

**Московский Авиационный Институт  
(национальный исследовательский университет)  
Факультет информационных технологий и прикладной математики  
Кафедра вычислительной математики и программирования**

**Лабораторная работа  
по курсу Базы Данных**

**Выполнил: Дюсекеев А.Е.**

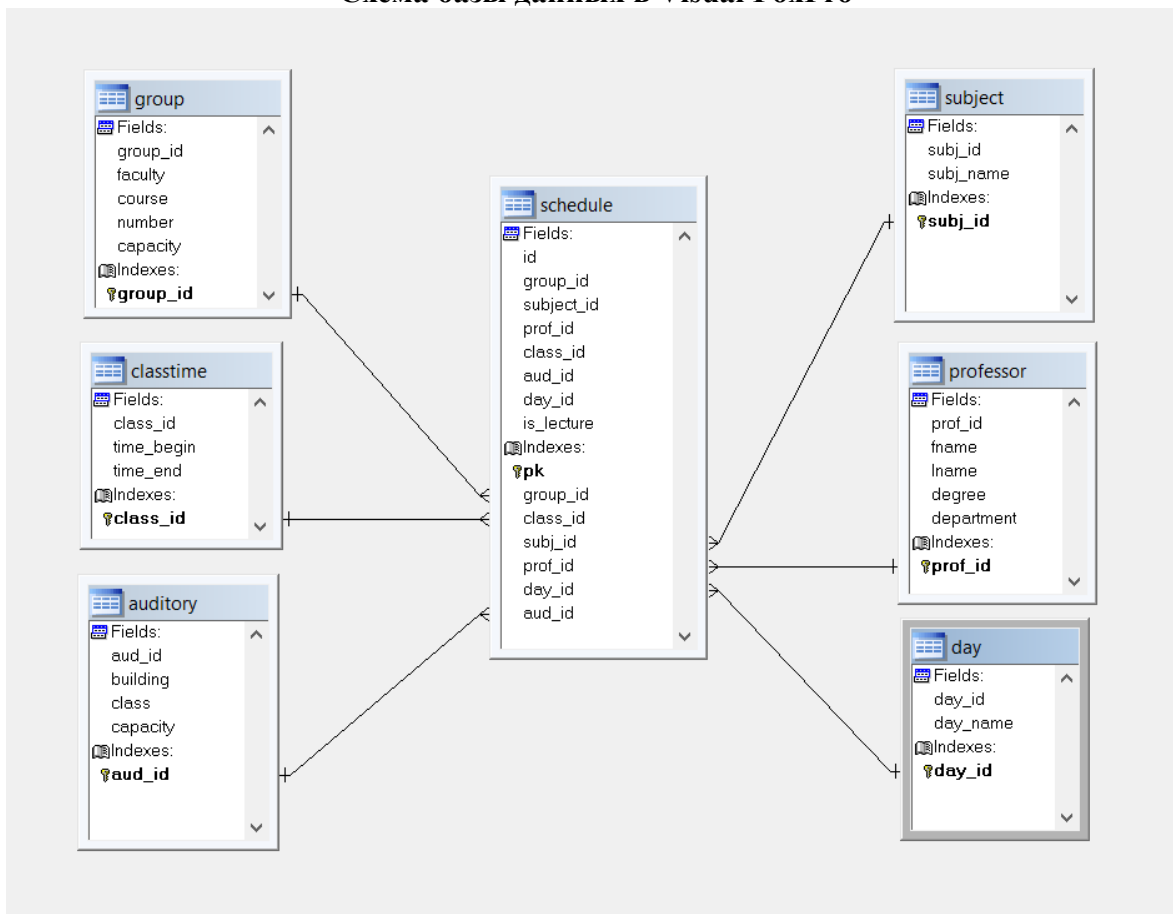
**Группа: М8О-304Б**

**Руководитель: Кузнецова Е.В.**

**Москва, 2019**

Задание: разработать базу данных для представления расписания занятий.

