Практикум на ЭВМ 5 семестр каф. СП

Общие сведения

Программное обеспечение

Для выполнения заданий практикума предлагается использовать СУБД PostgreSQL 11.5, дистрибутив которой можно скачать по ссылке:

https://www.postgresql.org/download/

В результате установки СУБД PostgreSQL должна быть установлена среда pgAdmin 4 для подключения к СУБД и выполнения SQL-запросов. Если этого не произошло, то ее необходимо установить отдельно, скачав по ссылке:

https://www.pgadmin.org/download/

СУБД PostgreSQL является бесплатным и свободно распространяющимся ПО, однако существуют также производные коммерческие разработки. К таким можно отнести СУБД PostgreSQL Pro. Для нее существует достаточно подробная документация на русском языке, которая также актуальна для СУБД PostgreSQL, так как СУБД PostgreSQL Pro является ее расширением и наследует всю функциональность:

https://postgrespro.ru/docs/postgrespro/11/index

Также можно воспользоваться англоязычной документацией для СУБД PostgreSQL:

https://www.postgresql.org/docs/11/index.html

Порядок сдачи и сроки

Практические задания предполагают предварительную теоретическую и практическую подготовку студента, которую он демонстрирует в процессе сдачи задания. Кроме того в процессе сдачи задания студент может получить дополнительные вопросы, ответы на которые необходимо дать до окончания занятия.

Практические задания могут быть с оценкой и без. Так практические задания 1, 2.1 и 3.1 предполагают сдачу в режиме зачет/незачет. Задания 2.2, 2.3, 2.4, 3.2, 3.3, 3.4 оцениваются по следующему принципу:

- Если домашняя часть практического задания признается преподавателем не выполненной полностью, и студент не успевает устранить недочеты до конца занятия, то студент отправляется на пересдачу.
- Если домашняя часть практического задания выполнена, однако студент не справился в должной мере с дополнительными вопросами, то ставится оценка -1 балл.
- Если домашняя часть практического задания выполнена и студент справился с дополнительными вопросами, то он получает 0 баллов.
- Если студент получил 0 баллов и хочет получить +1 балл, то он может попробовать ответить на несколько блиц вопросов, однако оценка может не измениться, а также уменьшиться до -1 балла, если преподаватель обнаружит грубую ошибку в рассуждениях студента.

Задания сдаются в порядке очереди. За одно занятие допускается сдача только одного задания. Записаться на сдачу можно в течение 15 минут от начала занятия, при этом необходимо личное присутствие. Прием задания должен начинаться не позднее, чем за 20 минут до окончания занятия, чтобы оставалось время на дополнительные вопросы.

Время, отведенное на сдачу каждого практического задания – одно занятие, однако для заданий 1, 2.1 и 3.1 может быть сделано исключение, если студент пытался, но не смог сдать с первого раза, в этом случае ему предоставляется еще одно занятие. Для записавшихся, но не успевших сдать задание студентов также предоставляется еще одно занятие. Если студент без уважительной причины пропустил занятие или не смог сдать задание до конца занятия, то он получает -1 балл и пробует сдать задание на следующем занятии. В этом случае график сдачи студентом практических заданий сдвигается на неделю вперед. Уважительность той или иной причины пропуска занятия рекомендуется заранее обсуждать с преподавателем.

Если студент имеет -6 баллов, то штрафные баллы больше не начисляются, однако оценка каждого последующего практического задания может быть только 0 баллов, иначе студент отправляется на пересдачу задания.

Практические задания выполняются по индивидуальным вариантам. Вариант содержит в себе предметную область для практических заданий 2.*, 3.*:

http://sp.cs.msu.ru/prak3/prak3_tasks.pdf

Узнать номер своего варианта можно в таблице успеваемости, где будут проставляться оценки и штрафные баллы за каждое практическое задание:

https://drive.google.com/open?id=163DkBrNsw4cXuJVwY07dO3s88TN96y_0LlpmI-pz-Bo

Просьба следить за своими оценками и сообщать преподавателям в случае обнаружения ошибок.

