1. Использование стандартных функций в PL/SQL.

*SQL Functions in PL/SQL*

Вы уже знакомы с функциями SQL. Вот пример использования одной из них:

*SELECT country\_name, LAST\_DAY(date\_of\_independence)*

*FROM wf\_countries*

*WHERE date\_of\_independence IS NOT NULL;* (речь идет о функции LAST\_DAY)

Вы также можете использовать эти функции в процедурах PL / SQL. Например:

*DECLARE*

*v\_last\_day DATE;*

*BEGIN*

*v\_last\_day := LAST\_DAY(SYSDATE);*

*DBMS\_OUTPUT.PUT\_LINE(v\_last\_day);*

*END;*

Функции, доступные в процедурах PL/SQL:

• Single-row character (символы/строки)

• Single-row number (числа)

• Date (дата)

• Data-type conversion (функции преобразования)

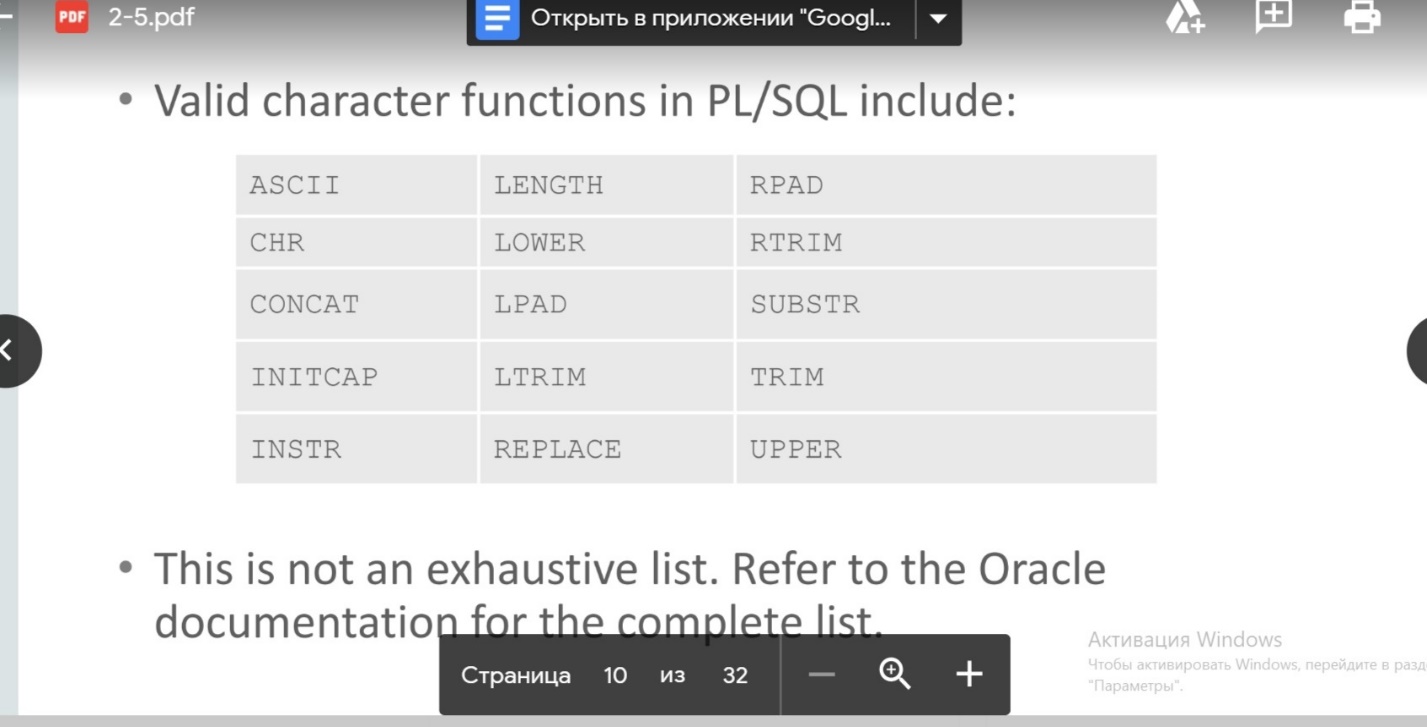
• Miscellaneous functions (иные функции)

Функции, недоступные в процедурах PL/SQL:

• DECODE (Декодирование)

• Group functions (Групповые функции)

Допустимые символьные функции в PL / SQL включают в себя:



**ПРИМЕРЫ:**

Получить длину строки:

*v\_desc\_size INTEGER(5);*

*v\_prod\_description VARCHAR2(70):='You can use this product with your*

*radios for higher frequency';*

*-- get the length of the string in prod\_description*

*v\_desc\_size:= LENGTH(v\_prod\_description);*

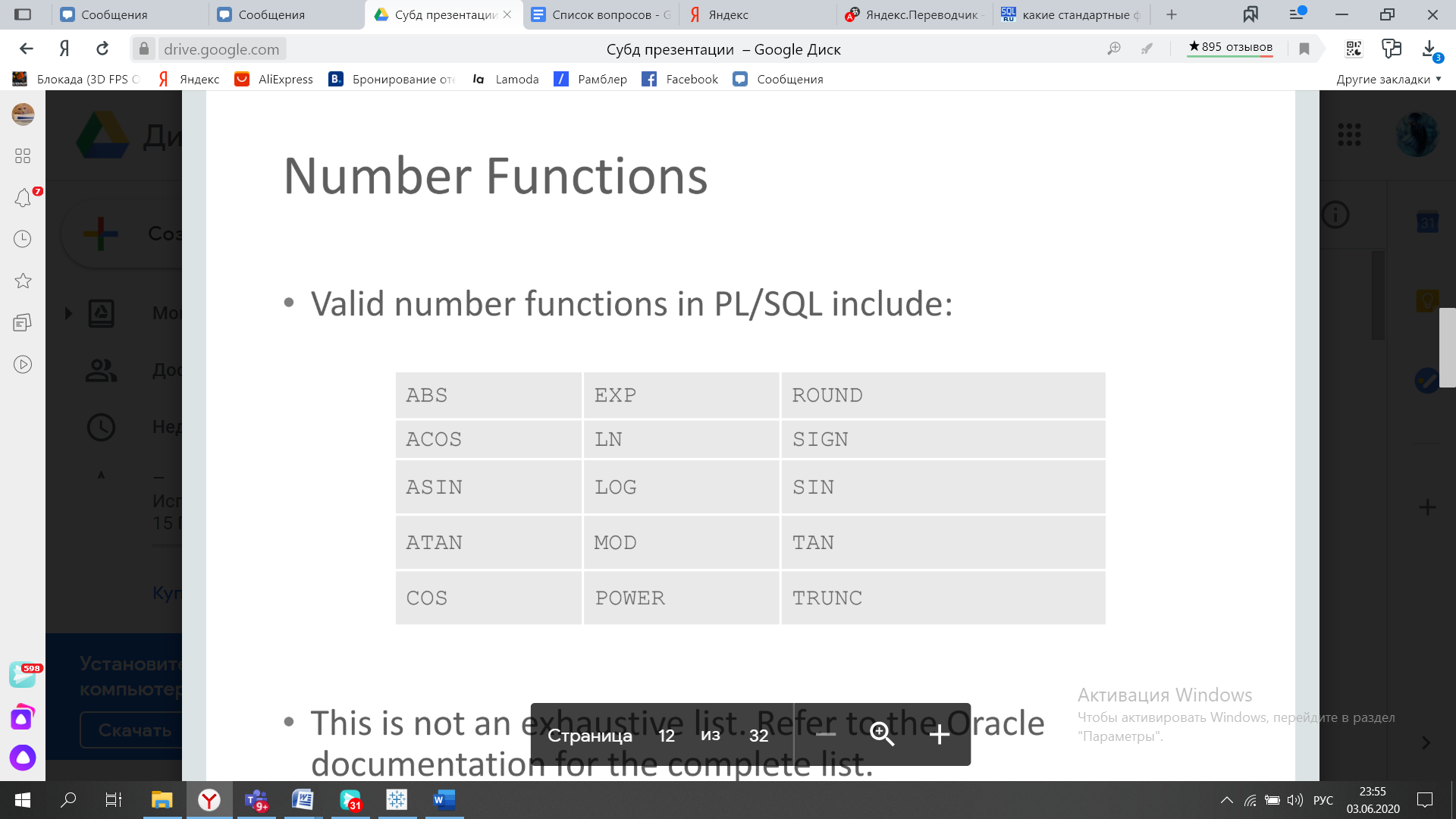
Название столицы страны в верхний регистр:

*v\_capitol\_name:= UPPER(v\_capitol\_name);*

Объединение строк:

*v\_emp\_name:= v\_first\_name||' '||v\_last\_name;*

Допустимые числовые функции в PL / SQL включают в себя:



**ПРИМЕРЫ:**

Получить знак числа:

*DECLARE*

*v\_my\_num BINARY\_INTEGER :=-56664;*

*BEGIN*

*DBMS\_OUTPUT.PUT\_LINE(SIGN(v\_my\_num));*

*END;*

Округление числа до 0 десятичных знаков:

*DECLARE*

*v\_median\_age NUMBER(6,2);*

*BEGIN*

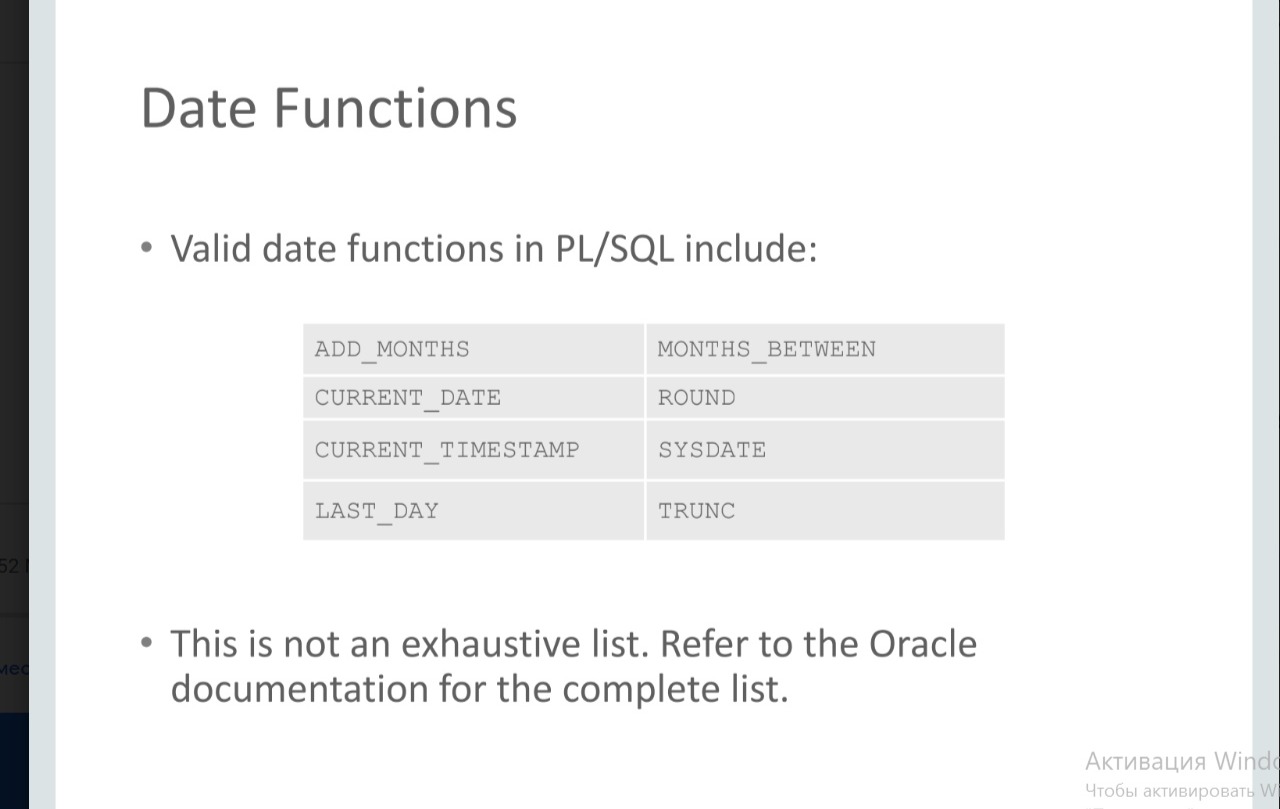
*SELECT median\_age INTO v\_median\_age*

*FROM wf\_countries WHERE country\_id=27;*

*DBMS\_OUTPUT.PUT\_LINE(ROUND(v\_median\_age,0));*

*END;*

Допустимые функции даты в PL / SQL включают в себя:



\*Везде на фотках строчка снизу есть, она означает, что это не весь список функций и надо чекать в Oracle. Но я думаю, что данного набора будет достаточно с головой.

