
Ред Монитор
Версия 0.5
Руководство пользователя

Содержание

1	Общие сведения о программе	3
1.1	Назначение программы	3
1.2	Минимальный состав аппаратных средств	4
1.3	Минимальный состав программных средств	4
2	Установка и настройка Ред Монитор	5
2.1	Сервер мониторинга	5
2.1.1	Prometheus	5
	Основные настройки Prometheus	5
	Секция global	5
	Секция rule_files	6
	Секция scrape_configs	6
	Секция alerting	6
	Запуск Prometheus	6
2.1.2	Настройка уведомлений	7
	Секция global	7
	Секция route	8
	Секция receivers	9
	Секция templates	9
	Секция inhibit_rules	10
	Секция time_intervals	10
	Включение уведомлений	11
2.1.3	Grafana	11
2.1.4	Ред База Данных	11
2.2	Сервер СУБД	11
2.2.1	Ред База Данных	11
2.2.2	Экспортёр	12
	Установка экспортёра	12
	Настройка экспортёра	13
3	Просмотр метрик	17
3.1	Просмотр метрик с помощью Prometheus	17
3.2	Просмотр метрик с помощью Grafana	17
3.3	Обзорная страница	19
3.4	Информация о выбранном сервере	19
	3.4.1 Базы данных	21
	3.4.2 Информация о выбранной базе данных	21
	SQL запросы	22
	Соединения	24
	3.4.3 Процессы	25
	Обзор процесса	26
	3.4.4 SQL-запросы	27
Приложение А	Собираемые метрики	29
А.1	Метрики таблиц мониторинга	29
А.2	Метрики агрегатного аудита	34
А.3	Метрики утилиты rdb_lock_print	36
А.4	Метрики репликации	38
А.5	Метрики операционной системы	39
А.6	Другие метрики	42

Глава 1

Общие сведения о программе

1.1 Назначение программы

Экспортёр СУБД Ред База Данных используется для сбора показателей состояния (метрик) СУБД и отправки их в систему мониторинга Prometheus. Экспортёр выступает инструментом для взаимодействия Prometheus и СУБД Ред База Данных. Для визуализации собранных метрик используется Grafana. Экспортёр, Ред База Данных, Prometheus и Grafana вместе образуют продукт Ред Монитор.

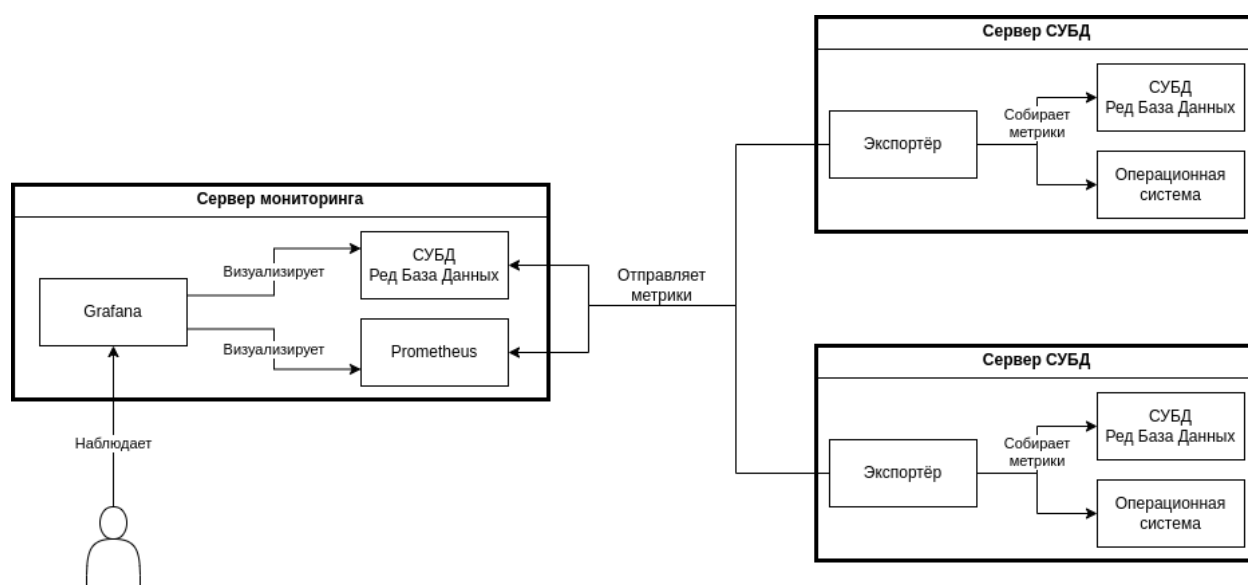


Рисунок 1.1 — Схема взаимодействия компонентов Ред Монитора

- Сервер мониторинга - компьютер, на котором работают программы Prometheus, СУБД Ред База Данных и Grafana.
- Сервер СУБД - компьютер, на котором работают программы СУБД Ред База Данных и Экспортёр.

Сервер мониторинга должен иметь возможность выполнения запроса к серверам СУБД. Сервер мониторинга выполняет функцию хранения и демонстрации данных, собранных с серверов СУБД. Prometheus отправляет запросы на экспортёры серверов СУБД. Экспортёры, получив запрос, собирают метрики и отправляют данные обратно в Prometheus. Некоторые данные (например, текст SQL-запроса) отправляются на хранение в базу данных.

Для отображения данных сервер мониторинга использует Grafana. Она получает данные из Prometheus и Ред Базы Данных.

1.2 Минимальный состав аппаратных средств

- Оперативная память от 16Гб;
- Процессор не менее 4х ядер;
- Запоминающее устройство объёмом не менее 64Гб.

1.3 Минимальный состав программных средств

Перед началом работы с Ред Монитор необходимо:

- Установить СУБД Ред База Данных версии не ниже 3.0. Узнать об этом подробнее можно в [Руководстве администратора](#);
- Установить и настроить Prometheus версии не ниже 2.37.5;
- Установить и настроить Grafana версии не ниже 9.3.2.

Глава 2

Установка и настройка Ред Монитор

2.1 Сервер мониторинга

2.1.1 Prometheus

Скачать Prometheus можно с официального сайта -- prometheus.io. Далее его необходимо установить.

Основные настройки Prometheus

Для настройки Prometheus используется файл `prometheus.yml`.

```
global:
  scrape_interval: 20s
  scrape_timeout: 15s
  evaluation_interval: 30s

scrape_configs:
  - job_name: "RedDatabase"
    static_configs:
      - targets: ["localhost:3051"]

rule_files:
  - rules.yml

# alerting:
#   alertmanagers:
#     - static_configs:
#       - targets: ["localhost:9093"]
```

Файл `prometheus.yml` разделён на четыре основные секции: `global`, `rule_files`, `scrape_configs`, `alerting`.

Секция `global`

Настройки, заданные в секции `global` по умолчанию распространяются на весь файл.

```
global:
  scrape_interval: 10s
  scrape_timeout: 10s
  evaluation_interval: 30s
```

Параметр `scrape_interval` определяет, как часто Prometheus должен собирать метрики.

Параметр `scrape_timeout` устанавливает время ожидания ответа на запрос Prometheus.

Параметр `evaluation_interval` устанавливает интервал, с которым будут проверяться правила предупреждений.

Секция `rule_files`

В секции `rule_files` указывается список файлов с правилами для предупреждений:

```
rule_files:
- rules.yml
- "side/*_rules.yml"
```

Секция `scrape_configs`

В секции `scrape_configs` задаются настройки для сбора метрик:

```
scrape_configs:
- job_name: "RedMonitor"
  scrape_interval: 15s
  scrape_timeout: 10s
  static_configs:
    - targets: ["192.168.0.100:8000", "192.168.0.100:8100", "192.168.0.150:8000"]
```

Параметр `job_name` задаёт уникальное имя для сборщика метрик. Можно указать несколько сборщиков.

Параметр `scrape_interval` определяет, как часто **Prometheus** должен собирать метрики. По умолчанию принимает значение, установленное в секции `global`.

Параметр `scrape_timeout` устанавливает время ожидания ответа на запрос **Prometheus**. По умолчанию принимает значение, установленное в секции `global`.

Параметр `targets` определяет сетевые узлы экспортеров, к которым будет обращаться **Prometheus**, чтобы извлечь метрики.

Секция `alerting`

Секция `alerting` определяет сущности **Alertmanager**, в которые **Prometheus** будет отправлять предупреждения.

```
alerting:
  alertmanagers:
    - static_configs:
      - targets: ["localhost:9093"]
```

Запуск Prometheus

Запуск **Prometheus** осуществляется следующей командой

```
./prometheus --config.file="prometheus.yml"
```

Более подробно о настройке **Prometheus** можно узнать на официальном сайте – prometheus.io.

2.1.2 Настройка уведомлений

Скачать Alertmanager можно с официального сайта -- prometheus.io. Далее его необходимо установить.

Чтобы получать уведомления, необходимо настроить Alertmanager. Для этого используется файл `alertmanager.yml`.

Пример конфигурации Alertmanager:

```
global:
  smtp_require_tls: true
  smtp_from: 'sender@example.com'
  smtp_smarthost: 'smtp.example.com:587'
  smtp_auth_username: 'user_name'
  smtp_auth_password: 'user_password'

route:
  group_by: ['alertname']
  group_wait: 30s
  group_interval: 5m
  repeat_interval: 1h
  receiver: 'receiver_name'

receivers:
  - name: 'receiver_name'
    email_configs:
      - to: 'receiver@example.com'

templates:
  - "/opt/RedMonitor/rdb_templates.lib"

inhibit_rules:
  - source_matchers: [severity="critical"]
    target_matchers: [severity="warning"]
```

Файл делится на следующие основные секции: `global`, `route`, `receivers`, `inhibit_rules`, `time_intervals`.

Секция `global`

В секции `global` указывается куда отправлять данные. Пример настройки секции для отправки уведомлений на почту:

```
global:
  smtp_smarthost: 'localhost:25'
  smtp_from: 'alertmanager@example.org'
  smtp_auth_username: 'alertmanager'
  smtp_auth_password: 'password'
  resolve_timeout: 1m
```

Параметр `smtp_smarthost` устанавливает адрес сервера почты.

Параметр `smtp_from` указывает имя отправителя.

Параметр `smtp_auth_username` определяет имя пользователя для аутентификации на сервере по-

чты.

Параметр `smtp_auth_password` определяет пароль пользователя для аутентификации на сервере почты.

