**Проект №10 — Базы данных.**

Привет, участник Школы 21!🤗 В этом проекте мы изучим базы данных, зачем они нужны, где их применять и как ими управлять. Наряду с этим мы познакомимся с декларативным языком программирования SQL, разберём типы баз данных и как происходит их тестирование. Приступаем!😎

**Instructions**

Вы уже дошли до предпоследнего проекта в нашем курсе, поздравляем! Изучение этого проекта поможет вам закрыть одну проблему на собеседовании — SQL. Хоть навык работы с реляционными базами данных и не так сильно связан с тестированием, он часто указан в вакансиях.

Напоминаем, что все созданные отчёты и файлы тебе нужно будет загрузить в папку src/ в корне проекта (обязательно в ветку *develop*).

**Contents**

1. [Chapter I](https://edu.21-school.ru/project/62673/task#chapter-i)  
   1.1. [Общая инструкция](https://edu.21-school.ru/project/62673/task#%D0%BE%D0%B1%D1%89%D0%B0%D1%8F-%D0%B8%D0%BD%D1%81%D1%82%D1%80%D1%83%D0%BA%D1%86%D0%B8%D1%8F)
2. [Chapter II](https://edu.21-school.ru/project/62673/task#chapter-ii)  
   2.1. [Базы данных](https://edu.21-school.ru/project/62673/task#%D0%B1%D0%B0%D0%B7%D1%8B-%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85)  
   2.2. [Задание №1](https://edu.21-school.ru/project/62673/task#%D0%B7%D0%B0%D0%B4%D0%B0%D0%BD%D0%B8%D0%B5-1-%D1%82%D0%B8%D0%BF%D1%8B-%D0%B1%D0%B0%D0%B7-%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85)  
   2.3. [Основы работы с СУБД](https://edu.21-school.ru/project/62673/task#%D0%BE%D1%81%D0%BD%D0%BE%D0%B2%D1%8B-%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D1%8B-%D1%81-%D1%81%D1%83%D0%B1%D0%B4)  
   2.4. [Веб-хранилища](https://edu.21-school.ru/project/62673/task#%D0%B2%D0%B5%D0%B1-%D1%85%D1%80%D0%B0%D0%BD%D0%B8%D0%BB%D0%B8%D1%89%D0%B0)  
   2.5. [Задание №2](https://edu.21-school.ru/project/62673/task#%D0%B7%D0%B0%D0%B4%D0%B0%D0%BD%D0%B8%D0%B5-2-%D0%B2%D0%B5%D0%B1-%D1%85%D1%80%D0%B0%D0%BD%D0%B8%D0%BB%D0%B8%D1%89%D0%B0-1)  
   2.6. [Задание №3](https://edu.21-school.ru/project/62673/task#%D0%B7%D0%B0%D0%B4%D0%B0%D0%BD%D0%B8%D0%B5-3-%D0%B2%D0%B5%D0%B1-%D1%85%D1%80%D0%B0%D0%BD%D0%B8%D0%BB%D0%B8%D1%89%D0%B0-2)  
   2.7. [Задание №4](https://edu.21-school.ru/project/62673/task#%D0%B7%D0%B0%D0%B4%D0%B0%D0%BD%D0%B8%D0%B5-4-%D1%81%D0%B2%D0%BE%D0%B9%D1%81%D1%82%D0%B2%D0%B0-%D0%B2%D0%B5%D0%B1-%D1%85%D1%80%D0%B0%D0%BD%D0%B8%D0%BB%D0%B8%D1%89)
3. [Chapter III](https://edu.21-school.ru/project/62673/task#chapter-iii)  
   3.1. [SQL](https://edu.21-school.ru/project/62673/task#sql)  
   3.2. [Задание №5](https://edu.21-school.ru/project/62673/task#%D0%B7%D0%B0%D0%B4%D0%B0%D0%BD%D0%B8%D0%B5-4-%D1%8F%D0%B7%D1%8B%D0%BA-sql)  
   3.3. [SQL-запросы](https://edu.21-school.ru/project/62673/task#sql-%D0%B7%D0%B0%D0%BF%D1%80%D0%BE%D1%81%D1%8B)  
   3.4. [Задание №6](https://edu.21-school.ru/project/62673/task#%D0%B7%D0%B0%D0%B4%D0%B0%D0%BD%D0%B8%D0%B5-6-%D1%81%D0%BE%D0%B7%D0%B4%D0%B0%D0%BD%D0%B8%D0%B5-sql-%D0%B7%D0%B0%D0%BF%D1%80%D0%BE%D1%81%D0%BE%D0%B2)
4. [Chapter IV](https://edu.21-school.ru/project/62673/task#chapter-iv)  
   4.1. [Тестирование базы данных](https://edu.21-school.ru/project/62673/task#%D1%82%D0%B5%D1%81%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5-%D0%B1%D0%B0%D0%B7%D1%8B-%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85)  
   4.2. [Классификация](https://edu.21-school.ru/project/62673/task#%D0%BA%D0%BB%D0%B0%D1%81%D1%81%D0%B8%D1%84%D0%B8%D0%BA%D0%B0%D1%86%D0%B8%D1%8F)  
   4.3. [Задание №7](https://edu.21-school.ru/project/62673/task#%D0%B7%D0%B0%D0%B4%D0%B0%D0%BD%D0%B8%D0%B5-7-%D0%BD%D0%B0%D0%B3%D1%80%D1%83%D0%B7%D0%BA%D0%B0-%D1%81%D1%82%D1%80%D0%B5%D1%81%D1%81-%D0%B8-%D0%BE%D1%82%D0%BA%D0%B0%D0%B7%D0%BE%D1%83%D1%81%D1%82%D0%BE%D0%B9%D1%87%D0%B8%D0%B2%D0%BE%D1%81%D1%82%D1%8C)

