МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное автономное образовательное учреждение высшего образования

«Санкт-Петербургский государственный университет аэрокосмического приборостроения»

ФАКУЛЬТЕТ СРЕДНЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ

ОТЧЕТ О ПРАКТИКЕ   
ЗАЩИЩЕН С ОЦЕНКОЙ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

РУКОВОДИТЕЛИ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| преподаватели |  |  |  | И.А.Юрьева, И.Л.Рохманько, |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |
|  |  |  |  |  |

|  |
| --- |
| ОТЧЕТ ПО УЧЕБНОЙ ПРАКТИКЕ |
| В СОСТАВЕ ПРОФЕССИОНАЛЬНОГО МОДУЛЯ  ПМ 04 Разработка, администрирование и защита баз данных |
|  |
|  |
|  |

ОТЧЕТ ВЫПОЛНИЛ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| СТУДЕНТ ГР. № | С021К |  |  |  | В. Д Панков |
|  |  |  | подпись, дата |  | инициалы, фамилия |

Санкт-Петербург 2023

ИНДИВИДУАЛЬНОЕ ЗАДАНИЕ

на прохождение учебной практики обучающегося по специальности

*вид практики*

09.02.07 «Информационные системы и программирование»

*код и наименование специальности*

1. Фамилия, имя, отчество обучающегося: Панков Василий Дмитриевич
2. Группа:С021 Период практики: с 26.01.23 по 08.02.23 г.
3. Тема задания: получение первичных профессиональных навыков, подготовка к освоению общих и профессиональных компетенций, по профессиональному модулю

ПМ.04 Разработка, администрирование и защита баз данных

*код и наименование профессионального модуля*

1. Вопросы, подлежащие изучению:

* Анализ предметной области «БД Театра»;
* Разработка диаграммы Чена;
* Реализация базы данных в конкретной СУБД;
* Разработка запросов к базе данных средствами языка DML;
* Импорт данных в базу данных.

1. Содержание отчетной документации:
   1. Отчёт, включающий в себя:

* титульный лист;
* индивидуальное задание;
* материалы о выполнении индивидуального задания (содержание определяется цикловой комиссией);
* список использованных источников.
  1. Аттестационный лист.

1. Срок представления отчета зам. декана по УПР « 08» февраля 2023 г.

Руководители практики от факультета СПО

преподаватель И.Л. Рохманько

должность, уч. степень, звание подпись, дата инициалы, фамилия

преподаватель И.А.Юрьева

должность, уч. степень, звание подпись, дата инициалы, фамилия

Задание принял к исполнению:

Обучающийся 26.01.2023

Дата подпись инициалы, фамилия

**СОДЕРЖАНИЕ**

[ВВЕДЕНИЕ 4](#_Toc126152232)

[1 Анализ предметной области 5](#_Toc126152233)

[2 Разработка диаграммы Чена 9](#_Toc126152234)

[3 Реализация базы данных в конкретной СУБД 12](#_Toc126152235)

[5 Импорт данных в базу данных 18](#_Toc126152236)

[ЗАКЛЮЧЕНИЕ 32](#_Toc126152237)

[ПРИЛОЖЕНИЕ А SQL- скрипт создания базы данных 33](#_Toc126152238)

[ПРИЛОЖЕНИЕ В Листинг приложения 37](#_Toc126152239)

# ВВЕДЕНИЕ

Решение широкого круга задач требует умения разрабатывать и создавать базы данных, подбирать соответствующие им системы управления базами данных, разрабатывать прикладные программы поиска и обработки информации.

Задачей учебной практики по разработке баз данных по МДК 04.01 «Технология разработки и защиты баз данных» является закрепление теоретического материала междисциплинарного курса МДК 04.01 и получение практических навыков по созданию и использованию баз данных.

# Анализ предметной области

* Предметная область: «БД Театра».
* Название ПО: «Расписание представлений»
* Описание ПО:

Вы работаете в театре, в который каждый день происходит множество представлений: оперы, балеты, хоровое пение и тому подобное.

С развитием технологий вы понимаете, что пользователям проще смотреть афишу в интернете, поэтому вам необходимо реализовать базу данных, которая обеспечит информационную базу для данного ресурса. Необходимо обеспечить поиск по театральным актёрам, режиссёрам и персонажам.

Учтите, что в вашем театре есть несколько корпусов.

* Анализ ПО «Расписание представлений»

В результате анализа предметной области выделены основные объекты, которые указаны в Таблице 1, и свойства объектов, которые указаны в таблицах с 2 по 14.

Таблица 1 – Список объектов

|  |  |
| --- | --- |
| № | Объекты |
| 1. | Вид постановки |
| 2 | Автор |
| 3. | Постановка |
| 4. | Вид персонала |
| 5. | Обслуживающий персонал |
| 6. | Актёр |
| 7. | Режиссёр |
| 8. | Роль |
| 9. | Сцена |
| 10. | Афиша |
| 11. | Актёры на дату |
| 12. | Звание актёра |
| 13. | Звание режиссёра |

Таблица 2 – Вид постановки

|  |  |  |
| --- | --- | --- |
| Идентификатор вида | Название вида | Особенности |
|  |  |  |

Таблица 3 – Автор

|  |  |
| --- | --- |
| Идентификатор автора | ФИО |
|  |  |

Таблица 4 – Постановка

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Идентификатор представления | Название представления | Вид представления | Акты | Длительность |
|  |  |  |  |  |

Таблица 5 – Должность персонала

|  |  |  |
| --- | --- | --- |
| Идентификатор должности | Наименование должности | Описание |
|  |  |  |

Таблица 6 – Обслуживающий персонал

|  |  |  |
| --- | --- | --- |
| Идентификатор персонала | ФИО | Должность |
|  |  |  |

Таблица 7 – Актёр

|  |  |  |  |
| --- | --- | --- | --- |
| Идентификатор актёра | ФИО | Дата рождения | Биография |
|  |  |  |  |

Таблица 8 – Режиссёр

|  |  |  |  |
| --- | --- | --- | --- |
| Идентификатор режиссёра | ФИО | Дата рождения | Биография |
|  |  |  |  |

Таблица 9 – Роль

|  |  |  |  |
| --- | --- | --- | --- |
| Идентификатор роли | Название роли | Идентификатор постановки | Описание |
|  |  |  |  |

Таблица 10 – Сцена

|  |  |  |  |
| --- | --- | --- | --- |
| Идентификатор сцены | Название сцены | Адрес | Описание |
|  |  |  |  |

Таблица 11 – Афиша

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Идентификатор в афиши | Идентификатор постановки | Идентификатор сцены | Дата/время проведения представления | Комментарий |
|  |  |  |  |  |

Таблица 12 – Актёры на дату

|  |  |  |  |
| --- | --- | --- | --- |
| Идентификатор афиши | Идентификатор актёра | Идентификатор роли | Комментарий |
|  |  |  |  |

Таблица 13 – Звание актёра

|  |  |
| --- | --- |
| Идентификатор звания актёра | Название звания актёра |
|  |  |

Таблица 14 – Звание режиссёра

|  |  |
| --- | --- |
| Идентификатор звания режиссёра | Название звания режиссёра |
|  |  |

* Объектная система.

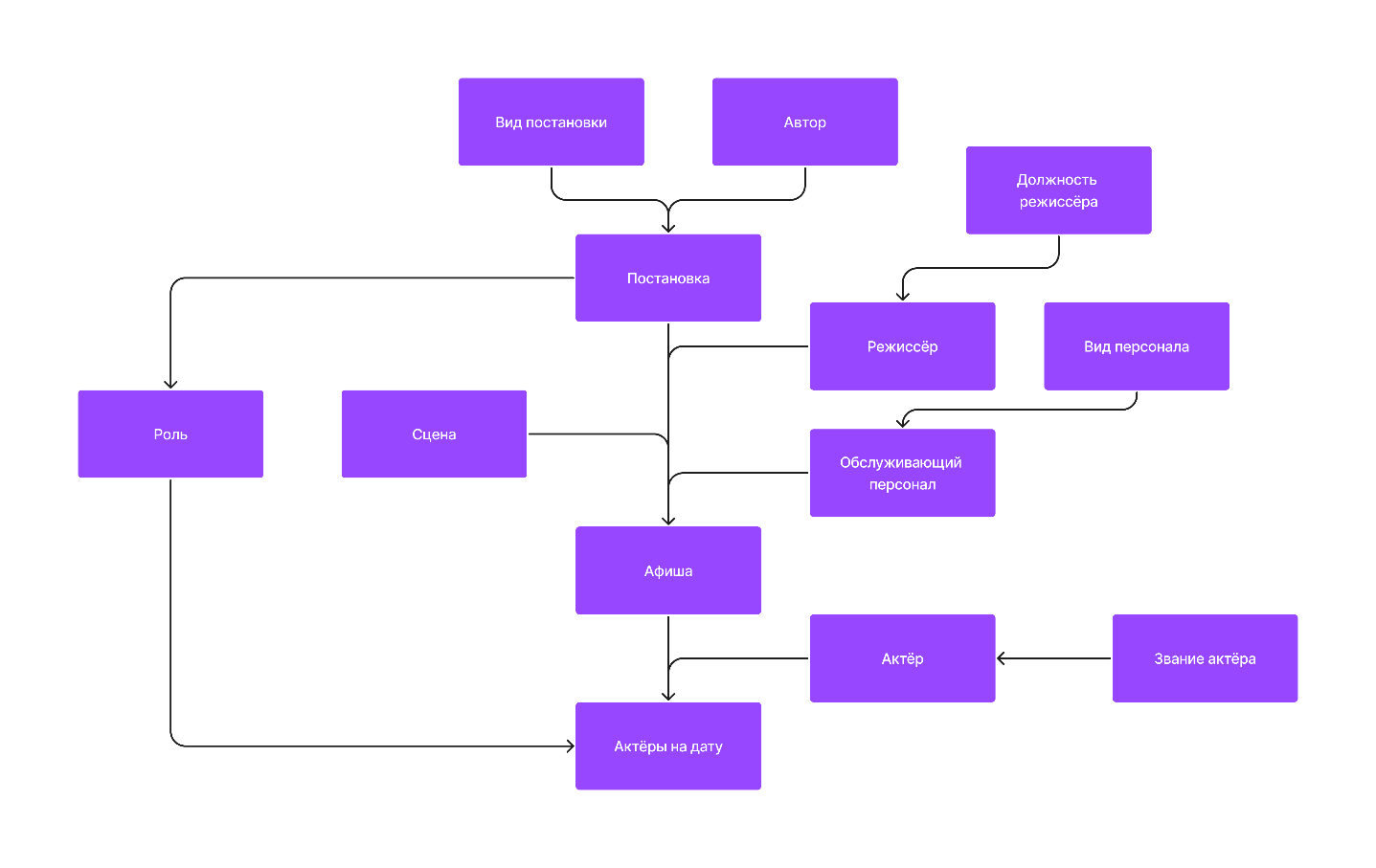


Рисунок 1 – Объектная система

# Разработка диаграммы Чена

Диаграммы "сущность-связь" (ERD) предназначены для разработки моделей данных и обеспечивают стандартный способ определения данных и отношений между ними.

