|  |  |  |
| --- | --- | --- |
|  | | |
| Федеральное государственное бюджетное  образовательное учреждение высшего образования Описание: Описание: FPMI_ngtu_neti_rgb_polya«Новосибирский государственный технический университет» | | |
|  | | |
| Кафедра теоретической и прикладной информатики | | |
| Лабораторная работа № 3 | | |
| по дисциплине « Технологии баз данных» | | |
|  | | |
|  | | |
|  | Факультет | фпми |
|  | Группа | пми - 12 |
| Бригада | 4 |
| Вариант | 1 |
| Студенты | КурдюкоВ И.  Омельницкая Е. |
| Преподаватели | стасышина т. л.  сивак м. а. |
|  |  |
| Новосибирск, 2024 | | |

**Цель работы**

Приобрести практические навыки работы со средствами динамического SQL при написании программ на ESQL/C.

**Задача**

Разработать и отладить набор ESQL/С-программ, решающих задачи из соответствующего варианта заданий. Результатом работы программ является одна или несколько строк, которые подлежат выводу на экран с соответствующими пояснительными заголовками.

**Запросы на эталонных данных**

1. Выдать число поставок, выполненных для изделий с деталями зеленого цвета.

SELECT COUNT(\*)

FROM spj

WHERE spj.n\_izd IN(SELECT spj.n\_izd

FROM spj

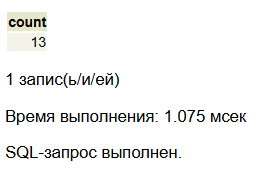
WHERE spj.n\_det IN(SELECT p.n\_det

FROM p

WHERE p.cvet = 'Зеленый'

)

)



1. Поменять местами города, где размещены изделия с самым коротким и самым длинным названием, т. е. изделия с самым коротким названием перевести в город, где размещено изделие с самым длинным названием, и наоборот, изделия с самым длинным названием перевести в город, где размещено изделие с самым коротким названием.

update j set town = (

case

when length(j.name) = (select max(length(name)) from j)

then (select j1.town

from j j1

where length(j1.name) = (select min(length(name)) from j)

order by j1.town

limit 1)

else (select j2.town

from j j2

where length(j2.name) = (select max(length(name)) from j)

order by j2.town

limit 1)

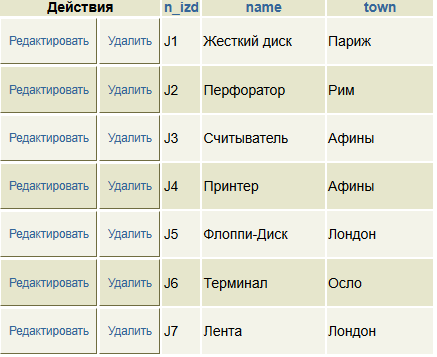
end

)

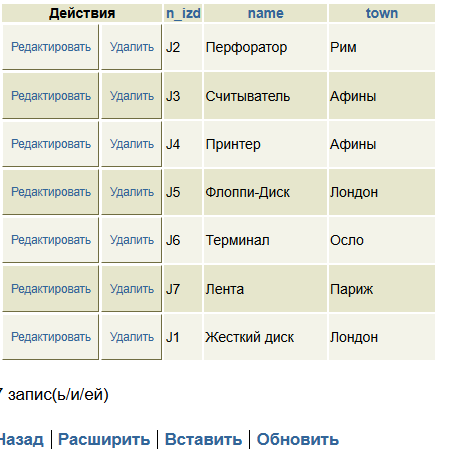
where length(j.name) = (select min(length(name)) from j)

or length(j.name) = (select max(length(name)) from j);

До:



После:



1. Найти детали, имеющие поставки, вес которых меньше среднего веса поставок этой детали для изделий из Лондона.

