**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ**

**Федеральное государственное бюджетное образовательное учреждение  
высшего профессионального образования  
Национальный исследовательский университет «МЭИ»**

Сидорова Н.П.

**Основы T-SQL.**

**Сборник лабораторных работ**

Методическое пособие

по курсу «Базы данных»

Для студентов, обучающихся по направлению «Прикладная математика и информатика»

Москва Издательский дом МЭИ 2013

**УДК**

**621.398**

**С 374**

Рецензент д.т.н., проф. Вагин В.Н.

Сидорова Н.П**. Основы T-SQL. Сборник лабораторных работ по курсу «Базы данных».** Методическое пособие. М.: Изд. дом МЭИ, 2013.- 32 с.

Сборник включает 8 лабораторных работ, отражающих основные возможности работы с базами данных в среде SQL Server. Он содержит необходимые сведения о синтаксисе языка T-SQL и особенностях его использования для решения задач обработки данных, программирования хранимых процедур и триггеров и решения задач администрирования баз данных.

Пособие предназначено для студентов, обучающихся по направлению «Прикладная математика и информатика», также будет полезно студентам других направлений подготовки при изучении особенностей использования T-SQL для работы с базами данных.

©Московский энергетический институт, 2013

[Введение 5](#_Toc375561947)

[Лабораторная работа 1. Создание схемы БД. 6](#_Toc375561948)

[1. Основные понятия 6](#_Toc375561949)

[2. Последовательность выполнения лабораторной работы 10](#_Toc375561950)

[3. Методические указания 10](#_Toc375561951)

[4. Требования к оформлению отчета 10](#_Toc375561952)

[Контрольные вопросы 11](#_Toc375561953)

[Лабораторная работа 2. Обработка данных из БД. 11](#_Toc375561954)

[1. Основные понятия 11](#_Toc375561955)

[2. Последовательность выполнения лабораторной работы 12](#_Toc375561956)

[3. Требования к оформлению отчета 12](#_Toc375561957)

[Контрольные вопросы 12](#_Toc375561958)

[Лабораторная работа 3. Поиск данных. 13](#_Toc375561959)

[1. Основные понятия 13](#_Toc375561960)

[2. Последовательность выполнения лабораторной работы 14](#_Toc375561961)

[3. Методические указания 16](#_Toc375561962)

[4. Требования к оформлению отчета 16](#_Toc375561963)

[Контрольные вопросы 16](#_Toc375561964)

[Лабораторная работа 4. Применение агрегатных функций 16](#_Toc375561965)

[1. Основные понятия 16](#_Toc375561966)

[2. Последовательность выполнения лабораторной работы 17](#_Toc375561967)

[3. Требования к оформлению отчета 18](#_Toc375561968)

[Контрольные вопросы 18](#_Toc375561969)

[Лабораторная работа 5. Реализация сложных запросов 18](#_Toc375561970)

[1. Основные понятия 18](#_Toc375561971)

[2. Последовательность выполнения лабораторной работы 20](#_Toc375561972)

[3. Методические указания 21](#_Toc375561973)

[4. Требования к оформлению отчета 21](#_Toc375561974)

[Контрольные вопросы 22](#_Toc375561975)

[Лабораторная работа 6. Разработка хранимых процедур 22](#_Toc375561976)

[1. Основные понятия 22](#_Toc375561977)

[2. Последовательность выполнения лабораторной работы 24](#_Toc375561978)

[3. Требования к оформлению отчета 24](#_Toc375561979)

[Контрольные вопросы 24](#_Toc375561980)

[Лабораторная работа 7. Реализация декларативных правил целостности 25](#_Toc375561981)

[1. Основные понятия 25](#_Toc375561982)

[2. Последовательность выполнения лабораторной работы 25](#_Toc375561983)

[3. Требования к оформлению отчета 26](#_Toc375561984)

[Контрольные вопросы 26](#_Toc375561985)

[Лабораторная работа 8. Реализация динамических правил целостности 27](#_Toc375561986)

[1. Основные понятия 27](#_Toc375561987)

[2. Последовательность выполнения лабораторной работы 28](#_Toc375561988)

[3. Требования к оформлению отчета 29](#_Toc375561989)

[Контрольные вопросы 29](#_Toc375561990)

[ПРИЛОЖЕНИЕ I. 30](#_Toc375561991)

# Введение

Разработка баз данных (БД) с использованием серверных СУБД, таких как Oracle, Microsoft SQL Server, MySql, предполагает использование мощного и простого языка, который позволяет выполнять все необходимые операции с БД её пользователям и администраторам. Таким языком для работы с реляционными БД стал язык SQL (Structured Query Language), достоинствам которого является наличие международных стандартов. В настоящее время действует стандарт языка, принятый в 2003 году (SQL:2003) с небольшими модификациями, внесёнными позже в 2008 г. Этот стандарт поддерживается практически всеми разработчиками современных коммерческих реляционных СУБД, таких как Oracle, Microsoft SQL Server, MySql, Sybase и др. SQL с самого начала разрабатывался как полный язык для работы с БД, т.е. он поддерживает выполнение любых действий по работе с БД.

В составе SQL выделяются следующие компоненты:

* ***SQL-DDL*** (Data Definition Language) – язык определения объектов БД и ограничений целостности. К ним относятся команды создания и удаления баз данных; создания, изменения и удаления таблиц; управления пользователями и т.д.
* ***SQL-DML*** (Data Manipulation Language) – язык манипулирования данными: добавление, изменение, удаление и извлечение данных, управления транзакциями
* Операторы программирования
* Средства администрирования

В настоящее время существует большое количество реализаций языка, которые имеют специфические отличия, подчиняясь, тем не менее, единому стандарту. При выполнении лабораторных работ используется реализация SQL в Microsoft SQL Server – Transact-SQL (T-SQL). Для этого используется компонент Database Engine — основная служба для хранения, обработки и защиты данных Microsoft SQL Server. Компонент Database Engine обеспечивает управляемый доступ и быструю обработку транзакций, достаточную для большинства приложений по обработке данных в организации. Он используется для создания реляционных БД, для оперативной обработки транзакций или оперативной аналитической обработки данных. Сборник включает 9 лабораторных работ, которые предусматривают работу с ***SQL-DDL*** для создания объектов БД: таблиц, индексов, хранимых процедур и триггеров; ***SQL-DML*** для разработки запросов на обработку данных, средства среды **SQL Server Management Studio** для решения задач администрирования БД. Сборник предназначен для студентов, обучающихся по направлению «Прикладная математика и информатика. Он может быть полезен студентам других направлений подготовки при изучении дисциплин, связанных с использованием Microsoft SQL Server .

# Лабораторная работа 1. Создание схемы БД.

1x1Цель работы – приобрести навыки использования оператора Create Table и Insert.

## 1. Основные понятия

Оператор Create Table используется для создания новых таблиц базы данных. В нем определяется состав столбцов таблицы, задается их описание, и определяются правила целостности на уровне столбца и таблицы. Обязательными описателями являются имя таблицы, имя столбца, тип данных столбца. Синтаксис оператора приведен ниже.