Таблица 15 – Описание связей

|  |  |
| --- | --- |
| № | Имена связей |
| 1. | Характеризует |
| 2. | Сочиняет |
| 3. | Обслуживает |
| 4. | Выступает |
| 5. | Руководит |
| 6. | Размещает |
| 7. | Выдается |
| 8. | Назначает |
| 9. | Награждает |
| 10. | Участвует |
| 11. | Составляет |
| 12. | Награждает |
| 13. | Распределяет |

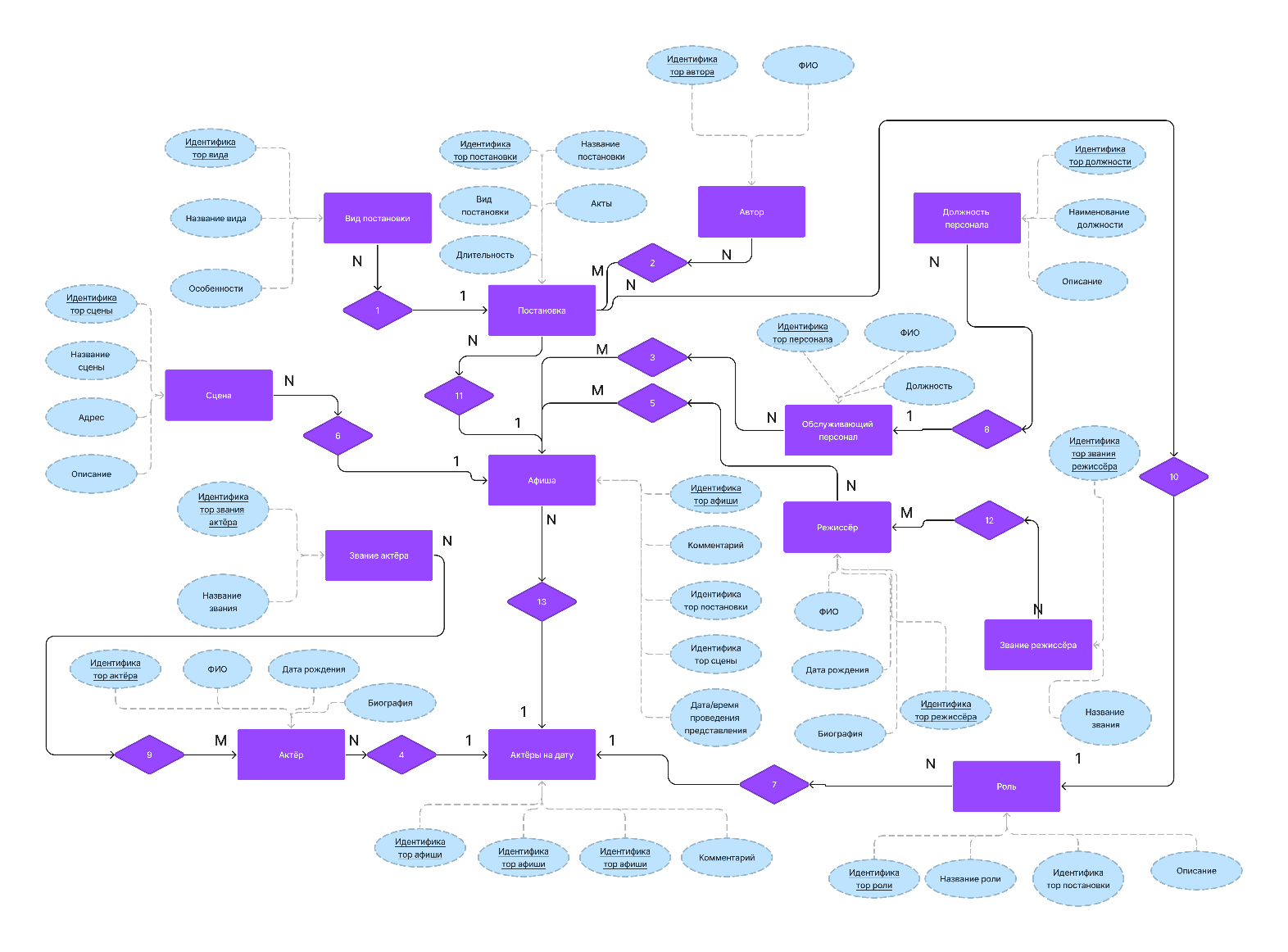


Рисунок 2 – Диаграмма Чена

Система создаётся для обслуживания следующих групп пользователей:

* Руководитель театра;
* Бухгалтерия;
* Администратор сайта;
* Пользователь сайта.
  + 1. Функциональные возможности руководителя театра:
       - Редактирование данных о артистах, персонале и режиссёре;
       - Удаление любых данных;
       - Добавление, удаление, редактирование данных о званиях режиссёров и актёров;
    2. Функциональные возможности бухгалтерии:
       - Обновление афиши, а также её редактирование;
       - Редактирование актёров, ролей, авторов и постановок;
       - Добавление новых постановок, видов постановок и авторов;
    3. Функциональные возможности администратора сайта:
       - Чтение всех данных из базы данных;
       - Реализация всех возможных запросов к базе данных;
    4. Функциональные возможности пользователя сайта:
       - Просмотр и поиск информацию об актёрах;
       - Ограниченные возможности фильтрации;
       - Нет возможности редактирования БД;

# Реализация базы данных в конкретной СУБД

Я реализовал базу данных по заданной предметной области в СУБД MySQL, с помощью Jetbrains DataGrip и MySQL Workbench, код созданных таблиц, показан в приложении А. Получилась диаграмма, представленная на рисунке 3. Также перечислил структуры созданных таблиц на рисунках 4-21.

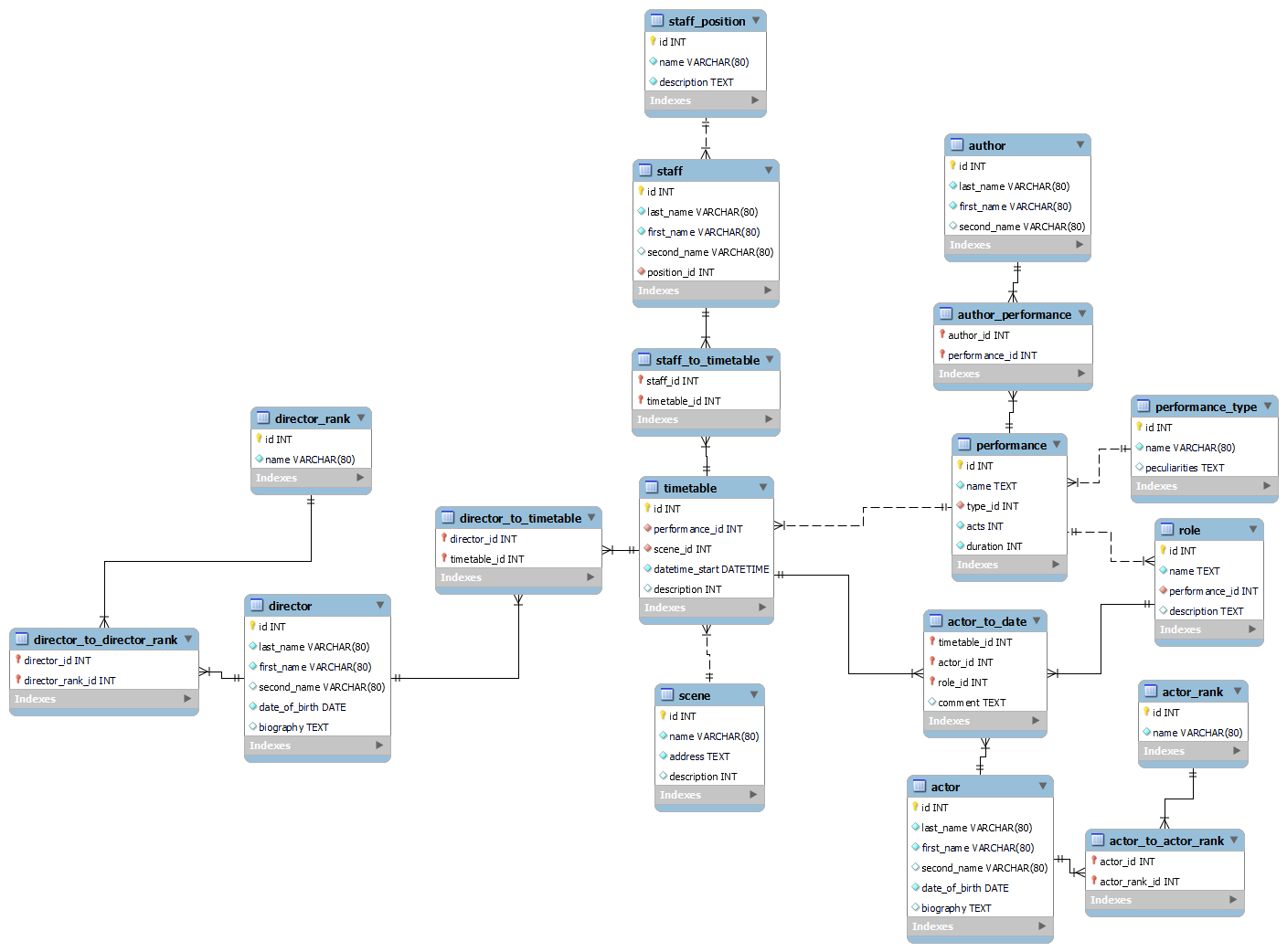


Рисунок 3 – Диаграмма таблиц в СУБД

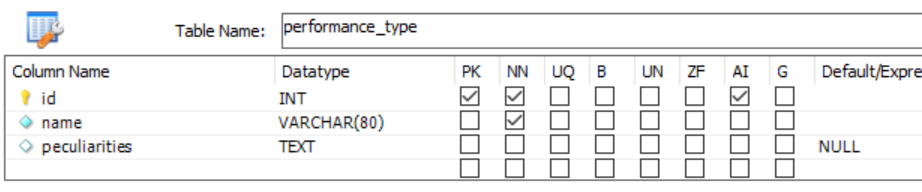


Рисунок 4 – Структура таблицы «performance\_type» (тип представления)

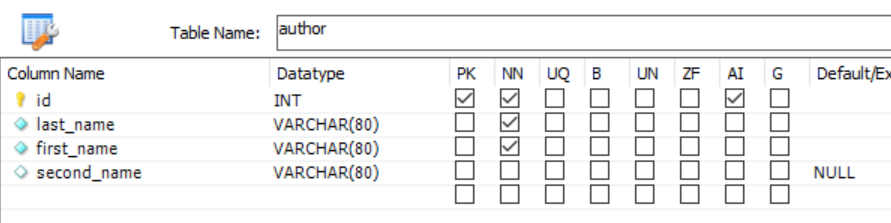


Рисунок 5 – Структура таблица «author» (Автор)

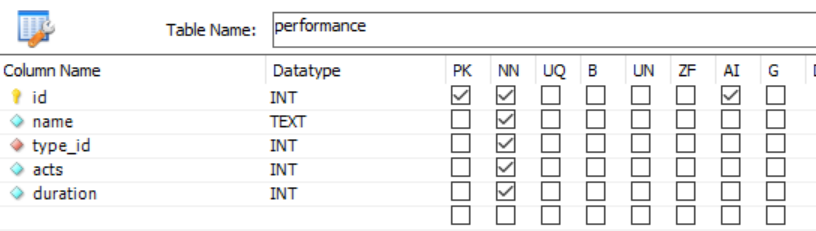


Рисунок 6 – Структура таблицы «performance» (Представление)

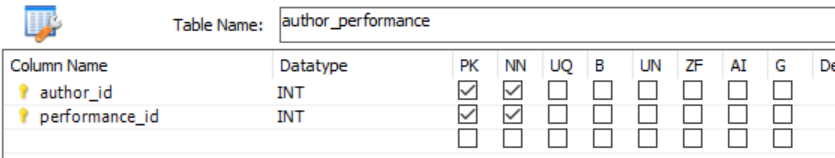


Рисунок 7 – Структура таблицы «author\_performance» (Автор к представлению)

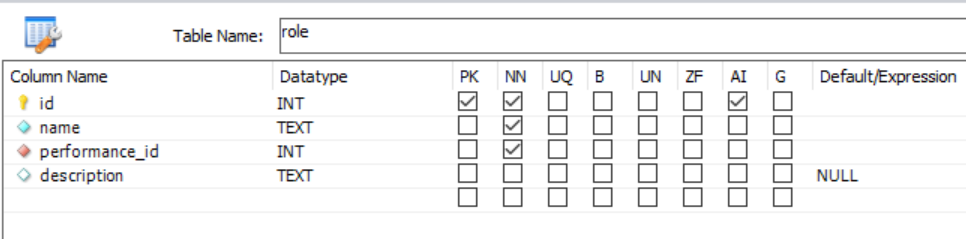


Рисунок 8 – Структура таблицы «role» (Роль)

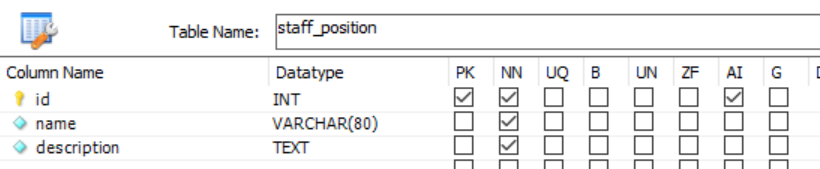


Рисунок 9 – Структура таблицы «staff\_position» (Должность персонала)

