Государственный Университет Молдовы

Факультет Математики и Информатики

Департамент Информатики

“Proiectarea sistemelor informatice”

Лабораторная работа 1

Эссе:“Системы с централизованным и децентрализованным хранением данных”

Проверил: Гладей Анатолий

Выполнил: Чобану Артём

Кишинев 2021

**Централизованное хранение:**

Многие компании вынуждены полагаться на централизованные базы данных для хранения информации своих клиентов. Подобные решения могут стать идеальным и экономически оправданным выбором для малого бизнеса. Но по мере роста компании необходимо расширять инфраструктуру и добавлять дополнительные сервера для обеспечения бесперебойной работы. В результате, масштабирование системы может стать основной статьей расходов. Не следует также забывать о проблемах, которые возникают в результате высокого трафика.

Основная идея централизованных баз данных заключается в том, что один все данные хранятся в одном месте определенной сети. Поскольку резервность данных часто не предусмотрено, при ее потере возобновление практически невозможно и его приходится делать вручную.

Централизованные базы данных подвержены риску взломов и утечки данных. Обычно конфигурация данных настолько слаба, что злоумышленники могут легко ее использовать в своих целях. Очень часто нет необходимости использовать сложные подходы к взлому из-за изначально небезопасной базы данных. Даже при наличии системы защиты можно использовать метод перебора возможных решений для взлома слабых или дефолтных имен пользователей и паролей, прибегнуть к уязвимостям базы данных, применить SQL-инъекцию или украсть резервную копию. Результатом неправомерных действий может быть идентификация личности по хранимых данных, что GDPR рассматривает как грубое нарушение закона.

Используя слабо защищенную базу данных, бизнес не может гарантировать безопасность пользовательских данных. GDPR очень строг в отношении защиты данных и вероятность нарушений ставит компанию под угрозу. Даже при соблюдении законов, одна успешная хакерская атака может обойтись в миллионные штрафы и поставить бизнес в опасность.

**Децентрализованное хранение с использованием block-chain:**

Децентрализованные решения – более надежный и безопасный способ взаимодействия с пользовательскими данными. Если централизованное хранилище использует единый сервер для хранения данных, в децентрализованном информация хранится на нескольких серверах, за счет чего данные можно легко восстановить. И когда мы добавляем блокчейн, мы получаем безопасное хранилище для любых данных, включая данные ваших пользователей и клиентов.

Анонимность пользователей и защита данных лежат в основе как GDPR, так и блокчейна. Последний призван быть безопасным и скрытым. Открытые и закрытые ключи позволяют участникам отправлять и получать данные почти с абсолютной анонимностью. Публичный ключ не привязан к конкретному человеку, а информация хэшируется, поэтому не может быть использована для идентификации его личности. В этом плане блокчейн соответствует требованиям GDPR.

Само понятие блокчейна подразумевает отсутствие единой уязвимости. Данные хранятся на многих нодах по всему миру, поэтому мы получаем распределенную систему без единой точки отказа. Компании, которые используют информацию с блокчейна, не будут работать под угрозой утечки данных. Вероятность сводится к минимуму, а то и вовсе равна нулю.

Практически неограниченная масштабируемость является главным основанных на блокчейне систем. Если при использовании централизованной базы данных вам нужно докупать новые сервера, то используя блокчейн, вам не придется выделять средства на их покупку и обслуживание. В результате, стоимость сервиса для вашей компании значительно снижается.

Различные блокчейн-компании предоставляют инструменты для создания собственных решений на основе блокчейна. Необходимо только нанять квалифицированных специалистов и оплатить транзакции. Цена разработки подобного решения может быть ниже цены обеспечения работы централизованного хранилища. Более того, оно нивелирует риск взломов, которые могут привести к многомиллионным штрафам.

Emercoin также работает над решением, которое поможет компаниям соответствовать требованиям GDPR. Блокчейн предоставляет более надежные решения для сбора, хранения и безопасности данных, поскольку он соответствует принципам «privacy by design», представленным в Общем регламенте защиты данных.