***CREATE TABLE*** *имя\_таблицы*

***(***

***{****имя\_столбца тип\_данных* ***[ NOT NULL] [ [PRIMARY KEY |***

***UNIQUE]***

***[DEFAULT <****значение****>]***

***[IDENTITY [(****начальное\_значение****)]]***

***[FOREIGN KEY***

***REFERENCES*** *имя\_род\_таблицы*

***[ (****имя\_столбца\_род\_таблицы* ***) ]***

***[ CHECK (<****условие* ***> ) ] [,...n]***

***[ON UPDATE {CASCADE | NO ACTION } ]***

***[ON DELETE {CASCADE | NO ACTION } ]***

***}***

***);***

Типы данных, хранящихся в столбце, определяются на основе разрешенных типов в данной реализации СУБД. Типы данных в SQL Server объединены в следующие категории:

* точные числа,
* символьные строки в Юникоде,
* приблизительные числа,
* двоичные данные,
* дата и время,
* символьные строки,
* дополнительные типы данных.

В зависимости от параметров хранения, некоторые типы данных в SQL Server относятся к следующим группам:

* типы данных больших значений: varchar(max), nvarchar(max) и varbinary(max)
* типы данных больших объектов: text, ntext, image, varchar(max), nvarchar(max), varbinary(max) и xml

Описание наиболее употребимых типов данных T-SQL приведены в табл.1, характеристики числовых типов приведены в табл. 2. Полное описание типов данных можно найти на сайте разработчика ***technet.microsoft.com/ru-ru/library.***

Таблица 1.

|  |  |
| --- | --- |
| **Тип** | **Описание** |
| ***binary [ ( n ) ]*** | Хранит двоичные данные фиксированной длины размером в n байт, где n — значение от 1 до 8000. Размер хранения составляет n байт. |
| ***varbinary [ ( n | max) ]*** | Хранит двоичные данные переменной длины n, n может иметь значение от 1 до 8000; max означает максимальную длину хранения, которая составляет 2^31-1 байт. Размер хранения — это фактическая длина введенных данных плюс 2 байта. Он представляет собой простые битовые потоки. |
| ***bit*** | Целочисленный тип данных, который может принимать значения 1, 0 или NULL. |
| ***char [ ( n ) ]*** | Строковые данные фиксированной длины не в Юникоде. Аргумент n определяет длину строки, он должен быть значением от 1 до 8000. Размер при хранении составляет n байт. |
| ***cursor*** | Тип данных для переменных или выходных параметров хранимых процедур, которые содержат ссылку на курсор. |
| ***date*** | Описывает дату. |
| ***datetime*** | Определяет дату, включающую время дня с долями секунды в 24-часовом формате |
| ***decimal[ (p[ ,s] )]*** | Хранит числовые данные с фиксированными точностью и масштабом. При использовании максимальной точности числа могут принимать значения в диапазоне от -10^38+1 до 10^38-1.  p (точность) определяет максимальное количество десятичных разрядов числа (как слева, так и справа от десятичной запятой). Точность должна принимать значение от 1 до 38. По умолчанию для точности принимается значение 18.  s (масштаб) - максимальное количество десятичных разрядов числа справа от десятичной запятой. Масштаб может принимать значение от 0 до p. Масштаб может быть указан только совместно с точностью. По умолчанию масштаб принимает значение 0; поэтому 0 <= s <= p. |
| ***float*** *[* ***(****n****)*** *]* | Используется для числовых данных с плавающей запятой. n- задает точность. |
| ***int, bigint, smallint*** и ***tinyint*** | Определяет точные числовые данные, использующие целые значения |
| ***money*** | Определяет данные, представляющие денежные (валютные) значения |
| ***nchar [ ( n ) ]*** | Строковые данные фиксированной длины в формате Юникод. Аргумент n определяет длину строки, он должен быть значением от 1 до 4000. Размер при хранении в два раза больше n байт. |
| ***ntext*** | Этот тип данных представляет данные в формате Юникод переменной длины с максимальной длиной строки 2^30 – 1 (1 073 741 823) символов. Объем занимаемого этим типом пространства (в байтах) в два раза превышает введенную длину строки. |
| ***numeric[ (p[ ,s] )]*** | Аналогично типу decimal |
| ***nvarchar [ ( n | max ) ]*** | Строковые данные переменной длины в формате Юникод. Аргумент n определяет длину строки, он может быть значением в диапазоне от 1 до 4000. max указывает, что максимальный размер для хранения равен 2^31-1 байт (2 ГБ). |
| ***real*** | Определяет числовые данные с плавающей запятой |
| ***smalldatetime*** | Определяет дату, сочетающуюся с временем дня. Время представлено в 24-часовом формате с секундами, всегда равными нулю (:00), без долей секунд |
| ***text*** | Этот тип данных представляет данные, отличные от данных Юникода с переменной длиной строки, представленные с использованием кодовой страницы сервера. Максимальная длина строки — 2^31 – 1 (2 147 483 647) символов. |
| ***time*** | Определяет время дня без учета часового пояса в 24-часовом формате. |
| ***varbinary [ ( n | max) ]*** | Двоичные данные переменной длины. n могжет иметь значение от 1 до 8000; max означает максимальную длину хранения, которая составляет 2^31-1 байт. Размер хранения — это фактическая длина введенных данных плюс 2 байта. |
| ***varchar [ ( n | max ) ]*** | Строковые данные переменной длины не в формате Юникод. Аргумент n определяет длину строки, он может быть значением в диапазоне от 1 до 8000. max указывает, что максимальный размер для хранения равен 2^31-1 байт (2 ГБ). Размер хранения (в байтах) — это значение фактической длины введенных данных плюс 2 байта. |
| ***xml (Transact-SQL)*** | Тип данных, в котором хранятся XML-данные. Можно хранить экземпляры xml в столбце либо в переменной типа xml. |

Табл.2

|  |  |  |
| --- | --- | --- |
| **Тип данных** | **Диапазон** | **Объем памяти** |
| ***bigint*** | от -2^63 (-9 223 372 036 854 775 808) до 2^63-1 (9 223 372 036 854 775 807) | 8 байт |
| ***int*** | от -2^31 (-2 147 483 648) до 2^31-1 (2 147 483 647) | 4 байта |
| ***smallint*** | от -2^15 (-32 768) до 2^15-1 (32 767) | 2 байта |
| ***tinyint*** | от 0 до 255 | 1 байт |
| ***float*** | - 1,79E+308 — -2,23E-308, 0 и 2,23E-308 — 1,79E+308 | Зависит от значения n: при n= **1÷24 -** 4 байта  при n= **25÷53 – 8 байтов** |
| ***real*** | - 3,40E + 38 — -1,18E - 38, 0 и 1,18E - 38 — 3,40E + 38 | 4 байта |
| ***money*** | От -922 337 203 685 477,5808 до 922 337 203 685 477,5807 | 8 байт |
| ***smallmoney*** | От -214 748,3648 до 214 748,3647 | 4 байта |

Описание типов данных даты и времени T-SQL определены в табл. 3.

