Engineer

SQLite

en hel database i én fil

Hvad det er

SQLite er en SQL-databasemotor, der kører som et bibliotek inde i det program, der bruger den, uden serverproces, netværksport eller administrator, og som samler tabeller, indeks, triggere og views i én almindelig fil på disken. Navnet optræder også som `sqlite3`, fordi kommandolinjeværktøjet hedder sådan, den samlede C-kildefil hedder `sqlite3.c`, og alle funktioner i C-grænsefladen begynder med `sqlite3_`. Projektet beskriver selv motoren som selvstændig, serverløs, konfigurationsfri og transaktionel, og det beder om at blive sammenlignet med `fopen()` frem for med Oracle. Den afløser den hjemmestrikkede datafil, ikke databaseserveren.

Projektet startede den 9. maj 2000, og koden er frigivet til det fri (public domain), så der er ingen licens at afklare, før den bygges ind i et produkt. Filformatet er en del af løftet: en database skrevet på en 32-bit maskine kan kopieres til en 64-bit maskine eller mellem big-endian- og little-endian-processorer og åbnes uændret. Udviklerne har som mål at vedligeholde SQLite frem til år 2050, og det amerikanske Library of Congress anbefaler formatet til langtidsopbevaring af data. Med alle funktioner slået til kan biblioteket fylde under 900 KiB, og derfor sidder det i telefoner, biler, ure og skrivebordsprogrammer lige så vel som på servere.

Under motorhjelmen er der tale om fuld SQL med joins, underforespørgsler, views og triggere, og transaktionerne holder ACID-egenskaberne, også når strømmen går midt i en skrivning. To valg adskiller den fra serverdatabaser. Det første er samtidighed: vilkårligt mange forbindelser kan læse på én gang, men kun én kan skrive ad gangen, og låsen dækker hele filen. Det andet er typer. En kolonnes erklærede type er en præference, som SQLite kalder affinitet, mens den faktiske type hører til hver enkelt værdi. En kolonne erklæret som `INTEGER` tager derfor gerne imod tekst, der ikke ligner et tal.

Grænsen til alternativerne er klar. PostgreSQL, MySQL og SQL Server er fælles tjenester, som mange maskiner taler med over netværket. SQLite er et privat lager for ét program på én maskine. Sammenlignet med CSV- eller JSON-filer får man forespørgsler, indeks, begrænsninger og transaktioner for prisen af én afhængighed, som de fleste sprog allerede har med.

Brug begrebet i en beslutning

Brug »Hvad er SQLite?« i en konkret beslutning. Vi kan knytte begrebet til det system, I bygger eller køber. En struktur, der passer til, hvordan forretningen faktisk fungerer, så de svar, I får brug for senere, stadig kan hentes.

Se vores ydelser Kontakt os →

Undersøg antagelserne bag ordet

Brug forklaringen og det tilknyttede arbejde til at formulere spørgsmål til jeres projekt. Vi kan undersøge antagelserne sammen, før et værktøj vælges. Har designet og drevet 6 ETL/ELT-pipelines — 6 ETL-pipelines på tre databasemotorer

Har designet og drevet 6 ETL/ELT-pipelines Kontakt os →

Hvorfor det betyder noget

Det svære spørgsmål ved en database er ofte ikke, hvor hurtig den er, men hvem der skal passe den. En server skal installeres, opdateres, sikres, overvåges og tages backup af, og nogen skal have løn for det. SQLite fjerner hele den post. At åbne databasen er at åbne en fil, en backup er en kopi af filen, taget med backup-API'et, mens den er i brug, og en testsuite kan give hver test sin egen friske database.

Den ændrer også, hvad en fil kan være. Et program, der ellers ville opfinde sit eget format, skrive en parser til det og håbe på, at et nedbrud aldrig rammer midt i en lagring, kan lægge sin tilstand i SQLite og få atomisk lagring foræret. Projektet nævner selv brugen som programmets filformat, som beholder til at flytte data mellem systemer, hvor modtageren skriver SQL i stedet for en parser, og som kladdedatabase til at sortere og opsummere data, der kom i en anden form.

Valget koger ned til ét spørgsmål, der kan besvares, før der skrives en linje kode: hvor mange processer skriver, og på hvor mange maskiner? Ét program på én maskine, der skriver i korte ryk, er SQLites hjemmebane, og de fleste websites med lav til moderat trafik kører fint på den. Flere servere, der skriver til de samme data, eller mange brugere, der skriver hele tiden, hører hjemme i en klient-server-database. Størrelsen afgør det sjældent, for én fil kan rumme op til 281 terabyte. Det gør antallet af skrivere.

Også på hjemmebanen er der omkostninger. `ALTER TABLE` kan kun omdøbe en tabel eller en kolonne og tilføje eller fjerne en kolonne, så de fleste andre ændringer af en tabels form betyder en ny tabel, en kopi af rækkerne og et navnebytte. Fremmednøgler er slået fra, indtil hver forbindelse slår dem til. Og den løse typning lukker forkerte data ind uden en lyd, medmindre tabellen er erklæret til at afvise dem.

Sådan ser det ud i praksis

I den dokumenterede udviklingsfase havde en generator til CV'er, referenceark, porteføljer og ansøgninger én kilde til sandheden: en SQLite-database med 23 tabeller, fra achievements og virksomheder til profiler og referencer, bygget af en pipeline i 19 trin, der gik fra en tom fil til en komplet database med én kommando. Hvert trin tålte at blive kørt igen, så pipelinen kunne køre mod en eksisterende database uden at fordoble en eneste række. Fra den ene fil gengav 41 Python-moduler på i alt 10.580 linjer seks outputformater med præcis to afhængigheder ved kørsel. Et dokument til en bestemt type læser var en forespørgsel, styret af en profil, ikke en redigeret kopi. Prisen stod der også: systemet var blevet den eneste vej til et dokument.

