**Выполнение задания на должность «аналитик-проектировщик»**

Пировских Виктория Николаевна

[pirovskikh\_viktoriya@mail.ru](mailto:pirovskikh_viktoriya@mail.ru)

01.05.2022

**ЗАДАНИЕ 1**

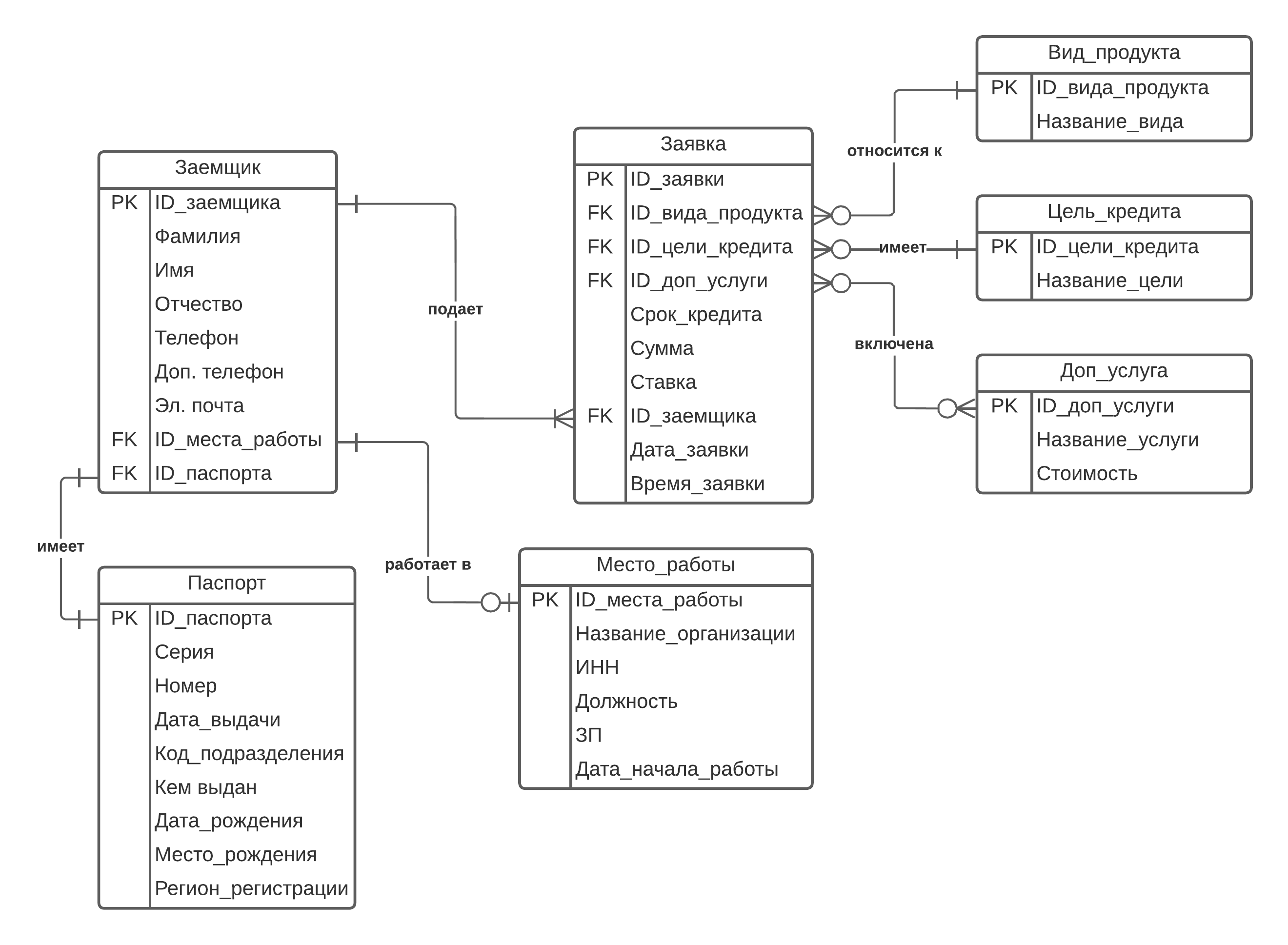


Рисунок 1 – ER-диаграмма

**Пояснение диаграммы**

Я выделила следующие сущности:

1. Заемщик;
2. Паспорт;
3. Место работы;
4. Заявка;
5. Вид продукта;
6. Цель кредита;
7. Дополнительная услуга.

