1)

-- 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 **enum**('messages', 'users', 'posts', 'media') **NOT** **NULL**,

created\_at **datetime** **DEFAULT** **current\_timestamp**

);

-- Временная таблица типов лайков

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

name **varchar**(100) **NOT** **NULL** **UNIQUE**

);

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

('messages'),

('users'),

('posts'),

('media');

-- Заполняем лайки

**INSERT** **INTO** likes

**SELECT**

id,

**floor** (1 + (**rand**()\*100)),

**floor** (1 + (**rand**()\*100)),

(**SELECT** name **FROM** target\_types **ORDER** **BY** **rand**() **LIMIT** 1),

**current\_timestamp**

**FROM** messages;

-- Создание таблицы постов

**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**

);

2)

**ALTER** **TABLE** profiles

**ADD** **CONSTRAINT** profiles\_user\_id\_fk

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

**ON** **DELETE** **cascade**;

**ALTER** **TABLE** profiles **DROP** **FOREIGN** **KEY** profiles\_user\_id\_fk;

**ALTER** **TABLE** messages

**ADD** **CONSTRAINT** messages\_from\_user\_id\_fk

**FOREIGN** **KEY** (from\_user\_id) **REFERENCES** users(id)

**ON** **DELETE** **CASCADE**,

**ADD** **CONSTRAINT** messages\_to\_user\_id\_fk

**FOREIGN** **KEY** (to\_user\_id) **REFERENCES** users(id)

**ON** **DELETE** **CASCADE**;

**ALTER** **TABLE** cities

**ADD** **CONSTRAINT** cities\_id\_fk

**FOREIGN** **KEY** (country\_id) **REFERENCES** countries(id)

**ON** **DELETE** **cascade**;

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

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

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

**ON** **DELETE** **CASCADE**,

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

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

**ON** **DELETE** **CASCADE**;

**ALTER** **TABLE** profiles

**ADD** **CONSTRAINT** profiles\_city\_id\_fk

**FOREIGN** **KEY** (city\_id) **REFERENCES** cities(id)

**ON** **DELETE** **cascade**;

**ALTER** **TABLE** friendship

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

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

**ON** **DELETE** **CASCADE**,

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

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

**ON** **DELETE** **CASCADE**,

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

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

**ON** **DELETE** **CASCADE**;

**ALTER** **TABLE** media

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

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

**ON** **DELETE** **CASCADE**,

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

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

**ON** **DELETE** **CASCADE**;

**ALTER** **TABLE** posts

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

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

**ON** **DELETE** **CASCADE**,

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

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

**ON** **DELETE** **CASCADE**,

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

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

**ON** **DELETE** **CASCADE**;

**ALTER** **TABLE** likes

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

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

**ON** **DELETE** **cascade**;

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

**SELECT** **IF** (

(**SELECT** **count**(\*)

**FROM** likes

**WHERE** user\_id **IN** (**SELECT** user\_id **FROM** profiles **WHERE** gender = 'F')) >

(**SELECT** **count**(\*)

**FROM** likes

**WHERE** user\_id **IN** (**SELECT** user\_id **FROM** profiles **WHERE** gender = 'M')),

'Females', 'Males') **AS** more\_likes\_put\_by;

Если учесть возможность одинакового количества лайков, поставленных мужчинами и женщинами, то можно выполнить такой запрос:

**SELECT** **IF** (

(**SELECT** **count**(\*)

**FROM** likes

**WHERE** user\_id **IN** (**SELECT** user\_id **FROM** profiles **WHERE** gender = 'F')) >

(**SELECT** **count**(\*)

**FROM** likes

**WHERE** user\_id **IN** (**SELECT** user\_id **FROM** profiles **WHERE** gender = 'M')),

'Females',

(**SELECT** **IF** (

(**SELECT** **count**(\*)

**FROM** likes

**WHERE** user\_id **IN** (**SELECT** user\_id **FROM** profiles **WHERE** gender = 'F')) <

(**SELECT** **count**(\*)

**FROM** likes

**WHERE** user\_id **IN** (**SELECT** user\_id **FROM** profiles **WHERE** gender = 'M')),

'Males','equally by females and males'))) **AS** more\_likes\_put\_by;

4) Вывести для каждого пользователя количество созданных сообщений, постов, загруженных медиафайлов и поставленных лайков

**SELECT**

(**SELECT** **concat**(first\_name, ' ', last\_name)) **AS** users,

(**SELECT** **count**(\*) **FROM** messages **WHERE** messages.from\_user\_id = users.id) **AS** Number\_of\_messages\_written,

(**SELECT** **count**(\*) **FROM** posts **WHERE** posts.user\_id = users.id) **AS** Number\_of\_posts\_created,

(**SELECT** **count**(\*) **FROM** media **WHERE** media.user\_id = users.id) **AS** Number\_of\_media\_submitted,

(**SELECT** **count**(\*) **FROM** likes **WHERE** likes.user\_id = users.id) **AS** Number\_of\_likes\_put

**FROM** users;

5) Подсчитать количество лайков которые получили 10 самых последних сообщений