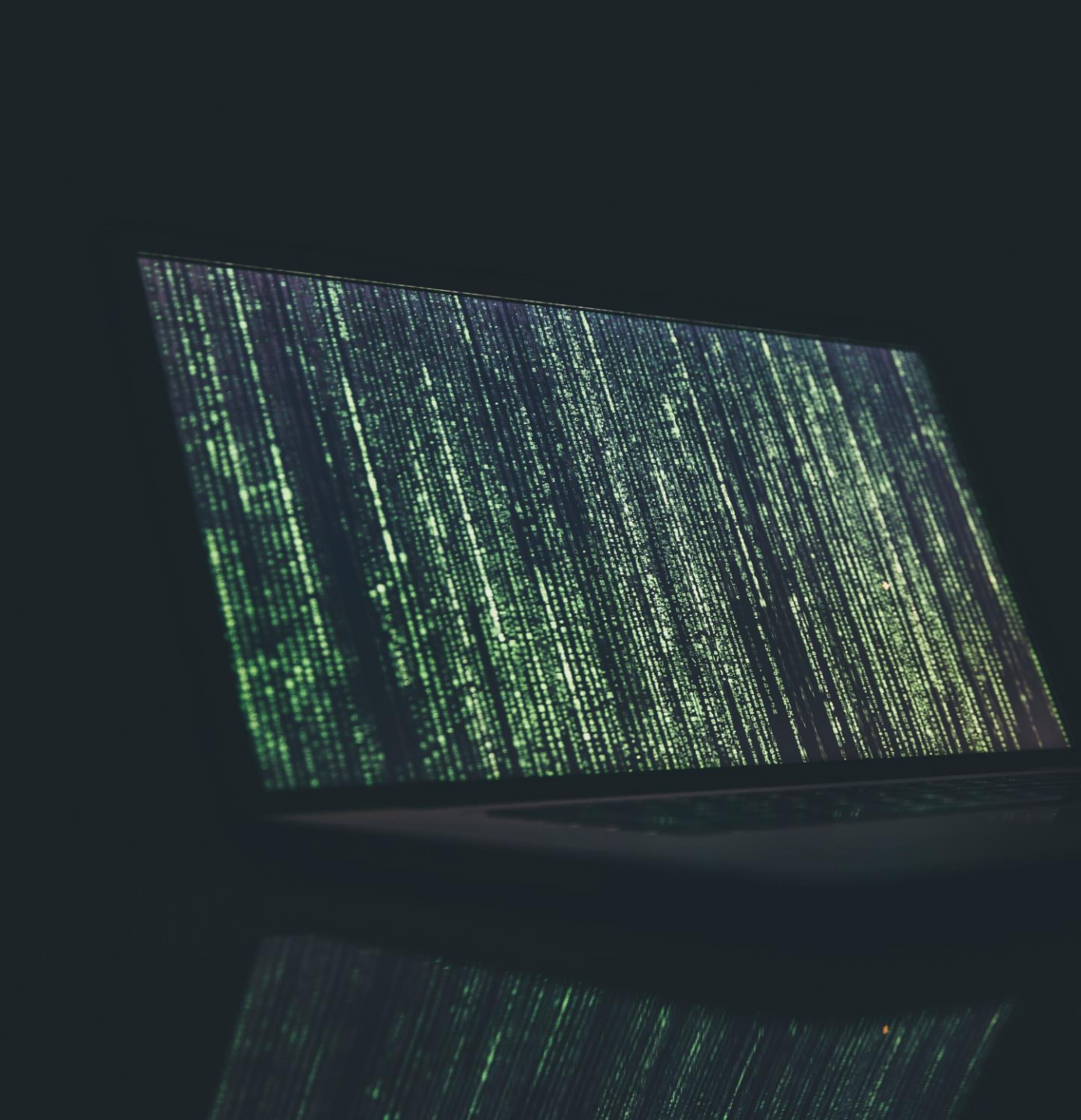


PERTEMUAN 6

TEKNIK PENYIMPANAN BIG TERDISTRIBUSI

Mata Kuliah:

Infrastruktur Dan Teknologi Big Data

Dosen: Galih Hermawan, S.Kom., M.T.

Prodi Teknik Informatika. FTIK.

