## Учреждение образования

## Белорусский государственный технологический университет

Кафедра полиграфического оборудования и

системы обработки информации

**ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 9**

Тема

«*Ознакомление со стандартом SQL*»

по дисциплине «Стандартизация и сертификация

информационных систем и технологий»

Выполнил

ФИТ 4 курс 7 группа Бобрович Г.С.

Проверил

кандидат технических наук

Сулим П.Е.

Отчет по лабораторной работе

защищен с отметкой баллов

Минск 2024

[**Стандарт *SQL*.**](https://aws.amazon.com/ru/what-is/sql/#:~:text=Стандарты%20SQL%20–%20это%20набор%20формально,стандарты%20SQL%20в%201986%20году.)

Язык структурированных запросов (SQL) – это язык программирования для хранения и обработки информации в реляционной базе данных. Реляционная база данных хранит информацию в табличной форме со строками и столбцами, представляющими различные атрибуты данных и различные связи между значениями данных. Инструкции SQL можно использовать для хранения, обновления, удаления, поиска и извлечения информации из базы данных. Можно также использовать SQL для поддержания и оптимизации производительности базы данных.

Решение SQL было изобретено в 1970-х годах на основе реляционной модели данных. Изначально оно было известен как структурированный английский язык запросов (SEQUEL). Позднее этот термин был сокращен до SQL. Компания Oracle, ранее – Relational Software, стала первым поставщиком, предложившим коммерческую систему управления реляционными базами данных SQL.

Системы управления реляционными базами данных используют язык структурированных запросов (SQL) для хранения данных и управления ими. В системе хранится несколько таблиц базы данных, связанных друг с другом. MS SQL Server, MySQL или MS Access являются примерами систем управления реляционными базами данных. Ниже перечислены компоненты такой системы.

### **Таблица SQL**

Таблица SQL – это базовый элемент реляционной базы данных. Таблица базы данных SQL состоит из строк и столбцов. Инженеры баз данных создают связи между несколькими таблицами базы данных, чтобы оптимизировать пространство для хранения данных.

### **Операторы SQL**

Инструкции SQL, или SQL-запросы, являются действительными инструкциями, понятными системам управления реляционными базами данных. Разработчики программного обеспечения создают инструкции SQL, используя различные языковые элементы SQL. Элементы языка SQL – это такие компоненты, как идентификаторы, переменные и условия поиска, которые формируют правильную инструкцию SQL.

### **Хранимые процедуры**

Хранимые процедуры – это набор из одной или нескольких инструкций SQL, хранящихся в реляционной базе данных. Разработчики программного обеспечения используют хранимые процедуры для повышения эффективности и производительности. Например, они могут создать хранимую процедуру обновления таблиц продаж вместо написания одной и той же инструкции SQL в разных приложениях.

Команды языка структурированных запросов (SQL) – это специальные ключевые слова или инструкции SQL, которые разработчики используют для управления данными, хранящимися в реляционной базе данных. Команды SQL можно классифицировать указанным ниже образом.

### **Язык определения данных**

Язык определения данных (DDL) относится к командам SQL, которые проектируют структуру базы данных. Инженеры баз данных используют DDL для создания и изменения объектов базы данных в соответствии с бизнес-требованиями. Например, инженер баз данных использует команду CREATE для создания объектов базы данных, таких как таблицы, представления и индексы.

### **Язык запроса данных**

Язык запросов данных (DQL) состоит из инструкций по извлечению данных, хранящихся в реляционных базах данных. Программные приложения используют команду SELECT для фильтрации и возврата определенных результатов из таблицы SQL.

### **Язык управления данными**

В инструкциях языка управления данными (DML) записывается новая информация или изменяются существующие записи в реляционной базе данных. Например, приложение использует команду INSERT для сохранения новой записи в базе данных.

### **Язык управления данными**

Администраторы баз данных используют язык управления данными (DCL) для управления или авторизации доступа к базе данных для других пользователей. Например, они могут использовать команду GRANT, чтобы разрешить определенным приложениям манипулировать одной или несколькими таблицами.