Таблица 3.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Тип данных** | **Формат** | **Диапазон** | **Точность** | **Объем памяти (в байтах)** | **Определяемая пользователем точность в долях секунды** |
| [***time***](http://msdn.microsoft.com/ru-ru/library/bb677243(v=sql.105).aspx) | чч:мм:сс[.ннннннн] | От 00:00:00.0000000 до 23:59:59.9999999 | 100 наносекунд | От 3 до 5 | Да |
| [***date***](http://msdn.microsoft.com/ru-ru/library/bb630352(v=sql.105).aspx) | ГГГГ-ММ-ДД | От 0001-01-01 до 9999-12-31 | 1 день | 3 | Нет |
| [***smalldatetime***](http://msdn.microsoft.com/ru-ru/library/ms182418(v=sql.105).aspx) | ГГГГ-ММ-ДД чч:мм:сс | От 01.01.1900 до 06.06.2079 | 1 минута | 4 | Нет |
| [***datetime***](http://msdn.microsoft.com/ru-ru/library/ms187819(v=sql.105).aspx) | ГГГГ-ММ-ДД чч:мм:сс[.ннн] | От 01.01.1753 до 9999-12-31 | 0,00333 секунда | 8 | Нет |

Для изменения структуры таблицы и описания её полей используется оператор ALTER TABLE, который позволяет добавлять и удалять столбцы, создавать и уничтожать индексы, переименовывать столбцы и саму таблицу:

***ALTER TABLE*** *имя\_таблицы*

***{***

***[ALTER COLUMN*** *имя\_столбца {новый\_тип\_данных* ***[NULL | NOT NULL ]}]***

***|***

***ADD { [****имя\_столбца тип\_данных] | имя\_столбца* ***AS*** *выражение* ***} [,...n]***

***|***

***DROP {COLUMN*** *имя\_столбца****}[,...n]***

***};***

Для удаления описания таблицы используется оператор

***DROP TABLE*** *имя\_таблицы* ***[RESTRICT | CASCADE***];

## 2. Последовательность выполнения лабораторной работы

1. Запустить программу SQL Server Management Studio, выполнить регистрацию и соединение с БД.
2. Средствами оператора Create Table создать описание таблиц БД, схема которой приведена в Приложении 1.
3. Используя оператор ALTER TABLE, изменить описание выбранного столбца таблицы.
4. Используя оператор ALTER TABLE, добавить описание нового столбца таблицы.
5. Используя оператор ALTER TABLE, удалить описание добавленного столбца таблицы.
6. Удалить описание таблицы (по указанию преподавателя).
7. Восстановить описание удаленной таблицы, используя встроенные средства SQL Management Studio.
8. Оформить отчет по лабораторной работе.

## 3. Методические указания

При определении структуры таблицы не учитывать ограничения целостности на значения полей.

## 4. Требования к оформлению отчета

Отчет по лабораторной работе должен включать следующие разделы:

1. Титульный лист.
2. Схема БД
3. Операторы описания структуры всех таблиц БД
4. Операторы изменения структуры таблицы
5. Оператор удаления описания таблицы.
6. Скриншоты работы со средствами SQL Management Studio.
7. Выводы по работе.
8. Ответы на контрольные вопросы.

## Контрольные вопросы

1. Определите назначение оператора Create Table.
2. Какие описатели полей в операторе Create Table являются обязательными?
3. Какие встроенные типы данных вам известны?
4. Почему следует избегать использования типов ntext, text и image при создании БД?
5. Для чего используется оператор Drop Table?
6. Определите назначение оператора Alter Table.

# Лабораторная работа 2. Обработка данных из БД.

Цель работы – приобрести навыки использования операторов Insert, Delete, Update для заполнения БД.

## 1. Основные понятия

Для добавления данных в таблицу используется оператор

***INSERT INTO <****имя\_таблицы****> [ (****имя\_столбца [,...n****] ) ]***

***VALUES (****значение[,...n****]);***

Он позволяет добавить одну строку в таблицу данных. Список столбцов задает имена столбцов, которым будут присвоены значения в добавляемых записях, указанных после ключевого слова **VALUES** . Список может быть опущен. Тогда должны быть определены значения всех столбцов столбцы таблицы (кроме объявленных как счетчик). Порядок вводимых значений в этом случае должен совпадать с порядком описания столбцов при создании таблицы. Если в команде **INSERT** указывается конкретный список имен столбцов, то любые пропущенные в нем столбцы должны быть определены при создании таблицы с описателями **NULL** или **DEFAULT**.

При задании значений данных при вводе может быть использованы функции преобразования типов. В Т-SQL определены универсальные функции **CONVERT** и **CAST**, которые преобразовывают значения одного типа в значения другого типа (если такие изменения возможны). Правила использования функций:

***CONVERT (****тип\_данных[(длина)], выражение [,стиль])*

***CAST (****выражение* ***AS*** *тип\_данных****)***

Аргумент ***стиль*** позволяет управлять стилем представления значений следующих типов данных*:* дата/время*,* денежный или нецелочисленный.

Изменения данных в таблице выполняет оператор

***UPDATE*** *имя\_таблицы* ***SET*** *имя\_столбца****=*** *<выраже ние>[,...n]* ***[WHERE*** *<условие\_отбора****>]***

В предложении ***SET*** указываются имена одного и более столбцов, данные в которых необходимо изменить. Поэтому одним оператором можно изменить сразу несколько столбцов.

Для удаления строк таблицы применяется оператор

***DELETE FROM <****имя\_таблицы****>***

***[WHERE <****условие\_отбора****>];***

Он позволяет удалить несколько строк из таблицы, которые удовлетворяют условию отбора. Если опустить предложение ***WHERE***, из таблицы будут удалены все записи*,* но сама таблица сохранится.

## 2. Последовательность выполнения лабораторной работы

1. Запустить программу SQL Server Management Studio, выполнить регистрацию и соединение с БД.
2. Средствами оператора Insert добавить данные в созданные таблицы.
3. Используя команду UPDATE, изменить значение выбранного столбца таблицы.
4. Используя оператор DELETE, удалить строки из таблицы (по указанию преподавателя).
5. Проверить заполненные данные, используя встроенные средства SQL Management Studio.
6. Оформить отчет по лабораторной работе.

## 3. Требования к оформлению отчета

Отчет по лабораторной работе должен включать следующие разделы:

1. Титульный лист.
2. Схема БД
3. Исходные данные для заполнения таблиц.
4. Операторы работы с данными БД
5. Скриншоты работы со средствами SQL Management Studio, отражающие содержимое таблиц.
6. Выводы по работе.
7. Ответы на контрольные вопросы

## Контрольные вопросы

1. Для чего используется оператор Insert?
2. Какие описатели полей в операторе Create Table являются обязательными?
3. Какие встроенные типы данных вам известны?
4. Почему следует избегать использования типов ntext, text и image при создании БД?
5. Для чего используется оператор Drop Table?
6. Определите назначение оператора Alter Table.
7. Какое соответствие должно быть между списком полей и списком значений в операторе Insert?

# Лабораторная работа 3. Поиск данных.

Цель работы – приобрести навыки использования оператора Select при поиске данных из одной таблицы

## 1. Основные понятия

Операция поиска данных в БД является наиболее часто используемой. Запрос на поиск данных выполняется в масштабе наборов данных, а не отдельных записей, поэтому один и тот же запрос может быть реализован несколькими способами, однако время выполнения таких запросов может существенно отличаться. Важно помнить, что оператор SELECT не изменяет данные в БД, а только производит их выборку в соответствии с заданными критериями.