Universitas Komputer Indonesia

KONSEP

- Teknik penyimpanan Big Data terdistribusi adalah teknik yang membagi dan menyimpan data dalam beberapa *node* atau komputer yang saling terhubung dalam sebuah jaringan.
- Tujuan → meningkatkan kinerja, skalabilitas, dan reliabilitas dalam mengakses dan memproses data yang berukuran sangat besar

ALASAN PENGGUNAAN

- Kebutuhan Skala Besar:
  - Penyimpanan terdistribusi dirancang untuk menangani volume data yang sangat besar, yang tidak mungkin ditangani oleh sistem penyimpanan tradisional.
- Kinerja yang Ditingkatkan:
  - Distribusi data memungkinkan untuk akses dan pemrosesan data secara paralel, meningkatkan kinerja dan mengurangi waktu respon.
- Ketersediaan Tinggi:
  - Dengan replikasi data, penyimpanan terdistribusi meningkatkan ketersediaan data dan mengurangi risiko kehilangan data karena kegagalan perangkat keras atau node.
- Elastisitas:
  - Sistem terdistribusi dapat dengan mudah diubah ukurannya untuk menyesuaikan dengan pertumbuhan atau penurunan kebutuhan penyimpanan.
- Toleransi Kesalahan:
  - Kemampuan untuk mendeteksi dan mengatasi kesalahan tanpa kehilangan data, meningkatkan kehandalan sistem secara keseluruhan.
- Pemrosesan Paralel:
  - Penyimpanan terdistribusi mendukung pemrosesan paralel, memungkinkan eksekusi tugas-tugas secara bersamaan untuk efisiensi maksimal.

CONTOH TEKNOLOGI

- Hadoop Distributed File System (HDFS)
  - Sistem fail terdistribusi yang dirancang untuk menyimpan sejumlah besar data pada klaster perangkat keras komoditas
  - Sangat dapat diskalakan dan tahan terhadap kegagalan
  - Cocok untuk menyimpan kumpulan data besar yang tidak terstruktur
  - Contoh kasus penggunaan: menyimpan data untuk aplikasi web, gudang data, dan pembelajaran mesin
- Apache HBase
  - Basis data terdistribusi yang dirancang untuk menyimpan sejumlah besar data semi-terstruktur
  - Dapat diskalakan dan tahan terhadap kegagalan
  - Cocok untuk menyimpan data yang tidak cocok dengan skema basis data relasional, misalnya model penyimpanan kunci-nilai.
  - Contoh kasus penggunaan: menyimpan data seri waktu, data media sosial, dan data klik aliran

CONTOH TEKNOLOGI (LANJUTAN)

- Amazon Simple Storage Service (S3)
  - Layanan penyimpanan objek yang menyediakan solusi penyimpanan yang dapat diskalakan dan tahan lama untuk berbagai jenis data
  - Sangat dapat diskalakan dan tahan lama
  - Efisien biaya
  - Contoh kasus penggunaan: menyimpan cadangan, pengarsipan data, dan menyajikan konten statis
- Google Cloud Storage
  - Layanan penyimpanan objek yang menyediakan solusi penyimpanan yang dapat diskalakan, tahan lama, dan berkinerja tinggi untuk berbagai jenis data
  - Sangat dapat diskalakan, tahan lama, dan berkinerja tinggi
  - Terintegrasi dengan layanan Google Cloud lainnya
  - Contoh kasus penggunaan: menyimpan data untuk aplikasi web, gudang data, dan pembelajaran mesin

