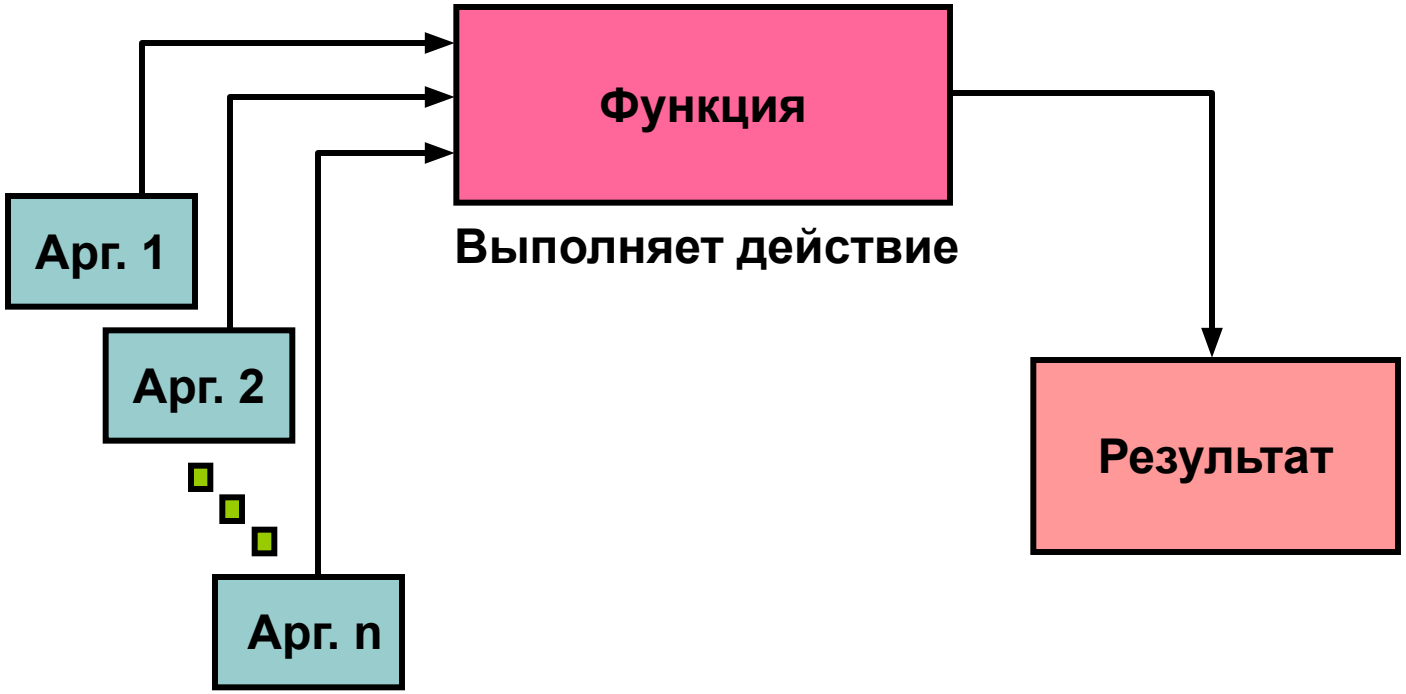
Использование SQL-функций

Цели

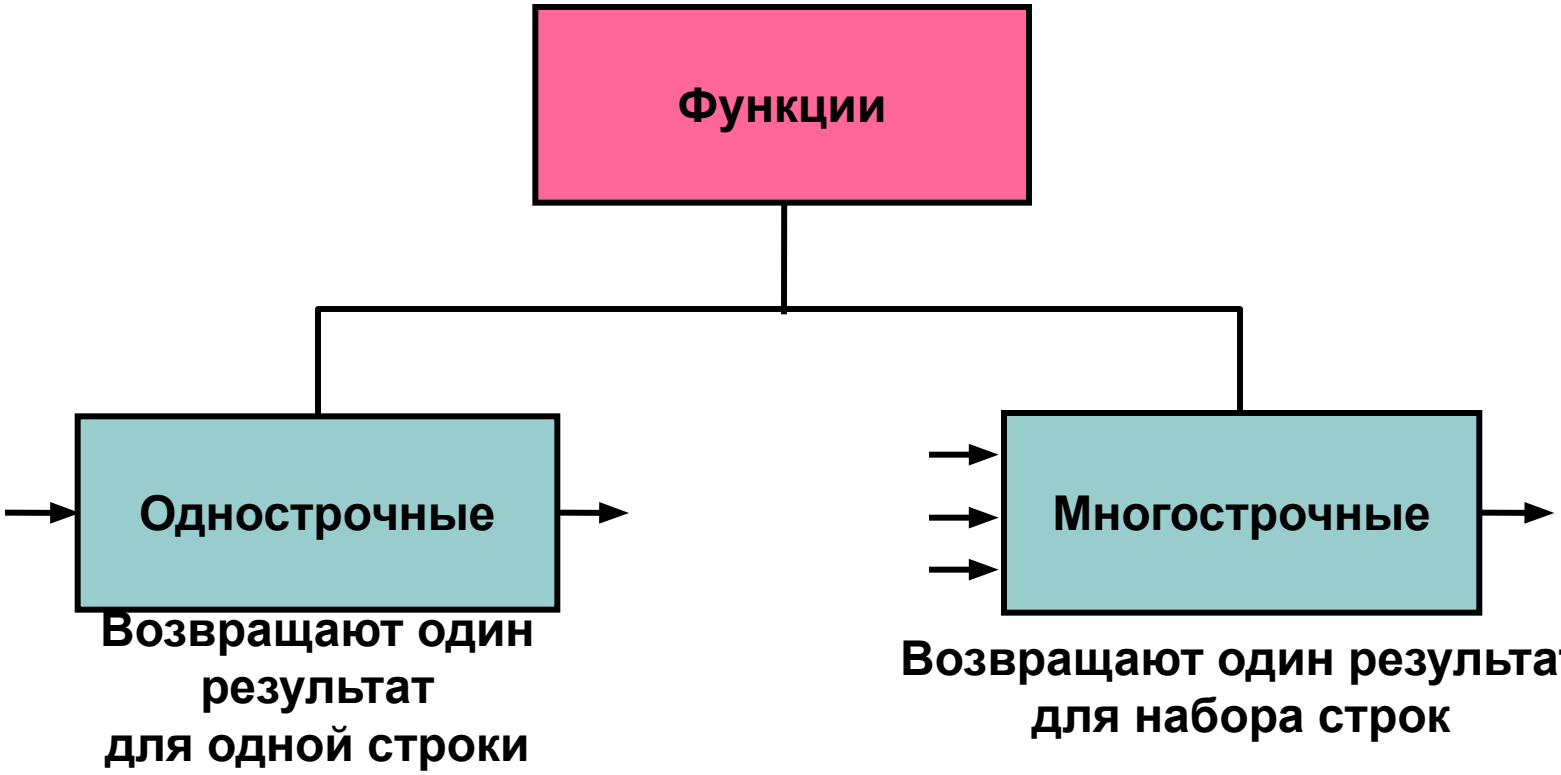
После освоения материала данной лекции, Вы должны быть в состоянии выполнить следующие действия:

- Описать типы функций языка Oracle SQL
- Использовать различные типы функций: символьные, числовые и типа «дата» - в команде SELECT
- Назвать и использовать функции преобразования данных

SQL Функции



Типы SQL Функций



Однострочные функции (Single-Row Functions)

