**Московский авиационный институт**

**(Национальный исследовательский университет)**

Институт: «Информационные технологии и прикладная математика»

Кафедра: 806 «Вычислительная математика и программирование»

Дисциплина: «Базы данных»

**Лабораторная работа**

Тема: Успеваемость студентов

Студенты: Ивенкова Л. В.

Королёв И.М.

Группа: М8О-308Б-19

Преподаватель: Киндинова В.В.

Дата: 30.10.21

Оценка:

Москва, 2021

Содержание

[**База данных в FoxPro** 3](#_Toc87014310)

[**Таблицы** 3](#_Toc87014311)

[1. Факультеты (faculty) 3](#_Toc87014312)

[2. Специальности (speciality) 4](#_Toc87014313)

[3. Кафедры (department) 4](#_Toc87014314)

[4. Группы (groups) 4](#_Toc87014315)

[5. Предметы (subjects) 4](#_Toc87014316)

[6. Преподаватели (teachers) 5](#_Toc87014317)

[7. Студенты (students) 5](#_Toc87014318)

[8. Экзамены (exams) 5](#_Toc87014319)

[9. Формы сдачи (passing\_form) 5](#_Toc87014320)

[10. Аудитории (classroom) 6](#_Toc87014321)

[11. Ведомость (grades) 6](#_Toc87014322)

[**Модель базы данных в ErWin** 7](#_Toc87014323)

[**Логическая модель** 7](#_Toc87014324)

[**Физическая модель** 7](#_Toc87014325)

[**Простые запросы** 8](#_Toc87014326)

[1. Определить дату самого последнего экзамена 8](#_Toc87014327)

[2. Список групп, обучающихся на заданной кафедре 8](#_Toc87014328)

[3. Для каждого задолжника посчитать количество его долгов 8](#_Toc87014329)

[**Сложные запросы** 9](#_Toc87014330)

[1. Пары групп одной специальности заданной кафедры 9](#_Toc87014331)

[2. Группы, которые не занимаются у заданного преподавателя 9](#_Toc87014332)

[3. Определить группы, имеющие максимальную среднюю оценку 10](#_Toc87014333)

[4. Определить задолжников по заданному предмету 10](#_Toc87014334)

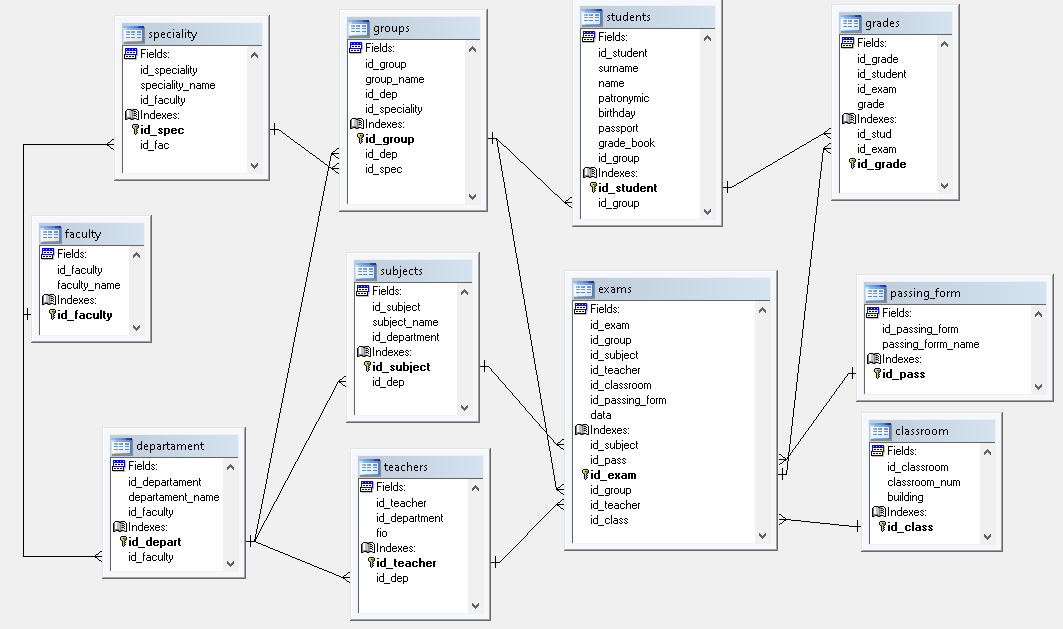
[5. Преподаватели, у которых выше среднего количество задолжников 11](#_Toc87014335)

