Лабораторный практикум по курсу

"Основы проектирования баз данных"

**Содержание**

[Лабораторная работа №1. Операторы SDL и DML 3](#_Toc485598867)

[Краткие теоретические данные. 3](#_Toc485598868)

[Задание к лабораторной работе №1 9](#_Toc485598869)

[Лабораторная работа №2. Оператор DML SELECT. Простые запросы 11](#_Toc485598870)

[Краткие теоретические данные. 11](#_Toc485598871)

[Задание к лабораторной работе №2 13](#_Toc485598872)

[Лабораторная работа №3. Вычисляемые столбцы в запросах. Функции работы с датами. Тип NULL. Ограничение количества строк в результирующем запросе. 14](#_Toc485598873)

[Краткие теоретические данные. 14](#_Toc485598874)

[Задание к лабораторной работе №3 19](#_Toc485598875)

[ПРИЛОЖЕНИЕ 1 Таблица для выполнения задания к лабораторным работам 2 и 3 21](#_Toc485598876)

# Лабораторная работа №1. Операторы SDL и DML

## Краткие теоретические данные.

**1.1 Операторы языка SDL**

Команды языка определения схемы данных (Schema Definition Language– SDL) представляют собой инструкции SQL, которые позволяют создавать и модифицировать элементы структуры базы данных. Например, используя SDL, можно создавать, удалять таблицы и изменять их структуру, создавать и удалять индексы.

**1.1.1 Создание таблицы**. Оператор создания таблицы имеет следующий вид:

**CREATE TABLE <tbl\_name>** (CREADE\_DEFINITION)

**<tbl\_name>** – имя создаваемой таблицы

**CREADE\_DEFINITION** включает в себя:

1) перечень столбцов, их типов, и ограничений для значений ячейки в столбце:

col\_name type [NOT NULL | NULL] [DEFAULT default\_value] [AUTO\_INCREMENT]

- для текстовых переменных NULL устанавливается автоматически

*- DEFAULT* – определяет значение по умолчанию

*- [AUTO\_INCREMENT]* – автоматически увеличивающееся значение

2) набор столбцов участвующих в первичном ключе

[CONSTRAINT <pk\_name>] PRIMARY KEY (index\_col\_name,...)

***CONSTRAINT <pk\_name***>- создает имя для первичного ключа

3) набор столбцов участвующих во внешнем ключе, а также таблица с которой осуществляется связь

[CONSTRAINT <fk\_name>] FOREIGN KEY index\_name (index\_col\_name,...)[reference \_definition]

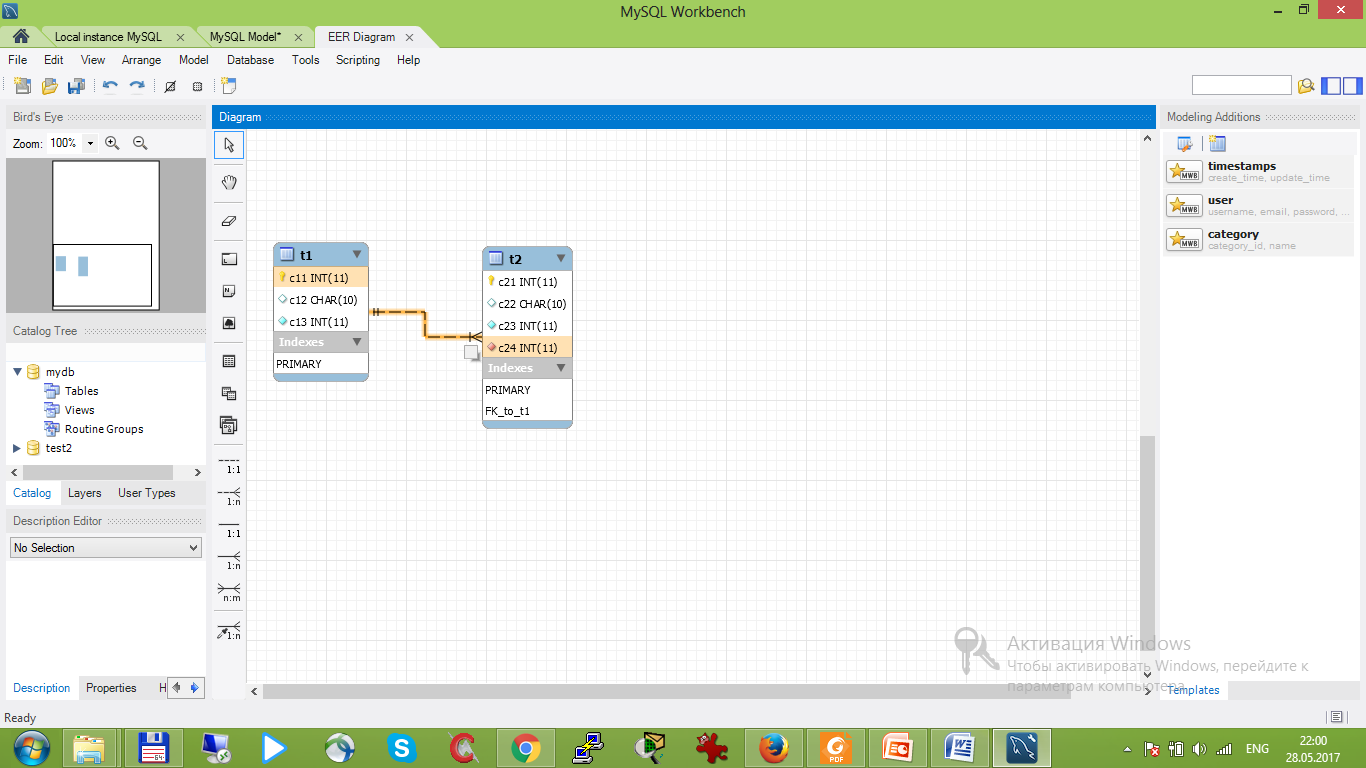
***CONSTRAINT <fk\_name****>-* создает имя для внешнего ключа

Обязательными операндами оператора являются **имя** создаваемой  
таблицы и имя **хотя бы одного столбца** (поля) с указанием типа данных,  
хранимых в этом столбце.