Итоговая оценка

Практикум состоит из 9 практических заданий, 3 из которых не оцениваются, а 6 заданий оцениваются от -1 до 1. Форма отчетности – зачет с оценкой. Для получения положительной оценки необходимо сдать все задания. По результатам выполнения практических заданий в конце семестра выставляется оценка:

«Удовлетворительно», если набрано от -6 до -3 баллов включительно.

«Хорошо», если набрано от -2 до 0 баллов включительно.

«Отлично», если набрано 1 и более баллов.

Практическое задание №1. Введение в SQL

Постановка задачи

Первое практическое задание заключается в знакомстве со средой pgAdmin и написании SQL-запросов с использованием оператора SELECT.

Для модельной базы данных должны быть составлены 4 произвольных SELECT-запроса, демонстрирующие полученные знания. Запросы должны охватывать проработанные темы. После составления запросов следует убедиться в их правильности при помощи более простых запросов. Дополнительные вопросы могут заключаться в построении более сложных запросов или объяснении работы подготовленных заданий.

Скрипт для создания и заполнения модельной базы данных и её описание:

https://postgrespro.ru/education/demodb

Темы для проработки

Основные понятия реляционных и SQL-ориентированных баз данных.

- Синтаксис SQL
  https://postgrespro.ru/docs/postgrespro/11/sql-syntax
- SELECT-запросы
  https://postgrespro.ru/docs/postgrespro/11/queries
- Функции и операторы
  https://postgrespro.ru/docs/postgrespro/11/functions
- Оконные функции
  https://postgrespro.ru/docs/postgrespro/9.5/tutorial-window
- Полное описание синтаксиса встретившихся команд
  https://postgrespro.ru/docs/postgrespro/11/sql-commands
- Оператор With и рекурсивные запросы
  https://postgrespro.ru/docs/postgrespro/11/queries-with
- Работа в среде pgAdmin.

Примеры вопросов

- Объяснить, как работают написанные запросы.
- Рассказать про операцию соединения (JOIN) и различные её разновидности.
- Рассказать про агрегатные функции, предложения GROUP BY и HAVING.
- Как выбрать только уникальные значения какого-либо столбца?
- Как осуществить сортировку по возрастанию/убыванию по значению какого-либо столбца?
- Как агрегатные функции ведут себя по отношению к неопределённым значениям?
- Рассказать о теоретико-множественных операциях в SQL.
- Чем отличаются UNION и UNION ALL?
- Чем отличаются COUNT(*) и COUNT(field)?
- Как подсчитать количество уникальных значений столбца?
- Как можно осуществить проверку на неопределенное значение?
- Рассказать про предикат LIKE.
- Как можно выбрать только определенное количество строк?
- Чем SQL-таблица отличается от отношения?

- Исправить неверно работающий запрос (запросы).
- Упростить один или несколько запросов.
- Округлить результирующее значение до 3 знаков после точки.
- Округлить вещественное число до целого без нулей после точки.
- Переписать запрос, не используя функцию MAX (MIN).
- Изменить формат вывода данных (например, формат даты и времени).
- Написать или модифицировать запрос по сформулированному заданию.

Практические задания № 2.* Создание БД для приложения

Одним из основных применений РСУБД является хранение и обработка данных небольших пользовательских приложений. Специфика такого использования заключается в том, что данные постоянно добавляются/изменяются/удаляются. При этом данных сравнительно немного. Схема данных достаточно устойчива и редко изменяется. В этом случае непосредственно с СУБД взаимодействует не человек, а программа, что уменьшает требования к способу взаимодействия с базой данных. Однако разработчику необходимо уметь проектировать гибкую и эффективную схему данных, использовать ограничения целостности, манипулировать данными и понимать механизмы поддержки согласованности базы данных, такие как транзакции и триггеры.

Блок практических заданий 2.1-2.4 призван сформировать у студента понимание особенностей хранения данных приложения в РСУБД и умение это хранение настраивать и поддерживать.