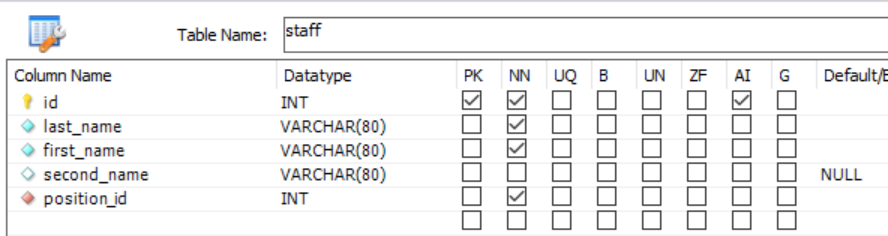


Рисунок 10 – Структура таблицы «staff» (Персонал)

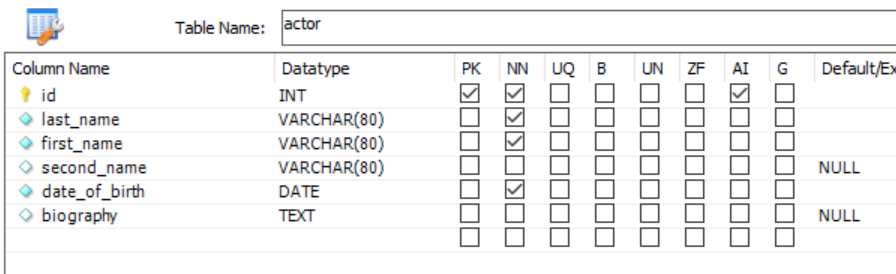


Рисунок 11 – Структура таблицы «actor» (Актёр)

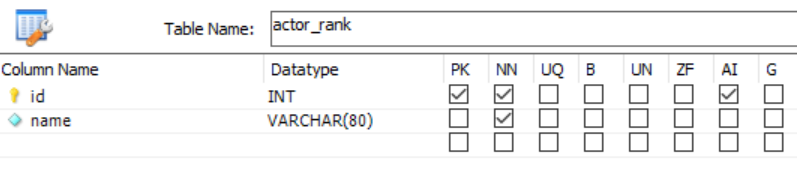


Рисунок 12 – Структура таблицы «actor\_rank» (Звание актёра)

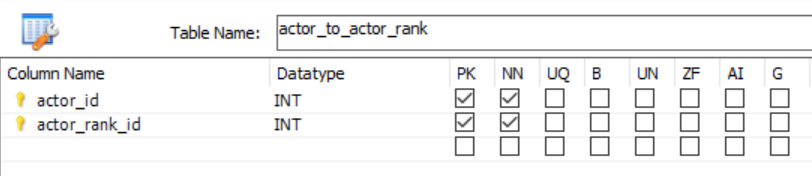


Рисунок 13 – Структура таблицы «actor\_to\_actor\_rank» (Актёр к его званию)

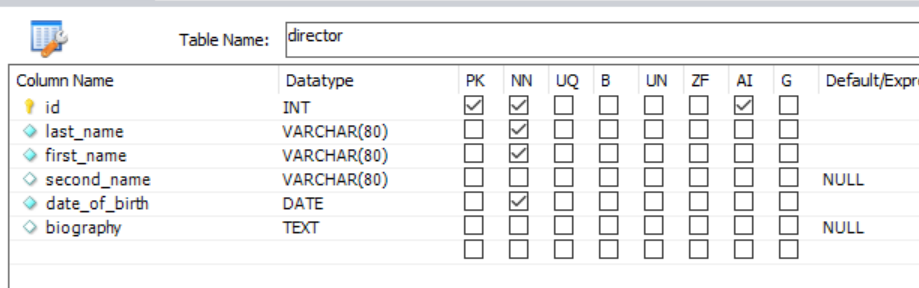


Рисунок 14 – Структура таблицы «director» (Режиссёр)

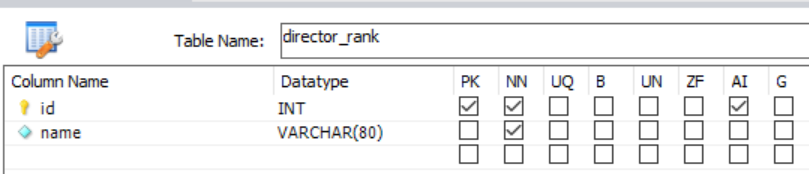


Рисунок 15 – Структура таблицы «director\_rank» (Звание режиссёра)

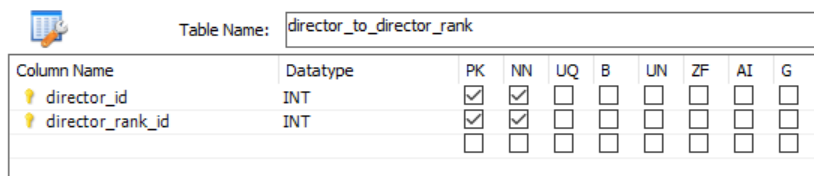


Рисунок 16 – Структура таблицы «director\_to\_director\_rank» (Режиссёр ко званию)



Рисунок 17 – Структура таблицы «scene» (Сцена)

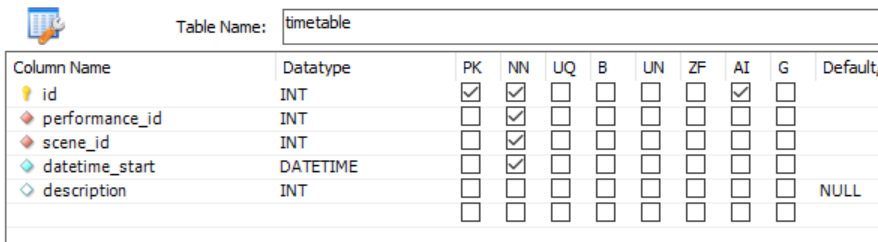


Рисунок 18 – Структура таблицы «timetable» (Афиша)

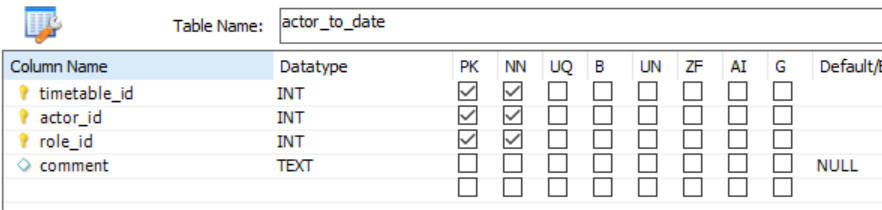


Рисунок 19 – Структура таблицы «actor\_to\_date» (Актёр на дату)

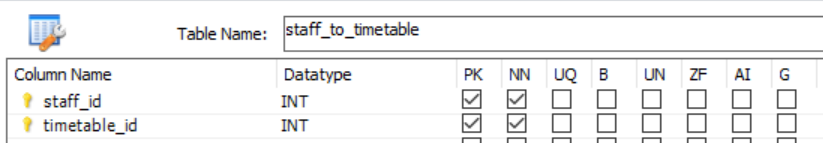


Рисунок 20 – Структура таблицы «staff\_to\_timetable» (Персонал к афише)

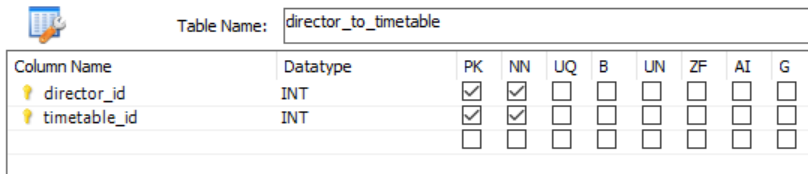


Рисунок 21 – Структура таблицы «director\_to\_timetable» (Режиссёр к афише)

# Импорт данных в базу данных

Задание 1.

В папке Ресурсы имеется файл Excel с именем Фамилии, который нужно импортировать. Мастер импорта работает с форматами csv и json. Необходимо подготовить файл для загрузки с расширением .csv. Из файла Excel он преобразуется в нужный формат командой Сохранить как (Формат CSV-разделители запятые). Выделить базу данных и через контекстное меню выбрать команду Table Data Wizard. Запустится мастер импорта. В окне выбрать необходимый файл для импорта.



Рисунок 22 Содержимое файла с фамилиями

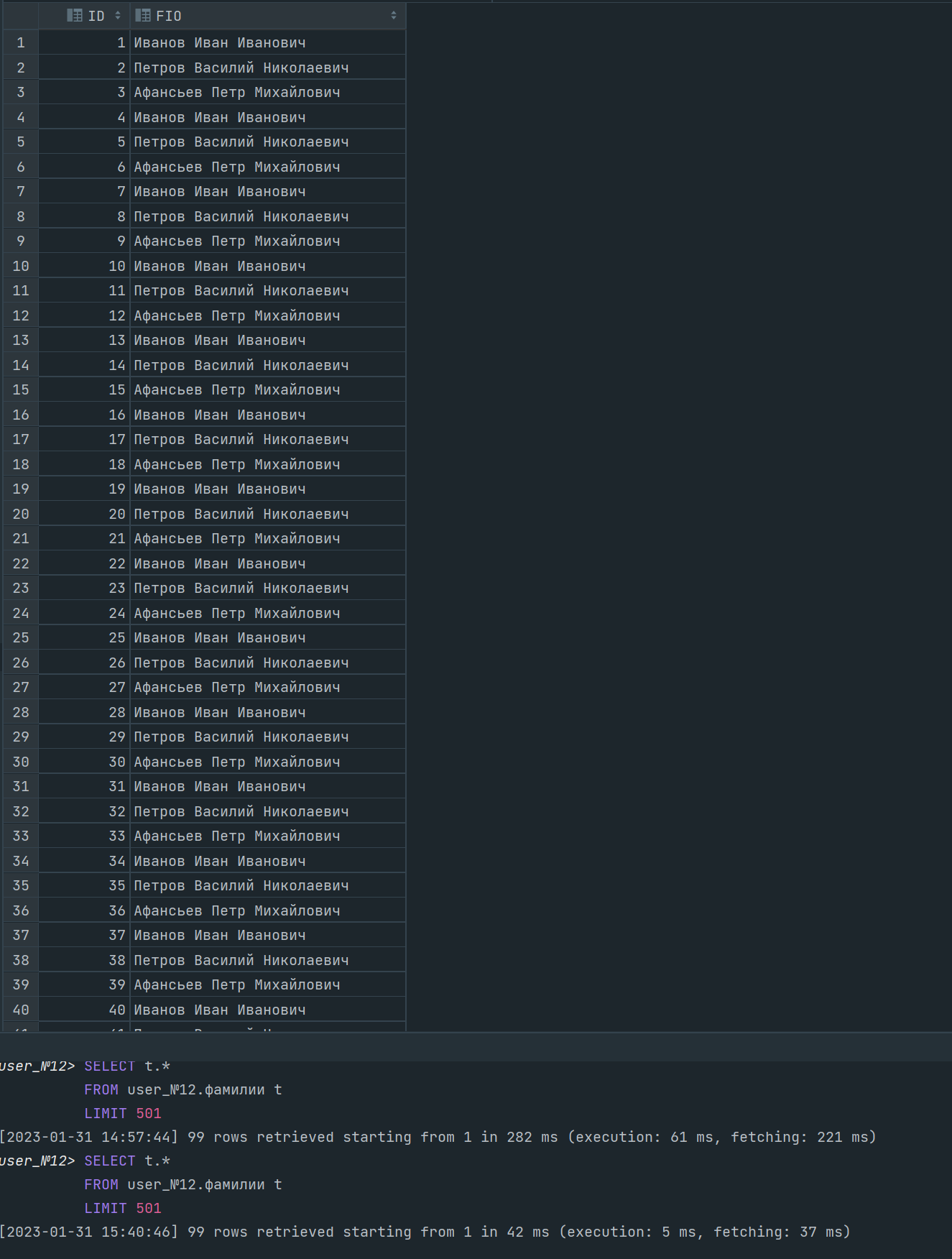


Рисунок 23 Результат импорта Фамилий

Задание 2. В папке Ресурсы имеются файлы для импорта (Материалы и Пользователи). Необходимо выполнить импорт этих файлов в соответствующие таблицы.