Пример 1.1

|  |
| --- |
| CREATE TABLE t1  (c11 INT NOT NULL IDENTITY,  c12 CHAR(10) DEFAULT 'no\_text',  c13 INT NOT NULL,  CONSTRAINT pk\_t1 PRIMARY KEY (c11));  CREATE TABLE t2  (c21 int not null IDENTITY,  c22 char(10) default 'no\_text',  c23 int not null,  c24 int not null,  CONSTRAINT pk\_t2 PRIMARY KEY (c21),  CONSTRAINT fk\_to\_t1 FOREIGN KEY (c24) REFERENCES t1(c11) ); |

Результат 1.1



**1.1.2 Изменение таблицы.** Для изменения **структуры таблицы** (добавление, удаление полей, изменения типов полей) используется оператор ALTER TABLE. в зависимости от необходимого изменения, оператор ALERT может иметь один из следующих видов:

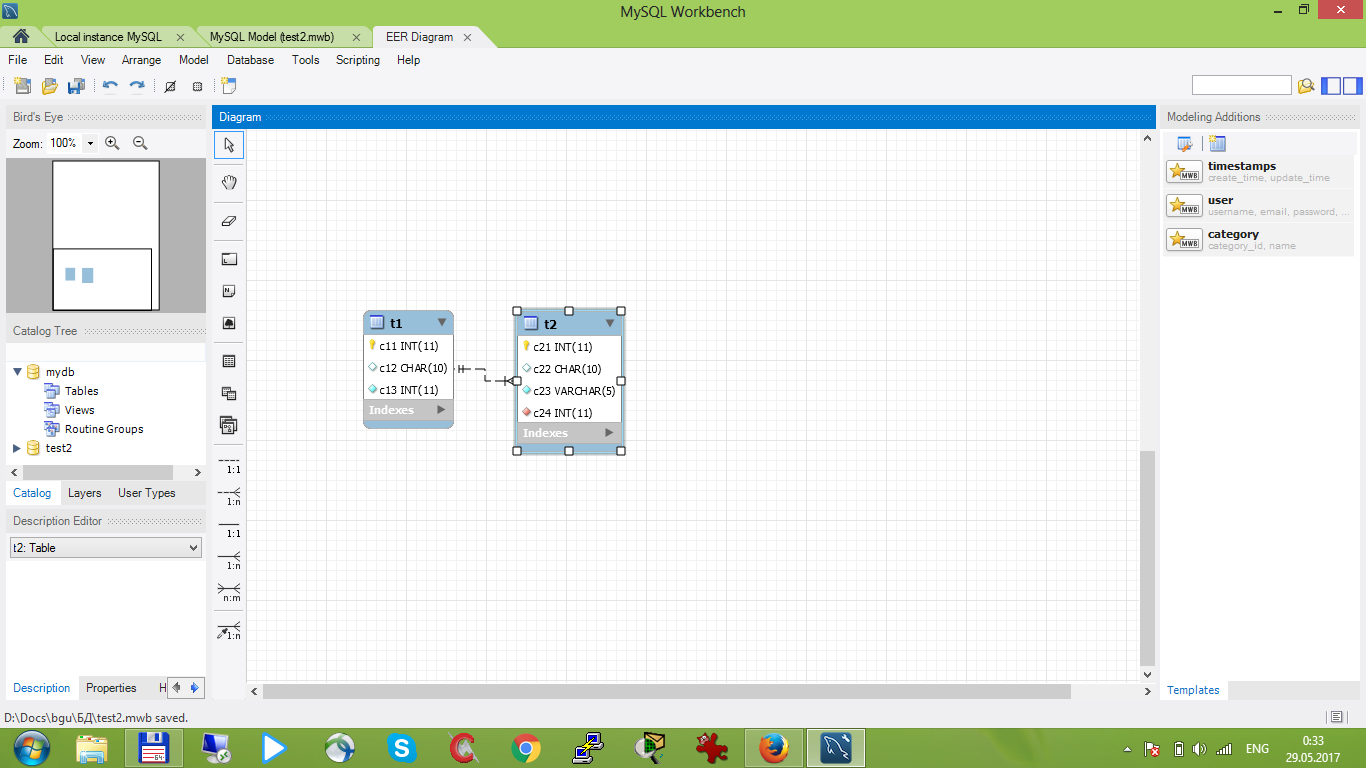
1) Изменение типа столбца:

ALTER TABLE <tab\_name> MODIFY <col\_name> < type> <NOT NULL>

Пример 1.2

|  |
| --- |
| ALTER TABLE t2 modify c23 varchar(5) NOT NULL; |

результат 1.2



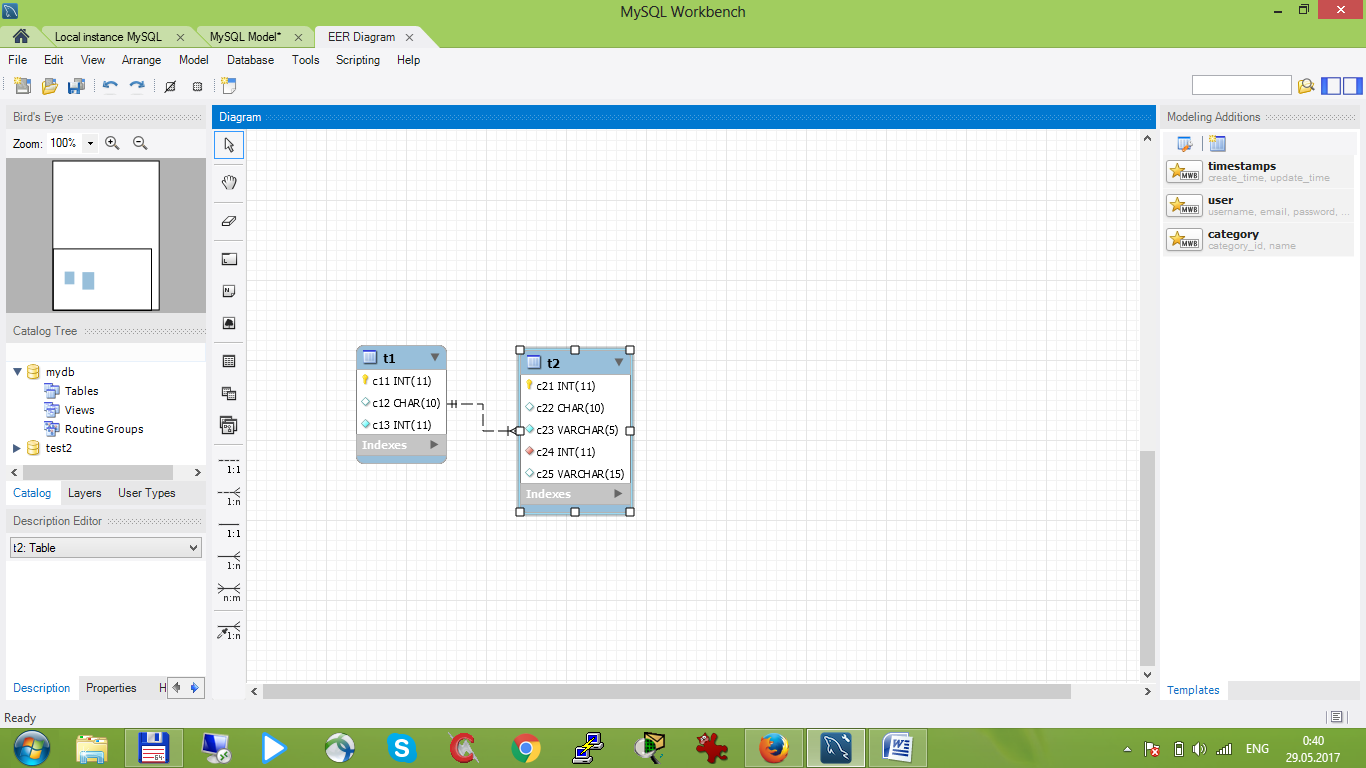
2) Добавление нового столбца:

ALTER TABLE <table\_name> ADD <col\_name> < type> [<NOT NULL> DEFAULT <val>];

Пример 1.3

|  |
| --- |
| ALTER TABLE t2 ADD c25 varchar(15) default 'no\_text'; |

результат 1.3



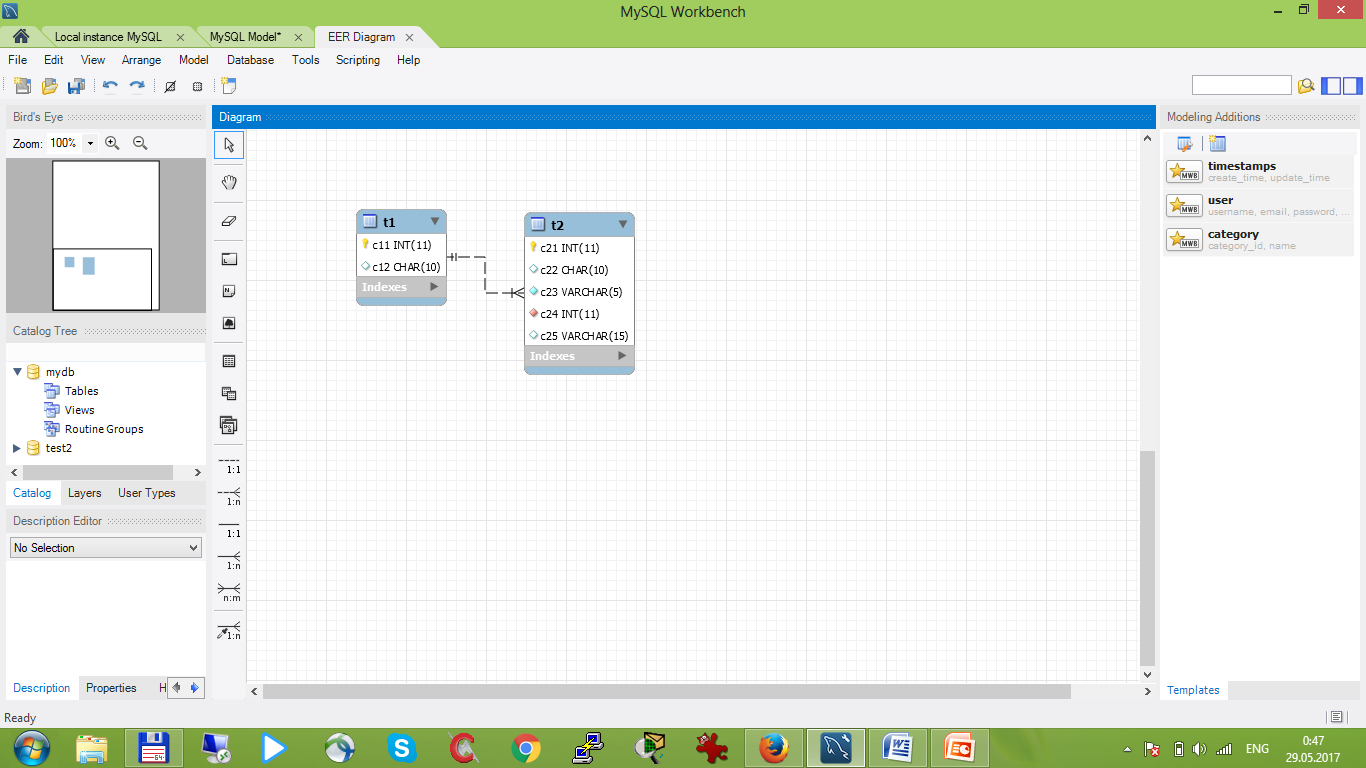
3) Удаление столбца

ALTER TABLE <tab\_name> DROP COLUMN <col\_name>

Пример 1.4

|  |
| --- |
| ALTER TABLE t1 DROP COLUMN c13 varchar(15) default 'no\_text'; |

результат 1.4



4) Изменение имени столбца

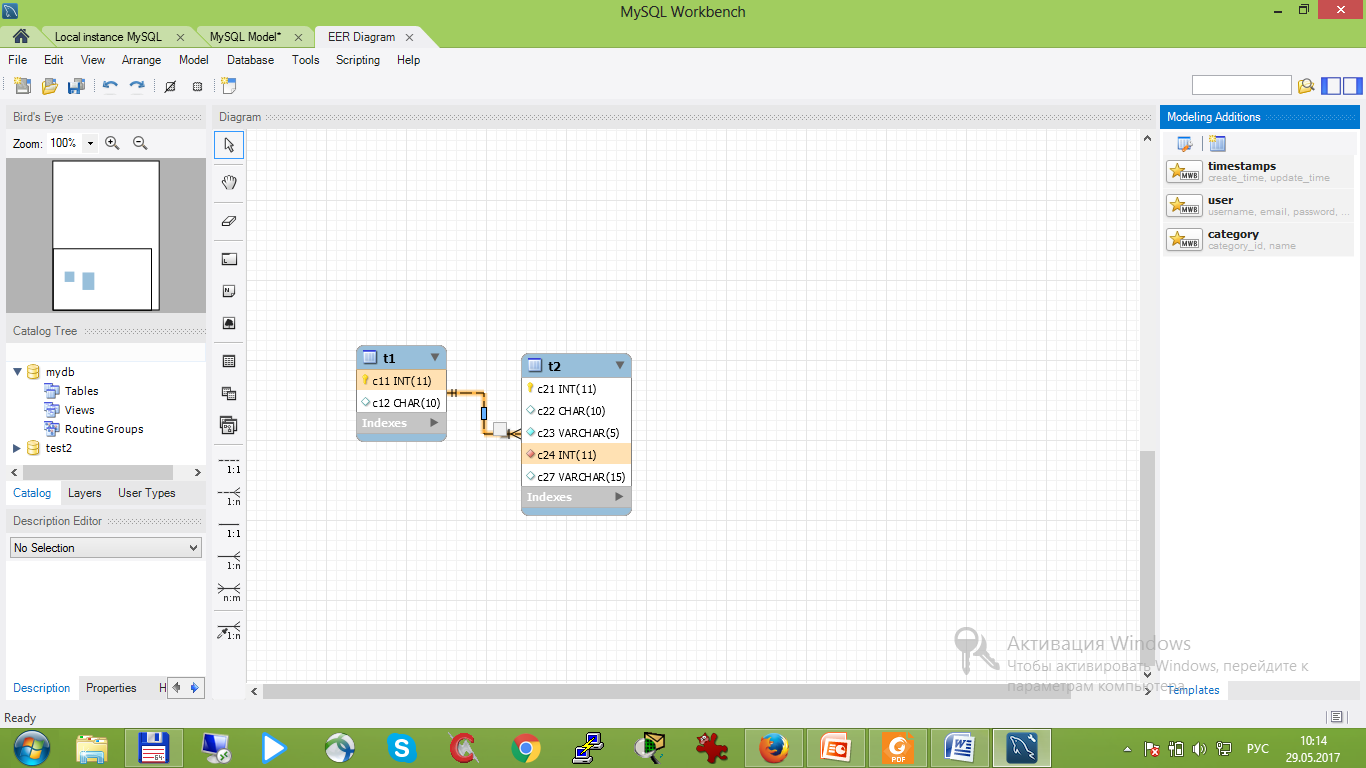
ALTER TABLE <tab\_name> CHANGE <col\_name\_old> <col\_name\_new> < type>

