

ЗАЩИЩЕННАЯ СИСТЕМА УПРАВЛЕНИЯ
БАЗАМИ ДАННЫХ «ЈАТОВА»

Руководство по настройке. Часть 6.
Формирование отчетов производительности СУБД.
Компонент «pg_Profile»

643.72410666.00067-07 98 01-06

Листов 78

Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

АННОТАЦИЯ

В документе приведены сведения, необходимые для настройки и использования компонента pg_Profile, который позволяет собирать и просматривать параметры и метрики системы управления базами данных «Jatoba» (далее – СУБД «Jatoba»). Настоящее руководство предназначено для администратора СУБД «Jatoba».

Администратор СУБД «Jatoba» должен иметь навыки по работе с системами управления базами данных (СУБД) PostgreSQL или защищенной СУБД «Jatoba» (ООО «Газинформсервис»).



Все примеры в данном документе приведены для СУБД «Jatoba» версии ядра 5.x, для других версий все шаги выполняются аналогично, разница состоит в именах директорий.

Например, СУБД «Jatoba» версии 6.x по умолчанию устанавливается в директорию:

- ОС Windows – «C:\Program Files\GIS\Jatoba\6\bin»;
- ОС Linux – «/usr/jatoba-6/bin».

Для СУБД «Jatoba» версии ядра 4/5/6 используется версия компонента — 4.10



Важная информация

Для сертифицированной версии СУБД «Jatoba» поддерживается работа только на ОС, указанных в формуляре на поставку!

Степени важности примечаний, применяемые в документе:

№ изменения: _____	Подпись отв. лица: _____	Дата внесения изм: _____
--------------------	--------------------------	--------------------------



Важная информация – указания, требующие особого внимания



Дополнительная информация – указания, позволяющие упростить работу с изделием

СОДЕРЖАНИЕ

1. Назначение компонента.....	7
1.1. Функциональные возможности	7
1.2. Условия применения.....	8
2. Установка.....	9
2.1. Установка компонента на ОС Windows.....	9
2.2. Установка компонента ОС GNU/Linux	10
2.3. Рекомендованные настройки postgresql.conf.....	11
2.4. Установка расширения pg_Profile.....	12
2.4.1. Установка расширений в БД «postgres».....	12
2.4.2. Установка расширений в служебную БД.....	13
2.4.3. Установка внешнего сервера для служебной БД.....	16
2.5. Дополнительные настройки «postgresql.conf».....	20
3. Использование	21
3.1. Управление соединениями с серверами	21
3.2. Сбор данных о размерах страниц	22
3.3. Снимки состояния БД	23
3.3.1. Снятие снапшотов	25
3.3.2. Время жизни снапшотов	26
3.3.3. Описание событий.....	26
3.4. Baseline	27
3.4.1. Функции управления baseline	27
3.5. Экспорт / импорт данных.....	29
3.5.1. Экспорт данных.....	29
3.5.2. Импорт данных.....	30
3.6. Создание отчетов	31
3.6.1. Обычные отчеты.....	31
3.6.2. Дифференциальные отчеты.....	32
4. Данные в отчетах	34
4.1. Server statistics (Статистика сервера).....	34
4.1.1. Database statistics (Статистика базы данных).....	34
4.1.2. Cluster I/O statistics (Статистика ввода/вывода кластера)	35
4.1.3. Cluster SLRU statistics (Статистика доступа к SLRU-кешам)	38
4.1.4. Statement statistics by database (Статистика по обращениям к базе данных)	38
4.1.5. Cluster statistics (Статистика кластера)	39
4.1.6. WAL statistics (Статистика WAL)	41
4.1.7. Tablespace statistics (Статистика табличных пространств).....	42
4.2. SQL Query statistics (Статистика SQL-запросов).....	42

4.2.1. Top SQL by execution time (Топ SQL-запросов по времени выполнения)	42
4.2.2. Top SQL by mean execution time (Топ SQL-запросов по среднему времени выполнений)	43
4.2.3. Top SQL by executions (Топ SQL-запросов по количеству выполнений)	45
4.2.4. Top SQL by I/O wait time (Топ SQL-запросов по времени ожидания ввода/вывода)	46
4.2.5. Top SQL by shared blocks fetched (Топ SQL-запросов по выбранным общим блокам)	47
4.2.6. Top SQL by shared blocks read (Топ SQL-запросов по количеству прочитанных разделяемых блоков)	47
4.2.7. Top SQL by shared blocks dirtied (Топ SQL-запросов по заполненным разделяемым блокам)	48
4.2.8. Top SQL by shared blocks written (Топ SQL-запросов по записи общих блоков)	49
4.2.9. Top SQL by WAL size (Топ SQL-запросов по размеру WAL)	50
4.2.10. Complete list of SQL texts (Полный список текстов SQL)	50
4.3. Schema object statistics (Статистика объекта схемы)	51
4.3.1. Top tables by estimated sequentially scanned volume (Топ таблиц по предполагаемому объему последовательного сканирования)	51
4.3.2. Top tables by blocks fetched (Топ таблиц по выбранным блокам)	52
4.3.3. Top tables by blocks read (Топ таблиц по прочитанным блокам)	53
4.3.4. Top DML tables (Топ таблиц по количеству операций DML)	54
4.3.5. Top tables by updated/deleted tuples (Топ таблиц по обновленным/удаленным записям)	54
4.3.6. Top tables by new-page updated tuples (Таблицы с наибольшим количеством изменённых кортежей, попавших на новую страницу)	55
4.3.7. Top growing tables (Топ таблиц по увеличению размера)	56
4.3.8. Top indexes by blocks fetched (Топ индексов по выбранным блокам)	57
4.3.9. Top indexes by blocks read (Топ индексов по прочитанным блокам)	57
4.3.10. Top growing indexes (Топ таблиц по увеличению объемов индексов)	58
4.3.11. Unused indexes (Неиспользуемые индексы)	59
4.4. User function statistics (Статистика функций пользователя)	59
4.4.1. Top functions by total time (Топ функций по общему времени)	60
4.4.2. Top functions by executions (Топ функций по исполнению)	60
4.5. Vacuum-related stats (Статистика, связанная с вакуумом)	61
4.5.1. Top tables by vacuum operations (Топ таблиц по статистике вакуума)	61
4.5.2. Top tables by analyze operations (Топ таблиц по операциям анализа)	62
4.5.3. Top tables by vacuum time spent	63
4.5.4. Top tables by analyze time spent (Топ таблиц, на анализ которых затрачено больше всего времени)	63
4.5.5. Top indexes by estimated vacuum load (Топ индексов, провоцирующие наибольшую нагрузку при очистке)	64
4.6. Cluster settings during the report interval (Настройка кластера во время отчетного интервала)	65
4.7. Отчеты по компоненту «ja_Hipe_Cluster» (Citrus)	66
4.7.1. Nodes	66
4.7.2. Connectivity between all nodes	66

4.7.3. Tables (citius_tables)	67
4.7.4. Shards	67
4.7.5. Blocked queries (citius_lock_waits).....	68
4.7.6. Query statistics (citius_stat_statements)	68
4.7.7. Rebalance progress (get_rebalance_progress)	69
4.7.8. Configuration parameters.....	69
4.7.9. Active tenants (citius_stat_tenants)	70
5. Дополнительная информация.....	71
6. Удаление компонента	72
6.1. Удаление компонента в ОС GNU/Linux	72
6.1.1. Удаление расширений компонента.....	72
6.1.2. Удаление пакета компонента	72
7. Обновление компонента	73
7.1. Обновление компонента в ОС GNU/Linux	73
Термины и определение	74
Перечень сокращений.....	77

1. НАЗНАЧЕНИЕ КОМПОНЕНТА

pg_Profile – компонент, позволяющий собирать и просматривать параметры и метрики функционирования БД в разное время, а также строить отчеты по этим данным и сравнивать их между собой для выявления проблемных мест.

Для сбора параметров и метрик БД используется система снимков.

Анализ снимков состояния позволяет выявить проблемные участки путем просмотра или сравнения данных из разных снимков.

Снимки можно снимать как с текущей БД, так и с любых других БД, к которым есть доступ.

pg_Profile допускает экспортировать и импортировать собранную статистику в свои служебные таблицы.

Компонент реализован в формате расширения для СУБД PostgreSQL/Jatoba, установка которого регламентирована соответствующими правилами СУБД.



Для установки расширения необходимо заранее установить другие дополнительные расширения, на работе которых основана работа pg_profile, а именно – plpgsql и dblink.

Для получения данных в разделе отчета rusage_statistics необходимо установить расширение pg_stat_kcache и включить его параметр pg_stat_kcache.track_planning = 'on' в postgresql.conf. Если эти данные не нужны, то расширение можно не устанавливать.

1.1. Функциональные возможности

Компонент автоматизации работы администратора СУБД обладает следующими функциональными возможностями:

- 1) выявление и анализ наиболее ресурсоемких операций и SQL-запросов;
- 2) гибкое формирование итоговых отчетов по накопленной статистической информации.

Компонент предоставляет для просмотра информацию о SQL-запросах, выполнение которых по разным характеристикам занимало больше всего времени. Количество таких

№ изменения: _____	Подпись отв. лица: _____	Дата внесения изм: _____
--------------------	--------------------------	--------------------------

запросов в формируемых отчетах определяется соответствующим параметром компонента, описанным в разделе 2.5.

Компонент обеспечивает сбор и хранение одномоментной и ретроспективной дискретной статистической информации о работе отдельных компонентов СУБД при выполнении SQL-запросов. Снятие статистической информации само по себе является затратной операцией, которая может вызывать блокировку отдельных объектов СУБД и влиять на производительность работы пользователей. В компоненте предусмотрен соответствующий механизм, основанный на использовании параметра СУБД `lock_timeout`, прерывающий построение снимка, который занимает более 3 секунд времени.

1.2. Условия применения

Компонент `pg_Profile` может использоваться совместно с СУБД «Jatoba» версий 1.x – 5.x.

2. УСТАНОВКА

2.1. Установка компонента на ОС Windows

Компонент устанавливается в составе СУБД «Jatoba» под управлением ОС Windows при первичной установке.

а) в окне «Выбор типа установки» следует выбрать тип установки «Выборочная» (см. рис. 2.1);

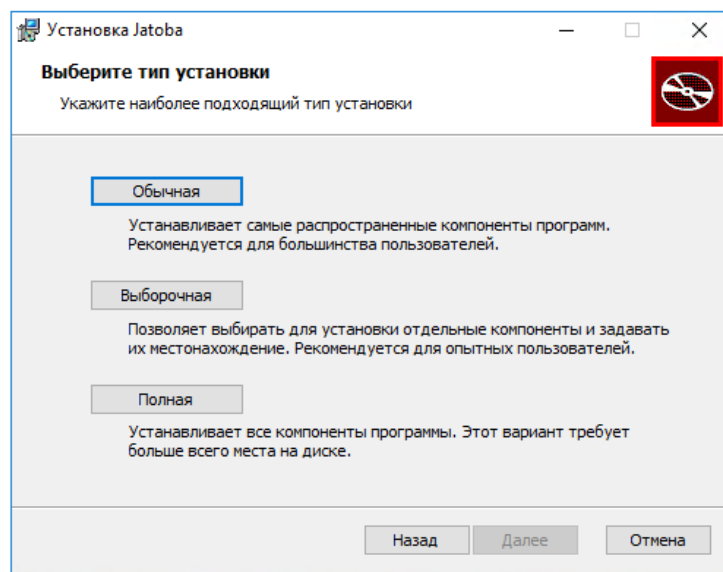


Рисунок 2.1 – Окно выбора типа установки

б) в окне «Выборочная установка» выбрать «Профилирование запросов (pg_profile)» (см. рис. 2.2);

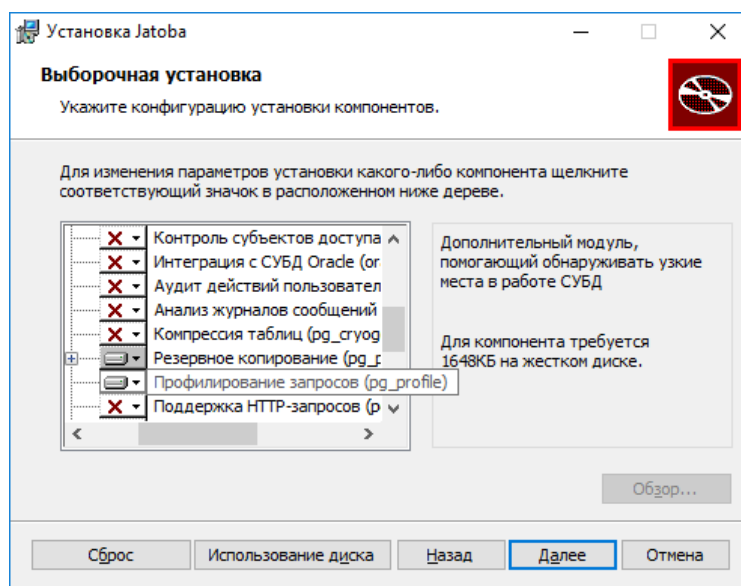


Рисунок 2.2 – Выбор устанавливаемых компонент

в) в открывшемся окне «Все готово к установке Jatoba» запустить процесс установки, нажав кнопку «Установить» (см. рис. 2.3);

№ изменения: _____	Подпись отв. лица: _____	Дата внесения изм: _____
--------------------	--------------------------	--------------------------

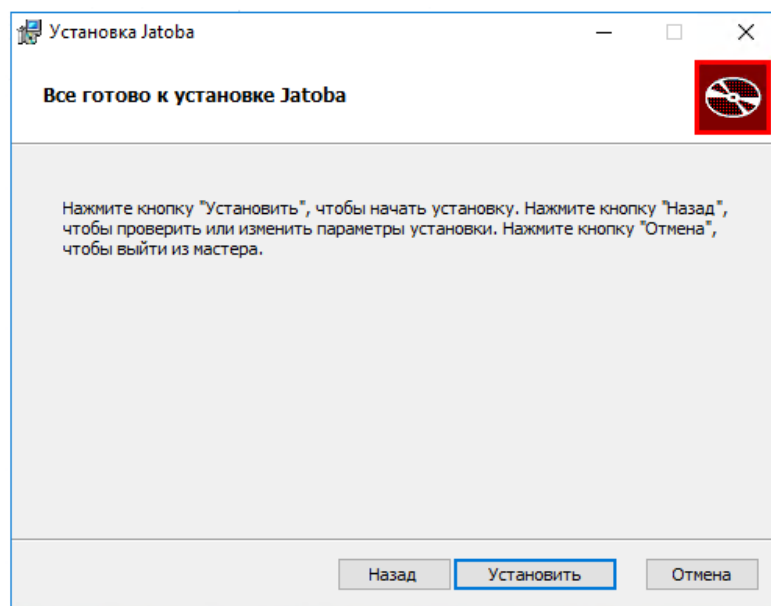


Рисунок 2.3 – Окно «Все готово к установке Jatoba»

2.2. Установка компонента ОС GNU/Linux

Компонент устанавливается в составе СУБД «Jatoba». Его возможно установить при первичной установке, либо доустановить.

Установку компонента возможно провести двумя способами:

- 1) установка из локального репозитория (CDROM) – производится из файлов, записанных на компакт-диск или скопированных с него;
- 2) установка непосредственно из deb/rpm-файлов – производится опционально, по усмотрению пользователя.

Компонент выполнен в виде отдельного deb или rpm-пакета. Установка компонента осуществляется средствами пакетного менеджера ОС. Для разных типов пакетных менеджеров команда установки немного отличается. Ниже приведены основные типы:

– для систем на основе пакетного менеджера APT (к таким системам относятся все ОС семейства Debian, использующие deb-пакеты) команда установки следующая:

```
apt-get install jatoba<ver>-pg-profile
```

– для систем на основе пакетных менеджеров YUM/DNF (к таким системам относятся все ОС семейства RedHat и вышедшие из нее, использующие rpm-пакеты) команда установки следующая:

```
yum install jatoba<ver>-pg_profile
```

Отдельного уточнения требуют операционные системы ALT Linux и openSUSE.

– ALT Linux использует пакетный менеджер APT, но распространяется в виде rpm-пакетов и для нее команда установки выглядит аналогично Debian:

```
apt-get install jatoba<ver>-pg-profile
```

Установка компонента в составе других версий СУБД «Jatoba» осуществляется аналогично. Отличие будет только в номере версии СУБД, в составе которой он распространяется. Например, jatoba3-pg-profile и т.п.

2.3. Рекомендованные настройки postgresql.conf

Для корректного функционирования расширения в разделе «Shared Library Preloading» в конфигурационном файле postgresql.conf необходимо добавить строку (рисунок 2.4):

```
shared_preload_libraries = 'pg_stat_statements'
```

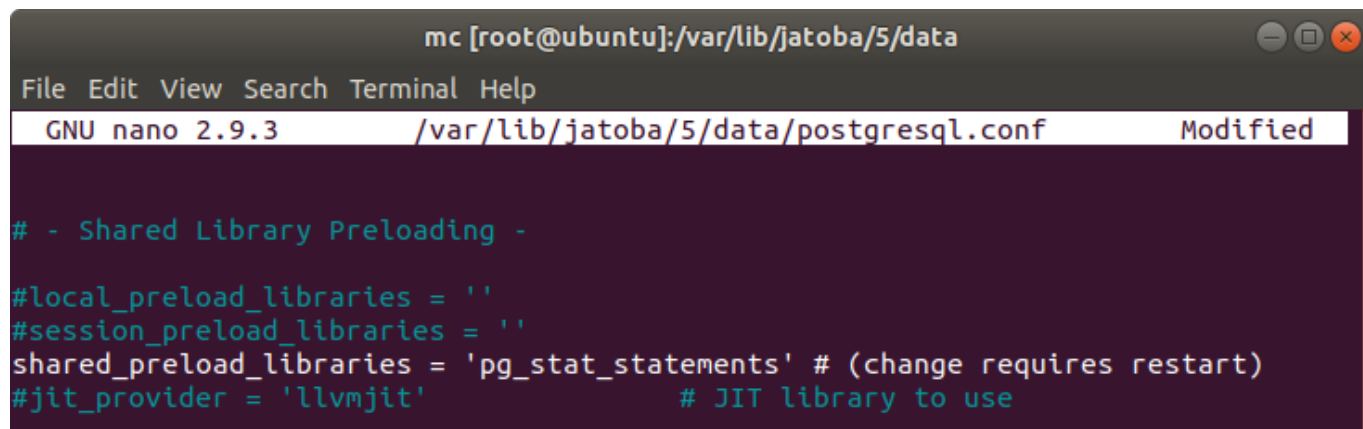
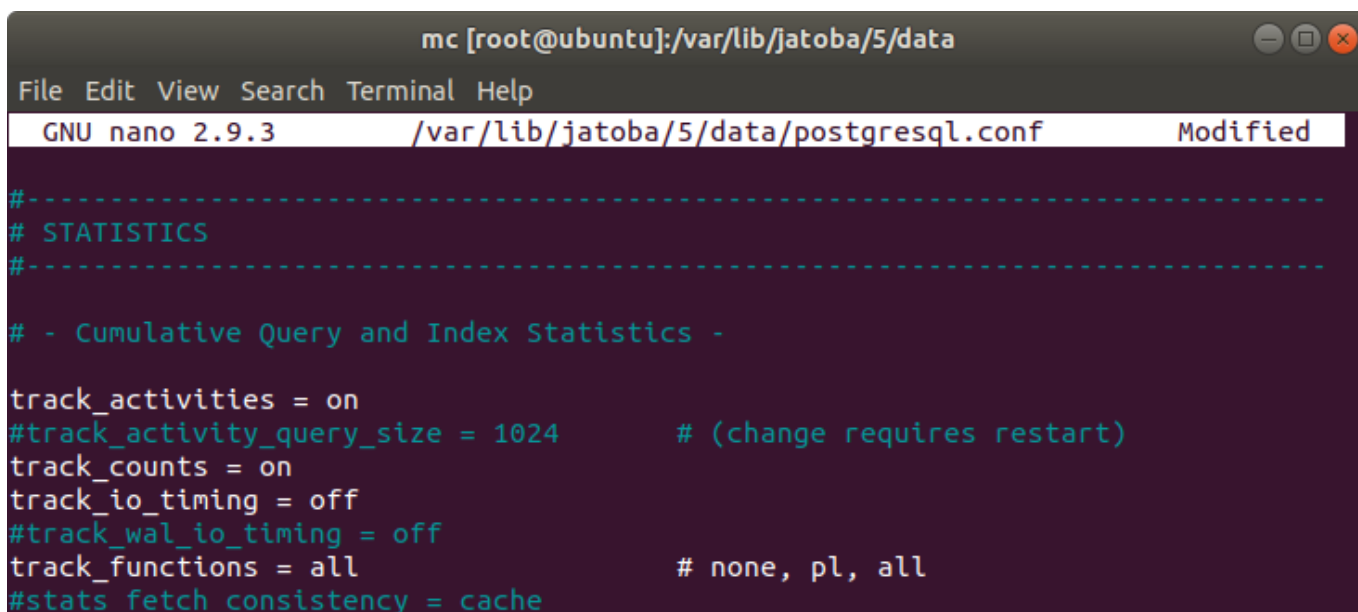


Рисунок 2.4 – Настройки postgresql.conf

В разделе «Statistics» – «Query and Index Statistics Collector» рекомендовано проверить и привести к следующему виду настройки для Statistic Collector (встроенный в СУБД механизм, позволяющий собирать метрики активности сервера БД) (рисунок 2.5):

```
track_activities = on  
track_counts = on  
track_io_timing = on  
track_functions = all
```



```
mc [root@ubuntu]:/var/lib/jatoba/5/data
File Edit View Search Terminal Help
GNU nano 2.9.3 /var/lib/jatoba/5/data/postgresql.conf Modified

#-----
# STATISTICS
#-----

# - Cumulative Query and Index Statistics -

track_activities = on
#track_activity_query_size = 1024      # (change requires restart)
track_counts = on
track_io_timing = off
#track_wal_io_timing = off
track_functions = all                  # none, pl, all
#stats_fetch_consistency = cache
```

Рисунок 2.5 – Настройки postgresql.conf



В настройке track_function допустимым значением является «pl», которое отслеживает время выполнения функций, написанных на процедурных языках. Значение «all» отслеживает все функции, к процедурным языкам добавляются SQL и C-функции.