### **Язык управления транзакциями**

Реляционный механизм использует язык управления транзакциями (TCL) для автоматического внесения изменений в базу данных. Например, база данных использует команду ROLLBACK для отмены ошибочной транзакции.

Стандарты SQL – это набор формально определенных рекомендаций языка структурированных запросов (SQL). Американский национальный институт стандартов (ANSI) и Международная организация по стандартизации (ISO) приняли стандарты SQL в 1986 году. Поставщики программного обеспечения используют стандарты ANSI SQL для создания программного обеспечения баз данных SQL для разработчиков.

[**Диалекты *SQL***](http://bseu.by/it/tohod/lekcii7_4.htm)***.*** [***DSL*.**](https://www.jetbrains.com/ru-ru/mps/concepts/domain-specific-languages/)

Несмотря на наличие международного стандарта ANSI SQL, многие компании, занимающиеся разработкой СУБД, вносят изменения в язык SQL, применяемый в разрабатываемой СУБД, тем самым отступая от стандарта. Каждая из реализаций языка SQL в конкретной СУБД называется *диалектом*. Функции, которые добавляются к стандарту языка разработчиками коммерческих реализаций, принято называть *расширениями*. Например, в стандарте языка SQL определены конкретные типы данных, которые могут храниться в базах данных. Во многих реализациях этот список расширяется за счет разнообразных дополнений.

Выделяют три уровня соответствия стандарту ANSI/ISO — начальный, промежуточный и полный. В настоящее время не существует ни одного диалекта, полностью соответствующего стандарту. Производители СУБД (например, [Oracle](http://ru.wikipedia.org/wiki/Oracle), [Microsoft](http://ru.wikipedia.org/wiki/Microsoft), [Borland](http://ru.wikipedia.org/wiki/Borland), Informix, Sybase) применяют собственные реализации SQL, отвечающие как минимум начальному уровню соответствия стандарту и содержащие некоторые расширения, специфические для данной СУБД.  Не существует двух совершенно идентичных диалектов. Более того, поскольку разработчики баз данных вводят в системы все новые функциональные средства, они постоянно расширяют свои диалекты языка SQL, в результате чего отдельные диалекты все больше и больше отличаются друг от друга. Это имеет свои достоинства и недостатки.

Конкретная реализация языка, может включать в себя более широкие возможности по сравнению со стандартом SQL, например, больше типов данных, большее количество команд, больше дополнительных возможностей у имеющихся команд. Такие возможности делают работу с конкретной СУБД более эффективной. Кроме того, такие нестандартные возможности языка проходят практическую апробацию и со временем могут быть включены в стандарт. Недостаток в том, что различия в синтаксисе реализаций SQL затрудняют перенос приложений из одной системы в другую. Например, если приложение было написано для базы данных MS SQL Server с использованием своего диалекта SQL – языка Transact-SQL, то при переносе системы в базу данных ORACLE, не все конструкции языка будут понятны соответствующему диалекту SQL – языку PL/SQL.

В широко распространенных в настоящее время СУБД используются следующие диалекты языка SQL:

* PL/SQL – в СУБД Oracle;
* Transact-SQL – в СУБД Microsoft SQL;
* Informix-SQL – в СУБД Informix;
* Jet SQL – Microsoft Access.

[Предметно-ориентированные языки (DSL)](https://www.jetbrains.com/ru-ru/mps/concepts/domain-specific-languages/) — это языки программирования с более высоким уровнем абстракции, которые отражают специфику решаемых с их помощью задач. Такие языки оперируют понятиями и правилами из определенной предметной области.

Предметно-ориентированные языки обычно не такие сложные, как языки общего назначения, например Java, C или Ruby. Как правило, предметно-ориентированные языки разрабатываются в сотрудничестве со специалистами в той сфере деятельности, для которой проектируется язык. Зачастую для использования таких языков не требуется квалификация разработчика и на них программируют люди, хорошо разбирающиеся в предметной области.

