راهنمایی برای هدوپ

(مقدمه‌ای بر هدوپ)

تألیف:

Kevin Sitto and Marshall Presser

مترجمین:

دکتر حامد تابش (دکترای آمار زیستی، عضو هیات علمی دانشگاه علوم پزشکی مشهد)

الهام نظری (دانشجوی دکترای انفورماتیک پزشکی دانشگاه علوم پزشکی مشهد)

بهرام هدایتی (کارشناسی ارشد مهندسی کامپیوتر، نرم‌افزار)

عنوان و نام پدیدآور: راهنمایی برای هدوپ: مقدمه‌ای بر هدوپ/ [کوین سیتو، مارشال پرسر]؛ مترجمین حامد تابش، الهام نظری، بهرام هدایتی؛ ویراستار معصومه عباسی.

مشخصات نشر: مشهد؛ قلم ذهن ۱۳۹۷

مشخصات ظاهری: ۱۶۲ ص.: جدول؛ ۲۲×۲۹ س‌م.

شابک: ۹۷۸-۶۰۰-۹۸۰۴۸-۴-۹ وضعیت فهرست نویسی: فیپا

یادداشت: عنوان اصلی: Field guide to Hadoop, ۲۰۱۰پ‌ودهر، ی‌ا ه‌مدقم :رگید ن‌اونع .

موضوع: Apache Hadoop؛ داده‌پردازی -- پردازش توزیع شده؛ Electronic data processing -- Distributed processing؛ محاسبات ابری؛ Cloud computing؛ سازماندهی فایل‌ها (کامپیوتر)؛ File organization (Computer science)

شناسه افزوده: پرسر،مارشال؛ Presser, Marshall

رده بندی کنگره: ۹/۷۶QA ۱۳۹۸ ۲ر۲۵د/ رده بندی دیویی: ۷۴/۰۰۵

آدرس: مشهد- بلوار آموزگار- بین آموزگار- پلاک ۵۲۰

تلفن: ۰۹۱۵۸۲۱۶۴۸۹ - ۰۵۱۳۸۹۲۴۰۸۱

شناسنامه کتاب

نام کتاب: راهنمایی برای هدوپ (مقدمه‌ای بر هدوپ)

مولف: *Kevin Sitto and Marshall Presser*

مترجمین: الهام نظری، بهرام هدایتی

تحت نظارت: دکتر حامد تابش

ویراستار: دکتر معصومه عباسی

ناشر: قلمِ ذهن

شمارگان: ۱۰۰۰ جلد

چاپ: اول ۱۳۹۷-

شابک: ۹-۴-۹۸۰۴۸-۶۰۰-۹۷۸

قیمت:

فهرست مطالب

پیشگفتار

هدوپ[1] چیست و چرا شما باید به آن اهمیت دهید؟ این کتاب به شما کمک خواهد کرد تا ماهیت هدوپ را درک کنید، اما اکنون اجازه دهید در مورد بخش دوم سؤال بحث کنیم. هدوپ رایج‌ترین پلت فرم واحد[2] برای ذخیره‌سازی[3] و تحلیل کلان داده‌ها[4] می‌باشد. اگر شما و سازمان‌تان در حال ورود به دنیای مهیج کلان داده‌ها هستید، باید تصمیم بگیرید که آیا هدوپ پلت فرم درستی است و از میان مؤلفه‌های[5] بسیار کدام یک برای فعالیت[6] مورد نظرتان مناسب‌تر است. هدف این کتاب، معرفی این موضوع و آماده‌سازی شما برای آغاز نمودن سفرتان است.

کتاب‌ها، وب‌سایت‌ها و کلاس‌های متعددی در مورد هدوپ و تکنولوژی‌های مربوط به آن وجود دارند؛ اما این کتاب متفاوت است. این کتاب به مقدمات آموزشی طولانی برای جنبه مشخصی از هدوپ یا هر یک از مؤلفه‌های متعدد اکوسیستم هدوپ نمی‌پردازد. یقیناً کتاب حاضر منبعی غنی و دقیق در مورد هر یک از این موضوعات نیست. در عوض، شبیه راهنمای نگهداری از درختان یا پرندگان می‌باشد. هر فصل بر روی قسمت‌هایی از اکوسیستم هدوپ که موضوع مشترکی دارند، متمرکز است. در هر فصل، عناوین و تکنولوژی‌های مرتبط به‌صورت مختصر ارائه شده‌اند. ما ارتباط آن‌ها با هدوپ را توضیح داده‌ایم و در مورد اینکه چرا آن‌ها می‌توانند برای نیازهای خاصی سودمند (و در بعضی موارد کم‌فایده) باشند، بحث کرده‌ایم. بدین منظور، این کتاب شامل بخش‌های کوتاه مختلفی مرتبط با بسیاری از پروژه‌ها و زیرپروژه‌های Apache Hadoop و تکنولوژی‌های وابسته است. بخش‌های یادشده به همراه اشاره‌ای به آموزش‌ها و لینک‌هایی به فرآیندها و تکنولوژی‌های مربوطه ارائه شده‌اند.

---

1. *Hadoop*
2. *Single Platform*
3. *Storing*
4. *Big Data*
5. *Components*
6. *Task*

در هر بخش، ما جدولی شبیه جدول زیر ارائه کرده‌ایم:

مجوز	نام مجوز
فعالیت	بدون فعالیت، کم، متوسط، زیاد
هدف	شرح هدف
صفحه رسمی	آدرس سایت
یکپارچگی هدوپ	کاملاً یکپارچه، سازگار با API، فاقد یکپارچگی، غیرقابل اجرا

بیایید نگاه عمیق‌تری به هر یک از این دسته‌ها داشته باشیم:

مجوز

درحالی‌که همه بخش‌های موجود در اولین نسخه این راهنمای حوزه، متن‌باز[۷] هستند، اما مجوز[۸]های گوناگون زیادی همراه با نرم‌افزار (اساساً شبیه هم با تفاوت‌های اندک) وجود دارند. درصورتی‌که قصد دارید از این نرم‌افزار در یک محصول استفاده نمایید، بایستی با شرایط مجوز آشنا شوید.

