доц. д-р Цветанка Георгиева-Трифонова

ИЗПОЛЗВАНЕ НА ИЗРАЗИ В SQL

Съдържание

- Изрази за дата/час и интервални изрази
- Функции за работа със символни низове и изрази за конвертиране на типа данни
- Условни (CASE) изрази

Използване на изрази в SQL

- Изразите могат да се използват в:
  - SELECT списъка;
  - в условието на WHERE;
  - в условието на HAVING.

ИЗПОЛЗВАНЕ НА ИЗРАЗИ В SQL (2)

- аритметични оператори: +, -, *, / , % (остатък при целочислено деление).
- Пример:

```
SELECT StudentNumber, SubjectID,
       2*Evaluation AS Score
FROM Evaluations
```

МАТЕМАТИЧЕСКИ ФУНКЦИИ

- ABS(*numeric_expr*) – абсолютната стойност;
- ACOS(*float_expr*) – ъгъла в радиани, чийто косинус е зададения числов израз *float_expr*;
- ASIN(*float_expr*) – ъгъла в радиани, чийто синус е зададения числов израз *float_expr*;
- ATAN(*float_expr*) – ъгъла в радиани, чийто тангенс е зададения числов израз *float_expr*;
- COS(*float_expr*) – тригонометричния косинус на даден ъгъл (в радиани), зададен в числовия израз *float_expr*;
- SIN(*float_expr*) – тригонометричния синус на даден ъгъл (в радиани), зададен в числов израз;
- TAN(*float_expr*) – тригонометричния тангенс на даден ъгъл (в радиани), зададен в числов израз;
- COT(*float_expr*) – тригонометричния котангенс на даден ъгъл (в радиани), зададен в числов израз;

МАТЕМАТИЧЕСКИ ФУНКЦИИ (2)

- DEGREES(*numeric_expr*) – градусите, превърнати от радиани;
- RADIANS(*numeric_expr*) – радианите, превърнати от градуси;
- PI() връща константата 3.1415...;
- POWER(*numeric_expr*, y) – стойността на числовия израз *numeric_expr* на степен y;
- SQUARE(*float_expr*) – квадрата на зададения числов израз;
- SQRT(*float_expr*) – квадратния корен;
- EXP(*float_expr*) – експоненциалната стойност;
- LOG(*float_expr*) – натуралния логаритъм;
- LOG10(*float_expr*) – логаритъма при основа 10;
- SIGN(*numeric_expr*) – знака на числовия израз, т.е. -1 – при отрицателна стойност; 0 – при стойност 0; 1 – при положителна стойност;

- CEILING(*numeric_expr*) – цяло число, най-малко от всички стойности, по-големи или равни на аргумента;
- FLOOR(*numeric_expr*) – цяло число, най-голямо от всички стойности, по-малки или равни на аргумента;
- ROUND(*numeric_expr*, *precision* [, *truncate*])
  - ако точността *precision* е **положително** число, числовият израз се закръгля до толкова позиции след десетичната запетая, колкото са посочени в точността;
  - ако точността е **отрицателно** число, числовият израз се закръгля до толкова позиции вляво от запетаята, колкото са посочени в точността;
  - числото *truncate* определя дали числовият израз се **закръгля** или **отрязва** – ако липсва или е 0, се закръгля; ако е различно от 0, се отрязват толкова знаци, колкото е точността.
    Например ROUND(70.46, 0, 1) връща цялата част на числото, т.е. 70; ROUND(75.81, -1, 0) връща 80; ROUND(75.81, -1, 1) връща 70;
- RAND([*seed*]) – случайно генерирана стойност – число с плаваща запетая между 0 и 1; аргументът *seed* е начална стойност за алгоритмите, генериращи случайни числа (от целочислен тип).

