



Министерство науки и высшего образования Российской Федерации  
Федеральное государственное бюджетное образовательное учреждение  
высшего образования  
«Московский государственный технический университет  
имени Н. Э. Баумана  
(национальный исследовательский университет)»  
(МГТУ им. Н. Э. Баумана)

---

---

ФАКУЛЬТЕТ «Информатика и системы управления (ИУ)»

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии (ИУ7)»

# РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

## *K KУРСОВОЙ РАБОТЕ*

*на тему:*

*«Проектирование базы данных характеристик  
комплектующих персонального стационарного  
компьютера»*

Студент ИУ7-53БВ  
(Группа)

(Подпись, дата)

К. А. Старовойтов  
(И. О. Фамилия)

Руководитель курсовой работы

(Подпись, дата)

А. А. Павлюк  
(И. О. Фамилия)

2024 г.

Министерство науки и высшего образования Российской Федерации  
Федеральное государственное бюджетное образовательное учреждение  
высшего образования  
«Московский государственный технический университет  
имени Н. Э. Баумана  
(национальный исследовательский университет)»  
(МГТУ им. Н. Э. Баумана)

---

---

УТВЕРЖДАЮ  
Заведующий кафедрой

ИУ7  
(Индекс)  
И. В. Рудаков  
(И.О. Фамилия)  
20 г.

## ЗАДАНИЕ на выполнение курсовой работы

по дисциплине: Базы данных

Студент группы ИУ7-53БВ

Старовойтов Кирилл Андреевич

(Фамилия, имя, отчество)

Тема курсовой работы: Проектирование базы данных характеристик комплектующих персонального стационарного компьютера

Направленность работы (учебная, исследовательская, практическая, производственная, др.) учебная

Источник тематики (кафедра, предприятие, НИР) кафедра

График выполнения работы: 25% к 4 нед., 50% к 7 нед., 75% к 11 нед., 100% к 14 нед.

1. Техническое задание:

Спроектировать базу данных, содержащую информацию о характеристиках компьютерных комплектующих: оперативной памяти, процессоров, графических карт, жестких дисков, твердотельных дисков, материнских плат

2. Оформление курсовой работы:

2.1 Расчетно-пояснительная записка на 25-30 листах формата А4.

Расчетно-пояснительная записка должна содержать постановку задачи, введение, аналитическую часть, конструкторскую часть, технологическую часть, исследовательский раздел, заключение, список литературы.

2.2 Перечень графического материала (плакаты, схемы, чертежи и т.п.). На защиту проекта должна быть представлена презентация, состоящая из 15-20 слайдов. На слайдах должны быть отражены: постановка задачи, использованные методы и алгоритмы, расчетные соотношения, структура комплекса программ, диаграмма классов, интерфейс, характеристики разработанного ПО, результаты проведенных исследований.

Дата выдачи задания « » 20 г.

Студент ИУ7-53БВ  
(Группа)

\_\_\_\_\_  
(Подпись, дата)

К. А. Старовойтов  
(И. О. Фамилия)

Руководитель курсовой работы

\_\_\_\_\_  
(Подпись, дата)

А. А. Павлюк  
(И. О. Фамилия)

# СОДЕРЖАНИЕ

|   |           |
|---|-----------|
| <b>ВВЕДЕНИЕ</b>                             | <b>4</b>  |
| <b>1 Аналитический раздел</b>               | <b>5</b>  |
| 1.1 Анализ предметной области . . . . .     | 5         |
| 1.2 Формализация данных . . . . .           | 8         |
| 1.3 Модели баз данных . . . . .             | 10        |
| 1.4 СУБД . . . . .                          | 11        |
| <b>2 Конструкторский раздел</b>             | <b>13</b> |
| 2.1 Формализация данных . . . . .           | 13        |
| 2.1.1 Сущности . . . . .                    | 13        |
| 2.1.2 Иерархия . . . . .                    | 13        |
| <b>3 Технологический раздел</b>             | <b>16</b> |
| 3.1 Выбор СУБД . . . . .                    | 16        |
| 3.2 Создание объектов базы данных . . . . . | 17        |
| 3.2.1 Таблицы . . . . .                     | 17        |
| 3.2.2 Функции . . . . .                     | 22        |
| <b>4 Исследовательский раздел</b>           | <b>26</b> |
| <b>ЗАКЛЮЧЕНИЕ</b>                           | <b>49</b> |
| <b>СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ</b>     | <b>50</b> |

# ВВЕДЕНИЕ

В современном мире информационных технологий базы данных играют ключевую роль в организации и хранении данных. Такие системы в отличии от традиционных способов хранения, например в обычном текстовом файле, имеют ряд преимуществ:

- компактное хранение
- быстрый доступ к информации
- изменение информации для множества записей
- сохранность информации
- доступность информации и т.д.

Однако, чтобы с такой информацией можно было работать, требуется предварительно спроектировать базу данных, применяя всевозможные решения в зависимости от требований, предъявляемых к системе, используя:

- заранее подготовленные функции для работы с данными в этой системе
- разграничение прав доступа для групп пользователей базой данных для защиты данных и т.д.

От того как качественно спроектирована система будет зависеть время доступа к данным, время на дополнение базы данных новыми записями, время на изменение имеющейся в базе данных информации.

Для каждого типа информации требуется свой подход к проектированию, т.к. та или иная группа информации может иметь схожие признаки, которые разумно выделить, разделить в отдельные сущности, а какие то объединить.

В данном курсовом проекте мы рассмотрим основные принципы проектирования баз данных, специфичные для отрасли информационных технологий, и разработаем структуру базы данных, позволяющую эффективно хранить и управлять информацией о комплектующих персонального компьютера.

Целью курсовой работы является проектирование и разработка базы данных характеристик комплектующих персонального стационарного компьютера.

# 1 Аналитический раздел

## 1.1 Анализ предметной области

Обычный персональный компьютер состоит из системного блока, состоящего из шасси и периферийных устройств.

Материнская (системная) плата — печатная плата, являющаяся основой построения модульного персонального компьютера. Системная плата содержит основную часть персонального компьютера. На материнской плате установлен чипсет — взаимосвязанный набор микросхем, логика которого в наибольшей степени определяет то, каковы прочие элементы компьютера и с каким типом устройств может работать данный чипсет, например какой тип стандарта оперативной памяти DDR поддерживает, с каким типом процессоров работает, какой тип PCI шины использует и т.д.

Центральный процессор — интегральная схема, исполняющая машинные инструкции (код программ), главная часть аппаратного обеспечения компьютера или программируемого логического контроллера.

Оперативная память (англ. Random Access Memory, RAM — память с произвольным доступом) — в большинстве случаев энергозависимая часть системы компьютерной памяти, в которой во время работы компьютера хранится выполняемый машинный код (программы), а также входные, выходные и промежуточные данные, обрабатываемые процессором. Оперативное запоминающее устройство (ОЗУ) — техническое устройство, реализующее функции оперативной памяти Для повышения надёжности, облегчения модернизации и экономии места на материнской плате микросхемы ОЗУ объединяют в модули, которые устанавливаются на плату вертикально. Наиболее распространенный стандарт оперативной памяти — DDR. Современные устройства еще используют версии DDR3, все больше DDR4, некоторые уже DDR5. Все типы являются несовместимыми и отличаются слотом подключения.

Видеокарта (видеоадаптер, графический адаптер) — устройство, преобразующее графический образ, хранящийся как содержимое памяти компьютера (или самого адаптера), в форму, пригодную для дальнейшего вывода на экран монитора. Обычно видеокарта выполнена в виде печатной платы (плата расширения) и вставляется в слот расширения материнской платы — универсальный, либо специализированный (AGP, PCI Express). Также широко

распространены и расположенные на системной плате видеокарты — как в виде дискретного отдельного чипа GPU, так и в качестве составляющей части северного моста чипсета или ЦПУ; в случае ЦПУ, встроенный (интегрированный) GPU, строго говоря, не может быть назван видеокартой. Видеокарты не ограничиваются простым выводом изображения. Они имеют встроенный графический процессор, который может производить дополнительную обработку, снимая эту задачу с центрального процессора компьютера. Например, видеокарты Nvidia и AMD (ATi) осуществляют рендеринг графического конвейера OpenGL, DirectX и Vulkan на аппаратном уровне. Также имеет место тенденция использовать вычислительные возможности графического процессора для решения неграфических задач (например, добычи криптовалюты или параллельных вычислений, таких как BOINC).

Энергонезависимая память (англ. Non-Volatile Random-Access Memory; NVRAM) — разновидность запоминающих устройств с произвольным доступом, которые способны хранить данные при отсутствии электрического питания. К энергонезависимым носителям: жесткие диски(HDD), твердотельные диски (SSD), флеш-накопители, лазерные диски и т.д.

Жёсткий диск (hard disk drive (HDD)) - запоминающее устройство (устройство хранения информации, накопитель) произвольного доступа, основанное на принципе магнитной записи. Является основным накопителем данных в большинстве компьютеров. Информация записывается на жёсткие (алюминиевые или стеклянные) пластины, покрытые слоем ферромагнитного материала, чаще всего диоксида хрома - магнитные диски. В HDD используется одна или несколько пластин на одной оси. Считывающие головки в рабочем режиме не касаются поверхности пластин благодаря прослойке набегающего потока воздуха, образующейся у поверхности при быстром вращении. Расстояние между головкой и диском составляет несколько нанометров (в современных дисках около 10 нм), а отсутствие механического контакта обеспечивает долгий срок службы устройства. При отсутствии вращения дисков головки находятся у шпинделя или за пределами диска в безопасной («парковочной») зоне, где исключён их нештатный контакт с поверхностью дисков. Также, в отличие от гибкого диска, носитель информации обычно совмещают с накопителем, приводом и блоком электроники. Такие жёсткие диски часто используются в качестве несъёмного носителя информации. Существуют

несколько различных технологий записи:

- Метод продольной записи — технология CMR (англ. Conventional Magnetic Recording) — это «обычная» магнитная запись, биты информации записываются с помощью маленькой головки, которая, проходя над поверхностью вращающегося диска, намагничивает миллиарды горизонтальных дискретных областей — доменов. При этом вектор намагченности домена расположен продольно, то есть параллельно поверхности диска. Каждая из этих областей является логическим нулём или единицей, в зависимости от направления намагченности.
- Метод перпендикулярной записи — технология PMR (англ. Perpendicular Magnetic Recording), при которой биты информации сохраняются в вертикальных доменах. Это позволяет использовать более сильные магнитные поля и снизить площадь материала, необходимую для записи 1 бита. Предыдущий метод записи, параллельно поверхности магнитной пластины, привёл к тому, что в определённый момент увеличивать плотность информации на дисках было невозможно. Плотность записи при этом методе резко возросла - более чем на 30 % ещё на первых образцах
- Метод черепичной магнитной записи (англ. Shingled Magnetic Recording, SMR) был реализован в начале 2010-х годов. В нём используется тот факт, что ширина области чтения меньше, чем ширина записывающей головки. Запись дорожек в этом методе производится с частичным наложением в рамках групп дорожек (пакетов). Каждая следующая дорожка пакета частично закрывает предыдущую (подобно черепичной кровле), оставляя от неё узкую часть, достаточную для считающей головки. По своей специфике она радикально отличается от более популярных технологий записи CMR и PMR. Черепичная запись увеличивает плотность записанной информации, однако усложняет перезапись - при каждом изменении требуется полностью перезаписать весь пакет перекрывающихся дорожек. Технология позволяет увеличить ёмкость жёстких дисков на 15-20 % в зависимости от конкретной реализации; при этом не лишена недостатков, главный из которых - низкая скорость записи/перезаписи, что критично при использовании в настольных компьютерах, поэтому применяется в основном в центрах обработки данных (ЦОД)

Со второй половины 2000-х годов получили распространение более производительные твердотельные накопители, вытесняющие дисковые накопители из ряда применений несмотря на более высокую стоимость единицы хранения; Твердотельный накопитель (англ. Solid-State Drive, SSD) - компьютерное энергонезависимое немеханическое запоминающее устройство на основе микросхем памяти, альтернатива жёстким дискам (HDD). Наиболее распространённый вид твердотельных накопителей использует для хранения данных флеш-память типа NAND(flash memory - разновидность полупроводниковой технологии электрически перепрограммируемой памяти (EEPROM)). Помимо микросхем памяти, накопитель содержит управляющую микросхему - контроллер. На 2016 год наиболее производительными выступали SSD формата M.2 с интерфейсом NVMe(NVM Express — от англ. Non-Volatile Memory) — интерфейс доступа к твердотельным накопителям, подключённым по шине PCI Express.), а к 2022 году их скорость записи/записи достигла 12000 мегабайт в секунду. По сравнению с традиционными жёсткими дисками твердотельные накопители имеют меньший размер и вес, являются бесшумными, а также многократно более устойчивы к повреждениям (например, при падении) и имеют гораздо большую скорость производимых операций. В то же время, они имеют в несколько раз большую стоимость в пересчёте на гигабайт и меньшую износостойкость (ресурс записи).

## 1.2 Формализация данных

В состав системного блока входят:

- Материнская плата в которую устанавливается или на которой имеются:
  - чипсет, определяющий тип используемого процессора, оперативной памяти, шины и т.д.
  - процессор,
  - оперативная память
  - «встроенные» контроллеры периферийных устройств, и разъёмы для подключения дополнительных взаимозаменяемых плат, называемых платами расширений, как правило подключённые к общейшине, такие как

- \* видеокарта,
- \* аудиокарта,
- \* сетевая карта и т.д.