فعالیت

ما تمام سعی خود را کرده‌ایم تا میزان توسعه فعال مربوط به این تکنولوژی را بسنجیم. این امکان وجود دارد که در بعضی موارد قضاوت نادرستی کرده باشیم و یا سطح فعالیت‌ها[۹] نسبت به زمانی که اقدام به نوشتن درباره‌ی این موضوع نمودیم، تغییر کرده باشد.

هدف

تکنولوژی چه کاری انجام می‌دهد؟ ما تلاش کرده‌ایم تا موضوعات را با توجه به اهداف مشترکشان گروه‌بندی نماییم و گاهی پی بردیم که یک موضوع می‌تواند در فصل‌های مختلفی قرار گیرد. زندگی یعنی انتخاب از میان گزینه‌ها؛ این‌ها انتخاب‌هایی هستند که ما انجام دادیم.

صفحه رسمی

اگر مسئولان این تکنولوژی، سایتی در اینترنت داشته باشند، صفحه خانگی پروژه محسوب می‌شود.

یکپارچگی هدوپ

زمانی که ما شروع به نوشتن کتاب کردیم، مطمئن نبودیم که دقیقاً چه موضوعاتی در اولین نسخه قرار داده می‌شوند. برخی موضوعات موجود در لیست اولیه با Apache Hadoop به‌شدت

---

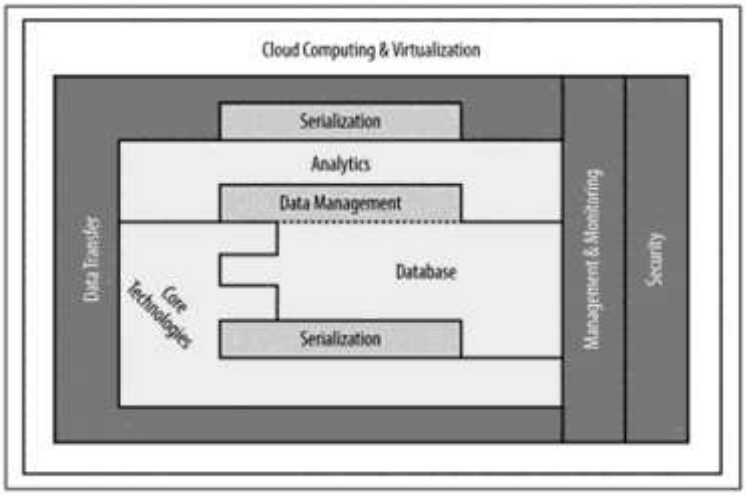
[7]. *Open Source*
[8]. *License*
[9]. *Activity*

درهم‌تنیده بودند. برخی عناوین دیگر، تکنولوژی‌های جایگزین یا تکنولوژی‌هایی بودند که اگرچه با هدوپ کار می‌کردند اما بخشی از خانواده Apache Hadoop نبودند. در آن موارد تلاش کردیم تا بهترین فهم از میزان یکپارچگی تکنولوژی یادشده در زمان نوشتن کتاب به دست آید. بدون شک این مطالب در طول زمان تغییر خواهند کرد.

تصور نکنید که این منبع، کتابی است که باید آن را از صفحه اول تا صفحه آخر بخوانید. اگر هدوپ برای شما کاملاً جدید است، باید مطالعه خود را از فصل مقدمه (فصل اول) آغاز کنید. سپس به دنبال موضوعات موردعلاقه‌تان باشید، بخش مربوط به آن مؤلفه را مطالعه کنید، عنوان فصل را بخوانید و در صورت امکان نگاهی اجمالی بر سایر گزینه‌های آن فصل داشته باشید. این روش به شما کمک می‌کند که درکی نسبت به موضوع به دست آورید. ما اغلب لینک‌هایی به سایر بخش‌های مرتبط با موضوع را در کتاب قرار داده‌ایم. به‌علاوه، ممکن است بخواهید به لینک‌های آموزشی مربوط به موضوع و یا به صفحه رسمی عنوان موردنظرتان نگاهی بیندازید.

ما بخش‌های این کتاب را مطابق الگوی نشان داده شده در شکل ۱ سازماندهی کرده‌ایم. اکثر عناوین در هدوپ عمومی (هسته هدوپ)، ابزارهای پایه و تکنیک‌های پشتیبانی ماژول[۱۰]های Apache Hadoop قرار گرفته‌اند. بااین‌حال، مجموعه ابزارهایی که نقش مهمی در اکوسیستم کلان داده‌ها ایفا می‌کنند، به تکنولوژی‌های موجود در هسته هدوپ محدود نمی‌شوند. در این کتاب تعدادی از تکنولوژی‌های مرتبط که نقش کلیدی در دورنما[۱۱]ی کلان داده‌ها دارند را نیز شرح داده‌ایم.

[10]. *Module*
[11]. *Landscape*



شکل ۱: نمای کلی از عناوین پوشش داده‌شده در این کتاب

در این نسخه (اول)، ما به اطلاعات مرتبط با توزیع‌های اختصاصی هدوپ نپرداخته‌ایم. ما پی بردیم که این پروژه‌ها مهم و مرتبط هستند، اما دورنمای تجاری به‌سرعت در حال تغییر است؛ بنابراین پیشنهاد ما تمرکز بر روی تکنولوژی متن‌باز است. در حال حاضر تکنولوژی متن‌باز، قدرتمندانه فروشگاه‌های کلان داده‌ها و هدوپ را تصرف کرده است و بسیاری از راه‌حل‌های تجاری اساساً بر مبنای تکنولوژی متن‌باز که در این کتاب شرح داده شده است، استوار هستند. آن دسته از خوانندگانی که علاقه‌مند به استفاده از تکنولوژی‌های متن‌باز توضیح داده شده در این کتاب هستند، می‌توانند در رابطه با توزیع‌های تجاری تکنولوژی‌های موردعلاقه خود مطالعه نمایند.

این کتاب سند ثابتی نیست که قرار باشد تنها هرسال یا هر دو سال به‌روزرسانی شود. هدف ما این است که آن را تا حد امکان به‌روز نگه داریم، با اضافه کردن محتوای جدید و یا جایگزین کردن تکنولوژی‌های پیشرفته‌تر و جدیدتر و کنار گذاشتن بعضی از تکنولوزی‌های قدیمی‌تر.

