Samenvatting Databanken III - NoSQL

TIN 3 - HoGent

Lorenz Verschingel

1 november 2015

1 Distributed Systems

Een gedistribueerd systeem bestaat uit verschillende computers en software componenten. Deze communiceren met elkaar via het netwerk en delen de resources. Het systeem kan mainframes, workstations... bevatten.

1.1 Voordelen van het gedistribueerd systeem

1. Reliability:

   Als één machine crashed wordt de rest hierdoor niet beïnvloed.

2. Scalability:

   Men kan makkelijk machines toevoegen als daar nood aan is.

3. Performance:

   De verzameling aan processoren van de verschillende machines kan een hoger performantie bieden dan een centralized computer.

4. Open system:

   Iedere service is toegankelijk voor iedere client.

5. Sharing of Resources
6. Flexibility
7. Speed

1.2 Nadelen van het gedistribueerd systeem

1. Software:

   Er is minder software support → Dit is het grootste probleem

2. Security:

   Het delen van data tussen de machines verhoogt het risico op het vlak van security.

3. Troubleshooting
4. Networking

1.3 Scalability

1.3.1 Verticaal schalen

Resources toevoegen aan een bestaande logische eenheid om de capaciteit te verhogen.

1.3.2 Horizontaal schalen

Meer machines (nodes) toevoegen aan het systeem om de capaciteit te verhogen.

2 NoSQL

2.1 Wat?

NoSQL = Not Only SQL

- Geen relationeel model
- Draait goed op clusters
- Meestal open-source
- Schema-less

2.2 Waarom?

NoSQL laat developers toe om met objecten onmiddellijk in de database te stoppen, zonder ze eerste te converteren naar het relationeel model.

NoSQL databanken zijn goedkoper dan relationele databanken. De laatste zijn ook niet bedoeld om foto's en dergelijke in op te slaan terwijl dit nu de snelst groeiende data is.

→ Gebruik de juiste databank voor de juiste toepassing

2.3 CAP theorem

1. Consistency: Alle nodes in het systeem zien dezelfde data op hetzelfde moment.
2. Availability: Wanneer één van de nodes in het systeem uitvalt beperkt het de andere nodes niet in hun functioneren.
3. Partition Tolerance: Het systeem blijft functioneren bij het willekeurig verliezen van berichten.

Je kan maar 2 tegelijk garanderen:

CA Single site cluster, hierdoor zijn de nodes altijd met elkaar in contact. Als een partitie bijkomt, blokkeert het systeem.

CP Niet alle data is beschikbaar. Wat beschikbaar is, is consistent.

AP Het syteem blijft altijd functioneren, maar de data die geretourneerd wordt is niet altijd consistent.

2.3.1 Voorbeeld: Riak

Riak is een distributed key-value database. In essentie zijn er drie variabelen:

1. r: het aantal nodes dat moet antwoorden alvorens een read request succesvol wordt beschouwd.
2. w: het aantal nodes dat moet antwoorden alvorens een write request succesvol wordt beschouwd.
3. n: de replicatie factor (aantal nodes waarop de data gekopieerd wordt.

Als een Riak cluster bijvoorbeeld vijf nodes heeft dan kan Riak zorgen voor hoge beschikbaarheid of hoge consistentie door r en w aan te passen.

- Hoge beschikbaarheid: r en w zijn beide laag
- Hoge consistentie: r en w zijn beide hoog

2.4 Verschillende soorten NoSQL databases

Er zijn vier soorten NoSSQL databases:

1. Key Value: Kan alles ingestopt worden
2. Document: Werkt makkelijk met objecten (JSON, XML)
3. Wide Column: Aggregaties berekenen
4. Graph: Relaties zien

2.4.1 Key-Value Databases

Dit soort databank kan beschouwd worden als een grote look-up table. Een key kan vrijwel van elk datatype zijn. Aan deze key kan dan eender welke value gekoppeld zijn. Het is aan de applicatie om te weten van welk type de value is.

vb: Riak, Redis

2.4.2 Document Databases

In dit type databank worden zelf gedefinieerde objecten opgeslagen in de vorm van JSON, XML, BSON... Deze opgeslagen objecten zijn zelf beschrijvend en worden dan documenten genoemd.

vb: MongoDB

2.4.3 Column Family Stores

De data wordt per kolom familie opgeslagen. Hierdoor kan men een stukje van een 'rij' opvragen en niet de hele 'rij'. Dit resulteert in grote performantie voor aggregaties.

vb: HBase, Cassandra

2.4.4 Graph Databases

Bij dit soort databanken komt het erop neer dat de databank wordt voorgesteld door een graaf. Hierbij zijn de nodes de objecten met al hun properties en de bogen zijn dan de relaties tussen de objecten. Ook de bogen kunnen in dit soort databank properties hebben. De relaties zijn ook persistent gemaakt. Dit is veel sneller dan ze telkens te berekenen voor iedere query.

2.5 Wanneer welke databank

1. Key-Value:
   - Opslaan van sessie informatie, user profile, preferences...
   - Als er geen er geen relaties zijn tussen de opgeslagen data
2. Document:
   - content management, blogging, web analytics
   - er zijn geen complexe transacties nodig zijn
3. Column:
   - content management, blogging, maintaining counters, expiring usage

- niet gebruiken in vroeg stadium van development of waar de queries nog veranderen

4. Graph:
    - social networks, spatial data, routing information, recommendation engines.

2.6 Schema-less ramification

Het belangrijkste gevolg hiervan is dat ontwikkelaars er rekening mee moeten houden dat sommige velden leeg kunnen zijn.

2.7 Conclusie

De opkomst van NoSQL databanken betekend niet dat relationele databanken zullen verdwijnen.

Wel wordt er gehamerd op polyglot persistence. Dit wil zeggen dat men de juiste databank moet gebruiken voor de juiste toepassing. Polyglot persistence kan toegepast worden binnen een bedrijf, maar ook binnen één applicatie.