**Подробнее о блокчейне**

Блокчейн переводится с английского как “цепочка блоков. Каждый блок хранит часть информации из предыдущего блока. В частности, туда входит хеш предыдущего блока, который работает как уникальный цифровой отпечаток.

Вероятность того, что два фрагмента данных дадут вам одинаковый результат хеш-функции, невероятно мала. То есть, если кто-то попытается изменить старый блок, его хеш также изменится, а значит, и хеш следующего блока тоже будет другим, и так далее. Следовательно, понять, был ли изменен блок, очень просто, ведь изменены будут и все следующие за ним блоки.

Важно отметить необходимость полной загрузки блокчейна на накопитель участника. Помните, мы говорили о том, что любой желающий может осуществлять валидацию (проверку) транзакций и подписей с помощью криптографии с открытым ключом? Когда нода получает блок, она выполняет ряд проверок; если что-то является недопустимым, блок отклоняется.

Когда нода получает валидный блок, она копирует его и распространяет этот блок на другие ноды. Они, в свою очередь, делают то же самое, пока блок не распространится по всей сети. Такой же процесс выполняется и для неподтвержденных транзакций, то есть тех, которые были объявлены, но еще не добавились в блокчейн.

Как блоки добавляются в блокчейн?

Поскольку система держится на взаимосвязанных между собой блоках, целостность всей блокчейн-сети подрывается в случае записи хотя бы одной ложной информации. В то же время в такой распределенной системе отсутствует администратор или руководитель, который бы поддерживал работу системного регистра или главной бухгалтерской книги системы. В связи с этим возникает вопрос: что выступает в качестве гаранта честной работы всех участники сети?

Сатоши предложил систему Proof of Work (Доказательство выполнения работы), которая дала возможность любому пользователю выдвигать блок на добавление в блокчейн. Чтобы выдвинуть блок, пользователи должны предоставить вычислительные мощности для решения задач, установленных протоколом.

Proof of Work – это проверенная схема достижения консенсуса среди пользователей, но далеко не единственная. Альтернативы, такие как Proof of Stake, продолжают тестироваться, однако они пока не достигли уровня реализации в их истинной форме (несмотря на то, что гибридные механизмы консенсуса существуют уже довольно долго).

Если майнер найдет решение задачи, построенный им блок расширит цепочку. В результате он получит вознаграждение в родной валюте блокчейна.

Задачи, которые майнеры должны решать, требуют постоянного хеширования данных для получения числа ниже определенного значения. Хеширование с помощью односторонней функции означает, что, имея выходные данные, угадать входные данные практически невозможно. И наоборот: при наличии входных данных проверить выходные данные уже очень легко. Таким образом, любой пользователь может проверить, что майнер создал «правильный» блок и отклонить невалидные блоки. В последнем случае майнер не получит вознаграждения и впустую потратит свои ресурсы.

В результате вырисовывается такая концепция игры, при которой попытка обмана обходится очень дорого, а честность поощряется. Ни у одного злоумышленника не хватит ресурсов, чтобы бесконечно атаковать сильную сеть. Таким образом, пользователи с ресурсами могут получать доход путем честного участия в процессе.

Минусы блокчейна:

К сожалению, криптовалюты могут быть безопасными и устойчивыми к цензуре лишь в том случае, если все ноды регулярно синхронизируют копию блокчейн-сети. И чем ниже требования к синхронной активности участников, тем проще процесс присоединения новых участников.

Таким образом, блокчейн, ориентированный на добавление маленьких блоков каждые десять минут, предпочтительнее, чем тот, который добавляет один огромный блок каждые пять минут. Вариант с большим блоком предполагает ноды с наличием высокопроизводительных компьютеров, чтобы поддерживать регулярную синхронизацию и подталкивать маломощные устройства к отключению. Подобный подход в итоге приводит к большей централизации, поскольку конкуренция в сети становится всё меньше.