### Схема базы данных в Visual FoxPro



### Таблица Schedule

Данная таблица является связующей для всех остальных. Все поля имеют целочисленный тип данных integer длиной 4 байта. Поле id – первичный ключ, а все остальные – внешние ключи к другим таблицам

Таблица

Id	Group_id	Subject_id	Prof_id	Class_id	Aud_id	Day_id	Is_lecture
1	1	1	1	1	3	1	T
2	2	1	1	1	3	1	T
3	3	1	1	1	3	1	T
4	4	1	1	1	3	1	T
						1	F
						1	F
						1	F
						2	T
						2	T
						2	T
						3	F
						3	F
						3	F
						4	F
						5	F
						6	F
						6	F

Group_id	Faculty	Course	Number	Capacity
1	1	1	1	15
2	1	1	2	13
3	1	1	3	14
4	1	1	4	18
5	1	2	1	13
6	1	2	2	16
7	1	2	3	18
8	1	3	1	12
9	1	3	2	13
10	1	4	1	20
11	2	1	1	24
12	2	1	2	26
13	2	1	3	10
14	2	2	1	20
15	2	2	2	15
16	2	3	1	21
17	2	4	1	20
18	3	1	1	16
19	3	1	2	18
20	3	1	3	17

Group

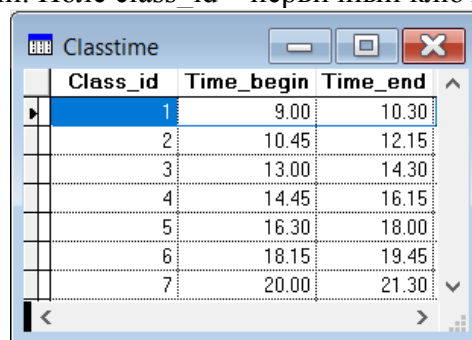
Описывает все таблички имеют данный integer

учебные группы. Все поля целочисленный тип длиной 4 байта. Поле

group\_id является первичным ключом.

### Таблица Classtime

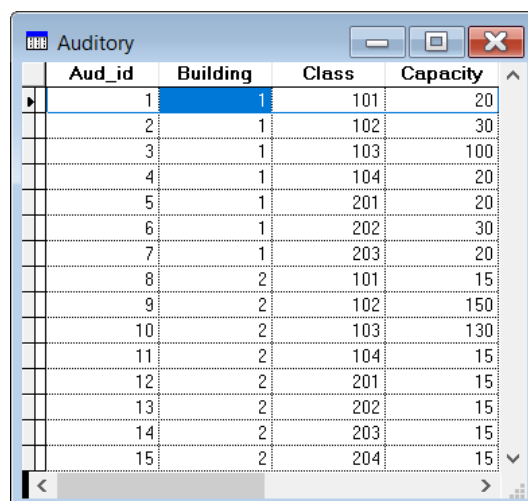
Задаёт время учебных занятий. Поле class\_id – первичный ключ типа integer длиной 4 байта.



Class_id	Time_begin	Time_end
1	9.00	10.30
2	10.45	12.15
3	13.00	14.30
4	14.45	16.15
5	16.30	18.00
6	18.15	19.45
7	20.00	21.30

### Таблица Auditory

Описывает все аудитории учебного заведения. Все поля имеют тип integer. Поле Aud\_id – первичный ключ.

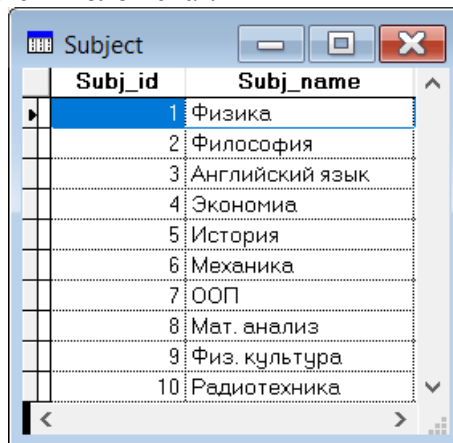


Aud_id	Building	Class	Capacity
1	1	101	20
2	1	102	30
3	1	103	100
4	1	104	20
5	1	201	20
6	1	202	30
7	1	203	20
8	2	101	15
9	2	102	150
10	2	103	130
11	2	104	15
12	2	201	15
13	2	202	15
14	2	203	15
15	2	204	15

### Таблица Subject

Задаёт учебные дисциплины. Поле subj\_id – первичный ключ типа integer длиной 4 байта.

Поле subj\_name – строка из 15 символов char.



