-- Практическое задание по теме “Операторы, фильтрация, сортировка и ограничение.

-- Агрегация данных”

-- Работаем с БД vk и тестовыми данными, которые вы сгенерировали ранее:

-- 1. Создать и заполнить таблицы лайков и постов.

-- Таблица лайков

**DROP** **TABLE** **IF** **EXISTS** likes;

**CREATE** **TABLE** likes (

id **INT** **UNSIGNED** **NOT** **NULL** **AUTO\_INCREMENT** **PRIMARY** **KEY**,

user\_id **INT** **UNSIGNED** **NOT** **NULL**,

target\_id **INT** **UNSIGNED** **NOT** **NULL**,

target\_type\_id **INT** **UNSIGNED** **NOT** **NULL**,

created\_at **DATETIME** **DEFAULT** **CURRENT\_TIMESTAMP**

);

-- Таблица типов лайков

**DROP** **TABLE** **IF** **EXISTS** target\_types;

**CREATE** **TABLE** target\_types (

id **INT** **UNSIGNED** **NOT** **NULL** **AUTO\_INCREMENT** **PRIMARY** **KEY**,

name **VARCHAR**(255) **NOT** **NULL** **UNIQUE**,

created\_at **DATETIME** **DEFAULT** **CURRENT\_TIMESTAMP**

);

**INSERT** **INTO** target\_types (name) **VALUES**

('messages'),

('users'),

('media'),

('posts');

-- Заполняем лайки

**INSERT** **INTO** likes

**SELECT**

id,

**FLOOR**(1 + (**RAND**() \* 100)),

**FLOOR**(1 + (**RAND**() \* 100)),

**FLOOR**(1 + (**RAND**() \* 4)),

**CURRENT\_TIMESTAMP**

**FROM** messages;

-- Создадим таблицу постов

**DROP** **TABLE** **IF** **EXISTS** posts;

**CREATE** **TABLE** posts (

id **INT** **UNSIGNED** **NOT** **NULL** **AUTO\_INCREMENT** **PRIMARY** **KEY**,

user\_id **INT** **UNSIGNED** **NOT** **NULL**,

community\_id **INT** **UNSIGNED**,

head **VARCHAR**(255),

body **TEXT** **NOT** **NULL**,

media\_id **INT** **UNSIGNED**,

is\_public **BOOLEAN** **DEFAULT** **TRUE**,

is\_archived **BOOLEAN** **DEFAULT** **FALSE**,

created\_at **DATETIME** **DEFAULT** **CURRENT\_TIMESTAMP**,

updated\_at **DATETIME** **DEFAULT** **CURRENT\_TIMESTAMP** **ON** **UPDATE** **CURRENT\_TIMESTAMP**

);

**INSERT** **INTO** posts

**SELECT**

id, -- id

**FLOOR**(1 + (**RAND**() \* 100)), -- user\_id

**FLOOR**(1 + (**RAND**() \* 20)), -- community\_id

**CONCAT**('Head №', id), -- head

**CONCAT**('Body №', id), -- body

**FLOOR**(1 + (**RAND**() \* 100)), -- media\_id

**FLOOR**((**RAND**() \* 2)), -- is\_public

**FLOOR**((**RAND**() \* 2)), -- is\_archived

**CURRENT\_TIMESTAMP**,

**CURRENT\_TIMESTAMP**

**FROM** messages;

-- 2. Создать все необходимые внешние ключи и диаграмму отношений.

