Engineer

BigQuery

Googles lager til forespørgsler på meget store tabeller

Hvad det er

BigQuery er en administreret SQL-tjeneste til analyse på Google Cloud: ingen skal stille en server op til den, tabellerne gemmes kolonne for kolonne, og hver SQL-forespørgsel betales enten efter de bytes, den læser, eller af regnekraft, der er reserveret på forhånd. Den kaldes også Google BigQuery, og dens SQL-dialekt hedder GoogleSQL, som følger ANSI-standarden og lægger sine egne funktioner oveni.

Grundideen er at holde lagring og beregning adskilt. Tabellerne ligger i Googles lagerlag, og en forespørgsel låner regnekraft, så længe den kører. Fordi hver kolonne gemmes for sig, læser en forespørgsel, der nævner tre kolonner i en tabel med halvtreds, kun de tre og lader resten ligge. Derfor ser tuning anderledes ud end på en PostgreSQL-server. I stedet for at designe B-tree-indeks og dimensionere hukommelse handler arbejdet om tabellens form: den kan partitioneres efter en dato- eller tidsstempelkolonne, efter indlæsningstidspunkt eller efter et heltalsinterval, og den kan klynges på op til fire kolonner, så rækker, der hører sammen, deler lagerblokke. Det sidste kaldes clustering, også på dansk.

BigQuery hører til familien af datawarehouses, og de produkter, den typisk vejes op imod, er Snowflake og Amazon Redshift, plus muligheden for at blive på den transaktionsdatabase, virksomheden allerede driver. Den er bygget til forespørgsler, der løber millioner eller milliarder af rækker igennem og returnerer en håndfuld summer. De enkeltrækkeopslag og opdateringer, en applikation laver ved hver anmodning, hører hjemme i en database som PostgreSQL eller SQL Server, og en udbredt opstilling kører begge dele med en pipeline, der fører data fra den ene til den anden.

Brug begrebet i en beslutning

Brug »Hvad er BigQuery?« i en konkret beslutning. Vi kan knytte begrebet til det system, I bygger eller køber. Tal fra salg, drift og økonomi samlet ét sted, så to afdelinger ikke længere kommer til det samme møde med hver sit svar.

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 gjort en geodatapipeline 50 gange hurtigere — 50x hurtigere geografiske datapipelines

Har gjort en geodatapipeline 50 gange hurtigere Kontakt os →

Hvorfor det betyder noget

Det valg, BigQuery stiller et team over for, handler om, hvordan det vil betale for analyse. Regningen følger de forespørgsler, folk skriver, og ikke størrelsen på en maskine, nogen engang valgte, og det trækker i begge retninger.

Med on-demand-prisen afregnes hver forespørgsel efter de bytes, den behandler. Det passer til uregelmæssigt arbejde: et team, der kører et par tunge rapporter om ugen, betaler for de rapporter og for intet derimellem, og ingen server står og summer om natten. Til gengæld får en skødesløs forespørgsel en pris. En `SELECT *` over et års sensormålinger koster det samme, uanset om analytikeren ville have én række eller dem alle. Kapacitetsprisen erstatter det med slots, enheder af regnekraft købt gennem reservationer, så regningen bliver et beløb, man kender på forhånd, og tungt, jævnt forbrug bliver lettere at budgettere. Et projekt kan skifte mellem de to modeller, så det første valg kan tages op igen, når det faktiske forbrug er til at se.

Lagring har sine egne regler. En tabel eller partition, der har stået uændret i 90 dage i træk, overgår til langtidsprisen, som er 50 % lavere, uden at nogen flytter den. Hvert datasæt afregnes enten efter den logiske, ukomprimerede størrelse af dataene eller efter den fysiske, komprimerede størrelse, og for data, der komprimerer godt, kan den anden model blive den billigste.

Tre ting fortjener vægt, før man skriver under. Et datasæts placering, en enkelt region eller multiregionen US eller EU, fastlægges, når datasættet oprettes, og ligger fast derefter, så det er en compliance-beslutning, der træffes på første dag. Dialekten, kommandolinjeværktøjet `bq` og klientbibliotekerne er Googles egne, så forespørgsler flytter til et andet datawarehouse med rettelser og ikke uændrede. Selve dataene slipper til gengæld let ud, for tabeller kan eksporteres

til CSV, Avro og Parquet. Og ansættelsesspørgsmålet er mildere, end det ser ud: analytikere, der kan SQL, er produktive i løbet af få dage, mens den sjældnere færdighed er at læse en forespørgsel i bytes og ikke kun i rækker.

Hvornår er BigQuery så det rigtige valg? Når data kommer i store mængder, der mest bliver føjet til, som hændelser, logs eller måleraflæsninger, når spørgsmålene til dem skifter fra uge til uge, og når teamet hellere vil være fri for at drive databaseservere til analyse. For data, som én PostgreSQL-server rummer uden besvær, tilføjer et datawarehouse et ekstra system, en ekstra regning og en pipeline imellem de to. Argumentet for det vokser med mængden og med antallet af folk, der stiller spørgsmål.

