PANEVROPSKI UNIVERZITET
APEIRON
za multidisciplinarne i virtuelne studije
Pan-European University for Multidiscipline & Virtual Studies
Banja Luka

FAKULTET INFORMACIONIH TEHNOLOGIJA

PROGRAMIRANJE I SOFTVERSKO INŽINJERSTVO

OPTIMIZACJA SKLADIŠTENJA PODATAKA U BIG DATA SISTEMIMA

-Diplomski rad-

Student:
Ana Stanimirović 62-19/RITP-S

Mentor:
Prof. Dr Dražen Marinković

Banja Luka, oktobar 2024

Ana Stanimirović
(Ime i prezime studenta)

IZJAVA O AKADEMSKOJ ČESTITOSTI

Izjavljujem i svojim potpisom potvrđujem da je diplomski rad

Optimizacija skladištenja podataka u Big Data sistemima

(naslov rada)

isključivo rezultat mog vlastitog rada uz preporuke i konsultacije sa mentorom. Rad se temelji na mojim istraživanjima i oslanja se na objavljenu literaturu, a što pokazuju korišćene bilješke i bibliografija. Izjavljujem da nijedan dio rada nije napisan na nedozvoljen način, odnosno da nije prepisan iz necitiranog rada, te da nijedan dio rada ne ugrožava bilo čija autorska prava. U izradi rada pridržavao(la) sam se Pravilnika o diplomskom/specijalističkom radu.

U B.Luci, 09.10.2024

Student

(potpis)

APEIRON

za multidisciplinarne i virtuelne studije
Pan-European University for Multidisciplinary & Virtual Studies
Banja Luka

Ana Stanimirović
(Ime i prezime studenta)

IZJAVA O KORIŠTENJU AUTORSKOG DJELA

Izjavljujem i svojim potpisom dajem saglasnost Panevropskom univerzitetu „APEIRON", kao nosiocu prava iskorištavanja, da moj diplomski/specijalistički rad pod nazivom

Optimizacija skladištenja podataka u Big Data sistemima

(naslov rada)

koristi na način da ga, u svrhu stavljanja na raspolaganje javnosti, kao cjeloviti tekst ili u skraćenom obliku trajno objavi u javno dostupni repozitorijum Panevropskog univerziteta „APEIRON", a sve u skladu sa Zakonom o autorskom pravu i drugim srodnim pravima i dobrom akademskom praksom. Korištenje diplomskog/specijalističkog rada na navedeni način ustupam bez naknade.

U B.Luci, 09.10.2024

Student

(potpis)

SADRŽAJ

SAŽETAK

Ovaj diplomski rad istražuje metode i tehnike za optimizaciju skladištenja podataka u Big Data sistemima, sa posebnim fokusom na efikasno korišćenje resursa i poboljšanje performansi sistema. Analiziraju se ključni pristupi kao što su kompresija podataka, indeksiranje, raspodela podataka, i upotreba distribuiranih sistema skladištenja poput HDFS-a. Pored tehničkih aspekata, razmatraju se i pitanja bezbjednosti i privatnosti podataka u ovim sistemima. Cilj rada je da pruži pregled postojećih rešenja, identifikuje izazove i predloži preporuke za unapređenje praksi optimizacije skladištenja podataka u industrijskim aplikacijama. Rezultati ovog istraživanja mogu poslužiti kao vodič za stručnjake u oblasti upravljanja velikim podacima, omogućavajući im da donesu informisane odluke o implementaciji optimizovanih rešenja u svojim organizacijama.

Ključne riječi: Big Data, HDFS, skladištenje podataka, kompresija, indeksiranje

ABSTRACT

This paper investigates methods and techniques for optimizing data storage in Big Data systems, with a special focus on efficient use of resources and improving system performance. Key approaches such as data compression, indexing, data distribution and the use of distributed storage systems such as HDFS are analyzed. In addition to technical aspects, security and data privacy issues in these systems are also considered. The goal of the work is to provide an overview of existing solutions, identify challenges and suggestions for improving the practice of data storage optimization in industrial applications. The results of this research can serve as a guide for experts in the field of big data management, enabling them to make informed decisions about the implementation of optimized solutions in their organizations.

Keywords: Big Data, HDFS, data storage, compression, indexing

Uvod

U današnjem digitalnom dobu, količina generisanih podataka raste ogromnom brzinom, stvarajući ogromne izazove za tradicionalne sisteme skladištenja i obrade podataka. Big Data sistemi, koji se koriste za upravljanje ovim ogromnim količinama podataka, pružaju mogućnosti za skladištenje, obradu i analizu podataka na nivou koja je ranije bila nezamisliva. Međutim, kako se vremenom obim podataka znatno povećava, tako raste i potreba za optimizacijom skladištenja podataka unutar ovih sistema kako bi se osigurala efikasnost, skalabilnost i dostupnost podataka. Optimizacija skladištenja podataka u Big Data sistemima je ključna za smanjenje troškova skladištenja, poboljšanje performansi sistema i omogućavanje brze analize podataka. U ovom radu će biti istražene tehnike i pristupi koji omogućavaju efikasnije upravljanje skladišnim resursima, uključujući kompresiju podataka, indeksiranje podataka, kao i korišćenje distribuiranih sistema skladištenja kao što su Hadoop Distributed File System (HDFS).

Pored tehničkih aspekata, ovaj rad će se baviti i izazovima koji se odnose na bezbjednosti i privatnosti podataka u Big Data sistemima. Optimizacija skladištenja podataka mora biti sprovedena uz održavanje visokog nivoa sigurnosti i poštovanja pravila o zaštiti podataka. Cilj ovog rada je da pruži pregled trenutnih praksi u optimizaciji skladištenja podataka u Big Data sistemima, identifikuje izazove i mogućnosti za dalja istraživanja, te ponudi preporuke za implementaciju optimizovanih skladišnih rešenja u industrijskim okruženjima. Skladištenje podataka je predstavlja ključni aspekt uspješnog upravljanja u Big Data sistemima.

1. Big Data

Big Data se odnose na izuzetno velike i različite kolekcije strukturisanih, polustrukturisanih i nestrukturisanih podataka koji nastavljaju ogromnom brzinom rasti tokom vremena. Ovi skupovi podataka su toliko ogromni i složeni po obimu, brzini i raznolikosti da ih tradicionalni sistemi za upravljanje podacima ne mogu pohraniti, analizirati i obraditi. Količina i dostupnost podataka ubrzano raste, podstaknuti su napretkom digitalne tehnologije, kao što su Internet stvari (IoT) i vještačka inteligencija (AI). Kako podaci nastavljaju da se proširuju i umnožavaju, pojavljuju se novi alati za velike podatke koji pomažu kompanijama da prikupljaju, obrađuju i analiziraju podatke brzinom potrebnom da od njih dobiju najveću vrijednost.

Veliki podaci opisuju velike i raznolike skupove podataka koji imaju ogroman volumen i koji takođe brzo rastu u veličini tokom vremena. Veliki podaci se koriste u mašinskom učenju, prediktivnom modelovanju i drugoj naprednoj analitici za rješavanje poslovnih problema i donošenje informiranih odluka. Veliki podaci su raznoliki, složeni i velikog obima, stoga zahtijevaju niz tehnika i tehnologija s novim oblicima integracije kako bi se otkrili uvidi iz skupova podataka. Veliki podaci nisu samo brojevi, datum i nizovi. To takođe mogu biti geoprostorni podaci, 3D podaci, audio i video.

1.2 Pojam 'Big Data'

Za definiciju velikih podataka može se reći da su to podaci koji sadrže veću raznolikost, koji stižu u sve većim količinama i sa većom brzinom.

Jednostavno rečeno, veliki podaci su veći, složeniji skupovi podataka, posebno iz novih izvora podataka. Termin veliki podaci odnosi se na brzo rastuću zbirku podataka u strukturisanim, polustrukturisanim i nestrukturisanim formatima. Zbog svoje složene prirode, veliki podaci zahtijevaju moćne tehnologije,alate i napredne algoritme za upravljanje i analizu jer tradicionalni poslovni alati nisu efikasni za rad sa velikim podacima. Definicija velikih podataka je tema oko koje se vode mnoge polemike.

U 2010, Apache Hadoop definiše velike podatke kao "*Skup podataka koji ima veliki volumen, brzinu ili raznolikost, a tradicionalne metode su ograničene u njihovoj sposobnosti da ih efikasno analiziraju*." [1]

Na osnovu ove definicije, u maju 2011, McKinsey & Company (globalni konsultant organizacije) uveo je velike podatke kao „*Sledeću granicu inovacije, konkurenciju i produktivnost*." [1]

1.3 Karakteristike Big Data

Sa velikim podacima potrebno je napraviti odgovarajuće kategorizacije kako bi se bolje razumjeli. Kao rezultat toga, karakteristike velikih podataka mogu se okarakterisati sa pet “V”: volumen, raznolikost, brzina, vrijednost i vjerodostojnost. Ove karakteristike ne samo da pomažu u dešifrovanju velikih podataka, već takođe pružaju ideju o tome kako se nositi sa ogromnim podacima kontrolisanom brzinom u prihvatljivom vremenskom periodu kako bi se mogla izvući vrijednost iz njih i raditi analize u realnom vremenu. [2] [3]

Slika 1 5V model, preuzeto sa: : https://datavalley.ai/characteristics-of-big-data/

1.3.1 Volumen

Volumen se odnosi na količinu podataka koja postoji, koji se pohranjuju i obrađuju. Volumen je poput baze velikih podataka, jer je to početna veličina i količina podataka koji se prikupljaju. Ako je količina podataka dovoljno velika, može se smatrati velikim podacima. Ove ogromne količine podataka zahtijevaju korištenje napredne tehnologije obrade.

1.3.2 Raznolikost

Raznolikost podrazumijeva tipove podataka koji se razlikuju po formatu i načinu na koji su organizovani i spremni za obradu. Velika imena kao što su Facebook, Twitter, Pinterest, Google Ads, CRM sistemi proizvode podatke koji se mogu prikupljati, pohranjivati i naknadno analizirati. [3] Organizacija može dobiti podatke iz nekoliko izvora podataka, koji se mogu razlikovati po vrijednosti. Podaci mogu doći i iz izvora unutar i izvan preduzeća. Izazov u raznolikosti odnosi se na standardizaciju i distribuciju svih podataka koji se prikupljaju.

1.3.3 Brzina

Brzina se odnosi na to koliko brzo se podaci generišu i koliko brzo se kreću. Ovo je važan aspekt za organizacije kojima je potreban brz protok podataka, tako da su dostupni u pravom trenutku za donošenje najboljih mogućih poslovnih odluka. Na primjer, budući da se podaci sa senzorskih uređaja uvijek kreću u skladišta baze podataka, ta količina nije tako mala da bi se lako obrađivala.[2]

1.3.4 Vrijednost

Vrijednost se odnosi na prednosti koje veliki podaci mogu pružiti i direktno se odnosi na ono što organizacije mogu učiniti s tim prikupljenim podacima. Biti u stanju izvući vrijednost iz velikih podataka je uslov, jer vrijednost velikih podataka značajno raste zavisno o uvidima koji se mogu steći iz njih. Organizacije mogu koristiti alate za velike podatke za prikupljanje i analizu podataka, ali način na koji izvlače vrijednost iz tih podataka treba biti jedinstven za njih. Alati kao što je Apache Hadoop mogu

pomoći organizacijama da skladište, čiste i brzo obrađuju ovu ogromnu količinu podataka.

1.3.5 Vjerodostojnost

Vjerodostojnost se odnosi na kvalitet, tačnost, integritet i kredibilitet podataka. Prikupljeni podaci mogu imati nedostajuće dijelove, mogu biti netačni ili ne mogu pružiti pravi, vrijedan uvid. Istinitost, sveukupno, odnosi se na nivo povjerenja u prikupljene podatke. Ovo je posebno važno kada se radi sa podacima koji se ažuriraju u realnom vremenu. Autentičnost podataka zahtijeva provjere i ravnoteže na svakom nivou procesa prikupljanja i obrade velikih podataka.[3]

2. Vrste velikih podataka

Kako doba interneta nastavlja rasti, svake sekunde se generiše neshvatljiva količinu podataka. Toliko da se procjenjuje da će broj podataka koji plutaju internetom dostići 163 zetabajta do 2025. [4] Ovi podaci se mogu podijeljenji prema sledećim tipovima:

- **Strukturisani podaci**

Strukturisani podaci imaju određena unaprijed definisana organizacijska svojstva i prisutni su u strukturisanoj ili tabelarnoj šemi, što olakšava analizu i sortiranje. Zahvaljujući svojoj unaprijed definisanoj prirodi, svako polje je diskretno i može mu se pristupiti odvojeno ili zajedno sa podacima iz drugih polja. Ovo čini strukturisane podatke izuzetno vrijednim, što omogućava brzo prikupljanje podataka sa različitih lokacija u bazi podataka.

- **Nestrukturisani podaci**

Nestrukturisani podaci podrazumijevaju informacije bez unaprijed definisanih konceptualnih definicija i nisu lako interpretirani ili analizirani standardnim bazama

podataka ili modelima podataka. Nestrukturisani podaci čine većinu velikih podataka i sadrže informacije kao što su datumi, brojevi i činjenice. Primjeri velikih podataka ovog tipa uključuju video i audio datoteke, mobilne aktivnosti, satelitske slike i No-SQL baze podataka.

- **Polustrukturisani podaci**

Polustrukturisani podaci ne prate format tabelarnog modela podataka ili relacionih baza podataka jer nemaju fiksnu šemu. Međutim, podaci nisu potpuno sirovi ili nestrukturisani i sadrže neke strukturne elemente kao što su oznake i organizacijski metapodaci koji olakšavaju analizu. Prednosti polustrukturisanih podataka su u tome što su fleksibilniji i jednostavniji za skaliranje u poređenju sa strukturisanim podacima. Na primjer, JSON i XML su tipični primjeri polustrukturisanih podataka.

2.1 Glavne komponente velikih podataka

Za efikasno upravljanje i izdvajanje vrednosti iz velikih podataka, nekoliko međusobno povezanih komponenti imaju ključnu ulogu:

- **Izvori podataka**: Ovo predstavlja porijeklo velikih podataka, uključujući uređaje, platforme društvenih medija, web stranice i IoT senzore. Oni generišu neobrađene podatke koji se unose u proces analize.
- **Prikupljanje podataka**: To je proces prikupljanja podataka iz različitih izvora, osiguravanja njihove tačnosti i organizovanja za analizu. U zavisnosti od aplikacije, podaci se mogu prikupljati u paketu ili u procesima u realnom vremenu.
- **Skladištenje podataka**: Rješenja za pohranu velikih podataka prihvaćaju ogromnu količinu i raznolikost podataka. Ova rješenja uključuju tradicionalne relacine baze podataka, NoSQL baze podataka, i distribuirane sisteme datoteka kao što je Hadoop HDFS.