**Chapter I**

**Общая инструкция**

Методология Школы 21 может быть не похожа на тот образовательный опыт, который случался с тобой ранее. Её отличает высокий уровень автономии: у тебя есть задача, ты должен её выполнить. По большей части тебе нужно будет самому добывать знания для её решения. Второй важный момент — это peer-to-peer обучение. В образовательном процессе нет менторов и экспертов, перед которыми ты защищаешь свой результат. Ты это делаешь перед таким же учащимися, как и ты сам. У них есть чек-лист, который поможет им качественно выполнить приемку вашей работы.

Роль Школы 21 заключается в том, чтобы обеспечить через последовательность заданий и оптимальный уровень поддержки такую траекторию обучения, при которой ты не только освоишь hard skills, но и научишься самообучаться.

* Не доверяй слухам и предположениям о том, как должно быть оформлено ваше решение. Этот документ является единственным источником, к которому стоит обращаться по большинству вопросов;
* твое решение будет оцениваться другими учащимися;
* подлежат оцениванию только те файлы, которые ты выложил в GIT (ветка develop, папка src);
* в твоей папке не должно быть лишних файлов — только те, что были указаны в задании;
* не забывай, что у вас есть доступ к интернету и поисковым системам;
* обсуждение заданий можно вести и в Rocket.Chat;
* будь внимателен к примерам, указанным в этом документе — они могут иметь важные детали, которые не были оговорены другим способом;
* и да пребудет с тобой Сила!

**Chapter II**

**Базы данных**

Тестирование любого приложения включает в себя проверку работы системы на различных входных данных, получая их из разных хранилищ (из разных баз данных). В этом разделе мы и изучим, что такое база данных, и поймём схему работы с ней.

Начнём с вводного определения. **База данных** — это набор структурированной информации, которая хранится в электронном виде. Другими словами, база данных является местом, в котором приложение хранит свои данные. Корзина в интернет-магазине, списки услуг на различных сервисах, ваши личные данные аккаунтов из социальных сетей — всё это хранится в базах данных.