В сущности Заемщик будут персональные данные заемщика. Сущности Паспорт и Место работы также будут относиться к заемщику, но они были выделены, чтобы в случае замены паспорта или смены места работы не нужно было прибегать к изменению персональных данных клиента, в целом так удобнее.

Связи:

1. Заемщик-Паспорт (1:1);
2. Заемщик-Место работы (1:1);
3. Заемщик-Заявка (1:М);
4. Заявка-Вид продукта (1:М);
5. Заявка-Цель кредита (1:М);
6. Заявка-Доп\_услуга (М:М).

Некоторые пояснения:

РАБОТА

Если есть запись о конкретном месте работы, то она обязательно должна принадлежать какому-то заемщику, иначе "ничья" запись не нужна.

Заемщик не обязательно должен иметь работу, даже если и не имеет, это тоже будет учитываться, когда будет приниматься решение о выдаче или невыдаче кредита.

ПАСПОРТ

Заемщик обязательно должен иметь паспорт, а если есть запись о паспорте, она должна обязательно быть привязана к какому-либо заемщику.

ЗАЯВКА

Атрибут "Сумма кредита с учетом доп. услуг" будет излишней, так как это можно вычислить с помощью других атрибутов.

**Типы данных:**

1) Заемщик

ID\_заемщика – int, так как это primary key,

Фамилия varchar(70) – так как часто бывают очень длинные фамилии,

Имя varchar(30) – та же причина,

Отчество varchar(30) – та же причина,

Телефон char(16) – так как телефон часто указывают с символами, например, тире и скобки, то лучше зарезервировать под них место,

Доп\_телефон char(16),

Эл\_почта varchar(100),

ID\_места\_работы int – тип такой же, как и в таблице Место\_работы,

ID\_паспорта int – тип такой же, как и в таблице Паспорт.

2) Заявка

ID\_заявки int, так как это primary key,

ID\_вида\_продукта int – тип такой же, как и в таблице Вид\_продукта,

ID\_цели\_кредита int – тип такой же, как и в таблице Цель\_кредита,

ID\_доп\_услуги int – тип такой же, как и в таблице Доп\_услуга,

Срок\_кредита tinyint – это будет целочисленное, количество месяцев, не слишком большое значение,

Сумма money – здесь будет содержаться денежный формат, тип соответствующий,

Ставка decimal – ставка это вещественное число,

ID\_заемщика int – тип такой же, как и в таблице

Дата\_заявки date – здесь будет содержаться дата, тип соответствующий,

Время\_заявки time – здесь будет содержаться время, тип соответствующий.

3) Паспорт

ID\_паспорта int , так как это primary key,

Серия tinyint – серия это число из 4 цифр,

Номер tinyint – номер это число из 6 цифр,

Дата\_выдачи date – здесь будет содержаться дата, тип соответствующий,

Код\_подразделения varchar(7) – код подразделения это 6 цифр плюс тире,

Кем\_выдан varchar(200),

Дата\_рождения date – здесь будет содержаться дата, тип соответствующий,

Место\_рождения varchar(100),

Регион\_регистрации varchar(100))

4) Место\_работы

ID\_места\_работы int , так как это primary key,

Название\_организации varchar(200),

ИНН int – ИНН это число из 12 цифр,

Должность varchar(100),

ЗП money – здесь будет содержаться денежный формат, тип соответствующий,

Дата\_начала\_работы date – здесь будет содержаться дата, тип соответствующий

Все остальные сущности, в принципе, понятны.