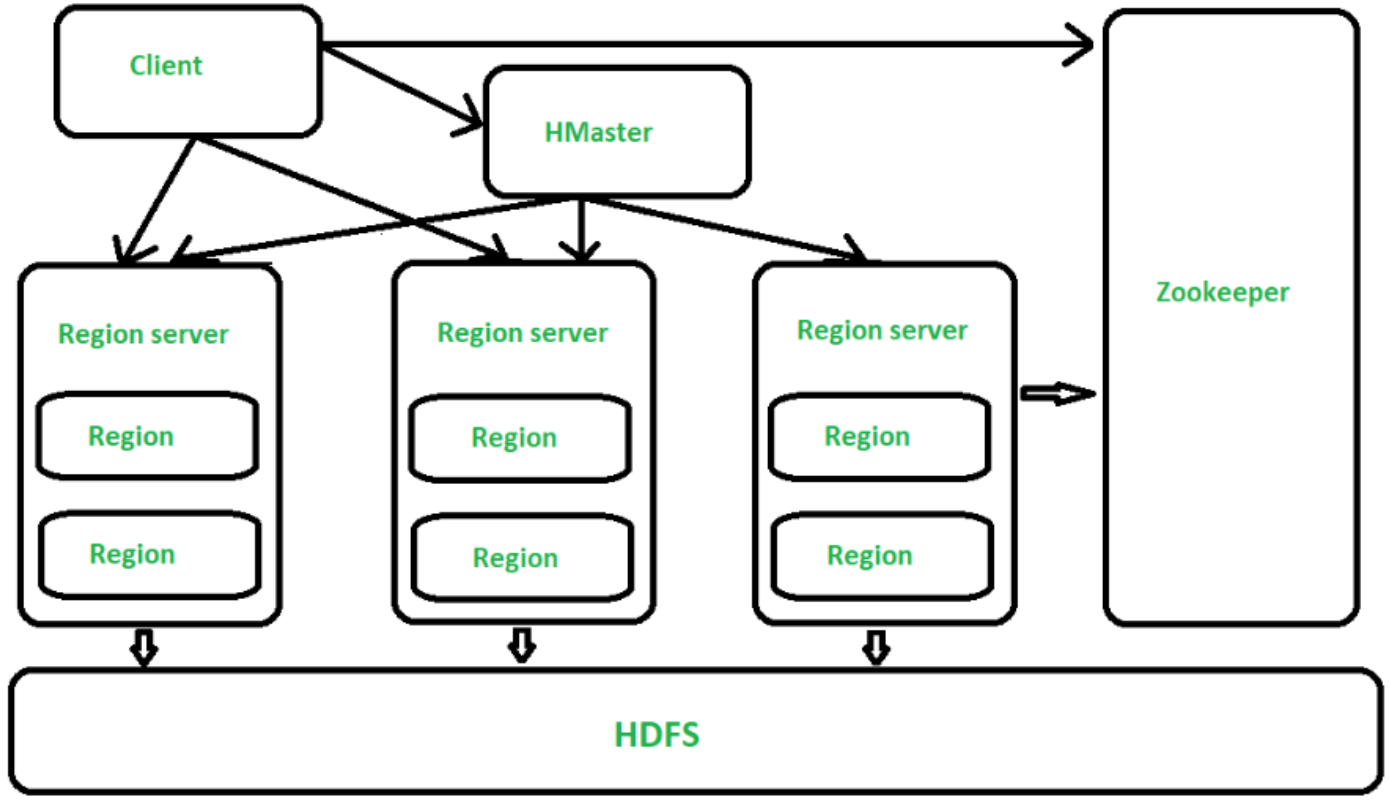
KONSEP DASAR HBASE

- **Model penyimpanan kunci-nilai**
  - HBase menyimpan data dalam format kunci-nilai, di mana setiap baris data terdiri dari kunci dan nilai.
  - Kunci adalah nilai unik yang digunakan untuk mengidentifikasi baris data, sedangkan nilai adalah data sebenarnya yang disimpan.
- **Skalabilitas horizontal**
  - HBase dapat diskalakan secara horizontal dengan menambahkan lebih banyak node ke cluster.
  - Hal ini memungkinkan HBase untuk menangani volume data yang besar dan beban kerja yang berat.
- **Toleransi kegagalan**
  - HBase menggunakan replikasi untuk memastikan ketersediaan data bahkan jika beberapa node gagal.
- ***Real-time data processing***
  - HBase mendukung pemrosesan data *real-time*, yang memungkinkannya untuk digunakan untuk aplikasi yang membutuhkan akses data yang cepat.

ARSITEKTUR HBASE

Terdiri dari tiga komponen utama:

- **Region Server**: Menyimpan data dalam format *key-value*
- **Master Server**: Mengelola Region Server dan mengatur data
- **ZooKeeper**: Mengelola *cluster HBase*



Sumber gambar.
https://www.geeksforgeeks.org/architecture-of-hbase/

PERBANDINGAN HBASE DAN HDFS

Fitur	HBase	HDFS
Model penyimpanan	Kunci-nilai	File
Data	Semi-terstruktur	Terstruktur
Skalabilitas	Horizontal	Horizontal
Toleransi kegagalan	Tinggi	Tinggi
Kinerja	Tinggi untuk operasi baca dan tulis	Tinggi untuk operasi baca
Biaya	Efisien biaya	Efisien biaya