ФУНКЦИИ ЗА ДАТА И ЧАС

- `DATEADD(interval, number, datetime)` – дата, добавяйки интервал към зададена дата.
  - *interval* може да има една от следните стойности:

година	year, yy, yyyy
тримесечие	quarter, q, qq
месец	month, m,mm
седмица	week, wk, ww
ден	day, d, dd
пореден ден от годината	dayofyear, dy, y
пореден ден от седмицата	weekday, dw, w
час	hour, hh
минута	minute, mi, n
секунда	second, s, ss
милисекунда	millisecond, ms

ФУНКЦИИ ЗА ДАТА И ЧАС (2)

- *number* е брой интервали (дни, месеци и т.н.):
  - при отрицателна стойност се извършва връщане назад от датата в *datetime*;
  - при положителна – добавяне към нея.
- *datetime* е датата, към която ще се добавя или връща интервал;
  - ако е константа, трябва да се огради с апострофи;
  - може да е колона от таблица, оградена при необходимост с [].
- Пример

```
DATEADD(day, 30, '1/1/2010')
```

- връща датата 31 януари 2010 г.

ФУНКЦИИ ЗА ДАТА И ЧАС (3)

- `DATEDIFF(`*`interval, datetime1, datetime2`*`)`
  - цяло число, означаващо разликата между две дати. Възможните стойности на параметъра *interval* са същите като при DATEADD.
  - Резултатът от функцията е цяло число, което е:
    - положително, ако *datetime1* е преди *datetime2*;
    - отрицателно, ако *datetime2* е преди *datetime1*;
    - 0, ако двете дати съвпадат по отношение на зададения интервал.
- Пример

```
DATEDIFF(year, '31/12/2009', '1/1/2010')
```

ФУНКЦИИ ЗА ДАТА И ЧАС (4)

- `DATEPART(`*`interval, datetime`*`)`
  - цяло число, представящо зададената чрез *interval* част от дата;
- `DATENAME(`*`interval, datetime`*`)`
  - низ, представящ зададената чрез *interval* част от дата;
- `GETDATE()`
  - текущата дата и час.
- Примери

```
DATEPART(month, '06/10/2009')
DATENAME(month, '06/10/2009')
DATEPART(month, GETDATE())
```

ФУНКЦИИ ЗА РАБОТА СЪС СИМВОЛНИ НИЗОВЕ

- LEN(*string*) – броя на символите в низа;
- LEFT(*string*, *number*) – *number* на брой символи от низа *string*, започвайки от началото на низа;
- RIGHT(*string*, *number*) – *number* на брой символи от низа *string*, започвайки от края на низа;
- SUBSTRING(*string*, *start*, *length*) – част от низа *string*, като взема *length* символа от *start* позиция;
- UPPER(*string*) – символния низ, като заменя малките букви в него с главни;
- LOWER(*string*) – символния низ, като заменя главните букви в него с малки;

ФУНКЦИИ ЗА РАБОТА СЪС СИМВОЛНИ НИЗОВЕ (2)

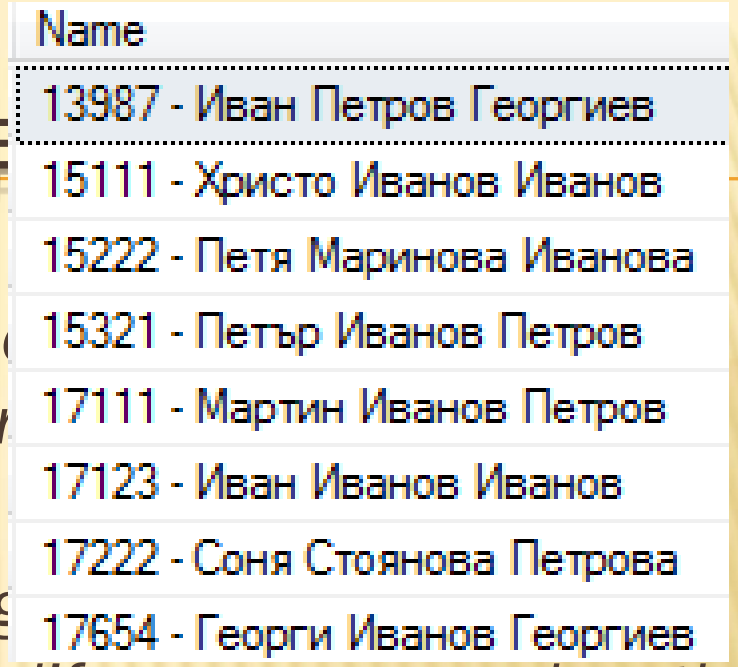
- CHARINDEX(*str_to_find*, *str_to_search* [, *start_position*]) – началната позиция на *str_to_find* в *str_to_search*;
- PATINDEX(*pattern*, *string*) – позицията в низа *string*, в който за първи път се среща съответствие на шаблона *pattern* или 0, ако шаблонът не е намерен;
- LTRIM(*string*) – низа, като отстранява евентуалните интервали, с които започва;
- RTRIM(*string*) – низа, като отстранява евентуалните интервали, с които завършва;
- REVERSE(*string*) – низа, като разполага знаците, от които се състои в обратен ред;
- REPLICATE(*string*, *number*) – низ, в който посоченият първи аргумент е повторен толкова пъти, колкото е *number*;
- SPACE(*number*) – низ от *number* на брой интервала.

