Engineer

SQL

мова запитів до реляційних баз даних

Що це таке

SQL — це декларативна мова, якою визначають, змінюють і читають дані в реляційних таблицях: ви описуєте потрібний результат, а база даних сама вирішує, як його отримати. Абревіатуру зазвичай розшифровують як Structured Query Language і вимовляють або по літерах, або як «сіквел», а міжнародний стандарт ISO/IEC 9075, що її визначає, називає мову просто Database Language SQL.

Ключове слово тут — «декларативна». Інструкція нижче називає таблицю, умову, групування й порядок, але нічого не каже про індекси, цикли чи про те, які рядки читати першими:

```
SELECT country, count(*) AS signups
FROM accounts
WHERE created_at >= now() - interval '30 days'
GROUP BY country
ORDER BY signups DESC;
```

У PostgreSQL цей текст проходить чотири етапи. Парсер перевіряє синтаксис і будує дерево, етап переписування розгортає представлення (views), згадані в інструкції, планувальник порівнює можливі способи виконання — скажімо, послідовне сканування проти індексу, — і лише тоді виконавець дістає рядки. Та сама інструкція може тривати мілісекунду або хвилину залежно від того, що знайде планувальник, тому вміння читати план — частина вміння писати SQL.

Мова складається з частин, які часто називають окремо. Визначення даних створює та змінює таблиці, схеми й індекси. Маніпулювання даними вставляє, оновлює, видаляє та зливає рядки. Запити читають їх через SELECT.

Привілеї й транзакції визначають, хто що може робити і які зміни застосовуються разом. Усе це передається текстом через одне з'єднання, тож SQL — спільна мова для застосунку, інструмента звітності й людини за терміналом.

Стандарт переглядали багато разів, востаннє — як SQL:2023, і кожна СКБД реалізує власну його підмножину. Додаток PostgreSQL про відповідність стандарту зазначає, що система виконує щонайменше 170 зі 177 обов'язкових можливостей рівня Core і що на момент написання жодна СКБД не заявляла повної відповідності Core. Понад це кожен рушій має свій діалект: Microsoft SQL Server говорить мовою Transact-SQL, BigQuery — мовою GoogleSQL, а PostgreSQL додає PL/pgSQL для процедурного коду, що виконується всередині бази.

Застосуйте поняття до рішення

Застосуйте «Що таке SQL?» 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

Рішення, яке SQL ставить перед командою, — де відбувається робота з даними: у самій базі, мовою, для виконання якої її створено, чи в коді застосунку, який вибирає рядки й обробляє їх сам.

Інструкція, що працює з множинами, виконується поруч із даними. Якщо відфільтрувати мільйон рядків до сорока в базі, мережею піде сорок рядків; якщо фільтрувати в застосунку, піде мільйон, і більшість із них викинуть. З'єднання, агрегації та прибирання дублікатів підкоряються тій самій арифметиці. Перенести таку роботу в SQL — часто найдешевший спосіб пришвидшити повільну систему, і нової інфраструктури для цього не треба.

Противага — відповідальність за код. Логіка на SQL живе в міграціях і функціях, які потребують такого самого рев'ю, тестів і контролю версій, як і решта коду, а довгу функцію на PL/pgSQL вільно читає менше інженерів, ніж

код на Go чи Python. Робочий поділ віддає базі все, що мусить бачити багато рядків, а правила, які змінюються разом із бізнесом, лишає в застосунку, де їх легше тестувати окремо.

Переносимість — другий компроміс. Ядро мови — SELECT зі з'єднаннями, GROUP BY та ORDER BY — переходить між рушіями з мінімальними правками. Краї переходять погано: арифметика дат, upsert, пагінація, функції JSON і процедурний код різняться, і перенесення з однієї бази на іншу витрачає на них більшу частину зусиль. Обираючи рушій, команда обирає й діалект, і що більше логіки живе в ньому, то дорожче коштуватиме перехід згодом.

