# Лабораторная работа № 5

## Типы данных и встроенные функции

## Цель работы

* 1. Изучить основные типы данных.
  2. Изучить встроенные функции для работы со строками.
  3. Изучить встроенные функции для работы с числами.
  4. Изучить встроенные функции для работы с датами и временем.
  5. Изучить встроенные функции преобразования данных.
  6. Изучить CASE и IIF.

## Теоретическая часть

* 1. ***Типы данных***

На языке Transact-SQL используется множество различных *типов данных*. Всех их можно разделить на следующие группы:

*Числовые типы данных*:

* BIT (значение 0 или 1),
* TINYINT (от 0 до 255),
* SMALLINT (от -32768 до 32767)
* INT (от -2147483648 до 2147483647),
* BIGINT (от -9223372036854775808 до 9223372036854775807),
* DECIMAL (числа c фиксированной точностью),
* NUMERIC: (аналоги чен типу DECIMAL),
* SMALLMONEY (дробные значения от -214748.3648 до 214748.3647),
* MONEY (дробные значения от -922337203685477.5808 до 922337203685477.5807),
* FLOAT (от -1.79E+308 до 1.79E+308),
* REAL (числа от –340E+38 до 3.40E+38);

*Типы данных, представляющие дату и время:*

* DATE (дата от 01/01/0001 до 31/12/9999),
* TIME (время в диапазоне от 00:00:00.0000000 до 23:59:59.9999999),
* DATETIME (дата и время от 01/01/1753 до 31/12/9999),
* DATETIME2 (дата и время от 01/01/0001 00:00:00.0000000 до 31/12/9999 23:59:59.9999999),
* SMALLDATETIME (дата и время от 01/01/1900 до 06/06/2079),
* DATETIMEOFFSET (дата и время от 01/01/0001 до 31/12/9999);

*Строковые типы данных:*

* CHAR (фиксированная строка длиной от 1 до 8000 символов),
* VARCHAR (переменная строка длиной от 1 до 8000 символов),
* NCHAR (Unicode – фик сированная строка длиной от 1 до 4000 символов),
* NVARCHAR (Unicode – переменная строка длиной от 1 до 4000 символов),
* TEXT и NTEXT (устаревшие, не рекомендуется использовать);

*Бинарные типы данных:*

* BINARY (фиксированные бинарные данные от 1 до 8000 байт),
* VARBINARY (переменные бинарные данные от 1 до 8000 байт),
* IMAGE (устаревшая, не рекомендуется использовать);

*Другие типы данных:*

* UNIQUEIDENTIFIER (уникальный идентификатор GUID),
* TIMESTAMP (номер версии строки в таблице),
* CURSOR (набор строк таблицы),
* HIERARCHYID (позиция в иерархии),
* SQL\_VARIANT (данные любого типа),
* XML (документы или фрагменты XML),
* TABLE (таблица),
* GEOGRAPHY (географические данные, такие как широта и долгота),
* GEOMETRY (координаты на плоскости).

## Встроенные функции Transact-SQL

*Функции SQL* производят действия с данными и возвращают результат. *Встроенные функции* делятся на три основные группы:

1. *скалярные функции* – обрабатывают одиночное значение и возвращают одно значение. Их можно использовать везде, где допускается применение выражений.
2. *агрегатные функции* – используются для получения обобщающих значений. Они, в отличие от скалярных функций, оперируют значениями столбцов множества строк;
3. *– функции для списка значений*.

*Скалярные функции* бывают следующих категорий:

* *строковые функции* – выполняют определенные действия над строками и возвращают строковые или числовые значения;
* *числовые функции* – возвращают числовые значения на основании заданных в аргументе значений того же типа;
* *функции времени и даты* – выполняют различные действия над входными значениями времени и даты и возвращают строковое, числовое значение или значение в формате даты и времени;
* *функции преобразования типа*.