SELECT DISTINCT spj.n\_det

FROM spj

JOIN p ON spj.n\_det = p.n\_det

WHERE spj.amount \* p.ves < (

SELECT AVG(spj\_inner.amount \* p\_inner.ves)

FROM spj AS spj\_inner

JOIN p AS p\_inner ON spj\_inner.n\_det = p\_inner.n\_det

JOIN j AS j\_inner ON spj\_inner.n\_izd = j\_inner.n\_izd

WHERE j\_inner.town = 'Лондон' AND spj\_inner.n\_det = spj.n\_det

);



1. Выбрать поставщиков, не поставляющих ни одной из деталей, поставляемых поставщиками, находящимися в Лондоне.`

SELECT n\_post

FROM s

EXCEPT

SELECT DISTINCT s1.n\_post

FROM spj s1

JOIN spj s2 ON s1.n\_det = s2.n\_det

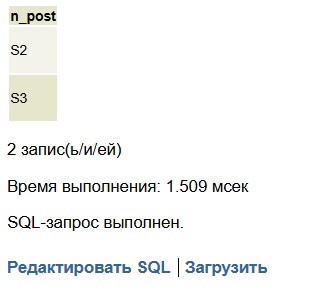
WHERE s2.n\_post IN (

SELECT n\_post

FROM s

WHERE town = 'Лондон'

);



1. Выдать полную информацию о поставщиках, выполнивших поставки ТОЛЬКО с объемом от 200 до 500 деталей.

SELECT \*  
FROM s  
EXCEPT  
SELECT \*  
FROM s  
WHERE s.n\_post IN(SELECT spj.n\_post  
FROM spj  
WHERE spj.amount < 200  
OR spj.amount > 500  
  
)



**Текст разработанной программы**

#include <sqlca.h>

#include <stdio.h>

#include <unistd.h>

#include <string.h>

#define SUCCESS 0

#define AUTH\_EXCEPTION 1

exec SQL begin declare section;

char db\_name[50];

char user\_name[10];

char user\_password[10];

unsigned int count=0;

exec SQL end declare section;

void print\_menu()

{

printf("\n --------------Меню--------------\n");

printf("\n 1 - Выполнить запрос №1 \n");

printf("\n 2 - Выполнить запрос №2 \n");

printf("\n 3 - Выполнить запрос №3 \n");

printf("\n 4 - Выполнить запрос №4 \n");

printf("\n 5 - Выполнить запрос №5 \n");

printf("\n 0 - Выход \n");

}

void close\_program()

{

exec SQL DISCONNECT CURRENT;

printf("Отключено от базы данных\n");

exit(SUCCESS);

}

void print\_table();

void task1();

void task2();

void task3();

void task4();

void task5();

int main()

{

strcpy(db\_name, "students");

strcpy(user\_name, "pmi-b1605");

strcpy(user\_password, "Kit5olya5");

printf("Подключение к базе данных...\n");

//Подключение к базе данных

exec SQL CONNECT TO :db\_name USER :user\_name using :user\_password;

if (sqlca.sqlcode < 0)

{

printf("Ошибка аутентификации. \n");

exit(1);

}

else

{

printf("Подключено к базе данных \"students\" под пользователем %s \n", user\_name);

}

exec SQL SET search\_path TO pmib1605, public;

int cmd = 1;

print\_menu();

while (cmd != 0)

{

printf("Введите команду: ");

scanf("%d", &cmd);

switch (cmd) {

case 1:

task1();

printf("\n 6 - Вывести все меню \n");

break;

case 2:

task2();

printf("\n 6 - Вывести все меню \n");

break;

case 3:

task3();

printf("\n 6 - Вывести все меню \n");

break;

case 4:

task4();

printf("\n 6 - Вывести все меню \n");

break;

case 5:

task5();

printf("\n 6 - Вывести все меню\n");

break;

case 6:

print\_menu();

break;

case 0:

close\_program();

break;

default:

printf("Некорректная программа \n");

printf("\n 6 - Вывести все меню \n");

}

}

return 0;

}

void print\_table()

{

EXEC SQL BEGIN DECLARE SECTION;

char n\_izd[6];

char name[40];

char town[40];

EXEC SQL END DECLARE SECTION;

exec SQL declare curs4 CURSOR for

Select j.\*

from j;

if (sqlca.sqlcode < 0)

{

printf("Ошибка объявления курсора. Код: %d %s \n Текст ошибки: %s\n ",

sqlca.sqlcode,

sqlca.sqlstate,

sqlca.sqlerrm.sqlerrmc);

exec SQL rollback;

return;

}

printf("\nОткрытие курсора...\n");

exec SQL begin;

exec SQL OPEN curs4;

if (sqlca.sqlcode < 0)

{

printf("Ошибка открытия курсора. Код: %d %s \n Текст ошибки: %s\n ",

sqlca.sqlcode,

sqlca.sqlstate,

sqlca.sqlerrm.sqlerrmc);

exec SQL close curs4;

exec SQL rollback;

return;