, разъемы для подключения устройств энергонезависимой постоянной памяти:

- \* жёсткий диск (их может быть несколько, они могут быть объединены в RAID-массив)
- \* SSD, SSD m2

, прочие разъемы для подключения устройств ввода-вывода:

- \* usb различных стандартов: 2.0, 3.1, type-c и т.д.
- \* видеоразъемы: hdmi, display-port, VGA
- \* RJ-45 для подключения к сети ethernet
- \* разъемы для подключения устройств ввода-вывода звука: для аналоговых сигналов и цифровых (SPDIF)
- \* разъемы подключения устаревших устройств ввода-вывода (мышь, клавиатура) PS2
- \* COM-порт (последовательный порт), LTP (параллельный порт)
- \* и т.д.

(некоторые разъемы могут устанавливаться на плату с внутренней стороны, например COM-порт или USB)

– ЦПУ. Главными характеристиками ЦПУ являются:

- \* тактовая частота,
- \* энергопотребление,
- \* нормы литографического процесса, используемого при производстве (для микропроцессоров),
- \* архитектура

Также ЦПУ Может иметь встроенный графический процессор ГПУ

– ОЗУ Существуют несколько основных форматов плат ОЗУ:

- \* DIMM(Dual In-line Memory Module).
- \* SO-DIMM (small outline - DIMM) - платы , используемые преимущественно на системных платах ноутбуков

- Основными характеристиками графического процессора видеокарт являются:
  - \* тактовая частота,
  - \* энергопотребление,
  - \* нормы литографического процесса, используемого при производстве (для микропроцессоров),
  - \* архитектура

Характеристики видеокарты:

- \* видеопамять:
  - объем (также как и для оперативной памяти самым распространенным сейчас используют GDDR5),
  - памяти
  - пропускная способность
  - разрядность шины
- \* разъемы подключения

### 1.3 Модели баз данных

База данных - это упорядоченный набор структурированной информации или данных, которые обычно хранятся в электронном виде в компьютерной системе. База данных обычно управляет системой управления базами данных (СУБД). Данные вместе с СУБД, а также приложения, которые с ними связаны, называются системой баз данных, или, для краткости, просто базой данных.

Данные в наиболее распространенных типах современных баз данных обычно хранятся в виде строк и столбцов формирующих таблицу. Этими данными можно легко управлять, изменять, обновлять, контролировать и упорядочивать. В большинстве баз данных для записи и запросов данных используется язык структурированных запросов (SQL). [1]

Рассмотрим основные типы использующихся различных типов баз данных:

- Документо-ориентированные базы данных
- Ключ-значение

- Поисковые
- Реляционные

Документо-ориентированные базы данных в основном используются для хранения и обработки данных в формате документов, таких как JSON или XML. Они обеспечивают гибкость в структуре данных и удобны для хранения неструктурированных данных, позволяют легко добавлять новые, но ограничены в возможности выполнять сложные запросы данных, а также не позволяют связать большое количество данных.

Базы данных типа ключ-значение просты, т.к. для каждого значения имеется уникальный ключ, что облегчает доступ к данным, а также высокопроизводительны, однако такие хранилища обычно используют для кэширования временных данных, которые сохраняются в оперативной памяти, редко используются для сохранения данных на диск, в связи с чем такие системы практически и не применимы для сохранения на внешнее хранилище данных, из-за чего высок риск потери данных. Также системы не подходят для сложных связных структур данных.

Поисковые базы данных имеют мощные возможности для поиска и индексации данных и высокой производительностью при работе с текстом, но не с другими типами данных, а сложные структуры данных потребуют дополнительные затраты на обработку сложно связных структур данных.

Реляционные базы данных в отличии от остальных менее производительны, могут потребовать больше ресурсов, в зависимости от того как была спроектирована база данных, в связи с чем более сложны, но основаны на теории реляционных моделей данных, т.е. формализованы, обеспечивают целостность данных, позволяют создавать сложно связанные структуры данных, что допускает нормализацию данных для избежания их избыточности, предоставляют возможность резервировать данные. Такие типы систем зарекомендовали себя временем. Исходя из перечисленных достоинств и недостатков всех типов баз данных выбираем реляционные.

## 1.4 СУБД

СУБД — это совокупность языковых и программных средств, предназначенных для создания, ведения и совместного использования БД многими

пользователями. [2] Современная СУБД содержит в своем составе программные средства создания баз данных, средства работы с данными и сервисные средства. С помощью средств создания БД проектировщик, используя язык описания данных, переводит логическую модель БД в физическую структуру, а на языке манипуляции данными разрабатывает программы, реализующие основные операции с данными (в реляционных БД – это реляционные операции). При проектировании привлекаются визуальные средства, т.е. объекты, и программа-отладчик, с помощью которой соединяются и тестируются отдельные блоки разработанной программы управления конкретной БД.

Существует множество различных реляционных СУБД, самые распространенные из которых:

- Microsoft Access. Разработана изначально под ОС Windows (имеется альтернатива LibreOffice Base, но не поддерживающая полный функционал и некоректно импортирует данные из MS Access)
- Oracle. Хорошо задокументирована, но весьма ограничена по функциональности для бесплатного использования, имеет высокую стоимость для доступа ко всем функциям.
- MySql и MariaDB. Одни из самых известных баз данных, ныне MySql принадлежит Oracle, MariaDB является ответвлением от MySQL с открытым исходным кодом. Поддерживает шифрование, множество функций (например хранение координат и выполнение запросов данных о местоположении), высоко производительна, стабильна, работает во множестве известных операционных системах. MariaDB предназначена для совместимости с MySQL, однако существует проблема с сопоставлениями версий, а также инженеры MySQL вводят некоторые платную функциональность, недоступную MariaDB. [3]
- Postgresql - не просто реляционная СУБД, а объектно-реляционная СУБД. Поддерживает вложенные и составные конструкции, которые не поддерживаются стандартными СУБД. Поддерживает обширный список типов данных, которые поддерживает PostgreSQL: uuid, денежный, перечисляемый, геометрический, бинарный, адресный (сети) и т.д. [4]

## **2 Конструкторский раздел**

### **2.1 Формализация данных**

#### **2.1.1 Сущности**

Перечислим компоненты

- Оперативная память
- Процессор
- Графическая карта
- Графический процессор
- Жесткие диски (HDD)
- Твердотельные диски (SSD)
- Твердотельные диски (SSD) формата m2
- Материнская плата

Все сущности являются компонентами. Некоторые компоненты являются подмножеством других, например множество энергонезависимой памяти

Таким образом образуется иерархия сущностей с общими атрибутами.

#### **2.1.2 Иерархия**

Выделение сущностей отобразим в формате иерархии, вложенные элементы в которой предполагают наличие общих атрибутов:

- Страны
- Производители
- Компоненты
  - Оперативная память
  - Процессор
  - Графическая карта

- \* Графический процессор
- Энергонезависимая память
  - \* Жесткие диски (HDD)
  - \* Твердотельные диски (SSD)
    - Твердотельные диски (SSD) формата m2
- Материнская плата

Общие сущности при этом служат также и для объединения всех компонентов и предоставления общей информации о них.

Отобразим все атрибуты и связи сущностей в диаграмме сущность-связь (ERD) (рисунок 2.1)

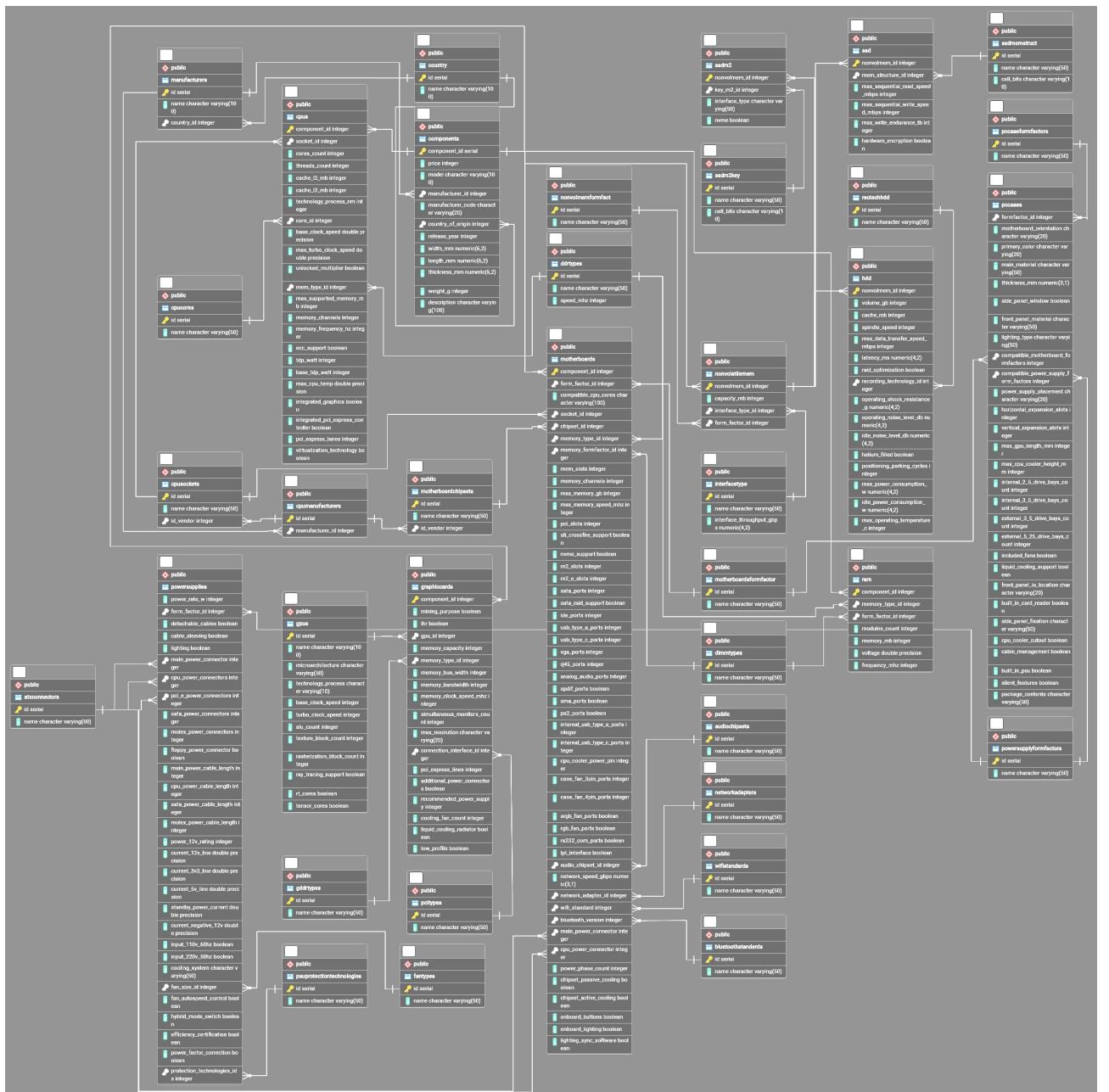


Рисунок 2.1 – ERD диаграмма

### 3 Технологический раздел

#### 3.1 Выбор СУБД

Сформируем требования, предъявляемые к СУБД:

- Одной из самых эффективных, распространенных и надёжных операционных систем является ОС Linux, поэтому СУБД должно работать в такой ОС
- В предыдущем разделе представлено множество сущностей, обладающих общими атрибутами, для которых требуется сформировать связь таблиц нескольких сущностей к одной. Можно сформировать для каждой из вложенных таблиц внешний ключ ссылающийся на одну главную таблицу, что потребует как дополнительный атрибут идентификатор, так и формирование для каждой записи вложенной таблицы, запись в главной таблице, а затем указание идентификатора на эту запись. Таким образом происходит усложнение работы как при формировании таблиц для сущностей, так и для обработки данных особенно при заполнении различных таблиц. Система должна иметь механизм работы, позволяющий облегчить процесс создания таблиц с частью одинаковых атрибутов

Была выбрана СУБД postgresql, в которой:

- встроена возможность наследования таблиц, которая позволяет одним ключевым словом с указанием таблицы родителя INHERITS(tablename) избежать дублирования атрибутов и организовать общую таблицу с множеством записей, имеющая общий первичный ключ для нескольких таблиц с одинаковой одной частью атрибутов и различной другой.
- поддержка ОС Linux
- бесплатное распространение ПО

Для работы с исходными текстами модулей был выбран следующий набор программного обеспечения:

- терминальный текстовый редактор VIM
- ПО с графическим интерфейсом pgAdmin

## 3.2 Создание объектов базы данных

### 3.2.1 Таблицы

В листинге 1 скрипт для создания и настройки таблиц для СУБД PostgreSQL

```
1 CREATE TABLE Countries(
2     id SERIAL PRIMARY KEY,
3     name VARCHAR(100)
4 );
5 CREATE TABLE Manufacturers(
6     id SERIAL PRIMARY KEY,
7     name VARCHAR(100),
8     country_id INT REFERENCES Countries(id)
9 );
10 -- Создание таблицы "Комплектующие"
11 CREATE TABLE Components (
12     component_id SERIAL PRIMARY KEY,
13     price int,
14     model VARCHAR(100),
15     manufacturer_code VARCHAR(50),
16     country_of_origin int REFERENCES Countries(id),
17     release_year INTEGER,
18     width_mm DECIMAL(6, 2),
19     length_mm DECIMAL(6, 2),
20     thickness_mm DECIMAL(6, 2),
21     weight_g INTEGER,
22
23     description VARCHAR(100)
24 );
25
26 -- Создание таблицы "Оперативная память"
27 CREATE TABLE RAMManufacturers(
28     id SERIAL PRIMARY KEY,
29     id_manufacturer INT REFERENCES Manufacturers(id)
30 );
31 CREATE TABLE DDRTypes(
32     id SERIAL PRIMARY KEY,
33     name VARCHAR(50),
34     speed_mhz INT
35 );
36 CREATE TABLE DIMMTypes(
37     id SERIAL PRIMARY KEY,
38     name VARCHAR(50)
39 );
40 CREATE TABLE RAM (
41     id_manufacturer INT REFERENCES RAMManufacturers(id),
42     memory_type_id int REFERENCES DDRTypes(id),
43     form_factor_id int REFERENCES DIMMTypes(id),
44     modules_count INT,
45     memory_mb INT,
46     voltage FLOAT,
47     frequency_mhz INT
48 ) INHERITS(Components);
49
50 -- Создание таблицы "Процессоры"
51 CREATE TABLE CPUManufacturers(
```

```

52     id SERIAL PRIMARY KEY,
53     id_manufacturer INT REFERENCES Manufacturers(id)
54 );
55 CREATE TABLE CPUcores (
56     id SERIAL PRIMARY KEY,
57     name VARCHAR(50)
58 );
59 CREATE TABLE CPUSockets(
60     id SERIAL PRIMARY KEY,
61     name VARCHAR(50),
62     id_manufacturer int REFERENCES CPUManufacturers(id)
63 );
64 CREATE TABLE CPUs (
65
66     socket_id INT REFERENCES CPUSockets(id) ,
67     --cores
68     cores_count INT,
69     threads_count INT,
70     cache_l2_mb int,
71     cache_l3_mb int,
72     technology_process_nm int,
73     core_id INT REFERENCES CPUcores(id) ,
74     -- speed
75     base_clock_speed FLOAT,
76     max_turbo_clock_speed FLOAT,
77     unlocked_multiplier BOOLEAN,
78     -- ram compatibility
79     mem_type_id INT REFERENCES DDRTypes(id) ,
80     max_supported_memory_mb INT,
81     memory_channels INT,
82     memory_frequency_hz INT,
83     ecc_support BOOLEAN,
84
85     tdp_watt INTEGER,
86     base_tdp_watt INTEGER,
87     max_cpu_temp FLOAT,
88
89     integrated_graphics BOOLEAN,
90
91     integrated_pci_express_controller BOOLEAN,
92     pci_express_lanes INTEGER,
93
94     virtualization_technology BOOLEAN
95 ) INHERITS(Components);
96
97
98     -- Создание таблицы "Графические карты"
99 CREATE TABLE GPUManufacturers(
100     id SERIAL PRIMARY KEY,
101     id_manufacturer INT REFERENCES Manufacturers(id)
102 );
103 CREATE TABLE GPUs(
104     id SERIAL PRIMARY KEY,
105     name VARCHAR(100),
106     id_manufacturer int REFERENCES GPUManufacturers(id) ,
107     microarchitecture VARCHAR(50),
108     technology_process VARCHAR(10),
109     base_clock_speed INTEGER,
110     turbo_clock_speed INTEGER,
111     alu_count INTEGER,

```

```

112     texture_block_count INTEGER,
113     rasterization_block_count INTEGER,
114     ray_tracing_support BOOLEAN,
115     rt_cores BOOLEAN,
116     tensor_cores BOOLEAN
117 );
118 CREATE TABLE GDDRTypes(
119     id SERIAL PRIMARY KEY,
120     name VARCHAR(50)
121 );
122 CREATE TABLE GraphOuts(
123     id SERIAL PRIMARY KEY,
124     name VARCHAR(50)
125 );
126 CREATE TABLE PCITypes(
127     id SERIAL PRIMARY KEY,
128     name VARCHAR(50)
129 );
130 CREATE TABLE GraphicCardsManufacturers(
131     id SERIAL PRIMARY KEY,
132     id_manufacturer INT REFERENCES Manufacturers(id)
133 );
134 CREATE TABLE GraphicCards(
135     id_manufacturer int REFERENCES GraphicCardsManufacturers(id),
136
137     mining_purpose BOOLEAN,
138     lhr BOOLEAN,
139
140     gpu_id INT REFERENCES GPUs(id),
141
142     memory_capacity INT,
143     memory_type_id INT REFERENCES GDDRTypes(id),
144     memory_bus_width INT,
145     memory_bandwidth INT,
146     memory_clock_speed_mhz INT,
147
148     simultaneous_monitors_count INTEGER,
149     max_resolution VARCHAR(20),
150
151     connection_interface_id INTEGER REFERENCES PCITypes(id),
152     pci_express_lines INTEGER,
153     additional_power_connectors BOOLEAN,
154     recommended_power_supply INTEGER,
155
156     cooling_fan_count INTEGER,
157     liquid_cooling_radiator BOOLEAN,
158
159     low_profile BOOLEAN
160 ) INHERITS(Components);

161
162 -- Создание таблиц "Энергонезависимая память"
163 CREATE TABLE NonVolMemManufacturers(
164     id SERIAL PRIMARY KEY,
165     id_manufacturer INT REFERENCES Manufacturers(id)
166 );
167 CREATE TABLE InterfaceType(
168     id SERIAL PRIMARY KEY,
169     name VARCHAR(50),
170     interface_throughput_gbps DECIMAL(4, 2)
171 );

```

```

172 | CREATE TABLE NonVolMemFormfact(
173 |   id SERIAL PRIMARY KEY,
174 |   name VARCHAR(50)
175 | );
176 | CREATE TABLE NonVolatileMem (
177 |   id_manufacturer int REFERENCES NonVolMemManufacturers(id),
178 |   capacity_mb INT,
179 |   interface_type_id INT REFERENCES InterfaceType(id),
180 |   form_factor_id INT REFERENCES NonVolMemFormfact(id)
181 | ) INHERITS(Components);
182 | CREATE TABLE RecTechHDD(
183 |   id SERIAL PRIMARY KEY,
184 |   name VARCHAR(50)
185 | );
186 |
187 | CREATE TABLE HDD (
188 |   volume_gb INTEGER,
189 |   cache_mb INTEGER,
190 |   spindle_speed INTEGER,
191 |   max_data_transfer_speed_mbps INTEGER,
192 |   latency_ms DECIMAL(4, 2),
193 |   raid_optimization BOOLEAN,
194 |
195 |   recording_technology_id int REFERENCES RecTechHDD(id),
196 |
197 |   operating_shock_resistance_g DECIMAL(4, 2),
198 |   operating_noise_level_db DECIMAL(4, 2),
199 |   idle_noise_level_db DECIMAL(4, 2),
200 |   helium_filled BOOLEAN,
201 |   positioning_parking_cycles INTEGER,
202 |
203 |   max_power_consumption_w DECIMAL(4, 2),
204 |   idle_power_consumption_w DECIMAL(4, 2),
205 |   max_operating_temperature_c INTEGER
206 | ) INHERITS(NonVolatileMem);
207 | -- Создание таблицы "SSD"
208 | CREATE TABLE SSDMemStruct(
209 |   id SERIAL PRIMARY KEY,
210 |   name VARCHAR(50),
211 |   cell_bits VARCHAR(10)
212 | );
213 |
214 | CREATE TABLE SSD (
215 |   mem_structure_id int REFERENCES SSDMemStruct(id),
216 |
217 |   max_sequential_read_speed_mbps INTEGER,
218 |   max_sequential_write_speed_mbps INTEGER,
219 |   max_write_endurance_tb INTEGER,
220 |   hardware_encryption BOOLEAN
221 | ) INHERITS(NonVolatileMem);
222 |
223 | CREATE TABLE SSDM2Key(
224 |   id SERIAL PRIMARY KEY,
225 |   name VARCHAR(50),
226 |   cell_bits VARCHAR(10)
227 | );
228 |
229 | CREATE TABLE SSDM2 (
230 |   key_m2_id INT REFERENCES SSDM2Key(id),
231 |   nvme BOOLEAN

```

```

232 ) INHERITS(SSD);
233
234 -- Создание таблицы "Материнские платы"
235 CREATE TABLE MotherboardManufacturers(
236     id SERIAL PRIMARY KEY,
237     id_manufacturer INT REFERENCES Manufacturers(id)
238 );
239 CREATE TABLE MotherboardsFormFactor(
240     id SERIAL PRIMARY KEY,
241     name VARCHAR(50)
242 );
243 CREATE TABLE AudioChipsets(
244     id SERIAL PRIMARY KEY,
245     name VARCHAR(50)
246 );
247 CREATE TABLE NetworkAdapters(
248     id SERIAL PRIMARY KEY,
249     name VARCHAR(50)
250 );
251 CREATE TABLE WifiStandards(
252     id SERIAL PRIMARY KEY,
253     name VARCHAR(50)
254 );
255 CREATE TABLE BluetoothStandards(
256     id SERIAL PRIMARY KEY,
257     name VARCHAR(50)
258 );
259 CREATE TABLE ATXConnectors(
260     id SERIAL PRIMARY KEY,
261     name VARCHAR(50)
262 );
263 CREATE TABLE MotherboardChipsets(
264     id SERIAL PRIMARY KEY,
265     name VARCHAR(50)
266 );
267 CREATE TABLE Motherboards (
268     form_factor_id INT REFERENCES MotherboardsFormFactor(id),
269     id_manufacturer int REFERENCES MotherboardManufacturers(id),
270
271     compatible_cpu_cores VARCHAR(100),
272     socket_id INT REFERENCES CPUSockets(id),
273     chipset_id INT REFERENCES MotherboardChipsets(id),
274
275     memory_type_id INT REFERENCES DDRTypes(id),
276     memory_formfactor_id INT REFERENCES DIMMTypes(id),
277     mem_slots INT,
278     memory_channels INT,
279     max_memory_gb INT,
280     max_memory_speed_mhz INT,
281
282     pci_slots int,
283     sli_crossfire_support BOOLEAN,
284
285     nvme_support BOOLEAN,
286     m2_slots INTEGER,
287     m2_e_slots INTEGER,
288
289     sata_ports INTEGER,
290     sata_raid_support BOOLEAN,
291     ide_ports INTEGER,

```

```

292     usb_2_count INTEGER,
293     usb_3_count INTEGER,
294     usb_type_a_ports INTEGER,
295     usb_type_c_ports INTEGER,
296     vga_ports INTEGER,
297     rj45_ports INTEGER,
298     analog_audio_ports INTEGER,
299     spdif_ports BOOLEAN,
300     sma_ports BOOLEAN,
301     ps2_ports BOOLEAN,
302     internal_usb_type_a_ports INTEGER,
303     internal_usb_type_c_ports INTEGER,
304
305     cpu_cooler_power_pin INTEGER,
306     case_fan_3pin_ports INTEGER,
307     case_fan_4pin_ports INTEGER,
308
309     argb_fan_ports BOOLEAN,
310     rgb_fan_ports BOOLEAN,
311
312     rs232_com_ports BOOLEAN,
313     lpt_interface BOOLEAN,
314     audio_chipset_id INT REFERENCES AudioChipsets(id),
315
316     network_speed_gbps DECIMAL(3,1),
317     network_adapter_id INT REFERENCES NetworkAdapters(id),
318
319     wifi_standard INT REFERENCES WifiStandards(id),
320     bluetooth_version INT REFERENCES BluetoothStandards(id),
321     main_power_connector INT REFERENCES ATXConnectors(id),
322     cpu_power_connector INT REFERENCES ATXConnectors(id),
323     power_phase_count INTEGER,
324     chipset_passive_cooling BOOLEAN,
325     chipset_active_cooling BOOLEAN,
326
327     onboard_buttons BOOLEAN,
328     onboard_lighting BOOLEAN,
329     lighting_sync_software BOOLEAN
330 ) INHERITS(Components);
331

```

Листинг 1 – Скрипт создания и настройки таблиц на языке PLpgSQL

### 3.2.2 Функции

Из предыдущего раздела видно, что количество атрибутов в таблицах относительно велико. Перед тем как добавлять новую запись, требуется подготовить данные в других вспомогательных таблицах, в которых запись может существовать или же отсутствовать. Сформируем вспомогательные функции для упрощения создания новых данных

1. для облегчения добавления элементов в промежуточные таблицы, кото-

рые имеют простую структуру:

- идентификатор
- имя

создадим функцию `find_or_create_el(_tbl, namevalue)`, которая ищет элемент по указанному значению атрибута (имени) и если не находит, то создает такой элемент и в любом случае возвращает идентификатор

2. `find_or_create_manufacturer` для условного добавления производителя по названию и стране. Функция является комбинацией использования предыдущей функции для двух таблиц Countries и Manufacturers
3. `find_or_create_manuf` условное добавления конкретной категории производителя (процессоров, оперативной памяти, графических карт и т.д.) по идентификатору производителя таблицы Manufacturers и названию таблицы категории производителя (CPUManufacturers, RAMManufacturers и т.д.)
4. `add_cpusocket` условное добавления сокета процессора по названию производителя, страны производителя и названию сокета
5. `add_gpu` добавление записи характеристик графического процессора в таблицу GPUs перед добавлением записей характеристик графических карт в таблицу GraphicCards