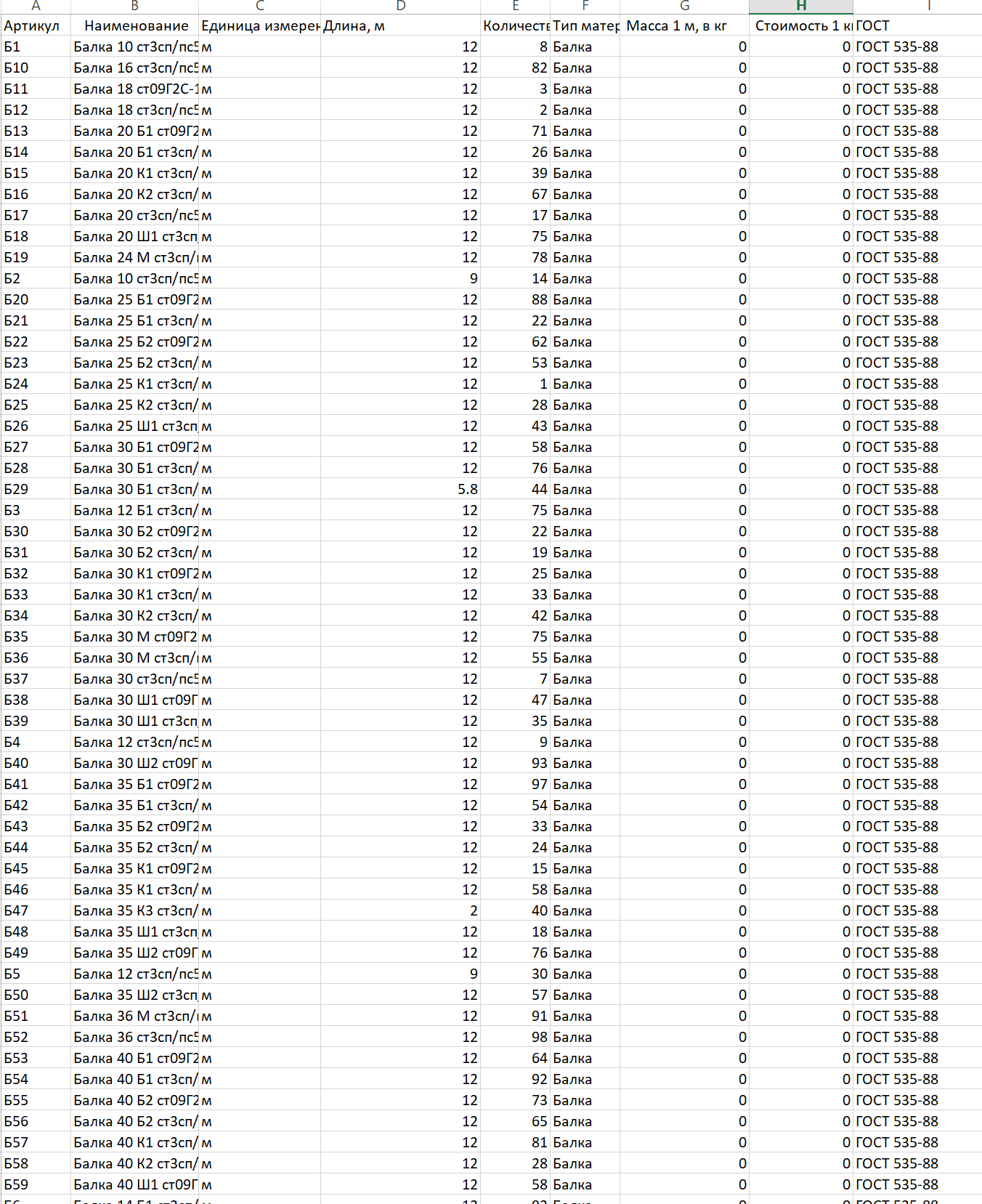


Рисунок 24 Исправленные исходный данные для импорта Материалов



Рисунок 25 Количество импортированных данных в таблицу Материалы

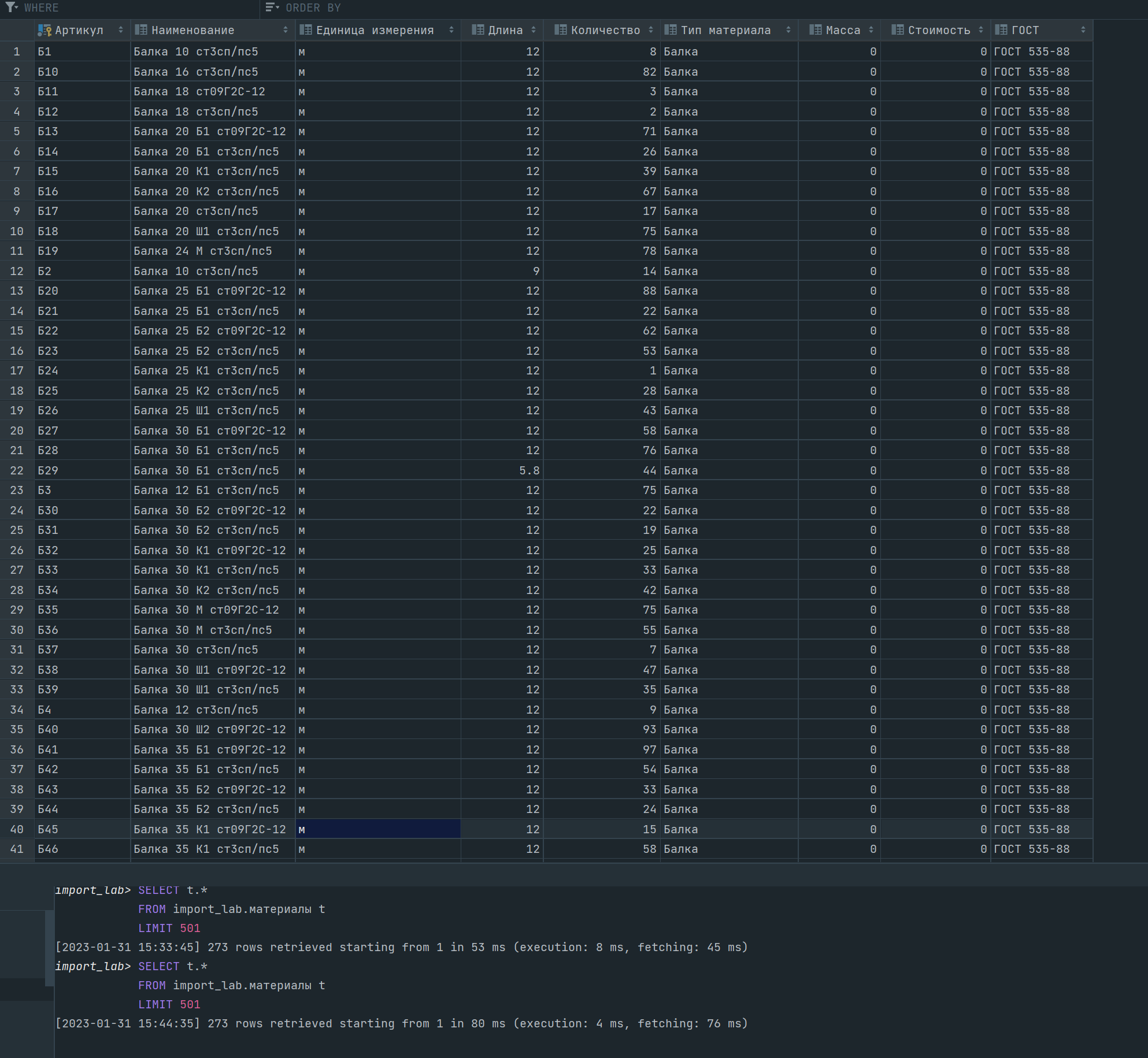


Рисунок 26 Полученная таблица после импорта Материалов

Для пользователей была изменена таблица следующий образом (Рисунок 27):