Оператор SELECTимеет следующий формат:

***SELECT [****предикат* ***]***

***{ \* | [****имя\_столбца* ***[AS*** *новое\_имя****] ] }*** *[,...n]*

***FROM*** *имя\_таблицы* ***[ [AS]*** *псевдоним****]*** *[,...n]*

***[WHERE <****условие\_отбора****>]***

***[GROUP BY*** *имя\_столбца [,...n****]]***

***[HAVING <****критерии выбора групп****>]***

***[ORDER BY*** *имя\_столбца [,...n]* ***];***

Ключевое слово WHERE позволяет определить условие поиска, определяющее те строки, которые должны быть выбраны при выполнении запроса. Условие отбора представляет собой логическое выражение, в котором могут быть использованы имена полей, логические операции ***AND, OR*** и ***NOT***, операции сравнения, встроенные функции. Набор операций, используемых при задании условия отбора, приведен в табл. 4

Таблица 4.

|  |  |
| --- | --- |
| **Оператор** | **Результат** |
| ***ALL*** | ***TRUE,*** если весь набор сравнений дает результат ***TRUE*** |
| ***AND*** | ***TRUE***, если оба булевых выражения дают результат ***TRUE*** |
| ***ANY*** | ***TRUE***, если хотя бы одно сравнение из набора дает результат ***TRUE*** |
| ***BETWEEN*** | ***TRUE***, если операнд находится внутри диапазона |
| ***EXISTS*** | ***TRUE***, если подзапрос возвращает хотя бы одну строку |
| ***IN*** | ***TRUE***, если операнд равен одному выражению из списка или одной или нескольким строкам, возвращаемым подзапросом |
| ***LIKE*** | ***TRUE***, если операнд совпадает с шаблоном |
| ***NOT*** | Логическое отрицание |
| ***OR*** | ***TRUE***, если любое булево выражение равно ***TRUE*** |
| ***SOME*** | ***TRUE***, если несколько сравнений из набора дают результат ***TRUE*** |

Ключевое слово ***GROUP BY*** задает правило группировки выбранных данных. Группирование данных – это размещение данных в столбцах с повторяющимися значениями в определенном логическом порядке.

Ключевое слово ***HAVING*** дает дополнительную возможность профильтровать» отобранные записи. При этом надо учитывать следующие отличия применения условий, определенных после ключевых слов WHERE и HAVING:

* WHERE накладывает ограничения на строки, HAVING – на группы;
* предложение WHERE отсеивает строки до группировки, а предложение HAVING – после;
* в условии поиска WHERE нельзя задавать агрегатные функции;
* в большинстве систем элементы предложения HAVING должны включаться в список выбора. На предложение WHERE подобное ограничение не распространяется.

Ключевое слово ***ORDER BY*** задаетсортировку данных выходного набора в заданной последовательности. Сортировка может выполняться по нескольким полям, в этом случае они перечисляются за ключевым словом ***ORDER BY*** через запятую. По умолчанию реализуется сортировка по возрастанию, она задается ключевым словом ***ASC***. Для выполнения сортировки в обратной последовательности необходимо указать ключевое слово ***DESC***.

## 2. Последовательность выполнения лабораторной работы

1. Запустить программу SQL Server Management Studio, выполнить регистрацию и соединение с БД.
2. Добавить с помощью оператора INSERT данные в таблицы БД для выполнения поиска по заданным в п.3 условиям.
3. Выполнить оператор SELECT для поиска данных из одной таблицы в соответствии с заданием
   1. Составить список названий курсов с указанием их стоимости, которые имеют тип «начальный».
   2. Составить список названий курсов, у которых стоимость менее 1500 руб., и с которые имеют тип «начальный». Список отсортировать по названию курсов.
   3. Составить список названий курсов, у которых стоимость более1500 руб., и с которые имеют тип, отличный от «начальный».
   4. Составить список названий курсов с указанием их стоимости, в названии которых есть слово ‘SQL’. В результирующий список включить название курса, количество часов, тип курса, его стоимость. Список отсортировать по стоимости.
   5. Составить список слушателей из заданной организации. Список должен содержать ФИО слушателя, контактные данные, размер скидки.
   6. Составить список слушателей с фамилиями, начинающимися на букву К. Список должен содержать ФИО слушателя, контактные данные, название организации. Список отсортировать по названию организации.
   7. Составить список слушателей-женщин с именем, начинающимися на букву А. Список должен содержать ФИО слушателя, контактные данные, название организации. Список отсортировать по названию организации.
   8. Составить список слушателей заданной организации только женского пола и с именами, не заканчивающимися на букву «я» или «ь» .
   9. Составить список слушателей, которые имеют скидку на обучение в диапазоне от 3 до 1. Список должен содержать ФИО слушателя, контактные данные, название организации. Список отсортировать по фамилии слушателя.
   10. Составить список преподавателей, у которых стоимость часа работы лежит в диапазоне от 600 руб. до 900 руб. Список должен содержать ФИО преподавателя, контактные данные, степень. Список отсортировать по фамилии преподавателя.
   11. Составить список преподавателей, не имеющих степени и имеющих стоимость часа более 1000 руб. Список должен содержать ФИО преподавателя, контактные данные, стоимость часа работы. Список отсортировать по стоимости часа.
   12. Составить список преподавателей только женского пола и с фамилиями, начинающимися на букву Б или В.
   13. Составить список преподавателей, имеющих степень и со стоимостью часа менее 900 руб. Список должен содержать ФИО преподавателя, контактные данные, стоимость часа работы, степень. Список отсортировать по фамилии преподавателя.
   14. Составить список преподавателей мужского пола, не имеющих степени, Имя которых начинается на букву С. Список должен содержать ФИО преподавателя, контактные данные, степень, стоимость часа работы. Список отсортировать по степени и фамилии преподавателя.
   15. Составить список преподавателей, имеющих степень, Фамилия которых начинается на букву М, и имеющих стоимость часа более 1000 руб. Список должен содержать ФИО преподавателя, контактные данные, стоимость часа работы. Список отсортировать по степени и стоимости часа.
4. Оформить отчет по лабораторной работе.

## 3. Методические указания

Если запрос допускает различные методы выборки данных из нескольких таблиц, обосновать используемый метод.

## 4. Требования к оформлению отчета

Отчет по лабораторной работе должен включать следующие разделы:

1. Титульный лист.
2. Схема БД.
3. Текст запроса и соответствующий ему оператор выборки данных.
4. Обоснование выбранного метода реализации запроса.
5. Скриншоты результатов выполнения запросов в БД.
6. Выводы по работе.
7. Ответы на контрольные вопросы.

## Контрольные вопросы

1. Как исключить повторяющиеся при выводе записи?
2. Какие символы могут использоваться в шаблоне?
3. Как задать сортировку данных по убыванию?
4. В чем отличие использования ключевых слов WHERE и HAVING?
5. Определите результат применения операции BETWEEN.
6. Какая операция проверяет принадлежность значения некоторому множеству?

# Лабораторная работа 4. Применение агрегатных функций

Цель работы – приобрести навыки использования агрегатных функций и группировки при реализации запросов к БД.

## Основные понятия

С помощью агрегатных (итоговых) функций в SQL-запросе можно получить статистические данные, полученные на основе множества отобранных значений. В T-SQL определены следующие основные агрегатные функции:

***Count (Выражение)*** – определяет количество записей в выходном наборе SQL-запроса;

***Min/Max (Выражение)*** – определяют наименьшее и наибольшее из множества значений в некотором поле запроса;

***Avg (Выражение)*** – эта функция позволяет рассчитать среднее для множества значений, хранящихся в определенном поле записей, отобранных запросом.

***Sum (Выражение)*** – вычисляет сумму множества значений, содержащихся в определенном поле записей, отобранных запросом.

