



МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение
высшего образования

«МИРЭА – Российский технологический университет»

РТУ МИРЭА

ЛЕКЦИОННЫЕ МАТЕРИАЛЫ

Технологии хранения в системах кибербезопасности

(наименование дисциплины (модуля) в соответствии с учебным планом)

Уровень

бакалавриат

(бакалавриат, магистратура, специалитет)

Форма обучения

очная

(очная, очно-заочная, заочная)

Направление(-я)
подготовки

10.05.04 Информационно-аналитические системы безопасности

(код(-ы) и наименование(-я))

Институт

Кибербезопасности и цифровых технологий (ИКБ)

(полное и краткое наименование)

Кафедра

КБ-2 «Прикладные информационные технологии»

(полное и краткое наименование кафедры, реализующей дисциплину (модуль))

Лектор

к.т.н., Селин Андрей Александрович

(сокращенно – ученая степень, ученое звание; полностью – ФИО)

Используются в данной редакции с учебного года

2024/2025

(учебный год цифрами)

Проверено и согласовано «___» _____ 2024 г.

А.А. Бакаев

*(подпись директора Института/Филиала
с расшифровкой)*

Москва 2024 г.



Технологии хранения в системах кибербезопасности

2024 год



Лекция 12.

БД временных рядов, графовые БД.

Учебные вопросы лекции:

1. Time Series Database (TSDB)
2. Распределенный режим
3. Столбцовые (колоночные) БД

Time Series Database (TSDB)

БД временных рядов (TSDB, Time Series DataBase) – это БД, оптимизированная для временных рядов – данных с отметками времени, которые представляют собой измерения или события, например, системные или сетевые метрики, показатели IoT-устройств, колебания валюты и пр. Будучи изначально созданной для данных с такой природой, TSDB оптимизирована для именно таких рабочих нагрузок: агрегации и сканирования большого диапазона множества записей. С одной стороны, модель данных в TSDB похожа на ключ значение, когда ключом выступает время, а значением – соответствующее ему измерение отслеживаемых данных. Однако, временные ряды постоянно растут. И, чтобы работать с этим огромным количеством данных, необходимо применять методы статистического анализа к различным диапазонам группировки измерений (по часам, минутам, секундам, микросекундам и т.д.).

Рост интереса к TSDB обусловлен развитием облачных технологий и активным внедрением интернета вещей в различные области жизни. Информационные системы, вычислительные кластера, беспилотные машины (автомобили, самолеты и пр.), датчики и IoT-устройства также постоянно генерируют события, которые надо наблюдать и анализировать.

Наиболее популярные СУБД временных рядов: InfluxDB, Kdb+, Prometheus, Graphite, TimescaleDB, DolphinDB, RRDTool, Apache Druid, TDengine, QuestDB, OpenTSDB, GridDB, Fauna, VictoriaMetrics, Amazon Timestream.

Time Series Database (TSDB)

Реляционные базы данных ([MySQL](#) или [SQL Server](#)) или с базы данных [NoSQL](#) ([MongoDB](#) или [DynamoDB](#)) основаны на том, что у них есть таблицы. Эти таблицы содержат столбцы и строки, каждая из которых определяет запись в вашей таблице. Часто эти таблицы специально предназначены для определённой цели. Одна может быть предназначена для хранения пользователей, другая — для фотографий или видео. Такие системы эффективны, масштабируемы и используются множеством гигантских компаний с миллионами запросов на своих серверах.

Базы данных временных рядов работают иначе. Данные по-прежнему хранятся в «коллекциях», но эти коллекции имеют общий знаменатель: они объединены со временем. Это означает, что для каждой точки, которую вы можете сохранить, у вас есть связанная с ней временная метка.