Список часто используемых *строковых функций*:

|  |  |
| --- | --- |
| LEN (строка) | возвращает количество символов в заданной строке |
| TRIM (строка)  TRIM ([символ FROM] строка) | удаляет символ пробела или другие заданные символы в начале и в конце строки. |
| LTRIM (строка) | удаляет начальные пробелы из заданной строки |
| RTRIM (строка) | удаляет конечные пробелы из заданной строки |
| CHARINDEX (подстрока, строка) CHARINDEX (подстрока, строка, начальная позиция) | возвращает индекс, по которому находится первое вхождение подстроки в строке. |
| PATINDEX ('%шаблон%', строка) | возвращает индекс, по которому находится первое вхождение определенного шаблона в строке |
| LEFT (строка, число) | возвращает с начала строки определенное количество символов |
| RIGHT (строка, число) | возвращает с конца строки определенное количество символов |
| SUBSTRING (строка, начальная позиция, длина) | возвращает подстроку заданной длиной, начиная с данной позиции |
| REPLACE (строка, подстрока, замена) | заменяет одну подстроку другой |
| REVERSE (строка) | переворачивает строку наоборот |
| CONCAT (строка1, строка2 [, строкаN ] ) | объединяет заданные строки в одну |
| LOWER (строка) | переводит строку в нижний регистр |
| UPPER (строка) | переводит строку в верхний регистр |
| SPACE (число) | возвращает заданное количество пробелов |
| REPLICATE (строка, число) | повторяет значение строки указанное число раз |
| STUFF (строка, начальная позиция, количество, замена) | удаляет указанное количество символов первой строки в начальной позиции и вставляет на их место замену. |

Список часто используемых *числовых функций*:

|  |  |
| --- | --- |
| ABS (число) | возвращает абсолютное значение числа |
| CEILING (число) | возвращает наименьшее целое, большее или равное заданного числа. |
| FLOOR (число) | возвращает наибольшее целое число, меньшее или равное заданного числа |
| POWER (число, степень) | возвращает значение указанного выражения, возведенное в заданную степень |
| RAND ([начальное значение]) | возвращает псевдослучайное значение от 0 до 1 |
| ROUND (число, точность) | возвращает число, округленное до указанной точности |
| SIGN (число) | возвращает положительное (+1), нулевое (0) или отрицательное (-1) значение, обозначающее знак заданного выражения |
| SQRT(число) | возвращает квадратный корень данного числа |
| SQUARE (число) | возвращает квадрат указанного числа |
| PI () | возвращает константное значение π |
| ACOS (число) | возвращает угол в радианах, косинус которого задан – арккосинус. |
| ASIN (число) | возвращает угол в радианах, синус которого задан – арксинус. |
| ATAN(число) | возвращает угол в радианах, тангенс которого задан – арктангенс. |
| COS(число) | возвращает косинус указанного угла в радианах. |
| SIN(число) | возвращает синус указанного угла в радианах. |
| TAN(число) | возвращает тангенс указанного угла в радианах. |
| COT(число) | возвращает котангенс указанного угла в радианах |
| DEGREES(число) | возвращает для значения угла в радианах соответствующее значение в градусах. |
| RADIANS(число) | возвращает для значения угла в градусах соответствующее значение в радианах |
| EXP(число) | возвращает экспонент заданного числа |
| LOG(число) | возвращает натуральный логарифм указанного числа |
| LOG(число, основа) | возвращает логарифм указанного числа |
| LOG10(число) | возвращает десятичный логарифм указанного числа |

Список часто используемых *функций времени и даты*:

|  |  |
| --- | --- |
| GETDATE() | возвращает текущую дату и время |
| CURRENT\_TIMEZONE() | возвращает имя часового пояса |
| GETUTCDATE() | возвращает текущую дату и время по Гринвичу (UTC/GMT) |
| DAY(дата) | возвращает день месяца указанной даты |
| MONTH(дата) | возвращает номер месяца указанной даты |
| YEAR(дата) | возвращает год указанной даты |
| DATEPART(часть, дата) | возвращает целое число, представляющее указанную часть заданной даты |
| DATENAME(часть, дата) | возвращает строку символов, представляющую указанную часть заданной даты |
| DATEADD(часть, число, дата) | добавляет указанное целое число со знаком к части входного значения даты, а затем возвращает это измененное значение |
| DATEDIFF(часть, начальная дата, конечная дата) | возвращает разницу как целое число со знаком между  частями заданных дат |
| EOMONTH(дата) | возвращает последний день месяца, заданной даты |

Для функций времени и даты используются следующие аргументы как часть даты и времени:

|  |  |
| --- | --- |
| ***Часть даты и времени*** | ***Сокращения*** |
| year | yy, yyyy |
| quarter | qq, q |
| month | mm, m |
| dayofyear | dy, y |
| day | dd, d |
| week | wk, ww |
| weekday | dw |
| hour | hh |
| minute | mi, n |
| second | ss, s |
| millisecond | ms |
| microsecond | mcs |
| nanosecond | ns |
| tzoffset | tz |
| iso\_week | isowk, isoww |

