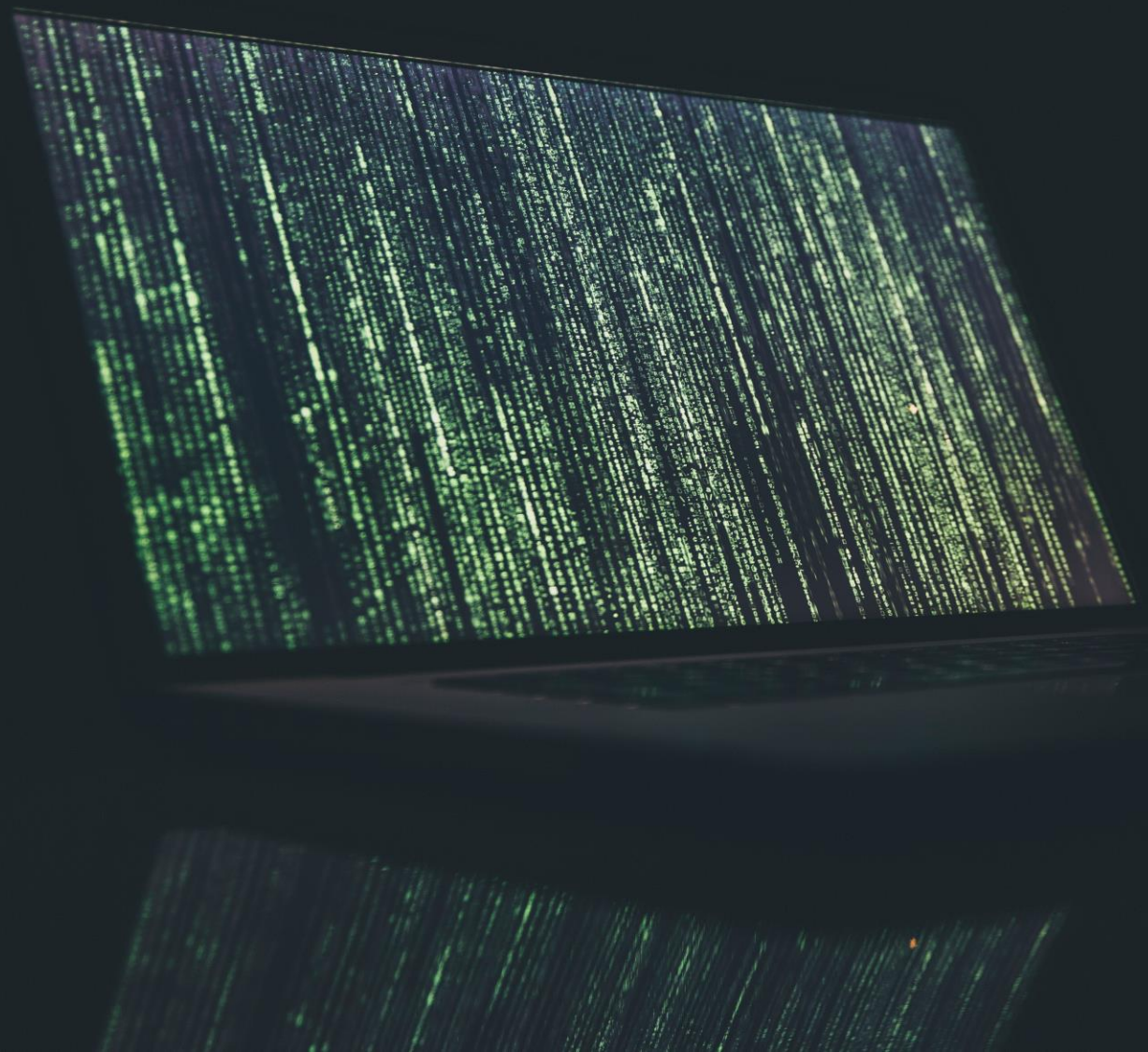

PERTEMUAN 4

TEKNOLOGI PEMROSESAN BIG DATA SECARA *BATCH*

Mata kuliah

Infrastruktur Dan Teknologi Big Data

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

Prodi Teknik Informatika. FTIK.

