**1. Тип tsvector**

Выполните запрос

SELECT to\_tsvector('The quick brown fox jumped over the lazy dog.');

В ответ будет возвращён список [лексем](https://en.wikipedia.org/wiki/Lexeme)

to\_tsvector

'brown': 3 'dog': 9 'fox': 4 'jump': 5 'lazy': 8 'quick': 2

Задание 1

1. Изучите документацию к функции to\_tsvector

*to\_tsvector разбирает текстовый документ на фрагменты, сводит фрагменты к лексемам и возвращает значение tsvector, в котором перечисляются лексемы и их позиции в документе. При обработке документа используется указанная конфигурация текстового поиска или конфигурация по умолчанию.*

1. Вызовите эту функцию для следующей строки: Съешь ещё этих мягких французских булок, да выпей чаю

*'булок':6 'вып':8 'ещ':2 'мягк':4 'съеш':1 'французск':5 'ча':9 'эт':3*

1. Почему в векторе нет слова да?

*Слово «Да» распознается словарем как стоп-слово.*

**2. Тип tsquery**

Выполните по очереди запросы

-- №1

SELECT to\_tsvector('The quick brown fox jumped over the lazy dog')

@@ to\_tsquery('fox');

***Ответ: True***

-- №2

SELECT to\_tsvector('The quick brown fox jumped over the lazy dog')

@@ to\_tsquery('foxes');

***Ответ: True***

-- №3

SELECT to\_tsvector('The quick brown fox jumped over the lazy dog')

@@ to\_tsquery('foxhound');

***Ответ: False***

Задание 2

1. Что означают символы @@

*Оператор @@ - используется для сопоставления*

1. Почему второй запрос возвращает true, а третий не возвращает

*Так как один из словарей понимает, что слово foxes – это слово fox во множественном числе. Слово foxhound* *не относится к этой лексеме.*

1. Выполните запрос

SELECT to\_tsvector('Russian', 'Съешь ещё этих мягких французских булок, да выпей чаю.')

@@ to\_tsquery('Russian','булка');

Почему слово булка не находится?

*Потому что в словаре слова булок и булка относятся к разным лексемам.*

4. Используйте функцию select ts\_lexize('russian\_stem', 'булок'); для того чтобы понять почему.

5. Замените в предложении слово булок, на слово пирожков Выполните запросы

--№1

SELECT to\_tsvector('Russian', 'Съешь ещё этих мягких французских пирожков, да выпей чаю.')

@@ to\_tsquery('Russian','пирожки');

--№2

SELECT to\_tsvector('Russian', 'Съешь ещё этих мягких французских пирожков, да выпей чаю.')

@@ to\_tsquery('Russian','пирожок');

Почему первый запрос возвращает true, а второй не возвращает?

*Потому что в словаре слова пирожков и пирожок относятся к разным лексемам.*

**3. Операторы**

Выполните запрос

-- И

SELECT to\_tsvector('The quick brown fox jumped over the lazy dog')

@@ to\_tsquery('fox & dog');

***Ответ: True***

- ИЛИ

SELECT to\_tsvector('The quick brown fox jumped over the lazy dog')

@@ to\_tsquery('fox | rat');

***Ответ: True***

- отрицание

SELECT to\_tsvector('The quick brown fox jumped over the lazy dog')

@@ to\_tsquery('!clown');

***Ответ: True***

- группировка

SELECT to\_tsvector('The quick brown fox jumped over the lazy dog' )

@@ to\_tsquery('fox & (dog | rat) & !mice');

***Ответ: True***

Задание 3

1. Приведите аналогичные запросы для любого предложения на русском

И

SELECT to\_tsvector('Russian','Защита диплома в УрФУ состоится не смотря ни на что')

@@ to\_tsquery('Диплома & УрФУ');

ИЛИ  
SELECT to\_tsvector('Russian','Защита диплома в УрФУ состоится не смотря ни на что')  
@@ to\_tsquery('Диплома | СИНХ');

Отрицание  
SELECT to\_tsvector('Russian','Защита диплома в УрФУ состоится не смотря ни на что')  
@@ to\_tsquery('!СИНХ');

Группировка  
SELECT to\_tsvector('Russian','Защита диплома в УрФУ состоится не смотря ни на что')  
@@ to\_tsquery('Состоится & (Диплома | СИНХ)');

1. Почему для английского языка не нужно указывать язык в первом аргументе и какой анализатор используется если никакой не указан?