Но с маленькими блоками мы не можем достичь большого количества транзакций в секунду (TPS). Кроме того, в загруженные периоды добавление блоков в блокчейн займет больше времени. Это неудобно, если вы хотите произвести быструю оплату, но ради децентрализации приходится идти на компромиссы.

Мы называем эту проблему дилеммой масштабируемости. Хорошо масштабируемая система может легко адаптироваться к повышенной нагрузке. Блокчейны же масштабируются плохо. Как мы уже объясняли, увеличение пропускной способности за счет блоков большего размера подрывает всю цель распределенной сети.

Чтобы увеличить TPS без ущерба децентрализации, применяется подход офчейн-масштабирования. Он включает широкий спектр решений (централизованных и децентрализованных), которые позволяют совершать транзакции без регистрации в блокчейне.

**Плюсы использования базы данных**

К сожалению, блокчейн имеет множество проблем, связанных с выскоим потреблением электроэнергии, если блоков очень много. Возможно, необходимо будет отказаться от него.

На замену может прийти обычное использование базы данных, но в нескольких экземплярах, их синхронизация.

**База данных** (БД) — это организованная структура, предназначенная для хранения, изменения и обработки взаимосвязанной информации, преимущественно больших объемов. **Базы данных** активно используются для динамических сайтов со значительными объемами **данных** — часто это интернет-магазины, порталы, корпоративные сайты.

**Реляционные** базы данных являются самыми распространёнными

**Реляционная база данных** — [база данных](https://ru.wikipedia.org/wiki/%D0%91%D0%B0%D0%B7%D0%B0_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85), основанная на [реляционной модели данных](https://ru.wikipedia.org/wiki/%D0%A0%D0%B5%D0%BB%D1%8F%D1%86%D0%B8%D0%BE%D0%BD%D0%BD%D0%B0%D1%8F_%D0%BC%D0%BE%D0%B4%D0%B5%D0%BB%D1%8C_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85). Понятие «реляционный» основано на [англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) [relation](https://ru.wiktionary.org/wiki/relation" \l "%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9" \o "wikt:relation) («[отношение](https://ru.wikipedia.org/wiki/%D0%9E%D1%82%D0%BD%D0%BE%D1%88%D0%B5%D0%BD%D0%B8%D0%B5_(%D1%80%D0%B5%D0%BB%D1%8F%D1%86%D0%B8%D0%BE%D0%BD%D0%BD%D0%B0%D1%8F_%D0%BC%D0%BE%D0%B4%D0%B5%D0%BB%D1%8C)), зависимость, связь»). Использование реляционных баз данных было предложено доктором [Коддом](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%B4%D0%B4,_%D0%AD%D0%B4%D0%B3%D0%B0%D1%80) из компании [IBM](https://ru.wikipedia.org/wiki/IBM) в [1970 году](https://ru.wikipedia.org/wiki/1970_%D0%B3%D0%BE%D0%B4). Для работы с реляционными БД применяют [реляционные СУБД](https://ru.wikipedia.org/wiki/%D0%A0%D0%B5%D0%BB%D1%8F%D1%86%D0%B8%D0%BE%D0%BD%D0%BD%D1%8B%D0%B5_%D0%A1%D0%A3%D0%91%D0%94).

Всё-таки, базы данных являются более привычным и изученным способом хранить данные, в отличие от блокчейна. Например, существует методика устранения избыточности, называемая **нормализацией**.

Целью нормализации реляционной базы данных является устранение недостатков структуры базы данных, приводящих к избыточности, которая, в свою очередь, потенциально приводит к различным аномалиям и нарушениям целостности данных.

Теоретики реляционных баз данных в процессе развития теории выявили и описали типичные примеры избыточности и способы их устранения.

Современные технологии предоставляют возможность не иметь собственные сервера, а хранить данные в “Облаке”.