Universitas komputer indonesia

PENTINGNYA BIG DATA DALAM KONTEKS MODERN

- **Informasi Mendalam**: Big Data menyediakan akses ke volume data besar dan beragam untuk mendapatkan wawasan mendalam.
- **Pemutusan Paradigma**: Keputusan didasarkan pada analisis data kuat, bukan hanya pengalaman atau intuisi.
- **Inovasi Produk dan Layanan**: Identifikasi kebutuhan pelanggan untuk pengembangan produk dan layanan yang lebih sesuai.
- **Efisiensi Operasional**: Optimalkan proses operasional dengan analisis data.
- **Personalisasi Pelanggan**: Memahami preferensi pelanggan untuk pengalaman yang lebih personal dan relevan.

PENTINGNYA BIG DATA DALAM KONTEKS MODERN (LANJUTAN)

- **Keamanan dan Deteksi Ancaman**: Menggunakan Big Data untuk deteksi ancaman dan keamanan.
- **Penelitian dan Pengembangan**: Digunakan dalam penelitian medis, pengembangan obat, dan industri lainnya.
- **Keputusan Bisnis Optimal**: Mendukung keputusan bisnis yang lebih tepat dan terukur.
- **Kompetisi Global**: Keunggulan kompetitif di pasar global.
- **Inovasi Terus-Menerus**: Identifikasi tren baru, peluang bisnis, dan cara baru untuk memenuhi kebutuhan pelanggan.

METODE PEMROSESAN BIG DATA

- **Pemrosesan *Batch***: Data diolah dalam batch besar pada interval waktu tertentu.
- **Pemrosesan *Real-time***: Data diolah segera setelah diterima.
- **Pemrosesan *Stream***: Data diproses secara kontinu saat mereka masuk.
- **Pemrosesan Interaktif**: Memungkinkan interaksi langsung dengan data untuk analisis exploratif.

KONSEP PEMROSESAN BATCH

- Merupakan sebuah metode pemrosesan data di mana sejumlah besar data dikumpulkan dan diproses secara bersamaan pada interval waktu tertentu.
- Proses ini melibatkan mengumpulkan, menyimpan, dan memproses data dalam kelompok atau *batch*, tanpa memperhatikan kapan data itu diterima.

Ringkasan
Data diakumulasi dan diproses secara periodik.
Terjadi pada interval waktu tertentu.
Cocok untuk analisis data yang tidak memerlukan respons instan.
Contoh: Laporan bulanan, analisis historis.

KERANGKA PEMROSESAN DATA

Apache Hadoop

- Merupakan kerangka kerja sumber terbuka yang memungkinkan pemrosesan dan penyimpanan data Big Data secara distribusi.
- Tujuan → Memungkinkan organisasi untuk mengelola, menyimpan, dan menganalisis data dalam skala besar dengan efisien.
- Komponen utama:
  a) **HDFS (*Hadoop Distributed File System*):**
     - Sistem penyimpanan terdistribusi yang dirancang untuk menyimpan data besar di berbagai node komputer.
     - Memungkinkan replikasi data untuk toleransi kesalahan dan akses cepat.
  b) **MapReduce:**
     - Model pemrograman untuk memproses data besar secara paralel di atas HDFS.
     - Terdiri dari dua tahap utama: Map (pemetaan) dan Reduce (pengurangan).
  c) **YARN (*Yet Another Resource Negotiator*):**
     - Manajer sumber daya yang mengelola dan menugaskan sumber daya komputasi (CPU, memori) kepada aplikasi yang berjalan di atas cluster Hadoop.
     - Memungkinkan untuk menjalankan berbagai jenis aplikasi di atas Hadoop.

KERANGKA PEMROSESAN DATA (LANJUTAN)

❑MapReduce

- Merupakan model pemrograman yang digunakan untuk memproses dan menganalisis data besar secara distribusi di atas infrastruktur komputasi terdistribusi.
- Tujuan → Memungkinkan pemrosesan efisien dan paralel data besar di dalam cluster Hadoop.
- Tahap pemrosesan:
  1) Map (Pemetaan)
     - Data besar dibagi menjadi bagian-bagian kecil.
     - Setiap bagian diolah oleh fungsi Map yang menghasilkan pasangan *key-value*.
  2) Shuffle and Sort (Pengacakan dan Pengurutan)
     - Hasil dari fungsi Map diurutkan dan dikelompokkan berdasarkan kunci.
  3) Reduce (Pengurangan)
     - Data hasil dari langkah sebelumnya diolah oleh fungsi Reduce.
     - Fungsi Reduce menghasilkan keluaran akhir atau hasil analisis.