Параметр `resolve_timeout` определяет время, в течении которого пришедшее уведомление будет считаться неразрешённым

Секция `route`

Секция `route` определяет маршруты уведомлений в виде структуры дерева. Маршрут - это список проверок, через которые проходит уведомление для нахождения получателя. Проверка происходит на каждом узле. Если параметр `continue` имеет значение `false`, то проверка прекращается после нахождения первого подходящего элемента дерева. Если `continue` имеет значение `true`, то уведомление продолжит поиск подходящих элементов среди соседних узлов. Если уведомление не соответствует ни одному дочернему узлу, то оно будет обрабатываться на основе параметров конфигурации текущего узла.

```
route:
group_by: ['alertname', 'cluster', 'service']
group_wait: 1m
group_interval: 5m
repeat_interval: 2h
receiver: A-team

routes:
- matchers:
  - service= "sells"
  receiver: B-team
  routes:
  - matchers:
    -severity="critical"
    receiver: C-team
- matchers:
  - service="docs"
  receiver: B-team
  routes:
  - matchers:
    -severity="critical"
    receiver: B-team
- matchers:
  - service="database"
  receiver: team-DB-pager
group_by: [alertname, cluster, database]
routes:
- matchers:
  - owner="team-X"
  receiver: team-X-pager
  continue: true
- matchers:
  - owner="team-Y"
  receiver: team-Y-pager
```

Параметр `group_by` определяет список меток, по которым происходит группировка входящих уведомлений.

Параметр `group_wait` устанавливает время ожидания перед отправкой новой группы уведомлений.

Параметр `group_interval` задаёт время между отправкой уведомлений по группам.

Параметр `repeat_interval` задаёт время между повторной отправкой уведомлений.

Параметр `receiver` определяет получателя.

Параметр `matchers` устанавливает регулярные выражения для проверки.

Параметр `routes` определяет новый узел дерева.

Секция `receivers`

В секции `receivers` указывается список получателей уведомлений.

```
receivers:
- name:
  email_configs:
    - to:

- name:
  email_configs:
    - to:
  pagerduty_configs:
    - service_key: <team-X-key>
```

Секция `templates`

В секции `templates` указываются пути до шаблонов, определяющих формат уведомлений, проходящих на почту.

```
receivers:
- name: 'email'
  email_configs:
- to: 'email'
  from: 'email'
  smarthost: 'smtp.gmail.com:587'
  auth_username: 'username'
  auth_password: 'password'
  html: '{{ range .Alerts }} {{ range .Labels.SortedPairs }} {{if eq (.Name "type")}}
{{ template "rdb_errors_mapping".Value | print}} {{ end }}{{ end }}{{ end }}'

templates:
- "/some/folder/template.lib"
```

Изменить оформление по умолчанию можно с помощью параметра `html`. Для фильтрации и обработки информации об уведомлениях используется язык шаблонов `G0`.

Можно создавать собственные шаблоны оформления. Для использования шаблона в секции `templates` нужно указать путь до него.

Секция `inhibit_rules`

Секция `inhibit_rules` устанавливает правила, по которым уведомления будут отключаться.

```
inhibit_rules:  
  - source_matchers: [severity="critical"]  
    target_matchers: [severity="warning"]
```

Параметры `source_matchers` и `target_matchers` это списки меток со значениями. Те предупреждения, значения меток которых совпадают с метками из параметра `source_matchers`, блокируют отправку уведомления для тех предупреждений, значения меток которых совпадают с метками из параметра `target_matchers`.

Секция `time_intervals`

В конфигурацию также может быть добавлена секция `time_intervals`. Она задает именованный интервал, на который можно ссылаться для отключения/активации определенных маршрутов в определенное время суток.

```
name: <string>  
time_intervals:  
  [ - <time_interval> ... ]
```

Параметр `time_interval` содержит фактическое определение интервала времени. Синтаксис поддерживает следующие поля:

```
- times:  
  [ - <time_range> ...]  
weekdays:  
  [ - <weekday_range> ...]  
days_of_month:  
  [ - <days_of_month_range> ...]  
months:  
  [ - <month_range> ...]  
years:  
  [ - <year_range> ...]  
location: <string>
```

Параметр `time_range` определяет диапазоны, включающее время начала и исключающее время окончания.

Параметр `weekday_range` определяет список дней недели, где неделя начинается в воскресенье и заканчивается в субботу. Дни следует указывать по имени (например, 'Sunday'). Для удобства диапазоны также принимаются в форме `<start_day>:<end_day>` и оба включают время окончания.

Параметр `days_of_month_range` определяет список дней в месяце. Дни начинаются с 1. Также принимаются отрицательные значения, которые отсчитываются с конца месяца, например, -1 в январе означает 31 января.

Параметр `month_range` определяет список календарных месяцев.

Параметр `year_range` определяет список лет.

Параметр `location` определяет строку, соответствующую местоположению в базе данных часовых поясов IANA. Местоположение указывает часовой пояс для временного интервала. Также можно использовать 'Local', чтобы использовать локальное время машины, на которой запущен `Alertmanager`, или 'UTC' для времени UTC.

Включение уведомлений

После настройки `alertmanager.yml` нужно указать в настройках Prometheus адрес, по которому доступен Alertmanager:

```
alerting:
  alertmanagers:
    - static_configs:
      - targets:
        - localhost:9093
```

Далее нужно добавить правила, по которым будут приходить уведомления. Можно импортировать готовый набор правил уведомлений, поставляемый вместе с Ред Монитор. Для этого нужно указать в настройках Prometheus путь к файлу с готовыми правилами:

```
rule_files:
  - rules.yml
```

Файл `rules.yml` необходимо предварительно скопировать из корневого каталога RedMonitor в корневой каталог Prometheus. Более подробно о настройке Alertmanager можно узнать в [официальной документации](#).

2.1.3 Grafana

Скачать Grafana можно с официального сайта – grafana.com. Далее её необходимо установить и запустить.

Для настройки откройте в браузере страницу запущенной Grafana (по умолчанию <http://localhost:3000/>). По умолчанию для входа используется логин `admin` и пароль `admin`. Чтобы установить источник данных, перейдите в настройки. Во вкладке `Data sources` нажмите на кнопку `Add data source`. В открывшемся окне из списка источников выберите Prometheus. Укажите URL-адрес для доступа к Prometheus (по умолчанию `http://localhost:9090/`) и нажмите кнопку `Save & test`.

2.1.4 Ред База Данных

Базу данных для хранения пользовательских запросов нужно создать самостоятельно на сервере мониторинга, то есть там, где запущены Prometheus, Grafana и СУБД. Для этого нужно выполнить скрипт `create_tables.sql`, указав в нём пользователя, от имени которого будет создана база данных.

```
<каталог Ред Базы Данных>/bin/isql -i ./dists/dictserver/create_tables.sql
```

2.2 Сервер СУБД

2.2.1 Ред База Данных

На сервере СУБД для сбора метрик пользовательских запросов необходимо:

1. В `firebird.conf` настроить параметры `TracePlugin` и `AuditTraceConfigFiles`:

```
TracePlugin = aggtrace
AuditTraceConfigFiles = fbtrace.conf
```

2. Включить агрегатный аудит, настроив `fbtrace.conf`:

```
database
{
    enabled = true
    format = 3
    reset_counters = true
}
```

2.2.2 Экспортёр

Установка экспортёра

Экспортёр необходимо скачать по [ссылке](#). После скачивания распакуйте архив, выполнив команду:

```
tar -xzf <архив>
```

Перейдите в распакованную папку:

```
cd <папка>
```

Установите Ред Монитор, выполнив `install.sh` с правами администратора:

```
sudo ./install.sh {<install> | <update>} {<grafana_plugins> | <exporter>}
```

Первый аргумент указывает, какое действие нужно выполнить: установку или обновление. Второй аргумент определяет, что нужно установить: плагины для **Grafana** или экспортёр.

При установке плагинов для **Grafana** нужно внести `redsoft-redmonitor-app` и `redsoft-rdbconnector-datasource` в список доверенных, разрешив редактировать файл `grafana.ini`:

```
Введите путь до папки плагинов: /opt/Grafana/data/plugins
Копирование файлов...
Успешно!
```

```
Редактировать файл /etc/grafana/grafana.ini для включения плагинов в
список доверенных?(д/н) да
Успешно!
```

```
Плагины Grafana установлены!
```

По-умолчанию проверяется наличие файла `/etc/grafana/grafana.ini`, если он существует, то нужно разрешить его редактирование. Если файла не существует, то нужно будет ввести путь до другого файла с плагинами **Grafana**.

Ред Монитор будет установлен в `/opt/RedMonitor`.

Для удаления Ред Монитора выполните команду:

```
sudo ./uninstall.sh [<опции>]
```

```
<опции>:=
exporter
```

(продолжение на следующей странице)

(продолжение с предыдущей страницы)

```
| grafana_plugins  
| full
```

Настройка экспортёра

Экспортёр должен находиться на том же устройстве, где и Ред База Данных. Для настройки параметров экспортёра используется файл `exporter_conf.json`, расположенный в `/opt/RedMonitor/exporter`.

Структура файла `exporter_conf.json`:

```
{  
  "exporter": {  
    "host": "<хост экспортёра>",  
    "port": <порт экспортёра>,  
    "scrape_rdb_lock_print": {  
      "enabled": true  
    },  
    "scrape_processes": {  
      "enabled": true  
    },  
    "scrape_log": {  
      "enabled": true,  
      "start_from_end": true  
    },  
    "scrape_atrace": {  
      "enabled": true  
    },  
    "scrape_replication": {  
      "enabled": true  
    }  
  },  
  "dictserver": {  
    "host": "<хост базы данных>",  
    "port": <порт базы данных>,  
    "database": "/db/statements.fdb"  
  },  
  "RedDatabase": {  
    "host": "<хост СУБД>",  
    "port": <порт СУБД>,  
    "login": "<логин>",  
    "password": "<пароль>",  
    "folder_path": "<путь к папке установки RedDatabase>",  
    "Auth_plugins": ["<плагин>", ..., "<плагин>"]  
  },  
  "databases": {  
    "<псевдоним>": {  
      "alias": "<алиас>",  
      "path": "<путь к базе данных>"  
    }  
  }  
}
```

Для каждого параметра должно быть указано значение. Значения строковых параметров должны заключаться в двойные кавычки. Значения для целочисленных параметров нужно указывать без кавычек.

