Формирование оценочного стенда для сравнительного анализа функциональности механизмов полнотекстового поиска баз данных в облачной инфраструктуре Yandex Cloud

А.И. Романенко

***Аннотация*—В наше время, когда количество информации, находящейся в сети Интернет, базах данных и электронных документах растет с каждым днем, поиск нужной информации становится все более сложным и трудоемким. Полнотекстовый поиск помогает решить эту проблему, позволяя пользователям быстро и эффективно находить нужную информацию. Полнотекстовый поиск является одним из самых распространенных и эффективных способов поиска информации в интернете. Он используется в различных областях, включая базы данных, электронную почту, сайты и другие источники информации. Например, в больших корпоративных базах данных полнотекстовый поиск может использоваться для быстрого поиска информации о клиентах, продуктах или услугах. В электронной почте полнотекстовый поиск может помочь пользователю быстро найти все сообщения, которые содержат определенные ключевые слова или фразы. На сайтах полнотекстовый поиск может использоваться для поиска информации о товарах, услугах или статьях.**

В данной статье демонстрируется формирование оценочного стенда для сравнительного анализа функциональности механизмов полнотекстового поиска в различных базах данных с использование облачной инфраструктуры Yandex Cloud. Для сравнительного анализа были выбраны механизмы полнотекстового поиска, встроенные в СУБД PostgreSQL, MySQL, Microsoft SQL Server, а также внешние системы полнотекстового поиска Sphinx и Elasticsearch. Описана подробная схема для установки и настройки выбранных механизмов в облачной инфраструктуре, а также приведены результаты тестирования на подготовленных данных.

***Ключевые слова*— полнотекстовый поиск, Sphinx, Elasticsearch, Yandex Cloud.**

I. Введение

С развитием информационных технологий и расширением объемов информации в интернете, задача поиска нужной информации становится все более актуальной. Сегодня пользователи в интернете ищут не только простые ответы на свои вопросы, но и более сложную информацию, которая может быть разбросана по разным источникам. Именно для решения этой задачи были разработаны механизмы полнотекстового поиска.

Механизмы полнотекстового поиска – это специальные программные средства, которые позволяют осуществлять поиск информации, по ключевым словам, в текстовых документах. Они позволяют находить не только точное совпадение слов, но и синонимы, словоформы, а также учитывать контекст и порядок слов в запросе.

На текущий момент существует множество систем полнотекстового поиска, которые обладают своими достоинствами и недостатками, поэтому важно выбрать наиболее подходящий механизм, который бы обеспечивал максимальную производительность для вашей системы и предоставлял все необходимые возможности поиска.

Данная стать посвящена рассмотрению механизмов полнотекстового поиска в различных базах данных в облачном хранилище Yandex Cloud.

II. Архитектура оценочного стенда

Для получения практических параметров был разработан оценочный стенд, который состоит из разработанного программного комплекса, а также необходимых для его работы внешних баз данных и систем полнотекстового поиска.

Архитектура оценочного стенда визуально отображена на схеме ниже и представлена следующими компонентами:

- Программный комплекс, развёрнутый на локальной машине пользователя
- Облачная виртуальная машина Yandex Cloud, на которой развёрнуты следующие компоненты:
  - СУБД PostgreSQL
  - СУБД MySQL
  - СУБД Microsoft SQL Server
  - Система полнотекстового поиска Sphinx
  - Система полнотекстового поиска Elasticsearch
  - Система сбора статистики

А. И. Романенко, МГУ им. М.В. Ломоносова (e-mail: alex.rom1519@gmail.com)

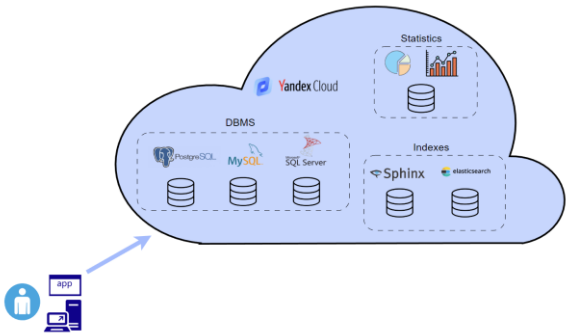

Рис. 1 - Архитектура оценочного стенда

III. Функционал программного комплекса

Для практического сравнения механизмов полнотекстового поиска определим следующие критерии.

- Скорость построения индекса
- Размер индекса
- Скорость выполнения следующих типов запросов:
  - Запрос на естественном языке
  - Запрос в логическом режиме
  - Префиксный поиск
  - Фразовый поиск

Для получения практических параметров был разработан программный комплекс, который позволяет выполнять операции по построению полнотекстового индекса для таблиц, заданных пользователем, выполнять запросы полнотекстового поиска в разных режимах для вышеописанных механизмов, имеет удобный графический интерфейс, а также собирает статистику по выполненным запросом в базу данных.

Функционал программного комплекса состоит из следующих модулей:

- Модуль создания полнотекстового индекса
- Модуль выполнения полнотекстовых запросов различных типов
- Модуль ручного ввода запросов, напрямую к СУБД
- Модуль работы со словарями
- Модуль конфигурации подключений к выбранным СУБД и внешним механизмам
- Модуль сбора статистики

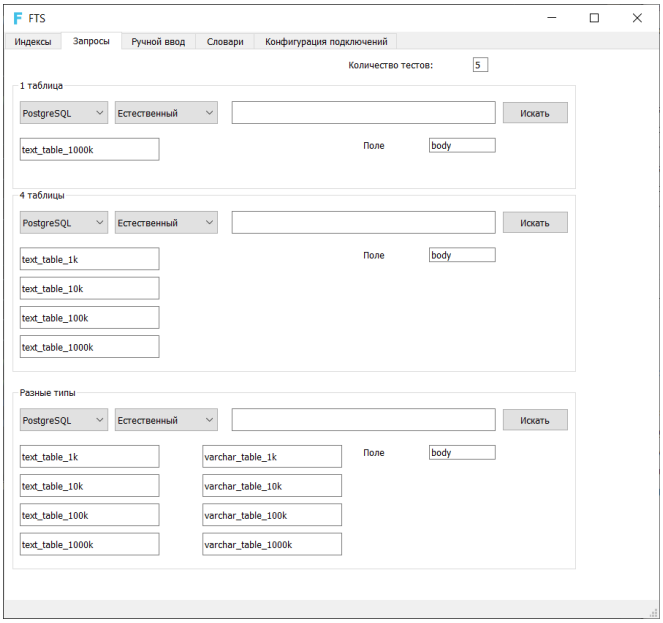

Рис. 2 - Интерфейс программного комплекса

IV. Создание виртуальной машины Yandex Cloud

Для работы оценочного стенда и получения практических параметров было использовано облачное хранилище данных «Yandex Cloud» и сервис «Yandex Compute Cloud», который предоставляет вычислительные мощности для создания виртуальных машин и размещения своих проектов.