- **Obrada podataka**: Ova komponenta uključuje transformaciju sirovih podataka u smislene uvide. Tehnologije kao što su Apache Spark, Hadoop MapReduce i okviri za obradu podataka omogućavaju efikasnu analizu i manipulaciju podacima.
- **Analiza podataka**: Analitika velikih podataka uključuje primjenu različitih tehnika za otkrivanje obrazaca, trendova i korelacija unutar podataka. Uključuje deskriptivnu, dijagnostičku, prediktivnu i preskriptivnu analitiku, kao i mašinsko učenje i vještačku inteligenciju.
- **Vizualizacija podataka**: Efikasno komuniciranje složenih uvida je jedna ključnih komponenti . Alati za vizualizaciju podataka kao što su Tableau, Power BI pomažu u stvaranju vizualno privlačnih reprezentacija podataka za lakšu interpretaciju.

2.2 Prednosti velikih podataka

Brojne su prednosti Big Data za organizacije. Razumijevanje prednosti sigurnosti velikih podataka je od suštinskog značaja ako organizacija želi da iskoristi svoj potencijal. Neki od ključnih su sledeći:

1) Poboljšano donošenje odluka - Preduzeća mogu imati koristi od analitike velikih podataka tako što stiču vrijedne uvide i obrasce koji im mogu pomoći u donošenju odluka zasnovanih na informacijama i podacima. Analiza velikih količina podataka omogućava preduzećima da identifikuju trendove, preferencije kupaca, tržišne prilike i potencijalne rizike, što rezultira efikasnijim donošenjem odluka..
2) Poboljšana operativna efikasnost - Analitika velikih podataka može optimizovati poslovne operacije identifikovanje neefikasnosti, uskih grla i područja za poboljšanje. Preduzeća mogu pojednostaviti procese, smanjiti troškove i povećati produktivnost analizom podataka iz različitih izvora.
3) Poboljšano razumijevanje kupaca - Veliki podaci omogućavaju preduzećima da bolje razumiju svoje klijente. Kompanije mogu identifikovati obrasce, ponašanja kako bi personizovale marketinške kampanje, poboljšalo korisničko iskustvo i izgradile jače veze sa kupcima.

4) Ciljani marketing i oglašavanje - Analitika velikih podataka pomaže preduzećima da bolje usmjere marketinške napore analizirajući podatke o kupcima i tržišne trendove. To dovodi do personalizovanih kampanja, viših stopa konverzije i poboljšanog povrata ulaganja.
5) Konkurentska prednost - Korištenje analitike velikih podataka daje preduzećima konkurentsku prednost. Pomaže im da uoče tržišne trendove, prate konkurente i donesu strateške odluke. Uz to, veliki podaci otkrivaju nove tržišne izglede i podstiču inovacije proizvoda i usluga.
6) Upravljanje rizicima. Analitika velikih podataka pomaže preduzećima u identifikaciji i ublažavanju rizika. Kroz analizu podataka iz različitih izvora, kompanije mogu otkriti potencijalne prevare, sigurnosne prijetnje i operativne rizike i proaktivno ublažiti ove izazove.
7) Inovacija proizvoda i usluga. Veliki podaci imaju potencijal da podstaknu inovacije nudeći preduzećima vrijedan uvid u zahtjeve kupaca, dinamiku tržišta i tehnologije u nastajanju. Kroz analizu podataka, kompanije mogu odrediti tržišne prilike, kreirati nove proizvode i usluge i unaprijediti postojeće kako bi zadovoljile potrebe kupaca.

Sve u svemu, veliki podaci nude preduzećima potencijal da steknu vrijedne uvide, poboljšaju donošenje odluka, poboljšaju operativnu efikasnost i steknu konkurentsku prednost na tržištu.

2.3 Istorija velikih podataka

Porijeklo velikih skupova podataka seže u 1960-ih i 70-ih godina kada je svijet podataka tek počeo sa prvim podatkovnim centrima i razvojem relacione baze podataka. Oko 2005. godine ljudi su počeli shvaćati koliko podataka su korisnici generisali putem Facebooka, YouTube-a i drugih online servisa. Hadoop (okvir otvorenog koda kreiran posebno za skladištenje i analizu velikih skupova podataka) razvijen je iste godine. NoSQL je takođe počeo da dobija na popularnosti tokom ovog vremena. Razvoj okvira otvorenog koda, kao što je Hadoop i Spark bio je od suštinskog značaja za rast velikih podataka jer čine rad s velikim podacima lakšim i jeftinijim za skladištenje. U godinama od tada, obim velikih podataka je naglo skočio.

S pojavom Interneta stvari (IoT), više objekata i uređaja se povezuje na internet, prikupljajući podatke o obrascima korištenja korisnika i performansama proizvoda. Pojava mašinskog učenja proizvela je još više podataka.

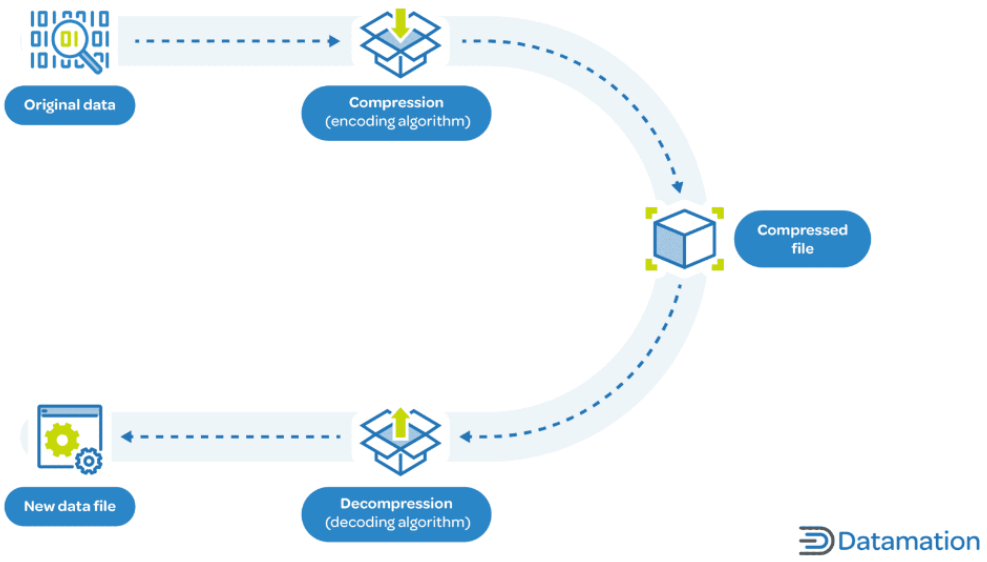
3. Metode i tehnike za optimizaciju skladištenja

3.1 Kompresija podataka

Kompresija podataka je proces korištenja kodiranja, restrukturisanja i drugih modifikacija za smanjenje veličine digitalnih datoteka podataka bez promjene njihovih osnovnih svojstava. Smanjenjem veličine datoteka, kompresija podataka smanjuje propusnost mreže potrebnu za njihovo dijeljenje i kapacitet potreban za njihovo pohranjivanje, smanjujući troškove. Na visokom nivou, kompresija podataka funkcioniše tako što kodira originalne, ciljne podatke u manje bitova, smanjujući veličinu podataka.

Kompresija prethodi digitalnoj tehnologiji, jer se koristila u Morzeovom kodu, koji je dodijelio najkraće kodove najčešćim znakovima. Danas, kada nekompresovana digitalna slika može zahtjevati 20 megabajta, kompresija podataka je važna za digitalno skladištenje informacija na računarskim diskovima i njihovo prenošenje preko komunikacionih mreža. Omjer kompresije mjeri koliko je datoteka ili skup podataka smanjen u veličini kroz kompresiju, obično izražen kao omjer ili postotak. Na primjer, ako je datoteka izvorno 1.000 kilobajta (KB), a nakon kompresije postane 500 KB, omjer kompresije je 50%.

Osnovni principi kompresije podataka su postizanje smanjenja veličine datoteke efikasnijim kodiranjem podataka. Kompresija može biti bez gubitka ili sa gubitkom. RLE, LZW, aritmetičko kodiranje, Huffman kodiranje, Shannon-Fano kodiranje su popularni algoritmi bez gubitaka. Neki od algoritama sa gubicima su kodiranje transformacije, DCT, DWT, fraktalna kompresija, segmentacija pravougaonika i RSSMS.



Slika 2 Kompresija podataka , preuzeto sa https://www.datamation.com/big-data/data-compression/

3.1.1 Vrste kompresije podataka

Postoje četiri različite vrste kompresije podataka:

- Kompresija teksta—primarno koristi kodove ili simbole za obrasce i redundancije; smanjuje ukupnu veličinu, ali informacije ostaju nepromijenjene.
- Kompresija zvuka—eliminiše šum radi smanjenja veličine fajla, ali smanjuje ukupni kvalitet.
- Kompresija slike—slično kompresiji teksta; zamjenjuje ponovljene uzorke boja koristeći kodove za smanjenje veličine.
- Video kompresija—kombinacija kompresije zvuka i slike, uklanja neželjene slike i pozadinsku buku kako bi se smanjila ukupna veličina nauštrb kvaliteta.

Algoritmi kompresije podataka nalaze primjenu u različitim poljima:

- Kompresija datoteka: Algoritmi kompresije podataka se obično koriste za kompresiju datoteka, smanjujući njihovu veličinu za efikasno skladištenje i prijenos.
- Kompresija baze podataka: Tehnike kompresije se koriste za smanjenje zahtjeva za skladištenjem baza podataka, omogućavajući efikasno upravljanje podacima.
- Kompresija slike i videa: Algoritmi kompresije sa gubitkom se uveliko koriste u obradi slike i videa kako bi se postigli visoki omjeri kompresije uz održavanje prihvatljivog vizualnog kvaliteta.
- Mrežni prenos: Algoritmi kompresije igraju ključnu ulogu u efikasnom prenosu podataka preko mreža, optimizujući korištenje propusnog opsega i smanjujući vrijeme prijenosa podataka.
- Analitika velikih podataka: Kompresovani podaci omogućavaju bržu obradu i analizu velikih skupova podataka, poboljšavajući performanse i skalabilnost platformi za analizu velikih podataka.

3.1.2 Tehnike kompresije podataka

Uopšteno govoreći, postoje dva sveobuhvatna pristupa kompresiji podataka. Svaki je na svoj način bolje prilagođen određenim aplikacijama i tipovima podataka zavisno o željenom rezultatu.

- **Kompresija bez gubitaka (Lossless Compression)**

Kompresija bez gubitaka je tehnika kompresije koja ne gubi nikakve podatke u procesu kompresije. Korišćenjem ove tehnike, nikakve informacije se zapravo ne uklanjaju. Kompresija bez gubitaka često će imati manji omjer kompresije, uz prednost da se ne izgube podaci u datoteci. Ovo je često veoma važno kada je potrebno da se održi apsolutni kvalitet, kao što su informacije iz baze podataka ili profesionalni medijski fajlovi. Formati kao što su ZIP, GIF, PDF, FLAC i PNG nude opcije kompresije bez gubitaka.

- **Kompresija sa gubitkom(Lossy Compression)**

Smanjuje veličinu brisanjem nepotrebnih informacija i smanjenjem složenosti postojećih informacija. Kompresija sa gubitkom može postići mnogo veće omjere kompresije, po cijenu moguće degradacije kvaliteta datoteke. JPEG nudi opcije kompresije sa gubicima, a MP3 se zasniva na kompresiji sa gubicima. Uobičajene primjene kompresije sa gubicima su multimedijalne datoteke kao što su audio, fotografije, grafika i video zapisi.

3.1.3 Algoritmi kompresije podataka

Kompresija podataka se oslanja na širok spektar algoritama za rad. Neki od najčešćih su:

Run Length Encoding (RLE) - Jednostavan oblik kompresije podataka koji je posebno efikasan za podatke sa dugim sekvencama iste vrijednosti. Zamjenjuje ove sekvence jednom vrijednošću i brojem, smanjujući redundantnost i veličinu datoteke.Ova metoda bez gubitaka smanjuje broj bitova koji se koriste u predstavljanju podataka. Ukupna veličina je smanjena, ali nijedna informacija se ne gubi. Na primjer, ako skup podataka uključuje nekoliko ponovljenih znakova—kao što je "aaaabbbbcccddee", RLE algoritam ga kodira kao "4a4b3c2de". Iste informacije su dostupne u manjem broju bajtova, ali niz podataka ostaje nepromijenjen.

Huffman Coding - Još jedan algoritam bez gubitaka, koji se prije svega koristi za skupove podataka koji se sastoje od znakova koji se često pojavljuju. Generiše jedinstveni kod za svaki znak na osnovu frekvencije – kada se niz predstavlja pomoću ovih kodova, ukupna veličina se smanjuje, ali podaci ostaju nepromenjeni.

Lempel-Ziv algoritam - Algoritam bez gubitaka koji se široko koristi za GIF i TIFF formate. Budući da kodovi zauzimaju manje prostora, ukupna veličina podataka je smanjena.

LZSS (Lempel-Ziv-Storer-Szymanski) algoritam - Ovaj algoritam bez gubitaka koristi princip tekstualne zamjene zasnovan na tehnici kodiranja rječnika. Prvo zamjenjuje niz simbola koristeći referencu. Zatim uklanja duple podatke i osigurava da je nova veličina datoteke manja od originalne. LZSS se može lako implementovati i široko se koristi za kompresiju GIF, TIFF, PDF i tekstualnih datoteka.

DEFLATE - Kombinacija LZSS i Huffman algoritama kodiranja, ova tehnika bez gubitaka je prvobitno razvijena za ZIP datoteke, ali se sada koristi i za gzip u HTTP kompresiji i PNG formatima. Radi tako što pronalazi ponovljene nizove znakova i kodira ih na osnovu njihove frekvencije. Format kompresovanih podataka DEFLATE sastoji se od niza blokova, koji odgovaraju uzastopnim blokovima ulaznih podataka. Svaki blok je kompresovan upotrebom kombinacije LZ77 algoritma i Huffmanovog kodiranja. LZ77 algoritam pronalazi ponovljene podnizove i zamjenjuje ih referencama unazad (relativne udaljenosti odstupanja). LZ77 algoritam može koristiti referencu na duplikovani niz koji se javlja u istim ili prethodnim blokovima, do 32K ulaznih bajtova unazad.

BZIP2 - je visoka stopa kompresije sa relativno velikom brzinom. Većina datoteka je kompresovana na manju veličinu datoteke koja je moguća s tradicionalnijim gzip i zip programima. Na primjer, ova metoda je kompresija bez gubitaka, da se nikakvi podaci ne gube tokom kompresije i tako se originalni fajlovi tačno regenerišu. Jedini nedostatak bzip2 je da je sporiji od gzipa i zip-a.