Описание функций представлено в листинге 2

```
1 -----  
2 CREATE OR REPLACE FUNCTION find_or_create_el(_tbl regclass, _name TEXT, OUT result  
3      INT)  
4 LANGUAGE plpgsql AS  
5 $func$  
6 BEGIN  
7     EXECUTE format('SELECT id FROM %s WHERE name = $1', _tbl)  
8     INTO result USING _name;  
9     IF result IS NULL THEN  
10        EXECUTE format('INSERT INTO %s (name) VALUES ($1) RETURNING id', _tbl) INTO  
11            result USING _name;  
12    END IF;  
13 END  
$func$;
```

```

14 CREATE OR REPLACE FUNCTION find_or_create_manufacturer(
15     _name TEXT, country_name TEXT
16 ) RETURNS INT AS $$$
17 DECLARE
18     _id INT;
19     _country_id INT;
20 BEGIN
21     SELECT find_or_create_el('Countries', country_name) INTO _country_id;
22     SELECT find_or_create_el('Manufacturers', _name) INTO _id;
23     UPDATE Manufacturers SET country_id=_country_id WHERE id=_id;
24     RETURN _id;
25 END;
26 $$ LANGUAGE plpgsql;
27 -----
28 CREATE OR REPLACE FUNCTION find_or_create_manuf(_tbl regclass, _id_manuf INT, OUT
29     ↪ result INT)
30     LANGUAGE plpgsql AS
31 $func$  

32 BEGIN
33     EXECUTE format('SELECT id FROM %s WHERE id_manufacturer = $1', _tbl)
34     INTO result USING _id_manuf;
35     IF result IS NULL THEN
36         EXECUTE format('INSERT INTO %s (id_manufacturer) VALUES ($1) RETURNING id',
37             ↪ _tbl) INTO result USING _id_manuf;
38     END IF;
39 END;
$func$;
40 -----
41 CREATE OR REPLACE FUNCTION add_cpusocket(
42     manufacturerName VARCHAR(100),
43     manufacturerCountry VARCHAR(100),
44     socket_name VARCHAR(100)
45 ) RETURNS INT AS $$$
46 DECLARE
47     _manufacturer_ID INT; _CPUmanufacturer_ID INT; _CPUsocketID INT; _CPUsockmanID
48     ↪ INT;
49 BEGIN
50     SELECT find_or_create_manufacturer(manufacturerName, manufacturerCountry) INTO
51     ↪ _manufacturer_ID;
52     SELECT find_or_create_manuf('CPUManufacturers', _manufacturer_ID) INTO
53     ↪ _CPUmanufacturer_ID;
54     SELECT find_or_create_el('CPUSockets', socket_name) INTO _CPUsocketID;
55     SELECT id_manufacturer INTO _CPUsockmanID FROM CPUSockets WHERE id=_CPUsocketID;
56     IF _CPUsockmanID IS NULL THEN
57         UPDATE CPUSockets SET id_manufacturer=_CPUmanufacturer_ID WHERE
58             ↪ id=_CPUsocketID;
59     END IF;
60     RETURN _CPUsocketID;
61 END;
62 $$ LANGUAGE plpgsql;
63 -----
64 CREATE OR REPLACE FUNCTION add_gpu(
65     manufacturerName VARCHAR(100),
66     manufacturerCountry VARCHAR(100),
67     ↪
68     _name VARCHAR(100),
69     microarchitecture VARCHAR(50),
70     technology_process VARCHAR(10),
71     base_clock_speed INTEGER,
72     turbo_clock_speed INTEGER,

```

```

68     alu_count INTEGER,
69     texture_block_count INTEGER,
70     rasterization_block_count INTEGER,
71     ray_tracing_support BOOLEAN,
72     rt_cores BOOLEAN,
73     tensor_cores BOOLEAN
74 ) RETURNS INT AS $$  

75 DECLARE  

76     _manufacturer_ID INT; _GPUmanufacturer_ID INT; _resultID INT;  

77 BEGIN
78     -- Добавление производителя
79     SELECT find_or_create_manufacturer(manufacturerName, manufacturerCountry) INTO
80         → _manufacturer_ID;
80     SELECT find_or_create_manuf('GPUManufacturers', _manufacturer_ID) INTO
81         → _GPUmanufacturer_ID;  

82
82     INSERT INTO GPUs ( name , id_manufacturer , microarchitecture ,
83         → technology_process , base_clock_speed , turbo_clock_speed , alu_count ,
83         → texture_block_count , rasterization_block_count , ray_tracing_support ,
83         → rt_cores , tensor_cores )
83     VALUES( _name , _GPUmanufacturer_ID, microarchitecture , technology_process ,
83         → base_clock_speed , turbo_clock_speed , alu_count , texture_block_count ,
83         → rasterization_block_count , ray_tracing_support , rt_cores , tensor_cores )
83         → RETURNING id INTO _resultID;  

84
84     RETURN _resultID;
85 END;  

86 $$ LANGUAGE plpgsql;
87 -----  

88
89

```

Листинг 2 – Скрипт создания функций на языке PLpgSQL

## 4 Исследовательский раздел

Наполним базу данных различными записями при помощи следующих функций:

- `add_rams()` для заполнения таблицы RAM (листинг 3)
- `add_cpus()` для заполнения таблицы CPUs(листинг 4)
- `add_gcards()` для заполнения таблицы GraphicCards(листинг 5), а также отдельно предварительного создания записей для GPUs
- `add_hdds()` для заполнения таблицы HDD(листинг 6)
- `add_ssds()` для заполнения таблицы SSD(листинг 7)
- `add_ssdsm2()` для заполнения таблицы SSDM2(листинг 8)
- `add_motherboards()` для заполнения таблицы Motherboards(листинг 9)

```
1 CREATE OR REPLACE FUNCTION add_rams(
2 ) RETURNS VOID AS $$
3 BEGIN
4     INSERT INTO RAM (
5         price, model, manufacturer_code, country_of_origin, release_year,
6         width_mm, length_mm, thickness_mm, weight_g, description,
7         id_manufacturer, memory_type_id, form_factor_id, modules_count, memory_mb,
8         → voltage, frequency_mhz
9     )
10    VALUES
11    (
12        100, 'HyperX Fury DDR4', 'HX426C16FB3K2/16',
13        (select find_or_create_el('Countries', 'CHINA')),
14        2020, 32.00, 133.35, 7.20, 60,
15        'DDR4-3200 16GB Kit (2x8GB) CL16 DIMM',
16        (find_or_create_manuf('RAMManufacturers', (find_or_create_manufacturer('Hyper
17        → X', 'USA')))),
18        →
19        (find_or_create_el('DDRTypes', 'DDR4')),
20        (find_or_create_el('DIMMTypes', 'DIMM')),
21        2, 16384, 1.35, 3200
22    ),
23    (
24        120, 'HyperX Fury DDR4 RGB', 'HX436C17FB3AK2/16',
25        (select find_or_create_el('Countries', 'CHINA')),
26        2021, 34.1, 133.35, 8, 70,
27        'DDR4-3600 CL17 16GB Kit (2x8GB)',
28        (find_or_create_manuf('RAMManufacturers', (find_or_create_manufacturer('Hyper
29        → X', 'USA')))),
30        →
31        (find_or_create_el('DDRTypes', 'DDR4')),
32        (find_or_create_el('DIMMTypes', 'DIMM')),
```

```

28      2, 16384, 1.35, 3600
29      ),
30      (
31      40, 'Kingston ValueRAM DDR4', 'KVR26S19S8/8',
32      (select find_or_create_el('Countries','CHINA')),
33      2022, 30, 69.6, 3.8, 20,
34      'DDR4-2666 SO-DIMM 8GB CL19',
35      (find_or_create_manuf('RAMManufacturers',(find_or_create_manufacturer('Kings
36      ↵ ton','USA'))),
37      ↵
38      (find_or_create_el('DDRTypes','DDR4')),
39      (find_or_create_el('DIMMTypes','SO-DIMM')),
40      1, 8192, 1.2, 2666
41      );
42
43  END;
44 $$ LANGUAGE plpgsql;

```

Листинг 3 – Функция добавления тестируемых записей в таблицу RAM

```

1 CREATE OR REPLACE FUNCTION add_cpus(
2 ) RETURNS VOID AS $$
3 BEGIN
4   INSERT INTO CPUs (
5     price, model, manufacturer_code, country_of_origin, release_year,
6     width_mm, length_mm, thickness_mm, weight_g, description,
7     socket_id, cores_count, threads_count, cache_12_mb, cache_13_mb,
8     ↵ technology_process_nm, core_id,
9     base_clock_speed, max_turbo_clock_speed, unlocked_multiplier, mem_type_id,
10    ↵ max_supported_memory_mb,
11    memory_channels, memory_frequency_hz, ecc_support, tdp_watt, base_tdp_watt,
12    ↵ max_cpu_temp,
13    integrated_graphics, integrated_pci_express_controller, pci_express_lanes,
14    ↵ virtualization_technology
15  ) VALUES (
16    589, 'Core i9-13900K', 'BX8071513900K',
17    (select find_or_create_el('Countries','CHINA')),
18    2022, 37.5, 37.5, 4.4, 100, '24-ядерный процессор 3.0 ГГц',
19    (add_cpusocket('Intel','USA','LGA 1700')),
20    24, 32, 32, 36, 10,
21    (find_or_create_el('CPUCores','Raptor Lake')),
22    3.0, 5.8, TRUE,
23    (find_or_create_el('DDRTypes','DDR4')),
24    128, 2, 5200, FALSE, 125, 125, 100,
25    TRUE, TRUE, 20, TRUE),
26    (
27      699, 'Ryzen 9 7950X', '100-000000514',
28      (select find_or_create_el('Countries','CHINA')),
29      2022, 40, 40, 4.4, 70, '16-ядерный процессор 4.5 ГГц',
30      (add_cpusocket('AMD','USA','AM5')),
31      16, 32, 16, 64, 5,
32      (find_or_create_el('CPUCores','Raptor Lake')),
33      4.5, 5.7, TRUE,
34      (find_or_create_el('DDRTypes','DDR5')),
35      128, 2, 5200, TRUE, 170, 105, 95,
36      FALSE, TRUE, 24, TRUE);
37
38  END;
39 $$ LANGUAGE plpgsql;

```

Листинг 4 – Функция добавления testируемых записей в таблицу CPUs

```
1 CREATE OR REPLACE FUNCTION add_gcards(
2 ) RETURNS VOID AS $$  
3 BEGIN  
4     PERFORM add_gpu( 'NVIDIA', 'USA', 'NVIDIA RTX 4090', 'NVIDIA Ada Lovelace',
5         ↳  '5nm', 2235, 2520, 16384, 512, true, true, true
6     ) ;  
7  
8     PERFORM add_gpu( 'AMD', 'USA', 'AMD Radeon RX 7900 XTX', 'AMD RDNA3', '5nm',
9         ↳  1900, 2500 , 6144, 384, 192, true, true, true
10    ) ;  
11  
12    INSERT INTO GraphicCards (
13        price, model, manufacturer_code, country_of_origin, release_year,
14        width_mm, length_mm, thickness_mm, weight_g, description,
15        id_manufacturer, mining_purpose, lhr, gpu_id, memory_capacity, memory_type_id,
16        memory_bus_width, memory_bandwidth, memory_clock_speed_mhz,
17        ↳  simultaneous_monitors_count,
18        max_resolution, connection_interface_id, pci_express_lines,
19        ↳  additional_power_connectors,
20        recommended_power_supply, cooling_fan_count, liquid_cooling_radiator, low_profile
21    ) VALUES (
22        1599, 'ROG Strix GeForce RTX 4090', 'ROG-STRIX-RTX4090-024G-GAMING',
23        (select find_or_create_el('Countries','CHINA')),
24        2022, 140.1, 357.6, 70.1, 2189, 'GeForce RTX 4090 24GB GDDR6X',
25        (find_or_create_manuf('GraphicCardsManufacturers',(find_or_create_manufacturer('
26        ↳  ASUS','Taiwan')))),
27        ↳
28        FALSE, TRUE,
29        (find_or_create_el('GPUs','NVIDIA RTX 4090')),
30        24,
31        (find_or_create_el('GDDRTypes','GDDR5')),
32        384, 1008, 21000, 4, '7680x4320',
33        (find_or_create_el('PCITypes','DisplayPort')),
34        16, TRUE, 850, 3, FALSE, FALSE
35    ), (
36        1999, 'AMD Radeon RX 7900XTX', 'RX7900XTX-24G',
37        3,
38        2023, 140, 300, 60, 2100, 'ASRock Radeon RX 7900XTX',
39        (find_or_create_manuf('GraphicCardsManufacturers',(find_or_create_manufacturer('
40        ↳  ASRock','Taiwan')))),
41        ↳
42        FALSE, TRUE,
43        (find_or_create_el('GPUs','AMD Radeon RX 7900 XTX')),
44        24,
45        (find_or_create_el('GDDRTypes','GDDR5')),
46        384, 10500, 22000, 4, '7680x4320',
47        (find_or_create_el('PCITypes','DisplayPort')),
48        16, TRUE, 850, 3, FALSE, FALSE);
49  
50  
51    END;  
52    $$ LANGUAGE plpgsql;
```

