**Лабораторная работа № 4. Многотабличные SELECT-запросы**

1. Ознакомиться с [приложением](#Прил1). Изучить сценарии на языке T-SQL, содержащие операторы для создания и заполнения таблиц базы данных **UNIVER**. Выполнить сценарии. Убедиться, что они работают корректно.

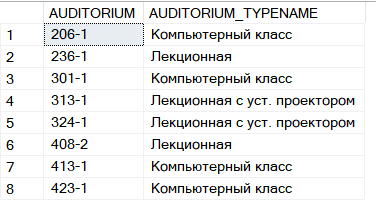
На основе таблиц **AUDITORIUM\_ TYPE** и **AUDITORIUM** сформировать перечень кодов аудиторий (столбец **AUDITORUM.AUDITORIUM**) и соответствующих им наименований типов аудиторий (столбец **AUDITORIUM\_ TYPE.AUDITORIUM\_ TYPENAME**). Примечание: использовать соединение таблиц INNER JOIN.

use UNIVER;

SELECT AUDITORIUM.AUDITORIUM, AUDITORIUM\_TYPE.AUDITORIUM\_TYPENAME

from AUDITORIUM INNER JOIN AUDITORIUM\_TYPE

on AUDITORIUM.AUDITORIUM\_TYPE = AUDITORIUM\_TYPE.AUDITORIUM\_TYPE;



1. На основе таблиц **AUDITORIUM\_TYPE** и **AUDITORIUM** сформировать перечень кодов аудиторий (столбец **AUDITORIUM.AUDITORIUM**) и соответствующих им наименований типов аудиторий (столбец **AUDITORIUM\_ TYPE.AUDITO-RIUM\_TYPENAME**). При этом следует выбрать только те аудитории, в наименовании которых присутствует подстрока **компьютер**. Примечание: использовать соединение таблиц INNER JOIN и предикат LIKE

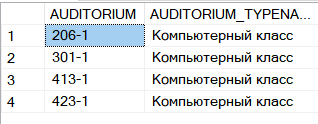
use UNIVER;

SELECT AUDITORIUM.AUDITORIUM, AUDITORIUM\_TYPE.AUDITORIUM\_TYPENAME

from AUDITORIUM INNER JOIN AUDITORIUM\_TYPE

on AUDITORIUM.AUDITORIUM\_TYPE = AUDITORIUM\_TYPE.AUDITORIUM\_TYPE

AND AUDITORIUM\_TYPE.AUDITORIUM\_TYPENAME LIKE '%компьютер%';



1. Написать два SELECT-запроса, формирующих результирующие наборы аналогичные запросам из заданий 1 и 2, но без применения INNER JOIN.

use UNIVER;

SELECT AUDITORIUM.AUDITORIUM, AUDITORIUM\_TYPE.AUDITORIUM\_TYPENAME

from AUDITORIUM, AUDITORIUM\_TYPE

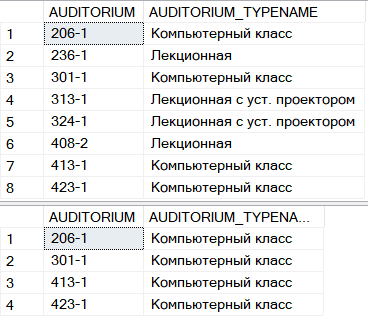
WHERE AUDITORIUM.AUDITORIUM\_TYPE = AUDITORIUM\_TYPE.AUDITORIUM\_TYPE;

SELECT AUDITORIUM.AUDITORIUM, AUDITORIUM\_TYPE.AUDITORIUM\_TYPENAME

from AUDITORIUM, AUDITORIUM\_TYPE

WHERE AUDITORIUM.AUDITORIUM\_TYPE = AUDITORIUM\_TYPE.AUDITORIUM\_TYPE

AND AUDITORIUM\_TYPE.AUDITORIUM\_TYPENAME LIKE '%компьютер%';



1. На основе таблиц **PRORGESS**, **STUDENT**, **GROUPS**, **SUBJECT**, **PULPIT** и **FACULTY** сформировать перечень студентов, получивших экзаменационные оценки (столбец **PROGRESS.NOTE**) от 6 до 8.