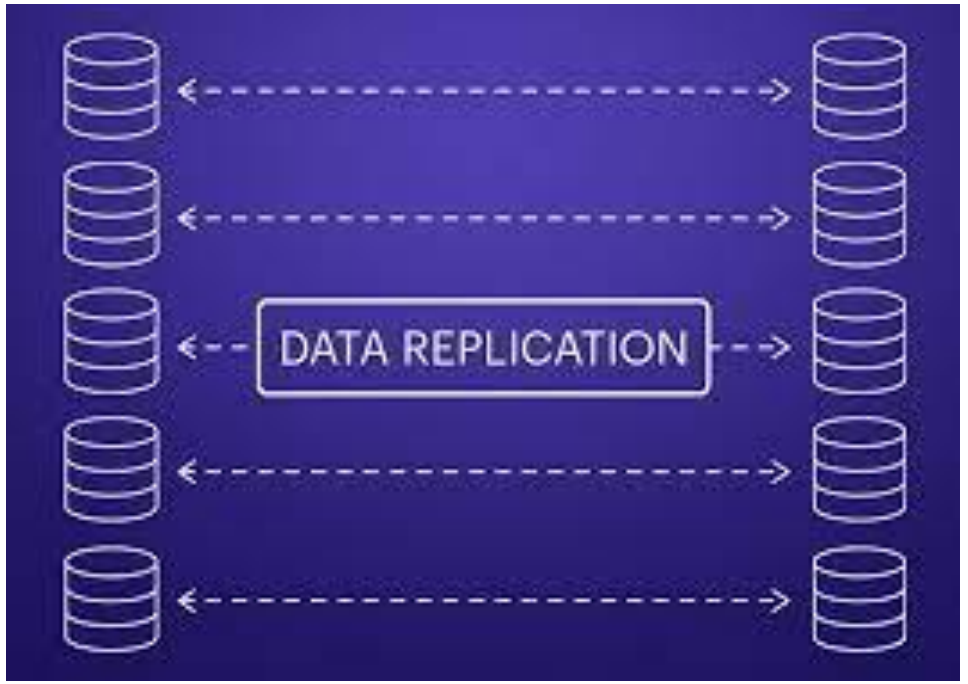
PENGGUNAAN HBASE

- Akses data real-time
  - HBase dapat digunakan untuk menyimpan data log dari aplikasi web. Dengan menggunakan HBase, data log dapat diakses secara cepat untuk analisis real-time.
- Analisis data besar
  - HBase dapat digunakan untuk menyimpan data transaksi dari bisnis. Dengan menggunakan HBase, data transaksi dapat dianalisis untuk mendapatkan insights yang berharga.
- Pembelajaran mesin
  - HBase dapat digunakan untuk menyimpan data pelatihan untuk model pembelajaran mesin. Dengan menggunakan HBase, data pelatihan dapat diakses secara cepat untuk meningkatkan kinerja model pembelajaran mesin.