ФУНКЦИИ ЗА РАБОТА СЪС СИМВОЛНИ НИЗОВЕ (3)

- REPLACE(*str_to_search*, *str_to_find*, *str_to_replace_with*) замества всички срещания на *str_to_find* в *str_to_search* със *str_to_replace_with*;
- STUFF(*string_to_modify*, *start*, *length*, *string_to_insert*) трансформира низа *string_to_modify*, като изтрива *length* на брой символи от позиция *start*, вмъквайки *string_to_insert* в *string_to_modify* от позиция *start*.
- *Конкатениране* на низове (+), например:

```
SELECT
   StudentNumber + ' - ' + StudentName AS Name
FROM Students
```

ФУНКЦИИ ЗА РАБОТА СЪС СИМВ



- REPLACE(*str_to_search*, *str_to_find*, *str_to_replace_with*) – замества всички срещания на *str_to_find* в *str_to_search* със *str_to_replace_with*;
- STUFF(*string_to_modify*, *start*, *length*, *string_to_insert*) – трансформира низа *string_to_modify*, като изтрива *length* на брой символи от позиция *start*, вмъквайки *string_to_insert* в *string_to_modify* от позиция *start*.
- *Конкатениране* на низове (+), например:

```
SELECT
    StudentNumber + ' - ' + StudentName AS Name
FROM Students
```

ФУНКЦИИ ЗА РАБОТА СЪС СИМВОЛНИ НИЗОВЕ – ПРИМЕРИ

```
SELECT LEFT(EGN, 2) AS Y,
       SUBSTRING(EGN, 3, 2) AS M,
       SUBSTRING(EGN, 5, 2) AS D,
       SUBSTRING(EGN, 5, 2) + '.' +
       SUBSTRING(EGN, 3, 2) + '.' +
       LEFT(EGN, 2) AS B
FROM Employees
```

	Y	M	D	B
1	71	12	11	11.12.71
2	80	01	11	11.01.80
3	77	07	01	01.07.77
4	65	09	10	10.09.65
5	80	01	11	11.01.80
6	83	03	12	12.03.83

ИЗРАЗИ ЗА КОНВЕРТИРАНЕ НА ТИПА ДАННИ

- CAST(*original_expression* AS *desired_datatype*) конвертира даден израз от определен тип данни в израз от друг тип данни;
- Пример

```
SELECT st.StudentNumber + ', ' +
       s.SubjectName + ', ' +
       CAST(e.Evaluation AS char(4)) AS
  Results
FROM Subjects s
INNER JOIN Evaluations e
   ON s.SubjectID = e.SubjectID
INNER JOIN Students st
  ON st.StudentNumber = e.StudentNumber
```