// тип переименнованого столбца указать обязательно

Пример 1.5

|  |
| --- |
| ALTER TABLE t2 CHANGE c25 c27 varchar(15); |

результат 1.5



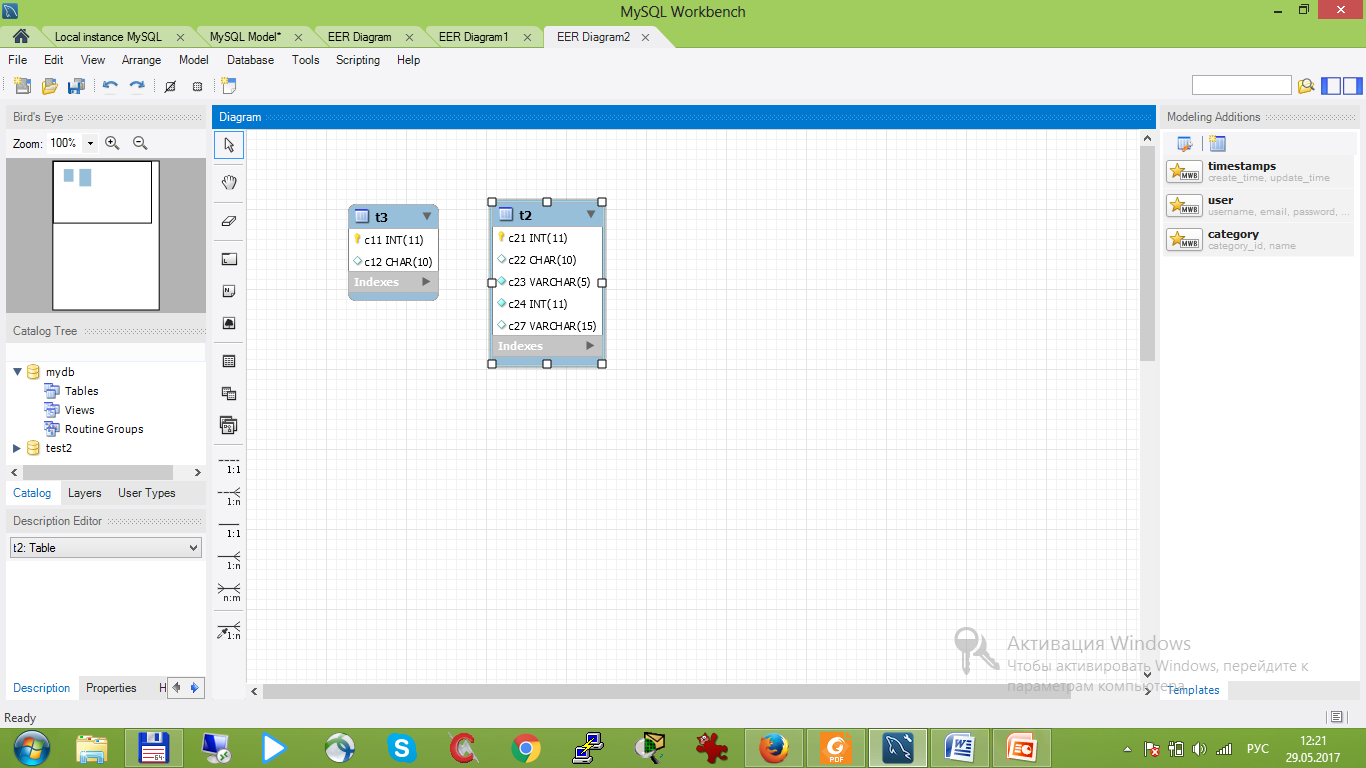
6) Удаление ключа внешнего ключа

ALTER TABLE <table\_name> drop foreign key <key\_name>

Пример 1.7

|  |
| --- |
| ALTER TABLE t2 drop foreign key FK\_to\_t1; |

результат 1.7



7) назначение нового внешнего ключа:

ALTER TABLE <tab\_name1> add [CONSTRAINT <fk\_name>] foreign key (<col\_name>) references <tab\_name2>(<col\_name3>)

Пример 1.8

|  |
| --- |
| ALTER TABLE t3 ADD CONSTRAINT FK\_to\_t1 FOREIGN KEY (c24) REFERENCES t1(c11); |

8) Назначение нового первичного ключа (не в MySQL)

ALTER TABLE <tab\_name1> add [CONSTRAINT <key\_name>] primary key(col\_name1, col\_nam2)

9) Удаление первичного ключа (не в MySQL)

ALTER TABLE <tab\_name1> DROP PRIMARY KEY <key\_name>

**1.1.3 Удаление таблицы**

DROP TABLE <table\_name>

**1.2 Операторы языка DML**

Язык DML (Data Manipulation Language) является средством управления данными в внутри объектов схемы (БД). К запросам языка манипуляции относятся запросы на добавление, удаление и модификацию кортежей.

**1.2.1 Добавление кортежа** производится командой:

*INSERT INTO имя\_таблицы [(<список столбцов>)] VALUES (<список*  
*значений>)*

Пример 1.8

|  |
| --- |
| INSERT INTO t3 (c12) VALUES ('Петров'), ('Иванов'), ('Сидоров'); |

Результат 1.8:

|  |  |
| --- | --- |
| C11 | C12 |
| 1 | Петров |
| 2 | Иванов |
| 3 | Сидоров |
|  |  |

**1.2.2** Операция **обновления данных** UPDATE требуется при изменении данных, которые надо отразить в базе данных.  
Запрос на обновление может изменить одну или несколько записей.

