**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ**

**Федеральное государственное бюджетное образовательное учреждение высшего образования**

**«Московский Авиационный Институт»**

**(Национальный Исследовательский Университет)**

**Институт: №8 «Информационные технологии   
и прикладная математика»   
Кафедра: 806 «Вычислительная математика   
и программирование»**

Лабораторная работа № 1   
по курсу «Виртуализация и облачные технологии»

Группа: М8О-206СВ-24

Студент: М.A. Фролов

Оценка:

Дата: 09.11.2025

Москва, 2025

**ОГЛАВЛЕНИЕ**

[1 Тема 3](#_Toc213586365)

[2 Задание 3](#_Toc213586366)

[3 Ход лабораторной работы 4](#_Toc213586367)

[4 Выводы 15](#_Toc213586368)

# **Тема**

Управляемые базы данных в облаке

# **Задание**

1. Сделать кластер (PostgreSQL)
2. В него загрузить любые >=5 датасетов с data.mos.ru с

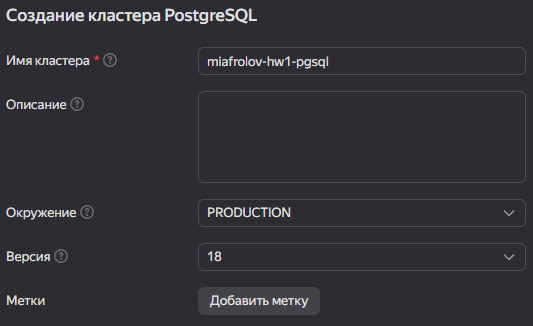
геоданными

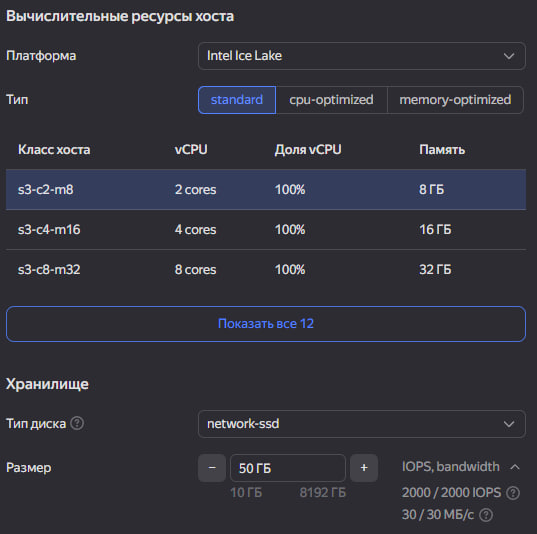
1. В даталенс визуализировать на дашборде эти датасеты

# **Ход лабораторной работы**

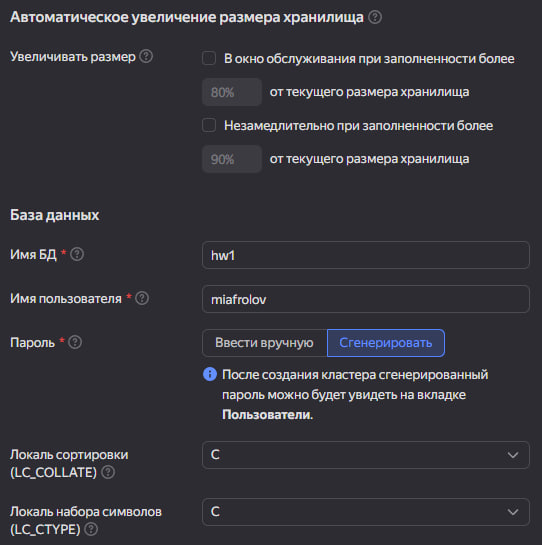
Создаю кластер PostgreSQL в облаке:

Задаю название кластера, окружение production (чтобы не было проблем с тестовыми версиями, новые возможности СУБД использовать не буду) и максимально доступную стабильную версию 18, чтобы как можно дольше не иметь необходимости обновлять мажорную версию PostgreSQL (из-за чего у сервиса может быть даунтайм).