[6. По данной специальности определить наиболее успешные предметы 11](#_Toc87014336)

[7. Придумать и реализовать запрос на EXIST 12](#_Toc87014337)

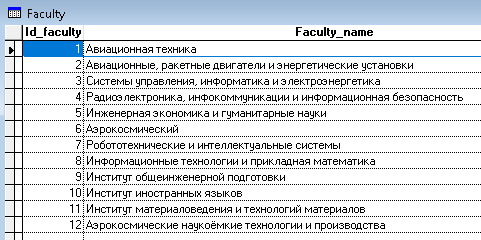
[8. Определить даты экзаменов с наихудшими средними результатами 13](#_Toc87014338)

# **База данных в FoxPro**

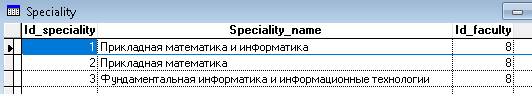


## **Таблицы**

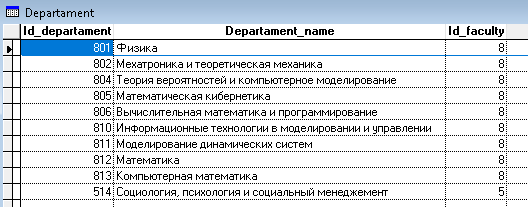
1. Факультеты (faculty)



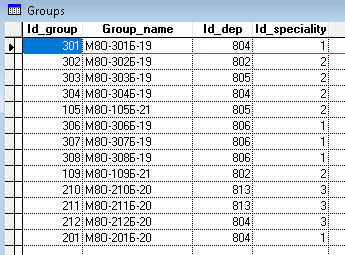
1. Специальности (speciality)



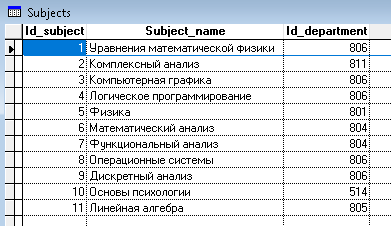
1. Кафедры (department)



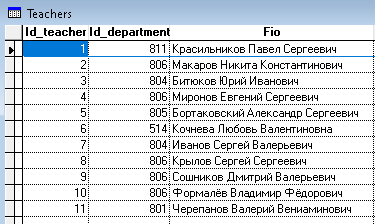
1. Группы (groups)



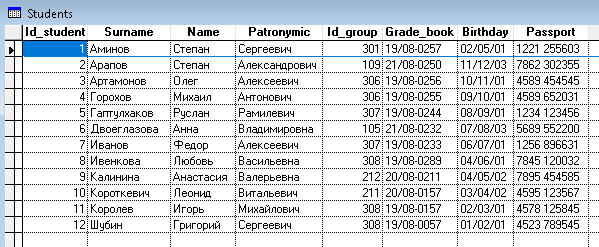
1. Предметы (subjects)



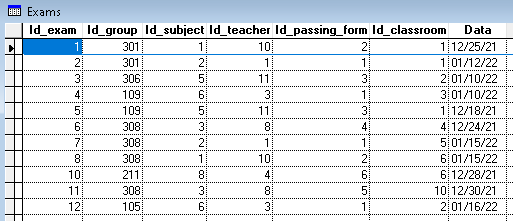
1. Преподаватели (teachers)



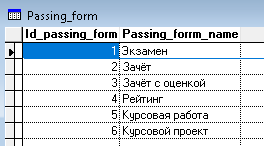
1. Студенты (students)



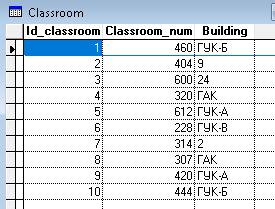
1. Экзамены (exams)



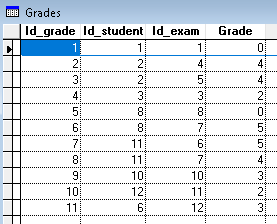
1. Формы сдачи (passing\_form)



1. Аудитории (classroom)

