Engineer

SQL

språket relasjonsdatabaser svarer på

Hva det er

SQL er et deklarativt språk for å definere, endre og lese data som ligger i relasjonelle tabeller: du skriver ned resultatet du vil ha, og databasen bestemmer hvordan det skal lages. Bokstavene står vanligvis for Structured Query Language og uttales enten bokstav for bokstav eller som det engelske «sequel», mens den internasjonale standarden som definerer språket, ISO/IEC 9075, bare kaller det Database Language SQL.

Deklarativ er ordet som bærer vekten. Setningen nedenfor navngir en tabell, en betingelse, en gruppering og en rekkefølge. Den sier ingenting om indekser, løkker eller hvilke rader som skal leses først:

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

I PostgreSQL går den teksten gjennom fire trinn. En parser sjekker syntaksen og bygger et tre, et omskrivingstrinn folder ut eventuelle visninger setningen nevner, en planlegger sammenligner de mulige måtene å kjøre den på, f.eks. et sekvensielt søk mot en indeks, og en utfører henter til slutt radene. Den samme setningen kan ta et millisekund eller et minutt, alt etter hva planleggeren finner, og derfor hører det å lese en plan med til å skrive SQL.

Språket har deler som folk gir egne navn. Datadefinisjon oppretter og endrer tabeller, skjemaer og indekser. Datamanipulering setter inn, oppdaterer, sletter og fletter rader. Spørringer leser dem med SELECT. Rettigheter og transaksjoner

avgjør hvem som får gjøre hva, og hvilke endringer som lander samtidig. Alt dette reiser som tekst over én enkelt tilkobling, så SQL er fellesgrunnen mellom en applikasjon, et rapportverktøy og et menneske ved en terminal.

Standarden er revidert mange ganger, sist som SQL:2023, og hver motor implementerer sin egen delmengde av den. Vedlegget om samsvar i PostgreSQL-dokumentasjonen oppgir at PostgreSQL oppfyller minst 170 av de 177 obligatoriske Core-funksjonene, og at ingen databasesystemer hevdet fullt samsvar med Core da teksten ble skrevet. Hver motor legger også til en egen dialekt: Microsoft SQL Server snakker Transact-SQL, BigQuery snakker GoogleSQL, og PostgreSQL legger til PL/pgSQL for prosedyrekode som kjører inne i databasen.

Bruk begrepet i en beslutning

Bruk «Hva er SQL?» i en konkret beslutning. Vi kan knytte begrepet til systemet dere bygger eller kjøper. En struktur som samsvarer med hvordan virksomheten faktisk fungerer, slik at spørsmålene du trenger svar på senere, fortsatt er mulige å stille.

Se tjenestene våre Ta kontakt →

Undersøk antakelsene bak ordet

Bruk forklaringen og det tilknyttede arbeidet til å formulere spørsmål for prosjektet. Vi kan undersøke antakelsene sammen før et verktøy velges. Designet og driftet 6 ETL/ELT-pipelines — 6 ETL-pipelines på tvers av tre databasemotorer

Designet og driftet 6 ETL/ELT-pipelines Ta kontakt →

Hvorfor det er viktig

Beslutningen SQL legger foran et team, er hvor arbeidet med dataene skal skje: inne i databasen, i språket den er bygd for å utføre, eller i applikasjonskode som henter rader og behandler dem selv.

En mengdebasert setning kjører ved siden av dataene. Å filtrere en million rader ned til førti inne i databasen sender førti rader over nettverket, mens filtrering i applikasjonen sender en million og kaster de fleste. Sammenkoblinger, aggregeringer og fjerning av duplikater følger samme regnestykke. Å flytte den typen arbeid inn i SQL er ofte den billigste hastighetsøkningen et tregt system har tilgjengelig, fordi den kjører på databasen som allerede står der.

Eierskapet er motvekten. Logikk skrevet i SQL lever i migreringer og funksjoner, som trenger den samme gjennomgangen, de samme testene og den samme versjonskontrollen som all annen kode, og færre utviklere leser en lang PL/pgSQL-funksjon like flytende som de leser Go eller Python. En fordeling som

fungerer, sender alt som må se mange rader, til databasen, og holder regler som endrer seg med forretningen, i applikasjonen, der de er lettere å teste hver for seg.

Flyttbarhet er den andre avveiningen. Kjernen i språket, SELECT med joins, GROUP BY og ORDER BY, flytter mellom motorer med få endringer. Kantene flytter dårlig: datoregning, upserts, paginering, JSON-funksjoner og prosedyrekode varierer fra motor til motor, og en migrering fra én database til en annen bruker mesteparten av innsatsen der. Å velge en motor er også å velge en dialekt, og jo mer logikk som bor i den dialekten, desto dyrere blir en senere flytting.