BZIP2 sintaksa je: bzip2 [option(s)] file_name(s)

Funkcija kompresije je: bzip2 file1 file2 file3

Gzip GZ format datoteke je kompresuje svaki pojedinačni fajl, ali ne može podržati više ulaznih datoteka u jednu kompresovanu arhivu kontejnera. Iz tog razloga se obično koristi u Unix/Linux svijetu za kompresovanje TAR arhivskog formata arhive koji može podržati više datoteka u jednom paketu i sačuvati informacije o sistemu datoteka (atributi datoteka, datum/vrijeme), ali nema mogućnosti kompresije za proizvodnju TGZ datoteke (ako se koristi konvencija o proširenju jedne datoteke).

3.1.4 Najbolje prakse za kompresiju podataka

Smanjenje veličine datoteke samo je jedan aspekt efikasne kompresije podataka. Kompresija takođe mora sačuvati performanse i integritet podataka i osigurati da su podaci i dalje dostupni kada je potrebno. Nekoliko najboljih praksi kako bi se osigurala sigurna i učinkovita kompresiju podataka:

1. Odrediti nivo kompresije

Procijeniti potreban nivo kompresije prije primjene kompresije. Sa nekim formatima datoteka, potrebno je težiti maksimalnoj kompresiji. Za druge, može se dati prioritet očuvanju integriteta podataka. Pronalaženje prave ravnoteže je ključ za postizanje najboljih rezultata koji zadovoljavaju određene poslovne potrebe. Na primjer, posao koji se bavi tekstualnim dokumentima, uključujući pravne ugovore i finansijske izvještaje, za ove dokumente integritet podataka je najvažniji. Korištenje nižeg nivoa kompresije ili čak kompresije bez gubitaka osigurava da se informacije ne izgube tokom procesa kompresije. Sa druge strane, ako preduzeće arhivira stare e-poruke ili manje kritične dokumente, mogu se primijeniti viši nivoi kompresije kako bi se uštedio prostor za skladištenje bez značajnog gubitka podataka.

2. Vrsta kompresije

Potrebno je odabrati odgovarajuću metodu kompresije na osnovu tipova podataka. Kompresija bez gubitaka je norma za tekstualne i programske datoteke, dok multimedijalne i video datoteke često koriste kompresiju sa gubicima. Na primjer, medijska kompanija pohranjuje ogromnu biblioteku video datoteka visoke definicije. Za distribuciju i online streaming, koristi tehnike kompresije sa gubitkom kako bi smanjio veličinu datoteka uz održavanje prihvatljivog nivoa kvaliteta videa. Ali kompanija koristi kompresiju bez gubitaka za svoju projektnu dokumentaciju, osiguravajući da se podaci ne izgube u procesu arhiviranja.

3. Razmotriti deduplikaciju

Deduplikacija podataka je tehnika koja identifikuje i eliminiše duplikovane podatke unutar skupa podataka kako bi se smanjila veličina podataka. Ovo dodatno smanjuje potrebu za dodatnim uređajima za pohranu podataka. Deduplikacija se može primijeniti prije ili nakon kompresije, zavisno o specifičnoj strategiji upravljanja podacima. Najkorisniji je u poslovnim rješenjima za pohranu gdje je redundantnost podataka uobičajna.

4. Pratiti efikasnost kompresije

Redovno praćenje performansi određenih procesa kompresije podataka je od suštinskog značaja kako bi se osiguralo da oni postižu željene rezultate bez gubitka kvaliteta podataka. Ovo može uključivati periodične procjene omjera kompresije, veličinu datoteka i brzine dekompresije. Na primjer, pružalac usluga skladištenja u cloud-u kontinuirano prati efikasnost svojih algoritama kompresije. Ako kompanija primijeti da određena vrsta podataka nije dobro kompresovana, može prilagoditi postavke kompresije ili istražiti alternativne algoritme kako bi optimizirala proces. Ova stalna evaluacija osigurava da kompresija podataka ostane efikasna i efektivna.

5. Optimizacija prema zadatku

Različiti zadaci ili projekti mogu zahtijevati različite metode kompresije, zavisno o veličini i vrsti podataka. Prilagođavanje vaše strategije kompresije specifičnim potrebama određenog projekta je od suštinskog značaja za postizanje najboljih rezultata.

6. Sigurnosna kopija originalnih podataka

Preporučljivo je uvijek čuvati sigurnosnu kopiju originalnih, nekompresovanih podataka. Ovo osigurava da postoji čista kopija na koju se može osloniti u slučaju gubitka ili oštećenja podataka tokom kompresije ili dekompresije. Redovne sigurnosne kopije originalnih podataka prestavljaju ključnu zaštitnu mjeru.

3.2 Indeksiranje podataka

Rast podataka i akumulacija složenih podataka zbirke su postale izazov za pronalaženje informacija. Rješenje za ovo je u izgradnji indeksa nad skupovima podataka. Uopšteno govoreći, indeksi su lista oznaka, imena, predmeta i grupe stavki koja upućuje na to gdje će se određene stavke pojaviti. Strategija indeksiranja je dizajn metoda pristupa traženoj stavci, ili jednostavnije rečeno, indeksu. To takođe opisuje kako su podaci organizovani u sistemu skladištenja kako bi se olakšalo pronalaženje određenih informacija. Ideja indeksiranja velikih podataka je da fragmentira skupove podataka prema kriterijumima koji će se često koristiti u upitima. Fragmenti su indeksirani tako da svaki sadrži vrijednost koja zadovoljava neke predikate upita. Indeksiranje ima za cilj skladištenje podataka na organizovaniji način, čime se olakšava pronalaženje informacija.

Složeni podaci se prikupljaju sa metapodacima koji opisuju njihov sadržaj. Takvi skupovi podataka mogu se ispitivati korištenjem metapodataka sadržaja. Umjesto pretraživanja cijele baze podataka što može oduzeti dosta vremena, efikasniji pristup je pretraživanje odgovarajućih grupa koje se odnose na upit. Ovo doprinosi smanjenju vremena u pronalaženja informacija, budući da proces pretraživanja uzima u obzir samo sadržaj određene grupe.

Da bi se olakšalo pronalaženje informacija, prikladna strategija indeksiranja mora se primijeniti na skupove podataka tokom obrade. Ovo takođe dolazi sa prednostima organizovanog sistema skladištenja koji olakšava pretragu i pronalaženje informacija. Prednosti ovoga su visoka propusnost za obradu podataka, smanjeno vrijeme pristupa za upite, replikacija podataka koja doprinosi povećanom dostupnošću i pouzdanošću, te skalabilnost strukture.

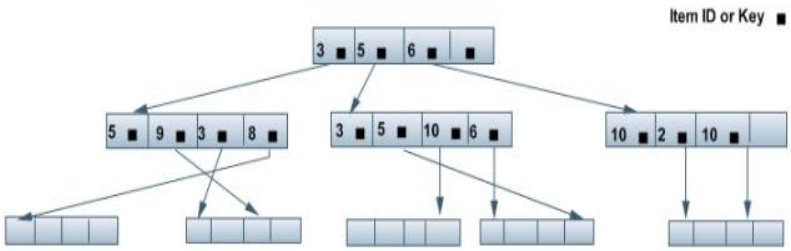
Dizajn metode pristupa ili vrste strategije indeksiranja koja će se koristiti u obradi određenog skupa podataka zavisi o vrsti upita koji će se izvršiti na skupu podataka, kao što su upiti sličnosti (pretraga najbližeg člana), upiti ključnih riječi i ad-hoc upit. Stoga, programer mora biti svjestan vrste podataka koje treba indeksirati (npr. evidencije, e-pošta, audio, video ili slike) i vrste upita koji će se izvršiti na indeksima.

3.2.1 Strategije indeksiranja zasnovane na stablu

Strukture indeksiranja stabla su B-stablo, R-stablo, Xtree, i mnogi drugi. U strategiji indeksiranja stabla, dohvaćanje podataka se vrši sortiranim redoslijedom, slijedeći odnose grananja stavke podataka. Ovo zadovoljava upite najbližeg člana. Strategije indeksiranja zasnovane na stablu su:

1) **B-stablo**: B-stablo radi kao pretraga binarnog stabla, ali na složeniji način. To je zato što čvorovi B-stabla imaju mnogo grana, za razliku od binarnog stabla koje ima dvije grane po čvoru. To znači da je B-stablo komplikovanije od binarnog stabla. Indeksi B-stabla zadovoljavaju upite opsega i upite sličnosti poznatih i kao pretraga najbližeg člana (NNS[1]), koristeći operatore poređenja (<, < =, =, >, > =).

U B-stablu, ključevi i svi zapisi se obično pohranjuju u listove, ali kopije (ključa) se pohranjuju u interne čvorove kao što je prikazano na slici ispod.

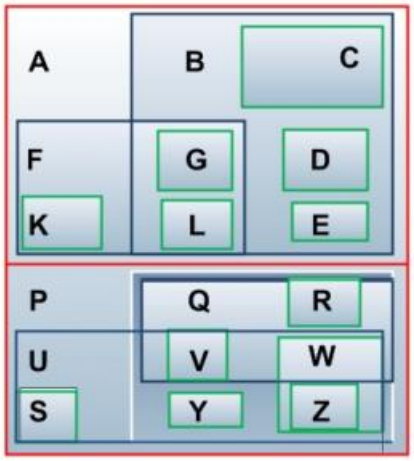


Slika 3 B-stablo , preuzeto sa https://www.slac.stanford.edu/pubs/slacpubs/16250/slac-pub-16460.pdf

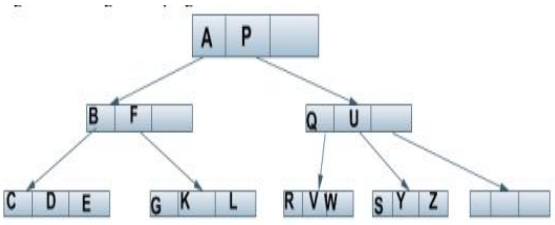
Istraživanja su pokazala da ova strategija nije uvijek brza pri pretraživanju velikih podataka i može potrošiti prostor za skladištenje jer čvorovi nisu uvijek puni. B-stablo se linearno skalira, ali je pogodno samo za jednodimenzionalni metod pristupa za razliku od drugih metoda pristupa zasnovanih na stablu ili strategija indeksiranja kao što je R-stablo. Takođe, algoritam B-stabla troši ogromne računarske resurse kada vrši indeksiranje velikih podataka.

[1] NNS(Nearest Neighbor Search) – Pretraga najbližeg člana, uključuje pronalaženje najbližih tačaka podataka datoj tački upita. Ova metoda pretraživanja služi kao osnovni alat u analizi podataka, omogućavajući efikasno pronalaženje sličnih tačaka podataka.

2) **R-stablo**: Ovo je strategija indeksiranja koja se koristi za prostorne ili upite opsega. Uglavnom se primjenjuje u geoprostornim sistemima gdje svaki unos ima koordinate X i Y sa minimalnim i maksimalnim vrijednostima. Prednost upotrebe R-stabla u odnosu na B-stablo je u tome što R-stablo zadovoljava višedimenzionalne upite ili upite opsega, dok B-stablo ne. S obzirom na raspon upita, korištenje R-stabla čini pronalaženje odgovora na upite brzim. Primjer je pronalaženje svih hostela unutar određenog kampusa ili pronalaženje svih hotela unutar određenog kilometra od određene lokacije. Ideja je grupisati stavke podataka prema njihovoj udaljenosti jedna od druge i dodijeliti im minimalne i maksimalne granice. Svaki zapis na lisnom čvoru opisuje jednu stavku (sa minimalnim i maksimalnim vrijednostima). Svaki unutrašnji čvor opisuje kolekciju stavki ili objekata kao što je prikazano na slikama 4 i 5.



Slika 4 Grupisanje opsega u R stablu, preuzeto sa: https://www.slac.stanford.edu/pubs/slacpubs/16250/slac-pub-16460.pdf



Slika 5 Indeksirasnje R stabla, izvor: https://www.slac.stanford.edu/pubs/slacpubs/16250/slac-pub-16460.pdf

3.2.2 Bitmap indeksi

Bitmap indeksiranje je tehnika indeksiranja podataka koja se koristi u sistemima za upravljanje bazama podataka za poboljšanje performansi upita samo za čitanje koji uključuju velike skupove podataka. To uključuje kreiranje bitmap indeksa, koji je struktura podataka koja predstavlja prisustvo ili odsustvo vrijednosti podataka u tabeli ili koloni. U bitmap indeksima svakoj različitoj vrijednosti u koloni dodjeljuje se vektor bita koji predstavlja prisustvo ili odsustvo te vrijednosti u svakom redu tabele. Vektor bita sadrži po jedan bit za svaki red u tabeli, gde postavljeni bit ukazuje na prisustvo odgovarajuće vrednosti u redu, a obrisani bit ukazuje na odsustvo vrednosti.

Bitmap indeks funkcioniše u osnovi u situacijama gdje se vrijednosti indeksa vrlo često ponavljaju za razliku od drugih tipova indeksa koji se obično koriste. Drugi tipovi su najefikasniji kada se indeksirane vrijednosti uopšte ne ponavljaju ili se ponavljaju manji broj puta. Polje roda u tablici baze podataka je dobar primjer za bitmap indeks. Ovo je tako, jer bez obzira na to koliko tuple ima u tabeli baze podataka, polje će imati samo 2 moguće vrijednosti: muško ili žensko. Struktura podataka bitmap indeksa prikazana je na slici 6.

Table

Record	Gender
Row 1	M
Row 2	F
Row 3	M
Row 4	F
Row 5	M
Row 6	F
Row 7	M

Bitmap Index

Gender	Row 1	Row 2	Row 3	Row 4	Row 5	Row 6	Row 7
Male	1	0	1	0	1	0	1
Female	0	1	0	1	0	1	0

Slika 6 Bitmap indeksi, izvor: https://www.researchgate.net/publication/350574309_Indexing_in_Big_Data_Mining_and_Analytics

4. Apache Hadoop

Apache Hadoop je softverska platforma otvorenog koda, zasnovana je na programskom jeziku Java, koja upravlja obradom i skladištenjem podataka za aplikacije velikih podataka. Platforma radi tako što vrši distribuciju Hadoop velike podatke i poslovne analitike preko čvorova koji se nalaze u računarskom klasteru, djeljeći ih na manja opterećenja koja se mogu izvoditi paralelno.

Neke ključne prednosti Hadoop-a su skalabilnost, otpornost i fleksibilnost. Hadoop Distributed File System (HDFS) pruža pouzdanost i otpornost replikacijom bilo kog čvora klastera na druge čvorove klastera radi zaštite od hardverskih ili softverskih kvarova. Hadoop fleksibilnost omogućava skladištenje bilo kojeg formata podataka uključujući strukturirane i nestrukturirane podatke. Hadoop je softverski ekosistem koji omogućava preduzećima da upravljaju ogromnim količinama podataka u veoma kratkom vremenskom periodu.

