BAB II

TINJAUAN PUSTAKA

Pada bab ini akan membahas tentang tinjauan pustaka yang melandasi penelitian yang diangkat dalam penyusunan tesis ini. Tinjauan pustaka yang digunakan meliputi penelitian sebelumnya, konsep ritel, definisi informasi, data, basis data, RDBMS, NoSQL, konsep transaksional dan teorema CAP.

2.1 Kajian Pustaka

Pada penelitian sebelumnya yang meneliti berkaitan basis data RDBMS dan basis data NoSQL. Berikut penelitian sebelumnya yang berkaitan dengan basis data RDBMS dan NoSQL diantaranya:

1. Pada tahun 2011 Carlos André Reis Fernandes Oliveira da Silva dari Universitas Do Porto melakukan penelitian dengan judul RDBMS vs NoSQL: Performance and Scaling Comparison. Penelitian yang dilakukan hanya berupa analisa dari teori dengan membuat pola desain prototipe lalu membandingkan kinerja basis data RDBMS MySQL dengan NoSQL MongoDB [3].
2. Pada tahun 2011 Robin Henricsson dari Universitas Blekinge Institute of Technology menguji kecepatan tulis dan baca juga efisiensi penyimpanan dari dua database NoSQL yang berorientasi dokumen yaitu MongoDB dan CouchDB, kedua database tersebut menggunakan JSON untuk menyimpan data. Hasil pengujian menunjukkan MongoDB sedikit lebih cepat dari CouchDB, berdasarkan dari hasil *query* tulis dan baca dengan

menggunakan bahasa pemrograman python dan pustakanya. MongoDB juga lebih efisien dari segi penyimpanan dibanding CouchDB. Total data maksimum yang diuji 500.000 dokumen dan minimum 5.000 dokumen [4].

3. Pada tahun 2012 Petter Nasholm dari Chalmers University of Technology mengkaji bagaimana aplikasi Spotfire berbasis data tabular yang umumunya menggunakan database RDBMS dapat diterapkan juga menggunakan database NoSQL. Untuk keperluan analisa dan visualisasi data dalam mencari dan menggali trends yang bisa digunakan sebagai input perusahaan dalam membuat keputusan bisnis. Hasil akhir dari pengujian hanya bersifat dasar dan bisa dijadikan sebagai fondasi atau landasan untuk penelitian lebih jauh [5].
4. Pada tahun 2013 Christoforos Hadjigeorgiou dari University of Edinburgh melakukan pengujian perbandingan kinerja dan penskalaan basis data RDBMS MySQL dengan NoSQL MongoDB. Dari hasil pengujian ditemukan bahwa MongoDB memiliki kinerja yang lebih baik untuk query yang kompleks tetapi menghasilkan data duplikat dan ukuran basis data menjadi membesar. Ukuran basis data tidak menjadi faktor penentu dari turunnya kinerja basis data secara signifikan untuk ukuran basis data yang lebih besar [6].

2.2 Konsep Ritel (Pengecer)

Kata ritel berasal dari kata Perancis yaitu pengecer, yang berarti untuk memotong sepotong atau untuk memecah massal. Dalam istilah sederhana, ini menyiratkan transaksi langsung dengan pelanggan. Ritel dapat didefinisikan sebagai pembelian dan penjualan barang dan jasa. Ritel adalah salah satu industri terbesar di dunia. Ini berada dalam kondisi perubahan permanen, dan laju perubahan ini telah meningkat selama dekade terakhir. Dari perspektif pemasaran, peritel, menurut definisi, lebih dekat ke konsumen daripada perusahaan manufaktur [7].

Pemasaran ritel secara konsisten menampilkan praktik pemasaran yang lebih efisien, lebih bermakna, dan lebih menguntungkan [8]. Berikut beberapa terminologi sebelum memahami konsep ritel [9].

1. Pasar: Setiap sistem atau tempat di mana para pihak terlibat dalam pertukaran barang atau jasa disebut pasar. Pihak-pihak tersebut sering disebut sebagai pembeli dan penjual. Penjual menawarkan barang atau jasanya kepada pembeli yang sebagai gantinya membelinya dalam pertukaran uang.
2. Barang: Benda nyata (hal-hal yang dapat dilihat dan disentuh) produk fisik yang ditransfer dari penjual ke pembeli (konsumen) untuk memenuhi kebutuhan yang terakhir disebut sebagai barang.

2.3 Penerapan TI dalam Operasi Ritel

Berikut beberapa peran teknologi yang dapat diterapkan di ritel.