Практическое задание №2.1 Проектирование схемы базы данных

Постановка задачи

Второе практическое задание связано с проектированием схемы базы данных для работы приложения (WEB/Mobile/Desktop). Каждый индивидуальный вариант содержит предметную область, из которой должна быть проектируемая база данных. Задачей студента является решить, для чего будет использоваться создаваемая база данных, и, исходя из этого, построить её концептуальную схему. Результатом данного практического задания является схема базы данных (в виде ER-диаграммы, содержащей таблицы и связи между ними, без уточнения типов столбцов). При сдаче задания студент должен обосновать соответствие созданной схемы поставленной задаче.

Для проектирования схемы и построения диаграммы можно использовать любые средства, один из вариантов использовать сайт:

https://www.lucidchart.com/pages/examples/er-diagram-tool

Темы для проработки

- Модель "сущность-связь" (ER-модель).
- Первичные и внешние ключи.
- Типы связей и их моделирование.
- Нормальные формы и нормализация.

Требования к схеме

- Схема должна соответствовать поставленной задаче.
- Связи между сущностями должны быть правильно смоделированы.
- Таблицы должны удовлетворять, по крайней мере, третьей нормальной форме.
- Желательно придерживаться какой-либо системы в именовании таблиц и столбцов.

Практическое задание №2.2 Создание и заполнение таблиц

Постановка задачи

Третье практическое задание заключается в подготовке SQL-скрипта для создания таблиц согласно схеме, полученной в предыдущем задании (с уточнением типов столбцов). Необходимо определить первичные и внешние ключи, а также декларативные ограничения целостности (возможность принимать неопределенное значение, уникальные ключи, проверочные ограничения и т. д.). Таблицы следует создавать в отдельной базе данных. Кроме того, нужно подготовить данные для заполнения созданных таблиц. Объем подготовленных данных должен составлять не менее 10 экземпляров для каждой из стержневых сущностей и 20 экземпляров для каждой из ассоциативных. На основе этих данных необходимо создать SQL-скрипт для вставки соответствующих строк в таблицы БД.

Темы для проработки

- Язык DDL, операторы CREATE TABLE и ALTER TABLE.
  https://postgrespro.ru/docs/postgrespro/11/ddl-basics
  https://postgrespro.ru/docs/postgrespro/11/ddl-default
  https://postgrespro.ru/docs/postgrespro/11/ddl-alter
- Типы данных.
  https://postgrespro.ru/docs/postgrespro/11/datatype
- Декларативные ограничения целостности.
  https://postgrespro.ru/docs/postgrespro/11/ddl-constraints
- Оператор INSERT.
  https://postgrespro.ru/docs/postgrespro/11/dml-insert
  https://postgrespro.ru/docs/postgrespro/11/dml-returning
- Полное описание синтаксиса встретившихся команд
  https://postgrespro.ru/docs/postgrespro/11/sql-commands

Примеры вопросов

- Объяснить, что делают написанные запросы.
- В чем различие типов CHAR и VARCHAR? VARCHAR и TEXT?
- Что такое внешний ключ?
- Какие существуют способы поддержания ссылочной целостности?
- Что такое уникальный ключ?
- Что такое SERIAL?
- Рассказать о значениях по умолчанию и неопределенных значениях.
- Как можно хранить даты и время?
- Рассказать о числовых типах данных.
- Каким образом можно вставить несколько строк с помощью одного оператора INSERT?

- Как ведет себя оператор INSERT, если в списке столбцов перечислены не все столбцы?
- Добавить какие-либо ограничения целостности.
- Добавить SERIAL.
- Исправить выявленные при проверке недочеты.

Практическое задание №2.3. Операторы манипулирования

Постановка задачи

Четвертое практическое задание посвящено манипулированию данными с помощью операторов SQL. В ходе выполнения четвертого практического задания необходимо:

- Нужно подготовить 3-4 выборки, которые имеют осмысленное значение для предметной области, и также составить для них SQL-скрипты.
- Сформулировать 3-4 запроса на изменение и удаление из базы данных. Запросы должны быть сформулированы в терминах предметной области. Среди запросов обязательно должны быть такие, которые будут вызывать срабатывание ограничений целостности. Составить SQL-скрипты для выполнения этих запросов.

