COMM1822

Term 2 2022

Introduction to Databases for Business Analytics

Week 9 Big Data 2

Lecturer-in-Charge: Kam-Fung (Henry) Cheung
Email: kf.cheung@unsw.edu.au
Tutors: Theresa Tran
Liam Li Chen
Kathy Xu
PASS Leader: Srilekha Chandrashekara Kolaki

Copyright

- There are some file-sharing websites that specialise in buying and selling academic work to and from university students.

- If you upload your original wor other student downloads and presents it as their own eit ight be found guilty of collusion **— even years after graduatio**

- These file-sharing websites may also accept purchase of course materials, **such as copies of <u>lecture slides</u> and <u>tutorial handouts</u>. By law, the copyright on course materials, developed by UNSW staff in the course of their employment, belongs to UNSW. It constitutes copyright infringement, if not academic misconduct, to trade these materials.**

Acknowledgement of Country

UNSW Business School acknowledges the Bidjigal (Kensington campus) and Gadigal (City campus) the traditional custodians of the lands where each campus is located.

We acknowledge all Aboriginal and Torres St Islander Elders, past and present and thei communities who have shared and practiced their teachings over thousands of years including business practices.

We recognise Aboriginal and Torres Strait Islander people's ongoing leadership and contributions, including to business, education and industry.

UNSW Business School. (2022, May 7). *Acknowledgement of Country* [online video]. Retrieved from https://vimeo.com/369229957/d995d8087f

Chapter 14
and NoSQL

W9 Learning Outcomes

- **Big Data Technologies**
  - Hadoop Ecosystem
    - Hadoop Distributed File System (HDFS)
    - MapReduce
    - Pig
    - Hive
    - HBase
    - Impala

- **NoSQL Database Types**
  - Key-value databases
  - Document databases
  - Column-oriented databases
  - Graph databases

- **Big Data Strategies**

Big Data Technologies

Big Data Infrastructure Challenges

- **Linear scalability**
  - To accommodate for the scalability of processing, thereby the storage management and architecture of traditional data management techniques become obsolete.

- **High throughput**
  - Infrastructure that is extremel processing, and storage.

- **Fault tolerance**
  - Any portion of the processing architecture should be able to take over and resume processing from the point of failure in any other part of the system.

Big Data Infrastructure Challenges

- **Auto recovery**
  - The processing architecture should be self-managing and recover from failure without manual intervention.

- **High degree of paralleli**

  - Distribute the load across multiple machines, eac n copy of the same data, but processing a different program. e.g., data analysis u ods: linear regression, random forests

- **Distributed data processing**
  - The underlying platform must be able to process distributed data to achieve extreme scalability.

What is Hadoop?

- Hadoop is an open-source framework for storing and analyzing massive amounts of distributed, unstructured data.

- Hadoop was created by D arella in 2005.

- Hadoop clusters run on inexpensive com are so projects can scale-out inexpensively.

- Open source - hundreds of contributors continuously improve the core technology.

- What is Hadoop? - https://www.youtube.com/watch?v=9s-vSeWej1U

Hadoop

- Not a single product, not a single database.
- A **collection of big data applications**.
- A **framework**, platform and ecosystem.
- Consisting of **different** .

- Most important components:
  - **Hadoop Distributed File System (HDFS)**
  - **MapReduce**
  - **Pig**
  - **Hive**
  - **HBase**
  - **Impala**

Why Hadoop?

- **Problems with relational database management system (RDBMS):**
  - Insufficiently scalable for big data
  - Insufficient speed for live data
  - Lack of sophisticated
  - Essentially a design b CPU and RAM (you can easily "scale up" to an extend, but cale out")

- **Polyglot persistence**: The coexistence of a variety of data storage and data management technologies within an organization's infrastructure.

Structured: Customer's data, e.g., date of birth, address, bank account, …
Unstructured: Customer's feedback (in text), …

Hadoop Ecosystem

Hadoop Ecosystem – Core Components

- **Hadoop Distributed File System (HDFS)**
- **MapReduce**

Hadoop Distributed File System (HDFS)

- **Hadoop** stores **files across networks** using **Hadoop Distributed File System (HDFS)**

- **Hence, Hadoop** is **not a** **sical database**, it is **a distributed file system** (s and tools in its ecosystem)

- **Networks** can be very large, **10,000s of computers**

- HDFS is a low-level **distributed file processing system** (can be used directly for data storage)

Hadoop Distributed File System (HDFS)

HDFS/Hadoop approach based on several **key assumptions**:

- **High volume:** Default physical **block sizes is 64 MB**, hence much fewer blocks per file (files are assumed to be very large)
- **Write-once, read-many:** M sues and improves data throughput
- **Streaming access:** Hadoop is optimized for ing of entire files as a continuous stream of data
- **Fault tolerance:** HDFS is designed to replicate data across many different devices so that when one fails, data is still available from another device (default **replication factor of three**)

Hadoop Distributed File System (HDFS)

- HDFS uses several types of nodes (computers): (see figure next slide)
  - **Data node** stores the actual file data
  - **Name node** contains file system metadata
  - **Client node** makes requests to the file system as needed to support user applications

- **Same computer** can ful **nctions**.