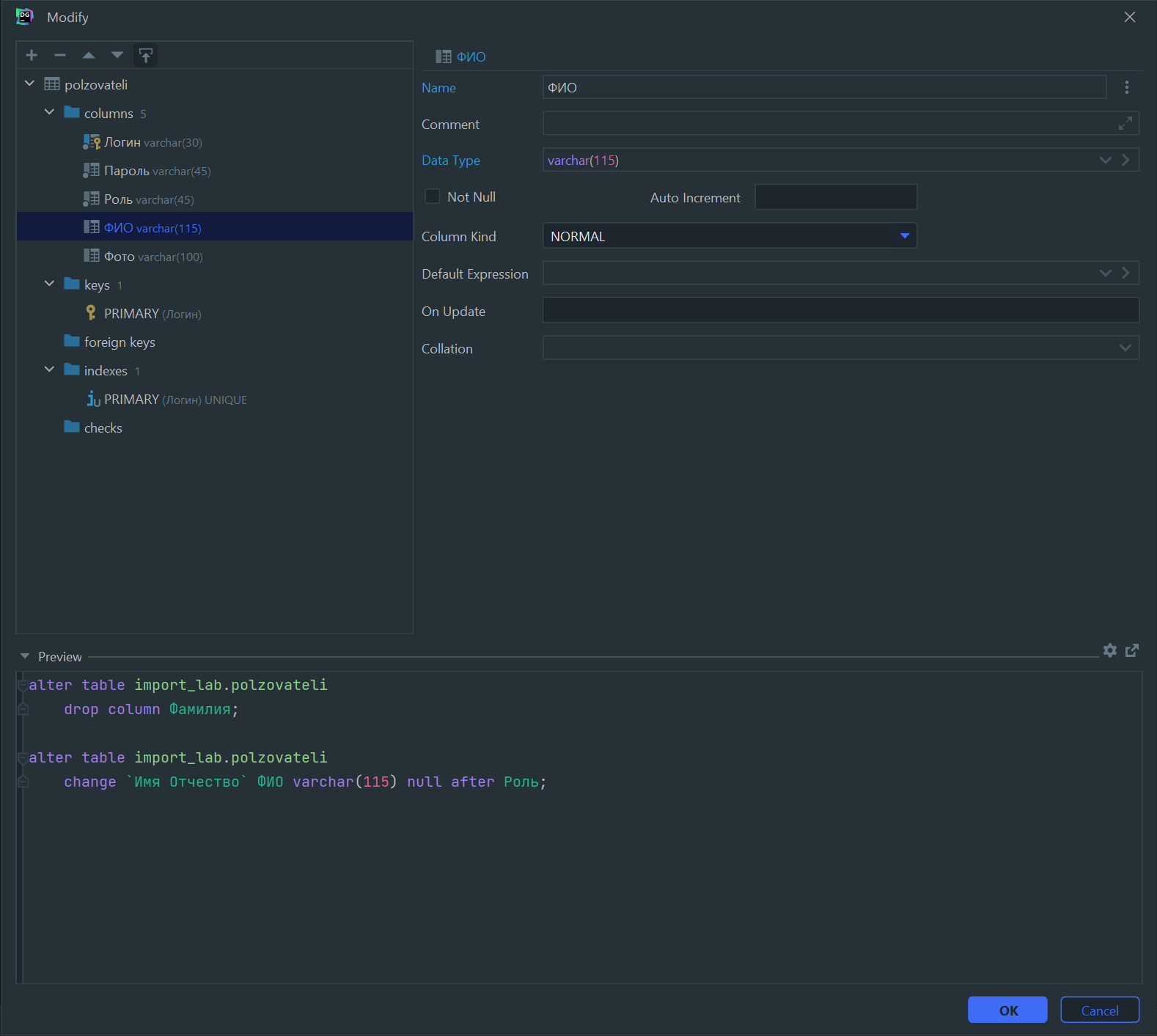


Рисунок 27 Изменение таблицы Пользователи

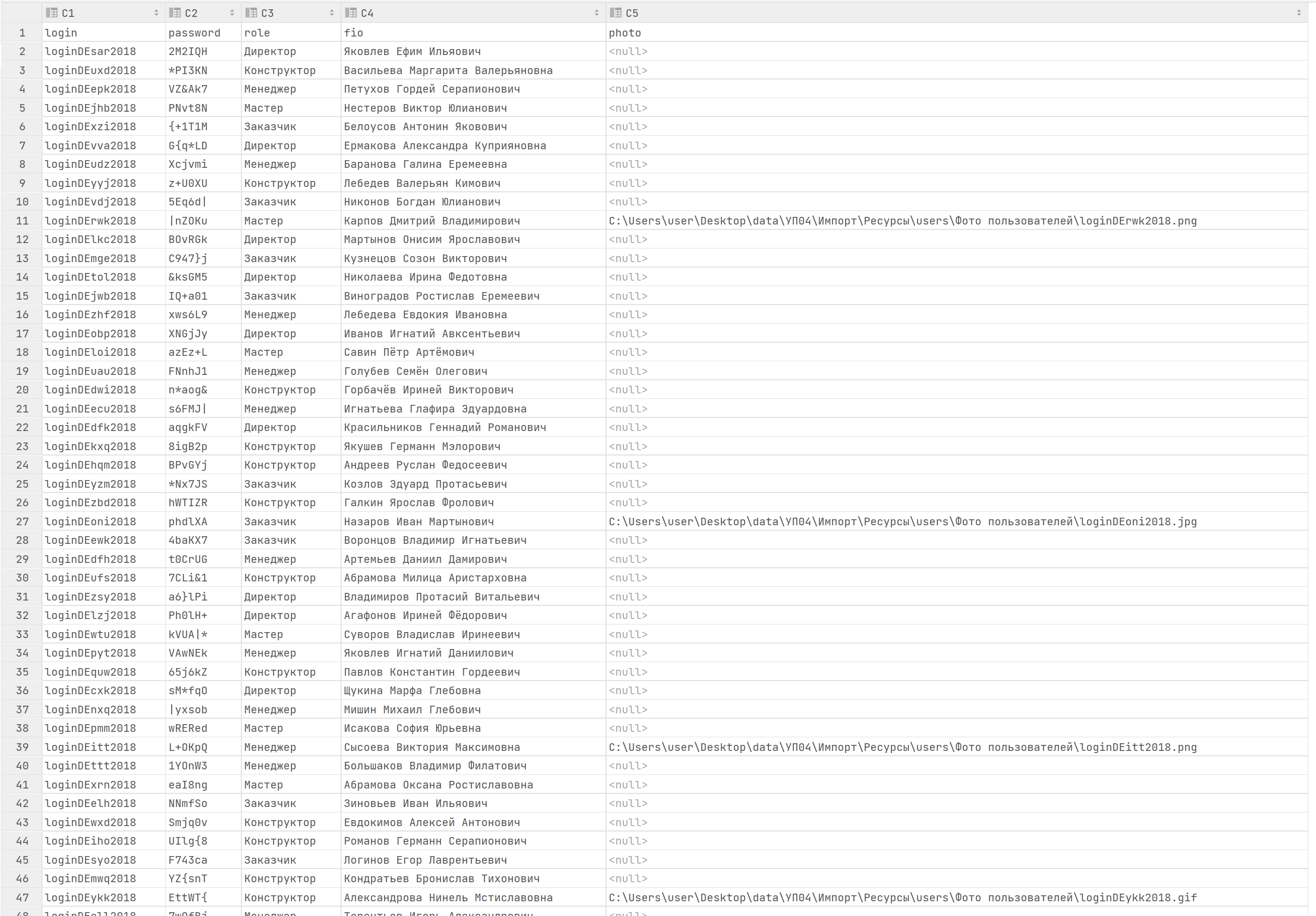


Рисунок 28 Подготовленный данные Пользователи

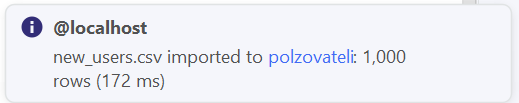


Рисунок 29 Количество добавленных данных в таблицу Пользователи

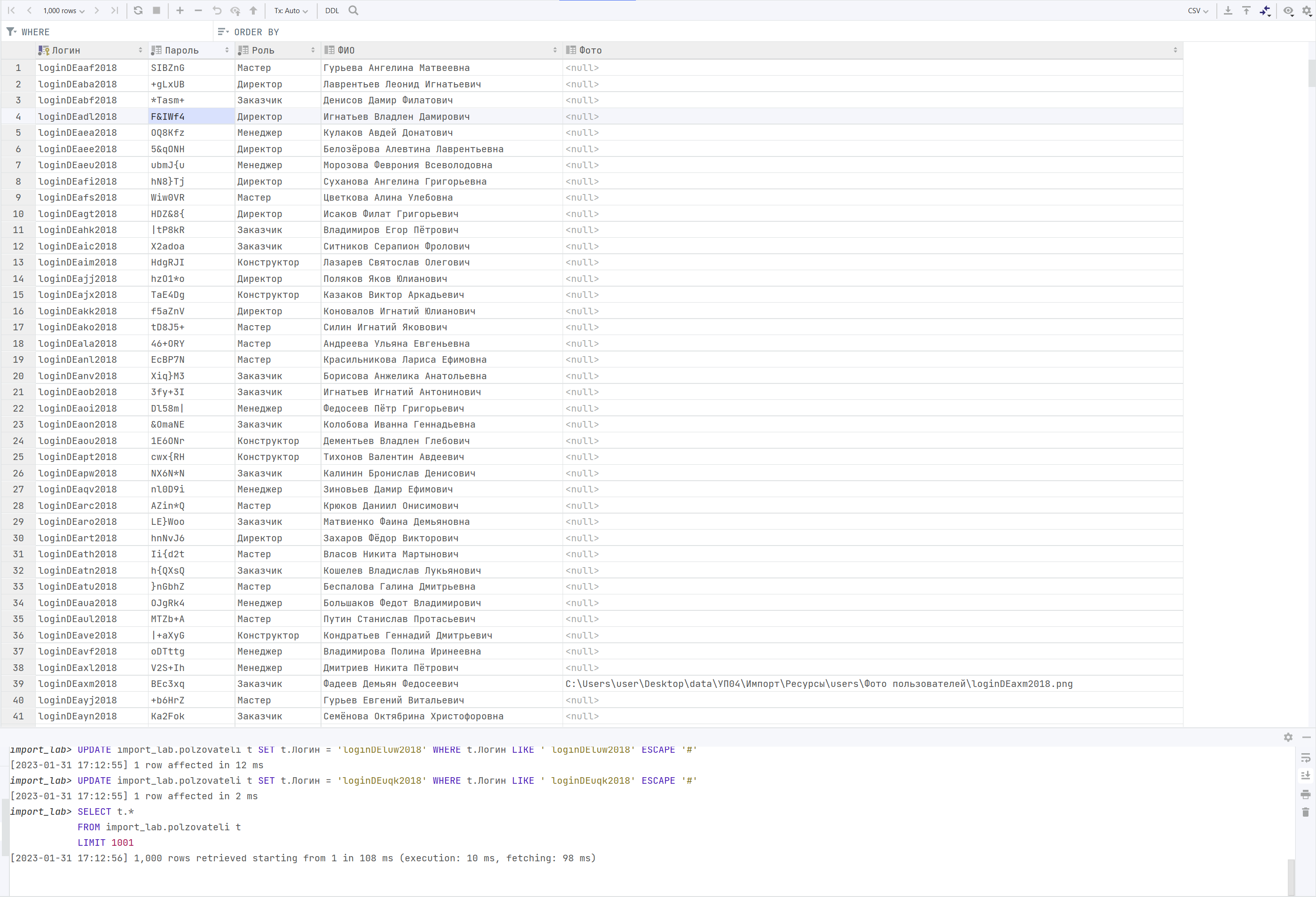


Рисунок 30 Данные в таблице Пользователи

Задание 3: используя представленные ресурсы, выполнить импорт данных в таблицу «client», предварительно выполнив необходимые преобразования.

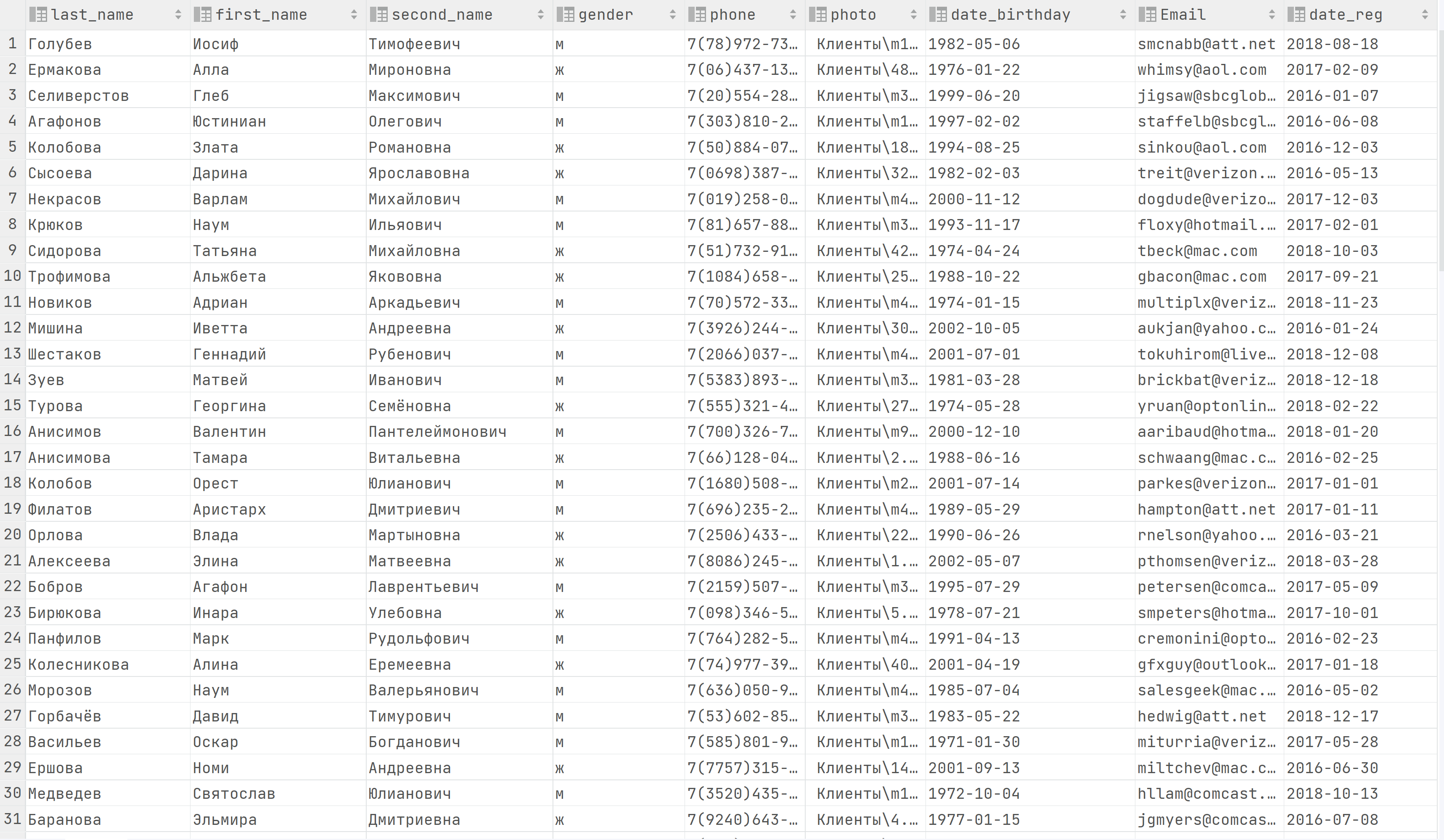


Рисунок 31 Содержимое файла "client"