Темы для проработки

- Оператор SELECT.
  https://postgrespro.ru/docs/postgrespro/11/queries
- Оператор UPDATE.
  https://postgrespro.ru/docs/postgrespro/11/dml-update
- Оператор DELETE.
  https://postgrespro.ru/docs/postgrespro/11/dml-delete
- Декларативные ограничения целостности.
  https://postgrespro.ru/docs/postgrespro/11/ddl-constraints
- Полное описание синтаксиса встретившихся команд
  https://postgrespro.ru/docs/postgrespro/11/sql-commands

Примеры вопросов

- Объяснить, как работают написанные запросы.
- Примеры вопросов по оператору SELECT см. в задании №1.
- Исправить неверно работающий запрос (запросы).
- Упростить один или несколько запросов.
- Написать или модифицировать запрос по сформулированному заданию.

Практическое задание №2.4 Контроль целостности данных

Постановка задачи

Пятое практическое задание посвящено контролю целостности данных, который производится с помощью механизма транзакций и триггеров. Транзакции позволяют рассматривать группу операций как единое целое, либо отрабатывают все операции, либо ни одной. Это позволяет избегать несогласованности данных. Триггеры позволяют проверять целостность данных в момент выполнения транзакций, поддерживать целостность, внося изменения, и откатывать транзакции, приводящие к потере целостности.

Необходимо подготовить SQL-скрипты для проверки наличия аномалий (потерянных изменений, грязных чтений, неповторяющихся чтений, фантомов) при параллельном исполнении транзакций на различных уровнях изолированности SQL/92 (READ UNCOMMITTED, READ COMMITTED, REPEATABLE READ, SERIALIZABLE). Подготовленные скрипты должны работать с одной из таблиц, созданных в практическом задании №2.1. Для проверки наличия аномалий потребуются два параллельных сеанса, операторы в которых выполняются пошагово:

- Установить в обоих сеансах уровень изоляции READ UNCOMMITTED. Выполнить сценарии проверки наличия аномалий потерянных изменений и грязных чтений.
- Установить в обоих сеансах уровень изоляции READ COMMITTED. Выполнить сценарии проверки наличия аномалий грязных чтений и неповторяющихся чтений.
- Установить в обоих сеансах уровень изоляции REPEATABLE READ. Выполнить сценарии проверки наличия аномалий неповторяющихся чтений и фантомов.
- Установить в обоих сеансах уровень изоляции SERIALIZABLE. Выполнить сценарий проверки наличия фантомов.

Необходимо составить скрипт для создания триггера, а также подготовить несколько запросов для проверки и демонстрации его полезных свойств:

- Изменение данных для сохранения целостности.
- Проверка транзакций и их откат в случае нарушения целостности.

Темы для проработки

- Понятие транзакции, свойства транзакций.
  https://postgrespro.ru/docs/postgrespro/11/tutorial-transactions
- Уровни изолированности и аномалии
  https://postgrespro.ru/docs/postgrespro/11/transaction-iso
- Триггеры и триггерные функции
  https://postgrespro.ru/docs/postgrespro/11/trigger-definition
  https://postgrespro.ru/docs/postgrespro/11/plpgsql-trigger
- Сообщения и ошибки
  https://postgrespro.ru/docs/postgrespro/11/plpgsql-errors-and-messages
- Полное описание синтаксиса встретившихся команд
  https://postgrespro.ru/docs/postgrespro/11/sql-commands

Примеры вопросов

- Рассказать об аномалиях доступа к БД.
- Перечислить аномалии, возникающие на каждом из уровней изолированности.
- Рассказать о свойствах транзакций.
- Рассказать об управлении транзакциями.
- Что такое тупики? Как бороться с тупиками?
- На каком уровне изолированности возможны тупики?
- Как обеспечивается изолированность транзакций в СУБД?
- Как бороться с проблемой фантомов?
- Что такое журнал транзакций?
- Как обеспечивается постоянство хранения (durability) в СУБД?