REPLIKASI DATA DALAM BIG DATA

» KEPENTINGAN

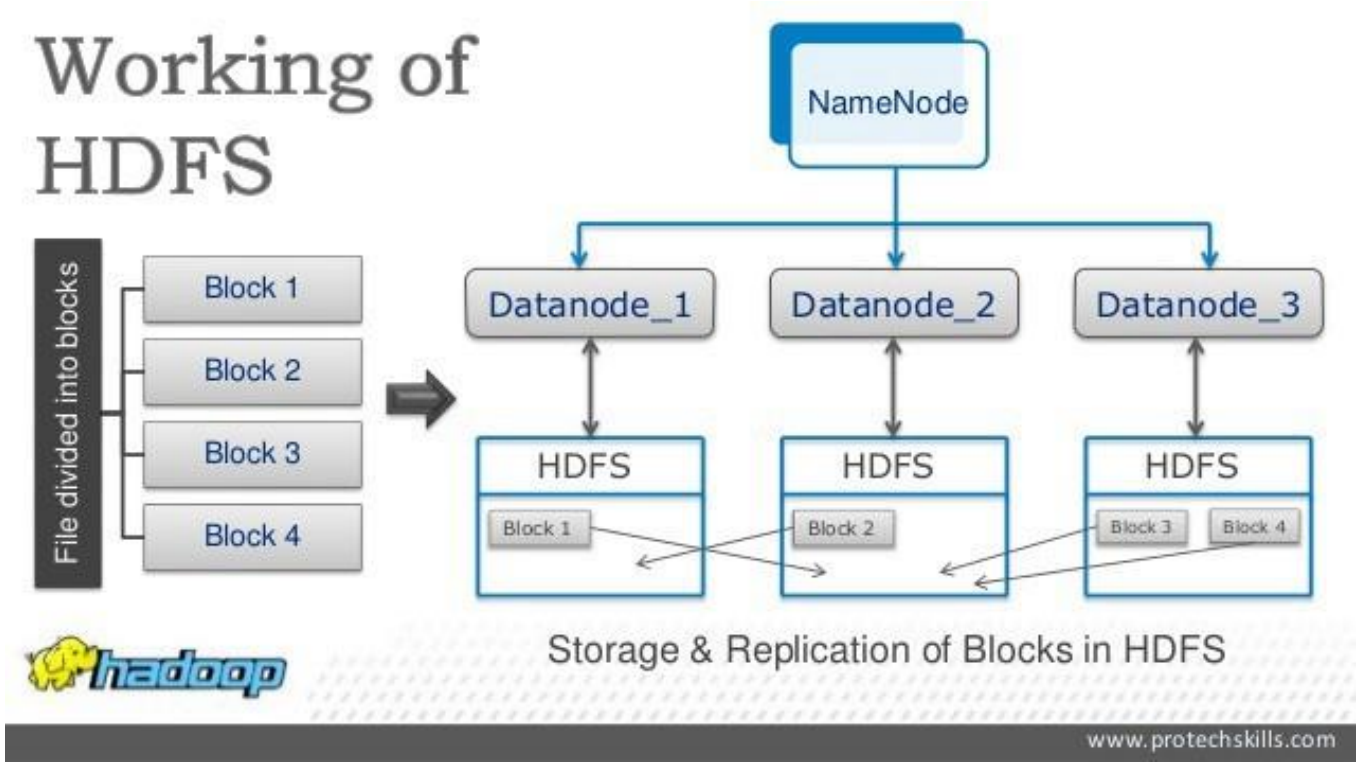
- Skala dan Ketersediaan:
  - Mengatasi tantangan ketersediaan pada skala besar dengan membuat salinan data di beberapa lokasi atau node.
- Pemrosesan Paralel:
  - Mendukung pemrosesan paralel dengan menyimpan salinan data di beberapa node, memungkinkan tugas-tugas dijalankan secara bersamaan.