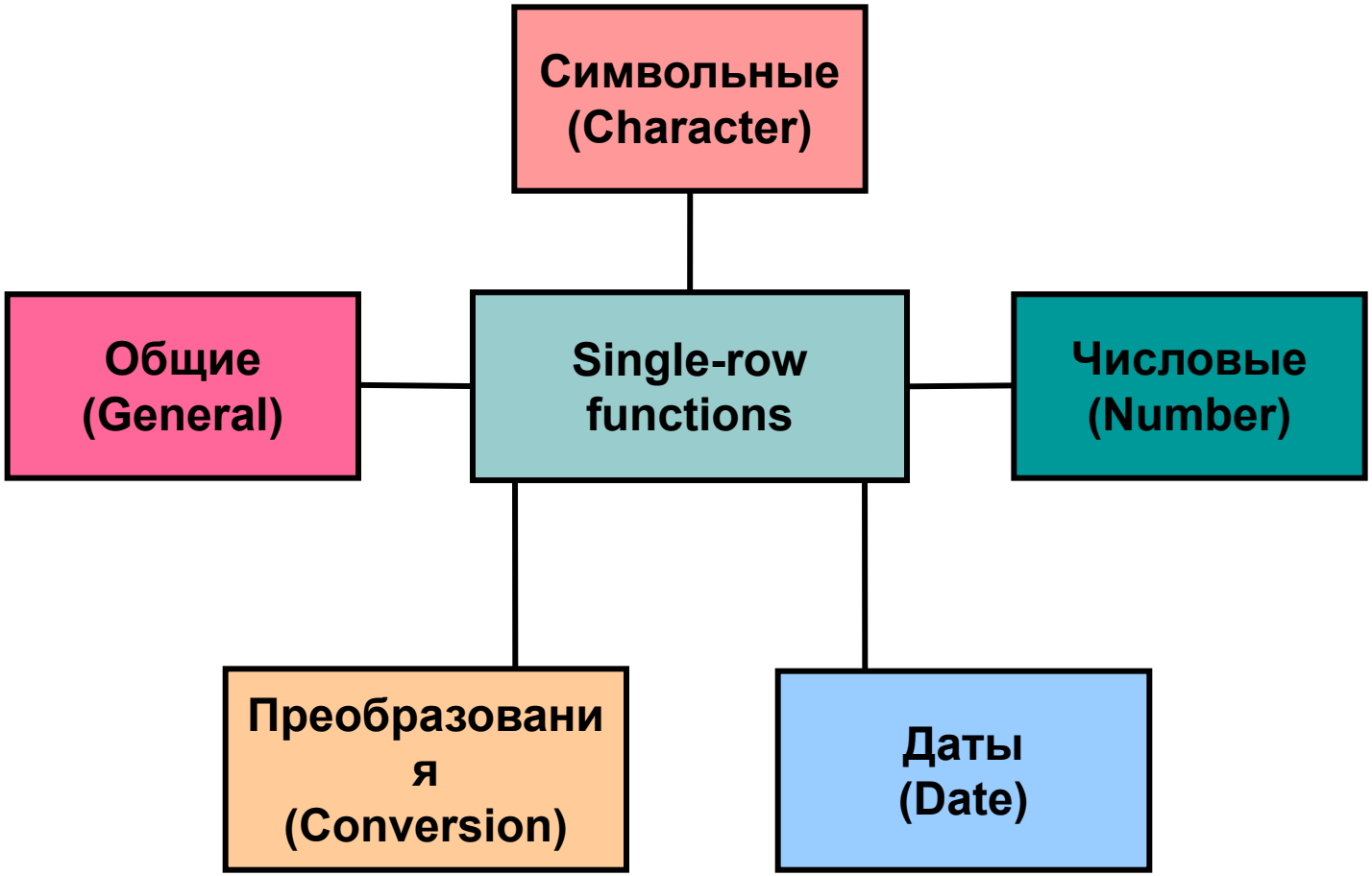
Особенности однострочных функций:

- Обрабатывает каждую строку, возвращаемую запросом
- Возвращает по одному результату для строки
- Могут изменять тип данных: тип данных на выходе может отличаться от типа данных, к которым обращается пользователь
- Могут принимать один или несколько аргументов
- Могут использоваться в предложениях SELECT,

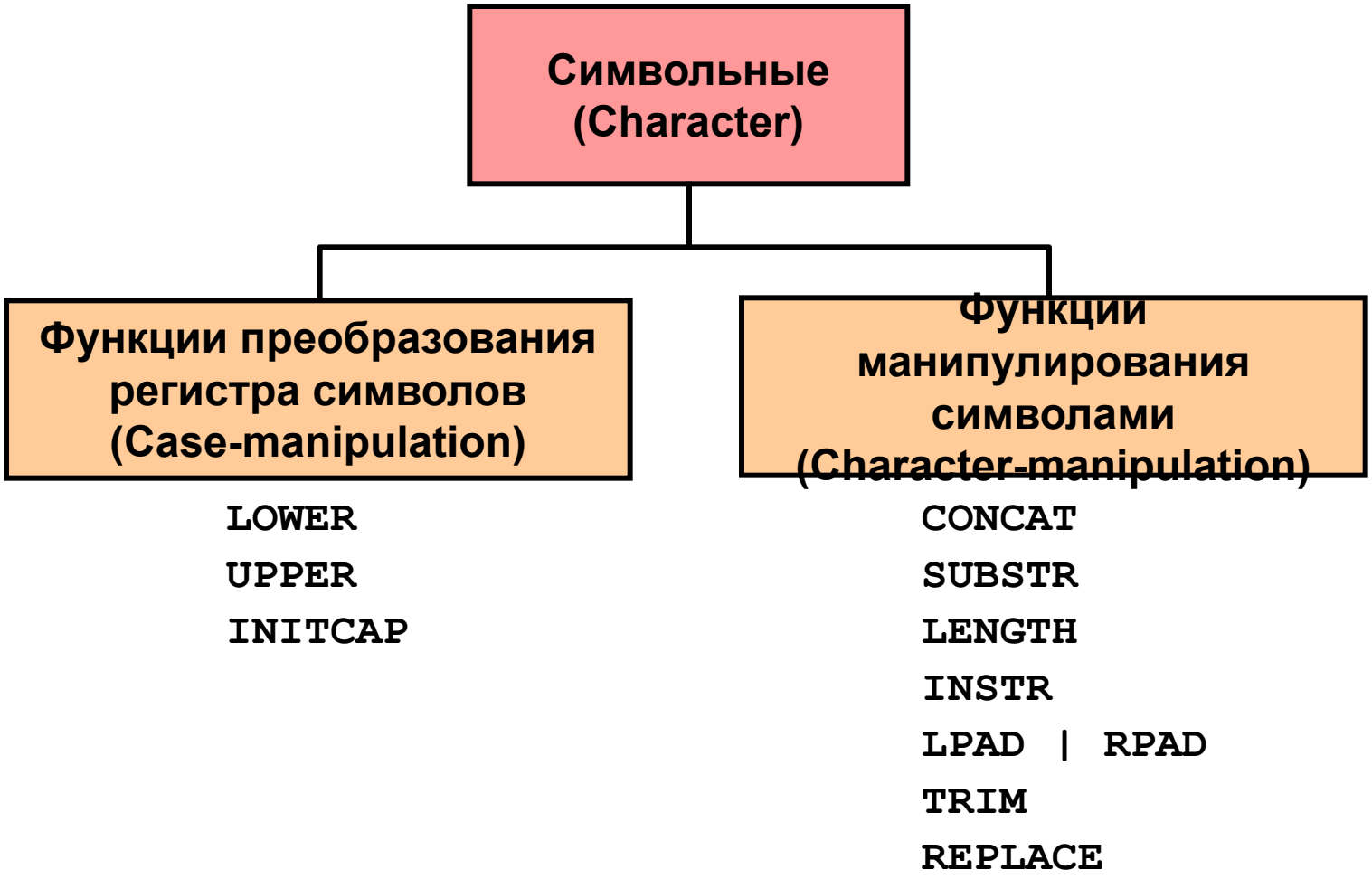
```
function_name [(arg1, arg2,...)]
```

- Могут быть вложенными

Однострочные функции (Single-Row Functions)



Символьные функции (Character Functions)



Функции преобразования регистра символов (Case-Manipulation Functions)

Функция	Результат
LOWER('SQL Course') – преобразует алфавитные символы в нижний регистр	sql course
UPPER('SQL Course') – преобразует алфавитные символы в верхний регистр	SQL COURSE
INITCAP('SQL Course') – преобразует символы по правилу: первая буква каждого слова становится заглавной, остальные - строчные	Sql Course

Пример использования Case-Manipulation Functions

Выбрать номер, фамилию и номер отдела, в котором работает сотрудник по фамилии Higgins:

```
SELECT employee_id, last_name, department_id
FROM   employees
WHERE  last_name = 'higgins';
no rows selected
```

```
SELECT employee_id, last_name, department_id
FROM   employees
WHERE  LOWER(last_name) = 'higgins';
```