Apache Hadoop je nastao iz potrebe da se obrađuju sve veće količine velikih podataka i brže isporučuju web rezultati kao što su pretrage poput Yahooa i Google-a počinjale. Inspirisani Google MapReduceom, modelom programiranja koji dijeli aplikaciju na male djeliće za pokretanje na različitim čvorovima, Doug Cutting i Mike Cafarella pokrenuli su Hadoop 2002. dok su radili na projektu „*Apache Nutch*“.[13] Nekoliko godina kasnije, Hadoop je odvojen od Nutcha. Nutch se fokusirao na element web pretraživača, a Hadoop je postao dio distribuiranog računanja i obrade. Dvije godine nakon što se Cutting pridružio Yahoou, Yahoo je 2008. godine objavio Hadoop kao projekat otvorenog koda. [13] Apache Software Foundation (ASF) je učinio Hadoop dostupnim javnosti u novembru 2012. kao „*Apache Hadoop*“.

Sa uvođenjem Hadoop-a, organizacije su brzo imale pristup mogućnosti skladištenja i obrade ogromnih količina podataka, te da povećanju računarske snage, toleranciju grešaka, fleksibilnosti u upravljanju podacima, nižim troškovima u poređenju sa DW-ovima i većoj skalabilnosti. Na kraju, Hadoop je otvorio put za budući razvoj analitike velikih podataka, poput uvođenja „*Apache Spark-a*“.

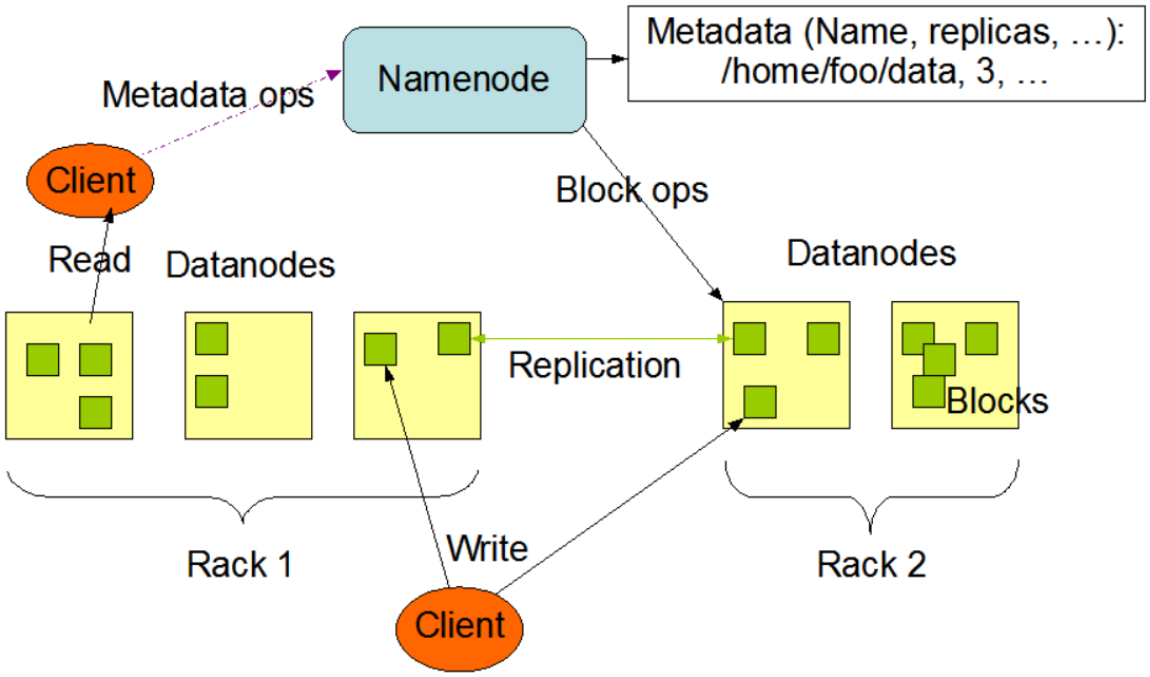
Nekoliko ključnih komponenti koje čine Apache Hadoop ekosistem su: *HDFS,MapReduce i YARN.*

4.1 HDFS

Hadoop distribuirani sistem datoteka(HDFS) je sistem koji omogućava skladištenje velikih skupova podataka na čvorovima u klasteru na način koji je otporan na greške. Poput desktopa, HDFS koristi hijerarhijski prostor imena sistema datoteka koji koristi strukturu direktorija za organiziranje datoteka pohranjenih u HDFS klasteru. Za razliku od desktopa, HDFS ne pohranjuje datoteku na jedno mjesto, umjesto toga razbija je na niz blokova pohranjenih na diskovima, serverima i klasterima.

Hadoop distribuirani sistem datoteka je mjesto gdje počinje i završava se skladište svih podataka. Ova komponenta upravlja velikim skupovima podataka u različitim strukturisanim i nestrukturisanim čvorovima podataka. Istovremeno, održava metapodatke u obliku log fajlova. Postoje dvije sekundarne komponente HDFS-a: *NameNode* i *DataNode*.

HDFS klaster se sastoji od jednog NameNodea, glavnog servera koji upravlja imenskim prostorom sistema datoteka i reguliše pristup datotekama od strane klijenata. Pored toga, postoji određeni broj DataNodes-a, obično jedan po čvoru u klasteru, koji upravljaju skladištem povezanim sa čvorovima na kojima se nalaze. HDFS izlaže prostor imena sistema datoteka i dozvoljava pohranjivanje korisničkih podataka u datoteke. Datoteka se dijeli na jedan ili više blokova i ti blokovi se pohranjuju u skup DataNodes. NameNode izvršava operacije imenskog prostora sistema datoteka kao što su otvaranje, zatvaranje i preimenovanje datoteka i direktorija. Takođe određuje mapiranje blokova u DataNodes. DataNodes su odgovorni za opsluživanje zahtjeva za čitanje i pisanje od klijenata sistema datoteka. DataNodes takođe izvode kreiranje blokova, brisanje i replikaciju prema instrukciji od NameNode.



Slika 7 Arhitektura HDFS , izvor: https://hadoop.apache.org/docs/stable/hadoop-project-dist/hadoop-hdfs/HdfsDesign.html#NameNode_and_DataNodes

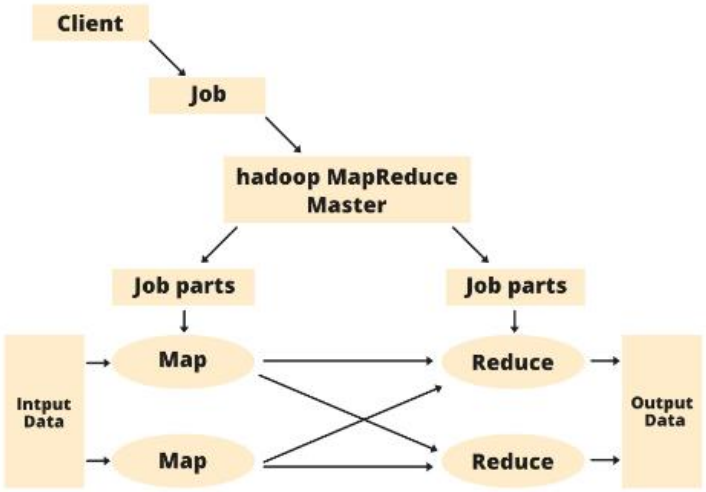
NameNode i DataNode su komadi softvera dizajnirani za rad na standardnim mašinama. Ove mašine obično koriste GNU/Linux operativni sistem (OS). Obzirom na to da je HDFS je napravljen korišćenjem jezika Java i bilo koja mašina koja podržava Javu može pokrenuti NameNode ili DataNode softver. Upotreba veoma popularnog Java jezika znači da se HDFS može postaviti na širok spektar mašina. Tipična implementacija ima namjensku mašinu koja pokreće samo softver NameNode. Svaka od ostalih mašina u klasteru pokreće jednu instancu softvera DataNode.

Postojanje jednog NameNode u klasteru značajno pojednostavljuje arhitekturu sistema. NameNode je i spremište za sve HDFS metapodatke. Sistem je dizajniran na takav način da korisnički podaci nikada ne prolaze kroz NameNode.

4.2 MapReduce

MapReduce je i model programiranja i mašina za obradu velikih podataka koja se koristi za paralelnu obradu velikih skupova podataka. Na početku MapReduce je bio jedini mehanizam za izvršavanje dostupan u Hadoop-u ali kasnije je Hadoop dodao podršku za druge, uključujući Apache Tez i Apache Spark. Hadoop MapReduce je osnovna komponenta za obradu Hadoop ekosistema. Ovaj softver pruža jednostavan okvir za pisanje aplikacija kada je u pitanju rukovanje ogromnim količinama strukturiranih i nestrukturiranih podataka.

MapReduce upravlja planiranjem poslova od strane klijenta. Zadaci koje zahtijeva korisnik podijeljeni su na nezavisne zadatke i procese. Klijent će poslati posao određene veličine Hadoop MapReduce Masteru. Nakon toga će MapReduce master podijeliti ovaj posao na daljnje ekvivalentne dijelove posla. Ovi dijelovi posla se zatim stavljaju na raspolaganje za Map i Reduce zadatak. Ovaj zadatak Map i Reduce će sadržati program u skladu sa zahtjevima slučaja upotrebe koji određena kompanija rješava. Programer piše svoju logiku da ispuni zahtjeve koji industrija zahtijeva. Ulazni podaci koji se koriste se zatim unose u zadatak mape i mapa će generisati srednji par ključ-vrijednost kao svoj izlaz. Izlaz Mape, tj. ovi parovi ključ/vrijednost se zatim unose u Reducer, a konačni izlaz se pohranjuje na HDFS. Može biti n broj zadataka Map i Reduce koji su dostupni za obradu podataka prema zahtjevu. Algoritam za Map and Reduce je napravljen na vrlo optimizovan način.



Slika 8 MapReduce Arhitektura, izvor: https://www.geeksforgeeks.org/mapreduce-

Faze MapReduce:

1. Mapa: Glavna upotreba je mapiranje ulaznih podataka u parovima ključ/vrijednost. Ulaz u mapu može biti par ključ-vrijednost gdje ključ može biti ID neke vrste adrese, a vrijednost je stvarna vrijednost koju čuva. Funkcija Map() će se izvršiti u svom memorijskom skladištu na svakom od ovih ulaznih parova ključ/vrijednost i generiše srednji par ključ/vrijednost koji radi kao ulaz za funkciju Reducer ili Reduce().

2. Reduce: Srednji parovi ključ/vrijednost koji rade kao ulaz za Reducer se miješaju i sortiraju i šalju funkciji Reduce(). Reducer grupiše podatke na osnovu svog para ključ/vrijednost prema algoritmu koji je napisao programer.

3. Job Tracker: Posao Job tracker-a je da upravlja svim resursima i svim poslovima u klasteru, kao i da zakaže svaku mapu na Task Tracker-u koja radi na istom čvoru podataka jer u klasteru može biti na stotine dostupnih čvorova podataka.

4. Task Tracker: Task Tracker se može smatrati stvarnim robovima koji rade na instrukciji koju daje Job Tracker. Ovaj Task Tracker je raspoređen na svakom od dostupnih čvorova u klasteru koji izvršava zadatak Map and Reduce prema uputama Job Tracker-a.

4.3 YARN

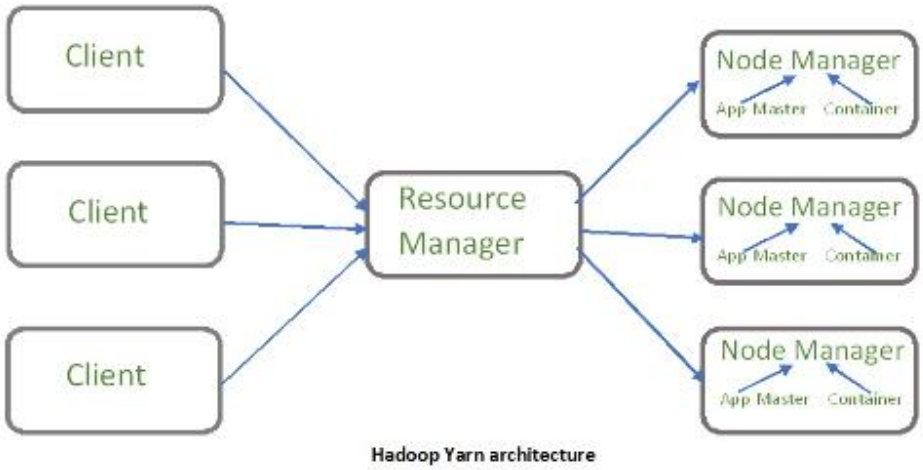
YARN (Yet Another Resource Negotiator) je kritična komponenta Hadoop ekosistema. Funkcioniše kao sloj upravljanja resursima klastera, odgovoran za upravljanje i dodjelu resursa kao što su CPU, memorija i skladište za distribuirane aplikacije koje rade na Hadoop klasteru. YARN se takodje koristi i za, zadatke planiranja poslova koji se izvode na Hadoop-u. YARN omogućava programerima da pokrenu onoliko aplikacija koliko je potrebno na istom klasteru. Pruža sigurnu i stabilnu osnovu za operativno upravljanje i dijeljenje sistemskih resursa za maksimalnu efikasnost i fleksibilnost.

Uveden je u Hadoop 2.0 kako bi se uklonilo usko grlo na Job Tracker-u koje je bilo prisutno u Hadoop-u 1.0. YARN je opisan kao „Redizajnirani menadžer resursa“ u vrijeme svog pokretanja, ali je sada evoluirao da bude poznat kao distribuirani operativni sistem velikih razmjera koji se koristi za obradu velikih podataka.

YARN arhitektura u osnovi odvaja sloj upravljanja resursima od sloja obrade. U Hadoop 1.0 verziji, odgovornost Job tracker-a je podijeljena između upravitelja resursa i upravitelja aplikacija YARN takođe dozvoljava različitim mašinama za obradu podataka kao što su obrada grafova, interaktivna obrada, obrada toka kao i grupna obrada za pokretanje i obradu podataka pohranjenih u HDFS (Hadoop Distributed File System) čime sistem čini mnogo efikasnijim. Kroz svoje različite komponente, može dinamički alocirati različite resurse i planirati obradu aplikacije. Za obradu velikih količina podataka, potrebno je pravilno upravljati dostupnim resursima kako bi ih svaka aplikacija mogla iskoristiti.

YARN je stekao popularnost zbog sledećih karakteristika:

- Skalabilnost: Planer u upravitelju resursa YARN arhitekture omogućava Hadoop-u da proširi i upravlja hiljadama čvorova i klastera.
- Kompatibilnost: YARN podržava postojeće aplikacije za smanjenje mapa bez prekida, što ga čini kompatibilnim i sa Hadoop 1.0.
- Korišćenje klastera: YARN podržava dinamičko korišćenje klastera u Hadoop-u, to omogućava optimizovano korišćenje klastera.