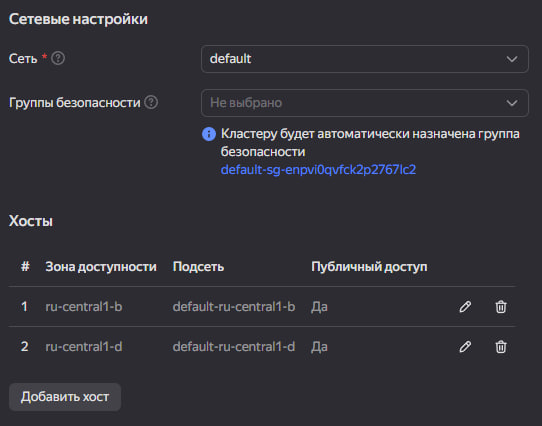




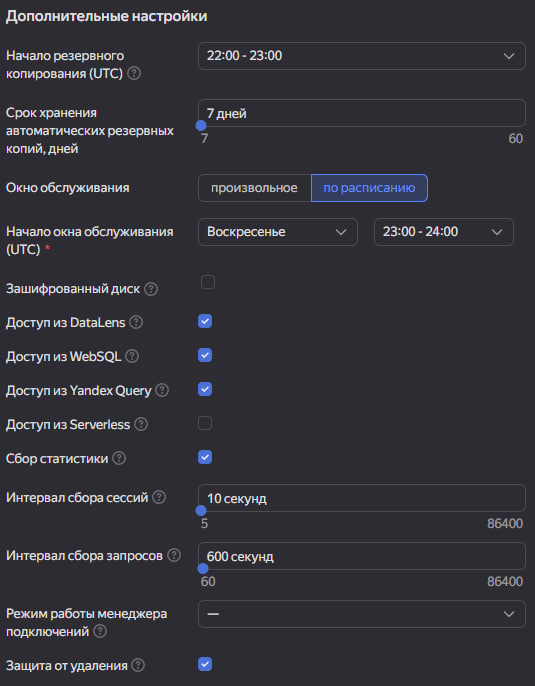
Задаю требуемые кластером ресурсы (выбираю минимальные 2 ядра – если будет меньше, то может быть проблема с hyperthreading и делением физического ядра между разными контейнерами), выбираю размер диска в 50Гб на случай использования тяжёлых датасетов в будущем (указываю network-ssd в качестве типа диска, потому что network-hdd слишком медленно, а local-ssd, т.е. расположенные в рамках серверах хостов, или network-ssd-io-m3, т.е. сетевые реплицируемые диски повышенной производительности, слишком дорогие и в рамках лабораторной работы их использование не требуется).



Не включаю автоматическое увеличение хранилища, потому что 50Гб должно быть достаточно, прописываю параметры базы данных (имя базы данных, имя пользователя, пароль делаю автогенерённый, потом его можно будет найти в разделе пользователи), локаль оставляю C (utf-8) как универсальную для многоязычных баз данных, в сортировке данных на русском языке потребности нет в ходе работы (если всё же будет неоходимо, то можно в рамках отдельных запросов поменять через collate).



Добавляю сетевые настройки - указываю сеть, в которой будут находиться хосты, а также группу безопасности (которая позволяет задавать правила отправки и получения трафика кластером), обязательно добавляю хостам публичный доступ, чтобы в них можно было загрузить датасеты и после этого визуализировать их в DataLens.

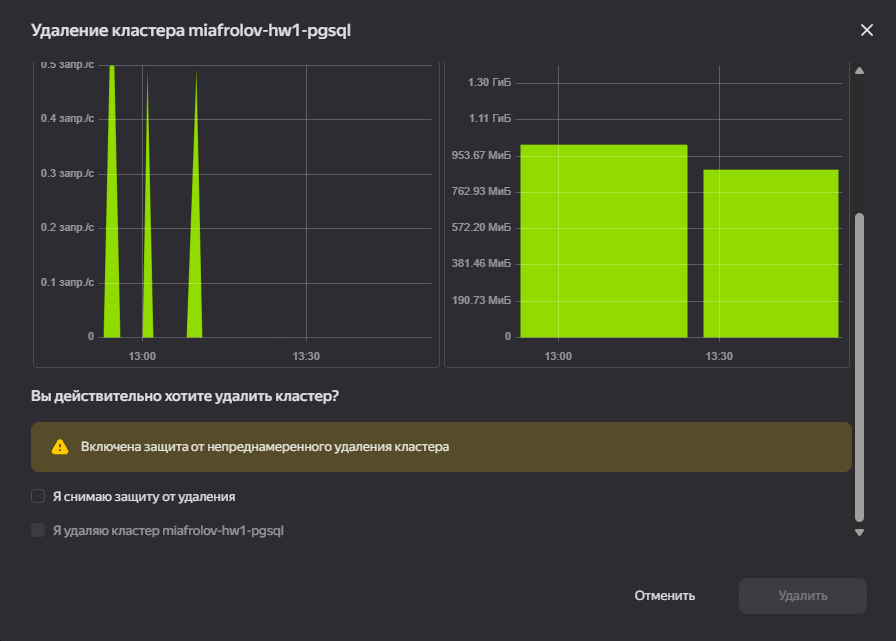


Выставляю резервное копирование на интервал с 22 до 23 часов (чтобы не нагружать кластер в дневное время, срок хранения резервных копий – 7 дней, чтобы иметь возможность откатиться к более ранней версии в случае проблем.