ازآنجایی‌که این موضوع به‌سرعت دچار تغییرات می‌شود، از خوانندگان دعوت می‌شود که پیشنهادات و نظرات خود در مورد مطالب درنظرگرفته شده در این کتاب را برای کوین (ksitto@gmail.com) و مارشال (bigmaish@gmail.com) ارسال نمایند.

قراردادهای مورداستفاده در این کتاب

قراردادهای چاپی که در این کتاب استفاده شده‌اند، عبارت‌اند از:

Italic (ایتالیک)

بیانگر واژه‌های جدید، آدرس سایت‌ها، آدرس ایمیل‌ها، اسامی فایل‌ها و پسوندهای فایل‌ها می‌باشد.

`Constant width`

جهت ارائه‌ی فهرست برنامه‌ها و همچنین درون پاراگراف‌ها برای اشاره به اجزای برنامه (نظیر اسامی توابع و متغیرها، پایگاه داده‌ها، انواع داده‌ها، متغیرهای محیطی، عبارات و کلمات کلیدی) به کار می‌رود.

`Constant width bold`

دستورات و سایر متن‌هایی را نشان می‌دهد که بایستی عیناً توسط کاربر نوشته شوند.

`Constant width italic`

نشان‌دهنده متن‌هایی است که باید با مقادیر موردنظر کاربر[۱۲] یا مقادیر تعیین‌شده به‌وسیله‌ی context جایگزین شوند.

از خوانندگان عزیز دعوت می‌نماییم با ارسال نظرات و انتقادات سازنده خود ما را در بهبود کیفیت کتاب در ویرایش بعدی یاری نمایند.

موفق باشید
تابش و نظری
بهار ۱۳۹۸
Nazarie951@mums.ac.ir
Tabeshh@mums.ac.ir

[12]. *User- supplied valuesl*

فصل اول:
تکنولوژی‌های هسته[۱۳]

در سال ۲۰۰۲، زمانی که شبکه جهانی وب نسبتاً جدید بود و قبل از اینکه شما بتوانید چیزی را از طریق گوگل جستجو کنید، داگ کاتینگ[۱۴] و مایک کافارلا[۱۵] غرق در وب بودند تا محتوای آن را اندیس‌گذاری کنند و بتوانند یک موتور جستجوی اینترنت[۱۶] تولید نمایند. برای انجام این کار آن‌ها پروژه‌ای به نام ناچ[۱۷] را آغاز کردند اما به روشی مقیاس‌پذیر[۱۸] برای ذخیره‌ی محتوای اندیس‌گذاری شان نیاز داشتند. روش استاندارد جهت سازماندهی و ذخیره داده‌ها در سال ۲۰۰۲ به‌وسیله سیستم‌های مدیریت پایگاه داده رابطه‌ای[۱۹] بود که از طریق زبانی به نام SQL می‌توانستند به آن دسترسی داشته باشند؛ اما تقریباًهمه‌ی ذخیره‌های رابطه‌ای و SQL برای ذخیره‌سازی و بازیابی[۲۰] موتور جستجوی اینترنت مناسب نبودند. آن‌ها به‌طور وحشتناکی مقیاس ناپذیر و پرهزینه بودند، در مواقع لزوم در برابر خطا، تحمل‌پذیری کمی داشتند و به‌طورکلی کارایی اندکی داشتند.

---

13. Core Technologies
14. Doug Cutting
15. Mike Cafarella
16. Internet Search Engine
17. Nutch
18. Scalable
19. Relational DataBase Management Systems (RDBMS)
20. Storage and Retrieval

در سال‌های ۲۰۰۳ و ۲۰۰۴ گوگل دو مقاله مهم را منتشر کرد. یکی از مقاله‌ها در مورد سیستم فایل گوگل و دیگری در مورد یک مدل برنامه‌نویسی برای سرورهای خوشه‌ای تحت عنوان MapReduce بود. کاتینگ و کافارلا از این دو تکنولوژی در پروژه خود استفاده کردند

و سرانجام هدوپ متولد شد. Hadoop یک سرواژه نیست. پسر کاتینگ یک فیل اسباب‌بازی زردرنگ به نام هدوپ داشت که این نام به طریقی روی پروژه گذاشته شد و آیکن[۱] پروژه نیز یک فیل کوچولوی بامزه است. یاهو به استفاده از هدوپ به‌عنوان پایه‌ی موتور جستجوی خود پرداخت و به‌سرعت استفاده از آن به بسیاری از سازمان‌های دیگر گسترش یافت. اکنون هدوپ به‌عنوان برجسته‌ترین پلت فرم کلان داده‌ها مطرح است. منابع بسیاری وجود دارند که هدوپ را با جزئیات زیاد شرح داده‌اند. در اینجا شما لیست مختصری از مؤلفه‌ها و ارجاع به منابعی را خواهید یافت که برای یادگیری بیشتر هدوپ مفید هستند.

هدوپ از سه منبع اصلی تشکیل شده است:

- سیستم فایل توزیع‌شده هدوپ[۲]
- پلت فرم[۳] برنامه‌نویسی MapReduce
- اکوسیستم هدوپ[۴]: مجموعه‌ای از ابزارهایی است که از MapReduce و HDFS برای

برای فرآیند ذخیره‌سازی و سازماندهی داده‌ها استفاده می‌کنند یا در کنار آن‌ها به این کارها می‌پردازند و ماشین‌هایی که هدوپ را اجرا می‌کنند را مدیریت می‌نمایند.

این ماشین‌ها، خوشه[۵] نامیده می‌شوند (گروهی از سرورها که تقریباً همیشه گونه‌ای از سیستم‌عامل لینوکس[۶] را اجرا می‌کنند) و به‌منظور اجرای یک فعالیت (وظیفه)[۷] باهم کار می‌کنند.

اکوسیستم هدوپ شامل ماژول‌هایی است که به برنامه‌ریزی سیستم، مدیریت و پیکربندی خوشه، مدیریت داده‌ها در خوشه، مدیریت ذخیره‌سازی در خوشه، اجرای فعالیت‌های تحلیلی و مانند آن کمک می‌کنند. اکثر ماژول‌های معرفی‌شده در این کتاب، به توضیح مؤلفه‌های اکوسیستم و تکنولوژی‌های مرتبط با آن‌ها پرداخته‌اند.

