**Сводная таблица операторов в системе PL/SQL**

|  |  |
| --- | --- |
| Разделы и операторы | **PL/SQL** |
| Тип языка программирования | Блочно-структурированный |
| Объявление переменных | Объявление переменных производится в начале программы, процедуры или функции в блоке DECLARE:  *<имя\_переменной> [CONSTANT] <тип\_переменной> [NOT NULL] [ { := | DEFAULT } <выражение>];*  Например:  *birthdate DATE;  emp\_count SMALLINT := 0;  emp\_count SMALLINT DEFAULT 0;  acct\_id VARCHAR2(5) NOT NULL := ‘AP001’;  v\_MYCITY OFFICES.CITY%TYPE;  a INTEGER := v\_RecOffices.OFFICE;* |
| Объявление констант | Объявление констант производится в начале программы, процедуры или функции в блоке DECLARE:  *<имя\_константы> CONSTANT <тип\_константы> := <значение\_константы>;*  Примеры:  *pi CONSTANT REAL := 3.14159;* |
| Общая структура оператора | *<Тело\_оператора>;* (оператор оканчивается знаком “точка с запятой”) |
| Операторы присвоения | *<переменная> := <значение>;* (используется знак “двоеточие и равно”)  Примеры:  *r := r + i;  name\_plsql\_table(index) := expr;* |
| Оператор безусловного перехода | *GOTO имя\_метки  <операторы<  ...  <<имя\_метки>>  <операторы<  ...*  Примеры:  *SQL< BEGIN 2 BEGIN 3 GOTO MID; 4 ..................... 5 END; 6 <> 7 ..................... 8 END; 9 /* |
| Метка | *<<имя\_метки>>  <исполнимый\_оператор>*  Примеры:  *<>  ...* |
| Оператор условного перехода | *IF (<условие> | <выражение типа Boolean (True/False)> ) THEN  (<блок выражений> | [BEGIN <блок выражений> END]) [ELSIF  (<блок выражений> | [BEGIN <блок выражений> END])]  [ELSE (<блок выражений> | [BEGIN <блок выражений> END])]  END IF*  Примеры:  IF l\_date < ’11-APR-03’ THEN l\_salary := l\_salary \* 1.15; ELSE  l\_salary := l\_salary \* 1.05; END IF; |
| Реализация условного оператора в одну строку | Аналогом однострочного условного оператора является функция DECODE:  *DECODE (<имя\_столбца>, '<значение\_для\_поиска1>',  '<отображаемое\_значение1>', [ '<значение\_для\_поиска2<',  '<отображаемое\_значение2>',  […], '<значение\_по\_умолчанию>' ] )*  Примеры:  *SELECT first\_name AS "Имя", last\_name As "Фамилия", DECODE(JOB\_ID, 'SA\_REP', 'Торговый представитель', 'SA\_MAN', 'Менеджер по продажам', 'Другое' ) AS "Должность" FROM hr.employees;* |
| Операторы цикла: Цикл со счетчиком | *FOR <имя\_переменной> IN [REVERSE] <начальное\_значение> .. <конечное\_значение>  LOOP  <блок выражений> END LOOP;*  Счетчик циклов <имя\_переменной> изменяется от начального значения до конечного значения с шагом 1, а при использовании опции “REVERSE” – от конечного до начального значения с шагом -1.  Между начальным и конечным значениями ставится знак “..” - “оператор диапазона”.  Примеры:  *FOR i IN 1..3 LOOP  // последовательность команд; -- цикл выполняется 3 раза (для i = 1, 2, 3) END LOOP;*  Цикл FOR для курсоров (аналогичен оператору ДЛЯ КАЖДОГО в языке 1С):  *FOR <переменная типа запись> IN <имя курсора>  LOOP  <блок выражений>  END LOOP;*  Примеры:  *FOR v\_gt IN get\_offices LOOP DBMS\_OUTPUT.put\_line(‘Get Data: ‘||TO\_CHAR(v\_gt.OFFICE) ); END LOOP;* |
| Цикл с условием | *LOOP <блок выражений с условием выхода из цикла>  END LOOP;*  - приводит к бесконечному повторению последовательности команд, если внутри нее нет команд EXIT (выход из цикла), RAISE (вызов обработчика исключительных ситуаций) или GOTO (безусловный переход).   Например:  LOOP *<блок выражений>  IF <условие> THEN EXIT; END LOOP;*  *- приведет к выходу из цикла после выполнения последовательности команд, как только условие станет истинным.*  *LOOP <блок выражений>  EXIT WHEN <условие>; END LOOP;* |
| Операторы досрочного выхода из цикла | Завершает цикл и передает управление оператору, следующему непосредственно после окончания цикла.  EXIT WHEN <условие>;  Примеры:  *LOOP*  *i:=i+1;*  *EXIT WHEN i>11;  END LOOP;* |
| Сообщения (вывод сообщений на экран) | *DBMS\_OUTPUT.put\_line('<текст\_сообщения<');*  Пример:  *SET SERVEROUTPUT ON  BEGIN  DBMS\_OUTPUT.enable;  DBMS\_OUTPUT.put\_line('Это чудесно!');  END;  /* |
| Обработка ошибок и исключений | *Оператор EXCEPTION*  *[WHEN имя\_исключения THEN  текст\_обработчика\_исключения;]* |
| Процедуры и функции | *Процедура:*  *CREATE [OR REPLACE] PROCEDURE <имя\_процедуры>  (<параметр> [IN] [OUT] [IN OUT] <тип\_параметра>, ..... ) AS [IS]  <зона объявления переменных>  BEGIN  .....<тело\_процедуры>  EXCEPTION  <раздел исключительных ситуаций<  END [имя процедуры]*  где OR REPLACE добавляется, если требуется изменение ранее созданной процедуры или создание процедуры с возможностью изменения;  IN – входной параметр;  OUT – выходной параметр;  IN OUT – параметр, одновременно являющийся и входным и выходным (т.е. в процессе действия процедуры предполагается изменение входного параметра)  *Функция:*  *CREATE [OR REPLACE] FUNCTION <имя\_функции<  (<параметр> [IN] [OUT] [IN OUT] <тип\_параметра>, ..... )  [RETURN <тип\_возвращаемого\_значения>]  AS [IS]  <зона объявления переменных<  BEGIN  .....<тело\_функции<  RETURN (<возвращаемое\_значение>); (или без скобок: RETURN <возвращаемое\_значение>;)  EXCEPTION  <раздел исключительных ситуаций<  END [имя процедуры]*  Примеры:  *CREATE OR REPLACE PROCEDURE TESTOUT(NUM IN NUMBER, DT OUT VARCHAR2) IS  BEGIN  SELECT COMPANY INTO DT FROM customers  WHERE customers.CUST\_NUM = NUM;  END TESTOUT;  /* |