Листинг 5 – Функция добавления testируемых записей в таблицу GraphicCards

```

1 CREATE OR REPLACE FUNCTION add_hdds(
2 ) RETURNS VOID AS $$*
3 BEGIN
4
5   INSERT INTO HDD (
6     price, model, manufacturer_code, country_of_origin, release_year,
7     width_mm, length_mm, thickness_mm, weight_g, description,
8     id_manufacturer, capacity_mb, interface_type_id, form_factor_id,
9     volume_gb, cache_mb, spindle_speed, max_data_transfer_speed_mbps,
10    latency_ms, raid_optimization, recording_technology_id,
11    → operating_shock_resistance_g,
12    operating_noise_level_db, idle_noise_level_db, helium_filled,
13    → positioning_parking_cycles,
14    max_power_consumption_w, idle_power_consumption_w, max_operating_temperature_c
15  ) VALUES (
16    529, 'IronWolf Pro 18TB', 'ST18000NE000',
17    (find_or_create_el('Countries', 'CHINA')),
18    2022, 101.6, 146.99, 26.11, 705, '18TB 7200RPM SATA 6Gb/s 3.5"',
19    (find_or_create_manuf('NonVolMemManufacturers', (find_or_create_manufacturer('Seagate', 'USA')))),
20    →
21    18000000,
22    (find_or_create_el('InterfaceType', 'SATA')),
23    (find_or_create_el('NonVolMemFormfact', '3.5')),
24    18, 256, 7200, 285, 4.16, TRUE,
25    (find_or_create_el('RecTechHDD', 'CMR')),
26    70, 28, 20, TRUE, 600000, 7.5, 4.5, 70 ),
27
28    (129, 'Seagate BarraCuda 4TB', 'ST4000DM004',
29    (find_or_create_el('Countries', 'USA')),
30    2021, 101.6, 146.99, 26.11, 705, '',
31    (find_or_create_manuf('NonVolMemManufacturers', (find_or_create_manufacturer('Seagate', 'USA')))),
32    →
33    400000,
34    (find_or_create_el('InterfaceType', 'SATA')),
35    → (find_or_create_el('NonVolMemFormfact', '3.5')),
36    40, 256, 5400, 190, 6, FALSE,
37    (find_or_create_el('RecTechHDD', 'SMR')),
38    70, 26, 20, TRUE, 700000, 8, 5, 60),
39
40    (249, 'WD Black 6TB', 'WD6003FZBX',
41    (find_or_create_el('Countries', 'Thailand')),
42    2020, 101.6, 146.99, 26.11, 705, '',
43    (find_or_create_manuf('NonVolMemManufacturers', (find_or_create_manufacturer('Western Digital', 'USA')))),
44    →
45    6000,
46    (find_or_create_el('InterfaceType', 'SATA')),
47    → (find_or_create_el('NonVolMemFormfact', '3.5')),
48    6000, 256, 7200, 256, 4.2, TRUE,
49    (find_or_create_el('RecTechHDD', 'CMR')),
50    65, 28, 22, TRUE, 80000, 10, 7, 65);
51
52  END;
53 $$ LANGUAGE plpgsql;

```

Листинг 6 – Функция добавления тестируемых записей в таблицу HDD

```

1 CREATE OR REPLACE FUNCTION add_ssds(
2 ) RETURNS VOID AS $$  

3 BEGIN  

4 INSERT INTO SSD (
5     price, model, manufacturer_code, country_of_origin, release_year,
6     width_mm, length_mm, thickness_mm, weight_g, description,
7     id_manufacturer, capacity_mb, interface_type_id, form_factor_id,
8     mem_structure_id, max_sequential_read_speed_mbps,
9     ↳ max_sequential_write_speed_mbps,
10    max_write_endurance_tb, hardware_encryption
11 ) VALUES (
12     229, 'Samsung 870 EVO', 'MZ-77E1T0BW/EU',
13     (select find_or_create_el('Countries','CHINA')),
14     2020, 69.85, 100, 6.8, 45, '2TB PCIe Gen 4.0 x4 NVMe M.2',
15     (find_or_create_manuf('NonVolMemManufacturers',(find_or_create_manufacturer('Sam
16     ↳ sung','South Korea'))),
17     ↳
18     10000000,
19     (find_or_create_el('InterfaceType', 'SATA')),
20     (find_or_create_el('NonVolMemFormfact', '2.5')),
21     (find_or_create_el('SSDMemStruct', '3DNand')),
22     560, 530, 600, TRUE),
23     -----
24     (179, 'Toshiba X300', 'HDWF180UZSVA',
25     (select find_or_create_el('Countries','PHILIPPINES')),
26     2021, 101.6, 147, 26.1, 450, '8TB SATA III 3.5"', (find_or_create_manuf('NonVolM
27     ↳ emManufacturers',(find_or_create_manufacturer('Toshiba','Japan'))),
28     ↳
29     8000000,
30     (find_or_create_el('InterfaceType', 'SATA')),
31     ↳ (find_or_create_el('NonVolMemFormfact', '3.5')),
32     ↳ (find_or_create_el('SSDMemStruct', 'HDD')),
33     210, 200, 4.5, FALSE),
34     -----
35     (149, 'Gigabyte AORUS RGB', 'GP-AG41TB',
36     (select find_or_create_el('Countries','TAIWAN')),
37     2020, 80.15, 22.15, 2.38, 15.5, '1TB PCIe Gen 4.0 x4 NVMe M.2',
38     ↳ (find_or_create_manuf('NonVolMemManufacturers',(find_or_create_manufacturer('
39     ↳ Gigabyte','Taiwan'))),
40     ↳
41     1000000,
42     (find_or_create_el('InterfaceType', 'NVMe')),
43     ↳ (find_or_create_el('NonVolMemFormfact', 'M.2')),
44     ↳ (find_or_create_el('SSDMemStruct', '3DNand')),
45     7000, 5500, 1.2, TRUE);
46 END;
47 $$ LANGUAGE plpgsql;

```

Листинг 7 – Функция добавления тестируемых записей в таблицу SSD

```

1 CREATE OR REPLACE FUNCTION add_ssdsm2(
2 ) RETURNS VOID AS $$  

3 BEGIN  

4 INSERT INTO SSDM2 (
5     price, model, manufacturer_code, country_of_origin, release_year,
6     width_mm, length_mm, thickness_mm, weight_g, description,

```

```

7   id_manufacturer, capacity_mb, interface_type_id, form_factor_id,
8   mem_structure_id, max_sequential_read_speed_mbps,
9   ↳ max_sequential_write_speed_mbps,
10  max_write_endurance_tb, hardware_encryption, key_m2_id, nvme
11 ) VALUES (
12   229, '980 PRO 2TB', 'MZ-V8P2T0BW',
13   (select find_or_create_el('Countries', 'CHINA'))),
14   2020, 22.15, 80.15, 2.38, 9, '2TB PCIe Gen 4.0 x4 NVMe M.2',
15   (find_or_create_manuf('NonVolMemManufacturers', (find_or_create_manufacturer('Sam
16   ↳ sung', 'South Korea'))),
17   ↳
18   2000000,
19   (find_or_create_el('InterfaceType', 'SATA')),
20   (find_or_create_el('NonVolMemFormfact', '3.5')),
21   (find_or_create_el('SSDMemStruct', '3DNand')),
22   7000, 5100, 1200, TRUE,
23   (find_or_create_el('SSDM2Key', 'M|B')),
24   TRUE),
25   -----
26   ( 149, 'KC2500 1TB', 'KC2500/1000G',
27   (select find_or_create_el('Countries', 'TAIWAN')), 2021,
28   22.15, 80.15, 2.38, 9, '1TB NVMe PCIe Gen 3.0 x4 M.2',
29   (find_or_create_manuf('NonVolMemManufacturers', (find_or_create_manufacturer('Kin
30   ↳ gston', 'TAIWAN'))),
31   1000,
32   (find_or_create_el('InterfaceType', 'NVMe PCIe Gen 3.0 x4')),
33   (find_or_create_el('NonVolMemFormfact', 'M.2 2280')),
34   (find_or_create_el('SSDMemStruct', '3DNand')), 3500, 2900,
35   800, TRUE,
36   (find_or_create_el('SSDM2Key', 'M|B')), TRUE),
37   -----
38   ( 79, 'NE-512', 'NE-512G',
39   (select find_or_create_el('Countries', 'CHINA')), 2020,
40   22.15, 80.15, 2.38, 9, '512GB SATA III M.2',
41   (find_or_create_manuf('NonVolMemManufacturers', (find_or_create_manufacturer('Kin
42   ↳ gSpec', 'CHINA'))),
43   512,
44   (find_or_create_el('InterfaceType', 'SATA III')),
45   (find_or_create_el('NonVolMemFormfact', 'M.2 2280')),
46   (find_or_create_el('SSDMemStruct', '3DNand')), 550, 420,
47   400, TRUE,
48   (find_or_create_el('SSDM2Key', 'M|B')), TRUE);
49 END;
50 $$ LANGUAGE plpgsql;

```

Листинг 8 – Функция добавления тестируемых записей в таблицу SSDM2

```

1 CREATE OR REPLACE FUNCTION add_motherboards(
2 ) RETURNS VOID AS $$
3 BEGIN
4   INSERT INTO Motherboards (

```

```

5     price, model, manufacturer_code, country_of_origin, release_year, width_mm,
6     ↳ length_mm, thickness_mm, weight_g, description, id_manufacturer, socket_id,
7     ↳ chipset_id, form_factor_id, memory_type_id, max_memory_gb, mem_slots,
8     ↳ memory_channels, sata_ports, m2_slots, sata_raid_support, usb_2_count,
9     ↳ usb_3_count, usb_type_c_ports, rj45_ports, network_speed_gbps,
10    ↳ wifi_standard, bluetooth_version, main_power_connector, cpu_power_connector,
11    ↳ power_phase_count, chipset_passive_cooling, chipset_active_cooling,
12    ↳ onboard_buttons, onboard_lighting, lighting_sync_software
13 ) VALUES
14 (
15     699, 'ROG Maximus Z790 Hero', 'ROG MAXIMUS Z790 HERO',
16     (select find_or_create_el('Countries','CHINA')),
17     2022, 305, 244, 46, 1850, 'Intel Z790 ATX',
18     (find_or_create_manuf('MotherboardManufacturers',(find_or_create_manufacturer('A
19     ↳ SUS','Taiwan')))),
20     ↳
21     (add_cpusocket('Intel','USA','LGA 1700')),
22     (find_or_create_el('MotherboardChipsets','Intel Z790')),
23     (find_or_create_el('MotherboardsFormFactor','ATX')),
24     (find_or_create_el('DDRTypes','DDR4')),
25     128, 4, 2, 4, 5, TRUE, 4, 8, 1, 2, 2.5,
26     (find_or_create_el('WifiStandards','Wi-Fi 6E')),
27     (find_or_create_el('BluetoothStandards','5.3')),
28     (find_or_create_el('ATXConnectors','24-pin')),
29     (find_or_create_el('ATXConnectors','8-pin')),
30     20, TRUE, FALSE, TRUE, TRUE
31 ), -----
32 (
33     189, 'GIGABYTE B550 AORUS ELITE', 'B550 AORUS ELITE',
34     (select find_or_create_el('Countries','TAIWAN')), 2021,
35     305, 244, 46, 1850, 'AMD B550 ATX',
36     (find_or_create_manuf('MotherboardManufacturers',(find_or_create_manufacturer('G
37     ↳ IGABYTE','Taiwan')))),
38     (add_cpusocket('AMD','USA','AM4')),
39     (find_or_create_el('MotherboardChipsets','AMD B550')),
40     (find_or_create_el('MotherboardsFormFactor','ATX')),
41     (find_or_create_el('DDRTypes','DDR4')),
42     128, 4, 2, 6, 2, TRUE, 4, 6, 1,
43     1, 2.5, (find_or_create_el('WifiStandards','Wi-Fi 6E')),
44     (find_or_create_el('BluetoothStandards','5.3')),
45     (find_or_create_el('ATXConnectors','24-pin')),
46     (find_or_create_el('ATXConnectors','8-pin')), 12,
47     TRUE, FALSE, TRUE, FALSE, FALSE
48 ), -----
49 (
50     279, 'MSI MPG Z590 GAMING CARBON WIFI', 'MPG Z590 GAMING CARBON WIFI',
51     (select find_or_create_el('Countries','CHINA')), 2021,
52     305, 244, 46, 1850, 'Intel Z590 ATX',
53     (find_or_create_manuf('MotherboardManufacturers',(find_or_create_manufacturer('M
54     ↳ SI','China')))),
55     (add_cpusocket('Intel','USA','LGA 1200')),
56     (find_or_create_el('MotherboardChipsets','Intel Z590')),
57     (find_or_create_el('MotherboardsFormFactor','ATX')),
58     (find_or_create_el('DDRTypes','DDR4')),
59     128, 4, 2, 6, 3, TRUE, 6, 8, 2,
60     1, 2.5,(find_or_create_el('WifiStandards','Wi-Fi 6E')),
61     (find_or_create_el('BluetoothStandards','5.2')),
62     (find_or_create_el('ATXConnectors','24-pin')),

```

```

54     (find_or_create_el('ATXConnectors', '8-pin')), 16,
55     TRUE, FALSE, FALSE, FALSE
56   );
57 END;
58 $$ LANGUAGE plpgsql;

```

Листинг 9 – Функция добавления тестируемых записей в таблицу Motherboards

Выдача прав пользователю для работы с данными в таблице выполняется при помощи интерпретатора в терминале psql от пользователя с правами администратора, подключенным к базе данных perscompcomps (листинг 10)

```

1 sudo -u postgres psql
2 postgres=# \c perscompcomps
3 perscompcomps=# grant all on all tables in schema public to kirpitcheq;
4 GRANT
5 perscompcomps=# grant all on all sequences in schema public to kirpitcheq;
6 GRANT

```

Листинг 10 – Добавление прав для работы с данными базы данных perscompcomps

Результаты добавления данных на рисунках 4.1, 4.2, 4.3, 4.4, 4.5, 4.6, 4.7. Все запросы выполнены успешно. Выполним запрос на получение данных из всех таблиц.

На рисунке 4.12 видим, что данных больше чем было добавлено в таблицу ssd, что верно. Остальные данные связаны с данными таблицы ssdm2, которая является наследуемой от ssd. На рисунке 4.13 подтверждение этому — записи таблицы ssdm2 действительно унаследованы от ssd и связаны по внешнему ключу.

На рисунке 4.14 отображен результат получения всех данных множества энергонезависимой памяти благодаря наследованию атрибутов таблицы nonvolatilemem.

