Engineer

База даних

організована сукупність даних, з якою працюють програми

Що це таке

База даних — це масив даних, що зберігається в сталій структурі з правилами про те, що в неї може потрапити, щоб програми могли знаходити, змінювати й перевіряти її вміст; цим самим словом, яке часто скорочують до БД, нестрого називають і програму, що веде це зберігання, хоча її точна назва — система керування базами даних, або СКБД (англійською DBMS). Глосарій NIST наводить два визначення, і ширше з них говорить про сховище інформації або даних, яке може бути традиційною реляційною системою баз даних, а може й не бути.

Слово виконує три ролі, і розмова ходить по колу, коли двоє мають на увазі різні. Воно називає самі дані: клієнтів, замовлення й показання сенсорів, які зберігає організація. Воно називає рушій, що їх береже: PostgreSQL, SQL Server, SQLite. А всередині сервера воно називає один контейнер серед кількох. Підручник PostgreSQL розкладає рівні так: рядки з іменованими типізованими стовпцями складають таблиці, таблиці групуються в бази даних, а бази, якими разом керує один екземпляр сервера, утворюють кластер баз даних. SQL Server так само розрізняє екземпляр і бази даних, які він обслуговує. Тож «база гальмує» може означати одну таблицю, один контейнер або цілу машину.

Більшість рушіїв у щоденному вжитку реляційні: вони зберігають дані як відношення, тобто, математичною мовою, як таблиці, і приймають запитання мовою SQL. Підручник PostgreSQL згадує й інші способи впорядкування. Файли й каталоги Unix-подібної системи утворюють ієрархічну базу даних, а об’єктно-орієнтовані бази з’явилися пізніше. Redis називає себе data structure server: замість таблиць він пропонує рядкові значення, хеші, списки, множини та інші вбудовані типи, які застосовують для кешування, черг і обробки подій.

Рушії різняться й тим, де вони працюють. Клієнт-серверний рушій — це процес, що працює постійно і до якого багато програм звертаються мережею. Вбудований рушій — це бібліотека всередині однієї програми, і SQLite тримає всю базу даних, разом із таблицями, індексами, тригерами та поданнями, в одному звичайному файлі; сам проєкт каже, що змагається з `fopen()`, а не з серверними базами даних. Різницю видно з того, як програма підключається:

```
# клієнт-сервер: підключитися до служби, яка вже працює
psql "host=db.internal dbname=shop user=app"

# вбудований: програма відкриває файл, і рушій працює всередині неї
sqlite3 shop.db
```

Третя родина створена для аналітики. BigQuery зберігає кожен стовпець окремо, тоді як багато традиційних систем тримають поля одного рядка поруч на диску. Рядкове зберігання підходить, щоб знайти один запис; стовпцеве — щоб підсумувати одне поле по мільйонах рядків, бо запит читає лише цей стовпець і решту залишає в спокої.

Застосуйте поняття до рішення

Застосуйте «Що таке база даних?» до конкретного рішення. Ми можемо пов'язати поняття із системою, яку ви створюєте або купуєте. Структура, що відповідає реальній роботі бізнесу, тож потрібні згодом запитання до даних усе ще можна буде поставити.

Переглянути наші послуги Зв'язатися з нами →

Перевірте припущення за терміном

Використайте пояснення та пов'язані роботи, щоб сформулювати запитання для проєкту. Ми можемо перевірити припущення разом до вибору інструмента. Спроєктувала й підтримувала 6 пайплайнів ETL/ELT — 6 пайплайнів ETL на трьох рушіях баз даних

Спроєктувала й підтримувала 6 пайплайнів ETL/ELT Зв'язатися з нами →

Чому це важливо

Дані можна тримати й у файлах. База даних додає до них обіцянки на той момент, коли з ними одночасно працює багато програм.

Перша обіцянка — узгоджена картина під час спільної роботи. PostgreSQL дає кожній інструкції знімок даних такими, якими вони були в певний момент; цей підхід називається multiversion concurrency control, або MVCC, і завдяки йому читання ніколи не блокує запис, а запис ніколи не блокує читання. SQLite дозволяє скільки завгодно читачів одночасно й одного

записувача в кожну мить, а решта записувачів чекає, доки той закінчить. Ці два підходи розраховані на різну кількість записувачів, і саме ця кількість багато в чому визначає вибір.

Друга обіцянка — дані зберігають свою форму. Схема визначає, які стовпці має таблиця і якого типу кожен; ключі та обмеження визначають, які рядки взагалі можуть існувати. Коли ці правила застосовує сама база даних, їм підкоряється кожна програма, що в неї пише. Коли правила живуть лише в коді застосунку, перша ж програма, яка їх оминає, залишає рядки, від яких доведеться захищатися кожному наступному запиту.

Третя — спільна мова з діалектами. SQL — це стандарт ISO/IEC 9075, востаннє переглянутий як SQL:2023, проте додаток PostgreSQL про відповідність стандарту зазначає, що повну відповідність його Core сьогодні не заявляє жодна система. Схема й запити належать своєму рушію, а перенесення їх на інший — це окремий проєкт із власним тестуванням.

Документація SQLite зводить вибір рушія до короткого списку: дані, відокремлені мережею від коду, що надсилає SQL, багато процесів, які пишуть в одну й ту саму мить, або обсяг, що може доповзти до терабайтів, — кожна з цих ознак вказує на клієнт-серверний рушій, а в решті випадків SQLite — простіша відповідь. Звітність у великому масштабі додає ще одне питання: чи варто винести важкі читання на окрему аналітичну копію, яку наповнює ETL, щоб довгий звіт ніколи не змагався із запитами, на які чекають клієнти.