**ЗАДАНИЕ 2**

В качестве СУБД был выбран Microsoft SQL Server (Таблица Список\_заказанных\_услуг – искусственная таблица для обеспечения связи М:М):

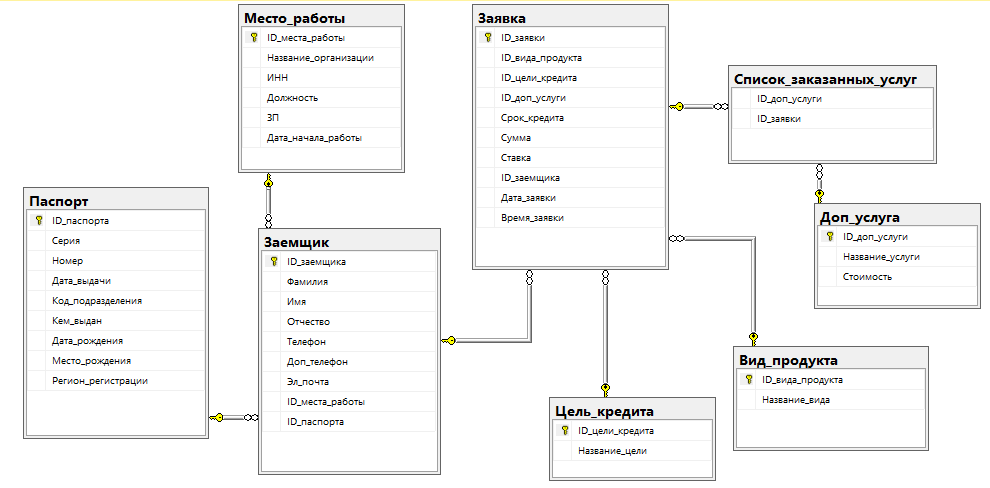


Рисунок 2 – База данных в Microsoft SQL Server

Запрос, возвращающий самый популярный вид продукта за текущий год:

select top 1 Название\_вида as 'Самый популярный вид продукта за текущий год:', Count(Заявка.ID\_вида\_продукта) as 'Количество заявок:'

from Заявка, Вид\_продукта

where Заявка.ID\_вида\_продукта = Вид\_продукта.ID\_вида\_продукта and Дата\_заявки >= '2022-01-01'

group by Заявка.ID\_вида\_продукта, Название\_вида

order by Название\_вида

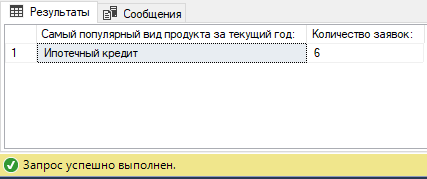


Рисунок 3 – Результат запроса

**ЗАДАНИЕ 3**

Листинг кода на языке С++ (работала вVisual Studio 2019):

#include <iostream>

#include <string>

using namespace std;

void credit(long double sum, long double stavka, int months)

{

int i;

long double everymonth = 0;

long double PS = 0;

long double procents = 0;

long double dolg\_proc = 0;

long double main\_dolg = 0;

long double ostatok\_main\_dolg = 0;

PS = stavka / (100 \* months);

everymonth = sum \* (PS / (1 - pow(1 + PS, -months)));

cout << "Сумма кредита: " << sum << " руб." << endl << "Ставка: " << stavka << "%" << endl << "Срок: " << months << " мес." << endl;

cout << "Месяц | Ежемесячный платеж | Основной долг | Долг по процентам | Остаток основного долга" << endl;

for (i = 1; i <= months; i++)

{

dolg\_proc = sum \* PS;

main\_dolg = everymonth - dolg\_proc;

sum = sum - main\_dolg;

if (i == months) { sum = 0;}

cout << i << " | " << everymonth << " | " << main\_dolg << " | " << dolg\_proc << " | " << sum << endl;

}

}

int main()

{

setlocale(LC\_ALL, "Russian");

long double sum;

long double stavka;

int months;

char wait;

cout << "Введите сумму кредита: " << endl;

cin >> sum;

cout << endl << "Введите процентную ставку: " << endl;

cin >> stavka;

cout << endl << "Введите срок (в месяцах): " << endl;

cin >> months;

credit(sum, stavka, months);

cout << endl << "Ваш кредит рассчитан!" << endl << "Нажмите любую клавишу...";

cin.get();

cin.get();

return 0;

}

Протестировала работу функции со следующими параметрами:

Сумма кредита: 100000 рублей

Ставка: 13%

Срок: 12 месяцев

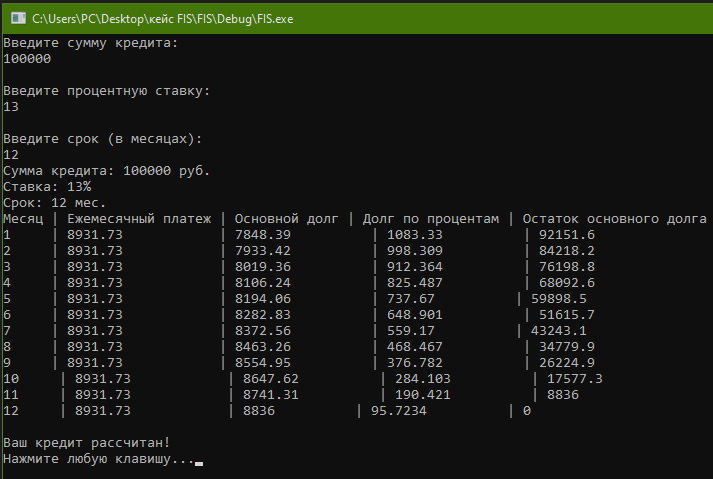


Рисунок 4 – Результат работы функции credit()

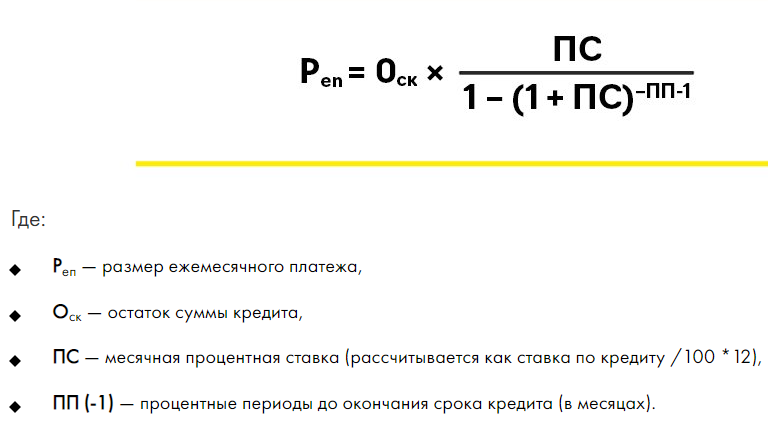
**Логика работы:**

Функция credit() принимает 3 параметра: сумму кредита, ставку и срок. В сущности Заявка из задания 1 как раз присутствуют такие атрибуты.

Сначала отдельно рассчитывается PS – месячная процентная ставка:

PS = ставка / 100 \* срок

Затем рассчитывается everymonth – ежемесячный платеж:



Далее просто выводятся поясняющие надписи.

После этого идет цикл for, который рассчитывает основной долг, долг по процентам и остаток основного долга:

Долг по процентам (dolg\_proc) = Остаток по кредиту (sum) \* Месячная процентная ставка

Основной долг (main\_dolg) = Ежемесячный платеж – долг по процентам

Остаток по кредиту = Остаток по кредиту – Основной долг

Далее, так как известно, что последний платеж аннулирует остаток основного долга (то есть кредит будет выплачен), я поставила условие, чтобы, если месяц был последним, sum бы обнулялся, чтобы избавиться от следующего «мусора»:



На каждой итерации цикла выводятся:

1. месяц выплаты (который и является счетчиком цикла);
2. ежемесячный платеж (который посчитан заранее и неизменен на протяжении всего срока выплаты);
3. основной долг;
4. долг по процентам;
5. остаток основного долга.

Некоторые уточнения:

Если рассматривать массив в качестве структур данных, то можно не только выводить из в консоль, но и записывать в массив.

Связывая с диаграммой из задания 1, стоит уточнить, что при передаче в функцию суммы кредита стоит сначала вычислить полную сумму кредита (в случае, когда выбраны доп. услуги), сложив основную сумму и сумму дополнительных услуг.

PS: Спасибо за интересное задание!