Параметры конфигурации:

В секции **exporter** обязательно должны быть указаны все параметры, иначе экспортёр не запустится. Параметры секции:

- **host** - определяет ip-адрес устройства, на котором работает экспортёр; параметр имеет строковый тип;
- **port** - порт, на котором работает экспортёр; параметр имеет целочисленный тип;
- **rdb_lock_print** - определяет, собирать ли данные файла блокировок. Аналогично `rdb_lock_print -n -l -o -c`. По умолчанию установлено значение **true**;
- **scrape_processes** - определяет, собирать ли информацию о процессах операционной системы. По умолчанию установлено значение **true**;
- **scrape_log** - определяет, собирать ли данные из лог-файла. Опция **start_from_end** определяет порядок чтения файла. Если установлено значение **true**, то чтение будет начато с конца лог-файла, то есть будут прочитаны только новые записи. Если установлено значение **false**, то с начала лог-файла будут прочитаны все записи;
- **scrape_atrace** - определяет, собирать ли данные с помощью агрегирующего трейса. По умолчанию установлено значение **false**;
- **scrape_replication** - определяет, собирать ли информацию о репликации. По умолчанию установлено значение **false**. В зависимости от вида репликации на сервере СУБД собираются разные метрики.

Секция **dictserver**:

- **host** - определяет ip-адрес устройства, на котором работает база данных для пользовательских запросов; параметр имеет строковый тип;
- **port** - порт, на котором работает база данных для пользовательских запросов; параметр имеет целочисленный тип;
- **database** - псевдоним базы данных пользовательских запросов.

Секция **RedDatabase**:

- **host** - определяет ip-адрес устройства, на котором работает Ред База Данных; параметр имеет строковый тип;
- **port** - порт, на котором работает Ред База Данных; параметр имеет целочисленный тип;
- **login** - имя пользователя, от которого экспортёр будет подключаться к СУБД. Желательно использовать пользователя с административными привилегиями для мониторинга всех подключений; параметр имеет строковый тип;
- **password** - пароль пользователя; параметр имеет строковый тип;
- **folder_path** - путь к установочной папке **RedDatabase**;
- **Auth_plugins** - определяет список плагинов, которые экспортёр использует для аутентификации, в качестве разделителя используется запятая, каждый плагин должен заключаться в двойные кавычки; список заключается в квадратные скобки;

Секция **databases** определяет список баз данных для мониторинга, указывающийся следующим образом:

```
"databases": {  
  "<псевдоним>": {  
    "alias": "<алиас>",  
    "path": "<путь к базе данных>"  
  },  
}
```

(продолжение на следующей странице)

(продолжение с предыдущей страницы)

```
...  
}
```

Где:

- **псевдоним**- уникальный псевдоним базы данных;
- **path** - путь до базы данных;
- **alias** - псевдоним базы данных, который используется СУБД. Если псевдонима у базы данных нет, то нужно указать значение **null**.

Пример настроенного файла `exporter_conf.json`:

```
{  
  "exporter": {  
    "host": "127.0.0.1",  
    "port": 3051  
    "scrape_rdb_lock_print": {  
      "enabled": true  
    },  
    "scrape_processes": {  
      "enabled": true  
    },  
    "scrape_log": {  
      "enabled": true,  
      "start_from_end": true  
    },  
    "scrape_atrace": {  
      "enabled": false  
    },  
    "scrape_replication": {  
      "enabled": false  
    }  
  },  
  "dictserver": {  
    "host": "172.17.0.2",  
    "port": 3050,  
    "database": "/db/statements.fdb"  
  },  
  "RedDatabase": {  
    "host": "127.0.0.1",  
    "port": 3050,  
    "login": "SYSDBA",  
    "password": "masterkey",  
    "folder_path": "/opt/RedDatabase",  
    "Auth_plugins": ["Srp"]  
  },  
  "databases": {  
    "employee_db": {  
      "alias": "employee",  
      "path": "/opt/RedDatabase/examples/empbuild/employee.fdb"  
    }  
  }  
}
```

Перед запуском экспортёра убедитесь, что сервер СУБД запущен.

Запуск и остановка утилиты осуществляется с помощью `systemd`:

```
systemctl start redmonitor  
systemctl stop redmonitor
```

При запуске через `systemd` используется файл конфигурации по умолчанию, расположенный по следующему пути: `/opt/RedMonitor/exporter/exporter_conf.json`. Чтобы использовать другой файл конфигурации (созданный самостоятельно), нужно изменить путь до файла конфигурации экспортёра, отредактировав файл `redmonitor.sh`, расположенный в `/usr/local/bin`. В этом файле нужно изменить значение переменной `$configure` на другой путь до файла конфигурации.

Также можно запустить бинарный файл. Для этого нужно выполнить в терминале:

```
./exporter <полный путь до файла конфигурации>
```

Глава 3

Просмотр метрик

Собираемые метрики описаны в приложении *Собираемые метрики*.

3.1 Просмотр метрик с помощью Prometheus

Для просмотра метрик с помощью Prometheus нужно открыть в браузере страницу запущенного Prometheus (по умолчанию <http://localhost:9090/>). Можете запустить Prometheus с другим портом:

```
sudo ./prometheus --web.listen-address=":8080"
```

Чтобы узнать значение конкретной метрики, нужно ввести PromQL запрос.

Примеры PromQL запросов:

1. Вывод объёма базы данных:

```
rdb_database_size{database="employee_db", instance="127.0.0.1:3051"}
```

2. Вывод скорости записи на диск в секунду:

```
irate(rdb_disks_io_bytes{instance="127.0.0.1:3051", database="employee_db",  
disk="sda1"}[1m])
```

3. Вывод разницы между Next transaction и Oldest interesting transaction:

```
rdb_transactions_markers{instance="127.0.0.1:3051", database=database=  
"employee_db", marker="NT"} - ignoring(marker)  
rdb_transactions_markers{instance="127.0.0.1:3051", database="employee_db",  
marker="OIT"}
```

Средства визуализации Prometheus довольно скудны, поэтому для просмотра метрик лучше использовать Grafana.

3.2 Просмотр метрик с помощью Grafana

Для просмотра метрик с помощью Grafana откройте в браузере страницу запущенной Grafana (по умолчанию <http://localhost:3000/>).

Для отображения собранной информации необходимо включить плагин RedMonitor:

1. Выберите Home → Administration → Plugins;
2. В открывшейся вкладке перейдите в Type → Applications → RedMonitor;
3. Нажмите на кнопку Enable.

После этого в меню Apps будет добавлен плагин RedMonitor:

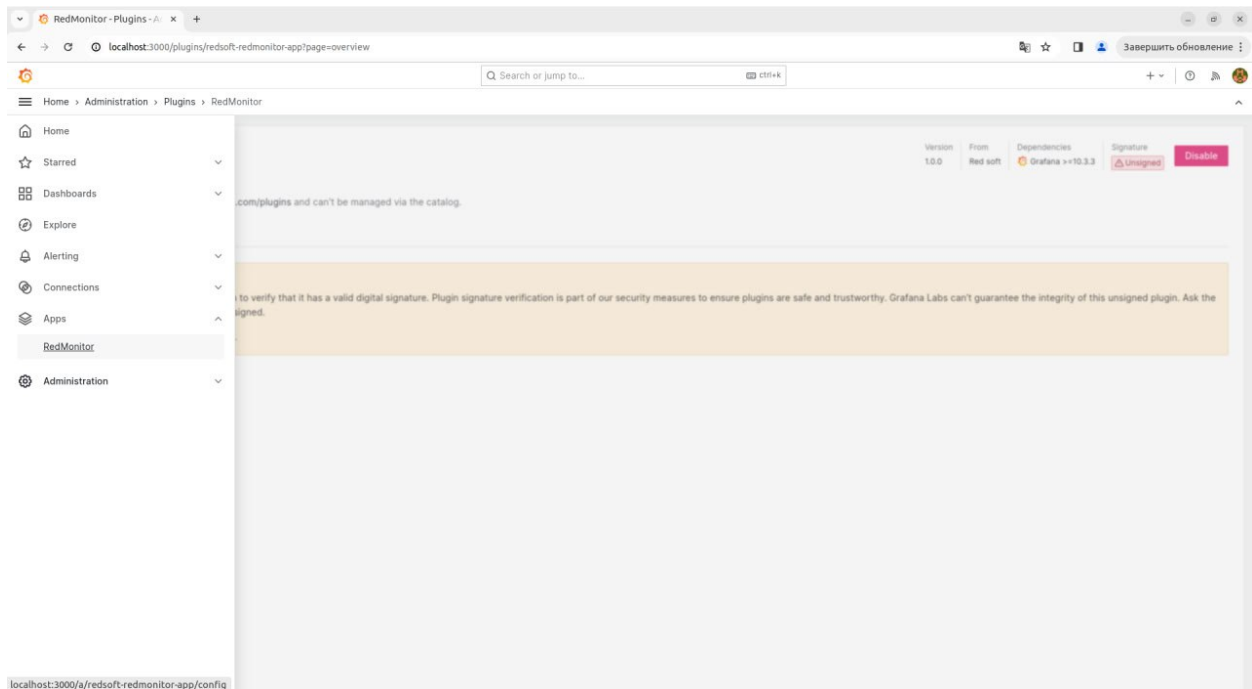


Рисунок 3.1 — Плагин Ред Монитор

Для настройки плагина нужно перейти на его страницу и выбрать источники данных для Prometheus и Ред Базы Данных.

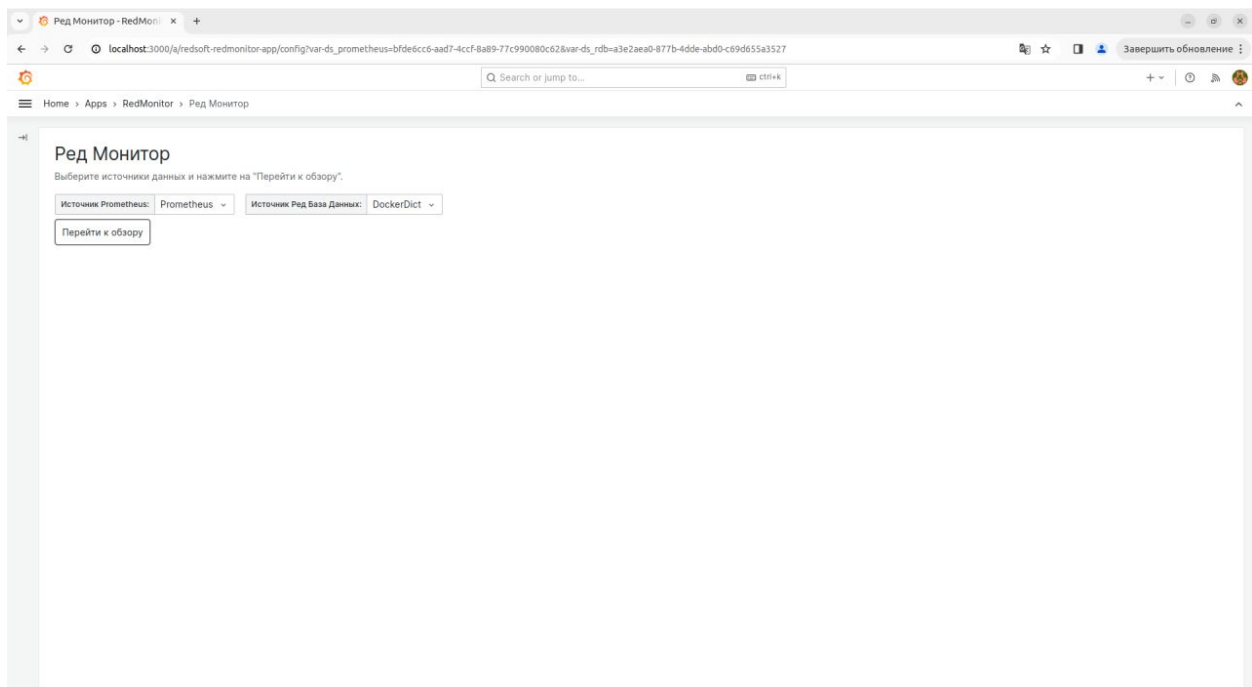


Рисунок 3.2 — Настройка плагина Ред Монитор

После нажатия на кнопку **Перейти к обзору** откроется **Обзорная страница**.

