РБДМонитор

Версия 0.9

Руководство пользователя

Содержание

Глава 1

Общие сведения о программе

1.1 Назначение программы

РБДМонитор - это система для мониторинга СУБД Ред База Данных.

РБДМонитор предназначен для наблюдения за состоянием СУБД. Он предоставляет возможность мониторинга сразу нескольких баз данных и серверов. Показывает подробную информацию о пользователях, соединениях, запросах, ошибках и отображает топ соединений и запросов сервера по указанной характеристике за определённое время. Например, самые долгие запросы, выполненные за последний час. Помогает следить за изменением планов запросов. Показывает, какую нагрузку на сервер и операционную систему оказывает наблюдаемая СУБД.

`Экспортёр`, `Ред База Данных`, `Prometheus`, `Alertmanager`, `Grafana`, `Loki` и `Promtail` вместе образуют продукт `РБДМонитор`. Экспортёр используется для сбора показателей состояния (метрик) СУБД и отправки их в систему мониторинга `Prometheus`. Для управления уведомлениями используется `Alertmanager`. Для визуализации собранных метрик используется `Grafana`.

Подробное описание собираемых метрик см. в приложении *Собираемые метрики*.

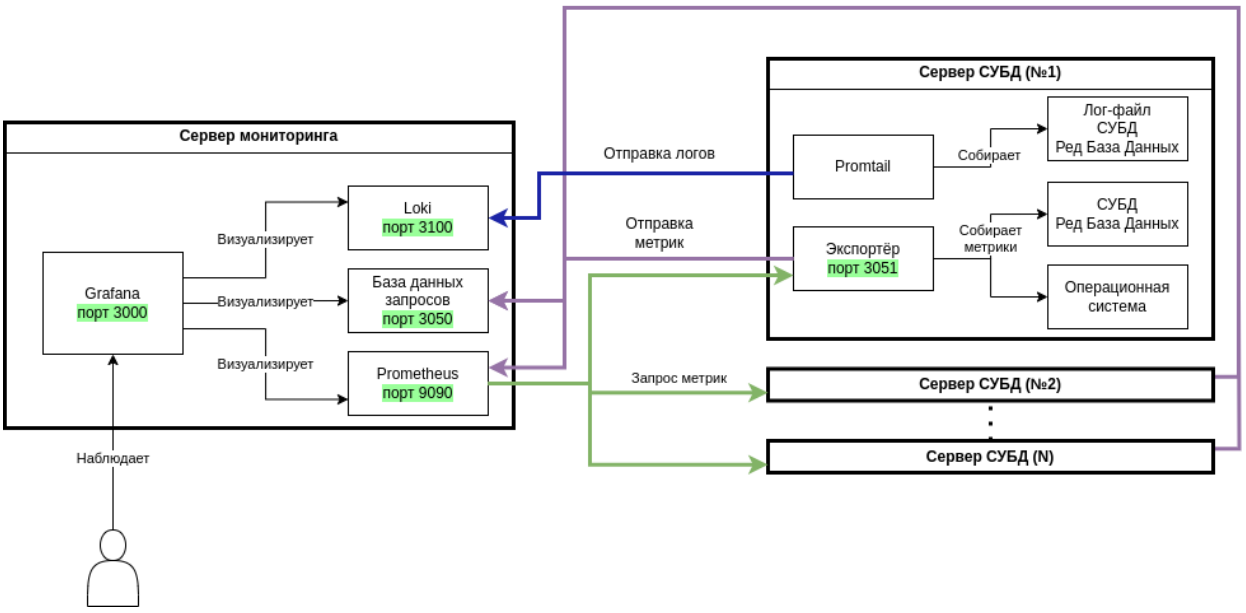

Рисунок 1.1 — Схема взаимодействия компонентов РБДМонитора

- `Сервер мониторинга` - сервер, на котором работают программы `Prometheus`, СУБД `Ред База Данных` и `Grafana`.
- `Сервер СУБД` - сервер, на котором работает `Наблюдаемая СУБД` 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 с `Наблюдаемой СУБД` и отправляют данные обратно в `Prometheus`, Некоторые данные (например, текст SQL-запроса) отправляются на хранение в базу данных пользовательских запросов.

Для отображения данных сервер мониторинга использует `Grafana`. Она получает данные из `Prometheus`, `Базы данных пользовательских запросов` и `Loki`.

Порты, используемые по умолчанию:

- Сервер мониторинга:
  - `Grafana` - порт 3000;
  - `Prometheus` - порт 9090;
  - `Loki` - порт 3100;
  - СУБД с `базой данных пользовательских запросов` - порт 3050.
- Сервер СУБД:
  - `Экспортёр` - порт 3051.

Сервер мониторинга и сервер СУБД можно установить как на одну машину, так и на разные.

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 РБДМонитор 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;
- Установить `Loki` версии не ниже 3.3.1;
- Установить `Promtail` версии не ниже 3.3.1.

Глава 2

Установка РБДМонитор

Все компоненты РБДМонитор можно устанавливать на один сервер.

Последовательность запуска компонентов важна. Первым необходимо запустить сервер мониторинга, после него запустить сервер СУБД.

2.1 Сервер мониторинга

Все элементы сервера мониторинга, а именно `Prometheus`, `Alertmanager`, `Grafana`, `Ред База Данных` и `Loki` должны быть установлены на одну машину или иметь между собой связь по сети.

2.1.1 Установка компонентов сервера мониторинга с помощью rpm-пакета

Из rpm-пакета установится `Prometheus`, `Alertmanager`, `Grafana` и `Loki`. Ред Базу Данных необходимо установить самостоятельно, следуя инструкции в Руководстве администратора.

1. Скачайте rpm-пакет сервера мониторинга с официального сайта.
2. Установите компоненты сервера мониторинга, выполнив следующую команду:

```
dnf install <путь_до_rpm_пакета_rdbmonitor-server>
```

Компоненты сервера мониторинга, входящие в состав rpm-пакета, будут расположены по следующим путям:

- `Prometheus` - `/etc/prometheus/`;
- `Alertmanager` - `/etc/alertmanager/`;
- `Grafana` - `/etc/grafana/`;
- `Loki` - `/etc/loki/`.

2.1.2 Установка компонентов сервера мониторинга по отдельности

Установка Prometheus

Установить `Prometheus` в РЕД ОС можно с помощью пакетного менеджера операционной системы:

Для установки `Prometheus` выполните команду:

```
dnf install golang-github-prometheus
```

Другой вариант установки - скачать `Prometheus` с официального сайта (prometheus.io) и выполнить следующее:

1. Распаковать скачанный архив:

```
tar -xf <путь_к_архиву>
```

2. Переместить распакованный архив в предварительно созданную папку:

```
sudo mv -Z ./<распакованный архив> /opt/prometheus
```

3. Создать пользователя и группу `prometheus`:

```
sudo useradd -M -U prometheus
```

4. Назначить права на папку `/opt/prometheus`:

```
sudo chown prometheus:prometheus -R /opt/prometheus
```

5. Создать unit-файл для запуска через службу:

```
[Unit]
Description=Prometheus Server
Documentation=https://prometheus.io/docs/introduction/overview/
After=network-online.target

[Service]
User=prometheus
Group=prometheus
Restart=on-failure
ExecStart=/opt/prometheus/prometheus \
--config.file=/opt/prometheus/prometheus.yml \
--storage.tsdb.path=/opt/prometheus/data \
--storage.tsdb.retention.time=30d

[Install]
WantedBy=multi-user.target
```

После создания unit-файла выполнить:

```
sudo systemctl daemon-reload
```

Установка Alertmanager

> Если уведомления не нужны, то этот пункт можно пропустить.

Установить `Alertmanager` в РЕД ОС можно с помощью пакетного менеджера операционной системы:

Для установки `Alertmanager` выполните команду:

```
dnf install golang-github-prometheus-alertmanager
```

Другой вариант - скачать `Alertmanager` с официального сайта -- prometheus.io.

Для установки нужно выполнить следующие действия:

1. Распаковать скачанный архив:

```
tar -xf <путь_к_архиву>
```

2. Переместить распакованный архив в предварительно созданную папку:

```
sudo mv -Z ./<распакованный архив> /opt/alertmanager
```

3. Создать пользователя и группу `alertmanager`:

```
sudo useradd -M -U alertmanager
```

4. Назначить права на папку `/opt/alertmanager`:

```
sudo chown alertmanager:alertmanager -R /opt/alertmanager
```

5. Создать unit-файл для запуска через службу:

```
[Unit]
Description=Prometheus Alertmanager
Documentation=https://prometheus.io/docs/alerting/latest/overview/
After=network-online.target

[Service]
User=alertmanager
Group=alertmanager
Restart=on-failure
ExecStart=/opt/alertmanager/alertmanager \
--config.file=/opt/alertmanager/alertmanager.yml \
--storage.path=/opt/alertmanager/data

[Install]
WantedBy=multi-user.target
```

После создания unit-файла выполнить:

```
sudo systemctl daemon-reload
```

Установка Grafana

На Ред ОС `Grafana` можно установить через пакетный менеджер:

```
sudo dnf install grafana
```

На других операционных системах нужно скачать `Grafana` с официального сайта и установить по инструкции.

Перенесите ранее скаченный архив Экспортёра на сервер мониторинга и установите плагины командой:

```
sudo ./install.sh -o install -p grafana_plugins
```

При установке плагинов для `Grafana` нужно внести `redsoft-rdbmonitor-app` и `redsoft-reddatabase-datasource` в список доверенных, разрешив редактировать файл `/etc/`

grafana/grafana.ini:

```
Установка плагинов для grafana.
Введите путь до папки плагинов: /var/lib/grafana/plugins
Копирование файлов...
Успешно!

Введите путь до файла конфигурации Grafana: /etc/grafana/grafana.ini
Включение плагинов в список доверенных...
Успешно!

Плагины Grafana установлены!

Копирование скрипта удаления...
Успешно!
```

В случае успешной установки в файле конфигурации Grafana (по умолчанию grafana.ini) будут указаны плагины redsoft-rdbmonitor-app и redsoft-reddatabase-datasource:

```
allow_loading_unsigned_plugins = redsoft-rdbmonitor-app,redsoft-reddatabase-
datasource, <другие плагины>
```

Плагины для Grafana будут установлены на сервер мониторинга.

Установка Ред Базы Данных

Инструкцию по установке Ред Базы Данных см. в Руководстве администратора.

Установка Loki

Loki - это инструмент, необходимый для хранения логов, собираемых с помощью экспортёра логов (Promtail).

Loki необходимо установить на сервер мониторинга, выполнив команду:

```
sudo ./install.sh -o install -p loki
```

2.2 Сервер СУБД

Все элементы сервера СУБД, а именно Ред База Данных (наблюдаемая СУБД), Экспортёр и Promtail должны быть установлены на одну машину.

2.2.1 Установка компонентов сервера СУБД с помощью rpm-пакета

Из rpm-пакета установится Экспортёр и Promtail. Ред Базу Данных необходимо установить самостоятельно.

1. Установите Ред Базу Данных, следуя инструкции в Руководстве администратора.
2. Скачайте rpm-пакет сервера СУБД с официального сайта.

3. Установите компоненты сервера СУБД, выполнив следующую команду:

```
dnf install <путь_до_rpm_пакета_rdbmonitor-client>
```

Компоненты сервера СУБД, входящие в состав rpm-пакета, будут расположены по следующим путям:

- Экспортёр - /opt/rdbmonitor/client/exporter/.
- Promtail - /opt/rdbmonitor/client/promtail/.

2.2.2 Установка компонентов сервера СУБД по отдельности

Установка Ред Базы Данных

Инструкцию по установке Ред Базы Данных см. в Руководстве администратора.

Установка Экспортёра

Экспортёр необходимо скачать по ссылке. После скачивания распакуйте архив, выполнив команду:

```
tar -xzf <архив>
```

Архив будет распакован в текущую директорию.

Перейдите в каталог экспортёра:

```
cd <каталог экспортёра>
```

Установите Экспортёр, выполнив install.sh с правами администратора:

```
sudo ./install.sh -o <install | update> -p <client | grafana_plugins [-d] | loki>
```

- -o указывает, какую операци нужно выполнить: установку или обновление.
- -p определяет, что нужно установить/обновить. Опция client используется для установки/обновления всего, что необходимо для сервера СУБД (экспортёр и Promtail). На сервер мониторинга необходимо установить плагины для Grafana и Loki, используя соответствующие опции. Установка плагинов для Grafana доступна, только если Grafana уже установлена на сервере мониторинга. При установке/обновлении плагинов для Grafana нужно будет указать путь до конфигурационного файла (/etc/grafana/grafana.ini) и путь до папки с плагинами Grafana. Если используются пути по умолчанию, то можно использовать опцию -d.
- -d указывает использовать пути по умолчанию при установке/обновлении плагинов для Grafana. Конфигурационный файл Grafana по умолчанию располагается по пути /etc/grafana/grafana.ini, а папка с плагинами Grafana по пути /var/lib/grafana/plugins/. Опция -d доступна, только если Grafana установлена на сервере мониторинга.
- -h выводит справку о доступных опциях.