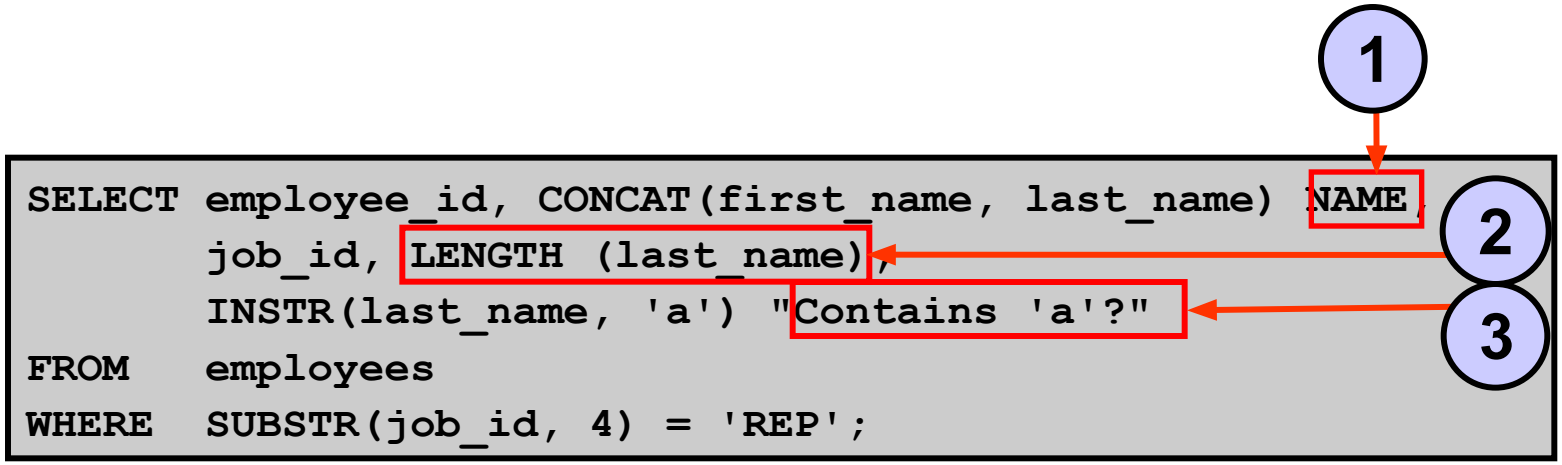
EMPLOYEE_ID	LAST_NAME	DEPARTMENT_ID
205	Higgins	110

Функции манипулирования символами (Character-manipulation)

Функция	Результат
`CONCAT('Hello', 'World')`	`HelloWorld`
`SUBSTR('HelloWorld',1,5)`	`Hello`
`LENGTH('HelloWorld')`	`10`
`INSTR('HelloWorld', 'W')`	`6`
`LPAD(salary,10,'*')`	`*****24000`
`RPAD(salary, 10, '*')`	`24000*****`
`REPLACE ('JACK and JUE','J','BL')`	`BLACK and BLUE`
`TRIM('H' FROM 'HelloWorld')`	`elloWorld`

Примеры использования Character-Manipulation Functions

```
SELECT employee_id, CONCAT(first_name, last_name) NAME,
       job_id, LENGTH (last_name),
       INSTR(last_name, 'a') "Contains 'a'?"
FROM   employees
WHERE  SUBSTR(job_id, 4) = 'REP';
```



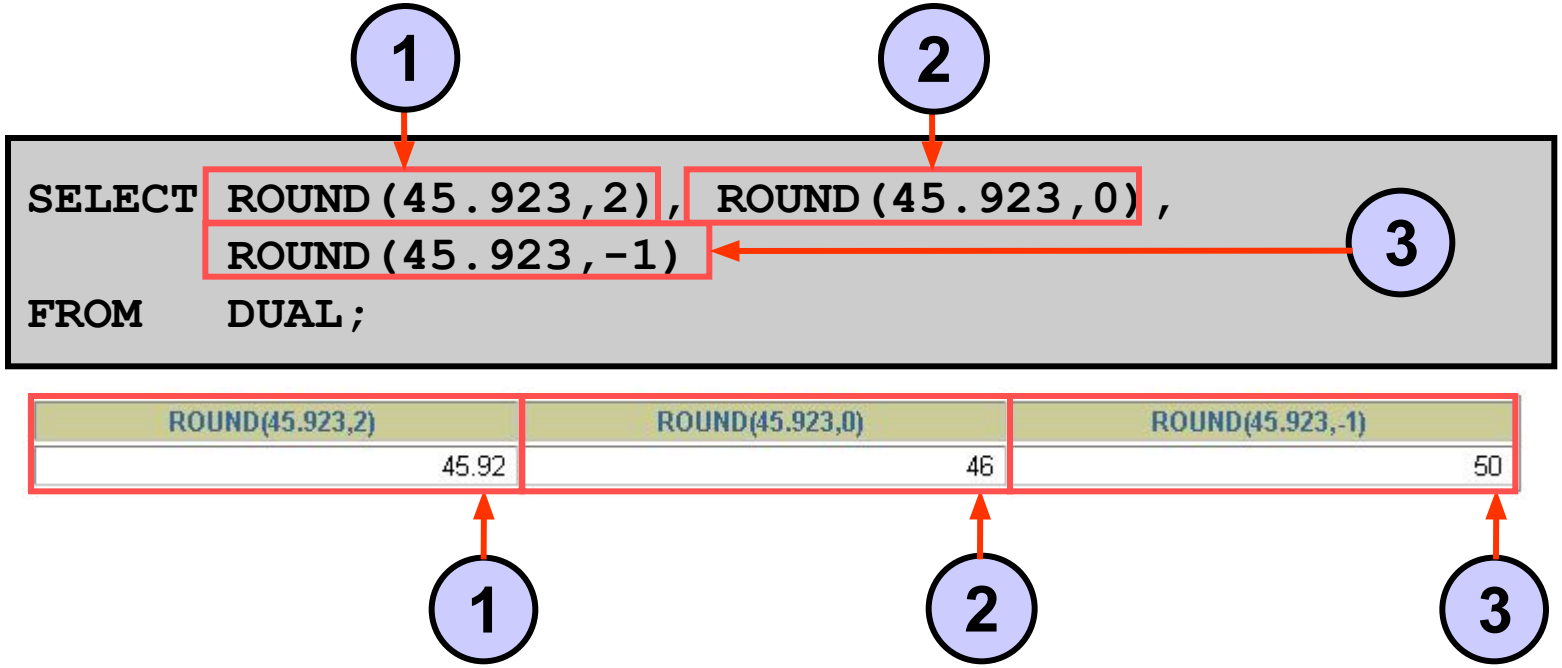
EMPLOYEE_ID	NAME	JOB_ID	LENGTH(LAST_NAME)	Contains 'a'?
174	EllenAbel	SA_REP	4	0
176	JonathonTaylor	SA_REP	6	2
178	KimberelyGrant	SA_REP	5	3
202	PatFay	MK_REP	3	2

Числовые функции (Number Functions)