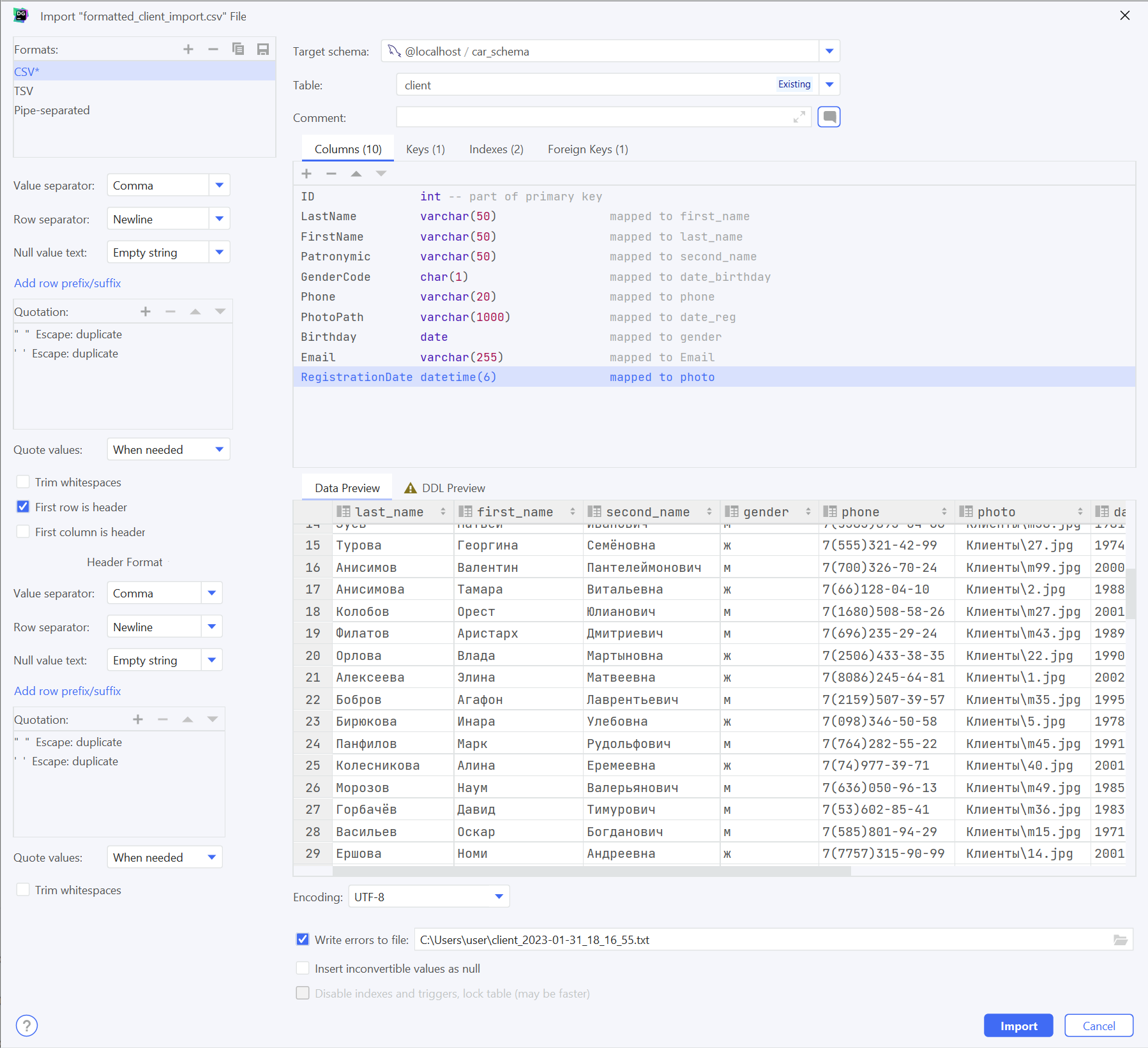


Рисунок 32 Настройки импорта

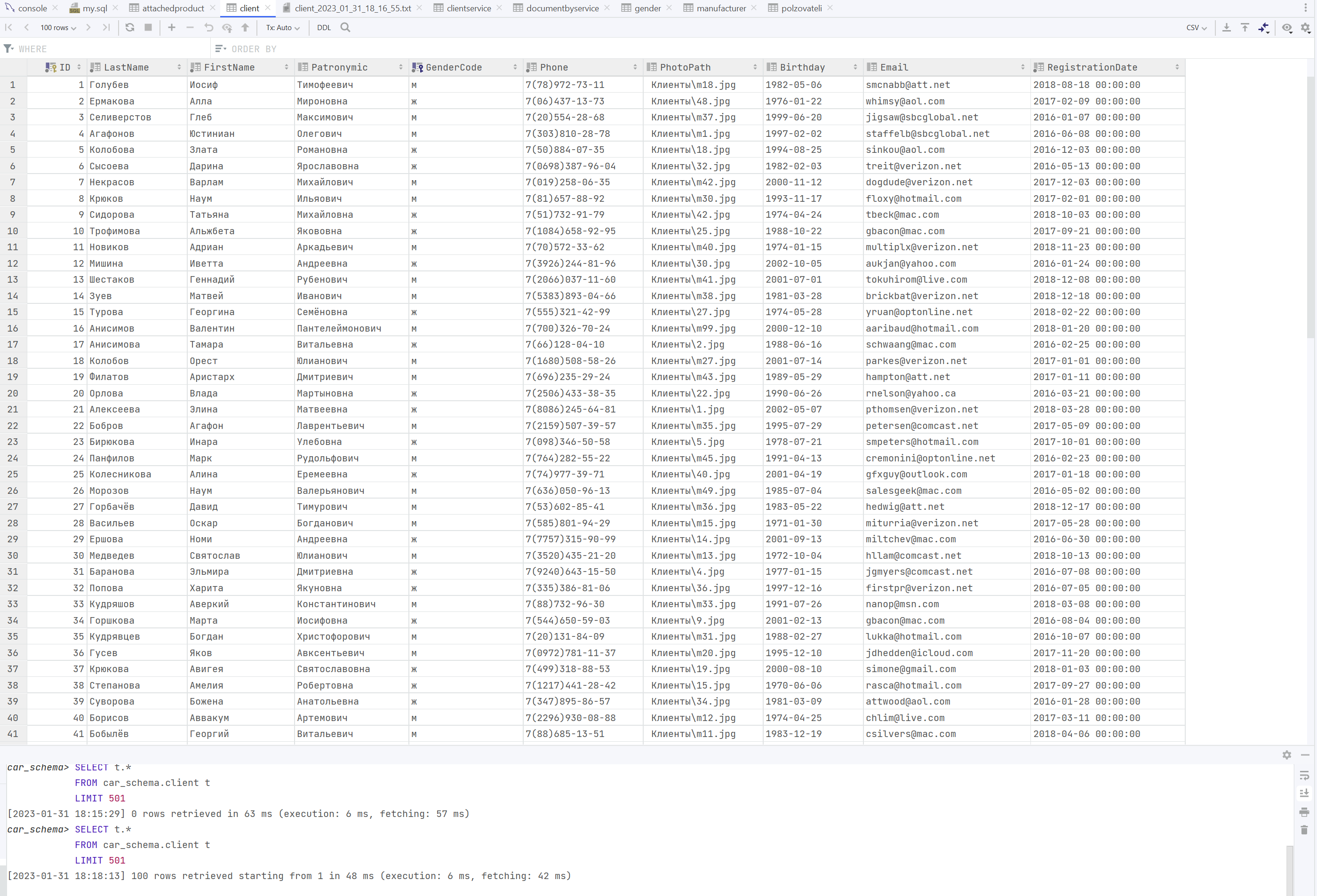


Рисунок 33 Результат импорта в таблицу «client»

Задание 2: используя представленные ресурсы, выполнить импорт данных в таблицу «service», предварительно выполнив необходимые преобразования.

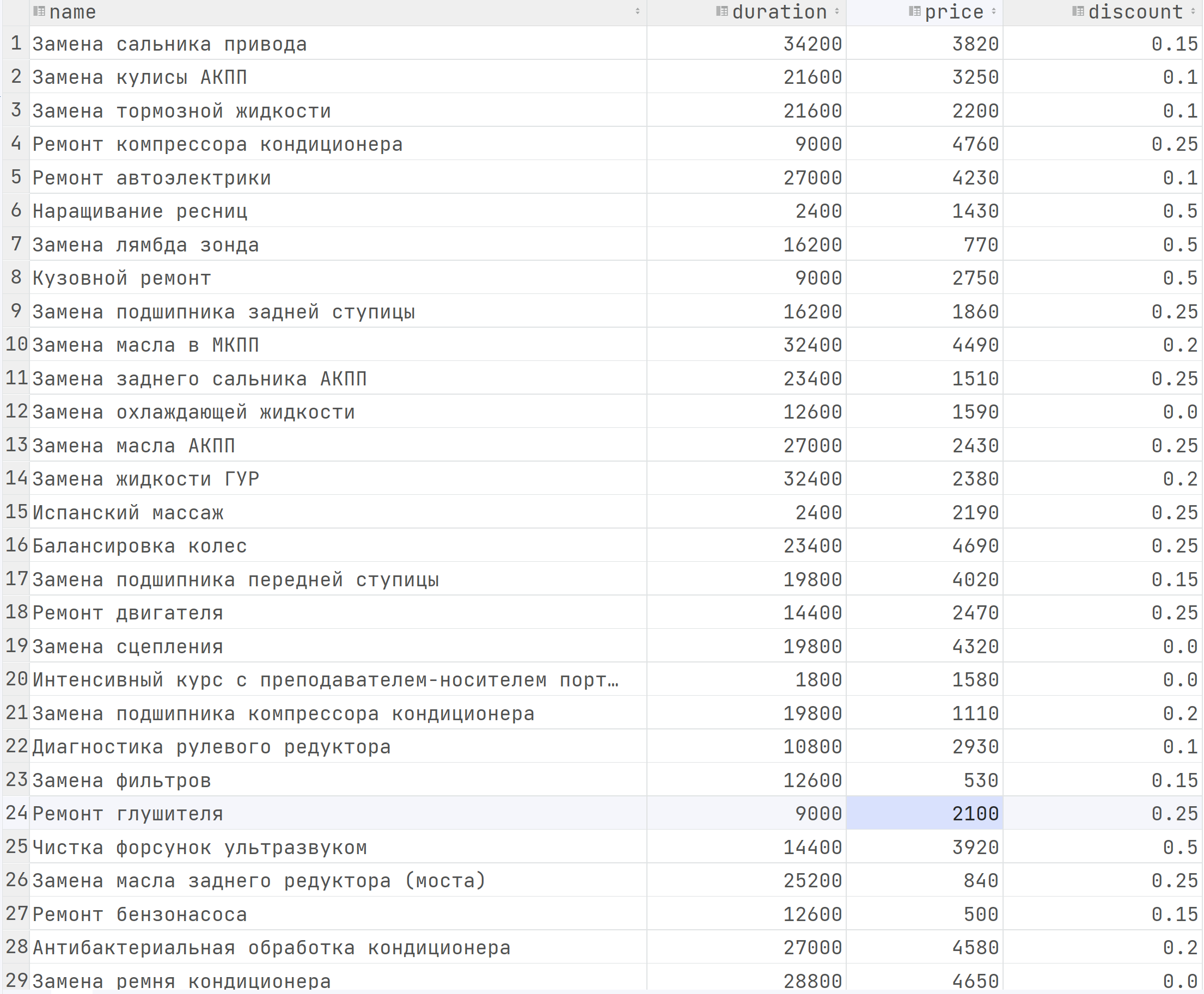


Рисунок 34 Содержимое файла «service»