Классическая реляционная база данных				Базы данных временных рядов		
Name ▲	Age ▲	Nickname	Employee ▼	Sensor Temperature ▲	Time	
Giacomo Guizzoni Founder & CEO	40	Pedi	<input type="radio"/>	39.5	12/04/19 @ 14:12	
Marco Botton Tuttofare	38		<input checked="" type="checkbox"/>	41.2	12/04/19 @ 14:13	
Mariah MacLachlan Better Half	41	Parata	<input type="checkbox"/>	12.4	14/04/19 @ 12:15	
Valerie Liberty Head Chef	46	Vai	<input checked="" type="checkbox"/>	18.5	16/04/19 @ 10:05	

Данные многомерны

Данные агрегируются по времени

Time Series Database (TSDB)

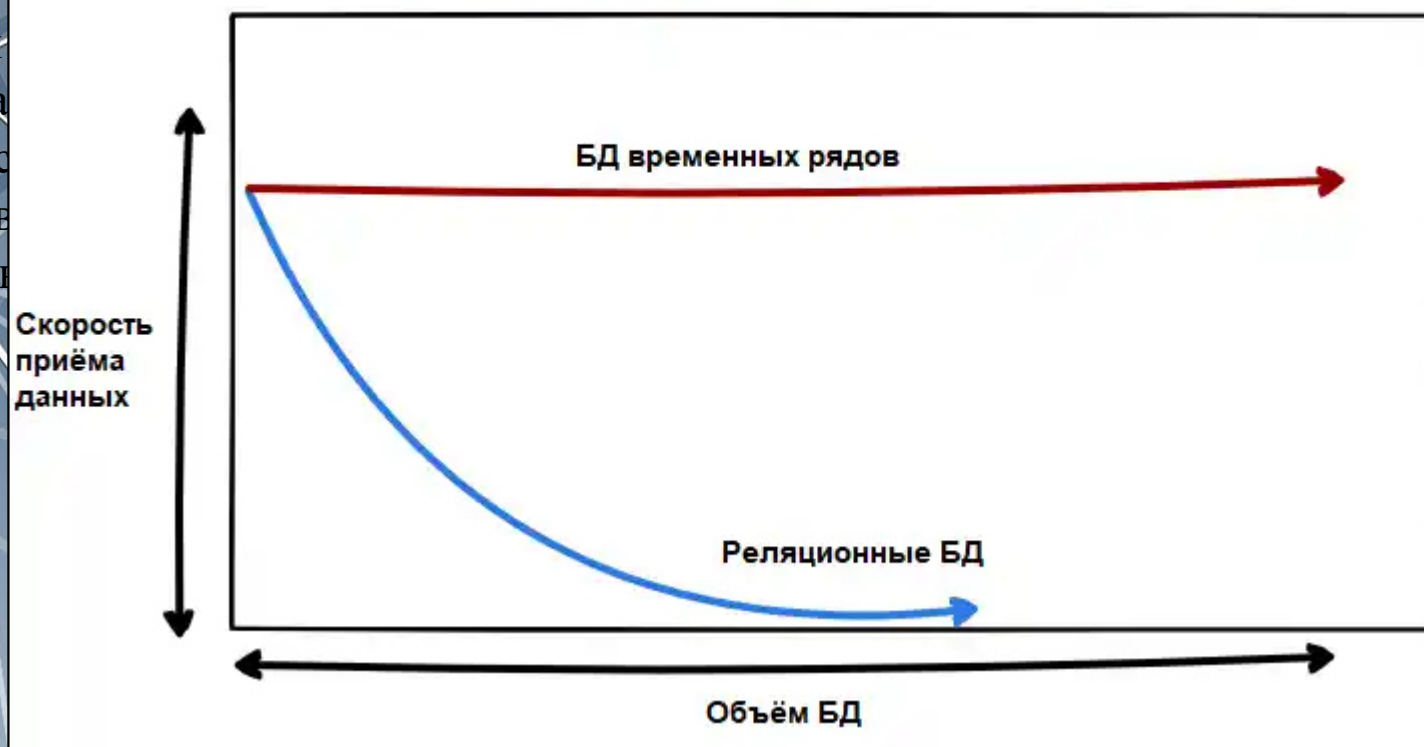
Системы баз данных временных рядов построены так, чтобы быстро и эффективно принимать данные. Реляционные базы данных тоже имеют большую скорость загрузки данных (от 20 000 до 100 000 строк в секунду). Тем не менее, приём не постоянен во времени. У реляционных баз данных есть один ключевой аспект, который делает их медленными при росте данных – индексы.