Subj_id	Subj_name
1	Физика
2	Философия
3	Английский язык
4	Экономика
5	История
6	Механика
7	ООП
8	Мат. анализ
9	Физ. культура
10	Радиотехника

### Таблица Professor

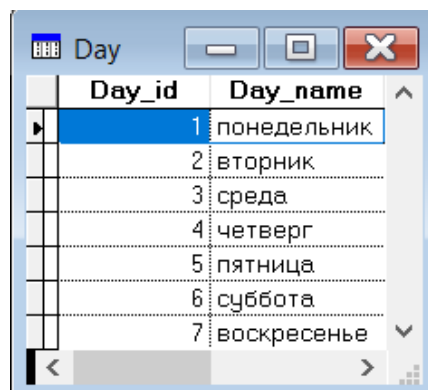
В этой таблице перечислены все преподаватели. Поля prof\_id и department – данные типа integer длиной 4 байта, причем prof\_id – первичный ключ. Остальные поля – строки из 10 символов.



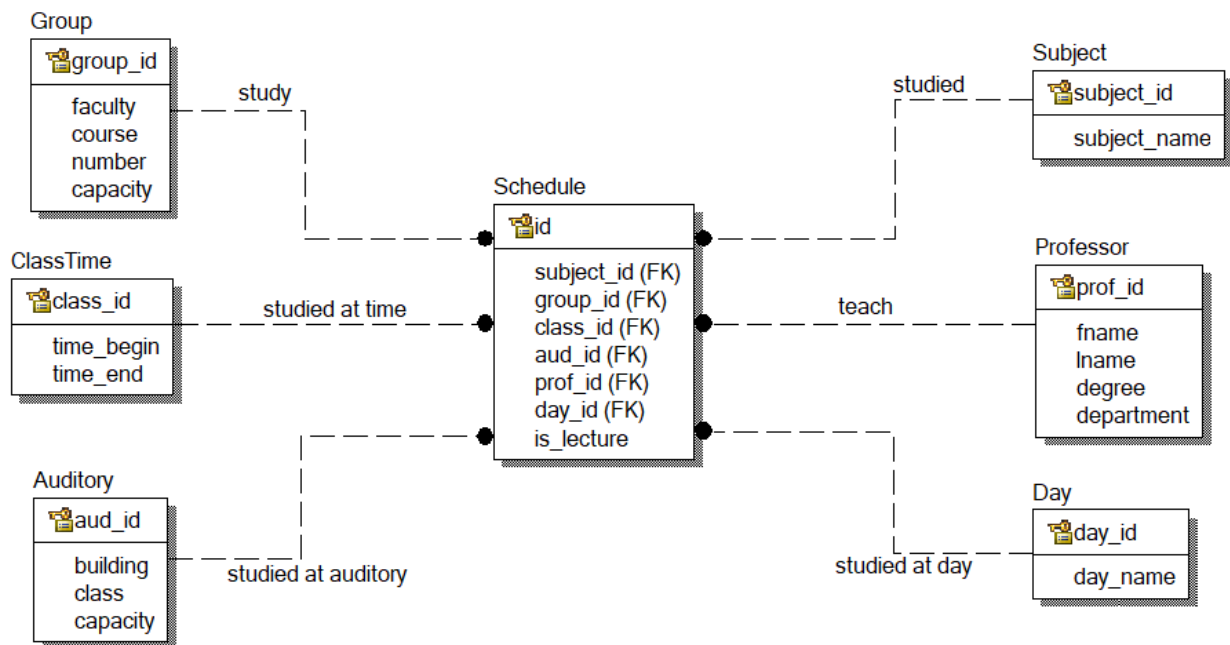
Prof_id	Fname	Lname	Degree	Department
1	Виталий	Раскалов	Профессор	1
2	Вадим	Махоров	Аспирант	1
3	Геннадий	Степанов	Профессор	1
4	Николай	Дорожников	Аспирант	1
5	Евгения	Городовая	Профессор	1
6	Юлия	Петрова	Аспирант	2
7	Валентин	Петренко	Профессор	2
8	Иван	Игнатов	Аспирант	2
9	Мария	Романова	Профессор	3
10	Николай	Рябинин	Аспирант	3