После внесения изменений в конфигурационный файл, для применения настроек необходимо перезагрузить СУБД.

2.4. Установка расширения pg_Profile

Установка основного расширения «pg_profile» и дополнительных расширений допустима в:

- БД по умолчанию «postgres» (п. 2.4.1);
- служебную БД «pg_profile» (п. 2.4.2).

Для дальнейшего использования с компонентом пользовательского веб-интерфейса для администраторов «Jatoba data safe» целесообразнее устанавливать расширения в служебную БД, с последующей установкой внешнего сервера для служебной БД (п. 2.4.3).

2.4.1. Установка расширений в БД «postgres»

Установка расширения может быть проведена следующим способом:

№ изменения: _____	Подпись отв. лица: _____	Дата внесения изм: _____
--------------------	--------------------------	--------------------------

```
postgres=# CREATE EXTENSION dblink;
```

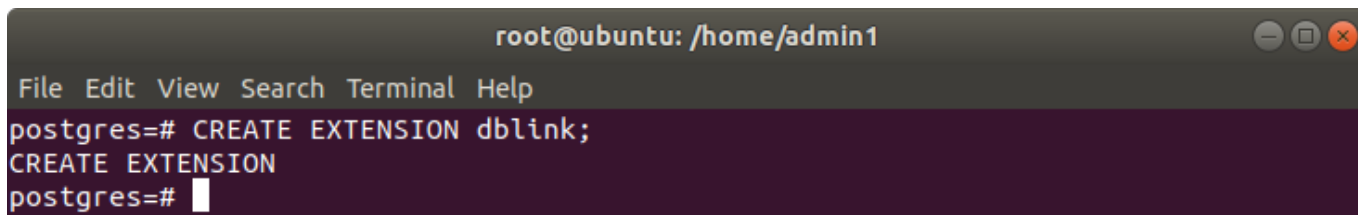


Рисунок 2.6 – Выполнение SQL-команды CREATE EXTENSION dblink

```
postgres=# CREATE EXTENSION pg_stat_statements;
postgres=# CREATE EXTENSION pg_profile;
```

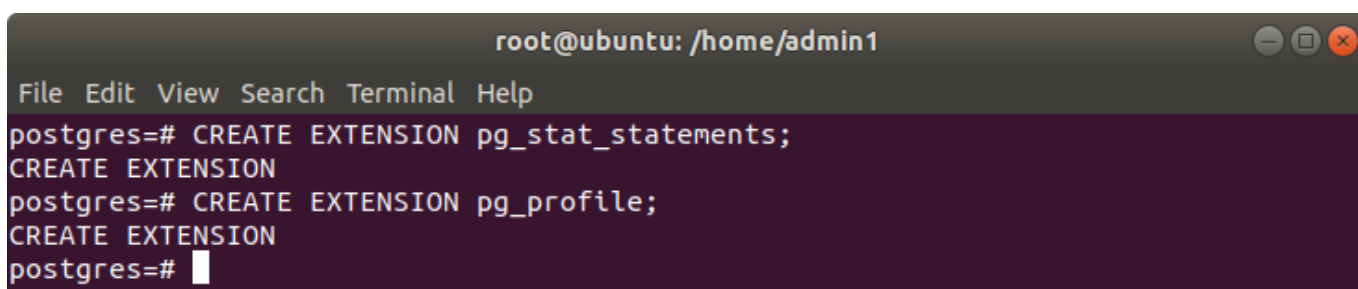


Рисунок 2.7 – Выполнение SQL-команд создания расширений

После выполнения данных команд будут созданы расширения СУБД (рисунок 2.8).

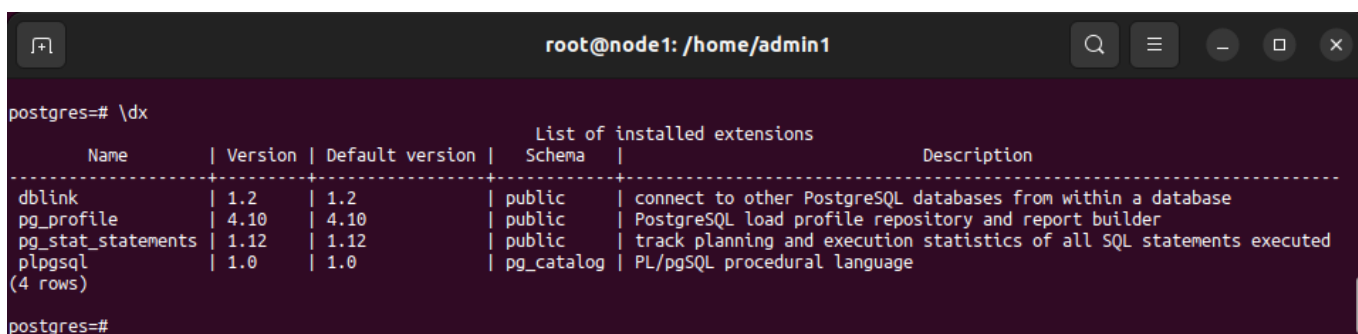


Рисунок 2.8 – Созданные расширения СУБД

Также есть возможность создать расширение в отдельной схеме:

```
postgres=# CREATE EXTENSION dblink;
postgres=# CREATE EXTENSION pg_stat_statements;
postgres=# CREATE SCHEMA profile;
postgres=# CREATE EXTENSION pg_profile SCHEMA profile;
```

2.4.2. Установка расширений в служебную БД

В случае, когда архитектура безопасности информационной системы следует принципу назначения минимально необходимых прав и привилегий пользователям и администраторам, использование компонента от имени и с правами суперпользователя

№ изменения: _____	Подпись отв. лица: _____	Дата внесения изм: _____
--------------------	--------------------------	--------------------------

становится невозможным. Кроме того, БД «postgres» является БД по умолчанию и использование ее для установки расширений нежелательно.

Существует наиболее безопасный способ использования компонента «pg_Profile» с установкой в служебную БД, в которой устанавливаются требуемые расширения, правами использования которых обладает пользователь СУБД с минимально достаточными правами.

Для этого необходимо проделать следующие шаги:

- создать БД «pg_profile»:

```
create database pg_profile;
```

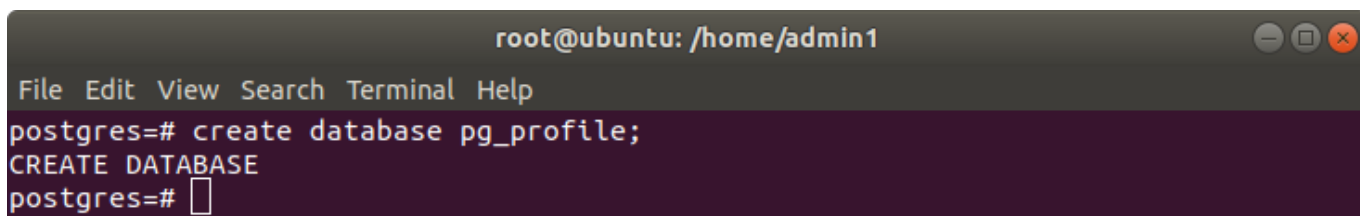


Рисунок 2.9 – Создание БД «pg_profile»

- подключиться к БД:

```
\connect pg_profile;
```

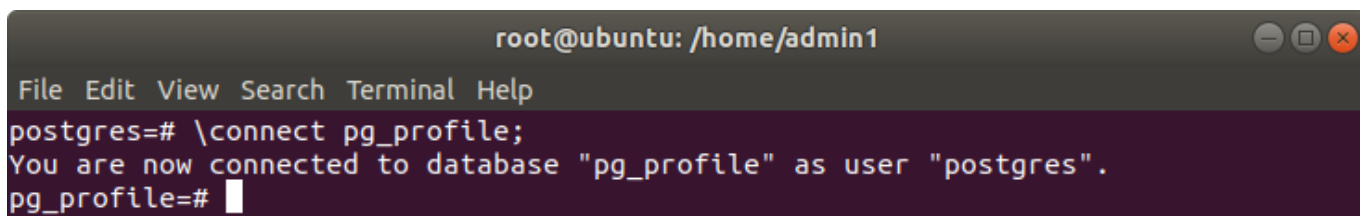


Рисунок 2.10 – Подключение к БД «pg_profile»

- создать пользователя «profile_usr»:

```
create role profile_usr login password 'pwd';
```

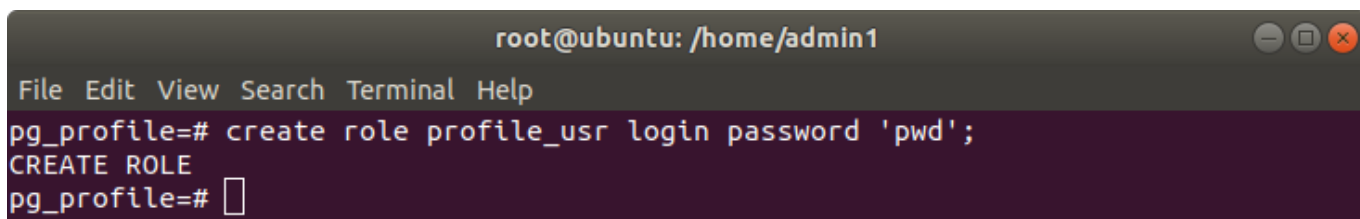
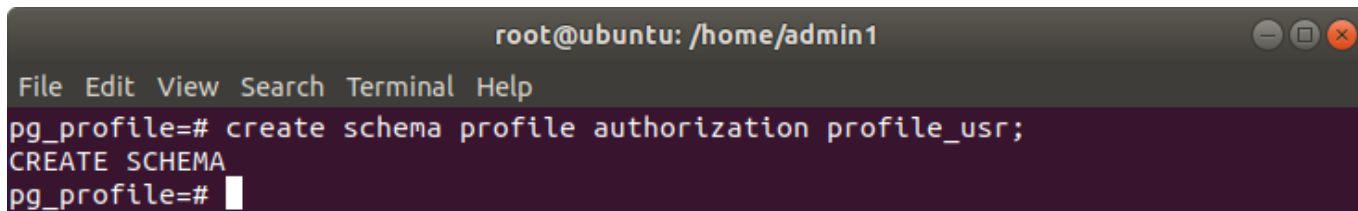


Рисунок 2.11 – Создание пользователя «profile_usr»

- создать схему для установки pg_profile:

```
create schema profile authorization profile_usr;
```

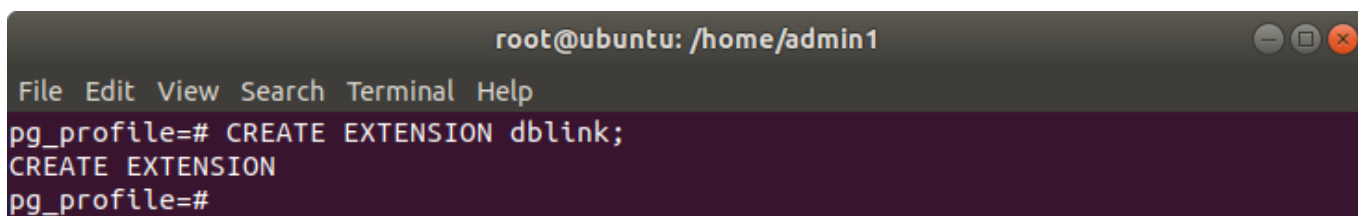


The screenshot shows a terminal window titled 'root@ubuntu: /home/admin1'. The command 'create schema profile authorization profile_usr;' has been entered and executed successfully, resulting in the output 'CREATE SCHEMA'. The prompt 'pg_profile=#' is visible at the end of the line.

Рисунок 2.12 – Создание схемы «profile»

- установить расширение «dblink»:

```
CREATE EXTENSION dblink;
```

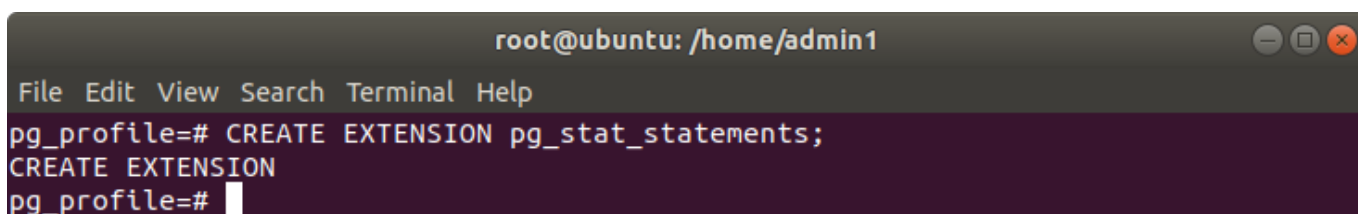


The screenshot shows a terminal window titled 'root@ubuntu: /home/admin1'. The command 'CREATE EXTENSION dblink;' has been entered and executed successfully, resulting in the output 'CREATE EXTENSION'. The prompt 'pg_profile=#' is visible at the end of the line.

Рисунок 2.13 – Установка расширения «dblink»

- установить расширение «pg_stat_statements»:

```
CREATE EXTENSION pg_stat_statements;
```



The screenshot shows a terminal window titled 'root@ubuntu: /home/admin1'. The command 'CREATE EXTENSION pg_stat_statements;' has been entered and executed successfully, resulting in the output 'CREATE EXTENSION'. The prompt 'pg_profile=#' is visible at the end of the line.

Рисунок 2.14 – Установка расширения «pg_stat_statements»

- предоставить разрешение на использование схемы public, в которой находится расширение dblink:

```
grant usage on schema public to profile_usr;
```

```

root@ubuntu: /home/admin1
File Edit View Search Terminal Help
pg_profile=# grant usage on schema public to profile_usr;
GRANT
pg_profile=#

```

Рисунок 2.15 – Разрешение на использование схемы public



Предполагается, что расширение dblink установлено в схему public

```

root@ubuntu: /home/admin1
File Edit View Search Terminal Help
pg_profile=# \dx

```

Name	Version	Schema	Description
dblink	1.2	public	connect to other PostgreSQL databases from within a database
pg_stat_statements	1.10	public	track planning and execution statistics of all SQL statements executed
plpgsql	1.0	pg_catalog	PL/pgSQL procedural language

```

(3 rows)
pg_profile=#

```

Рисунок 2.16 – Список установленных расширений в схемах данных

- создать расширение, «pg_profile» в схеме данных «profile»:

```
create extension pg_profile schema profile;
```

```

root@ubuntu: /home/admin1
File Edit View Search Terminal Help
pg_profile=# create extension pg_profile schema profile;
CREATE EXTENSION
pg_profile=#

```

Рисунок 2.17 – Создание расширения, используя учетную запись profile_usr

На этом шаге установка компонента в отдельной БД закончена.

2.4.3. Установка внешнего сервера для служебной БД

Компонент обладает функциональной возможностью предоставления внешнего подключения пользователю или приложению. Для этого используется виртуальный сервер.

В разбираемом примере:

- создана служебная БД «pg_profile»;
- в служебной БД «pg_profile» установлены требуемые расширения;
- сервер СУБД имеет IP-адрес 10.96.1.70;

№ изменения: _____	Подпись отв. лица: _____	Дата внесения изм: _____
--------------------	--------------------------	--------------------------

- создан пользователь «admin_bd» с атрибутом «Login».



В синтаксисе SQL-команды при вызове функций следует учитывать, что они установлены в отдельной схеме данных и для их вызова потребуется указывать схему данных, в которых они установлены.

Имея исходные данные, создать внешний сервер при помощи SQL-команды:

```
select
profile.create_server('pg_profile_server','host=10.96.1.70
port=5432 user=admin_bd password=P@ssword dbname=pg_profile');
```

```
root@ubuntu: /home/admin1
File Edit View Search Terminal Help
pg_profile=# select profile.create_server('pg_profile_server','host=10.96.1.70 p
ort=5432 user=admin_bd password=P@ssword dbname=pg_profile');
 create_server
-----
                2
(1 row)
pg_profile=#
```

Рисунок 2.18 – Создание внешнего сервера

После выполнения SQL-команды в таблице «server» будет создана запись с строкой подключения внешнего пользователя СУБД.

Пользователю СУБД от имени и с правами которого будет производиться подключение, достаточно иметь атрибут «Login» и дополнительные привилегии.