*Если аргумент не определен, то используется конфигурация по умолчанию, записанная в параметре default\_text\_search\_config в файле postgresql.conf. Встроенное значение по умолчанию для данного параметра — pg\_catalog.simple, но initdb при инициализации записывает в файл конфигурации сервера значение, соответствующее выбранной локали lc\_ctype, если удастся найти такую конфигурацию текстового поиска.*

**4. Поиск фраз**

Изучите документацию по [операторам](https://www.postgresql.org/docs/current/functions-textsearch.html) FTS Выполните запрос

SELECT to\_tsvector('Russian', 'Съешь ещё этих мягких французских булок, да выпей чаю.')

@@ to\_tsquery('Russian','мягких<2>булок');

Задание 4

1. Что означает число 2 в операторе <->

*Оператор <-> показывает, предшествует ли одно слово другому.  
Число 2 в операторе <-> показывает расстояние между словами. В запросе мы хотим узнать, стоит ли слово мягких от слова булок в 2 шагах. В данном случаем результат запроса будет true.*

1. Модифицируйте запрос так, чтобы можно было найти фразу съешь ещё

*SELECT to\_tsvector('Russian', 'Съешь ещё этих мягких французских булок, да выпей чаю.') @@ to\_tsquery('Russian','Съешь<->ещё');*

1. Для чего нужно использовать функцию phraseto\_tsquery

*Для поиска фраз, игнорируя знаки препинания*

**5. Утилиты**

**ts\_debug**  
С помощью функции **ts\_debug** можно легко протестировать конфигурацию текстового поиска.  
**ts\_debug** выводит информацию обо всех фрагментах данного документа, которые были выданы анализатором и обработаны настроенными словарями.  
**ts\_debug** возвращает по одной строке для каждого фрагмента, найденного в тексте анализатором. Эта строка содержит следующие столбцы:

* синоним text — краткое имя типа фрагмента;
* описание text — описание типа фрагмента;
* фрагмент text — текст фрагмента;
* словари regdictionary[] — словари, назначенные в конфигурации для фрагментов такого типа;
* словарь regdictionary — словарь, распознавший этот фрагмент, или NULL, если подходящего словаря не нашлось;
* лексемы text[] — лексемы, выданные словарём, распознавшим фрагмент, или NULL, если подходящий словарь не нашёлся; может быть также пустым массивом ({}), если фрагмент распознан как стоп-слово.

Например, выполнив запрос

SELECT ts\_debug('Russian', 'Съешь ещё этих мягких французских булок, да выпей чаю.');

можно увидеть следующий результат

(word,"Word, all letters",Съешь,{russian\_stem},russian\_stem,{съеш})

(blank,"Space symbols"," ",{},,)

(word,"Word, all letters",ещё,{russian\_stem},russian\_stem,{ещ})

(blank,"Space symbols"," ",{},,)

(word,"Word, all letters",этих,{russian\_stem},russian\_stem,{эт})

(blank,"Space symbols"," ",{},,)

(word,"Word, all letters",мягких,{russian\_stem},russian\_stem,{мягк})

(blank,"Space symbols"," ",{},,)

(word,"Word, all letters",французских,{russian\_stem},russian\_stem,{французск})

(blank,"Space symbols"," ",{},,)

(word,"Word, all letters",булок,{russian\_stem},russian\_stem,{булок})

(blank,"Space symbols",", ",{},,)

(word,"Word, all letters",да,{russian\_stem},russian\_stem,{})

(blank,"Space symbols"," ",{},,)

(word,"Word, all letters",выпей,{russian\_stem},russian\_stem,{вып})

(blank,"Space symbols"," ",{},,)

(word,"Word, all letters",чаю,{russian\_stem},russian\_stem,{ча})

(blank,"Space symbols",.,{},,)

**ts\_headline**  
Функция **ts\_headline** предоставляет возможность показать фрагменты документа с отмеченными искомыми словами. **ts\_headline** принимает документ вместе с запросом и возвращает выдержку из документа, в которой выделяются слова из запроса.  
Например, результатом запроса

SELECT ts\_headline('Russian', 'Гарри вскочил с постели. Счастье распирало его изнутри, словно он проглотил воздушный шар. Гарри подошел к окну и распахнул его. Сова влетела в комнату и уронила газету прямо на Хагрида, но тот не проснулся. Затем сова спикировала на пол и набросилась на куртку Хагрида.',

to\_tsquery('сова & газета & счастье'), 'StartSel = <, StopSel = >');

где символами <> будут выделены искомые слова.