**Облачное хранилище данных** ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *cloud storage*) — модель онлайн-хранилища, в котором данные хранятся на многочисленных распределённых в сети [серверах](https://ru.wikipedia.org/wiki/%D0%A1%D0%B5%D1%80%D0%B2%D0%B5%D1%80_(%D0%B0%D0%BF%D0%BF%D0%B0%D1%80%D0%B0%D1%82%D0%BD%D0%BE%D0%B5_%D0%BE%D0%B1%D0%B5%D1%81%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%B8%D0%B5)), предоставляемых в пользование клиентам, в основном, третьей стороной. В отличие от модели хранения данных на собственных выделенных серверах, приобретаемых или арендуемых специально для подобных целей, количество или какая-либо внутренняя структура серверов клиенту, в общем случае, не видна. Данные хранятся и обрабатываются в так называемом «[облаке](https://ru.wikipedia.org/wiki/%D0%9E%D0%B1%D0%BB%D0%B0%D1%87%D0%BD%D1%8B%D0%B5_%D0%B2%D1%8B%D1%87%D0%B8%D1%81%D0%BB%D0%B5%D0%BD%D0%B8%D1%8F)», которое представляет собой, с точки зрения клиента, один большой виртуальный сервер. Физически же такие серверы могут располагаться удалённо друг от друга географически.

Облачными хранилищами являются такие интернет-сервисы, как: [Dropbox](https://ru.wikipedia.org/wiki/Dropbox" \o "Dropbox), [OneDrive](https://ru.wikipedia.org/wiki/OneDrive" \o "OneDrive), [Google Drive](https://ru.wikipedia.org/wiki/Google_Drive) и другие.

**Преимущества облачного хранилища:**

* Возможность доступа к данным из любого компьютера, имеющего выход в интернет.
* Возможность организации совместной работы с данными.
* Высокая вероятность сохранения данных даже в случае аппаратных сбоев.
* Клиент платит только за то место в хранилище, которое фактически использует, но не за аренду сервера, все ресурсы которого он может и не использовать.
* Клиенту нет необходимости заниматься приобретением, поддержкой и обслуживанием собственной инфраструктуры по хранению данных, что, в конечном счёте, уменьшает общие издержки производства.
* Все процедуры по резервированию и сохранению целостности данных предоставляются провайдером «облачного» центра, который не вовлекает в этот процесс клиента.

**Проблемы облачного хранилища:**

В марте 2011 года аналитическая фирма [IDC](https://ru.wikipedia.org/wiki/International_Data_Corporation) провела исследование, которое показало, что многие компании в первую очередь связывают с «облачными» сервисами большие проблемы по части безопасности. А независимая исследовательская организация [Portio Research](https://ru.wikipedia.org/w/index.php?title=Portio_Research&action=edit&redlink=1" \o "Portio Research (страница отсутствует)) только подтвердила это, указав конкретные цифры: 68 % опрошенных руководителей европейских IT-компаний в целях безопасности отказываются использовать «облачные» технологии. Также руководитель компании [G Data Security Labs](https://ru.wikipedia.org/wiki/G-DATA) и эксперт по безопасности Ральф Бенцмюллер не рекомендовал использовать «облачные» сервисы в связи с тем, что данные могут стать достоянием общественности.