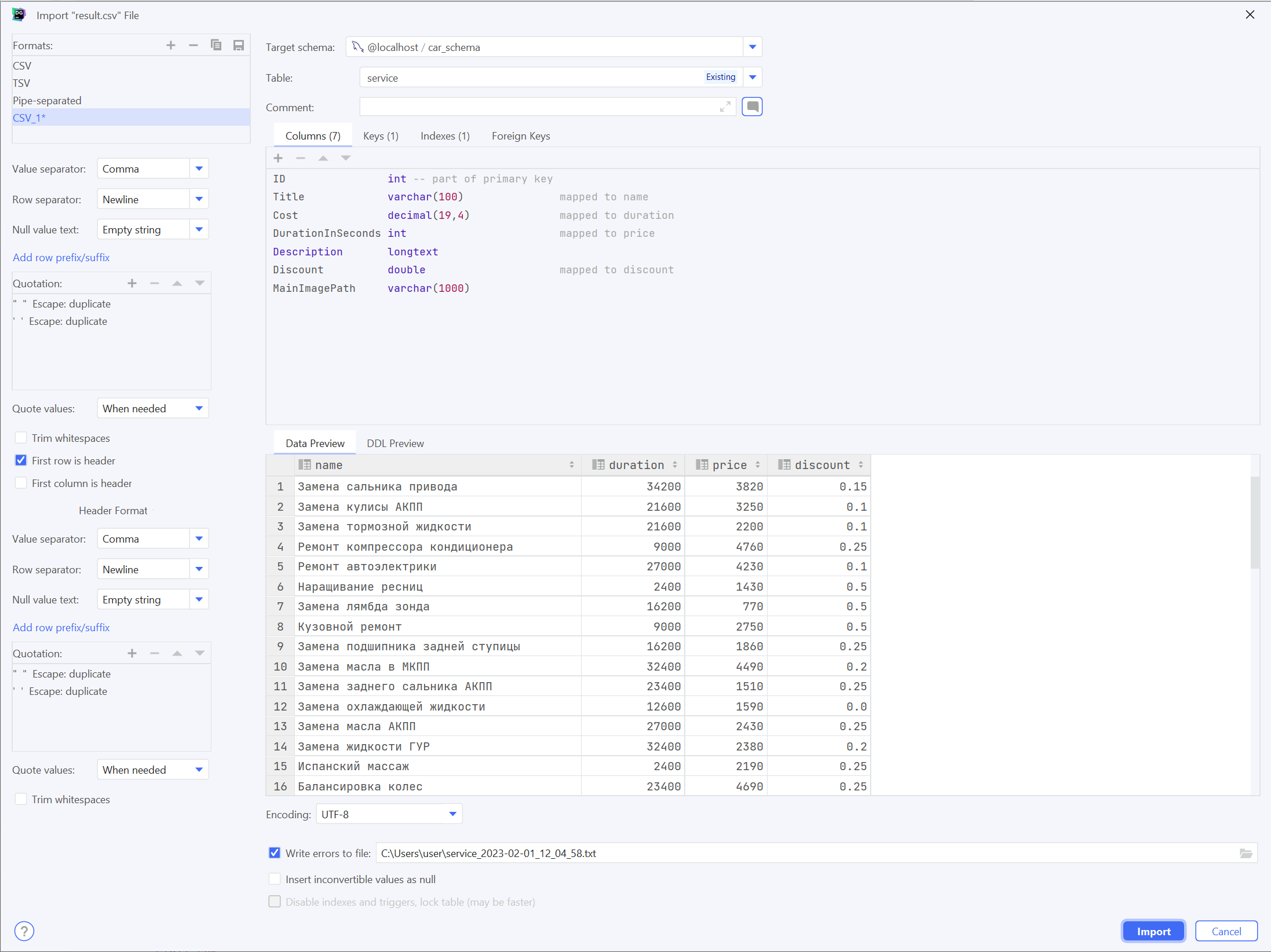


Рисунок 35 Настройки импорта в таблицу «client»

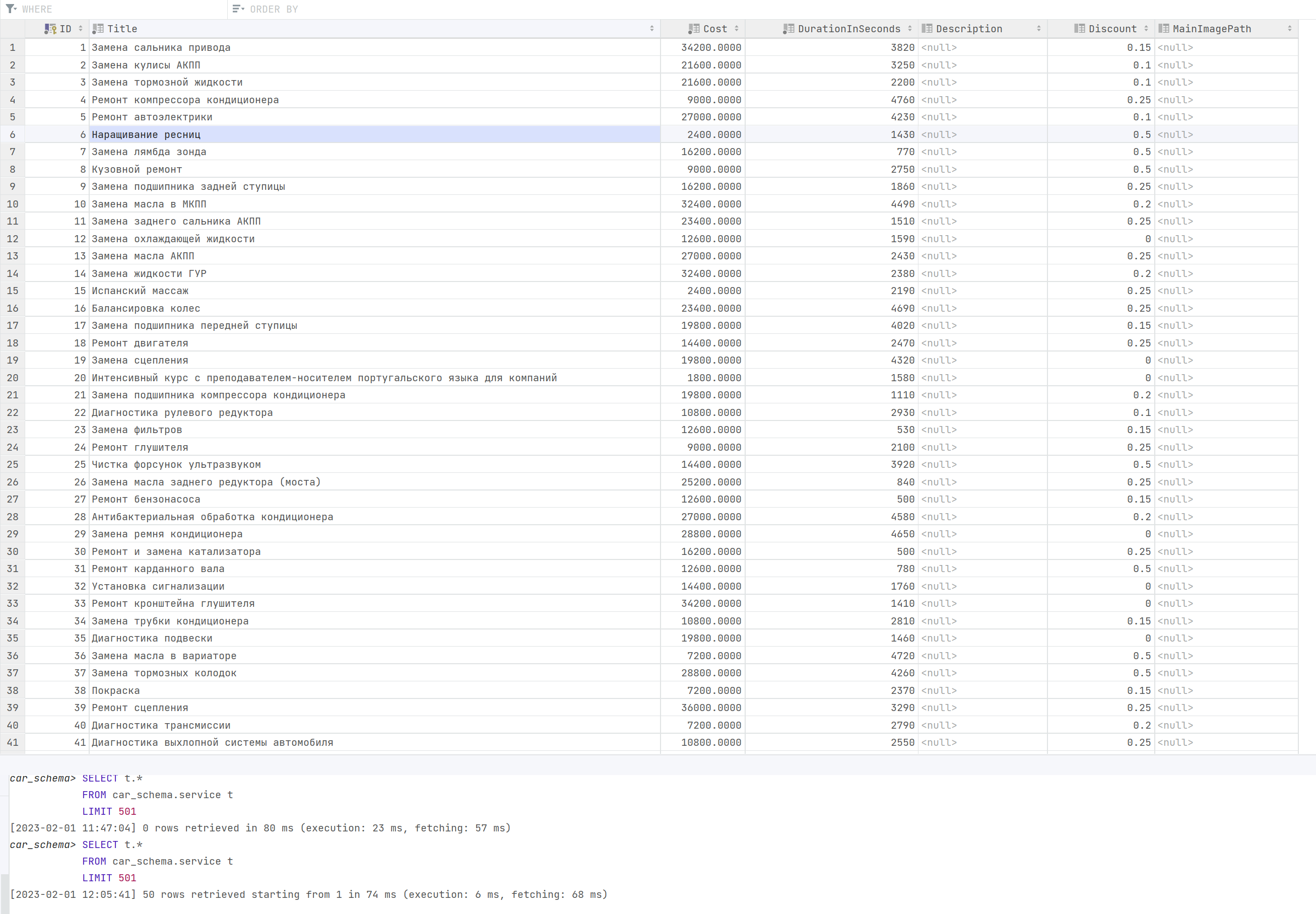


Рисунок 36 Результат импорта в таблицу «service»

Задание 3: используя представленные ресурсы, выполнить импорт данных в таблицу «clientservice», предварительно выполнив необходимые преобразования.

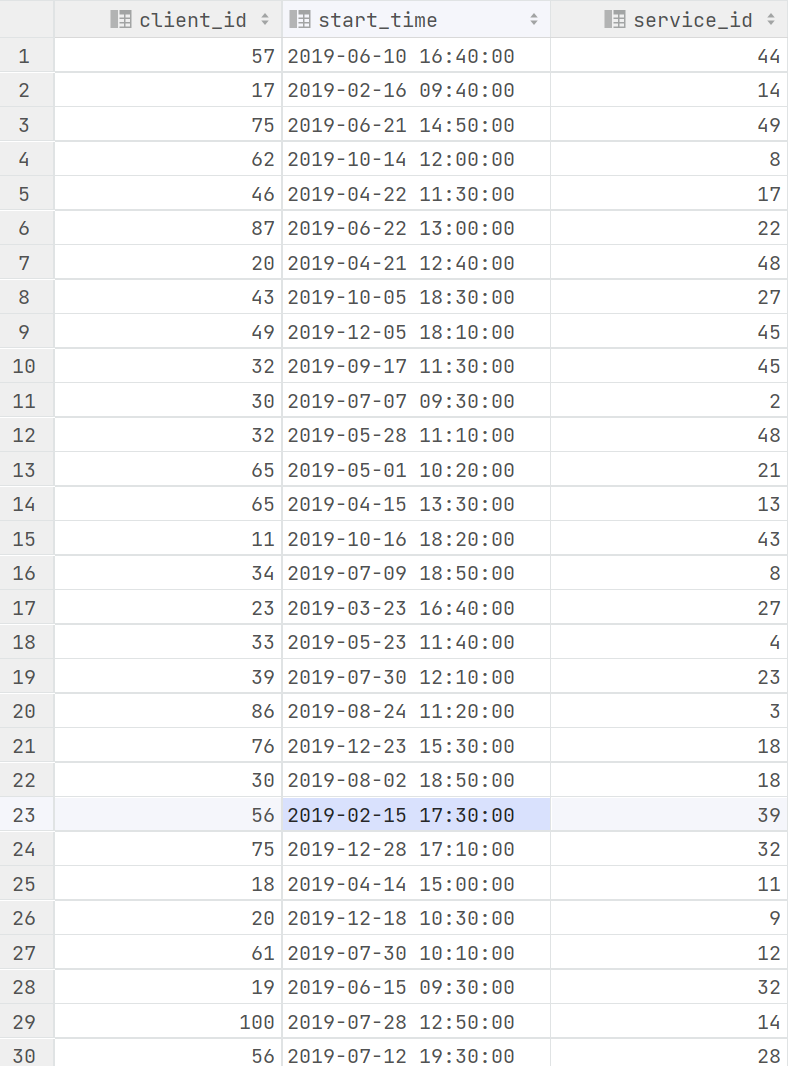


Рисунок 37 Содержимое файла «clientservice»

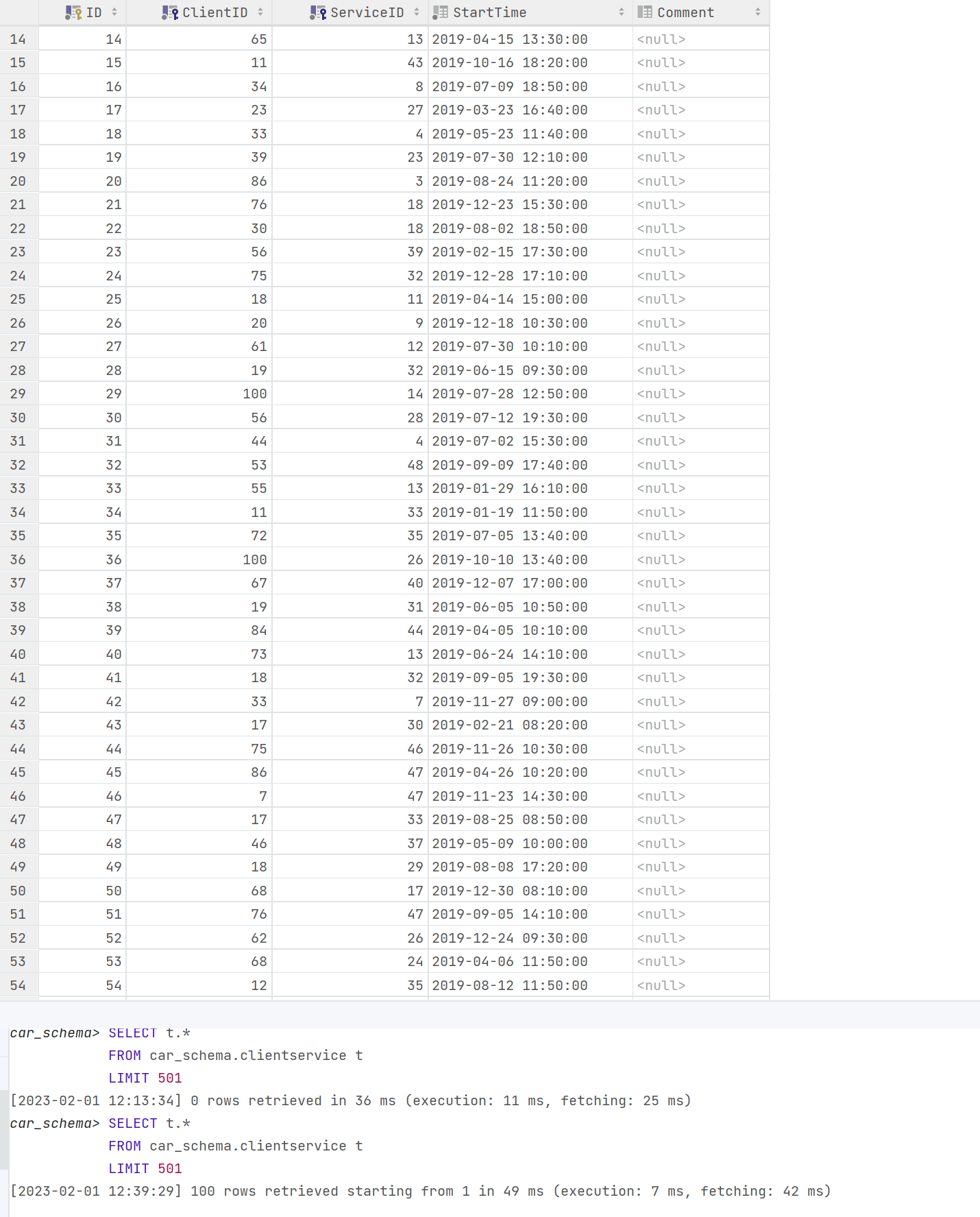


Рисунок 38 Результат импорта в таблицу «clientservice»

# ЗАКЛЮЧЕНИЕ

В период учебной практики по ПМ04 «Разработка, администрирование и защита баз данных» мною получены навыки анализа предметной области, разработки объектной системы и диаграммы Чена, реализации базы данных в СУБД MySQL, с помощью MySQL Workbench 8.0 и Jetbrains DataGrip, а также запросов к базе данных средствами языка DML. Помимо этого, изучил и получил опыт работы, связанный с нормализацией данных в формате CSV для импорта в базу данных.

