SaaS: Data as a Service

Pengantar

Database adalah sebuah kumpulan data yang disusun berdasarkan cara tertentu dan merupakan suatu kesatuan yang utuh. Dengan sistem database tersebut data yang terhimpun dapat diolah dan digunakan oleh client sehingga menjadi informasi yang berguna. Hubungan antar data dapat ditunjukan dengan adanya field/kolom kunci dari tiap file/tabel yang ada. Dapat dilakukan operasi penyimpanan, pengaturan, penambahan, penghapusan, dan pembaruan database. Lebih singkatnya lagi, database (basis data) merupakan kumpulan data yang saling berhubungan. Hubungan antar data dapat ditunjukan dengan adanya field/kolom kunci dari tiap file/tabel yang ada.

Database dapat bekerja karena adanya aplikasi yang bertugas untuk mengatur, menyimpan, memodifikasi data. Aplikasi itu disebut dengan software database engine dan lebih resminya disebut dengan DBMS *(Database Management System)*. Ada banyak sekali aplikasi DBMS ini, mulai yang berjalan di komputer Personal (PC) sampai ke komputer skala mainframe.
Contoh dari aplikasi database engine, misalnya seperti SQL Server oleh microsoft, MS Access oleh Microsoft, Oracle Database dibuat oleh Oracle, MySQL dibuat oleh MySQL AB, dan lain-lainnya.

Sedangkan Cloud database adalah sebuah database yang dapat di akses oleh client dari cloud service yang didistribusikan ke user melalui internet oleh cloud provider. Dengan adanya cloud database, pengguna tidak perlu lagi menyimpan data pada hard driver, CD, ataupun hardware lainnya. Pengguna cukup menyimpan data pada remote database yang disediakan oleh pihak ketiga, untuk mengaksesnya, pengguna hanya memerlukan koneksi internet. Kelemahan Cloud database adalah dari segi keamanan karena data yang tersimpan di dalam internet bisa saja semua orang di dunia ini mendapatkannya dengan cara hacking. Cloud Database berkaitan erat dengan Cloud computing yang mana merupakan sebuah metode komputasi dimana kemampuan TI disediakan sebagai layanan berbasis internet.

SQL vs NoSQL

1. SQL

SQL merupakan singkatan dari *Structured Query Language*. Bahasa yang digunakan untuk mengatur/mengelola data dalam database relasional. Database relasional menggunakan 'relasi' (yang biasanya disebut tabel) untuk menyimpan data dan mencocokkan data tersebut dengan memakai karakteristik umum di setiap dataset. Beberapa contoh database management system yang menggunakan SQL antara lain Oracle, Sybase, Microsoft SQL Server, PostgreSQL.

SQL untuk membuat data (dalam objek yang disebut tabel) dan skema untuk data tersebut yang mendeskripsikan isian dalam beberapa kolom. Sedangkan untuk setiap record dalam SQL database disebut 'baris'.

Query atau sintaks pemanggilan data pada database SQL memanfaatkan media relasi tabel. Beberapa kelompok data dari tabel-tabel yang berbeda dapat dipanggil bersamaan secara serentak.

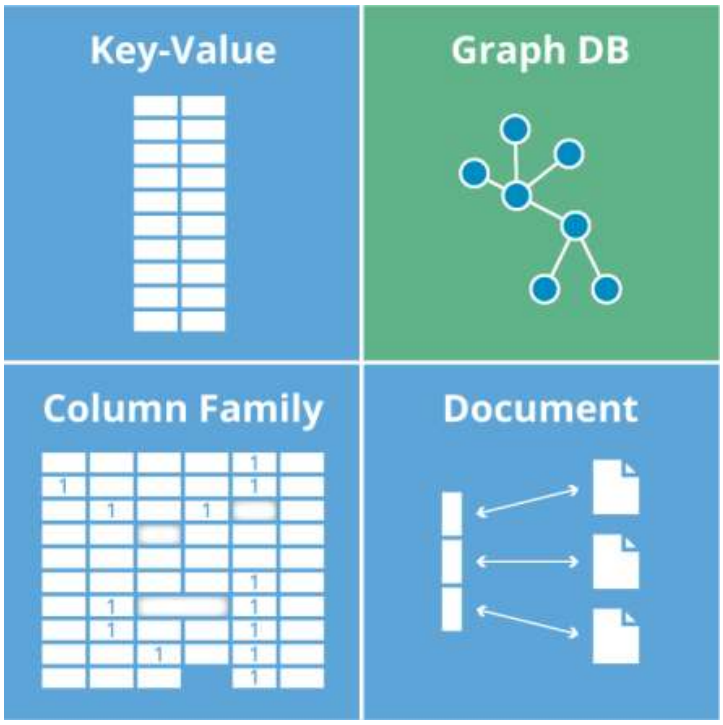
Apabila user ingin menampilkan data-data yang diinginkan saja, maka dapat dilakukan dengan desain bahasa query tertentu. Tabel-tabel dalam database SQL merupakan sekumpulan tabel yang solid dan *fixed*. Karena itu, sedikit saja perubahan struktur pada satu tabel bisa jadi dapat mengakibatkan kegagalan query yang sudah terprogram di bagian View ataupun Trigger. Kendala yang dihadapi pada database SQL terletak pada kompleksitas *maintain* (perawatan sistem) serta pengembangan skala datanya.

