



# SQL

Базовый уровень

---

# Функции CAST и CONVERT

# Функции CAST и CONVERT

CAST ( expression AS data\_type [ ( length ) ] )

CONVERT ( data\_type [ ( length ) ] ,  
expression  
[ , style ] )

- *expression* – любое допустимое выражение.
- *data\_type* – целевой тип данных.
- *length* – длина целевого типа данных.
- *style* – целочисленное выражение, определяющее, как функция CONVERT преобразует параметр *expression*.

# Функции CAST и CONVERT

-- преобразование даты к числу

```
select cast(getdate() as numeric(8,2))
```

(No column name)

41721.88

--преобразование даты к строке

```
select cast(getdate() as varchar(30))
```

(No column name)

Mar 25 2014 9:09PM

--преобразование числа к формату

```
select cast('10.0005' as numeric(10,8))
```

(No column name)

10.00050000

--преобразование числа к формату

```
select cast(10.6496 as int);
```

(No column name)

10

# Функции CAST и CONVERT

--127 - гggг-мм-ддТчч:мм:сс.mmmП (без пробелов)

```
select convert(varchar(40), getdate(), 127)
```

(No column name)

2014-03-25T21:15:54.477

--113 - дд мес гggг чч:мм:сс:mmm (24-часовой формат)

```
select convert(varchar(40), getdate(), 113)
```

(No column name)

25 Mar 2014 21:16:10.223

--114 - чч:мм:сс:mmm (24-часовой формат)

```
select convert(varchar(40), getdate(), 114)
```

(No column name)

21:16:32.223

# Функции CAST и CONVERT

По умолчанию SQL Server интерпретирует двузначные значения года с пороговым значением 2049.

Год, обозначенный двумя цифрами 49, интерпретируется как 2049, а год, обозначенный двумя цифрами 50, интерпретируется как 1950.

select **cast**('01.01.49' as date);



(No column name)

2049-01-01

select **cast**('01.01.50' as date);



(No column name)

1950-01-01

# Пример

- Вывести информацию о продукте (ProductID) и средней цене продукта ("AVG price = " + средняя цена продукта) из таблицы SalesOrderDetail.

SalesOrderDetail (Sales)	
🔑	SalesOrderID
🔑	SalesOrderDetailID
	CarrierTrackingNumber
	OrderQty
	ProductID
	SpecialOfferID
	UnitPrice
	UnitPriceDiscount
	LineTotal
	rowguid
	ModifiedDate

```
select s.ProductID,  
       'AVG price =' + avg(s.UnitPrice) as avg  
from Sales.SalesOrderDetail s  
group by s.ProductID
```



*Cannot convert a char value to money. The char value has incorrect syntax.*

```
select s.ProductID,  
       'AVG price =' +  
       cast(avg(s.UnitPrice) as varchar(25))  
       as avg  
from Sales.SalesOrderDetail s  
group by s.ProductID
```



# Модификация данных



## ВСТАВКА ДАННЫХ

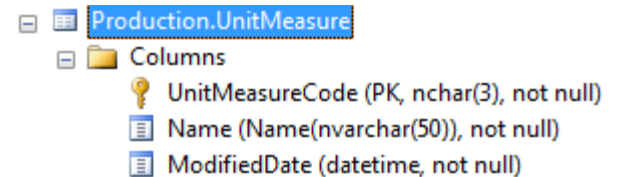
Вставка новой записи в таблицу осуществляется оператором Insert, который в общем виде имеет формат:


```
INSERT INTO <имя таблицы>  
            (<список полей>)  
VALUES (<список значений>)
```

В списке перечисляются только те поля, значения которых известны, остальные могут опускаться. Для пропущенных полей значения берутся по умолчанию, или остаются пустыми.

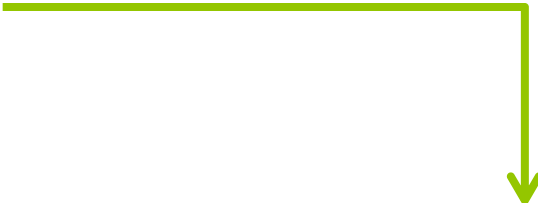
# ВСТАВКА ДАННЫХ

```
INSERT INTO Production.UnitMeasure  
VALUES ('FT2', 'Square Feet ', GETDATE()),  
          ('Y', 'Yards', GETDATE()),  
          ('Y3', 'Cubic Yards', GETDATE());
```



 Messages

(3 row(s) affected)



UnitMeasureCode	Name	ModifiedDate
FT2	Square Feet	2014-03-25 14:53:40.380
Y	Yards	2014-03-25 14:53:40.380
Y3	Cubic Yards	2014-03-25 14:53:40.380