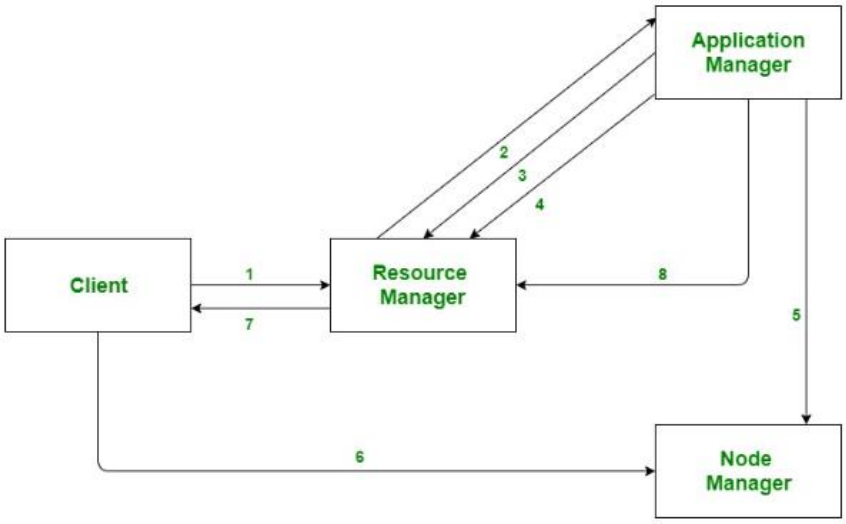


Slika 9 Hadoop Yarn arhitektura, izvor: https://www.geeksforgeeks.org/hadoop-yarn-architecture/

Glavne komponente YARN arhitekture uključuju:

- Klijent: Podnosi poslove smanjenja mape.
- Resource Manager: To je glavni proces YARN-a i odgovoran je za dodjelu resursa i upravljanje svim aplikacijama. Kad god primi zahtjev za obradu, prosljeđuje ga odgovarajućem upravitelju čvora i u skladu s tim dodjeljuje resurse za završetak zahtjeva.
- Planer: Izvodi zakazivanje na osnovu dodijeljene aplikacije i raspoloživih resursa. To je čisti planer, što znači da ne obavlja druge zadatke kao što su praćenje i ne garantuje ponovno pokretanje ako zadatak ne uspije. YARN planer podržava dodatke kao što su Capacity Scheduler i Fair Scheduler za particioniranje resursa klastera.
- Upravitelj aplikacija: Odgovoran je za prihvaćanje aplikacije i pregovaranje o prvom kontejneru od upravitelja resursa. Također ponovo pokreće Application Master kontejner ako zadatak ne uspije.
- Node manager: brine o pojedinačnom čvoru na Hadoop klasteru i upravlja aplikacijom i radnim tokom i tim određenim čvorom. Njegov primarni zadatak je da prati Resource Manager. Nadzire korištenje resursa, vrši upravljanje dnevnikom. Također je odgovoran za kreiranje procesa kontejnera i njegovo pokretanje na zahtjev Master aplikacije.

- Master aplikacije: Aplikacija je jedan posao koji se podnosi okviru. Master aplikacije je odgovoran za pregovaranje o resursima sa upraviteljem resursa, praćenje statusa i praćenje napretka jedne aplikacije. Master aplikacije traži kontejner od upravitelja čvorova slanjem konteksta za pokretanje kontejnera (CLC) koji uključuje sve što je potrebno aplikaciji za pokretanje. Jednom kada se aplikacija pokrene, s vremena na vrijeme šalje izvještaj o stanju aplikacije upravitelju resursa.
- Kontejner: To je zbirka fizičkih resursa kao što su RAM, CPU jezgra i disk na jednom čvoru. Kontejneri se pozivaju pomoću Container Launch Context (CLC) koji je zapis koji sadrži informacije kao što su varijable okruženja, sigurnosni tokeni.



Slika 10 Tok rada aplikacije u Hadoop YARN okruženju, izvor: https://www.geeksforgeeks.org/hadoop-yarn-architecture/

Tok rada aplikacije u Hadoop YARN:

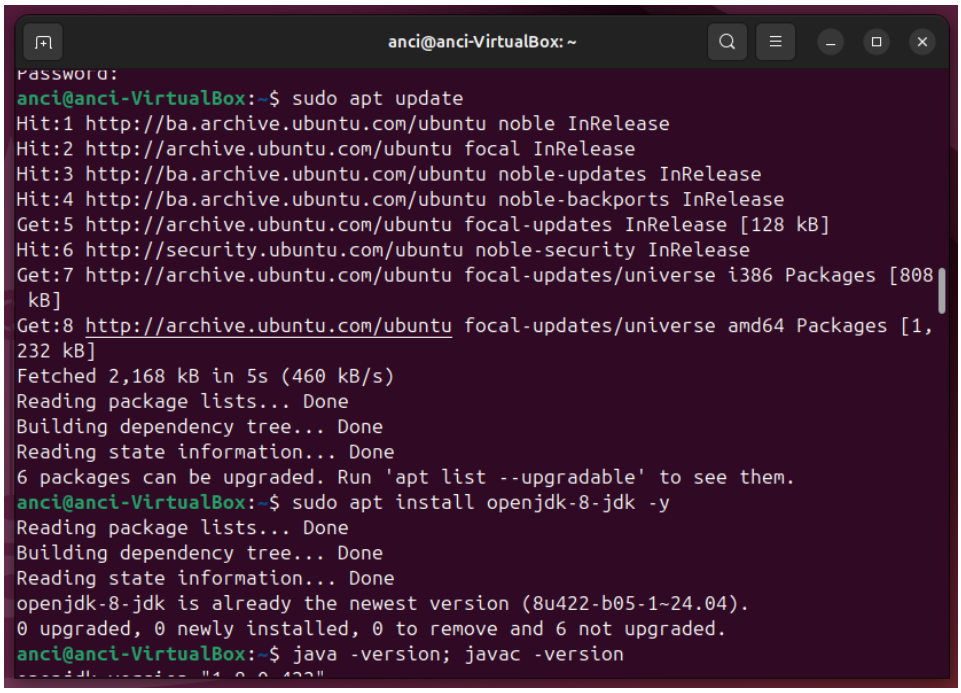
1. Klijent podnosi prijavu
2. Resource Manager dodjeljuje kontejner za pokretanje menadžera aplikacija
3. Menadžer aplikacija(Application Manager) se registruje kod menadžera resursa
4. Menadžer aplikacija pregovara o kontejnerima iz Menadžera resursima
5. Menadžer aplikacija obavještava Node Managera da pokrene kontejnere
6. Aplikacijski kod se izvršava u kontejneru
7. Klijent kontaktira Resource Manager/Application Manager da nadgleda status aplikacije
8. Kada se obrada završi, Menadžer aplikacija poništava registraciju kod Menadžera resursima

Prednosti:

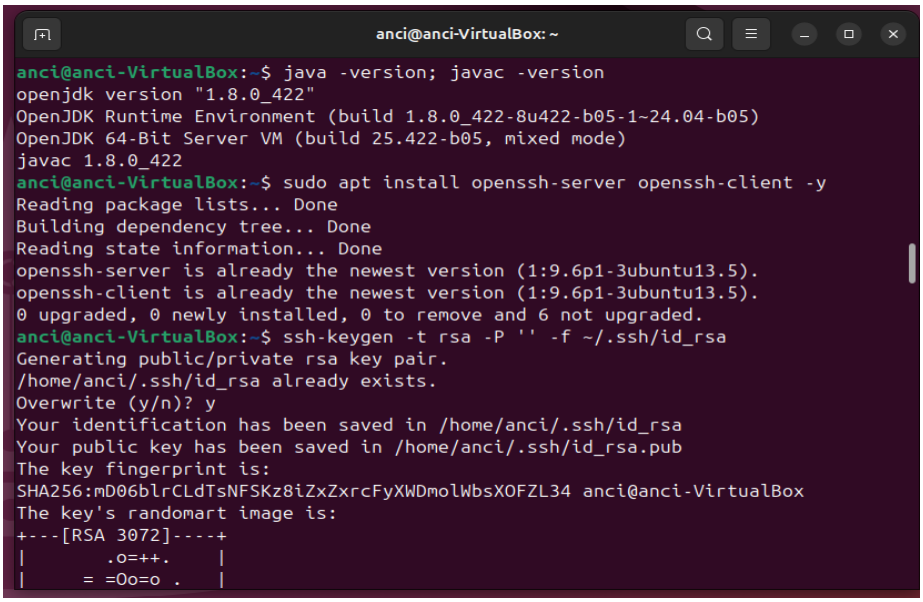
- Fleksibilnost: YARN nudi fleksibilnost za pokretanje različitih tipova distribuiranih sistema za obradu kao što su Apache Spark, Apache Flink, Apache Storm i drugi. Omogućava da više procesora za obradu istovremeno radi na jednom Hadoop klasteru.
- Upravljanje resursima: YARN pruža efikasan način upravljanja resursima u Hadoop klasteru. Omogućava administratorima da dodijele i nadgledaju resurse potrebne svakoj aplikaciji u klasteru, kao što su CPU, memorija i prostor na disku.
- Skalabilnost: YARN je dizajniran da bude visoko skalabilan i može da obrađuje hiljade čvorova u klasteru. Može se povećati ili smanjiti ovisno o zahtjevima aplikacija koje rade na klasteru.
- Poboljšane performanse: YARN nudi bolje performanse pružanjem centralizovanog sistema upravljanja resursima. Osigurava da se resursi optimalno koriste i da se aplikacije efikasno raspoređuju na raspoloživim resursima.
- Sigurnost: YARN pruža robusne sigurnosne karakteristike kao što su Kerberos autentifikacija, Secure Shell (SSH) pristup i siguran prijenos podataka. Osigurava da su podaci pohranjeni i obrađeni na Hadoop klasteru sigurni.

4.4 Optimizacija skladištenja podataka u Apache Hadoop sistemu

Za potrebnu praktičnog dijela istraživanja, potrebno je bilo preuzeti neki Linux operativni sistem, u ovom slucaju je korišten Ubuntu na Virtual Box zbog njegove kompatibilnosti sa Big data alatima kao sto je Apache Hadoop. Nakon instalacije Ubuntu-a, potrebno je instalirati i konfigurisati Apache Hadoop-a. Ova faza je ključna za postavljanje distribuiranog Sistema za skladištenje i obradu podataka. Proces instalacije i konfiguracije Hadoop ekosistema na virtuelnoj masini kao i smještanje dataset podataka u csv fajlu prikazan je u sledećim koracima.