3.3 Обзорная страница

На этой странице отображается краткая информация о каждом сервере. Для получения подробной информации о конкретном сервере нужно нажать на его адрес.

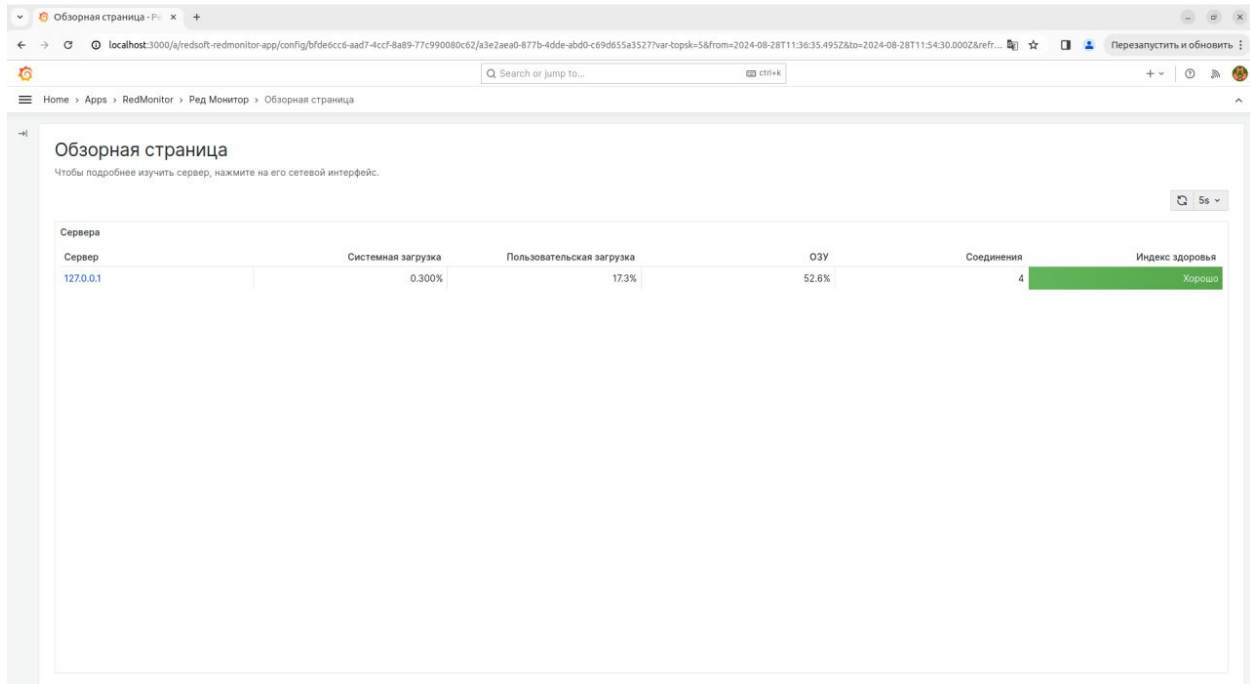


Рисунок 3.3 — Обзорная страница

3.4 Информация о выбранном сервере

На этой странице находится подробная информация о выбранном сервере.

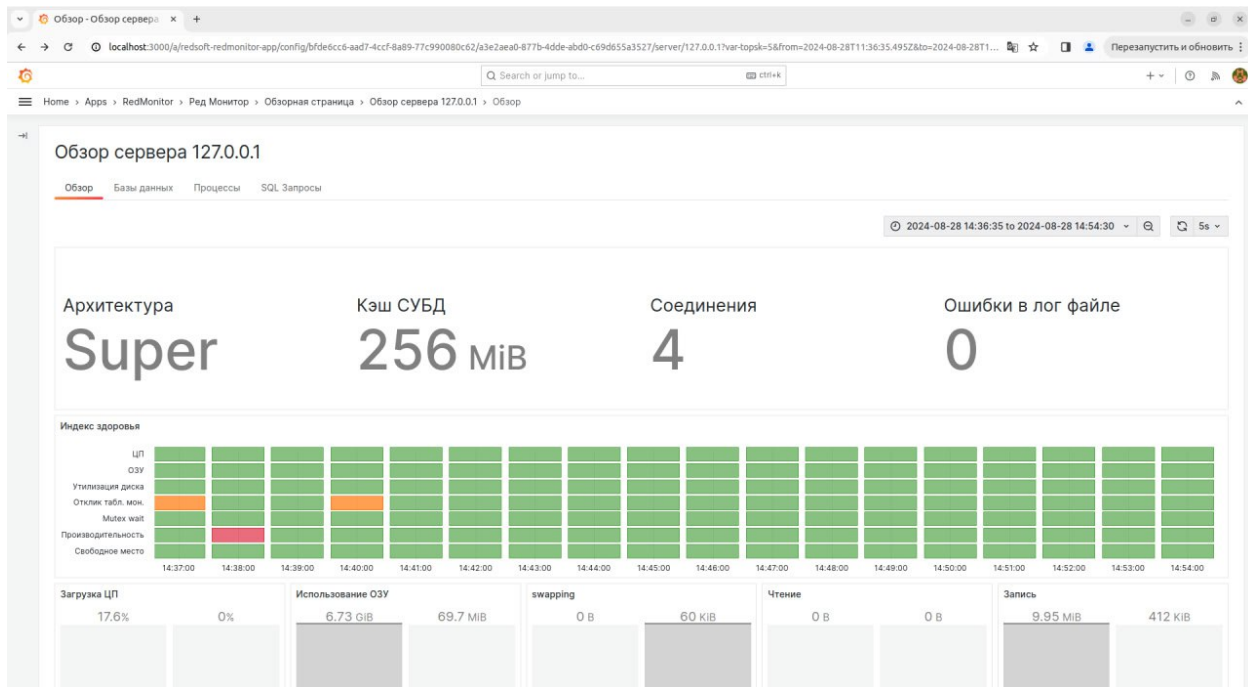


Рисунок 3.4 — Обзор выбранного сервера

- Кэш – размер, выделенный в оперативной памяти для кэша;
- Архитектура – архитектура СУБД;
- Соединения - количество соединений, установленных в данный момент;
- Ошибки в лог файле - количество ошибок, записанных в лог файл.

График **Количество соединений** показывает количество активных соединений в зависимости от времени.

График **Ошибки в лог файле** по группам отображает количество ошибок разной критичности в зависимости от времени.

График **Загрузка ЦПУ** показывает пользовательскую и системную нагрузку на процессор в зависимости от времени.

График **Использование ОЗУ** показывает использование оперативной памяти в зависимости от времени.

График **Количество прерываний** показывает сколько прерываний процессора было совершено за последний интервал сбора метрик.

График **Чтение/запись** показывает количество выполненных операций чтения с дисков и записи на них в зависимости от времени.

График **Переключения контекста** показывает сколько переключений контекста процессора было совершено за последний интервал сбора метрик.

График **SWAP** показывает использование SWAP в зависимости от времени.

График **Тактовая частота процессора** показывает текущую, максимальную и минимальную частоту процессора в зависимости от времени.

График **Swapping** показывает объём информации, который был загружен в SWAP и выгружен из него за время, прошедшее с предыдущего измерения.

3.4.1 Базы данных

На этой странице отображается краткая информация о каждой базе данных. Для получения подробной информации о конкретной базе нужно нажать на её имя.

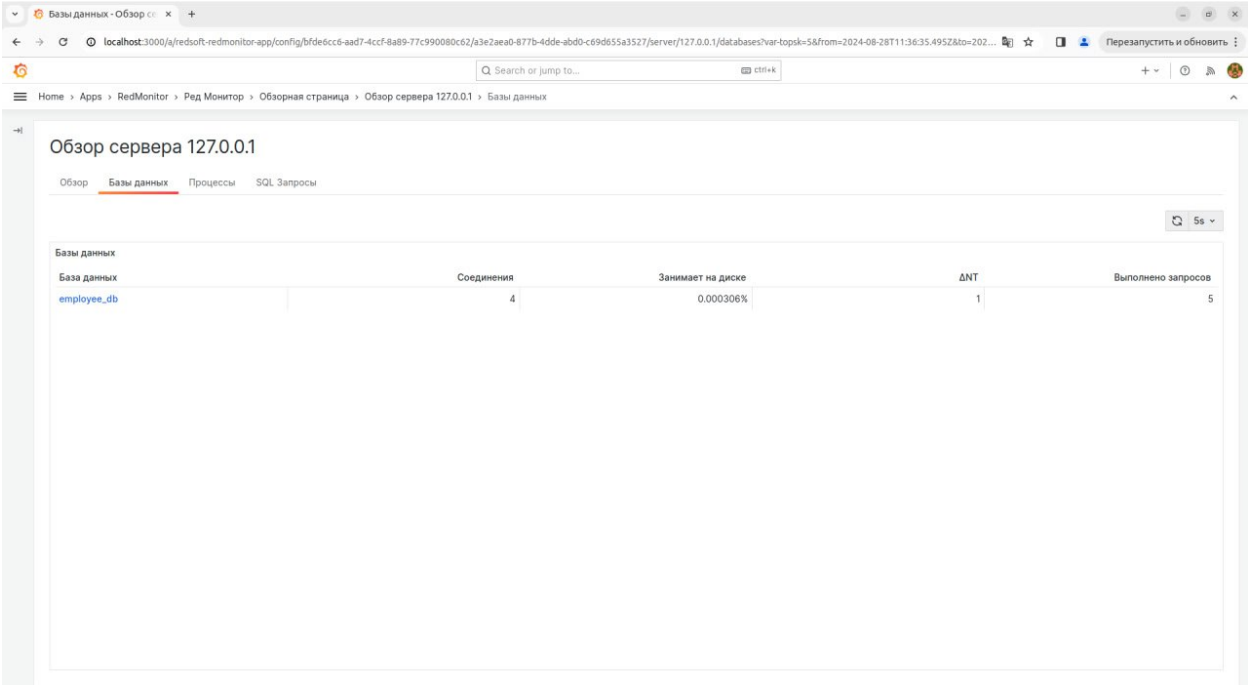


Рисунок 3.5 — Обзор баз данных

3.4.2 Информация о выбранной базе данных

На этой странице находится подробная информация о базе данных.