* Безопасность при хранении и пересылке данных является одним из основных вопросов при работе с «облаком», особенно в отношении конфиденциальных и приватных данных. Так, например, провайдер имеет возможность просматривать данные клиента (если они не защищены [паролем](https://ru.wikipedia.org/wiki/%D0%9F%D0%B0%D1%80%D0%BE%D0%BB%D1%8C)), которые также могут попасть в руки [хакеров](https://ru.wikipedia.org/wiki/%D0%A5%D0%B0%D0%BA%D0%B5%D1%80), сумевших взломать системы защиты провайдера.
* Надёжность, своевременность получения и доступность данных в «облаке» очень сильно зависит от многих промежуточных параметров, таких как: каналы передачи данных на пути от клиента к «облаку», надёжность [последней мили](https://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D1%81%D0%BB%D0%B5%D0%B4%D0%BD%D1%8F%D1%8F_%D0%BC%D0%B8%D0%BB%D1%8F), качество работы [интернет-провайдера](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%82%D0%B5%D1%80%D0%BD%D0%B5%D1%82-%D0%BF%D1%80%D0%BE%D0%B2%D0%B0%D0%B9%D0%B4%D0%B5%D1%80) клиента, доступность самого «облака» в данный момент времени. Если же сама компания, предоставляющая онлайновое хранилище, будет ликвидирована, клиент может потерять все свои данные.
* Общая производительность при работе с данными в «облаке» может быть ниже, чем при работе с локальными копиями данных.
* Абонентская плата за дополнительные возможности (увеличенный объём хранения данных, передача больших файлов и т. д.).

### **Известные инциденты**

19 июня 2011 года в течение четырёх часов любой желающий мог получить доступ к данным пользователей [Dropbox](https://ru.wikipedia.org/wiki/Dropbox" \o "Dropbox).

Некорректная настройка прав доступа к облачным хранилищам Amazon S3 несколько раз приводила к крупным утечкам данных, например, 200 миллионов записей предвыборной аналитики, о 3 миллионах фанатов WWE, 6 миллионах клиентов Verizon.

Обрушения линий связи и датацентров могут приводить к временной недоступности данных из облачных хранилищ или даже потере данных.

**Облачные шлюзы**

Облачные шлюзы — технология, которая может быть использована для более удобного предоставления «облака» клиенту. К примеру, с помощью соответствующего программного обеспечения, хранилище в «облаке» может быть предоставлено для клиента как локальный диск на компьютере. Таким образом, работа с данными в «облаке» для клиента становится абсолютно прозрачной. И при наличии хорошей, быстрой связи с «облаком» клиент может даже не замечать, что работает не с локальными данными у себя на компьютере, а с данными, хранящимися, возможно, за много километров от него.

В базах данных широко используется язык **SQL**

**SQL** (structured query language — «язык структурированных запросов») — [декларативный](https://ru.wikipedia.org/wiki/%D0%94%D0%B5%D0%BA%D0%BB%D0%B0%D1%80%D0%B0%D1%82%D0%B8%D0%B2%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5) [язык программирования](https://ru.wikipedia.org/wiki/%D0%AF%D0%B7%D1%8B%D0%BA_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F), применяемый для создания, модификации и управления данными в [реляционной базе данных](https://ru.wikipedia.org/wiki/%D0%A0%D0%B5%D0%BB%D1%8F%D1%86%D0%B8%D0%BE%D0%BD%D0%BD%D1%8B%D0%B5_%D0%B1%D0%B0%D0%B7%D1%8B_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85), управляемой соответствующей [системой управления базами данных](https://ru.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D1%83%D0%BF%D1%80%D0%B0%D0%B2%D0%BB%D0%B5%D0%BD%D0%B8%D1%8F_%D0%B1%D0%B0%D0%B7%D0%B0%D0%BC%D0%B8_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85).

Является, прежде всего, [информационно-логическим языком](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%86%D0%B8%D0%BE%D0%BD%D0%BD%D1%8B%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA), предназначенным для описания, изменения и извлечения данных, хранимых в [реляционных базах данных](https://ru.wikipedia.org/wiki/%D0%A0%D0%B5%D0%BB%D1%8F%D1%86%D0%B8%D0%BE%D0%BD%D0%BD%D1%8B%D0%B5_%D0%B1%D0%B0%D0%B7%D1%8B_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85). SQL считается [языком программирования](https://ru.wikipedia.org/wiki/%D0%AF%D0%B7%D1%8B%D0%BA_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F), в общем случае (без ряда современных расширений) не является [тьюринг-полным](https://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D0%BB%D0%BD%D0%BE%D1%82%D0%B0_%D0%BF%D0%BE_%D0%A2%D1%8C%D1%8E%D1%80%D0%B8%D0%BD%D0%B3%D1%83" \o "Полнота по Тьюрингу), но вместе с тем стандарт языка спецификацией [SQL/PSM](https://ru.wikipedia.org/wiki/SQL/PSM) предусматривает возможность его [процедурных расширений](https://ru.wikipedia.org/wiki/SQL#%D0%9F%D1%80%D0%BE%D1%86%D0%B5%D0%B4%D1%83%D1%80%D0%BD%D1%8B%D0%B5_%D1%80%D0%B0%D1%81%D1%88%D0%B8%D1%80%D0%B5%D0%BD%D0%B8%D1%8F).