На рисунке 4.16 отображен результат запроса всех записей из таблицы components. Все записи множества компонентов наследуемых таблиц от components имеются

Убедимся, что функции выполнены верно, для чего выполним запросы получения данных из таблиц стран (рисунок 4.17), типов оперативной памяти (рисунок 4.18), производителей (рисунок 4.19), сокетов процессоров (рисунок 4.20), графических процессоров (рисунок 4.21)

The screenshot shows the MySQL Workbench interface. The left sidebar lists database objects: Aggregates, Collations, Domains, FTS Collations, FTS Directories, FTS Parameters, FTS Tables, Foreign Keys, and Functions. The 'Functions' node is expanded, showing several functions starting with 'add\_'. The main pane displays a query editor with the following content:

```
1 select add_rams();
2 select add_cpus();
3 select add_gcards();
4 select add_hdds();
```

Below the queries, the results of the first query are shown in a table:

|   | add_rams | lock |
|---|----------|------|
| 1 |          |      |

At the bottom of the interface, status information is displayed: Total rows: 1 of 1, Query complete 00:00:00.160, and Ln 2, Col 1.

Рисунок 4.1 – Отображение результата выполненного запроса функции  
add\_rams

Видим, что данные добавлены, не продублированы, были условно добавлены при отсутствии или найдены при использовании реализованных функций.

The screenshot shows the MySQL Workbench interface. The left sidebar lists database objects: Aggregates, Collations, Domains, FTS Caches, FTS Directories, FTS Parameters, FTS Tables, Foreign Keys, and Functions. The 'Functions' section is expanded, showing several functions starting with 'add\_'. The main area displays a query window with the following content:

```
1 select add_rams();  
2 select add_cpus();  
3 select add_gcards();  
4 select add_hdds();
```

The second line, 'select add\_cpus();', is highlighted. Below the query window, the 'Data Output' tab is selected, showing a table with one row:

|   | add_cpus | void |
|---|----------|------|
| 1 |          |      |

At the bottom of the interface, status bars show 'Total rows: 1 of 1', 'Query complete 00:00:00.070', and 'Ln 3, Col 1'.

Рисунок 4.2 – Отображение результата выполненного запроса функции  
add\_cpus

The screenshot shows the MySQL Workbench interface. The left sidebar displays various database objects like Aggregates, Collations, Domains, FTS Components, FTS Directories, FTS Parameters, FTS Tables, Foreign Keys, and Functions. The main area has tabs for 'Query' (selected), 'Query History', and 'Scratch Pad'. The 'Query' tab contains the following SQL code:

```
2 select add_cpus();
3 select add_gcards();
```

The results pane shows the output of the last query:

|   | add_gcards | void |
|---|------------|------|
| 1 |            |      |

At the bottom, status messages indicate: 'Total rows: 1 of 1', 'Query complete 00:00:00.073', and 'Ln 4, Col 1'.

Рисунок 4.3 – Отображение результата выполненного запроса функции  
add\_gcards

The screenshot shows the MySQL Workbench interface. The left sidebar contains a tree view of database objects under the 'Functions' category. The main area displays a query editor with the following SQL code:

```
3 select add_gcards();  
4 select add_hdds();  
5 select add_ssds();  
6 select add_ssdsm2();
```

The result set for the selected query (line 4) is shown in a table:

|   | add_hdds | void |
|---|----------|------|
| 1 |          |      |

At the bottom, the status bar indicates: Total rows: 1 of 1 | Query complete 00:00:00.092 | Ln 5, Col 1.

Рисунок 4.4 – Отображение результата выполненного запроса функции  
add\_hdds

The screenshot shows the MySQL Workbench interface. The left sidebar lists database objects: Aggregates, Collations, Domains, FTS Collations, FTS Directories, FTS Parameters, FTS Tables, Foreign Keys, and Functions. The Functions section is expanded, showing several functions starting with 'add\_'. The main area displays a query window with the following content:

```
4 select add_hdds();  
5 select add_ssds();  
6 select add_ssdsm2();  
7 select add_motherboard();
```

The fifth line, 'select add\_ssds();', is highlighted. Below the query window, there are tabs for Data Output, Messages, and Notifications. The Data Output tab shows a table with one row:

|   | add_ssds | void |
|---|----------|------|
| 1 |          |      |

At the bottom of the interface, status bars show 'Total rows: 1 of 1', 'Query complete 00:00:00.058', and 'Ln 6, Col 1'.

Рисунок 4.5 – Отображение результата выполненного запроса функции  
add\_ssds

The screenshot shows the MySQL Workbench interface. The left sidebar lists database objects: Aggregates, Collations, Domains, FTS Columns, FTS Directories, FTS Parameters, FTS Tables, Foreign Keys, and Functions. The Functions section is expanded, showing several functions starting with 'add\_'. The main area displays a query window with the following SQL code:

```
5 select add_ssds();  
6 select add_ssdsm2();  
7 select add_motherboards  
8
```

The result pane shows one row of data for the function 'add\_ssdsm2':

|   | add_ssdsm2 | void |
|---|------------|------|
| 1 |            |      |

At the bottom, status information indicates: Total rows: 1 of 1, Query complete 00:00:00.048, and Ln 7, Col 1.

Рисунок 4.6 – Отображение результата выполненного запроса функции add\_ssdsm2

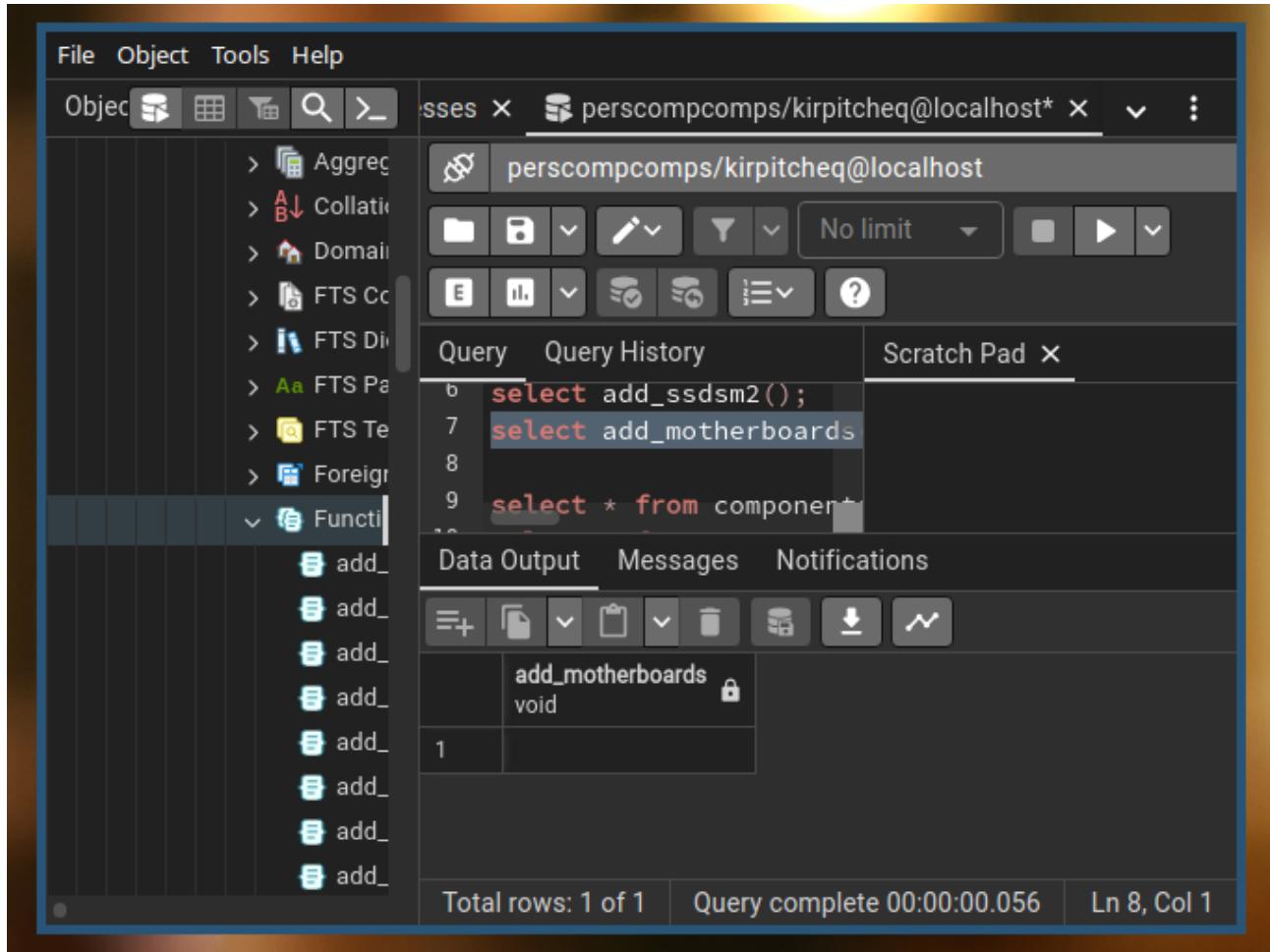


Рисунок 4.7 – Отображение результата выполненного запроса функции add\_motherboards

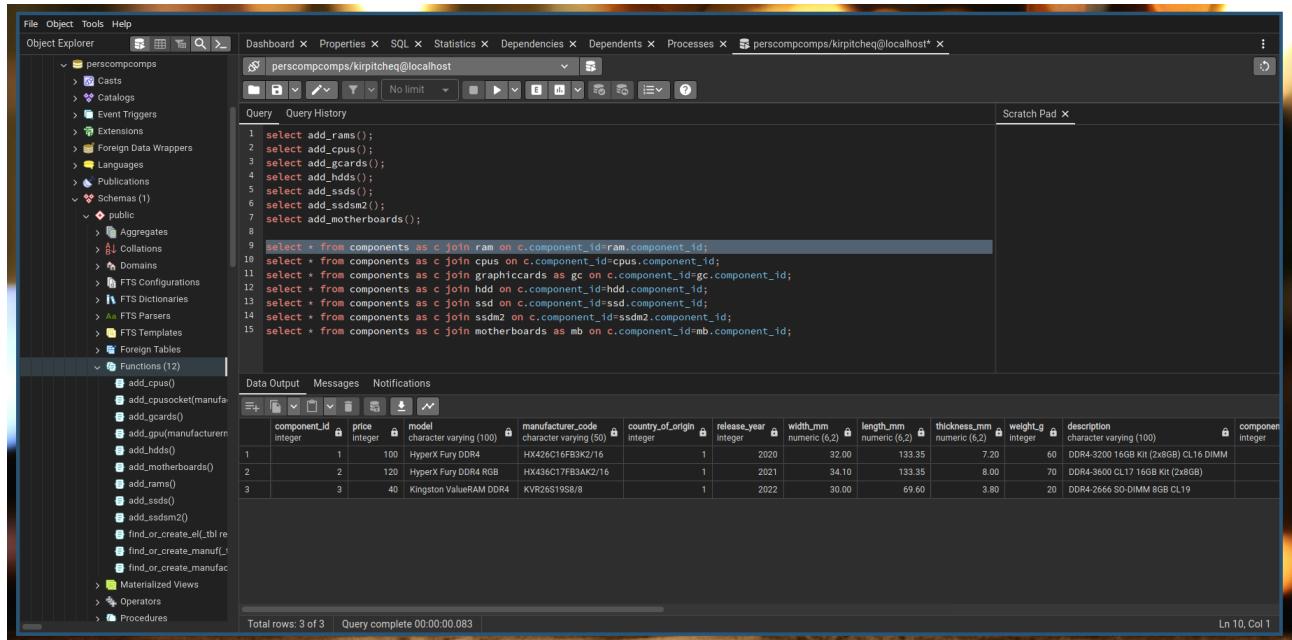


Рисунок 4.8 – Отображение результата выполненного запроса получения данных из таблицы rams с объединением данных от наследуемой таблицы components

The screenshot shows a database interface with the following details:

- Object Explorer:** Shows the schema structure of the database, including Schemas (1), Functions (12), and other objects.
- Query Editor:** Displays the SQL query being executed:
 

```

1 select add_rams();
2 select add_cpus();
3 select add_gcards();
4 select add_hdds();
5 select add_ssds();
6 select add_ssdm2();
7 select add_motherboards();
8
9 select * from components as c join ram on c.component_id=ram.component_id;
10 select * from components as c join cpus on c.component_id=cpus.component_id;
11 select * from components as c join graphiccards as gc on c.component_id=gc.component_id;
12 select * from components as c join hdd on c.component_id=hdd.component_id;
13 select * from components as c join ssd on c.component_id=ssd.component_id;
14 select * from components as c join ssdm2 on c.component_id=ssdm2.component_id;
15 select * from components as c join motherboards as mb on c.component_id=mb.component_id;
      
```
- Data Output:** Shows the results of the query, which include two rows of data. The columns are component\_id, price, model, manufacturer\_code, country\_of\_origin, release\_year, width\_mm, length\_mm, thickness\_mm, weight\_g, description, and component\_id. The data is as follows:
 

| component_id | price | model              | manufacturer_code | country_of_origin | release_year | width_mm | length_mm | thickness_mm | weight_g | description                  | component_id                 |   |
|--------------|-------|--------------------|-------------------|-------------------|--------------|----------|-----------|--------------|----------|------------------------------|------------------------------|---|
| 1            | 4     | 589 Core i9-13900K | BX8071513900K     | 1                 | 2022         | 37.50    | 37.50     | 4.40         | 100      | 24-ядерный процессор 3.0 ГГц | 4                            |   |
| 2            | 5     | 699 Ryzen 9 7950X  | 100-000000514     |                   | 1            | 2022     | 40.00     | 40.00        | 4.40     | 70                           | 16-ядерный процессор 4.5 ГГц | 5 |
- Messages:** Shows "Total rows: 2 of 2 | Query complete 00:00:00.077 | Ln 11, Col 1".