1. Perkiraan Permintaan (*Demand Forecasting*)

Peramalan adalah proses estimasi dalam situasi yang tidak diketahui. Ini adalah proses yang esensial dan sangat penting dalam perusahaan bisnis apa pun. Para pemimpin bisnis dan ekonom terus terlibat dalam proses mencoba meramalkan atau memprediksi, masa depan bisnis dalam ekonomi. Para pemimpin bisnis terlibat dalam proses ini karena banyak dari apa yang terjadi dalam bisnis saat ini bergantung pada apa yang akan terjadi di masa depan [9].

Sistem peramalan permintaan modern memberikan peluang baru untuk meningkatkan kinerja ritel. Sistem skala besar saat ini mampu menangani massa data transaksi ritel - mengelolanya, menambangnya, dan memproyeksikannya ke dalam perilaku pelanggan di masa depan. Sistem peramalan permintaan di ritel akan berkontribusi pada keakuratan rencana masa depan, kepuasan pelanggan masa depan, efisiensi keseluruhan dan profitabilitas operasi perusahaan [9].

2. Manajemen Persediaan (*Inventory Management*)

Persediaan dapat berupa bahan baku, barang jadi yang sudah tersedia untuk dijual, atau barang dalam proses pembuatan. Persediaan dicatat sebagai aset dalam neraca perusahaan. Untuk mengoptimalkan penyebaran inventaris, pengecer perlu mengelola ketidakpastian, kendala, dan kompleksitas di seluruh rantai pasokan global mereka secara berkelanjutan. Ini memungkinkan mereka untuk meningkatkan kemampuan perkiraan persediaan dan secara akurat menetapkan target persediaan [9].

Solusi TI adalah solusi yang terbukti dan memimpin pasar untuk menentukan target persediaan barang yang bervariasi berdasarkan waktu untuk setiap item, di setiap lokasi di sepanjang rantai pasokan. Ini memungkinkan para

pengecer untuk mengurangi persediaan secara signifikan tanpa mempengaruhi tingkat layanan [9].

3. Manajemen Toko (*Store Management*)

Contoh lain dimana teknologi informasi dapat bermanfaat adalah manajemen toko yang dapat memberikan informasi barang tidak pada tempatnya atau persediaan habis. Biasanya toko atau kios untuk penjualan eceran komoditas, tetapi juga sekaligus tempat menyimpan persediaan grosir, ditampilkan (*display*), atau dijual. Menggunakan perangkat untuk memantau lokasi produk yang sebenarnya versus lokasi yang diinginkan di lantai atau di ruang stok. Sistem keuangan dan pembayaran berbasis sistem yaitu POS, Point of Sales sebagai perangkat untuk memudahkan transaksi pembayaran dengan konsumen dan menjadi penyimpanan data transaksi [9].

2.4 Definisi Informasi

Informasi adalah sekumpulan data atau fakta yang telah diproses dan dikelola sedemikian rupa sehingga menjadi sesuatu yang mudah dimengerti dan bermanfaat bagi penerimanya. Dari definisi tersebut dapat kita pahami bahwa kata “informasi” memiliki arti yang berbeda dengan kata “data”. Data adalah fakta yang masih bersifat mentah atau belum diolah, setelah mengalami proses atau diolah maka data itu bisa menjadi suatu informasi yang bermanfaat [10].

Tidak semua data atau fakta dapat diolah menjadi sebuah informasi bagi penerimanya. Jika suatu data yang diolah ternyata tidak bermanfaat bagi penerimanya, maka hal tersebut belum bisa disebut sebagai sebuah informasi [10].

Secara etimologis istilah “informasi” berasal dari bahasa Latin, yaitu “Informatinem” yang artinya ide, kode, atau garis besar. Informasi dapat disajikan dalam beragam bentuk, mulai dari tulisan, gambar, tabel, diagram, audio, video, dan lain sebagainya [11]. Berikut pengertian informasi menurut para ahli:

1. Data yang telah diolah menjadi bentuk yang mempunyai arti bagi si penerima dan bermanfaat dalam pengambilan keputusan saatu ini atau di masa mendatang [10].
2. Hasil dari pengolahan data ke dalam bentuk yang lebih bermanfaat bagi penerimanya yang menggambarkan kejadian-kejadian yang nyata untuk digunakan dalam pengambilan keputusan [11].
3. Data yang disajikan dalam bentuk yang lebih berguna untuk mengambil suatu keputusan [12].
4. Suatu data penting yang memberikan pengetahuan yang berguna bagi penerimanya [13].
5. Suatu hasil pengolahan data yang memberikan arti dan manfaat bagi penerimanya [14].

Berikut ciri ciri informasi yang berkualitas [10][14], yaitu:

1. Akurat, artinya informasi mencerminkan keadaan sebenarnya.
2. Tepat waktu, artinya informasi harus ada saat diperlukan.
3. Relevan, informasi yang diberikan harus sesuai yang dibutuhkan.
4. Lengkap, artinya informasi harus utuh, tidak setengah-setengah.