Перед созданием виртуальной машины необходимо:

- Зарегистрировать аккаунт «Яндекс ID», который позволяет проходить авторизацию в сервисах Яндекса.
- Авторизоваться с помощью аккаунта «Яндекс ID» в сервисе «Yandex Cloud».
- На странице биллинга подключить платёжный аккаунт, указав свои персональные данные и убедиться, что он находится в статусе ACTIVE или TRIAL_ACTIVE.
- Создать каталог в сервисе «Yandex Cloud», указав его имя.
- Создать сервисный аккаунт (аккаунт, от имени которого программы могут управлять ресурсами в Yandex Cloud), указав имя и роль в каталоге editor.

Также, для доступа к виртуальной машине по протоколу SSH, нужна пара ключей: открытый ключ размещается на ВМ, а закрытый ключ хранится у пользователя. Такой способ более безопасен, чем подключение по логину и паролю. Для создания пары ключей в ОС «Windows 10» необходимо:

- Ввести команду
  ssh-keygen -t ed25519
- Указать имена файлов, в которые будут сохранены ключи.
- Ввести пароль для закрытого ключа.

Для создания виртуальной машины необходимо перейти на официальный сайт «https://cloud.yandex.ru/», зайти в личный кабинет, выбрать раздел «Виртуальные

машины» и выбрать пункт «Создать ВМ». Далее откроется меню создания виртуальной машины, где потребуется ввести базовые параметры:

- Имя виртуальной машины
- Описание
- Зона доступности

В качестве операционной системы была выбрана Ubuntu версии 22.04.

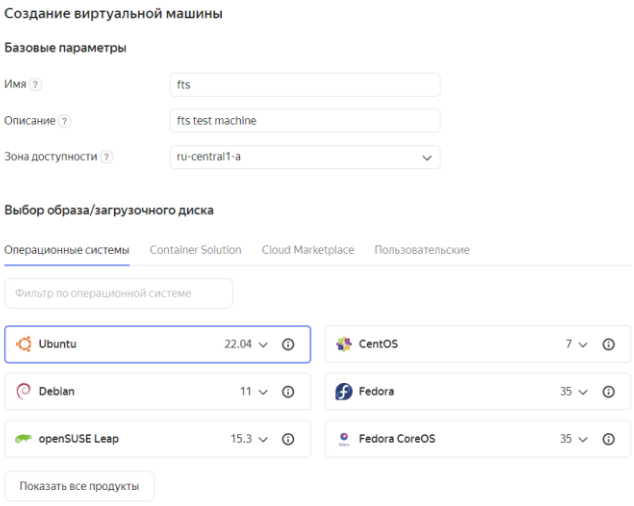

Рис. 3 - Базовые параметры

В процессе настройки облачного хранилища под задачу формирования оценочного стенда были определены следующие характеристики создаваемого диска, такие как:

- Тип диска
- Размер диска

Был выбран загрузочный диск для операционной системы типа SSD, размером 50 ГБ с максимальным значением IOPS (количество операций чтения / записи, выполняемых диском в секунду) 4000/4000 и максимальной пропускной способность 60/60 МБ/с.

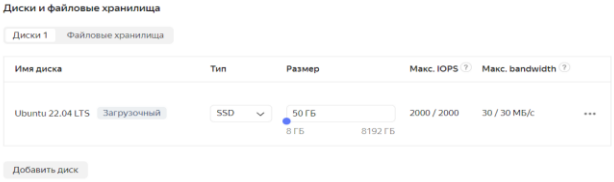

Рис. 4 - Диски и файловые хранилища

Далее определяются параметры вычислительных ресурсов машины. Yandex Compute Cloud предоставляет различные виды физических процессоров. Выбор платформы гарантирует тип физического процессора в дата-центре и определяет набор допустимых конфигураций vCPU (виртуальный процессор) и RAM (оперативная память). Также, к виртуальной машине можно добавить графический ускоритель (GPU).

Для нашей задачи была выбрана платформа «Intel Ice Lake», которая предоставляет в пользование процессор «Intel Xeon Gold 6338» с максимальным количеством ядер 96 и базовой частотой в 2.00 Гц. Для нашей задачи достаточно 2-х ядер. Гарантированная доля vCPU, которая будет выделена виртуальной машине – была выбрана 100%, так как виртуальная машина, с гарантированной долей меньше 100%, могла бы не предоставить нам всю заявленную вычислительную мощность, что могло бы повлиять на результаты наших исследований. Объём оперативной памяти был установлен в 2 ГБ. Опция «Прерываемая» означает, что сервисы Яндекса могут остановить машину в любой момент и гарантированно, по истечению 24 часов с момента запуска, однако она предоставляет скидку на услугу.

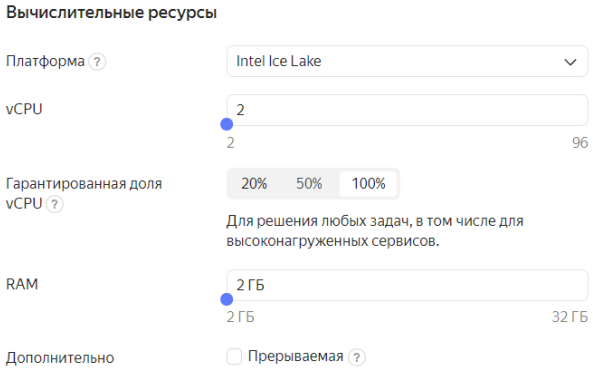

Рис. 5 - Вычислительные ресурсы

В пункте сетевых настроек подсеть была определена по умолчанию, публичный и внутренний адрес предоставлялись автоматически. В качестве дополнительной опции можно выбрать статический адрес из списка, который не будет меняться со временем.