Рисунок 4.9 – Отображение результата выполненного запроса получения данных из таблицы cpus с объединением данных от наследуемой таблицы components

The screenshot shows a database interface with the following details:

- Object Explorer:** Shows the schema structure of the database, including Schemas (1), Functions (12), and other objects.
- Query Editor:** Displays the SQL query being executed:
 

```

1 select add_rams();
2 select add_cpus();
3 select add_gcards();
4 select add_hdds();
5 select add_ssds();
6 select add_ssdm2();
7 select add_motherboards();
8
9 select * from components as c join ram on c.component_id=ram.component_id;
10 select * from components as c join cpus on c.component_id=cpus.component_id;
11 select * from components as c join graphiccards as gc on c.component_id=gc.component_id;
12 select * from components as c join hdd on c.component_id=hdd.component_id;
13 select * from components as c join ssd on c.component_id=ssd.component_id;
14 select * from components as c join ssdm2 on c.component_id=ssdm2.component_id;
15 select * from components as c join motherboards as mb on c.component_id=mb.component_id;
      
```
- Data Output:** Shows the results of the query, which include two rows of data. The columns are component\_id, price, model, manufacturer\_code, country\_of\_origin, release\_year, width\_mm, length\_mm, thickness\_mm, weight\_g, description, and component\_id. The data is as follows:
 

| component_id | price | model                           | manufacturer_code             | country_of_origin | release_year | width_mm | length_mm | thickness_mm | weight_g | description                  | component_id             |   |
|--------------|-------|---------------------------------|-------------------------------|-------------------|--------------|----------|-----------|--------------|----------|------------------------------|--------------------------|---|
| 1            | 6     | 1599 ROG Strix GeForce RTX 4090 | ROG-STRIX-RTX4090-024G-GAMING | 1                 | 2022         | 140.10   | 357.60    | 70.10        | 2189     | GeForce RTX 4090 24GB GDDR6X | 6                        |   |
| 2            | 7     | 1999 AMD Radeon RX 7900XTX      | RX7900XTX-24G                 |                   | 3            | 2023     | 140.00    | 360.00       | 60.00    | 2100                         | ASRock Radeon RX 7900XTX | 7 |
- Messages:** Shows "Total rows: 2 of 2 | Query complete 00:00:00.091 | Ln 12, Col 1".

Рисунок 4.10 – Отображение результата выполненного запроса получения данных из таблицы graphiccards с объединением данных от наследуемой таблицы components

The screenshot shows a database interface with the following details:

- Object Explorer:** Shows the schema structure, including tables like `Components`, `RAM`, `CPU`, `GPU`, `HDD`, `SSD`, and `Motherboards`.
- Query Pad:** Displays the SQL query used to retrieve data from the `hdd` table:
 

```

1 select add_hdds();
2 select add_cpus();
3 select add_gcards();
4 select add_hdds();
5 select add_ssds();
6 select add_ssdm2();
7 select add_motherboards();
8
9 select * from components as c join ram on c.component_id=ram.component_id;
10 select * from components as c join cpus on c.component_id=cpus.component_id;
11 select * from components as c join graphicards as gc on c.component_id=gc.component_id;
12 select * from components as c join hdd on c.component_id=hdd.component_id;
13 select * from components as c join ssd on c.component_id=ssd.component_id;
14 select * from components as c join ssdm2 on c.component_id=ssdm2.component_id;
15 select * from components as c join motherboards as mb on c.component_id=mb.component_id;
      
```
- Data Output:** Shows the results of the query, which include three rows of data for the `hdd` table.

| component_id | price | model | manufacturer_code     | country_of_origin | release_year | width_mm | length_mm | thickness_mm | weight_g | description | component_id                 |
|--------------|-------|-------|-----------------------|-------------------|--------------|----------|-----------|--------------|----------|-------------|------------------------------|
| 1            | 8     | 529   | IronWolf Pro 1TB      | ST1800NE000       | 1            | 2022     | 101.60    | 146.99       | 26.11    | 705         | 18TB 7200RPM SATA 6Gb/s 3.5" |
| 2            | 9     | 129   | Seagate Barracuda 4TB | ST4000DM004       | 2            | 2021     | 101.60    | 146.99       | 26.11    | 705         |                              |
| 3            | 10    | 249   | WD Black 6TB          | WD6003FBX         | 4            | 2020     | 101.60    | 146.99       | 26.11    | 705         |                              |

Рисунок 4.11 – Отображение результата выполненного запроса получения данных из таблицы `hdd` с объединением данных от наследуемой таблицы `components`

The screenshot shows a database interface with the following details:

- Object Explorer:** Shows the schema structure, including tables like `Components`, `RAM`, `CPU`, `GPU`, `HDD`, `SSD`, and `Motherboards`.
- Query Pad:** Displays the SQL query used to retrieve data from the `ssd` table:
 

```

1 select add_ssds();
2 select add_cpus();
3 select add_gcards();
4 select add_hdds();
5 select add_ssds();
6 select add_ssdm2();
7 select add_motherboards();
8
9 select * from components as c join ram on c.component_id=ram.component_id;
10 select * from components as c join cpus on c.component_id=cpus.component_id;
11 select * from components as c join graphicards as gc on c.component_id=gc.component_id;
12 select * from components as c join hdd on c.component_id=hdd.component_id;
13 select * from components as c join ssd on c.component_id=ssd.component_id;
14 select * from components as c join ssdm2 on c.component_id=ssdm2.component_id;
15 select * from components as c join motherboards as mb on c.component_id=mb.component_id;
      
```
- Data Output:** Shows the results of the query, which include six rows of data for the `ssd` table.

| componentId | price | model | manufacturer_code  | country_of_origin | release_year | width_mm | length_mm | thickness_mm | weight_g | description       | componentId                  |
|-------------|-------|-------|--------------------|-------------------|--------------|----------|-----------|--------------|----------|-------------------|------------------------------|
| 1           | 11    | 229   | Samsung 870 EVO    | MZ-77E1T0BW/EU    | 1            | 2020     | 69.85     | 100.00       | 6.80     | 45                | 2TB PCIe Gen 4.0 x4 NVMe M.2 |
| 2           | 12    | 179   | Toshiba X300       | HDWF180UZSVA      | 6            | 2021     | 101.60    | 147.00       | 450      | 8TB SATA III 3.5" |                              |
| 3           | 13    | 149   | Gigabyte AORUS RGB | GP-A641TB         | 8            | 2020     | 80.15     | 22.15        | 2.38     | 16                | 1TB PCIe Gen 4.0 x4 NVMe M.2 |
| 4           | 14    | 229   | 980 PRO 2TB        | MZ-V8P2T0BW       | 1            | 2020     | 22.15     | 80.15        | 2.38     | 9                 | 2TB PCIe Gen 4.0 x4 NVMe M.2 |
| 5           | 15    | 149   | KC2500 1TB         | KC2500/1000G      | 8            | 2021     | 22.15     | 80.15        | 2.38     | 9                 | 1TB NVMe PCIe Gen 3.0 x4 M.2 |
| 6           | 16    | 79    | NE-512             | NE-512G           | 1            | 2020     | 22.15     | 80.15        | 2.38     | 9                 | 512GB SATA III M.2           |

Рисунок 4.12 – Отображение результата выполненного запроса получения данных из таблицы `ssd` с объединением данных от наследуемой таблицы `components`

```

1 select add_rams();
2 select add_cpus();
3 select add_gcards();
4 select add_hdds();
5 select add_ssds();
6 select add_ssdm2();
7 select add_motherboards();
8
9 select * from components as c join ram on c.component_id=ram.component_id;
10 select * from components as c join cpus on c.component_id=cpus.component_id;
11 select * from components as c join graphcards as gc on c.component_id=gc.component_id;
12 select * from components as c join hdd on c.component_id=hdd.component_id;
13 select * from components as c join ssd on c.component_id=ssd.component_id;
14 select * from components as c join ssdm2 on c.component_id=ssdm2.component_id;
15 select * from components as c join motherboards as mb on c.component_id=mb.component_id;

```

|   | component_id | price | model       | manufacturer_code | country_of_origin | release_year | width_mm | length_mm | thickness_mm | weight_g | description                  | component_id       |    |
|---|--------------|-------|-------------|-------------------|-------------------|--------------|----------|-----------|--------------|----------|------------------------------|--------------------|----|
| 1 | 14           | 229   | 980 PRO 2TB | MZ-V8P2T0BW       | 1                 | 2020         | 22.15    | 80.15     | 2.38         | 9        | 2TB PCIe Gen 4.0 x4 NVMe M.2 | 14                 |    |
| 2 | 15           | 149   | KC2500 1TB  | KC2500/1000G      | 8                 | 2021         | 22.15    | 80.15     | 2.38         | 9        | 1TB NVMe PCIe Gen 3.0 x4 M.2 | 15                 |    |
| 3 | 16           | 79    | NE-512      | NE-512G           |                   | 1            | 2020     | 22.15     | 80.15        | 2.38     | 9                            | 512GB SATA III M.2 | 16 |

Рисунок 4.13 – Отображение результата выполненного запроса получения данных из таблицы ssdm2 с объединением данных от наследуемой таблицы components

```

9 select * from components as c join ram on c.component_id=ram.component_id;
10 select * from components as c join cpus on c.component_id=cpus.component_id;
11 select * from components as c join graphcards as gc on c.component_id=gc.component_id;
12 select * from components as c join hdd on c.component_id=hdd.component_id;
13 select * from components as c join ssd on c.component_id=ssd.component_id;
14 select * from components as c join ssdm2 on c.component_id=ssdm2.component_id;
15
16 select * from components as c join nonvolatilemem as nvm on c.component_id=nvm.component_id;
17
18 select * from components as c join motherboards as mb on c.component_id=mb.component_id;
19
20 select * from components;

```

|   | component_id | price | model                 | manufacturer_code | country_of_origin | release_year | width_mm | length_mm | thickness_mm | weight_g | description         | component_id |
|---|--------------|-------|-----------------------|-------------------|-------------------|--------------|----------|-----------|--------------|----------|---------------------|--------------|
| 1 | 8            | 529   | IronWolf Pro 1TB      | ST1800NE000       | 1                 | 2022         | 101.60   | 146.99    | 26.11        | 705      | 18TB 7200RPM SAT.   |              |
| 2 | 9            | 129   | Seagate BarraCuda 4TB | ST4000DM004       | 2                 | 2021         | 101.60   | 146.99    | 26.11        | 705      |                     |              |
| 3 | 10           | 249   | WD Black 6TB          | WD6003FZBX        | 4                 | 2020         | 101.60   | 146.99    | 26.11        | 705      |                     |              |
| 4 | 11           | 229   | Samsung 870 EVO       | MZ-77E1T0BW/EU    | 1                 | 2020         | 69.85    | 100.00    | 6.80         | 45       | 2TB PCIe Gen 4.0 x4 |              |
| 5 | 12           | 179   | Toshiba X300          | HDW-F180UZSVA     | 6                 | 2021         | 101.60   | 147.00    | 26.10        | 450      | 8TB SATA III 3.5"   |              |
| 6 | 13           | 149   | Gigabyte AORUS RGB    | GP-A641TB         | 8                 | 2020         | 80.15    | 22.15     | 2.38         | 16       | 1TB PCIe Gen 4.0 x4 |              |
| 7 | 14           | 229   | 980 PRO 2TB           | MZ-V8P2T0BW       | 1                 | 2020         | 22.15    | 80.15     | 2.38         | 9        | 2TB PCIe Gen 4.0 x4 |              |
| 8 | 15           | 149   | KC2500 1TB            | KC2500/1000G      | 8                 | 2021         | 22.15    | 80.15     | 2.38         | 9        | 1TB NVMe PCIe Gen   |              |
| 9 | 16           | 79    | NE-512                | NE-512G           | 1                 | 2020         | 22.15    | 80.15     | 2.38         | 9        | 512GB SATA III M.2  |              |

Рисунок 4.14 – Отображение результата выполненного запроса получения данных из таблицы nonvolatilemem с объединением данных от наследуемой таблицы components

```

File Object Tools Help
Object Explorer Dashboard Properties SQL Statistics Dependencies Dependents Processes perscompcomps/kirpitcheq@localhost*
Query Query History
1 select add_rams();
2 select add_cpus();
3 select add_gcards();
4 select add_hdds();
5 select add_ssds();
6 select add_ssdm2();
7 select add_motherboards();
8
9 select * from components as c join ram on c.component_id=ram.component_id;
10 select * from components as c join cpus on c.component_id=cpus.component_id;
11 select * from components as c join graphiccards as gc on c.component_id=gc.component_id;
12 select * from components as c join hdd on c.component_id=hdd.component_id;
13 select * from components as c join ssd on c.component_id=ssd.component_id;
14 select * from components as c join ssdm2 on c.component_id=ssdm2.component_id;
15 select * from components as c join motherboards as mb on c.component_id=mb.component_id;

```

| component_id | price | model                           | manufacturer_code           | country_of_origin | release_year | width_mm | length_mm | thickness_mm | weight_g | description         |
|--------------|-------|---------------------------------|-----------------------------|-------------------|--------------|----------|-----------|--------------|----------|---------------------|
| 17           | 699   | ROG Maximus Z790 Hero           | ROG MAXIMUS Z790 HERO       |                   | 1            | 2022     | 305.00    | 244.00       | 46.00    | 1850 Intel Z790 ATX |
| 18           | 189   | GIGABYTE B550 AORUS ELITE       | B550 AORUS ELITE            |                   | 8            | 2021     | 305.00    | 244.00       | 46.00    | 1850 AMD B550 ATX   |
| 19           | 279   | MSI MPG Z590 GAMING CARBON WIFI | MPG Z590 GAMING CARBON WIFI |                   | 1            | 2021     | 305.00    | 244.00       | 46.00    | 1850 Intel Z590 ATX |

