Федеральное государственное образовательное бюджетное

учреждение высшего образования

**«Финансовый университет при Правительстве Российской Федерации»**

**(Финансовый университет)**

Факультет «Информационных технологий и анализа больших данных»

**ОТЧЕТ**

**по практической работе № 7**

**по теме «Функции и процедуры. Язык PL/pgSQL»**

**по дисциплине Б.1.1.3.4 Системы управления базами данных**

**Вариант 1**

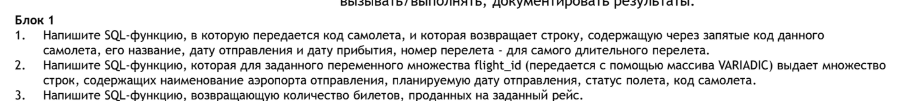
Выполнил: студент группы ИД23-1

Маслов Александр Николаевич

Проверил преподаватель: Цветков И.Е.

**Москва – 2025**

**Задание №1**

****

**Решение:**

**1.**

CREATE OR REPLACE FUNCTION get\_aircraft\_longest\_flight\_info(ac\_code CHAR(3))

RETURNS TEXT AS $$

DECLARE

result TEXT;

BEGIN

SELECT CONCAT\_WS(', ',

f.aircraft\_code,

ad.model,

f.scheduled\_departure,

f.scheduled\_arrival,

f.flight\_no)

INTO result

FROM bookings.flights f

JOIN bookings.aircrafts\_data ad ON f.aircraft\_code = ad.aircraft\_code

WHERE f.aircraft\_code = ac\_code

ORDER BY (f.scheduled\_arrival - f.scheduled\_departure) DESC

LIMIT 1;

RETURN result;

END;

$$ LANGUAGE plpgsql;

**Проверка:**

SELECT bookings.get\_aircraft\_longest\_flight\_info(

'319'

)

Изображение выглядит как текст, Шрифт, снимок экрана, линия

Контент, сгенерированный ИИ, может содержать ошибки.

**2.**

CREATE OR REPLACE FUNCTION get\_flights\_info(VARIADIC flight\_ids INTEGER[])

RETURNS TABLE (

departure\_airport\_name TEXT,

scheduled\_departure TIMESTAMPTZ,

flight\_status VARCHAR,

aircraft\_code CHAR(3)

) AS $$

BEGIN

RETURN QUERY

SELECT

(a.airport\_name ->> 'en')::TEXT,

f.scheduled\_departure,

f.status,

f.aircraft\_code

FROM bookings.flights f

JOIN bookings.airports\_data a ON f.departure\_airport = a.airport\_code

WHERE f.flight\_id = ANY(flight\_ids);

END;

$$ LANGUAGE plpgsql;

**Проверка:**

SELECT bookings.get\_flights\_info(

550,10000,23500

)

Изображение выглядит как текст, Шрифт, программное обеспечение, линия

Контент, сгенерированный ИИ, может содержать ошибки.

**3.**

CREATE OR REPLACE FUNCTION get\_ticket\_count(flight\_id\_input INTEGER)

RETURNS INTEGER AS $$

DECLARE

count\_result INTEGER;

BEGIN

SELECT COUNT(\*)

INTO count\_result

FROM bookings.boarding\_passes

WHERE flight\_id = flight\_id\_input;

RETURN count\_result;

END;

$$ LANGUAGE plpgsql;

**Проверка:**

SELECT bookings.get\_ticket\_count(

239

)

Изображение выглядит как текст, Шрифт, число, линия

Контент, сгенерированный ИИ, может содержать ошибки.

**Задание №2**

**Изображение выглядит как текст, Шрифт, снимок экрана, документ

Контент, сгенерированный ИИ, может содержать ошибки.**

**Решение:  
1.**

CREATE TABLE bookings\_copy AS TABLE bookings.bookings;

CREATE OR REPLACE PROCEDURE increase\_total\_amount(percentage NUMERIC)

LANGUAGE plpgsql

AS $$

BEGIN

UPDATE bookings\_copy

SET total\_amount = total\_amount + (total\_amount \* percentage / 100);

END;

$$;

**Проверка:**

CALL increase\_total\_amount(10);

Изображение выглядит как текст, снимок экрана, Шрифт, число

Контент, сгенерированный ИИ, может содержать ошибки.

До выполнения процедуры.

Изображение выглядит как текст, снимок экрана, Шрифт, число

Контент, сгенерированный ИИ, может содержать ошибки.  
После выполнения процедуры.