}

int row\_count = 0;

while (1)

{

exec SQL FETCH curs4 INTO :n\_izd, :name, :town;

if (sqlca.sqlcode < 0) {

printf("Ошибка получения строки! Код: %d (%s)\n Текст ошибки: %s\n",

sqlca.sqlcode,

sqlca.sqlstate,

sqlca.sqlerrm.sqlerrmc);

exec SQL close curs4;

exec SQL rollback;

return;

}

if (sqlca.sqlcode == 100) break;

if (row\_count == 0) printf("n\_izd\tname\t\ttown\n");

printf("%s\t%s\t%s\n", n\_izd, name, town);

row\_count++;

}

printf("Закрытие курсора...\n\n");

exec SQL close curs4;

exec SQL commit;

if(row\_count==0){

printf("Нет данных.\n");

}

printf("Результат: %d строк(-а/и).\n", row\_count);

return;

}

void task1()

{

printf("\n Текст запроса: \n");

printf("1.Выдать число поставок, выполненных для изделий с деталями зеленого цвета.\n\n");

printf("Выполнение запроса...\n");

exec SQL begin;

exec SQL

SELECT COUNT(\*) INTO:count

FROM spj

WHERE spj.n\_izd IN(SELECT spj.n\_izd

FROM spj

WHERE spj.n\_det IN(SELECT p.n\_det

FROM p

WHERE p.cvet = 'Зеленый'

)

);

if (sqlca.sqlcode < 0) {

printf("Ошибка запроса select! Код: %d %s \n Текст ошибки: %s\n ", sqlca.sqlcode, sqlca.sqlstate, sqlca.sqlerrm.sqlerrmc);

exec SQL rollback;

return;

}

printf("Result: %d \n", count);

exec SQL commit;

return;

}

void task2()

{

printf("Текст запроса: \n");

printf("2.Поменять местами города, где размещены изделия с самым коротким и самым длинным названием, т. е. изделия с самым коротким названием перевести в город, где размещено изделие с самым длинным названием, и наоборот, изделия с самым длинным названием перевести в город, где размещено изделие с самым коротким названием.\n\n");

printf("Таблица до выполнения запроса:\n");

print\_table();

exec SQL begin;

printf("Выполение запроса... \n");

exec SQL

update j set town = (

case

when length(j.name) = (select max(length(name)) from j)

then (select j1.town

from j j1

where length(j1.name) = (select min(length(name)) from j)

order by j1.town

limit 1)

else (select j2.town

from j j2

where length(j2.name) = (select max(length(name)) from j)

order by j2.town

limit 1)

end

)

where length(j.name) = (select min(length(name)) from j)

or length(j.name) = (select max(length(name)) from j);

if (sqlca.sqlcode < 0)

{

printf("Ошибка запроса update! Код: %d (%s)\n Текст ошибки%s\n",

sqlca.sqlcode,

sqlca.sqlstate,

sqlca.sqlerrm.sqlerrmc);

exec SQL rollback;

return;

}

printf("Обновлено(-а) %d строк(-и). \n", sqlca.sqlerrd[2]);

exec SQL commit;

printf("\nТаблица после выполнения запроса:\n");

print\_table();

return;

}

void task3()

{

printf("\nТекст запроса: \n");

printf("3. Найти детали, имеющие поставки, вес которых меньше среднего веса поставок этой детали для изделий из Лондона.\n\n");

printf("Объявление курсора...\n");

EXEC SQL BEGIN DECLARE SECTION;

char n\_det[6];

int amount;

float avg;

EXEC SQL END DECLARE SECTION;

exec SQL declare curs1 CURSOR for

SELECT DISTINCT spj.n\_det

FROM spj

JOIN p ON spj.n\_det = p.n\_det

WHERE spj.amount \* p.ves < (

SELECT AVG(spj\_inner.amount \* p\_inner.ves)

FROM spj AS spj\_inner

JOIN p AS p\_inner ON spj\_inner.n\_det = p\_inner.n\_det

JOIN j AS j\_inner ON spj\_inner.n\_izd = j\_inner.n\_izd

WHERE j\_inner.town = 'Лондон' AND spj\_inner.n\_det = spj.n\_det

);

if (sqlca.sqlcode < 0)