ИЗРАЗИ ЗА КОНВЕРТИРАНЕ НА Т

- CAST(*original_expression* AS *desir*
  конвертира даден израз от опред
  от друг тип данни;
- Пример

```
SELECT st.StudentNumber + ',
       s.SubjectName + ', '
       CAST(e.Evaluation AS char(4)) AS
  Results
FROM Subjects s
INNER JOIN Evaluations e
   ON s.SubjectID = e.SubjectID
INNER JOIN Students st
  ON st.StudentNumber = e.StudentNumber
```

Results
13987, Програмиране I, 6.00
13987, Програмиране II, 3.00
13987, Бази от данни, 5.00
13987, Информационни системи, 5.00
13987, XML, 6.00
13987, DHTML, 4.00
15111, Програмиране I, 4.00
15111, Програмиране II, 5.00
15321, Програмиране I, 4.00
15321, Програмиране II, 6.00

ИЗРАЗИ ЗА КОНВЕРТИРАНЕ НА ТИПА ДАННИ (2)

- STR(*float_expression* [, *length* [, *decimal*]]) – низ, представящ число с плаваща точка;
  - *Length* – общия брой знаци, които да съставят получения низ, включвайки (.) и (-), ако съществуват;
  - *decimal* – брой на разрешените позиции след десетичната точка.
- Пример

```
SELECT st.StudentNumber + ', ' +
       s.SubjectName + ', ' +
       STR(e.Evaluation, 4, 2) AS Results
FROM Subjects s
INNER JOIN Evaluations e
   ON s.SubjectID = e.SubjectID
INNER JOIN Students st
   ON st.StudentNumber = e.StudentNumber
```

ИЗРАЗИ ЗА КОНВЕРТИРАНЕ НА Т

- STR(*float_expression* [, *length* [, *d*
  представящ число с плаваща точ
  - *Length* – общия брой знаци, които
    низ, включвайки (.) и (-), ако същес
  - *decimal* – брой на разрешените по
    точка.
- Пример

```
SELECT st.StudentNumber + ', ' +
       s.SubjectName + ', ' +
       STR(e.Evaluation, 4, 2) AS Results
FROM Subjects s
INNER JOIN Evaluations e
   ON s.SubjectID = e.SubjectID
INNER JOIN Students st
   ON st.StudentNumber = e.StudentNumber
```

Results
13987, Програмиране I, 6.00
13987, Програмиране II, 3.00
13987, Бази от данни, 5.00
13987, Информационни системи, 5.00
13987, XML, 6.00
13987, DHTML, 4.00
15111, Програмиране I, 4.00
15111, Програмиране II, 5.00
15321, Програмиране I, 4.00
15321, Програмиране II, 6.00

ИЗРАЗИ ЗА КОНВЕРТИРАНЕ НА ТИПА ДАННИ (3)

- CONVERT(*datatype*, *expression* [, *style*]) преобразува израза *expression* от един тип данни в друг;
  - опционалният параметър *style* се използва при преобразуване на *datetime* или *smalldatetime* в тип *char*, *nchar*, *varchar* или *nvarchar*:

0 или 100	mon dd yyyy hh:mm AM (или PM)		
1	mm/dd/yy	101	mm/dd/yyyy
2	yy.mm.dd	102	yyyy.mm.dd
3	dd/mm/yy	103	dd/mm/yyyy
4	dd.mm.yy	104	dd.mm.yyyy
…			

ИЗРАЗИ ЗА КОНВЕРТИРАНЕ НА ТИПА ДАННИ – ПРИМЕР

```
SELECT st.StudentNumber + ', ' +
       s.SubjectName + ', ' +
       CONVERT(char(4), e.Evaluation) AS Results
FROM Subjects s
INNER JOIN Evaluations e
   ON s.SubjectID = e.SubjectID
INNER JOIN Students st
  ON st.StudentNumber =
      e.StudentNumber
```

Results
13987, Програмиране I, 6.00
13987, Програмиране II, 3.00
13987, Бази от данни, 5.00
13987, Информационни системи, 5.00
13987, XML, 6.00
13987, DHTML, 4.00
15111, Програмиране I, 4.00
15111, Програмиране II, 5.00
15321, Програмиране I, 4.00
15321, Програмиране II, 6.00