Наймання — третій чинник. SQL — одна з небагатьох навичок, спільних для аналітиків, бекенд-розробників, інженерів даних і адміністраторів баз даних, тож шар даних на звичайному SQL може прочитати й перевірити будь-хто з них. ORM ховає SQL, доки сторінка не стане повільною, а тоді комусь однаково доведеться читати те, що вона згенерувала.

Як це виглядає на практиці

На одній платформі функції PostgreSQL збирають кожну відповідь API як готовий JSON-документ, а обробник на Go пересилає її як `json.RawMessage`, не розбираючи й не кодуючи заново. В архітектурі наскрізної передачі JSON формування відповіді стало завданням SQL і виконується там, де дані вже лежать. Невеликий помічник QueryJSON зробив цей шаблон найпростішим шляхом, і DTO, мапери та структури відповідей, які зазвичай накопичує багатошаровий API, зникли. Код обробників суттєво скоротився, а зміна того, що повертає функція, доходить до клієнта без жодної правки в обробнику.

У конвеєрі даних та сама мова виглядає інакше. У шести конвеєрах ETL/ELT на BigQuery, MSSQL і PostgreSQL сирі файли завантажували в проміжні (staging) схеми за допомогою bcp, psql і SSIS, а очищення, нормалізацію та збагачення даних написали як процедури на PL/pgSQL і T-SQL. Три рушії, кожен зі своїм діалектом, в одній роботі: описане вище питання переносимості, побачене зсередини. На сторінці зафіксовано результат: щонайменше три години ручної праці скоротилися до менш ніж 20 хвилин на весь цикл, оновлення даних перейшло зі щотижневого на щоденне, а Python API, які їх читають, стали на 30% швидшими.

Де виникають проблеми

Інструкції, склеєні зі шматків тексту. Клієнт на прізвище O'Brien ламає сторінку пошуку, або підступно складене введення читає чужі дані. Причина — введення користувача, вклеєне в текст SQL, де база не може відрізнити дані від команди. Передавайте значення як параметри, окремо від інструкції, — це дозволяє кожен поширений драйвер. У клієнтській бібліотеці самого PostgreSQL параметризований виклик до того ж приймає лише одну команду в тексті, що дає додатковий захист від SQL-ін'єкцій.

Порядок, який був випадковістю. Список, що місяцями приходив відсортованим, раптом перемішується після завантаження даних або зміни плану. Без ORDER BY PostgreSQL повертає рядки в тому порядку, який найшвидше отримати, а LIMIT без ORDER BY може навіть щоразу повертати іншу підмножину. Кожен результат, для якого важливий порядок, мусить задавати його в самому запиті.

NULL як звичайне значення. Фільтр `WHERE discount = NULL` повертає нуль рядків, і `NOT IN` щодо підзапиту з NULL поводиться так само. Будь-яке порівняння з NULL дає «невідомо», а не істину чи хибність, а WHERE залишає лише рядки, для яких умова істинна. Пишіть `IS NULL` і віддавайте перевагу `NOT EXISTS` перед `NOT IN`, коли внутрішній стовпець може містити NULL.

Діалект, що просочується в тести. Тести проходять на SQLite, а той самий код падає на PostgreSQL у продакшні; або переїзд на SQL Server спотикається об кожен `LIMIT`, який цей рушій записує як `TOP` чи `OFFSET ... FETCH`. Кожен рушій приймає те, що інші відкидають. Тестуйте на тому рушії, який справді розгортаєте, і тримайте специфічний для рушія синтаксис у якомога меншій кількості місць.

Що почитати далі

- Part II. The SQL Language — повний посібник і довідник PostgreSQL з мови, від синтаксису до порад щодо продуктивності.
- Appendix D. SQL Conformance — частини ISO/IEC 9075 і те, які можливості з них реалізує PostgreSQL.
- Transact-SQL Reference — діалект SQL Server, описаний самою Microsoft.
- Introduction to SQL in BigQuery — GoogleSQL і застарілий діалект, який він замінив.

- Query Language Understood by SQLite — синтаксис вбудованого рушія, інструкція за інструкцією.

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 SQL?» 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/sql/