{

printf("Ошибка обявления курсора! Код: %d (%s)\n Текст ошибки: %s\n",

sqlca.sqlcode,

sqlca.sqlstate,

sqlca.sqlerrm.sqlerrmc);

exec SQL rollback;

return;

}

printf("Открытие курсора...\n");

exec SQL begin;

exec SQL OPEN curs1;

if (sqlca.sqlcode < 0)

{

printf("Ошибка открытия курсора! Код: %d (%s)\n Текст ошибки: %s\n",

sqlca.sqlcode,

sqlca.sqlstate,

sqlca.sqlerrm.sqlerrmc);

exec SQL close curs1;

exec SQL rollback;

return;

}

printf("Получение результата...\n");

int row\_count = 0;

while (1)

{

exec SQL FETCH curs1 INTO :n\_det;

if (sqlca.sqlcode < 0) {

printf("Ошибка получения строки! Код: %d (%s)\n Текст ошибки: %s\n",

sqlca.sqlcode,

sqlca.sqlstate,

sqlca.sqlerrm.sqlerrmc);

exec SQL close curs1;

exec SQL rollback;

return;

}

if (sqlca.sqlcode == 100) break;

if (row\_count == 0) printf("n\_det\n");

printf("%s\n", n\_det);

row\_count++;

}

printf("Закрытие курсора...\n");

exec SQL close curs1;

exec SQL commit;

if (row\_count == 0) printf("Данные не найдены.");

else printf("Результат: %d строк(-а/и).\n", row\_count);

return;

}

void task4()

{

printf("Текст запроса: \n");

printf("4.Выбрать поставщиков, не поставляющих ни одной из деталей, поставляемых поставщиками, находящимися в Лондоне.\n\n");

printf("Объявление курсора...\n");

EXEC SQL BEGIN DECLARE SECTION;

char n\_post[6];

EXEC SQL END DECLARE SECTION;

exec SQL declare curs2 CURSOR for

SELECT n\_post

FROM s

EXCEPT

SELECT DISTINCT s1.n\_post

FROM spj s1

JOIN spj s2 ON s1.n\_det = s2.n\_det

WHERE s2.n\_post IN (SELECT n\_post

FROM s

WHERE town = 'Лондон'

);

if (sqlca.sqlcode < 0)

{

printf("Ошибка объявления курсора! Код: %d (%s)\n Текст ошибки: %s\n",

sqlca.sqlcode,

sqlca.sqlstate,

sqlca.sqlerrm.sqlerrmc);

exec SQL rollback;

return;

}

printf("Открытие курсора...\n");

exec SQL begin;

exec SQL OPEN curs2;

if (sqlca.sqlcode < 0)

{

printf("Ошибка открытия курсора! Код: %d(%s)\n Текст ошибки: %s\n",

sqlca.sqlcode,

sqlca.sqlstate,

sqlca.sqlerrm.sqlerrmc);

exec SQL close curs2;

exec SQL rollback;

return;

}

printf("Получение результата...\n");

int row\_count = 0;

while (1)

{

exec SQL FETCH curs2 INTO :n\_post;

if (sqlca.sqlcode < 0) {

printf("Ошибка получения строки! Код: %d (%s)\n Текст ошибки: %s\n",

sqlca.sqlcode,

sqlca.sqlstate,

sqlca.sqlerrm.sqlerrmc);

exec SQL close curs2;

exec SQL rollback;

return;

}

if (sqlca.sqlcode == 100) break;

if (row\_count == 0) printf("n\_post\n");

printf("%s\n", n\_post);

row\_count++;

}

printf("Закрытие курсора...\n");

exec SQL close curs2;

exec SQL commit;

if (row\_count == 0) printf("Данные не найдены.");

else printf("Результат: %d строк(-а/и).\n", row\_count);

return;

}

void task5()

{

printf("\nТекст запроса: \n");

printf("5.Выдать полную информацию о поставщиках, выполнивших поставки ТОЛЬКО с объемом от 200 до 500 деталей.\n");

printf("Объявление курсора...\n");

EXEC SQL BEGIN DECLARE SECTION;

char n\_post[6];

char name[40];

char reiting[6];

char town[40];

int name\_;

int town\_;

int reiting\_;