ИЗРАЗИ ЗА КОНВЕРТИРАНЕ НА ТИПА ДАННИ – ПРИМЕР (2)

- Брой продажби по дати

	SaleID	CustomerID	EmployeeID	SaleDate	TotalForSale	Discount
1	1	1	1	2010-01-30 09:08:00.000	8.75	0.00
2	2	3	1	2010-01-30 10:08:00.000	42.00	0.15
3	3	2	3	2010-01-31 12:08:00.000	1.80	0.00
4	4	2	4	2010-01-31 13:08:00.000	10.10	0.00

```
SELECT CONVERT(char(10), SaleDate, 104) AS Date,
       COUNT(*) AS CountOfSales
FROM Sales
GROUP BY CONVERT(char(10), SaleDate, 104)
```

	Date	CountOfSales
1	30.01.2010	2
2	31.01.2010	2

УСЛОВНИ (CASE) ИЗРАЗИ

```
CASE value_expression
    { WHEN value_expression
          THEN {value_expression | NULL} }…
      [ELSE {value_expression | NULL}]
END
```

- Пример

```
SELECT StudentNumber,
       CASE Evaluation
         WHEN 2 THEN 'слаб'
         WHEN 3 THEN 'среден'
         WHEN 4 THEN 'добър'
         WHEN 5 THEN 'мн.добър'
         WHEN 6 THEN 'отличен'
         ELSE 'неизвестна оценка'
       END AS StudentEvaluation,
       Evaluation
FROM Evaluations
```

УСЛОВНИ (CASE) ИЗРАЗИ (2

```
CASE value_expression
    { WHEN value_expressi
          THEN {value_ex
     [ELSE {value_expres
END
```

- Пример

```
SELECT StudentNumber,
       CASE Evaluation
         WHEN 2 THEN 'слаб'
         WHEN 3 THEN 'среден'
         WHEN 4 THEN 'добър'
         WHEN 5 THEN 'мн.добър'
         WHEN 6 THEN 'отличен'
         ELSE 'неизвестна оценка'
       END AS StudentEvaluation,
       Evaluation
FROM Evaluations
```

StudentNumber	StudentEvaluation	Evaluation
13987	отличен	6.00
13987	среден	3.00
13987	мн.добър	5.00
13987	мн.добър	5.00
13987	отличен	6.00
13987	добър	4.00
15111	добър	4.00
15111	мн.добър	5.00
15321	добър	4.00
15321	отличен	6.00

УСЛОВНИ (CASE) ИЗРАЗИ (3)

```
CASE
   {WHEN boolean_expression
    THEN {value_expression | NULL}}…
   [ELSE {value_expression | NULL}]
END
```

- Пример

```
SELECT StudentNumber,
       CASE
          WHEN Evaluation < 3 THEN 'слаб'
          WHEN Evaluation < 3.5 THEN 'среден'
          WHEN Evaluation < 4.5 THEN 'добър'
          WHEN Evaluation < 5.5 THEN 'мн.добър'
          WHEN Evaluation <= 6 THEN 'отличен'
          ELSE 'неизвестна оценка'
       END AS StudentEvaluation,
       Evaluation
FROM Evaluations
```

УСЛОВНИ (CASE) ИЗРАЗИ (4

```
CASE
   {WHEN boolean_expressi
    THEN {value_expressio
   [ELSE {value_expressio
END
```

- Пример

```
SELECT StudentNumber,
       CASE
          WHEN Evaluation < 3 THEN 'слаб'
          WHEN Evaluation < 3.5 THEN 'среден'
          WHEN Evaluation < 4.5 THEN 'добър'
          WHEN Evaluation < 5.5 THEN 'мн.добър'
          WHEN Evaluation <= 6 THEN 'отличен'
          ELSE 'неизвестна оценка'
       END AS StudentEvaluation,
       Evaluation
FROM Evaluations
```