Для этого, необходимо назначить права на:

- схему данных «profile» SQL-командой:

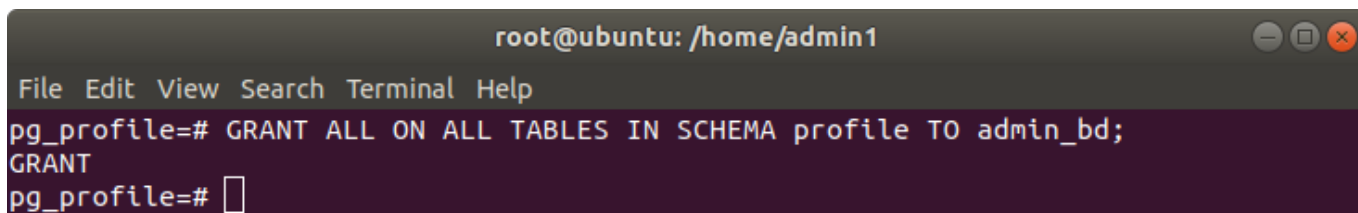
```
GRANT ALL ON SCHEMA profile TO [username];
```

```
root@ubuntu: /home/admin1
File Edit View Search Terminal Help
pg_profile=# GRANT ALL ON SCHEMA profile TO admin_bd;
GRANT
pg_profile=#
```

Рисунок 2.19 – Предоставление прав на схему данных

- таблицы схемы данных «profile» SQL-командой:

```
GRANT ALL ON ALL TABLES IN SCHEMA profile TO [username];
```

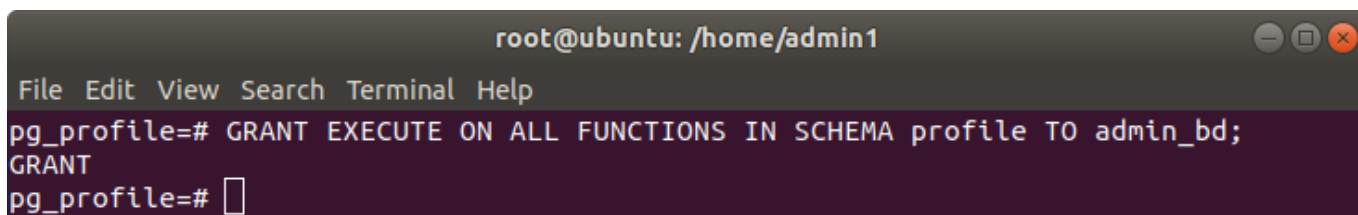


```
root@ubuntu: /home/admin1
File Edit View Search Terminal Help
pg_profile=# GRANT ALL ON ALL TABLES IN SCHEMA profile TO admin_bd;
GRANT
pg_profile=#
```

Рисунок 2.20 – Предоставление прав на таблицы схемы данных «profile»

- функции схемы данных «profile» SQL-командой:

```
GRANT EXECUTE ON ALL FUNCTIONS IN SCHEMA profile TO [username];
```



```
root@ubuntu: /home/admin1
File Edit View Search Terminal Help
pg_profile=# GRANT EXECUTE ON ALL FUNCTIONS IN SCHEMA profile TO admin_bd;
GRANT
pg_profile=#
```

Рисунок 2.21 – Предоставление прав на функции схемы «profile»

Необходимо убедиться, что пользователь имеет права на подключение к любой БД в СУБД (по умолчанию это так) и в конфигурационном файле «pg_hba.conf» прописаны разрешения для подключения с узла БД «pg_profile».

Кроме того, необходимо, чтобы пользователь СУБД от имени и с правами которого производится подключение, был включен в групповую роль «pg_read_all_stats» и имел привилегию «execute» на функцию «pg_stat_statements_reset».

```
# GRANT pg_read_all_stats to [username];
# GRANT execute on function pg_stat_statements_reset TO
[username];
```

```

root@ubuntu: /home/admin1
File Edit View Search Terminal Help
pg_profile=# grant pg_read_all_stats to admin_bd;
GRANT ROLE
pg_profile=# grant execute on function pg_stat_statements_reset TO admin_bd;
GRANT
pg_profile=#

```

Рисунок 2.22 – Включение в групповую роль «pg_stat_statements_reset» и назначение привилегии «execute» на функцию «pg_stat_statements_reset»

По умолчанию, вновь созданный сервер будет неактивен. Это отразится при просмотре состояния снапшотов:

```
SELECT profile.take_sample();
```

```

root@ubuntu: /home/admin1
File Edit View Search Terminal Help
pg_profile=# select profile.take_sample();
               take_sample
-----
(local,OK,00:00:01.62)
(pg_profile_server,"could not establish connection
SQL statement ""SELECT dblink_connect('server_connection', server_properties #>> '{properties,server_connstr}')""
PL/pgSQL function init_sample(integer) line 60 at PERFORM
PL/pgSQL function take_sample(integer,boolean) line 14 at assignment
PL/pgSQL function take_sample_subset(integer,integer) line 27 at assignment
SQL function ""take_sample"" statement 1
connection to server at ""10.96.1.70"", port 5432 failed: No route to host
Is the server running on that host and accepting TCP/IP connections?",00:00:03.06)
(2 rows)

```

Рисунок 2.23 – Состояние снятия снапшотов

Включение сервера выполняется командой:

```
SELECT profile.enable_server('pg_profile_server');
```

```

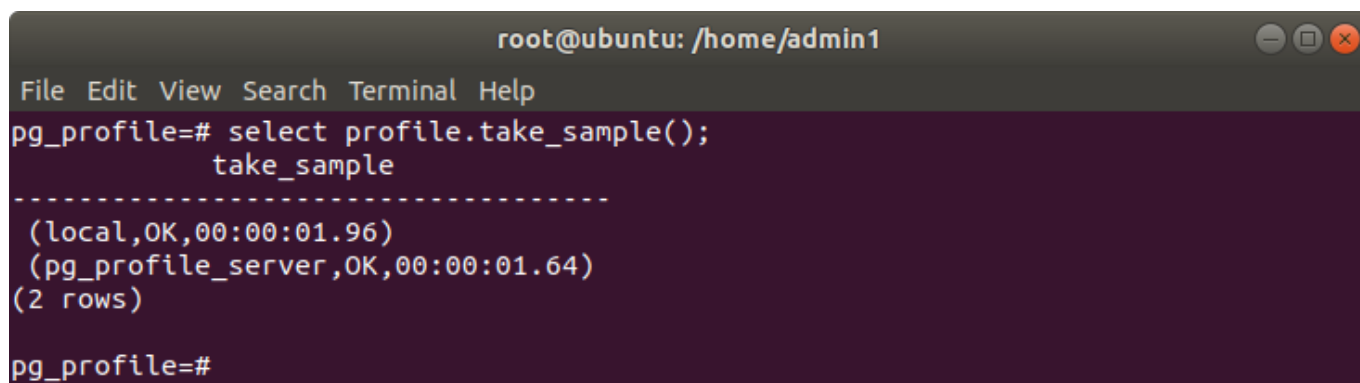
root@ubuntu: /home/admin1
File Edit View Search Terminal Help
pg_profile=# SELECT profile.enable_server('pg_profile_server');
enable_server
-----
1
(1 row)
pg_profile=#

```

Рисунок 2.24 – Включение сервера с именем 'pg_profile_server'

После чего оба сервера будут активны, но для снижения нагрузки на СУБД, рекомендуется оставлять работающим один сервер.

№ изменения: _____	Подпись отв. лица: _____	Дата внесения изм: _____
--------------------	--------------------------	--------------------------



```
root@ubuntu: /home/admin1
File Edit View Search Terminal Help
pg_profile=# select profile.take_sample();
               take_sample
-----
(local,OK,00:00:01.96)
(pg_profile_server,OK,00:00:01.64)
(2 rows)

pg_profile=#
```

Рисунок 2.25 – Активное состояние серверов

На этом формирование подключения к внешнему серверу закончено.

2.5. Дополнительные настройки «postgresql.conf»

По умолчанию в postgresql.conf устанавливаются следующие параметры, которые можно поменять вручную:

- pg_profile.topn = 20 – количество основных объектов для каждого типа, которые будут выводиться в отчете;
 - pg_profile.max_sample_age = 7 – время жизни снимка (снимки состояния, которые старше 7 дней будут удаляться при создании нового);
 - pg_profile.track_sample_timings = off – настройка записи детальных данных о времени в снимок;
 - pg_profile.max_query_length = 20000 – ограничение длины запроса для отчетов.
- Все запросы в отчете будут усечены до этой длины. Этот параметр не влияет на сбор текста запроса – во время выборки собираются полные тексты запросов, таким образом, они могут быть получены.

3. ИСПОЛЬЗОВАНИЕ

3.1. Управление соединениями с серверами

После установки расширение автоматически создаст соединение с локальным сервером, на котором оно установлено. Для управления другими соединениями имеются следующие функции:

```
create_server(server name, server_connstr text, server_enabled  
boolean = TRUE, max_sample_age integer = NULL, description text  
= NULL)
```

– создание нового соединения с сервером со следующими аргументами:

- `server` – уникальное имя сервера;
- `server_connstr` – строка соединения с сервером;
- `enabled` – включение сервера в набор серверов, на которых снимаются метрики при создании снапшота;
- `max_sample_age` – время жизни снапшота на сервере;
- `description` – описание сервера, которое будет указано в отчетах.

```
drop_server(server name)
```

– удаление сервера из настроек и удаление его снапшотов;

```
enable_server(server name)
```

– включает сервер в набор серверов, на которых снимаются метрики при создании снапшота;

```
disable_server(server name)
```

– исключает сервер из набора серверов, на которых снимаются метрики при создании снапшота;

```
rename_server(server name, new_name name)
```

– переименование сервера;

```
set_server_max_sample_age(server name, max_sample_age integer)
```

– определение времени жизни снапшота на сервере;



Функция `set_server_max_sample_age` перезаписывает значение `pg_profile.max_sample_age` в `postgresql.conf` для сервера.

Для отмены времени жизни снапшота, необходимо установить значение NULL.

```
set_server_db_exclude(server name, exclude_db name[])
```

– исключение из снапшота определенных БД на сервере (не будут указаны в отчетах);

```
set_server_connstr(server name, new_connstr text)
```

– установка правил соединения с сервером;

```
set_server_description(server name, description text)
```

– описание сервера (будет указано в отчетах);

```
show_servers()
```

– отображение созданных соединений с серверами.

Пример создания сервера:

```
SELECT profile.create_server('omega', 'host=name_or_ip  
dbname=postgres port=5432')
```

3.2. Сбор данных о размерах страниц

Для определения сбора данных о размерах таблиц реализована политика сбора данных.

Сбор данных о размерах таблиц в СУБД ресурсозатратный процесс и требует, чтобы таблица была эксклюзивно заблокирована на момент подсчета размера.



Данные о размерах таблиц можно собирать не постоянно, а, например, раз в сутки в выделенный промежуток времени.

Также можно установить временной интервал между двумя снимками состояния, где просчитываются данные о размерах таблиц.

- Функция `set_server_size_sampling()` определяет данные политики сбора размеров.

```
set_server_size_sampling(server name, window_start time with  
time zone = NULL, window_duration interval hour to second =  
NULL, sample_interval interval day to minute = NULL)
```

`server` – имя сервера;

`window_start` – начало временного промежутка, когда разрешено собирать данные о размерах таблиц;

`window_duration` – продолжительность временного промежутка;

`sample_interval` – минимальное время между двумя снимками с собранными размерами таблиц.



Все три параметра являются обязательными для установки политики сбора данных о размерах таблиц.

Пример:

```
SELECT set_server_size_sampling('local','23:00+03', interval '2  
hour',interval '8 hour')
```

- Функция `show_servers_size_sampling()` показывает все установленные политики.

3.3. Снимки состояния БД

Каждый снимок показывает статистическую информацию о метриках БД и нагрузке с момента снятия предыдущего состояния.

Для работы со снимками используются следующие команды:

```
take_sample(server name [, skip_sizes boolean])
```

- снятие снимков со всех серверов.

server_name – имя сервера для снятия снимка;

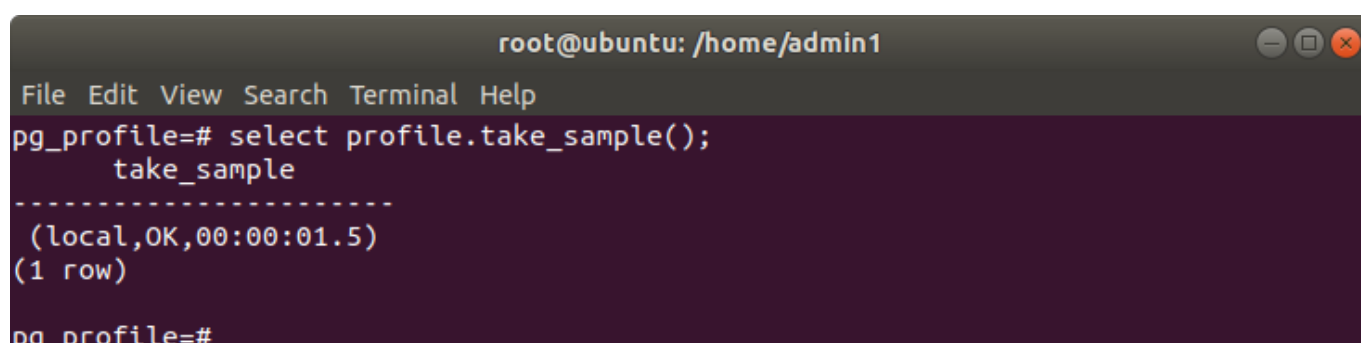
skip_sizes – переопределение установленной политики сбора данных о размерах.



Снимки для каждого сервера будут сняты последовательно, один за другим.

Если не указывать параметры server_name, то будут сняты снимки со всех доступных серверов.

При запросе функции без указания параметров пользователю будет возвращена таблица вида (см. рис. 3.1):



```
root@ubuntu: /home/admin1
File Edit View Search Terminal Help
pg_profile=# select profile.take_sample();
      take_sample
-----
 (local,0K,00:00:01.5)
(1 row)
pg_profile=#
```

Рисунок 3.1 – Возвращение таблицы

server name – имя сервера;

result text – результат (ОК – успешно, текст ошибки – неудача);

elapsed interval – время, затраченное на создание снимка.



Работа функции take_sample() может занять значительное время из-за последовательного снятия снимков.

```
take_sample_subset([sets_cnt integer], [current_set integer])
```

– функция take_sample() обрабатывает данные и создает снимки последовательно. В случае, если пользователь регистрирует множество серверов через функцию create_server(), то для получения всех снимков может быть затрачено значительное время. Функция take_sample_subset() предоставляет возможность использовать параллельную обработку серверов, тем самым, сокращая время на получение всех снимков.

`sets_cnt` – число, на которое будет разделено все множество серверов. К примеру, если у пользователя есть 20 серверов, то указывая `sets_cnt = 5`, будет создано 5 подмножеств по 4 сервера.

`current_set` – номер подмножества, которое будет обрабатываться в данный момент. Счет идет от 0, максимальное значение `current_set = sets_cnt – 1`.

Можно запускать как один за другим, контролируя таким образом время выполнения, так и запустить параллельно в нескольких сеансах, указывая разный `current_set`.

```
show_samples([server name,] [days integer])
```

– возвращение таблицы с данными об имеющихся снапшотах. Если не вводить имя сервера, то автоматически подставится локальный сервер.

- `sample integer` – ID снапшота;
- `sample_time timestamp (0) with time zone` – время, когда был снят снапшот;
- `dbstats_reset / clustats_reset / archstats_reset timestamp (0) with time zone` – время сброса статистики в представлениях `pg_stat_database`, `pg_stat_bgwriter` с `pg_stat_archiver` с момента последнего снятия снапшота.

3.3.1. Снятие снапшотов

Для построения отчета необходимо, как минимум, 2 снапшота.

Для автоматического получения снимков по расписанию, к примеру, раз в 30 минут, нужно поместить вызов функции в планировщик задач системы.

В стандартной утилите Linux – `crontab`:

```
*/30 * * * * psql -c 'SELECT [profile].take_sample();' > /dev/null 2>&1
```

В MS Windows (начиная с Windows Server 2003) – стандартной утилитой `schtasks`:

```
schtasks /create /sc minute /mo 30 /tn "30_min_sample" /tr psql -c "SELECT [profile].take_sample();" >
```

Для получения списка существующих снапшотов в хранилище используется функция `show_samples()`. Эта функция также покажет обнаруженные сбросы статистики БД.

3.3.2. Время жизни снимков

Для определения срока хранения снимков реализована политика хранения. Определить политику хранения можно на трех уровнях:

- 1) установка параметра `pg_profile.max_sample_age` в файле `postgresql.conf` – общая настройка, которая действует, если не определено ни одно из других;
- 2) определение параметра `max_sample_age` сервера при создании сервера или использование функции `set_server_max_sample_age()` для существующего сервера – настройка переопределяет глобальную настройку `pg_profile.max_sample_age` для конкретного сервера;
- 3) создание `baseline` (см. пункт 3.4) – `baseline` будет переопределять период хранения для включенных снимков с наивысшим приоритетом.

3.3.3. Описание событий

`pg_profile` собирает подробную статистику времени снятия снимков, если включен параметр `pg_profile.track_sample_timings`. Результаты можно получить из представления `v_sample_timings`.

Описание полей `v_sample_timings`:

- `server_name` – имя сервера с которого создавался снимок;
- `sample_id` – идентификатор снимка;
- `sample_time` – время взятия снимка;
- `event` – событие на котором измерялось время;
- `time_spent` – количество времени, проведенного в событии.

В таблице 3.1 описаны все события и метрики, которые доступны пользователю в созданном отчете.

Таблица 3.1 – Описание событий и метрик

Описание	Значение
total	снятие снимка (все этапы)
connect	установка соединения <code>dblink</code> с сервером
get server environment	получение конфигурационных параметров сервера, доступных расширений и т.д.

№ изменения: _____	Подпись отв. лица: _____	Дата внесения изм: _____
--------------------	--------------------------	--------------------------

Описание		Значение
collect database stats		запрос представления pg_stat_database для получения статистики по БД
calculate database stats		вычисление дифференциальной статистики по БД с момента предыдущего снимка
collect tablespace stats		запрос представления pg_tablespace для получения статистики по табличным пространствам
collect statement stats		сбор статистики по операторам с помощью расширений pg_stat_statements и pg_stat_kcache
query pg_stat_bgwriter		сбор статистики кластера с помощью представления pg_stat_bgwriter
query pg_stat_archiver		сбор статистики кластера с помощью представления pg_stat_archiver
collect object stats		сбор статистики по объектам БД. Включает события
	db:dbname collect tables stats	сбор статистики по таблицам для БД dbname
	db:dbname collect indexes stats	сбор статистики по индексам для БД dbname
	db:dbname collect functions stats	сбор статистики по функциям для БД dbname
maintain repository		выполнение процедур поддержки
calculate tablespace stats		вычисление дифференциальной статистики по табличным пространствам. Включает события
	calculate object stats	вычисление дифференциальной статистики по объектам БД
	calculate tables stats	вычисление дифференциальной статистики по таблицам БД
	calculate indexes stats	вычисление дифференциальной статистики по индексам БД
calculate functions stats		вычисление дифференциальной статистики по функциям БД
calculate cluster stats		вычисление дифференциальной статистики кластера
calculate archiver stats		вычисление дифференциальной статистики архиватора
delete obsolete samples		удаление устаревших baseline и снимков

3.4. Baseline

Baseline – именованная последовательность снимков, которая имеет отдельную от настроенной политику хранения. Можно задать как определенное время хранения в днях, так и бесконечное время хранения, оставив соответственный параметр пустым. Также можно создать последовательность снимков только для определенного периода времени.

Можно сохранить снимки, собранные во время нагрузочного тестирования или во время обычной нагрузки на систему для дальнейшего использования.