Результирующий набор должен содержать столбцы: **Факультет**, **Кафедра**, **Специальность**, **Дисциплина**, **Имя Студента**, **Оценка**. В столбце **Оценка** должны быть записаны экзаменационные оценки прописью: **шесть**, **семь**, **восемь**.

Результирующий набор отсортировать в порядке возрастания по столбцам **FACULTY.FACULTY**, **PULPIT.PULPIT**, **PROFESSION.PROFESSION**, **STUDENT. STUDENT\_NAME** и в порядке убывания по столбцу **PROGRESS.NOTE**. Примечание: использовать соединение INNER JOIN, предикат BETWEEN и выражение CASE.

use UNIVER;

SELECT FACULTY.FACULTY, PULPIT.PULPIT, PROFESSION.PROFESSION\_NAME, SUBJECT\_T.SUBJECT\_NAME, STUDENT.NAME,

CASE

when(PROGRESS.NOTE = 6) then 'шесть'

when(PROGRESS.NOTE = 7) then 'семь'

when(PROGRESS.NOTE = 8) then 'восемь'

END [NOTE]

FROM STUDENT inner Join PROGRESS ON STUDENT.IDSTUDENT = PROGRESS.IDSTUDENT

inner join SUBJECT\_T ON PROGRESS.SUBJECT\_T = SUBJECT\_T.SUBJECT\_T and (PROGRESS.NOTE between 6 and 8)

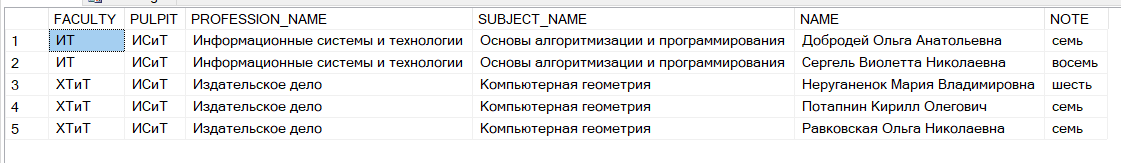
inner join GROUPS ON STUDENT.IDGROUP = GROUPS.IDGROUP

inner join PULPIT ON SUBJECT\_T.PULPIT = PULPIT.PULPIT

inner join PROFESSION ON PROFESSION.PROFESSION = GROUPS.PROFESSION

inner join FACULTY ON GROUPS.FACULTY = FACULTY.FACULTY

ORDER BY FACULTY.FACULTY, PULPIT.PULPIT, PROFESSION.PROFESSION, STUDENT.NAME ASC, PROGRESS.NOTE DESC;



1. Переписать запрос, реализующий задание 4 таким образом, чтобы в результирующем наборе сортировка по экзаменационным оценкам была следующей: сначала выводились строки с оценкой **7**, затем строки с оценкой **8** и далее строки с оценкой **6**. Примечание: использовать выражение CASE в секции ORDER BY.

use UNIVER;

SELECT FACULTY.FACULTY, PULPIT.PULPIT, PROFESSION.PROFESSION\_NAME, SUBJECT\_T.SUBJECT\_NAME, STUDENT.NAME,

CASE

when(PROGRESS.NOTE = 6) then 'шесть'

when(PROGRESS.NOTE = 7) then 'семь'

when(PROGRESS.NOTE = 8) then 'восемь'

END [NOTE]

FROM STUDENT inner Join PROGRESS ON STUDENT.IDSTUDENT = PROGRESS.IDSTUDENT

inner join SUBJECT\_T ON PROGRESS.SUBJECT\_T = SUBJECT\_T.SUBJECT\_T and (PROGRESS.NOTE between 6 and 8)