Всем ли приложениям обязательно иметь свою базу данных? Совсем нет. Например, если это сайт-лендинг, который лишь предоставляет информацию, или если это игровое приложение, где не требуется хранение результата вашей игры, то база данных тут не потребуется. Однако отсутствие базы данных зачастую говорит о том, что приложение не является масштабным, и тестирование такого приложения не требует большого количества усилий. В таком случае приложение хранит данные в памяти компьютера, телефона и иных технических средствах, что при большом объёме данных может замедлять работу приложения.

Как мы выяснили, память технического устройства — не лучшее место для хранения. Поэтому когда данных много, их нужно куда-то сложить. Можно писать в файлы, а можно сохранять информацию в базу данных (далее часто будем пользоваться аббревиатурой — БД).

**Задание №1. Типы баз данных**

Типов баз данных достаточно большое количество, и для их тестирования приложений стоит понимать основные особенности каждой из них. Создайте файл exercise1.md, в котором опишите следующие виды баз данных:

1. Простейшие базы данных:
   * иерархические;
   * сетевые.
2. Реляционные базы данных
3. Нереляционные базы данных:
   * на основе пар "ключ‑значение";
   * документная;
   * графовая;
   * колоночная;
   * временных рядов.
4. NewSQL базы данных
5. Многомодельные базы данных

Каждому типу дайте определение, укажите их особенности (преимущества и недостатки). К каждому типу приведите по 2-3 примера баз данных.

**Основы работы с СУБД**

При проектировании современных высоконагруженных систем зачастую говорят не о БД, а о системе управления базами данных (СУБД). СУБД – это программные средства, которые позволяют создавать структуру новой базы данных, наполнять её содержимым, редактировать и обрабатывать информацию внутри БД. СУБД предназначена для централизованного управления базой данных.

По степени универсальности различают два класса СУБД:

* системы общего назначения
* специализированные системы

*P.S.: обязательно изучите эти классы СУБД самостоятельно!*

Чтобы оценить производительность СУБД достаточно проверить:

* время выполнения запросов;
* скорость поиска информации в неиндексированных полях;
* время выполнения операций импортирования базы данных из других форматов;
* скорость создания индексов и выполнения таких массовых операций, как вставка, удаление данных;
* максимальное число параллельных обращений к данным в многопользовательском режиме.

Некоторые СУБД предусматривают средства обеспечения безопасности данных, которые позволяют:

* шифровать данные;
* защищать данные паролем;
* ограничить уровень доступа (к базе данных, к таблице, к словарю).

**Веб-хранилища**

Как мы уже говорили ранее, у приложений есть возможность хранить данные в памяти, в случае веб-приложений – это браузер, а конкретнее – веб-хранилища браузера (web-storage). В них легче хранить второстепенную информацию, например, пользовательские настройки (скажем, параметры, которые определяют способ отображения веб-страницы) и состояние приложения (снимок текущего состояния веб-приложения), чтобы посетитель мог продолжить его выполнение с того же самого места позже. Эта информация может быть кратковременной, которая удаляется после выключения браузера, или долговременной, которая остается доступной при последующих посещениях веб-страницы.

Сохраняемая в веб-хранилище информация в действительности сохраняется не в интернете, а на компьютере посетителя веб-страницы. Иными словами, веб-хранилище означает хранение данных не в интернете, а хранение данных из интернета.

**Задание №2. Веб-хранилища №1**

Создайте файл exercise2.md, в котором ответьте на следующие вопросы:

* Что такое Local Storage и Session Storage? В чём заключается их различие?
* Какие существуют основные методы взаимодействия с localStorage и sessionStorage через консоль в браузере? Как производить сохранение, получение, удаление данных, очистку хранилища?
* Что такое IndexedDB? Чем этот вид хранилища отличается от описанных выше?
* Как можно просмотреть содержимое всех веб-хранилищ в браузере?