Окно обслуживания (например установка минорных обновлений) выставляю в воскресенье вечером, чтобы в дневное время не обновлять кластер.

Зашифрованный диск не нужен, потому что секретных данных нет, включаю доступы из DateLens для следующей части лабораторной работы, доступ WebSQL, чтобы в интерфейсе можно было выполнять запросы и смотреть успешно ли загрузились данные через скрипт. Через сбор статистики подключал для того, чтобы возможно по диагностике запросов увидеть какими запросами DataLens выбирает данные, но их не оказалось в той вкладке по окончании выполнения лабораторной работы.

Защита от удаления представляет из себя просто дополнительное диалоговое окно при удалении кластера



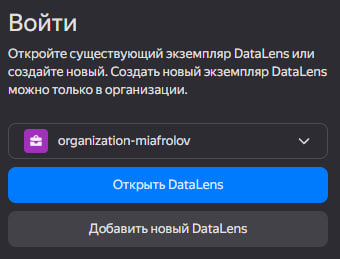
В качестве датасетов решил использовать

* <https://data.mos.ru/opendata/1957>
* <https://data.mos.ru/opendata/62888>
* <https://data.mos.ru/opendata/507>
* <https://data.mos.ru/opendata/516>
* <https://data.mos.ru/opendata/514>

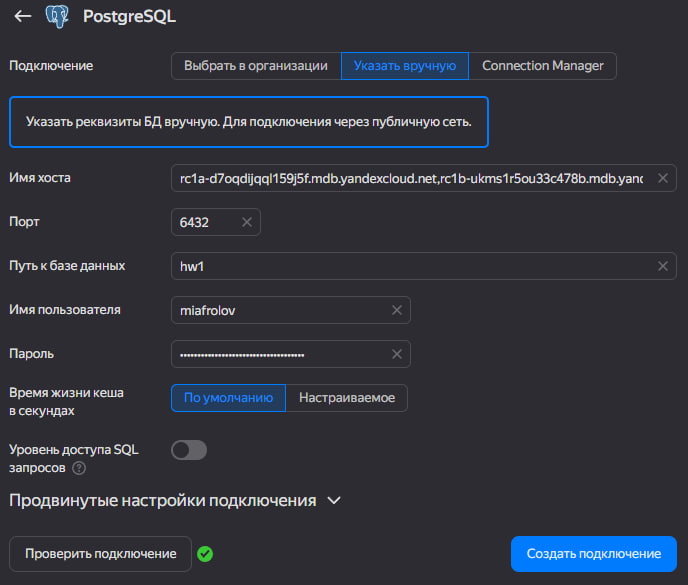
После скачивания CSV с сайта, почистил датасеты и предобработал python скриптом (удалил пустые поля и поля с вложенными данными, заменил координаты из формата coordinates=…, в [lat, lon], для случаев, когда это была multipoint (т.е. набор точек, например филиалы больниц), сделал копии записей, у каждой из которых была только одна пара координат.

Сам скрипт доступен на Github - <https://github.com/Qworl/cloud-hws/blob/main/hw1/migrate_to_cloud.py>, после того как были созданы и наполнены таблицы в БД из датасетов, я приступил к созданию дашборда DataLens.

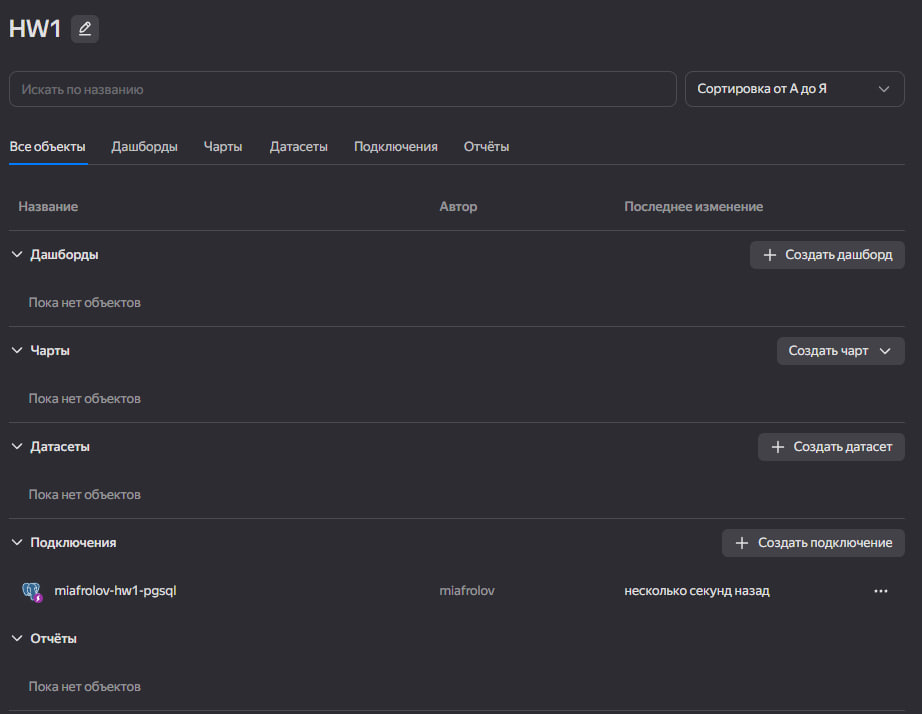
Создаю экземпляр DataLens в своей организации