[DSL](https://sky.pro/wiki/javascript/dsl-v-programmirovanii-chto-eto-primery-i-kak-sozdat/) (Domain-Specific Languages), или языки специального назначения, представляют собой мощный инструмент в арсенале разработчика. В отличие от языков программирования общего назначения, таких как Python или Java, которые предназначены для решения широкого спектра задач, DSL фокусируется на конкретной области, предлагая более простой и интуитивно понятный способ работы.

[Примеры повседневных DSL](https://sky.pro/wiki/javascript/dsl-v-programmirovanii-chto-eto-primery-i-kak-sozdat/)

SQL (Structured Query Language): используется для запросов к базам данных, позволяя эффективно извлекать и обрабатывать данные.

HTML (HyperText Markup Language): определяет структуру и оформление веб-страниц, делая интернет визуально доступным и интерактивным.

CSS (Cascading Style Sheets): используется для описания внешнего вида документа, написанного на языке разметки.

Эти языки делают разработку более доступной и понятной для специалистов в соответствующих областях, позволяя им сосредоточиться на решении задач, а не на технических деталях.

DSL делятся на два основных типа: внутренние (EDSL) и внешние. Внутренние DSL являются частью других языков программирования, расширяя их функциональность для специфических задач. Внешние DSL, в свою очередь, существуют как самостоятельные языки с собственным синтаксисом и правилами.

[Примеры использования](https://sky.pro/wiki/javascript/dsl-v-programmirovanii-chto-eto-primery-i-kak-sozdat/)

* Внутренние DSL: LINQ в C# для работы с данными, DSL в Kotlin для создания андроид-приложений.
* Внешние DSL: CSS для стилизации веб-страниц, SQL для работы с базами данных.

Выбор между внутренним и внешним DSL зависит от специфики задачи и предпочтений разработчика. Внутренние DSL облегчают интеграцию и использование в рамках существующих проектов, в то время как внешние предлагают большую гибкость и специализацию.

[Шаги создания DSL:](https://sky.pro/wiki/javascript/dsl-v-programmirovanii-chto-eto-primery-i-kak-sozdat/)

* Определение целей и задач, которые должен решать DSL.
* Проектирование синтаксиса и структуры языка.
* Реализация интерпретатора или компилятора, который будет обрабатывать код на вашем DSL.
* Тестирование и документирование, чтобы убедиться, что язык работает как задумано и доступен для изучения другими разработчиками.

Использование существующих инструментов и библиотек, таких как Kotlin для создания внутренних DSL, может значительно упростить этот процесс.

[Преимущества и вызовы использования DSL](https://sky.pro/wiki/javascript/dsl-v-programmirovanii-chto-eto-primery-i-kak-sozdat/)

Преимущества DSL включают в себя упрощение и ускорение разработки, сокращение затрат на обучение и поддержку, а также повышение точности и надежности решений. Однако использование и разработка DSL также сопряжены с вызовами, включая сложность поддержки и ограниченную область применения.

[Почему стоит использовать DSL:](https://sky.pro/wiki/javascript/dsl-v-programmirovanii-chto-eto-primery-i-kak-sozdat/)

* Специализация: DSL позволяет решать задачи более эффективно, благодаря фокусу на конкретной области.
* Простота и доступность: Языки специального назначения часто проще для понимания и использования специалистами в данной области.
* Ускорение разработки: Специализированный инструментарий упрощает и ускоряет процесс создания и тестирования решений.

[Возможные сложности:](https://sky.pro/wiki/javascript/dsl-v-programmirovanii-chto-eto-primery-i-kak-sozdat/)

* Ограниченная область применения: DSL хорошо работают в рамках своей специализации, но могут быть неэффективны за ее пределами.
* Сложность поддержки: Разработка и поддержка собственного DSL требует дополнительных ресурсов и знаний.

В заключение, DSL в программировании – это мощный инструмент, который позволяет сосредоточиться на решении конкретных задач, делая процесс разработки более эффективным и приятным. Создание собственного DSL может быть сложным, но оно открывает новые возможности для специализации и оптимизации процессов.

**Вывод:**

В ходе выполнения лабораторной работы был изучен стандарт SQL, его общие понятия и термины, также было выполнено исследование диалектов SQL и предметно-ориентированных языков.