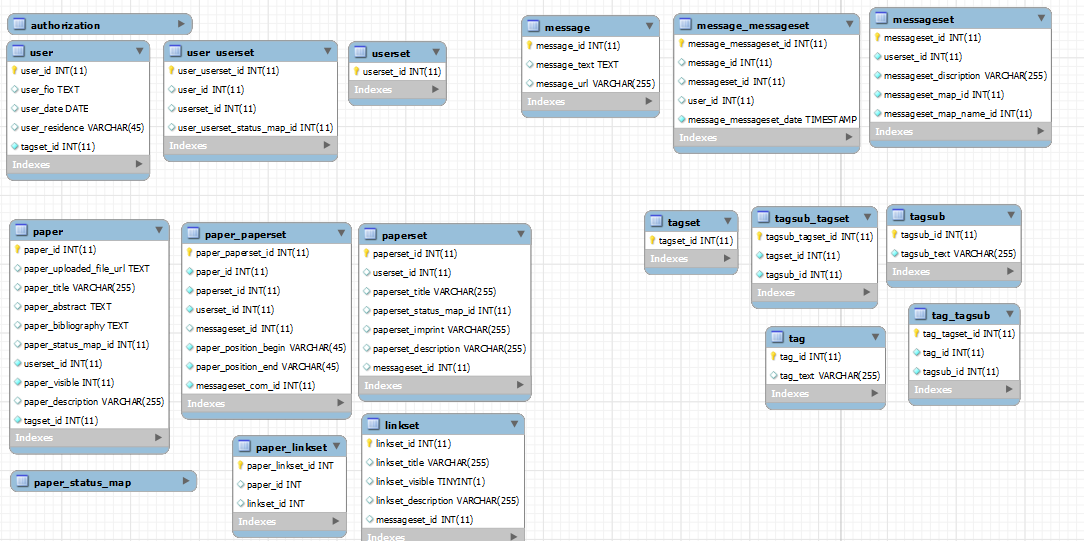
**Описание структуры БД:**

****

!!!Изображены не все таблицы (только те которые нам понадобятся для альфа запуска)

Данная БД разрабатывается для Электронно-научного журнала.

Весь проект реализован с помощью объектно-ориентированного программирования.

К каждому объекту в нашей системе есть определенный доступ, который зависит от ролей пользователей. В нашей системе рассматривается 5 ролей

1. Автор(автор статьи)
2. Эксперт
3. Владелец(создатель журнала)
4. Создатель группы сообщений
5. Обычный пользователь(данная роль требуется для комментариев)

!!!!5-я роль под сомнением

В данной системе рассматриваются несколько объектов(все приведенные объекты сложные, т.е в них входят другие объекты):

1. Paper(статья)
2. Paper\_paperset(связка paper и paperset) – данную связь рассматриваем в виде объекта, потому что в нее входят другие объекты при этом создавая более сложный объект
3. Paperset(журнал)
4. Messageset(группа сообщений)
5. Event(подсобытие)
6. Eventset(событие)
7. Stand(значение по умолчанию) требуется для создания обычной переписки пользователей, т.е если объект messageset не прикреплен ни к какому объекту.

Статья(paper) в нашем проекте имеет статусы(От этих статусов будет ясно на какой стадии она находится):

1. В разработке(developing)
2. На экспертизе(examination)
3. Подтверждена(confirmed)
4. Опубликована(published)

**Рассмотрим таблицы БД.**

Где сперва будет описание самой таблицы а затем будут рассмотрены поля таблицы. В данном описании таблиц опущены поля типа auto increment.

Таблица **authorization** хранит информацию об авторизационных данных каждого пользователя.