REPLIKASI DATA DALAM BIG DATA

» TEKNIK DAN STRATEGI REPLIKASI DI HADOOP

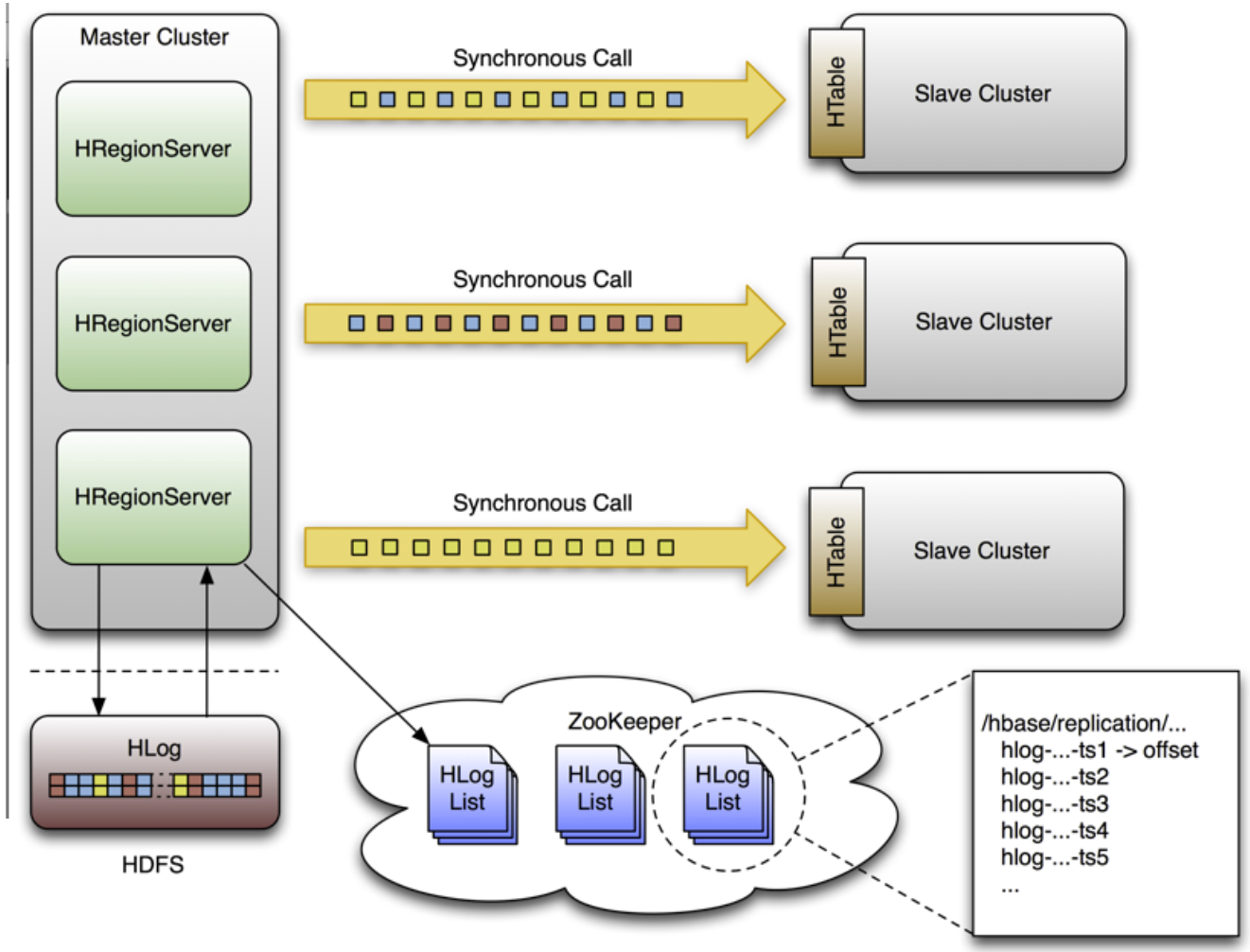
- Block-level Replication (HDFS):
  - Data dipecah menjadi blok-blok, dan setiap blok direplikasi di beberapa node.
  - Defaultnya, HDFS menggunakan faktor replikasi 3 (tiga salinan).
- Rack Awareness:
  - Strategi replikasi yang mempertimbangkan lokasi fisik node dalam satu *rack*, mengurangi latensi dan beban jaringan.



REPLIKASI DATA DALAM BIG DATA

» TEKNIK DAN STRATEGI REPLIKASI DI HBASE

- Column-family Replication:
  - Menawarkan replikasi tingkat kolom, memungkinkan kontrol yang lebih granular terhadap data yang direplikasi.
  - Cocok untuk aplikasi yang memerlukan akses cepat terhadap subset data tertentu.
- Multi-Master Replication:
  - Beberapa node dapat menerima dan menulis data, meningkatkan ketersediaan dan distribusi beban.



Sumber gambar.
http://people.apache.org/~stack/site/replication.html

MANFAAT REPLIKASI DATA

1. Ketersediaan Tinggi:
   - Memastikan ketersediaan data yang tinggi dengan menyediakan salinan di berbagai node atau lokasi.
2. Redundansi:
   - Mencegah kehilangan data akibat kegagalan perangkat keras atau node dengan adanya salinan yang dapat diakses.
3. Efisiensi Pemrosesan:
   - Mendukung pemrosesan paralel dan distribusi beban, meningkatkan efisiensi dan kinerja keseluruhan.
4. Toleransi Kesalahan:
   - Menyediakan mekanisme toleransi kesalahan yang dapat mendeteksi dan menangani kegagalan dengan cepat.

TANTANGAN REPLIKASI DATA

1. Konsistensi Data:
   - Mengelola konsistensi antara salinan data menjadi tantangan, terutama dalam lingkungan dengan tulisan data yang sering terjadi.
2. Penggunaan Sumber Daya:
   - Replikasi memerlukan penggunaan sumber daya tambahan, termasuk ruang penyimpanan dan *bandwidth* jaringan.
3. Manajemen Konflik:
   - Konflik dapat muncul saat menyinkronkan salinan data, dan perlu strategi manajemen konflik yang baik.
4. Biaya:
   - Biaya penyimpanan tambahan untuk salinan data dapat menjadi faktor, terutama dalam skala besar.