Список часто используемых *функций преобразования*:

|  |  |
| --- | --- |
| CAST(выражение AS тип) | преобразуют выражение в заданный тип |
| CONVERT(тип, выражение [, стиль]) |
| ASCII(строка) | возвращает код ASCII первого символа указанного  символьного выражения |
| UNICODE(строка) | возвращает код Юникод первого символа указанного  символьного выражения |
| CHAR(число) | возвращает символ ASCII с указанным кодом |
| NCHAR(число) | возвращает символ Юникода с указанным кодом |
| STR(число) | возвращает символьные данные, преобразованные из  числовых данных |

Список часто используемых *функций проверки значений*:

|  |  |
| --- | --- |
| ISDATE(выражение) | возвращает 1, если выражение имеет допустимое значение типа даты и времени, иначе возвращает значение 0 |
| ISNUMERIC(выражение) | возвращает 1, если выражение имеет допустимое значение числовой тип данных, иначе возвращает 0 |
| ISNULL(выражение, замена) | заменяет значение NULL указанным замещающим значением |
| COALESCE(выражение[,...n ]) | вычисляет аргументы по порядку и возвращает текущее значение первого выражения, изначально не вычисленного как NULL. |

Особое место среди встроенных скалярных функций языка SQL занимают функции вывода, которые являются разновидностью CASE-выражений. Функция CASE проверяет значе ние некоторого выражения, и в зависимости от результата проверки может возвращать тот или иной результат.

Выражение CASE имеет два формата:

* простое выражение CASE для определения результата сравнивает выражение с набо ром простых выражений;
* поисковое выражение CASE для определения результата вычисляет набор логических выражений.

Оба формата поддерживают дополнительный аргумент ELSE.

Функция IIF(условие, выражение\_если\_истина, выражение\_если\_ложь) – возвращает одно из двух значений в зависимости от того, принимает логическое выражение значение true или false.

## Практическая часть

Дана таблица ***Академики:***

|  |  |  |  |
| --- | --- | --- | --- |
| **ФИО** | **Дата\_рождения** | **Специализация** | **Год\_присвоения\_звания** |
| Аничков Николай Николаевич | 1885-11-03 | медицина | 1939 |
| Бартольд Василий Владимирович | 1869-11-15 | историк | 1913 |
| Белопольский Аристарх Аполлонович | 1854-07-13 | астрофизик | 1903 |
| Бородин Иван Парфеньевич | 1847-01-30 | ботаник | 1902 |
| Вальден Павел Иванович | 1863-07-26 | химик-технолог | 1910 |
| Вернадский Владимир Иванович | 1863-03-12 | геохимик | 1908 |
| Виноградов Павел Гаврилович | 1854-11-30 | историк | 1914 |
| Ипатьев Владимир Николаевич | 1867-11-21 | химик | 1916 |
| Истрин Василий Михайлович | 1865-02-22 | филолог | 1907 |
| Карпинский Александр Петрович | 1847-01-07 | геолог | 1889 |
| Коковцов Павел Константинович | 1861-07-01 | историк | 1906 |
| Курнаков Николай Семёнович | 1860-12-06 | химик | 1913 |
| Марр Николай Яковлевич | 1865-01-06 | лингвист | 1912 |
| Насонов Николай Викторович | 1855-02-26 | зоолог | 1906 |
| Ольденбург Сергей Фёдорович | 1863-09-26 | историк | 1903 |
| Павлов Иван Петрович | 1849-09-26 | физиолог | 1907 |
| Перетц Владимир Николаевич | 1870-01-31 | филолог | 1914 |
| Соболевский Алексей Иванович | 1857-01-07 | лингвист | 1900 |
| Стеклов Владимир Андреевич | 1864-01-09 | математик | 1912 |