* user\_id(int) – id пользователя
* authorization\_login(varchar(255)) – логин пользователя
* authorization\_password(varchar(255)) – пароль пользователя

Таблица **user** хранит информацию о пользователе

* User\_name(varchar(255))- имя пользователя
* User\_surneme(varchar(255)) – фамилия пользователя
* User\_ patronymic(varchar(255)) – отчество пользователя
* User\_date(date) – дата рождения пользователя
* User\_residence(varchar(255)) – место проживания пользователя
* User\_gender(varchar(10)) – пол
* User\_education(varchar(255)) - образование
* Tagset\_id(varchar(255)) – id группы тегов прикрепленных к данному пользователю

!!!Пока плохо продумано какие поля будут у пользователя

Таблица **userset** требуется для группировки пользователей(подробнее система группировки будет рассмотрена ниже)

Таблица **user\_userset** – таблица связка м/у таблицами user и userset

* User\_id(int) – id пользователя
* Userset\_id(int) – id группы пользователей к которой прикреплен пользователь
* User\_userset\_status\_map\_id(int) – хранит информацию роли пользователя в группе

Таблица **message** хранит текст и прикрепленные фалы сообщения

* message\_text(text) – текст сообщения
* message\_url(varchar(255)) – прикрепленный файл

Таблица **messageset** группирует сообщения

* messageset\_title(varchar(255)) – название группы
* userset\_id(int) – прикрепленная группа пользователей
* messageset\_ description(varchar(255)) – описание группы сообщений

Таблица **message\_messageset** – таблица связка м/у таблицей message и messageset

* Message\_id(int) – id сообщения
* Messageset\_id(int) – id группы к которому прикрепляется сообщение
* message\_messageset\_data(timestamp) – хранится время и дата отправки сообщения(данное время задается автоматически, потому что значение по умолчанию стоит CURRENT\_TIMESTAMP)
* user\_id(int) – id отправителя сообщения

Таблица **user\_message\_read** – хранит информацию о прочитанных сообщениях пользователя

* User\_id(int) – id пользователя
* Messageset\_id(int) – id группы сообщений к которой прикреплен пользователь
* user\_message\_read\_date (timestamp)– хранит информацию о последнем посещении пользователя(данное время задается автоматически, потому что значение по умолчанию стоит CURRENT\_TIMESTAMP)

Таблица **paper** хранит информацию о статье

* Paper\_uploaded\_file\_url(varchar(255)) – ссылка на статью
* Paper\_title(varchar(255)) – название статьи
* Paper\_abstract(text)
* Paper\_bibliography(text) – Хранится список литературы
* Paper\_status\_map\_id(int) – хранится статус статьи(статусы статьи будут рассмотрены ниже)
* Userset\_id(int) – id прикрепленной группы пользователей(в данной группе пользователей могут быть только пользователи роли автор)
* Paper\_discription(varchar(255)) – описание статьи
* Tagset\_id(int) – id группы тегов прикрепленных к данной статье

Таблица **paperset** группирует статьи образовывая журналы

* Userset\_id (int)– id группы пользователей(пользователи роли владелец, эксперт, юзер)
* Paperset\_title(varchar(255)) – название журнала
* Paperset\_status\_map\_id (int)– id статуса журнала
* Paperset\_imprint(varchar(255)) –
* Paperset\_discription(varchar(255)) – описание журнала
* Messageset\_id - id группы сообщений(тут служат комментариями)

Таблица **Paper\_paperset** – таблица связка м/у таблицами paper и paperset, так же служит для экспертизы статьи

* Paper\_id(int) – id статьи
* Paperset\_id(int) – id журнала к которому прикреплена стаья
* Userset\_id(int) – id группы пользователей(только пользователи роли эксперт) относящихся к статье
* Messageset\_id(int) – id группы сообщений(данная группа сообщений служит для экспертизы статьи)
* Messageset\_com\_id(id) – id группы сообщений(данная группа сообщений служит для комментариев)
* Paper\_position\_begin(varchar(255)) – хранит информацию о начальной странице в журнале
* Paper\_position\_end(varchar(255)) – хранит информацию о конечной странице в жунале

Таблица **paper \_status\_map** хранит в себе этапы разработки статьи

* Paper\_status\_map\_name(varchar(45)) – название этапа разработки

Таблица **tag** хранит в себе общие разделы тегов

* tag\_text(varchar(255)) – название раздела тегов

Таблица **tagsub** хранит в себе все теги

* tag\_text(varchar(255)) – название тега

Таблица **tag\_tagsub** требуется для того чтобы понять, какой тег к какому разделу тегов относится