I den dokumenterede eksportfase var SQLite også input til et build. Dette sites egen portefølje blev genereret fra den samme database. En eksportør åbnede SQLite-filen skrivebeskyttet og skrev 495 sider på tre sprog med kun Pythons standardbibliotek, og resultatet var committet, så sitet kunne bygges uden generatorens miljø. Sideadresser lå som rækker, én pr. adresse pr. achievement, så en omformuleret sætning ikke længere flyttede siden. Samme arbejde dokumenterede fælden i næste afsnit: en periodesammenligning mod en tekstkolonne ramte alle toogtredive rækker.

Den mest typiske rolle er også den mindste: et sted at efterse resultater. En pipeline, der lavede rå elmålinger om til timeforbrug pr. zone, skrev sine resultater som CSV eller i en let SQLite-database til eftersyn, og Python-versionen behandlede et døgns data på 34 ms og tredive dages data på 180 ms. En database brugt på den måde kræver ingen opsætning: alle med `sqlite3`-værktøjet kan åbne filen og stille spørgsmål til den.

Hvor det går galt

Det meste besvær med SQLite opstår, når den behandles som en lille PostgreSQL. Den er en anden slags maskine med andre kanter.

Et filter på årstal rammer hver eneste række. SQLite konverterer kun en værdi før en sammenligning, når den ene side har en affinitet. Resultatet af en funktion som `substr()` har ingen, og det har et løst tal heller ikke, så teksten `'2019'` sammenlignes med heltallet `2025`, som de er. I SQLites sorteringsorden kommer ethvert tal før enhver tekst:

```
CREATE TABLE stint (period TEXT, hours TEXT);
INSERT INTO stint VALUES
  ('2019-03', '9'), ('2024-11', '10'), ('2026-01', '42');

SELECT count(*) FROM stint
WHERE substr(period, 1, 4) >= 2025;                   -- 3

SELECT count(*) FROM stint
WHERE CAST(substr(period, 1, 4) AS INTEGER) >= 2025;  -- 1
```

Cast eksplicit, hvor tekst møder tal. Giv nye tabeller rigtige typer og nøgleordet `STRICT`, som findes fra version 3.37.0: en streng `INTEGER`-kolonne tager stadig imod `'42'` og gemmer det som et tal, men afviser `'ten'` med en fejl.

Skrivninger begynder at fejle med `database is locked`. En forbindelse holder skrivelåsen, og den næste skriver venter eller giver op med det samme. Årsagen er som regel en lang transaktion eller en læser i standardtilstanden med rollback-journal, der holder en skriver tilbage. Slå write-ahead logging til, så læsere og skriver holder op med at blokere hinanden, hold skrivetransaktioner korte, og sæt en busy-timeout, så en skriver venter lidt i stedet for at fejle. Er skriverne reelt mange og konstante, er det tegnet til at flytte til en server.

Filen bliver korrupt på et netværksdrev. SQLite læner sig op ad styresystemets fillåse, og mange netværksfilsystemer har fejl i netop dem, så to maskiner kan komme til at skrive i samme del af filen samtidig. Write-ahead logging virker slet ikke over et netværksfilsystem, fordi alle processer, der bruger databasen, skal dele hukommelse på én maskine. Læg databasen på en lokal disk hos det ene program, der skriver i den, og sæt en server foran, hvis andre maskiner skal have adgang til dataene.

En række overlever den forælder, den peger på. Fremmednøgler står i skemaet, men er inaktive, indtil hver forbindelse kører `PRAGMA foreign_keys = ON`. En sletning fra en forbindelse, der sprang den over, efterlader forældreløse

rækker uden en lyd. Sæt pragmaen dér, hvor programmet åbner sine forbindelser, ét sted i koden, og test, at den er slået til.

Læs mere

- About SQLite — projektets egen beskrivelse: i programmets egen proces, serverløs, public domain og én fil på tværs af platforme.
- Datatypes In SQLite — lagringsklasser, typeaffinitet og de sammenligningsregler, der ligger bag fælden ovenfor.
- STRICT Tables — sådan får man en tabel til at håndhæve sine erklærede kolonnetyper.
- Write-Ahead Logging — læsere og en skriver side om side, og hvorfor det kræver én maskine.
- Appropriate Uses For SQLite — hvor den passer, hvor en klient-server-database passer bedre, og en tjekliste.
- SQLite Foreign Key Support — hvorfor begrænsningerne er slået fra som standard, og hvordan de slås til.

Ordliste

Vælg ud fra jeres faktiske vilkår

Medbring et repræsentativt datasæt, de spørgsmål det skal besvare, og den aktuelle flaskehals. Vi kan aftale målinger af kvalitet, gennemløb og gendannelse.

Se, hvad det koster Kontakt os →

Gør begreberne praktiske

Spørg, hvordan »Hvad er SQLite?« gælder i jeres organisation. Et konkret eksempel fra jeres nuværende system er nok til at begynde samtalen.

Se vores ydelser

welcome@engineer.company +45 50 10 61 84 København, Danmark

Tag denne side med dig

Visitkort

Lavet med kærlighed af D.

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

Copyright © 2025 – nu · Engineer ApS · Alle rettigheder forbeholdes.

Udskrevet fra https://engineer.company/da/glossary/sqlite/