/*

Курс по СУБД **Oracle:**

основы администрирования, **SQL, PL/SQL**

*/

```
CREATE OR REPLACE PACKAGE "Занятие 2" AS
```

T · · Systems ·

План занятия

- Типы данных в Oracle SQL
- Оператор **SELECT**
- Фильтрация **WHERE**
- Сортировка **ORDER BY**
- Однострочные функции
- Условные выражения
- Представления (**VIEW**)

END;

T · · Systems ·

```
CREATE OR REPLACE PACKAGE BODY “Занятие 2”
AS
   l_alert VARCHAR2(10);
BEGIN
   l_alert := ‘Продолжаем?’;
   dbms_output.put_line(l_alert);
```

Типы данных

Символьные

- **CHAR** — строки фиксированной длины (в байтах)
- **VARCHAR2** — строки переменной длины (в байтах)
- **NCHAR** и **NVARCHAR2** — только Unicode строки (в символах)

Примеры:

- **VARCHAR2(20 BYTE)** = **VARCHAR2(20)** в байтах
- **VARCHAR2(20 CHAR)** — в символах

В зависимости от параметра **NLS_LENGTH_SEMANTICS**

```
SELECT value
  FROM nls_database_parameters
 WHERE parameter = 'NLS_LENGTH_SEMANTICS';
```

T · · Systems ·

Типы данных

Числовые

- **`NUMBER`** — фиксированные и с плавающей точкой
  - **precision** — **всего** знаков в числе (вплоть до 38)
  - **scale** — знаков после десятичной запятой
- **`BINARY_FLOAT`** и **`BINARY_DOUBLE`** — вещественные типы

Примеры хранения числа 123,456.7[illegible]

Разделители заданы параметром **`NLS_NUMERIC_CHARACTERS`**

- **`NUMBER(*, 2)`** — 123456,79
- **`NUMBER(6, -3)`** — 123000
- **`NUMBER(3)`** — ошибка, превышение точности

T · · Systems ·

Типы данных

Дата и время

- **DATE** — значения момента времени (дата и время)

Oracle хранит формат в 7 байтах: век, год, месяц, день, часы, минуты, секунды

Стандартный формат — '**DD-MON-YY**': '**12-MAR-**

При этом Oracle всё равно хранит год как 4 символа

- **TIMESTAMP** = **DATE** + дробные секунды + пояса
  - **TIMESTAMP WITH LOCAL TIME ZONE**

Полезные функции:

- **CURRENT_DATE** — время сессии
- **SYSDATE** — время сервера

Из таблицы **DUAL**

T · · Systems ·

Типы данных

ROWID

ROWID — псевдоколонка, хранящее двоичное значение, однозначно определяющее физический

```
SELECT * FROM dual;
```

DUMMY
X

```
SELECT d.*, rowid FROM dual d;
```

DUMMY	ROWID
X	AAAACOAABAAAAWJAAA

AAAACO — сегмент БД
AAB — номер файла данных
AAAAWJ — номер блока
AAA — строка в блоке

Особенности:

- Используется в создании индексов (ключ => ROWID)
- Самый быстрый способ доступа к определённой строке
- Меняется при любых физ. манипуляциях —

NULL

- **NULL** — отсутствие значения, неопределённость
- **NULL** ≠ 0
- При сортировке считается наибольшим значением
  - Можно регулировать **`NULLS FIRST`** и **`NULLS LAST`**
- **NULL** ≠ **NULL**
- Результат арифметических операций с **NULL** равен **NULL**

- ~~**`value = / <> NULL`**~~ **`value IS / IS NOT NULL`**
- Oracle «трактует» пустую строку как **NULL** (при INSERT'е)

T··Systems·

Написание кода на SQL

- В SQL не различаются регистры символов
- Операторы — прописными буквами, названия — строчными
- Предложения SQL могут занимать одну или несколько строк
- Каждое предложение обычно пишется на отдельной строке
- Ключевые слова нельзя сокращать и размещать на