Изначально SQL был основным способом работы пользователя с [базой данных](https://ru.wikipedia.org/wiki/%D0%91%D0%B0%D0%B7%D0%B0_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85) и позволял выполнять следующий набор операций:

* создание в базе данных новой таблицы;
* добавление в таблицу новых записей;
* изменение записей;
* удаление записей;
* выборка записей из одной или нескольких таблиц (в соответствии с заданным условием);
* изменение структур таблиц.

Со временем SQL усложнился — обогатился новыми конструкциями, обеспечил возможность описания и управления новыми хранимыми объектами (например, индексы, представления, триггеры и хранимые процедуры) — и стал приобретать черты, свойственные языкам программирования.

При всех своих изменениях SQL остаётся самым распространённым лингвистическим средством для взаимодействия прикладного программного обеспечения с базами данных. В то же время современные [СУБД](https://ru.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D1%83%D0%BF%D1%80%D0%B0%D0%B2%D0%BB%D0%B5%D0%BD%D0%B8%D1%8F_%D0%B1%D0%B0%D0%B7%D0%B0%D0%BC%D0%B8_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85), а также информационные системы, использующие СУБД, предоставляют пользователю развитые средства визуального построения запросов.

**SQL как стандарт**

Поскольку к началу 1980-х годов существовало несколько вариантов СУБД от разных производителей, причём каждый из них обладал собственной реализацией языка запросов, было принято решение разработать стандарт языка, который будет гарантировать переносимость ПО с одной СУБД на другую (при условии, что они будут поддерживать этот стандарт).

В 1983 году [Международная организация по стандартизации](https://ru.wikipedia.org/wiki/%D0%9C%D0%B5%D0%B6%D0%B4%D1%83%D0%BD%D0%B0%D1%80%D0%BE%D0%B4%D0%BD%D0%B0%D1%8F_%D0%BE%D1%80%D0%B3%D0%B0%D0%BD%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D1%8F_%D0%BF%D0%BE_%D1%81%D1%82%D0%B0%D0%BD%D0%B4%D0%B0%D1%80%D1%82%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D0%B8) (ISO) и [Американский национальный институт стандартов](https://ru.wikipedia.org/wiki/%D0%90%D0%BC%D0%B5%D1%80%D0%B8%D0%BA%D0%B0%D0%BD%D1%81%D0%BA%D0%B8%D0%B9_%D0%BD%D0%B0%D1%86%D0%B8%D0%BE%D0%BD%D0%B0%D0%BB%D1%8C%D0%BD%D1%8B%D0%B9_%D0%B8%D0%BD%D1%81%D1%82%D0%B8%D1%82%D1%83%D1%82_%D1%81%D1%82%D0%B0%D0%BD%D0%B4%D0%B0%D1%80%D1%82%D0%BE%D0%B2) (ANSI) приступили к разработке стандарта языка SQL. По прошествии множества консультаций и отклонения нескольких предварительных вариантов, в 1986 году ANSI представил свою первую версию стандарта, описанную в документе ANSI X3.135-1986 под названием «Database Language SQL» (Язык баз данных SQL). Неофициально этот стандарт SQL-86 получил название SQL1. Год спустя была завершена работа над версией стандарта ISO 9075-1987 под тем же названием. Разработка этого стандарта велась под патронажем Технического Комитета TC97 ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) Technical Committee TC97), областью деятельности которого являлись процессы вычисления и обработки информации ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) Computing and Information Processing). Именно его подразделение, именуемое как Подкомитет SC21 ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) Subcommittee SC21), курировало разработку стандарта, что стало залогом идентичности стандартов ISO и ANSI для SQL1 (SQL-86).