- Объяснить принцип работы написанного триггера.
- Какие бывают типы триггеров?
- Когда может срабатывать триггер?
- В каком порядке срабатывают триггеры?
- Можно ли менять порядок срабатывания триггеров?
- Сработает ли триггер, если оператор, выполненный пользователем, не затрагивает ни одну строку таблицы?
- Продемонстрировать откат транзакции при возникновении ошибок.
- Продемонстрировать возникновение тупика.
- Исправить неверные сценарии проверки аномалий
- Исправить ошибки в работе триггера.
- Модифицировать триггер каким-либо образом.

Практические задания № 3.* Создание БД для аналитики

РСУБД часто используются в аналитических целях. В этом случае объем данных постоянно увеличивается. При этом их изменение или удаление затрудненно из-за большого объема. Хранение данных в денормализованном виде с помощью массивов и json-формата упрощает их обработку. В этом случае с СУБД взаимодействует человек-аналитик, который может иметь доступ не ко всем данным. Ему может быть неудобно работать с большими SELECT-запросами. Для него сложно заранее предугадать все задачи, которые перед ним будут стоять. При этом разработчику такой базы данных необходимо уметь проектировать высокопроизводительную схему данных ориентированную на большие объемы и регулярно появляющиеся и устаревающие данные, использовать механизмы секционирования и построения индексов, анализировать планы запросов и оптимизировать их, упрощать интерфейс базы данных с помощью процедур и представлений, ограничивать доступ с помощью ролей и прав.

Блок практических заданий 3.1-3.4 призван сформировать у студента понимание особенностей создания аналитических баз данных и умение их настраивать и поддерживать.

Практическое задание №3.1 Проектирование схемы базы данных

Постановка задачи

Шестое практическое задание связано с проектированием схемы базы данных для аналитики. Будем исходить из того, что приложение, для которого была сделана база данных в задании №2.1, стало очень популярным и по нему каждый день можно собирать большой объем статистической информации. Что это будет за статистика? Почему именно ее необходимо собирать, обрабатывать и анализировать? Задачей студента является ответить на эти вопросы, и, исходя из этого, разработать базу данных и заполнить ее данными. Результатом данного практического задания является схема базы данных, скрипты создания базы данных и ее заполнения, обладающие следующими свойствами:

- Как минимум одна таблица должна содержать не меньше 100 млн. записей, которые со временем теряют актуальность.
- Другая таблица, связанная с первой, должна содержать не меньше 1 млн. записей.

- В одной из таблиц с количество записей больше 1 млн. должна быть колонка с текстом, по которой будет необходимо настроить полнотекстовый поиск.
- В одной из таблиц с количество записей больше 1 млн. должна быть колонка с данными в json-формате.
- В одной из таблиц с количество записей больше 1 млн. должна быть колонка с массивом.

При выполнении задания важно учитывать плюсы и минусы денормализации схемы данных и использования массивов и json-формата. При сдаче задания студент должен обосновать соответствие созданной схемы поставленной задаче.

Для проектирования схемы и построения диаграммы можно использовать любые средства, один из вариантов использовать сайт:

https://www.lucidchart.com/pages/examples/er-diagram-tool

Темы для проработки

- Денормализация
  https://habr.com/ru/company/latera/blog/281262/
  https://habr.com/ru/post/64524/
- Массивы
  https://postgrespro.ru/docs/postgrespro/11/arrays
  https://postgrespro.ru/docs/postgrespro/11/functions-array
- Json
  https://postgrespro.ru/docs/postgrespro/11/datatype-json
  https://postgrespro.ru/docs/postgrespro/11/functions-json
- Наполнение базы данных
  https://postgrespro.ru/docs/postgrespro/11/populate