Sådan ser det ud i praksis

I et miljø med over 100 databaser stod BigQuery ved siden af PostgreSQL og Microsoft SQL Server, og de fleste data var geografiske eller tidsserier: koordinater, sensormålinger, logs og målinger i realtid. På BigQuery-siden blev de store tidsserietabeller tidspartitioneret og klynget, og komprimering og kolonnebaseret lagring blev brugt til at holde lageromkostningerne nede og scanningerne hurtige. Sammen med indeksering, materialiserede views og caching på tværs af miljøet faldt svartiderne for forespørgsler på GIS- og tidsseriedata med 40–60 %.

I en geografisk datapipeline, der var blevet langsom, blev SQL-koden og datamodellerne i PostgreSQL, MS SQL og Google Cloud BigQuery først analyseret, og det afslørede svag indeksering, ineffektive forespørgsler og manglende constraints. Derefter fik hver motor det værktøj, den har: B-tree- og GiST-indeks i PostgreSQL, filtrerede indeks i MS SQL og clustering i BigQuery, sammen med normaliserede modeller, partitionering og omskrevne forespørgsler. Grundlæggerens resultatbeskrivelse angiver, at pipelinen som helhed blev 50 gange hurtigere; den angiver ikke benchmarkets datasæt, hardware eller sammenligningsprocedure.

På data fra smart-målere i flere regioner blev over 500 analyserapporter skrevet i ren SQL mod PostgreSQL og BigQuery, med CTE'er, vinduesfunktioner, underforespørgsler og geospatiale joins over måler-id'er, tidsstempler, koordinater og miljømetadata. Hver rapport fulgte med sin SQL kommenteret, så en kollega kunne køre den igen og efterprøve den, og med noter om kendte

problemer med datakvaliteten som manglende GPS-koordinater og ødelagte målerværdier. Rapporterne blev et fast opslagsværk på tværs af afdelinger og bidrog til, at den operationelle planlægning blev 10–15 % mere effektiv.

I alle tre tilfælde arbejdede BigQuery sammen med PostgreSQL i stedet for at erstatte den, og hver motor fik det arbejde, den er bedst til.

Hvor det går galt

En forespørgsel med `LIMIT 10` bliver afregnet for hele tabellen. På en tabel uden clustering begrænser `LIMIT` resultatet, ikke de læste bytes, og `SELECT *` læser hver eneste kolonne. Et hurtigt kig på dataene bliver til en fuld scanning. Nævn de kolonner, spørgsmålet kræver, og se eksempelrækker gennem konsollens forhåndsvisning af tabellen eller `bq head`, som begge er gratis.

En partitioneret tabel bliver scannet fra ende til anden alligevel. Partitioner springes kun over, når filteret på partitioneringskolonnen er en konstant, som motoren kan beregne, før den læser. Sammenlignes kolonnen med resultatet af en underforespørgsel, pakkes den ind i en funktion, eller regnes der på den, går forespørgslen gennem hver eneste partition. To værn hjælper. En tabel kan kræve et partitionsfilter, så en forespørgsel uden et brugbart filter stopper med en fejl i stedet for at køre. Og en dry run oplyser, hvor mange bytes en forespørgsel ville behandle, uden at køre den eller tage betaling for den:

```
bq query --use_legacy_sql=false --dry_run \
  'SELECT meter_id, kwh
   FROM energy.readings
   WHERE reading_date = DATE "2026-10-01"'
```

Der bliver tilføjet clustering, og de læste bytes rykker sig knap. Clustering sorterer lagerblokke efter op til fire kolonner i rækkefølge, og rækkefølgen betyder noget: en tabel, der er klynget efter ordredato, land og status, er ikke organiseret til et filter på land og status alene. Tabeller under 64 MB får kun lidt ud af det, og på en klynget tabel kendes de bytes, der faktisk scannes, først, når forespørgslen er færdig. Sæt clustering-kolonnerne i den rækkefølge, forespørgslerne filtrerer i. Hvor omkostningen skal have et loft, før noget overhovedet kører, så støt dig til partitionering, hvis beskæring en dry run allerede regner med, og sæt et maksimum for afregnede bytes, så en forespørgsel over grænsen stopper, før den koster noget.

En forespørgsel, der joiner to datasæt, bliver afvist. Hvert datasæt bor ét sted, og en forespørgsel kører ét sted, så alle de datasæt, den rører, skal dele placering. Teams, der lader hvert projekt tage sin egen standard, ender med tabeller, der kun kan joines, når den ene er kopieret over. Vælg placeringen én gang, ud fra compliance og ud fra, hvor dataene kommer fra, og opret hvert datasæt der med vilje.

Læs mere

- BigQuery overview — Googles egen beskrivelse af tjenesten, de adskilte lag til lagring og beregning, og hvad der bliver administreret for dig.
- Introduction to partitioned tables — partitioneringstyperne, deres granularitet og deres begrænsninger.
- Query partitioned tables — hvilke filtre der beskærer partitioner, og hvilke der sender en forespørgsel gennem dem alle.
- Introduction to clustered tables — hvordan kolonnernes rækkefølge styrer, hvilke blokke der springes over, og hvad det betyder for omkostningsoverslag.
- Estimate and control costs — dry runs, maksimum for afregnede bytes og de gratis måder at se en tabel på.
- BigQuery locations — regioner, multiregioner og reglerne for forespørgsler, der rører mere end ét datasæt.

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 BigQuery?« 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/bigquery/