***Пример 1:*** Вывести ФИО академиков и длину ФИО: SELECT

ФИО

,LEN(ФИО) AS Количество\_символов

FROM

Академики

***Пример 2:*** Вывести список академиков, убрать лишние пробелы в ФИО:

SELECT

TRIM(ФИО) AS ФИО

,Дата\_рождения

,Специализация

,Год\_присвоения\_звания

FROM

Академики

***Пример 3:*** Найти позиции буквы «о» в ФИО каждого академика. Вывести ФИО и по зицию:

SELECT

ФИО

,CHARINDEX('о',ФИО) AS Позиция\_о

FROM

Академики

***Пример 4:*** Вывести ФИО и первые три буквы специализации каждого академика:

SELECT

ФИО

,LEFT(Специализация, 3) AS Спец\_3

FROM

Академики

***Пример 5:*** Вывести ФИО и от второй до пятой буквы специализации каждого академика.

SELECT

ФИО

,SUBSTRING(Специализация, 2, 4) AS Спец\_2\_5

FROM

Академики

***Пример 6:*** Вывести список академиков, заменить специализацию «лингвист» на «язы ковед»:

SELECT

ФИО

,Дата\_рождения

,REPLACE(Специализация, 'лингвист', 'языковед') AS Спец

,Год\_присвоения\_звания

FROM

Академики

***Пример 7:*** Вывести список академиков, специализацию на верхнем регистре:

SELECT

ФИО

,Дата\_рождения

,UPPER(Специализация) AS Спец

,Год\_присвоения\_звания

FROM

Академики

***Пример 8:*** Вывести ФИО академиков в правильном и обратном виде:

SELECT

ФИО

,REVERSE(ФИО) AS ФИО\_Обр

FROM

Академики

Название

***Пример 9:*** Вывести каждую специализацию 4 раза в одной строке. Убрать дубликаты:

SELECT DISTINCT

REPLICATE(Специализация, 4) AS Спец\_4

FROM

Академики

***Пример 10:*** Вывести абсолютное значение тригонометрических функций на точке π:

SELECT

ABS(COS(PI())) AS Косинус\_Пи

,ABS(SIN(PI())) AS Синус\_Пи

,ABS(TAN(PI())) AS Тангенс\_Пи

,ABS(COT(PI())) AS КоТангенс\_Пи

***Пример 11:***Вывести число 132.456, округленное с точностью от 3 до -3:

SELECT

ROUND(123.456, 3) AS Окр3

,ROUND(123.456, 2) AS Окр2

,ROUND(123.456, 1) AS Окр1

,ROUND(123.456, 0) AS Окр0

,ROUND(123.456, -1) AS Окр\_1

,ROUND(123.456, -2) AS Окр\_2

,ROUND(123.456, -3) AS Окр\_3

***Пример 12:*** Вывести наименьшее целое число, которое больше или равно 123.456, и наибольшее целое число, которое меньше или равно 123.456:

SELECT

CEILING(123.456) AS Больше

,FLOOR(123.456) AS Меньше

***Пример 13:*** Вывести квадратный корень, квадрат и куб числа 25:

SELECT

SQRT(25) AS Корень

,SQUARE(25) AS Квадрат

,POWER(25, 3) AS Куб

***Пример 14:*** Вывести текущую дату и время:

SELECT

GETDATE() AS Сейчас

***Пример 15:*** Вывести день, месяц, год, час, минуту, секунду, номер квартала, номер недели, день года, день недели для текущей даты и времени:

SELECT

DAY(GETDATE()) AS День

,MONTH(GETDATE()) AS Месяц

,YEAR(GETDATE()) AS Год

,DATEPART(HOUR, GETDATE()) AS Час

,DATEPART(MINUTE, GETDATE()) AS Минута

,DATEPART(SECOND, GETDATE()) AS Секунда

,DATEPART(QUARTER, GETDATE()) AS Квартал

,DATEPART(WEEK, GETDATE()) AS Неделя

,DATEPART(DAYOFYEAR, GETDATE()) AS День\_года

,DATEPART(WEEKDAY, GETDATE()) AS День\_недели

***Пример 16:*** Вывести дату 100 дней назад от текущей:

SELECT

DATEADD(DAY, -100, GETDATE()) AS День\_100\_Назад

***Пример 17:*** Академик Игорь Евгеньевич Тамм родился 8 июля 1895 года. И.Е. Тамм скончался 12 апреля 1971 года. Вывести количество прожитых дней:

SELECT

DATEDIFF(DAY, '18950708', '19710412') AS Количество\_прожитых\_дней

***Пример 18:*** Вывести ФИО и время года рождения каждого академика:

SELECT

ФИО

, CASE MONTH(Дата\_рождения)

WHEN 3 THEN 'Весна'

WHEN 4 THEN 'Весна'

WHEN 5 THEN 'Весна'

WHEN 6 THEN 'Лето'

WHEN 7 THEN 'Лето'

WHEN 8 THEN 'Лето'

WHEN 9 THEN 'Осень'

WHEN 10 THEN 'Осень'

WHEN 11 THEN 'Осень'

ELSE 'Зима'