StudentNumber	StudentEvaluation	Evaluation
13987	отличен	6.00
13987	среден	3.00
13987	мн.добър	5.00
13987	мн.добър	5.00
13987	отличен	6.00
13987	добър	4.00
15111	добър	4.00
15111	мн.добър	5.00
15321	добър	4.00
15321	отличен	6.00

УСЛОВНИ (CASE) ИЗРАЗИ (5)

- NULLIF(*value_expression1*, *value_expression2*)
  - ако двете стойности съвпадат, резултатът е NULL;
  - в противен случай, резултатът е първата от двете стойности:

```
CASE value_expression1
  WHEN value_expression2 THEN NULL
  ELSE value_expression1
END
```

УСЛОВНИ (CASE) ИЗРАЗИ (6)

- COALESCE(*value_expression1*, ..., *value_expressionN*) връща първия израз, който не е NULL в списъка от изрази:

```
CASE
    WHEN value_expression1 IS NOT NULL
        THEN value_expression1
    WHEN value_expression2 IS NOT NULL
        THEN value_expression2
    …
    ELSE value_expressionN
END
```

УСЛОВНИ (CASE) ИЗРАЗИ (7)

- ISNULL(*value_expression1*, *value_expression2*) връща първата, ако тя не е NULL и втората, в противен случай:

```
CASE
     WHEN value_expression1 IS NOT NULL
          THEN value_expression1
     ELSE value_expression2
END
```

- **Пример**

```
SELECT StudentNumber,
       ISNULL(Address, 'неизвестен адрес')
       AS StudentAddress
FROM Students
```

УСЛОВНИ (CASE) ИЗРАЗИ (8)

- ISNULL(*value_expression1*, *value*
  първата, ако тя не е NULL и втор

```
CASE
     WHEN value_expressio
          THEN value_expressio
     ELSE value_expression2
END
```

- **Пример**

```
SELECT StudentNumber,
       ISNULL(Address, 'неизвестен адрес')
       AS StudentAddress
FROM Students
```

StudentNumber	StudentAddress
13987	ул. Страцин, 19
15111	ул. В. Левски, 32
15222	неизвестен адрес
15321	бул. България, 10
17111	ул. Христо Ботев, 12
17123	ул. В. Левски, 7
17222	неизвестен адрес
17654	ул. Иван Вазов, 12

ПРИМЕРИ

- Определете резултата от заявката:

```
SELECT FirstName + ' ' + LastName AS Name,
       DATEDIFF(year, HireDate,
                ISNULL(TerminationDate, GetDate()))
       AS Duration
FROM Employees
```

	Name	Duration
1	Стоян Георгиев	10
2	Георги Хростов	5
3	Ваня Хростова	13
4	Стела Миланова	2
5	Атанас Лазаров	10
6	Катя Цветанова	5

- Имената на служителите и броя на годините между датата на назначаване и датата на напускане или текущата дата за служителите, които не са напуснали.

ПРИМЕРИ (2)

- Определете резултата от заявката:

```
SELECT FirstName + ' ' + LastName AS Name,
       CONVERT(char(10), HireDate, 104) AS HireDate
FROM Employees
WHERE DateDiff(year, HireDate, GetDate()) <= 5
```

	Name	HireDate
1	Стела Миланова	08.10.2008
2	Катя Цветанова	25.05.2005

- Имената и датите на назначаване на служителите, назначени за последните 5 години.

ПРИМЕРИ (3)

- Определете резултата от заявката:

```
SELECT DATEPART(year, HireDate) AS Year,
       COUNT(*) AS CountOfEmpl
FROM Employees
GROUP BY DATEPART(year, HireDate)
```