PARTISI DATA DALAM BIG DATA

- Partisi data adalah teknik untuk membagi dataset besar menjadi bagian yang lebih kecil, disebut partisi, berdasarkan suatu kriteria.
- Tujuan → memungkinkan pemrosesan dan pencarian data menjadi lebih efisien.
- Skala Besar:
  - Partisi data diperlukan ketika dataset terlalu besar untuk diproses atau dicari secara keseluruhan.
- Optimasi Pemrosesan:
  - Meningkatkan efisiensi pemrosesan dengan memfokuskan operasi pada subset data yang relevan.

PARTISI DATA DI APACHE HBASE

- Partisi Berbasis Kolom:
  - HBase menggunakan partisi berbasis kolom untuk mendistribusikan data terkait ke dalam partisi.
  - Setiap partisi berisi data dari kolom yang sama.
  - Misalnya, jika kita memiliki tabel HBase yang menyimpan data tentang produk, kita dapat membagi data berdasarkan kolom **kategori**. Dengan demikian, semua data tentang produk dalam kategori yang sama akan berada dalam partisi yang sama.
- Key Range Partitioning:
  - Key range partitioning memastikan bahwa data dengan kunci yang serupa berada dalam partisi yang sama.
  - Misalnya, jika kita memiliki tabel HBase yang menyimpan data tentang transaksi, kita dapat membagi data berdasarkan **tanggal transaksi**. Dengan demikian, semua transaksi yang terjadi pada tanggal yang sama akan berada dalam partisi yang sama.
- Pemilihan Jenis Partisi
  - Partisi berbasis kolom cocok untuk aplikasi yang membutuhkan akses cepat ke data yang terkait.
  - Key range partitioning cocok untuk aplikasi yang membutuhkan akses cepat ke data yang tersebar secara merata.

MANFAAT PARTISI DATA

1. Distribusi Beban:
   - Pemrosesan Paralel:
     - Memungkinkan pemrosesan paralel data di beberapa node, mengurangi beban di setiap node tunggal.
   - Skalabilitas:
     - Menyederhanakan skala horizontal dengan menambahkan node atau partisi baru.
2. Kinerja yang Ditingkatkan:
   - Akses Cepat:
     - Memungkinkan akses cepat ke data yang relevan tanpa memerlukan pencarian di seluruh dataset.
   - Pemrosesan Efisien:
     - Operasi pencarian dan analisis lebih efisien karena terfokus pada subset data yang lebih kecil.
3. Optimasi Penggunaan Sumber Daya:
   - Penggunaan Sumber Daya yang Lebih Efektif:
     - Meminimalkan penggunaan sumber daya dengan memproses hanya sebagian kecil data yang diperlukan.
   - Optimasi Penyimpanan:
     - Mengurangi kebutuhan ruang penyimpanan dengan menyimpan hanya data yang diperlukan dalam setiap partisi.

KETERSEDIAAN DAN KEANDALAN

Strategi untuk Memastikan Ketersediaan dan Keandalan.

- Replikasi Data:
  - Membuat salinan data di beberapa node atau lokasi untuk meningkatkan ketersediaan dan keandalan.
- Partisi Data:
  - Memisahkan data menjadi partisi untuk mengoptimalkan pemrosesan dan distribusi beban, serta meminimalkan dampak kesalahan terhadap keseluruhan sistem.

KETERSEDIAAN DAN KEANDALAN (LANJUTAN)

Deteksi dan Manajemen Kesalahan.

- Monitoring Sistem:
  - Pemantauan terus-menerus untuk mendeteksi anomali atau tanda-tanda kegagalan.
- Sistem Notifikasi:
  - Pengaturan sistem notifikasi untuk memberi tahu secara cepat ketika terjadi kesalahan atau ancaman ketersediaan.

KETERSEDIAAN DAN KEANDALAN (LANJUTAN)

Penggunaan Teknik Redundansi untuk Meningkatkan Keandalan.