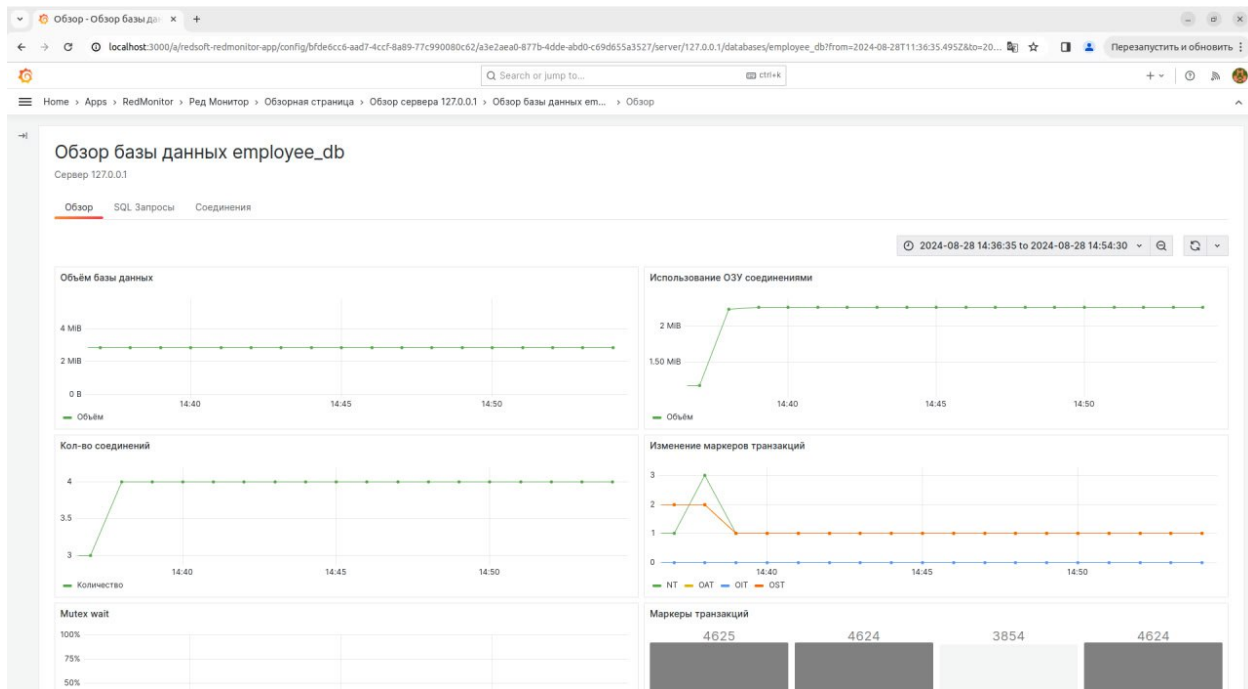


Рисунок 3.6 — Обзор выбранной базы данных

График **Объём базы данных** показывает размер базы данных в зависимости от времени.

График **Использование ОЗУ соединениями** показывает использование оперативной памяти соединениями с выбранной базой данных.

График **Количество соединений** показывает количество активных соединений с выбранной базой данных.

График **Изменение маркеров транзакций** показывает изменение значения маркеров транзакций, произошедшее с прошлого запроса информации.

График **Маркеры транзакций** отражает значения маркеров транзакций.

График **Mutex wait** показывает процент попыток, которые были заблокированы, когда владелец старался обратиться к таблице блокировок.

График **Выполнение запросов** показывает количество выполненных запросов в зависимости от времени.

График **Количество блокировок** показывает количество блокировок в зависимости от времени.

SQL запросы

На этой странице отображается краткая информация о выполненных запросах к выбранной базе данных. Для получения подробной информации о запросе нужно нажать на его хэш.

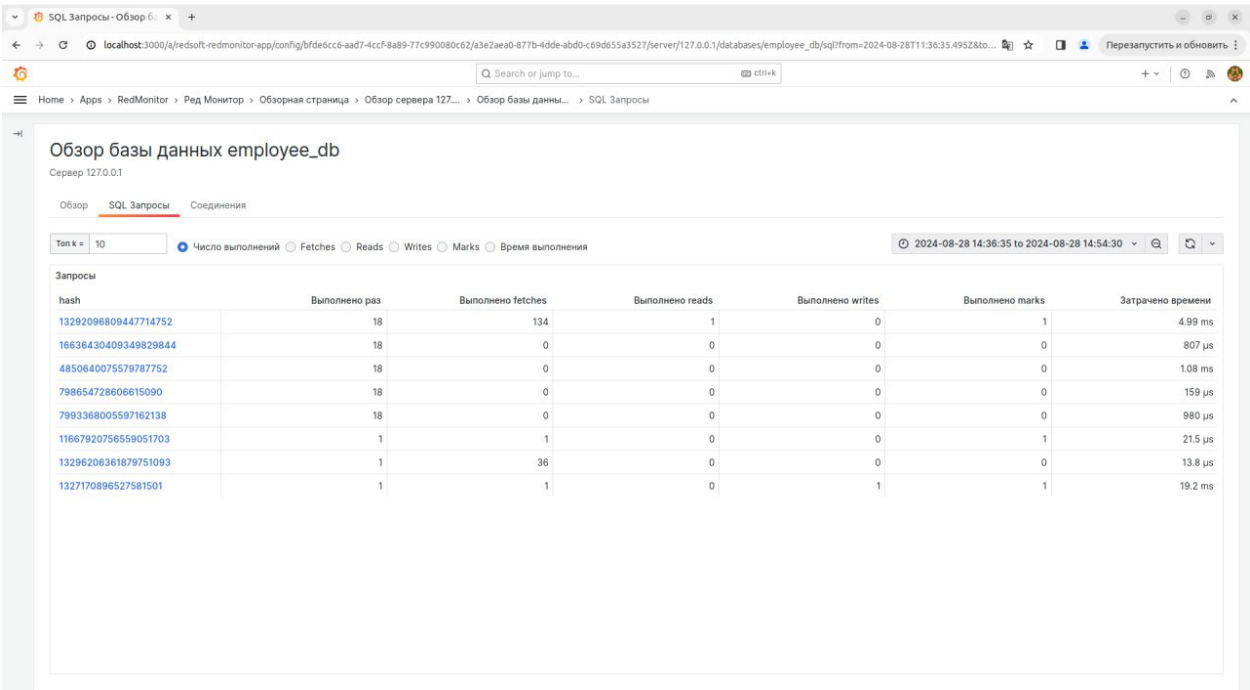


Рисунок 3.7 — SQL запросы выбранной базы данных

Обзор запроса

На данной странице находится подробная информация о выбранном запросе.

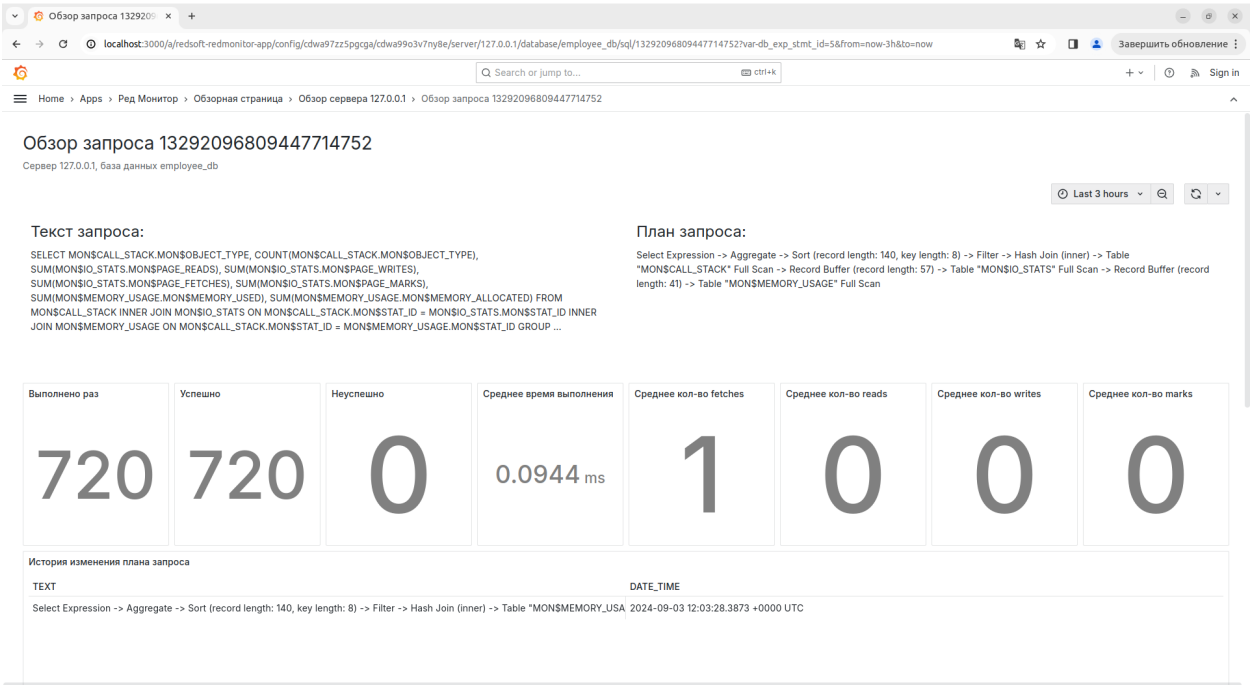


Рисунок 3.8 — Обзор запроса

Страница содержит текст запроса, план запроса и его изменения, информацию о количестве и времени выполнений. График **Операции над страницами** показывает количество страниц считанных из страничного кэша, считанных с диска, записанных на него и изменённых в страничном кэше в зависимости от времени.