---

1. *Icon*
2. *The Hadoop Distributed File System (HDFS)*
3. *platforme*
4. *Hadoop Ecosystem*
5. *Cluster*
6. *linux*
7. *Task*

سیستم فایل توزیع‌شده هدوپ

مجوز	*Apache License, Version 2*
فعالیت	زیاد
هدف	ظرفیت بالا، تحمل‌پذیری خطا[۱]، ذخیره‌سازی مجموعه داده‌های بسیار بزرگ با هزینه مناسب
صفحه رسمی	http://hadoop.apache.org/docs/current/hadoop-project-dist/hadoop-hdfs/HdfsUserGuide.html
یکپارچگی هدوپ	کاملاً یکپارچه

سیستم فایل توزیع‌شده هدوپ (HDFS) مکانی در یک خوشه هدوپ است که داده‌ها در آنجا ذخیره می‌شوند. HDFS که برای کاربردهای داده های فشرده ساخته شده برای اجرا روی خوشه‌های سرورهای ارزان طراحی شده است. HDFS برای عملیات‌های خواندن سریع و دارای عملکرد بالا بهینه‌سازی می‌شود و در برابر شکست‌های[۲] رخ دهنده در خوشه مقاوم است. باوجودآنکه HDFS از وقوع شکست‌ها جلوگیری نمی‌کند، اما بعید است که هنگام بروز شکست، داده‌ای از بین رود زیرا طبق پیش‌فرض[۳] HDFS از بلوک‌های داده‌های خود کپی‌های متعددی تهیه می‌کند. HDFS یک سیستم فایل نوشتن یکبار، خواندن چندبار[۴] است: زمانی که یک فایل ایجاد شود، واسط برنامه‌نویسی کاربرد[۵] سیستم فایل تنها اجازه می‌دهد که مواردی را به فایل اضافه[۶] کنید، نه اینکه بر روی آن رونویسی[۷] انجام دهید. درنتیجه، HDFS معمولاً برای کاربردهای پردازش تراکنش برخط[۸] نرمال نامناسب است. بیشتر استفاده‌هایی که از HDFS می‌شود، مربوط به خواندن متوالی[۹] فایل‌های بزرگ است. این فایل‌ها به بلوک‌های بزرگ که

---

1. *Fault Tolerant*
2. *Failure*
3. *Default*
4. *Write Once Read Many (WORM)*
5. *Application Programming Interface*
6. *Append*
7. *Overwrite*
8. *Online Transaction Processing (OLTP) Applications*
9. *Sequential Read*

اندازه آن‌ها معمولاً ۶۴ مگابایت یا بیشتر است، تجزیه می‌شوند و این بلوک‌ها بین گره‌های[۱] موجود در سرور توزیع می‌شوند.

برخلاف آنچه در سیستم‌عامل‌های Linux، Mac OS X و برخی از پلت فرم‌های ویندوزی ملاحظه کرده‌اید، HDFS یک سیستم فایل سازگار با POSIX نیست (به‌منظور یافتن توضیحات مختصری در مورد POSIX، صفحه ویکی پدیای[۲] آن را ببینید). HDFS توسط کرنل‌های سیستم‌عامل موجود در گره‌های سرور مدیریت نمی‌شود. بلوک‌ها در HDFS به فایل‌هایی در سیستم فایل میزبان[۳] (در سیستم‌عامل لینوکس اغلب ext3)، نگاشت[۴] می‌شوند. HDFS دیسک‌های سیستم میزبان را تحت حفاظت RAID نمی‌داند؛ بنابراین طبق پیش‌فرض سه کپی از هر بلوک تهیه و بر روی گره‌های مختلف خوشه قرار داده می‌شوند. این مکانیزم، داده‌ها را در برابر از دست رفتن محافظت می‌کند. از دست رفتن داده‌ها هنگامی اتفاق می‌افتد که گره‌ها یا دیسک‌ها در زمان انجام فعالیت با شکست مواجه شوند. به‌علاوه، مکانیزم یاد شده طرح هدوپ دسترسی[۵] به داده‌ها در همان‌جایی که قرار دارند را تسهیل می‌نماید (به‌جای جابجایی آن‌ها در یک شبکه جهت دسترسی).

اگرچه توضیح این مطلب فراتر از حوزه این کتاب است، اما جالب است بدانید فراداده[۶] مربوط به فایل‌های موجود در HDFS از طریق یک گره نام[۷] مدیریت می‌شود. گره نام معادل سوپر بلوک هدوپ سیستم‌های یونیکس[۸] یا لینوکس است.

لینک‌های آموزشی

1. *Node*
2. *Wikipedia*
3. *Host*
4. *Mapping*
5. *Access*
6. *Meta Data*
7. *Name Node*
8. *Unix*

در آینده بارها با HDFS از طریق ابزارهایی نظیر Hive یا Pig تعامل خواهید داشت. به‌عبارت‌دیگر، زمان‌های زیادی وجود خواهد داشت که بخواهید به‌طور مستقیم با HDFS کار کنید. یاهو یک راهنمای عالی برای پیکربندی[۱] و بررسی[۲] یک سیستم پایه منتشر کرده است.