Добавляю подключение к PostgreSQL кластеру вручную (потому что кластер и DataLens в разных организациях) – здесь же можно перед созданием подключения проверить его работоспособность и поправить настройки при необходимости. Уровень доступа SQL запросов не нужен, потому что хотим работать с содержимым исходной таблицы.

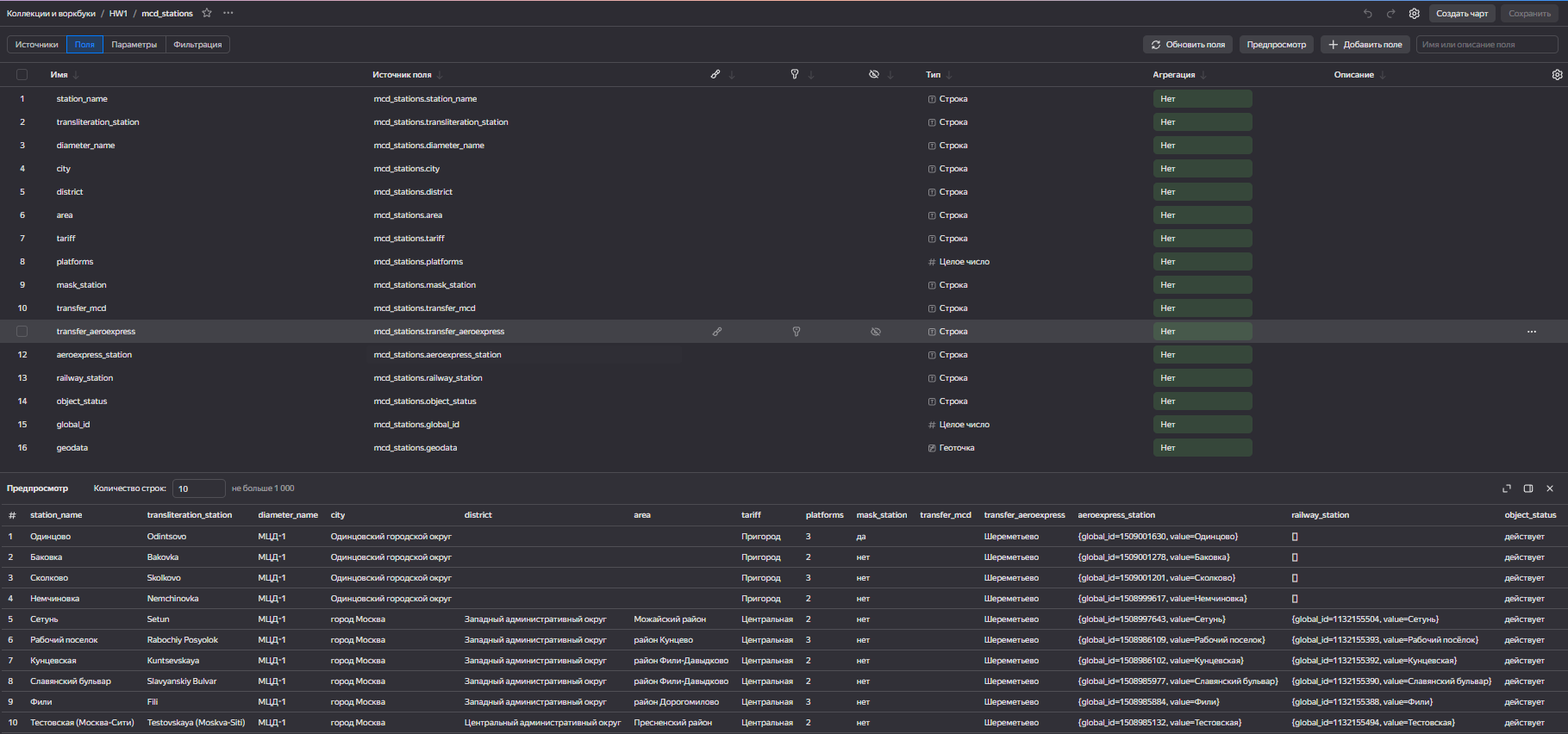


После того как подключение успешно сохранено в DataLens оно будет отображаться в разделе с подключениями

