**Лабораторная работа № 5. [Использование подзапросов](#Лаб7)**

1. На основе таблиц **FACULTY**, **PULPIT** и **PROFESSION** сформировать список наименований кафедр (столбец **PULPIT\_NAME**), которые находятся на факультете (таблица **FACULTY**), обеспечивающем подготовку по специальности, в наименовании (столбец **PROFESSION\_ NAME**) которого содержится слово ***технология*** или ***технологии***. Примечание: использовать в секции WHERE предикат IN c некоррелированным подзапросом к таблице **PROFESSION**.

use UNIVER;

SELECT PULPIT.PULPIT\_NAME

FROM FACULTY, PULPIT

WHERE FACULTY.FACULTY = PULPIT.FACULTY AND

FACULTY.FACULTY IN (

SELECT PROFESSION.FACULTY

FROM PROFESSION

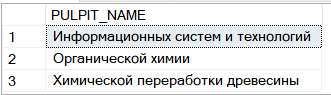
WHERE (

PROFESSION.PROFESSION\_NAME LIKE '%технология%' OR

PROFESSION.PROFESSION\_NAME LIKE '%технологии%'

)

);



1. Переписать запрос пункта 1 таким образом, чтобы тот же подзапрос был записан в конструкции INNER JOIN секции FROM внешнего запроса. При этом результат выполнения запроса должен быть аналогичным результату исходного запроса.

SELECT PULPIT.PULPIT\_NAME

FROM FACULTY JOIN PULPIT

ON FACULTY.FACULTY = PULPIT.FACULTY AND

FACULTY.FACULTY IN (

SELECT PROFESSION.FACULTY

FROM PROFESSION

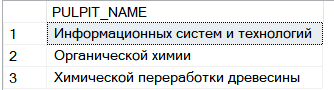
WHERE (

PROFESSION.PROFESSION\_NAME LIKE '%технология%' OR

PROFESSION.PROFESSION\_NAME LIKE '%технологии%'

)

);



1. Переписать запрос, реализующий 1 пункт без использования подзапроса. Примечание: использовать соединение INNER JOIN трех таблиц.

SELECT DISTINCT PULPIT.PULPIT\_NAME

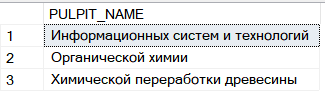
FROM FACULTY

JOIN PULPIT ON FACULTY.FACULTY = PULPIT.FACULTY

JOIN PROFESSION ON PROFESSION.FACULTY = FACULTY.FACULTY AND

(PROFESSION.PROFESSION\_NAME LIKE '%технология%' OR

PROFESSION.PROFESSION\_NAME LIKE '%технологии%');



1. На основе таблицы **AUDITORIUM** сформировать список аудиторий самых больших вместимостей (столбец **AUDITORIUM\_CAPACITY**) для каждого типа аудитории (**AUDITORIUM\_TYPE**). При этом результат следует отсортировать в порядке убывания вместимости. Примечание: использовать коррелируемый подзапрос c секциями TOP и ORDER BY.

SELECT a.AUDITORIUM,

a.AUDITORIUM\_CAPACITY,

a.AUDITORIUM\_TYPE

FROM AUDITORIUM AS a

WHERE a.AUDITORIUM = (

SELECT TOP(1) b.AUDITORIUM

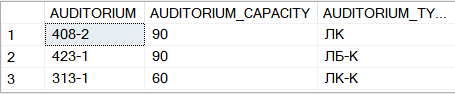
FROM AUDITORIUM AS b

WHERE b.AUDITORIUM\_TYPE = a.AUDITORIUM\_TYPE

ORDER BY b.AUDITORIUM\_CAPACITY DESC

)

ORDER BY a.AUDITORIUM\_CAPACITY DESC;



1. На основе таблиц **FACULTY** и **PULPIT** сформировать список наименований факультетов (столбец **FACULTY\_NAME**) на котором нет ни одной кафедры (таблица **PULPIT**). Примечание: использовать предикат EXISTS и коррелированный подзапрос.

SELECT FACULTY.FACULTY\_NAME

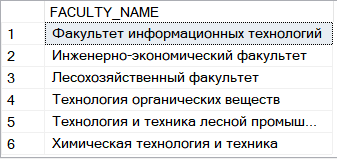
FROM FACULTY

WHERE EXISTS (

SELECT \* FROM PULPIT

WHERE PULPIT.FACULTY = FACULTY.FACULTY

);



1. На основе таблицы **PROGRESS** сформировать строку, содержащую средние значения оценок (столбец **NOTE**) по дисциплинам, имеющим следующие коды: **ОАиП**, **БД** и **СУБД**. Примечание: использовать три некоррелированных подзапроса в списке SELECT; в подзапросах применить агрегатные функции AVG.