При добавлении новых записей в реляционную БД и при наличии в таблице индексов СУБД будет многократно переиндексировать данные для быстрого и

эффект
снижае
чтении
База
Такие с
следств
стабил

нем
при
ных.
Как
очно

Разница между реляционными БД и БД временных рядов



Time Series Database (TSDB)

Системы баз данных временных рядов построены так, чтобы быстро и эффективно принимать данные. Реляционные базы данных тоже имеют большую скорость загрузки данных (от 20 000 до 100 000 строк в секунду). Тем не менее, приём не постоянен во времени. У реляционных баз данных есть один ключевой аспект, который делает их медленными при росте данных – индексы.

При добавлении новых записей в реляционную БД и при наличии в таблице индексов СУБД будет многократно переиндексировать данные для быстрого и эффективного доступа к ним. Как следствие, производительность со временем снижается. При этом увеличивается нагрузка, что приводит к трудностям при чтении данных.

База данных временных рядов оптимизирована для быстрого приёма данных. Такие системы используют индексацию данных, объединённых со временем. Как следствие, скорость загрузки не уменьшается со временем и остается достаточно стабильной (от 50 до 100 тыс. строк в секунду на одном узле).

Временные ряды не ограничиваются метрическими показателями базы данных. Метриками может быть что угодно – изменение потока людей, входящих в торговый центр, с течением времени, изменение трафика в городе, использование общественного транспорта в течение дня, течение воды в реке или ручье, количество энергии, вырабатываемое водной установкой – все это и все остальное, что можно измерить во времени, является примером временных рядов. Такие данные можно запросить, построить, проанализировать, чтобы найти корреляционную зависимость между различными метриками.

Структура данных в базе данных временных рядов

Самая важная составляющая данных в базе данных временных рядов – это время. Существует два основных способа хранения данных. Первый способ чем-то похож на хранилище «ключ-значение» и выглядит так:

<i>Метка времени</i>	<i>Метрика 1</i>
2019-03-28 00:00:01	2356
2019-03-28 00:00:02	6874
2019-03-28 00:00:03	3245
2019-03-28 00:00:04	2340

Для каждой метки времени имеется некоторое значение метрики.

Структура данных в базе данных временных рядов


Второй способ подразумевает хранения большего числа показателей. Вместо того, чтобы хранить каждую метрику в отдельной таблице или коллекции, их можно хранить вместе.

<i>Метка времени</i>	<i>Метрика 1</i>	<i>Метрика 2</i>	<i>Метрика 3</i>	<i>Метрика 4</i>	<i>Метрика 5</i>
2019-03-28 00:00:01	765	873	124	98	0
2019-03-28 00:00:02	5876	765	872	7864	634
2019-03-28 00:00:03	234	7679	98	65	34
2019-03-28 00:00:04	345	3	598	0	7345

Такая структура данных, когда все метрики связаны, позволяет более эффективно запрашивать данные. Вместо того, чтобы читать несколько таблиц и объединять их для получения всех метрик, достаточно прочитать лишь одну единственную таблицу, чтобы подготовить данные к обработке и представлению.

Time Series Database (TSDB)

Примеры баз данных временных рядов. Существует множество баз данных временных рядов. Несколько примеров баз данных временных рядов:



**InfluxDB;
Prometheus;
TimescaleDB;
Graphite;
QuestDB;
AWS Timestream;
OpenTSDB.**

InfluxDB

InfluxDB

InfluxDB contains everything you need in a time series data platform in a single binary.



Особенности:

- Высокая производительность для данных временных рядов с высоким уровнем приема и запросов в реальном времени
- InfluxQL для взаимодействия с данными, которые схож с языком запросов SQL.
- Основной компонент стека TICK (Telegraf, InfluxDB, Chronograf и Kapacitor)
- Поддержка плагинов для таких протоколов, как collectd, Graphite, OpenTSDB для приема данных
- Может обрабатывать миллионы точек данных всего за 1 секунду
- Политики хранения для автоматического удаления устаревших данных

Prometheus

Prometheus - это решение для мониторинга с открытым исходным кодом, используемое для анализа данных метрик и отправки необходимых предупреждений. Он имеет локальную базу данных временных рядов на диске, которая хранит данные в пользовательском формате на диске.