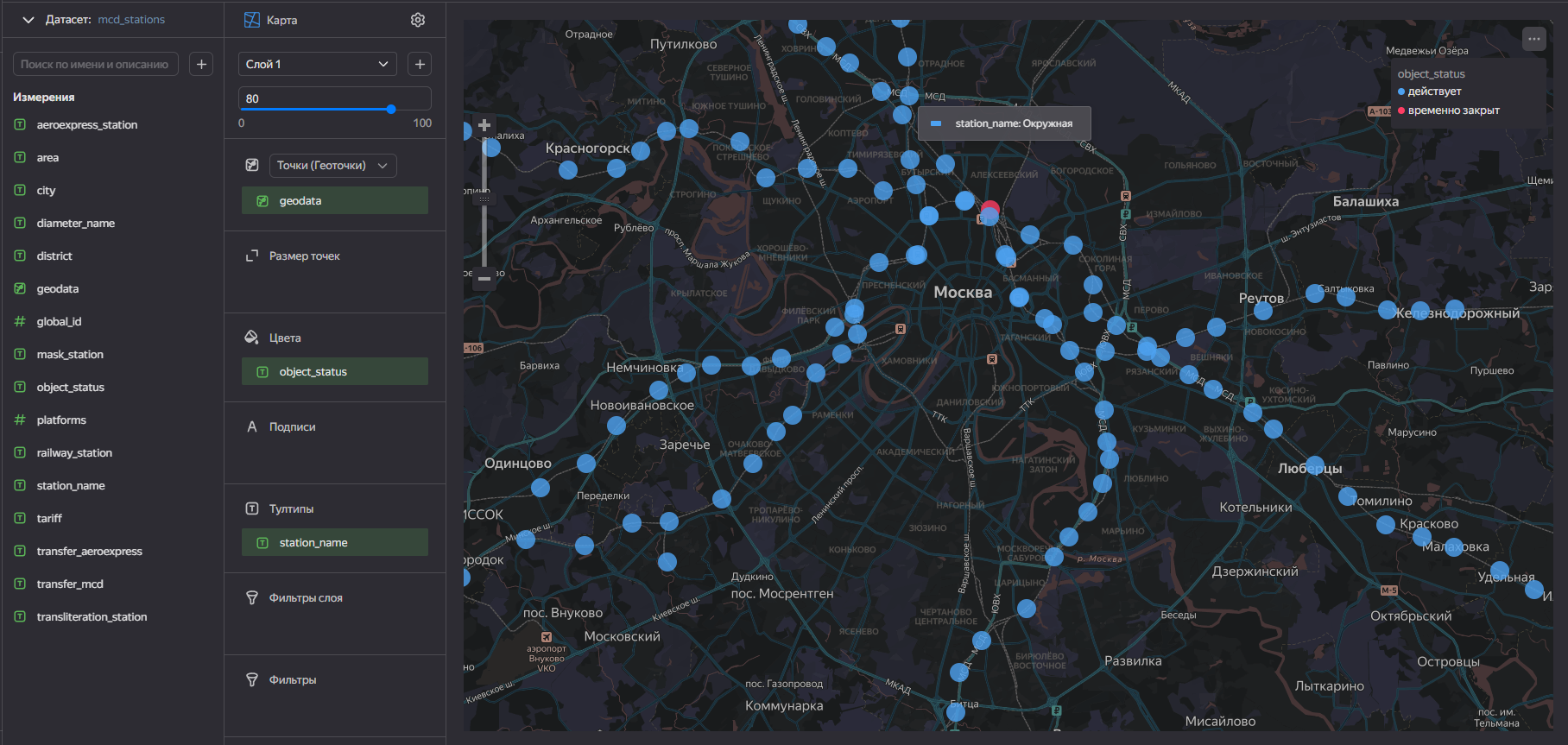


После того как создано подключение можно переходить к добавлению датасетов и созданию чартов из них:

Создаю датасет для станций Московского Центрального Диаметра, для поля geodata изменяю тип на Геоточка, чтобы в дальнейшем значения в нём можно было использовать для построения карты