* tag\_id(int) – id раздела тегов
* tagsub\_id – id тега

Таблица **tagset** требуется для группировки тегов

Таблица **tagsub\_tagset** требуется для связки тегов с группой тегов

* tagsub\_id(int) – id тега
* tagset\_id(int) – id группы тегов

Таблица **linkset** хранит информацию о личной коллекции статей пользователя

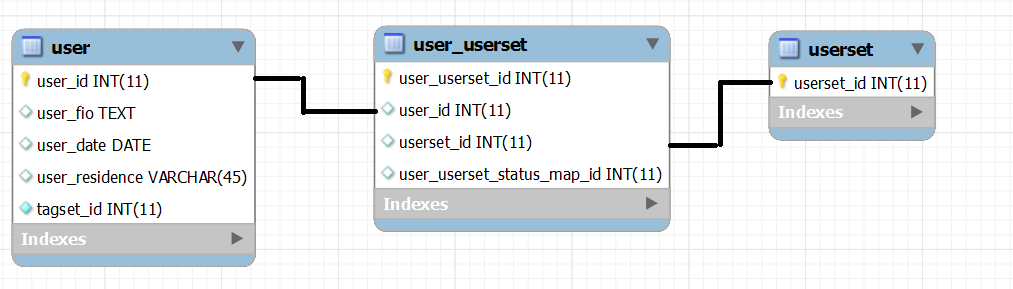
* Linkset\_title(varchar(255)) – название коллекции
* Linkset\_visible(int) – видимость личной коллекции другим пользователям
* Linkset\_discription(varchar(255)) – описание личной коллекции
* Messageset\_id(int) – id группы сообщений(тут хранится в виде комментариев)

Таблица **type\_object\_map** хранит в себе типы обьетов

* Type\_object\_map\_text(varchar(45)) – название типа обьекта(обращение к типам происходит по их id-шникам)

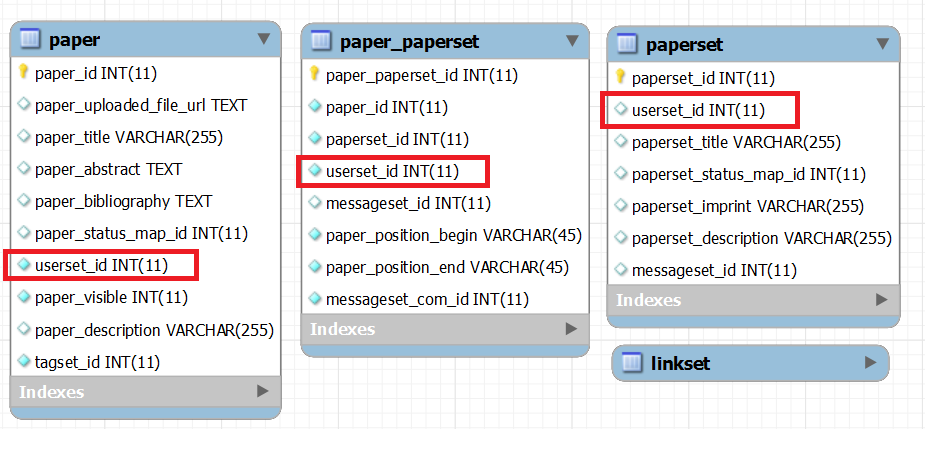
**Система группировки пользователей**

Ниже представлены таблицы участвующие в данной системе



Все пользователи нашего электронно-научного журнала хранятся в таблице user. С помощью таблицы **user\_userset** мы можем группировать пользователей. Идентификаторы групп находятся в таблице **userset**.

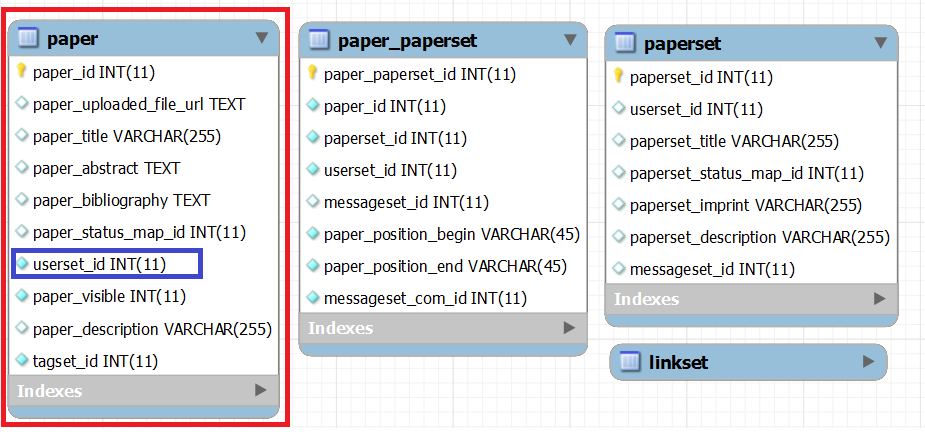
Данная система используется для того чтобы мы могли объект «группа пользователей» прикреплять к любому другому объекту. Доступ к объекту группа пользователей мы можем получить из объекта к которому они прекреплены т.е если мы хотим узнать какие пользователи у нас прикреплены к объекту, то мы обращаемся сперва к самому объекту и ищем в нем поле **userset\_id**.