- Data node communicates with name no rly sending block reports (list of blocks, every 6 hours) and heartbeats (every 3 seconds)
  - If heartbeat stops, data blocks of that node are replicated elsewhere

Hadoop Distributed File System (HDFS)

How Does HDFS Work? [Writing]

1. The **client node** needs to create a new file, and communicates with the **name node**.
2. The **name node**
   - adds the new file name to the metadata;
   - determines a new (first) block
   - determines a list of on which d rect;
   - and passes that information b
3. The **client node**
   - contacts the first data node specified by the name n riting;
   - sends the data node the list of replicating data nodes.
4. First **data node** contacts the second data node in the list for replication while receiving it from the client node.
5. The **client node** gets further block numbers from the name node … until file is written.

MapReduce

- Implementation **complements HDFS structure**.
- **Open-source** application programming interface (API).
- **Framework** used to process large data sets across clusters.
- **"Divide and conquer"** st into smaller subtasks, performed at node level i ated to final result.
- Based on **batch processing**, runs tasks ing to end with no user interaction.

- **YARN** (Yet Another Resource Negotiator), or MapReduce 2, can do
  - Batch processing
  - **Stream processing** (for data that comes in/out continuously)
  - **Graph processing** (for social networks)

MapReduce

- **Map function** takes a collection of data and sorts and filters it into a set of key-value pairs.
  - **Mapper** program performs the map function
- **Reduce function** summ n to produce a single result.
  - **Reducer** program performs the reduce function

- **Map and reduce functions** are written as **Java** programs.
- Instead of central program retrieving the data for processing in a central location, **copies of the program are "pushed" to the nodes**.
- Typically **1 mapper *per block***, **1 reducer *per node***.

MapReduce

MapReduce

- **Job tracker** or central control program to accept, distribute, monitor and report on jobs in a Hadoop environment
  - Typically on **name node**.
- **Task tracker** is a pr e responsible for reducing tasks on a node
  - Typically on **data node**.

How Does MapReduce Work? [Reading/Analyzing]

1. A **client node** (client application) submits a **MapReduce job** to the job tracker.
2. The **job tracker** (on server that is also the **name node**):
   - communicates with name n ata node;
   - determines which task track d be busy);
   - send portions of work to task trackers.
3. The **task tracker** (on server that is al ode)
   - runs map and reduce functions (in virtual machine);
   - sends heartbeat ("still working") and "complete" message to job tracker.
4. The **client** node
   - periodically queries job tracker if all task trackers are completed;
   - receives completed job.

Hadoop Ecosystem – Data Ingestion Applications

- **Flume**
- **Sqoop**

- Why? **Help getting data** **into Hadoop clusters**. These tools “ingest” or gather data into Hadoop.

Flume

- Flume is a component for **ingesting data in Hadoop**.
- Primarily for harvesting large sets of data such **clickstream data/server logs**.
- Simple query processing **some transformation**.
- Can move data into **HDFS** or **HBase**.

Sqoop

- **“SQL-to-Hadoop.”**
- **Sqoop** is a tool for converting data back and forth between **relational databases** and **HDFS** (both directions).
- Works with Oracle, MySQ

- Example of Hadoop-to-SQL: MapReduce rted back into a traditional (relational) data warehouse.

Hadoop Ecosystem – MapReduce Simplification Applications

- **Hive**
- **Pig**

- Why? **They help creating MapRedu**

  - Creating MapReduce jobs requires signific ing skills.
  - As the mapper and reducer programs become more complex, the skill requirements increase and the time to produce the programs becomes significant.

Hive

- **Hive** is a **data warehousing system** that sites on top of HDFS.
  - Supports its own **SQL-like language: HiveQL** (declarative / non-procedural)
  - **Summarizes queries, analyzes data**

This is the component tha t use in terms of how to actually work with the data!

Pig

- Hadoop platform to **write MapReduce programs.**
- Has its own **high-level scripting/programming language: Pig Latin** (procedural).
- **Pig** compiles Pig Latin scripts **into MapR** for executing in Hadoop.