SQL Style Guide:

- Для облегчения чтения используются отступы
- Желательно не превышать 120 символов в строке

T · · Systems ·

SELECT

```
SELECT [DISTINCT] *|{column|expression [alias],...}
  FROM table;
```

- **SELECT** — какие столбцы будут выбраны
- **FROM** — откуда столбцы будут выбраны

- Все столбцы: **SELECT ***
- Определённые столбцы: **SELECT col1, col3, col5**

- Служебная таблица **dual**

T · · Systems ·

SELECT

- Можно использовать **арифметические выражения**
- Можно изменить заголовок столбца
  - **Псевдонимы** (alias) задаются ключевым словом **AS**
  - Если есть пробелы и/или спец.символы, ставятся кавычки
- Можно **соединять** столбцы или символы со столбцами (||)

```
SELECT film_title || ' (' || film_release_year || ')' AS Фильм,
       film_rental_rate                               AS "Цена в $",
       film_rental_rate * 60                          AS "Цена в ₽"
  FROM films;
```

T-Systems

WHERE

```
SELECT [DISTINCT] *|{column|expression [alias],...}
  FROM table
[WHERE condition(s)];
```

- Отбор конкретных строк по какому-то условию (условиям)
- Операторы сравнения:
  - =, >, >=, <, <=, <> или !=
  - **BETWEEN**...**AND**... — находится между двумя значениями
  - **(NOT) IN (**...,...,...**)** — совпадает со значением из списка
  - **LIKE** '...%' — совпадает с шаблоном (% для нечёткого поиска)

WHERE

- Логические операторы: **AND**, **OR**, **NOT**

AND	TRUE	FALSE	NULL
TRUE	TRUE	FALSE	NULL
FALSE	FALSE	FALSE	FALSE
NULL	NULL	FALSE	NULL

OR	TRUE	FALSE	NULL
TRUE	TRUE	TRUE	TRUE
FALSE	TRUE	FALSE	NULL
NULL	TRUE	NULL	NULL

NOT	
TRUE	FALSE
FALSE	TRUE
NULL	NULL

- Приоритет операторов:
  1. Арифметические (+, -, *)
  2. Конкатенация (||)
  3. Сравнение (>, =, <)
  4. **IS (NOT) NULL**, **LIKE**, **(NOT) IN**
  5. **(NOT) BETWEEN**
  6. **NOT**
  7. **AND**
  8. **OR**

T · · Systems ·

ORDER BY

```
SELECT [DISTINCT] *|{column|expression [alias],...}
  FROM table
[WHERE condition(s)]
[ORDER BY {column|alias|expr|numeric_position} [ASC|DESC]];
```

- **ASC**ending — по возрастанию, **DESC**ending — по убыванию
- **ORDER BY** — последнее предложение оператора SELECT
- Сортировка возможна по:
  - Названию или псевдониму столбца
  - Номеру столбца
  - Столбцу с применённой к нему функцией
  - Нескольким столбцам