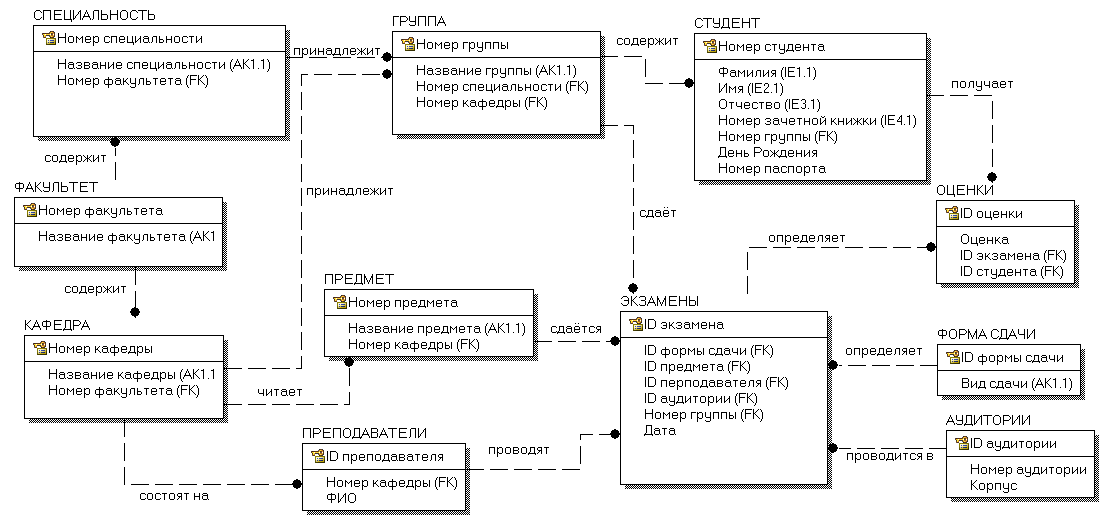


1. Ведомость (grades)

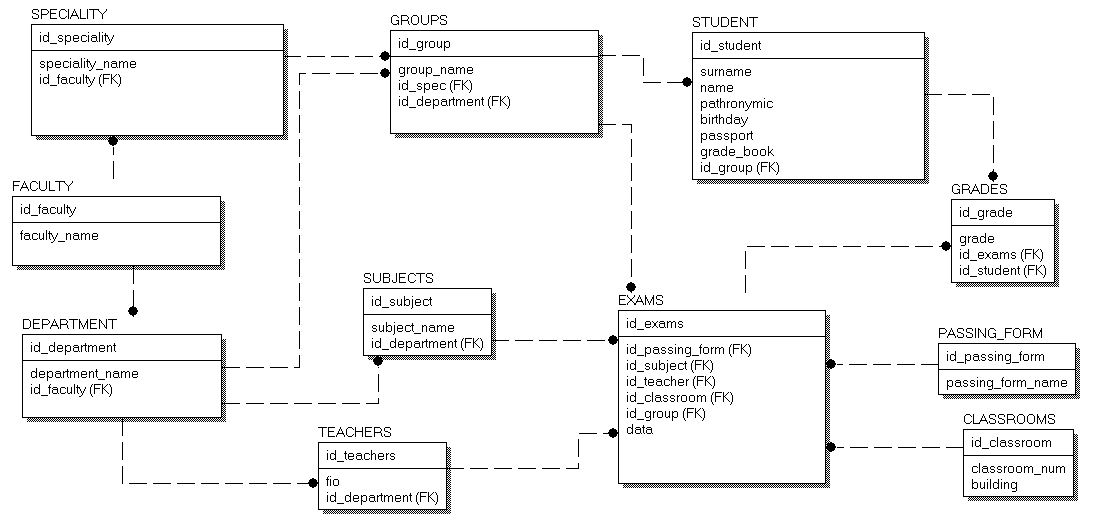


# **Модель базы данных в ErWin**

## **Логическая модель**



## **Физическая модель**



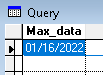
# **Простые запросы**

1. Определить дату самого последнего экзамена

**Код:**

SELECT MAX(data);

FROM exams;  
**Результат:**



1. Список групп, обучающихся на заданной кафедре

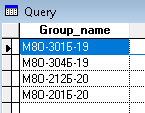
**Код:**

SELECT group\_name;