2.5 Definisi Data

Data adalah sekumpulan keterangan atau fakta mentah berupa simbol, angka, kata-kata, atau citra, yang didapatkan melalui proses pengamatan atau pencarian ke sumber-sumber tertentu [15]. Pendapat lain mengatakan, definisi data adalah kumpulan keterangan-keterangan atau deskripsi dasar dari suatu hal (objek atau kejadian) yang diperoleh dari hasil pengamatan (observasi) dan dapat diolah menjadi bentuk yang lebih kompleks, seperti; informasi, database, atau solusi untuk masalah tertentu [16].

Secara etimologis, istilah "data" berasal dari bahasa Latin, yaitu "Datum" yang artinya sesuatu yang diberikan. Dengan kata lain, data merupakan hasil pengukuran atau pengamatan suatu variabel yang bentuknya dapat berupa simbol, warna, kata-kata, angka, atau citra [17].

2.5.1 Jenis-Jenis Data

Jenis-jenis data dapat dikelompokkan berdasarkan sumbernya, sifatnya, cara memperolehnya dan waktu pengumpulannya. Data dapat diklasifikasikan sebagai berikut [17] :

1. Data Berdasarkan Cara Memperolehnya

Data Primer, yaitu data asli atau data baru yang dikumpulkan langsung oleh orang yang melakukan penelitian.

Data Sekunder, yaitu data tersedia yang dikumpulkan dari berbagai sumber yang sudah ada sebelumnya. Misalnya; dari perpustakaan, dokumen penelitian terdahulu, dan lain-lain.

2. Jenis Data Berdasarkan Sumbernya

Data Internal, yaitu data yang didapatkan dari internal suatu perusahaan yang menggambarkan keadaan perusahaan tersebut. Misalnya; informasi jumlah pegawai, jumlah modal, jumlah produksi, dan sebagainya.

Data Eksternal, yaitu data yang diperoleh dari luar perusahaan yang menggambarkan berbagai faktor yang dapat mempengaruhi kinerja perusahaan tersebut. Misalnya; informasi tentang daya beli masyarakat, perubahan kebiasaan masyarakat, dan lain sebagainya.

3. Jenis Data Berdasarkan Sifatnya

Data Kualitatif yaitu suatu data yang dinyatakan dalam bentuk verbal, simbol atau gambar. Misalnya kuesioner mengenai tingkat kepuasan konsumen terhadap pelayanan suatu perusahaan.

Data Kuantitatif yaitu suatu data yang dinyatakan dalam bentuk angka atau bilangan. Misalnya harga saham, nilai pendapatan, dan lain-lain.

4. Data Berdasarkan Waktu Pengumpulannya

Data *Cross Section* yaitu data yang dikumpulkan hanya pada waktu-waktu tertentu saja untuk mengetahui keadaan pada waktu tersebut. Misalnya data penelitian dengan kuesioner.

Data Berkala yaitu data yang dikumpulkan secara berkala dari waktu ke waktu untuk mengetahui perkembangan suatu kejadian pada periode tertentu. Misalnya data harga elektronik.

2.6 Sistem Manajemen Basis Data (DBMS)

Database-management system (DBMS) adalah kumpulan data yang saling terkait dan satu set program untuk mengakses data tersebut. Pengumpulan data, biasanya disebut sebagai basis data, berisi informasi yang relevan dengan suatu perusahaan. Tujuan utama dari DBMS adalah menyediakan cara untuk menyimpan dan mengambil informasi basis data yang ada nyaman dan efisien [18].

Sistem basis data dirancang untuk mengelola banyak informasi. Manajemen data melibatkan kedua struktur pendefinisian untuk penyimpanan informasi dan menyediakan mekanisme untuk manipulasi informasi. Selain itu sistem database harus memastikan keamanan informasi yang disimpan, terlepas dari kegagalan sistem atau upaya akses yang tidak sah. Jika data akan dibagikan di antara beberapa pengguna, sistem harus menghindari kemungkinan hasil yang tidak normal atau tidak sesuai. Karena informasi sangat penting di sebagian besar organisasi, perusahaan dan para ilmuwan komputer telah mengembangkan banyak konsep dan teknik untuk mengelola data [18].

2.6.1 Basis Data Relasional (RDBMS)

Dalam sistem komputasi (web dan aplikasi bisnis), ada data yang besar yang keluar setiap hari dari aplikasi. Sebuah bagian besar dari data tersebut ditangani oleh sistem manajemen basis data relasional (RDBMS). Ide model relasional datang dari E.F.Codd’s pada tahun 1970 dalam sebuah paper "*Sebuah model relasional data untuk besar bersama bank data*" yang membuat pemodelan data dan pemrograman aplikasi lebih mudah. Model relasional sesuai untuk

pemrograman client-server dan hari ini yang merupakan teknologi dominan untuk menyimpan data terstruktur di web dan aplikasi bisnis [19].