Hadoop Ecosystem – Direct Query Applications

- **HBase**
- **Impala**

- Why? **To provide faste****y to HDFS** (without going through the MapReduce processing la

HBase

- **HBase** is a **NoSQL** database
- **Column-oriented**
- Designed to **sit on top of**
- **Quickly** processes **smaller subsets** of t
- **No SQL** support, instead uses **Java**

Impala

- First **SQL-on-Hadoop** application
- Produced by **Cloudera**
- **SQL queries** directly aga l in **HDFS**
- Makes heavy use of **in-memory caching** **des**

NoSQL Database Types

NoSQL

- **Non-relational database technologies** developed to address **Big Data challenges**
- **NoSQL** = “not modelled using relational model” (“non-SQL” / “not-only SQL”)
- **Category** emerged from organizations such as **Google, Amazon** and **Facebook** that faced problems of their data sets reac
- **Much larger data** volumes can
- **Flexible structure** and often **faster**
- No standardized query language – no SQL! (mayb

- Less adopted than RDBMS:
  - Was at peak in 2015-2016
  - Survey 2016, **16%** of companies use **NoSQL** databases and **79%** of companies use **relational databases**
- *NoSQL seems to be in decline nowadays ??!!*

NoSQL

NoSQL

NoSQL – Key-Value Database

- Store data as a collection of **key-value pairs** (keys ~ primary keys, there are no foreign keys)
- Key-value pairs are organiz
  logical groupings, **buckets** (~ tables)
- Key values must be unique (only) within a bucket.
- **Queries** are based on **buckets and keys** (not values)
- ***get*, *store* and *delete*** operations

NoSQL – Document Databases

- **Document databases** store data in **key-value pairs** in which the value components are **tag-encoded documents.**
- Document can be encoded in **JSON** or **BSON** (Binary JSON)
- Have tags, but still **schema-less** (not schemas, documents may have different tags).
- Documents are grouped into logical groups called **collections** (buckets).
- Tags can be queried (e.g., where balance = 0).

NoSQL – Column-Centric Databases

- **Column-centric (columnar) databases** focuses on storing data in columns, not rows, but still relational logic.

- **Column-centric storage:** Data in blocks which hold data from **column across many rows**

- **Row-centric storage:** Data stored in blocks which hold data from **all columns of a given set of rows**

NoSQL – Column-Centric Databases

- **Column-oriented (column family) databases** in NoSQL:
  - Organizes data in key-value pairs.
  - Keys are mapped to columns in the value component.
  - The columns vary by row.
- **Key-value pair:** name of the c xample: "cus_lname: Ramas". (~**cell** in relational model)
- **Super column:** group of columns that are logically posite attribute)
- **Rows keys**: created to identify objects (~**entity instances**) in the environment
- **Column family:** All of the columns (or super columns) that describe objects are grouped (~**table**)

NoSQL – Graph Databases

- Suitable for **relationship-rich data**
- A collection of **nodes and edges**
- **Properties** are the **attributes of a node or edge** of interest to a user
- **Traversal** is a **query** in a graph databases

Applications of NoSQL

- Twitter app generating 7 Tbs+ of daily tweets and displaying it back.
- Property details in a real-estate website, redundant in nature but accessed in huge numbers.
- Online coupon sites distributing coupons ... rket.
- Update of railway schedules and accessed by thousands of users at peak time.
- Real time score update of baseball / cricket match.

Big Data Strategies

What is Big Data Strategy?

A Big Data strategy defines and lays out a comprehensive vision across the enterprise and sets a foundation for the organization to employ data-related or ilities.

Week 9 - Big Data II

Pre-Class Activities

- How Do You Create A Data Strategy?
- How to Define a Big Data Strategy
- How to Develop a Data Strategy (Bernard Marr)
- Types and Examples of NoSQL Databases
- Week 9 Pre-Class Tasks

Source: https://www.bigdataframework.org/formulating-a-big-data-strategy/

Challenges of Implementing Big Data Strategy

- Technological
  - Lack of managerial analytics knowledge
  - Technical misundersta s and data scientists
  - Inherent challenges re s)
  - Technical requirements in compliance wnership and privacy regulations (e.g., NSW Transport d n could lead to app deluge https://www.itnews.com.au/news/nsw-%20transport-data-liberation-could-lead-to-app-deluge-418406)
  - Costly data management tools

(Tabesh et al. 2019)

Challenges of Implementing Big Data Strategy

- Cultural
  - Extensive reliance on intuitive or experiential decision-making approaches
  - Dominance of manag ing process
  - Lack of a shared unde its goals

(Tabesh et al. 2019)

Reference:
Tabesh, P., Mousavidin, E. and Hasani, S., 2019. Implementing big data strategies: A managerial perspective. *Business Horizons, 62*(3), pp.347-358. https://doi.org/10.1016/j.bushor.2019.02.001

Implementing Big Data Strategy

Source: https://www.bigdataframework.org/formulating-a-big-data-strategy/

Questions

Source: stacker.com