2. NoSQL

Sebuah konsep serta model basis data yang fleksibel. Secara general maupun spesifik NoSQL tidak mengikuti kaidah-kaidah database relasional (RDBMS). NoSQL tidak pula menggunakan bahasa query SQL. NoSQL adalah sebuah model database yang berbeda dibandingkan dengan SQL.

Database NoSQL dikembangkan sebagai harapan untuk menyelesaikan masalah-masalah klasik pada database SQL. Database NoSQL berbeda dengan database SQL yang metode penyimpanan

datanya hanya satu tipe saja. Struktur database NoSQL lebih dinamis dan fleksibel dengan empat jenis model penyimpanan data, dapat digambarkan sebagai berikut,



Gambar 1. Model Penyimpanan Data NoSQL

a. **Key-value database**

Jenis *database* ini berisi *key* dan *value* sebagai tempat akses data. Sebuah *value* atau *nilai* biasanya hanya diambil dengan mereferensikan dari *key* atau kuncinya. Setiap atribut unik akan disimpan sebagai kunci (*key*) dengan nilai (*value*) terkait. Nilai (*value*) dapat berupa string, array, struktur data dan lainnya

Contoh :

Key	Value
K1	AAA,BBB,CCC
K2	AAA,BBB
K3	AAA,DDD
K4	AAA,2,01/01/2015
K5	3,ZZZ,5623

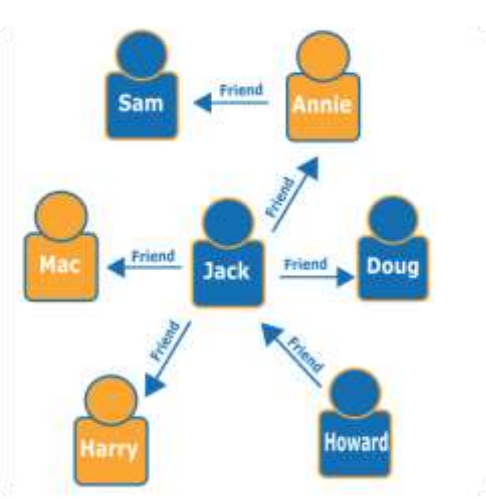
b. Graph database

Graph database menyimpan data dalam node dan edge. Node biasanya menyimpan informasi tentang orang, tempat, dan benda-benda.

Sementara itu, edge menyimpan informasi tentang hubungan antar node.

Jenis *database* yang satu ini lebih unggul dalam penggunaan di mana pengguna perlu mencari tahu hubungan atau pola.

Contoh :



c. Column-based database

Column-based database memberikan banyak fleksibilitas daripada *database* yang relasional karena setiap baris tidak diharuskan memiliki kolom yang sama. Setiap kolom dibuat secara terpisah dan nilai dalam *database* kolom tunggal disimpan secara berdekatan.

Jenis *database* ini memberikan kinerja tinggi pada *aggregation queries* seperti SUM, Count, AVG, hingga MIN karena datanya sudah tersedia di kolom.

Contoh :

ColumnFamily			
Row Key	Nama Column		
	Key	Key	Key
	Value	Value	Value
	Nama Column		
	Key	Key	Key
	Value	Value	Value

d. Document database

Dalam *document database* menyimpan data dalam dokumen yang mirip dengan objek *JSON (JavaScript Object Notation).* Konsep dari *document database* ini lebih efisien dan fleksibel. Jadi, program akan lebih mudah dikembangkan karena *document database* akan menyesuaikan penyimpanan data berdasarkan kebutuhan program.

Jenis database ini sangat cocok digunakan untuk *database* yang bertujuan umum. Selain itu, *document database* juga mampu mengakomodasi volume data yang besar.