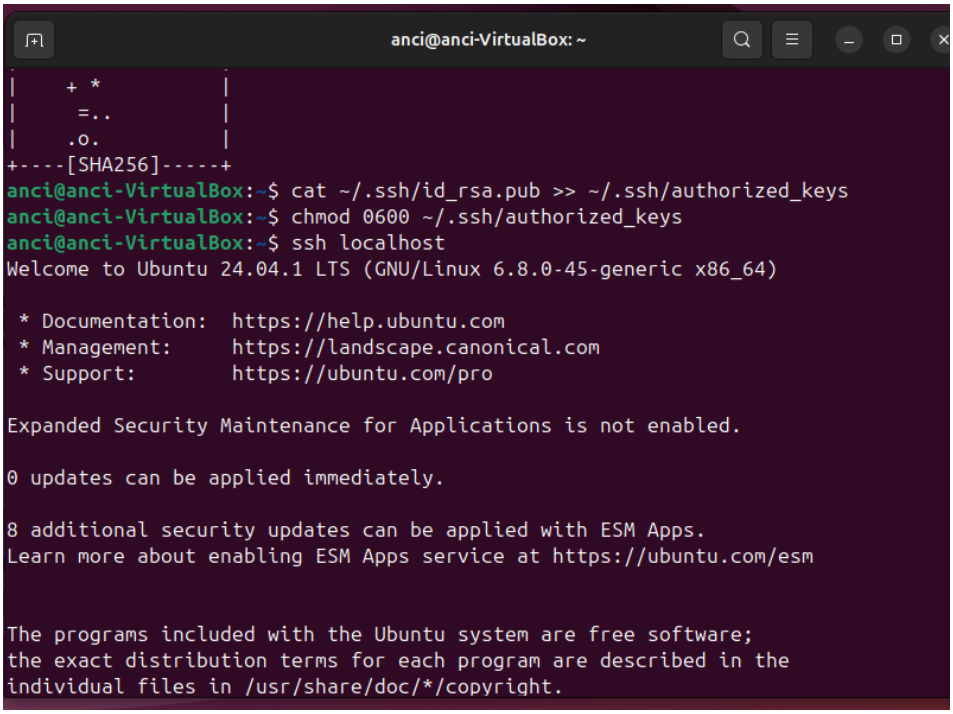


Slika 11 Proces instalacije JAVA JDK 8 paketa na Ubuntu Virtual Box

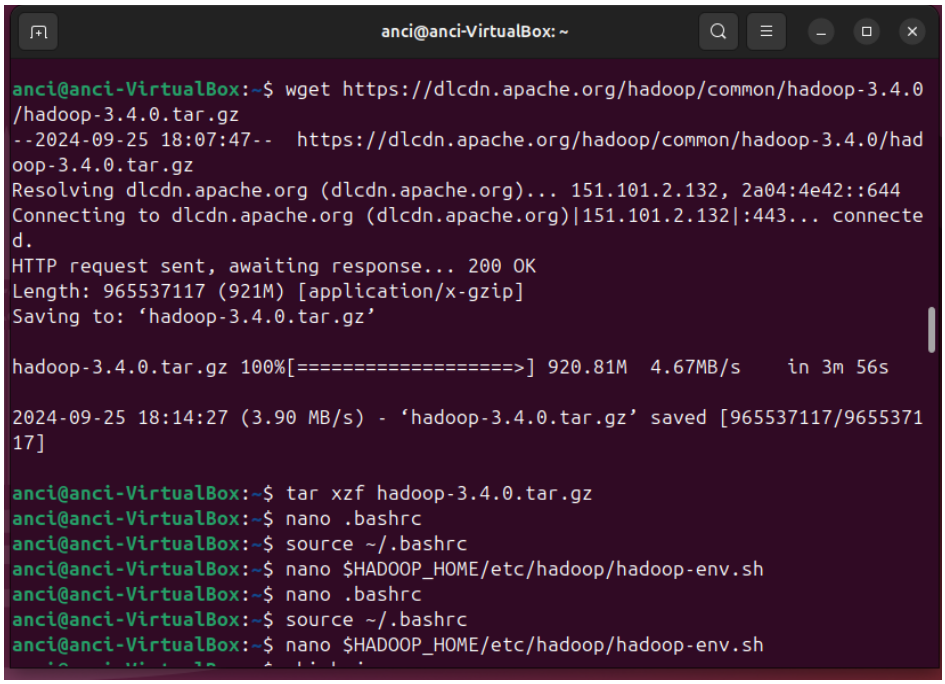


Slika 12 SSH Secure Shell instalacija

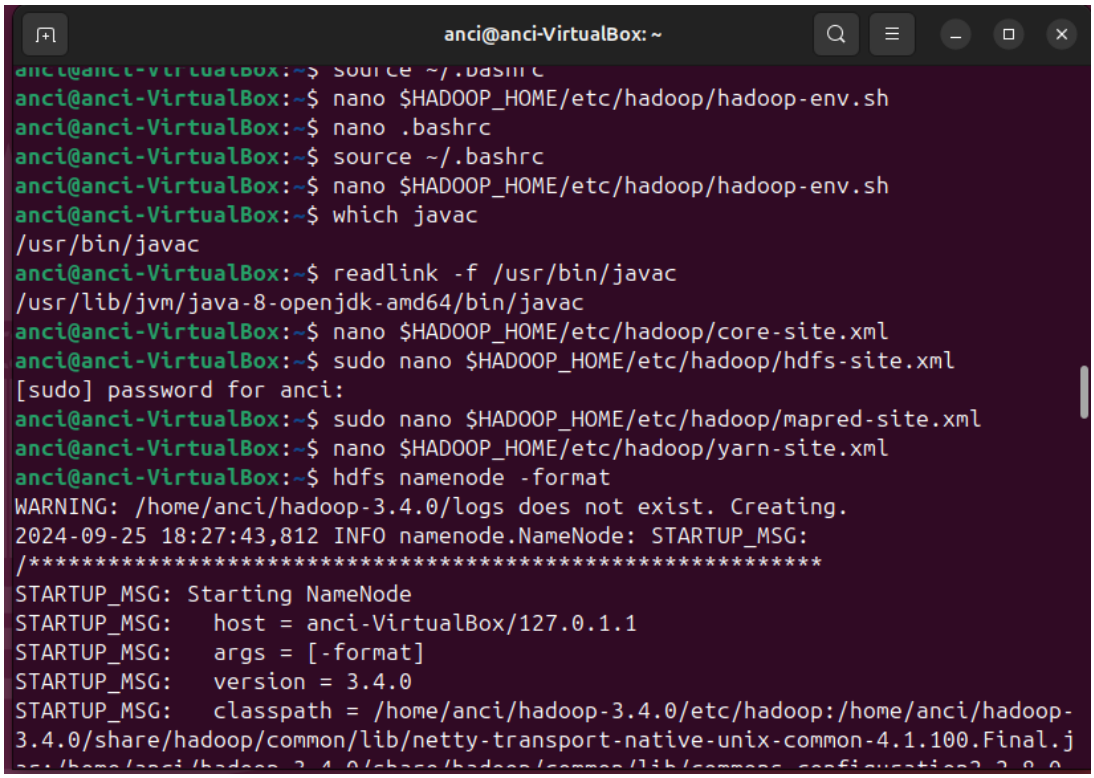
SSH (Secure Shell) instalacija je od vitalnog značaja za Hadoop jer omogućava sigurnu komunikaciju između čvorova u Hadoop klasteru. Ovo osigurava integritet podataka, povjerljivost i omogućava efikasnu distribuiranu obradu podataka u cijelom klasteru



Slika 13 Kopiranje generisanog javnog ključa u autorizovanu datoteku ključa i autentifikacija lokalnog hosta



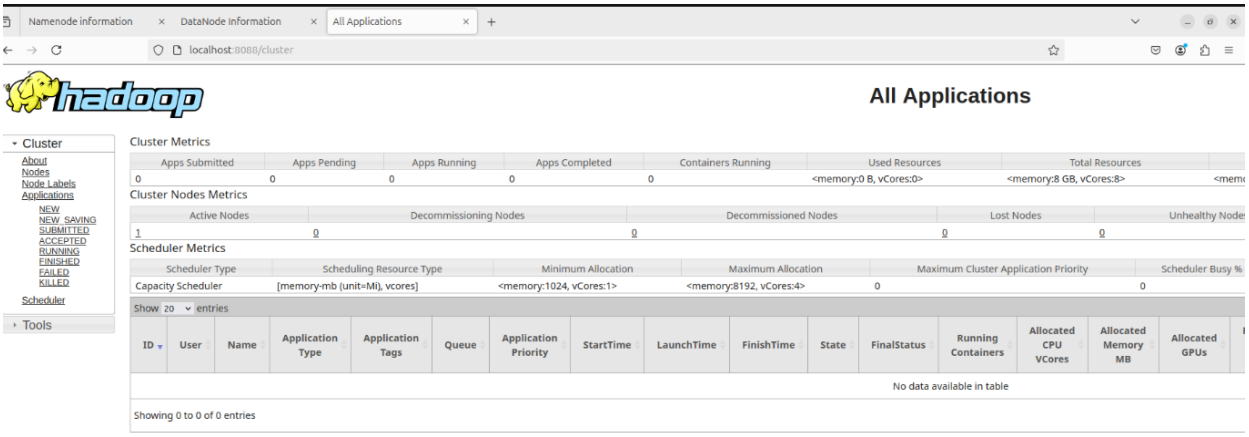
Slika 14 Konfiguracija Hadoopa



Slika 15 Konfiguracija JAVA_HOME i kofiguracionih fajlova

Konfiguracija Hadoop okruženja, uključujući podešavanje JAVA_HOME promenljive i konfiguracionih fajlova kao što su *core-site.xml* i *hdfs-site.xml*

Za pristup Upravljanju resursima, potrebno je otvoriti web pretraživač i posjetiti URL http://your-server-ip:8088, koji prikazuje ekran kao što je prikazano na slici ispod:



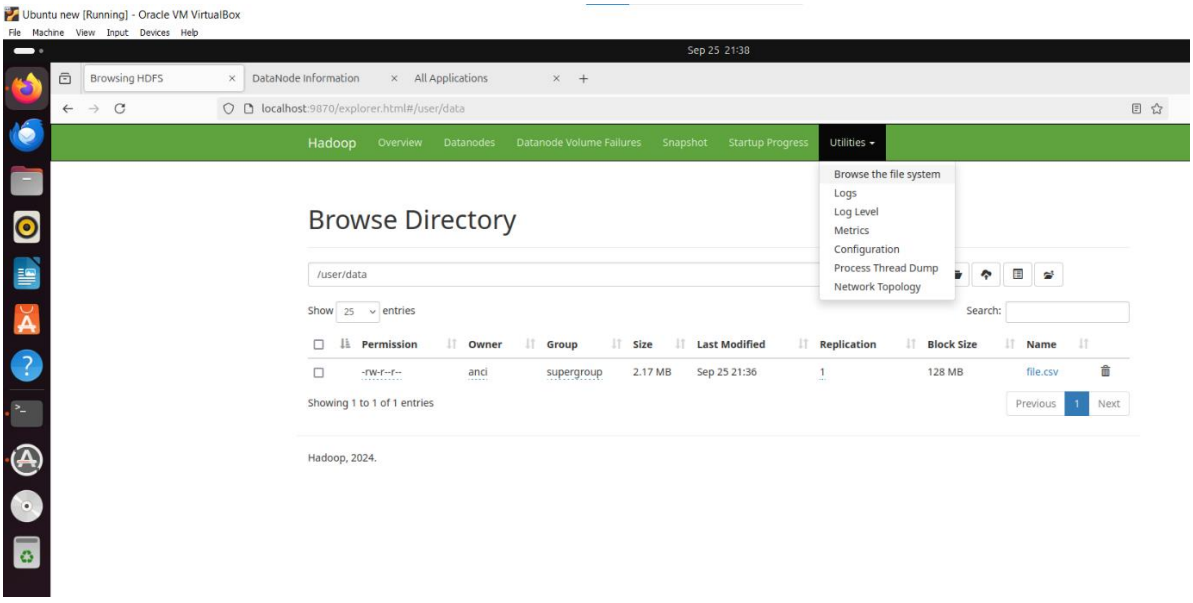
Slika 16 Prikaz interfejsa Hadoop okruženja

Za smještanje i učitavanje određenog dataset podataka u csv formatu potrebno je ukucati sledeće komande:

```
anci@anci-VirtualBox:~$ hdfs dfs -mkdir /user/data
anci@anci-VirtualBox:~$ hdfs dfs -put /home/anci/file.csv /user/data/file.csv
anci@anci-VirtualBox:~$
```

Slika 17 Prikaz komandi za učitavanje fajlova

Nakon učitavanja datoteka se pojavljuje u Browse direktorijumu.



Slika 18 Prikaz Hadoop Browse direktorijuma

5. BIG DATA alati za skladištenje i analizu podataka

Sa nevjerovatnim porastom generisanja podataka, veliki podaci su se pojavili kao ključni koncept koja pokreće inovacije i rast za preduzeća na globalnom nivou. Kako organizacije prikupljaju više podataka nego ikad, potrebni su im pravi alati za efikasno upravljanje i analizu podataka. Platforme velikih podataka pružaju infrastrukturu i alate koji su potrebni za skladištenje, obradu i analizu opsežnih i složenih skupova podataka. Odabir prave platforme za velike podatke je veoma bitna odluka koja može značajno uticati na sposobnost organizacije da dobije vrijedne uvide i donese odluke zasnovane na podacima. U eri digitalne transformacije, velika količina generisanih podataka zahtijevala je razvoj specijalizovanih platformi za rukovanje i analizu ovog ogromnog toka informacija. Platforme velikih podataka su sveobuhvatni okviri koji omogućavaju organizacijama da pohranjuju, obrađuju i analiziraju ogromne količine strukturiranih i nestrukturiranih podataka.

Platforme za upravljanje velikim podacima osnažuju preduzeća da identifikuju trendove i optimizuju operacije korišćenjem distribuiranog računarstva, paralelne obrade i naprednih tehnika analitike. Od unosa i skladištenja podataka do obrade i vizualizacije podataka, platforme za analizu velikih podataka nude sveobuhvatno rješenje za upravljanje i iskorištavanje moći podataka u modernom dobu.

Nekoliko velikih platformi podataka nudi sveobuhvatne funkcije i rješenja za preduzeća za upravljanje i analizu složenih skupova podataka. Najistaknutije platforme velikih podataka koje koriste kompanije uključuju sledeće:

Apache Hadoop, Apache Spark, Apache Flink, Apache Storm i mnogi drugi.

5.1 Apache Spark

Apache Spark je mehanizam za obradu podataka u memoriji i analitku koji može raditi na klasterima kojima upravljaju Hadoop YARN, Mesos i Kubernetes ili u samostalnom načinu rada. Omogućava velike transformacije podataka i analizu i može se koristiti i za paketne i streaming aplikacije, kao i za slučajeve korištenja mašinskog učenja i obrade grafova.

Apache Spark je platforma za obradu velikih podataka koja prilagođava hibridni okvir. Hibridni okvir nudi podršku za batch i stream mogućnosti obrade. Iako Spark koristi mnogo sličnih principa kao i Hadoop MapReduce, Spark nadmašuje Hadoop MapReduce u smislu performansi. Na primjer, obzirom na isto opterećenje grupne obrade, Spark može biti brži od MapReduce-a zbog funkcije „računanja pune memorije“ koju koristi Spark u poređenju sa tradicionalnim čitanjem i pisanjem na disk koji koristi MapReduce. Spark može raditi u samostalnom načinu rada ili se može kombinovati sa Hadoop-om kako bi zamijenio MapReduce.

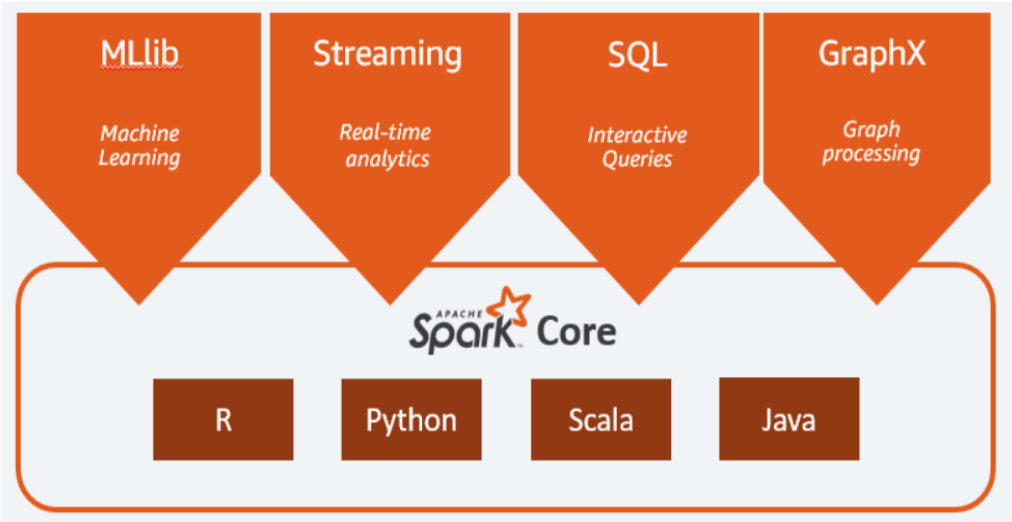
1) *Model paketne obrade Spark-a*: Jedna od prednosti Spark-a u odnosu na MapReduce je računanje u memoriji. Spark komunicira sa diskom samo za dva zadatka: početno učitavanje podataka u memoriju i pohranjivanje konačnih rezultata nazad u memoriju. Svi ostali rezultati između se obrađuju u memoriji. Ova obrada u memoriji čini Spark znatno bržim od svog konkurentnog okvira za grupnu obradu Hadoop-a. Takođe, optimizacija koju koristi Spark dodatno doprinosi njegovoj velikoj brzini gdje se kompletan skup zadataka može analizirati unaprijed. To se postiže generisanjem usmjerenih acikličkih grafova (DAG[2]) koji se koriste za predstavljanje svih operacija, podataka i odnosa između njih. Da bi podržao funkciju računanja u memoriji, Spark koristi otporne distribuirane skupove podataka (RDD). RDD[3] je samo prostorna struktura podataka koja se održava u memoriji kako bi se napravio Sparkov okvir tolerancije grešaka bez potrebe za pisanjem na disk nakon svake operacije.

2) *Model obrade Spark Stream-a:* Pored batchprocesiranja, Spark pruža mogućnosti obrade toka uz korištenje mikro-batcha. U mikro-batchingu tokovi podataka se

[2] DAG (Directed Acyclic Graph) - je konceptualni prikaz niza aktivnosti. Redoslijed aktivnosti prikazan je grafikonom koji je vizualno predstavljen kao skup krugova, od kojih svaki predstavlja aktivnost, od kojih su neke povezane linijama, koje predstavljaju tok od jedne aktivnosti do druge.

[3] RDD (Resilient Distributed Datasets) - omogućavaju ponovnu upotrebu podataka na način otporan na greške.

tretiraju kao grupa vrlo malih serija kojima Spark batch engine zauzvrat rukuje kao redovni zadatak. Iako ovaj mikro-batching postupak dobro funkcioniše, ipak može dovesti do nekih razlika u pogledu performansi za razliku od pravih okvira za obradu toka.