3.4.1. Функции управления baseline

Для управления baseline предназначены следующие функции:

№ изменения: _____	Подпись отв. лица: _____	Дата внесения изм: _____
--------------------	--------------------------	--------------------------

```
CREATE_baseline([server name,] baseline_name varchar(25),  
start_id integer, end_id integer [, days integer])
```

– создание baseline,

server name – имя сервера (если параметр опущен, предполагается локальный сервер);

baseline_name – имя baseline. Каждый baseline должен иметь уникальное имя в пределах сервера;

start_id, end_id – первый и последний образцы, включенные в baseline;

days – время хранения baseline. Определяется в целых днях с момента времени now(). Этот параметр может быть опущен (или установлен в null), что означает неограниченное хранение.

```
CREATE_baseline([server name,] baseline_name varchar(25),  
time_range tstzrange [, days integer])
```

– создание baseline,

server name – имя сервера (если параметр опущен, предполагается локальный сервер);

baseline_name – имя baseline. Каждый baseline должен иметь уникальное имя в пределах сервера;

time_range – временной интервал для baseline. Baseline будет включать все доступные снимки, перекрывающие этот интервал;

days – время хранения baseline. Определяется в целых днях с момента времени now(). Этот параметр может быть опущен (или иметь значение null), что означает неограниченное хранение.

```
drop_baseline([server name,] name varchar(25))
```

– удаление baseline,

server name – имя сервера (если параметр опущен, предполагается локальный сервер);

name – имя baseline для удаления. Удаление baseline не означает удаление его снимков, они исключаются из baseline.

```
keep_baseline([server name,] name varchar(25) [, days integer])
```

– изменение времени хранения baseline,

server name – имя сервера (если параметр опущен, предполагается локальный сервер);

name – имя baseline. Каждый baseline должен иметь уникальное имя в пределах сервера;

days – время хранения baseline. Определяется в целых днях с момента времени now(). Этот параметр может быть опущен (или иметь значение null), что означает неограниченное сохранение.

```
show_baselines([server name])
```

– просмотр существующих baseline и их информация,

server name – имя сервера (если параметр опущен, предполагается локальный сервер).

Пример просмотра существующих baseline:

```
postgres=# SELECT * FROM show_baselines('local');
```

3.5. Экспорт / импорт данных

Собранные снапшоты можно экспортировать из экземпляра расширения pg_Profile и импортировать на другой сервер.

3.5.1. Экспорт данных

Данные экспортируются в виде обычной таблицы с помощью функции export_data(). Можно использовать любой метод для экспорта этой таблицы из БД. Например, можно использовать команду copy в psql для получения одного файла .csv:

```
postgres=# \copy (select * from export_data()) to 'export.csv'
```

По умолчанию функция export_data() экспортирует все снапшоты всех настроенных серверов. Можно ограничить экспорт только одним сервером, а также ограничить диапазон снапшотов:

```
export_data([server name, [min_sample_id integer,]  
[max_sample_id integer]] [, obfuscate_queries boolean])
```

– экспорт собранных данных,

`server` – имя сервера (если параметр опущен, предполагаются все серверы);

`min_sample_id` и `max_sample_id` – экспорт идентификаторов граничных снапшотов (включительно). Нулевое значение `min_sample_id` приводит к экспорту всех снапшотов до `max_sample_id`, а нулевое значение `max_sample_id` приводит к экспорту всех снапшотов, начиная с `min_sample_id`;

`obfuscate_queries` – скрытие текста запросов, они будут экспортированы в виде MD5-хэша.

3.5.2. Импорт данных

Данные могут быть импортированы только из локальной таблицы, поэтому ранее экспортированные данные необходимо сначала загрузить с помощью команды `copy`:

```
# CREATE TABLE import (section_id bigint, row_data json);  
# \copy import from 'export.csv'
```

После загрузки можно выполнить импорт данных, предоставив эту таблицу функции `import_data()`:

```
# SELECT * FROM import_data('import');
```



Одинаково названные `pg_profile` сервера на разных физических серверах при импорте данных не будут обработаны из-за внутренних ограничений функции. Во избежание данной ситуации нужно временно переименовать их через встроенную функцию `rename_server(server name, new_name name)`

Если понадобится импортировать новые данные для ранее импортированных серверов, они будут сопоставлены по системному идентификатору. Все серверы импортируются с опцией `server_enabled = FALSE` (см. пункт 3.1).

Функция `import_data()` принимает только импортируемую таблицу:

```
import_data(data regclass)
```

data – таблица, содержащая экспортируемые данные. Эта функция возвращает количество строк, фактически загруженных в таблицы расширения. После успешного завершения операции можно удалить таблицу импорта.

3.6. Создание отчетов

Отчеты создаются в формате HTML. Существует два вида отчетов: обычные отчеты и дифференциальные отчеты.

3.6.1. Обычные отчеты

Обычные отчеты содержат статистическую информацию для определенного периода времени.

Функции обычного отчета:

```
get_report([server name,] start_id integer, end_id integer [,  
description text [, with_growth boolean]])
```

– создание отчета, исходя из идентификаторов снапшотов;

```
get_report([server name,] time_range tstzrange [, description  
text [, with_growth boolean]])
```

– генерирование отчета для самого короткого интервала снапшотов, охватывающего указанный time_range;

```
get_report([server name], baseline varchar(25) [, description  
text [, with_growth boolean]])
```

– генерирование отчета, используя baseline в качестве интервала снапшотов;

```
get_report_latest([server name])
```

– создание отчета, исходя из двух последних снапшотов,

server – имя сервера (если параметр опущен, предполагается локальный сервер);

start1_id, end1_id – идентификаторы снапшота первого интервала;

start2_id, end2_id – идентификаторы снапшота второго интервала;

baseline1 – имя baseline первого интервала;

baseline2 – имя baseline второго интервала;

time_range1 – временной диапазон первого интервала;

time_range2 – временной диапазон второго интервала;

description – текстовая заметка, которая будет включена в отчет, как описание отчета;

with_growth – параметр, который означает, что будет браться ближайший снимок с данными о размерах.

3.6.2. Дифференциальные отчеты

Дифференциальные отчеты содержат статистическую информацию за два выбранных периода, позволяя легко сравнить показатели.

Функции дифференциального отчета:

```
get_diffreport([server name,] start1_id integer, end1_id  
integer, start2_id integer, end2_id integer [, description text  
[, with_growth boolean]])
```

– создание отчета на основе двух периодов времени;

```
get_diffreport([server name,] baseline1 varchar(25), baseline2  
varchar(25) [, description text [, with_growth boolean]])
```

– создание дифференциального отчета по двум интервалам, заданными именами baseline;

```
get_diffreport([server name,] time_range1 tstzrange,  
time_range2 tstzrange [, description text [, with_growth  
boolean]])
```

– создание дифференциального отчета по двум интервалам, заданным временными диапазонами,

server – имя сервера (если параметр опущен, предполагается локальный сервер);

start1_id, end1_id – идентификаторы снимков первого интервала;

start2_id, end2_id – идентификаторы снимков второго интервала;

baseline1 – название базовой линии первого интервала;

baseline2 – название базовой линии второго интервала;

time_range1 – временной диапазон первого интервала;

№ изменения: _____	Подпись отв. лица: _____	Дата внесения изм: _____
--------------------	--------------------------	--------------------------

time_range2 – временной диапазон второго интервала;

with_growth – параметр, который означает, что будет браться ближайший снимок с данными о размерах;

description – текстовая информация, которая будет выведена в отчете.

Пример формирования отчета:

```
$ psql -AqtC "SELECT profile.get_report('omega',12,14)" -o  
report_omega_12_14.html
```

4. ДАННЫЕ В ОТЧЕТАХ

В данном разделе описаны таблицы отчетов:

- 1) статистики сервера (Server statistics п. 4.1);
- 2) статистики SQL-запросов (SQL query statistics п. 4.2);
- 3) статистики объекта схемы (Schema object statistics п. 4.3);
- 4) статистики функций пользователя (User function statistics п. 4.4);
- 5) статистики, связанной с вакуумом (Vacuum-related statistics п. 4.5);
- 6) настройки кластера во время отчетного интервала (Cluster settings during the report interval п. 4.6);
- 7) Отчеты по компоненту «ja_Hipe_Cluster» (Citrus) (п. 4.7).

В каждом подразделе приведены подробные виды отчета.



Наименование таблиц в отчетах отображаются на английском языке.

4.1. Server statistics (Статистика сервера)

Данный раздел содержит сведения о метриках и статистических данных, собранных в генерируемом отчете.

4.1.1. Database statistics (Статистика базы данных)

Таблица 4.1 содержит статистику по БД в течение интервала отчета, основанную на представлении pg_stat_database.

Database	Transactions			Block statistics			Block I/O times		Tuples					Temp files		Size	Growth
	Commits	Rollbacks	Deadlocks	Hit(%)	Read	Hit	Read	Write	Ret	Fet	Ins	Upd	Del	Size	Files		
postgres	85			100		179052			218960	72218	2162	2671	1307			14 MB	816 kB
Total	85			100		179052			218960	72218	2162	2671	1307			14 MB	816 kB

Рисунок 4.1 – Пример отчета Database statistics

Таблица 4.1 – Описание параметров отчета Database statistics

Параметр		Описание
Database		имя БД
Transaction		статистика транзакций БД
	Commits	количество зафиксированных транзакций (xact_commit)
	Rollbacks	количество откатенных транзакций (xact_rollback)
	Deadlocks	количество обнаруженных взаимных блокировок (deadlocks)

№ изменения: _____	Подпись отв. лица: _____	Дата внесения изм: _____
--------------------	--------------------------	--------------------------

Параметр		Описание
Block statistics		статистика чтения и попаданий в блоки БД
	Hit(%)	коэффициент попадания в буферный кэш
	Read	количество прочитанных дисковых блоков в данной БД (blks_read)
	Hit	количество раз, когда дисковые блоки были найдены в буферном кэше (blks_hit)
Block I/O times		время, затраченное на операции ввода/вывода (I/O)
	Read	на операции чтения
	Write	на операции записи
Tuples		раздел статистики записей
	Ret	количество возвращенных записей (tup_returned)
	Fet	количество извлеченных записей (tup_fetched)
	Ins	количество вставленных записей (tup_inserted)
	Upd	количество обновленных записей (tup_updated)
	Del	количество удаленных записей (tup_deleted)
Temp files		статистика временных файлов
	Size	общий объем данных, записанных во временные файлы запросами в этой БД (temp_bytes)
	Files	количество временных файлов, созданных запросами в этой БД (temp_files)
Size		размер БД в конце интервала отчета (pg_database_size())
Growth		рост БД в течение интервала отчета (разница pg_database_size())

4.1.2. Cluster I/O statistics (Статистика ввода/вывода кластера)

Таблица 4.2 содержит статистику по кластеру на операции ввода/вывода (I/O) в течение интервала отчет, основанную на представлении pg_stat_io.

Object	Backend	Context	Reads			Writes			Writebacks			Extends			Hits	Evictions	Reuses	Fsyncs	
			Count	Bytes	Time	Count	Bytes	Time	Count	Bytes	Time	Count	Bytes	Time				Count	Time
relation	autovacuum worker	normal										13	136 kB		17130				
relation	autovacuum worker	vacuum													885				
relation	autovacuum worker	*										13	136 kB		18015				
relation	checkpointer	normal				742	5936 kB	0.03	742	5936 kB	0.03							231	0.02
relation	checkpointer	*				742	5936 kB	0.03	742	5936 kB	0.03							231	0.02
relation	client backend	normal										109	904 kB		162783				
relation	client backend	vacuum													179				
relation	client backend	*										109	904 kB		162962				
relation	Total	*				742	5936 kB	0.03	742	5936 kB	0.03	122	1040 kB		180977			231	0.02
wal	autovacuum worker	normal				22	664 kB											22	
wal	autovacuum worker	*				22	664 kB											22	
wal	checkpointer	normal				3	24 kB											3	
wal	checkpointer	*				3	24 kB											3	
wal	client backend	normal				3	2304 kB											3	
wal	client backend	*				3	2304 kB											3	
wal	walwriter	normal				21	1608 kB											18	
wal	walwriter	*				21	1608 kB											18	
wal	Total	*				49	4608 kB											46	
*	Total	*				791	10 MB	0.03	742	5936 kB	0.03	122	1040 kB		180977			277	0.02

Рисунок 4.2 – Пример отчета Cluster I/O statistics

Таблица 4.2 – Описание параметров отчета Cluster I/O statistics

Параметр	Описание
Object	целевой объект операции ввода-вывода. Возможные значения: <ul style="list-style-type: none"> relation: Постоянные связи. temp relation: Временные связи. wal: Журналы предварительной записи.
Backend	Тип бэкенда (например, background worker, autovacuum worker). Дополнительную информацию о типах бэкендов см. в pg_stat_activity. Некоторые типы бэкендов не накапливают статистику операций ввода-вывода и не будут включены в представление.
Context	Контекст операции ввода-вывода. Возможные значения: <ul style="list-style-type: none"> normal: Стандартный контекст для определенного типа операций ввода-вывода. Например, по умолчанию данные отношений считываются и записываются из общих буферов. Таким образом, операции чтения и записи данных отношений в общие буферы и из них отслеживаются в контексте normal. init: Операции ввода-вывода, выполняемые при создании сегментов WAL, отслеживаются в контексте init. vacuum: Операции ввода-вывода, выполняемые вне общих буферов при очистке и анализе постоянных отношений. Очистка временных таблиц использует тот же локальный пул буферов, что и другие операции ввода-вывода временных таблиц, и отслеживается в контексте normal. bulkread: Некоторые операции чтения/вывода больших объемов данных, выполняемые вне общих буферов, например, последовательное сканирование большой таблицы.

Параметр		Описание
		– bulkwrite: Некоторые операции записи/вывода больших объемов данных, выполняемые вне общих буферов, такие как COPY.
Reads		статистика операций чтений
	Count	количество операций чтения
	Bytes	количество общее операций чтения в байтах
	Time	время ожидания операций чтения в миллисекундах (если track_io_timing включен и объект не является wal, или если track_wal_io_timing включен и объект является wal, в противном случае — ноль)
Writes		статистика операций записи
	Count	количество операций записи
	Bytes	количество общее операций записи в байтах
	Time	время ожидания операций записи в миллисекундах (если track_io_timing включен и объект не является wal, или если track_wal_io_timing включен и объект является wal, в противном случае — ноль)
Writebacks		статистика операций запрошенной записи
	Count	количество операции запрошенной записи
	Bytes	количество общее операций запрошенной записи в байтах
	Time	время ожидания операций запрошенной записи в миллисекундах (если включен track_io_timing, в противном случае — ноль). Это включает время, затраченное на постановку запросов на запись в очередь, и, возможно, время, затраченное на запись измененных данных.
Extends		
	Count	количество операции расширения файла данных
	Bytes	количество общее операций расширения файла данных в байтах
	Time	время ожидания операций расширения файла данных в миллисекундах. (если track_io_timing включен и объект не является wal, или если track_wal_io_timing включен и объект является wal, в противном случае - ноль)
Hits		количество раз, когда нужный блок был найден в общем буфере.
Evictions		количество раз, когда блок был записан из общего или локального буфера, чтобы сделать его доступным для другого использования.
Reuses		количество раз, когда существующий буфер в кольцевом буфере с ограниченным размером вне разделяемых буферов был повторно использован в рамках операции ввода-вывода в контексте массового чтения, массовой записи или очистки.
Fsyncs		статистика вызовов fsync
	Count	количество вызовов fsync. Они отслеживаются только в нормальном контексте.

Параметр		Описание
	Time	время ожидания операций fsync в миллисекундах (если track_io_timing включен и объект не является wal, или если track_wal_io_timing включен и объект является wal, в противном случае — ноль)

4.1.3. Cluster SLRU statistics (Статистика доступа к SLRU-кешам)

Таблица 4.3 содержит статистику доступа к SLRU-кешам (simple least-recently-used, простое вытеснение давно не используемых), основанную на представлении pg_stat_slru.

Name	Zeroed	Hits	Reads	%Hit	Writes	Checked	Flushes	Truncates
multixact_member		5		100				
multixact_offset		5		100				
subtransaction		4		100				
transaction		277		100				
Total		291		100				

Рисунок 4.3 – Пример отчета Cluster SLRU statistics

Таблица 4.3 – Описание параметров отчета Cluster SLRU statistics

Параметр	Описание
Name	имя SLRU-кеша
Zeroed	количество блоков, обнулённых при инициализации
Hits	сколько раз дисковые блоки обнаруживались в SLRU-кеше и чтение с диска не требовалось (учитываются только случаи обнаружения в этом кеше, а не в файловом кеше ОС)
Reads	количество дисковых блоков, прочитанных для этого SLRU-кеша
Hit	процент обнаружения дисковых блоков в SLRU-кеше
Writes	количество дисковых блоков, записанных для этого SLRU-кеша
Checked	количество блоков, проверенных на существование для этого SLRU-кеша
Flushes	количество сбросов «грязных» данных для этого SLRU-кеша.
Truncates	

4.1.4. Statement statistics by database (Статистика по обращениям к базе данных)

Таблица 4.4 содержит агрегированные общие статистические данные по базе данных pg_stat_statements (если расширение pg_stat_statements было доступно в течение интервала отчета).

Database	Calls	Time (s)				Fetched (blk)		Read (blk)		Dirtied (blk)		Temp (blk)		Local (blk)		Statements	WAL size
		Exec	Read	Write	Trg	Shared	Local	Shared	Local	Shared	Local	Read	Write	Read	Write		
postgres	70	0.46				157291				331						58	2912 kB
Total	70	0.46				157291				331						58	2912 kB

Рисунок 4.4 – Пример отчета Statement statistics by database

№ изменения: _____	Подпись отв. лица: _____	Дата внесения изм: _____
--------------------	--------------------------	--------------------------

Таблица 4.4 – Описание параметров отчета Statement statistics by database

Параметр		Описание
Database		имя БД
Calls		общее количество исполнений всех запросов (сумма вызовов).
Time (s)		затраченное время в секундах
	Plan	время, затраченное на планирование (сумма total_plan_time) – доступно с версии pg_stat_statements 1.8
	Exec	время, затраченное на выполнение (сумма total_time или total_exec_time)
	Read	время, затраченное на чтение блоков (сумма blk_read_time)
	Write	время, затраченное на запись блоков (сумма blk_write_time)
	Trg	время, затраченное на выполнение функций триггера
Fetched (blk)		общее количество блоков, извлеченных с диска и кэша буфера
	Shared	общее количество найденных общих блоков (сумма shared_blks_read + shared_blks_hit)
	Local	общее количество локальных блоков (сумма local_blks_read + local_blks_hit)
Read (blk)		общее количество блоков, прочитанных с диска и кэша буфера
	Shared	общее количество прочитанных общих блоков
	Local	общее количество прочитанных локальных блоков
Dirtied (blk)		общее количество блоков, заполненных в БД
	Shared	общее количество общих блоков, заполненных в БД (сумма shared_blks_dirtied)
	Local	общее количество локальных блоков, заполненных в БД (сумма local_blks_dirtied)
Temp (blk)		блоки, используемые для операций соединения и сортировки
	Read	прочитанные блоки (сумма temp_blks_read)
	Write	записанные блоки (сумма temp_blks_written)
Local (blk)		блоки, используемые для временных таблиц
	Read	прочитанные блоки (сумма local_blks_read)
	Write	записанные блоки (сумма local_blks_written)
Statements		общее количество перехваченных заявлений
WAL size		общий объем WAL, сгенерированный операторами (сумма wal_bytes)

4.1.5. Cluster statistics (Статистика кластера)

Таблица 4.5 содержит данные из представления pg_stat_bgwriter.