inner join GROUPS ON STUDENT.IDGROUP = GROUPS.IDGROUP

inner join PULPIT ON SUBJECT\_T.PULPIT = PULPIT.PULPIT

inner join PROFESSION ON PROFESSION.PROFESSION = GROUPS.PROFESSION

inner join FACULTY ON GROUPS.FACULTY = FACULTY.FACULTY

ORDER BY FACULTY.FACULTY, PULPIT.PULPIT, PROFESSION.PROFESSION, STUDENT.NAME ASC,

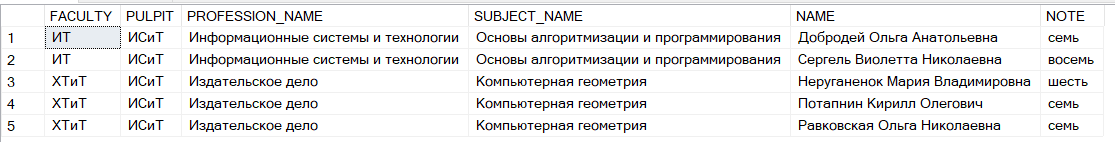
(CASE

when(PROGRESS.NOTE = 7) then 1

when(PROGRESS.NOTE = 8) then 2

when(PROGRESS.NOTE = 6) then 3

END);



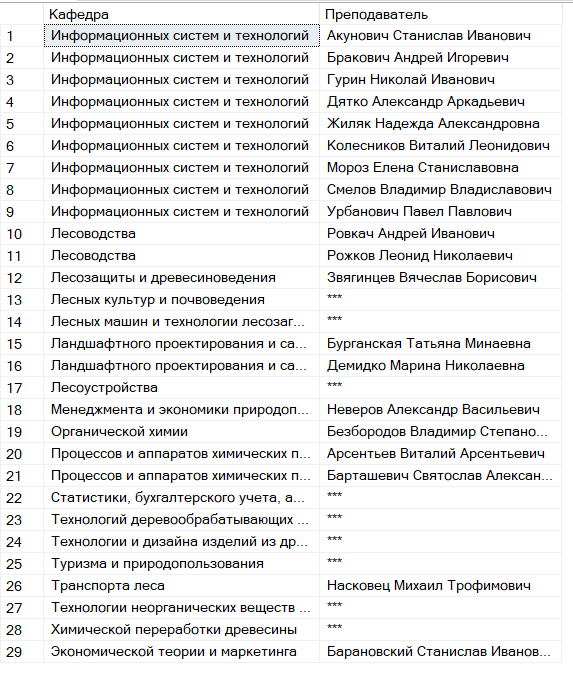
1. На основе таблиц **PULPIT** и **TEACHER** получить полный перечень кафедр (столбец **PULPIT.PULPIT\_ NAME)** и преподавателей (столбец **TEACHER.TEACHER\_NAME**) на этих кафедрах. Результирующий набор должен содержать два столбца: **Кафедра** и **Преподаватель**. Если на кафедре нет преподавателей, то в столбце **Преподаватель** должна быть выведена строка **\*\*\*.** Примечание: использовать соединение таблиц LEFT OUTER JOIN и функцию **isnull**.

use UNIVER;

SELECT PULPIT.PULPIT\_NAME [Кафедра], ISNULL(TEACHER.TEACHER\_NAME, '\*\*\*') [Преподаватель]

FROM PULPIT LEFT OUTER JOIN TEACHER

ON TEACHER.PULPIT = PULPIT.PULPIT;



1. В запросе, реализующем пункт 6, поменять порядок таблиц в выражении LEFT OUTER JOIN. Объяснить: почему в столбце **Кафедра** не может быть значения NULL. Переписать запрос таким образом, чтобы получился аналогичный результат, но применялось соединение таблиц RIGHT OUTER JOIN.

use UNIVER;

SELECT PULPIT.PULPIT\_NAME [Кафедра], ISNULL(TEACHER.TEACHER\_NAME, '\*\*\*') [Преподаватель]