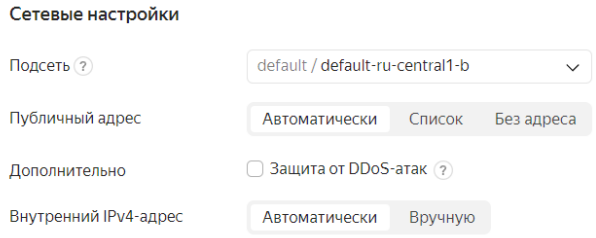

Рис. 6 - Сетевые настройки

В блоке «Доступ» были указаны данные для доступа к виртуальной машине:

- Сервисный аккаунт, который мы создали раннее
- Имя пользователя
- В поле SSH-ключ скопировано содержимое файла открытого ключа, сгенерированного раннее

Также опционально можно разрешить доступ к серийной консоли. Серийная консоль — это способ получить доступ к виртуальной машине вне зависимости от состояния сети или операционной системы. Таким образом, вы можете использовать консоль, например, для устранения неисправностей ВМ или при возникновении проблем с доступом через SSH.

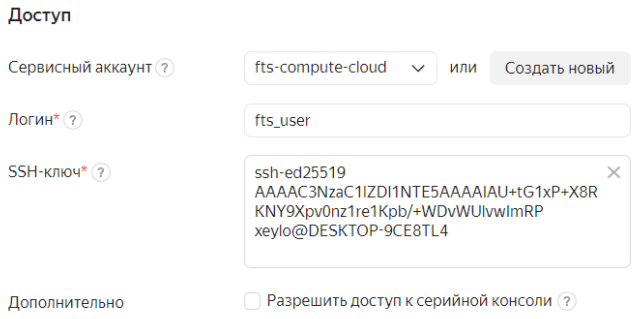

Рис. 7 - Доступ

После создания виртуальной машины мы можем узнать публичный IPv4 адрес для подключения в разделе «Сеть» виртуальной машины и подключиться к ней по протоколу SSH с помощью ключа, сгенерированного раннее.

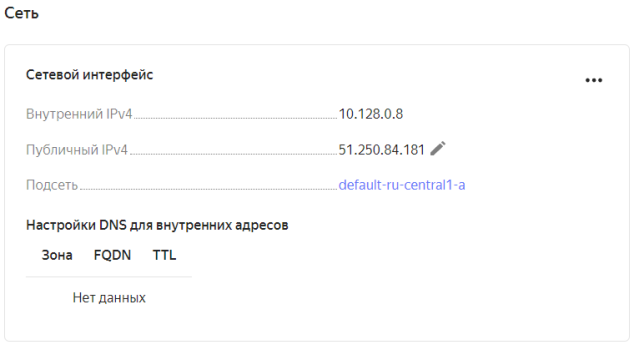

Рис. 8 - Сеть

Для подключения к виртуальной машине использовалась команда

```
ssh -i c:\_tmp\ssh\yc_key_fts fts_user@51.250.84.181
```

После этого предоставляется управление виртуальной машиной.

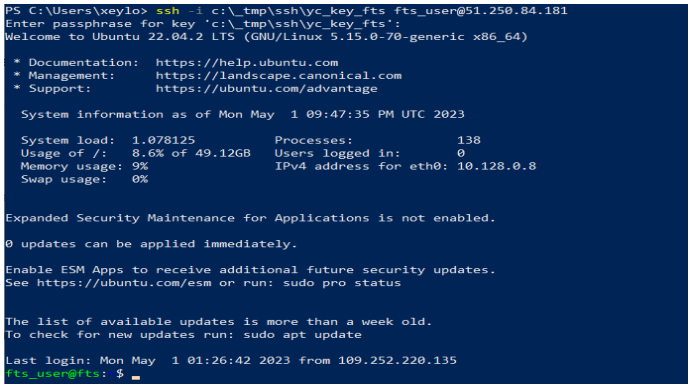

Рис. 9 - Управление виртуальной машиной

V. Установка и настройка PostgreSQL

Для установки PostgreSQL в ОС Ubuntu с помощью apt (advanced packaging tool) — программы для установки, обновления и удаления программных пакетов в операционных системах Debian и основанных на них есть два способа: через репозиторий Ubuntu и через официальный репозиторий PostgreSQL. В первом случае установка требует меньше команд, однако версия PostgreSQL будет установлена та, которая лежит в репозитории Ubuntu и может быть не последней. Второй способ требует большего числа команд, однако гарантирует, что версия СУБД будет последней. Воспользуемся вторым способом. Создадим файл pgdg.list в директории /etc/apt/sources.list.d, который будет содержать адрес официального репозитория PostgreSQL, в зависимости от конкретно вашей версии Ubuntu.

```
sudo sh -c 'echo "deb http://apt.postgresql.org/pub/repos/apt $(lsb_release -cs)-pgdg main" > /etc/apt/sources.list.d/pgdg.list'
```

Затем добавим PostgreSQL GPG (GNU Privacy Guard) ключ в менеджер пакетов. APT будет использовать этот ключ для проверки подлинности пакетов в репозитории.

```
wget --quiet -O - https://www.postgresql.org/media/keys/ACCC4CF8.asc | sudo apt-key add -
```

Теперь обновим списки пакетов, чтобы APT знал, где найти официальные пакеты PostgreSQL.

```
sudo apt update -y
```

Можно заметить, что команда теперь включает адрес официального репозитория PostgreSQL при проверке наличия пакетов.