IMPLEMENTASI APACHE HADOOP DAN MAPREDUCE

- Cara menginstal dan mengkonfigurasi Apache Hadoop di Windows
  - Unduh Hadoop dari situs web resminya.
  - Unzip file yang diunduh ke direktori yang diinginkan.
  - Edit file **hadoop-env.cmd** untuk menyesuaikan konfigurasi Hadoop.
  - Jalankan perintah **start-dfs.cmd** untuk memulai Hadoop Distributed File System (HDFS).
  - Jalankan perintah **start-yarn.cmd** untuk memulai Hadoop YARN.
- Konfigurasi
  - Edit file **core-site.xml** untuk menentukan path ke HDFS dan YARN.
  - Edit file **hdfs-site.xml** untuk menentukan konfigurasi HDFS.
  - Edit file **yarn-site.xml** untuk menentukan konfigurasi YARN.

Info terbaru ada di laman web https://hadoop.apache.org/

IMPLEMENTASI APACHE HADOOP DAN MAPREDUCE (LANJUTAN)

- Contoh sederhana penggunaan MapReduce untuk pemrosesan batch
  - Buat program Java MapReduce

```java
public class KataJumlah {

    public static void main(String[] args) throws Exception {
        // Buat job MapReduce
        Configuration conf = new Configuration();
        Job job = new Job(conf, "Kata jumlah");

        // Set mapper dan reducer
        job.setMapperClass(KataJumlahMapper.class);
        job.setReducerClass(KataJumlahReducer.class);

        // Set input dan output format
        job.setInputFormatClass(TextInputFormat.class);
        job.setOutputFormatClass(TextOutputFormat.class);

        // Set input dan output path
        FileInputFormat.addInputPath(job, new Path(args[0]));
        FileOutputFormat.setOutputPath(job, new Path(args[1]));

        // Jalankan job
        job.waitForCompletion(true);
    }
}
```

Kelas **KataJumlah** adalah kelas utama yang digunakan untuk menjalankan job MapReduce.

IMPLEMENTASI APACHE HADOOP DAN MAPREDUCE (LANJUTAN)

- Buat kelas Mapper → untuk memproses data input.

```java
public class KataJumlahMapper extends Mapper<Object, Text, Text, IntWritable> {

    private final static IntWritable SATU = new IntWritable(1);

    @Override
    public void map(Object key, Text value, Context context) throws IOException, InterruptedException {
        // Split text menjadi kata-kata
        String[] kata = value.toString().split(" ");

        // Hitung jumlah kemunculan setiap kata
        for (String kata : kata) {
            context.write(new Text(kata), SATU);
        }
    }
}
```

IMPLEMENTASI APACHE HADOOP DAN MAPREDUCE (LANJUTAN)

- Buat kelas Reducer → untuk menggabungkan hasil pemrosesan dari mapper.

```java
public class KataJumlahReducer extends Reducer<Text, IntWritable, Text, IntWritable> {

    @Override
    public void reduce(Text key, Iterable<IntWritable> values, Context context) throws IOException,
InterruptedException {
        // Hitung total kemunculan kata
        int jumlah = 0;
        for (IntWritable value : values) {
            jumlah += value.get();
        }

        // Tampilkan hasil
        context.write(key, new IntWritable(jumlah));
    }
}
```

IMPLEMENTASI APACHE HADOOP DAN MAPREDUCE (LANJUTAN)

- Jalankan di *console* (sesuaikan dengan nama fail dan kelas yang Anda buat):

```
hadoop jar katajumlah.jar KataJumlah input output
```