Чаще всего в качестве выражениявыступают имена столбцов. Выражение может вычисляться и по значениям нескольких таблиц. Агрегатные функции оперируют со значениями в единственном столбце таблицы или с арифметическим выражением и возвращают единственное значение. Функции ***COUNT***, ***MIN*** и ***MAX*** применимы как к числовым, так и к нечисловым полям. Функции ***SUM*** и ***AVG*** могут использоваться только в случае числовых полей. При вычислении результатов любых функций сначала исключаются все пустые значения. После этого требуемая операция применяется только к оставшимся конкретным значениям столбца. Исключением является функция ***COUNT(\*),*** котораявозвращаетобщее количество строк результирующей таблицы. – особый случай использования функции

Агрегатные функции могут использоваться только в списке предложения ***SELECT*** и в составе предложения ***HAVING***.

## Последовательность выполнения лабораторной работы

1. Запустить программу SQL Server Management Studio, выполнить регистрацию и соединение с БД.
2. Реализовать запросы к БД:
   1. Составить отчет по количеству учебных курсов каждого типа. Отчет должен начинаться заголовком и цифрой, определяющей ­общее количество курсов. Далее следует таблица, включающая перечень типов учебных курсов с цифрами, не равными нулю.
   2. Составить отчет по количеству слушателей, приехавших из разных городов. Отчет должен начинаться с соответствующего заголовка и цифры - общего количества слушателей. Далее следует таблица, включающая перечень городов с циф­рами, не равными нулю.
   3. Составить отчет преподавателям учебных курсов. Отчет должен начинаться с заголовка и цифры - общего количества преподавателей. Далее таблица, включающая перечень учебных степеней и количество преподавателей с заданной степенью с цифрами, не равными нулю.
   4. Составить отчет по стоимости часа преподавателей учебных курсов. Отчет должен начинаться с заголовка и цифры - общего количества преподавателей. Далее таблица, включающая стоимость часа и количество преподавателей (с цифрами, не равными нулю).
   5. Составить отчет по длительности курсов. Отчет должен начинаться с заголовка и цифры - общего количества курсов. Далее таблица, включающая перечень продолжительности курса и их количество (цифрами, не равными нулю).
   6. Составить отчет по курсам с учетом их типа. Отчет должен начинаться с заголовка и цифры - общего количества курсов, далее таблица, включающая тип курса и максимальную стоимость для этого типа.
   7. Составить отчет по курсам с учетом их типа. Отчет должен начинаться с заголовка и цифры - общего количества курсов, далее таблица, включающая тип курса, количество и среднюю стоимость для этого типа.
   8. Составить отчет преподавателям учебных курсов. Отчет должен начинаться с заголовка и цифры - общего количества преподавателей. Далее таблица, включающая перечень учебных степеней, минимальную и максимальную стоимость часа для преподавателей с данной степенью.
   9. Составить отчет по средней стоимости часа преподавателей учебных курсов. Отчет должен включать перечень учебных степеней и среднюю стоимость часа для преподавателей с данной степенью.
   10. Составить отчет о размере скидок на обучение. Отчет должен включать общее количество слушателей. Далее таблица с указанием размера скидки и числа слушателей, имеющих эту скидку.
3. Оформить отчет по лабораторной работе.

## Требования к оформлению отчета

Отчет по лабораторной работе должен включать следующие разделы:

1. Титульный лист.
2. Схема БД.
3. Текст запроса и соответствующий ему оператор выборки данных.
4. Скриншоты результатов выполнения запросов в БД.
5. Выводы по работе.
6. Ответы на контрольные вопросы.

Ответы на контрольные вопросы

## Контрольные вопросы

1. Для чего используются агрегатные функции?
2. Перечислите агрегатные функции, которые используются при работе с числовыми полями.
3. Как исключить суммирование повторяющихся значений?
4. В каких случаях используется группировка выводимых записей?
5. Какова особенность использования опции GROUP BY совместно с агрегатными функциями?

# Лабораторная работа 5. Реализация сложных запросов

Цель работы – приобрести навыки использования оператора Select при поиске данных из нескольких взаимосвязанных таблиц.

## Основные понятия

Как правило, извлекаемые из БД данные, содержатся в разных таблицах. При составлении запросов на выборку данных из нескольких таблиц можно использовать различные возможности T-SQL:

* команду соединения
* запросы с подзапросами

В составе языка T-SQL существует несколько команд, реализующих соединение: внутреннее соединение (INNER JOIN), внешнее левое соединение (LEFT JOIN), внешнее правое соединение (RIGHT JOIN), внешнее полное внешнее соединение (FULL JOIN)

Результатом внутреннего соединения, реализующего операцию естественного соединения реляционной алгебры, являются строки соединяемых таблиц, значения которых по соединяемым столбцам совпадают. При внешнем соединении в результирующую таблицу помещаются не только строки с совпадающими значениями, но и строки, для которых таких значений не нашлось.

Команда join используется внутри опции FROM и имеет вид:

[ { INNER | { { LEFT | RIGHT | FULL }    JOIN ON *условие\_соединения*

Применение запросов с подзапросами связано с теми случаями, когда невозможно решить поставленную задачу путем одного запроса. Если условие соединения указывает столбцы, их имена и типы данных могут не совпадать, однако, если типы данных не совпадают, столбцы должны либо быть совместимыми, либо иметь типы, которые SQL Server может неявно преобразовать. Если типы данных не могут быть преобразованы неявно, условие должно проводить явное преобразование типа данных при помощи функции CONVERT.

Подзапрос– это запрос, содержащийся в выражении ключевого слова ***WHERE*** другого запроса с целью дополнительных ограничений на выводимые данные. Подзапросы называют также вложенными запросами*.*

Синтаксис оператора Select с подзапросом:

SELECT *имя\_столбца*

FROM *таблица*

WHERE *предикат* (SELECT *имя\_\_столбца*

FROM *таблица*

WHERE *условие*);

Подзапрос можно также использовать в выражении ключевых слов в ***WHERE*** или ***HAVING*** внешних операторов выбора ***SELECT***, вставки ***INSERT,*** обновления ***UPDATE*** или удаления ***DELETE***.

Существуют два основных типа подзапросов:

подзапросы-выражения или скалярные подзапросы, они возвращают единственное значение;

 предикатные подзапросы, они возвращают список значений.

В тех случаях, когда результатом выполнения подзапроса является множество значений, для использования полученных в подзапросе значений, в основном операторе SELECT используются операции, которые применяются к множеству: предикаты ***IN, NOT IN, ALL*** и ***ANY (SOME), EXISTS*** (***NOT EXISTS***). Фактически любой запрос, который выражается через ***IN***, может быть альтернативным образом сформулирован также с помощью ***EXISTS***. Однако обратное высказывание несправедливо.