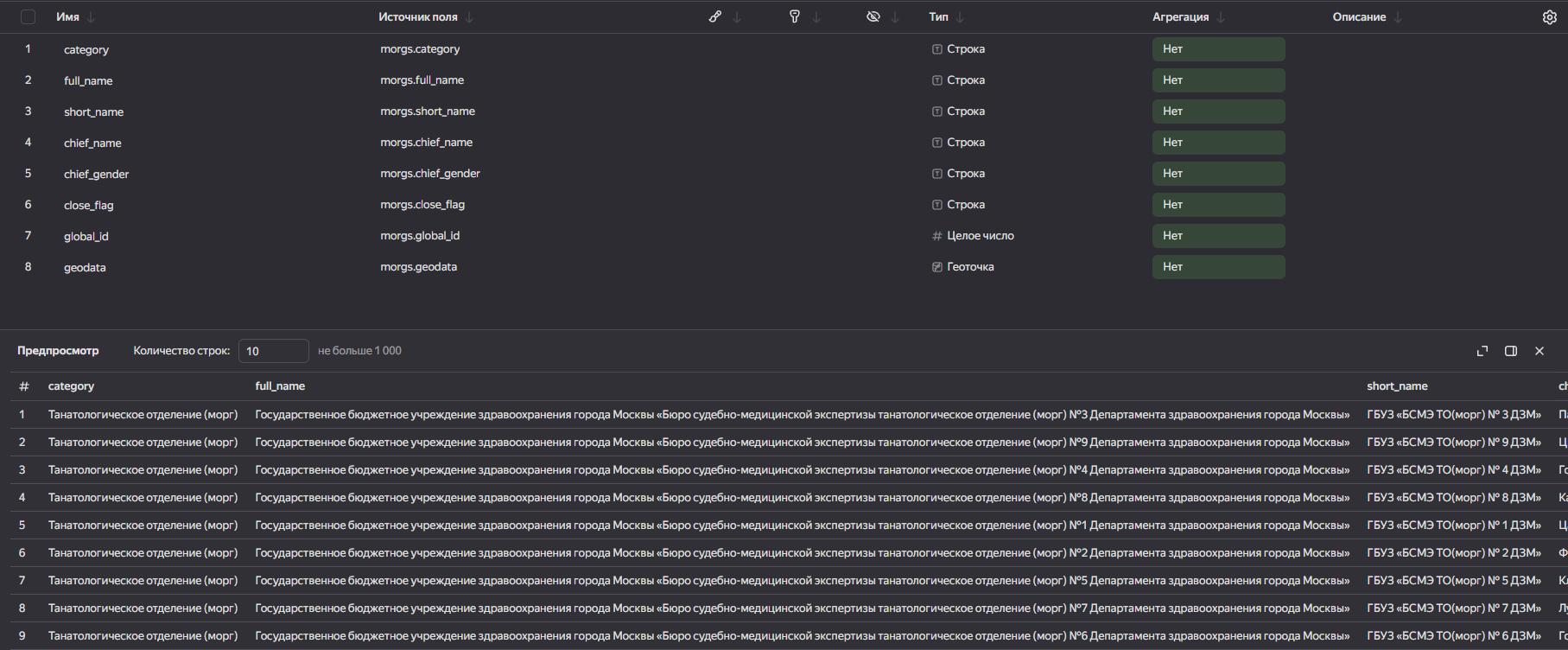
Из датасета создаю чарт с картой, на которую нанесены координаты каждой станции и подписано ее название и её статус (закрыта или действует)



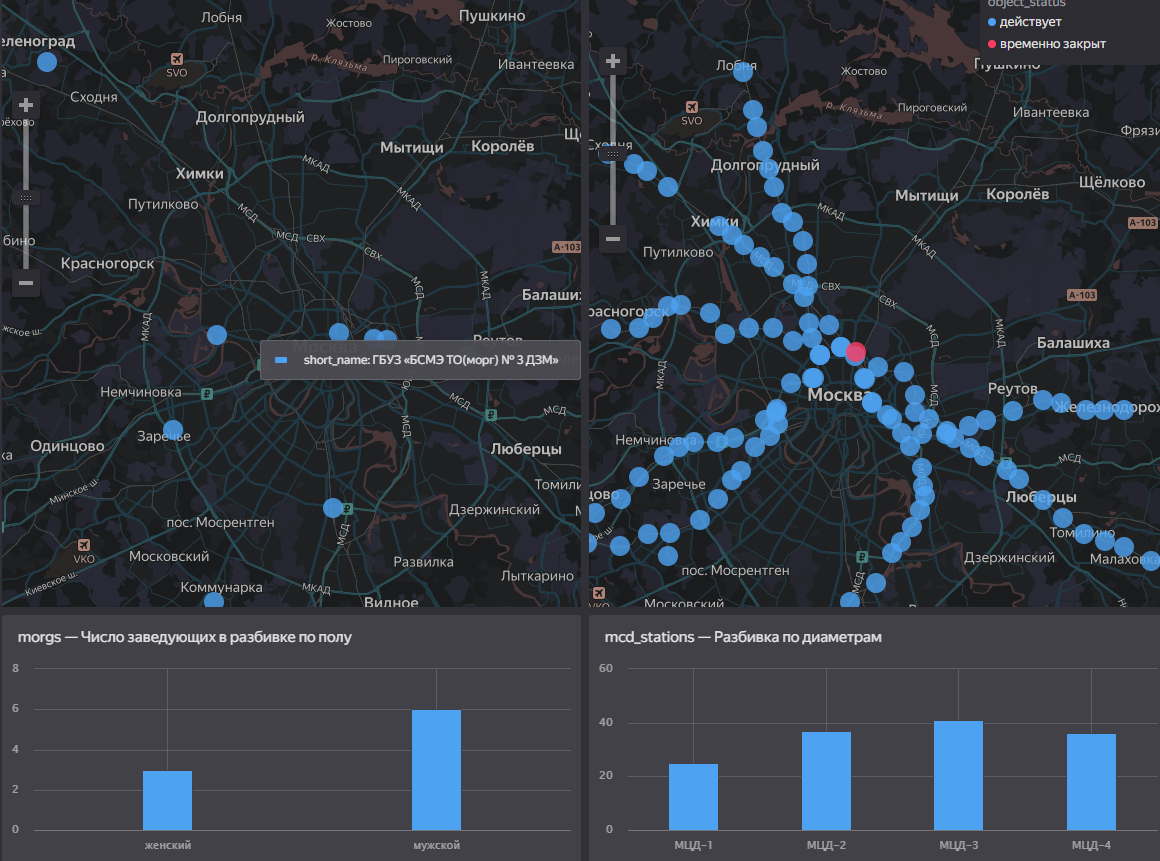
Из этого же датасета создаю ещё один чарт с разбивкой числа станций по диаметру