EXEC SQL END DECLARE SECTION;

exec SQL

declare curs3 CURSOR for

SELECT \*

FROM s

EXCEPT

SELECT \*

FROM s

WHERE s.n\_post IN(SELECT spj.n\_post

FROM spj

WHERE spj.amount < 200

OR spj.amount > 500 );

if (sqlca.sqlcode < 0)

{

printf("Ошибка объявления курсора! Код: %d (%s)\n Текст ошибки: %s\n",

sqlca.sqlcode,

sqlca.sqlstate,

sqlca.sqlerrm.sqlerrmc);

exec SQL rollback;

return;

}

printf("Открытие курсора...\n");

exec SQL begin;

exec SQL OPEN curs3;

if (sqlca.sqlcode < 0)

{

printf("Ошибка открытия курсора! Код: %d(%s)\n Текст ошибки: %s\n",

sqlca.sqlcode,

sqlca.sqlstate,

sqlca.sqlerrm.sqlerrmc);

exec SQL close curs3;

exec SQL rollback;

return;

}

printf("Получение результата...\n");

int row\_count = 0;

while (1)

{

exec SQL FETCH curs3 INTO :n\_post, :name indicator name\_,:reiting indicator reiting\_, :town indicator town\_;

if (sqlca.sqlcode < 0) {

printf("Ошибка получения строки! Код: %d (%s)\n Текст ошибки: %s\n",

sqlca.sqlcode,

sqlca.sqlstate,

sqlca.sqlerrm.sqlerrmc);

exec SQL close curs3;

exec SQL rollback;

return;

}

if (sqlca.sqlcode == 100) break;

if (row\_count == 0) printf("n\_post\tname\t\t\ttown\n");

if(name\_<0) strcpy(name,"Нет данных");

if(town\_<0) strcpy(town,"Нет данных");

if(reiting\_<0) strcpy(reiting,"Нет данных");

printf("%s\t%s\t%s\t%s\n", n\_post, name, reiting, town);

row\_count++;

}

printf("Закрытие курсора...\n");

exec SQL close curs3;

exec SQL commit;

if (row\_count == 0) printf("Данные не найдены.");

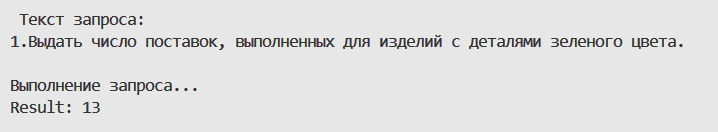
else printf("Результат: %d строк(-а/и).\n", row\_count);