### Таблица day

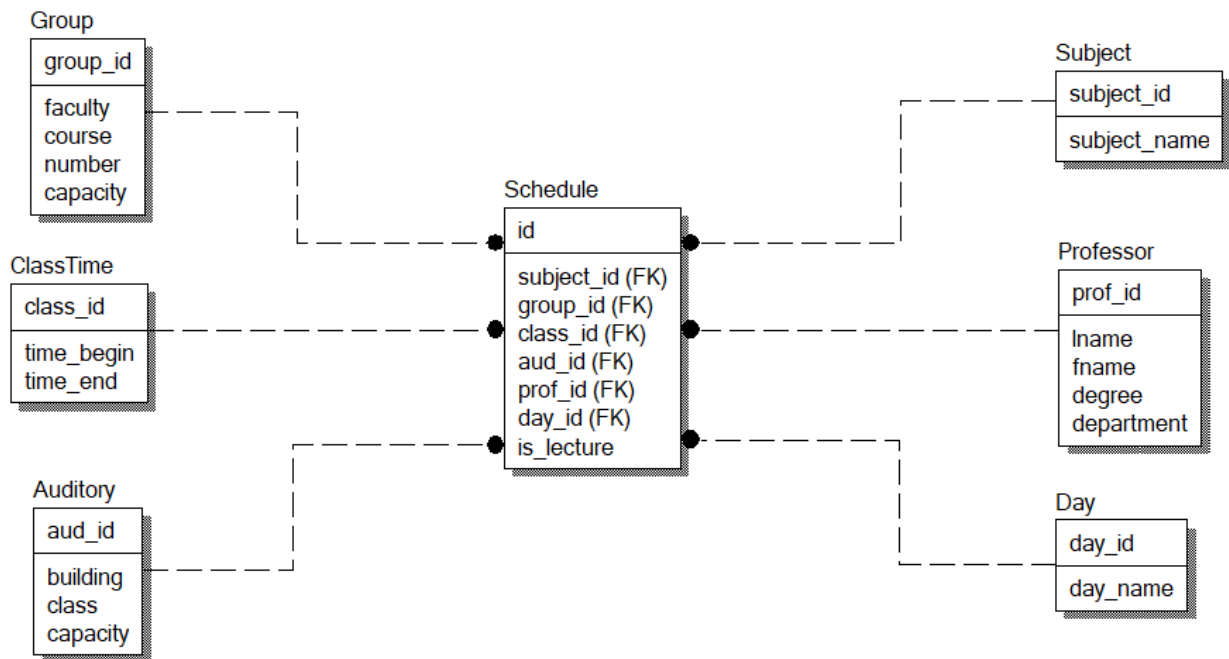
Задаёт учебные дни. Поле day\_id – первичный ключ типа integer. Поле day\_name – строка из 15 символов.



Day_id	Day_name
1	понедельник
2	вторник
3	среда
4	четверг
5	пятница
6	суббота
7	воскресенье



### Физическое представление в Erwin



### Сгенерированный код на DDLSQL

```
CREATE TABLE Auditory
(
    aud_id          Numeric NOT NULL,
    building        Numeric NULL,
    class           Numeric NULL,
    capacity        Numeric NULL
);

CREATE UNIQUE INDEX XPKAuditory ON Auditory
(
    aud_id          ASC
)
```

);

CREATE TABLE ClassTime

(  
    class\_id        Numeric NOT NULL,  
    time\_begin      Float NULL,  
    time\_end        Float NULL

);

CREATE UNIQUE INDEX XPKClassTime ON ClassTime

(  
    class\_id        ASC

);

CREATE TABLE Day

(  
    day\_id          Numeric NOT NULL,  
    day\_name        Character(15) NULL

);

CREATE UNIQUE INDEX XPKDay ON Day

(  
    day\_id          ASC

);

CREATE TABLE Group

(  
    group\_id        Numeric NOT NULL,  
    faculty         Numeric NULL,  
    course          Numeric NULL,  
    number          Numeric NULL,  
    capacity         Numeric NULL

);

CREATE UNIQUE INDEX XPKGroup ON Group

(  
    group\_id        ASC

);

CREATE TABLE Professor

(  
    prof\_id         Numeric NOT NULL,  
    lname           Character(10) NULL,  
    fname           Character(10) NULL,  
    degree          Character(10) NULL,  
    department      Numeric NULL

);