Contoh :

Relational SQL

Kolom1	Kolom2	Kolom3
Data	Data	Data
Data	Data	Data
Data	Data	Data

Document-oriented NoSQL

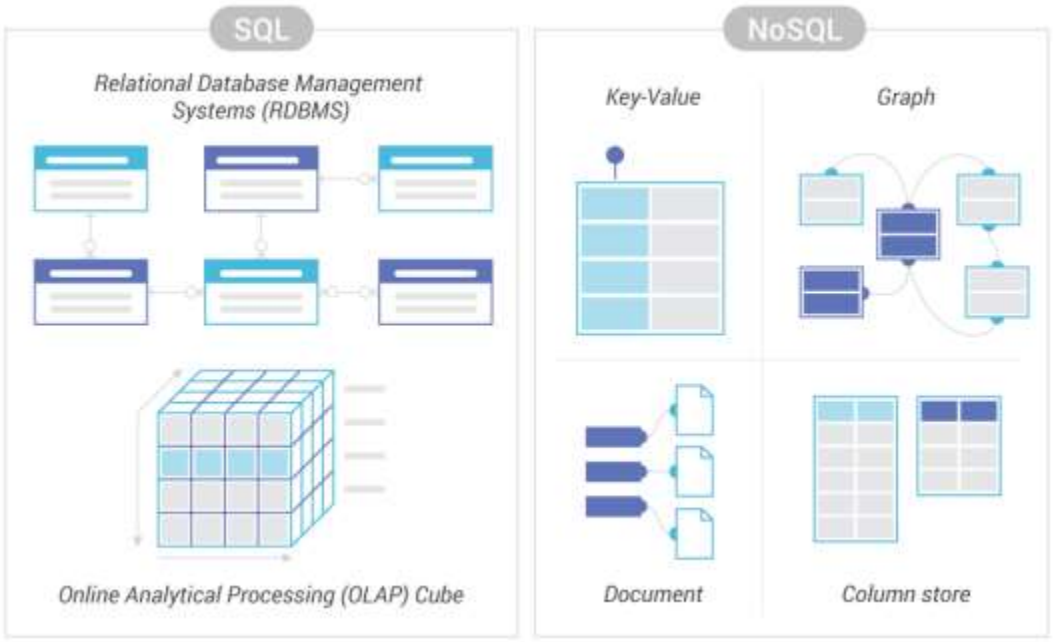
Document 1	Document 2	Document 3
{ "prop1":data, "prop2":data, "prop3":data, "prop4":data, }	{ "prop1":data, "prop2":data, "prop3":data, "prop4":data, }	{ "prop1":data, "prop2":data, "prop3":data, "prop4":data, }

```
[
    {
        "dibuat" : 01-02-2020,
        "judul" : "Cara membuat program seminggu",
        "info" : {
            "penulis" : "faqih",
            "dilihat" : 112,
            "genres" : "programming"
        }
    },
    {
        "dibuat" : 21-02-2020,
        "judul" : "Tutorial membuat web scraping",
        "info" : {
            "penulis" : "faqih",
            "dilihat" : 112,
            "share" : 10"
        }
    }
]
```

3. Perbedaan NoSQL dengan SQL

Fungsi database NoSQL sama dengan fungsi database SQL klasik, Seperti RDBMS SQL pada umumnya, NoSQL berguna untuk mengolah data-data dan menyimpannya untuk dimanfaatkan kembali atau diakses baik oleh Server maupun Client. Data-data tersebut tersimpan secara konsisten sehingga baik dari Server maupun Client melihat data yang sama. NoSQL menganut kaidah BASE, yaitu *Basically Available, Soft state,* dan *Eventual consistency*. Data di dalamnya dapat berubah meski tidak ada input, ia seperti RDBMS dengan *Trigger* yang terpasang secara otomatis. Karena itu NoSQL lebih cocok dan sesuai digunakan untuk aplikasi yang memerlukan perubahan yang *rapidly* (cepat) karena sifatnya yang lebih dinamis dan fleksibel.

Perbedaan SQL dan NoSQL dapat digambarkan sebagai berikut,



Gambar 2. Skema Perbedaan SQL dan NoSQL

4. Kelebihan database NoSQL dibandingkan database SQL

NoSQL dinilai dapat bekerja lebih baik daripada SQL yang berbasis relasional, diantaranya sebagai berikut.

1. Ketika perlu menyimpan data dalam jumlah yang besar dengan skema yang tidak konsisten. Skema data pada NoSQL tidak *fixed* seperti halnya pada SQL, perubahan struktur dan skema yang sewaktu-waktu bisa berubah bisa ditangani dengan mudah tanpa harus merubah konsistensi data di database itu sendiri.
2. Ketika memerlukan komputasi dan penyimpanan data secara *cloud-based*. Sebagian besar database NoSQL dibuat dan dirancang agar bisa bekerja di *data-center* yang berbeda dan dijalankan sebagai sistem terdistribusi. Dalam kasus ini sebagai pegguna NoSQL akan diuntungkan, karena dapat memanfaatkan akses ke berbagai infrastruktur komputasi berbasis cloud.
3. Ketika menjadi *web-developer* atau *app-developer* dan memerlukan *update* yang cepat (*rapidly*). Dengan NoSQL tidak perlu mempersiapkan data sebagaimana apabila menggunakan RDBMS (SQL), bahkan dapat memigrasikan data yang sudah terstruktur dari satu versi aplikasi ke versi update setelahnya, kapan saja desain aplikasinya di-update. Sifat dinamis NoSQL dapat ikut berkembang bersama dengan perubahan aplikasinya.