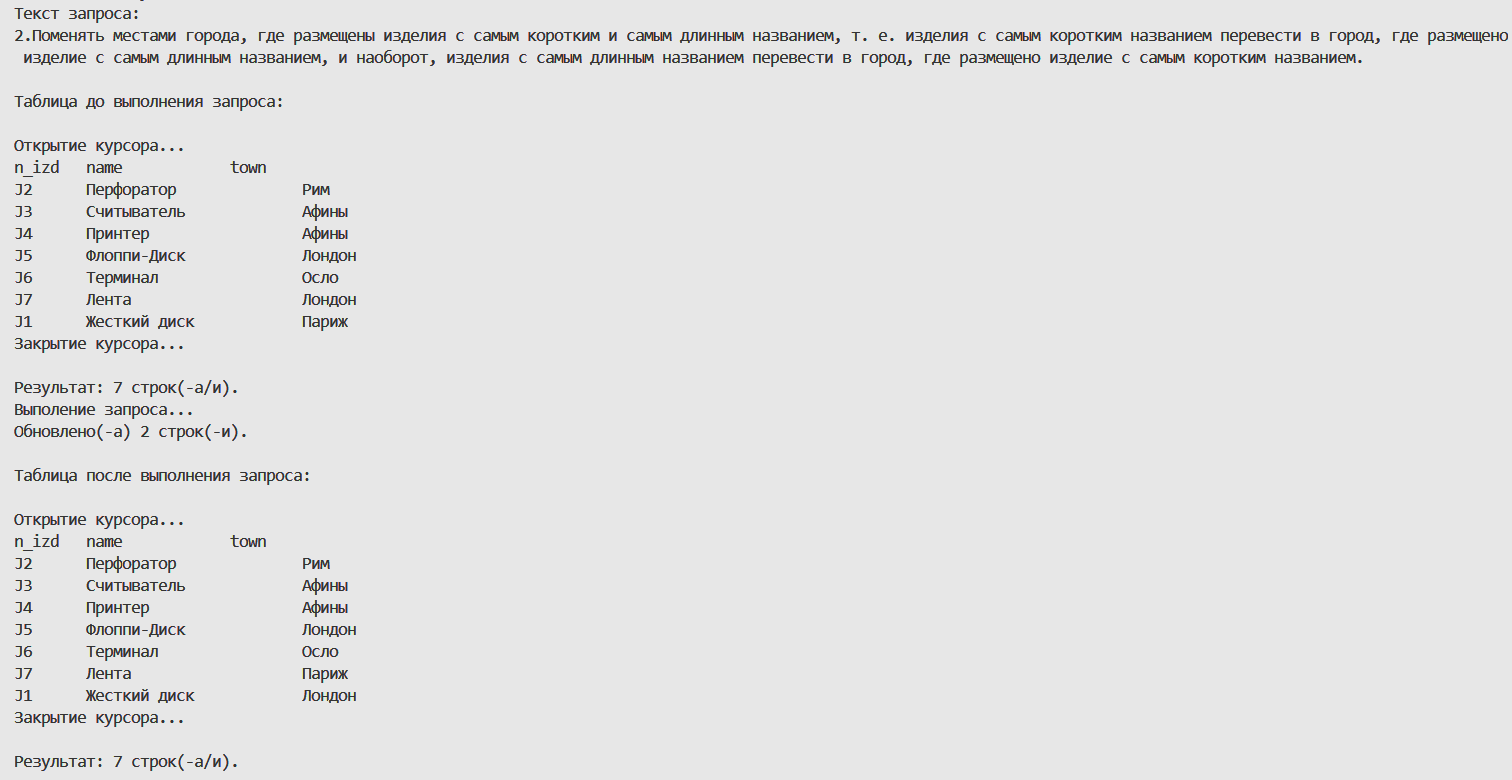
//return;