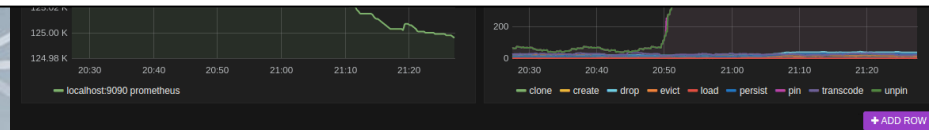
Модель данных Prometheus многомерна на основе временных рядов и сохраняет все данные в виде потоков значений с временной меткой. Это очень полезно при работе с полностью числовым временным рядом. Сбор данных о микросервисах и их запрос - одна из сильных сторон Prometheus.

Prometheus плотно интегрируется с **Grafana** для визуализации.

Особенности

- Имеет многомерную модель, в которой использовались пары "имя метрики" и "ключ-значение" (метки)
- PromQL используется для запроса данных временных рядов для создания таблиц, оповещений и графиков Adhoc
- Использует режим HTTP pull для сбора данных временных рядов
- Использует промежуточный шлюз для передачи временных рядов

У Prometheus есть сотни экспортеров для экспорта данных из Windows, Linux, Java, базы данных, API, веб-сайта, серверного оборудования, PHP, обмена сообщениями и т.д.



TimescaleDB

TimescaleDB - реляционная база данных с открытым исходным кодом, которая делает SQL масштабируемым для данных временных рядов. Эта база данных построена на PostgreSQL.



PostgreSQL for time-series

TimescaleDB is the leading open-source relational database for time-series data. Fully managed or self-hosted.

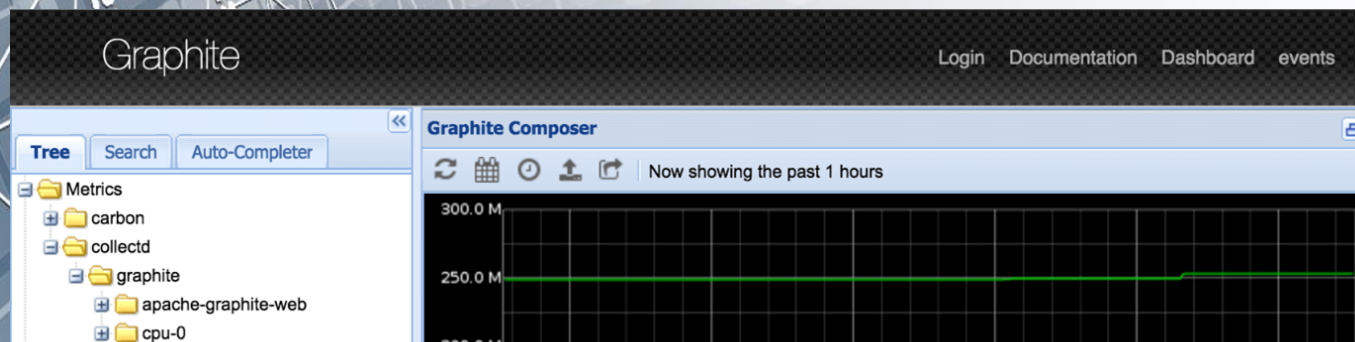
Особенности

- Выполнение запросов 10-100X быстрее, чем PostgreSQL, [MongoDB](#)
- Возможность горизонтального масштабирования до петабайт и записи миллионов точек данных в секунду
- Очень похож на PostgreSQL, что облегчает работу с ним разработчиков и администраторов.
- Сочетание функций реляционных баз данных и баз данных временных рядов для создания мощных приложений.
- Встроенные алгоритмы и функции производительности для защиты от больших затрат.

Graphite

Graphite - это универсальное решение для хранения и эффективной визуализации данных в реальном времени. Выполняет две функции: хранит данные временных рядов и визуализирует графики по требованию.

БД имеет три компонента - Carbon, Whisper и Graphite-Web. Carbon получает данные временных рядов, агрегирует их и сохраняет на диске. Whisper - это хранилище базы данных временных рядов, в котором хранятся данные. Graphite-Web - это интерфейс для создания панелей мониторинга и визуализации данных.