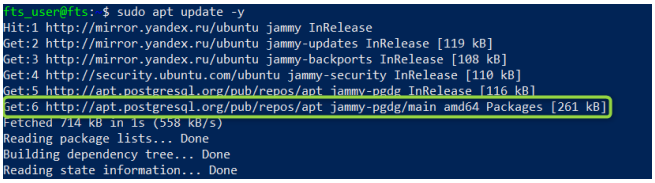

Рис. 10 - Проверка наличия пакетов

Установим PostgreSQL с помощью команды

```
sudo apt install -y postgresql postgresql-contrib postgresql-client
```

Проверим установленную версию PostgreSQL с помощью команды

```
sudo dpkg --status postgresql
```

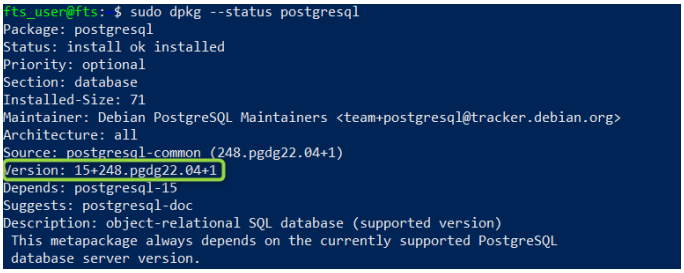

Рис. 11 - Проверка установленной версии

Установка PostgreSQL завершена, запустим PostgreSQL сервис

```
sudo systemctl start postgresql.service
```

Проверим статус PostgreSQL, выполнив приведенную ниже команду

```
sudo systemctl status postgresql.service
```

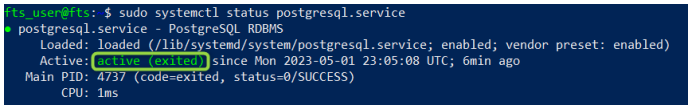

Рис. 12 - Проверка статуса службы

Служба должна быть в статусе active.

Проверим подключение к базе данных. По умолчанию для доступа пользователя к базе данных пароль не требуется. Зайдём в PostgreSQL shell и выведем список всех доступных баз на сервере.

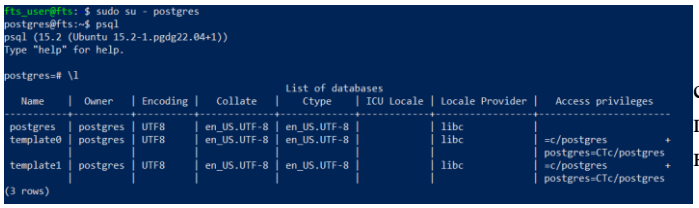

Рис. 13 - Список доступных баз

Создадим базу данных fts_db с помощью команды

```
create database fts_db;
```

Создадим учётную запись пользователя fts_db_user с паролем fts_pass

```
create user fts_db_user with encrypted password 'fts_pass';
```

Выдадим пользователю права доступа к созданной БД

```
grant all privileges on database fts_db to fts_db_user;
```

Проверим подключение с правами пользователя

```
psql -hlocalhost -U fts_db_user -W fts_db
```

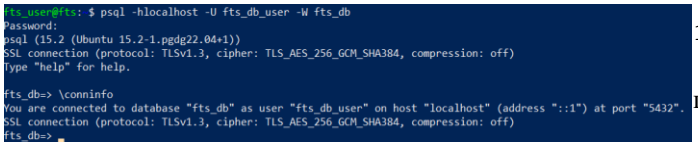

Рис. 14 - Проверка подключения

Создадим схему FTS, которая будет содержать целевые таблицы с данными для использования функционала разрабатываемого модуля

```
CREATE SCHEMA FTS;
```

Создадим в новой схеме таблицу, по которой в дальнейшем будет происходить поиск.

```
CREATE TABLE FTS.text_table_1k (
  id int4,
  title text,
  body text
);
```

Загрузим в созданную схему подготовленный дамп данных

```
COPY FTS.text_table_1k (id, title, body) FROM '/home/fts_user/fts_data/text_table_1k.csv' DELIMITER ',' CSV HEADER;
```

Так как PostgreSQL по умолчанию принимает только локальные подключения, настроим возможность подключения к базе извне. Для этого изменим текуший конфигурационный файл postgresql.conf, который лежит по адресу /etc/postgresql/15/main. Установим значение параметра listen_addresses = '*', чтобы PostgreSQL мог принимать запросы на подключение с любого IP адреса.

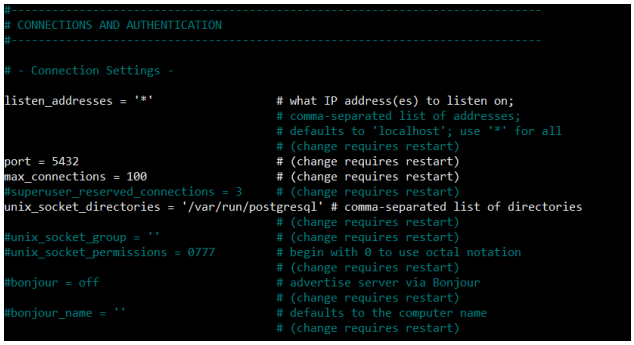

Рис. 15 - Настройка внешнего подключения

Далее выдадим нашему пользователю fts_db_user сетевой доступ к базе данных fts_db с любого адреса после авторизации по md5-паролю. Для этого добавим в конфигурационный файл pg_hba.conf следующую строку

```
host fts_db fts_db_user 0.0.0.0/0 md5
```

Затем перезапустим PostgreSQL.

Настройка PostgreSQL завершена.

VI. Установка и настройка MySQL

Для установки MySQL с официального репозитория разработчиков перейдём на страницу загрузки официального сайта dev.mysql.com, выберем последнюю версию MySQL APT Repository и выполним команду для её загрузки

```
wget -c https://dev.mysql.com/get/mysql-apt-config_0.8.25-1_all.deb
```

Установим пакет

```
sudo dpkg -i mysql-apt-config_0.8.25-1_all.deb
```

В процессе установки выберем следующие параметры:

- MySQL Server & Cluster: mysql-8.0
- MySQL Tools & Connectors: Enabled
- MySQL Preview Peckages: Enabled

Обновим репозиторий

```
sudo apt-get update
```

Установим MySQL сервер

```
sudo apt-get install mysql-server
```

В процессе установки зададим пароль для root пользователя fts_root_pass и выберем плагин аутентификации по умолчанию «Strong Password Encryption».

Для настройки безопасного доступа к MySQL запустим специальный MySQL скрипт, который изменяет некоторые наиболее уязвимые опции СУБД, использующиеся по умолчанию, такие как:

- Использование компонента VALIDATE PASSWORD (установить уровень валидации, выбрав из: 0 – низкий, 1 – средний, 2 – высокий)
- Изменение пароля от root (скрипт оценит надёжность введённого пароля и запросит ваше согласие на его использование)
- Удаление анонимных пользователей
- Запрет на удалённый логин под root
- Удаление тестовой базы и доступа к ней
- Обновление таблицы привилегий

```
sudo mysql_secure_installation
```

Настройка безопасности завершена.