2.6.2 Basis Data NoSQL

NoSQL adalah sekumpulan konsep yang memungkinkan pengolahan cepat dan efisien dari sekumpulan data dengan fokus pada kinerja, kehandalan, dan kelincahan [20]. Dalam konsep RDBMS penambahan *field* atau kolom merupakan tantangan tersendiri dan harus dihindari sekecil mungkin namun dalam prakteknya sulit untuk dapat menghindari hal tersebut. Konsep inilah yang mendasari adanya konsep NoSQL. NoSQL tidak membutuhkan skema tabel (*schema-less*) dan umumnya menghindari operasi join dan berkembang secara horizontal. Akademisi menyebut basis data seperti ini sebagai *structured storage*. Selain itu Konsep ini didasarkan pemikiran, kedepan basis data dapat diambil dari mana saja dan penempatannya akan terpencar-pencar, sehingga suatu data akan dapat mengambil master dari tempat lain [20].

Nama NoSQL menunjukkan bahwa basis data ini tidak mendukung bahasa SQL *query*. Namun, tidak membahas karakteristik utama mereka yang adalah kenyataan bahwa mereka adalah non-relasional [30]. Nama harus ditafsirkan sebagai “tidak hanya sql”, yang menunjukkan perubahan dalam mentalitas strategi saat ini bekerja bahwa satu jenis basis data berguna untuk memecahkan segala macam masalah. Dalam 25 tahun terakhir pembangunan DBMS komersial dapat diringkas dalam satu ungkapan yaitu satu ukuran cocok untuk semua [28] Dengan demikian, peralihan ini mendorong terciptanya berbagai jenis basis data,

masing-masing dengan satu set spesifik karakteristik dan berlaku untuk domain masalah yang berbeda.

Peralihan NoSQL dimulai pada awal 2009 dan telah berkembang pesat [30]. Peralihan ini merupakan generasi baru dari basis data non-relasional yang berbagi beberapa karakteristik seperti: fokus pada distribusi sistem, skalabilitas (terutama horizontal) representasi skema-bebas non-relasional data. Sebagian besar basis data NoSQL menawarkan rentang yang lebih terbatas sifat dari sistem RDBMS saat ini, mudah saat kendala konsistensi dan memang tidak mendukung sifat ACID penuh. Pengurangan kemampuan relatif ini memberikan dorongan sistem dalam hal kinerja, sementara pada saat yang sama memfasilitasi distribusi.

Penerapan NoSQL menggunakan berbagai variasi tipe penyimpanan data (*database*). Dari penyimpanan kunci-nilai sederhana yang mengaitkan kunci unik dengan nilai, untuk penyimpanan dalam bentuk grafik dengan cara mengasosiasikan relasi, untuk penyimpanan berbentuk dokumen berupa variabel data, setiap tipe penyimpanan data NoSQL memiliki atribut yang unik dan digunakan seperti yang diidentifikasi dalam tabel IV-1.

Tabel IV-1 Empat Kategori dan Jenis penyimpanan data NoSQL

Type	Typical usage	Examples
Key-value store—A simple data storage system that uses a key to	•Image stores •Key-based filesystems •Object cache	• Berkeley DB • Memcache •Redis

access a value	•Systems designed to scale	•Riak •DynamoDB
Column family store—A sparse matrix system that uses a row and a column as keys	•Web crawler results •Big data problems that can relax consistency rules	•Apache HBase •Apache Cassandra •Hypertable •Apache Accumulo
Graph store—For relationship-intensive problems	•Social networks •Fraud detection •Relationship-heavy data	•Neo4j •AllegroGraph •Big Data (RDF data store) •InfiniteGraph (Objectivity)
Document store—Storing hierarchical data structures directly in the database	•High-variability data •Document search •Integration hubs •Web content management •Publishing	•MongoDB (10Gen) •CouchDB •Couchbase •MarkLogic •eXist-db

2.7 Konsep Transaksional Basis Data

Transaksi adalah unit eksekusi program yang mengakses dan mungkin memperbarui berbagai item data. Biasanya, sebuah transaksi diprakarsai oleh program pengguna yang ditulis dalam bahasa manipulasi data tingkat tinggi (biasanya SQL), atau bahasa pemrograman (misalnya, C++, atau Java), dengan akses basis data tertanam di JDBC atau ODBC. Sebuah transaksi dibatasi oleh pernyataan atau panggilan fungsi dari sebuah form untuk memulai transaksi dan mengakhiri transaksi. Transaksi terdiri dari semua operasi dieksekusi antara transaksi awal dan akhir transaksi [23].