CREATE UNIQUE INDEX XPKProfessor ON Professor

(  
    prof\_id         ASC

);

```
CREATE TABLE Schedule
(
    group_id      Numeric NOT NULL,
    class_id      Numeric NOT NULL,
    aud_id        Numeric NOT NULL,
    subject_id    Numeric NOT NULL,
    prof_id       Numeric NOT NULL,
    day_id        Numeric NOT NULL,
    is_lecture    Logical NULL,
    id            Numeric NOT NULL
);
```

```
CREATE UNIQUE INDEX XPKSchedule ON Schedule
(
    id            ASC
);
```

```
CREATE INDEX XIF1Schedule ON Schedule
(
    group_id      ASC
);
```

```
CREATE INDEX XIF2Schedule ON Schedule
(
    class_id      ASC
);
```

```
CREATE INDEX XIF3Schedule ON Schedule
(
    aud_id        ASC
);
```

```
CREATE INDEX XIF4Schedule ON Schedule
(
    subject_id    ASC
);
```

```
CREATE INDEX XIF5Schedule ON Schedule
(
    prof_id       ASC
);
```

```
CREATE INDEX XIF6Schedule ON Schedule
(
    day_id        ASC
);
```

```
CREATE TABLE Subject
(
    subject_id    Numeric NOT NULL,
    subject_name  Character(15) NULL
);
```

);

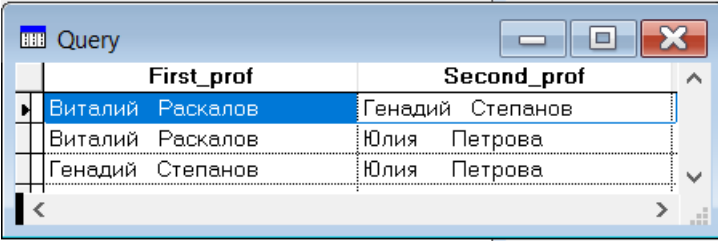
CREATE UNIQUE INDEX XPKSubject ON Subject

(  
    subject\_id        ASC  
);

## Запросы

1. Найти пары преподавателей одной группы, работающие в заданный день недели.

```
SELECT Schedule.prof_id; /* Выбираем в таблице Schedule столбец prof_id */  
  
FROM ; /* из */  
  
    schedule; /* таблицы графика */  
  
WHERE Schedule.day_id = ( 1 ); /* где день проведения занятий */  
  
AND Schedule.group_id = ( 1 ); /* где группы */  
  
INTO CURSOR temp NOFILTER /* таблица переходит во временную переменную */  
  
SELECT pr.prof_id, pr.firstname, pr.lastname; /* Выбираем профессоров */  
  
FROM; /* из */  
    temp; /* Временная переменная */  
INNER JOIN professor AS pr ; /* соединяется с таблицей профессоров */  
  
    ON temp.prof_id = pr.prof_id; /* при этом условии */  
  
INTO CURSOR temp NOFILTER /* таблица переходит во временную переменную */  
  
SELECT DISTINCT first.lastname AS first_prof, second.lastname AS second_prof;  
    /* выборка элементов из этих таблиц */  
FROM; /* из */  
    temp AS first; /* */  
  
INNER JOIN temp AS second ON first.prof_id < second.prof_id  
/* соединение таблиц */
```



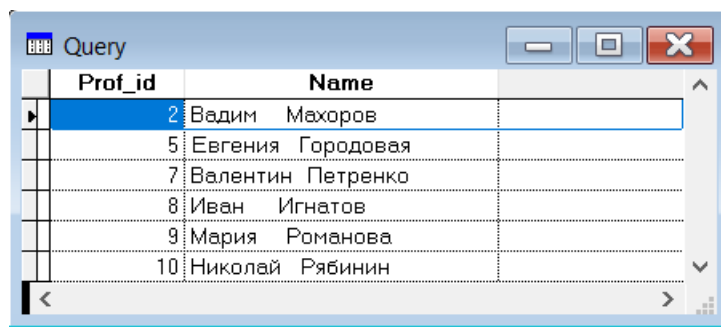
First_prof	Second_prof
Виталий Раскалов	Геннадий Степанов
Виталий Раскалов	Юлия Петрова
Геннадий Степанов	Юлия Петрова

2. Преподаватели, которые не работают в понедельник

```
SELECT Professor.prof_id, firstname+" "+lastname AS name;  
/* выбираем профессоров */  
  
FROM ; /* из */  
  
professor; /* этой таблицы */
```



```
WHERE ( prof_id ) NOT IN (SELECT prof_id FROM schedule WHERE (day_id) = (1))
/* где преподаватели не работают в первый день недели */
```



Prof_id	Name
2	Вадим Махоров
5	Евгения Городовая
7	Валентин Петренко
8	Иван Игнатов
9	Мария Романова
10	Николай Рябинин