کد مثال

زمانی که از واسط خط فرمان[۳] یک سرویس‌گیرنده[۴] هدوپ استفاده کنید، می‌توانید یک فایل از سیستم فایل محلی[۵] خود را بر روی HDFS کپی کنید. سپس به ۱۰ خط اول مطابق با کد زیر نگاه کنید.

```
[hadoop@client-host ~]$ hadoop fs -ls /data
Found 4 items
drwxr-xr-x - hadoop supergroup 0 2012-07-12 08:55 /data/faa
-rw-r--r-- 1 hadoop supergroup 100 2012-08-02 13:29
/data/sample.txt
drwxr-xr-x - hadoop supergroup 0 2012-08-09 19:19 /data/wc
drwxr-xr-x - hadoop supergroup 0 2012-09-11 11:14 /data/weblogs
[hadoop@client-host ~]$ hadoop fs -ls /data/weblogs/
[hadoop@client-host ~]$ hadoop fs -mkdir /data/weblogs/in
[hadoop@client-host ~]$ hadoop fs -copyFromLocal
weblogs_Aug_2008.ORIG /data/weblogs/in
[hadoop@client-host ~]$ hadoop fs -ls /data/weblogs/in
Found 1 items
-rw-r--r-- 1 hadoop supergroup 9000 2012-09-11 11:15
/data/weblogs/in/weblogs_Aug_2008.ORIG
[hadoop@client-host ~]$ hadoop fs -cat
/data/weblogs/in/weblogs_Aug_2008.ORIG \
| head
```

[1] *Configuration*
[2] *Exploring*
[3] *Command Line Interface (CLI)*
[4] *Client*
[5] *Local*

10.254.0.51 - - [29/Aug/2008:12:29:13 -0700] "GGGG / HTTP/1.1"
200 1456
10.254.0.52 - - [29/Aug/2008:12:29:13 -0700] "GET / HTTP/1.1"
200 1456
10.254.0.53 - - [29/Aug/2008:12:29:13 -0700] "GET /apache_pb.gif
HTTP/1.1" 200 2326
10.254.0.54 - - [29/Aug/2008:12:29:13 -0700] "GET /favicon.ico
HTTP/1.1" 404 209
10.254.0.55 - - [29/Aug/2008:12:29:16 -0700] "GET /favicon.ico
HTTP/1.1"
404 209
10.254.0.56 - - [29/Aug/2008:12:29:21 -0700] "GET /mapreduce
HTTP/1.1" 301 236
10.254.0.57 - - [29/Aug/2008:12:29:21 -0700] "GET /develop/
HTTP/1.1" 200 2657
10.254.0.58 - - [29/Aug/2008:12:29:21 -0700] "GET
/develop/images/gradient.jpg
HTTP/1.1" 200 16624
10.254.0.59 - - [29/Aug/2008:12:29:27 -0700] "GET /manual/
HTTP/1.1" 200 7559
10.254.0.62 - - [29/Aug/2008:12:29:27 -0700] "GET
/manual/style/css/manual.css
HTTP/1.1" 200 18674

MapReduce

مجوز	مجوزApache ، ورژن ۲/۰
فعالیت	زیاد
هدف	یک رویکرد برنامه‌نویسی جهت پردازش کلان داده‌ها
صفحه رسمی	https://hadoop.apache.org
یکپارچگی هدوپ	کاملاً یکپارچه

MapReduce چارچوب اصلی برنامه‌نویسی و نخستین روش جهت توسعه کاربردها در هدوپ بود. برای استفاده از MapReduce در قالب اصلی‌اش نیاز خواهید داشت که با Java کار کنید. بهتر است که برنامه شمارش کلمه Hello World هدوپ را مطالعه فرمایید. کد این برنامه برای کلیه توزیع‌های هدوپ موجود است. مسئله شما در برنامه شمارش کلمه این است: شما یک مجموعه داده‌ها در اختیار دارید که شامل مجموعه بزرگی از اسناد می‌باشد. هدف این است که لیستی از کل کلمات به همراه تعداد دفعات ظاهر شدن آن‌ها در مجموعه داده‌ها تهیه شود.

کارهای MapReduce از دو برنامه Java به نام‌های نگاشت کننده[۱] و کاهش‌دهنده[۲] تشکیل شده است. با هماهنگی نرم‌افزار هدوپ، هر یک از نگاشت کننده‌ها دسته‌هایی[۳] از داده‌ها را برای تحلیل در اختیار دارند. فرض کنید یک جمله داریم:

The dog ate the food

در این حالت نگاشت کننده پنج نقشه[۴] یا زوج نام ـ مقدار[۵] را به‌عنوان خروجی تولید می‌کند:

“the”: 1, “dog”: 1, “ate”: 1, “the”: 1, “food”: 1

در زوج نام ـ مقدار، قسمت نام نشان‌دهنده کلمه و قسمت مقدار بیانگر تعداد دفعاتی می‌باشد که کلمه در متن مشاهده شده است. هدوپ این خروجی را می‌گیرد و آن را مرتب[۶] می‌کند. به ازای هر نقشه، یک مقدار درهم سازی شده[۷] ایجاد می‌گردد و در مرحله‌ای که نام آن

---

1. *Mapper*
2. *Reducer*
3. *Chunk*
4. *Map*
5. *Name-Value Pair*
6. *Sort*
7. *Hash value*

مخلوط کردن[1] است به کاهش‌دهنده اختصاص داده می‌شود. کاهش‌دهنده برای هر کلمه‌ای که در جریان[2] ورودی‌اش قرار دارد، نقشه‌ها را باهم جمع می‌کند و لیست دسته بندی شده‌ای از کلمات موجود در سند تولید می‌نماید. شما می‌توانید نگاشت کننده‌ها را به‌عنوان برنامه‌هایی در نظر بگیرید که داده‌ها را از فایل‌های HDFS استخراج نموده و آن‌ها را به نقشه‌ها منتقل می‌کنند. به‌علاوه، کاهش‌دهنده‌ها برنامه‌هایی هستند که خروجی را از نگاشت کننده‌ها تحویل گرفته و نتایج را باهم جمع می‌کنند[3]. لینک‌های آموزشی که در بخش بعدی معرفی می‌شوند، این روال را با جزئیات بیشتری توضیح داده‌اند.

بسیار خوشحال خواهید شد وقتی بفهمید که اکثر کارهای سخت (تقسیم مجموعه‌های داده‌های ورودی، تخصیص نگاشت کننده‌ها و کاهش‌دهنده‌ها به گره‌ها، آمیختن داده‌های موجود در نگاشت کننده‌ها با کاهش‌دهنده‌ها، ثبت نتایج نهایی در HDFS) توسط خود هدوپ مدیریت می‌شوند. برنامه‌نویس‌ها صرفاً باید توابع نگاشت و کاهش را بنویسند. نگاشت کننده‌ها و کاهش‌دهنده‌ها معمولاً در Java نوشته می‌شوند (همان‌طور که در مثال موجود در قسمت نتیجه این بخش ذکر شده است) و نوشتن کد MapReduce برای افراد تازه کار، پر از نکته و پیچیدگی است. برای این مورد ابزارهای سطح بالا تهیه شده است. یکی از مثال‌ها Pig است. مثال دیگری در این مورد، جریان هدوپ[4] می‌باشد.