Экспортёр будет установлен на сервер СУБД по пути /opt/rdbmonitor.

Для удаления РБДМонитор выполните команду:

```
sudo ./uninstall.sh [<опции>]

<опции>::=
  client
```

(продолжение на следующей странице)

(продолжение с предыдущей страницы)

```
| grafana_plugins
| loki
```

Установка Promtail

`Promtail` - это инструмент, необходимый для сбора логов СУБД и операционной системы.

`Promtail` устанавливается на сервер СУБД вместе с экспортёром командой:

```
sudo ./install.sh -o install -p client
```

Глава 3

Настройка РБДМонитор

Последовательность запуска компонентов важна. Первым необходимо запустить сервер мониторинга, после него запустить сервер СУБД.

3.1 Сервер мониторинга

В этом разделе находится описание настроек программ, которые должны быть установлены на сервере мониторинга.

3.1.1 Ред База Данных

Базу данных для хранения пользовательских запросов нужно создать самостоятельно на `Сервере мониторинга`, то есть там, где запущены `Prometheus`, `Grafana` и СУБД:

1. Создайте каталог `/db`:

```
sudo mkdir /db
```

2. Назначьте на каталог `/db` права пользователю `reddatabase` - для Ред Базы Данных 5.0 и выше, или пользователю `firebird`:

```
sudo chown "имя_пользователя" /db/
```

3. Выполните скрипт `create_tables.sql`, расположенный по пути `/opt/rdbmonitor/server/rdb/create_tables.sql`, если Экспортёр был установлен через rpm-пакет, или по пути `/распакованный архив/dists/dictserver/create_tables.sql`, если Экспортёр был установлен, как отдельный компонент. В скрипте необходимо указать пользователя, от имени которого будет создана база данных и каталог, где она будет расположена. По умолчанию база будет создана по пути `/db/statements.fdb`.

Пример настройки скрипта:

```
CREATE DATABASE 'localhost:<путь_до_бд>'
   USER '<имя пользователя>' PASSWORD '<пароль>';
...
```

Запуск скрипта:

```
<каталог Ред Базы Данных>/bin/isql -i <путь до create_tables.sql>
```

По умолчанию доступ к СУБД с базой данных пользовательских запросов осуществляется по порту 3050.

Запуск Ред Базы Данных осуществляется следующей командой:

```
systemctl start firebird
```

3.1.2 Loki

Запуск Loki выполняется следующей командой:

```
sudo systemctl start loki.service
```

3.1.3 Prometheus

Основные настройки Prometheus

Prometheus собирает и хранит метрики СУБД. Но обращаться к наблюдаемой базе напрямую он не может, поэтому для передачи данных от СУБД к Prometheus используется Экспортёр.

Для получения уведомлений о состоянии наблюдаемой базы нужно настроить правила проверки значений собранных метрик в секции rule_files. Prometheus проверяет собранные метрики на соблюдение указанным правилам с заданной периодичностью. Если правило соблюдается (например, правило для проверки загрузки процессора rate(rdb_system_cpu_times[1m]) * 100 > 20), то Prometheus сообщит об этом Alertmanager, отправив предупреждение. Alertmanager сформирует из предупреждений уведомления и отправит их на указанный адрес.

Для настройки Prometheus используется файл prometheus.yml. При установке через rpm-пакет файл будет расположен по пути /etc/prometheus/prometheus.yml.

Пример настройки prometheus.yml:

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

scrape_configs:
  - job_name: "RDBMonitor"
    static_configs:
      - targets: ["192.168.0.100:3051", "192.168.0.100:8100", "192.168.0.150:8000"]

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 нужно собирать метрики.

Параметр `scrape_timeout` устанавливает время ожидания получения метрик.

Параметр `evaluation_interval` устанавливает интервал, с которым собранные метрики будут проверяться на соответствие правилам, указанным в секции `rule_files`.

Секция rule_files

Файл `rules.yml` необходимо предварительно скопировать из корневого каталога `RDBMonitor` в корневой каталог `Prometheus`.

В секции `rule_files` указывается список файлов с правилами, на соответствие которым нужно проверять собранные метрики:

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

Секция scrape_configs

В секции `scrape_configs` задаются настройки для сбора метрик:

Пример настройки секции:

```
scrape_configs:
  - job_name: "RDBMonitor"
    scrape_interval: 15s
    scrape_timeout: 10s
    static_configs:
      - targets: ["192.168.0.100:3051", "192.168.0.100:8100", "192.168.0.150:8000"]
```

Параметр `job_name` задаёт имя группы экспортёров. Экспортёры, входящие в группу, определяются в параметре `targets`. Изменять значение параметра `job_name` нельзя.

Параметр `scrape_interval` определяет, как часто должны собираться метрики. По умолчанию принимает значение, установленное в секции `global`.

Параметр `scrape_timeout` устанавливает время ожидания получения метрик. По умолчанию принимает значение, установленное в секции `global`.

Параметр `targets` определяет сетевые узлы экспортёров, к которым будет обращаться `Prometheus`, чтобы получить метрики. Портом экспортёра по умолчанию является 3051.

Секция alerting

Секция `alerting` определяет сущности `Alertmanager`, в которые `Prometheus` будет отправлять предупреждения, когда собранные метрики соблюдают правила, указанные в секции `rule_files`.

Если нет необходимости в уведомлениях, то эту секцию нужно оставить закомментированной.

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

Запуск Prometheus

Запуск `Prometheus` осуществляется следующей командой:

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

Или можно запустить службу:

```
sudo systemctl start prometheus
```

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

3.1.4 Alertmanager

Если уведомления не нужны, то этот пункт можно пропустить.

Основные настройки Alertmanager

Чтобы получать уведомления, необходимо настроить `Alertmanager`. Для этого используется файл `/etc/alertmanager/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'

 inhibit_rules:
   - source_match:
       severity: 'critical'
     target_match:
```

(продолжение на следующей странице)

(продолжение с предыдущей страницы)

```
        severity: 'warning'
```

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

Секция global

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

```
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'
```

Параметр `smtp_require_tls` указывает, использовать ли протокол `TLS`.

Параметр `smtp_smarthost` устанавливает адрес сервера почты.

Параметр `smtp_from` указывает почту отправителя.

Параметр `smtp_auth_username` определят имя пользователя для аутентификации на сервере почты.

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

Секция route

Секция `route` определяет маршруты уведомлений в виде структуры дерева. Маршрут - это список проверок, через которые проходит уведомление для нахождения получателя.

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

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.

Секция receivers

В секции `receivers` указывается список получателей уведомлений.

```
receivers:
  - name: 'receiver_name'
```

(продолжение на следующей странице)

(продолжение с предыдущей страницы)

```
    email_configs:
    - to: 'receiver@example.com'
```

Параметр `name` указывает имя получателя.

Параметр `to` определяет почту получателя.

Секция inhibit_rules

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

```
inhibit_rules:
  - source_match:
      severity: 'critical'
    target_match:
      severity: 'warning'
```

Параметры `source_matchers` и `target_matchers` это списки меток со значениями. Те предпреждения, 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`.

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 РБДМонитор.

Файл `rules.yml` содержит правила для следующих предупреждения:

- Высокое потребление процессора в режиме ядра (СУБД);
- Высокое потребление процессора (СУБД);
- Ухудшение производительности (СУБД);
- Увеличение нагрузки (СУБД);
- Среднее время выполнения запроса увеличилось (СУБД);
- Высокая нагрузка на процессор (Сервер);
- Насыщение процессора (Сервер);
- Мало места в файловой системе (Сервер);
- Высокая утилизация диска по времени (Сервер);
- Высокое потребление памяти (Сервер);
- Высокое использование объёма подкачки (Сервер);
- Наличие процесса подкачки (Сервер);
- Экспортёр не ответчает (Экспортёр).

Файл `cluster_rules.yml` содержит правила для предупреждений о работе кластера:

- Нет ведущего узла в кластере;
- Узел стал мастером;
- Узел стал репликой;
- Узел долго находится в состоянии standalone;
- Уменьшение количества узлов кластера;
- Увеличение количества узлов кластера;
- Уменьшение количества баз данных кластера;
- Увеличение количества баз данных кластера;
- Кластер перешёл в режим обслуживания;
- Узел не синхронизирован с мастером;
- Время работы узла было сброшено;
- Длительное отсутствие синхронизации базы данных;
- Критическое расхождение номеров журналов репликации;
- Узел не может стать ведущим;
- Высокое время опроса статуса кластера;
- Высокое время обновления информации об узле.

Нужно указать в настройках `Prometheus` путь к файлу с правилами. Можно указать несколько файлов с настройками предупреждений:

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

Файлы `rules.yml` и `cluster_rules.yml`, расположенные по пути `/opt/rdbmonitor/server/prometheus/`, необходимо предварительно скопировать в корневой каталог `Prometheus`. Более подробно о настройке `Alertmanager` можно узнать в официальной документации.

Запуск Alertmanager

Запуск `Alertmanager` осуществляется следующей командой:

```
sudo -u alertmanager ./alertmanager --config.file="alertmanager.yml"
```

Или можно запустить службу `Alertmanager`:

```
sudo systemctl start alertmanager
```

3.1.5 Grafana

Запуск `Grafana` выполняется командой:

```
sudo systemctl start grafana-server
```

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/`). Необходимо настроить интервал сбора метрик. Для параметра `Scrape interval` укажите число, совпадающее со значением параметра

scrape_interval в конфигурационном файле Prometheus (prometheus.yml). Нажмите на кнопку Save & test.

Также в качестве источника нужно установить базу данных, в которой хранятся пользовательские запросы. Для этого во вкладке Data sources нажмите на кнопку Add new data source. В открывшемся окне из списка источников выберите Ред База Данных. Заполните поля для соединения с базой данных и нажмите на кнопку Save & test.

Для хранения логов операционной системы и СУБД также добавьте Loki в качестве источника данных. Во вкладке Data sources нажмите на кнопку Add data source. В открывшемся окне из списка источников выберите Loki. Укажите URL-адрес для доступа к Loki (по умолчанию http://127.0.0.1:3100) и нажмите на кнопку Save & test.

3.2 Сервер СУБД

3.2.1 Агрегатный аудит

Агрегатный аудит собирает метрики, которые агрегируются по событиям и транзакциям. Метриками являются следующие значения запросов: read, write, fetch, mark и время выполнения. Аудит хранит значения метрик во время работы сервера. При выключении/перезагрузке сервера сохранённая статистика будет очищена.

1. Для настройки агрегатного аудита нужно создать конфигурационный файл в корневом каталоге наблюдаемой СУБД, например, aggtrace.conf, и указать в нём следующие параметры:

```
database
{
  format = 3
  max_log_size = 2048
}

database = /путь/к/бд
{
  enabled = true
  format = 3
  max_sql_length = 0
  max_plan_length = 0
}
```

Файл конфигурации состоит из двух секций. В глобальной секции (database) указываются настройки для плагина aggtrace. Указанные параметры распространяющиеся на все базы данных. В ней должен быть указан параметр max_log_size.

В секции database = /путь/к/бд указываются параметры, распространяющиеся на заданную базу данных.

Опция enabled = true включает ведение аудита.

Опция format = aggtrace указывает, что нужно использовать агрегатный аудит.

Опция max_sql_length определяет максимальную допустимую длину SQL-запроса, которая может храниться. Запросы, длина которых больше, будут обрезаны до указанного значения. По умолчанию значение равно 0. Значение 0 указывает, что длина не ограничена.

Опция max_plan_length определяет максимальную допустимую длину плана SQL-запроса, которая может храниться. Планы, длина которых больше, будут обрезаны до указанного значения. По умолчанию значение равно 0. Значение 0 указывает, что длина не ограничена.

Опция max_log_size задает максимальный размер log-файлов в мегабайтах. Допускается значение от 5 до 4096. Значение параметра по умолчанию равно 2048. Параметр должен

быть определён в глобальной секции.

Настройки агрегатного аудита перечитываются для каждого соединения.

3.2.2 Ред База Данных

На `Наблюдаемой СУБД` для сбора метрик пользовательских запросов необходимо в `firebird.conf` настроить параметры `TracePlugin` и `AuditTraceConfigFiles`:

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

После настройки запустите Ред Базу Данных:

```
systemctl start firebird
```

3.2.3 Экспортёр

Настройка экспортёра

Экспортёр должен находиться на том же устройстве, где и наблюдаемая СУБД. Для настройки параметров экспортёра используется файл `exporter_conf.json`, расположенный в `/opt/rdbmonitor/client/exporter/exporter_conf.json`.

Структура файла `exporter_conf.json`:

```
{
 "exporter": {
   "port": <порт экспортёра>,
   "rdb_lock_print": {
     "enabled": true
   },
   "scrape_system":{
     "enabled": false
   },
   "scrape_atrace": {
     "enabled": true,
     "filters": "<путь к файлу с настройками фильтрации>"
   },
   "scrape_mon_tables":{
     "enabled": true
   },
   "user_metrics": {
     "enabled": true,
     "timeout": <таймаут сбора метрики в секундах>,
     "max_metric_labels": <максимальное кол-во метрик>,
     "max_user_metrics": <максимальное кол-во пользовательских метрик>,
     "max_metric_data_blocks": <максимальное кол-во блоков метрики>
   }
 },
 "promtail": {
   "host": "<хост наблюдаемого сервера>"
 },
 },
```

(продолжение на следующей странице)

(продолжение с предыдущей страницы)

```
  "dictserver": {
   "queries_threads": <количество потоков записи для запросов>,
   "atts_threads": <количество потоков записи для соединений>,
   "host": "<хост базы данных>",
   "port": <порт базы данных>,
   "database": "/db/statements.fdb",
   "user": "<пользователь>",
   "password": "<пароль>",
   "Auth_plugins": ["<плагин>", ..., "<плагин>"],
   "loki": {
     "port": <порт loki>
   }
  },
  "RedDatabase": {
    "host": "<хост СУБД>",
    "port": <порт СУБД>,
    "user": "<логин>",
    "password": "<пароль>",
    "folder_path": "<путь к папке установки RedDatabase>",
    "Auth_plugins": ["<плагин>", ..., "<плагин>"]
  },
  "databases": {
    "<псевдоним>": {
      "alias": "<алиас>"
    }
  }
 }
```

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:

- `port` - Порт, на котором работает экспортёр, по умолчанию 3051. 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_system` - Определяет, собирать ли информацио об операционной системе. По умолчанию установлено значение `true`.
- `scrape_mon_tables` - Определяет, собирать ли данные из таблиц мониторинга. По умолчанию установлено значение `true`.
- `scrape_atrace` - Определяет, собирать ли данные с помощью агрегирующего трейса. По умолчанию установлено значение `false`. Агрегатный аудит необходимо предварительно настроить *по инструкции*. Опция `filters` определяет путь к предварительно созданному файлу с настройками фильтрации `filters.ini`. Собираться будут только запросы, характеристики которых превышают указанные значения параметров в `filters.ini`. Например, если `time = 100`, то соберутся только запросы, выполняющиеся дольше 100 миллисекунд.

  Структура `filters.ini`:

```
[limits.statements]
<параметр>
...
<параметр>
```

Параметры фильтрации:

- `time` - время выполнения запроса в миллисекундах;
- `finish` - количество завершённых запросов (событий finish);
- `finish failed` - количество неудачно завершённых запросов;
- `start` - количество запущенных запросов (событий start);
- `prepare` - количество подготовленных запросов (событий prepare);
- `free` - количество освобождённых запросов (событий free);
- `reads` - количество страниц, считанных с диска;
- `marks` - количество отмеченных (mark) страниц;
- `writes` - количество страниц, записанных диск;
- `fetches` - количество страниц, считанных из страничного кэша (fetch);
- `mem use by sort in total` - количество байт, выделенных под сортировку;
- `mem use by sort in cache` - количество байт, выделенных под сортировку в кеше;
- `mem use by sort in disk` - количество байт, выделенных под сортировку на диске.

Значения всех параметров фильтрации должны быть целым положительным числом.

Пример настройки `filters.ini`:

```
[limits.statements]
time = 60000
finish = 100
```

- `user_metrics` - определяет, собирать ли пользовательские метрики. По умолчанию выключено (`"enabled": false`). Опция `timeout` указывает таймаут в секундах, по истечении которого метрика перестанет запрашиваться, значение должно быть дробным числом (по умолчанию значение 0.1). Опция `max_metric_labels` определяет максимальное количество собираемых метрик (по умолчанию значение 5). Опция `max_user_metrics` максимальное количество пользовательских метрик (по умолчанию значение 100). Опция `max_metric_data_blocks` максимальное количество блоков метрики (по умолчанию значение 10).

Секция `promtail`:

- `host` - определяет ip-адрес устройства наблюдаемого сервера, указанное значение должно совпадать со значением в конфигурационном файле Prometheus. параметр имеет строковый тип.

Секция `dictserver`:

- `queries_threads` - количество потоков записи для запросов; значение должно быть целым положительным числом, не рекомендуется указывать число, превышающее количество ядер процессора; по умолчанию значение 2;
- `atts_threads` - количество потоков записи для соединений; значение должно быть целым положительным числом, не рекомендуется указывать число, превышающее количество ядер процессора; по умолчанию значение 1;
- `host` - определяет ip-адрес устройства, на котором работает база даннных для пользовательских запросов; параметр имеет строковый тип;
- `port` - порт, на котором работает база даннных для пользовательских запросов, по умолчанию 3050; параметр имеет целочисленный тип;

- `database` - псевдоним базы данных пользовательских запросов;
- `user` - имя пользователя, от которого экспортёр будет подключаться к базе данных пользовательских запросов; параметр имеет строковый тип;
- `password` - пароль пользователя для подключения к базе данных пользовательских запросов; параметр имеет строковый тип;
- `Auth_plugins` - определяет список плагинов, которые используются для аутентификации при подключении к базе данных пользовательских запросов, в качестве разделителя используется запятая, каждый плагин должен заключаться в двойные кавычки; список заключается в квадратные скобки;
- `loki` - определяет порт, на котором работает `Loki`. По умолчанию 3100.

Секция `RedDatabase`:

- `host`- определяет ip-адрес устройства, на котором работает наблюдаемая СУБД; параметр имеет строковый тип;
- `port` - порт, на котором работает наблюдаемая СУБД; 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>"
 },
 ...
}
```

Где:

- `псевдоним` - уникальный псевдоним базы данных;
- `alias` - псевдоним базы данных, который используется СУБД.

Пример настроенного файла `exporter_conf.json`:

```
{
   "exporter": {
      "port": 3051,
      "rdb_lock_print": {
         "enabled": true
      },
      "scrape_system": {
         "enabled": true
      },
      "scrape_atrace": {
         "enabled": true,
         "filters": "/opt/rdbmonitor/client/exporter/filters.ini"
      },
```

(продолжение на следующей странице)

(продолжение с предыдущей страницы)

```
        "scrape_mon_tables": {
           "enabled": true
        },
        "user_metrics": {
           "enabled": false,
           "timeout": 0.1,
           "max_metric_labels": 5,
           "max_user_metrics": 100,
           "max_metric_data_blocks": 10
        }
    },
    "promtail": {
        "host": "127.0.0.1"
    },
    "dictserver": {
        "queries_threads": 2,
        "atts_threads": 1,
        "host": "127.0.0.1",
        "port": 3050,
        "database": "/db/statements.fdb",
        "user": "SYSDBA",
        "password": "masterkey",
        "Auth_plugins": ["Srp"],
        "loki": {
           "port": 3100
        }
    },
    "RedDatabase": {
        "host": "127.0.0.1",
        "port": 3050,
        "user": "SYSDBA",
        "password": "masterkey",
        "folder_path": "/opt/RedDatabase",
        "Auth_plugins": ["Srp"]
    },
    "databases": {
        "employee_db": {
           "alias": "employee"
        }
    }
}
```

Перед запуском экспортёра убедитесь, что Ред База Данных запущена. В конфигурационном файле экспортёра в секции `scrape_atrace` должно быть установлено значение `"enabled": true`.

`Экспортёр` и `Promtail` запускаются вместе с помощью `systemd`:

```
systemctl start rdbmonitor-client.target
systemctl stop rdbmonitor-client.target
```

3.3 Настройки при установке на один сервер

При установке всех компонентов РБДМонитор на один сервер необходимо указать в конфигурационном файле аудита наблюдаемой СУБД следующие настройки (`fbtrace.conf`):

```
database = <путь или alias к базе данных>
   {
      enabled = true
      format = 3
   }
```

Такие настройки нужно произвести для каждой базы данных, которая указана в секции `databases` файла конфигурации экспортёра.

3.3.1 Запуск РБДМонитор при установке на один сервер

Запустите все компоненты РБДМонитор, выполнив следующее:

1. Запустите Ред Базу Данных:

   ```
   systemctl start firebird
   ```

2. Запустите `Loki`:

   ```
   systemctl start loki
   ```

3. Запустите `Prometheus`:

   ```
   systemctl start prometheus
   ```

4. Запустите `Grafana`:

   ```
   systemctl start grafana-server
   ```

5. Запустите `Экспортёр` и `Promtail`:

   ```
   systemctl start rdbmonitor-client.tartget
   ```

Глава 4

Просмотр метрик

Собираемые метрики описаны в приложении *Собираемые метрики*.

4.1 Пользовательские метрики

Для добавления пользовательской метрики необходимо создать описывающий её файл с расширением `.sh` по пути `/opt/rdbmonitor/exporter/UserMetrics`.

Структура описания метрики:

```
#!/bin/bash

# Название метрики
echo "имя_метрики"

echo "#" # Разделительный символ

# Опциональный список меток
echo "имя_метки" # Название метки 1
echo "имя_метки" # Название метки 2

echo "#" # Разделительный символ

# Опциональный список характеристик меток
# Характеристика метки 1
echo "" # Состояние метки
echo "lable_var1" # Название метки 1
echo "lable_var2" # Название метки 2

echo "#" # Разделительный символ

# Характеристика метки 2
...
# Характеристика метки n
```

Пример описания метки:

```
#!/bin/bash

# Чтение значений памяти (в килобайтах)
read TOTAL_MEMORY FREE_MEMORY <<< $(awk '/MemTotal/ {t=$2} /MemFree/ {f=$2} END {print
t, f}' /proc/meminfo)

# Расчет использованной памяти
USED_MEMORY=$((TOTAL_MEMORY - FREE_MEMORY))

# Блок с названием метрики
echo "meminfo"
```

(продолжение на следующей странице)

(продолжение с предыдущей страницы)

```
# Разделительный символ
echo "#"

# Блок с названиями меток
echo "type"  # Название метки

# Разделительный символ
echo "#"

# Блок со значением метрики (total)
echo "$TOTAL_MEMORY"  # Вывод значения метрики
echo "total"          # Значение метки

# Разделительный символ
echo "#"

# Блок со значением метрики (free)
echo "$FREE_MEMORY"   # Вывод значения метрики
echo "free"           # Значение метки

# Разделительный символ
echo "#"

# Блок со значением метрики (used)
echo "$USED_MEMORY"   # Вывод значения метрики
echo "used"           # Значение метки
```

Далее нужно включить отображение пользовательских метрик в `exporter_conf.json`, настроив секцию `"user_metrics"`:

```
"user_metrics": {
  "enabled": true,
  "timeout": <таймаут сбора метрики в секундах>,
  "max_metric_labels": <максимальное кол-во метрик>,
  "max_user_metrics": <максимальное кол-во пользовательских метрик>,
  "max_metric_data_blocks": <максимальное кол-во блоков метрики>
}
```

Пользовательские метрики просматриваются только с помощью `Prometheus`. Для просмотра пользовательских метрик в `Grafana` нужно самостоятельно создать панель.