**-- обновление всех записей в столбце**

**UPDATE tab SET <col\_name>=<val>**

Пример 1.9

|  |
| --- |
| UPDATE t3 SET c12=null |

Результат 1.9:

|  |  |
| --- | --- |
| C11 | C12 |
| 1 | null |
| 2 | null |
| 3 | null |
|  |  |

**-- обновление одной или группы записей (по условию)**

**UPDATE tab SET <col\_name>=<val> WHERE <col\_name1>=<val>**

Пример 1.9

|  |
| --- |
| UPDATE t3 SET c12=null c12='Иванов' where c11=2; |

Результат 1.10:

|  |  |
| --- | --- |
| C11 | C12 |
| 1 | Null |
| 2 | Иванов |
| 3 | Null |

**1.2.3** Оператор **удаления данных** DELETE позволяет удалить одну или  
несколько строк из таблицы в соответствии с условиями, которые задаются  
для удаляемых строк. Синтаксис оператора DELETE следующий:

*DELETE FROM <имя\_таблицы> [WHERE <условия\_отбора>]*

Если условия отбора не задаются, то из таблицы удаляются все строки.

## Задание к лабораторной работе №1

1) Создайте следующие таблицы

1. таблица STUDENTS содержащая поля

- идентификатор студента

- имя

-фамилию

-отчество

-идентификатор группы

поле идентификатора студента должно быть первичным ключом в таблице

1. Таблица TEACHERS содержащая поля

- идентификатор преподавателя

- имя

-фамилию

-отчество

поле идентификатора преподавателя должно быть первичным ключом в таблице

1. Таблица GROUPS содержащая поля

- идентификатор группы

- наименование

- курс

поле идентификатора группы должно быть первичным ключом в таблице

1. 4 Таблица PLAN содержащая поля

- идентификатор группы

- идентификатор преподавателя

- идентификатор предмета

Все поля образуют первичный ключ

1. 5 Таблица SUBJECTS содержащая поля

- идентификатор предмета

- наименование предмета

- количество часов по предмету

2. Измените таблицы, создав связи между ними

- STUDENTS. *идентификатор группы* -- GROUPS.*идентификатор группы*

- PLAN.*идентификатор преподавателя* -- TEACHERS.*идентификатор преподавателя*

- PLAN.*идентификатор предмета* -- SUBJECTS.*идентификатор предмета*

- PLAN.*идентификатор группы* -- GROUPS.*идентификатор группы*

3. Заполнить таблицы согласно данным Таблицы 1.1 и Таблицы 1.2

Таблица 1.1

|  |  |  |  |
| --- | --- | --- | --- |
| **ФИО** | **дата рождения** | **Номер группы** | **Курс** |
| Федоренко П.Р. | 25.12.1997 | ПО135 | 1 |
| Зингел О. | 25.12.1985 | ПО135 | 1 |
| Михеенок П.Г. | 05.02.1993 | ПО134 | 1 |
| Савицкаян Н. | 22.09.1987 | ПО235 | 2 |
| Ковальчук М.Е. | 17.06.1992 | ПО235 | 2 |
| Заболотная Н.Г. | 18.06.1992 | ПО335 | 3 |
| КовригоТ.Р. | 13.05.1992 | ПО335 | 3 |
| Шарапо Н. | 14.08.1992 | ПО335 | 3 |

Таблица 1.2

|  |  |  |  |
| --- | --- | --- | --- |
| **Фио** | **Предмет** | **Курс** | **Количество часов** |
| Сафроненко Н | Физика | 1 | 200 |
| Зайцева Н.У. | Математика | 1 | 120 |
| Лисопад П.Г. | Основы алгоритмизации | 2 | 70 |
| Клюев К.Н. | Проектирование БД | 2 | 130 |
| Рогачевский П.Н. | Средства визуального программирования | 3 | 90 |
| Макаров Н.Г. | Объектно-ориентированное программирование | 3 | 70 |

4) В таблице STUDENTS внесите изменения таким образом, чтобы все студенты учащиеся в группе ПО135 числились в группе 134.

5) Из таблицы групп удалите запись о группе по 135

6) в таблице SUBJECTS увеличьте количество часов на 30 для предметов

Средства визуального программирования, Объектно-ориентированное программирование

7) в таблицу SUBJECTS добавьте столбец, в который будут внесены данные для формы контроля по предметам (по всем предметам сдаются экзамены, кроме " Основы алгоритмизации " -зачет). обновите таблицу SUBJECTS с учетом форм контроля по предметам.

8) Из таблиц STUDENTS и TEACHERS удалите столбцы с данными отчества.

# Лабораторная работа №2. Оператор DML SELECT. Простые запросы

## Краткие теоретические данные.

Для выполнения запросов в SQL существует один единственный оператор – SELECT. Синтаксис оператора SELECT имеет следующий вид:

*SELECT [ ALL| DISTINCT] <Список полей>|\**

*FROM <Список таблиц>*

*[WHERE <Предикат-условие выборки или соединения>]*

*[GROUP BY <Список полей результата>]*

*[HAVING <Предикат-условие для группы>]*

*[ORDER BY <Список полей, по которым упорядочить вывод>];*

**SELECT** – ключевое слово, которое сообщает СУБД, что эта команда –  
запрос. За ним может следовать способ выборки:

- **ALL -** ключевое слово, означает, что в результирующий набор строк  
включаются **все строки** (не столбцы), удовлетворяющие условиям запроса, т.е. в результирующий набор могут попасть одинаковые строки.

- **DISTINCT -** ключевое слово,означает, что в результирующий набор  
включаются только различные строки, то есть дубликаты строк результата не  
включаются в набор.

- список полей – это список перечисленных через запятую столбцов,  
которые выбираются запросом из таблиц.

- символ **\*** (звездочка) означает, что в результирующий набор  
включаются все столбцы из исходных таблиц запроса.

В разделе **FROM** задается перечень исходных отношений (таблиц)  
запроса. В случае, если указано более одного имени таблицы, неявно  
подразумевается, что над перечисленными таблицами осуществляется  
операция **декартова произведения**.   
Разделы SELECT и FROM являются обязательными, все другие разделы  
являются необязательными.

Допустимо в раздел SELECT включать не только имена полей и \*.

В разделе **WHERE** задаются условия отбора строк результата или  
условия соединения кортежей исходных таблиц, подобно операции  
условного соединения в реляционной алгебре.  
В выражении условий раздела WHERE могут быть использованы  
следующие предикаты:

**- Предикаты сравнения (=, <>, >, >=, <, <=)**, которые имеют  
традиционный смысл.

**- Предикат Between A and B** – принимает значения между А и В.  
Предикат истинен, когда сравниваемое значение попадает в заданный  
диапазон, включая границы диапазона. Одновременно в стандарте задан и  
противоположный предикат Not Between A and B, который истинен тогда,  
когда сравниваемое значение не попадает в заданный интервал, включая его  
границы.