لینک‌های آموزشی

1. *Suffle*
2. *Stream*
3. *Aggregate*
4. *Hadoop Streaming*

تعدادی زیادی از آموزش‌های بسیار خوب در مورد کار با MapReduce وجود دارند. یک نقطه‌ی خوب برای شروع، مستندسازی آپاچی رسمی است. یاهو نیز یک ماژول آموزشی منتشر کرده است. گروه MapR (یک شرکت نرم‌افزاری تجاری می‌باشد که توزیعی از هدوپ را تهیه کرده است) ارائه بسیار خوبی در رابطه با نوشتن MapReduce دارد.

کد مثال

نوشتن MapReduce می‌تواند نسبتاً پیچیده باشد و فراتر از حوزه این کتاب می‌باشد. یک نمونه کاربردی از MapReduce که افراد برای شروع کدنویسی می‌کنند، برنامه ساده شمارش کلمه است. مستندات رسمی حاوی آموزش ایجاد این برنامه می‌باشد.

YARN

مجوز	مجوزApache ، ورژن[1] ۲/۰
فعالیت	متوسط
هدف	پردازش
صفحه رسمی	*https://hadoop.apache.org/docs/current/hadoop-yarn/hadoop-yarn-site/YARN.html*
یکپارچگی هدوپ	کاملاً یکپارچه

زمانی که بسیاری از افراد هدوپ را بررسی می‌کنند، درواقع در حال فکر کردن به دو تکنولوژی مرتبط با آن هستند. این دو تکنولوژی عبارت‌اند از: سیستم فایل توزیع‌شده هدوپ (HDFS) که مکانی برای استقرار داده‌هاست و MapReduce که امکان انجام کار با داده‌ها را فراهم می‌نماید. باوجودآنکه MapReduce برای دسته معینی از کارها واقعاً خوب عمل می‌کند، اما در سایر کارها عملکرد مطلوبی ندارد. این امر منجر به ازهم‌گسیختگی در اکوسیستم و برخی ابزارها می‌شود. ابزارهای مذکور، آن‌هایی هستند که در خارج از خوشه هدوپ مشغول به فعالیت هستند اما جهت ارتباط با HDFS تلاش می‌کنند.

در ماه مِی[2] سال ۲۰۱۲، نسخه ۲ هدوپ منتشر شد. با آمدن این نسخه، تغییر مهیجی در روش تقابل شما با داده‌ها رخ داد. این تغییر با معرفی YARN[3] همراه شد.

YARN در فضای داده‌های شما و جایی که هم‌اکنون MapReduce در آنجا قرار دارد، وجود دارد. YARN برای بسیاری از ابزارهای دیگر که قبلاً در خارج از سیستم هدوپ قرار داشتند (نظیر Spark و Giraph)، امکانی را فراهم می‌کند که اکنون در درون یک خوشه هدوپ واقع شوند. مهم است بدانید که YARN جایگزینی برای MapReduce نیست. درواقع YARN به‌خودی‌خود هیچ کاری انجام نمی‌دهد. کاری که YARN انجام می‌دهد عبارت است از

---

1. *Version*
2. *May*
3. *Yet Another Resource Negotiator*

فراهم کردن یک روش یکپارچه[1] و راحت برای انواع ابزارها نظیر MapReduce، HBase یا هر برنامه سودمند سفارشی[2] که احتمالاً به‌منظور اجرا بر روی خوشه هدوپ خود ساخته‌اید.

لینک‌های آموزشی

YARN هنوز یک تکنولوژی در حال رشد می‌باشد و «راهنمای رسمی آپاچی» حقیقتاً بهترین مکان برای شروع است.

کد مثال

حقیقت آن است که نوشتن برنامه‌ها در YARN همچنان مبهم و پیچیده و سطح آن برای این کتاب بسیار بالا است. بااین‌حال شما می‌توانید در بخش قبلی (لینک‌های آموزشی) لینکی عالی برای ساخت اولین برنامه‌تان در YARN پیدا کنید.

---

1. *Uniform*
2. *Custom Utility*

Spark

مجوز	مجوزApache ، ورژن ۲/۰
فعالیت	زیاد
هدف	ذخیره‌سازی / پردازش
صفحه رسمی	http://spark.apache.org
یکپارچگی هدوپ	سازگار با API

MapReduce نخستین تکنولوژی حائز اهمیت در هسته اکثر خوشه‌های هدوپ است. باوجود تأثیر بالای MapReduce در کارهای تحلیل دسته‌ای بسیار بزرگ[۱]، ثابت شده است که این تکنولوژی برای کاربردهایی مثل تحلیل گراف که به پردازش و اشتراک داده تکراری[۲] نیاز دارند، زیربهینه[۳] می‌باشد.

Spark طراحی شد تا مدلی با انعطاف بیشتر فراهم کند به‌گونه‌ای که بتواند بسیاری از کاربردهای چندگذری[۴] که در MapReduce متزلزل بودند را پشتیبانی نماید. این هدف با بهره‌گیری از مزایای حافظه در هر زمان ممکن جهت کاهش حجم داده‌ها برای نوشتن بر روی دیسک و خواندن آن‌ها از دیسک تحقق می‌یابد. برخلاف Pig و Hive، تکنولوژی Spark ابزاری نیست که استفاده از MapReduce را آسان‌تر کند. Spark یک جایگزین کامل برای MapReduce محسوب می‌شود، به‌طوری‌که موتور اجرای کار[۵] خودش را دارد.

Spark با در نظر داشتن سه ایده اصلی کار می‌کند:

---

1. *Very large batch-analytic jobs*
2. *Iterative processing and data sharing*
3. *Suboptimal*
4. *Multi-pass applications*
5. *Work execution engine*

مجموعه داده‌های توزیع شده واکنش‌گرا (انعطاف‌پذیر)[1]

RDD حاوی داده‌هایی است که می‌خواهید آن‌ها را تبدیل[2] یا تحلیل کنید. این داده‌ها می‌توانند از یک منبع خارجی (نظیر یک فایل یا یک پایگاه داده) خوانده شوند یا اینکه به‌وسیله‌ی تبدیل به وجود آیند.