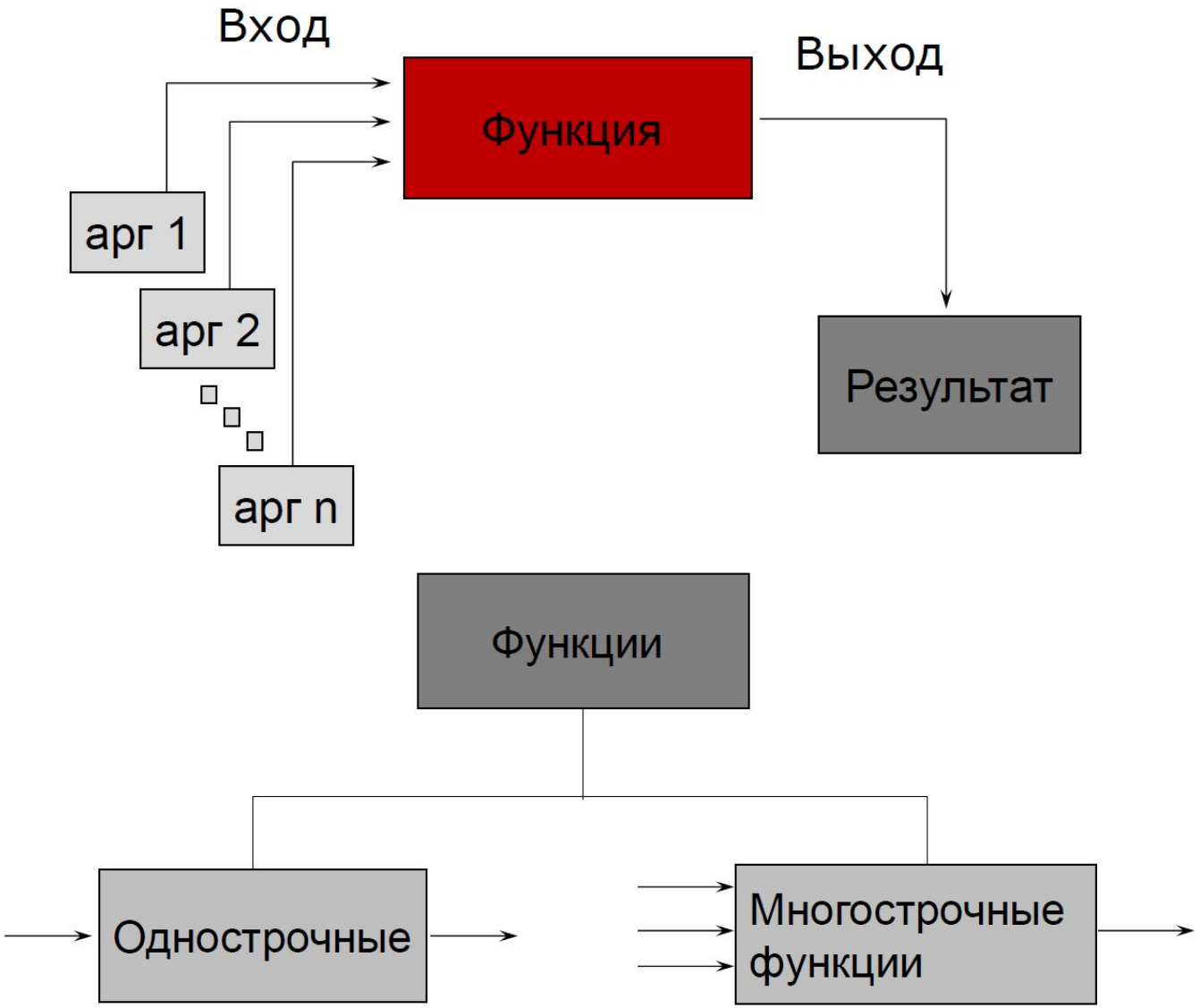
SELECT + WHERE + ORDER BY

◆Вывести названия («Фильм») и рейтинг («Рейтинг») фильмов в порядке уменьшения продолжительности, которые были сняты в 2006 году, по продолжительности превышают полтора часа, к которым есть трейлеры, у которых нет перевода и чья цена за прокат варьируется от $2 до $3 в сутки.

```
SELECT film_title AS Фильм, film_rating AS Рейтинг
  FROM films
 WHERE film_release_year = 2006
   AND film_length > 90
   AND film_special_features LIKE '%Trailers%'
   AND film_orig_lang_id IS NULL
   AND film_rental_rate BETWEEN 2 AND 3
 ORDER BY film_length DESC, Фильм
```