Аналогично создаём датасет с моргами, удаляя незаполненные колонки из датасета

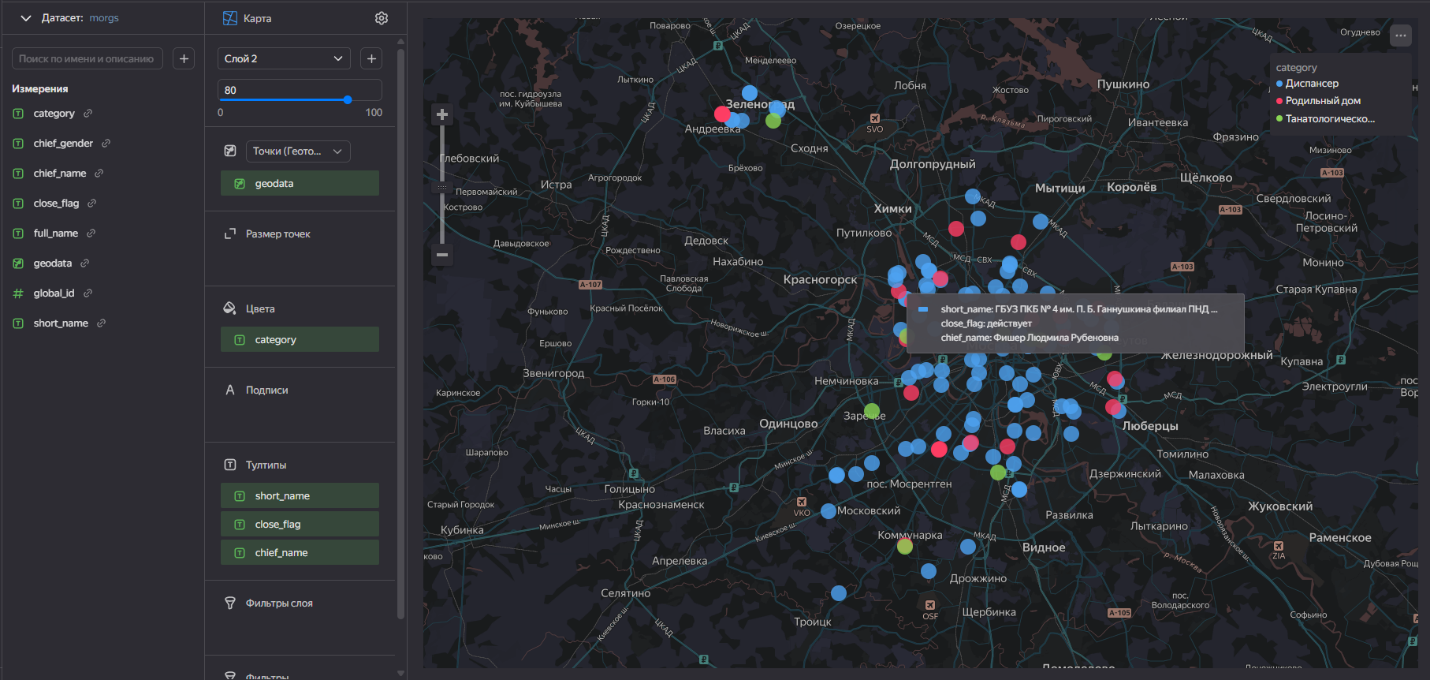
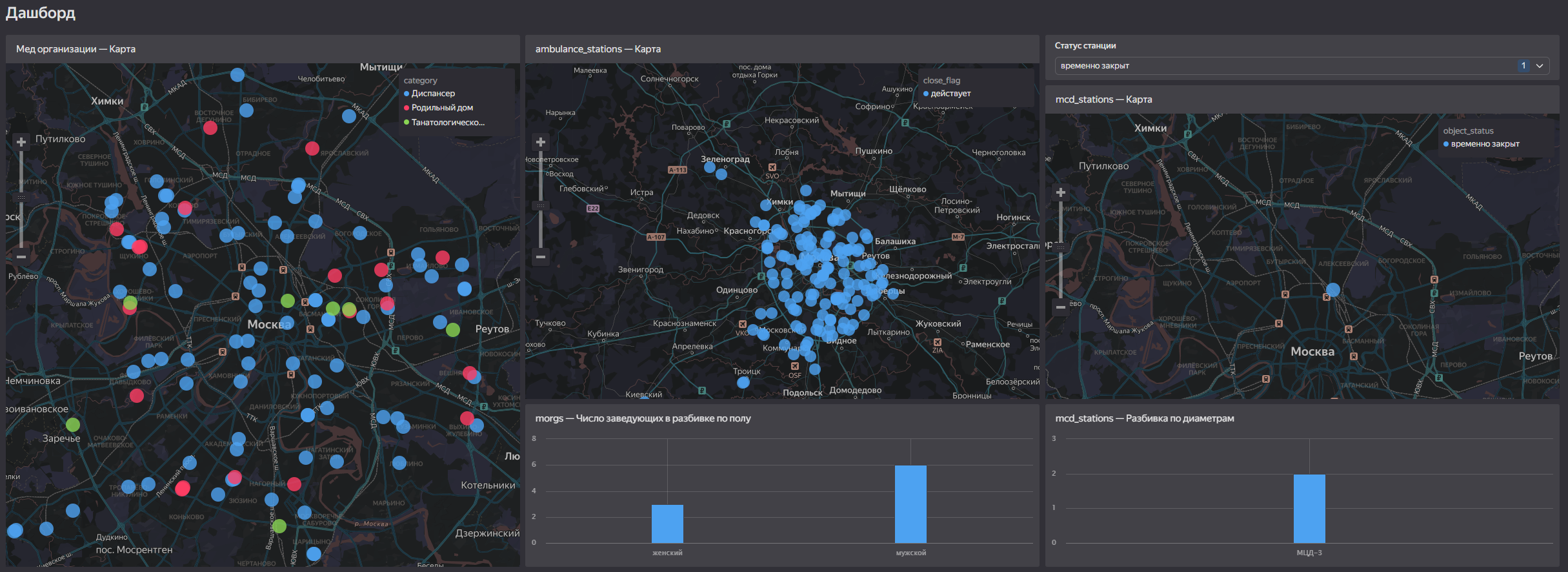