Создадим новую базу данных fts_db

```
CREATE DATABASE fts_db;
```

Так как изначально MySQL слушает только localhost, для прослушивание всех адресов необходимо отредактировать файл конфигурации MySQL и добавить или изменить значение опции bind-address. Вы можете установить один IP-адрес или диапазоны IP-адресов. Если адрес установлен 0.0.0.0, сервер MySQL принимает соединения на всех интерфейсах IPv4 хоста. Установим опцию в значение 0.0.0.0, для этого откроем файл mysqld.cnf в директории etc/mysql/mysql.conf.d и добавим туда строку

```
bind-address = 0.0.0.0
```

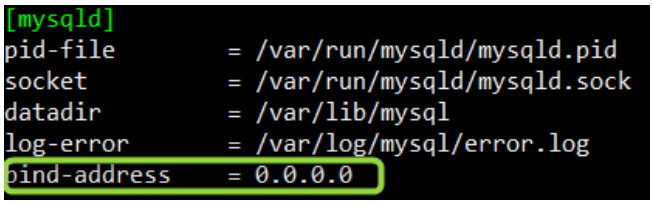

Рис. 16 - Настройка опции bind-address

Перезапустим службу MySQL

```
sudo systemctl restart mysql
```

Создадим пользователя fts_db_user с паролем fts_pass, который будет иметь доступ к базе с любого IP-адреса, для этого укажем параметр '%' после имени пользователя

```
CREATE USER 'fts_db_user'@'%' IDENTIFIED BY 'fts_pass';
```

Выдадим пользователю права доступа к созданной БД

```
GRANT ALL PRIVILEGES ON fts_db.* TO 'fts_db_user'@'%';
```

Для подключения к базе с локального компьютера будем использовать следующие параметры:

- Хост: 51.250.84.181
- Порт: 3306
- База данных: fts_db
- Пользователь: fts_db_user
- Пароль: fts_pass

Также, для подключения необходимо добавить параметры соединения

```
allowPublicKeyRetrieval = true
useSSL = false
```

Создание таблиц и загрузка данных проводятся по аналогии с запросами для PostgreSQL.

VII. Установка и настройка SQL Server

Чтобы настроить SQL Server в Ubuntu, выполним следующие команды в терминале для установки пакета mssql-server.

Импортируем открытые ключи GPG из репозитория

```
wget -qO- https://packages.microsoft.com/keys/microsoft.asc | sudo tee /etc/apt/trusted.gpg.d/microsoft.asc
```

Зарегистрируем репозиторий Ubuntu для SQL Server

```
sudo add-apt-repository "$(wget -qO- https://packages.microsoft.com/config/ubuntu/20.04/mssql-server-2022.list)"
```

Теперь обновляем список пакетов

```
sudo apt-get update
```

И устанавливаем ms sql server

```
sudo apt-get install -y mssql-server
```

После установки пакета настроим конфигурацию SQL Server

```
sudo /opt/mssql/bin/mssql-conf setup
```

Выберем выпуск SQL Server Developer и зададим пароль.

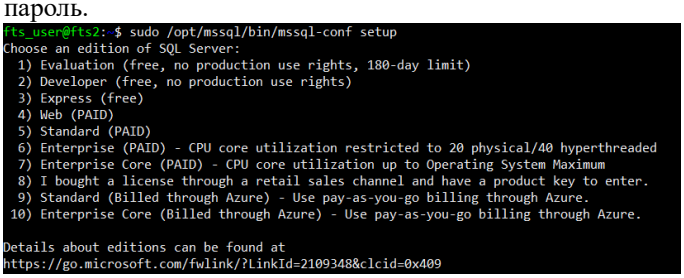

Рис. 17 - Выбор выпуска SQL Server Developer

Чтобы создать базу данных, необходимо подключиться с помощью средства, которое позволяет выполнять инструкции Transact-SQL в SQL Server. Инструмент для подключения к СУБД, по умолчанию, не устанавливается с сервером. Также для его установки используется другой репозиторий, нежели чем для самого SQL Server.

Импортируем открытые ключи GPG из репозитория:

```
curl https://packages.microsoft.com/keys/microsoft.asc | sudo tee /etc/apt/trusted.gpg.d/microsoft.asc
```

Зарегистрируем репозиторий Ubuntu

```
curl https://packages.microsoft.com/config/ubuntu/20.04/prod.list | sudo tee /etc/apt/sources.list.d/msprod.list
```

Обновим список источников и выполним команду установки с помощью пакета разработчика unixODBC

```
sudo apt-get update
sudo apt-get install mssql-tools unixodbc-dev
```

Для удобства добавим /opt/mssql-tools/bin/ в переменную среды PATH, чтобы sqlcmd или bcp можно было открыть из оболочки bash. Изменим переменную среды PATH в файле ~/.bash_profile с помощью следующей команды

```
export PATH="$PATH:/opt/mssql-tools/bin"
```

Для локального подключения к базе будем использовать команду

```
sqlcmd -S localhost -U sa
```

Создадим базу данных fts_db

```
CREATE DATABASE fts_db;
```

Создадим логин для подключения к базе, пользователя и выдадим ему права для работы с созданной базой.

```
USE fts_db;
CREATE LOGIN fts_db_login WITH PASSWORD='Ftspass968';
CREATE USER fts_db_user FOR LOGIN fts_db_login;
EXEC sp_addrolemember 'db_owner', 'fts_db_user';
```

По умолчанию SQL Server слушает внешние IP-адреса, следовательно необходимости в дополнительных сетевых настройках нет. Для подключения будем использовать следующие параметры:

- Хост: 51.250.84.181
- Порт: 1433
- База данных: fts_db
- Пользователь: fts_db_login
- Пароль: Ftspass968

Создание таблиц и загрузка данных проводятся по аналогии с запросами раннее.

VIII. Установка и настройка Sphinx

Для установки Sphinx добавим Sphinxsearch репозиторий и обновим список пакетов

```
sudo add-apt-repository ppa:builds/sphinxsearch-rel22
sudo apt-get update
```

Установим пакет Sphinxsearch

```
sudo apt-get install sphinxsearch
```

Поисковая система Sphinx успешно установлена на сервер. После установки Sphinx нуждается в дополнительной настройке. Конфигурации Sphinx должны храниться в файле sphinx.conf в каталоге /etc/sphinxsearch. Они состоят из трёх основных блоков: index, searchd и source.