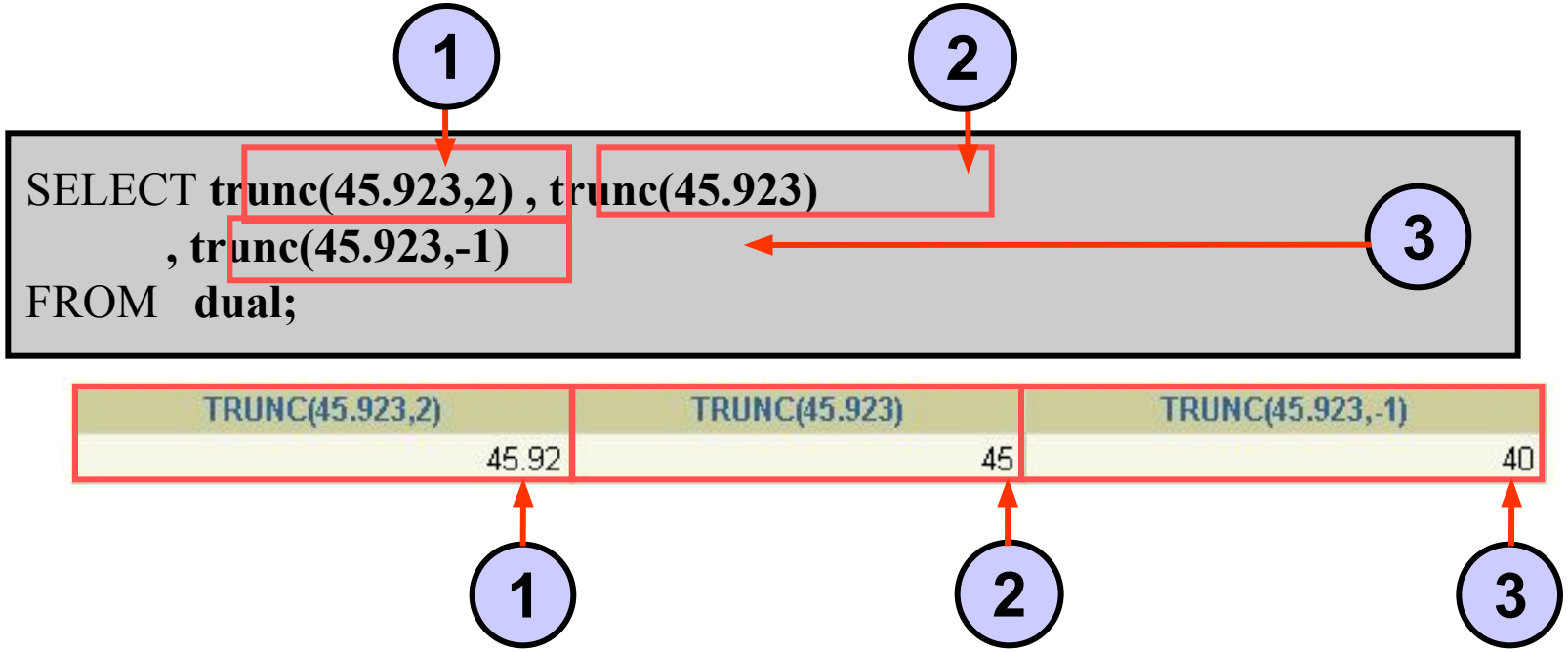
- ROUND: Округляет значение до десятичных разрядов
- TRUNC: Усекает значение до десятичных разрядов

Функция	Результат
ROUND(45.926, 2)	45.93
TRUNC(45.926, 2)	45.92
MOD(1600, 300)	100

Пример использования функции ROUND



Пример использования функции TRUNC



Пример использования функции MOD

```
SELECT last_name, salary, MOD(salary, 5000)
FROM   employees
WHERE  job_id = 'SA_REP';
```

LAST_NAME	SALARY	MOD(SALARY,5000)
Abel	11000	1000
Taylor	8600	3600
Grant	7000	2000

Работа с датами

- **База данных Oracle хранит даты во внутреннем числовом формате: век, год, месяц, день, часы, минуты и секунды.**
- **Стандартный формат отображения DD-MON-RR.**

Можно перенастроить для всей бызы или отдельно сессии dd.mm.yyyy

```
SELECT last_name, hire_date
FROM   employees
WHERE  hire_date < '01-FEB-88';
```

LAST_NAME	HIRE_DATE
King	17-JUN-87
Whalen	17-SEP-87

Работа с датами

Функция SYSDATE возвращает:

- Текущую дату
- Текущее время

Функция current_date возвращает:

- Текущую дату
- Текущее время

Arithmetic with Dates

- Прибавить или вычесть число из даты
- Найти разность между двумя датами.

Использование арфиметических операций с датами

```
SELECT last_name, (SYSDATE-hire_date)/7 AS WEEKS
FROM   employees
WHERE  department_id = 90;
```

LAST_NAME	WEEKS
King	744.245395
Kochhar	626.102538
De Haan	453.245395

Date Functions

Function	Result
MONTHS_BETWEEN	Number of months between two dates
ADD_MONTHS	Add calendar months to date
NEXT_DAY	Next day of the date specified
LAST_DAY	Last day of the month
ROUND	Round date
TRUNC	Truncate date

Using Date Functions

Function	Result
`MONTHS_BETWEEN ('01-SEP-95','11-JAN-94')`	`19.6774194`
`ADD_MONTHS ('11-JAN-94',6)`	`'11-JUL-94'`
`NEXT_DAY ('01-SEP-95','FRIDAY')`	`'08-SEP-95'`
`LAST_DAY ('01-FEB-95')`	`'28-FEB-95'`

Using Date Functions

Assume `SYSDATE = '25-JUL-03'`:

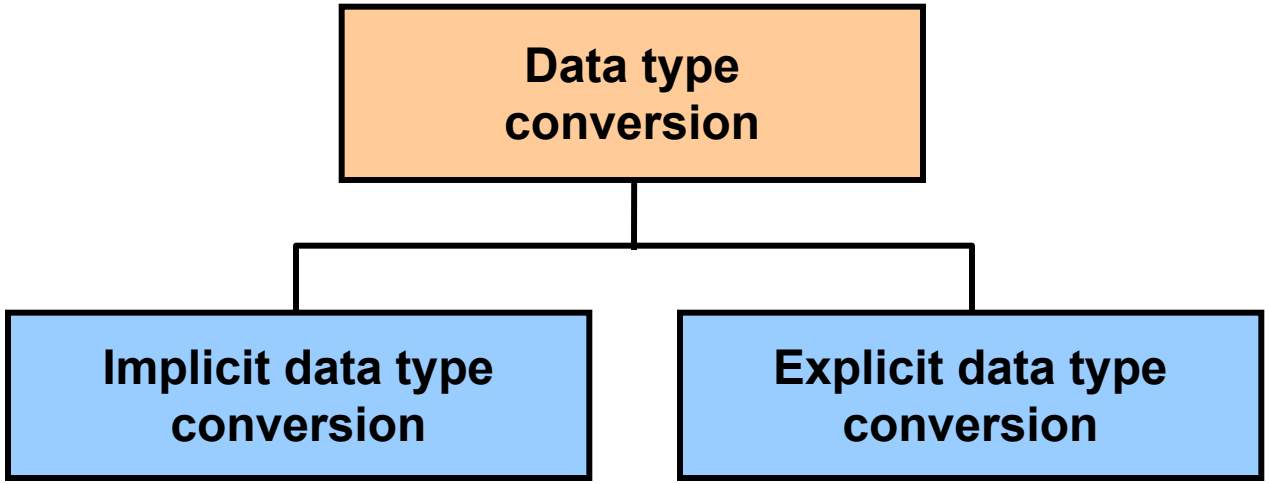
Function	Result
`ROUND(SYSDATE,'MONTH')`	`01-AUG-03`
`ROUND(SYSDATE ,'YEAR')`	`01-JAN-04`
`TRUNC(SYSDATE ,'MONTH')`	`01-JUL-03`
`TRUNC(SYSDATE ,'YEAR')`	`01-JAN-03`

Practice 3: Overview of Part 1

This practice covers the following topics:

- Writing a query that displays the current date
- Creating queries that require the use of numeric, character, and date functions
- Performing calculations of years and months of service for an employee

Conversion Functions



Implicit Data Type Conversion

For assignments, the Oracle server can automatically convert the following:

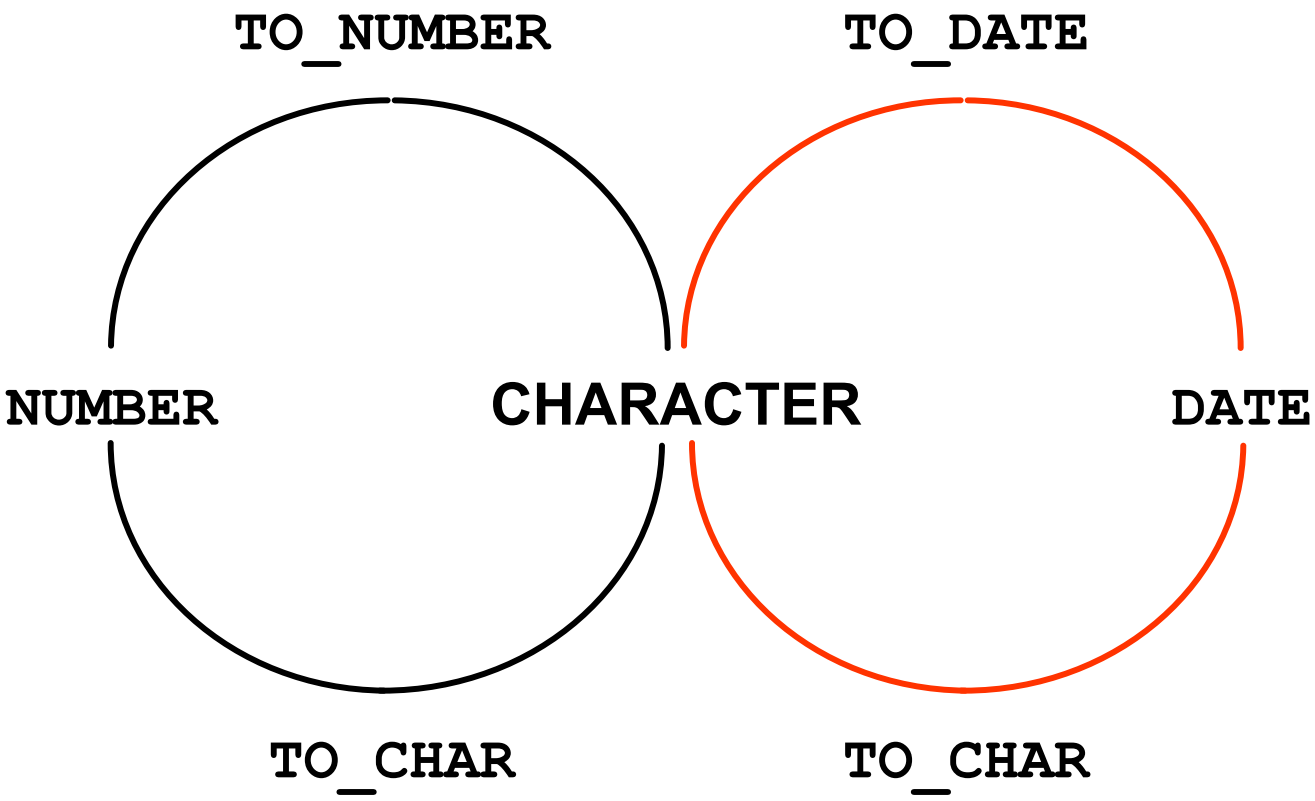
From	To
VARCHAR2 or CHAR	NUMBER
VARCHAR2 or CHAR	DATE
NUMBER	VARCHAR2
DATE	VARCHAR2

Implicit Data Type Conversion

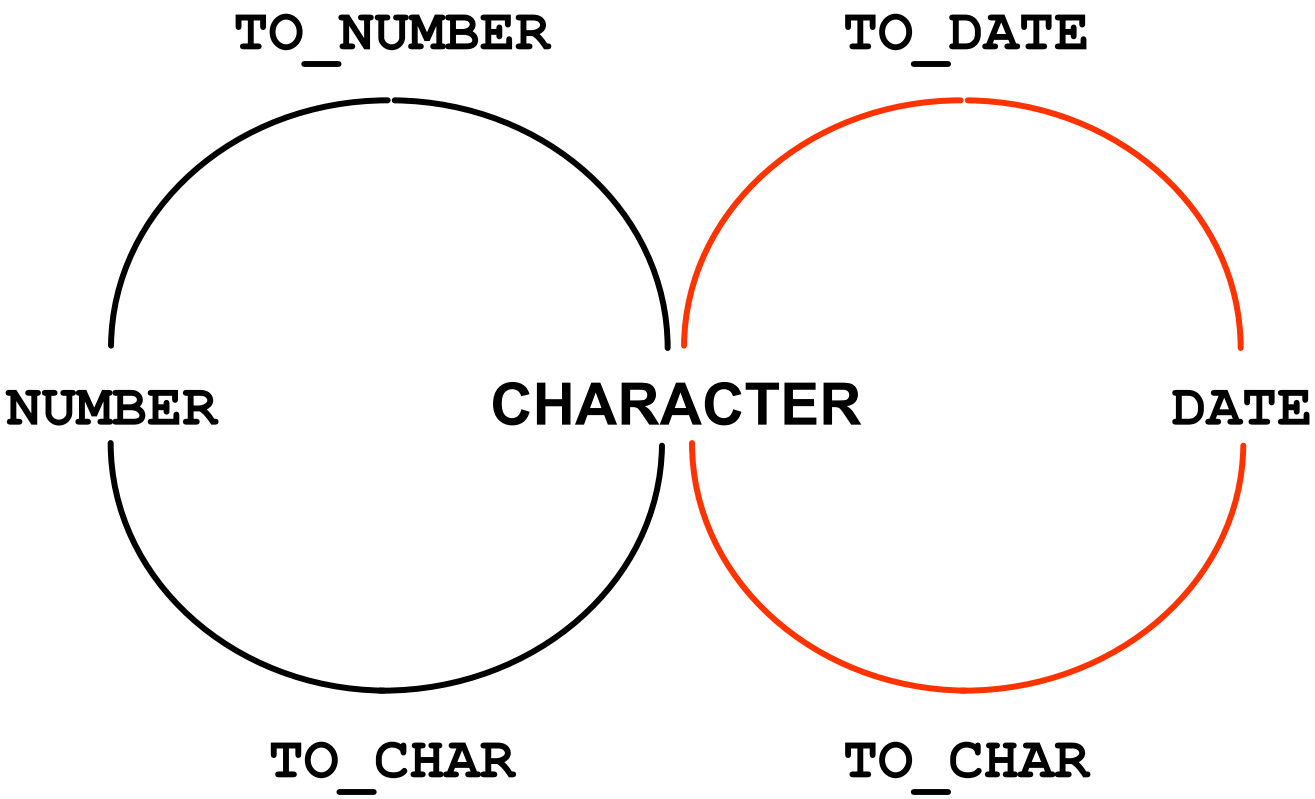
Oracle Server может автоматически неявно преобразовывать типы:

From	To
`VARCHAR2 or CHAR`	`NUMBER`
`VARCHAR2 or CHAR`	`DATE`

Explicit Data Type Conversion



Explicit Data Type Conversion



Using the TO_CHAR Function with Dates

```
TO_CHAR(date, 'format_model')
```

The format model:

- Must be enclosed by single quotation marks
- Is case-sensitive
- Can include any valid date format element
- Has an `fm` element to remove padded blanks or suppress leading zeros
- Is separated from the date value by a comma

Elements of the Date Format Model

Element	Result
YYYY	Full year in numbers
YEAR	Year spelled out (in English)
MM	Two-digit value for month
MONTH	Full name of the month
MON	Three-letter abbreviation of the month
DY	Three-letter abbreviation of the day of the week
DAY	Full name of the day of the week
DD	Numeric day of the month

Elements of the Date Format Model

- Time elements format the time portion of the date:

`HH24:MI:SS AM`	`15:45:32 PM`

- Add character strings by enclosing them in double quotation marks:

`DD "of" MONTH`	`12 of OCTOBER`

- Number suffixes spell out numbers:

`ddspth`	`fourteenth`

Using the TO_CHAR Function with Dates

```
SELECT last_name,
       TO_CHAR(hire_date, 'fmDD Month YYYY')
       AS HIREDATE
FROM   employees;
```

LAST_NAME	HIREDATE
King	17 June 1987
Kochhar	21 September 1989
De Haan	13 January 1993
Hunold	3 January 1990
Ernst	21 May 1991
Lorentz	7 February 1999
Mourgos	16 November 1999

...

20 rows selected.