Metric	Value
Scheduled checkpoints	3
Requested checkpoints	
Checkpoints done	3
Checkpoint write time (s)	74.59
Checkpoint sync time (s)	0.02
Checkpoint buffers written	742
SLRU buffers written by checkpoint	8
Background buffers written	
Bgwriter interrupts (too many buffers)	
Number of buffers allocated	136
WAL generated	4288 kB
Start LSN	0/1ED2000
End LSN	0/2302000
WAL segments archived	
WAL segments archive failed	

Рисунок 4.5 – Пример отчета Cluster statistics

Таблица 4.5 – Описание параметров отчета Cluster statistics

Параметр	Описание
Scheduled checkpoints	общее количество контрольных точек, завершенных по расписанию благодаря параметру checkpoint_timeout (поле checkpoints_timed)
Requested checkpoints	общее количество других контрольных точек из-за значений параметров max_wal_size, archive_timeout и команды CHECKPOINT (поле checkpoints_req)
Checkpoints done	количество завершенных контрольных точек
Checkpoint write time(s)	общее время записи контрольных точек в секундах (поле checkpoint_write_time)
Checkpoint sync time(s)	общее время синхронизации контрольных точек в секундах (поле checkpoint_sync_time)
Checkpoints buffers written	общее количество буферов, записанных процессом Checkpointer (поле buffers_checkpoint)
SLRU buffers written by checkpoint	количество записанных буферов SLRU (Simple Least Recently Used) во время контрольной точки (checkpoint)
Background buffers written	общее количество буферов, записанных фоновым процессом записи (поле buffers_clean)
Bgwriter interrupts (too many buffers)	общее количество прерываний фонового писателя из-за достижения значения параметра bgwriter_lru_maxpages
Number of buffers allocated	общее количество выделенных буферов (поле buffers_alloc)
WAL generated	общее количество сгенерированных WAL (на основе pg_current_wal_lsn())
Start LSN	начальный номер LSN

№ изменения: _____	Подпись отв. лица: _____	Дата внесения изм: _____
--------------------	--------------------------	--------------------------

Параметр	Описание
End LSN	конечный номер LSN
WAL segments archived	количество архивированных сегментов WAL (на основе archived_count представления pg_stat_archiver)
WAL segments archive failed	количество неудач архивирования сегментов WAL (на основе failed_count представления pg_stat_archiver)

4.1.6. WAL statistics (Статистика WAL)

Таблица 4.6 содержит данные из представления pg_stat_wal1.

Metric	Value
WAL generated	3999 kB
WAL per second	4394 bytes
WAL records	16672
WAL FPI	257
WAL buffers full	
WAL writes	49
WAL writes per second	0.05
WAL sync	46
WAL syncs per second	0.05
WAL write time (s)	
WAL write duty	
WAL sync time (s)	
WAL sync duty	

Рисунок 4.6 – Пример отчета WAL statistics

Таблица 4.6 – Описание параметров отчета WAL statistics

Параметр	Описание
WAL generated	Общий объем сгенерированного WAL (wal_bytes)
WAL per second	Средний объем WAL, генерируемого в секунду
WAL records	Общее количество сгенерированных записей WAL (wal_records)
WAL FPI	Общее количество созданных полных изображений WAL-страниц (wal_fpi)
WAL buffers full	Количество раз, когда данные WAL записывались на диск из-за того, что буферы WAL заполнялись (wal_buffers_full)
WAL writes	Количество раз, когда буферы WAL записывались на диск с помощью запроса XLogWrite (wal_write)
WAL writes per second	Среднее количество раз, которое буферы WAL записываются на диск посредством запроса XLogWrite в секунду
WAL sync	Количество раз, когда файлы WAL были синхронизированы с диском посредством запроса issue_xlog_fsync

¹ для версии компонента 4.2

Параметр	Описание
WAL syncs per second	Среднее количество раз, когда файлы WAL синхронизировались с диском с помощью запроса <code>issue_xlog_fsync</code> в секунду
WAL write time (s)	Общее время, затраченное на запись буферов WAL на диск с использованием запроса <code>XLogWrite</code> , в секундах Это включает время синхронизации, когда <code>wal_sync_method</code> является <code>open_datasync</code> или <code>open_sync</code> (<code>wal_write_time</code>)
WAL write duty	Процентное соотношение времени записи WAL от общего времени отчета
WAL sync time (s)	Общее количество времени, затраченного на синхронизацию файлов WAL с диском посредством выполнения запроса <code>issue_xlog_fsync</code> , в секундах
WAL sync duty	Процентное соотношение времени синхронизации WAL от общего времени отчета

4.1.7. Tablespace statistics (Статистика табличных пространств)

Таблица 4.7 содержит информацию о размерах и росте табличных пространств.

Tablespace	Path	Size	Growth
pg_default		29 MB	808 kB
pg_global		549 kB	

Рисунок 4.7 – Пример отчета Tablespace statistics

Таблица 4.7 – Описание параметров отчета Tablespace statistics

Параметр	Описание
Tablespace	имя табличного пространства
Path	путь к табличному пространству
Size	размер табличного пространства на момент последнего снимка в интервале отчета
Growth	рост табличного пространства в течение интервала отчета

4.2. SQL Query statistics (Статистика SQL-запросов)

В данном разделе отчета содержатся таблицы топ запросов за интервал отчета, отсортированные по нескольким важным статистикам. Данные берутся из представления `pg_stat_statements`, если оно было доступно во время снятия снимка.

4.2.1. Top SQL by execution time (Топ SQL-запросов по времени выполнения)

Таблица 4.8 содержит топ запросы в количестве, которое указано в параметре `pg_profile.topn`, отсортированные по полю `total_time` (или `total_exec_time`) представления `pg_stat_statements`.

№ изменения: _____	Подпись отв. лица: _____	Дата внесения изм: _____
--------------------	--------------------------	--------------------------

Query ID	Database	User	Exec (s)	%Total	I/O time (s)		Rows	Execution times (ms)				Executions	%Cvr
					Read	Write		Mean	Min	Max	StdErr		
6f063ef5a156594 [e6bfa9c351]	postgres	postgres	0.42	91.25			2	212.12	208.13	216.11	3.99	2	100
c0e170b3804411d8 [61a7c10aab]	postgres	postgres	0.01	1.97			480	4.58	3.87	5.3	0.72	2	100
e78e3efd9edcc620 [d123620a7c]	postgres	postgres	0.01	1.52			658	3.53	3.52	3.54	0.01	2	100

Рисунок 4.8 – Пример отчета Top SQL by execution time

Таблица 4.8 – Описание параметров отчета Top SQL by execution time

Параметр		Описание
Query ID		идентификатор запроса в виде хэша БД, пользователя и текста запроса. Совместим с утилитой pgcenter. Родное поле pg_stat_statements queryid в шестнадцатеричной нотации показано в квадратных скобках
Database		имя БД запроса (получено из поля dbid)
User		имя пользователя БД
Exec(s)		время, затраченное на выполнение данного запроса (поле total_exec_time)
%Total		время выполнения данного запроса в процентах от общего времени выполнения всех операторов в кластере
I/O time(s):		
	Read	время, затраченное на чтение блоков (поле blk_read_time)
	Write	время, затраченное на запись блоков (поле blk_write_time)
Rows		количество строк, извлеченных или затронутых запросом (поле rows)
Execution times (ms)		подробная статистика времени выполнения данного запроса (в миллисекундах)
	Mean	среднее время выполнения данного запроса (поле mean_exec_time)
	Min	минимальное время, затраченное на выполнение данного запроса (поле min_exec_time)
	Max	максимальное время выполнения данного запроса (поле max_exec_time)
	StdErr	популяционное стандартное отклонение времени, затраченное на выполнение данного запроса (поле stddev_exec_time)
Executions		количество раз, за которое было выполнено данное запроса (поле calls)
%Cvr		охват: продолжительность сбора статистики по операторам в процентах от продолжительности отчетного интервала

4.2.2. Top SQL by mean execution time (Топ SQL-запросов по среднему времени выполнений)

Таблица 4.9 содержит топ запросы в количестве, которое указано в параметре pg_profile.topn, отсортированные по полю total_time (или total_exec_time) представления pg_stat_statements.

№ изменения: _____	Подпись отв. лица: _____	Дата внесения изм: _____
--------------------	--------------------------	--------------------------

Query ID	Database	User	Execution times (ms)				Exec (s)	%Total	I/O time (s)		Rows	Executions	%Cvr
			Mean	Min	Max	StdErr			Read	Write			
6f063ef5a156594 [e6bfa9c351]	postgres	postgres	212.12	208.13	216.11	3.99	0.42	91.25			2	2	100
c0e170b3804411d8 [61a7c10aab]	postgres	postgres	4.58	3.87	5.3	0.72	0.01	1.97			480	2	100
e78e3efd9edcc620 [d123620a7c]	postgres	postgres	3.53	3.52	3.54	0.01	0.01	1.52			658	2	100
9efde8cd54d9d346 [711ac4ec16]	postgres	postgres	2.4	2.23	2.56	0.17		1.03			6	2	
4dd87e7f9cd51b98 [fd1d75d066]	postgres	postgres	2.09	1.93	2.25	0.16		0.9			4	2	
4e43a3dea89c8c2 [193a531a61]	postgres	postgres	1.65	1.64	1.65	0.01		0.71			824	2	
56b14a3a00283994 [6228910825]	postgres	postgres	1.59	1.46	1.72	0.13		0.68			42	2	100

Рисунок 4.9 – Пример отчета Top SQL by mean execution time

Таблица 4.9 – Описание параметров отчета 4.2.4. Top SQL by mean execution time

Параметр		Описание
Query ID		идентификатор запроса в виде хэша БД, пользователя и текста запроса. Совместим с утилитой pgcenter. Родное поле pg_stat_statements queryid в шестнадцатеричной нотации показано в квадратных скобках
Database		имя БД запроса (получено из поля dbid)
User		имя пользователя БД
Exec(s)		время, затраченное на выполнение данного запроса (поле total_exec_time)
%Total		время выполнения данного запроса в процентах от общего времени выполнения всех операторов в кластере
I/O time(s):		
	Read	время, затраченное на чтение блоков (поле blk_read_time)
	Write	время, затраченное на запись блоков (поле blk_write_time)
Rows		количество строк, извлеченных или затронутых запросом (поле rows)
Execution times (ms)		подробная статистика времени выполнения данного запроса (в миллисекундах)
	Mean	среднее время выполнения данного запроса (поле mean_exec_time)
	Min	минимальное время, затраченное на выполнение данного запроса (поле min_exec_time)
	Max	максимальное время выполнения данного запроса (поле max_exec_time)
	StdErr	популяционное стандартное отклонение времени, затраченное на выполнение данного запроса (поле stddev_exec_time)
Executions		количество раз, за которое было выполнено данное запроса (поле calls)
%Cvr		охват: продолжительность сбора статистики по операторам в процентах от продолжительности отчётного интервала

4.2.3. Top SQL by executions (Топ SQL-запросов по количеству выполнений)

Таблица 4.10 содержит топ запросы в количестве, которое указано в параметре pg_profile.topn, отсортированные по полю calls представления pg_stat_statements.

Query ID	Database	User	Executions	%Total	Rows	Mean(ms)	Min(ms)	Max(ms)	StdErr(ms)	Elapsed(s)	%Cvr
e230f8b1f320897b [e8835a881a]	postgres	postgres	4	5.71		0.01	0.01	0.01			100
865fda18c2fc1bbb [ee0843dd06]	postgres	postgres	4	5.71	4	0.01	0.01	0.01			100
1318292377921ee6 [d27fb35ac4]	postgres	postgres	4	5.71		0.01	0.01	0.01			100
51582704af81013d [c2c79b4805]	postgres	postgres	4	5.71		0.01	0.01	0.01			100
a3e03f68e3e01562 [e666e859ae]	postgres	postgres	4	5.71							100

Рисунок 4.10 – Пример отчета Top SQL by executions

Таблица 4.10 – Описание параметров отчета Top SQL by executions

Параметр	Описание
Query ID	идентификатор запроса в виде хэша БД, пользователя и текста запроса. Совместим с утилитой pgcenter. Родное поле pg_stat_statements queryid в шестнадцатеричной нотации показано в квадратных скобках
Database	имя БД запроса (получено из поля dbid)
Executions	количество выполнений запроса (поле calls)
%Total	вызовы данного запроса в процентах от общего количества вызовов всех запросов в кластере
Rows	количество строк, извлеченных или затронутых запросом (поле rows)
Mean(ms)	среднее время, проведенное в запросе, в миллисекундах (поле mean_time или mean_exec_time)
Min(ms)	минимальное время, затраченное в запросе, в миллисекундах (поле min_time или min_exec_time)
Max(ms)	максимальное время, проведенное в запросе, в миллисекундах (поле max_time или max_exec_time)
StdErr(ms)	популяционное стандартное отклонение времени, проведенное в запросе, в миллисекундах (поле stddev_time или stddev_exec_time)
Elapsed(s)	количество времени, затраченное на выполнение данного запроса, в секундах (поле total_time или total_exec_time)
%Cvr	охват: продолжительность сбора статистики по операторам в процентах от продолжительности отчетного интервала

4.2.4. Top SQL by I/O wait time (Топ SQL-запросов по времени ожидания ввода/вывода)

Таблица 4.11 содержит топ запросы в количестве, которое указано в параметре `pg_profile.topn`, отсортированные по времени чтения и записи (`blk_read_time + blk_write_time`).

Query ID	Database	User	IO(s)	R(s)	W(s)	%Total	Reads			Writes			Elapsed(s)	Executions	%Cvr
							Shr	Loc	Tmp	Shr	Loc	Tmp			
6f063ef5a156594 [e6bfa9c351]	postgres	postgres	0.01	0.01		95.76	87			301			0.68	3	100

Рисунок 4.11 – Пример отчета Top SQL by I/O wait time

Таблица 4.11 – Описание параметров отчета Top SQL by I/O wait time

Параметр		Описание
Query ID		идентификатор запроса в виде хэша БД, пользователя и текста запроса. Совместим с утилитой <code>pgcenter</code> . Родное поле <code>pg_stat_statements queryid</code> в шестнадцатеричной нотации показано в квадратных скобках
Database		имя БД запроса (получено из поля <code>dbid</code>)
IO(s)		количество времени, затраченное на чтение и запись (время ввода/вывода) данным утверждением в секундах (<code>blk_read_time + blk_write_time</code>)
R(s)		количество времени, затраченное на чтение, данным оператором в секундах (<code>blk_read_time</code>)
W(s)		количество времени, затраченное на запись, данным оператором в секундах (<code>blk_write_time</code>)
%Total		время ввода/вывода данного оператора в процентах от общего времени ввода/вывода для всех операторов в кластере
Reads		количество блоков, прочитанных данным оператором, разделенное на три подстолбца
	Shr	общие чтения (поле <code>shared_blks_read</code>)
	Loc	локальные чтения (поле <code>local_blks_read</code>)
	Tmp	временные чтения (поле <code>temp_blks_read</code>)
Writes		количество блоков, записанных данным оператором, разделенное на три подстолбца
	Shr	общие записи (поле <code>shared_blks_written</code>)
	Loc	локальные записи (поле <code>local_blks_written</code>)
	Tmp	временные записи (поле <code>temp_blks_written</code>)
Elapsed(s)		количество времени, затраченное на выполнение данного запроса в секундах (поле <code>total_time</code> или <code>total_exec_time</code>)
Executions		количество выполнений для данного запроса (поле <code>calls</code>)
%Cvr		охват: продолжительность сбора статистики по операторам в процентах от продолжительности отчётного интервала

4.2.5. Top SQL by shared blocks fetched (Топ SQL-запросов по выбранным общим блокам)

Таблица 4.12 содержит топ запросы в количестве, которое указано в параметре `pg_profile.topn`, отсортированные по прочитанным и перехваченным блокам, помогающие обнаружить запросы которые обрабатывают наибольшее количество данных.




Query ID	Database	User	blks fetched	%Total	Hits(%)	Elapsed(s)	Rows	Executions	%Cvr
 6f063ef5a156594 [e6bfa9c351]	postgres	postgres	209676	91.35	99.96	0.68	3	3	100
 e78e3efd9edcc620 [d123620a7c]	postgres	postgres	10713	4.67	100	0.01	987	3	100
 c0e170b3804411d8 [61a7c10aab]	postgres	postgres	7968	3.47	100	0.01	720	3	100

Рисунок 4.12 – Пример отчета Top SQL by shared blocks fetched

Таблица 4.12 – Описание параметров отчета Top SQL by shared blocks fetched

Параметр	Описание
Query ID	идентификатор запроса в виде хэша БД, пользователя и текста запроса. Совместим с утилитой <code>pgcenter</code> . Родное поле <code>pg_stat_statements queryid</code> в шестнадцатеричной нотации показано в квадратных скобках
Database	имя БД запроса (получено из поля <code>dbid</code>)
blks fetched	количество найденных блоков (выражение <code>shared_blks_hit + shared_blks_read</code>)
%Total	количество блоков, найденных для данного запроса в процентах от общего количества блоков, найденных для всех утверждений в кластере
Hits(%)	процент блоков, полученных из буферов в пределах всех полученных блоков
Elapsed(s)	количество времени, затраченное на выполнение данного запроса, в секундах (поле <code>total_time</code> или <code>total_exec_time+total_plan_time</code>)
Rows	количество строк, извлеченных или затронутых данным запросом (поле <code>rows</code>)
Executions	количество выполнений для данного запроса (поле <code>calls</code>)
%Cvr	охват: продолжительность сбора статистики по операторам в процентах от продолжительности отчётного интервала

4.2.6. Top SQL by shared blocks read (Топ SQL-запросов по количеству прочитанных разделяемых блоков)

Таблица 4.13 содержит топ запросы в количестве, которое указано в параметре `pg_profile.topn`, отсортированные по общим прочитанным блокам, помогающие обнаружить наиболее интенсивные по чтению операторы.



Query ID	Database	User	Reads	%Total	Hits(%)	Elapsed(s)	Rows	Executions	%Cvr
 6f063ef5a156594 [e6bfa9c351]	postgres	postgres	87	71.31	99.96	0.68	3	3	100
 56b14a3a00283994 [6228910825]	postgres	postgres	35	28.69	96.82	0.01	59	3	100

Рисунок 4.13 – Пример отчета Top SQL by shared blocks read

Таблица 4.13 – Описание параметров отчета Top SQL by shared blocks read

Параметр	Описание
Query ID	идентификатор запроса в виде хэша БД, пользователя и текста запроса. Совместим с утилитой pgcenter. Родное поле pg_stat_statements queryid в шестнадцатеричной нотации показано в квадратных скобках
Database	имя БД запроса (получено из поля dbid)
Reads	количество общих блоков чтения для данного запроса (поле shared_blks_read)
%Total	количество общих чтений для данного запроса в процентах от общего количества общих чтений для всех утверждений в кластере
Hits(%)	процент блоков, полученных из буферов, среди всех полученных блоков
Elapsed(s)	количество времени, затраченное на выполнение данного запроса в секундах (поле total_time или total_exec_time+total_plan_time)
Rows	количество строк, извлеченных или затронутых данным запросом (поле rows)
Executions	количество выполнений для данного запроса (поле calls)
%Cvr	охват: продолжительность сбора статистики по операторам в процентах от продолжительности отчетного интервала

4.2.7. Top SQL by shared blocks dirtied (Топ SQL-запросов по заполненным разделяемым блокам)

Таблица 4.14 содержит топ запросы в количестве, которое указано в параметре pg_profile.topn, отсортированные по количеству общих заполненных буферов, помогающие обнаружить наиболее изменяющие данные запросы.