- **Предикат вхождения в множество IN** (множество) истинен тогда,  
когда сравниваемое значение входит в множество заданных значений. При  
этом множество значений может быть задано простым перечислением или  
встроенным подзапросом. Одновременно существует противоположный  
предикат NOT IN (множество), который истинен тогда, когда сравниваемое  
значение не входит в заданное множество.

**- Предикаты сравнения с образцом LIKE и NOT LIKE**. Предикат LIKE  
требует задания шаблона, с которым сравнивается заданное значение,  
предикат истинен, если сравниваемое значение соответствует шаблону, и  
ложен в противном случае. Предикат NOT LIKE имеет противоположный  
смысл. Шаблон может содержать % (\* для Access) для обозначения любого  
числа любых символов; \_ (? для Access) для обозначения любого одного  
символа.

- **Предикат сравнения с неопределенным значением IS NULL**. Для  
выявления равенства значения некоторого атрибута неопределенному  
значению применяют специальные стандартные предикаты: <имя атрибута> IS NULL и <имя атрибута> IS NOT NULL

- **Предикат существования EXIST и не существования NOT EXIST**.  
Применяется во вложенных запросах для определения непустого или пустого  
множества, являющегося результатом выборки

В разделе **GROUP BY** задается список полей группировки. **GROUP BY**  
группирует записи данных и объединяет в одну запись все записи данных,  
которые содержат идентичные значения в указанном поле (или полях).  
**WHERE** определяет, какие записи должны участвовать в группировании, т.е.  
фильтрует до группирования

В разделе **HAVING** задаются предикаты-условия, накладываемые на  
каждую группу. **HAVING** используется для фильтрации записей,  
полученных в результате группировки.

|  |
| --- |
| !!! WHERE фильтрует до группирования.  !!! HAVING определяет, какие из получившихся в результате группировки записей будут включены в результирующую выборку, т.е. фильтрует записи после группирования. |

В части **ORDER BY** задается список полей упорядочения результата, то  
есть список полей, который определяет порядок сортировки в  
результирующем отношении. Например, если первым полем списка будет  
указан Шифр группы, а вторым Фамилия, то в результирующем отношении  
записи сначала будут расположены в порядке возрастания шифра группы, а  
затем в рамках одной группы записи будут отсортированы по фамилии в  
алфавитном порядке. Также можно определить направление порядка: по убыванию **ASC** – сортировка по возрастанию (стоит по умолчанию), **DESC** – сортировка по убыванию

**Применение агрегатных функций**

В SQL добавлены дополнительные функции, которые позволяют  
вычислять обобщенные групповые значения. Для применения агрегатных  
функций предполагается предварительная операция группировки. При отсутствии группировки вся выборка является одной группой.

Некоторые агрегатные функции представлены в таблице 2.1.

**Таблица 2.1**

|  |  |  |
| --- | --- | --- |
| **Функция** | **Результат** | **Примечание** |
| COUNT | Количество строк или непустых значений полей, которые выбрал запрос | **Тип поля - не имеет значение** |
| SUM | Сумма всех выбранных значений данного поля | **Тип поля - числовой** |
| AVG | Среднеарифметическое значение всех выбранных значений данного поля | **Тип поля - числовой** |
| MIN | Наименьшее из всех выбранных значений данного поля | **Тип поля - числовой/символьный** |
| MAX | Наибольшее из всех выбранных значений данного поля | **Тип поля - числовой/символьный** |

## Задание к лабораторной работе №2

1. Вывести фамилии студентов, в которых присутствуют буквы "б" и/или "о" в любых комбинациях и порядке
2. Вывести сведения о студентах, фамилии которых начинаются с буквы К
3. Вывести сведения о студентах, фамилии которых содержат 8 символов и более
4. Вывести сведения о студентах, фамилии которых содержат или больше, или меньше семи символов
5. Найти всех студентов учащихся на ФПМ очном 1, 2 и 3 курса, отсортировать результаты по полю отчества
6. Найти всех студентов учащихся на ФПК очном, отсортировать результаты по полю отчества в убывающем порядке
7. Определить средний балл на заочной форме обучения каждого факультета.
8. Найти значения максимальных средних баллов для каждого факультета и курса
9. Определить список факультетов, средний балл успеваемости на которых больше 7.5
10. Определить список курсов с указанием факультета, средний балл успеваемости на которых больше 7.5
11. Определить форму обучения с привязкой к факультету, для которой минимальный балл успеваемости для одного из студентов меньше 7.
12. Определить, какое количество часов студенты ФПК 3-го курса заочной формы обучения тратят на самостоятельную подготовку
13. Вывести список факультетов, курсов, форм обучения на которых студенты тратят более 150 часов на самостоятельную подготовку
14. определить среднее число часов, которое тратит студент факультета для каждой формы обучения за один год на этой форме.

# Лабораторная работа №3. Вычисляемые столбцы в запросах. Функции работы с датами. Тип NULL. Ограничение количества строк в результирующем запросе.

## Краткие теоретические данные.

1. Вычисляемые столбцы в запросах SELECT

Оператор SELECT может содержать не только перечень столбцов, которые должны войти в результирующую выборку, то и вычисляемые поля, т.е:

*SELECT < поле1>, < поле1>-< поле2>*

*FROM <table1>;*

*- возвращает выборку всех строк первого поля и разности первого и второго поля*

*SELECT < поле1>, 1.05\*< поле2>*

*FROM <table1>;*

*- возвращает выборку всех строк первого поля и второго поля, увеличенного на 5%*

Для наглядности результирующей выборки для вычисляемого столбца можно определить имя

*SELECT < поле1>, 1.05\*< поле2> as increase*

*FROM <table1>;*

Агрегатные функции также могут быть использованы как вычисляемый столбец

*SELECT Сount(\*) FROM <table1>;*

*- возвращает количество всех строк*

*SELECT < поле1>, MAX(< поле2>) as maximum FROM <table1> GROUP BY < поле1>;*

*- в результате запроса поля таблицы будут сгруппированы по полю 1 и в каждой группе найдено максимальное значение поля 2*

При наличии в запросе группировки (GROUP BY) перечень столбцов результирующей выборки может содержать только те столбцы по которым выполняется группировка, а также столбцы к которым применена одна из агрегативных функций.

2. Функции работы с датами.

В СУБД реализовано достаточно большое количество функций для работы с данными в формате времени и даты. Наиболее часто используемые приведены ниже:

**DAY** ( *date* )  -  целочисленное представление дня указанной даты. Эта функция эквивалентна функции **DATEPART**(*dd*, *date*).