Total rows: 3 of 3    Query complete 00:00:00.117    Ln 15, Col 89

Рисунок 4.15 – Отображение результата выполненного запроса получения данных из таблицы motherboards с объединением данных от наследуемой таблицы components

```

File Object Tools Help
Object Explorer Dashboard Properties SQL Statistics Dependencies Dependents Processes perscompcomps/kirpitcheq@localhost*
Query Query History
9 select * from components as c join ram on c.component_id=ram.component_id;
10 select * from components as c join cpus on c.component_id=cpus.component_id;
11 select * from components as c join graphiccards as gc on c.component_id=gc.component_id;
12 select * from components as c join hdd on c.component_id=hdd.component_id;
13 select * from components as c join ssd on c.component_id=ssd.component_id;
14 select * from components as c join ssdm2 on c.component_id=ssdm2.component_id;
15 select * from components as c join motherboards as mb on c.component_id=mb.component_id;
16
17 select * from components;

```

| component_id | price | model                           | manufacturer_code             | country_of_origin | release_year | width_mm | length_mm | thickness_mm | weight_g | description                        |
|--------------|-------|---------------------------------|-------------------------------|-------------------|--------------|----------|-----------|--------------|----------|------------------------------------|
| 1            | 100   | HyperX Fury DDR4                | HX426C16FBK2/16               |                   | 1            | 2020     | 32.00     | 133.35       | 7.20     | 60 DDR4-3200 16GB Kit (2x8GB) CL   |
| 2            | 120   | HyperX Fury DDR4 RGB            | HX436C17FB3AK2/16             |                   | 1            | 2021     | 34.10     | 133.35       | 8.00     | 70 DDR4-3600 CL17 16GB Kit (2x8GB) |
| 3            | 40    | Kingston ValueRAM DDR4          | KVR26S19SS8/8                 |                   | 1            | 2022     | 30.00     | 69.60        | 3.80     | 20 DDR4-2666 SO-DIMM 8GB CL19      |
| 4            | 589   | Core i9-13900K                  | BX8071513900K                 |                   | 1            | 2022     | 37.50     | 37.50        | 4.40     | 100 24-ядерный процессор 3.0 ГГц   |
| 5            | 699   | Ryzen 9 7950X                   | 100-000000514                 |                   | 1            | 2022     | 40.00     | 40.00        | 4.40     | 70 16-ядерный процессор 4.5 ГГц    |
| 6            | 1599  | ROG Strix GeForce RTX 4090      | ROG-STRIX-RTX4090-024G-GAMING |                   | 1            | 2022     | 140.10    | 357.60       | 70.10    | 2189 GeForce RTX 4090 24Gb GDDR6   |
| 7            | 1999  | AMD Radeon RX 7900XTX           | RX7900XTX-24G                 |                   | 3            | 2023     | 140.00    | 300.00       | 60.00    | 2100 ASRock Radeon RX 7900XTX      |
| 8            | 17    | ROG Maximus Z790 Hero           | ROG MAXIMUS Z790 HERO         |                   | 1            | 2022     | 305.00    | 244.00       | 46.00    | 1850 Intel Z790 ATX                |
| 9            | 189   | GIGABYTE B550 AORUS ELITE       | B550 AORUS ELITE              |                   | 8            | 2021     | 305.00    | 244.00       | 46.00    | 1850 AMD B550 ATX                  |
| 10           | 279   | MSI MPG Z590 GAMING CARBON WIFI | MPG Z590 GAMING CARBON WIFI   |                   | 1            | 2021     | 305.00    | 244.00       | 46.00    | 1850 Intel Z590 ATX                |
| 11           | 8     | 529 IronWolf Pro 1TB            | ST1000NE000                   |                   | 1            | 2022     | 101.60    | 146.99       | 26.11    | 705 1TB 7200RPM SATA 6Gb/s 3.5"    |
| 12           | 9     | Seagate Barracuda 4TB           | ST4000DM004                   |                   | 2            | 2021     | 101.60    | 146.99       | 26.11    | 705                                |
| 13           | 10    | WD Black 6TB                    | WD6003FZBX                    |                   | 4            | 2020     | 101.60    | 146.99       | 26.11    | 705                                |
| 14           | 11    | Samsung 870 EVO                 | MZ-77E1T0BW/EU                |                   | 1            | 2020     | 69.85     | 100.00       | 6.80     | 45 2TB PCIe Gen 4.0 x4 NVMe M.2    |
| 15           | 12    | Toshiba X300                    | HDWF180UZSVA                  |                   | 6            | 2021     | 101.60    | 147.00       | 26.10    | 450 8TB SATA III 3.5"              |
| 16           | 13    | Gigabyte AORUS RGB              | GP-AG411TB                    |                   | 8            | 2020     | 80.15     | 22.15        | 2.38     | 16 1TB PCIe Gen 4.0 x4 NVMe M.2    |
| 17           | 14    | 980 PRO 2TB                     | MZ-V8P2T0BW                   |                   | 1            | 2020     | 22.15     | 80.15        | 2.38     | 9 2TB PCIe Gen 4.0 x4 NVMe M.2     |
| 18           | 15    | KC2500 1TB                      | KC2500/1000G                  |                   | 8            | 2021     | 22.15     | 80.15        | 2.38     | 9 1TB NVMe PCIe Gen 3.0 x4 M.2     |
| 19           | 16    | NE-512                          | NE-512G                       |                   | 1            | 2020     | 22.15     | 80.15        | 2.38     | 9 512GB SATA III M.2               |

Total rows: 19 of 19    Query complete 00:00:00.143    Ln 17, Col 26

Рисунок 4.16 – Отображение результата выполненного запроса получения данных из таблицы components

The screenshot shows the pgAdmin 4 interface with the following details:

- File Object Tools Help**: The top menu bar.
- Processes X perscompcomps/kirpitcheq@localhost\***: The title bar of the active connection window.
- Explorer**: The left sidebar showing database objects like Aggregates, Collations, Domains, FTS Configs, FTS Dictionaries, FTS Parameters, FTS Tables, Foreign Tables, Functions, Materials, and Operators.
- Query History**: A tabbed section showing the history of queries run on the current connection.
- Scratch Pad X**: A tabbed section for temporary scratch work.
- Query**: The active tab where the SQL query `select * from countries;` is entered.
- Data Output**: The main pane displaying the results of the query as a table.
- Messages Notifications**: Tabs for monitoring system messages and notifications.
- Table Results**: The data output table with columns `id` and `name`. The data is as follows:

|   | <code>id</code><br>[PK] integer | <code>name</code><br>character varying (100) |
|---|---------------------------------|--|
| 1 | 1                               | CHINA  |
| 2 | 2                               | USA  |
| 3 | 3                               | Taiwan                                       |
| 4 | 4                               | Thailand                                     |
| 5 | 5                               | South Korea                                  |
| 6 | 6                               | PHILIPPINES                                  |
| 7 | 7                               | Japan  |
| 8 | 8                               | TAIWAN                                       |
| 9 | 9                               | China  |

**Total rows: 9 of 9    Query complete 00:00:00.112    Ln 22, Col 25**

Рисунок 4.17 – Отображение результата выполненного запроса получения данных из таблицы countries

The screenshot shows a PostgreSQL client interface with the following details:

- File Object Tools Help**: The menu bar at the top.
- Explorer**: A sidebar on the left listing database objects: Aggregates, Collations, Domains, FTS Configs, FTS Dictionaries, FTS Parameters, FTS Tables, Foreign Tables, Functions, and Views.
- Connections**: A list of connections on the right, showing "perscompcomps/kirpitcheq@localhost\*".
- Query History**: A tabbed interface with "Query History" selected, showing the query `select * from ddrtypes;`.
- Data Output**: The main pane displaying the results of the query:

|   | <b>Id</b><br>[PK] integer | <b>name</b><br>character varying (50) | <b>speed_mhz</b><br>integer |
|---|---------------------------|---------------------------------------|-----------------------------|
| 1 | 1                         | DDR4                                  | [null]                      |
| 2 | 2                         | DDR5                                  | [null]                      |

- Messages**: Shows "Total rows: 2 of 2" and "Query complete 00:00:00.095".
- Notifications**: Shows "Ln 24, Col 24".

Рисунок 4.18 – Отображение результата выполненного запроса получения данных из таблицы ddrtypes

The screenshot shows the pgAdmin 4 interface. On the left is a tree view of database objects under the 'Tables' node. In the center is a query editor window with the following content:

```
* from manufacturers;
```

Below the query editor is a data grid displaying the results of the query. The columns are:

|    | <b>Id</b><br>[PK] integer | <b>name</b><br>character varying (100) | <b>country_id</b><br>integer |
|----|---------------------------|--|------------------------------|
| 1  | 1                         | HyperX                                 | 2                            |
| 2  | 5                         | NVIDIA                                 | 2                            |
| 3  | 7                         | ASRock                                 | 3                            |
| 4  | 8                         | Seagate                                | 2                            |
| 5  | 9                         | Western Digital                        | 2                            |
| 6  | 11                        | Toshiba                                | 7                            |
| 7  | 12                        | Gigabyte                               | 3                            |
| 8  | 10                        | Samsung                                | 5                            |
| 9  | 2                         | Kingston                               | 8                            |
| 10 | 13                        | KingSpec                               | 1                            |
| 11 | 6                         | ASUS                                   | 3                            |
| 12 | 14                        | GIGABYTE                               | 3                            |
| 13 | 4                         | AMD                                    | 2                            |
| 14 | 15                        | MSI                                    | 9                            |
| 15 | 3                         | Intel                                  | 2                            |

At the bottom of the data grid, it says "Total rows: 15 of 15". Below the data grid, the status bar shows "Query complete 00:00:00.068" and "Ln 28, Col 29".

Рисунок 4.19 – Отображение результата выполненного запроса получения данных из таблицы manufacturers

The screenshot shows the pgAdmin 4 interface. The left sidebar displays the Object Explorer with various database objects like add\_, find\_, and Tables. The main area shows a query editor with the following SQL code:

```
25
26 select * from cpusockets;
```

The results pane displays a table with four rows of data:

|   | <b>id</b><br>[PK] integer | <b>name</b><br>character varying (50) | <b>id_manufacturer</b><br>integer |
|---|---------------------------|---------------------------------------|-----------------------------------|
| 1 | 1                         | LGA 1700                              | 1                                 |
| 2 | 2                         | AM5                                   | 2                                 |
| 3 | 3                         | AM4                                   | 2                                 |
| 4 | 4                         | LGA 1200                              | 1                                 |

At the bottom, it says "Total rows: 4 of 4" and "Query complete 00:00:00.073". The status bar indicates "Ln 26, Col 26".

Рисунок 4.20 – Отображение результата выполненного запроса получения данных из таблицы cpusockets

The screenshot shows the pgAdmin 4 interface. The left sidebar displays the Object Explorer with various database objects like add\_, find\_, and Tables (45). The main area shows a query editor with the following SQL code:

```
17
18 select * from components as c join motherboards as mb on c.component_id=mb.component_id;
19
```

The results pane displays a table with two rows of data:

| <b>id</b><br>[PK] integer | <b>name</b><br>character varying (100) | <b>id_manufacturer</b><br>integer | <b>microarchitecture</b><br>character varying (50) | <b>technology_process</b><br>character varying (10) | <b>base_clock_speed</b><br>integer | <b>turbo_clock_speed</b><br>integer | <b>alu_count</b><br>integer | <b>texture_block_count</b><br>integer |
|---------------------------|--|-----------------------------------|--|---|------------------------------------|-------------------------------------|-----------------------------|---------------------------------------|
| 1                         | NVIDIA RTX 4090                        | 1                                 | NVIDIA Ada Lovelace                                | 5nm   | 2235                               | 2520                                | 16384                       |                                       |
| 2                         | AMD Radeon RX 7900 XTX                 | 2                                 | AMD RDNA3  | 5nm   | 1900                               | 2500                                | 6144                        |                                       |

At the bottom, it says "Total rows: 2 of 2" and "Query complete 00:00:00.091". The status bar indicates "Ln 31, Col 1".

Рисунок 4.21 – Отображение результата выполненного запроса получения данных из таблицы gpus

## ЗАКЛЮЧЕНИЕ

В ходе выполнения курсовой работы были выполнены следующие задачи:

- выполнен анализ предметной области, формализованы данные, рассмотрены типы баз данных
- спроектированы атрибуты баз данных, рассмотрена иерархия, разработана диаграмма сущность-связь
- сформированы требования, предъявляемые к СУБД, выбрана конкретная наиболее пригодная
- на основе спроектированных решений разработаны скрипты создания базы данных, вспомогательных функций
- был разработан исследовательский скрипт заполняющий базу данных записями, использующие созданные функции, проверена корректность работы функций, исследованы полученные ожидаемые результаты сформированных данных, их связей, системы наследования, а также корректность работы функций

## **СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

1. Oracle СНГ[Электронный ресурс]. — URL: <https://www.oracle.com/cis/database/what-is-database/> (дата обращения: 05.01.2024).
2. БГЭУ[Электронный ресурс]. — URL: <http://bseu.by/it/tohod/lekciis.htm> (дата обращения: 05.01.2024).
3. Сайт Владимира Драча[Электронный ресурс]. — URL: <https://drach.pro/blog/hi-tech/item/196-popular-relational-dbms-2022> (дата обращения: 05.01.2024).
4. Чем PostgreSQL лучше других SQL баз данных с открытым исходным кодом. Часть 1 [Электронный ресурс]. — URL: <https://drach.pro/blog/hi-tech/item/196-popular-relational-dbms-2022> (дата обращения: 05.01.2024).