**ПРИМЕРЫ:**

Добавить месяцы к дате:

*DECLARE*

*v\_new\_date DATE;*

*v\_num\_months NUMBER := 6;*

*BEGIN*

*v\_new\_date := ADD\_MONTHS(SYSDATE,v\_num\_months);*

*DBMS\_OUTPUT.PUT\_LINE(v\_new\_date);*

*END;*

Посчитать количество месяцев между двумя датами:

*DECLARE*

*v\_no\_months PLS\_INTEGER:=0;*

*BEGIN*

*v\_no\_months := MONTHS\_BETWEEN('31-JAN-2006','31-MAY-2005');*

*DBMS\_OUTPUT.PUT\_LINE(v\_no\_months);*

*END;*

Преобразование типов.

В любом языке программирования преобразование одного типа данных в другой это достаточно стандартное требование. PL / SQL может справиться с такими преобразованиями скалярных типов данных. Преобразование бывает **явным** и **неявным.**

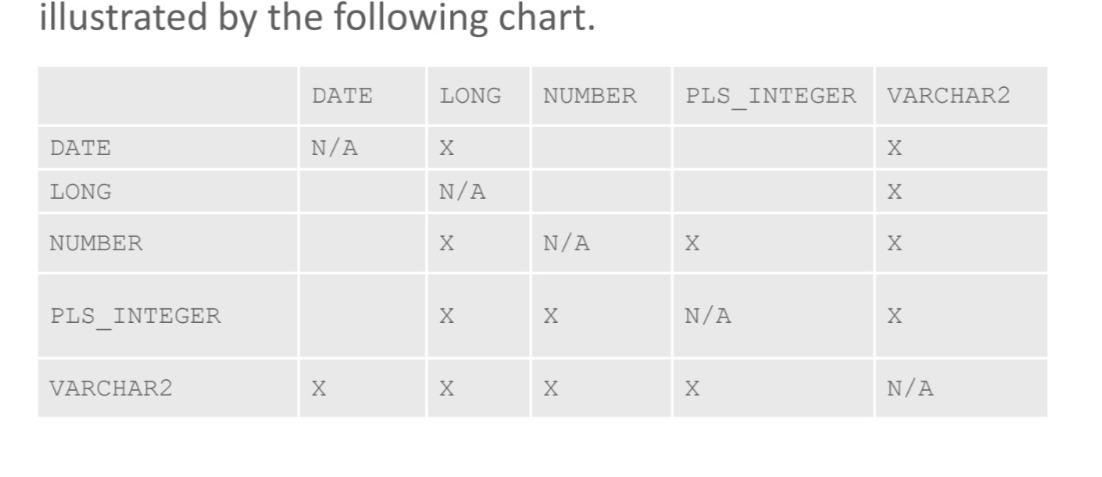
Неявное преобразование типов.

В неявных преобразованиях PL / SQL пытается преобразовать типы данных

динамически, если они смешаны в операторе. Неявный

преобразования могут происходить между многими типами в PL / SQL, как

проиллюстрировано следующей диаграммой:



В этом примере переменная v\_sal\_increase имеет тип

VARCHAR2. При расчете общей заработной платы сначала PL / SQL

преобразует v\_sal\_increase в NUMBER и затем выполняет операцию сложения. Результатом операции является тип числа:

*DECLARE*

*v\_salary NUMBER(6):=6000;*

*v\_sal\_increase VARCHAR2(5):='1000';*

*v\_total\_salary v\_salary%TYPE;*

*BEGIN*

*v\_total\_salary:= v\_salary + v\_sal\_increase;*

*DBMS\_OUTPUT.PUT\_LINE(v\_total\_salary);*

*END;*

Недостатки неявных преобразований

На первый взгляд неявные преобразования могут показаться полезными;

однако есть и несколько недостатков:

\* Неявные преобразования могут быть медленнее.

• Когда вы используете неявные преобразования, вы теряете контроль над своей программой, потому что вы делаете предположение о том, как

Oracle обрабатывает эти данные. Если Oracle изменит преобразование

правила, то ваш код может быть изменен.

\* Неявные правила преобразования зависят от среды в которой вы работаете.

– Например, формат даты варьируется в зависимости от языковой настройки

и тип установки. Код, использующий неявное преобразование, может не выполняться на другом сервере или на другом языке.