2.7.1 Konsep ACID (RDBMS)

ACID adalah akronim untuk dasar dari Atomicity, Consistency, Isolation dan Durability. Basis data relasional umumnya mampu menyediakan dan mematuhi properti ACID yang merupakan kumpulan properti yang menjamin transaksi basis data diproses dengan andal dan bahwa basis data konsisten dalam menghadapi akses bersamaan dan kegagalan sistem, berikut penjelasannya [23].

1. **Atomicity**

   Atomicity berarti bahwa transaksi basis data harus mengikuti “semua atau tidak” dari sebuah aturan. Transaksi adalah serangkaian instruksi yang akan dieksekusi (sebagai salah satu), untuk transaksi menjadi “atom”, semua instruksi harus dijalankan, atau jika salah satu (atau lebih) gagal, seluruh transaksi harus gagal, dan basis data harus tetap tidak berubah.

2. **Consistency**

   Konsistensi memastikan hanya data yang valid ditulis ke basis data,

menjamin bahwa jika transaksi berhasil dijalankan basis data diambil dari keadaan yang konsisten untuk baru keadaan konsisten.

3. Isolation

Isolasi mensyaratkan bahwa transaksi dalam proses dan belum berkomitmen harus tetap terisolasi dari transaksi lainnya, oleh karena pelaksanaan transaksi tidak boleh mempengaruhi pelaksanaan transaksi konkuren lainnya.

4. Durability

Daya Tahan memastikan bahwa setiap transaksi yang dilakukan ke basis data adalah permanen dan tidak akan hilang. Sistem basis data harus dapat memulihkan *update* transaksi berkomitmen terhadap segala jenis kegagalan sistem (*hardware* atau *software*).

2.7.2 Konsep BASE (NoSQL)

BASE adalah akronim untuk dasar dari *Basically Available, Soft state, Eventual consistency* [24]. BASE adalah kebalikan logis dari ACID [31]. Sementara ACID lebih memfokuskan konsistensi yang kuat dengan menjadi pesimis, BASE mengambil ketersediaan seperti itu tujuan utama yaitu optimis dan memberikan akhir yang konsistensi. Hal ini secara intrinsik terhubung dengan CAP teorema yang pernah diusulkan oleh Eric Brewer pada tahun 2000 pada Simposium Prinsip Komputasi Terdistribusi [24].

Dengan mempertimbangkan teorema CAP, konsistensi akhirnya memainkan peran penting pada basis data NoSQL karena memberikan basis data

yang berkemampuan untuk konsistensi dan efektifitas transaksi untuk ketersediaan atau toleransi partisi [25].

2.8 Teorema CAP (Brewer)

Teorema CAP juga dikenal sebagai teorema Brewer yang merupakan teorema pada sistem terdistribusi yang menyatakan bahwa sifat berikut diharapkan dalam sistem terdistribusi: konsistensi, pada tahapan ketersediaan partisi dan toleransi tetapi sangat kecil kemungkinan untuk mencapai kesemua tiga sifat itu sekaligus. Oleh karena itu, paling banyak hanya dua sifat ini yang dapat dipenuhi [24].

1. ***Consistency***

   Dalam konteks sistem terdistribusi, konsistensi berarti bahwa semua node sistem melihat data yang sama pada waktu yang sama [26].

2. ***Availability***

   Ketersediaan berarti bahwa sistem ini memastikan untuk tetap beroperasi selama periode waktu. Dalam konteks ini, itu merupakan sistem yang terjadi kegagalan di beberapa *node* tidak mencegah *node* yang lain untuk tetap beroperasi [26].