END AS Времени\_года

FROM Академики

***Пример 19:*** Вывести ФИО, дату рождения и знак зодиака каждого академика:

SELECT

ФИО

, Дата\_рождения

, CASE

WHEN (MONTH(Дата\_рождения)=3 AND DAY(Дата\_рождения) >= 21)

OR (MONTH(Дата\_рождения)=4 AND DAY(Дата\_рождения) <= 20) THEN 'Овен'

WHEN (MONTH(Дата\_рождения)=4 AND DAY(Дата\_рождения) >= 21)

OR (MONTH(Дата\_рождения)=5 AND DAY(Дата\_рождения) <= 21) THEN 'Телец'

WHEN (MONTH(Дата\_рождения)=5 AND DAY(Дата\_рождения) >= 22)

OR (MONTH(Дата\_рождения)=6 AND DAY(Дата\_рождения) <= 21) THEN 'Близнецы'

WHEN (MONTH(Дата\_рождения)=6 AND DAY(Дата\_рождения) >= 22)

OR (MONTH(Дата\_рождения)=7 AND DAY(Дата\_рождения) <= 22) THEN 'Рак'

WHEN (MONTH(Дата\_рождения)=7 AND DAY(Дата\_рождения) >= 23)

OR (MONTH(Дата\_рождения)=8 AND DAY(Дата\_рождения) <= 21) THEN 'Лев'

WHEN (MONTH(Дата\_рождения)=8 AND DAY(Дата\_рождения) >= 22)

OR (MONTH(Дата\_рождения)=9 AND DAY(Дата\_рождения) <= 23) THEN 'Дева'

WHEN (MONTH(Дата\_рождения)=9 AND DAY(Дата\_рождения) >= 24)

OR (MONTH(Дата\_рождения)=10 AND DAY(Дата\_рождения) <= 23) THEN 'Весы'

WHEN (MONTH(Дата\_рождения)=10 AND DAY(Дата\_рождения) >= 24)

OR (MONTH(Дата\_рождения)=11 AND DAY(Дата\_рождения) <= 22) THEN 'Скорпион'

WHEN (MONTH(Дата\_рождения)=11 AND DAY(Дата\_рождения) >= 23)

OR (MONTH(Дата\_рождения)=12 AND DAY(Дата\_рождения) <= 22) THEN 'Стрелец'

WHEN (MONTH(Дата\_рождения)=12 AND DAY(Дата\_рождения) >= 23)

OR (MONTH(Дата\_рождения)=1 AND DAY(Дата\_рождения) <= 20) THEN 'Козерог'

WHEN (MONTH(Дата\_рождения)=1 AND DAY(Дата\_рождения) >= 21)

OR (MONTH(Дата\_рождения)=2 AND DAY(Дата\_рождения) <= 19) THEN 'Водолей'

WHEN (MONTH(Дата\_рождения)=2 AND DAY(Дата\_рождения) >= 20)

OR (MONTH(Дата\_рождения)=3 AND DAY(Дата\_рождения) <= 20) THEN 'Рыбы'

END AS Знак\_зодиака

FROM Академики

***Пример 20:*** Вывести список академиков. Для каждого академика, в зависимости от возраста, при присвоении звания вывести «молодой» или «старый» в дополнительном столбце:

SELECT

ФИО

,Дата\_рождения

,Специализация

,Год\_присвоения\_звания

,IIF(Год\_присвоения\_звания - Year(Дата\_рождения) <= 45, 'Молодой','Старый')

AS Возраст\_при\_присвоении

FROM Академики

## Задание

* 1. Вывести список академиков, отсортированный по количеству символов в ФИО.
  2. Вывести список академиков, убрать лишние пробелы в ФИО.
  3. Найти позиции «ов» в ФИО каждого академика. Вывести ФИО и номер позиции.
  4. Вывести ФИО и последние две буквы специализации для каждого академика.
  5. Вывести список академиков, ФИО в формате Фамилия и Инициалы.
  6. Вывести список специализаций в правильном и обратном виде. Убрать дубликаты.
  7. Вывести свою фамилию в одной строке столько раз, сколько вам лет.
  8. Вывести абсолютное значение функций с точностью два знака после десятичной запятой.
  9. Вывести количество дней до конца семестра.
  10. Вывести количество месяцев от вашего рождения.
  11. Вывести ФИО и високосность года рождения каждого академика.
  12. Вывести список специализаций без повторений. Для каждой специализации выве сти «длинный» или «короткий», в зависимости от количества символов.