Query ID	Database	User	Dirtied	%Total	Hits	%Total	WAL	%Total	Elapsed(s)	Rows	Executions	%Cvr
 6f063ef5a156594 [e6bfa9c351]	postgres	postgres	582	92.82	209589	99.96	3985 kB	66.42	0.68	3	3	100
 56b14a3a00283994 [6228910825]	postgres	postgres	45	7.18	1065	96.82	352 kB	5.87	0.01	59	3	100

Рисунок 4.14 – Пример отчета Top SQL by shared blocks dirtied

Таблица 4.14 – Описание параметров отчета Top SQL by shared blocks dirtied

Параметр	Описание
Query ID	идентификатор запроса в виде хэша БД, пользователя и текста запроса. Совместим с утилитой pgcenter. Родное поле pg_stat_statements queryid в шестнадцатеричной нотации показано в квадратных скобках
Database	имя БД запроса (получено из поля dbid)
№ изменения: _____	
Подпись отв. лица: _____	
Дата внесения изм: _____	

Параметр	Описание
User	Имя пользователя БД
Dirtied	количество общих блоков, заполненных этим запросом (поле shared_blks_dirtied)
%Total	количество общих блоков, заполненных этим запросом, в процентах от общего количества общих блоков, заполненными всеми операторами в кластере
Hits	количество блоков, полученных из буферов, среди всех полученных блоков
%Total	отношение числа блоков, полученных из буферов, к общему числу блоков, полученных при выполнении этого плана
WAL	общий объём WAL (в байтах), сгенерированный при выполнении плана
%Total	отношение объёма WAL, сгенерированного при выполнении плана, ко всему объёму WAL, сгенерированному в кластере
Elapsed(s)	количество времени, затраченное на выполнение данного запроса в секундах (поле total_time или total_exec_time+total_plan_time)
Rows	количество строк, извлеченных или затронутых данным запросом (поле rows)
Executions	количество выполнений для данного запроса (поле calls)
%Cvr	охват: продолжительность сбора статистики по операторам в процентах от продолжительности отчётного интервала

4.2.8. Top SQL by shared blocks written (Топ SQL-запросов по записи общих блоков)

Таблица 4.15 содержит топ запросы в количестве, которое указано в параметре pg_profile.topn, выполняющие записи отсортированные по количеству записанных блоков.


Query ID	Database	User	Written	%Total	%BackendW	Hits(%)	Elapsed(s)	Rows	Executions	%Cvr
 6f063ef5a156594 [e6bfa9c351]	postgres	postgres	301	39.92	7525	99.96	0.68	3	3	100

Рисунок 4.15 – Пример отчета Top SQL by shared blocks written

Таблица 4.15 – Описание параметров отчета Top SQL by shared blocks written

Параметр	Описание
Query ID	идентификатор запроса в виде хэша БД, пользователя и текста запроса. Совместим с утилитой pgcenter. Родное поле pg_stat_statements queryid в шестнадцатеричной нотации показано в квадратных скобках
Database	имя БД запроса (получено из поля dbid)
User	Имя пользователя
Written	количество блоков, записанных данным запросом (поле shared_blks_written)
%Total	количество блоков, записанных данным запросом в процентах от общего количества блоков, записанных всеми операторами в кластере
%BackendW	количество блоков, записанных данным запросом, в процентах от всех блоков, записанных backend в кластере (поле buffers_backend представления pg_stat_bgwriter)
Hits(%)	процент блоков, полученных из буферов, среди всех полученных блоков

№ изменения: _____	Подпись отв. лица: _____	Дата внесения изм: _____
--------------------	--------------------------	--------------------------

Параметр	Описание
Elapsed(s)	количество времени, затраченное на выполнение данного запроса в секундах (поле total_time или total_exec_time+total_plan_time)
Rows	количество строк, извлеченных или затронутых данным запросом (поле rows)
Executions	количество выполнений данного запроса (поле calls)
%Cvr	охват: продолжительность сбора статистики по операторам в процентах от продолжительности отчётного интервала

4.2.9. Top SQL by WAL size (Топ SQL-запросов по размеру WAL)

Таблица 4.16 содержит топ запросы в количестве, которое указано в параметре pg_profile.topn, отсортированные по размеру WAL (доступно в pg_stat_statements, начиная с версии 1.8).



Query ID	Database	User	WAL	%Total	Dirtied	WAL FPI	WAL records	%Cvr
 6f063ef5a156594 [e6bfa9c351]	postgres	postgres	3985 kB	66.42	582	180	20093	100
 56b14a3a00283994 [6228910825]	postgres	postgres	352 kB	5.87	45	45	45	100

Рисунок 4.16 – Пример отчета Top SQL by WAL size

Таблица 4.16 – Описание параметров отчета Top SQL by WAL size

Параметр	Описание
Query ID	идентификатор запроса в виде хэша БД, пользователя и текста запроса. Совместим с утилитой pgcenter. Родное поле pg_stat_statements queryid в шестнадцатеричной нотации показано в квадратных скобках
Database	имя БД запроса (получено из поля dbid)
User	Имя пользователя
WAL	объем WAL, сгенерированный запросом (поле wal_bytes)
%Total	объем WAL, сгенерированный оператором, как процент от общего объема WAL, сгенерированного в кластере (приращение pg_current_wal_lsn())
Dirtied	количество общих блоков, заполненных этим запросом (поле shared_blks_dirtied)
WAL FPI	общее количество полностраничных образов WAL, сгенерированных этим запросом (поле wal_fpi)
WAL records	общее количество байт WAL, сгенерированных запросом (поле wal_bytes)
%Cvr	охват: продолжительность сбора статистики по операторам в процентах от продолжительности отчётного интервала

4.2.10. Complete list of SQL texts (Полный список текстов SQL)

Таблица содержит полный текст SQL-запросов и состоит из столбцов:

- Query ID – идентификатор запроса в виде хэша БД;

№ изменения: _____	Подпись отв. лица: _____	Дата внесения изм: _____
--------------------	--------------------------	--------------------------

– Query Text – текст SQLзапроса.

Query ID	Query Text
838639b6b471ed84	SELECT now() as subsample ts, backend_type, datid, datname, usesysid, username, application_name, client_addr, count(*) count(*) FILTER (WHERE state = \$1) as active, count(*) FILTER (WHERE state = \$2) as idle, count(*) FILTER (WHERE state idle_t, count(*) FILTER (WHERE state = \$4) as idle ta, count(*) FILTER (WHERE state IS NULL) as state null, count(*) F wait_event_type = \$5) as lwlock, count(*) FILTER (WHERE wait_event_type = \$6) as lock, count(*) FILTER (WHERE wait eve as bufferpin, count(*) FILTER (WHERE wait_event_type = \$8) as activity, count(*) FILTER (WHERE wait_event_type = \$9) a count(*) FILTER (WHERE wait_event_type = \$10) as client, count(*) FILTER (WHERE wait_event_type = \$11) as ipc, count(*) (WHERE wait_event_type = \$12) as timeout, count(*) FILTER (WHERE wait_event_type = \$13) as io FROM pg_stat_activity GR backend_type, datid, datname, usesysid, username, application_name, client_addr
865fda18c2fc1bbb	SELECT current_setting(\$1)::interval = \$2::interval
9efde8cd54d9d346	SELECT dbs.datid, dbs.datname, dbs.xact commit, dbs.xact rollback, dbs.blks_read, dbs.blks_hit, dbs.tup returned, dbs. dbs.tup inserted, dbs.tup updated, dbs.tup deleted, dbs.conflicts, dbs.temp_files, dbs.temp bytes, dbs.deadlocks, dbs.checksum failures, dbs.checksum last failure, dbs.blk read time, dbs.blk write time, dbs.session time, dbs.active_ dbs.idle_in_transaction time, dbs.sessions, dbs.sessions_abandoned, dbs.sessions_fatal, dbs.sessions_killed, dbs.parallel_workers_to_launch, dbs.parallel_workers_launched, dbs.stats_reset, pg_database_size(dbs.datid) as datsize datsize delta, db.datistemplate, db.dattablespace, db.dataallowconn FROM pg_catalog.pg_stat_database dbs JOIN pg_catalo db ON (dbs.datid = db.oid) WHERE dbs.datname IS NOT NULL
a3e03f68e3e01562	SET application_name=\$1

Рисунок 4.17 - Пример отчета Complete list of SQL texts

4.3. Schema object statistics (Статистика объекта схемы)

Данный раздел отчета содержит топ объектов БД, используя статистику из представлений Statistics Collector.

4.3.1. Top tables by estimated sequentially scanned volume (Топ таблиц по предполагаемому объему последовательного сканирования)

Таблица 4.18 содержит топ запросы БД, отсортированные по предполагаемому объему, прочитанному последовательным сканированием. Основано на представлении pg_stat_all_tables. Здесь можно искать таблицы, в которых, возможно, отсутствует какой-либо индекс.

DB	Tablespace	Schema	Table	~SeqBytes	SeqScan	IxScan	IxFet	Ins	Upd	Del	Upd(HOT)
postgres	pg_default	pg_catalog	pg_attribute	[29 MB]	33	1866	5993				
postgres	pg_default	pg_catalog	pg_class	[16 MB]	82	3726	3206				
postgres	pg_default	public	last_stat_database_srv1	[11 MB]	1360	932	932	6		6	
postgres	pg_default	public	last_stat_tables_srv1	[10200 kB]	30	5921	37416	480	805	480	10
postgres	pg_default	public	last_stat_tables_srv1(TOAST)								
postgres	pg_default	public	tables_list	[8600 kB]	215	488	424	64	68		
postgres	pg_default	public	last_stat_indexes_srv1	[7392 kB]	24	3800	1746	658	799	658	11
postgres	pg_default	public	last_stat_indexes_srv1(TOAST)								

Рисунок 4.18 – Пример отчета Top tables by estimated sequentially scanned volume

Таблица 4.17 – Описание параметров отчета Top tables by estimated sequentially scanned volume

Параметр	Описание
Database	имя БД, в которой находится таблица
Tablespace	имя табличного пространства, в котором находится таблица
Schema	имя схемы таблицы
Table	имя таблицы

№ изменения: _____	Подпись отв. лица: _____	Дата внесения изм: _____
--------------------	--------------------------	--------------------------

Параметр	Описание
~SeqBytes	предполагаемый объем, прочитанный при последовательном сканировании. Рассчитывается как сумма размера отношения, умноженная на seq_scan для всех снапшотов отчета
SeqScan	количество последовательных сканирований, выполненных над таблицей (поле seq_scan)
IxScan	количество индексных сканирований, инициированных для этой таблицы (поле idx_scan)
IxFet	количество живых строк, извлеченных при индексном сканировании (поле idx_tup_fetch)
Ins	количество вставленных строк (поле n_tup_ins)
Upd	количество обновленных строк (включая HOT) (поле n_tup_upd)
Del	количество удаленных строк (поле n_tup_del)
Upd(HOT)	количество строк, обновленных по HOT (поле n_tup_hot_upd)

4.3.2. Top tables by blocks fetched (Топ таблиц по выбранным блокам)

Таблица 4.19 содержит захваченный блок – это блок, обрабатываемый с диска (read) или из общих буферов (hit). Таблицы в этом списке сортируются по сумме найденных блоков для отношения таблицы, ее индексов, TOAST-таблицы (если существует) и индекса TOAST (если существует). Этот раздел может привлечь внимание к таблицам с избыточной обработкой блоков. Основано на данных представления pg_statio_all_tables.

DB	Tablespace	Schema	Table	Heap		Ix		TOAST		TOAST-Ix	
				Blks	%Total	Blks	%Total	Blks	%Total	Blks	%Total
postgres	pg_default	public	last_stat_tables_srv1	68747	38.26	14728	8.2				
postgres	pg_default	public	last_stat_indexes_srv1	11330	6.31	10710	5.96				
postgres	pg_default	pg_catalog	pg_class	4704	2.62	7477	4.16				
postgres	pg_default	public	sample_stat_database	6810	3.79	3408	1.9				
postgres	pg_default	pg_catalog	pg_statistic	4588	2.55	4527	2.52	44	0.02	46	0.03

Рисунок 4.19 – Пример отчета Top tables by blocks fetched

Таблица 4.18 – Описание параметров отчета Top tables by blocks fetched

Параметр	Описание
Database	имя БД таблицы
Tablespace	имя табличного пространства, в котором находится таблица
Schema	имя схемы таблицы
Table	имя таблицы
Heap	статистика по найденным блокам таблиц (heap_blks_read + heap_blks_hit)
Ix	статистика для всех найденных блоков индексов таблиц (idx_blks_read + idx_blks_hit)
TOAST	статистика для блоков, полученных для TOAST-таблиц (toast_blks_read + toast_blks_hit)

Параметр	Описание
TOAST-Ix	статистика для найденных блоков индекса TOAST (tidx_blks_read + tidx_blks_hit)
Поля Heap и Ix в этой таблице разделены на два столбца	
Blks	количество блоков, найденных для heap таблиц, индекса, TOAST или индекса TOAST
%Total	блоки, найденные для heap таблиц, индекса, TOAST или индекса TOAST в процентах от всех блоков, найденных во всем кластере

4.3.3. Top tables by blocks read (Топ таблиц по прочитанным блокам)

Таблица 4.20 содержит топ таблицы, отсортированные по количеству прочитанных блоков. Таблицы в этом списке отсортированы по сумме прочитанных блоков для таблицы, ее индексов, TOAST-таблицы (если существует) и индекса TOAST (если существует). Этот раздел может сфокусировать внимание на таблицах с чрезмерным количеством прочитанных блоков. Основано на данных представления pg_statio_all_tables.

DB	Tablespace	Schema	Table	Heap		Ix		TOAST		TOAST-Ix		Hit(%)
				Blks	%Total	Blks	%Total	Blks	%Total	Blks	%Total	
postgres	pg_default	pg_catalog	pg_proc	56	38.89	6	4.17					98.11
postgres	pg_default	pg_catalog	pg_rewrite	7	4.86			9	6.25			89.68
postgres	pg_default	pg_toast	pg_toast_2618	9	6.25							85
postgres	pg_default	pg_catalog	pg_range			4	2.78					
postgres	pg_default	public	sample_stat_tables			3	2.08					99.98

Рисунок 4.20 – Пример отчета Top tables by blocks read

Таблица 4.19 – Описание параметров отчета Top tables by blocks read

Параметр	Описание
Database	имя БД таблицы
Tablespace	имя табличного пространства, в котором находится таблица
Schema	имя схемы таблицы
Table	имя таблицы
Heap	статистика чтения блоков таблиц (heap_blks_read)
Ix	статистика чтения блоков всех индексов таблицы (idx_blks_read)
TOAST	статистика чтений блоков TOAST-таблиц (toast_blks_read)
TOAST-Ix	статистика чтения блоков индексов TOAST (tidx_blks_read)
Hit(%)	процент блоков, полученных из буферов, среди всех полученных блоков
Поля Heap и Ix в этой таблице разделены на два столбца	
Blks	количество блочных чтений для heap таблиц, index, TOAST или индекса TOAST
%Total	блочные чтения для heap таблиц, индекса, TOAST или индекса TOAST как процент от всех блочных чтений во всем кластере

4.3.4. Top DML tables (Топ таблиц по количеству операций DML)

Таблица 4.21 содержит топ таблицы, отсортированные по количеству строк, затронутых DML, т.е. сумма n_tup_ins, n_tup_upd и n_tup_del (включая таблицы TOAST).

DB	Tablespace	Schema	Table	Ins	Upd	Del	Upd(HOT)	SeqScan	SeqFet	IxScan	IxFet
postgres	pg_default	public	last_stat_indexes_srv1	987	866	658	11	33	18753	4596	1814
			last_stat_indexes_srv1(TOAST)								
postgres	pg_default	public	last_stat_tables_srv1	720	947	480	10	39	16560	7487	54221
			last_stat_tables_srv1(TOAST)								
postgres	pg_default	pg_catalog	pg_statistic	346	910		172			2708	1594
			pg_statistic(TOAST)	10		9				38	28

Рисунок 4.21 – Пример отчета Top DML tables

Таблица 4.20 – Описание параметров отчета Top DML tables

Параметр	Описание
Database	имя БД таблиц
Tablespace	имя табличного пространства, в котором находится таблица
Schema	имя схемы таблицы
Table	имя таблицы
Ins	количество вставленных строк (поле n_tup_ins)
Upd	количество обновленных строк (включая HOT) (поле n_tup_upd)
Del	количество удаленных строк (поле n_tup_del)
Upd(HOT)	количество обновленных строк HOT (поле n_tup_hot_upd)
SeqScan	количество последовательных сканирований, выполненных над таблицей (поле seq_scan)
SeqFet	количество живых строк, извлеченных в результате последовательного сканирования (поле seq_tup_read)
IxScan	количество индексных сканирований, инициированных для этой таблицы (поле idx_scan)
IxFet	количество живых строк, полученных при индексном сканировании (поле idx_tup_fetch)

4.3.5. Top tables by updated/deleted tuples (Топ таблиц по обновленным/удаленным записям)

Таблица 4.22 содержит топ таблицы, отсортированные по количеству операций, вызывающих автовакуумную нагрузку, т.е. сумма n_tup_upd и n_tup_del (включая таблицы TOAST).

DB	Tablespace	Schema	Table	Upd	Upd(HOT)	Del	Vacuum count		Analyze count	
							Vacuum	AutoVacuum	Analyze	AutoAnalyze
postgres	pg_default	public	last_stat_indexes_srv1	866	11	658		2	3	3
			last_stat_indexes_srv1(TOAST)							
postgres	pg_default	public	last_stat_tables_srv1	947	10	480		3	3	3
			last_stat_tables_srv1(TOAST)							
postgres	pg_default	pg_catalog	pg_statistic	910	172			3		
			pg_statistic(TOAST)			9				
postgres	pg_default	public	last_stat_statements_srv1	152	6	47		2	3	3
			last_stat_statements_srv1(TOAST)							

Рисунок 4.22 – Пример отчета Top tables by updated/deleted tuples

Таблица 4.21 – Описание параметров отчета Top tables by updated/deleted tuples

Параметр	Описание
Database	имя БД таблицы
Tablespace	имя табличного пространства, в котором находится таблица
Schema	имя схемы таблицы
Table	имя таблицы
Upd	количество обновленных строк (включая HOT) (поле n_tup_upd)
Upd(HOT)	количество строк, обновленных HOT (поле n_tup_hot_upd)
Del	количество удаленных строк (поле n_tup_del)
Vacuum	количество раз, когда эта таблица была обработана вручную (не считая VACUUM FULL) (поле vacuum_count)
AutoVacuum	количество раз, когда эта таблица была обработана фоновым процессом autovacuum (поле autovacuum_count)
Analyze	количество раз, когда эта таблица была проанализирована вручную (поле analyze_count)
AutoAnalyze	количество раз, когда эта таблица была проанализирована процессом autovacuum (поле autoanalyze_count)

4.3.6. Top tables by new-page updated tuples (Таблицы с наибольшим количеством изменённых кортежей, попавших на новую страницу)

Таблица 4.22 показывает таблицы с наибольшим количеством изменённых строк, новая версия которых переходит на новую страницу кучи, оставляя исходную версию с полем t_ctid, которое указывает на другую страницу кучи. Учитываются только изменения не по схеме HOT.

