Практикум СУБД ВШПИ

Общие сведения

Программное обеспечение

Для выполнения заданий практикума предлагается использовать СУБД PostgreSQL

После установки/развертывания СУБД PostgreSQL можно использовать встроенные клиенты pgAdmin 4 (GUI), psql (cmd) или сторонние клиенты, например, DataGrip от JetBrains (GUI).

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/14/index

Полезный ресурс по СУБД Postgres:
https://www.postgresqltutorial.com

Также можно воспользоваться англоязычной документацией для СУБД PostgreSQL:
https://www.postgresql.org/docs/14/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.
Сроки сдачи работ: 1-ая часть 15.02-15.03, 2-ая часть 16.03-15.04, 3-я часть 16.04-15.05

За одно занятие допускается сдача не более трех заданий.

Практические задания выполняются по индивидуальным вариантам. Вариант содержит в себе предметную область для практических заданий 1.*, 2.*:
Файл вариантов <Варианты заданий.pdf>.

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

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

Итоговая оценка

Для получения положительной оценки необходимо сдать все задания. По результатам выполнения практических заданий в конце семестра выставляется оценка.

Практические задания № 1.* Создание БД для приложения

Одним из основных применений РСУБД является хранение и обработка данных пользовательских приложений. Специфика такого использования заключается в том, что данные постоянно добавляются/изменяются/удаляются. 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 1.1-1.4 призван сформировать у студента понимание особенностей хранения, умение настраивать и поддерживать данные.

Практическое задание №1.1 Проектирование схемы базы данных

Постановка задачи

Практическое задание связано с проектированием схемы базы данных для работы приложения. 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 (т.е. каждая таблица должна состоять из ключа и других взаимно независимых атрибутов).

Желательно придерживаться какой-либо системы в именовании таблиц и столбцов.

Практическое задание №1.2 Создание и заполнение таблиц

Постановка задачи

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 (неподчиняющиеся основные сущности) сущностей и 15 экземпляров для каждой из ассоциативных (подчиненные многие-ко-многим). 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/14/ddl-basics
  https://postgrespro.ru/docs/postgrespro/14/ddl-default
  https://postgrespro.ru/docs/postgrespro/14/ddl-alter
- Типы данных.
  https://postgrespro.ru/docs/postgrespro/14/datatype
- Декларативные ограничения целостности.
  https://postgrespro.ru/docs/postgrespro/14/ddl-constraints
- Оператор INSERT.
  https://postgrespro.ru/docs/postgrespro/14/dml-insert
  https://postgrespro.ru/docs/postgrespro/14/dml-returning
- Полное описание синтаксиса встретившихся команд
  https://postgrespro.ru/docs/postgrespro/14/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 №1.3. Операторы манипулирования

Постановка задачи

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. Составить SQL-скрипты для выполнения этих запросов.

Темы для проработки

- Синтаксис SQL
  https://postgrespro.ru/docs/postgrespro/14/sql-syntax
- Оператор SELECT.
  https://postgrespro.ru/docs/postgrespro/14/queries
- Оператор With и рекурсивные запросы
  https://postgrespro.ru/docs/postgrespro/14/queries-with
- Оператор UPDATE.
  https://postgrespro.ru/docs/postgrespro/14/dml-update
- Оператор DELETE.
  https://postgrespro.ru/docs/postgrespro/14/dml-delete
- Функции и операторы
  https://postgrespro.ru/docs/postgrespro/14/functions
- Оконные функции
  https://postgrespro.ru/docs/postgrespro/14/tutorial-window
- Полное описание синтаксиса встретившихся команд
  https://postgrespro.ru/docs/postgrespro/14/sql-commands

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.

Практическое задание №1.4 Контроль целостности данных

Постановка задачи

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/14/tutorial-transactions
- Уровни изолированности и аномалии
  https://postgrespro.ru/docs/postgrespro/14/transaction-iso
- Триггеры и триггерные функции
  https://postgrespro.ru/docs/postgrespro/14/trigger-definition
  https://postgrespro.ru/docs/postgrespro/14/plpgsql-trigger
- Сообщения и ошибки
  https://postgrespro.ru/docs/postgrespro/14/plpgsql-errors-and-messages
- Полное описание синтаксиса встретившихся команд
  https://postgrespro.ru/docs/postgrespro/14/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 № 2.* Роли, права, хранимые процедуры

Разработчику 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 2.1-2.3 призван сформировать у студента понимание особенностей настроек прав доступа к базе данных и умение их настраивать и поддерживать, научиться работать с хранимыми процедурами.

Практическое задание №2.1 Управление доступом

Постановка задачи

Целью 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 №1.1. 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.
- 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.
- Присвоить новому пользователю право доступа (SELECT) к одному из представлений

- Создать стандартную роль уровня базы данных, присвоить ей право доступа (UPDATE на некоторые столбцы) к одному из представлений, назначить новому пользователю созданную роль.
- Выполнить от имени нового пользователя некоторые выборки из таблиц и представлений. Убедиться в правильности контроля прав доступа.
- Выполнить от имени нового пользователя операторы изменения таблиц с ограниченными правами доступа. Убедиться в правильности контроля прав доступа.

Темы для проработки

- Роли и пользователи.
  https://postgrespro.ru/docs/postgresql/14/user-manag
- https://postgrespro.ru/docs/postgrespro/14/app-createuser
- Директивы GRANT и REVOKE.
  https://postgrespro.ru/docs/postgrespro/14/ddl-priv
  https://postgrespro.ru/docs/postgrespro/14/ddl-schemas#DDL-SCHEMAS-PRIV
- Представления.
  https://postgrespro.ru/docs/postgrespro/14/sql-createview
  https://postgrespro.ru/docs/postgrespro/14/rules-views
  https://postgrespro.ru/docs/postgrespro/14/rules-materializedviews
- Полное описание синтаксиса встретившихся команд
  https://postgrespro.ru/docs/postgrespro/14/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.
- 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 №2.2 Функции и язык PL/pgSQL

Постановка задачи

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

- Составить SQL-скрипты для создания нескольких (2-3) функций, упрощающих работу 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/14/xfunc-sql
- PL/PgSQL https://postgrespro.ru/docs/postgrespro/14/plpgsql-errors-and-messages
- Основные операторы https://postgrespro.ru/docs/postgrespro/14/plpgsql-statements
- Управляющие структуры https://postgrespro.ru/docs/postgrespro/14/plpgsql-control-structures
- Курсоры https://postgrespro.ru/docs/postgrespro/14/plpgsql-cursors
- Полное описание синтаксиса встретившихся команд https://postgrespro.ru/docs/postgrespro/14/sql-commands

Примеры вопросов

- Описать в каких случаях целесообразно создавать функции.
- Рассказать о курсорах, как и зачем используются.
- Рассказать о работе с циклами