3. ***Partition Tolerance***

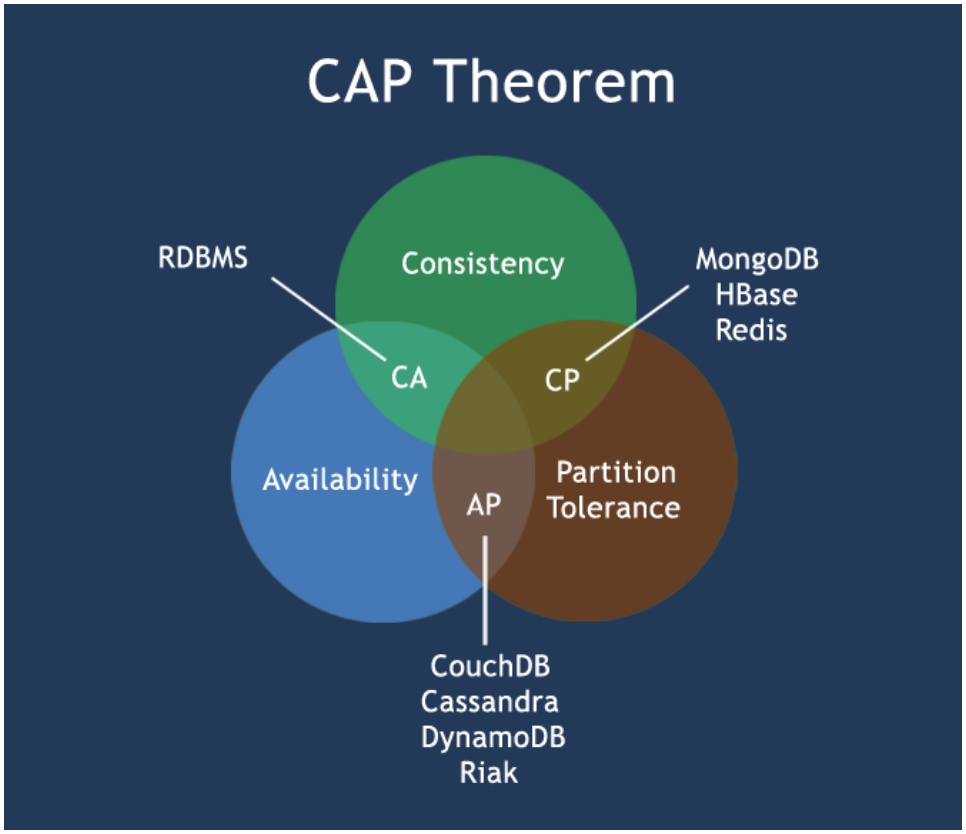
   Partisi Toleransi berarti bahwa sistem harus terus beroperasi meskipun kehilangan pesan perintah antara *node* yang berbeda dari sistem terdistribusi [26].

Berikut adalah deskripsi singkat dari tiga kombinasi CA, CP, AP [26]:

CA - situs klaster *Single*, oleh karena itu semua node selalu bersinggungan. Ketika partisi terjadi, blok sistem.

CP - Beberapa data tidak dapat diakses, tapi sisanya masih konsisten / akurat.

AP - Sistem masih tersedia di bawah partisi, tetapi beberapa data yang dikembalikan mungkin tidak akurat.



Gambar IV-1 Teorema CAP (teorema Brewer)

Sumber: http://www.w3resource.com

2.9 Multiversion Concurrency Control

Multiversion concurrency control (MVCC) adalah mekanisme kontrol konkurensi yang memungkinkan akses bersamaan ke sumber daya tertentu misalnya basis data. Pertama kali dijelaskan pada 1978 disertasi oleh David Reed [27] dan sejak saat itu telah banyak diterapkan pada basis data relasional, seperti, PostgreSQL [32], MySQL, Oracle [33] dan juga pada beberapa basis data NoSQL.

Multi-versi kontrol konkurensi adalah sebuah alternatif untuk mengunci dan mampu memberikan akses paralel efisien untuk data yang menghindari kemungkinan korupsi data dan kebuntuan. Mengunci karya dengan memberikan akses eksklusif ke *record* tertentu untuk proses meminta. Sampai proses yang melepaskan kunci, setiap proses lain mencoba untuk mengakses record (bahkan untuk membaca) harus menunggu. Model ini secara efektif *serializes* permintaan dan memberikan akses eksklusif kepada pihak yang meminta. Hal ini dapat dilihat sebagai sikap pesimis pada *concurrency*.