## Последовательность выполнения лабораторной работы

1. Запустить программу SQL Server Management Studio, выполнить регистрацию и соединение с БД.
2. Выполнить оператор SELECT для поиска данных из нескольких таблиц в соответствии с заданием.
   1. Составить список слушателей заданного курса. Список включает ФИО слушателя, его контактные данные, название курса, его тип. Список отсортировать по фамилии слушателя
   2. Составить список слушателей курсов, стоимость которых менее 1500 руб., и с которые имеют тип «начальный». Список отсортировать по названию курсов и фамилии слушателя.
   3. Составить список контрактов для слушателей, фамилия которых начинается на букву К. Список должен содержать ФИО слушателя, название курсов, на которые он заключал контракт, стоимость каждого курса. Список отсортировать по стоимости курса.
   4. Составить список слушателей, с которыми занимается заданный преподаватель. В результирующий список включить название курса, количество часов, тип курса, ФИО слушателя, его контактные данные. Список отсортировать по названию курса и ФИО слушателя.
   5. Составить список курсов, на которых обучались слушателей из заданной организации. Список должен содержать ФИО слушателя, контактные данные, название курса, тип курса, количество часов. Список отсортировать по названию курса и ФИО слушателя.
   6. Составить список слушателей с фамилиями, начинающимися на букву К, которые обучались на курсах, имеющих тип «начальный». Список должен содержать ФИО слушателя, контактные данные, название организации, название курса, его стоимость Список отсортировать данные по названию организации и ФИО слушателя.
   7. Составить список курсов, на которые отведено максимальное количество часов. Список должен содержать название, тип, стоимость, количество часов. Список отсортировать названию курса.
   8. Составить список курсов, стоимость которых выше средней. Список должен содержать название курса, тип, стоимость и количество часов.
   9. Составить список слушателей, которые имеют максимальную скидку на обучение. Список должен содержать ФИО слушателя, контактные данные, название организации. Список отсортировать данные по фамилии.
   10. Составить список преподавателей, у которых стоимость часа работы выше средней. Список должен содержать ФИО преподавателя, контактные данные, степень. Список отсортировать данные по фамилии преподавателя.
   11. Составить список слушателей, которые обучаются на курсах с максимальной стоимостью. Список должен содержать ФИО слушателя, контактные данные, название курса, его стоимость. Список отсортировать по названию курса и ФИО слушателя.
   12. Составить список преподавателей, которые проводят занятия на курсах, в названии которых есть слово «SQL». Список должен содержать ФИО преподавателя, название курса, его тип и продолжительность. Данные отсортировать по ФИО преподавателя.
   13. Составить список преподавателей, имеющих степень и преподающих на курсах, длительность которых более 36 часов. Список должен содержать ФИО преподавателя, контактные данные, степень, название курса и его стоимость. Список отсортировать данные по фамилии преподавателя.
   14. Составить список преподавателей мужского пола, имеющих степень, стоимость часа работы которых меньше средней стоимости часа. Список должен содержать ФИО преподавателя, контактные данные, степень, стоимость часа работы. Список отсортировать по степени и фамилии преподавателя.
   15. Составить список преподавателей, имеющих степень и стоимость часа более 3000 руб., которые ведут занятия на курсах, имеющих тип «начальный» Список должен содержать ФИО преподавателя, контактные данные, стоимость часа работы, название курса и его стоимость. Список отсортировать по степени и стоимости часа.
3. Оформить отчет по лабораторной работе.

## Методические указания

Если запрос допускает различные методы выборки данных из нескольких таблиц, обосновать используемый метод.

## Требования к оформлению отчета

Отчет по лабораторной работе должен включать следующие разделы:

1. Титульный лист.
2. Схема БД.
3. Текст запроса и соответствующий ему оператор выборки данных.
4. Обоснование выбранного метода реализации запроса.
5. Скриншоты результатов выполнения запросов в БД.
6. Выводы по работе.
7. Ответы на контрольные вопросы.

## Контрольные вопросы

1. Для чего используется команда Join?
2. Назовите разновидности команды Join.
3. Что такое подзапрос?
4. Почему возникает необходимость создания подзапросов?
5. Что такое скалярный подзапрос?
6. Какие подзапросы возвращают множество значений?

# Лабораторная работа 6. Разработка хранимых процедур

Цель работы – приобрести навыки разработки пользовательских хранимых процедур.

## Основные понятия

Хранимая процедура – это набор операторов T-SQL, который компилируется системой SQL Server и сохраняется в виде отдельного объекта БД. Различают системные, пользовательские и временные хранимые процедуры. Хранимые процедуры выполняются непосредственно на сервере БД, что обеспечивает более высокое их быстродействие, чем выполнение тех же операций средствами клиента БД. В теле хранимой процедуре могут быть использованы операторы ЯМД и операторы, реализующие логику обработки (присваивания, логического ветвления и т.п.).

Создание новой и изменение имеющейся хранимой процедуры осуществляется с помощью следующей команды:

***{CREATE | ALTER } PROC[EDURE]*** *имя\_процедуры*

***[{@****имя\_параметра тип\_данных* ***} [VARYING ]***

***[=****значение\_по\_умолчанию****][OUTPUT] ][,...n]***

***[WITH { RECOMPILE | ENCRYPTION | RECOMPILE,***

***ENCRYPTION }]***

***[FOR REPLICATION]***

***AS***

*Тело процедуры****;***

Операторы, используемые для реализации логику обработки, приведены в табл.5.

Таблица 5.

|  |  |
| --- | --- |
| **Оператор** | **Описание** |
| ***BEGIN...END***. | Определяет блок операторов |
| ***GOTO label*** | Безусловный переход к метке ***label***. |
| ***IF...ELSE***. | Условный оператор |
| ***RETURN***. | Безусловный выход |
| ***RETURN ([*integer\_expression*])*** | Возвращаемое значение – это код возврата, причем часть |
| ***WHILE*** …***[BREAK]*** [***CONTINUE***] | Цикл с предусловием |
| ***...BREAK*** | Выход из цикла ***WHILE*** |
| ***...CONTINUE*** | Продолжение цикла ***WHILE*** |
| ***DECLARE*** | Объявление локальных переменных. |
| ***PRINT***. | Выдает заданное значение на экран |
| ***CASE*** | Позволяет выражениям принимать значение в зависимости от условий. |

Запрос к реляционной БД может возвращать несколько строк (записей) данных. Однако приложение может обрабатывать лишь одну запись. В этой ситуации используется концепция курсора, который представляет собой указатель на ряд строк. Операторы работы с курсором приведены в табл. 6.

Таблица 6.

|  |  |
| --- | --- |
| **Синтаксис оператора** | **Описание** |
| ***DECLARE*** *имя\_курсора* ***CURSOR [LOCAL | GLOBAL]***  ***[FORWARD\_ONLY | SCROLL]***  ***[STATIC | KEYSET | DYNAMIC | FAST\_FORWARD]***  ***[READ\_ONLY | SCROLL\_LOCKS | OPTIMISTIC]***  ***[TYPE\_WARNING]***  ***FOR*** *SELECT\_оператор*  ***[FOR UPDATE [OF*** *имя\_столбца[,...n]]]* | Описание курсора |
| ***OPEN*** *{{[****GLOBAL***]*имя\_курсора* } | Открыть курсор |
| ***FETCH [[NEXT | PRIOR | FIRST | LAST***  ***| ABSOLUTE*** *{номер\_строки*  *| @переменная\_номера\_строки}*  *|* ***RELATIVE*** *{номер\_строки |*  *@переменная\_номера\_строки}]*  ***FROM ]{{[GLOBAL*** *]имя\_курсора }|*  *@имя\_переменной\_курсора }*  *[****INTO*** *@имя\_переменной [,...n]]* | Выбрать данные |
| ***CLOSE*** *{имя\_курсора | имя\_переменной\_курсора}* | Закрыть курсор |
| ***DEALLOCATE*** *{ имя\_курсора |@имя\_переменной\_курсора }* | Освободить курсор |

## Последовательность выполнения лабораторной работы

1. Запустить программу SQL Server Management Studio, выполнить регистрацию и соединение с БД.
2. Реализовать в виде хранимой процедуры статистические запросы, из задания к лабораторной работе №5 (по заданию преподавателя)
3. Выполнить разработанную процедуру.
4. Разработать хранимую процедуру с входным и выходным параметрами для определения общей суммы контрактов заданного слушателя.
5. Выполнить разработанную процедуру.
6. Разработать процедуру по заданию преподавателя с использованием курсора (Приложение II).
7. Выполнить разработанную процедуру.
8. Оформить отчет по лабораторной работе.