Slika 19 Spark ekosistem, izvor: https://www.chaossearch.io/blog/apache-spark-analytics

Spark ekosistem se sastoji od pet komponenti:

1) Spark Core

Spark Core je osnova Apache Spark platforme. Odgovoran je za poslove kao što planiranje, uklanjanje grešaka, upravljanje memorijom, distribuciju i nadgledanje poslova i interakciju sa sistemima za skladištenje podataka. Spark Core je izložen kroz interfejs za programiranje aplikacija (API) izgrađen za Javu, Scalu, Python i R. Ovi API-ji skrivaju složenost distribuirane obrade iza jednostavnih operatora visokog nivoa.

2) MLlib (Mašinsko učenje)

Spark uključuje MLlib, biblioteku visokokvalitetnih algoritama sa velikom brzinom za mašinsko učenje podataka na visokom nivou. Modele mašinskog učenja mogu obučiti naučnici podataka sa R ili Python-om na bilo kom Hadoop izvoru podataka, sačuvati pomoću MLlib-a i uvesti u cijevovod zasnovan na Javi ili Scali. Spark je dizajniran za brzo, interaktivno računanje koje radi u memoriji, omogućavajući brzo pokretanje mašinskog učenja. Algoritmi uključuju sposobnost klasifikacije, regresije,

grupisanja i lienearnu algebru. Takođe pruža i druge mogućnosti kao što su procjena modela i uvoz podataka. Može se koristiti u Javi, R, Scali i Pythonu.

3) Spark Streaming

Spark Streaming je riješenje u realnom vremenu koje koristi Spark Core koje koristi mogućnost da obrađuju streaming podatake u realnom vremenu. Unosi podatke u mini-serijama i omogućava analizu tih podataka sa istim kodom aplikacije napisanim za grupnu analitiku. Ovo poboljšava produktivnost programera, jer oni mogu koristiti isti kod za grupnu obradu i za streaming aplikacije u realnom vremenu. Spark Streaming podržava podatke sa HDFS-a, Twitter-a, Kafke, Flume-a i ZeroMQ-a i mnoge druge koji se nalaze u ekosistemu Spark Packages-a.

4) Spark SQL (Interaktivni upiti)

Spark SQL je distribuirani mehanizam upita koji pruža interaktivne upite niske latencije i do 100x brže od MapReducea. Uključuje optimizator zasnovan na troškovima, skladištenje i generisanje koda za brze upite, uz skaliranje na hiljade čvorova. Poslovni analitičari mogu koristiti standardni SQL ili Hive Query Language za upite podataka. Programeri mogu koristiti API-je, dostupne u Scala, Java, Python i R. Spark SQL podržava različite izvore podataka uključujući JDBC, ODBC, JSON, HDFS, Hive, ORC i Parquet. Druge popularne prodavnice—Amazon Redshift, Amazon S3, Couchbase, Cassandra, MongoDB, Salesforce.com, Elasticsearch i mnoge druge mogu se pronaći iz ekosistema Spark Packages.

5) GraphX (Obrada grafikona)

Spark GraphX je Spark-ova komponenta za obradu podataka grafa. On proširuje funkcionalnost Spark-a kako bi se bavio paralelnim proračunima grafova, omogućavajući korisnicima da modeluju i transformišu svoje podatke u grafove. Ova karakteristika značajno pojednostavljuje složeno rukovanje i analizu odnosa podataka.

5.2 Apache Kafka

Apache Kafka je distribuirana platforma za strimovanje podataka koja može da objavljuje, skladišti i obrađuje tokove zapisa u realnom vremenu. Tehnologija razdvaja tokove podataka i sisteme, zadržavajući tokove podataka tako da se mogu koristiti negdje drugo. Radi u distribuiranom okruženju i koristi TCP mrežni protokol visokih performansi za komunikaciju sa sistemima i aplikacijama.

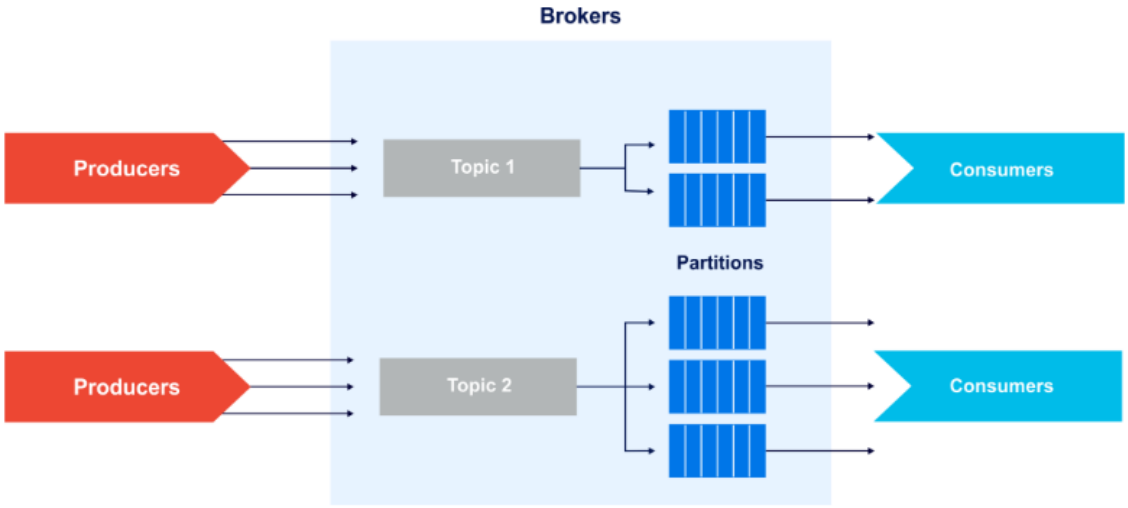
Kafka je sistem za obradu toka koji se koristi za razmjenu poruka, praćenje aktivnosti na web stranici, prikupljanje i praćenje metrika, evidentiranje i analitiku u realnom vremenu. Dobro se uklapa u aplikacije za obradu poruka velikih razmjera jer je pouzdaniji i tolerantniji na greške kada ga uporedimo sa tradicionalnim redovima poruka. Veoma je pouzdan, ima visoku dostupnost i omogućava geografski raspoređene tokove podataka i aplikacije za obradu tokova.

Doniran je Fondaciji Apache od strane LinkedIn-a 2011. godine, Kafka je izazvao veliko interesovanje i sada ga široko koriste mnoge organizacije širom svijeta, uključujući Netflix, Twitter, Spotify.

U bilo kojoj Big Data aplikaciji, Kafka ima tri osnovne funkcije a to su:

1. Prenos poruka - Omogućavanje transporta podataka između različitih krajnjih tačaka izdavača i pretplatnika.
2. Agregacija poruka - Agregiranje niza različitih tokova podataka za korištenje od strane aplikacija za distribuiranu obradu.
3. Message Store - Pohranjivanje tokova podataka kao keš memorije u repliciranom okruženju za skladištenje otpornom na greške.

Kafka funkcioniše tako što prihvata tokove događaja koje pišu proizvođači podataka. Zatim, pohranjuje zapise hronološki u particijama među brokerima (serverima), a više brokera čine klaster. Svaki zapis sadrži informacije o događaju i sastoji se od para ključ/vrijednost a vremenska oznaka i zaglavlje su opcione dodatne informacije. Kafka grupiše zapise u teme (primjer: Topic 1 i Topic 2, kao što je prikazano na slici ispod) i nakon toga potrošači podataka dobijaju svoje podatke pretplatom na teme koje žele.



Slika 20 Prikaz toka rada Apache Kafka, izvor: https://www.instaclustr.com/education/apache-kafka/

Neki od osnovnih koncepata Apache Kafka su:

1. Događaji (Events) - Događaj je poruka sa podacima koji opisuju događaj. Na primjer, kada se novi korisnik registruje na web-stranici, sistem kreira događaj registracije, koji može uključivati korisničko ime, e-mail, lozinku ili lokaciju..

2. Potrošači i proizvođači (Consumers and Producers) - Proizvođač je sve što stvara podatke i oni pišu događaje Kafki. Primjeri proizvođača uključuju web servere, druge aplikacije (ili komponente aplikacije), IoT uređaje, agente za praćenje itd. na primjer:

- Komponenta web stranice odgovorna za registraciju korisnika proizvodi događaj “novi korisnik je registriran”.
- Vremenski senzor (IoT uređaj) proizvodi "vremenske" događaje po satu sa informacijama o temperaturi, vlažnosti, brzini vjetra i tako dalje.

Potrošači su subjekti koji koriste podatke koje su napisali proizvođači. Ponekad entitet može biti i proizvođač i potrošač u zavisnosti od arhitekture sistema. Na primjer, skladište podataka može konzumirati podatke iz Kafke, zatim ih obraditi i proizvesti pripremljeni podskup za preusmjeravanje preko Kafke u ML ili AI aplikaciju. Baze podataka, i aplikacije za analizu podataka

uopšteno djeluju kao potrošači podataka, pohranjujući ili analizirajući podatke koje primaju od Kafke.

3. Brokeri i klasteri (Brokers and Clusters) - Kafka radi na klasterima, iako sada postoji verzija Kafke bez servera u pregledu na AWS-u. Svaki klaster se sastoji od više servera, koji se uopšteno nazivaju brokeri (a ponekad i čvorovi). To je ono što Kafku čini distribuiranim sistemom a podaci u Kafka klasteru se distribuišu među više brokera i više kopija (replika) istih podataka postoji u Kafka klasteru. Ovaj mehanizam čini Kafku stabilnijim, otpornijim na greške i pouzdanijim ako dođe do greške ili kvara kod jednog brokera, drugi broker stupa u funkciju neispravne komponente, a informacije se ne gube.

4. Teme (Topic) - Kafkina tema je nepromjenjivi dnevnik događaja (sekvence). Proizvođači objavljuju događaje na Kafkine teme a potrošači se pretplate na teme kako bi pristupili svojim željenim podacima. Svaka tema može poslužiti podacima mnogim potrošačima. Nastavljajući sa primjerom pomenitim iznad , komponenta registracije web stranice objavljuje događaje kao „novi korisnik“ (preko Kafke) u temu „registracije“. Pretplatnici kao što su analitičke aplikacije, aplikacije za vijesti, aplikacije za praćenje, baze podataka zauzvrat koriste događaje iz teme „registracije“ i koriste ih sa drugim podacima kao osnovu za isporuku vlastitih proizvoda ili usluga.

5. Particije (Partitions) - Particija je najmanja jedinica za skladištenje u Kafki. Particije služe za podjelu podataka među brokerima kako bi se ubrzale performanse. Svaka Kafka tema je podijeljena na particije i svaka particija se može postaviti na zasebnog brokera.

5.3 Apache Flink

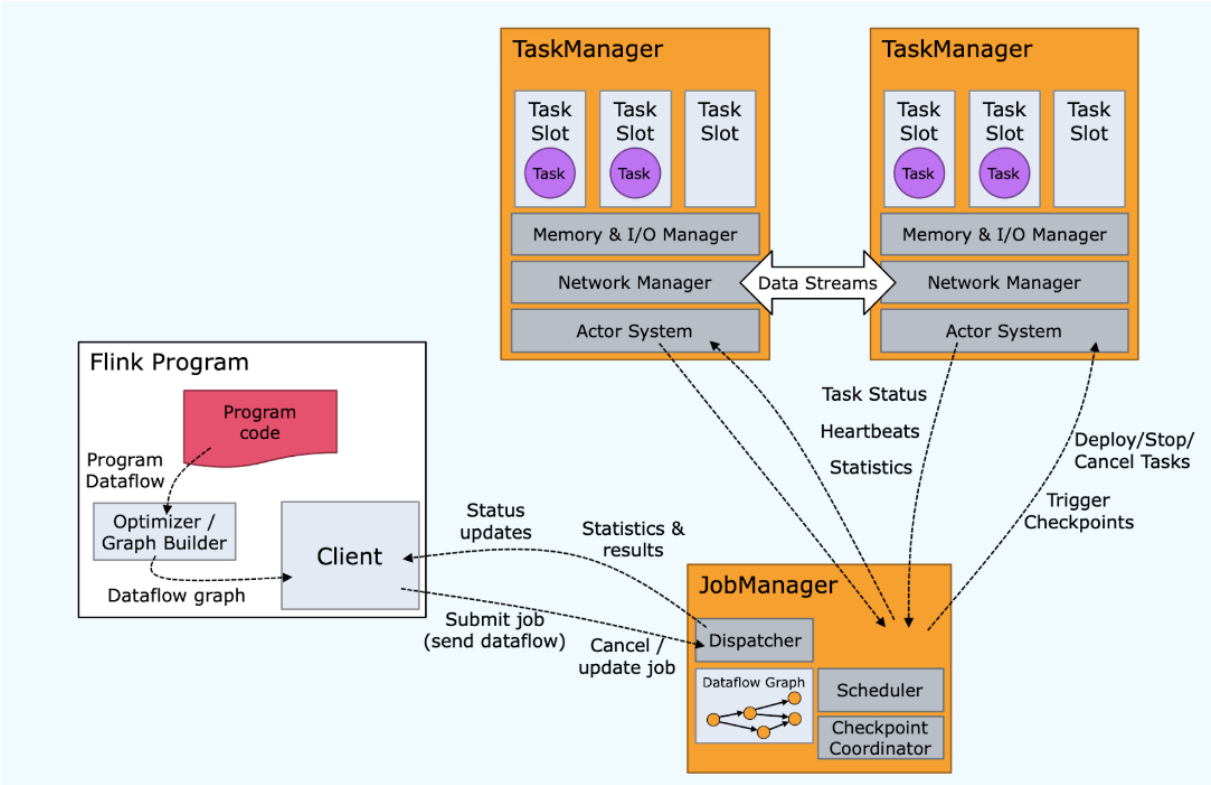
Apache Flink je okvir i mehanizam za distribuiranu obradu za proračune sa stanjem preko neograničenih i ograničenih tokova podataka. Flink je dizajniran tako da radi u svim uobičajenim okruženjima klastera i da izvodi proračune brzinom u memoriji, na bilo kojoj skali. Programeri grade aplikacije za Flink koristeći API-je kao što su Java ili SQL, koji se izvršavaju na Flink klasteru od strane okvira. Dok su drugi popularni sistemi za obradu podataka poput Apache Spark i Kafka Streams ograničeni na strimovanje podataka ili grupnu obradu, Flink podržava oboje. To ga čini svestranim alatom za kompanije u industrijama kao što su finansije, e-trgovina i telekomunikacije koje sada mogu obavljati i grupnu i stream obradu u jednoj objedinjenoj platformi. Preduzeća mogu koristiti Apache Flink za moderne aplikacije kao što su otkrivanje prevara, personalizovane preporuke, analiza tržišta dionica i mnoge druge.