Особенности Graphite:

- Формат метрик, в котором передаются данные, прост
- Комплексный API для визуализации данных и создания диаграмм, панелей мониторинга, графиков
- Предоставляет богатый набор статистических библиотек и функций преобразования
- Связывает несколько функций визуализации для создания целевого запроса

QuestDB

QuestDB - это реляционная база данных, ориентированная на столбцы, которая может выполнять анализ данных временных рядов в реальном времени. Работает с SQL и некоторыми расширениями для создания реляционной модели для данных временных рядов. QuestDB был создан с нуля и не имеет зависимостей, повышающих его производительность.

QuestDB поддерживает реляционные соединения и соединения временных рядов, что помогает сопоставлять данные. Самый простой способ начать работу с QuestDB - развернуть его внутри контейнера Docker.

Функции QuestDB:

- Интерактивная консоль для импорта данных с помощью перетаскивания и запроса
- Поддерживается работа как на облачных технологиях (AWS, Azure, GCP), так и локально.
- Поддерживает такие корпоративные возможности, как работа с Active Directory, обеспечение высокой доступности, корпоративная безопасность, кластеризация
- Предоставляет информацию в режиме реального времени с использованием оперативной и прогнозируемой аналитики

AWS Timestream

AWS Timestream - это служба базы данных временных рядов без сервера, которая является быстрой и масштабируемой. Он используется главным образом для приложений Интернета вещей, чтобы хранить триллионы событий в день и в 1000 раз быстрее при 1/10 стоимости реляционных баз данных.

С помощью специализированного механизма запросов можно одновременно запрашивать последние данные и архивные сохраненные данные. Она предоставляет множество встроенных функций для анализа данных временных рядов для поиска полезной информации.

Функции Amazon Timestream:

- Нет серверов для управления или экземпляров для выделения; все обрабатывается автоматически.
- Экономичный, платите только за то, что вы принимаете, храните и запрашиваете.
- Способен ежедневно принимать триллионы событий без снижения производительности
- Встроенная аналитика со стандартными функциями SQL, интерполяции и сглаживания для определения тенденций, шаблонов и аномалий
- Все данные шифруются с помощью системы управления ключами AWS (KMS) с ключами управления клиента (CMK)

```
8 AND cell = 'ap-northeast-1-cell-5'
9 AND silo = 'ap-northeast-1-cell-5-silo-2'
10 AND availability_zone = 'ap-northeast-1-3'
11 AND microservice_name = 'zeus'
12 GROUP BY region,
13 cell,
```


OpenTSDB

OpenTSDB - масштабируемая БД временных рядов, написанная поверх Hbase, способная хранить триллионы точек данных при миллионах операций записи в секунду. Данные в OpenTSDB можно хранить вечно с его исходной меткой времени и точным значением, чтобы не потерять данные.



Функции OpenTSDB:

- Может агрегировать, фильтровать, понижать метрики на огромной скорости
- Хранение и запись данных с точностью до миллисекунды
- Работает на Hadoop и HBase и легко масштабируется, добавляя узлы в кластер
- Использование графического интерфейса для создания графиков

Специфичные концепции баз данных временных рядов

Кроме высокой скорости приёма, базы данных временных рядов вводят специфичные для этих технологий концепции.

Одной из них является организация хранения данных. В традиционной РБД данные хранятся до тех пор, пока вы не решите их удалить. Учитывая сценарии использования БД временных рядов, вы можете не хранить ваши данные слишком долго: это или слишком дорого, или данные со временем теряют актуальность.

Системы вроде InfluxDB могут позаботиться об удалении данных через определённое время, используя концепцию, называемую политикой хранения. Вы также можете выполнять непрерывные запросы к оперативным данным для выполнения определённых операций. В реляционной БД можно найти эквивалентные операции (например «задания» в SQL), которые могут выполняться по заданному расписанию.