**Задание №3. Веб-хранилища №2**

Изучите различные веб-приложения и сайты на предмет того, какие пользовательские данные они хранят в "Local Storage", "Session Storage" и "IndexedDB". Подбор веб-приложений производите на своё усмотрение, это могут быть социальные сети, интернет-магазины, различные сервисы и т.п. Создайте файл exercise3.md, опишите в нём по 5-7 объектов из "Local Storage" и "Session Storage" в виде таблицы: Ключ / Значение / Краткое описание того, как вы понимаете, с какой целью нужно хранить данный объект. Некоторые объекты могут содержать личные данные, поэтому при выполнении этого задания лучше их лишний раз не распространять 😁.

Когда вы опишете объекты веб-хранилища, попробуйте сами создать парочку! С помощью метода setItem(key, value) нужно добавить в sessionStorage и в localStorage по одному объекту. Для этого в DevTools необходимо открыть вкладку "Console", после чего ввести соответствующую команду. В тот же файл exercise3.md необходимо будет прикрепить 4 скриншота (не забудь добавить их в src!):

1. Результата выполнения команд для добавления и получения объекта в "Session Storage"
2. Состояния "Session Storage" после добавления объекта
3. Результата выполнения команд для добавления и получения объекта в "Local Storage"
4. Состояния "Local Storage" после добавления объекта

**Задание №4. Свойства веб-хранилищ**

Аналогично второй части предыдущего задания, давайте создадим по одному объекту в "Local Storage" и "Session Storage". Только теперь вам предстоит исследовать их при различных действиях с браузером, а именно проверить, сохраняются ли объекты при:

* обновлении страницы
* дублировании вкладки браузера
* восстановлении страницы (это можно сделать сочетанием клавиш "Ctrl + Shift + T")
* закрытии и открытии браузера (перезапуска браузера)

Создайте файл exercise4.md. Сопоставьте полученные результаты и представьте их в виде таблицы "Local Storage" / "Session Storage", где в каждой ячейке укажите "+" или "-" в соответствии с тем, сохраняется ли объект при определённом действии.

**Chapter III**

**SQL**

Стандартным языком реляционных СУБД является язык SQL (Structured Query Language) – это структурированный язык запросов, применяемый для создания, модификации и управления данными. Язык SQL позволяет находить среди большого объёма данных те записи, которые нужны для конкретных целей, сортировать и структурировать их.

Общение пользователя с базой данных не осуществляет напрямую и выглядит примерно так:

1. Пользователь в клиентской части приложения, которая используется для создания запроса, вводит определённые данные (например, в поисковую строку вводит какой-то текст).
2. Формируется запрос и передаётся на серверную часть приложение через API.
3. При помощи СУБД создаётся запрос к базе данных с определёнными фильтрами (это делают с помощью специальных команд — операторов).
4. После получения ответа от БД, на серверной части формируется ответ на пользовательский запрос и далее вы уже всё знаете!

**Задание №5. Язык SQL**

И снова теоретическое задание! Создайте файл exercise5.md. Опишите в нём:

* Что такое операторы в SQL?
* Каких типов бывают операторы SQL, для чего каждый из них используются?
* Часто ли применяется SQL? Приведите 3 примера реальных приложений, в которых используется SQL (укажите для чего именно).