Добавляем несколько чартов из этого датасета на дашборд



Добавляем ещё датасеты с диспансерами, родильными домами, моргами и станциями скорой помощи

Родильные дома, диспансеры и морги отображаем на карте и подписываем статус заведения (открыто или нет) и ФИО начальника.

  
 На дашборд добавляем этот чарт, а также чарты с станциями скорой помощи на карте (их очень много так, что не добавлял в предыдущий чарт), а также разбивку числа заведующих в моргах по полу. Дополнительно добавляем фильтр по статусу станции (работает или закрыта).

# **Выводы**

В ходе выполнения лабораторной работы я освоил работу с облачной СУБД PostgreSQL в Yandex Cloud: создал кластер, выполнил подключение к нему, отработал базовые операции по добавлению и изменению данных. Далее осуществил интеграцию базы данных с инструментом BI‑аналитики Yandex DataLens — настроил источник данных, сформировал датасеты и приступил к созданию визуализаций (чартов и карт).

В процессе работы возникла сложность с отображением геоданных на карте, связанная с несоответствием форматов; задача была решена путём преобразования данных к требуемому виду. Также я поработал с настройками подписей и фильтров в чартах.

На основании проведённой работы можно сделать следующие выводы:

* использование Managed‑решений для СУБД существенно упрощает администрирование (не требуется вручную выполнять резервное копирование, масштабирование и др.);
* DataLens эффективно интегрируется с PostgreSQL и позволяет оперативно создавать наглядные визуализации;
* критически важно соблюдать соответствие форматов данных и корректно настраивать права доступа, чтобы избежать ошибок при подключении и отображении информации.

Таким образом, связка «облачная PostgreSQL + DataLens» продемонстрировала свою работоспособность и удобство для решения аналитических задач.