# ПРИЛОЖЕНИЕ А SQL- скрипт создания базы данных

Ниже перечислен код для создания таблиц в базе данных:

create table pr\_02.actor

(

id int auto\_increment

primary key,

last\_name varchar(80) not null,

first\_name varchar(80) not null,

second\_name varchar(80) null,

date\_of\_birth date not null,

biography text null

);

create table pr\_02.actor\_rank

(

id int auto\_increment

primary key,

name varchar(80) not null

);

create table pr\_02.actor\_to\_actor\_rank

(

actor\_id int not null,

actor\_rank\_id int not null,

primary key (actor\_id, actor\_rank\_id),

constraint actor\_to\_actor\_rank\_actor\_null\_fk

foreign key (actor\_id) references pr\_02.actor (id)

on update cascade,

constraint actor\_to\_actor\_rank\_actor\_rank\_null\_fk

foreign key (actor\_rank\_id) references pr\_02.actor\_rank (id)

on update cascade

);

create table pr\_02.author

(

id int auto\_increment

primary key,

last\_name varchar(80) not null,

first\_name varchar(80) not null,

second\_name varchar(80) null

);

create table pr\_02.director

(

id int auto\_increment

primary key,

last\_name varchar(80) not null,

first\_name varchar(80) not null,

second\_name varchar(80) null,

date\_of\_birth date not null,

biography text null

);

create table pr\_02.director\_rank

(

id int auto\_increment

primary key,

name varchar(80) not null

);

create table pr\_02.director\_to\_director\_rank

(

director\_id int not null,

director\_rank\_id int not null,

primary key (director\_id, director\_rank\_id),

constraint director\_to\_director\_rank\_director\_null\_fk

foreign key (director\_id) references pr\_02.director (id)

on update cascade,

constraint director\_to\_director\_rank\_director\_rank\_null\_fk

foreign key (director\_rank\_id) references pr\_02.director\_rank (id)

on update cascade

);

create table pr\_02.performance\_type

(

id int auto\_increment

primary key,

name varchar(80) not null,

peculiarities text null

);

create table pr\_02.performance

(

id int auto\_increment

primary key,

name text not null,

type\_id int not null,

acts int not null,

duration int not null comment 'В минутах',

constraint perfomance\_performance\_type\_null\_fk

foreign key (type\_id) references pr\_02.performance\_type (id)

on update cascade,

constraint check\_name

check (`acts` > 0)

);

create table pr\_02.author\_performance

(

author\_id int not null,

performance\_id int not null,

primary key (author\_id, performance\_id),

constraint author\_performance\_author\_null\_fk

foreign key (author\_id) references pr\_02.author (id),

constraint author\_performance\_perfomance\_null\_fk

foreign key (performance\_id) references pr\_02.performance (id)

);

create table pr\_02.role

(

id int auto\_increment

primary key,

name text not null,

performance\_id int not null,

description text null,

constraint role\_performance\_null\_fk

foreign key (performance\_id) references pr\_02.performance (id)

on update cascade

);

create table pr\_02.scene

(

id int auto\_increment

primary key,

name varchar(80) not null,

address text not null,

description int null

);

create table pr\_02.staff\_position

(

id int auto\_increment

primary key,

name varchar(80) not null,

description text not null

)

comment 'Должность персонала';

create table pr\_02.staff

(

id int auto\_increment

primary key,

last\_name varchar(80) not null,

first\_name varchar(80) not null,

second\_name varchar(80) null,

position\_id int not null,

constraint staff\_staff\_position\_null\_fk

foreign key (position\_id) references pr\_02.staff\_position (id)

on update cascade

);

create table pr\_02.timetable

(

id int auto\_increment

primary key,

performance\_id int not null,

scene\_id int not null,

datetime\_start datetime not null,

description int null,

constraint timetable\_pk

unique (performance\_id, scene\_id, datetime\_start),

constraint timetable\_performance\_null\_fk

foreign key (performance\_id) references pr\_02.performance (id)

on update cascade,

constraint timetable\_scene\_null\_fk

foreign key (scene\_id) references pr\_02.scene (id)

on update cascade

)

comment 'Афиша';

create table pr\_02.actor\_to\_date

(

timetable\_id int not null,

actor\_id int not null,

role\_id int not null,

comment text null,

primary key (role\_id, actor\_id, timetable\_id),

constraint actor\_to\_date\_actor\_null\_fk

foreign key (actor\_id) references pr\_02.actor (id)

on update cascade,

constraint actor\_to\_date\_role\_null\_fk

foreign key (role\_id) references pr\_02.role (id)

on update cascade,

constraint actor\_to\_date\_timetable\_null\_fk

foreign key (timetable\_id) references pr\_02.timetable (id)

on update cascade

);

create table pr\_02.director\_to\_timetable

(

director\_id int not null,

timetable\_id int not null,

primary key (director\_id, timetable\_id),

constraint director\_to\_timetable\_director\_null\_fk

foreign key (director\_id) references pr\_02.director (id)

on update cascade,

constraint director\_to\_timetable\_timetable\_null\_fk

foreign key (timetable\_id) references pr\_02.timetable (id)

on update cascade

);

create table pr\_02.staff\_to\_timetable

(

staff\_id int not null,

timetable\_id int not null,

primary key (staff\_id, timetable\_id),

constraint staff\_to\_timetable\_staff\_\_fk

foreign key (staff\_id) references pr\_02.staff (id)

on update cascade,

constraint staff\_to\_timetable\_timetable\_null\_fk

foreign key (timetable\_id) references pr\_02.timetable (id)

on update cascade

);

# ПРИЛОЖЕНИЕ В Листинг приложения

Приложение для генерации CSV для импорта на основе данных Пользователи:

from dataclasses import dataclass

from dataclass\_csv import DataclassWriter

import csv

from typing import List

import os

import re

@dataclass

class User:

login: str

password: str

role: str

fio: str = ""

photo: str = ""

DIRECTORY\_NAME = "Фото пользователей"

users: List[User] = []

photos: List[str] = list(filter(lambda f: re.match(r'.\*\.(gif|png|jpg)', f), os.listdir(DIRECTORY\_NAME)))

if \_\_name\_\_ == '\_\_main\_\_':

with open("Пользователи.csv", 'r', encoding='utf-8', newline="") as user\_f:

csvr = csv.reader(user\_f, dialect=csv.excel)

# Пропускаем первую строчку

csvr.\_\_next\_\_()

for el in csvr:

users.append(User(el[2], el[3], el[4], "".join(el[:2])))

try:

# Можно оптимизировать скрипт и проходить по фото и искать к ним пользователей, но тут вопрос,

# что у нас больше фото или пользователей

users[-1].photo = os.path.abspath(

DIRECTORY\_NAME + "\\" + list(filter(lambda f: re.match(rf'{el[2]}\..\*', f), photos))[0])

except IndexError:

pass

with open("new\_users.csv", "w", encoding='utf-8', newline="") as user\_w:

DataclassWriter(user\_w, users, User).write()

Приложение для генерации данных CSV для импорта на основе данных clients:

from datetime import date

from dataclass\_csv import DataclassReader, dateformat, accept\_whitespaces, DataclassWriter

from dataclasses import dataclass, field

from typing import List

@dateformat(' %Y-%m-%d')

@accept\_whitespaces

@dataclass

class Client:

last\_name: str

first\_name: str

second\_name: str

gender: str

phone: str

photo: str

date\_birthday: date

Email: str

date\_reg: date

formatted\_clients: List[Client] = []

if \_\_name\_\_ == '\_\_main\_\_':

with open("stable\_client\_import.csv", "r", encoding="windows-1251", newline='') as f:

dt = DataclassReader(f, Client, dialect='excel')

dt.map("Фамилия").to("last\_name")

dt.map("Имя").to("first\_name")

dt.map("Отчество").to("second\_name")

dt.map("Пол").to("gender")

dt.map("Телефон").to("phone")

dt.map("Фотография клиента").to("photo")

dt.map("Дата рождения").to("date\_birthday")

dt.map("Дата регистрации").to("date\_reg")

for client in dt:

client.first\_name = client.first\_name.lstrip()

client.second\_name = client.second\_name.lstrip()

if len(client.first\_name) == 0:

(client.last\_name, client.first\_name, client.second\_name) = client.last\_name.split()

client.gender = client.gender.lstrip().lower()

if len(client.gender) > 1:

client.gender = client.gender[0]

client.phone = client.phone.lstrip()[:-1]

client.Email = client.Email.lstrip()

formatted\_clients.append(client)

with open("formatted\_client\_import.csv", "w", encoding="utf-8", newline='') as fw:

DataclassWriter(fw, formatted\_clients, Client).write()

Приложение для генерации данных CSV для импорта на основе данных services:

import csv

import re

from dataclasses import dataclass

from typing import List

from dataclass\_csv import DataclassWriter

@dataclass

class Service:

name: str

duration: int

price: int

discount: float

services: List[Service] = []

with open("service\_a\_import.txt", "r", encoding="utf-8", newline="") as f:

reader = csv.reader(f, dialect=csv.excel)

# пропускаем заголовки

reader.\_\_next\_\_()

for el in reader:

new\_el = list(map(lambda l: l.lstrip(), el))

(num, unit) = int(re.findall(r"\d+", new\_el[1])[0]), re.findall(r"[а-я]+\.", new\_el[1])[0]

duration: int

match unit:

case "мин.":

duration = num \* 60

case "час.":

duration = num \* 3600

case \_:

duration = num

price = int(re.findall(r"\d+", new\_el[2])[0])

try:

discount = float("0." + re.findall(r"\d+", new\_el[3])[0])

except IndexError:

discount = 0.

services.append(Service(new\_el[0], duration, price, discount))

with open("result.csv", "w", encoding="utf-8", newline="") as f:

DataclassWriter(f, services, Service).write()

Приложение для генерации данных CSV для импорта на основе данных client\_service:

import csv

import datetime

from dataclasses import dataclass

from dataclass\_csv import DataclassWriter, dateformat

from typing import Dict, List

@dataclass

class ClientService:

client\_id: int

start\_time: datetime.datetime

service\_id: int

id\_last\_names: Dict[str, int] = {}

id\_services: Dict[str, int] = {}

client\_services: List[ClientService] = []

with open("client\_from\_db.csv", "r", encoding="utf-8", newline='') as f:

reader = csv.reader(f, dialect=csv.excel)

reader.\_\_next\_\_()

id\_last\_names = dict(map(lambda row: (row[1], int(row[0])), reader))

with open("service\_from\_db.csv", "r", encoding="utf-8", newline='') as f:

reader = csv.reader(f, dialect=csv.excel)

reader.\_\_next\_\_()

id\_services = dict(map(lambda row: (row[1], int(row[0])), reader))

with open("clientservice\_a\_import.csv", "r", encoding="windows-1251", newline='') as f:

reader = csv.reader(f, dialect=csv.excel)

reader.\_\_next\_\_()

client\_services = list(

map(lambda row: ClientService(id\_last\_names[row[0]], datetime.datetime.strptime(row[1], "%m/%d/%Y %H:%M"),

id\_services[row[2]]), reader))

with open("result.csv", "w", encoding="utf-8", newline='') as fw:

DataclassWriter(fw, client\_services, ClientService)