Соединения

На этой странице отображается краткая информация об активных соединениях с выбранной базой данных. Для получения подробной информации о подключении нужно нажать на его ID.

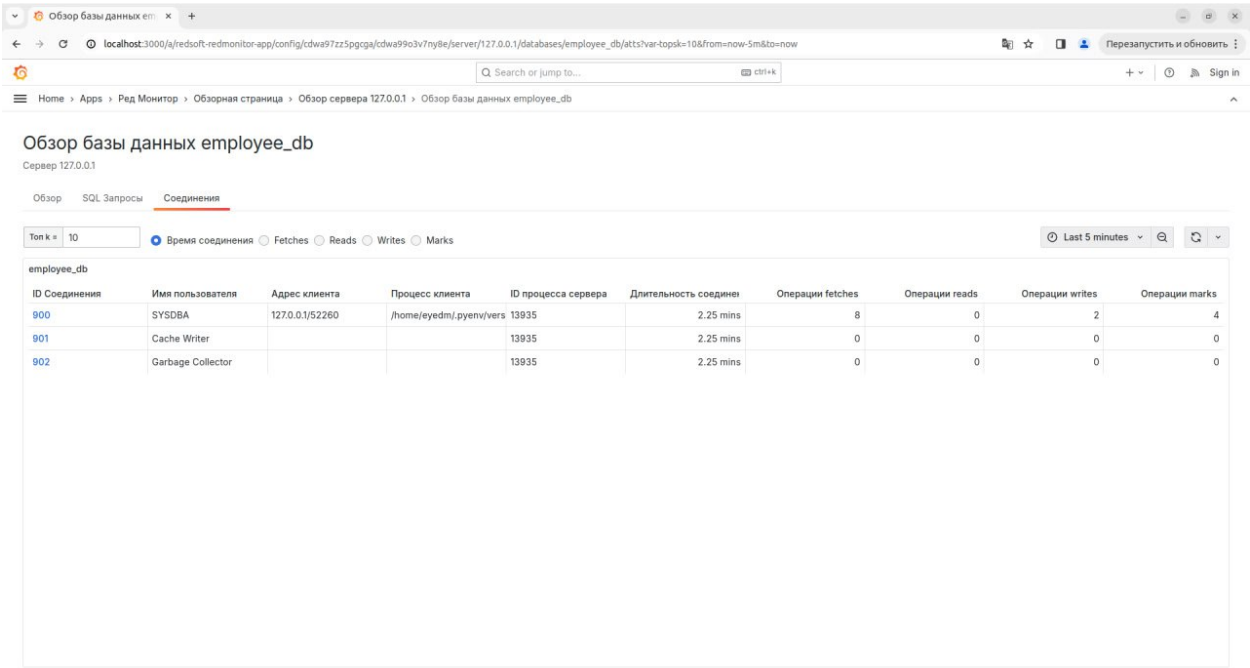


Рисунок 3.9 — Соединения с базой данных

Обзор соединения

На данной странице находится подробная информация о выбранном соединении.

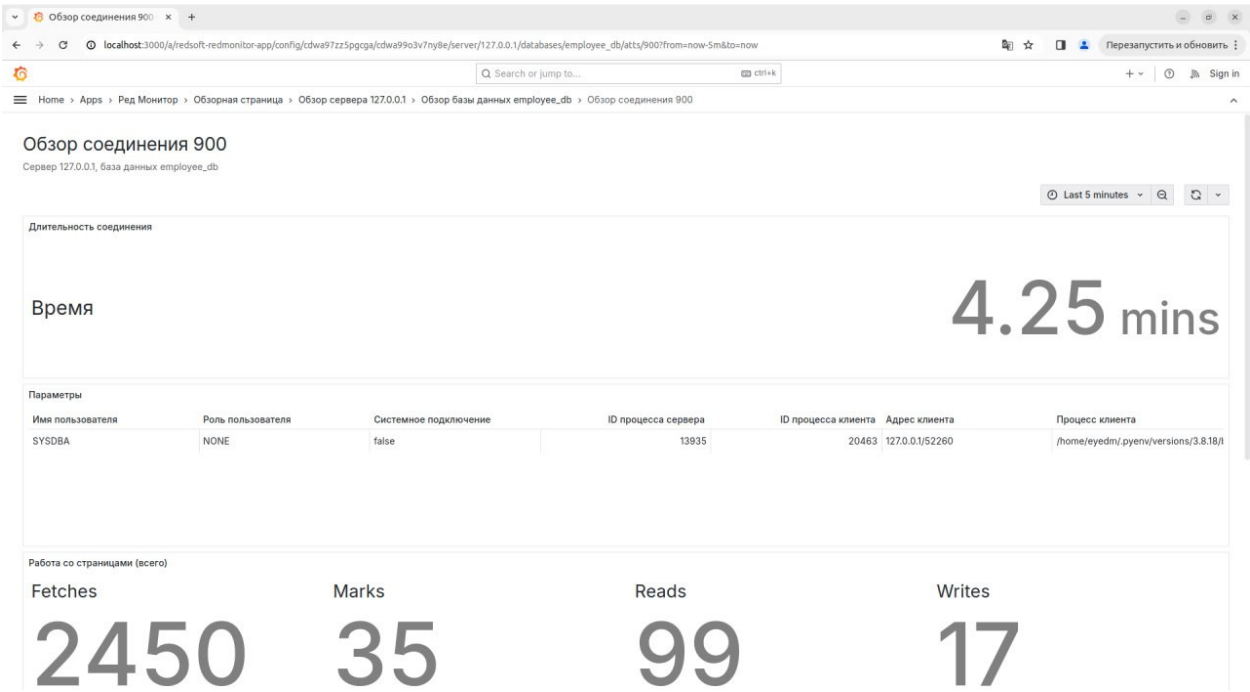


Рисунок 3.10 — Обзор соединения

Страница содержит информацию о длительности соединения, параметрах подключения, количестве обработанных страниц и объёме оперативной памяти, которая используется соединением.

3.4.3 Процессы

На этой странице отображается краткая информация о процессах СУБД. Для получения подробной информации о процессе нужно нажать на его ID.

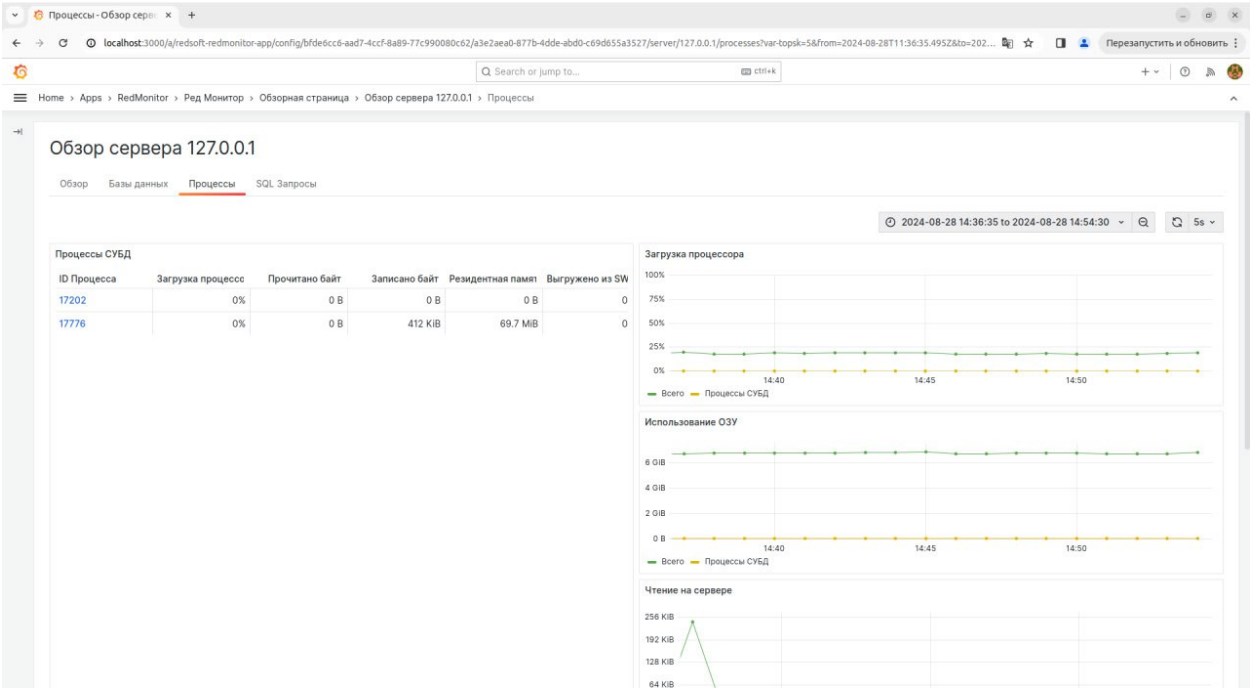


Рисунок 3.11 — Процессы СУБД

Страница показывает текущую нагрузку на процессор, использование памяти, объём записи и чтения диска, объём памяти, выгруженной в SWAP.

Обзор процесса

На данной странице находится подробная информация о выбранном процессе.