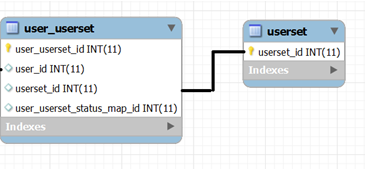
С помощью этого идентификатора из таблицы **userset** и таблицы связи **user\_userset** мы узнаем какие пользователи у нас прикреплены к объекту.

Цепочка поиска пользователей.

1. Выбираем объект (например paper)



1. Далее выбираем идентификатор группы пользователей **userset\_id** и с помощью него ссылаемся на таблицу **user\_userset**

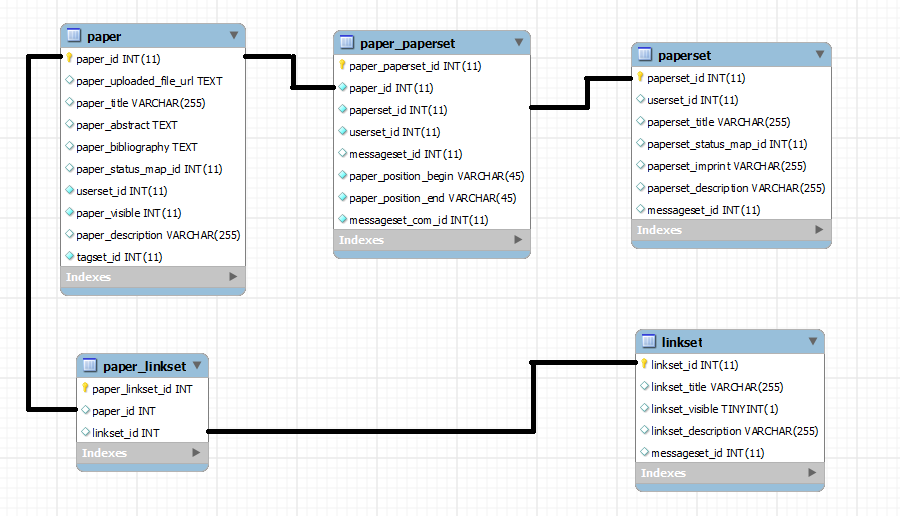


1. А уже с помощью таблицы **user\_userset** мы находим необходимых нам пользователей

**!!!Данная схема используется и для системы тегов.**

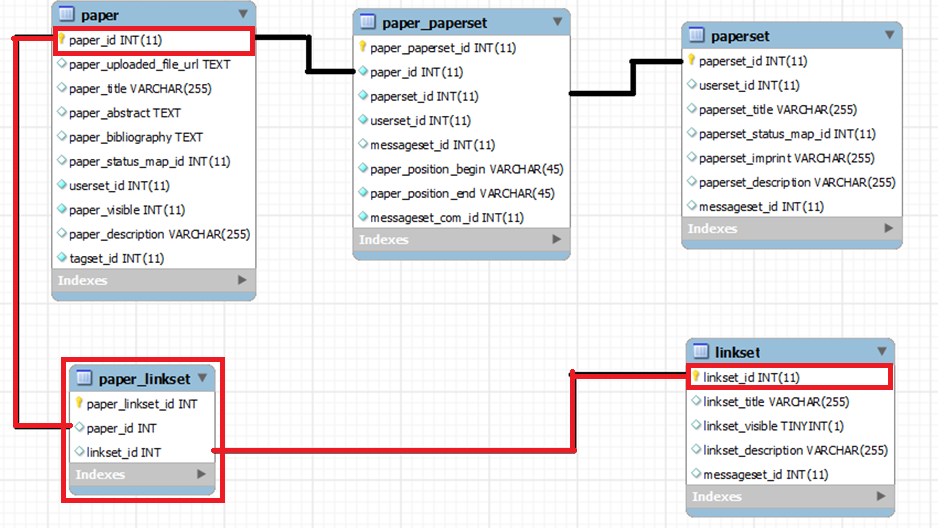
**Система создания журнала и пополнение ее статьями**

