# Отчет по лабораторной работе №5 – Шардирование.

## Исходники:

<https://github.com/filatkinen/socialnet>

Docker-compose и папка с необходимыми файлами:

<https://github.com/filatkinen/socialnet/tree/main/labs/lab05>

bash -скрипт с командами curl для проверки api:

<https://github.com/filatkinen/socialnet/blob/main/labs/lab05/check.sh>

## Описание решения:

1. В таблицу users добавлено поле dialog\_shard\_id. Данные в таблице обновлены, каждой записи присвоено случайное значение от 1 до3:

update users set dialog\_shard\_id=(*floor*(*random*() \* 3 + 1)::int);

1. В БД добавлена таблица shards:

CREATE TABLE IF NOT EXISTS shards  
(  
 shard\_id INTEGER NOT NULL,  
 dsn TEXT NOT NULL,  
 active BOOLEAN NOT NULL  
);

в dsn указывается строка подключения к БД, где расположена таблица dialogs

CREATE TABLE IF NOT EXISTS dialogs  
(  
 dialog\_id CHAR(36) NOT NULL,  
 user\_id CHAR(36),  
 friend\_id CHAR(36),  
 message TEXT  
)

поле active опционально, активная/неактвная шарда

1. Добавлены 3 БД – по количеству шардов и в них созданы таблицы dialogs: snet\_shard01, snet\_shard01, snet\_shard01
2. В таблице shards основной БД добавлены записи:

1,postgres://socialnet:socialnet@localhost:5432/snet\_shard01?sslmode=disable&Timezone=UTC,true

2,postgres://socialnet:socialnet@localhost:5432/snet\_shard02?sslmode=disable&Timezone=UTC,true

3,postgres://socialnet:socialnet@localhost:5432/snet\_shard03?sslmode=disable&Timezone=UTC,true

1. При старте приложения создается мапа, состоящая из структур, одним из элементов которых которых является поинтер на db.sql, который инициализируется по dsn, указанный в значении поля dsn таблицы shards. Доступ к элементу мапы по номеру шарда.

func (s \*Storage) GetShards(ctx context.Context) (err error) {  
 // closing connections shard and clear  
 for k := range s.shards {  
 if s.shards[k].db != nil {  
 \_ = s.shards[k].db.Close()  
 delete(s.shards, k)  
 }  
 }  
  
 query := `select shard\_id, dsn, active from shards`  
 rows, err := s.db.QueryContext(ctx, query)  
 if err != nil {  
 return ErrorConnectShard  
 }  
 defer rows.Close()  
 for rows.Next() {  
 var shard Shard  
 var e = rows.Scan(  
 &shard.shardid,  
 &shard.dsn,  
 &shard.active,  
 )  
 if e != nil {  
 return ErrorConnectShard  
 }  
 db, e := sql.Open("postgres", shard.dsn)  
 if e != nil {  
 err = errors.Join(err, e)  
 continue  
 }  
 e = s.db.PingContext(ctx)  
 if e != nil {  
 err = errors.Join(err, e)  
 continue  
 }  
 shard.db = db  
 s.shards[shard.shardid] = shard  
 }  
 return err  
}

1. Шардинг устроен следующим образом: при записи постов диалога, либо при получении списка диалогов определяется максимум значения **dialog\_shard\_id** от 2-ух пользователей(отправитель/получатель). Это максимальное значение и является номер шарда, в котором будятт храниться сообщения.

func (s \*Storage) GetShardNumber(ctx context.Context, userID string, friendID string) (int, error) {  
 query := `select dialog\_shard\_id from users where user\_id=$1`  
 var shard1, shard2 int  
 if err := s.db.QueryRowContext(ctx, query, userID).Scan(&shard1); err != nil {  
 return 0, err  
 }  
 if err := s.db.QueryRowContext(ctx, query, friendID).Scan(&shard2); err != nil {  
 return 0, err  
 }  
 if shard1 > shard2 {  
 return shard1, nil  
 }  
 return shard2, nil  
}

1. При добавлении пользователя номер шарда указывается как максимальное значение, хранящееся в таблице shards:

query := `select max(shard\_id) FROM shards where active='true'`  
var shardID int  
if err := s.db.QueryRowContext(ctx, query).Scan(&shardID); err != nil {  
 if err == sql.ErrNoRows {  
 return storage.ErrRecordNotFound  
 }  
 return err  
}

1. Добавление шарда:
   1. Добавить запись в таблицу shards, указав необходимый dsn и следующий номер шарда.
   2. На целевом сервере создать БД, в которой нужно создать однотипную таблицу **dialogs**
   3. С помощью прикрученной ручки к приложению /shardupdate обновить список шардов, доступных приложению.
2. Решардинг: Диалог с новыми пользователями, у которых будет указан новый номер шарда будет храниться в этом шарде. Решардинг не требует останова системы. Вот такой способ.