4.2 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"
```

По умолчанию для просмотра метрик используется нулевой часовой пояс, чтобы использовать локальное время укажите флаг `Use local time`:

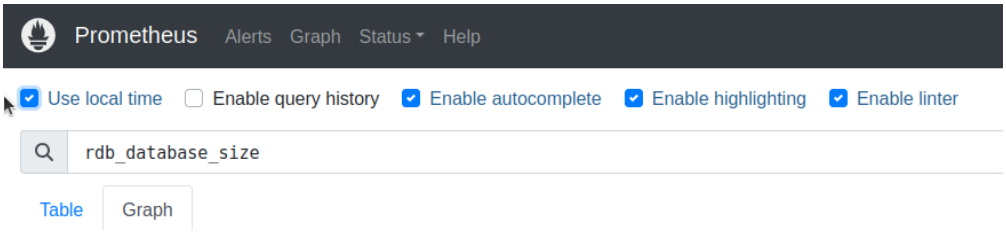

Рисунок 4.1 — Использование локального времени

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="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.

4.3 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 РБДМонитор:

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

После этого в меню Apps будет добавлен плагин РБДМонитор.

После добавления источников (подробнее см. *Grafana*) необходимо указать их в настройках плагина РБДМонитор:

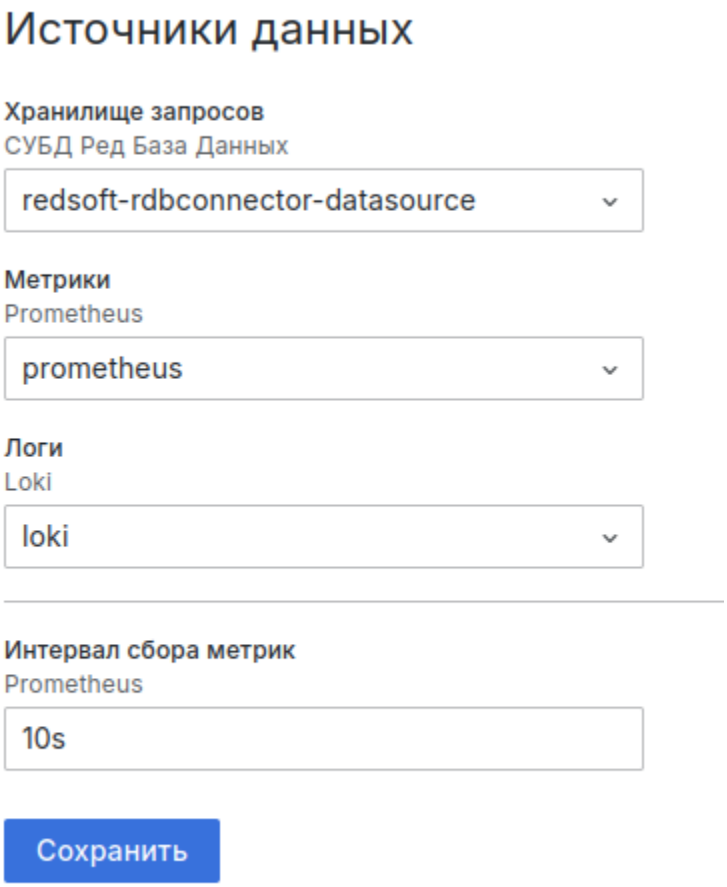

Рисунок 4.2 — Настройка плагина

После нажатия на `РБДМонитор` в списке приложений (`Apps`) откроется `Обзорная страница`.

4.4 Обзорная страница

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

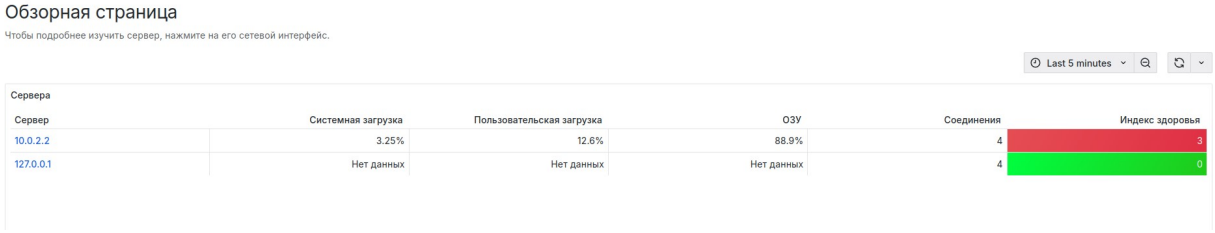

Рисунок 4.3 — Обзорная страница

4.5 Информация о выбранном сервере

На этой странице находится подробная информация о выбранном сервере.

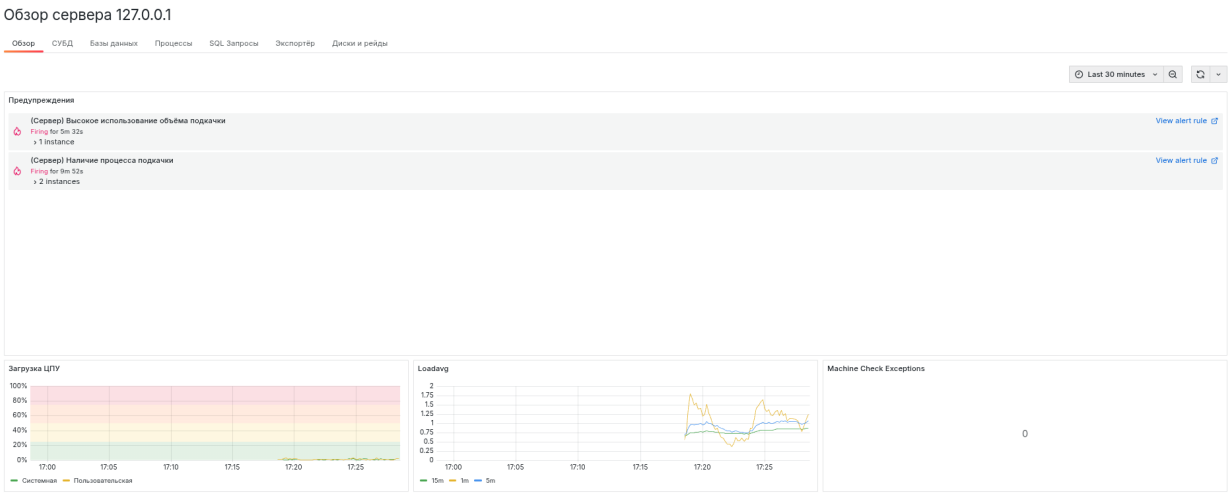

Рисунок 4.4 — Обзор выбранного сервера - Часть 1

Рисунок 4.5 — Обзор выбранного сервера - Часть 2

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

График `Использование памяти` показывает использование оперативной памяти в зависимости от времени.

Графиг `Loadavg` отображает среднее количество готовых к выполнению процессов в системе за определённое время: 1 минута, 5 минут и 15 минут.

График `Насыщение памяти` показывает использование `SWAP` в зависимости от времени.

Панель `MCC и Machine Check Exceptions` отображает количество ошибок в логе системы с сообщением, содержащим `MCC` или `Machine Check Exceptions`.

Панель `Error-correction code` отображает количество ошибок в логе системы с сообщением, содержащим `ECC` или `Error-correction code`.

График `Использование дисков` показывает какой процент времени занимают операции чтения с дисков и записи на них в зависимости от времени.

График `Насыщение дисков` показывает среднюю длину очереди для операций чтения с дисков и записи на них в зависимости от времени.

График `Использование пространства дисков` отображает среднюю заполненность файловых систем, находящихся на диске, в зависимости от времени.

График `Использование сети` показывает скорость отправки/получения пакетов (количество пакетов в секунду) в зависимости от времени.

График `Насыщение сети` показывает скорость потери пакетов (количество пакетов в секунду) в зависимости от времени.

График `Ошибки сети` показывает скорость возникновения ошибок (количество ошибок в секунду) отправки/получения пакетов в зависимости от времени.

4.5.1 СУБД

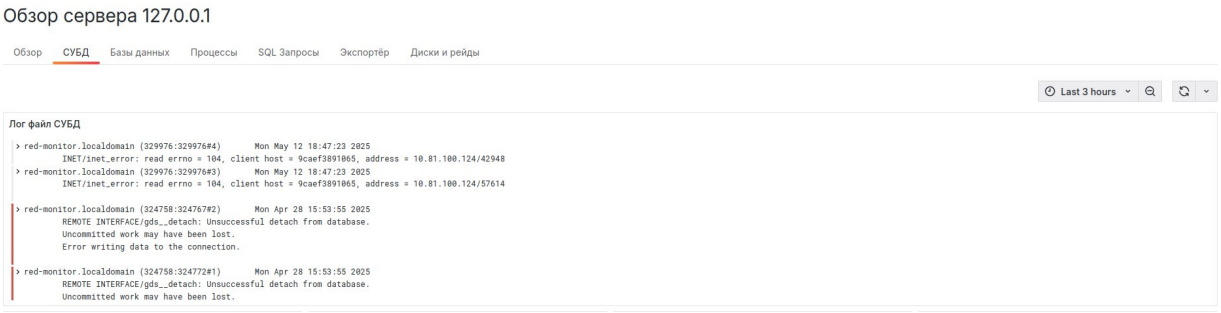

Рисунок 4.6 — Лог СУБД

Панель `Лог файл СУБД` показывает содержание файла `firebird.log`.

Рисунок 4.7 — Общая информация о СУБД

График **Скорость подготовки запросов** показывает скорость возникновения событий `PREPARE_STATEMENT` (количество событий в секунду) в зависимости от времени.

График **Время обработки запросов** отображает общее время выполнения запросов.

График **Средняя скорость возникновения ошибок выполнения запросов** показывает скорость возникновения ошибок при выполнении запроса.

График **Средняя скорость завершения транзакций** показывает скорость возникновения событий, связанных с завершением транзакции (`COMMIT`, `ROLLBACK`, `COMMIT_RETAINING`, `ROLLBACK_RETAINING`), в зависимости от времени.

График **Время обработки событий завершения транзакций** отображает время, потраченное на выполнение событий, связанных с завершением транзакции.

График **Средняя скорость возникновения ошибок выполнения транзакций** показывает скорость возникновения ошибок при завершении транзакции.

4.5.2 Базы данных

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

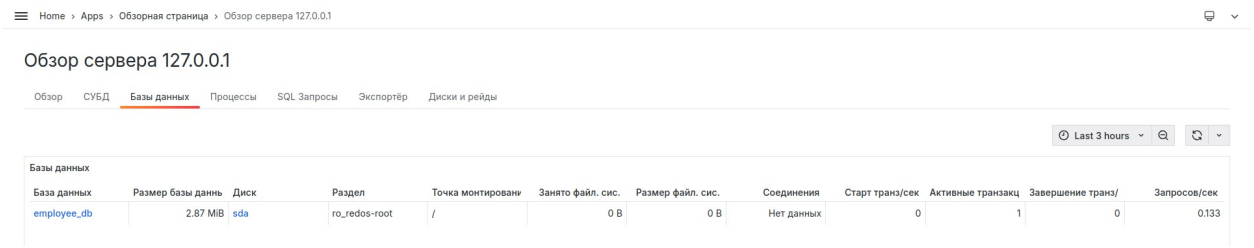

Рисунок 4.8 — Обзор баз данных

4.5.3 Информация о выбранной базе данных

На этой странице находится подробная информация о базе данных.

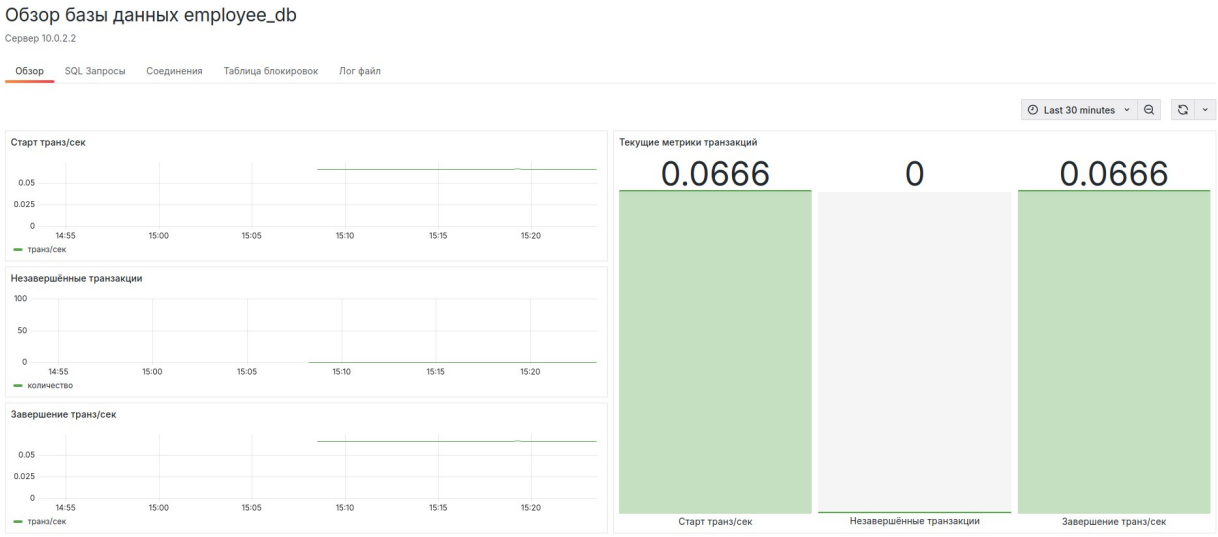

Рисунок 4.9 — Обзор выбранной базы даннных - Часть 1

График `Частота стартов транзакций` показывает количество транзакций, запущенных за секунду, в зависимости от времени.

График `Незавершённые транзакции` показывает количество незавершённых транзакций (активных и бездействующих).

График `Частота завершений транзакций` показывает скорость возникновения транзакций завершения в зависимости от времени.

Рисунок 4.10 — Обзор выбранной базы даннных - Часть 2

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 `Запросов/сек` показывает скорость возникновения запросов (количество запросов в секунду) в зависимости от времени.