# INSERT - SELECT

**Insert into** HumanResources.EmployeeDepartmentHistory  
(BusinessEntityID,DepartmentID,ShiftID,StartDate)

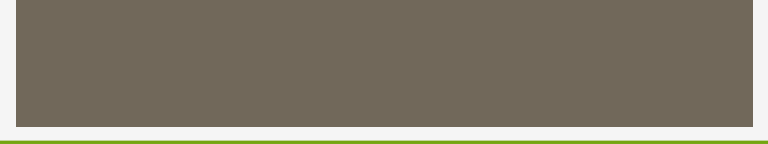
**select** edh.BusinessEntityID  
    , edh.DepartmentID  
    , s.ShiftID  
    , getdate()

from HumanResources.EmployeeDepartmentHistory edh,  
      HumanResources.Shift s

where s.Name like 'test'

    and edh.ShiftID in (select ShiftID  
                          from HumanResources.Shift  
                          where name like 'day')

    and edh.DepartmentID = 16

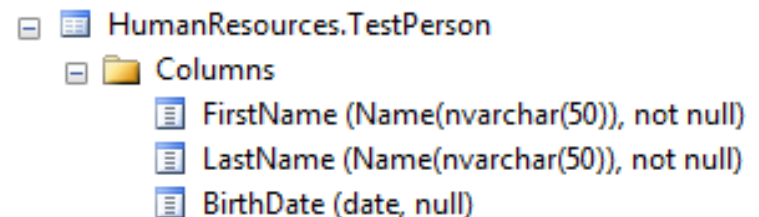


HumanResources.EmployeeDepartmentHistory
Columns
BusinessEntityID (PK, FK, int, not null)
DepartmentID (PK, FK, smallint, not null)
ShiftID (PK, FK, tinyint, not null)
StartDate (PK, date, not null)
EndDate (date, null)
ModifiedDate (datetime, not null)

# INSERT - INTO

- Данная конструкция позволяет:
  - Создать таблицу указанной структуры
  - Заполнить её данными

```
select top(15) p.FirstName,  
              p.LastName,  
              cast ('01.01.2000' as date) BirthDate  
into HumanResources.TestPerson  
from Person.Person p
```



Создание таблицы

# Задания

1. С помощью оператора INSERT – SELECT создать таблицу HumanResources.MyEmployee с такими полями:

- BusinessEntityID
- FirstName,
- LastName,
- Title,
- Gender,
- MaritalStatus,
- BirthDate,
- JobTitle
- EmailAddress

Person.Person

HumanResources.Employee

Person.EmailAddress

2. Заполнить её 15 строками (произвольными)
3. С помощью оператора INSERT добавить в таблицу информацию о себе.

# Редактирование данных

Редактирование записей осуществляется оператором UPDATE:

```
UPDATE <имя таблицы>  
SET <поле1 = выражение1>,  
      <поле2 = выражение2>,  
...  
WHERE <условие>
```

# Редактирование данных

**UPDATE** Production.Product

**SET** ListPrice = ListPrice \* 2

**WHERE** ProductID IN

( SELECT ProductID

FROM Purchasing.ProductVendor

WHERE BusinessEntityID = 1540);

# Задание

- Изменить значение поля EmailAddress в созданной таблице MyEmployee:
  - для мужчин изменить адрес на домен «@EPAM.COM»
  - для незамужних дам изменить адрес на домен «@HotLove.COM»

**Помощь:** для вырезания части e-mail использовать  
substring(EmailAddress,  
0,  
CHARINDEX('@', EmailAddress)+1)



## Удаление данных

Удаление записей осуществляется оператором DELETE или оператором TRUNCATE TABLE:

**DELETE**

**FROM** <имя таблицы>

**WHERE** <условие>

# Удаление данных

**DELETE**

**FROM** Production.ProductCostHistory

**WHERE** StandardCost > 1000.00;



Messages

(50 row(s) affected)

|

# Удаление данных

*--удалить первые 2.5% записей*

```
DELETE TOP (2.5) PERCENT  
FROM Production.ProductInventory;
```

*--удалить историю квот по условию*

```
DELETE  
FROM Sales.SalesPersonQuotaHistory  
WHERE BusinessEntityID IN  
      (SELECT BusinessEntityID  
       FROM Sales.SalesPerson  
       WHERE SalesYTD > 25000000.00);
```

# Удаление данных

--дочерние записи - ошибка удаления

delete

from HumanResources.Shift

where name like '%test%'

*The DELETE statement conflicted with the REFERENCE constraint "FK\_EmployeeDepartmentHistory\_Shift\_ShiftID". The conflict occurred in database "AdventureWorks2012", table "HumanResources.EmployeeDepartmentHistory", column 'ShiftID'.*

*The statement has been terminated.*

# Очистка таблицы TRUNCATE

Особенности TRUNCATE TABLE:

- DDL-операция (невосстанавливаемая операция);
- быстро выполняется (ощутимо на больших таблицах), поскольку строки удаляются как результат укорачивания структуры хранения данных таблицы в БД, а не поштучно, как при DELETE.

# Очистка таблицы TRUNCATE

```
select count(*)  
from HumanResources.JobCandidateTest
```

(No column name)

13

--удаление данных

```
truncate table HumanResources.JobCandidateTest
```

*Command(s) completed successfully.*

-- проверка заполненности

```
HumanResources.JobCandidateTest
```

```
select count(*)
```

```
from HumanResources.JobCandidateTest
```

(No column name)

0

# Задания

- В созданной таблице MyEmployee удалить записи о сотрудниках, которые за прошлый год продали < 1000 (Sales.SalesPerson, поле SalesLastYear)