\* Код, использующий неявное преобразование, труднее читать и

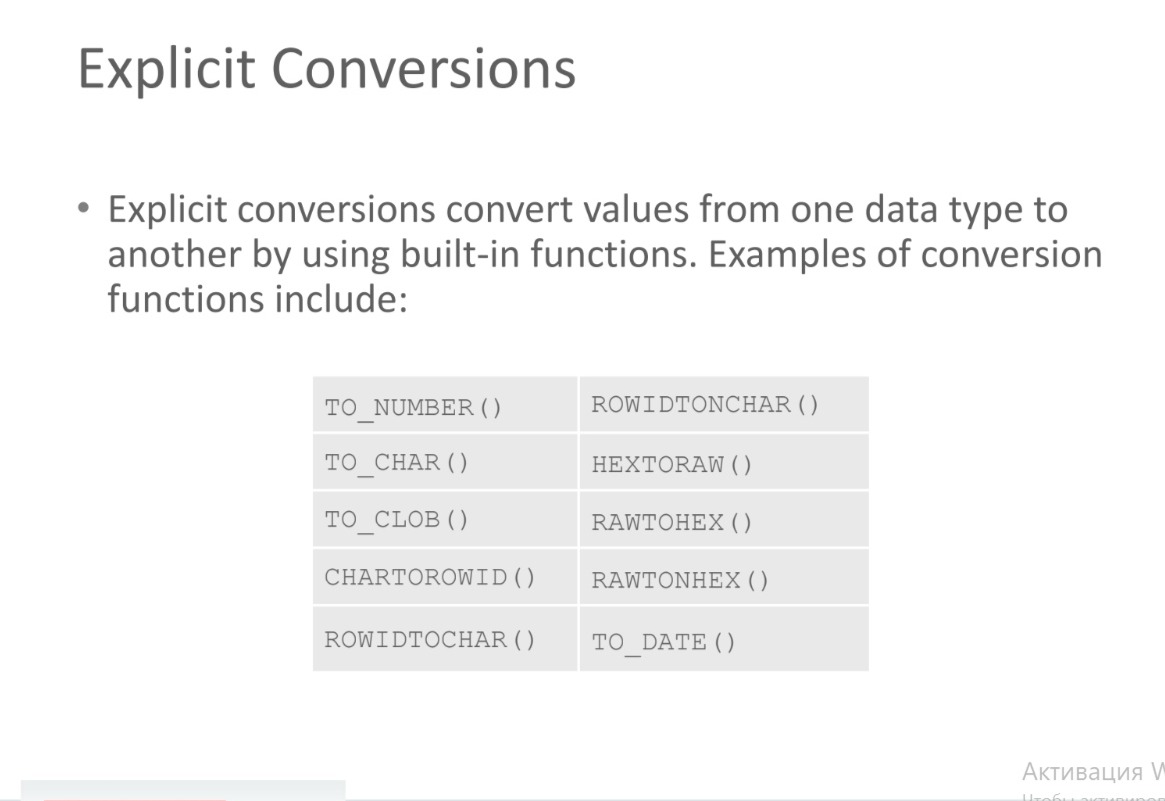
понимать.

\* Ответственность программиста заключается в том, чтобы гарантировать, что значения могут быть преобразованы. Например, PL / SQL может преобразовать CHAR значение '02-JUN-1992' в значение даты, но не может преобразовать значение CHAR ‘Yesterday ' в значение даты. Аналогично, PL / SQL не сможет преобразовать значение VARCHAR2, содержащее буквы или иные символы (не цифры), в числовое значение.

Явное преобразование типов.

Явные преобразования преобразуют значения из одного типа данных в другой с помощью встроенных функций. Примеры преобразования

функции включают в себя:



**ПРИМЕРЫ:**

• TO\_CHAR

*BEGIN*

*DBMS\_OUTPUT.PUT\_LINE(TO\_CHAR(SYSDATE,'Month YYYY'));*

*END;*

\* TO\_DATE

*BEGIN*

*DBMS\_OUTPUT.PUT\_LINE(TO\_DATE('April-1999','Month-YYYY'));*

*END;*

\* TO\_NUMBER

*DECLARE*

*v\_a VARCHAR2(10) := '-123456';*

*v\_b VARCHAR2(10) := '+987654';*

*v\_c PLS\_INTEGER;*

*BEGIN*

*v\_c := TO\_NUMBER(v\_a) + TO\_NUMBER(v\_b);*

*DBMS\_OUTPUT.PUT\_LINE(v\_c);*

*END;*

Примеры преобразования типа CHAR --- DATE:

*v\_date\_of\_joining DATE:= '02-Feb-2000';*

*v\_date\_of\_joining DATE:= 'February 02,2000';*

*v\_date\_of\_joining DATE:= TO\_DATE('February 02,2000','Month DD,YYYY');*