T··Systems·

Функции SQL

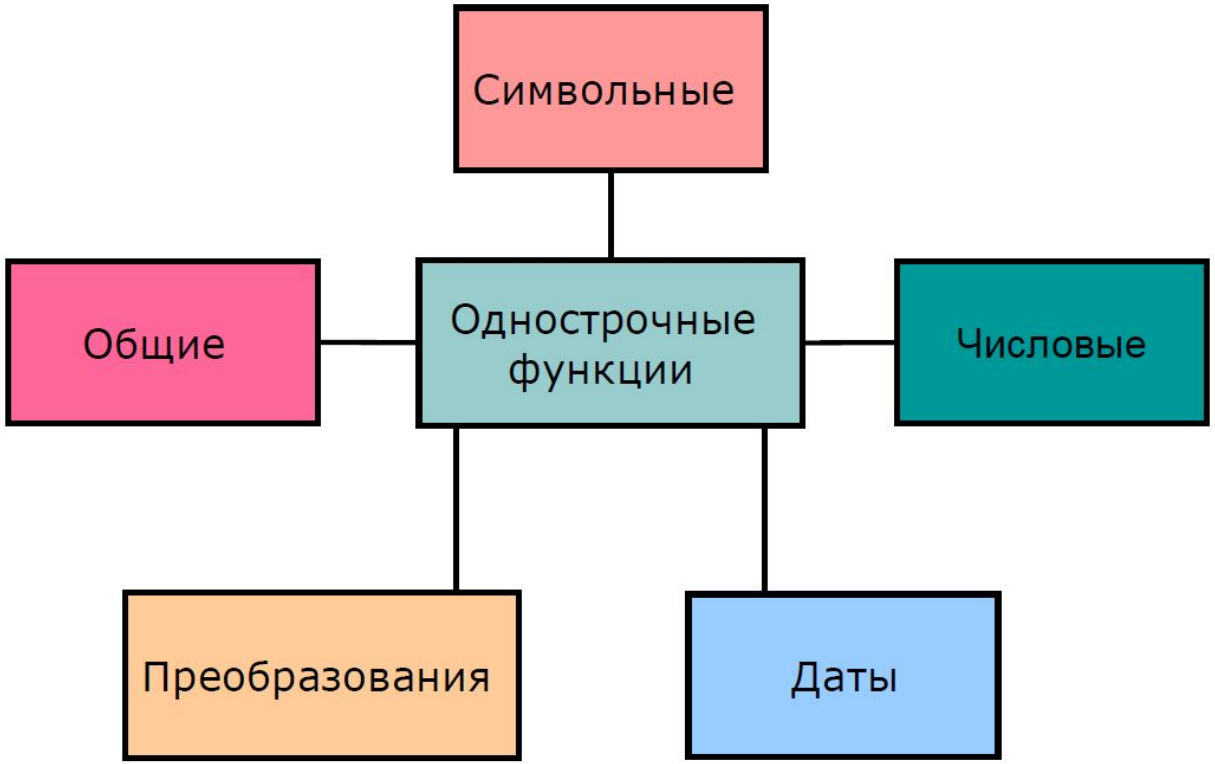


T · · Systems ·

Однострочные функции

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

- Принимают аргументы — столбец или выражение — и возвращают одно значение



T · · Systems ·

Символьные функции

- **LOWER**(*строка*), **UPPER**(*строка*), **INITCAP**(*строка*) — преобразование регистра: строчные буквы, прописные буквы, каждое слово со строчной буквы, соответственно
- **CONCAT**(*строка 1*, *строка 2*) — конкатенация двух строк
- **LENGTH**(*строка*) — длина строки
- **LPAD**(*строка*, *кол-во символов*, [*символы*]), **RPAD** (…) — дополнение строки символами слева/справа
- **CHR**(*код*) — возврат символа по числовому коду
- **TRIM**([[**LEADING**|**TRAILING**|**BOTH**] *символ* **FROM**] *строка*) — удаление символов в начале/конце (пробел — по умолчанию)