-- communities\_users

**ALTER** **TABLE** communities\_users

**ADD** **CONSTRAINT** communities\_users\_community\_id\_fk

**FOREIGN** **KEY** (community\_id) **REFERENCES** communities(id);

**ALTER** **TABLE** communities\_users

**ADD** **CONSTRAINT** communities\_users\_user\_id\_fk

**FOREIGN** **KEY** (user\_id) **REFERENCES** users(id);

-- friendship

**ALTER** **TABLE** friendship

**ADD** **CONSTRAINT** friendship\_user\_id\_fk

**FOREIGN** **KEY** (user\_id) **REFERENCES** users(id),

**ADD** **CONSTRAINT** friendship\_friend\_id\_fk

**FOREIGN** **KEY** (friend\_id) **REFERENCES** users(id),

**ADD** **CONSTRAINT** friendship\_friendship\_status\_id\_fk

**FOREIGN** **KEY** (friendship\_status\_id) **REFERENCES** friendship\_statuses(id);

-- likes

**ALTER** **TABLE** likes

**ADD** **CONSTRAINT** likes\_user\_id\_fk

**FOREIGN** **KEY** (user\_id) **REFERENCES** users(id),

**ADD** **CONSTRAINT** likes\_target\_type\_id\_fk

**FOREIGN** **KEY** (target\_type\_id) **REFERENCES** target\_types(id);

-- media

**ALTER** **TABLE** media

**ADD** **CONSTRAINT** media\_user\_id\_fk

**FOREIGN** **KEY** (user\_id) **REFERENCES** users(id),

**ADD** **CONSTRAINT** media\_media\_type\_id\_fk

**FOREIGN** **KEY** (media\_type\_id) **REFERENCES** media\_types(id);

-- posts

**alter** **table** posts **modify** **column** media\_id **bigint**; -- меняем тип, чтобы была совместимость типов

**ALTER** **TABLE** posts

**ADD** **CONSTRAINT** posts\_user\_id\_fk

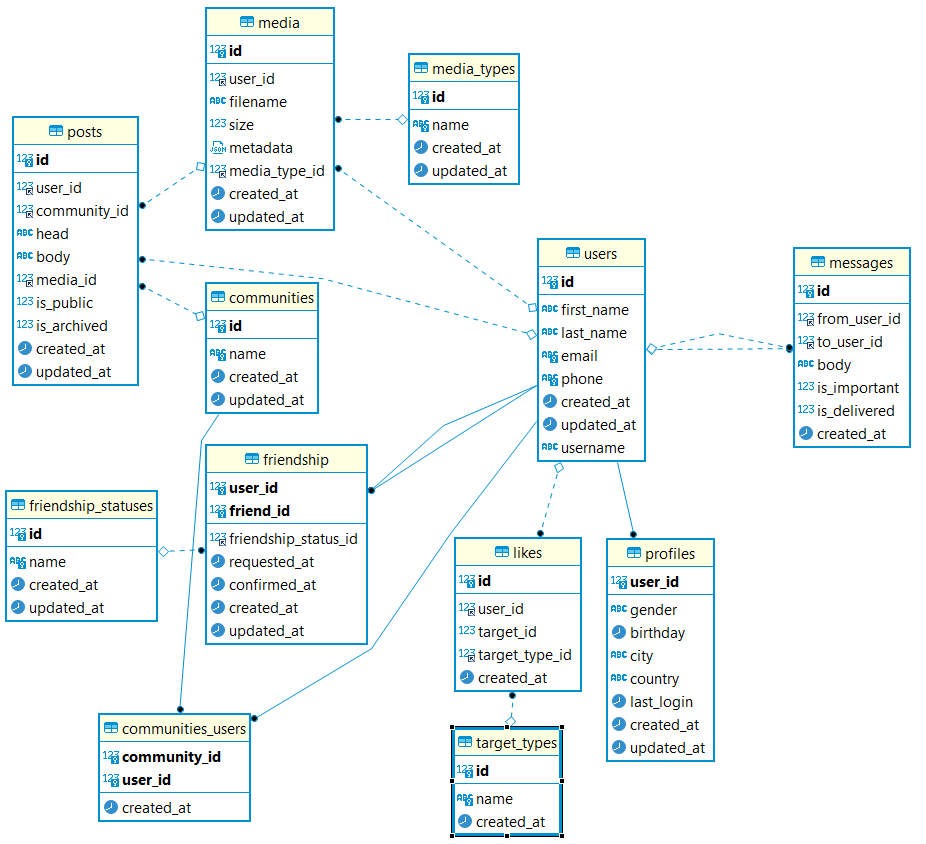
**FOREIGN** **KEY** (user\_id) **REFERENCES** users(id),

**ADD** **CONSTRAINT** posts\_community\_id\_fk

**FOREIGN** **KEY** (community\_id) **REFERENCES** communities(id),

**ADD** **CONSTRAINT** posts\_media\_id\_fk

**FOREIGN** **KEY** (media\_id) **REFERENCES** media(id);



-- 3. Определить кто больше поставил лайков (всего) - мужчины или женщины?

-- не ипользуем конструкции join

**select**

t.sex "Пол",

t.cnt "Кол-во лайков"

**from** (

**select**

'М' sex,

**count**(\*) cnt

**from**

likes l

**where**

l.user\_id **in** (**select** p.user\_id

**from** profiles p

**where** p.gender = 'm') -- мужчины

**union**

**select**

'Ж' sex,

**count**(\*) cnt

**from**

likes l

**where**

l.user\_id **in** (**select** p.user\_id

**from** profiles p

**where** p.gender = 'w') -- мужчины

) t

**order** **by** t.cnt **desc**

**limit** 1;

-- 4. Подсчитать количество лайков которые получили 10 самых молодых пользователей.

-- Не используем конструкцию JOIN

-- Данный запрос НЕ работает, т.к. This version of MySQL doesn't yet support 'LIMIT & IN/ALL/ANY/SOME subquery'

**select** **count**(\*)

**from** likes l

**where** l.target\_id **in** (**select** p.user\_id

**from** profiles p

**order** **by** p.birthday **desc**

**limit** 10);

-- Пишем по другому

**select** **sum**(cnt)

**from** (

-- Для 10 самых молодых пользователей

**select**

(**select** **count**(\*) **from** likes l **where** l.user\_id = p.user\_id) **as** cnt

**from** profiles p

**order** **by** p.birthday **desc**

**limit** 10

) t;

-- 5. Найти 10 пользователей, которые проявляют наименьшую активность в

-- использовании социальной сети

-- (критерии активности необходимо определить самостоятельно).

-- Не используем конструкцию JOIN

/\* Критерием активности будем считать сумму статистик:

\* - количество постов пользователя

\* - количество лайков, поставленных пользователем

\*/

**select**

**concat**(first\_name, ' ', last\_name) "ФИО",

(**coalesce**((**select** **count**(\*) **from** posts p **where** p.user\_id = u.id), 0) +

**coalesce**((**select** **count**(\*) **from** likes l **where** l.user\_id = u.id), 0)

) **as** total\_activity

**from**

users u

**order** **by** total\_activity

**limit** 10;