Создадим конфигурационный файл и зададим конфигурацию

```
sudo nano /etc/sphinxsearch/sphinx.conf
```

Блок source содержит тип источника данных, имя пользователя и пароль. Первый столбец sql_query должен содержать уникальный ID. Запрос SQL будет выполняться для каждого индекса, а затем передавать данные в индексный файл Sphinx. Блок source состоит из таких полей:

- type: тип источника данных. В нашем случае это mysql (также система поддерживает типы pgsql, mssql, xmlpipe2, odbc и т.д.)
- sql_host: имя хоста; в нашем случае это localhost. В это поле нужно внести домен или IP-адрес
- sql_user: имя пользователя (в нашем случае это root)
- sql_pass: пароль
- sql_db: имя БД, в которой хранятся нужные данные
- sql_query: запрос, который сбрасывает данные в индексный файл

Блок index содержит данные об источнике и путь к местонахождению данных.

- source: имя блока source. В нашем случае это src1.
- path: путь к индексному файлу.

Блок searchd содержит порты и переменные для запуска демона Sphinx

- listen: порт, на котором нужно запустить Sphinx, и используемый протокол. Популярные протоколы Sphinx – sphinx (SphinxAPI) и :mysql41 (SphinxQL)
- query_log: путь к логу запросов
- pid_file: путь к PID-файлу Sphinx
- max_matches: максимальное количество совпадений, которое нужно выводить за один раз
- seamless_rotate: предотвращает останов searchd при кэшировании большого объема данных
- preopen_indexes: указывает, нужно ли предварительно открывать все индексы
- unlink_old: указывает, нужно ли отключить старые копии индекс-файлов

Ниже приведены все конфигурации файла sphinx.conf.

```
source text_table_1k_source
{
  type          = pgsql
  sql_host      = 51.250.84.181
  sql_user      = fts_db_user
  sql_pass      = fts_pass
  sql_db        = fts_db
  sql_port      = 5432
  sql_query     = \
  SELECT ID, title, body AS ID, title, body \
  FROM FTS.text_table_1k
}

index text_table_1k_index
{
  source            = text_table_1k_source
  path                                  = /var/lib/sphinxsearch/data/text_table_1k_index
  docinfo           = extern
}

searchd
{
  listen            = 9306:mysql41
  log                                   = /var/log/sphinxsearch/searchd.log
  query_log                             = /var/log/sphinxsearch/query.log
  read_timeout      = 5
  max_children      = 30
  pid_file                              = /var/run/sphinxsearch/searchd.pid
  seamless_rotate   = 1
  preopen_indexes   = 1
  unlink_old        = 1
  binlog_path                           = /var/lib/sphinxsearch/data
}
```

Добавим данные новой конфигурации в индекс Sphinx

```
sudo indexer -all
```

```
fts_user@fts:~$ sudo indexer --all
Sphinx 2.2.11-id64-release (95ae9a6)
Copyright (c) 2001-2016, Andrew Aksyonoff
Copyright (c) 2008-2016, Sphinx Technologies Inc (http://sphinxsearch.com)

using config file '/etc/sphinxsearch/sphinx.conf'...
indexing index 'text_table_1k_index'...
WARNING: Attribute count is 0: switching to none docinfo
collected 991 docs, 5.5 MB
sorted 0.4 Mhits, 100.0% done
total 991 docs, 5502856 bytes
total 0.236 sec, 23279406 bytes/sec, 4192.34 docs/sec
total 3 reads, 0.001 sec, 1078.5 kb/call avg, 0.5 msec/call avg
total 16 writes, 0.003 sec, 366.7 kb/call avg, 0.2 msec/call avg
```

Рис. 18 - Добавление индекса Sphinx

По умолчанию демон Sphinx отключен. Чтобы включить Sphinx, откроем /etc/default/sphinxsearch и изменим START=no на START=yes

```
sudo sed -i 's/START=no/START=yes/g' /etc/default/sphinxsearch
```

Перезапустим Sphinx с помощью systemctl

```
sudo systemctl restart sphinxsearch.service
```

Для подключения к Sphinx с локального компьютера дополнительных настроек не требуется, так как в блоке searchd мы указали параметр listen = 9306:mysql41, Sphinx будет слушать все IP-адреса порта 9306. Аутентификации не требуется. Для подключения будем использовать следующие параметры:

- Хост: 51.250.84.181
- Порт: 9306

Настройка Sphinx завершена.

IX. Установка и настройка Elasticsearch

Для установки Sphinx добавим Sphinxsearch репозиторий и обновим список пакетов

Компоненты Elasticsearch отсутствуют в репозиториях пакетов Ubuntu по умолчанию. Однако их можно установить с помощью APT после добавления списка источников пакетов Elastic.

Установим пакет OpenJDK 11, который необходим для работы Elasticsearch:

```
sudo apt install openjdk-11-jdk
```

Добавим ключ и репозиторий Elasticsearch в APT:

```
wget -qO - https://artifacts.elastic.co/GPG-KEY-elasticsearch | sudo apt-key add -
echo "deb https://artifacts.elastic.co/packages/7.x/apt stable main" | sudo tee /etc/apt/sources.list.d/elastic-7.x.list
```

Обновим список пакетов и установим Elasticsearch:

```
sudo apt update
sudo apt install elasticsearch
```

Настроим Elasticsearch, отредактировав файл конфигурации /etc/elasticsearch/elasticsearch.yml. Рассмотрим некоторые наиболее важные настройки:

- cluster.name: имя кластера Elasticsearch
- node.name: имя узла Elasticsearch
- network.host: IP-адрес или имя хоста, на котором будет работать Elasticsearch
- http.port: порт HTTP API Elasticsearch
- bootstrap.memory_lock: true, чтобы разрешить Elasticsearch заблокировать память в области страницы, что уменьшает вероятность переключения контекста и улучшает производительность.