**Сопоставление встроенных функций VB, PL/SQL и 1С:**

|  |  |
| --- | --- |
| Математические функции: | **ROUND(x, [,y])** округляет число до заданной точности  **TRUNC(x, [,y])** - Возвращает x усеченное (не округленное) до y десятичных разрядов.  **MOD(x, y) -** Возвращает остаток от деления числа х нацело на y. Если y равно 0, то возвращается x.  **ABS(x)** возвращает абсолютное значение числа  **EXP(х)** используется для вычисления числа e  **COS(х)** вычисляет косинус заданного угла  **LN(х) -** Возвращает натуральный логарифм х.  **LOG(y, x)** вычисляет значение натурального логарифма  **SIN(х)**  **TAN(х)**  **SIGN(x)** определяет знак числа (отрицательное оно или положительное)  **SQRT(x)** вычисляет квадратный корень из числа  **ACOS(х) ASIN(х) -** Возвращает арккосинус и арксинус для х соответственно.  **ATAN(х) -** Возвращает арктангенс и тангенс х соответственно.  **CEIL(х) -** Возвращает наименьшее целое число, большее или равное х.  **POWER(x, y) -** Возвращает число x в степени y. Основание x и порядок y могут быть неположительными целыми числами, но если x - отрицательное число, то y должен быть целым числом. |
| Функции даты и времени: | **SYSDATE** возвращает текущую системную дату  **ADD\_MONTHS(d,x)** - Возвращает дату d плюс x месяцев.  **MONTHS\_BETWEEN(дата 1, дата 2)** - Возвращает число месяцев между "датой 1" и "датой 2". |
|  |  |
|  | **LAST\_DAY(d)** - Возвращает дату последнего дня того месяца, в который входит d.  **NEXT\_DAY(d, день\_недели)** - Возвращает дату первого "дня\_недели", наступающего после даты d.  **NEW\_TIME(d, пояс 1, пояс 2)** - Возвращает дату и время часового "пояса 2" для того момента, когда датой и временем часового "пояса 1" является d (Эта функция используется редко).  **ROUND(d, [, формат]) -** Округляет дату d до единицы указанной форматом.  **TRUNC(d, [, формат]) -** Возвращает дату d, усеченную до единицы, указанной в параметре "формат". |
| Функции обработки строк | **ASCII(строка) -** Возвращает десятичное представление первого байта "строка", согласно применяемому набору символов.  **CHR(x)** - Возвращает символ, имеющий код, равный x в наборе символов БД.  **INSTR(строка 1, строка 2, [,a[,b]])**  **LTRIM(строка 1, строка 2)** (функция удаляет не только пробелы, но и любые символы, заданные в параметре “строка 2”)  **RTRIM(строка 1, строка 2)** (функция удаляет не только пробелы, но и любые символы, заданные в параметре “строка 2”)  **LOWER(строка) -** Возвращает "строку" со строчными символами.  **UPPER(строка)**  **INITCAP(строка) -** Возвращает "строку", в которой каждое слово начинается с прописной буквы и продолжается строчными.  **LENGTH(строка)**  **REPLACE(строка\_символов, строка\_поиска, [строка\_замены])** |
|  | **SUBSTR(строка 1, a, [,b]) -** Возвращает часть "строки 1", начинающуюся с символа с номером a и имеющую длину b символов.  **CONCAT(строка 1, строка2)** - функция соединяет (конкатенирует) строки.  **LPAD(строка 1, х, строка 2) -** Возвращает "строку 1", дополненную слева до размера х символами "строки 2".  **RPAD(строка 1, х, строка 2) -** Возвращает "строку 1", дополненную справа до размера х символами "строка 2".  **TRANSLATE(строка\_символов, заменяемая\_строка, вносимая\_строка) -** Возвращает "строку\_символов", в которой все вхождения каждого символа из "заменяемой\_строки" замещаются соответствующим символом из "вносимой\_строки". |
| Функции преобразования и форматирования данных | **TO\_NUMBER(строка\_символов, [, формат [, nls\_параметр]])**  **TO\_DATE(строка\_символов, [, формат [, nls\_параметр]])**  **TO\_CHAR(число, [, формат[, nls\_параметр]])** |