T · · Systems ·

Символьные функции

- *__INSTR__(*строка*, *подстрока*, [*нач. позиция,* [*n-ное вхождение*]]) — возврат n-ного вхождения под у
- *__SUBSTR__(*строка*, *нач. позиция*, [*длина*]) — возврат подстроки
- *__REPLACE__(*строка*, *набор для замены*, [*заменяющий набор*]) — цельная замена одного набора символов другим
- __TRANSLATE__(*строка*, *набор 1*, *набор 2*) — последовательная замена одного набора символов другим

В отличие от многих языков, нумерация идёт с 1, не с 0

* Имеют эквиваленты с префиксом REGEXP_ для работы с регулярными выражениями

T··Systems·

Символьные функции
Примеры:

-- Вывести названия всех фильмов, начало каждого слова сделав с прописной буквы и заменив, где есть, слово “angels” на “daemons”

```
SELECT INITCAP(REPLACE(film_title, UPPER('angels'), 'daemons')) AS
film
  FROM films;
```

-- Вывести с указанием года выпуска в формате «оглавления» названия всех фильмов, описание которых длиннее 100 символов

```
SELECT CONCAT(RPAD(film_title, 30, '.'), film_release_year) AS film
  FROM films
 WHERE LENGTH(film_description) > 100;
```

-- Вывести названия всех фильмов, в описании которых больше 5 раз есть слово “oracle”

```
SELECT film_title AS film
  FROM films
 WHERE INSTR(LOWER(film_description), 'oracle', 1, 6) <> 0;
```

Числовые функции

- **ROUND**(*число*, [*позиция после запятой*]) — округление числа
- **TRUNC**(*число*, [*цифр после запятой*]) — отбрасывание части числа
- **FLOOR**(*число*), **CEIL**(*число*) — возврат наименьшего/наибольшего целого, которое меньше или равно числу/которое больше или равно числу
- **POWER**(*число*, *степень*), **SQRT**(*число*) — возведение числа в определённую степень/извлечение квадратного корня из числа
- **SIGN**(*число*) — возврат значения, определяющего знак числа:
  - `-1`, если число отрицательное
  - `0`, если число = 0
  - `1`, если число положительное

T · · Systems ·

+ математические функции: **ABS**, **SIN**/**COS**, **LOG**/**LN**, **EXP**

Числовые функции

Пример:

-- Вывести название и продолжительность фильмов в порядке её убывания, где она представляет собой квадрат целого числа

```
SELECT film_title  AS film,
       film_length AS duration
  FROM films
 WHERE SQRT(film_length) = TRUNC(SQRT(film_length))
 ORDER BY film_length DESC;
```

-- Вывести название фильмов с минимальной суммой налички, с которого уже потребуется сдача за аренду фильмов на 7 дней

```
SELECT film_title                    AS film,
       CEIL(film_rental_rate * 7) AS cash
  FROM films;
```

T · · Systems ·

Функции для работы с датами

- **MONTHS_BETWEEN**(*дата 1*, *дата 2*) —кол-во месяцев между датами
- **ADD_MONTHS**(*дата*, *кол-во месяцев*)
- **NEXT_DAY**(*дата*, *день недели*) — дата ближайшего дня недели
- **LAST_DAY**(*дата*) — дата последнего дня месяца на основе даты
- **ROUND**(*дата*, [*формат*]), **TRUNC**(…) — округлённая/усечённая дата
- **EX**
- **SYSDATE**, **CURRENT_DATE** — ???