Using the TO_CHAR Function with Numbers

```
TO_CHAR(number, 'format_model')
```

These are some of the format elements that you can use with the TO_CHAR function to display a number value as a character:

Element	Result
9	Represents a number
0	Forces a zero to be displayed
$	Places a floating dollar sign
L	Uses the floating local currency symbol
.	Prints a decimal point
,	Prints a comma as thousands indicator

Using the TO_CHAR Function with Numbers

```
SELECT TO_CHAR(salary, '$99,999.00') SALARY
FROM   employees
WHERE  last_name = 'Ernst';
```

SALARY
$6,000.00

Using the TO_NUMBER and TO_DATE Functions

- Convert a character string to a number format using the TO_NUMBER function:

```
TO_NUMBER(char[, 'format_model'])
```

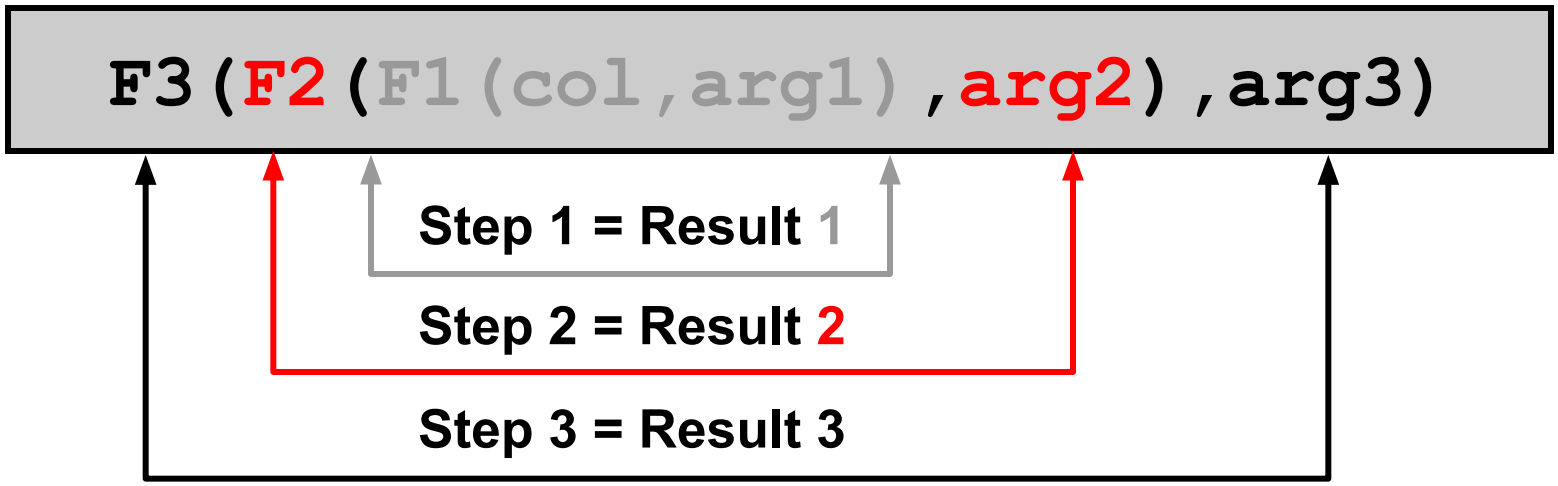
- Convert a character string to a date format using the TO_DATE function:

```
TO_DATE(char[, 'format_model'])
```

- These functions have an fx modifier. This modifier specifies the exact matching for the character argument and date format model of a TO_DATE function.

Nesting Functions

- Single-row functions can be nested to any level.
- Nested functions are evaluated from deepest level to the least deep level.



Nesting Functions

```
SELECT last_name,
  UPPER(CONCAT(SUBSTR (LAST_NAME, 1, 8), '_US'))
FROM   employees
WHERE  department_id = 60;
```

LAST_NAME	UPPER(CONCAT(SUBSTR(LAST_NAME,1,8
Hunold	HUNOLD_US
Ernst	ERNST_US
Lorentz	LORENTZ_US

General Functions

The following functions work with any data type and pertain to using nulls:

- **`NVL (expr1, expr2)`**
- **`NVL2 (expr1, expr2, expr3)`**
- **`NULLIF (expr1, expr2)`**
- **`COALESCE (expr1, expr2, ..., exprn)`**

NVL Function

Converts a null value to an actual value:

- Data types that can be used are date, character, and number.
- Data types must match:
  - `NVL(commission_pct,0)`
  - `NVL(hire_date,'01-JAN-97')`
  - `NVL(job_id,'No Job Yet')`

Using the NVL Function

```
SELECT last_name, salary, NVL(commission_pct, 0),
    (salary*12) + (salary*12*NVL(commission_pct, 0)) AN_SAL
FROM employees;
```

1

2

LAST_NAME	SALARY	NVL(COMMISSION_PCT,0)	AN_SAL
King	24000	0	288000
Kochhar	17000	0	204000
De Haan	17000	0	204000
Hunold	9000	0	108000
Ernst	6000	0	72000
Lorentz	4200	0	50400
Mourgos	5800	0	69600
Rajs	3500	0	42000

...

20 rows selected.

1

2

Using the NVL2 Function

```
SELECT last_name,  salary, commission_pct,
       NVL2(commission_pct,
            'SAL+COMM', 'SAL') income
FROM   employees WHERE department_id IN (50, 80);
```

1

2

LAST_NAME	SALARY	COMMISSION_PCT	INCOME
Zlotkey	10500	.2	SAL+COMM
Abel	11000	.3	SAL+COMM
Taylor	8600	.2	SAL+COMM
Mourgos	5800		SAL
Rajs	3500		SAL
Davies	3100		SAL
Matos	2600		SAL
Vargas	2500		SAL

8 rows selected.

1

2

Using the NULLIF Function

```
SELECT first_name, LENGTH(first_name) "expr1",
       last_name,  LENGTH(last_name)  "expr2",
       NULLIF(LENGTH(first_name), LENGTH(last_name)) result
FROM   employees;
```

FIRST_NAME	expr1	LAST_NAME	expr2	RESULT
Steven	6	King	4	6
Neena	5	Kochhar	7	5
Lex	3	De Haan	7	3
Alexander	9	Hunold	6	9
Bruce	5	Ernst	5	
Diana	5	Lorentz	7	5
Kevin	5	Mourgos	7	5
Trenna	6	Rajs	4	6
Curtis	6	Davies	6	

...

20 rows selected.

Using the COALESCE Function

- The advantage of the COALESCE function over the NVL function is that the COALESCE function can take multiple alternate values.
- If the first expression is not null, the COALESCE function returns that expression; otherwise, it does a COALESCE of the remaining expressions.

Using the COALESCE Function

```
SELECT last_name,
       COALESCE(manager_id,commission_pct, -1) comm
FROM   employees
ORDER BY commission_pct;
```

LAST_NAME	COMM
Grant	149
Zlotkey	100
Taylor	149
Abel	149
King	-1
Kochhar	100
De Haan	100

...

20 rows selected.

Conditional Expressions

- Provide the use of IF-THEN-ELSE logic within a SQL statement
- Use two methods:
  - `CASE` expression
  - `DECODE` function

CASE Expression

Facilitates conditional inquiries by doing the work of an IF-THEN-ELSE statement:

```
CASE expr WHEN comparison_expr1 THEN return_expr1
          [WHEN comparison_expr2 THEN return_expr2
           WHEN comparison_exprn THEN return_exprn
           ELSE else_expr]
END
```

Using the CASE Expression

Facilitates conditional inquiries by doing the work of an IF-THEN-ELSE statement:

```
SELECT last_name, job_id, salary,
       CASE job_id WHEN 'IT_PROG'  THEN  1.10*salary
                   WHEN 'ST_CLERK' THEN  1.15*salary
                   WHEN 'SA_REP'   THEN  1.20*salary
       ELSE       salary END     "REVISED_SALARY"
FROM   employees;
```

LAST_NAME	JOB_ID	SALARY	REVISED_SALARY
...			
Lorentz	IT_PROG	4200	4620
Mourgos	ST_MAN	5800	5800
Rajs	ST_CLERK	3500	4025
...			
Gietz	AC_ACCOUNT	8300	8300

20 rows selected.

DECODE Function

Facilitates conditional inquiries by doing the work of a CASE expression or an IF-THEN-ELSE statement:

```
DECODE(col|expression, search1, result1
                  [, search2, result2,...,]
                  [, default])
```

Using the DECODE Function

```
SELECT last_name, job_id, salary,
       DECODE(job_id, 'IT_PROG',  1.10*salary,
                      'ST_CLERK', 1.15*salary,
                      'SA_REP',   1.20*salary,
              salary)
       REVISED_SALARY
FROM   employees;
```

LAST_NAME	JOB_ID	SALARY	REVISED_SALARY
...			
Lorentz	IT_PROG	4200	4620
Mourgos	ST_MAN	5800	5800
Rajs	ST_CLERK	3500	4025
...			
Gietz	AC_ACCOUNT	8300	8300

20 rows selected.