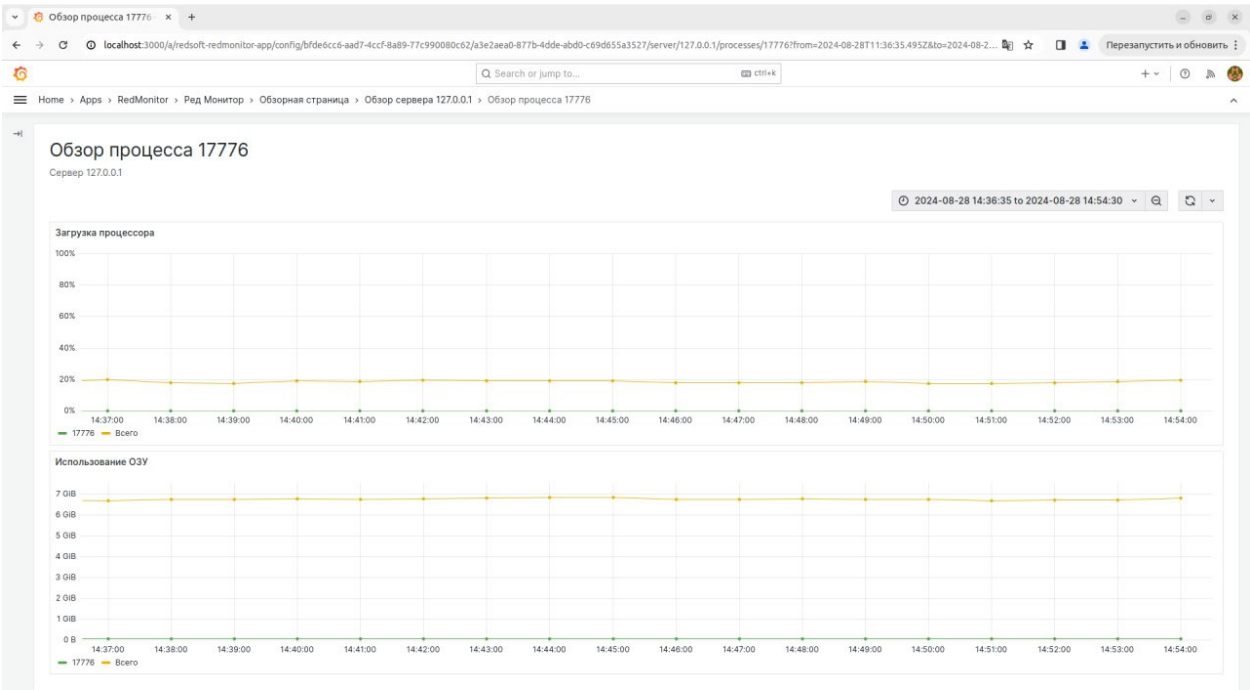


Рисунок 3.12 — Обзор процесса

3.4.4 SQL-запросы

На странице находится краткая информация о выполненных запросах, которые соответствуют заданному фильтру. По умолчанию отображаются 10 запросов, которые выполнялись чаще всего.

Для получения подробной информации о запросе нужно нажать на его хэш. Подробнее см. [Обзор запроса](#).

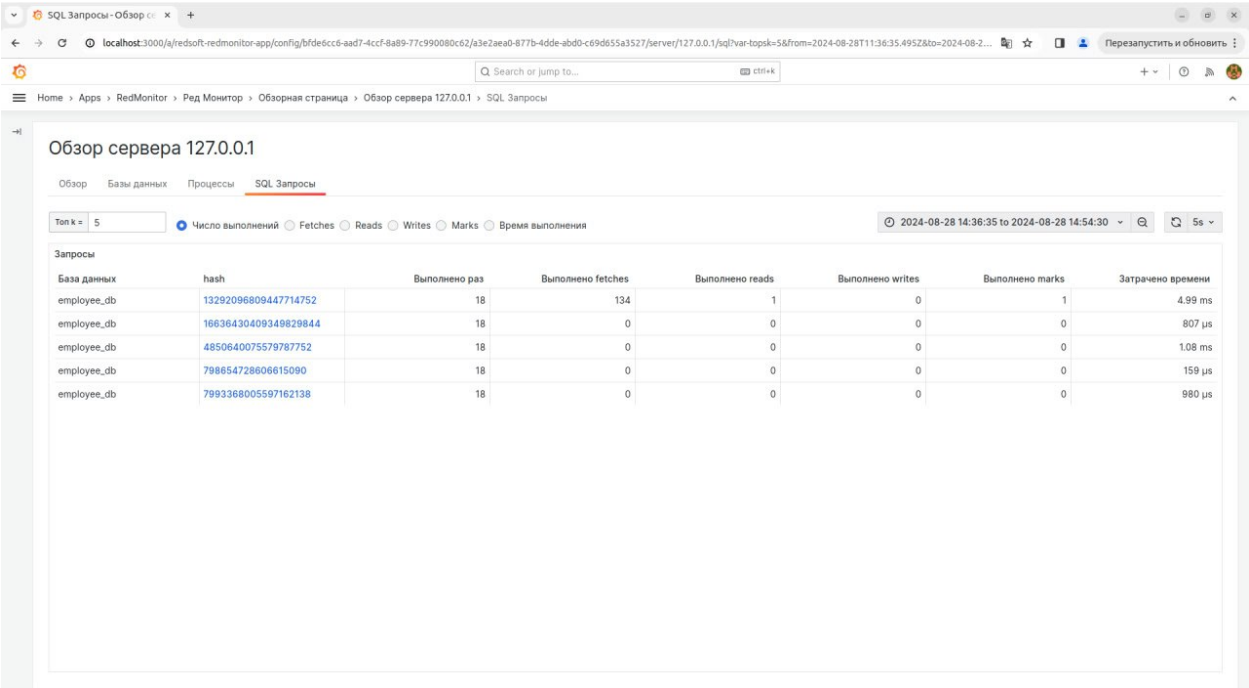


Рисунок 3.13 — Запросы

Приложение А Собираемые метрики

А.1 Метрики таблиц мониторинга

Таблица А.1 — Метрики таблиц мониторинга

Название метрики	Метки	Описание метрики
rdb_call_stack_pages	database - база данных. object_type - тип объекта базы данных: • procedure; • trigger; • function. operation - тип операции: • reads; • fetches; • marks; • writes.	Количество операций указанного типа, выполненных вызовами со страницами базы данных в момент сбора информации.
rdb_call_stack_memory	database - база данных. object_type - тип объекта базы данных: • procedure; • trigger; • function; usage - использование памяти: • used - объём используемой памяти; • allocated - количество выделенной памяти.	Объём оперативной памяти в байтах, используемой для указанного типа вызова в момент сбора информации.
rdb_call_stack_count	database - база данных. object_type - тип объекта базы данных: • procedure; • trigger; • function;	Количество вызовов указанного типа, выполняемых в момент сбора информации.
rdb_attachments_pages	database - база данных. att_id - идентификатор соединения. operation - тип операции: • reads; • fetches; • marks; • writes.	Количество операций, выполненных соединением со страницами базы данных в момент сбора информации.

(разрыв таблицы)

(разрыв таблицы)

Название метрики	Метки	Описание метрики
rdb_attachments_memory	database - база данных. att_id - идентификатор соединения. usage - использование памяти: • used - объём используемой памяти; • allocated - количество выделенной памяти.	Объём оперативной памяти в байтах, используемой соединением в момент сбора информации.
rdb_attachments_connection_time	database - база данных. att_id - идентификатор соединения.	Продолжительность соединения в наносекундах.
rdb_attachments_count	database - база данных.	Количество соединений с базой данных в момент сбора информации.
rdb_transactions_pages	database - база данных. is_active - активна ли транзакция; isolation_mode - уровень изоляции: • consistency • concurrency • read_committed_rec_ver • read_committed_no_rec_ver • read_committed_read_consistency read_only - выполняется ли транзакция в режиме "read_only"; auto_commit - используется ли режим автоматической фиксации; auto_undo - используется ли автоматическая отмена транзакции; operation - тип операции: • reads; • fetches; • marks; • writes.	Количество операций, выполненных транзакциями.

(разрыв таблицы)

(разрыв таблицы)

Название метрики	Метки	Описание метрики
rdb_transactions_memory	database - база данных. is_active - активна ли транзакция. isolation_mode - уровень изоляции: • consistency • concurrency • read_committed_rec_ver • read_committed_no_rec_ver • read_committed_read_consistency read_only - выполняется ли транзакция в режиме read_only. auto_commit - используется ли режим автоматической фиксации. auto_undo - используется ли автоматическая отмена транзакции. usage - использование памяти: • used - объём используемой памяти. • allocated - количество выделенной памяти.	Объём памяти (в байтах), используемой транзакциями, выполняющимися в момент сбора информации.
rdb_transactions_count	database - база данных. is_active - активна ли транзакция. isolation_mode - уровень изоляции: • consistency • concurrency • read_committed_rec_ver • read_committed_no_rec_ver • read_committed_read_consistency read_only - выполняется ли транзакция в режиме read_only. auto_commit - используется ли режим автоматической фиксации. auto_undo - используется ли автоматическая отмена транзакции.	Количество транзакций, выполняющихся в момент сбора информации.

(разрыв таблицы)

(разрыв таблицы)

Название метрики	Метки	Описание метрики
rdb_statements_pages	database - база данных. state - состояние запроса: • idle — бездействующий; • active — активный; • stalled — приостановленный, то есть запрос или курсор "живой" , но в данный момент не выполняется. operation - тип операции: • reads; • fetches; • marks; • writes.	Количество запросов, которые выполняются над страницами базы данных в момент сбора информации.
rdb_statements_memory	database - база данных. state - состояние запроса: • idle — бездействующий; • active — активный; • stalled — приостановленный, то есть запрос или курсор "живой" , но в данный момент не выполняется. usage - использование памяти: • used - объём используемой памяти. • allocated - количество выделенной памяти.	Объём памяти (в байтах), используемой запросами, выполняющимися в момент сбора информации.
rdb_statements_count	database - база данных. state - состояние запроса: • idle — бездействующий; • active — активный; • stalled — приостановленный, то есть запрос или курсор "живой" , но в данный момент не выполняется.	Количество запросов, выполняющихся в момент сбора информации.
rdb_response_times	database - база данных. table - таблица MON\$DATABASE.	Время выполнения запроса к таблице MON\$DATABASE в секундах.
rdb_database_sql_dialect	database - база данных.	SQL диалект.
rdb_database_page_size	database - база данных.	Размер страницы файлов базы данных в байтах.
rdb_database_page_buffers	database - база данных.	Количество страниц, выделенных в оперативной памяти для кэша;

(разрыв таблицы)

(разрыв таблицы)

Название метрики	Метки	Описание метрики
<code>rdb_database_sweep_interval</code>	database - база данных.	Интервал автоматической сборки мусора;
<code>rdb_database_read_only</code>	database - база данных.	Является ли база данных доступной только для чтения. Read-only - значение 1, read-write значение 0.
<code>rdb_database_forced_writes</code>	database - база данных.	Указывает, установлен для базы режим синхронного вывода (forced writes , значение 1) или режим асинхронного вывода (значение 0).
<code>rdb_database_backup_state</code>	database - база данных.	Указывает состояние бэкапа базы данных: • 0 — база не затронута бэкапом, • 1 — база заблокирована для резервирования, • 2 — объединение временного файла дельты и основного файла базы данных.
<code>rdb_database_pages</code>	database - база данных.	Количество страниц, выделенных для базы данных на внешнем устройстве.
<code>rdb_database_shutdown_mode</code>	database - база данных.	текущее состояние остановки (shutdown) базы данных: • 0 — база данных активна (online); • 1 — остановлена для нескольких пользователей (multi-user shutdown); • 2 — остановлена для одного пользователя (single-user shutdown); • 3 — полностью остановка (full shutdown).
<code>rdb_database_size</code>	database - база данных.	Объём памяти в байтах, занимаемый базой данных на диске. Это произведение количества выделенных страниц и размера одной страницы базы данных.
<code>rdb_database_cache</code>	database - база данных.	Размер выделенной оперативной памяти под кеш СУБД в байтах.