}

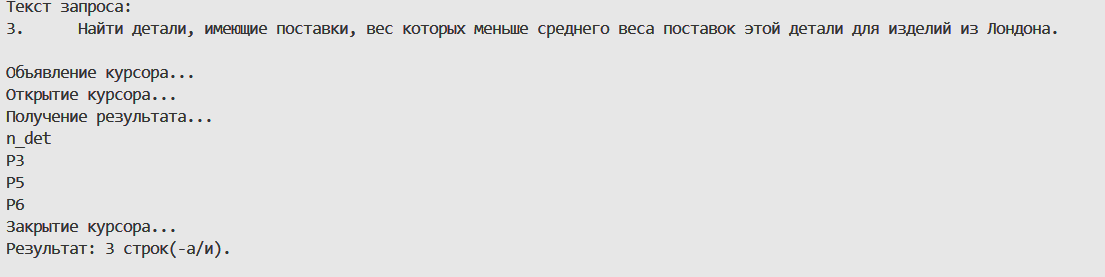
Тесты программы

1.Первый запрос

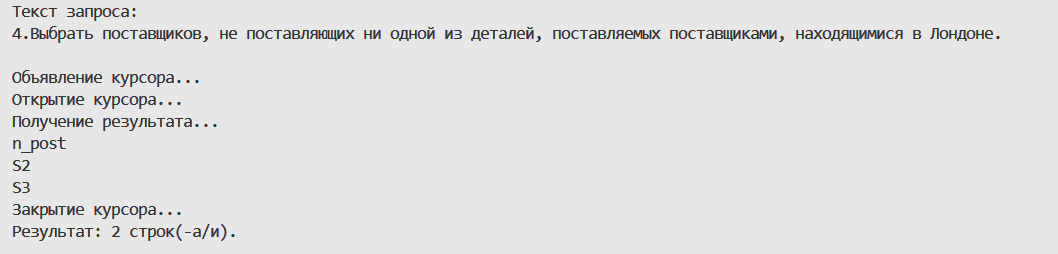


2.Второй запрос

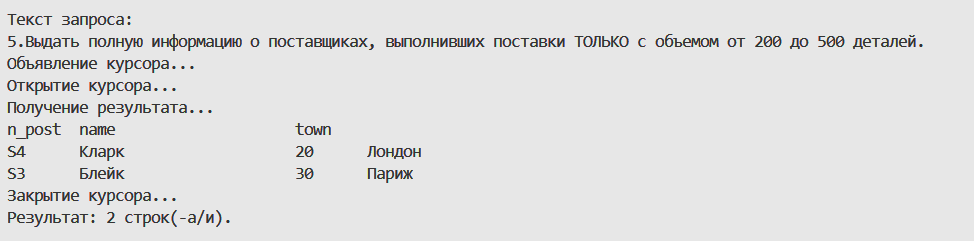
3.Третий запрос



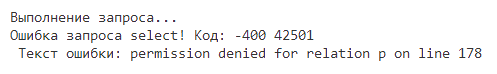
4.Четвертый запрос



5.Пятый запрос

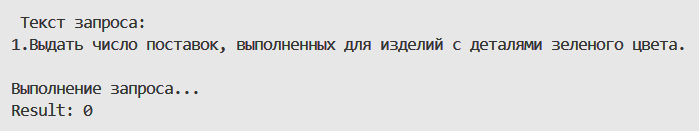


6.Таблица p не существует

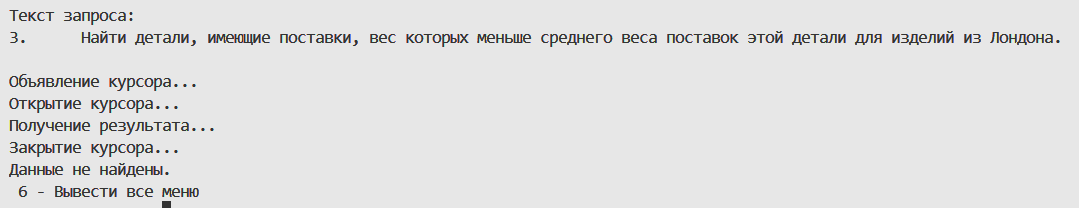


7.Таблица p пустая

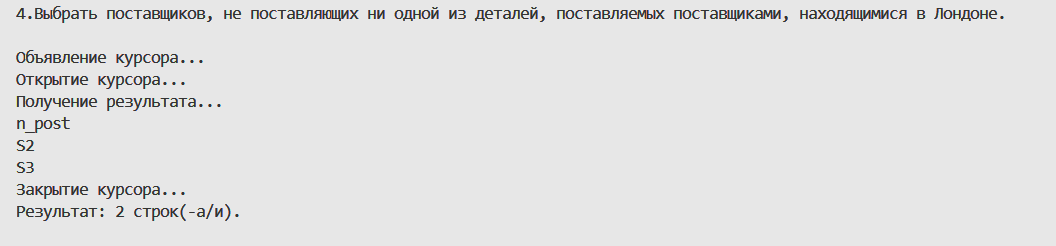
Для 1 запроса



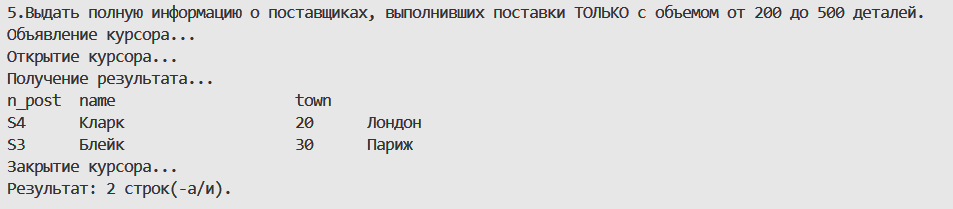