3. Кто из преподавателей может заменить Иванова в заданной группе в заданный день недели.

```
SELECT a.prof_id; /* выбираем таблицу */
FROM; /* из */
SELECT DISTINCT prof_id FROM schedule WHERE subject_id = 1) AS a;
/* выбираем разных профессоров по заданному предмету */

INNER JOIN /* внутреннее соединение */

(SELECT DISTINCT prof_id FROM schedule WHERE day_id = 1) AS b;
/* выбираем разных профессоров по заданному дню */
ON a.prof_id = b.prof_id; /* условие объединения */

INTO CURSOR tmp NOFILTER /* таблица переходит во временную переменную */

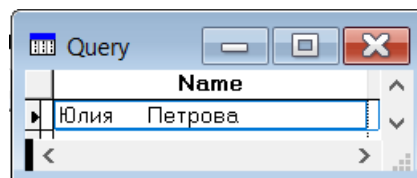
SELECT a.prof_id; /* выбираем таблицу */

FROM; /* из */
tmp AS a;

WHERE NOT exist (SELECT * FROM schedule as sch WHERE sch.prof_id =
a.prof_id AND day_id = 1 AND class_id = 1);

INTO CURSOR tmp2

SELECT pr.fname + ' ' + pr.lname AS name;
FROM;
tmp2 AS res;
INNER JOIN professor AS pr;
ON res.prof_id = pr.prof_id
```



Name
Юлия Петрова

4. Найти дни, в которые больше всего занятий вообще

```
SELECT Schedule.day_id, COUNT(*) AS count; /* выбрать график учебных дней */

FROM ; /* из */

schedule; /* график */
```

```

GROUP BY Schedule.day_id; /* */

INTO CURSOR tmp NOFILTER /* в курсор */

SELECT day_id; /* выбрать */

FROM; /* из */

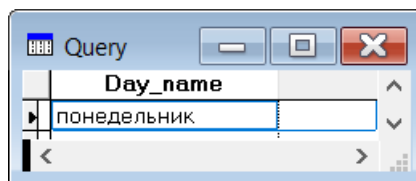
    tmp;

WHERE (count) = (SELECT MAX(count) FROM tmp));

INTO CURSOR temp NOFILTER /* где */

SELECT day_name; /* выбрать day_name */
    FROM; /* из */
        day INNER JOIN temp ON day.day_id = tmp.day_id /* внутреннее
соединение */

```



5. Определить лекционные аудитории (больше 100 человек), свободные в заданный день в заданное время

```

SELECT Auditory.aud_id; /*Выбраем лекционные аудитории*/

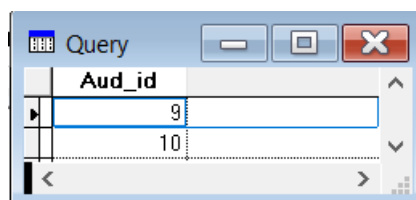
FROM ; /* из */

    auditory; /* таблицы аудитории */

WHERE ( aud_id ) NOT IN (SELECT aud_id FROM schedule WHERE day_id = 1 AND
class_id = 1); /* где свободные аудитории в заданный день в заданное время */

AND ( capacity ) >= ( 100 )/* и вместимость больше 100 */

```



6. Определить аудитории, которые заняты выше среднего

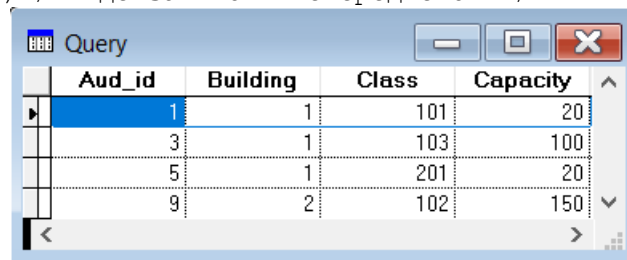
```

SELECT Schedule.aud_id, COUNT(*) AS count; /* выбрать в таблице Schedule столбец
aud_id */
FROM ; /* из */
    schedule; /* график */
GROUP BY Schedule.aud_id; /* объединения результатов выборки по одному или
нескольким столбцам */
INTO CURSOR tmp NOFILTER /* в курсор */

SELECT *; /* выбрать */
    FROM; /* из */
        auditory; /* таблица аудитория */
WHERE auditor_id IN (SELECT aud_id FROM tmp WHERE count > (SELECT

```

```
AVG(count) FROM tmp)) /* где занято выше среднего */
```



	Aud_id	Building	Class	Capacity	
▶	1	1	101	20	^
	3	1	103	100	
	5	1	201	20	
	9	2	102	150	▼