График `Маркеры транзакций` отражает значения маркеров транзакций.

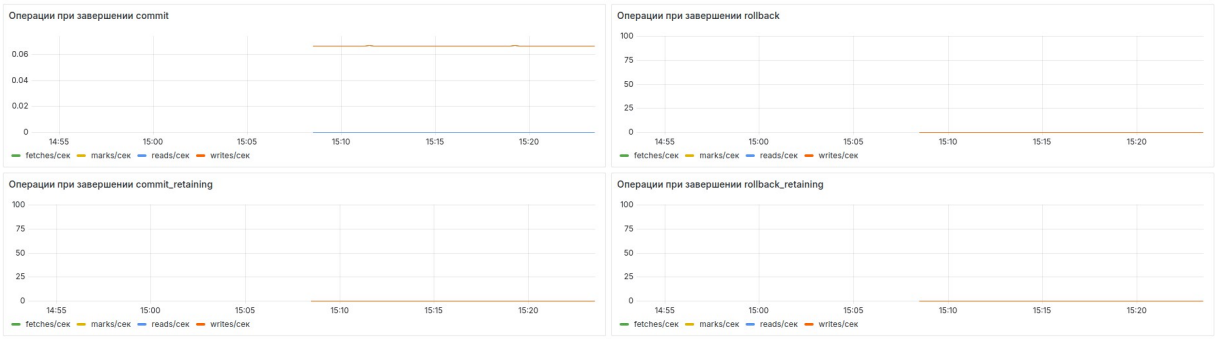

Рисунок 4.11 — Обзор выбранной базы даннных - Часть 3

График `Операции при завершении commit` показывают количество операций `reads`, `fetches`, `marks`, `writes`, выполнившихся за секунду, при завершении транзакции `commit`.

График `Операции при завершении rollback` показывают количество операций `reads`, `fetches`, `marks`, `writes`, выполнившихся за секунду, при завершении транзакции `rollback`.

График `Операции при завершении commit_retaining` показывают количество операций `reads`, `fetches`, `marks`, `writes`, выполнившихся за секунду, при завершении транзакции `commit_retaining`.

График `Операции при завершении rollback_retaining` показывают количество операций `reads`, `fetches`, `marks`, `writes`, выполнившихся за секунду, при завершении транзакции `rollback_retaining`.

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.

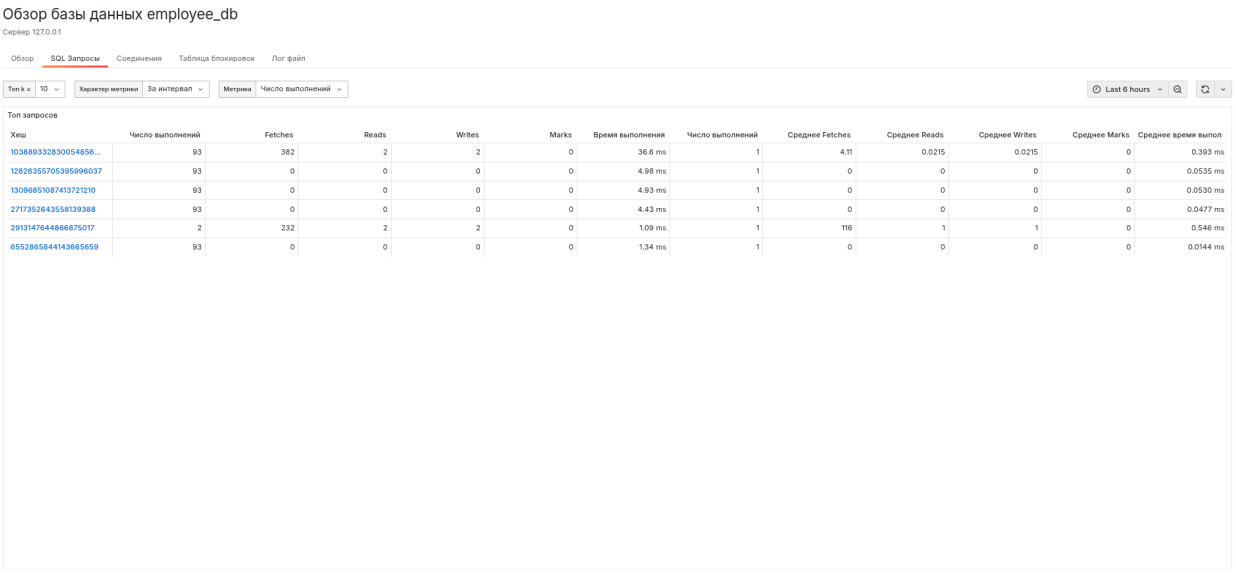

Рисунок 4.12 — SQL запросы выбранной базы данных

Обзор запроса

На данной странице находится подробная информация о выбранном запросе.

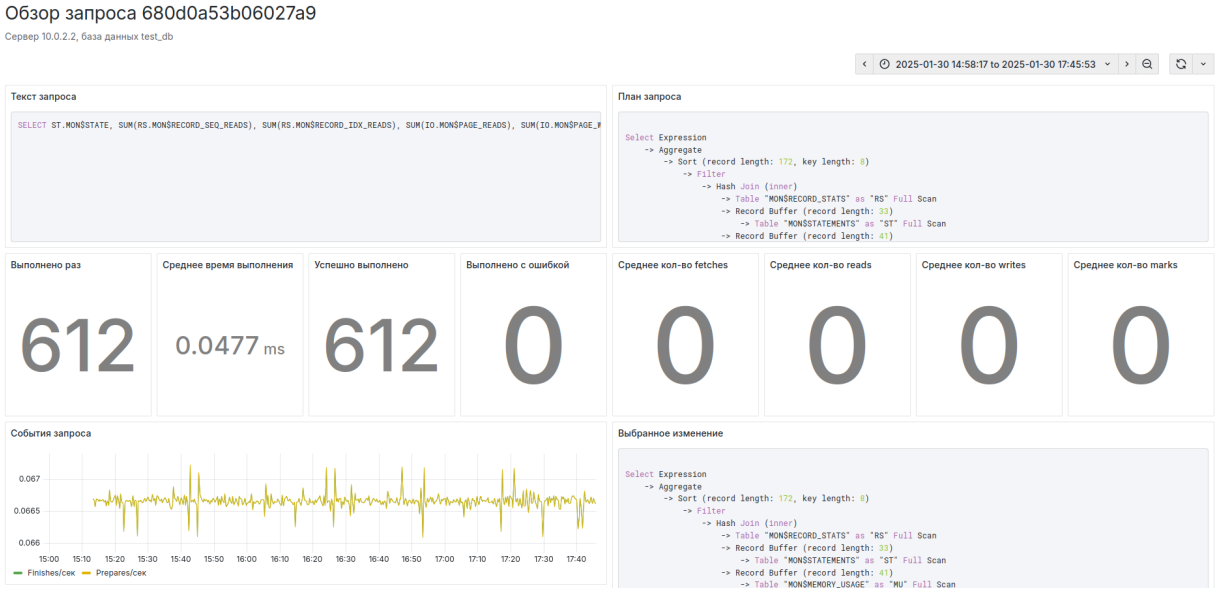

Рисунок 4.13 — Обзор запроса - Часть 1

Страница содержит текст запроса, план запроса и его изменения, информацию о количестве и времени выполнений.

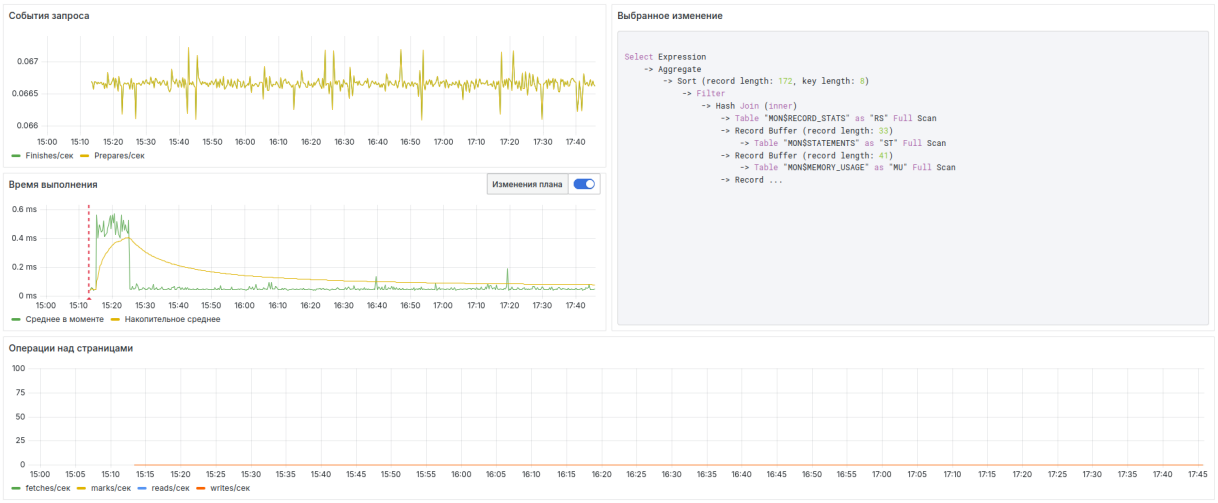

Рисунок 4.14 — Обзор запроса - Часть 2

График `События запроса` показывает частоту возникновения событий подготовки и завершения запроса за выбранный промежуток времени.

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.

График `Время выполнения` показывает время выполнения запроса. `Среднее в моменте` - среднее время выполнения по двум последним сборам метрик. `Накопительное среднее` - среднее время выполнения за указанный промежуток времени. `Изменение плана` включает/отключает отображение моментов времени, когда план запроса изменялся.

Соединения

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

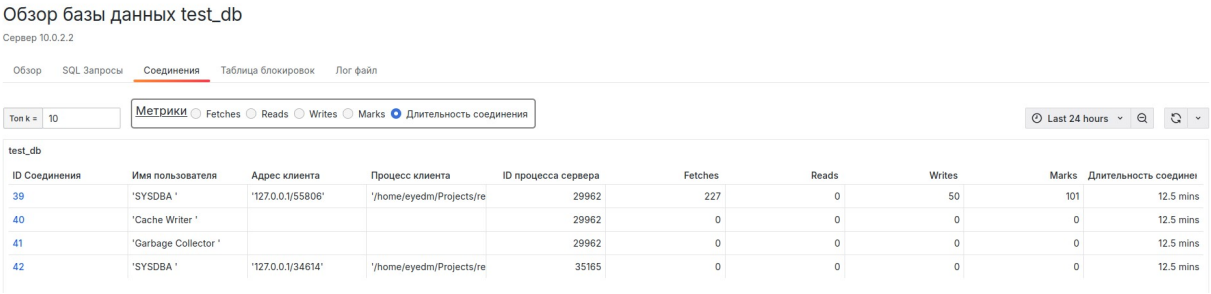

Рисунок 4.15 — Соединения с базой данных

Обзор соединения

На данной странице находится подробная информация о выбранном соединении.

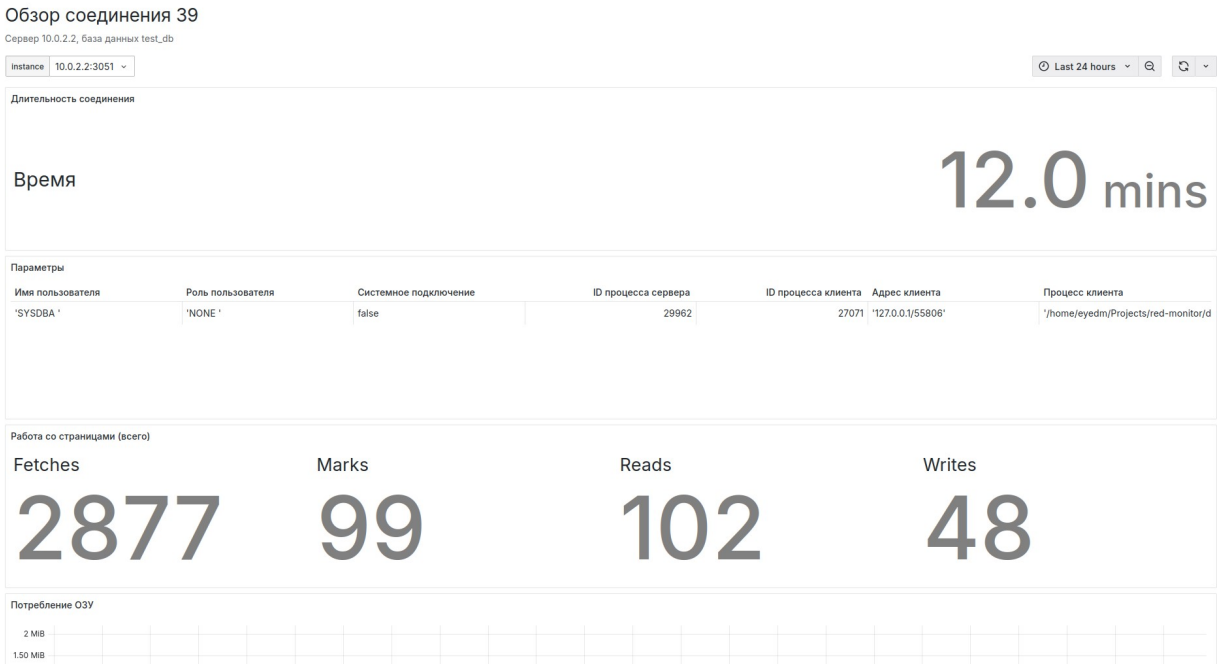

Рисунок 4.16 — Обзор соединения

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

Таблица блокировок

Содержание данной вкладки соответствует выводу утилиты `rdb_lock_print` в интерактивном режиме. Также показано использование памяти таблицей блокировок.

Рисунок 4.17 — Использование памяти таблицей блокировок

Лог файл

Вкладка содержит результат анализа файла `firebird.log`. Учитываются только события, относящиеся к выбранной базе данных.

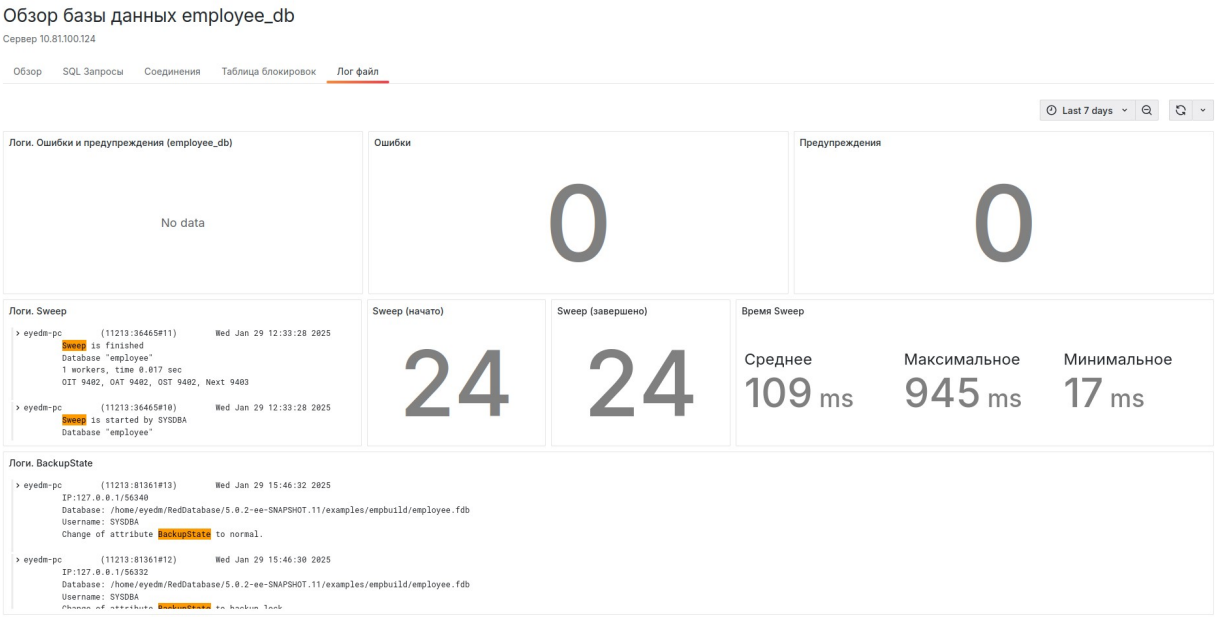

Рисунок 4.18 — Анализ лог файла

4.5.4 Процессы

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

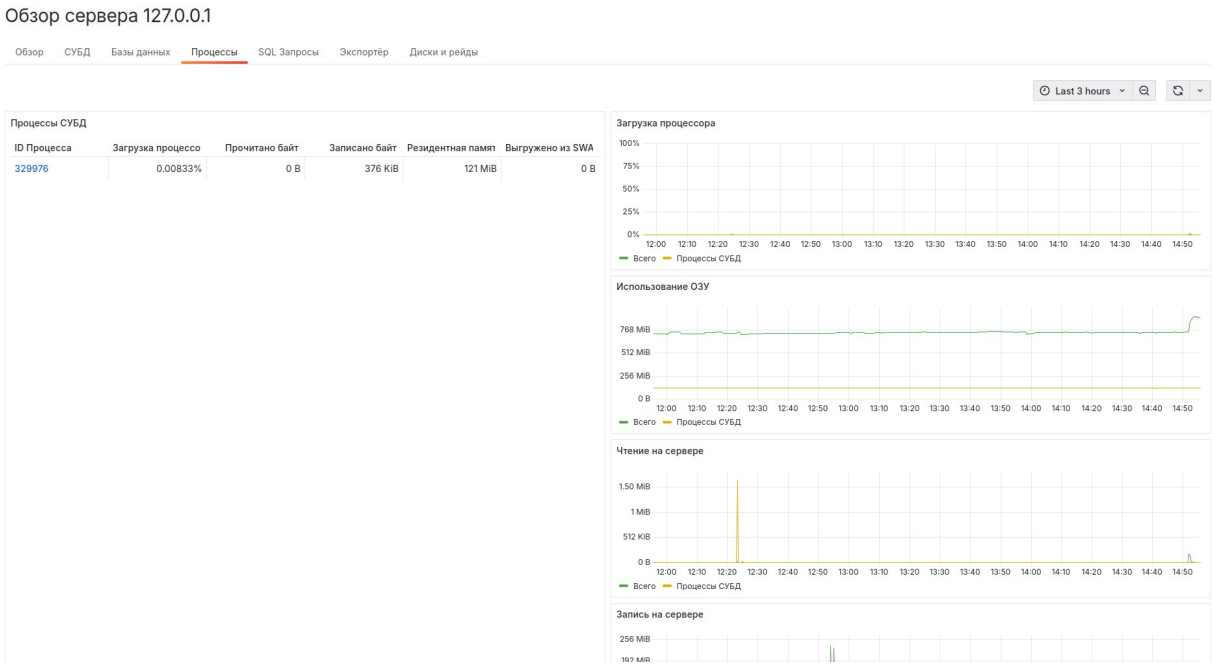

Рисунок 4.19 — Процессы СУБД

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

Обзор процесса

На данной странице находится подробная информация о выбранном процессе.

Рисунок 4.20 — Обзор процесса

График `Загрузка процессора` показывает нагрузку на процессор всеми процессами в системе и выбранным процессом СУБД.

График `Использование ОЗУ` показывает использование оперативной памяти всеми процессами в системе и выбранным процессом СУБД.

График `Потоки процесса` показывает количество потоков, открытых для выполнения выбранного процесса.

График `Использование процессора по типу нагрузки` показывает системную и пользовательскую нагрузку при выполнении выбранного процесса.

4.5.5 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*.

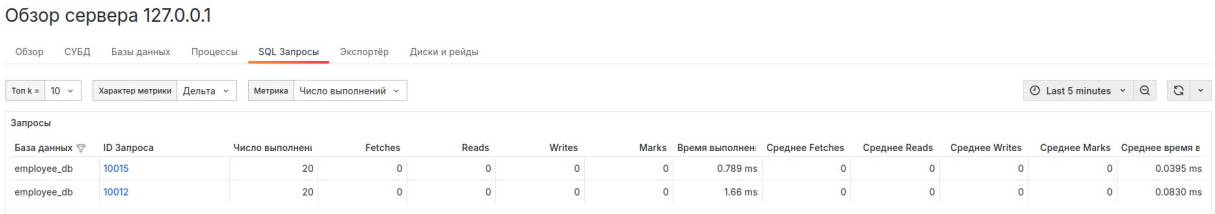

Рисунок 4.21 — Запросы

4.5.6 Экспортёр

На странице находится информация о сборе метрик экспортёром.

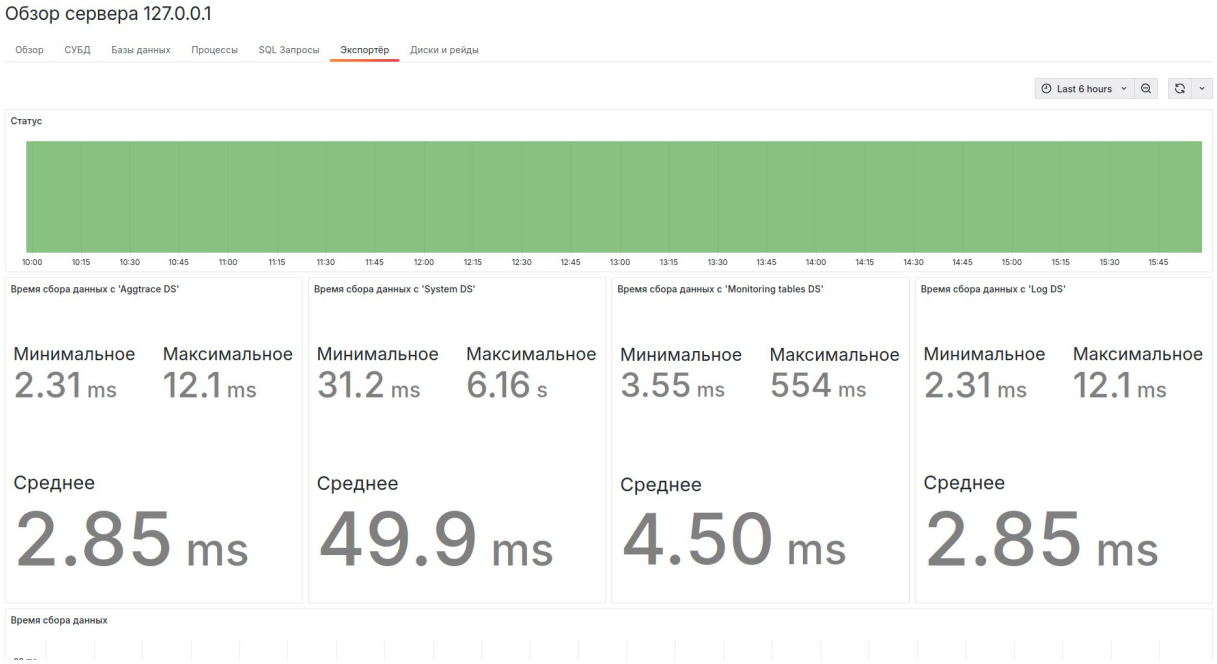

Рисунок 4.22 — Экспортёр

Панель статус показывает интервалы работы экспортёра: зелёным цветом показано время, когда экспортёр работает и собирает метрики, а красным, когда он не запущен.

На других панелях находится информация о продолжительности сбора метрик с каждого источника данных.

4.5.7 Диски и рейды

На странице отображается информация о выбранных дисках.

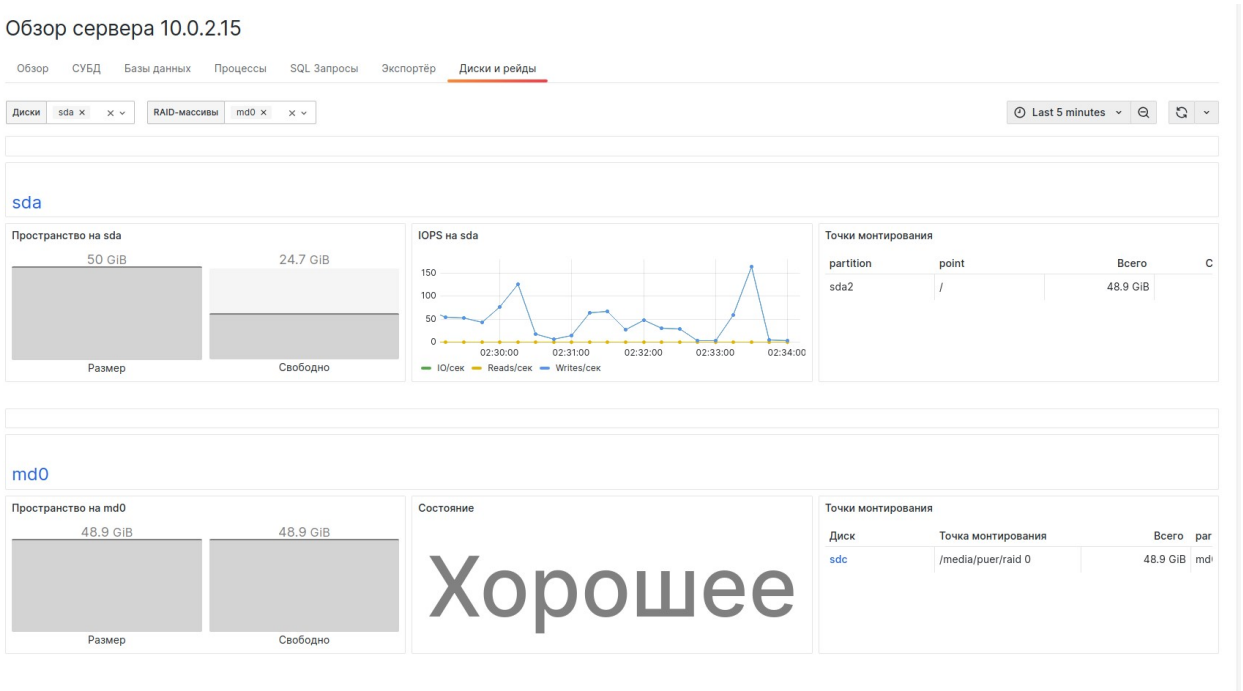

Рисунок 4.23 — Информация о диске

Панель `Пространство` показывает общий и свободный объём памяти на диске и RAID-массиве.

График `IOPS` отображает количество операций ввода/вывода в секунду.

График `Состояние` показывает состояние RAID-массива:

- Хорошее (`clean`)
- Повреждён (`degraded`)
- Синхронизация (`resyncing`)
- Восстановление (`recovering`)
- Неисправен (`faulty`)

Обзор RAID-массива

На странице отображается подробная информация о выбранном RAID-массиве.