По умолчанию Elasticsearch доступен только с локального адреса, для прослушивания внешних адресов зададим параметр network.host: 0.0.0.0 и http.port: 9200, также зададим имя кластера cluster.name: fts_cluster и имя узла node.name: fts_node.

```
cluster.name: fts_cluster
node.name: fts_node
network.host: 0.0.0.0
http.port: 9200
bootstrap.memory_lock: true
```

Запустим службу Elasticsearch

```
sudo systemctl start Elasticsearch
```

Проверим, что Elasticsearch работает, отправив запрос к его API:

```
curl http://51.250.84.181:9200/
```

Ответ сервера

```
{
  "name" : "fts_node",
  "cluster_name" : "fts_cluster",
  "cluster_uuid" : "niwmztQkSpO9ODZGmHuQ",
  "version" : {
    "number" : "7.17.9",
    "build_flavor" : "default",
    "build_type" : "deb",
    "build_hash":
"83c34f456ae29d60e94d886e455e6a3409bba9ed",
    "build_date" : "2023-02-
02T21:56:19.031608185Z",
    "build_snapshot" : false,
    "lucene_version" : "9.5.0",
    "minimum_wire_compatibility_version" :
"6.8.0",
    "minimum_index_compatibility_version"
: "6.0.0-beta1"
  },
  "tagline" : "You Know, for Search"
}
```

Настройка Elasticsearch завершена.

X. Настройка системы сбора статистики

Для сбора статистических данных развернём ещё одну базу данных fts_statistics с помощью СУБД PostgreSQL

create database fts_statistics;

Создадим в ней схему fts и таблицы для сбора статистики по результатам выполнения запросов полнотекстового поиска «query_log_table» и результатам работы модуля построения полнотекстового индекса «index_log_table». Состав полей продемонстрирован на диаграмме.

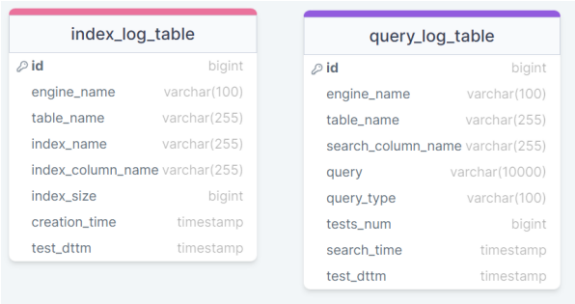

Рис. 19 - Диаграмма базы данных статистики

Настройка системы сбора статистики завершена.

XI. Результаты работы оценочного стенда

Для получения практических параметров был скачан дамп русскоязычной википедии с сайта https://dumps.wikimedia.org/backup-index.html. Он имеет формат xml и содержит более 1 миллиона статей. Для проведения тестов были подготовлены таблицы размерностью 1000, 10000, 100000, 1000000 записей с полями title и body, содержащие заголовок и тело статей соответственно.

Для тестирования скорости работы поиска на естественном языке, поиск проводился по фразе «американский фильм». Время работы механизмов представлено на рисунке 20.

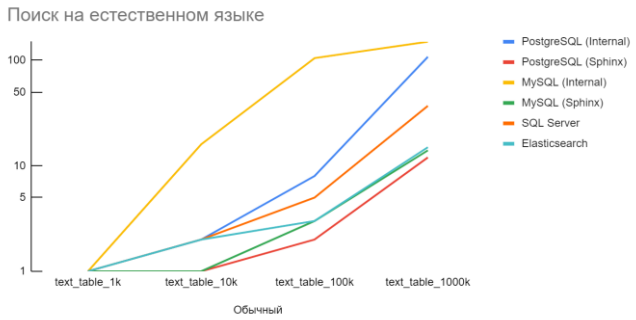

Рис. 20 – Поиск на естественном языке

Поиск на естественном языке дольше всех выполнял встроенный в MySQL механизм, наиболее быстрым оказался Sphinx. Среди встроенных механизмов быстрее всех был SQL Server.

Для тестирования скорости работы поиска в логическом режиме, поиск проводился по запросу «+фильм +боевик -драма».

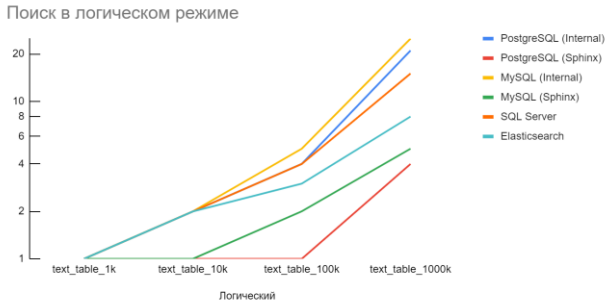

Рис. 21 – Логический поиск

Префиксный поиск тестировался запросом «рус*» такой запрос должен найти все фрагменты, начинающиеся на «рус».

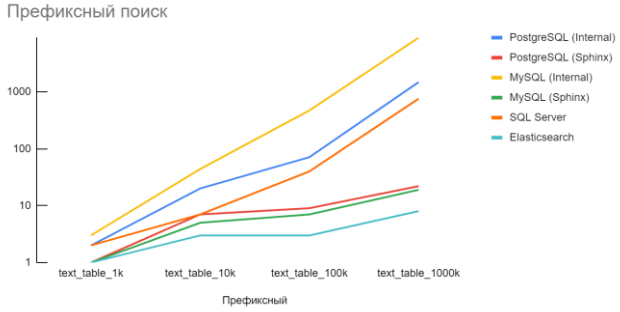

Рис. 22 – Префиксный поиск

Фразовый поиск тестировался на фразе «драматический сериал». Данный вид поиска ищет документы, в точности содержащие искомую фразу.

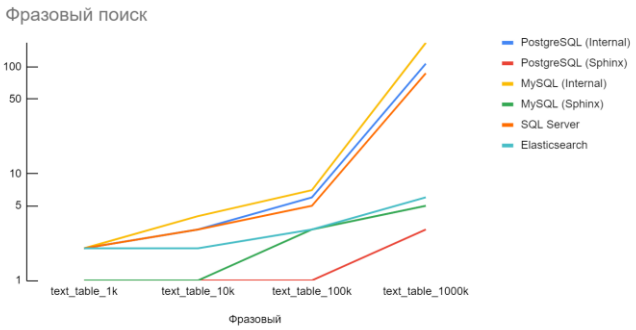

Рис. 23 – Фразовый поиск

Поиск на естественном языке, в логическом режиме, фразовый поиск и префиксный поиск дольше всех выполнял встроенный в MySQL механизм, наиболее быстрым в поиске на естественном языке, в логическом режиме и фразовом поиске оказался Sphinx, быстрее всего с префиксным поиском справился Elasticsearch. Среди встроенных механизмов быстрее всех на всех типах запросов отработал SQL Server.

