**Виды баз данных, их сравнение и выбор наиболее подходящего решения для проекта ZLP TECH**

Ещё до появления информационных технологий у людей возникала необходимость в упорядоченном хранении тех или иных данных. Для удобства их разделяли по определённому признаку, объединяли в группы, создавали иерархическое представление и применяли множество других способов.

База данных с точки зрения информатики - это хранение информации в упорядоченном виде, следуя определённой, заранее установленной разработчиком, системе.

Существует несколько видов баз данных, различающихся по структуре:

* иерархические;
* сетевые;
* реляционные.

Под *иерархической* понимается такая база данных, в которой хранение данных и их структурирование осуществляется по принципу разделения элементов на родительские и дочерние. Преимуществом таких баз является лёгкость в чтении запрашиваемой информации и её быстрое предоставление пользователю.

Компьютер способен быстро ориентироваться в ней. Иерархический принцип взят за основу в структурировании файлов и папок в операционной системе Windows, а реестр хранит информацию о параметрах работы тех или иных приложений в структурированном иерархическим способом виде.

Все интернет-ресурсы также построены по иерархическому принципу, так как при его использовании ориентироваться в рамках сайта очень легко.

Под *сетевой* базой данных понимается модифицированная иерархическая. Её особенность заключается в том, что элементы могут быть связаны друг с другом в нарушение иерархии. То есть дочерний элемент одновременно может иметь несколько предков.

Под *реляционным* типом баз данных понимается их представление в рамках двумерной таблицы. Она имеет несколько столбцов, в которых устанавливаются такие параметры, как, например, тип вводимых данных (текст, число, дата и др.).

Таблица здесь является способом хранения введённых в неё данных и способна реагировать на любые обращения со стороны СУБД. Главная проблема в работе с реляционными базами данных состоит в их правильном проектировании.

Во время проектирования базы данных следует учесть следующие два фактора:

* база данных должна быть компактной и не содержать избыточных компонентов;
* обработка базы данных должны происходить просто.

В крупных проектах задействовано множество таблиц, которых может быть более сотни. При этом обойтись без них невозможно, если человек имеет дело с важным и сложным проектом.

Система управления базами данных представляет собой встраивыемый модуль или полноценную программу, которая способна работать с данными и вносить изменения в базы.

Существует две модели СУБД - реляционная и бессхемная. Бессхемные СУБД, основанные на принципах неструктурированного подхода, избавляют программиста от проблем реляционной модели, в число которых входит низкая производительность и трудное масштабирование данных в горизонтальном формате.

Неструктурированные базы данных (NoSQL) создают структуру по ходу и убирают необходимость в создании жёстко определённых связей между данными. Здесь можно экспериментировать с разными способами доступа к тем или иным видам данных.

К реляционным базам данных относятся:

* SQLite;
* MySQL;
* PostgreSQL.

Из них наиболее распространённой является база данных MySQL. Принцип работы таких систем заключается в слежении за строгой структурой данных, которая представлена в виде комплекса таблиц. В свою очередь, внутри таблицы есть ячейки и поля, которыми также управляет MySQL.

По принципу NoSQL работает база данных MongoDB. Они хранят все данные как единое целое в одной базе. При этом данные могут быть и одиночным объектом, но в то же время любой запрос не останется без ответа.

Каждая NoSQL имеет собственную систему запросов, что требует дополнительного изучения данной системы.

Сравнение SQL и NoSQL:

1. Если SQL-системы основаны исключительно на строгом представлении данных, то NoSQL-системы предоставляют свободу и способны работать с любым типом данных.
2. SQL-системы стандартизированы, за счёт чего запросы формируются с использованием языка SQL. В то же время NoSQL-системы базируются на специфической для каждой из них технологии, что является недостатком.
3. Масштабируемость. Обе СУБД способны обеспечить вертикальное масштабирование, то есть увеличить объём системных ресурсов на обработку данных. При этом NoSQL, будучи более новой разновидностью баз данных, позволяет применять простые методы горизонтального масштабирования.
4. В плане надёжности SQL обладает уверенным лидерством.
5. У SQL-баз есть качественная техническая поддержка за счёт их продолжительной истории, в то время как NoSQL-системы весьма молоды и и решить какую-либо проблему сложнее.
6. Хранение данных и доступ к их структурам в рамках реляционных систем лучше всего происходит в SQL-системах.

Таким образом, хоть NoSQL и является стремительно развивающейся разновидностью систем управления базами данных, однако на данном этапе рекомендуется остановить свой выбор на SQL [1].

Теперь рассмотрим виды SQL-систем. Выше были упомянуты SQLite, MySQL и PostgreSQL. Кратко их можно охарактеризовать так:

* SQLite - очень мощная встраиваемая система управления;
* MySQL - самая популярная и распространённая СУБД;
* PostgreSQL - наиболее продвинутая СУБД.