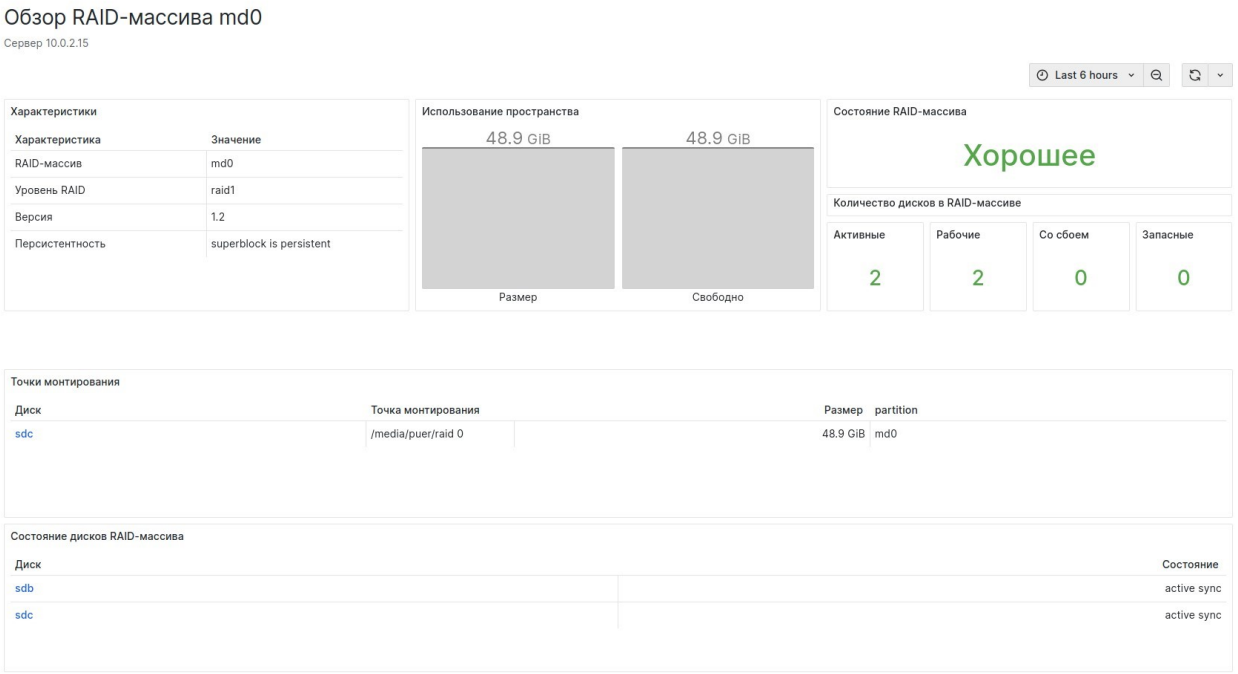

Рисунок 4.24 — Информация о RAID-массиве

Панель `Количество дисков в RAID-массиве` показывает число дисков, находящихся в конкретном состоянии.

Панель `Состояние дисков RAID-массива` показывает статус каждого диска:

- `unknown` (неизвестное состояние диска)
- `absent` (диск отсутствует)
- `active` (диск активен)
- `sync` (диск синхронизирован)
- `spare` (запасной диск)
- `write-mostly` (диск используется преимущественно для записи)
- `blocked` (диск заблокирован)
- `faulty` (диск неисправен)

Диск может находиться сразу в нескольких состояниях, например `"active sync"`.

4.6 Мониторинг кластеров

4.6.1 Необходимые настройки

Необходимо установить `Сервер СУБД` на каждый узел кластера *по инструкции*.

В конфигурационном файле `prometheus.yml` в параметре `targets` нужно указать сокет каждого узла. Подробнее о настройке `Prometheus` см. *Основные настройки Prometheus*.

4.6.2 Обзор кластеров

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

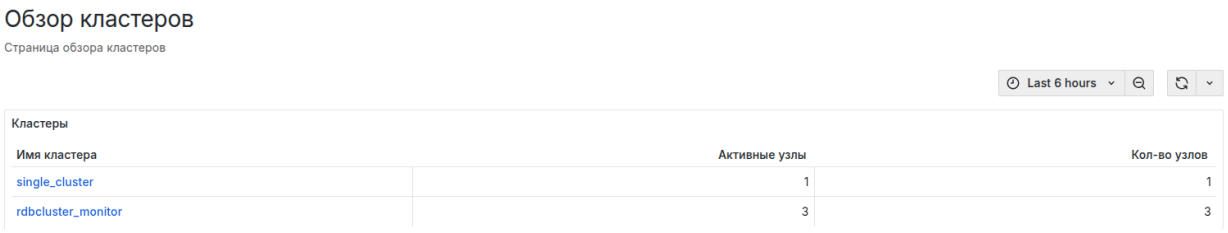

Рисунок 4.25 — Обзор кластеров

4.6.3 Информация о выбранном кластере

Обзор

На данной странице представлена краткая информация о выбранном кластере. Подробнее о режиме обслуживания и настройках отказоустойчивого кластера см. в Документации по настройке отказоустойчивого кластера.

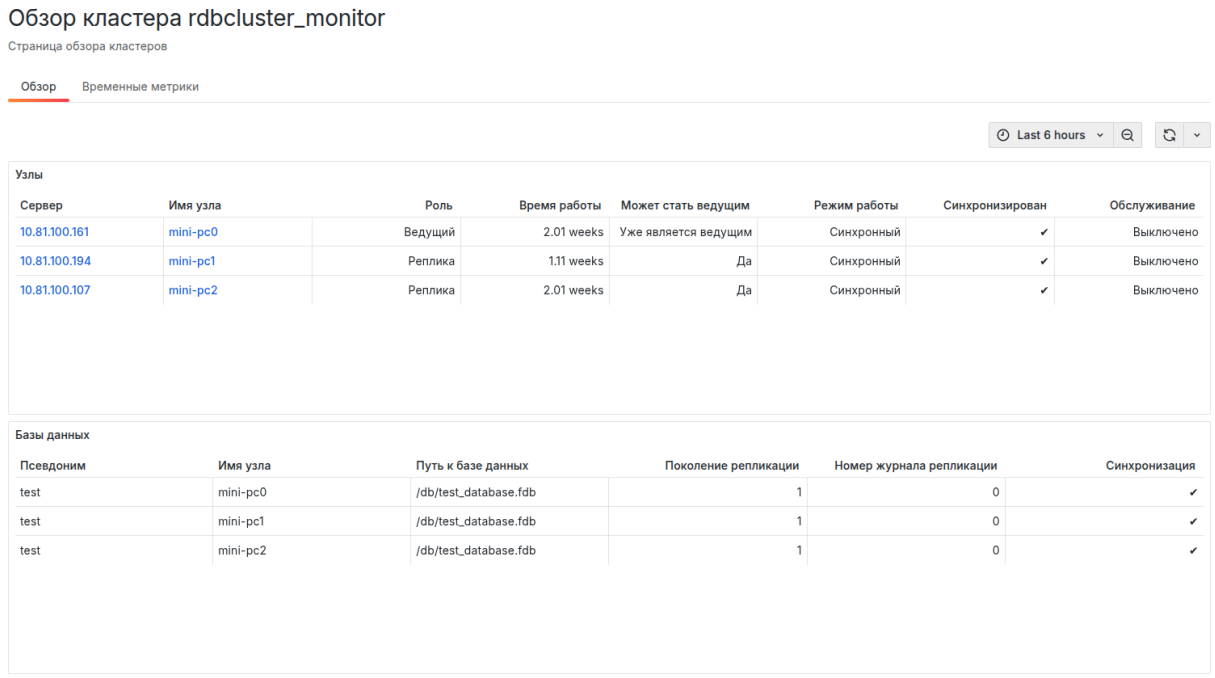

Рисунок 4.26 — Информация о выбранном кластере

Временные метки

Временные метки отображают время, затраченное на опрос состояния узла и кластера и публикацию собранной информации.

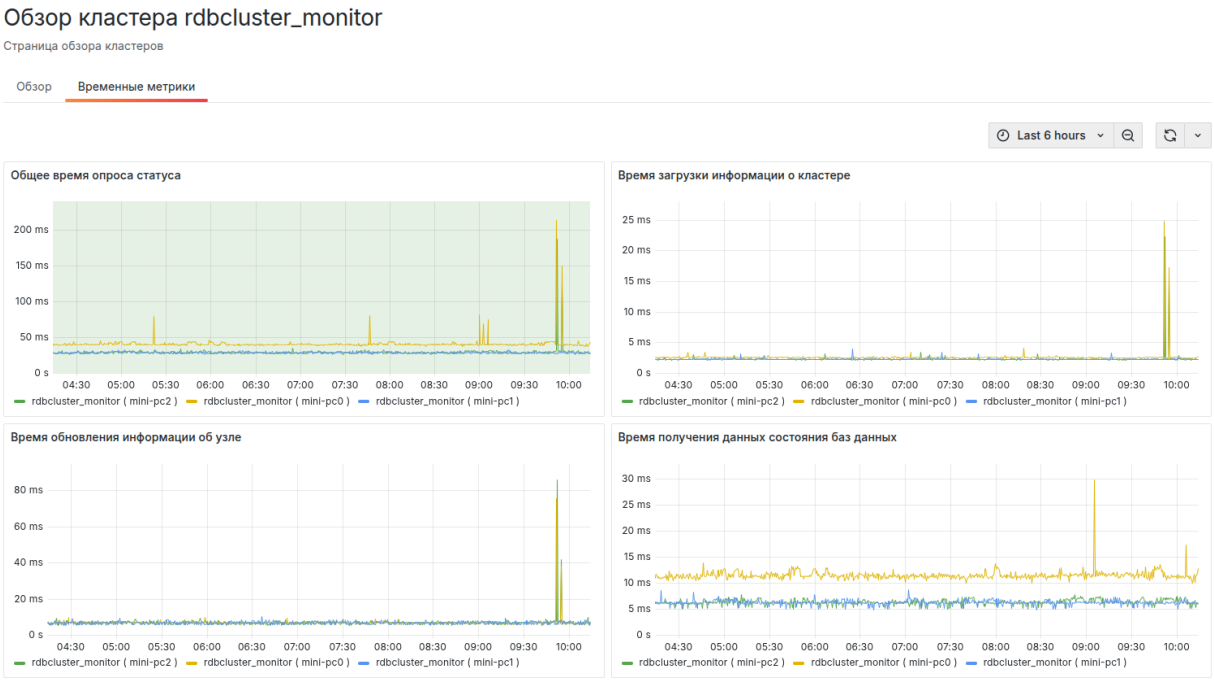

Рисунок 4.27 — Временные метки

4.6.4 Информация об узле кластера

На этой странице отображается информация о состоянии и временных метках конкретного узла кластера.

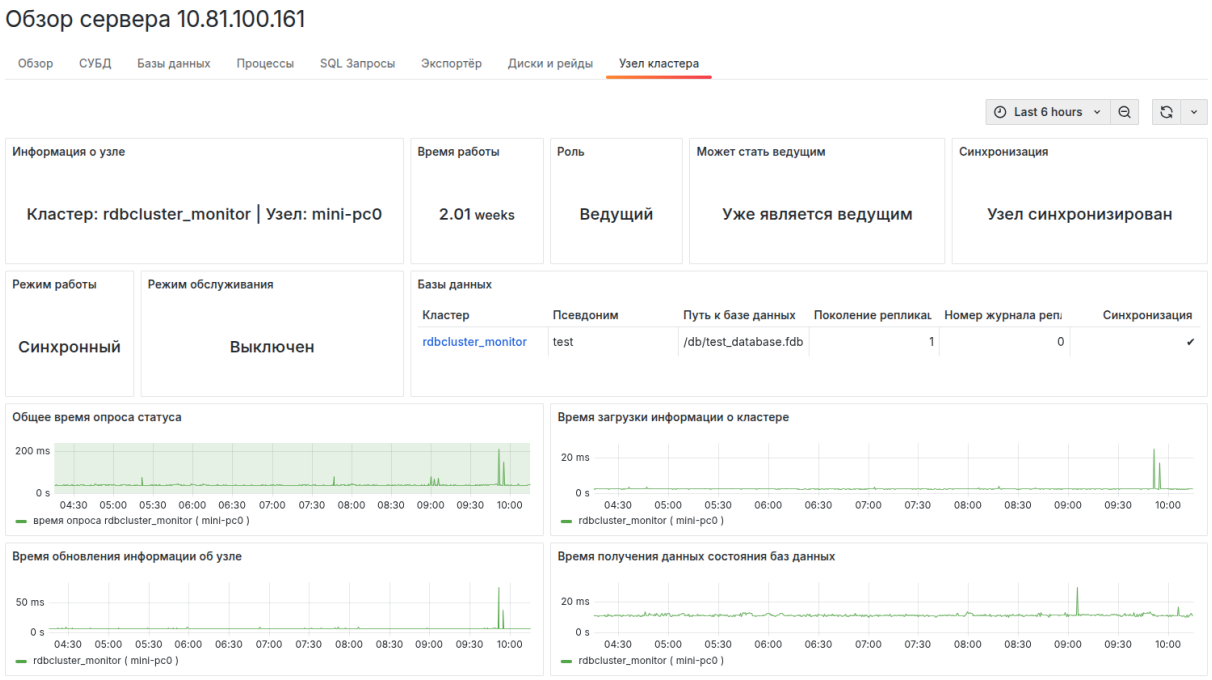

Рисунок 4.28 — Информация об узле кластера

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 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.	Количесво 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
rdb_database_sweep _interval	database - база данных.	Интервал автоматической сборки мусора;
rdb_database_read_only	database - база данных.	Является ли база данных доступной только для чтения. Read-only - значение 1, read-write значение 0.
rdb_database_forced _writes	database - база данных.	Указывает, установлен для базы режим синхронного вывода (forced writes, значение 1) или режим асинхронного вывода (значение 0).
rdb_database_backup _state	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.
rdb_database_pages	database - база данных.	Количество страниц, выделенных для базы данных на внешнем устройстве.
rdb_database_shutdown _mode	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).
rdb_database_size	database - база данных.	Объём памяти в байтах, занимаемый базой данных на диске. Это произведение количества выделенных страниц и размера одной страницы базы данных.
rdb_database_cache	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.

А.2 Метрики агрегатного аудита

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

Таблица А.2 — Метрики агрегатного аудита

Название метрики	Метки	Описание метрики
rdb_aggtrace_stmt_finish_count	database - база данных; hash - хэш запроса;	Количество завершённых запросов (событий FINISH STATEMENT).
rdb_aggtrace_stmt_finish_count_failed	database - база данных; hash - хэш запроса;	Количество неудачно завершённых запросов.
rdb_aggtrace_stmt_start_count	database - база данных; hash - хэш запроса;	Количество запусков запросов (событий START STATEMENT).
rdb_aggtrace_stmt_free_count	database - база данных; hash - хэш запроса;	Количество освобождений запросов (событий FREE STATEMENT).
rdb_aggtrace_stmt_prepare_count	database - база данных; hash - хэш запроса;	Количество подготовки запросов (событий PREPARE STATEMENT).
rdb_aggtrace_stmt_times	database - база данных; hash - хэш запроса;	Время выполнения запроса в миллисекундах.
rdb_aggtrace_stmt_pages_reads	database - база данных; hash - хэш запроса;	Количество страниц, считанных с диска.
rdb_aggtrace_stmt_pages_writes	database - база данных; hash - хэш запроса;	Количество страниц, записанных из кэша на диск.
rdb_aggtrace_stmt_pages_fetches	database - база данных; hash - хэш запроса;	Количество страниц, считанных из страничного кэша.
rdb_aggtrace_stmt_sort_total	database - база данных; hash - хэш запроса;	Количество памяти, выделенной под сортировку запроса.
rdb_aggtrace_stmt_sort_cache	database - база данных; hash - хэш запроса;	Количество памяти, выделенной в кеше под сортировку запроса.
rdb_aggtrace_stmt_sort_disk	database - база данных; hash - хэш запроса;	Количество памяти, выделенной на диске под сортировку запроса.