SELECT TOP(1)

ISNULL((

SELECT AVG(PROGRESS.NOTE) FROM PROGRESS

WHERE PROGRESS.SUBJECT\_T = 'ОАиП'

), 0) [ОАиП],

ISNULL((

SELECT AVG(PROGRESS.NOTE) FROM PROGRESS

WHERE PROGRESS.SUBJECT\_T = 'БД'

), 0) [БД],

ISNULL((

SELECT AVG(PROGRESS.NOTE) FROM PROGRESS

WHERE PROGRESS.SUBJECT\_T = 'СУБД'

), 0) [СУБД]

FROM PROGRESS;



1. Разработать SELECT-запрос, демонстрирующий принцип применения ALL совместно с подзапросом.

SELECT AUDITORIUM.AUDITORIUM, AUDITORIUM.AUDITORIUM\_CAPACITY

FROM AUDITORIUM

WHERE AUDITORIUM.AUDITORIUM\_CAPACITY >= ALL (

SELECT AUDITORIUM.AUDITORIUM\_CAPACITY

FROM AUDITORIUM

WHERE AUDITORIUM.AUDITORIUM\_TYPE = 'ЛБ-К'

) AND AUDITORIUM.AUDITORIUM\_TYPE = 'ЛБ-К';



1. Разработать SELECT-запрос, демонстрирующий принцип применения ANY совместно с подзапросом.

SELECT AUDITORIUM.AUDITORIUM, AUDITORIUM.AUDITORIUM\_CAPACITY

FROM AUDITORIUM

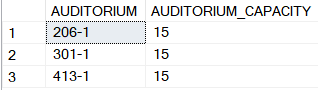
WHERE AUDITORIUM.AUDITORIUM\_CAPACITY < ANY (

SELECT AUDITORIUM.AUDITORIUM\_CAPACITY

FROM AUDITORIUM

WHERE AUDITORIUM.AUDITORIUM\_TYPE = 'ЛБ-К'

) AND AUDITORIUM.AUDITORIUM\_TYPE = 'ЛБ-К';



1. Разработать и выполнить аналогичные запросы для базы данных **X\_MyBASE**.

use Dobriyan\_MyBase;

-- №1

SELECT Groups.Number, Specialities.Speciality

FROM Groups, Specialities

WHERE Specialities.id = Groups.Speciality AND

Specialities.Speciality IN (

SELECT Specialities.Speciality

FROM Specialities

WHERE (

Specialities.Speciality LIKE '%ПОИТ%' OR

Specialities.Speciality LIKE '%ДЭиВИ%'

)

);

-- №2

SELECT Groups.Number, Specialities.Speciality

FROM Groups JOIN Specialities

ON Specialities.id = Groups.Speciality AND

Specialities.Speciality IN (

SELECT Specialities.Speciality

FROM Specialities

WHERE (

Specialities.Speciality LIKE '%ПОИТ%' OR

Specialities.Speciality LIKE '%ДЭиВИ%'

)

);

-- №3

SELECT DISTINCT Groups.Number, Specialities.Speciality

FROM Groups

JOIN Specialities ON Specialities.id = Groups.Speciality

JOIN Teachers ON Teachers.Active\_Group = Groups.Number AND

Specialities.Speciality LIKE '%ПОИТ%' OR

Specialities.Speciality LIKE '%ДЭиВИ%';

-- №4

SELECT a.Number,

a.Students\_Count,

Specialities.Speciality

FROM Groups AS a, Specialities

WHERE a.Number = (

SELECT TOP(1) b.Number

FROM Groups AS b

WHERE b.Speciality = a.Speciality

ORDER BY b.Students\_Count DESC

)

AND a.Speciality = Specialities.id

ORDER BY a.Students\_Count DESC;

-- №5

SELECT Specialities.Speciality, Groups.Number

FROM Groups, Specialities

WHERE EXISTS (

SELECT \* FROM Specialities

WHERE Specialities.id = Groups.Speciality

)

AND Groups.Speciality = Specialities.id;

-- №6

SELECT TOP(1)

ISNULL((

SELECT AVG(Groups.Students\_Count) FROM Groups

WHERE Groups.Speciality = '1'

), 0) [ПОИТ],

ISNULL((

SELECT AVG(Groups.Students\_Count) FROM Groups

WHERE Groups.Speciality = '2'

), 0) [ДЭиВИ],

ISNULL((

SELECT AVG(Groups.Students\_Count) FROM Groups

WHERE Groups.Speciality = '3'

), 0) [ИСИТ],

ISNULL((

SELECT AVG(Groups.Students\_Count) FROM Groups

WHERE Groups.Speciality = '4'

), 0) [ПОИБМС]

FROM Groups;

-- №7

SELECT Specialities.Speciality, Groups.Students\_Count

FROM Groups, Specialities

WHERE Groups.Students\_Count >= ALL (

SELECT Groups.Students\_Count

FROM Groups

WHERE Groups.Speciality = 2

)

AND Groups.Speciality = 2

AND Groups.Speciality = Specialities.id;

-- №8

SELECT Specialities.Speciality, Groups.Students\_Count

FROM Groups, Specialities

WHERE Groups.Students\_Count < ANY (

SELECT Groups.Students\_Count

FROM Groups

WHERE Groups.Speciality = 2

)

AND Groups.Speciality = 2

AND Groups.Speciality = Specialities.id;