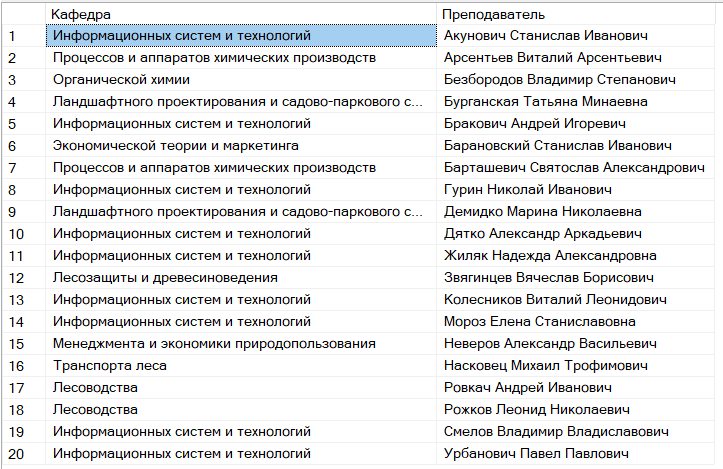
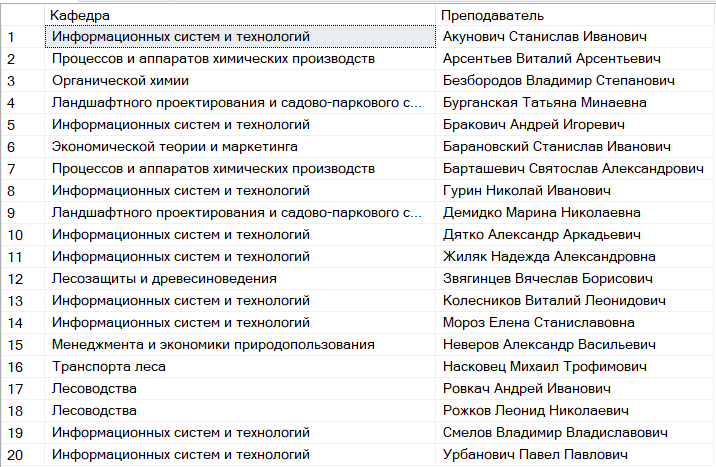
FROM TEACHER LEFT OUTER JOIN PULPIT

ON TEACHER.PULPIT = PULPIT.PULPIT;

SELECT PULPIT.PULPIT\_NAME [Кафедра], ISNULL(TEACHER.TEACHER\_NAME, '\*\*\*') [Преподаватель]

FROM PULPIT RIGHT OUTER JOIN TEACHER

ON TEACHER.PULPIT = PULPIT.PULPIT;



1. Показать на примере, что соединение FULL OUTER JOIN двух таблиц:

− является коммутативной операцией;

− является объединением LEFT OUTER JOIN и RIGHT OUTER JOIN соединений этих таблиц;

− включает соединение INNER JOIN этих таблиц. Примечание: создать две таблицы, заполнить их данными. Разработать SELECT-запросы, реализующие задания.

Создать три новых запроса:

− запрос, результат которого содержит данные левой (в операции FULL OUTER JOIN) таблицы и не содержит данные правой;

− запрос, результат которого содержит данные правой таблицы и не содержащие данные левой;

− запрос, результат которого содержит данные правой таблицы и левой таблиц;

Примечание: использовать в запросах выражение IS NULL и IS NOT NULL.

use UNIVER;

SELECT ISNULL(TEACHER.TEACHER\_NAME, '\*\*\*') [TEACHER], SUBJECT\_T.SUBJECT\_NAME [SUBJECT]

FROM TEACHER INNER JOIN PULPIT

ON TEACHER.PULPIT = PULPIT.PULPIT

FULL OUTER JOIN SUBJECT\_T

ON PULPIT.PULPIT = SUBJECT\_T.PULPIT

WHERE TEACHER.TEACHER\_NAME IS NULL;

SELECT ISNULL(SUBJECT\_T.SUBJECT\_NAME, '\*\*\*') [SUBJECT], TEACHER.TEACHER\_NAME [TEACHER]

FROM PULPIT INNER JOIN TEACHER

ON TEACHER.PULPIT = PULPIT.PULPIT

FULL OUTER JOIN SUBJECT\_T

ON PULPIT.PULPIT = SUBJECT\_T.PULPIT

WHERE SUBJECT\_T.SUBJECT\_NAME IS NULL;