**SQL-запросы**

Общая структура sql-запросов выглядит следующим образом:

SELECT (указываются столбцы) | обязательно

FROM (указывается таблица) | обязательно

WHERE (указывается условие для отбора данных) | необязательно

GROUP BY (указывается столбец для группировки данных) | необязательно

HAVING (указывается условие для фильтрации на уровне сгруппированных данных) | необязательно

ORDER BY (столбец, по которому необходими отсортировать вывод) | необязательно

Например, запрос:

SELECT \* FROM Customers

ORDER BY Country, City;

Вернёт все записи из таблицы Customers, отсортируя их сначала по стране (по столбцу стран), затем по городу (по столбцу городов)

Пример запроса, содержащего WHERE и HAVING:

SELECT City, count(CustomerID) AS number\_of\_clients FROM Customers

WHERE CustomerName NOT IN ('Around the Horn','Drachenblut Delikatessend')

GROUP BY City

HAVING number\_of\_clients >= 5

В данном запросе берутся города и число клиентов (в качестве переменной number\_of\_clients) из таблицы Customers. Затем исходная таблица фильтруется по именам клиентов (отсеиваются записи, не подходящие под указанные условия), полученные данные группируются по городам, а затем остаются только те записи, количество клиентов в которых не менее 5.

**Задание №6. Создание SQL-запросов**

Ну что, пришло время самим попробовать написать несколько запросов к базам данных!🤗 Для этого перейдите на [сайт](https://www.w3schools.com/sql/trysql.asp?filename=trysql_op_in), который был создан как раз для обучения SQL-запросам. Создайте файл exercise6.md, в который прикрепите скриншоты таблиц, которые выдаёт сервис после совершения запросов к базам данных. Причём на каждую базу данных (они расположены в правой части сайта) необходимо будет сделать по три запроса, один из которых должен быть сложным (содержать по 4-5 операторов запроса). К каждому скриншоту сделайте краткое описание получаемых данных.

*P.S.: Дополнительно потренироваться в написании SQL-запросов можно вот*[*на этом сайте*](https://www.w3schools.com/sql/exercise.asp)

**Chapter IV**

**Тестирование базы данных**

**Тестирование базы данных** – это выполнение проверки достоверности данных, проверка целостности данных, проверка производительности, связанная с базой данных, и тестирование процедур (различных запросов к базе данных) и функций в базе данных. Тестирование базы данных может включать в себя проверки: схем, таблицы базы данных, столбцов, ключей и индексов, хранимых процедур, триггеров (процедур, выполняющихся автоматически при определённом заданном событии на сервере), сервера базы данных, дублирования данных.

**Классификация**

В зависимости от функции и структуры базы данных тестирование БД можно разделить на три категории:

* **Функциональное тестирование** − проверка функциональности базы данных (отвечает на вопрос "соответствуют ли требуемые операции, выполняемые конечными пользователями, бизнес-спецификациям").
* **Структурное тестирование** базы данных − это процесс проверки структуры базы данных на точность и полноту. Примером такого вида тестирования является тестирование схемы − проверка типов данных столбцов, взаимосвязей между таблицами и так далее.
* **Нефункциональное тестирование** − включает в себя выполнение нагрузочного тестирования, стресс-тестирование, проверку минимальных системных требований на соответствие бизнес-спецификациям, выявление рисков и оптимизацию производительности базы данных

**Задание №7. Нагрузка, стресс и отказоустойчивость**

Создайте файл exercise7.md и в нём дайте ответы на следующие вопросы:

* Что такое нагрузочное тестирование, тестирование отказоустойчивости и стресс-тестирование? Что их отличает друг от друга?
* Приведите по одному примеру выполнения каждого из трёх видов тестирования (в контексте проверки базы данных большого веб-приложения).
* С помощью каких инструментов можно провести нагрузочное тестирование базы данных?
* Какие существуют способы оптимизации (улучшения производительности) базы данных?

**Double-check**

Перед загрузкой выполненного проекта в репозиторий перепроверь наличие всех необходимых файлов, которые требовалось создать во время его выполнения:

exercise1.md

exercise2.md

exercise3.md

exercise4.md

exercise5.md

exercise6.md

exercise7.md

💡 [Нажми здесь](https://forms.gle/PxQBA5TMjmX9rv9x7) **чтобы отправить обратную связь по проекту**.