Ниже представлены таблицы участвующие в данной системе

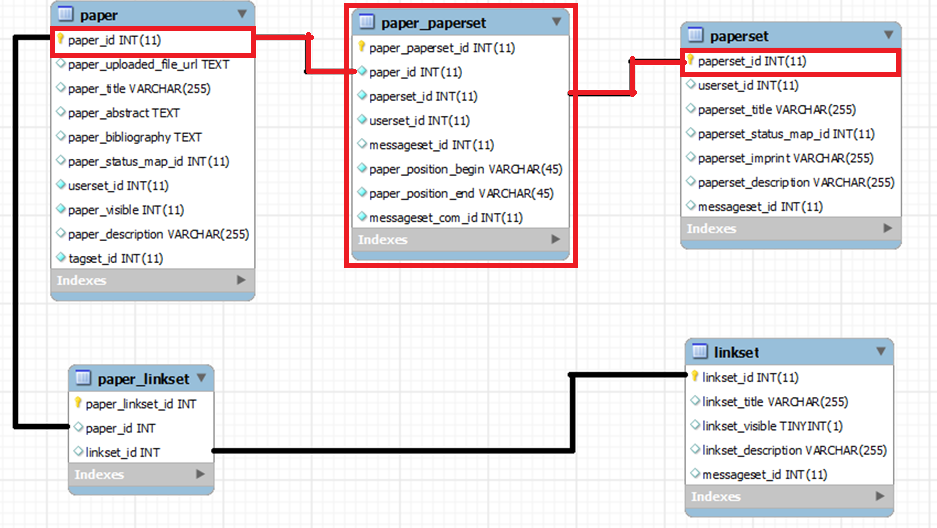


В нашей системе рассматриваются две коллекции, это **paperset**(журнал) и **linkset**(личная коллекция).

При создании статьи(paper), она автоматически связывается с **linkset** с помощью таблицы **paper\_linkset** (Используется ХП **insert\_paper**).



Только после этого мы можем связать статью и журнал через таблицу **paper\_paperset** (используется ХП **insert\_paper\_paperset**).



**!!!Пока не рассмотрена система сообщений**

**Описание хранимых процедур**

Процедура **create\_message**

Данная процедура требуется для создания сообщения.

PROCEDURE `create\_message`(IN `text\_message` TEXT, IN `url` VARCHAR(255), OUT `id\_message` INT)

Входные параметры:

* text\_message( TEXT )- содержит в себе текст сообщения
* url(VARCHAR(255)) – содержит в себе ссылку на прикрепленный файл

Возвращаемое значение:

* Id\_message(INT) – id созданного сообщения

Процедура **create\_messageset**

Данная процедура требуется для создания группы сообщений прикрепленного к объекту.

PROCEDURE `create\_messageset`(IN `title` VARCHAR(255), IN `description` VARCHAR(255), OUT `rez` INT)

Входные параметры

* Title(VARCHAR(255)) – Название группы
* description(VARCHAR(255)) – описание группы сообщений

Возвращаемое значение

* Id\_messageset(INT) – id созданной группы сообщений

Процедура **create\_userset**

Требуется для создания группы пользователей

PROCEDURE `create\_userset`(OUT `id\_userset` INT)

Входные параметры –

Возвращаемое значение

* Id\_userset(INT) – id созданной группы пользователей

Процедура **insert\_message\_messageset**

Требуется для создания сообщения входящего в известную нам группу сообщений

PROCEDURE `insert\_message\_messageset`(IN `text\_message` TEXT, IN `url` VARCHAR(255), IN `id\_messageset` INT, IN `id\_user` INT)