SELECT SUBJECT\_T.SUBJECT\_NAME [SUBJECT], TEACHER.TEACHER\_NAME [TEACHER]

FROM PULPIT INNER JOIN TEACHER

ON TEACHER.PULPIT = PULPIT.PULPIT

FULL OUTER JOIN SUBJECT\_T

ON PULPIT.PULPIT = SUBJECT\_T.PULPIT

WHERE SUBJECT\_T.SUBJECT\_NAME IS NOT NULL AND TEACHER.TEACHER\_NAME IS NOT NULL;

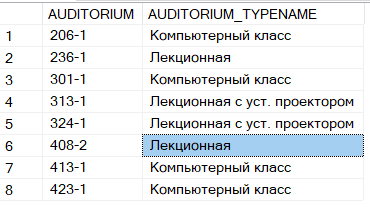
1. Разработать SELECT-запрос на основе CROSS JOIN-соединения таблиц **AUDITORIUM\_TYPE** и **AUDITORIUM**, формирующего результат, аналогичный результату, полученному при выполнении запроса в задании 1.

use UNIVER;

SELECT AUDITORIUM.AUDITORIUM, AUDITORIUM\_TYPE.AUDITORIUM\_TYPENAME

FROM AUDITORIUM CROSS JOIN AUDITORIUM\_TYPE

WHERE AUDITORIUM.AUDITORIUM\_TYPE = AUDITORIUM\_TYPE.AUDITORIUM\_TYPE;



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

use Dobriyan\_MyBase;

-- №1

SELECT Groups.Number [Group], Specialities.Speciality

FROM Specialities INNER JOIN Groups

ON Specialities.id = Groups.Speciality;

-- №2

SELECT Groups.Number [Group], Specialities.Speciality

FROM Specialities INNER JOIN Groups

ON Specialities.id = Groups.Speciality

AND Specialities.Speciality LIKE '%П%';

-- №3

SELECT Groups.Number [Group], Specialities.Speciality

FROM Specialities, Groups

WHERE Specialities.id = Groups.Speciality;

SELECT Groups.Number [Group], Specialities.Speciality

FROM Specialities, Groups

WHERE Specialities.id = Groups.Speciality

AND Specialities.Speciality LIKE '%П%';

-- №4-5

SELECT Courses.Course\_Subject [Subject],

Teachers.Surname [Teacher Surname],

Teachers.Name [Teacher Name],

Teachers.Patronimic [Teacher Patronimic],

Specialities.Speciality [Group Speciality],

CASE

when(Groups.Number = 1) then 'первая'

when(Groups.Number = 2) then 'вторая'

when(Groups.Number = 3) then 'третья'

when(Groups.Number = 4) then 'четвертая'

END [Group Number]

FROM Teachers

INNER JOIN Courses on Teachers.Active\_Course = Courses.Course\_Subject

INNER JOIN Groups on Teachers.Active\_Group = Groups.Number AND Groups.Number BETWEEN 1 and 4

INNER JOIN Specialities on Groups.Speciality = Specialities.id

ORDER BY

(CASE

when(Groups.Number = 1) then 1

when(Groups.Number = 2) then 4

when(Groups.Number = 3) then 2

when(Groups.Number = 4) then 3

END);

-- №6

SELECT Courses.Course\_Subject [Group], ISNULL(Teachers.Surname, '---') [Teacher Surname]

FROM Courses

LEFT JOIN Teachers on Courses.Course\_Subject = Teachers.Active\_Course;

-- №7

SELECT Courses.Course\_Subject [Group], ISNULL(Teachers.Surname, '---') [Teacher Surname]

FROM Teachers

LEFT JOIN Courses on Courses.Course\_Subject = Teachers.Active\_Course;

SELECT Courses.Course\_Subject [Group], ISNULL(Teachers.Surname, '---') [Teacher Surname]

FROM Courses

RIGHT JOIN Teachers on Courses.Course\_Subject = Teachers.Active\_Course;

-- №9

SELECT Groups.Number [Group], Specialities.Speciality

FROM Specialities CROSS JOIN Groups

WHERE Specialities.id = Groups.Speciality;