DB	Tablespace	Schema	Table	NP Upd	%Upd	Upd	Upd (HOT)
postgres	pg_default	public	last_stat_tables_srv1	937	98.9	947	10
			last_stat_tables_srv1(TOAST)				
postgres	pg_default	public	last_stat_indexes_srv1	855	98.7	866	11
			last_stat_indexes_srv1(TOAST)				
postgres	pg_default	pg_catalog	pg_statistic	738	81.1	910	172
			pg_statistic(TOAST)				
postgres	pg_default	public	last_stat_statements_srv1	146	96.1	152	6
			last_stat_statements_srv1(TOAST)				

Рисунок 4.23 – Пример отчета Top tables by new-page updated tuples

Таблица 4.22 – Описание параметров отчета Top tables by new-page updated tuples

Параметр	Описание
Database	имя БД таблицы
Tablespace	имя табличного пространства, в котором находится таблица
Schema	имя схемы таблицы
Table	имя таблицы
NP Upd	количество изменённых строк, попавших на новую страницу кучи
%Upd	количество изменённых строк, попавших на новую страницу, в процентах от количества всех изменённых строк
Upd	количество изменённых строк, включая изменения по схеме HOT
Upd(HOT)	количество строк, обновленных по HOT (поле n_tup_hot_upd)

4.3.7. Top growing tables (Топ таблиц по увеличению размера)

Таблица 4.23 показывает таблицы, которые увеличились в объёме больше других. Эта информация основана на представлении pg_stat_all_tables. В отсутствие данных о размере, собираемых функцией pg_relation_size(), оценка размера берётся из поля pg_class.relpages. Для обозначения меньшей точности оценки она отображается в квадратных скобках..

DB	Tablespace	Schema	Table	Size	Growth	Ins	Upd	Del	Upd(HOT)
postgres	pg_default	public	last_stat_tables_srv1	[288 kB]	[288 kB]	720	947	480	10
			last_stat_tables_srv1(TOAST)						
postgres	pg_default	public	last_stat_indexes_srv1	[240 kB]	[240 kB]	987	866	658	11
			last_stat_indexes_srv1(TOAST)						
postgres	pg_default	pg_catalog	pg_statistic	[416 kB]	[192 kB]	346	910		172
			pg_statistic(TOAST)	[24 kB]		10		9	
postgres	pg_default	public	sample_stat_tables	[144 kB]	[144 kB]	467			

Рисунок 4.24 – Пример отчета Top growing tables

Таблица 4.23 – Описание параметров отчета Top growing tables

Параметр	Описание
Database	имя БД таблицы
Tablespace	имя табличного пространства, в котором находится таблица
Schema	имя схемы таблицы

№ изменения: _____	Подпись отв. лица: _____	Дата внесения изм: _____
--------------------	--------------------------	--------------------------

Параметр	Описание
Table	имя таблицы
Size	размер таблицы, каким он был на момент последнего снапшота в интервале отчета
Growth	рост таблицы
Ins	количество вставленных строк (поле n_tup_ins)
Upd	количество обновленных строк (включая HOT) (поле n_tup_upd)
Del	количество удаленных строк (поле n_tup_del)
Upd(HOT)	количество строк, обновленных по HOT (поле n_tup_hot_upd)

4.3.8. Top indexes by blocks fetched (Топ индексов по выбранным блокам)

Таблица 4.25 содержит список, где полученными блоками считаются блоки как прочитанные с диска (read), так и найденные в общих буферах (hit). Эта информация основана на представлении pg_statio_all_indexes.

DB	Tablespace	Schema	Table	Index	Scans	Blks	%Total
postgres	pg_default	public	last_stat_tables_srv1	pk_last_stat_tables_srv1	7487	17416	6.72
postgres	pg_default	public	last_stat_indexes_srv1	pk_last_stat_indexes_srv1	4596	12810	4.94
postgres	pg_default	pg_catalog	pg_class	pg_class_oid_index	5534	11091	4.28
postgres	pg_default	pg_catalog	pg_statistic	pg_statistic_relid_att_inh_index	2708	7947	3.07
postgres	pg_default	pg_catalog	pg_attribute	pg_attribute_relid_attnum_index	3065	6159	2.38
postgres	pg_default	public	sample_stat_database	pk_sample_stat_database	4441	4452	1.72

Рисунок 4.25 – Пример отчета Top indexes by blocks fetched

Таблица 4.24 – Описание параметров отчета Top indexes by blocks fetched

Параметр	Описание
Database	имя БД индекса
Tablespace	имя табличного пространства, в котором расположен индекс
Schema	имя схемы индекса
Table	имя таблицы
Index	имя индекса
Scans	количество сканирований, выполненных по индексу (поле idx_scan)
Blks	блоки, извлеченные из данного индекса (idx_blks_read + idx_blks_hit)
%Total	блоки, считанные для этого индекса, в процентах от всех блоков, считанных во всем кластере

4.3.9. Top indexes by blocks read (Топ индексов по прочитанным блокам)

Таблица 4.26 содержит топ индексы, отсортированные по количеству прочитанных блоков. Основано на данных представления pg_statio_all_indexes.

DB	Tablespace	Schema	Table	Index	Scans	Blks	Reads	%Total	Hits(%)
postgres	pg_default	pg_catalog	pg_proc	pg_proc_prname_args_nsp_index	121		4	2.78	98.4
postgres	pg_default	pg_catalog	pg_range	pg_range_rngmultitypid_index	1		2	1.39	
postgres	pg_default	pg_catalog	pg_proc	pg_proc_oid_index	666		2	1.39	99.85
postgres	pg_default	pg_catalog	pg_range	pg_range_rngtypid_index	1		2	1.39	
postgres	pg_default	public	sample_stat_io	pk_sample_stat_io			1	0.69	96.77
postgres	pg_default	public	sample_stat_slru	pk_sample_stat_slru			1	0.69	93.33

Рисунок 4.26 – Пример отчета Top indexes by blocks read

Таблица 4.25 – Описание параметров отчета Top indexes by blocks read

Параметр	Описание
Database	имя БД индекса
Tablespace	имя табличного пространства, в котором расположен индекс
Schema	имя схемы индекса
Table	имя таблицы
Index	имя индекса
Scans	количество сканирований, выполненных по индексу (поле idx_scan)
Blk Reads	количество дисковых блоков, прочитанных из данного индекса (поле idx_blks_read)
%Total	чтение блоков из данного индекса в процентах от всех чтений блоков во всем кластере
Hits(%)	процент индексных блоков, полученных из буферного кэша, среди всех индексных блоков, найденных для этого индекса

4.3.10. Top growing indexes (Топ таблиц по увеличению объемов индексов)

Таблица 4.27 содержит топ индексы, отсортированные по росту.

DB	Tablespace	Schema	Table	Index	Index		Table		
					Size	Growth	Ins	Upd	Del
postgres	pg_default	public	last_stat_indexes_srv1	pk_last_stat_indexes_srv1	[80 kB]	[72 kB]	987	866	658
postgres	pg_default	public	last_stat_tables_srv1	pk_last_stat_tables_srv1	[64 kB]	[56 kB]	720	947	480
postgres	pg_default	public	sample_stat_tables	ix_sample_stat_tables_rel	[40 kB]	[32 kB]	467		
postgres	pg_default	public	sample_settings	pk_sample_settings	[40 kB]	[32 kB]	413		
postgres	pg_default	public	sample_stat_tables	pk_sample_stat_tables	[32 kB]	[24 kB]	467		
postgres	pg_default	pg_catalog	pg_statistic	pg_statistic_relid_att_inh_index	[56 kB]	[16 kB]	346	910	

Рисунок 4.27 – Пример отчета Top growing indexes

Таблица 4.26 – Описание параметров отчета Top growing indexes

Параметр	Описание
Database	имя БД индекса
Tablespace	имя табличного пространства, в котором находится индекс
Schema	имя схемы индекса
Table	имя таблицы
Index	имя индекса
Index	статистика индекса

№ изменения: _____	Подпись отв. лица: _____	Дата внесения изм: _____
--------------------	--------------------------	--------------------------

Параметр		Описание
	Size	размер индекса, каким он был на момент последнего снимка в интервале отчета
	Growth	рост индекса в течение интервала отчета
Table		статистика базовой таблицы
	Ins	количество вставленных строк в базовую таблицу (поле n_tup_ins)
	Upd	количество строк, обновленных в базовой таблице (без HOT) (n_tup_upd - n_tup_hot_upd)
	Del	количество строк, удаленных из базовой таблицы (поле n_tup_del)

4.3.11. Unused indexes (Неиспользуемые индексы)

Таблица 4.28 содержит неиспользуемые индексы в течение интервала отчета, отсортированные по операциям DML над базовыми таблицами, вызывающими поддержку индексов. Индексы ограничений исключены.

DB	Tablespace	Schema	Table	Index	Index		Table		
					Size	Growth	Ins	Upd	Del
postgres	pg_default	public	sample_stat_indexes	ix_sample_stat_indexes_il	[16 kB]	[8192 bytes]	208		
postgres	pg_default	public	sample_stat_indexes	ix_sample_stat_indexes_ts	[16 kB]	[8192 bytes]	208		
postgres	pg_default	public	sample_stat_user_functions	ix_sample_stat_user_functions_fl	[8192 bytes]		39		

Рисунок 4.28 – Пример отчета Unused indexes

Таблица 4.27 – Описание параметров отчета Unused indexes

Параметр		Описание
Database		имя БД индекса
Tablespace		имя табличного пространства, в котором расположен индекс
Schema		имя схемы индекса
Table		имя таблицы
Index		имя индекса
Index		статистика индекса
	Size	размер индекса, каким он был на момент последнего снимка в интервале отчета
	Growth	рост индекса в течение интервала отчета
Table		статистика базовой таблицы
	Ins	количество вставленных строк в базовую таблицу (поле n_tup_ins)
	Upd	количество строк, обновленных в базовой таблице (без HOT) (n_tup_upd - n_tup_hot_upd)
	Del	количество строк, удаленных из базовой таблицы (поле n_tup_del)

4.4. User function statistics (Статистика функций пользователя)

Данный раздел отчета содержит топ функции в кластере, основанные на представлении pg_stat_user_functions, которое наполняется данными о статистике выполнения функций.

№ изменения: _____	Подпись отв. лица: _____	Дата внесения изм: _____
--------------------	--------------------------	--------------------------

4.4.1. Top functions by total time (Топ функций по общему времени)

Таблица 4.29 содержит топ функций, отсортированных по затраченному времени.

DB	Schema	Function	Executions	Time (s)			
				Total	Self	Mean	Mean self
postgres	public	take_sample	3	0.68		0.23	
postgres	public	take_sample_subset	3	0.68		0.23	
postgres	public	take_sample	3	0.68	0.07	0.23	0.02
postgres	public	sample_dbobj_delta	3	0.23	0.23	0.08	0.08
postgres	public	collect_obj_stats	3	0.16	0.05	0.05	0.02
postgres	public	dblink	107	0.14	0.14		

Рисунок 4.29 – Пример отчета Top functions by total time

Таблица 4.28 – Описание параметров отчета Top functions by total time

Параметр		Описание
Database		имя БД функции
Schema		имя схемы индекса
Function		имя функции
Executions		количество вызовов данной функции (поле calls)
Time (s)		статистика времени выполнения функции в секундах
	Total	общее время, затраченное на эту функцию и все другие вызванные ей функции (поле total_time)
	Self	общее время, затраченное на выполнение самой функции без учета других вызванных ей функций (поле self_time)
	Mean	среднее время выполнения одной функции
	Mean self	среднее собственное время выполнения одной функции

4.4.2. Top functions by executions (Топ функций по исполнению)

Таблица 4.30 содержит топ функции, отсортированные по количеству выполнений.

DB	Schema	Function	Executions	Time (s)			
				Total	Self	Mean	Mean self
postgres	public	dblink	107	0.14	0.14		
postgres	public	dblink_connect	6	0.08	0.08	0.01	0.01
postgres	public	dblink_disconnect	6	0.01	0.01		
postgres	public	dblink_get_connections	6				
postgres	public	pg_stat_statements	6				
postgres	public	pg_stat_statements_reset	3				
postgres	public	get_connstr	3	0.01	0.01		

Рисунок 4.30 – Пример отчета Top functions by executions

Таблица 4.29 – Описание параметров отчета Top functions by executions

Параметр		Описание
Database		имя БД функции
Schema		имя схемы индекса
Function		имя функции
Executions		количество вызовов данной функции (поле calls)
Time(s)		статистика времени выполнения функции в секундах
	Total	общее время, затраченное на эту функцию и все другие вызванные ей функции (поле total_time)
	Self	общее время, затраченное на выполнение самой функции, без учета других вызванных ей функций (поле self_time)
	Mean	среднее время выполнения одной функции
	Mean self	среднее собственное время выполнения одной функции

4.5. Vacuum-related stats (Статистика, связанная с вакуумом)

Данный раздел отчета содержит статистику работы процесса autovacuum. Если во время работы между снимками не было вакуумных операций – данный раздел останется пустым.

4.5.1. Top tables by vacuum operations (Топ таблиц по статистике вакуума)

Таблица 4.31 содержит топ таблицы, отсортированные по очистке (ручной и автоматической).

DB	Tablespace	Schema	Table	Vacuum count		Vacuum time (s)		Ins	Upd	Del	Upd(HOT)
				Vacuum	AutoVacuum	Vacuum	AutoVacuum				
postgres	pg_default	pg_catalog	pg_statistic		3	0.02		346	910		172
postgres	pg_default	public	last_stat_tables_srv1		3	0.01		720	947	480	10
postgres	pg_default	public	last_stat_statements_srv1		2			76	152	47	6
postgres	pg_default	public	last_stat_indexes_srv1		2	0.01		987	866	658	11
postgres	pg_default	public	last_stat_user_functions_srv1		1			59	39	38	24

Рисунок 4.31 – Пример отчета Top tables by vacuum operations

Таблица 4.30 – Описание параметров отчета Top tables by vacuum operations

Параметр		Описание
Database		имя БД таблицы
Tablespace		имя табличного пространства, в котором находится таблица
Schema		имя схемы таблицы
Table		имя таблицы
Vacuum count		
	Vacuum	количество раз очистки таблицы выполнена вручную (VACUUM FULL не учитывается)
	Autovacuum	количество раз очистки таблицы выполнена фоновым процессом автоочистки

№ изменения: _____	Подпись отв. лица: _____	Дата внесения изм: _____
--------------------	--------------------------	--------------------------

Параметр		Описание
Vacuum time (s)		
	Vacuum	общее время, затраченное на ручную очистку этой таблицы. Включает время ожидания, вызванное задержками на основе стоимости
	Autovacuum	общее время, затраченное демоном автоочистки на очистку этой таблицы. Включает время ожидания, вызванное задержками на основе стоимости
Ins		количество вставленных строк (поле n_tup_ins)
Upd		количество обновленных строк (включая HOT) (поле n_tup_upd)
Del		количество удаленных строк (поле n_tup_del)
Upd(HOT)		количество обновленных строк HOT (поле n_tup_hot_upd)

4.5.2. Top tables by analyze operations (Топ таблиц по операциям анализа)

Таблица 4.32 содержит топ таблицы, отсортированные по количеству запусков анализа (ручных и автоматических).

DB	Tablespace	Schema	Table	Analyze count		Analyze time (s)		Ins	Upd	Del	Upd(HOT)
				Analyze	AutoAnalyze	Analyze	AutoAnalyze				
postgres	pg_default	public	last_stat_statements_srv1	3	3	0.01	0.03	76	152	47	6
postgres	pg_default	public	last_stat_tables_srv1	3	3	0.01	0.02	720	947	480	10
postgres	pg_default	public	last_stat_indexes_srv1	3	3	0.01	0.01	987	866	658	11
postgres	pg_default	public	last_stat_user_functions_srv1	3	2			59	39	38	24
postgres	pg_default	public	sample_stat_tables		3		0.02	467			

Рисунок 4.32 – Пример отчета Top tables by analyze operations

Таблица 4.31 – Описание параметров отчета Top tables by analyze operations

Параметр		Описание
Database		имя БД таблицы
Tablespace		имя табличного пространства, в котором находится таблица
Schema		имя схемы таблицы
Table		имя таблицы
Analyze count		
	Vacuum	количество раз сбора статистики таблицы выполнен вручную
	Autovacuum	количество раз сбора статистики выполнен фоновым процессом автоочистки
Autoanalyze time (s)		
	Vacuum	общее время, затраченное на ручной анализ этой таблицы. Включает время ожидания, вызванное задержками на основе стоимости
	Autovacuum	общее время, затраченное демоном автоочистки на анализ этой таблицы. Включает время ожидания, вызванное задержками на основе стоимости
Ins		количество вставленных строк (поле n_tup_ins)
Upd		количество обновленных строк (включая HOT) (поле n_tup_upd)
Del		количество удаленных строк (поле n_tup_del)
Upd(HOT)		количество строк, обновленных по HOT (поле n_tup_hot_upd)

№ изменения: _____	Подпись отв. лица: _____	Дата внесения изм: _____
--------------------	--------------------------	--------------------------

4.5.3. Top tables by vacuum time spent

Данная статистика отображается в случае, если pgpro_stats может предоставлять расширенную статистику очистки. Таблица 4.32 содержит информацию о таблицах, на очистку которых было потрачено больше всего времени. Эта информация основана на представлении pgpro_stats_vacuum_tables.

DB	Tablespace	Schema	Table	Vacuum time (s)		Vacuum count		Ins	Upd	Del	Upd(HOT)
				Vacuum	AutoVacuum	Vacuum	AutoVacuum				
postgres	pg_default	pg_catalog	pg_statistic		0.02			3 346 910			172
postgres	pg_default	public	last_stat_tables_srv1		0.01			3 720 947 480			10
postgres	pg_default	public	last_stat_indexes_srv1		0.01			2 987 866 658			11
postgres	pg_default	public	last_stat_statements_srv1					2 76 152 47			6
postgres	pg_default	public	last_stat_user_functions_srv1					1 59 39 38			24

Рисунок 4.33 – Пример отчета Top tables by vacuum time spent

Таблица 4.32 – Описание параметров отчета Top tables by vacuum time spent

Параметр		Описание
Database		имя БД таблицы
Tablespace		имя табличного пространства, в котором находится таблица
Schema		имя схемы таблицы
Table		имя таблицы
Analyze count		
	Vacuum	количество раз сбора статистики таблицы выполнен вручную
	Autovacuum	количество раз сбора статистики выполнен фоновым процессом автоочистки
Autoanalyze time (s)		
	Vacuum	общее время, затраченное на ручной анализ этой таблицы. Включает время ожидания, вызванное задержками на основе стоимости
	Autovacuum	общее время, затраченное демоном автоочистки на анализ этой таблицы. Включает время ожидания, вызванное задержками на основе стоимости
Ins		количество вставленных строк (поле n_tup_ins)
Upd		количество обновленных строк (включая HOT) (поле n_tup_upd)
Del		количество удаленных строк (поле n_tup_del)
Upd(HOT)		количество строк, обновленных по HOT (поле n_tup_hot_upd)

4.5.4. Top tables by analyze time spent (Топ таблиц, на анализ которых затрачено больше всего времени)

В таблице 4.33 представлена информация о таблицах, на ручной и автоматический анализ которых было затрачено больше всего времени.