- Redundansi Perangkat Keras:
  - Penggunaan perangkat keras cadangan atau salinan untuk mengurangi dampak kegagalan perangkat keras utama.
  - RAID (Redundant Array of Independent Disks):
    - Sistem penyimpanan yang menggunakan kombinasi dari beberapa disk untuk meningkatkan kinerja dan redundansi.
  - Hot Standby:
    - Perangkat keras cadangan yang siap digunakan jika perangkat utama mengalami kegagalan.
- Redundansi Jaringan:
  - Jaringan cadangan atau *multiple paths* untuk menghindari kegagalan titik tunggal dalam komunikasi data.
  - Multipath Routing:
    - Pengaturan jaringan dengan rute ganda untuk menghindari kegagalan satu jalur tunggal.
  - Load Balancing:
    - Distribusi beban trafik di antara jalur-jalur yang tersedia untuk mencegah overload atau kegagalan satu titik.

PEMULIHAN BENCANA

1. Strategi Pemulihan Bencana:
   - Rencana Pemulihan Bencana (*Disaster Recovery Plan*):
     - Dokumentasi rinci langkah-langkah yang harus diambil untuk memulihkan operasi normal setelah terjadinya bencana.
   - Pengelolaan Risiko:
     - Identifikasi potensi risiko dan pengembangan strategi untuk mengurangi dampaknya.
2. Teknik Pemulihan Bencana:
   - Backups Reguler:
     - Melakukan cadangan data secara reguler untuk memastikan adanya salinan yang dapat dikembalikan.
   - Replicasi Data:
     - Mereplikasi data di lokasi geografis yang berbeda untuk memitigasi risiko bencana yang melibatkan satu lokasi.

MANAJEMEN DATA

1. Prinsip-prinsip Manajemen Data dalam Skala Besar:
   - Distribusi Data:
     - Prinsip mengelola data yang tersebar di berbagai node dan lokasi untuk memastikan ketersediaan dan kinerja.
   - Skalabilitas:
     - Berfokus pada kemampuan sistem untuk dengan mudah berkembang seiring dengan pertumbuhan volume data.
2. Pengelolaan dan Katalogisasi Data:
   - Metadata dalam Big Data:
     - Merupakan informasi tambahan tentang data, seperti asal, format, dan sejarah perubahan.
     - Manfaat: meningkatkan visibilitas, pencarian, dan pemahaman tentang data dalam ekosistem Big Data
   - Manajemen Katalog Data:
     - Pengelolaan katalog data untuk membuat metadata mudah diakses dan dicari.

TREN DAN PERKEMBANGAN TERKINI

- Peningkatan penggunaan teknologi *cloud*:
  - Teknologi *cloud* semakin populer untuk penyimpanan Big Data.
  - *Cloud* menawarkan berbagai manfaat, seperti skalabilitas, fleksibilitas, dan biaya yang lebih rendah.
- Pertumbuhan data semi-terstruktur dan tidak terstruktur:
  - Data semi-terstruktur dan tidak terstruktur semakin mendominasi Big Data.
  - Teknologi penyimpanan baru yang dirancang untuk menangani data jenis ini terus dikembangkan.
- Peningkatan kebutuhan akan analisis *real-time*:
  - Kebutuhan akan analisis *real-time* untuk Big Data semakin meningkat.
  - Teknologi penyimpanan baru yang mendukung analisis *real-time* terus dikembangkan.

STUDI KASUS: ANALISIS SENTIMEN TWITTER DENGAN HADOOP DAN HBASE

- Studi kasus ini membahas bagaimana menggunakan Hadoop dan HBase untuk mengumpulkan, menyimpan, dan menganalisis data tweet dari Twitter.
- Tujuan dari studi kasus ini adalah untuk mengetahui sentimen publik terhadap topik tertentu, seperti produk, layanan, atau isu sosial.
- Studi kasus ini meliputi langkah-langkah berikut:
  - Menggunakan Twitter Streaming API untuk mengambil data tweet secara real-time dan menyimpannya dalam HDFS.
  - Menggunakan Apache Flume untuk mentransfer data tweet dari HDFS ke HBase.
  - Menggunakan Apache Pig untuk melakukan pra-pemrosesan data tweet, seperti membersihkan, menormalkan, dan mengekstraksi fitur.
  - Menggunakan Apache Mahout untuk melakukan klasifikasi sentimen data tweet, seperti positif, negatif, atau netral.
  - Menggunakan Apache Hive untuk melakukan agregasi dan visualisasi hasil analisis sentimen.

TANYA JAWAB

Terima Kasih