А.3 Метрики утилиты rdb_lock_print

Таблица А.3 — Метрики утилиты rdb_lock_print

Название метрики	Метки	Описание метрики
rdb_lck_table_length	database - база данных	Общий объем памяти, выделенный таблице блокировок (в байтах)
rdb_lck_table_used	database - база данных	Наибольшая величина смещения в таблице блокировок, которая используется в настоящий момент
rdb_lck_table_hash_slots	database - база данных	Число слотов кэширования блокировок
rdb_lck_table_scan_interval	database - база данных	Время (в секундах), которое ожидает менеджер блокировок до того как запустить к поиску взаимных блокировок
rdb_lck_table_acquire_spins	database - база данных	Режим ожидания взаимной блокировки, когда повторяется запрос к таблице блокировок
rdb_lck_table_acquires	database - база данных	Сколько раз владелец запрашивает исключительное управление таблицей блокировок, чтобы выполнить изменения
rdb_lck_table_acquire_blocks	database - база данных	Сколько раз владелец находился в состоянии ожидания при запросе исключительного управления таблицей блокировок.
rdb_lck_table_acquire_retries	database - база данных	
rdb_lck_table_retry_success	database - база данных	
rdb_lck_table_enqs	database - база данных	Число запросов, полученных на блокировку (не включает запросы, которые пришли и ушли)
rdb_lck_table_converts	database - база данных	Запросы на повышение уровня блокировки
rdb_lck_table_downgrades	database - база данных	
rdb_lck_table_deqs	database - база данных	
rdb_lck_read_data	database - база данных	
rdb_lck_table_write_data	database - база данных	
rdb_lck_table_query_data	database - база данных	

(разрыв таблицы)

(разрыв таблицы)

Название метрики	Метки	Описание метрики
rdb_lck_table_waits	database - база данных	
rdb_lck_table_denies	database - база данных	
rdb_lck_table_timeouts	database - база данных	
rdb_lck_table_blocks	database - база данных	Запросы, которые не могут быть удовлетворены немедленно
rdb_lck_table_wakeups	database - база данных	
rdb_lck_table_scans	database - база данных	Показывает число просмотров менеджером блокировок цепочки блокировок и владельцев для поиска взаимных блокировок
rdb_lck_table_deadlocks	database - база данных	Число найденных взаимных блокировок
rdb_lck_table_lops	database - база данных lop - операция над блокировкой	

А.4 Метрики операционной системы

Таблица А.4 — Метрики операционной системы

Название метрики	Метки	Описание метрики
rdb_disks_io_counters	disk - диск. operation - тип операции: • reads • writes	Количество операций чтения и записи на диск.
rdb_disks_io_bytes	disk - диск. operation - тип операции: • reads • writes	Объём прочитанной/записанной на диск информации в байтах.
rdb_disks_io_wtime	database - база данных. disk - диск.	Взвешенное время, потраченное на операции ввода-вывода.
rdb_disks_busy_time	disk - диск. kind - тип времени, потраченного на ввод/вывод: • normal - время, затраченное на фактический ввод/выводж; • weighted - взвешенное время, затраченное на фактический ввод/вывод.	Время, потраченное на фактический ввод-вывод.
rdb_disk_size	disk - диск.	Объём диска

(разрыв таблицы)

(разрыв таблицы)

Название метрики	Метки	Описание метрики
rdb_disks_info	disk - диск. vendor - поставщик. revision - ревизия. model - модель. serial - серийный номер. scheduler - планировщик диска.	Общая информация о диске.
rdb_mount_point_usage	disk - диск. partition - раздел диска. point - точка монтирования. usage: • used • free • total	Использование точки монитрования.
rdb_link_mp_db	disk - диск partition - раздел диска point - точка монтирования database - база данных.	Устанавливает связь с другими метриками.
rdb_procs_open_files_size	pid - идентификатор процесса СУБД kind - тип временного объекта: • table • blob • undo • redbuf • merge • sort • tpc • snap	Размер временных файлов, созданных процессами СУБД (в байтах).
rdb_procs_open_files_count	pid - идентификатор процесса СУБД kind - тип временного объекта: • table • blob • undo • redbuf • merge • sort • tpc • snap	Количество временных файлов, созданных процессами СУБД.
rdb_procs_cpu_times	pid- идентификатор процесса СУБД kind - тип времени, потраченного процессором: • system • user • iowait	Время, потраченное процессором на выполнение процесса.

(разрыв таблицы)

(разрыв таблицы)

Название метрики	Метки	Описание метрики
rdb_procs_memory_usage	pid - идентификатор процесса СУБД usage: • resident - резидентная память • virtual - виртуальная память • swapped_out - занимаемая память в пространстве swap • shared - разделяемая память: text, lib, data, dirty, uss, pss.	Объём резидентной, виртуальной и выгруженной в swap памяти в байтах.
rdb_procs_io_counters	pid - идентификатор процесса СУБД operation - тип операции: • reads • writes	Количество операций ввода/вывода, выполняемых процессом СУБД.