	Year	CountOfEmpl
1	1997	2
2	2000	3
3	2008	1

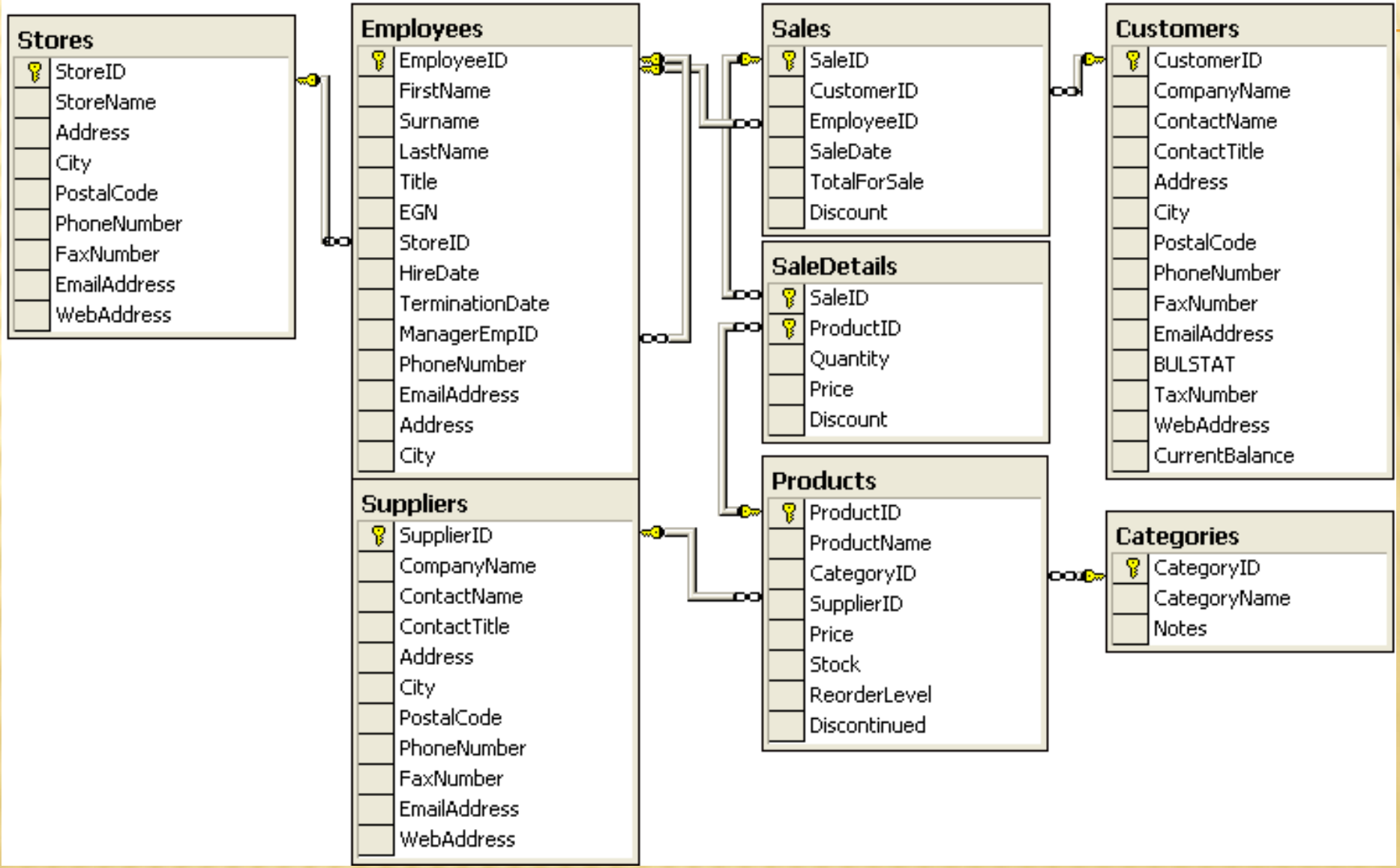
- Годините, през които са назначавани служители и броя на назначените служители през съответните години.

```
SELECT DATEPART(year, HireDate) AS [година],
       COUNT(CASE Title
              WHEN 'мениджър' THEN 1
             END) AS [мениджър],
       COUNT(CASE Title
              WHEN 'управител' THEN 1
             END) AS [управител],
       COUNT(CASE Title
              WHEN 'продавач' THEN 1
             END) AS [продавач],
       COUNT(CASE Title
              WHEN 'касиер' THEN 1
             END) AS [касиер]
FROM Employees
GROUP BY DATEPART(year, HireDate)
```

```
SELECT DATEPART(year, HireDate) AS [година],
       COUNT(CASE Title
              WHEN 'мениджър' THEN 1
             END) AS [мениджър],
       COUNT(CASE Title
              WHEN 'управител' THEN 1
             END) AS [управител],
       COUNT(CASE Title
              WHEN 'продавач' THEN 1
             END) AS [продавач],
       COUNT(CASE Title
              WHEN 'касиер' THEN 1
```