**SQLite** — легко встраиваемая в приложения база данных. Так как это система базируется на файлах, то она предоставляет довольно широкий набор инструментов для работы с ней, по сравнению с сетевыми СУБД. При работе с этой СУБД обращения происходят напрямую к файлам (в эти файлах хранятся данные), вместо портов и сокетов в сетевых СУБД. Именно поэтому SQLite очень быстрая, а также мощная благодаря технологиям обслуживающих библиотек.

*Когда стоит использовать SQLite:*

* Встроенные приложения: все портируемые не предназначенные для масштабирования приложения — например, локальные однопользовательские приложения, мобильные приложения или игры.
* Система доступа к дисковой памяти: в большинстве случаев приложения, часто производящие прямые операции чтения/записи на диск, можно перевести на SQLite для повышения производительности.
* Тестирование: отлично подойдёт для большинства приложений, частью функционала которых является тестирование бизнес-логики.

*Когда не стоит использовать SQLite:*

* Многопользовательские приложения: если вы работаете над приложением, доступом к БД в котором будут одновременно пользоваться несколько человек, лучше выбрать полнофункциональную РСУБД — например, MySQL.
* Приложения, записывающие большие объёмы данных: одним из ограничений SQLite являются операции записи. Эта РСУБД допускает единовременное исполнение лишь одной операции записи.

**MySQL** - это самая распространенная полноценная серверная СУБД. MySQL очень функциональная, свободно распространяемая СУБД, которая успешно работает с различными сайтами и веб приложениями. Обучиться использованию этой СУБД довольно просто, так как на просторах интернета существует большое количество информации. Благодаря популярности этой СУБД существует огромное количество различных плагинов и расширений, облегчающих работу с системой.

Несмотря на то, что в ней не реализован весь SQL функционал, MySQL предлагает довольно много инструментов для разработки приложений. Так как это серверная СУБД, приложения для доступа к данным, в отличие от SQLite, работают со службами MySQL.

*Когда стоит использовать MySQL:*

* Распределённые операции: когда нужен функционал бо́льший, чем может предоставить SQLite, стоит использовать MySQL.
* Высокая безопасность: функции безопасности MySQL предоставляют надёжную защиту доступа и использования данных.
* Веб-сайты и приложения: большая часть веб-ресурсов вполне может работать с MySQL, несмотря на ограничения. Этот инструмент весьма гибок и прост в обращении, что только на руку в длительной перспективе.
* Кастомные решения: если вы работаете над очень специфичным продуктом, MySQL подстроится под ваши потребности благодаря широкому спектру настроек и режимов работы.

*Когда не стоит использовать MySQL:*

* SQL-совместимость: поскольку MySQL не пытается полностью реализовать стандарты SQL, она не является полностью совместимой с SQL. Из-за этого могут возникнуть проблемы при интеграции с другими РСУБД.
* Конкурентность: хотя MySQL неплохо справляется с операциями чтения, одновременные операции чтения-записи могут вызвать проблемы.
* Недостаток функций: в зависимости от выбора движка MySQL может недоставать некоторых функций.

**PostgreSQL** является самым профессиональным из всех трех рассмотренных нами СУБД. Она свободно распространяемая и максимально соответствует стандартам SQL. PostgreSQL или Postgres стараются полностью применять ANSI/ISO SQL стандарты своевременно с выходом новых версий.

От других СУБД PostgreSQL отличается поддержкой востребованного объектно-ориентированного и/или реляционного подхода к базам данных. Например, полная поддержка надежных транзакций, т.е. атомарность, последовательность, изоляционность, прочность (Atomicity, Consistency, Isolation, Durability - ACID). Благодаря мощным технологиям Postgre очень производительна. Параллельность достигнута не за счет блокировки операций чтения, а благодаря реализации управления многовариантным параллелизмом (MVCC), что также обеспечивает соответствие ACID. PostgreSQL очень легко расширять своими процедурами, которые называются хранимые процедуры. Эти функции упрощают использование постоянно повторяемых операций [2].

*Когда стоит использовать PostgreSQL:*

* Целостность данных: если приоритет стоит на надёжность и целостность данных, PostgreSQL — лучший выбор.
* Сложные процедуры: если БД должна выполнять сложные процедуры, стоит выбрать PostgreSQL в силу её расширяемости.
* Интеграция: если в будущем предстоит перемещать всю базу на другое решение, меньше всего проблем возникнет с PostgreSQL.

*Когда не стоит использовать PostgreSQL:*

* Скорость: если всё, что нужно — это быстрые операции чтения, не стоит использовать PostgreSQL.
* Простые ситуации: если не требуется повышенная надёжность, поддержка ACID и так далее, использовать PostgreSQL ни к чему [3].

В итоге, рассмотрев все варианты, мы решили использовать для нашего проекта MySQL. В этой системе нас привлекло обилие справочного материала, простота в обращении, совместимость со многими веб-ресурсами и приложениями, большое количество инструментов для разработки приложений и приемлемая скорость работы.