## Требования к оформлению отчета

Отчет по лабораторной работе должен включать следующие разделы:

1. Титульный лист.
2. Задание на разработку хранимой процедуры.
3. Алгоритм реализации хранимой процедуры.
4. Операторы описания хранимых процедур.
5. Результаты выполнения хранимых процедур.
6. Выводы по работе.
7. Ответы на контрольные вопросы.

## Контрольные вопросы

1. Для чего используются хранимые процедуры?
2. Какие виды хранимых процедур вы знаете?
3. Как задать параметры хранимой процедуры?
4. Для чего используется курсор в хранимых процедурах?
5. Как определить курсор в программе?
6. Назовите команды работы с курсором.
7. Какие режимы навигации определены в операторе Fetch?
8. Какой оператор позволяет удалить курсор?

# Лабораторная работа 7. Реализация декларативных правил целостности

Цель работы – применение статических правил для контроля целостности данных

## Основные понятия

При обработке данных необходима гарантия сохранения целостности данных в базе, поэтому важным этапом проектирования РБД является обеспечение целостности базы данных. Правила целостности задаются в виде ограничений целостности, которые представляют собой это некоторое логическое утверждение. Оно может быть истинным или ложным в зависимости от состояния БД. Ограничение целостности – это набор определенных правил, которые устанавливают допустимость данных и связей между ними в любой момент времени. Ограничения могут применяться на уровне столбцов и на уровне таблиц. Ограничения на уровне столбцов объявляются при создании столбца и применимы только к нему. Ограничения на уровне таблиц объявляются независимо от определений столбцов и могут применяться к одному или нескольким столбцам таблицы. Ограничения целостности имеют приоритет над триггерами, правилами и значениями по умолчанию. При задании ограничений на уровне столбца можно использовать следующие виды ограничений:

* ограничение типа данных,
* ограничение первичного ключа - ***PRIMARY KEY***;
* ограничение внешнего ключа - ***FOREIGN KEY***;
* ограничение уникальности значений - ***UNIQUE***;
* ограничение значения ***NULL***;
* ограничение, задаваемое пользователем - ***CHECK***.

Ограничения, задаваемые пользователем на уровне столбца, определяются опцией ***CHECK:***

***CHECK (<логическое\_****условие* ***> ) ] [,…n]***

Ограничения на уровне таблицы задаются опцией ***CONSTRAINT:***

***CONSTRAINT [****имя\_ограничения] тип\_ограничения [(стол-*

*бец[,…])] [предикат]*

## Последовательность выполнения лабораторной работы

1. Запустить программу SQL Server Management Studio, выполнить регистрацию и соединение с БД.
2. Добавить в таблицу Course столбец Doc, имеющий следующее описание:

| **Имя столбца** | **Тип данных** | **комментарий** |
| --- | --- | --- |
| Doc | символьный | Вид документа, может принимать значение: удостоверение, диплом, сертификат, справка |

1. Определить правила целостности для поля Doc.
2. Изменить описание таблиц, добавив описание правил целостности на основе данных табл.3 Приложения I.
3. Проверить заданные правила, используя оператор INSERT.
4. Определить правило целостности для таблицы Course на основе следующего описания:

Поле Doc может принимать значение «справка» или «сертификат» для курсов длительностью не более 36 часов. Если длительность курса находится в диапазоне от 72 до 126 часов, то поле Doc может принимать значение «сертификат» или «удостоверение». Если длительность курса более 126 часов, то поле Doc может принимать значение «удостоверение» или «диплом».

1. Проверить заданное правило, используя команду UPDATE для поля Hur.
2. Сформулировать свое ограничение целостности для таблицы Tсhr и определить для него правило целостности.
3. Проверить заданное правило целостности, используя оператор INSERT.
4. Оформить отчет по лабораторной работе.

## Требования к оформлению отчета

Отчет по лабораторной работе должен включать следующие разделы:

1. Титульный лист.
2. Описание правил целостности.
3. Описание правил целостности на языке SQL.
4. Тестовые наборы данных для проверки правил целостности.
5. Результаты проверки правил целостности.
6. Выводы по работе.
7. Ответы на контрольные вопросы.

## Контрольные вопросы

1. Для чего используются статические правила целостности?
2. Определите назначение конструкций check и constraint.
3. Назовите ограничения использования оператора create rule.
4. Какие правила применяются для задания ограничений на значение атрибута?
5. Для чего используется описатель NOT NULL?
6. В чем важность задания ограничений целостности?

# Лабораторная работа 8. Реализация динамических правил целостности

Цель работы – приобрести навыки использования внешних ключей разработки триггеров для обеспечения целостности данных.

## Основные понятия

Для обеспечения целостности данных, содержащихся в разных таблицах, можно использовать ограничения FOREIGN KEY, задаваемые при описании структуры таблицы, и триггеры.

Ограничения FOREIGN KEY поддерживают связи между таблицами.

Внешний ключ применяется для принудительного установления связи между данными в двух таблицах. ***В столбцах, включенных во внешний ключ, могут использоваться только значения из столбцов первичного ключа***.

Ограничение FOREIGN KEY может ссылаться на столбцы в таблицах этой же БД или столбцы этой же самой таблицы. Ограничение FOREIGN KEY не обязательно должно быть связано с ограничением PRIMARY KEY в другой таблице. Кроме того, с помощью этого ограничения могут указываться столбцы ограничения UNIQUE в другой таблице. Ограничение FOREIGN KEY может содержать значения NULL.

Ограничение FOREIGN KEY обеспечивает целостность ссылок по следующим правилам: запрещается изменение данных в таблице первичного ключа, если такие изменения сделают недопустимой ссылку в таблице внешнего ключа. Если при попытке удалить строку в таблице первичного ключа или изменить значение этого ключа окажется, что удаленному или измененному значению первичного ключа соответствует значение в ограничении FOREIGN KEY в другой таблице, то действие выполнено не будет. Для успешного изменения или удаления строки с ограничением FOREIGN KEY можно использовать дополнительные предложения ON DELETE и ON UPDATE, определяющими порядок действий с помощью специальных параметров:

* NO ACTION: удаления (изменения) не произойдет и будет выведено сообщение об ошибке.
* CASCADE: все строки с внешними ключами, указывающими на удаленную строку, также будут удалены (изменены).
* SET NULL: всем строкам с внешними ключами, указывающими на удаленную (измененную) строку, присваивается значение NULL.
* SET DEFAULT: всем строкам с внешними ключами, указывающим на удаленную (измененную) строку, присваивается установленное для них значение по умолчанию.

Процедурные правила целостности позволяют проводить согласованные изменения данных, хранящихся во взаимосвязанных таблицах. Триггеры - это особый вид хранимых процедур, которые вызываются автоматически при возникновении ситуации, с которой связан триггер. Их удобно использовать в следующих случаях:

* для каскадных изменений в связанных таблицах БД (в большинстве случаев эти изменения можно более эффективно выполнить при помощи каскадных ограничений ссылочной целостности);
* для предотвращения случайных или неправильных операций INSERT, UPDATE и DELETE и реализации других более сложных ограничений, чем те, которые определены при помощи ограничения CHECK.   
  В отличие от ограничений CHECK, триггеры могут ссылаться на столбцы других таблиц;
* для оценки состояние таблицы до и после изменения данных и предпринять действия на основе этого различия.