Входные параметры

* Text\_message(TEXT) – текст написанного сообщения
* url(VARCHAR(255)) – ссылка прикрепленного файла к сообщению
* id\_messageset(INT) – id группы сообщений в которую входит данное сообщение
* id\_user(INT) – id отправителя данного сообщения

Возвращаемое значение –

Процедура **insert\_paper**

Требуется для создания статьи

PROCEDURE `insert\_paper`(`url` TEXT, `title` TEXT, `abstract` TEXT, `bibliography` TEXT, `status` INT, `description` VARCHAR(255), `users` VARCHAR(255), `tags` VARCHAR(255))

Входные параметры

* url(TEXT) – ссылка на статью
* title(TEXT) – название статьи
* abstract(TEXT) – аннотация статьи
* bibliography(TEXT) – список литературы
* status(INT) – статус статьи(на каком этапе разработки она находится)
* description(VARCHAR(255)) – описание статьи
* users(VARCHAR(255)) – список пользователей(авторов) прикрепленных к данной статье(оформляется в виде <id пользователя>.<его роль>,<id пользователя>.<его роль>,…)
* tags(VARCHAR(255)) – список тегов прикрепленных к данной статье(оформляется в виде <id тега>,<id тега>,….)

Возвращаемое значение –

Процедура **insert\_paperset**

Требуется для создания журнала

PROCEDURE `insert\_paperset`(title varchar(255), status\_paperset int, imprint varchar(255), description varchar(255), users varchar(255))

Входные параметры

* Title(VARCHAR(255)) – название коллекции статей(журнала)
* status\_paperset(INT) – этап разработки журнала
* imprint(VARCHAR(255)) – выходные сведения
* description(VARCHAR(255)) – описание журнала
* users(VARCHAR(255)) – список пользователей(авторов) прикрепленных к данному журналу(оформляется в виде <id пользователя>.<его роль>,<id пользователя>.<его роль>,…)

Возвращаемое значение –

Процедура **insert\_paper\_paperset**

Требуется для связывания статьи и известного нам журнала

PROCEDURE `insert\_paper\_paperset`(id\_paper int, id\_paperset int, position\_begin varchar(255), position\_end varchar(255), users\_expert varchar(255))

Входные параметры

* Id\_paper(INT) – id статьи
* id\_paperset(INT) – id журнала в который входит статья
* position\_begin(VARCHAR(255)) – начальная позиция в журнале
* position\_end(VARCHAR(255)) – конечная позиция в журнале
* users\_expert(VARCHAR(255)) – список пользователей(экспертов) прикрепленных к данной статье(оформляется в виде <id пользователя>.<его роль>,<id пользователя>.<его роль>,…)

Возвращаемое значение –

Процедура **create\_tagset**

Требуется для создания группы тегов.

PROCEDURE `create\_tagset`(OUT `id\_tagset` INT)

Входные параметры –

Возвращаемое значение

* Id\_tagset(INT) – id созданной группы тегов

Процедура **insert\_tagsub\_tagset**

Требуется для добавления тегов в известную нам группы тегов

PROCEDURE `insert\_tagsub\_tagset`(IN `tags` VARCHAR(255), IN `id\_tagset` INT)

Входные параметры

* tags(VARCHAR(255)) – список тегов (оформляется в виде <id тега>,<id тега>,….)
* id\_tagset(INT) – id группы тегов к которой прикреплены список тегов

Возвращаемое значение –

Процедура **insert\_user**

Требуется для создания пользователя (автоматически создается личная коллекция для пользователя)

PROCEDURE `insert\_user`(IN `name` VARCHAR(255), surname varchar(255), patronymic varchar(255), IN `rdate` VARCHAR(255), IN `residence` VARCHAR(255), gender varchar(10), education varchar(255), IN `tags` VARCHAR(255)

Входные параметры

* name(varchar(255))- имя пользователя
* surneme(varchar(255)) – фамилия пользователя
* patronymic(varchar(255)) – отчество пользователя
* rdate(date) – дата рождения пользователя
* residence(varchar(255)) – место проживания пользователя
* gender(varchar(10)) – пол
* education(varchar(255)) - образование
* tags(VARCHAR(255)) – список тегов прикрепленных к данному пользователю (оформляется в виде <id тега>,<id тега>,….)