**2.**CREATE OR REPLACE FUNCTION get\_passenger\_info(

ticket CHAR(13),

OUT passenger\_id VARCHAR,

OUT full\_name TEXT,

OUT contact JSONB

)

AS $$

BEGIN

SELECT

t.passenger\_id,

t.passenger\_name,

t.contact\_data

INTO

passenger\_id,

full\_name,

contact

FROM bookings.tickets t

WHERE t.ticket\_no = ticket;

END;

$$ LANGUAGE plpgsql;

**Проверка:**

SELECT \* FROM get\_passenger\_info('0005432000987');

**Изображение выглядит как текст, снимок экрана, Шрифт, линия

Контент, сгенерированный ИИ, может содержать ошибки.**

**3.**

CREATE OR REPLACE FUNCTION sum\_variadic(VARIADIC nums NUMERIC[])

RETURNS NUMERIC AS $$

DECLARE

total NUMERIC := 0;

n NUMERIC;

BEGIN

FOREACH n IN ARRAY nums LOOP

total := total + n;

END LOOP;

RETURN total;

END;

$$ LANGUAGE plpgsql;

**Проверка:**

SELECT sum\_variadic( 5, 3, 10)

Изображение выглядит как текст, Шрифт, линия, Прямоугольник

Контент, сгенерированный ИИ, может содержать ошибки.

**4.**

CREATE OR REPLACE FUNCTION get\_seats\_by\_fare(fare\_cond VARCHAR)

RETURNS TABLE (

aircraft\_code CHAR(3),

seat\_no VARCHAR,

fare\_conditions VARCHAR

) AS $$

BEGIN

RETURN QUERY

SELECT

s.aircraft\_code,

s.seat\_no,

s.fare\_conditions

FROM bookings.seats s

WHERE s.fare\_conditions = fare\_cond;

END;

$$ LANGUAGE plpgsql;

**Проверка:**

SELECT get\_seats\_by\_fare('Business')

Изображение выглядит как текст, снимок экрана, Шрифт, число

Контент, сгенерированный ИИ, может содержать ошибки.

**5.**

CREATE OR REPLACE FUNCTION get\_seats\_by\_fare\_record(fare\_cond VARCHAR)

RETURNS SETOF RECORD AS $$

BEGIN

RETURN QUERY

SELECT \*

FROM bookings.seats

WHERE fare\_conditions = fare\_cond;

END;

$$ LANGUAGE plpgsql;

**Проверка:**

SELECT \* FROM get\_seats\_by\_fare\_record('Economy') AS (

aircraft\_code CHAR(3),

seat\_no VARCHAR,

fare\_conditions VARCHAR

);

**Изображение выглядит как текст, снимок экрана, число, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.**

**Задание №3**

**Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.**

**Решение:**

**1.**

CREATE OR REPLACE FUNCTION nested\_variables\_demo()

RETURNS TEXT AS $$

DECLARE

var\_name TEXT := 'Уровень 1';

BEGIN

RAISE NOTICE 'Внешний блок: %', var\_name;

DECLARE

var\_name TEXT := 'Уровень 2';

BEGIN

RAISE NOTICE 'Вложенный блок 1: %', var\_name;

DECLARE

var\_name TEXT := 'Уровень 3';

BEGIN

RAISE NOTICE 'Вложенный блок 2: %', var\_name;

var\_name := 'Изменено на уровне 3';

RAISE NOTICE 'Вложенный блок 2 (после изменения): %', var\_name;

END;

RAISE NOTICE 'Вложенный блок 1 (после вложенного блока 2): %', var\_name;

var\_name := 'Изменено на уровне 2';

RAISE NOTICE 'Вложенный блок 1 (после изменения): %', var\_name;

END;

RAISE NOTICE 'Внешний блок (после всех вложенных блоков): %', var\_name;

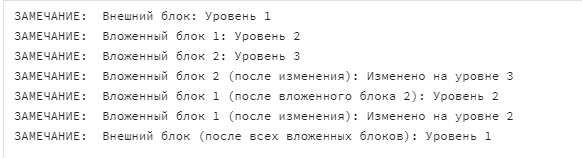
RETURN 'Демонстрация завершена';

END;

$$ LANGUAGE plpgsql;

**Проверка:**

SELECT nested\_variables\_demo();



**2.**

**Проверка:**

**3.**

**Проверка:**

**4.**

**Проверка:**

**5.**

**Проверка:**