Graph databases

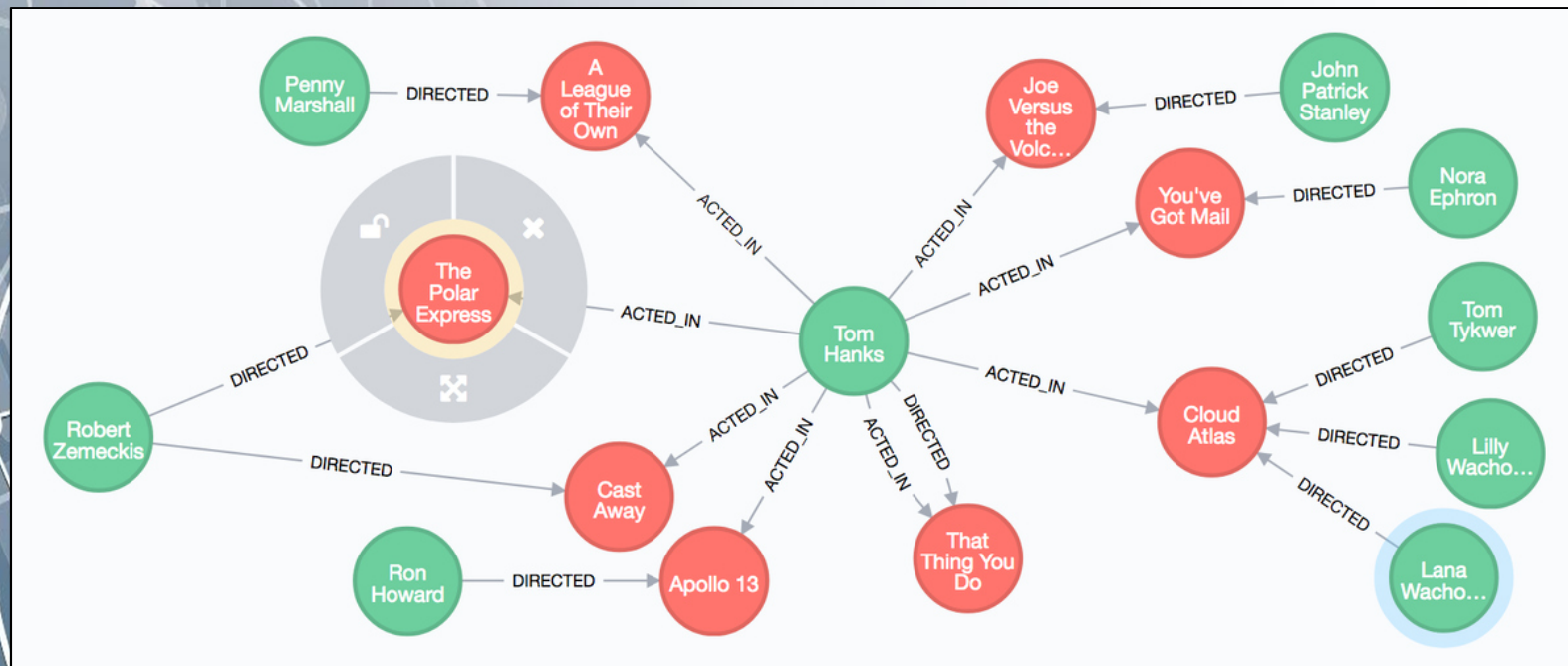
Графовые БД (Graph databases) – это NoSQL-системы, которые определяют корреляции между сложно взаимосвязанными сущностями. Такая структура позволяет обойти ограничения реляционных БД и уделяет больше внимания отношениям между данными.

Графовая БД позволяет аккуратно определять взаимосвязи и дает ответы на сложные вопросы о том, как точки данных соотносятся друг с другом.

Графовая база данных – это нереляционный тип баз данных, основанный на топографической структуре сети. Идея этой БД восходит к математической теории графов. Графы представляют наборы данных в виде узлов, ребер и свойств.

- Узлы, или точки (nodes) – это экземпляры или сущности данных; ими является любой объект, который вы планируете отслеживать. Например, люди, заказчики, подразделения и т.д.
- Ребра, или линии (edges) – это важнейшие концепции в графовых БД. Они отображают взаимосвязь между узлами. Эти связи имеют направление и могут быть одно- или двунаправленными.
- Свойства (properties) содержат описательную информацию, связанную с узлами. В некоторых случаях свойства бывают и у ребер.