Возвращаемое значение –

Процедура **insert\_user\_userset**

Требуется для добавления пользователей в известную нам группу пользователей

PROCEDURE `insert\_user\_userset`(IN `users` VARCHAR(255), IN `id\_userset` INT)

Входные параметры

* users(VARCHAR(255)) – список пользователей(авторов) (оформляется в виде <id пользователя>.<его роль>,<id пользователя>.<его роль>,…)
* id\_userset(INT) – id группы пользователей

Процедура **create\_linkset**

Требуется для создания личной коллекции

PROCEDURE `create\_linkset`(id\_user int)

Входные параметры

* Id\_user(int) – id пользователя, для кого будет создана личная коллекция

Процедура **update\_paper**

Требуется для обновления статьи(paper)

PROCEDURE `update\_paper`( `url` TEXT, `title` TEXT, `abstract` TEXT, `bibliography` TEXT, `status` INT, `description` VARCHAR(255), `users` VARCHAR(255), `tags` VARCHAR(255), `id\_paper` INT)

Входные параметры

* url(TEXT) – ссылка на статью
* title(TEXT) – название статьи
* abstract(TEXT) – аннотация статьи
* bibliography(TEXT) – список литературы
* status(INT) – статус статьи(на каком этапе разработки она находится)
* description(VARCHAR(255)) – описание статьи
* users(VARCHAR(255)) – список пользователей(авторов) прикрепленных к данной статье(оформляется в виде <id пользователя>.<его роль>,<id пользователя>.<его роль>,…)
* tags(VARCHAR(255)) – список тегов прикрепленных к данной статье(оформляется в виде <id тега>,<id тега>,….)
* id\_paper(INT) – id статьи которую мы обновляем

Возвращаемое значение –

Процедура **update\_paperset**

Требуется для обновления журнала(paperset)

PROCEDURE `update\_paperset`(title varchar(255), status\_paperset int, imprint varchar(255), description varchar(255), users varchar(255), id\_paperset int)

Входные параметры

* Title(VARCHAR(255)) – название коллекции статей(журнала)
* status\_paperset(INT) – этап разработки журнала
* imprint(VARCHAR(255)) – выходные сведения
* description(VARCHAR(255)) – описание журнала
* users(VARCHAR(255)) – список пользователей(авторов) прикрепленных к данному журналу(оформляется в виде <id пользователя>.<его роль>,<id пользователя>.<его роль>,…)
* id\_paperset(INT) – id журнала который мы обновляем

Возвращаемое значение –

Процедура **update\_tagsub\_tagset**

Требуется для обновления тегов входящих в группу тегов

PROCEDURE `update\_tagsub\_tagset`(IN `tags` VARCHAR(255), IN `id\_tagset` INT)

Входные параметры

* tags(VARCHAR(255)) – список тегов (оформляется в виде <id тега>,<id тега>,….)
* id\_tagset(INT) – id группы тегов которой мы обновляем

Возвращаемое значение –

Процедура **update\_user**

Требуется для обновления данных пользователя

PROCEDURE `update\_user`( `name` VARCHAR(255), `surname` VARCHAR(255), `patronymic` VARCHAR(255), `rdate` VARCHAR(255), `residence` VARCHAR(255), `gender` VARCHAR(10), `education` VARCHAR(255), `tags` VARCHAR(255), id\_user int)

Входные параметры

* name(varchar(255))- имя пользователя
* surneme(varchar(255)) – фамилия пользователя
* patronymic(varchar(255)) – отчество пользователя
* rdate(date) – дата рождения пользователя
* residence(varchar(255)) – место проживания пользователя
* gender(varchar(10)) – пол
* education(varchar(255)) - образование
* tags(VARCHAR(255)) – список тегов прикрепленных к данному пользователю (оформляется в виде <id тега>,<id тега>,….)
* id\_user(INT) – id пользователя которого мы обновляем

Возвращаемое значение –

Процедура **update\_user\_userset**

Требуется для обновления списка пользователей в группе пользователей

PROCEDURE `update\_user\_userset`(IN `users` VARCHAR(255), IN `id\_userset` INT)

Входные параметры

* users(VARCHAR(255)) – список пользователей(авторов) (оформляется в виде <id пользователя>.<его роль>,<id пользователя>.<его роль>,…)
* id\_userset(INT) – id группы пользователей которую мы обновляем