تبدیل

یک تبدیل، RDD موجود را جهت ایجاد یک RDD جدید اصلاح می‌نماید. به‌عنوان‌مثال فیلتری که پیام‌های خطا را از یک فایل سابقه[3] استخراج می‌کند، یک تبدیل محسوب می‌شود.

اقدام

اقدام[4]، یک RDD را تحلیل می‌کند و یک نتیجه مشخص را برمی‌گرداند. برای مثال، یک اقدام می‌تواند تعداد نتایج شناسایی‌شده به‌وسیله‌ی فیلتر ERROR ما را بشمارد.

اگر می‌خواهید کار مهمی در Spark انجام دهید، بهتر است که در مورد Scala (یک زبان برنامه‌نویسی تابعی[5]) اطلاعات کاملی کسب کنید. Scala برنامه‌نویسی تابعی را با شئ گرایی[6] ترکیب می‌کند. ازآنجایی‌که Lisp یک زبان برنامه‌نویسی تابعی قدیمی‌تر است، ممکن است Scala «الحاق Lisp به قرن بیست و یکم[7]» نیز نامیده شود. این مطلب به آن معنی نیست که Scala تنها راه برای کار با Spark است. به‌علاوه، این پروژه دارای پشتیبانی قدرتمندی برای جاوا[8] و پایتون[9] می‌باشد؛ اما زمانی که واسط‌های برنامه‌نویسی کاربردی[10] یا ویژگی‌های جدید اضافه می‌گردند، ابتدا در Scala ظاهر می‌شوند.

---

1. *Resilient Distributed Dataset (RDD)*
2. *Transform*
3. *Log file*
4. *Action*
5. *Functional*
6. *Object Orientation*
7. *Lisp joins the 21st century*
8. *java*
9. *python*
10. *Application Programming Interface (API)*

لینک‌های آموزشی

در صفحه اصلی سایت پروژه، یک «شروع سریع» برای Spark وجود دارد.

کد مثال

این مثال را با باز کردن پوسته[1] Spark از طریق اجرای /bin/spark-shell از همان دایرکتوری که Spark آنجا نصب شده است، آغاز می‌کنیم.

در این مثال قصد داریم که تعداد مشاهدات کلمه Dune در فایل یادآوری[2] خود را بشماریم:

```
// Read the csv file containing our reviews
scala> val reviews = spark.textFile("hdfs://reviews.csv")
testFile: spark.RDD[String] = spark.MappedRDD@3d7e837f
// This is a two-part operation:
// first we'll filter down to the two
// lines that contain Dune reviews
// then we'll count those lines
scala> val dune_reviews = reviews.filter(line =>
line.contains("Dune")).count()
res0: Long = 2
```

[1]. Shell
[2]. Review file

فصل دوم:

مدیریت داده‌ها و پایگاه داده‌ها

اگر قصد دارید از هدوپ استفاده کنید، احتمالاً می‌خواهید حجم زیادی از داده‌ها را مدیریت نمایید و علاوه بر کارهایی که MapReduce انجام می‌دهد، به گونه‌هایی از پایگاه داده‌ها احتیاج دارید. از زمان ظهور جدول بزرگ گوگل[۱]، هدوپ به مدیریت داده‌ها توجه داشته است. در حالی که تعداد زیادی از پایگاه داده‌های رابطه‌ای SQL یا واسطهای SQL برای داده‌های HDFS نظیر Hive وجود دارند، اما اکثر روش‌های مدیریت داده‌ها در هدوپ از تکنیک‌های غیر SQL جهت ذخیره‌سازی و دسترسی به داده‌ها استفاده می‌کنند. بایگانی غیر SQL [۲] بیش از ۱۵۰ پایگاه داده غیر SQL را لیست می‌کند که در گروه‌های زیر دسته‌بندی می‌شوند:

- ذخیره‌سازی‌های ستونی
- ذخیره‌سازی‌های سند
- ذخیره‌سازی‌های تاپل / کلید ـ مقدار[۳]
- پایگاه داده‌های گراف
- پایگاه داده‌های چندمدلی[۴]

---

1. *Google's BigTable*
2. *NoSQL Archive*
3. *Key-value / tuple stores*
4. *Multimodel databases*

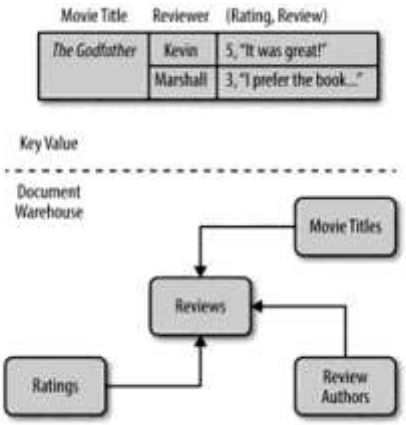
- پایگاه داده‌های شئ
- پایگاه داده‌های ابری و شبکه‌ای[۱]
- پایگاه داده‌های چندمقداری[۲]
- ذخیره‌سازی‌های جدولی[۳]
- سایر

پایگاه داده‌های غیر SQL عموماً از عملگرهای الحاق رابطه‌ای[۴]، تراکنش‌های پیچیده یا قیدهای کلید خارجی[۵] که در سیستم‌های رابطه‌ای رایج هستند، پشتیبانی نمی‌کنند؛ اما با حجم انبوهی از داده‌ها تناسب بهتری دارند. شما باید تصمیم بگیرید کدام یک از انواع پایگاه داده‌ها برای مجموعه داده‌ها و اطلاعاتی که می‌خواهید از آن‌ها استخراج کنید، بهترین کارایی را به همراه خواهد داشت. این امکان وجود دارد که برای رسیدن به بهترین کارایی از چند پایگاه داده استفاده نمایید.

کتاب حاضر، مثال‌های آموزشی زیادی را در هر بخش بررسی نموده است، اما تمرکز اصلی بر روی دو دسته عمده خواهد بود: ذخیره‌سازی‌های کلید ـ مقدار و ذخیره‌سازی‌های سند (در شکل 2-1 نشان داده شده است) .