Як це виглядає на практиці

Найбільший масштаб мала роль інженера баз даних у технологічній компанії, що стрімко зростала: вона полягала в тому, щоб створити, оптимізувати й вести 100+ баз даних у PostgreSQL, Microsoft SQL Server та Google BigQuery, переважно з GIS-даними й часовими рядами: показаннями сенсорів, логами та метриками в реальному часі. З цієї роботи з’явилися нормалізовані й денормалізовані схеми для PostgreSQL і SQL Server із просторовою індексацією (PostGIS на боці PostgreSQL) і партиціонуванням часових рядів, а великі обсяги часових рядів у BigQuery лягли в партиціоновані за часом і кластеризовані таблиці. Коли додалися індексація, матеріалізовані подання й кешування, час відповіді на запити до GIS-даних і часових рядів скоротився на 40–60%, а автоматизований моніторинг скоротив час простою на 50%.

На задокументованому етапі розробки менша система зберігала факти для CV, списку рекомендацій, портфоліо й супровідного листа в одній базі даних SQLite трьома мовами. Двадцять три таблиці цієї бази даних збирав пайплайн із дев'ятнадцяти впорядкованих кроків, який однією командою перетворював порожній файл на повну базу даних, і кожен крок можна було безпечно повторювати. Сорок один модуль на Python, загалом 10 580 рядків, генерував з неї шість вихідних форматів. Профіль визначав, які досягнення з'являлися і в якому порядку, тож документ для певного типу читачів, словами сторінки, був «запитом, а не копією». Ціною було те, що відтоді документ можна було отримати лише через цю базу даних. Ці кількісні характеристики описують задокументований етап реалізації, а не поточний репозиторій чи перелік підтримуваних мов.

Між цими розмірами стоїть питання, з яким стикається кожна навантажена база даних, і нотатка про оптимізацію перед шардуванням відповідає на нього порядком дій: подивитися, підтягнути, зняти читання з основної бази, розділити таблиці навколо гарячої точки на партиції і шардувати лише тоді, коли одна машина повністю завантажена корисною роботою. Подивитися означає спитати сам PostgreSQL: `pg_stat_statements` покаже інструкції, що коштують найбільше сумарно, а `EXPLAIN (ANALYZE, BUFFERS)` — що робить найдорожча з них.

Де виникають проблеми

Більшість проблем із базами даних починається з рішення, ухваленого рано, коли даних було мало, а рішення здавалося безкоштовним.

Дві копії одного факту розходяться. Генератор CV, описаний вище, існує саме тому, що факти, які ведуть як чотири документи, доводиться виправляти чотири рази, аж поки два з них не почнуть суперечити один одному, і ніщо не вкаже, котрий актуальний. Те саме стається з таблицею, яку експортували в електронну таблицю й правлять там, або з другим сервісом, що веде власний список клієнтів. Дайте кожному факту одне місце зберігання, яке ним володіє, і нехай кожна інша копія каже, звідки й коли вона взялася. Кеш із нотатки дотримується цього правила суворо: він може сказати покупцеві, що товар повністю зарезервовано, але кожне «так» дає база даних.

Машин більшає раніше, ніж хтось прочитає плани. Розкидання даних по серверах розносить повільні інструкції на більше заліза, не роблячи жодну з них дешевшою, і кладе тривалий рахунок на все інше: як перелічує нотатка, кожна міграція, резервна копія, відновлення і звіт виконуються раз на кожен шард, а join між шардами стає кодом застосунку. Спершу прочитайте `pg_stat_statements` і плани запитів і збережіть те, що вони показали, як точку відліку.

Читання потрапляє на копію, що відстає. Потокова репліка знімає читання з основної бази й трохи від неї відстає. Користувач, який щойно щось записав і читає це з репліки, може побачити старе значення. Власні щойно зроблені записи користувача читайте з основної бази.

Звіти й клієнти ділять один рушій. Звіт, що сканує два роки історії, займає пам'ять і диск, потрібні власним запитам магазину, і покупці це відчувають. Перенесіть важку аналітику на копію в сховищі даних, а робочій базі залиште безліч дрібних читань і записів.

Що почитати далі

- NIST glossary: database — два короткі визначення з публікацій NIST, одне з яких охоплює й нереляційні бази даних.
- PostgreSQL tutorial: Concepts — таблиці, рядки, стовпці, бази даних і кластер баз даних на одній сторінці.
- PostgreSQL: Introduction to MVCC — як знімки дають читачам і записувачам працювати, не блокуючи одне одного.
- PostgreSQL: SQL Conformance — частини й версії ISO/IEC 9075 і що означає відповідність Core.
- Appropriate Uses For SQLite — де доречний вбудований рушій, де краще підходить клієнт-серверний, і контрольний список для вибору.
- Overview of BigQuery storage — рядкове й стовпцеве зберігання і чому аналітика тяжіє до другого.

Словник

Обирайте за фактичними умовами

Надайте репрезентативний набір даних, запитання до нього та поточне вузьке місце. Узгодимо вимірювання якості, пропускної здатності та відновлення.

Дивитися, скільки це коштує Зв'язатися з нами →

Зробіть термінологію практичною

Запитайте, як «Що таке база даних?» стосується вашої організації.
Конкретного прикладу з поточної системи достатньо для початку розмови.

Переглянути наші послуги

welcome@engineer.company +45 50 10 61 84 Копенгаген, Данія

Візьміть цю сторінку із собою

Візитівка

Дбайливо зроблено з любов'ю — D.

CVR 45969703 · Tagensvej 105C, 3. th, 2200 København N

Авторське право © 2025 – дотепер · Engineer ApS · Усі права захищено.

Надруковано з https://engineer.company/uk/glossary/database/