DB	Tablespace	Schema	Table	Analyze time (s)		Analyze count		Ins	Upd	Del	Upd(HOT)
				Analyze	AutoAnalyze	Analyze	AutoAnalyze				
postgres	pg_default	public	last_stat_tables_srv1	0.01	0.02	3		3 720	947	480	10
postgres	pg_default	public	last_stat_statements_srv1	0.01	0.03	3		3 76	152	47	6
postgres	pg_default	public	last_stat_indexes_srv1	0.01	0.01	3		3 987	866	658	11
postgres	pg_default	public	sample_stat_tables		0.02			3 467			
postgres	pg_default	public	last_stat_io		0.01			1 39		26	
postgres	pg_default	public	last_stat_user_functions_srv1			3		2 59	39	38	24

Рисунок 4.34 – Пример отчета Top tables by analyze time spent

Таблица 4.33 – Описание параметров отчета Top tables by analyze time spent

Параметр		Описание
Database		имя БД таблицы
Tablespace		имя табличного пространства, в котором находится таблица
Schema		имя схемы таблицы
Table		имя таблицы
Analyze count		
	Vacuum	количество раз сбора статистики таблицы выполнен вручную
	Autovacuum	количество раз сбора статистики выполнен фоновым процессом автоочистки
Autoanalyze time (s)		
	Vacuum	общее время, затраченное на ручной анализ этой таблицы. Включает время ожидания, вызванное задержками на основе стоимости
	Autovacuum	общее время, затраченное демоном автоочистки на анализ этой таблицы. Включает время ожидания, вызванное задержками на основе стоимости
Ins		количество вставленных строк (поле n_tup_ins)
Upd		количество обновленных строк (включая HOT) (поле n_tup_upd)
Del		количество удаленных строк (поле n_tup_del)
Upd(HOT)		количество строк, обновленных по HOT (поле n_tup_hot_upd)

4.5.5. Top indexes by estimated vacuum load (Топ индексов, провоцирующие наибольшую нагрузку при очистке)

В таблице 4.35 представлена оценка неявной вакуумной нагрузки, вызванной индексами таблицы. Здесь представлены топ индексы, отсортированные по количеству вакуумов, выполненные для базовой таблицы, умноженные на размер индекса.

DB	Tablespace	Schema	Table	Index	Vacuum bytes	Vacuum count		IX size	Relsize
						Vacuum	AutoVacuum		
postgres	pg_default	pg_catalog	pg_statistic	pg_statistic_relid_att_inh_index	[168 kB]			3 [56 kB]	[384 kB]
postgres	pg_default	public	last_stat_tables_srv1	pk_last_stat_tables_srv1	[144 kB]			3 [48 kB]	[283 kB]
postgres	pg_default	public	last_stat_indexes_srv1	pk_last_stat_indexes_srv1	[136 kB]			2 [68 kB]	[308 kB]
postgres	pg_default	public	last_stat_statements_srv1	pk_last_stat_statements_srv1	[32 kB]			2 [16 kB]	[64 kB]

Рисунок 4.35 – Пример отчета Top indexes by estimated vacuum load

Таблица 4.34 – Описание параметров отчета Top indexes by estimated vacuum load

Параметр	Описание
Database	имя БД индекса
Tablespace	имя табличного пространства, в котором расположен индекс
Schema	имя схемы индекса
Table	имя таблицы
Index	имя индекса
~Vacuum bytes	оценка вакуумной нагрузки, рассчитываемая как (vacuum_count + autovacuum_count) * index_size
Vacuum cnt	количество раз, когда эта таблица была обработана вручную (не считая VACUUM FULL) (поле vacuum_count)
Autovacuum cnt	количество раз, когда эта таблица была обработана фоновым процессом autovacuum (поле autovacuum_count)
IX size	средний размер индекса в течение интервала отчета
Relsize	средний размер отношения в течение интервала отчета

4.6. Cluster settings during the report interval (Настройка кластера во время отчетного интервала)

Данный раздел отчета содержит параметры СУБД «Jatoba», а также значения встроенных системных функций version(), pg_postmaster_start_time(), pg_conf_load_time() и поле system_identifier функции pg_control_system() в течение интервала отчета.

Setting	Defined settings	Unit	Source	Notes
client_encoding	reset_val	UTF8		
commit_timestamp_buffers		32 8kB		
config_file	/var/lib/jatoba/18/data/postgresql.conf			
data_checksums		on		
data_directory	/var/lib/jatoba/18/data			
default_text_search_config	pg_catalog.english		/var/lib/jatoba/18/data/postgresql.conf:808	
hba_file	/var/lib/jatoba/18/data/pg_hba.conf			
huge_pages_status		off		
ident_file	/var/lib/jatoba/18/data/pg_ident.conf			
io_max_concurrency		64		
jatoba_version	Jatoba 18.0.1-61911 (build-id: 39a84569; build-os: UBUNTU22.04)			

Рисунок 4.36 – Пример отчета Cluster settings during the report interval

Таблица 4.35 – Описание параметров отчета Cluster settings during the report interval

Параметр	Описание
Setting	имя параметра
reset_val	поле reset_val представления pg_settings. Жирный шрифт используется для отображения настроек, измененных в течение интервала отчета

№ изменения: _____	Подпись отв. лица: _____	Дата внесения изм: _____
--------------------	--------------------------	--------------------------

Параметр	Описание
Unit	единица измерения параметра
Source	конфигурационный файл, в котором была определена данная настройка, номер строки после точки с запятой
Notes	поле, содержащее временную метку образца, когда это значение наблюдалось впервые в течение интервала отчета

4.7. Отчеты по компоненту «ja_Hipe_Cluster» (Citrus)

Данный раздел отчетов содержит параметры высокопроизводительного кластера, созданного компонентом «ja_Hipe_Cluster» (Citrus).

4.7.1. Nodes

Отчет содержит информацию об узлах кластера. В нем используются citrus_stat_activity, pg_dist_node и функция run_command_on_all_nodes.

Таблица 4.36 - Описание параметров отчета Nodes

Название параметра	Название параметра в pg_profile	Тип данных параметра	Описание параметра
Nodeid	Node id	Integer	Id узла
Groupid	Group id	Integer	Id группы
Nodename	Node name	Text	Наименование узла
nodeport	Node port	Integer	Порт
noderole	Node role	Noderole	Является ли узел первичным или вторичным
nodecluster	Node cluster	Name	Имя кластера
available	Available	integer	Доступность узла
count_conn	Count connection	integer	Количество соединений

4.7.2. Connectivity between all nodes

Отчет содержит информацию о связь между всеми узлами кластера. В нем используется функция citrus_check_cluster_node_health.

Таблица 4.37 – Описание параметров отчета Connectivity between all nodes

Название параметра	Название параметра в pg_profile	Тип данных параметра	Описание параметра
From_nodename	From node name	Text	Имя исходного узла
From_nodeport	From node port	integer	Порт на исходном узле
To_nodename	To node name	text	Имя узла назначения
To_nodeport	To node port	Integer	Порт на узле назначения
Available	Available	integer	Доступность узла

№ изменения: _____	Подпись отв. лица: _____	Дата внесения изм: _____
--------------------	--------------------------	--------------------------

4.7.3. Tables (cit_{us}_tables)

Отчет содержит информацию о таблицах, распределенных и справочных.

Таблица 4.38 – Описание параметров отчета Tables

Название параметра	Название параметра в pg_profile	Тип данных параметра	Описание параметра
table_name	Table	Text	Наименование таблицы
cit _{us} _table_type	Type	Text	Тип таблицы (distributed - распределенная, reference - справочная)
distribution_column	Distribution column	Text	Наименование столбца распределения
colocation_id	Colocation id	Integer	Идентификатор группы совместного размещения
table_size	Table size	Text	Размер таблицы
shard_count	Shard count	Bigint	Число шардов
access_method	Access method	Text	Метод доступа (heap, columnar)
	Growth size	Text	Размер таблицы (разница между текущим и предыдущим снимками)
	Growth shard count	Bigint	Число шардов (разница между текущим и предыдущим снимками)

4.7.4. Shards

В отчет входит информация о том, к какой распределенной таблице принадлежит шард, где находится каждый шард (нода/узел и порт), к какому типу таблицы он принадлежит, его размер и статистика о столбце распределения для этого шарда. В случае хэш-распределенных таблиц это диапазоны хэш-токенов, назначенных этому шарду.

Представление cit_{us}_shards помогает изучать сегменты, позволяя, среди прочего, находить дисбаланс по размеру между узлами.

Таблица 4.39 – Описание параметров отчета Shards

Название параметра	Название параметра в pg_profile	Тип данных параметра	Описание параметра
cit _{us} _shards.table_name	Table	Regclass	Наименование таблицы и схемы
cit _{us} _shards.shardid	Shard id	Bigint	Id шарда
cit _{us} _shards.shard_name	Shard name	Text	Наименование шарда
cit _{us} _shards.cit _{us} _table_type	Table type	Text	Тип таблицы

№ изменения: _____	Подпись отв. лица: _____	Дата внесения изм: _____
--------------------	--------------------------	--------------------------

Название параметра	Название параметра в pg_profile	Тип данных параметра	Описание параметра
citus_shards.colocation_id	Colocation id	Integer	Идентификатор группы совместного размещения
citus_shards.nodename	Node name	Text	Наименование ноды на которой расположен шард
citus_shards.nodeport	Node port	Integer	Порт
citus_shards.shard_size	Shard size	Bigint	Размер шарда
pg_dist_shard.shardstorage	Shard storage	character	Тип хранилища, используемый для этого шарда ('t', 'c', 'f').
pg_dist_shard.shardminvalue	Min value	Text	Для распределенных хэш-таблиц минимальное значение
pg_dist_shard.shardmaxvalue	Max value	text	Для распределенных хэш-таблиц максимальное значение
	Growth size	Bigint	Размер шарда (разница между текущим и предыдущим снимками)

4.7.5. Blocked queries (citus_lock_waits)

Отчет содержит информацию о запросах, заблокированных во всем кластере.

Таблица 4.40 – Описание параметров отчета Blocked queries

Название параметра	Название параметра в pg_profile	Тип данных параметра	Описание параметра
waiting_gpid	Waiting id	bigint	Id ожидающего запроса
blocking_gpid	Blocking id	bigint	Id блокирующего запроса
blocked_statement	Waiting query	Text	Заблокированный запрос
current_statement_in_blocking_process	Blocking query	Text	Блокирующий запрос
waiting_nodeid,	Waiting node id	Integer	Id ноды ожидающего запроса
blocking_nodeid	Blocking node id	integer	Id ноды блокирующего запроса

4.7.6. Query statistics (citus_stat_statements)

Отчет содержит статистику, как и для кого выполняются запросы.

№ изменения: _____	Подпись отв. лица: _____	Дата внесения изм: _____
--------------------	--------------------------	--------------------------

Таблица 4.41 – Описание параметров отчета Query statistics

Название параметра	Название параметра в pg_profile	Тип данных параметра	Описание параметра
queryid	Query id	bigint	Id запроса
rolname	User	oid	Наименование пользователя выполнившего запрос
datname	Database	oid	Наименование бд
query	Query	text	Строка запроса
partition_key	Partition key	text	Значения столбца распределения в запросах
calls	Calls	bigint	Количество запусков запросов

4.7.7. Rebalance progress (get_rebalance_progress)

Отчет содержит информацию о процессе ребалансировки.

Таблица 4.42 - Описание параметров отчета Rebalance progress

Название параметра	Название параметра в pg_profile	Тип данных параметра	Описание параметра
Table_name	Table	Regclass	Наименование таблицы
Shardid	Shard id	Bigint	Id шарда
Source_shard_size	Source shard size	Text	Размер исходного шарда
Target_shard_size	Target shard size	Text	Размер целевого шарда
Percent_completed_estimate	Percent	numeric	Процент выполнения

4.7.8. Configuration parameters

Отчет о параметрах конфигурации.

Таблица 4.43 - Описание параметров отчета Configuration parameters

Название параметра	Название параметра в pg_profile	Тип данных параметра	Описание параметра
Name	name	Text	Наименование параметра конфигурации
Setting	setting	Text	Текущее значение параметра конфигурации

4.7.9. Active tenants (citus_stat_tenants)

В случае если установленная версия Citus меньше чем 11.3, то отчет по таблице Active tenants не отображается.

Таблица 4.44 – Описание параметров отчета Active tenants

Название параметра	Название параметра в pg_profile	Тип данных параметра	Описание параметра
Nodename	Node name	Text	Наименование ноды
Tenant_attribute	Tenant attribute	Text	Значение в столбце распределения, идентифицирующее tenant
Read_count_in_this_period	Read count in this period	Int	Количество запросов на чтение (select) за период
Read_count_in_last_peroid	Read count in last peroid	Int	Количество запросов чтения за прошлый период
Query_count_in_this_period	Query count in this period	Int	Количество запросов на чтение/запись для tenant за период времени
Query_count_in_last_period	Query count in last period	int	Количество запросов на чтение/запись для tenant за прошлый период
Cpu_usage_in_this_period	Cpu usage in this period	Float	Секунды процессорного времени, затраченные этим tenant за период
Cpu_usage_in_last_period	Cpu usage in last period	float	Секунды процессорного времени, затраченные этим tenant за прошлый период

5. ДОПОЛНИТЕЛЬНАЯ ИНФОРМАЦИЯ

1) СУБД собирает статистику после завершения выполнения запроса. Если выполнение запроса длится в течение нескольких снапшотов, это повлияет на статистику только последнего снапшота (когда он был снят). И получить статистику по все еще выполняющимся операторам будет нельзя. Процессы обслуживания, такие как vacuum и checkpoint, обновляют статистику только после завершения.

2) Сброс любой статистики СУБД может повлиять на корректность формирования последующих запросов (планировщик запросов опирается на данные статистики при выборе методов обхода таблиц, использования индексов и т.д.).

3) Эксклюзивные блокировки на таблицах конфликтуют с процедурой вычисления размера таблицы. Снапшот не будет собирать размеры таблиц с AccessExclusiveLock, принадлежащим какой-либо сессии. Сессия может получить AccessExclusiveLock на таблицу во время обработки выборки. Для того, чтобы обойти эту проблему, lock_timeout (настройка СУБД «Jatoba», отвечающая за максимальное разрешенное время блокировки таблицы/индекса, строки или иного объекта БД) установлен на 3 секунды, поэтому если функция take_sample() не сможет получить блокировку в течение 3 секунд, она завершится неудачей, и снапшот не будет снят.

6. УДАЛЕНИЕ КОМПОНЕНТА

6.1. Удаление компонента в ОС GNU/Linux

6.1.1. Удаление расширений компонента

Удаление из БД расширений компонента может быть проведено следующим способом:

```
postgres=# DROP EXTENSION pg_profile;  
postgres=# DROP EXTENSION pg_stat_statements;  
postgres=# DROP EXTENSION dblink CASCADE;
```

В случае, если расширения были установлены в отдельную служебную БД (см. п. 2.4.2), то выполняется подключение к ней и команды удаления расширений.

После удаления расширений из БД необходимо открыть конфигурационный файл `postgres.conf` и удалить/закомментировать параметры, настроенные согласно п. 2.3 данного руководства:

```
# shared_preload_libraries = 'pg_stat_statements'  
# track_activities = on  
# track_counts = on  
# track_io_timing = on  
# track_functions = all
```

После внесения изменений в конфигурационный файл `postgres.conf`, для применения настроек необходимо перезагрузить СУБД.

6.1.2. Удаление пакета компонента

Удаление пакета компонента для систем на основе пакетного менеджера АРТ (к таким системам относятся все ОС семейства Debian, использующие deb-пакеты) применяется следующая команда удаления:

```
# apt-get remove jatoba<ver>-pg-profile
```

Удаление пакета компонента из других типов ОС также осуществляется средствами пакетного менеджера ОС. Вместо команды `install` нужно использовать соответствующую данному пакетному менеджеру команду удаления (`remove`, `purge`, `erase` и т.п.).

Для получения детальной информации по пакетному менеджеру рекомендуется обратиться к документации по ОС.

№ изменения: _____	Подпись отв. лица: _____	Дата внесения изм: _____
--------------------	--------------------------	--------------------------

7. ОБНОВЛЕНИЕ КОМПОНЕНТА



В данном разделе описаны процедуры обновления компонента pg_Profile с версии 4.6 включительно до версии 4.10.

При обновлении компонента будет утеряна информация по созданным подключениям до серверов, снимки состояния БД, отчеты по данным статистики.

7.1. Обновление компонента в ОС GNU/Linux

Для обновления компонента pg_Profile с версии 4.6 включительно до версии 4.10 необходимо выполнить следующие шаги:


- 1) Удалить расширение компонента pg_Profile версии 4.6 согласно п. 6.1 данного руководства;
- 2) Обновить репозиторий с пакетами согласно документу «Руководство по установке» 643.72410666.00067-07 97 01 и установить компонент pg_Profile версии 4.10 согласно п. 2.2;
- 3) Указать в конфигурационном файле postgres.conf параметры согласно п. 2.3;
- 4) Установить расширение pg_profile в БД согласно п. 2.4.1. В случае использования служебной БД установка расширения выполняется согласно п. 2.4.2;
- 5) Обновить расширения dblink и pg_stat_statements при помощи следующих команд:

```
ALTER EXTENSION dblink UPDATE;  
ALTER EXTENSION pg_stat_statements UPDATE;
```

- 6) Выполнить настройку компонента согласно разделу 3 данного руководства.

ТЕРМИНЫ И ОПРЕДЕЛЕНИЕ

AccessExclusiveLock	вид блокировки таблицы, гарантирующий, что доступ на момент блокировки имеет только та транзакция, которая вызвала блокировку
baseline	именованная последовательность снапшотов, которая имеет отдельную от настроенной политику хранения. Можно задать как определенное время хранения в днях, так и бесконечное время хранения, оставив соответственный параметр пустым. Также можно создать последовательность снапшотов только для определенного периода времени, например, в пиковые часы загрузки
dblink	расширение, позволяющее выполнить запрос к удаленной БД. Распространяется в составе СУБД PostgreSQL/Jatoba
HOT (Heap-only Tuple) обновление	механизм оптимизации выполнения команд UPDATE по изменению строк таблиц в СУБД. Данный механизм добавляет новую версию строки рядом со старой версией строки на той же странице данных и устанавливает соответствующие флаги следования записей в порядке их обновления. При таком подходе не нужно обновлять все индексы таблицы и записывать в них местоположение новой строки на диске. Данный механизм работает только для полей таблиц, не входящих в индексы, и призван уменьшить нагрузку на файловую систему
lock_timeout	параметр, задающий максимально возможное время блокировки таблицы / индекса / строки

<code>pg_conf_load_time()</code>	функция СУБД, которая возвращает данные о времени (моменте) загрузки конфигурации СУБД
<code>pg_control_system()</code>	функция СУБД, которая возвращает информацию о текущем состоянии управляющего файла СУБД
<code>pg_stat_kcache</code>	расширение, собирающее статистику по операциям чтения и записи на уровне файловой системы. Распространяется отдельно от СУБД PostgreSQL/Jatoba
	 Работает только на Linux системах
<code>pg_stat_kcache.track_planning = on/off</code>	параметр расширения <code>pg_stat_kcache</code> , контролирующий сбор данных об операциях планирования SQL-запросов и времени планирования
<code>pg_postmaster_start_time()</code>	функция СУБД, которая возвращает данные о времени (моменте) запуска СУБД
<code>pg_stat_statements</code>	расширение, собирающее статистику выполнения SQL-запросов на сервере БД. Распространяется в составе СУБД PostgreSQL/Jatoba
<code>plpgsql</code>	расширение языка SQL (процедурный язык), используемое в СУБД PostgreSQL/Jatoba. Этот язык предназначен для написания хранимых процедур и функций
<code>postgresql.conf</code>	файл конфигурации СУБД
Statistic Collector (сборщик статистики)	встроенный в СУБД механизм, позволяющий собирать метрики активности сервера БД
<code>vacuum</code> (вакуум)	механизм СУБД, удаляющий старые версии строк таблиц в БД

version() и jatoba_version()

функции СУБД, которые возвращают информацию о версии СУБД

снэпшот

снимок состояния БД на конкретный момент времени

ПЕРЕЧЕНЬ СОКРАЩЕНИЙ

ID	–	Identifier
SQL	–	Structured Query Language
БД	–	База данных
ОС	–	Операционная система
СУБД	–	Система управления базами данных

Лист регистрации изменений

№ изменения: _____	Подпись отв. лица: _____	Дата внесения изм: _____
--------------------	--------------------------	--------------------------