Для 3 запроса



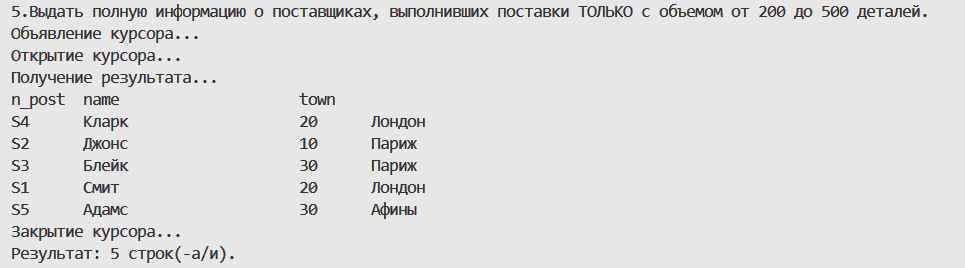
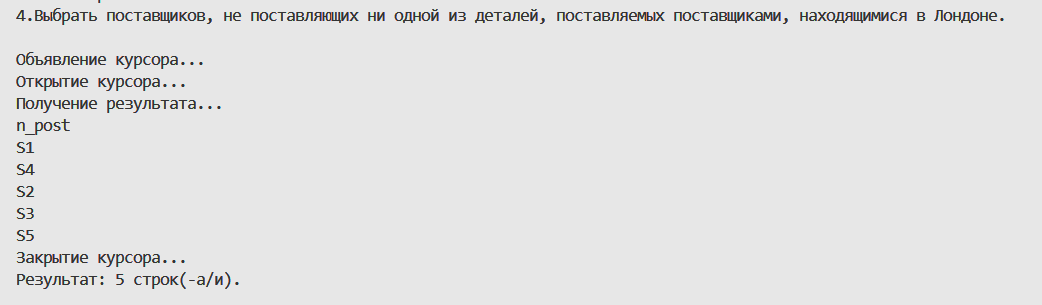
Для 4 запроса



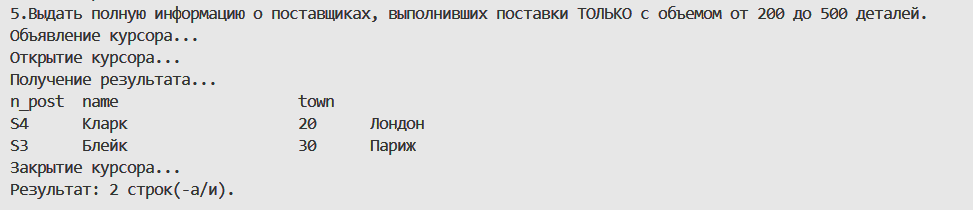
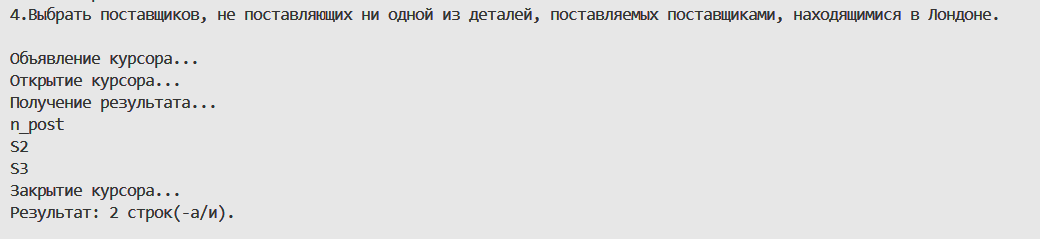
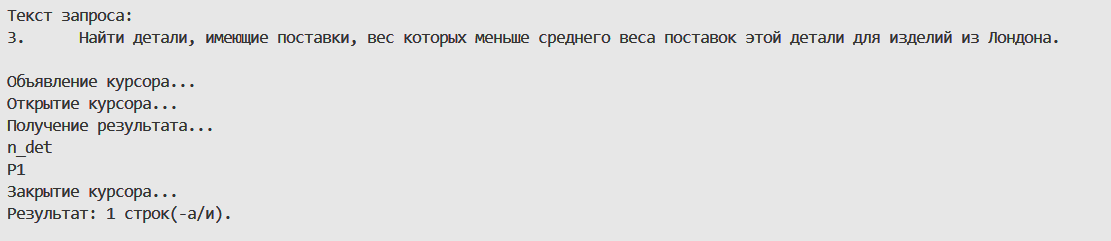
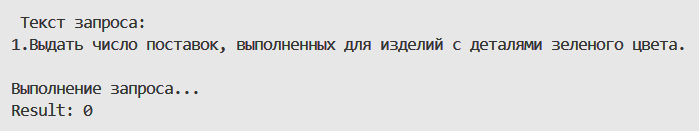
Для 5 запроса



8.Таблица spj пустая

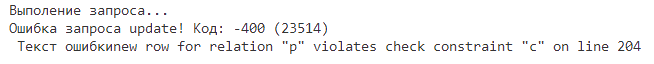


9.В таблице p 1 запись



10. Работа с ограничением

alter table p add constraint c check(n\_det!='P1' or name!='Блюм')



11.Не открылся курсор