FROM groups;

WHERE id\_dep = 804

**Результат:**



1. Для каждого задолжника посчитать количество его долгов

**Код:**

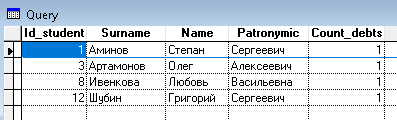
SELECT students.id\_student, surname, name, patronymic, COUNT(id\_grade) as Count\_debts;

FROM grades, students;

WHERE (grade = 2 OR grade = 0) AND students.id\_student = grades.id\_student;

GROUP BY students.id\_student, surname, name, patronymic

**Результат:**



# **Сложные запросы**

1. Пары групп одной специальности заданной кафедры

**Код:**

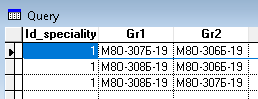
**/\* Соединяем таблицу групп с собой же, выбираем всевозможные пары групп одной специальности по заданной кафедре. \*/**

SELECT DISTINCT a.id\_speciality, a.group\_name as gr1, b.group\_name as gr2;

FROM departament, speciality, groups as a, groups as b;

WHERE a.id\_speciality = b.id\_speciality AND a.id\_group > b.id\_group AND a.id\_dep = 806 AND b.id\_dep = 806

**Результат:**



1. Группы, которые не занимаются у заданного преподавателя

**Код:**

**/\* Отбираем в временную таблицу все группы, которые учатся у заданного преподавателя \*/**

SELECT exams.id\_group;

FROM exams, teachers INTO CURSOR tmp;

WHERE exams.id\_teacher = teachers.id\_teacher AND teachers.fio = 'Формалёв Владимир Фёдорович'

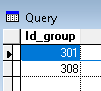
**/\* Проходимся по списку всех групп и выводим те, чьих номеров нет в временной таблице. \*/**

SELECT groups.id\_group;

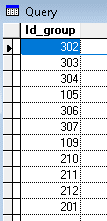
FROM groups;

WHERE groups.id\_group NOT IN (SELECT tmp.id\_group FROM tmp)

**Промежуточная таблица:**



**Результат:**



1. Определить группы, имеющие максимальную среднюю оценку

**Код:**

**/\* Считаем среднюю оценку для каждой группы – записываем это всё во временную таблицу.\*/**

SELECT students.id\_group, AVG(CAST(grades.grade as float)) as avg\_grade;

FROM grades, students INTO CURSOR tmp;

WHERE grades.id\_student = students.id\_student AND grades.grade <> ( 1 ) AND grades.grade <> ( 0 ) ;

GROUP BY students.id\_group

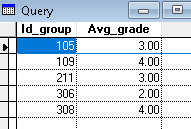
**/\* Во временной таблице находим максимальную среднюю оценку. \*/**

SELECT groups.group\_name, tmp.avg\_grade ;

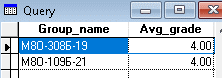
FROM groups, tmp;

WHERE tmp.avg\_grade = (SELECT MAX(avg\_grade) FROM tmp) AND tmp.id\_group = groups.id\_group

**Промежуточная таблица:**

****

**Результат:**



1. Определить задолжников по заданному предмету (2 = 2, 0 = неявка)

**Код:**

SELECT surname, name, patronymic FROM students, grades, exams, subjects;

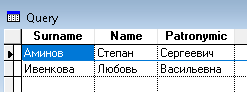
WHERE grades.id\_student = students.id\_student AND;

(grades.grade = (0) OR grades.grade = (2));

AND (exams.id\_subject = subjects.id\_subject) AND subjects.subject\_name = 'Уравнения математической физики' AND (exams.id\_exam = grades.id\_exam);

ORDER BY surname

**Результат:**



1. Преподаватели, у которых выше среднего количество задолжников

**Код:**

**/\* Находим количество должников для каждого преподавателя – записываем это во временную таблицу. \*/**

SELECT DISTINCT teachers.fio as fio, COUNT(grades.id\_student) as Count\_debtors;

FROM teachers, exams, grades INTO CURSOR tmp;

WHERE (grades.grade = 2 OR grades.grade = 0) AND (exams.id\_exam = grades.id\_exam) AND;

(teachers.id\_teacher = exams.id\_teacher);