Практическое задание №3.2 Управление доступом

Постановка задачи

Целью седьмого практического задания является освоение работы с представлениями и другими способами управления доступом. При выполнении задания необходимо:

- Создать пользователя test и выдать ему доступ к базе данных.
- Составить и выполнить скрипты присвоения новому пользователю прав доступа к таблицам, созданным в практическом задании №3.1. При этом права доступа к различным таблицам должны быть различными, а именно:
  - По крайней мере, для одной таблицы новому пользователю присваиваются права SELECT, INSERT, UPDATE в полном объеме.
  - По крайней мере, для одной таблицы новому пользователю присваиваются права SELECT и UPDATE только избранных столбцов.
  - По крайней мере, для одной таблицы новому пользователю присваивается только право SELECT.
- Присвоить новому пользователю право доступа (SELECT) к представлению, созданному в практическом задании №3.1

- Создать стандартную роль уровня базы данных, присвоить ей право доступа (UPDATE на некоторые столбцы) к представлению, созданному в практическом задании №3.3, назначить новому пользователю созданную роль.
- Выполнить от имени нового пользователя некоторые выборки из таблиц и представления. Убедиться в правильности контроля прав доступа.
- Выполнить от имени нового пользователя операторы изменения таблиц с ограниченными правами доступа. Убедиться в правильности контроля прав доступа.
- Составить SQL-скрипты для создания нескольких представлений, которые позволяли бы упростить манипуляции с данными или позволяли бы ограничить доступ к данным, предоставляя только необходимую информацию.

Темы для проработки

- Роли и пользователи.
  https://postgrespro.ru/docs/postgresql/11/user-manag
  https://postgrespro.ru/docs/postgrespro/11/app-createuser
- Директивы GRANT и REVOKE.
  https://postgrespro.ru/docs/postgrespro/11/ddl-priv
  https://postgrespro.ru/docs/postgrespro/11/ddl-schemas#DDL-SCHEMAS-PRIV
- Представления.
  https://postgrespro.ru/docs/postgrespro/11/sql-createview
  https://postgrespro.ru/docs/postgrespro/11/rules-views
  https://postgrespro.ru/docs/postgrespro/11/rules-materializedviews
- Полное описание синтаксиса встретившихся команд
  https://postgrespro.ru/docs/postgrespro/11/sql-commands

Примеры вопросов

- Для чего нужны роли?
- Что такое схема?
- Рассказать про директивы GRANT и REVOKE.
- Для чего нужна роль PUBLIC?
- Как добавить нового пользователя в текущую базу данных?
- Как позволить пользователю заходить на сервер?
- Какие существуют права?
- Исправить ошибки в обязательной части.
- Сменить владельца базы данных.
- Сменить пароль для пользователя.
- Определить роль с заданными правами.
- Объяснить, как работают написанные запросы.
- Рассказать о CHECK OPTION.
- Рассказать о модификации данных через представления.
- Рассказать о вставке данных через представления.
- Примеры вопросов по оператору SELECT см. в задании №1.
- Исправить неверно работающий запрос (запросы).
- Упростить один или несколько запросов.
- Продемонстрировать изменение и вставку данных через представления.

- Написать или модифицировать запрос по сформулированному заданию.
- Продемонстрировать полезность материализованного представления.

Практическое задание №3.3 Функции и язык PL/pgSQL

Постановка задачи

Восьмое практическое задание посвящено упрощению работы аналитика с помощью создания и использования функций. При выполнении задания необходимо:

- Составить SQL-скрипты для создания нескольких функций, упрощающих манипуляции с данными.
- Продемонстрировать полученные знания о возможностях языка PL/pgSQL. В скриптах должны использоваться:
    - Циклы.
    - Ветвления.
    - Переменные.
    - Курсоры.
    - Исключения.
- Обосновать преимущества механизма функций перед механизмом представлений.