- Contoh luaran

```
halo 2
dunia 1
```

Contoh dalam bahasa Python

- Implementasi MapReduce menggunakan Python, dapat menggunakan **Hadoop Streaming API**.
- Fungsi **mapper**() digunakan untuk memproses data input. Fungsi ini menerima dua parameter, yaitu key dan value. Key adalah nilai kunci yang digunakan untuk menggabungkan hasil pemrosesan dari mapper. Value adalah nilai data yang akan diproses.
- Fungsi **reducer**() digunakan untuk menggabungkan hasil pemrosesan dari mapper. Fungsi ini menerima dua parameter, yaitu key dan values. Key adalah nilai kunci yang digunakan untuk menggabungkan hasil pemrosesan dari mapper. Values adalah nilai data yang akan digabungkan.
- Fungsi **yield** digunakan untuk menghasilkan pasangan key-value. Pasangan key-value ini akan dikirim ke reducer.
- Fungsi **sys.argv** digunakan untuk mengambil argumen dari baris perintah. Argumen input_path digunakan untuk menentukan path ke data input. Argumen output_path digunakan untuk menentukan path ke data output.

```python
# Buat fungsi mapper
def mapper(key, value):
    # Split text menjadi kata-kata
    kata = value.split(" ")

    # Hitung jumlah kemunculan setiap kata
    for kata in kata:
        yield kata, 1

# Buat fungsi reducer
def reducer(key, values):
    # Hitung total kemunculan kata
    jumlah = 0
    for value in values:
        jumlah += value

    # Tampilkan hasil
    yield key, jumlah

# Jalankan job MapReduce
import sys

input_path = sys.argv[1]
output_path = sys.argv[2]

with open(input_path, "r") as f:
    for line in f:
        # Mapper
        for key, value in mapper(None, line):
            # Reducer
            for key, value in reducer(key, [value]):
                print(key, value)
```

TEKNIK PEMROSESAN BATCH

a) Analisis Data.
   - Langkah 1: Mengumpulkan dan mempersiapkan data besar untuk analisis.
   - Langkah 2: Memproses data dalam *batch* menggunakan algoritma atau model analisis tertentu.
   - Langkah 3: Menganalisis hasil untuk mendapatkan wawasan dan informasi yang berharga.

b) Transformasi Data.
   - Langkah 1: Menentukan tujuan transformasi data (misalnya, normalisasi, agregasi, penggabungan).
   - Langkah 2: Mengidentifikasi dan menerapkan teknik transformasi yang sesuai pada data dalam *batch*.
   - Langkah 3: Memastikan data hasil transformasi dapat digunakan untuk keperluan analisis atau aplikasi lainnya.

CONTOH KASUS NYATA: ANALISIS DATA PENJUALAN E-COMMERCE

Deskripsi Kasus:

- Sebuah perusahaan e-commerce besar ingin meningkatkan efisiensi operasional dan memahami pola pembelian pelanggan. Mereka memiliki dataset besar yang mencakup informasi tentang transaksi penjualan, produk yang dibeli, dan profil pelanggan.

CONTOH KASUS NYATA: ANALISIS DATA PENJUALAN E-COMMERCE (LANJUTAN)

Langkah Implementasi:

1. Pengumpulan dan Pemrosesan Data:
   - Data penjualan dari berbagai cabang dan kategori produk dikumpulkan dan disimpan dalam sistem.
2. Pemrosesan Batch dengan MapReduce:
   - Menggunakan Apache Hadoop dan MapReduce untuk menghitung total penjualan per kategori produk secara bulanan.
   - Hasilnya adalah dataset baru yang berisi informasi agregat tentang total penjualan.
3. Analisis Pola Pembelian:
   - Memanfaatkan teknik analisis data untuk mengidentifikasi pola pembelian yang signifikan, seperti produk terlaris, tren musiman, dan preferensi pelanggan.
4. Segmentasi Pelanggan:
   - Melakukan segmentasi pelanggan berdasarkan perilaku pembelian, seperti pelanggan yang sering berbelanja, pelanggan baru, dan pelanggan yang kurang aktif.

CONTOH KASUS NYATA: ANALISIS DATA PENJUALAN E-COMMERCE (LANJUTAN)

Hasil atau Manfaat yang Dihasilkan:

- Memahami preferensi dan kebiasaan pembelian pelanggan.
- Meningkatkan strategi pemasaran dengan menargetkan promosi yang lebih tepat sasaran.
- Mengoptimalkan persediaan dengan memahami permintaan produk.
- Mengidentifikasi produk yang paling menguntungkan dan memprioritaskan fokus pemasaran.

Catatan: Studi kasus ini adalah contoh hipotetis dan dapat bervariasi tergantung pada data dan tujuan bisnis aktual dari perusahaan *e-commerce* tersebut.

TANYA JAWAB

Terima Kasih