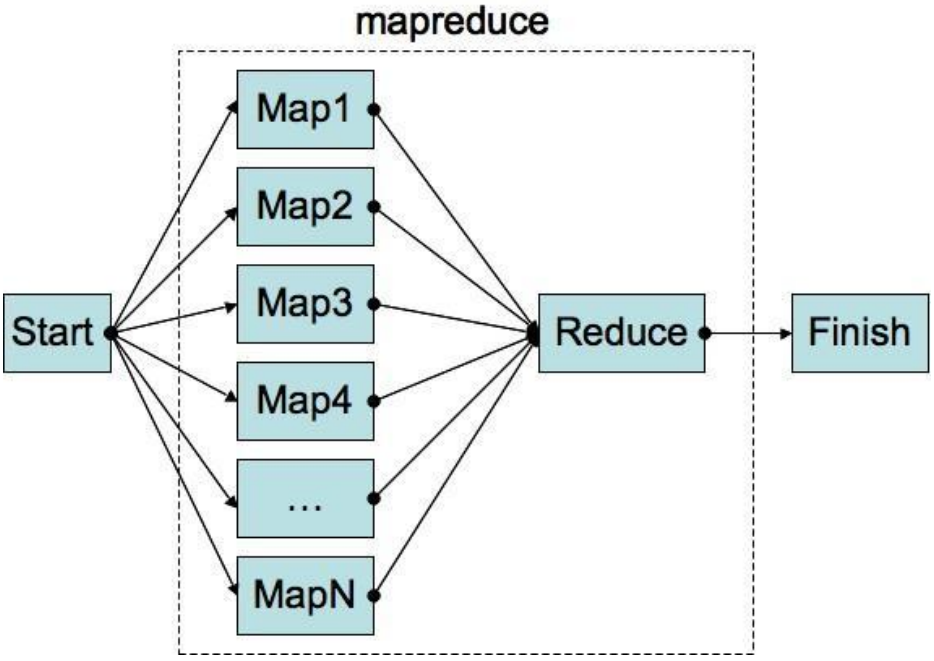
Untuk mencapai konsistensi, setiap item data yang melekat padanya cap waktu (atau nomor revisi). Setiap kali proses membaca sumber daya, itu mengambil nilainya serta cap ini. Jika itu sama proses ingin memperbarui catatan ini, setiap kali mencoba untuk menulis ke database nilai baru dikirim serta cap asli. Jika timestamp revisi terbaru adalah sama, maka nilai baru ini ditulis dan timestamp diperbarui, efektif membuat salinan modifikasi dari data asli.

Pada beberapa sistem database (misalnya CouchDB) MVCC juga memungkinkan untuk melakukan optimasi akses disk. Karena banyak update

berarti bahwa salinan baru dari item data benar-benar diciptakan memungkinkan untuk menulis seluruh item data ke bagian yang berdekatan dari disk [29]. Meskipun teknik ini menawarkan beberapa keunggulan dibandingkan penguncian, ia datang dengan kelemahan karena harus menyimpan beberapa versi item data, sehingga meningkatkan ukuran basis data.

2.10 MapReduce

MapReduce adalah model pemrograman dan implementasi terkait untuk diproses dan menghasilkan set data yang besar. Dikembangkan oleh Google untuk mendukung sistem pengindeksan mereka [30]. Meskipun *MapReduce* pertama kali digunakan sebagai perangkat lunak berpemilik, oleh Google, saat ini ada implementasi berbasis terbuka (*open source*) seperti Apache Hadoop [31] . Hal ini juga tersedia dan digunakan secara luas di banyak basis data NoSQL. Sebagai basis data NoSQL cenderung didistribusikan dan biasanya berurusan dengan set data yang sangat besar , pengolahan data dan operasi agregasi (SQL COUNT, SUM, dll) biasanya diimplementasikan sebagai operasi *MapReduce*, sehingga bisa merata mendistribusikan setiap pekerjaan di seluruh sistem, juga akan memberikan kerangka yang mudah bagi pengembang untuk menangani.



Gambar IV-2 Teknik MapReduce

Sumber: http://www.datasciencecentral.com

Pada Gambar 2.8.1 [36] representasi dari fase MapReduce dan distribusi kerja disajikan. Untuk dapat menggunakan kerangka MapReduce, pengguna hanya harus memberikan peta yang tepat dan mengurangi fungsi yang akan digunakan dalam setiap langkah. Fungsi peta mengambil pasangan input dan menghasilkan satu set antara kunci / nilai pasangan. Mengurangi fungsi menerima kunci intermediate dan satu set nilai untuk kunci itu. Ini menggabungkan nilai-nilai ini bersama-sama untuk membentuk satu set mungkin lebih kecil dari nilai-nilai, biasanya hanya nol atau satu nilai output yang dihasilkan [30].

```
1 map(String key, String value):
2 // key: document name
3 // value: document contents
4 for each word w in value:
```

```
5 EmitIntermediate(w, "1");
6
7 reduce(String key, Iterator values):
8 // key: a word
9 // values: a list of counts
10 int result = 0;
11 for each v in values:
12 result += ParseInt(v);
13
14 Emit(AsString(result))
```

Listing II-1: Contoh map dan reduce fungsi untuk menghitung kata

1. **Map** adalah setiap node pekerja menerapkan fungsi peta ke data lokal, dan menulis output ke penyimpanan sementara. Node master memastikan bahwa hanya satu salinan dari data input yang berlebihan yang diproses.
2. **Reduce** adalah node pekerja saat memproses setiap kelompok data keluaran, per kunci, secara paralel.