GROUP BY teachers.fio

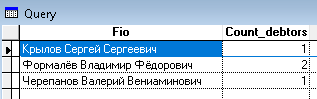
**/\* Во временной таблице ищем преподавателей, у которых количество должников выше среднего \*/**

SELECT tmp.fio;

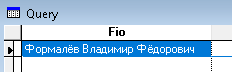
FROM tmp;

WHERE (tmp.Count\_debtors > (SELECT AVG(CAST(Count\_debtors as float)) FROM tmp))

**Промежуточная таблица:**

****

**Результат:**

****

1. По данной специальности определить наиболее успешные предметы (их средняя оценка выше средней по всем предметам)

**Код:**

**/\* Для студентов, принадлежащих заданной специальности, отбираем их оценки и номера экзаменов, когда они получили эти оценки – заносим всё это во временную таблицу. \*/**

SELECT DISTINCT grades.id\_exam as ex, grades.grade;

FROM exams, grades, groups, speciality INTO CURSOR tmp;

WHERE (grade <> 1 AND grade <> 0) AND (grades.id\_exam = exams.id\_exam) AND (exams.id\_group = groups.id\_group) AND;

(groups.id\_speciality = speciality.id\_speciality) AND (speciality\_name == 'Прикладная математика и информатика');

GROUP BY grades.id\_exam, grades.grade

**/\* По созданной временной таблице вычисляем среднюю оценку для каждого предмета. \*/**

SELECT DISTINCT subject\_name as sub, AVG(CAST(grade as float)) as Avg\_grade;

FROM subjects, exams, tmp INTO CURSOR tmp1;

WHERE (exams.id\_exam = tmp.ex) AND (subjects.id\_subject = exams.id\_subject);

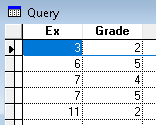
GROUP BY subjects.subject\_name

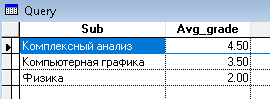
**/\* Вычисляем среднюю оценку по всем предметам, и находим такие средние оценки предметов, которые выше её. \*/**

SELECT tmp1.sub, tmp1.Avg\_grade;

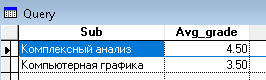
FROM tmp1 WHERE (tmp1.Avg\_grade > (SELECT AVG(CAST(tmp1.Avg\_grade as float)) FROM tmp1))

**Промежуточная таблица 1:**

****

**Промежуточная таблица 2:**

**Результат:**



1. Придумать и реализовать запрос на EXIST (Придумали: «Список студентов, которые не сдали хотя бы один экзамен/зачёт»)

**Код:**

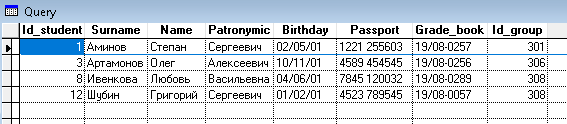
**/\* Во внутреннем подзапросе находим все оценки студента, которые равны 0 или 2. Проверяем, что существует хотя бы одна. \*/**

SELECT students.\*;

FROM students;

WHERE EXISTS (SELECT grades.id\_grade FROM grades WHERE students.id\_student = grades.id\_student AND (grades.grade = 0 OR grades.grade = 2))

**Результат:**

****

1. Определить даты экзаменов с наихудшими средними результатами

**Код:**

**/\* Для каждого экзамена находим среднюю оценку. \*/**

SELECT exams.id\_exam, AVG(CAST(grades.grade AS float)) AS avg\_grade;

FROM exams, grades INTO CURSOR tmp;

WHERE grades.id\_exam = exams.id\_exam;

AND grades.grade <> ( 0 ) ;

AND grades.grade <> ( 1 );

GROUP BY exams.id\_exam

**/\* Находим те экзамены, у которых средняя оценка самая низкая. \*/**

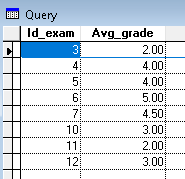
SELECT exams.data, tmp.avg\_grade ;

FROM exams, tmp;

WHERE tmp.avg\_grade = (SELECT MIN(tmp.avg\_grade) FROM tmp);

AND exams.id\_exam = tmp.id\_exam

**Промежуточная таблица:**



**Результат:**