Graph databases



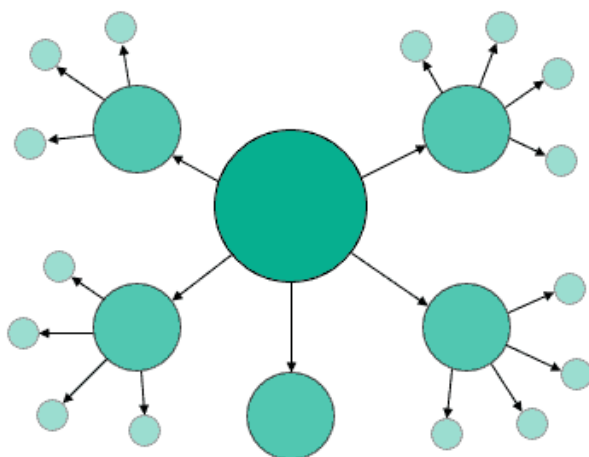
Узлы с пояснительными свойствами создают взаимосвязи, представленные через ребра.

Графовые БД предлагают концептуальное представление данных, тесно связанных с реальным миром. Моделировать сложные связи гораздо проще, поскольку отношениям между точками данных уделяется такое же внимание, как и самим данным.

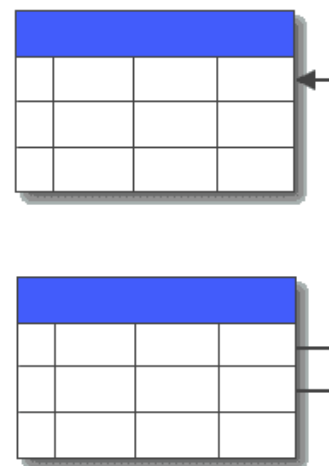
Graph databases

Сравнение графовых и реляционных баз данных

Graph



Relational



Тип	Графовые БД	Реляционные БД
Формат	Узлы и ребра со свойствами	Таблицы со строками и столбцами
Связи	Представлены в виде ребер между узлами	Создаются с помощью внешних ключей между таблицами
Гибкость	Гибкие	Жестко заданные
Сложные запросы	Быстрые и отзывчивые	Необходимы сложные соединения
Варианты использования	Системы с взаимосвязанными зависимостями	Системы с транзакциями и более простыми отношениями

Graph databases

Как работают графовые базы данных?

Графовые базы данных одинаково относятся к данным и взаимосвязям между ними. Связанные узлы физически связываются, и эта связь рассматривается как часть данных.

При таком моделировании данных вы можете запрашивать взаимосвязи также, как и сами данные. Вместо вычисления и запросов на подключение, графовые БД считывают взаимосвязи напрямую из хранилища.

По гибкости, производительности и адаптивности графовые БД близки к другим нереляционным моделям данных. В них, как и в других нереляционных БД, отсутствуют схемы, что делает данную модель гибкой и легко изменяемой.

Примеры использования графовых БД

Есть много примеров, когда графовые БД превосходят все прочие методы моделирования данных. Среди таких примеров можно выделить:

- Рекомендательные сервисы в режиме реального времени. Динамичные рекомендации по продуктам и электронным товарам улучшают пользовательский опыт и максимизируют прибыль. Из известных компаний можно упомянуть Netflix, eBay и Walmart.
- Управление основными данными. Привязка всех данных к одной общей точке обеспечивает постоянство и точность данных. Управление основными данными крайне важно для крупномасштабных компаний мирового уровня.
- GDPR и соблюдение нормативных требований. С графами гораздо проще управлять безопасностью и отслеживать перемещение данных. Базы данных снижают вероятность утечки информации и обеспечивают большую согласованность при удалении данных, чем повышается общее доверие к конфиденциальной информации.
- Управление цифровыми ресурсами. Объем цифрового контента огромен и постоянно растет. Графовые БД предлагают масштабируемую и простую модель данных, позволяющую отслеживать цифровые ресурсы: документы, расчеты, контракты и т.д.