Стандарт SQL1 разделялся на два уровня. Первый уровень представлял собой подмножество второго уровня, описывавшего весь документ в целом. То есть, такая структура предусматривала, что не все спецификации стандарта SQL1 будут относиться к Уровню 1. Тем самым поставщик, заявлявший о поддержке данного стандарта, должен был заявлять об уровне, которому соответствует его реализация языка SQL. Это значительно облегчило принятие и поддержку стандарта, поскольку производители могли реализовывать его поддержку в два этапа.

Со временем к стандарту накопилось несколько замечаний и пожеланий, особенно с точки зрения обеспечения целостности и корректности данных, в результате чего в 1989 году данный стандарт был расширен, получив название SQL89. В частности, в него была добавлена концепция [первичного](https://ru.wikipedia.org/wiki/%D0%9F%D0%B5%D1%80%D0%B2%D0%B8%D1%87%D0%BD%D1%8B%D0%B9_%D0%BA%D0%BB%D1%8E%D1%87) и [внешнего ключей](https://ru.wikipedia.org/wiki/%D0%92%D0%BD%D0%B5%D1%88%D0%BD%D0%B8%D0%B9_%D0%BA%D0%BB%D1%8E%D1%87). ISO-версия документа получила название ISO 9075:1989 «Database Language SQL with Integrity Enhancements» (Язык баз данных SQL с добавлением контроля целостности). Параллельно была закончена и ANSI-версия.

Сразу после завершения работы над стандартом SQL1 в 1987 году была начата работа над новой версией стандарта, который должен был заменить стандарт SQL89, получив название SQL2, поскольку дата принятия документа на тот момент была неизвестна. Таким образом, фактически SQL89 и SQL2 разрабатывались параллельно. Новая версия стандарта была принята в 1992 году, заменив стандарт SQL89. Новый стандарт, озаглавленный как SQL92, представлял собой по сути расширение стандарта SQL1, включив в себя множество дополнений, имевшихся в предыдущих версиях инструкций.

Как и SQL1, SQL92 также был разделён на несколько уровней, однако, во-первых, число уровней было увеличено с двух до трёх, а во-вторых, они получили названия вместо порядковых цифр: начальный ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) entry), средний ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) intermediate), полный ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) full). Уровень «полный», как и Уровень 2 в SQL1 подразумевал весь стандарт целиком. Уровень «начальный» представлял собой подмножество уровня «средний», в свою очередь, представлявшего собой подмножество уровня «полный». Уровень «начальный» был сравним с Уровнем 2 стандарта SQL1, но спецификации этого уровня были несколько расширены. Таким образом, цепочка включений уровней стандартов выглядела примерно следующим образом: SQL1 Уровень 1 → SQL1 Уровень 2 → SQL92 «Начальный» → SQL92 «Средний» → SQL92 «Полный».

После принятия стандарта SQL92 к нему были добавлены ещё несколько документов, расширявших функциональность языка. Так, в 1995 году был принят стандарт [SQL/CLI](https://ru.wikipedia.org/wiki/SQL/CLI) (Call Level Interface, интерфейс уровня вызовов), впоследствии переименованный в CLI95. На следующий год был принят стандарт [SQL/PSM](https://ru.wikipedia.org/wiki/SQL/PSM) (Persistent Stored Modules, постоянно хранимые модули), получивший название PSM-96.

На данный момент, каждый может создать собственную версию языка SQL. Существуют версии от компаний Microsoft, Oracle и другие.