**MONTH** ( *date* )   -  целочисленное представление месяца указанной даты. Эта функция эквивалентна функции **DATEPART**(*mm*, *date*).

**YEAR** ( *date* )   -  целочисленное представление года указанной даты.Эта функция эквивалентна функции **DATEPART**(*yy*, *date*).

**CURRENT\_DATE** *– возвращает текущее значение даты*

**CURRENT\_TIME-** *возвращает текущее значение времени*

**DAYNAME(date)** *– возвращает название дня недели указанной даты*

**DAYOFYEAR(date)***- возвращает номер дня в году*

**WEEKOFYEAR(date)** *– возвращает номер недели в году*

Функции даты и времени могут использоваться в выражениях SELECT при вычислении значений столбцов, в условии выборки сток, сортировки, группировки, а также условиях выборки групп.

|  |  |  |  |
| --- | --- | --- | --- |
| **tab\_1** | | | |
| **Id** | **Department** | **Name** | **Birthday** |
| 1 | Продаж | Иванов | 06-12-1985 |
| 2 | Продаж | Петрова | 04-23-1976 |
| 3 | Кадров | Сидоров | 08-27-1976 |
| 4 | Кадров | Кунин | 05-13-1983 |
| 5 | Технический | Мартынов | 11-11-1984 |

*SELECT department, name, YEAR(CURRENT\_DATE) -YEAR(Birthday) as age*

*FROM tab\_1;*

*- возвращает выборку с названием отдела, фамилией сотрудника, возрастом сотрудника*

*SELECT department, name, birthday*

*FROM tab\_1*

*WHERE YEAR(CURRENT\_DATE -YEAR(Birthday) >40 ORDER BY birthday;*

*- возвращает выборку, содержащую сведения о сотрудниках старше 40 лет отсортированную по дате рождения*

*SELECT department, COUNT(\*)*

*FROM tab\_1*

*GROUP BY department*

*HAVING MAX(YEAR(CURRENT\_DATE -YEAR(Birthday))<40;*

*- возвращает выборку с перечнем отделов, сотрудники которых младше 40 лет;*

3. Тип NULL, его влияние на значения агрегатных функции

 Тип NULL особым образом обрабатывается как в условиях оператора SELECT, так и агрегатными функциями. Для нахождения ячеек со значением NULL необходимо использовать оператор **IS** или **NOT IS**.

|  |  |
| --- | --- |
| **tbl1** | |
| **id** | **f3** |
| 1 | по1701 |
| 2 | NULL |
| 3 | по1708 |

*SELECT id FROM tbl1 WHERE f3 = NULL*

*- вернет пустое множество*

*SELECT id FROM tbl1 WHERE f3 IS NULL*

*- вернет одну строку*

*SELECT id FROM tbl1 WHERE f3 IS NOT NULL*

*- вернет ДВЕ строки*

Для типа **AVG()**, **MIN()**, **MAX()**, **SUM()**, **COUNT()**, если значение столбца будет содержать **NULL то такая строка игнорируется**

Если какие либо из значений содержащихся в столбце, равны **NULL**, при вычислении результата функции они исключаются!

Если все значения в столбце равны **NULL**, то функции **AVG()**, **SUM()**, **MIN()**, **MAX()** возвращают значения **NULL.** Функция **COUNT()** возвращает ноль.

Если в столбце нет значений (т.е. столбец пуст), то функции **AVG()**, **SUM()**, **MIN()**, **MAX()** возвращают значения **NULL.** Функция **COUNT()** возвращает ноль!

Функция **COUNT(\*)** подсчитывает количество строк и не зависит от наличия или отсутствия в столбце значений **NULL.** Если строк в столбце нет, то эта функция возвращает ноль

**SUM(значение столбца - NULL)** возвращает **NULL.**

4. Ограничение количества строк в результирующем запросе

Для ограничения количества выводимых строк в операторе SELECT можно использовать ключевое слово **LIMIT n**, где n- целое число, количество выводимых срок

|  |  |  |  |
| --- | --- | --- | --- |
| **tab\_1** | | | |
| **Id** | **Department** | **Name** | **Birthday** |
| 1 | Продаж | Иванов | 06-12-1985 |
| 2 | Продаж | Петрова | 04-23-1976 |
| 3 | Кадров | Сидоров | 08-27-1976 |
| 4 | Кадров | Кунин | 05-13-1983 |
| 5 | Технический | Мартынов | 11-11-1984 |

*SELECT \* FROM tab\_1 WHERE Birthday IS NOT NULL*

*- возвращает выборку из 5 строк*

*SELECT \* FROM tab\_1 WHERE Birthday IS NOT NULL LIMIT 3*

*- возвращает выборку из первых 3 строк*

Ограничение LIMIT можно использовать в запросах поиска максимального или минимального значения в некоторых задачах. Например при необходимости поиска возраста самого младшего сотрудника можно использовать запрос:

*SELECT MAX(Birthday) FROM tab\_1*

*- возвращает ТОЛЬКО дату рождения*

Для вывода информации об этом сотруднике возможно использовать вложенный запрос, так как использование агрегатной функции делает невозможным включить в перечень выбираемых столбцов Name и Department, однако можно использовать и запрос вида

*SELECT \* FROM tab\_1 ORDER BY Birthday DESC LIMIT 1*

*- возвращает все столбцы и строки исходной таблицы, при этом все строки отсортированы по столбцу Birthday, а итоговая выборка ограничена первой строкой которая содержит максимальное значение столбца Birthday.*

Сочетания DESC LIMIT 1 и ASC LIMIT 1 можно использовать для поиска макси мальных и минимальных значений, однако следует учитывать возможность наличия нескольких максимумов и минимумов.

При необходимости исключения дублирующих строк из результата запроса следует использовать ключевое слово **DISTINCT**

*SELECT Department DISTINCT FROM tab\_1;*

*- возвращает перечень отделов компании.*

Ключевое слово**DISTINCT** возможно использовать совместно с НЕКОТОРЫМИ агрегатными функциями, например с **COUNT()**

*SELECT COUNT(DISTINCT Department) FROM tab\_1;*

*- возвращает количество отделов компании.*

При этом следует помнить:

- если **DISTINCT** используется с агрегатной функцией, то ее аргументом может быть только имя столбца, выражение не может быть аргументом.

- в функциях **MIN()**, **MAX()** нет смысла использовать **DISTINCT;**

- к функции **COUNT(\*)** не применимо **DISTINCT**, так как она просто подсчитывает число строк;

- в одном запросе **DISTINCT** можно употреблять только один раз.