5.3.1 Apache Flink arhitektura i ključne komponente

Dok je Kafka Streams biblioteka koja radi kao dio neke aplikacije, Flink je samostalni alat za obradu toka koji se postavlja nezavisno. Pruža rješenja za teške probleme sa kojima se suočavaju distribuirani sistemi za obradu toka, kao što su tolerancija grešaka, isporuka tačno jednom, visoka propusnost i nisko kašnjenje.

Flinkova arhitektura je dizajnirana i za stream i za batch obradu. Može da upravlja neograničenim tokovima podataka i ograničenim skupovima podataka, omogućavajući obradu podataka i analizu podataka u realnom vremenu. Flink takođe osigurava integritet podataka, čak i u složenim scenarijima obrade događaja.

Slika ispod prikazuje Flink komponente kao i Flink tok vremena izvođenja. Programski kod ili SQL upit sastavljen je u graf operatora koji klijent zatim šalje JobManager-u. JobManager razbija posao na operatore koji se izvršavaju kao zadaci na čvorovima koji pokreću TaskManager-i. Ovi zadaci obrađuju podatke koji se prenose i komuniciraju sa različitim izvorima podataka, kao što su Hadoop Distributed File System (HDFS) i Apache Kafka.



Slika 21 Prikaz toka rada Apache Flink-a,izvor: https://www.confluent.io/learn/apache-flink/

Kljucne komponente Apache Flink-a su:

- Program - To je dio koda koji pokrećete na Flink Clusteru.
- Klijent - Odgovoran je za preuzimanje koda (programa) i konstruisanje grafa toka podataka posla, a zatim ga prosljeđuje JobManageru. Takođe preuzima rezultate posla.
- JobManager - Nakon što od klijenta dobije graf toka podataka posla, odgovoran je za kreiranje grafa izvršenja. Dodjeljuje posao upraviteljima zadataka u klasteru i nadzire izvršenje posla.
- TaskManager - Odgovoran je za izvršavanje svih zadataka koje je dodijelio JobManager. Svi TaskManageri pokreću zadatke u svojim zasebnim slotovima u određenom paralelizmu. Odgovoran je za slanje statusa zadataka JobManageru.

6. Bezbjednost i sigurnost u Big Data

U današnjem digitalnom svijetu, gdje informacije posvuda prostiru, veliki podaci su se pojavili kao prilika za inovacije, ali takođe kriju potencijalni rizik od raznih ranjivosti. Velike količine podataka koje obrađuju kompanije i pojedinci sadrže vrijedne uvide u ponašanje kupaca, operativne performanse i tržišne trendove koji mogu poboljšati procese donošenja odluka, optimizovati poslovne operacije i razviti nove proizvode. Međutim, sigurnost velikih podataka ostaje veliki izazov za organizacije i može uzrokovati značajne gubitke pojedincima i preduzećima.

Sigurnost velikih podataka je skup mjera i praksi sigurnosti podataka za zaštitu velikih količina podataka, poznatih kao "veliki podaci", od napada zlonamjernog softvera, neovlaštenog pristupa i drugih bezbjednosnih prijetnji. Proces uključuje zaštitu povjerljivosti, integriteta i dostupnosti podataka. Upravljanje sigurnošću velikih podataka uključuje enkripciju podataka, kontrolu pristupa, autentifikaciju, autorizaciju, praćenje, otkrivanje prijetnji, kao i obuku zaposlenih.

Arhitektura sigurnosti velikih podataka odnosi se na strukturu i komponente postavljene kako bi se osigurala sigurnost i zaštita. Uključuje različite faze i mjere za smanjenje rizika i zaštitu osjetljivih podataka. Iako se specifična arhitektura može razlikovati u zavinosti od organizacije i njenim zahtjevima. Nekoliko bitnih standardnih komponenti i razmatranja su:

- Kontrola pristupa
  Da bi podaci bili sigurni, važno je ograničiti i fizički i digitalni pristup centralnim sistemima i podacima. Cilj je osigurati da svi računari i uređaji budu zaštićeni lozinkom i da su fizičke lokacije dostupne samo ovlaštenim osobama.

- Autentifikacija
  Prije nego što se odobri pristup podacima, preporučljivo je obezbjediti mjere autentifikacije, kao što su ograničenja pristupa i pravilna identifikacija ljudi. Biometrija, lozinke, PIN-ovi ili sigurnosni tokeni.

- Sigurnosne kopije i oporavak od katastrofe (Backup)
  Efikasna sigurnost podrazumijeva postojanje plana za siguran pristup podacima tokom kvarova sistema, katastrofa, oštećenja podataka. Da bi se

olakšao oporavak, rezervna kopija podataka mora biti pohranjena u zasebnom formatu, kao što je hard disk, lokalna mreža ili cloud.

- Brisanje podataka
  Redovno i pravilno odlaganje podataka je neophodno. Brisanje podataka, koje koristi softver za potpuno brisanje podataka s bilo kojeg uređaja za pohranu, sigurnija je metoda od konvencionalnog brisanja podataka. Garantuje da se podaci ne mogu povratiti, sprečavajući da završe u neovlašćenim rukama.

- Maskiranje podataka
  Softver za maskiranje podataka koristi proxy znakove da sakrije slova i brojeve, efektivno prikrivajući informacije. U slučaju neovlaštenog pristupa, podaci ostaju skriveni, postaju vidljivi samo kada im pristupi ovlašteni korisnik.

- Enkripcija
  Pomoću ključeva za šifrovanje, kompjuterski algoritam pretvara tekstualne znakove u nerazumljiv oblik, osiguravajući da samo ovlaštene osobe sa potrebnim ključevima mogu otključati i pristupiti sadržaju. U određenoj mjeri, bitno je osigurati različite oblike podataka, uključujući datoteke, baze podataka i komunikaciju putem e-pošte.

6.1 Šifromanje i anonimizacija podataka

Anonimizacija podataka uklanja lične podatke iz skupa podataka, što otežava ili čini nemogućim povezivanje podataka sa pojedincem ili organizacijom. Djeluje tako što zamjenjuje polja podataka koja se mogu identifikovati neidentifikovanim poljima podataka. Na primjer, broj socijalnog osiguranja može se zamijeniti nasumičnim skupom brojeva. Posebno je važno kada se radi o osjetljivim podacima, kao što su medicinski, financijski ili lični podaci, jer organizacijama daje mogućnost da koriste podatke za neophodne poslovne funkcije, istovremeno štiteći privatnost pojedinca.

Anonimizacija podataka je opšti pojam koji uključuje više tehnika, kao što su maskiranje podataka, generalizacija podataka, pseudonimizacija i razmjena podataka.

Šifrirovanje je moćna tehnika koja pretvara podatke u format koji mogu razumjeti samo oni koji posjeduju pravi ključ ili lozinku. Pruža dodatni sloj sigurnosti za osjetljive informacije kao što su brojevi bankovnih računa i lozinke. Pored toga, šifrovanje se može koristiti za zaštitu vrijedne imovine kao što su intelektualna svojina, poslovne tajne i druge povjerljive informacije koje treba čuvati privatno. Šifrovanje se često koristi kada podaci treba da budu zaštićeni od neovlašćenog pristupa tokom prenosa ili skladištenja.

Algoritmi pretvaraju originalne podatke u kodirani oblik koji mogu razumjeti samo ovlaštene strane sa ključem za dešifrovanje. Kada su podaci šifrovani, oni postaju nečitljivi osim ako nemate ispravan ključ ili lozinku za njihovo dešifrovanje. Iz tog razloga, šifrovanje se često koristi kako bi podaci bili sigurni dok su u pohranjeni ili u pokretu (koji se šalju preko mreže). Na primjer, šifrovanje se često koristi za zaštitu informacija o kreditnoj kartici koje se dijele tokom online transakcija.

Anonimizacija i šifrovanje podataka(enkripcija) omogućavaju preduzećima da zaštite lične podatke i da se pridržavaju standarda privatnosti podataka kao što su Zakon o zaštiti ličnih podataka (GDPR). Glavna razlika između njih je u nivou zaštite koji nude i kako tu zaštitu postižu. Anonimizacija podataka pruža visok nivo zaštite privatnosti jer su podaci potpuno anonimni. Šifrovanje pruža visok nivo sigurnosti jer se podaci pretvaraju u nečitljiv format.

Alati za anonimizaciju podataka uklanjaju lične podatke iz skupa podataka, mijenjajući same podatke kako bi zaštitili pojedinačne korisnike od identifikacije. Enkripcija, s druge strane, ne mijenja same podatke, već umjesto toga transformiše u nečitljiv format koji se može dešifrovati samo odgovarajućim ključem. Šifrovanje se obično koristi za zaštitu osjetljivih podataka u prenosu ili u mirovanju. Uklanjanjem identifikacionih informacija, podaci se i dalje mogu koristiti za istraživanje ili analizu bez ugrožavanja privatnosti uključenih pojedinaca.

6.1.1 Tehnike anonimizacije podataka

Postoji mnogo različitih tehnika anonimizacije podataka koje se mogu koristiti u zavisnosti od potreba organizacije. Na primjer, preduzeće može grupisati podatke o dobi u starosne raspone, umjesto da bilježi svačiju tačnu dob. Anonimizacija podataka uključuje djelomično prikrivanje podataka kako bi ih bilo što teže identifikovati. Pseudonimizacija zamjenjuje podatke kao što su imena, adrese i datumi rođenja jedinstvenim identifikatorom ili kodom. Razmjena podataka funkcionište tako što se stvarne vrijednosti podataka zamjenjuju fiktivnim, ali sličnim. Na primjer, pravo ime ili telefonski broj može se zamijeniti izmišljenim. Prilikom odabira tehnike anonimizacije podataka, organizacije moraju uzeti u obzir kompromis između korisnosti i privatnosti. Jako anonmizovani podaci mogu biti manje korisni za analizu i donošenje odluka. Međutim, ako lični podaci nisu dovoljno anonimizovani, mogli bi se koristiti za identifikaciju pojedinaca i ugrožavanje njihove privatnosti. Preduzeća bi trebala odabrati odgovarajući metod anonimizacije na osnovu osjetljivosti podataka i rizika od ponovne identifikacije. Na primjer, anonimizacija podataka može biti prikladno za manje osjetljive informacije, dok zamjena može biti neophodna za visoko povjerljive podatke.

6.1.2 Tehnike šifrovanja podataka

Postoje dvije glavne tehnike šifrovanja podataka koje se mogu koristiti za zaštitu ličnih ili osjetljivih informacija: Simetrična enkripcija poznata kao šifrovanje zajedničke tajne, koristi isti tajni ključ za šifrovanje i dešifrovanje informacija. Često se koristi za prenos podataka između dvije strane. Asimetrična enkripcija, poznata i kao šifrovanje sa javnim ključem, koristi par ključeva: javni ključ i privatni ključ. Javni ključ se koristi za šifrovanje podataka i svako mu može pristupiti, dok se privatni ključ koristi za dešifrovanje podataka, a posjeduje ga samo primalac. Asimetrična enkripcija se široko koristi za sigurnu komunikaciju putem Interneta, kao što je u online bankarstvu i transakcije e-trgovine, jer omogućava siguran prenos podataka između svije strane. Pored toga, asimetrična enkripcija se često koristi za digitalne potpise, gdje se privatni ključ koristi za kreiranje potpisa koji može samo biti provjereni odgovarajućim javnim ključem, čime se osigurava autentičnost i integritet podataka.

ZAKLJUČAK

Količina i dostupnost podataka ubrzano raste, tako raste i potreba za optimizacijom skladištenja podataka unutar ovih sistema kako bi se osigurala efikasnost, skalabilnost i dostupnost podataka. Optimizacija skladištenja podataka u Big Data sistemima je ključna za smanjenje troškova skladištenja, poboljšanje performansi sistema i omogućavanje brze analize podataka.

U svom dosadašnjem radu i istraživanju primjetila sam kako efikasno skladištenje podataka može značajno smanjiti operativne troškove, poboljšati performanse sistema i omogućiti brže donošenje odluka na osnovu analize podataka. S obzirom na sve veću količinu podataka koja se generiše i povećava u modernim poslovnim okruženjima, od izuzetnog značaja je istražiti i razviti metode koje će omogućiti efikasnije upravljanje tim podacima.

LITERATURA

[1] Min Chen, Shiwen Mao, Yunhao Liu, Big Data: A Survey, Springer Science+Business Media New York 2014 http://www2.egr.uh.edu/~zhan2/ECE6111/class/BigDataSurvey2014.pdf

[2] Wisam A. Qader, Musa M. Ameen and 2Bilal Ismael Ahmed , Big Data Characteristics, Architecture, Technologies and Applications, Department of Computer Engineering, Faculty of Engineering, Tishk International University, Erbil, Iraq, https://thescipub.com/pdf/jcssp.2020.817.824.pdf

[3] Characteristics of Big Data: Understanding the Five V's, https://datavalley.ai/characteristics-of-big-data/

[4] Mahshad Mahmoudian, s. Mohammadali Zanjani, Hossein Shahinzadeh, Yasin Kabalci, Ersan Kabalcı, Farshad Ebrahimi, An Overview of Big Data Concepts, Methods, and Analytics: Challenges, Issues, and Opportunities, Tyrkey, 2024, https://www.researchgate.net/publication/372368521_An_Overview_of_Big_Data_Concepts_Methods_and_Analytics_Challenges_Issues_and_Opportunities

[5] Rajkumar Buyya, Manjrasoft Pty Ltd, Australia Rodrigo N, Australia Amir Vahid Dastjerdi, Big Data Principles and Paradigms, USA 2014, https://dhoto.lecturer.pens.ac.id/lecture_notes/internet_of_things/Big%20Data%20Principles%20and%20Paradigms.pdf

[6] Fatima Binta Adamu , Adib Habbal , Suhaidi Hassan , R. Les Cottrell , Bebo White , Ibrahim Abdullahi, A Survey On Big Data Indexing Strategies, https://www.slac.stanford.edu/pubs/slacpubs/16250/slac-pub-16460.pdf

[7] Indurani P., A SURVEY ON BIG DATA COMPRESSION, https://www.researchgate.net/publication/344376816_A_SURVEY_ON_BIG_DATA_COMPRESSION

[8] Dhruba Borthakur , HDFS Architecture Guide, https://hadoop.apache.org/docs/r1.2.1/hdfs_design.pdf

[9] Apache Hadoop Tutorial, https://enos.itcollege.ee/~jpoial/allalaadimised/reading/Apache-Hadoop-Tutorial.pdf

[10] Exploring Big Data with Apache Spark: Introduction and Key Components , https://medium.com/nerd-for-tech/exploring-big-data-with-apache-spark-introduction-and-key-components-a6872c581ce6

[11] Big Data Security: Advantages, Challenges, and Best Practices

https://www.turing.com/resources/big-data-security

[12] Petr Moskalev, Big Data Security Best Practices, https://maddevs.io/blog/big-data-security-best-practices/

[13] History of Hadoop Project, https://www.vskills.in/certification/tutorial/history-of-hadoop-project-3/

POPIS SLIKA