Триггер не может иметь входных параметров и возвращать результат. В SQL Server существует два вида триггеров, которые различаются по способу его выполнения: **AFTER** и **INSTEAD OF**. Триггеры различают также по типу команд, на которые они реагируют. Существует три типа триггеров: **INSERT TRIGGER**, **UPDATE TRIGGER, DELETE TRIGGER.**  Триггер создается операцией

***CREATE TRIGGER*** *имя\_триггера*

***BEFORE | AFTER*** *<триггерное\_событие>*

***ON*** *<имя\_таблицы>*

***[REFERENCING***

*<список\_старых\_или\_новых\_псевдонимов>]*

*[****FOR EACH { ROW | STATEMENT}]***

*[****WHEN****(условие\_триггера)]*

*<тело\_триггера>*

В теле триггера могут быть использованы специальные таблицы: *inserted* и *deleted*. В них содержатся списки строк, которые будут вставлены или удалены по завершении транзакции. В случае нарушения ограничений целостности для прерывания выполнения триггера и отмены всех изменений, которые пытался выполнить пользователь, необходимо использовать команду ***ROLLBACK TRANSACTION.***

## Последовательность выполнения лабораторной работы

1. Запустить программу SQL Server Management Studio, выполнить регистрацию и соединение с БД.
2. Изменить описание таблицы Contract, описав поле Id\_Crs как внешний ключ, запрещающий изменение записей родительской таблицы.
3. Выполнить оператор удаления курса, с которым связана хотя бы одна запись таблицы Contract.
4. Написать триггер, который реализует каскадное изменение полей в связанных записях в таблицах Contract и Tchr при изменении кода преподавателя.
5. Проверить работоспособность триггера, выполнив команду изменения кода преподавателя в таблице Tchr.
6. Разработать триггер, изменяющий величину скидки для слушателя по следующему правилу:

Если стоимость всех контрактов слушателя лежит в диапазоне 50000÷70000 руб., то размер скидки – 5%; если стоимость всех контрактов слушателя лежит в диапазоне 70000÷80000 руб., то размер скидки – 6%; если стоимость всех контрактов слушателя лежит в диапазоне 80000÷100000 руб., то размер скидки – 10%; если стоимость всех контрактов слушателя превышает 100000 руб., то размер скидки устанавливается в 15%.

1. Проверить работоспособность триггера, добавив контракт для слушателя.
2. Оформить отчет по лабораторной работе.

## Требования к оформлению отчета

1. Титульный лист.
2. Описание правила целостности для каждого правила.
3. Алгоритм реализации правила целостности.
4. Описание правила целостности на языке SQL.
5. Тестовые наборы данных для проверки правила целостности.
6. Результаты проверки правил целостности.
7. Выводы по работе.
8. Ответы на контрольные вопросы.

## Контрольные вопросы

1. Определите назначение триггеров.
2. Для чего используются вспомогательные таблицы deleted и inserted?
3. В чем разница выполнения триггеров after и instead of?
4. Опишите алгоритм стандартных правил целостности, обеспечивающих каскадное обновление данных.
5. Что такое ссылочная целостность?
6. В чем суть применения триггеров для контроля целостности данных?
7. Как можно реализовывать ограничения целостности на «домен»?

## ПРИЛОЖЕНИЕ I.

**Схема БД и описание таблиц**



**Рис. 1. Схема БД**

**Состав и описание таблиц БД**

Таблица 1 определяет состав таблиц БД, в таблицах 2 - 5 приведены описания атрибутов каждой таблицы. В таблице 6 определены правила целостности для отдельных атрибутов.

**Таблица 1. Состав таблиц БД**

| **Имя таблицы** | **Описание таблицы** |
| --- | --- |
| Contract | договор на оказание образовательной услуги |
| Course | описание курса |
| Student | Сведения о слушателях |
| Tchr | Преподаватель |

Таблица 2. Описание столбцов таблицы ***Contract***

| **Имя столбца** | **Тип данных** | **первичный ключ** | **внешний ключ** | **Комментарий** |
| --- | --- | --- | --- | --- |
| Id\_Cont | Integer | Yes | No | Номер контракта |
| Id\_Stud | Integer | No | Yes | код слушателя в БД |
| Dte | Datetime | No | No | дата заключения |
| Id\_Crs | Integer | No | Yes | Идентификатор курса |

Таблица 2. Описание столбцов таблицы ***Course***

| **Имя столбца** | **Тип данных** | **первичный ключ** | **внешний ключ** | **Комментарий** |
| --- | --- | --- | --- | --- |
| Id\_Crs | Integer | Yes | No | Идентификатор курса |
| Title | varchar(20) | No | No | Название курса |
| DefCrs | varchar(20) | No | No | Кракое содержание курса (до 256 знаков) |
| Cost | Integer | No | No | стоимость курса (не менее 500 руб) |
| Id\_Tch | Integer | No | Yes | код препоадвателя |
| Hur | Integer | No | No | количество часов, не менее 16 часов |
| TypCrs | varchar(20) | No | No | тип курса, может принимать значения начальный, продвинутый, профессиональный |

Таблица 4. Описание столбцов таблицы ***Student***

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Имя столбца** | **Тип данных** | **первичный ключ** | **внешний ключ** | **Комментарий** |
| Id\_Stud | Integer | Yes | No | код слушателя в БД |
| Name\_stud | varchar(20) | No | No | Фамилия Имя отчество |
| Org | varchar(20) | No | No | Название организации |
| Cont | varchar(20) | No | No | Контактные данные |
| Cty | varchar(20) | No | No | город проживания |
| Cntr | varchar(20) | No | No | страна проживания |
| disCnt | Integer | No | No | размер скидки на обучение |
| Sx | Char | No | No | Пол, может принимать значение «м» и «ж» |

Таблица 5. Описание столбцов таблицы ***Tchr***

| **Имя столбца** | **Тип данных** | **первичный ключ** | **внешний ключ** | **Комментарий** |
| --- | --- | --- | --- | --- |
| Id\_Tch | Integer | Yes | No | код преподавателя |
| NameTch | char(18) | No | No | Фамилия, имя отчество |
| ContTch | varchar(20) | No | No | Контактные данные |
| Sal | Integer | No | No | Стоимость 1 часа работы |
| Dol | varchar(20) | No | No | степень |
| Sx\_Tc | Char | No | No | Пол: «м» или «ж» |
| Id\_Tch | Integer | Yes | No | код преподавателя |

Таблица 3. Правила целостности для атрибутов

| **Имя атрибута** | **Имя таблицы** | **Min** | **Max** |
| --- | --- | --- | --- |
| disCnt | Student | 0 | 15 |
| Sal | Tchr | 500 | 10000 |
| Hur | Course | 16 |  |

Учебное издание

Сидорова Наталья Петровна

**Основы T-SQL.**

**Сборник лабораторных работ**

Методическое пособие

по курсу «Базы данных»

для студентов, обучающихся по направлению Прикладная математика и информатика

Редактор издательства

Темплан издания МЭИ 2013 (), метод. Подписано к печати

Формат 60×84/16 Печать офсетная Физ. печ.л

Тираж 100 экз. Изд. № Заказ №

ЗАО «издательский дом МЭИ», 111250, Москва, ул. Красноказарменная, д. 14

Отпечатано в типографии ФГУП «НИИ «Геодезия», 141292, Московская область,

Г. Красноармейск, пр-т Испытателей, д.14.