## Задание к лабораторной работе №3

1. Вывести информацию о студентах, которым на конец текущего года не исполниться 31 года
2. Вывести средний возраст студентов по факультету, курсу и форме обучения
3. Вывести информацию о студентах, которым на текущий момент исполнился 31 год
4. Найти количество студентов от 32 до 45 лет
5. Определить сколько студентов старше 45 лет учатся на каждом факультете на каждом курсе
6. Определит факультет, на котором учится максимальное количество студентов до 35
7. Определить факультет с минимальным числом студентов на заочной форме обучения
8. Определить факультет, на котором учится максимальное число первокурсников очников или заочников
9. Определить факультет и курс, на которых средний возраст студентов меньше 32 лет
10. Определить факультет и курс и общую численность курса на котором учится студенты не младше 25
11. Найти факультет и курс, на которых студентов без отчества обучается больше одного
12. Вывести информацию о самом успевающем студенте университета.
13. Вывести список форм обучения в университете
14. Определить число форм обучения в университете на текущий момент
15. Определить число форм обучения на каждом факультете
16. Определить общее количество иностранцев и студентов не иностранцев фамилии которых начинаются на букву С
17. Определить процент учащихся иностранцев к числу студентов в университете\*

\* для корректного вывода результатов следует использовать функцию явного преобразования типов cast(X as TYPE), в данном случае следует приводить к типу float

# ПРИЛОЖЕНИЕ 1 Таблица для выполнения задания к лабораторным работам 2 и 3

create database UNIVER;

use UNIVER;

Create table stud

(id int not null IDENTITY(1,1),

last\_name nvarchar(25) not null,

f\_name nvarchar(25) not null,

s\_name nvarchar(25),

form nvarchar(10) not null default 'очно',

faculty nvarchar(10) not null default 'ФПМ',

year int not null default 1,

all\_h int default NULL,

inclass\_h int default NULL,

br\_date date,

in\_date date,

exm float default NULL,

CONSTRAINT PK\_stud Primary key (id)

);

insert into stud

(last\_name,f\_name,s\_name,form,faculty,year,all\_h,inclass\_h,br\_date,in\_date,exm) values

(N'Стрынгель',N'К',null,N'заочная',N'ФПК',1,300,100,'19831212','20160901',8),

(N'Козлова',N'Д',N'Е',N'заочная',N'ФПК',2,300,100,'19831012','20150901',8.4),

(N'Федоров',N'Н',N'Н',N'заочная',N'ФПК',3,300,100,'19811207','20140901',7),

(N'Рингель',N'П',N'О',N'заочная',N'ФПК',3,300,100,'19730215','20160901',8),

(N'Бежик',N'Н',N'Н',N'вечерняя',N'ФПК',1,500,400,'19931211','2016-09-01',4.5),

(N'Осипчик',N'Н',N'Н',N'вечерняя',N'ФПК',1,500,400,'19831216','20150901',7.7),

(N'Белый',N'С',N'С',N'вечерняя',N'ФПК',2,450,370,'19870627','20150901',6.7),

(N'Ботяновский',N'А',N'С',N'вечерняя',N'ФПК',2,450,370,'19870723','20150901',7.6),

(N'Слободницкий',N'С',N'А',N'вечерняя',N'ФПК',2,450,370,'19870803','20150901',6.7),

(N'Рогатка',N'П',N'Р',N'очная',N'ФПМ',1,500,450,'19861027','20160901',7.4),

(N'Федоренко',N'П',N'Р',N'очная',N'ФПМ',1,500,450,'19950426','20160901',5.6),

(N'Зингель',N'П',N'В',N'очная',N'ФПМ',2,500,450,'19900425','20150901',3.4),

(N'Михеенок',N'Л',N'Н',N'очная',N'ФПМ',2,500,450,'19890313','20150901',5.3),

(N'Савицкая',N'Л',N'Н',N'очная',N'ФПМ',3,450,400, '19950705','20140901',7.7),

(N'Ковальчук',N'О',N'Е',N'заочная',N'ФПМ',1,350,100,'19640523','20160901',7.6),

(N'Заболотная',N'Л',N'И',N'заочная',N'ФПМ',1,350,100,'19860914','20160901',4.7),

(N'Ковриго',N'И',null,N'заочная',N'ФПМ',2,360,120,'19920301', '20150901',7.7),

(N'Шарапо',N'М',null,N'заочная',N'ФПМ',2,360,120,'19970325', '20150901',8.7),

(N'Сафроненко',N'Н',N'Л',N'заочная',N'ФПМ',3,370,130, '19920525','20140901',7.7),

(N'Зайцева',N'Т',N'Я',N'заочная',N'ФПМ',3,370,130,'19940725','20140901',5.6);

**id last\_name f\_name s\_name form faculty year all\_h inclass\_h br\_date in\_date exm**

1 Стрынгель К NULL заочная ФПК 1 300 100 1983-12-12 2016-09-01 8

2 Козлова Д Е заочная ФПК 2 300 100 1983-10-12 2015-09-01 8,4

3 Федоров Н Н заочная ФПК 3 300 100 1981-12-07 2014-09-01 7

4 Рингель П О заочная ФПК 3 300 100 1973-02-15 2016-09-01 8

5 Бежик Н Н вечерняя ФПК 1 500 400 1993-12-11 2016-09-01 4,5

6 Осипчик Н Н вечерняя ФПК 1 500 400 1983-12-16 2015-09-01 7,7

7 Белый С С вечерняя ФПК 2 450 370 1987-06-27 2015-09-01 6,7

8 Ботяновский А С вечерняя ФПК 2 450 370 1987-07-23 2015-09-01 7,6

9 Слободницкий С А вечерняя ФПК 2 450 370 1987-08-03 2015-09-01 6,7

10 Рогатка П Р очная ФПМ 1 500 450 1986-10-27 2016-09-01 7,4

11 Федоренко П Р очная ФПМ 1 500 450 1995-04-26 2016-09-01 5,6

12 Зингель П В очная ФПМ 2 500 450 1990-04-25 2015-09-01 3,4

13 Михеенок Л Н очная ФПМ 2 500 450 1989-03-13 2015-09-01 5,3

14 Савицкая Л Н очная ФПМ 3 450 400 1995-07-05 2014-09-01 7,7

15 Ковальчук О Е заочная ФПМ 1 350 100 1964-05-23 2016-09-01 7,6

16 Заболотная Л И заочная ФПМ 1 350 100 1986-09-14 2016-09-01 4,7

17 Ковриго И NULL заочная ФПМ 2 360 120 1992-03-01 2015-09-01 7,7

18 Шарапо М NULL заочная ФПМ 2 360 120 1997-03-25 2015-09-01 8,7

19 Сафроненко Н Л заочная ФПМ 3 370 130 1992-05-25 2014-09-01 7,7

20 Зайцева Т Я заочная ФПМ 3 370 130 1994-07-25 2014-09-01 5,6