---

1. *Grid and cloud databases*
2. *Multivalue databases*
3. *Tabular stores*
4. *Relational Joins*
5. *Foreign-key constraints*



شکل ۲-۱: روش‌های شاخص‌گذاری

یک ذخیره‌سازی کلید ـ مقدار را می‌توان به‌عنوان یک کاتالوگ در نظر گرفت. همه اقلامی که در یک کاتالوگ قرار دارند (مقدارها) بر اساس یک اندیس (کلیدها) سازماندهی شده‌اند. دقیقاً همانند کاتالوگ اگر بدانید که به دنبال کدام کلید هستید، یک ذخیره‌سازی کلید ـ مقدار بسیار سریع و مؤثر است؛ درحالی‌که اگر کلیدی برای جستجو نداشته باشید، یک ذخیره‌سازی کلید ـ مقدار کمک چندانی نمی‌کند.

برای مثال، فرض کنید من به دنبال بازنگری مارشال برای فیلم پدرخوانده[1] هستم. در اینجا من می‌توانم سریعاً به اندیس خودم رجوع کنم، کلیه بازنگری‌های مربوط به این فیلم را پیدا کنم و نتایج را پیمایش کنم تا به بازنگری مارشال برسم: من کتاب را ترجیح می‌دهم ...

از سوی دیگر، یک انبار سند، نوعی پایگاه داده منعطف‌تر محسوب می‌شود. انبار سند بیش از آنکه شما را به سازماندهی داده‌هایتان بر اساس یک کلید مشخص مجبور کند، این امکان را فراهم می‌کند تا داده‌ها را اندیس‌گذاری کنید و به جستجوی آن‌ها بر اساس تعدادی از پارامترها بپردازید. بیایید مثال قبلی را بسط دهیم و بگوییم که می‌خواهیم فیلمی تماشا کنیم که آن فیلم بر اساس یک کتاب باشد. یک راه ساده برای یافتن چنین فیلمی، جستجو برای یافتن بازنگری‌هایی است که حاوی کلمه کتاب هستند. ازآنجایی‌که در این مورد کلید به‌طور شفاف تعیین نشده است، یک ذخیره‌سازی کلید ـ مقدار نمی‌تواند کمک چندانی نماید. چیزی که در اینجا موردنیاز می‌باشد، یک انبار سند است که امکان جستجوی سریع کلیه متن‌های بازنگری‌ها و یافتن آن‌هایی را که حاوی کلمه کتاب هستند را فراهم کند.

---

[1]. *The Godfather*

Cassandra

مجوز	GPL v2
فعالیت	زیاد
هدف	ذخیره‌سازی کلید ـ مقدار
صفحه رسمی	http://cassandra.apache.org
یکپارچگی هدوپ	سازگار با API

ممکن است خیلی اوقات خواسته باشید که به‌سادگی برخی از کلان داده‌های خود را به‌منظور بازیابی آسان[1]، سازماندهی کنید. یک روش رایج برای انجام این کار، استفاده از ذخیره‌سازی کلید ـ مقدار می‌باشد. این نوع از پایگاه داده شبیه صفحات سفید یک دفترچه تلفن است. داده‌های شما بر اساس یک کلید منحصربفرد سازماندهی می‌شوند و مقادیر به کلیدهای خود پیوند زده می‌شوند. به‌عنوان‌مثال اگر شما بخواهید اطلاعات مشتریان خود را به این روش ذخیره کنید، می‌توانید نام کاربری مشتریان را به‌عنوان کلید و اطلاعات آن‌ها نظیر تاریخچه تراکنش و آدرس ایشان را به‌عنوان مقادیر پیوند زده شده با آن کلید، در نظر بگیرید.

ذخیره‌سازی داده‌ها به روش کلید ـ مقدار یک عضو ثابت شناخته‌شده در هر سیستم کلان داده‌ها محسوب می‌شود، زیرا این روش مقیاس‌پذیر و سریع است و دارای کاربرد آسان می‌باشد. Cassandra یک پایگاه داده کلید ـ مقدار توزیع شده است که بر مبنای سهولت و مقیاس‌پذیری طراحی شده است. باوجودآنکه Cassandra اغلب با HBase مقایسه می‌شود، اما در بعضی موارد کلیدی متفاوت می‌باشد:

[1]. *Easy retrieval*

- Cassandra یک سیستم کامل[۱] است؛ به این معنی که به یک محیط هدوپ یا هر ابزار کلان داده دیگر نیازی ندارد.
- Cassandra کاملاً بدون مدیر[۲] است؛ یعنی به‌عنوان یک سیستم همتا به همتا[۳] عمل می‌کند. این ویژگی Cassandra را قادر می‌سازد تا به‌آسانی تنظیم شود و واکنش‌گرایی بالایی داشته باشد.

لینک‌های آموزشی

کمپانی DataStax که پشتیبانی تجاری را برای Cassandra فراهم می‌کند، مجموعه‌ای از ویدئوهای رایگان را در این زمینه ارائه نموده است.

کد مثال

آسان‌ترین روش برای تعامل با Cassandra از طریق واسط پوسته آن است. برای آغاز کار با پوسته، bin/cqlsh از دایرکتوری نصب را اجرا کنید.

سپس باید یک فضای کلید[۴] ایجاد نمایید. فضاهای کلید شبیه شِماها[۵] در پایگاه داده‌های رابطه‌ای سنتی هستند. درواقع، فضاهای کلید روشی آسان جهت سازماندهی جداول شِما محسوب می‌شوند. یک الگوی معمولی به‌منظور استفاده از یک فضای کلید متفاوت برای هر کاربرد به شرح زیر است:

```
CREATE KEYSPACE field_guide
WITH REPLICATION = {
'class': 'SimpleStrategy', 'replication factor' : 3 };

USE field_guide;
```

اکنون شما یک فضای کلید دارید و می‌توانید جدولی درون این فضای کلید ایجاد نمایید تا بازنگری‌های خود را در آنجا نگه دارید. این جدول سه ستون و یک کلید اصلی خواهد داشت.

---

1. *All-inclusive*
2. *Master-less*
3. *Peer to peer*
4. *Keyspace*
5. *Schemas*