- «дата + число» или «дата – число» = «дата»
- «дата – дата» = «кол-во дней между этими датами»

T · · Systems ·

Функции для работы с датами

Пример:

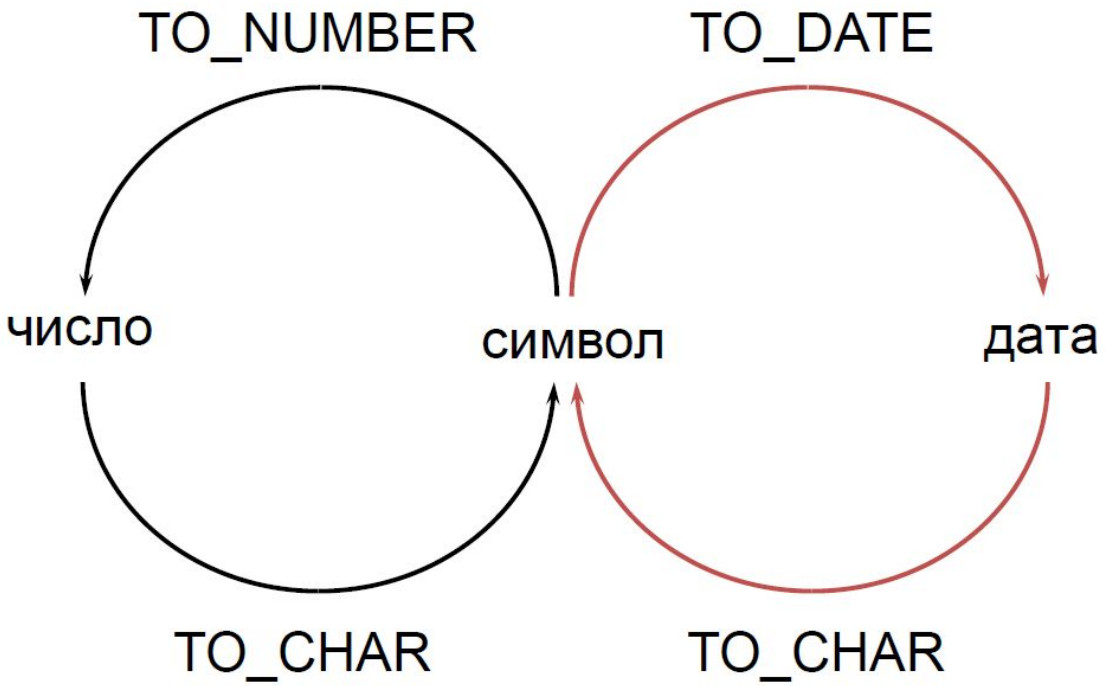
```
-- Вывести ID счастливых покупателей, которые сделали заказы 29-го февраля

SELECT orde_customer_id AS happy_customers
  FROM orders
 WHERE EXTRACT(MONTH FROM orde_made_date) = 2
   AND EXTRACT(DAY FROM orde_made_date) = 29;


-- Вывести ID копий фильмов и соответствующую дату квартала, когда эти
фильмы наиболее популярны (фильм популярен, если пробыл в аренде больше
недели)
SELECT orde_inventory_id          AS best_rentals,
       TRUNC(orde_made_date, 'Q') AS best_quarter
  FROM orders
 WHERE FLOOR(MONTHS_BETWEEN(orde_return_date, orde_made_date) * 31) >
7;
```

T · · Systems ·

Функции преобразований типов



- **TO_NUMBER**(*строка*, [*маска*]) — преобразование строки в число
- **TO_DATE**(*строка*, [*маска*]) — преобразование строки в дату по маске
- **TO_CHAR**(*значение*, [*маска*]) — преобразование числа или даты в строку

- **CAST**(*выражение* **AS** *тип*) — преобразование выражения в

Функции преобразований типов

Пример:

-- Вывести в человеческом формате все даты взятия фильмов в аренду, указав в скобках день недели. Пример: «**Май 29, 2017 (понедельник)**»

```
SELECT TO_CHAR(orde_made_date, 'FMMonth DD, YYYY (DAY)') AS rental
  FROM orders;
```