The MapReduce, makalah yang diterbitkan oleh Google menyajikan contoh sederhana tentang penggunaan kerangka MapReduce yang membantu memahami konsep-konsep kunci [30]. Untuk menghitung jumlah kejadian dari setiap kata dalam satu set yang sangat besar dokumen, dapat berguna dengan mengandalkan teknik MapReduce untuk mendistribusikan tugas. Untuk melakukannya, perlu menentukan peta (*map*) dan mengurangi (*reduce*) fungsi.

Fungsi peta menentukan pasangan kunci atau nilai untuk setiap kata yang ditemukan dalam dokumen tertentu. Kuncinya merupakan kata dan nilai merupakan jumlah kejadian (1 dalam kasus). Mengurangi fungsi lalu merangkum bersama-sama semua nilai (*value*) yang dikeluarkan untuk kata tertentu (*key*) dan memunculkan hasil akhir.

Meskipun MapReduce pertama kali diterapkan sebagai perangkat lunak yang dimiliki dan dikembangkan oleh Google, saat ini sudah ada implementasi berbasis *open-source* seperti Apache Hadoop [31]. Hal ini juga tersedia dan digunakan secara luas di banyak basis data NoSQL. Umumnya basis data NoSQL cenderung didistribusikan dan biasanya berurusan dengan kumpulan data yang sangat besar, pengolahan data dan operasi agregasi (SQL COUNT, SUM, dll) biasanya diimplementasikan sebagai operasi MapReduce, sehingga bisa merata pendistribusian pekerjaan di seluruh sistem dan memberikan kerangka yang mudah untuk ditangani oleh para pengembang (*programmer*).

2.11 Taxonomy NoSQL

Jenis metode yang digunakan NoSQL untuk menyimpan ke dalam *storage* [20] :

1. *Document Oriented*, dibandingkan menggunakan tabel sebagai struktur data yang akan disimpan, NoSQL menggunakan *Document Oriented* sebagai struktur penyimpanannya. Bisa menambahkan *field* dengan panjang value dengan lebih mudah, fleksibel dan tidak terlalu terikat dengan ukuran dari struktur tabel. Contohnya MongoDB, CouchDB.

2. *Key Value Stores* digunakan di NoSQL, Key Values ini berfungsi untuk menyimpan key *unique*, sebagai penanda *index*. Untuk penggunaanya dapat terstruktur dan tidak terstruktur. Contoh SimpleDB, Redis, Memcached.
3. *Table Oriented*, NoSQL menggunakan tabel sebagai cara untuk penyimpanan data. Cara ini digunakan oleh Google Big Table, HBase.
4. *Graph database*, cara terakhir menggunakan konsep *graph* sebagai cara melakukan penyimpanan di dalam basis data NoSQL. Di antara ketiga yang lain, cara *Graph* masih terbilang baru di dalam implementasinya dan digunakan kebanyakan untuk situs media sosial. Contoh Neo4J, OrientDB.

Jumlah database non-relasional yang tersedia relatif besar dan terus meningkat. Pada penelitian ini basis data yang mendekati RDBMS adalah yang berorientasi dokumen.

2.12 Karakteristik NoSQL Berorientasi Dokumen

Berdasarkan taxonomy diatas akan dijelaskan beberapa teknologi NoSQL yang berorientasi dokumen tersedia. Basis data ini dipilih berdasarkan tingkat kematangan, relevansi dan penerapan pada studi kasus yang akan dikembangkan.

Basis data NoSQL yang berorientasi dokumen memiliki karakteristik yang hampir sama dengan basis data relasional. Basis data berorientasi dokumen dirancang untuk menyimpan, mengambil, dan mengelola informasi berorientasi dokumen, juga dikenal sebagai data semi-terstruktur. Berbeda dengan basis data relasional dan gagasan tentang relasi dari tabel, pada basis data NoSQL sistem ini

dirancang berdasarkan abstrak dari dokumen. Berikut Tabel Karakteristik NoSQL *Document Oriented* [37].

Tabel IV-2 Karakteristik NoSQL Document Oriented

Objek dapat disimpan sebagai dokumen	Hanya ada model objek dokumen dan langsung siap digunakan.
Dokumen dapat menjadi kompleks	Model seluruh objek dapat dibaca dan ditulis sekaligus. Tidak perlu melakukan serangkaian pernyataan insert atau membuat store prosedur yang kompleks.
Dokumen yang independen	Meningkatkan kinerja dan mengurangi efek samping concurrency
Format Terbuka	Dokumen dijelaskan menggunakan JSON atau XML atau turunannya. Clean & self-describing.
Tanpa/Bebas Skema	Skema bebas memberikan fleksibilitas untuk sistem berkembang tanpa memaksa data yang ada harus direstrukturisasi.
Built-in Versioning	Kebanyakan database dokumen mendukung versioning dokumen dengan flip dari switch.