Using the DECODE Function

Display the applicable tax rate for each employee in department 80:

```
SELECT last_name, salary,
       DECODE (TRUNC(salary/2000, 0),
                          0, 0.00,
                          1, 0.09,
                          2, 0.20,
                          3, 0.30,
                          4, 0.40,
                          5, 0.42,
                          6, 0.44,
                             0.45) TAX_RATE
FROM   employees
WHERE  department_id = 80;
```

Использование групповых функций

Objectives

После завершения этого урока вы должны знать :

- Что такое групповые функции и как их использовать
- Как производить группировку с помощью GROUP BY
- Как производить включение или исключение сгруппированных строк с помощью HAVING

Что такое групповая функция?

Групповые функции работают с наборами строк, чтобы дать один результат в каждой группе.

DEPARTMENT_ID	SALARY
90	24000
90	17000
90	17000
60	9000
60	6000
60	4200
50	5800
50	3500
50	3100
50	2600
50	2500
80	10500
80	11000
80	8600
	7000
10	4400

20 rows selected.

Maximum salary in EMPLOYEES table

MAX(SALARY)
24000

Типы групповых функций

AVG

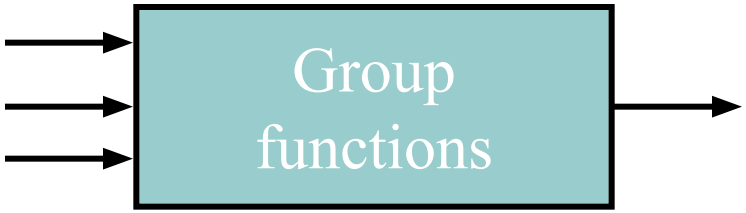
COUNT

MAX

MIN

SUM

...



Функции Группа: Синтаксис

```
SELECT      [column,] group_function(column), ...
FROM      table
[WHERE    condition]
[GROUP BY   column]
[ORDER BY   column];
```

Использование функций AVG и SUM

Вы можете использовать AVG и SUM для числовых данных.

```
SELECT AVG(salary), MAX(salary),
       MIN(salary), SUM(salary)
FROM   employees
WHERE  job_id LIKE '%REP%';
```

AVG(SALARY)	MAX(SALARY)	MIN(SALARY)	SUM(SALARY)
8150	11000	6000	32600

Использование функций MIN и MAX

Вы можете использовать MAX и MIN для типов **numeric, character, date**

```
SELECT MIN(hire_date), MAX(hire_date)
FROM     employees;
```

MIN(HIRE_	MAX(HIRE_
17-JUN-87	29-JAN-00

Использование COUNT функции

COUNT(*) возвращает количество строк в таблице :

1

```
SELECT COUNT(*)
FROM   employees
WHERE  department_id = 50;
```

COUNT(*)
5

COUNT(*expr*) возвращает количество строк с ненулевых значений для *expr*:

2

```
SELECT COUNT(commission_pct)
FROM   employees
WHERE  department_id = 80;
```

COUNT(COMMISSION_PCT)
3

Использование DISTINCT

`COUNT(DISTINCT expr)` возвращает число различных ненулевых значениях *`expr`*.

Для того, чтобы отобразить количество различных значений отдела в таблице EMPLOYEES :

```
SELECT COUNT(DISTINCT department_id)
FROM   employees;
```

COUNT(DISTINCTDEPARTMENT_ID)
7

Групповые функции и значения Null

Групповые функции игнорируют столбцы со значением null :

1

```
SELECT AVG(commission_pct)
FROM   employees;
```

AVG(COMMISSION_PCT)
.2125

Функция NVL позволяет включать нулевые значения:

2

```
SELECT AVG(NVL(commission_pct, 0))
FROM   employees;
```

AVG(NVL(COMMISSION_PCT,0))
.0425

Создание групп данных

DEPARTMENT_ID	SALARY
10	4400
20	13000
20	6000
50	5800
50	3500
50	3100
50	2500
50	2600
60	9000
60	6000
60	4200
80	10500
80	8600
80	11000
90	24000
90	17000

20 rows selected.

4400

9500

3500

6400

10033

Average salary in EMPLOYEES table for each department

DEPARTMENT_ID	AVG(SALARY)
10	4400
20	9500
50	3500
60	6400
80	10033.3333
90	19333.3333
110	10150
	7000

Создание групп данных: Синтаксис предложения GROUP BY

```
SELECT     column, group_function(column)
FROM       table
[WHERE     condition]
[GROUP BY group_by_expression]
[ORDER BY column];
```

Вы можете разделить строки в таблице на более мелкие группы при помощи предложения GROUP BY.

Использование предложения GROUP BY

Все столбцы в списке выбора, к которым не применяются групповые функции должны быть описаны в предложении GROUP BY.

```
SELECT    department_id, AVG(salary)
FROM      employees
GROUP BY department_id ;
```

DEPARTMENT_ID	AVG(SALARY)
10	4400
20	9500
50	3500
60	6400
80	10033.3333
90	19333.3333
110	10150
	7000

8 rows selected.

Использование предложения GROUP BY

Столбец в GROUP не обязательно должен находиться в SELECT.

```
SELECT   AVG(salary)
FROM     employees
GROUP BY department_id ;
```

AVG(SALARY)
4400
9500
3500
6400
10033.3333
19333.3333
10150
7000

Группировка по нескольким столбцам

DEPARTMENT_ID	JOB_ID	SALARY
90	AD_PRES	24000
90	AD_VP	17000
90	AD_VP	17000
60	IT_PROG	9000
60	IT_PROG	6000
60	IT_PROG	4200
50	ST_MAN	5800
50	ST_CLERK	3500
50	ST_CLERK	3100
50	ST_CLERK	2600
50	ST_CLERK	2500
80	SA_MAN	10500
80	SA_REP	11000
80	SA_REP	8600
20	MK_REP	6000
110	AC_MGR	12000
110	AC_ACCOUNT	8300

20 rows selected.

Add the salaries in the EMPLOYEES table for each job, grouped by department

DEPARTMENT_ID	JOB_ID	SUM(SALARY)
10	AD_ASST	4400
20	MK_MAN	13000
20	MK_REP	6000
50	ST_CLERK	11700
50	ST_MAN	5800
60	IT_PROG	19200
80	SA_MAN	10500
80	SA_REP	19600
90	AD_PRES	24000
90	AD_VP	34000
110	AC_ACCOUNT	8300
110	AC_MGR	12000
	SA_REP	7000

13 rows selected.

Использование GROUP BY на несколько колонок

```
SELECT   department_id dept_id, job_id,
  SUM(salary)
FROM     employees
GROUP BY department_id, job_id ;
```

DEPT_ID	JOB_ID	SUM(SALARY)
10	AD_ASST	4400
20	MK_MAN	13000
20	MK_REP	6000
50	ST_CLERK	11700
50	ST_MAN	5800
60	IT_PROG	19200
80	SA_MAN	10500
80	SA_REP	19600
90	AD_PRES	24000
90	AD_VP	34000
110	AC_ACCOUNT	8300
110	AC_MGR	12000
	SA_REP	7000

13 rows selected.

Некорректные Запросы
Использование групповых функций

Любой столбец или выражение в списке SELECT, который не является агрегатной функцией должен быть описан в предложении GROUP BY:

```
SELECT department_id, COUNT(last_name)
FROM   employees;
```

```
SELECT department_id, COUNT(last_name)
       *
ERROR at line 1:
ORA-00937: not a single-group group function
```

Колонка отсутствуют в списке предложения GROUP BY

Вы не можете использовать групповые функции в WHERE. Для этой цели используйте HAVING.

```
SELECT   department_id, AVG(salary)
FROM     employees
WHERE    AVG(salary) > 8000
GROUP BY department_id;
```

```
WHERE  AVG(salary) > 8000
       *
ERROR at line 3:
ORA-00934: group function is not allowed here
```

Cannot use the WHERE clause to restrict groups

Ограничение результатов группировки

DEPARTMENT_ID	SALARY
90	24000
90	17000
90	17000
60	9000
60	6000
60	4200
50	5800
50	3500
50	3100
50	2600
50	2500
80	10500
80	11000
80	8600
20	6000
110	12000
110	8300

20 rows selected.