T · · Systems ·

Общие функции

- **NVL**(*строка*, *на что заменить*) — замена строки, если она NULL
- **NVL2**(*строка*, *заменить если не NULL*, *заменить если NULL*) — замена строки в любом случае: когда она **NULL** или нет
- **NULLIF**(*выражение 1*, *выражение 2*) — возврат **NULL**, если выражения равны, и *выражения 1* в противном случае
- **COALESCE**(*вып. 1*, *вып. 2*, …) — возврат первого не-**NULL**-значения
- **GREATEST**(*вып. 1*, *вып. 2*, …), **LEAST**(…) — наибольшее/наименьшее значение в списке выражений (числа, строки, **NULL**)

T · · Systems ·

Общие функции
Пример:

-- Вывести ФИО заказчика с его основным контактом (первоочердно это телефон, но если его нет, это E-Mail)

```
SELECT cust_first_name || ' ' || cust_last_name AS customer,
       LOWER(COALESCE(cust_phone, cust_email))  AS contact
FROM customers;
```

T · · Systems ·

Условные выражения

~~IF-THEN-ELSE~~

- **DECODE**(*выражение*,
  значение 1, *результат 1*
  [, *значение* ***n***, *результат* ***n***]…
  [, *результат по умолчанию*]) — возврат результата **n**, если значение **n** соответствует выражению **n**, иначе результата по умолчанию

- **CASE** [*выражение*]
  WHEN *условие 1* **THEN** *результат 1*
  [**WHEN** *условие* ***n*** **THEN** *результат* ***n***]
  [**ELSE** *результат по умолчанию*]
  END

T · · Systems ·

Условные выражения
Пример:

```
-- Вывести фамилии сотрудников с указанием их статуса: активный сотрудник или уволенный

SELECT empl_last_name    AS employee,
       CASE empl_active
         WHEN 0 THEN 'Resigned'
         WHEN 1 THEN 'Active'
       END               AS status
FROM employees;

SELECT empl_last_name        AS employee,
       DECODE(empl_active,
              1, 'Active',
              0, 'Resigned') AS status
FROM employees;
```

Представления (VIEW)

- Представление — логическая таблица, «именованный

```
CREATE [OR REPLACE] [FORCE|NO FORCE] VIEW view [(alias[, alias]...)]
    AS subquery
  [WITH CHECK OPTION [CONSTRAINT constraint]]
  [WITH READ ONLY [CONSTRAINT constraint]];
```

- DML-операции могут выполняться с простыми представлениями
- Не выполняются, если в запросе есть:
  - Групповые функции, ROWNUM, WITH READ ONLY
  - Столбцы с выражениями
  - Столбцы NOT NULL, которые не находятся в представлении

```
DROP VIEW view;
```

T-Systems

Практика

1. Создать в своей схеме представление HW2_1 со списком клиентов:
   - «**ФИО**»
     - необходимо по-человечески «капитализировать» ФИО
     - вместо имени “Willie” корректнее писать “Willy” (* без использования REPLACE)
   - «**Email**», где домен “sakilacustomer.org” нужно поменять на “sakila.ru”
   - «**Год**», когда клиент внесён в базу
   - Отсортировать список по ID города в обратном порядке
2. Создать в своей схеме представление HW2_2 со списком заказов:
   - «**Взято за**», где указано кол-во полных дней, оставшихся до конца месяца с момента, как был сделан заказ
   - «**Статус**», где указано, фильм «Сдан» или «На руках»
   - Нужны только трёхзначные ID копий фильмов, по которым нужно отсортировать
3. Создать в своей схеме представление HW2_3 со списком городов:
   - «**Город**», в названии которого больше двух слов

```
  EXCEPTION
    WHEN questions
      just_ask;
    WHEN others
      NULL;
  END;
END "Занятие 2";

ALTER PACKAGE "Занятие 2" COMPILE BODY;
ALTER PACKAGE "Занятие 2" COMPILE PACKAGE;
```