Rekruttering er den tredje. SQL er en av få ferdigheter som analytikere, backendutviklere, dataingeniører og databaseadministratorer har felles, så et datalag skrevet i ren SQL kan leses og gjennomgås av hvem som helst av dem. En ORM skjuler SQL-en helt til en side blir treg, og da må noen likevel lese det den genererte.

Slik ser det ut i praksis

På én plattform setter PostgreSQL-funksjoner sammen hvert API-svar som et komplett JSON-dokument, og Go-handleren sender det videre som `json.RawMessage` uten å pakke det ut eller kode det på nytt. JSON-passthrough-arkitekturen gjør formingen av svaret til en SQL-oppgave, utført der dataene allerede ligger. En liten QueryJSON-hjelper gjorde det mønsteret til det enkleste å følge, og DTO-ene, mappingkoden og respons-structene som et lagdelt API vanligvis samler opp, forsvant. Handlerkoden ble mye kortere, og en endring i hva en funksjon returnerer, når klienten uten at noen redigerer en handler.

Det samme språket ser annerledes ut i en datapipeline. På tvers av seks ETL/ELT-pipeliner som spente over BigQuery, MSSQL og PostgreSQL, ble rå filer lastet inn i staging-skjemaer med bcp, psql og SSIS, og rensingen, normaliseringen og berikingen ble skrevet som prosedyrer i PL/pgSQL og T-SQL. Tre motorer, hver med sin dialekt, i én jobb: spørsmålet om flyttbarhet ovenfor, sett innenfra. Siden måler resultatet for hele pipelinen, SQL-trinnene medregnet: en jobb på minst tre timer for hånd tar nå under 20 minutter, dataene oppdateres daglig i stedet for ukentlig, og Python-API-ene som leser dem, har fått 30 % bedre ytelse.

Hvor det går galt

Setninger limt sammen av strenger. En kunde som heter `O'Brien`, får en søkeside til å feile, eller en nøye tilpasset verdi i et skjemafelt leser andres data. Årsaken er det brukeren skrev, limt rett inn i SQL-teksten, der databasen har mistet evnen til å skille data fra kommando. Send verdiene som parametere i stedet, atskilt fra setningen, slik hver utbredt driver tillater. I PostgreSQLs eget klientbibliotek godtar et parameterisert kall dessuten bare én kommando per streng, noe som gir et ekstra forsvar mot injeksjon.

Rekkefølge som bare var flaks. En liste som kom tilbake sortert i månedsvis, kommer i tilfeldig rekkefølge etter en datainnlasting eller en ny plan. Mangler ORDER BY, returnerer PostgreSQL radene i den rekkefølgen som er raskest å produsere, og en LIMIT uten ORDER BY kan til og med gi et annet utvalg hver gang. Hvert resultat der rekkefølgen betyr noe, må si det i spørringen.

NULL behandlet som en verdi. Et filter skrevet `WHERE discount = NULL` returnerer ingenting, og `NOT IN` mot en underspørring som inneholder en NULL, returnerer heller ingenting. En sammenligning med NULL gir ukjent i stedet for sann eller usann, og en WHERE-klausul beholder bare radene der betingelsen er sann. Skriv `IS NULL`, og foretrekk `NOT EXISTS` framfor `NOT IN` når den indre kolonnen kan inneholde NULL.

Dialekt som lekker inn i testene. En testsuite består mot SQLite, og den samme koden feiler mot PostgreSQL i produksjon, eller en flytting til SQL Server snubler på hver `LIMIT`, som den motoren skriver `TOP` eller `OFFSET ... FETCH`. Hver motor godtar ting de andre avviser. Test mot motoren du produksjonssetter, og hold motorspesifikk syntaks på så få steder som mulig.

Les mer

- Part II. The SQL Language — PostgreSQLs fullstendige innføring i og referanse for språket, fra syntaks til råd om ytelse.
- Appendix D. SQL Conformance — delene av ISO/IEC 9075 og hvilke funksjoner PostgreSQL implementerer.
- Transact-SQL Reference — dialekten i SQL Server, dokumentert av Microsoft.
- Introduction to SQL in BigQuery — GoogleSQL og den eldre dialekten den erstattet.

- Query Language Understood by SQLite — syntaksen til en innebygd motor, setning for setning.

Ordliste

Velg ut fra deres faktiske rammer

Ta med et representativt datasett, spørsmålene det skal besvare og dagens flaskehals. Vi kan avtale målinger av kvalitet, gjennomstrømning og gjenoppretting.

Se hva det koster Ta kontakt →

Gjør begrepene praktiske

Spør hvordan «Hva er SQL?» gjelder i organisasjonen deres. Et konkret eksempel fra dagens system er nok til å starte samtalen.

Se tjenestene våre

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

Ta med deg denne siden

Visittkort

Laget med kjærlighet av D.

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

Opphavsrett © 2025 – nå · Engineer ApS · Alle rettigheter forbeholdt.

Skrevet ut fra https://engineer.company/nb/glossary/sql/