Темы для проработки

- Функции.
  https://postgrespro.ru/docs/postgrespro/11/xfunc-sql
- PL/PgSQL
  https://postgrespro.ru/docs/postgrespro/11/plpgsql
- Основные операторы
  https://postgrespro.ru/docs/postgrespro/11/plpgsql-statements
- Управляющие структуры
  https://postgrespro.ru/docs/postgrespro/11/plpgsql-control-structures
- Курсоры
  https://postgrespro.ru/docs/postgrespro/11/plpgsql-cursors
- Полное описание синтаксиса встретившихся команд
  https://postgrespro.ru/docs/postgrespro/11/sql-commands

Примеры вопросов

- Объяснить, как работают написанные запросы.
- Примеры вопросов по оператору SELECT см. в задании №1.
- Исправить неверно работающий запрос (запросы).
- Упростить один или несколько запросов.
- Написать или модифицировать запрос по сформулированному заданию.
- Описать в каких случаях целесообразно создавать функции.
- Рассказать о курсорах, как и зачем используются.
- Рассказать о работе с циклами

Практическое задание №3.4 Индексы

Постановка задачи

Девятое практическое задание посвящено ускорению выполнения запросов. Для этого могут быть использованы механизмы секционирования, наследования и индексов. Для выполнения задания необходим достаточно большой объем данных, чтобы оптимизация была целесообразной (порядка 1 млн. строк в каждой таблице). Необходимо подготовить два запроса:

- Запрос к одной таблице, содержащий фильтрацию по нескольким полям.
- Запрос к нескольким связанным таблицам, содержащий фильтрацию по нескольким полям.

Для каждого из этих запросов необходимо провести следующие шаги:

• Получить план выполнения запроса без использования индексов.

- Получить статистику (IO и Time) выполнения запроса без использования индексов.
- Создать нужные индексы, позволяющие ускорить запрос.
- Получить план выполнения запроса с использованием индексов и сравнить с первоначальным планом.
- Получить статистику выполнения запроса с использованием индексов и сравнить с первоначальной статистикой.
- Оценить эффективность выполнения оптимизированного запроса.

Также необходимо продемонстрировать полезность индексов для организации полнотекстового поиска, фильтрации с использованием массива и json-формата.

Для таблицы объемом больше 100 млн. записей произвести оптимизацию, позволяющую быстро удалять старые данные, ускорить вставку и чтение данных.

Темы для проработки

- EXPLAIN
  https://postgrespro.ru/docs/postgrespro/11/using-explain
  https://postgrespro.ru/docs/postgrespro/11/planner-stats
  https://postgrespro.ru/docs/postgrespro/11/explicit-joins
- ANALYZE
  https://postgrespro.ru/docs/postgrespro/11/routine-vacuuming#VACUUM-FOR-STATISTICS
- Индексы
  https://postgrespro.ru/docs/postgrespro/11/indexes
- Полнотекстовый поиск
  https://postgrespro.ru/docs/postgrespro/11/textsearch
- Наследование таблиц
  https://postgrespro.ru/docs/postgrespro/11/ddl-inherit
- Секционирование таблиц
  https://postgrespro.ru/docs/postgrespro/11/ddl-partitioning
- Полное описание синтаксиса встретившихся команд
  https://postgrespro.ru/docs/postgrespro/11/sql-commands

Примеры вопросов

- В чем отличие первичного ключа и уникального индекса?

- В каких случаях имеет смысл создавать индексы? Какие колонки следует включать в индекс и почему?
- Какие существуют способы внутренней организации индексов?
- Рассказать о проблеме фрагментации индексов. Как бороться с фрагментацией?
- Имеет ли значение порядок указания колонок при создании индекса?
- В чем разница между Index Scan и Index Seek?
- В чем разница между секционированием и наследованием?
- Зачем нужен ANALYZE?
- Исправить ошибки в подготовленных выборках.
- Могут ли индексы ухудшить производительность? Если да, то продемонстрировать это.
- На что влияет порядок сортировки (ASC\DESC) при создании индекса? Продемонстрировать это.
- Продемонстрировать полезность индекса по выражению.
- Продемонстрировать полезность частичного индекса.