The maximum salary per department when it is greater than $10,000

DEPARTMENT_ID	MAX(SALARY)
20	13000
80	11000
90	24000
110	12000

Ограничение результатов группировки с использованием HAVING

При использовании предложения HAVING, сервер Oracle ограничивает группы следующим образом:

1. Строки сгруппированы.
2. Применяется групповая функция.
3. Отображаются группы, соответствующие предложения HAVING.

```
SELECT     column, group_function
FROM       table
[WHERE     condition]
[GROUP BY  group_by_expression]
[HAVING    group_condition]
[ORDER BY  column];
```

Использование предложения HAVING

```
SELECT    department_id, MAX(salary)
FROM      employees
GROUP BY  department id
HAVING    MAX(salary)>10000 ;
```

DEPARTMENT_ID	MAX(SALARY)
20	13000
80	11000
90	24000
110	12000

Использование предложения HAVING

```
SELECT   job_id, SUM(salary) PAYROLL
FROM     employees
WHERE    job_id NOT LIKE '%REP%'
GROUP BY job_id
HAVING   SUM(salary) > 13000
ORDER BY SUM(salary);
```

JOB_ID	PAYROLL
IT_PROG	19200
AD_PRES	24000
AD_VP	34000

Nesting Group Functions

Отображение максимальной средней заработной платы:

```
SELECT   MAX(AVG(salary))
FROM     employees
GROUP BY department_id;
```

MAX(AVG(SALARY))
19333.3333

Summary

Вы научились:

Использовать групповые функции COUNT, MAX, MIN и AVG

Писать запросы, которые используют GROUP BY

Писать запросы, которые используют HAVING

```
SELECT    column, group_function
FROM      table
[WHERE    condition]
[GROUP BY group_by_expression]
[HAVING   group_condition]
[ORDER BY column];
```

Задания на занятие

1. Количество сотрудников по департаментам.
   - Оставить только департаменты с средней ЗП > 7000

2. Количество уникальных JOB_ID в разрезе департаментов

3. Вывести одним запросом дату найма первого сотрудника и дату найма последнего сотрудника в компании.

4. Вывести количество сотрудников, являющихся руководителями.

5.* Вывести среднюю ЗП руководителей.

6.* Размер 2 по величине ЗП.

* - Задачки «на вырост». Темы мы еще не проходили.
Можно погуглить «использование подзапросов»

Домашнее задание

Employees

1. Часть сотрудников, при выполнении планов получает премию к ЗП. Доля этой премии указана в поле comission_pct.

A) Выведите список сотрудников в алфавитном порядке с указанием ЗП с учетом премии.

ФИО	ЗП	премия	ЗП с премией
aGsYXbwOEi JLIVINGS	1488	0,2	1785,6
MysmtkShkf KGRANT	1488	0,15	1711,2
pgGjYbRHca CJOHNSON	1488	0,1	1636,8
KefjgAnopz WTAYLOR	1488	0	1488
qtdVNEYqcO JFLEAUR	1488	0	1488
RlwxITtsXfY MSULLIVA	1488	0	1488

б) Выведите список подразделений, среднюю ЗП с учетом прем среднюю ЗП без учета премии.

Отсортировать по абсолютному размеру премии.

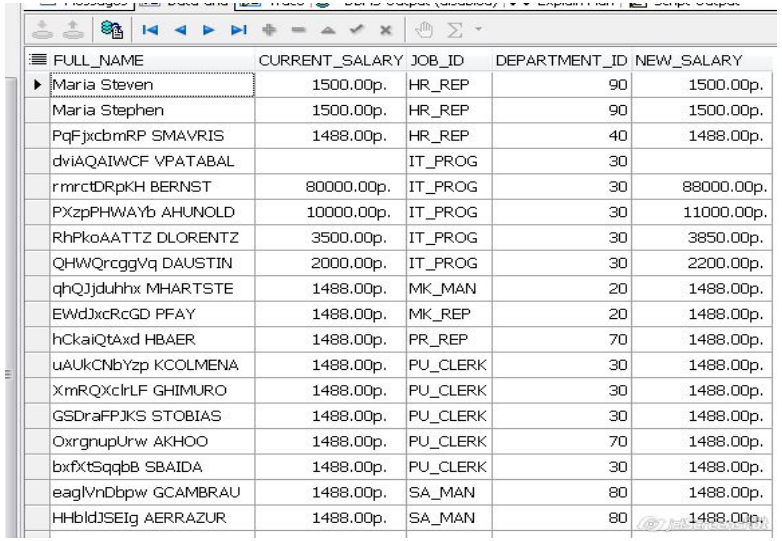
DEPARTMENT_ID	AVG_SALARY	AVG_SAL_2
80	1488	1822,8
	1488	1711,2
90	2874,06666666667	2874,06666666667
70	1488	1488
10	1488	1488

2. Служба найма попросила предоставить данные, в какие месяцы чаще всего люди устраивались в компанию? Результат вывести в формате месяц, количество сотрудников.

Отсортировать в порядке возрастания по количеству сотрудников.

MONTH	CNT
сентябрь	5
октябрь	6
ноябрь	6
июль	7
апрель	7
май	7
декабрь	8
август	10
июнь	10
февраль	13
январь	14
март	17

3. В компании приняли решение увеличить заработную плату всем сотрудникам с должностью IT_PROG на 10%, сотрудникам с должностью SH_CLERK уменьшить заработную плату на 10%.
Выведите таблицу, которая покажет разницу в ЗП по каждому сотруднику.
Результат отсортируйте по JOB_ID (по возрастанию), внутри группы JOB_ID по новой ЗП (по убыванию)



FULL_NAME	CURRENT_SALARY	JOB_ID	DEPARTMENT_ID	NEW_SALARY
Maria Steven	1500.00р.	HR_REP	90	1500.00р.
Maria Stephen	1500.00р.	HR_REP	90	1500.00р.
PqFjxcbmRP SMAVRIS	1488.00р.	HR_REP	40	1488.00р.
dviAQAIWCF VPATABAL		IT_PROG	30	
rmrctDRpKH BERNST	80000.00р.	IT_PROG	30	88000.00р.
PXzpPHWAYb AHUNOLD	10000.00р.	IT_PROG	30	11000.00р.
RhPkoAATTZ DLORENTZ	3500.00р.	IT_PROG	30	3850.00р.
QHWQrcggVq DAUSTIN	2000.00р.	IT_PROG	30	2200.00р.
qhQJjduhhx MHARTSTE	1488.00р.	MK_MAN	20	1488.00р.
EWdJxcRcGD PFAY	1488.00р.	MK_REP	20	1488.00р.
hCkaiQtAxd HBAER	1488.00р.	PR_REP	70	1488.00р.
uAUkCNbYzp KCOLMENA	1488.00р.	PU_CLERK	30	1488.00р.
XmRQXclrLF GHIMURO	1488.00р.	PU_CLERK	30	1488.00р.
GSDraFPJKS STOBIAS	1488.00р.	PU_CLERK	30	1488.00р.
OxrgnupUrw AKHOO	1488.00р.	PU_CLERK	70	1488.00р.
bxfXtSqqbB SBAIDA	1488.00р.	PU_CLERK	30	1488.00р.
eaglVnDbpw GCAMBRAU	1488.00р.	SA_MAN	80	1488.00р.
HHbldJSEIg AERRAZUR	1488.00р.	SA_MAN	80	1488.00р.

4. Для каждого менеджера вывести количество непосредственных подчиненных

Бонус: дополнительно выведите средний стаж подчиненных в неделях.
Бонус: оставьте в итоговой выборке только список менеджеров, у которых есть подчиненные трудоустроенные в текущем году.

MANAGER_ID	COUNT(EMPLOYEE_ID)	СТАЖ в НЕДЕЛЯХ
100	17	613
124	8	524
	1	47

5. Вывести количество сотрудников, в разрезе стажа

СТАЖ (ЛЕТ)	Количество
0	4
1	1
2	1
3	1
4	1