	година	мениджър	управител	продавач	касиер
1	1997	0	1	0	0
2	2000	0	1	2	0
3	2001	1	0	0	0
4	2008	0	0	1	0

ЗАДАЧИ


Stores
StoreID
StoreName
Address
City
PostalCode
PhoneNumber
FaxNumber
EmailAddress
WebAddress
Employees
EmployeeID
FirstName
Surname
LastName
Title
EGN
StoreID
HireDate
TerminationDate
ManagerEmpID
PhoneNumber
EmailAddress
Address
City
Suppliers
SupplierID
CompanyName
ContactName
ContactTitle
Address
City
PostalCode
PhoneNumber
FaxNumber
EmailAddress
WebAddress
Sales
SaleID
CustomerID
EmployeeID
SaleDate
TotalForSale
Discount
SaleDetails
SaleID
ProductID
Quantity
Price
Discount
Products
ProductID
ProductName
CategoryID
SupplierID
Price
Stock
ReorderLevel
Discontinued
Customers
CustomerID
CompanyName
ContactName
ContactTitle
Address
City
PostalCode
PhoneNumber
FaxNumber
EmailAddress
BULSTAT
TaxNumber
WebAddress
CurrentBalance
Categories
CategoryID
CategoryName
Notes

ЗАДАЧИ (ИЗРАЗИ ЗА ДАТА/ЧАС И ИНТЕРВАЛНИ ИЗРАЗИ)

- ***Задача 1***. Да се напише заявка, която да изведе идентификатор на клиент и сума от общата стойност на покупките на съответния клиент през изминалия месец.
- ***Задача 2***. Да се напише заявка, която да изведе данните за всички служители, назначени:
  - ***2.1.*** след датата 01 май 1999 г.;
  - ***2.2.*** през юни 2000 година.
- ***Задача 3***. Да се напише заявка, която да изведе идентификатор на служител и брой на осъществените продажби от съответния служител:
  - ***3.1.*** за текущата дата;
  - ***3.2.*** за вчерашна дата.

ЗАДАЧИ (ИЗРАЗИ ЗА ДАТА/ЧАС И ИНТЕРВАЛНИ ИЗРАЗИ)

- ***Задача 4***. Да се напише заявка, която да извежда сума от общата стойност на продажбите по месеци за текущата година. Резултатът от заявката да има вида:

Num	Month	SumTotal
1	January	632.9750
2	February	785.7500
3	March	122.7750

ЗАДАЧИ (СИМВОЛНИ НИЗОВЕ И ИЗРАЗИ ЗА КОНВЕРТИРАНЕ НА ТИПА ДАННИ)

- ***Задача 1***. Да се напише заявка, която извежда имената и датата на раждане и възрастта на служителите, като се използва ЕГН.
- ***Задача 2***. Да се напише заявка, която извежда дата на продажба, броя на клиентите, направили покупки на тази дата, броя на служителите, осъществили продажби, броя на продажбите, средната стойност на продажбите и общата стойност на продажбите.

ЗАДАЧИ (УСЛОВНИ (CASE) ИЗРАЗИ)

- ***Задача 1***. Да се напише заявка, която извежда данните за продажбите и изчислената отстъпка в зависимост от стойността на TotalForSale: за обща сума на продажбите над 100 – 15%; между 80 и 100 – 10%; между 50 и 80 – 5%; под 50 – 0%.
- ***Задача 2***. Да се напише заявка, която извежда имената и доставните цени на продуктите, класифицирани като евтини, средно скъпи и скъпи.