(разрыв таблицы)

(разрыв таблицы)

Название метрики	Метки	Описание метрики
rdb_database_up	database - база данных.	Указывает на наличие соединения экспортёра с базой данных: • 0 - соединение отсутствует; • 1 - соединение установлено.

A.2 Метрики агрегатного аудита

Эти метрики показывают на сколько изменилось значение с предыдущего сбора информации.

Таблица A.2 — Метрики агрегатного аудита

Название метрики	Метки	Описание метрики
rdb_aggtrace_stmt_pages	database - база данных; hash - хэш запроса; q_operation - тип запроса: • select • insert • update • create • delete • drop operation - тип операции: • reads; • fetches; • marks; • writes. marker - количество операций: • min; • max; • avg.	Количество операций, выполненных запросом.
rdb_aggtrace_stmt_times	database - база данных; hash - хэш запроса; q_operation - тип запроса: • select • insert • update • create • delete • drop marker - затраченное время: • min; • max; • avg.	Время, затраченное на выполнение запросов.

(разрыв таблицы)

(разрыв таблицы)

Название метрики	Метки	Описание метрики
rdb_aggtrace_stmt_count	database - база данных; hash - хэш запроса; q_operation - тип запроса: • select • insert • update • create • delete • drop marker - результат выполнения: • failed • succeed • total	Количество выполненных запросов выбранного типа.
rdb_aggtrace_sort_mu_total	database - база данных; hash - хэш запроса; q_operation - тип запроса: • select • insert • update • create • delete • drop marker - результат выполнения: • failed • succeed • total	Объём памяти (в байтах), выделенной под сортировку.
rdb_aggtrace_sort_mu_disk	database - база данных; hash - хэш запроса; q_operation - тип запроса: • select • insert • update • create • delete • drop marker - результат выполнения: • failed • succeed • total	Объём памяти (в байтах), выделенной под сортировку на диске.

(разрыв таблицы)

(разрыв таблицы)

Название метрики	Метки	Описание метрики
rdb_aggtrace_sort_mu_cache	database - база данных; hash - хэш запроса; q_operation - тип запроса: • select • insert • update • create • delete • drop marker - результат выполнения: • failed • succeed • total	Объём памяти (в байтах), выделенной под сортировку в кэше.

A.3 Метрики утилиты rdb_lock_print

Таблица A.3 — Метрики утилиты rdb_lock_print

Название метрики	Метки	Описание метрики
rdb_lock_table_memory_usage	database - база данных; usage: • length - общий объем памяти, выделенный таблице блокировок (в байтах); • used - наибольшая величина смещения в таблице блокировок, которая используется в настоящий момент.	Объем памяти, выделенный таблице блокировок (в байтах).
rdb_lock_table_requests_count	database - база данных; kind - тип запроса: • enqs - число запросов, полученных на блокировку (не включает запросы, которые пришли и ушли); • converts - запросы на повышение уровня блокировки; • rejects - запросы, которые не могут быть удовлетворены; • blocks - Запросы, которые не могут быть удовлетворены немедленно.	Количество запросов выбранного типа.

(разрыв таблицы)

(разрыв таблицы)

Название метрики	Метки	Описание метрики
rdb_lock_table_deadlocks_count	database - база данных; counter: • scans - количество проверок на взаимные блокировки; • found - количество найденных взаимных блокировок.	Подсчёт взаимных блокировок.
rdb_lock_table_scan_interval	database - база данных;	Время (в секундах), которое ожидает менеджер блокировок до того как запустить к поиску взаимных блокировок.
rdb_lock_table_acquires	database - база данных; counter: • requires - сколько раз владелец запрашивает исключительное управление таблицей блокировок, чтобы выполнить изменения; • blocks - сколько раз владелец находился в состоянии ожидания при запросе исключительного управления таблицей блокировок.	Запросы на исключительное управление таблицей блокировок.
rdb_lock_table_mutex_wait	database - база данных;	Процент попыток, которые были заблокированы, когда владелец старался обратиться к таблице блокировок.
rdb_lock_table_hash_slots_count	database - база данных;	Число слотов кэширования блокировок.
rdb_lock_table_hash_slots_lengths	database - база данных; length: • min; • max; • avg.	Длина цепочки кэширования.
rdb_lock_table_owners_count	database - база данных; kind - тип владельца: • process • database • connection • transaction • dummy_process	Количество владельцев, соединенных с таблицей блокировок.
rdb_lock_table_owners_requests_avg	database - база данных;	Среднее количество групп запросов, которые были освобождены и не использованы повторно.

(разрыв таблицы)

(разрыв таблицы)

Название метрики	Метки	Описание метрики
rdb_lock_table_locks_count	database - база данных; series - тип ресурса; is_pended - ожидается ли кем-то.	Количество блокировок указанного типа.
rdb_lock_table_locks_memory	database - база данных; series - тип ресурса; is_pended - ожидается ли кем-то; usage - использование памяти: - used - объём используемой памяти. - allocated - количество выделенной памяти.	Объём выделенной памяти в байтах для указанной блокировки.
rdb_lock_table_pending_req_count_avg	database - база данных; series - тип ресурса;	Среднее количество ожидающих владельцев для указанного типа ресурса.

A.4 Метрики репликации

Таблица A.4 — Метрики репликации

Название метрики	Метки	Описание метрики
rdb_master_segments_count	database - база данных; state - состояние сегмента: - full - arch - free - used	Количество сегментов указанного состояния.
rdb_master_current_sequence	database - база данных.	Номер текущего сегмента
rdb_master_segments_size_in_use	database - база данных.	Общий размер журнала репликации в байтах.
rdb_slave_current_segment	database - база данных.	Номер текущего сегмента
rdb_slave_oldest_segment	database - база данных.	Сегмент, который начал самую раннюю транзакцию, которая не была завершена во время обработки последней последовательности.
rdb_slave_active_transaction	database - база данных.	Количество активных транзакций.
rdb_slave_segments_in_queue	database - база данных.	Количество сегментов в очереди.
rdb_master_pings	from to	Задержка ответа между серверами в секундах.

(разрыв таблицы)

(разрыв таблицы)

Название метрики	Метки	Описание метрики
rdb_repl_log_size	database - база данных.	Общий размер журнала репликации в байтах.

A.5 Метрики операционной системы

Таблица A.5 — Метрики операционной системы

Название метрики	Метки	Описание метрики
rdb_disks_io_counters	database - база данных. disk - диск. operation - тип операции: • reads • writes	Количество операций чтения и записи на диск.
rdb_disks_io_bytes	database - база данных. disk - диск. operation - тип операции: • reads • writes	Объём прочитанной/записанной на диск информации в байтах.
rdb_disks_io_times	database - база данных. disk - диск. operation - тип операции: • reads • writes	Время (в миллисекундах), затраченное на выполнение операций чтения/записи на диск.
rdb_disks_io_wtime	database - база данных. disk - диск.	Взвешенное время, потраченное на операции ввода-вывода.
rdb_disks_io_time	database - база данных. disk - диск.	Время, потраченное на фактический ввод-вывод.
rdb_mount_point_usage	database - база данных. point - точка монтирования. usage: • used • free • total	Использование точки мониторинга.

(разрыв таблицы)

(разрыв таблицы)

Название метрики	Метки	Описание метрики
rdb_other_procs_memory_usage	usage - тип операции: • resident - объём резидентной памяти, используемой процессами, не относящимся к СУБД. • virtual - объём виртуальной памяти, выделенной под процессы, не относящиеся к СУБД. • swapped_out - количество записанных на диск байт при нехватке ОЗУ.	Количество прочитанной и записанной информации в байтах процессами, которые не относятся к СУБД.
rdb_other_procs_io_bytes	operation - тип операции: • reads • writes	Количество прочитанной и записанной информации в байтах процессами, которые не относятся к СУБД.
rdb_opened_temp_files_size	temp_kind - тип временного объекта: • table • blob • undo • redbuf • merge • sort • tpc • snap	Размер временных файлов, созданных процессами СУБД (в байтах).
rdb_opened_temp_files_count	temp_kind - тип временного объекта: • table • blob • undo • redbuf • merge • sort • tpc • snap	Количество временных файлов, созданных процессами СУБД.
rdb_procs_cpu_usage	pid	Загрузка процессора процессами СУБД.
rdb_procs_memory_usage	pid usage: • resident • virtual • swapped_out	Объём резидентной, виртуальной и выгруженной в swap памяти в байтах.

(разрыв таблицы)

(разрыв таблицы)

Название метрики	Метки	Описание метрики
rdb_procs_io_bytes	pid operation - тип операции: • reads • writes	Размер прочитанной и записанной информации в байтах процессами СУБД.
rdb_procs_count	is_rdb - относится ли процесс к СУБД.	Количество процессов.
rdb_system_memory	field: • used • availabel • total	Использование оперативной памяти.
rdb_system_swap_usage	usage: • used • free • total	Использование памяти, выделенной для swap.
rdb_system_swapped	direction: • in • out	Объём данных (в байтах), затронутый swap.
rdb_system_cpu_load	group: • user • system	Загрузка процессора.
rdb_system_cpu_freq	marker: • max • min • current	Частота работы процессора.
rdb_system_cpu_ctx		Число изменений контекста процессора.
rdb_system_cpu_interrupts	marker: • soft • casual	Число прерываний процессора.
rdb_tmp_mount_point_usage	point usage: • total • used • free	Использование точки монтирования.
rdb_tmp_disks_io_bytes	disk operation: • reads • writes	Показывает количество байт прочитанных с диска и записанных на него.

(разрыв таблицы)

(разрыв таблицы)

Название метрики	Метки	Описание метрики
rdb_tmp_disks_io_counters	disk operation: • reads • writes	Показывает количество операций чтения с диска и записи на него.
rdb_tmp_disks_io_times	disk operation: • reads • writes	Показывает количество времени в миллисекундах, потраченного на чтение с диска и запись на него.
rdb_tmp_dir_size	name point	Показывает объём временных файлов СУБД.

А.6 Другие метрики

Таблица А.6 — Другие метрики

Название метрики	Метки	Описание метрики
rdb_transactions_markers	database - база данных; marker: • NT • OST • OAT • OIT	Значения маркеров транзакций в момент сбора информации.
rdb_log_errors	kind - тип ошибки: • fatal • critical • normal • event	Количество ошибок указанного типа.