rdb_procs_io_bytes	pid - идентификатор процесса СУБД operation - тип операции: • reads • writes	Размер прочитанной и записанной информации в байтах процессами СУБД.
rdb_procs_threads_count	pid - идентификатор процесса СУБД	Количество открытых потоков для процесса
rdb_memory_usage	field: • free • used • total	Использование оперативной памяти.
rdb_swap_usage	usage: • used • free • total	Использование памяти, выделенной для swap.
rdb_swapped	direction: • in • out	Объём данных (в байтах), затронутый swap.
rdb_cpu_freq	marker: • current	Частота работы процессора.
rdb_cpu_times	kind: • system • user • iowait • irq • soft_irq	Время, потраченное процессором.

(разрыв таблицы)

(разрыв таблицы)

Название метрики	Метки	Описание метрики
rdb_cpu_ctx		Число изменений контекста процессора.
rdb_cpu_cores	kind: • logical • physical	Количество ядер процессора.
rdb_cpu_interrupts	marker: • soft • casual	Число прерываний процессора.
rdb_tmp_mount_point_usage	point usage: • total • used • free	Использование точки монтирования.
rdb_sys_loadavg	period: 1 минута, 5 минут, 15 минут.	Количество готовых к выполнению процессов.
rdb_net_io_bytes	interface - сетевой интерфейс direction - направление трафика (in, out)	Количество байт, переданных по сети.
rdb_net_io_packets	interface - сетевой интерфейс direction - направление трафика (in, out)	Количество пакетов, переданных по сети.
rdb_net_io_errs	interface - сетевой интерфейс direction - направление трафика (in, out)	Количество ошибок, возникших при передаче по сети.
rdb_net_io_drops	interface - сетевой интерфейс direction - направление трафика (in, out)	Количество потерянных пакетов.

A.5 Метрики RAID-массивов

Таблица А.5 — Метрики RAID-массивов

Название метрики	**Метки**	**Описание метрики**
`rdb_raid_disks_count`	`raid` - имя массива; `state` - состояние диска: • `active` - количество активных дисков • `working` - количество рабочих дисков • `failed` - количество неисправных дисков • `spare` - количество запасных дисков	Количество дисков, находящихся в конкретном состоянии.
`rdb_raid_state`	`raid` - имя массива.	Текущее состояние RAID-массива: • 1 - `clean` • 2 - `degraded` • 3 - `resyncing` • 4 - `recovering` • 5 - `faulty`
`rdb_raid_info`	raid - имя RAID-массива `raid_level` - уровень `RAID`; `version` - версия `RAID`; `consistency_policy` - политика согласованности; `persistence` - информация о персистентности.	Общая информация о RAID-массиве.
`rdb_link_raid_disk`	`raid` - имя массива; `disk` - диск	Сопоставление RAID-массива и диска.

(разрыв таблицы)

(разрыв таблицы)

Название метрики	Метки	Описание метрики
rdb_raid_device_state	raid - имя массива; disk - диск	Состояние каждого диска в RAID-массиве: • 0 - unknown (неизвестное состояние диска) • 1 - absent (диск отсутствует) • 2 - active (диск активен) • 3 - sync (диск синхронизирован) • 4 - spare (запасной диск) • 5 - write-mostly (диск используется преимущественно для записи) • 6 - blocked (диск заблокирован) • 7 - faulty (диск неисправен) Если диск находится в нескольких состояниях, например "active sync", код формируется как комбинация чисел ("23" - 2 для active, 3 для sync).
rdb_raid_size	raid - имя массива.	Размер RAID-массива в байтах.

А.6 Метрики экспортёра

Таблица А.6 — Метрики экспортёра

Название метрики	Метки	Описание метрики
rdb_registry_blocks_count	ds - источник данных.	Количество блоков в регистре.
rdb_registry_free_blocks_count	ds - источник данных.	Количество пустых блоков в регистре.
rdb_registry_state	ds - источник данных.	Состояние регистра: 0 - готов к отправке; 1 - в процессе формирования метрик/не готов к отправке.
rdb_exporter_mem_rss		Количество резидентной памяти в байтах.
rdb_exporter_mem_vms		Количество виртуальной памяти в байтах.
rdb_exporter_mem_uss		Количество USS памяти в байтах.
rdb_exporter_mem_pss		Количество PSS памяти в байтах.
rdb_exporter_mem_data		Количество памяти в байтах, выделенной под секцию data.

(разрыв таблицы)

(разрыв таблицы)

Название метрики	Метки	Описание метрики
rdb_exporter_mem_lib		Количество памяти в байтах, занимаемое библиотеками (*.so).
rdb_exporter_mem_shared		Количество памяти в байтах, разделяемой между процессами.
rdb_exporter_mem_swap		Количество памяти в байтах, записанной в swap.
rdb_exporter_mem_text		Количество памяти в байтах, выделенной под исполняемый код.
rdb_exporter_cpu_user		Время работы процессора (в секундах) в режиме пользователя.
rdb_exporter_cpu_sys		Время работы процессора (в секундах) в режиме ядра.
rdb_exporter_threads_count		Количество потоков процесса экспортёра.

А.7 Другие метрики

Таблица А.7 — 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.
rdb_server_version	kind: major, minor, fix	Версия СУБД.