГЭ и тд)

NAME\_EXAM - VARCHAR2(50) - Название экзамена

«PPE» - сущность пунктов проведения экзаменов. Имеет поля:

Id - NUMBER(5,0) NOT NULL - искусственный первичный ключ

CODE\_PPE - VARCHAR2(10) - код пункта проведения экзамена

ADRESSPPE - VARCHAR2(150) - адрес пункта проведения экзамена

SHCOOL - VARCHAR2(200) - название общеобразовательного учреждения

«REPORTS» - сущность отчетов. Имеет поля:

Id – NUMBER – искусственный первичный ключ

PLAN\_ID – NUMBER – внешний ключ в поле Id таблицы PLAN

COUNT\_ROOM - NUMBER(3,0) – количество аудиторий

FIO\_ON - VARCHAR2(1000) – ФИО общественных наблюдателей

FIO\_FON - VARCHAR2(1000) – ФИО федеральных общественных наблюдателей

CONTROL\_START – DATE - время начала контроля

ENTER\_TIME – DATE – время прихода наблюдателя в ППЭ

FIO\_GEK - VARCHAR2(1000) – ФИО ГЭК

FIO\_SHEF\_PPE - VARCHAR2(30) – ФИО начальника ППЭ

ENTRY\_ERR - VARCHAR2(1000) – Нарушения на входе в ППЭ

EXAM\_START – DATE – Время начала экзамена

EXAM\_ERR - VARCHAR2(1000) – Нарушения во время проведения экзамена

EXAM\_END – DATE – Время окончания экзамена

ANY\_ERR - VARCHAR2(1000) – Иные нарушения

CONTROL\_END – DATE – время окончания наблюдения

STATE - VARCHAR2(1) – состояние отчета

«PLAN» - сущность записей плана проверок ППЭ. Имеет поля:

Id – NUMBER – искусственный первичный ключ

REPORT\_ID – NUMBER – внешний ключ в поле Id таблицы REPORTS

EXAM\_DT – DATE – дата проведения экзамена

EXAM\_ID - NUMBER(5,0) – внешний ключ в поле Id таблицы EXAMS

USER\_ID - NUMBER(5,0) - внешний ключ в поле Id таблицы USERS

PPE\_ID - NUMBER(5,0) - внешний ключ в поле Id таблицы PPE

STATE - VARCHAR2(1) - состояние

MANAGER\_ID - NUMBER(5,0) - внешний ключ в поле Id таблицы USERS

«FILES» - сущность файлов. Имеет поля:

Id – NUMBER - искусственный первичный ключ

USER\_ID – NUMBER – внешний ключ в поле Id таблицы USERS

FILE\_ID - NUMBER(5,0) – идентификатор файла

DESCRIPT - VARCHAR2(100) – описание файла

FILE\_URL - VARCHAR2(128) – адрес файла в файловой системе

«USERS» - сущность пользоватлей. Имеет поля:

Id - NUMBER(5,0) - искусственный первичный ключ

FIO - VARCHAR2(40) – ФИО пользователя

LOGIN - VARCHAR2(20) – Логин пользователя

PASSW - VARCHAR2(50) – Пароль пользователя

PHOTO\_ID – NUMBER – внешний ключ в поле Id таблицы FILES

PHONE - VARCHAR2(20) – номер телефона пользователя

MAIL - VARCHAR2(35) – email пользователя

RANG - VARCHAR2(1) – ранг пользователя внутри системы

STATE - VARCHAR2(1) – состояние пользователя

SEX - VARCHAR2(1) - пол

BIRTHDAY – DATE – дата рождения пользователя

BIRTHDAY\_PLACE - VARCHAR2(150) – место рождения

PASSPORT - VARCHAR2(12) – серия и номер паспорта

UFMS - VARCHAR2(150) – кем выдан паспорт

PASSPORT\_DATE – DATE – дата выдачи паспорта

UFMS\_CODE - VARCHAR2(8) – код подразделения, выдавшего паспорт

WORK\_PLACE - VARCHAR2(150) – место работы/ учебы

POST - VARCHAR2(150) – занимаемая должность

COURSE - VARCHAR2(1) – курс обучения

STUDY\_GROUP - VARCHAR2(64) – группа обучения

FACULTY - VARCHAR2(64) - факультет

STUDY\_FORM - VARCHAR2(30) – форма обучения

P\_OBLAST - VARCHAR2(64) – облась проживания по прописке

P\_DISTRICT - VARCHAR2(64) – район проживания по прописке

P\_TOWN - VARCHAR2(64) – город проживания по прописке

P\_STREET - VARCHAR2(64) – улица проживания по прописке

P\_HOUSE - VARCHAR2(4) – дом проживания по прописке

P\_FLAT - VARCHAR2(4) – квартира проживания по прописке

R\_OBLAST - VARCHAR2(64) – область фактического проживания

R\_DISTRICT - VARCHAR2(64) – район фактического проживания

R\_TOWN - VARCHAR2(64) – город фактического проживания

R\_STREET - VARCHAR2(64) – улица фактического проживания

R\_HOUSE - VARCHAR2(4) – дом фактического проживания

R\_FLAT - VARCHAR2(4) – квартира фактического проживания

EXPERIENCE - NUMBER(4,0) – год вступления в КОН

AGREED – DATE – согласие на обработку персональных данных

«LESSON» - сущность занятий. Имеет поля:

Id - NUMBER(2,0) - искусственный первичный ключ

LESSON\_DT – DATE – дата и время проведения занятия

DLIT\_HH - NUMBER(2,0) – длительность занятия

NUM\_TEMA - VARCHAR2(5) – номер темы

TEMA - VARCHAR2(256) – название темы

LINK - VARCHAR2(256) – ссылка на сторонний ресурс

LEKTOR\_ID - NUMBER(5,0) – внешний ключ в поле Id таблицы USERS

«JURNAL» - сущность журнала. Имеет поля:

Id - NUMBER(5,0) – искусственный первичный ключ

USER\_ID - NUMBER(5,0) ) – внешний ключ в поле Id таблицы USERS

T1 - NUMBER(1,0) – оценка за занятие 1

T2 - NUMBER(1,0) – оценка за занятие 2

T3 - NUMBER(1,0) – оценка за занятие 3

T4 - NUMBER(1,0) – оценка за занятие 4

T5 - NUMBER(1,0) – оценка за занятие 5

T6 - NUMBER(1,0) – оценка за занятие 6

T7 - NUMBER(1,0) – оценка за занятие 7

T8 - NUMBER(1,0) – оценка за занятие 8

T9 - NUMBER(1,0) – оценка за занятие 9

T10 - NUMBER(1,0) – оценка за занятие 10

T11 - NUMBER(1,0) – оценка за занятие 11

T12 - NUMBER(1,0) – оценка за занятие 12

TEST - NUMBER(1,0) – оценка за экзамен

**3.3 Sequence**

Из-за отсутствия в базе данный автоинкрементируемых полей, для генерации идентификаторов к каждой таблице создана последовательнось - структура для генерации уникальных целочисленных значений.

Для таблицы «EXAMS»

***CREATE SEQUENCE "SEQ\_EXAMS" MINVALUE 1 MAXVALUE 999999999999999999999999999 INCREMENT BY 1 START WITH 103 CACHE 20 NOORDER NOCYCLE***

Поледовательность от 1 до 999999999999999999999999999 с шагом в 1, начинается со значения 103, заранее создается 20 значений, для ускорения работы, параметр NOORDER указывает, что генерируемые значения не обязательно будут идти по порядку, последовательность не циклична, что приведет к ошибке при попытке запросить новый член последовательности при достижении максимального значения.

3.4 Хранимые процедуры и функции

Для повышения производительности и безопасности в проекте используются хранимые процедуры и функции. Производительность увеличивается за счет того, что хранимые процедуры и функции компилируются заранее и план выполнения создаетя один раз, а затем повторно использует этот план выполнения. Это приводит к повышению производительности в случае многократного вызова хранимой процедуры, так как обработчику запросов не нужно создавать новый план и поэтому обработка процедуры или функции занимает меньше времени.

Для сохранения файлов используется процедура:

***create or replace PROCEDURE save\_user\_file(p\_user\_id in number, p\_descript in VARCHAR2, p\_FileName in VARCHAR2, f\_id out number) as ST\_SQL varchar2(1000) := 'Insert into files values (SEQ\_Files.nextval,null,:1,:2,:3) returning ID into :4'; begin execute immediate ST\_SQL using p\_user\_id, p\_descript, p\_FileName, out f\_id; end save\_user\_file;***

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |

В поцедуре используются 3 входных параметра и один выходной, в блоке begin выполняется динамический SQL запрос, параметры в который подставляются последовательно после ключевого слова **using.** В выходной параметр записывается идентификатор файла последством **returning**.

Для аутентификации используется функция с возвращаемым значением типа boolean.

create or replace function password\_auth (p\_username in VARCHAR2, p\_password in VARCHAR2) return BOOLEAN is

l\_password varchar2(50); l\_stored\_password varchar2(50);

l\_count number;

begin select count(\*) into l\_count from users where login = p\_username and State = '1'; if l\_count > 0 then select passw into l\_stored\_password from users where login = p\_username;

l\_password := password\_hash(p\_username, p\_password);

if l\_password = l\_stored\_password then return true; else return false;

end if;

else return false;

end if;

end;

Получаем на вход логин и пароль пользователя, проверяем на наличие в базе данных пользователей с таким логином и активным состоянием, если такие есть, то вычисляем hash из полученного пароля и сравниваем его с паролем, хранимым в базе данных. Если пароли совпадают – возвращаем true, если нет – false. False вернется и в том случае, если пользователей с таким логином не найдено.

Вызываемая password\_hash – тоже хранимая функция.

3.5 Триггеры

В базе данных также используются триггеры, некоторые для поддержания целостности базы данных, некоторые из них для реализации логики. Одним из триггеров, используемых для реализаци логики является триггер «TR\_EXAMS». Он предназначен для получения первичного ключа и сохранения его в запись перед выполнением DML оператора:

***CREATE OR REPLACE TRIGGER "TR\_EXAMS"***

***before insert on EXAMS***

***for each row***

***begin***

***select SEQ\_EXAMS.nextval into :new.ID from dual;***

***end TR\_EXAMS;***

***/***

***ALTER TRIGGER "TR\_EXAMS" ENABLE***

***/***

Для простоты аутентификации используется триггер «TR\_USERS\_2»

***CREATE OR REPLACE TRIGGER "TR\_USERS\_2"***

***BEFORE***

***insert or update on USERS***

***for each row***

***begin***

***:NEW.LOGIN := upper(:NEW.LOGIN);***

***end TR\_Users\_2;***

***/***

***ALTER TRIGGER "TR\_USERS\_2" ENABLE***

***/***

Он преобразует логин в верхний регистр. При аутентификации введенный логин тоже преобразуется в верхний регистр.

3.6 Нормализация базы данных

База данных соответствует третьей нормальной форме ???????????.