XII. Заключение

В данной статье было продемонстрировано создание оценочного стенда, позволяющего выполнять сравнение функциональных характеристик механизмов полнотекстового поиска для различных баз данных в облачной инфраструктуре Yandex Cloud. Описана архитектура оценочного стенда и взаимодействие его компонент. Подробно продемонстрирована установка и настройка СУБД PostgreSQl, MySQL, SQL Server, а также внешних систем полнотекстового поиска Sphinx и Elasticsearch. Приведены результаты тестирования на подготовленных данных.

Библиография

[1] Официальная документация по полнотекстовому поиску PostgreSQL [Электронный ресурс]. - Режим доступа: https://www.postgresql.org/docs/current/textsearch.html

[2] Официальная документация по полнотекстовому поиску MySQL [Электронный ресурс]. - Режим доступа: https://dev.mysql.com/doc/refman/8.0/en/fulltext-search.html

[3] Официальная документация по полнотекстовому поиску Microsoft SQL Server [Электронный ресурс]. - Режим доступа: https://docs.microsoft.com/ru-ru/sql/relational-databases/search/full-text-search?view=sql-server-ver16

[4] Официальная документация Sphinx [Электронный ресурс]. - Режим доступа: http://sphinxsearch.com/docs/current.html

[5] Официальная документация Elasticsearch [Электронный ресурс]. - Режим доступа: https://www.elastic.co/guide/index.html

[6] Виноградова М.В. Обзор системы полнотекстового поиска в постреляционной базе данных PostgreSQL [Электронный ресурс]. - Режим доступа: https://cyberleninka.ru/article/n/obzor-sistemy-polnotekstovogo-poiska-v-postrelyatsionnoy-baze-dannyh-postgresql/viewer

[7] Бартунов О., Сигаев, Ф. Введение в полнотекстовый поиск в PostgreSQL [Электронный ресурс]. - Режим доступа: http://citforum.ru/database/postgres/fts/

[8] Полнотекстовый поиск (Full-Text Search) в Microsoft SQL Server [Электронный ресурс]. - Режим доступа: https://info-comp.ru/sisadminst/486-full-text-search-ms-sql-server.html

[9] Виноградова М.В. Обзор системы полнотекстового поиска в постреляционной базе данных PostgreSQL [Электронный ресурс]. - Режим доступа: https://cyberleninka.ru/article/n/obzor-sistemy-polnotekstovogo-poiska-v-postrelyatsionnoy-baze-dannyh-postgresql/viewer

[10] Адаманский А. Обзор методов и алгоритмов полнотекстового поиска. [Электронный ресурс]. - Режим доступа: http://grusha-store.narod.ru/olderfiles/1/Obzor_metodov_polnotekstovogo_poiska.pdf

[11] Рязанова Н.Ю. Анализ вопросов автоматизации поиска информации [Электронный ресурс]. - Режим доступа: https://cyberleninka.ru/article/n/analiz-voprosov-avtomatizatsii-poiska-informatsii

Creation of an Evaluation Stand for a Comparative Analysis of the Functionality of Full-text Database Search Mechanisms in the Yandex Cloud Infrastructure

A.I. Romanenko

Abstract—Nowadays, when the amount of information on the Internet, databases and electronic documents is growing every day, the search for the necessary information is becoming more and more complex and time-consuming. Full-text search helps to solve this problem, allowing users to quickly and efficiently find the information they need. Full-text search is one of the most common and effective ways to search for information on the Internet. It is used in various fields, including databases, email, websites and other sources of information. For example, in large corporate databases, full-text search can be used to quickly find information about customers, products or services. In email, a full-text search can help the user quickly find all messages that contain certain keywords or phrases. On websites, full-text search can be used to find information about products, services or articles.

This article demonstrates the formation of an evaluation stand for a comparative analysis of the functionality of full-text search mechanisms in various databases using the Yandex Cloud infrastructure. For comparative analysis, full-text search mechanisms built into PostgreSQL, MySQL, Microsoft SQL Server DBMS, as well as external full-text search systems Sphinx and Elasticsearch were selected. A detailed scheme for installing and configuring the selected mechanisms in the cloud infrastructure is described, as well as the results of testing on the prepared data.

Keywords— full-text search, Sphinx, Elasticsearch, Yandex Cloud.

REFERENCES

[1] Official documentation on PostgreSQL full-text search [Electronic resource]. - Access mode: https://www.postgresql.org/docs/current/textsearch.html
[2] Official MySQL Full-text Search Documentation [Electronic resource]. - Access mode: https://dev.mysql.com/doc/refman/8.0/en/fulltext-search.html
[3] Official documentation on Microsoft SQL Server full-text search [Electronic resource]. - Access mode: https://docs.microsoft.com/ru-ru/sql/relational-databases/search/full-text-search ?view=sql-server-ver16
[4] Official Sphinx Documentation [Electronic resource]. - Access mode: http://sphinxsearch.com/docs/current.html
[5] Official Elasticsearch documentation [Electronic resource]. - Access mode: https://www.elastic.co/guide/index.html
[6] Vinogradova M.V. Review of the full-text search system in the post-relational database PostgreSQL [Electronic resource]. - Access mode: https://cyberleninka.ru/article/n/obzor-sistemy-polnotekstovogo-poiska-v-postrelyatsionnoy-baze-dannyh-postgresql/viewer
[7] Bartunov O., Sigaev, F. Introduction to full-text search in PostgreSQL [Electronic resource]. - Access mode: http://citforum.ru/database/postgres/fts /
[8] Full-Text search in Microsoft SQL Server [Electronic resource]. - Access mode: https://info-comp.ru/sisadminst/486-full-text-search-ms-sql-server.html
[9] Vinogradova M.V. Review of the full-text search system in the post-relational database PostgreSQL [Electronic resource]. - Access mode: https://cyberleninka.ru/article/n/obzor-sistemy-polnotekstovogo-poiska-v-postrelyatsionnoy-baze-dannyh-postgresql/viewer
[10] Adamansky A. Review of methods and algorithms of full-text search. [electronic resource]. - Access mode: http://grusha-store.narod.ru/olderfiles/1/Obzor_metodov_polnotekstovogo_poiska.pdf
[11] Ryazanova N.Yu. Analysis of automation issues of information retrieval [Electronic resource]. - Access mode: https://cyberleninka.ru/article/n/analiz-voprosov-avtomatizatsii-poiska-informatsii