Примеры использования графовых БД

- Контекстно-зависимые сервисы. Графы помогают в предоставлении сервисов, приближенных к актуальным характеристиками мира. Будь то предупреждения о стихийных бедствиях, информация о пробках или рекомендации по товарам для конкретного местоположения, – графовые базы данных предлагают логическое решение для реальных обстоятельств.
- Выявление мошенничества. Поиск подозрительных закономерностей и раскрытие мошеннических платежных схем выполняется в режиме реального времени. Выявление и изоляция частей графа позволяет быстрее обнаружить мошенническое поведение.
- Семантический поиск. Обработка естественного языка бывает неоднозначной. Семантический поиск помогает определить значение ключевых слов и выдает более подходящие варианты, которые, в свою очередь проще отобразить с помощью графовых БД.
- Сетевое управление. Сети – это не что иное, как связанные графы. Графовые БД снижают время, необходимое для оповещения сетевого администратора о проблемах в сети.
- Маршрутизация. Информация передается по сети за счет поиска оптимальных маршрутов, и это делает графовые БД идеальным вариантом для маршрутизации.

Популярные графовые БД

JanusGraph – это распределенная, масштабируемая система графовых БД с открытым кодом и широким набором возможностей по интеграции и аналитике больших данных. Ниже приведен перечень основных функций JanusGraph:

- Поддержка ACID-транзакций с возможностью одновременного обслуживания тысяч пользователей
- Несколько вариантов хранения графических данных, включая Cassandra и HBase
- Встроенный сложный поиск, а также дополнительная (опциональная) поддержка Elasticsearch
- Полная интеграция Apache Spark для расширенной аналитики данных
- JanusGraph использует полный по Тьюрингу язык запросов для обхода графов

Neo4j (Network Exploration and Optimization 4 Java, что переводится как «исследование сети и оптимизация для Java») – это графовая база данных, написанная на Java с нативным хранением и обработкой графов. Основные возможности:

- Масштабируемость БД за счет разделения данных на части – сегменты
- Высокая доступность благодаря непрерывному резервному копированию и последовательным обновлениям
- Высокий уровень безопасности: несколько экземпляров баз данных можно разделить, оставив их на одном выделенном сервере
- Neo4j использует Cypher – язык запросов для графовых БД, который очень удобен для программирования

Популярные графовые БД

DGraph (Distributed graph, что переводится как «распределенный граф») – это распределенная система графовых БД с открытым исходным кодом и хорошей масштабируемостью. Вот несколько интересных возможностей DGraph:

Горизонтальная масштабируемость для работы в реальной среде с ACID-транзакциями

DGraph – это свободно распространяемая система с поддержкой множества открытых стандартов

Язык запросов – GraphQL, который был разработан для API

DataStax Enterprise Graph – это распределенная графовая БД на базе Cassandra. Она оптимизирована под предприятия. Несколько функций:

DataStax обеспечивает постоянную доступность для корпоративных нужд

База данных легко интегрируется с автономной платформой Apache Spark

Полная интеграция аналитики и поиска в реальном времени

Масштабируемость за счет наличия нескольких центров обработки данных

Поддержка Gremlin и CQL для запросов

Graph databases

Плюсы и минусы графовых баз данных

В каждом типе баз данных есть свои плюсы и минусы. Именно поэтому так важно понимать отличия между моделями и доступные возможности для решения конкретных проблем. Графовые БД – это развивающаяся технология с целями, отличными от других типов БД.

Плюсы

- Гибкая и адаптивная структура
- Четкое представление взаимосвязей между сущностями
- Запросы выводят результаты в реальном времени. Скорость зависит от количества связей

Минусы

- Отсутствует стандартизированный язык запросов. Язык зависит от используемой платформы
- Графы не подходят для систем на основе транзакций
- Небольшая база пользователей; при возникновении проблема сложно получить поддержку

Заключение

Поскольку в наши дни используются все больше и больше IoT или умных устройств, на веб-сайтах с миллионами событий в день в реальном времени генерируется огромный трафик, увеличивается торговля на рынке, что и привело к созданию база данных временных рядов. Базы данных временных рядов являются обязательным элементом производственного стека для мониторинга.

В свою очередь, графовые базы данных – это отличный подход для анализа сложных отношений между объектами данных. Быстрота запросов и результаты в режиме реального времени хорошо вписываются в требования современных и стремительно растущих исследований данных. Графы – это развивающаяся технология, которую ждет еще много улучшений.

СПАСИБО ЗА ВНИМАНИЕ!

