مقدمه

هدف از این تمرین آشنایی شما با چهار دیتابیس **NoSQL** محبوب یعنی کاساندرا، مانگودی‌بی ، نئوفورجی و الستیک سرچ است.

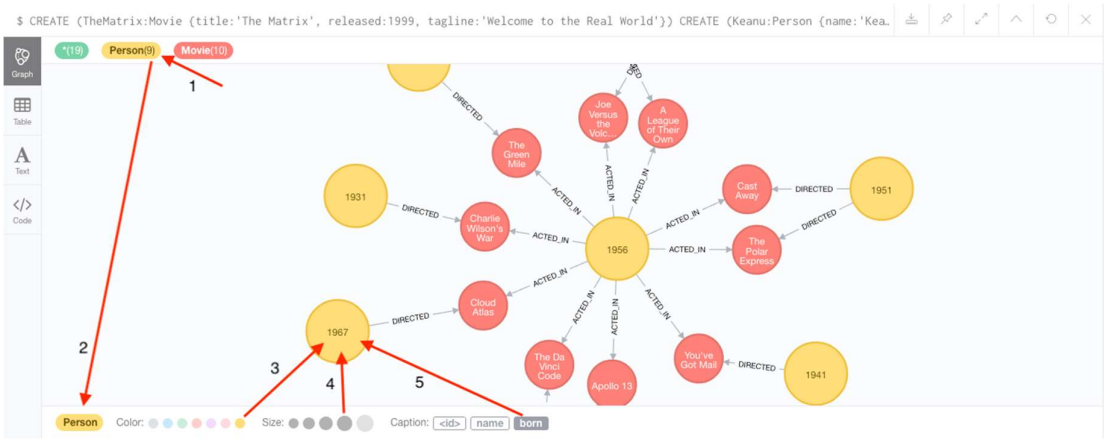
در این تمرین با چهار سامانه مدیریت بانک‌های اطلاعاتی **NoSQL** زیر کار خواهید کرد :

1. **MongoDB** به عنوان نماینده بانک‌های اطلاعاتی **Document Oriented**
2. **Cassandra** به عنوان نماینده بانک‌های اطلاعاتی سطرگسترده (**Wide Row**)
3. **Neo4j** به عنوان نماینده بانک‌های اطلاعاتی گراف‌محور
4. **Elasticsearch** به عنوان نماینده سرچ‌انجین‌های متنی

در این تمرین می‌توانید از نسخه‌های داکر و یا نصبی و حتی نسخه های آنلاین این دیتابیس‌ها (سندباکس نئوفورجی) استفاده کنید (سعی کنید از آخرین نسخه هر کدام از این دیتابیس ها استفاده کنید). برای هر یک از دیتابیس‌های این تمرین، توصیه میشود از ابزارهای زیر استفاده کنید :

مانگودی‌بی :Mongo Compass / Robo3T

نئوفورجی : Neo4j Browser[1]



کاساندرا : ترجیحا کار با خود **CLI** کاساندرا و یا استفاده از **DBSchema**[2] – نسخه سازمانی **DBeaver** هم از کاساندرا پشتیبانی می کند که می‌توانید دوره **trial** آنرا فعال و با کاساندرا به راحتی کار کنید. (البته از طریق **jdbc** و افزودن درایورمربوطه، با نسخه رایگان **DBeaver** هم میتوانید با کاساندرا از طریق **JDBC** کار کنید که تنظیمات درایور آن مشابه با تنظیمات **DBSchema** خواهد بود)

[1] https://neo4j.com/developer/neo4j-browser/
[2] https://github.com/wise-coders/cassandra-jdbc-driver

الستیک سرچ :محیط کیبانا که معمولا همراه با الستیک سرچ نصب میشود.

سوالات و مسایل خود را در گروه تلگرامی درس مطرح کنید تا توسط استاد درس، دستیاران و یا سایر دانشجویان پاسخ داده شود.

دقت کنید که بانک‌های اطلاعاتی NoSQL را معمولاً به عنوان دیبابیس‌های جانبی (Not Only SQL) و بسته به نیاز خاصی که در کسب‌وکار داریم استفاده می‌کنیم. بنابراین آنها را به عنوان جایگزین‌های SQL در نظر نگیرید.(No SQL)

با توجه به این که بخش‌های مختلف این تمرین به همدیگر مربوط نیستند می توانید طبق ترتیب زیر (از ساده به سخت) آنها را انجام دهید :

- **کار با الستیک سرچ**
- **کار با مانگودی‌بی**
- **دیتابیس نئوفورجی**
- **کار با کاساندرا**

بخش اول - ذخیره اطلاعات بدون ساختار / کار با MongoDB

در بخش اول تمرین، برای ذخیره توییت‌های سایت سهامیاب از مانگو استفاده میکنیم و بعد از ذخیره اطلاعات، با انجام چند پرس و جوی ساده، نحوه کار با این دیتابیس محبوب را فراخواهیم گرفت.

دریافت اطلاعات

با استفاده از https://www.sahamyab.com/guest/twiter/list?v=0.1 ده توییت آخر سایت سهامیاب با تمامی مشخصات را در فرمت جی سان دریافت میکنیم (با پستمن با روش GET می‌توانید خروجی را تست کنید). توییت‌ها در فیلد **items** پاسخ ، قابل مشاهده هستند. برای این تمرین، به کمک API فوق به جمع‌آوری و پردازش توئیت‌های فارسی خواهیم پرداخت.

برای دریافت اطلاعات میتوانید از کد زیر استفاده کنید:

```
import requests, json
response = requests.get('https://www.sahamyab.com/guest/twiter/list?v=0.1', headers={'User-Agent': 'Chrome/61'})
data = json.loads(response.text)
```

نصب مانگو و ساخت کالکشن توییت ها

مانگو دی بی را نصب کرده[1] و کالکشن **tweets** را در دیتابیس **sahamyab** (این دیتابیس هم باید ایجاد شود) بسازید. میتوانید از خط فرمان مانگودی‌بی یا ابزارهای گرافیکی رایج مانند **MongoDB Compass**[2] برای این منظور استفاده کنید.

علاوه بر کتاب درسی معرفی شده، کتاب کوچک **The Little MongoDB**[3] می‌تواند راهنمای سریع شما برای کار با مانگو در این تمرین باشد.

گام اول تمرین

در این گام، با فراخوانی آدرس https://www.sahamyab.com/guest/twiter/list?v=0.1 ده توییت آخر را دریافت کرده و به صورت دستی در مانگو ذخیره کنید (از خروجی پستمن هم می‌توانید در این مرحله استفاده کنید و نیاز به کدنویسی نخواهد بود) و بررسی کنید چه فیلدهایی توسط خود مانگو به صورت خودکار به داده‌ها افزوده میشود.(هر توئیت را به عنوان یک داکیومنت ذخیره کنید یعنی با فراخوانی کدفوق، ده توئیت را ذخیره خواهیم کرد.)

سپس با استفاده از کتابخانه **pymongo**[1] کد دریافت اطلاعات فوق را به گونه‌ای تغییر دهید که هر یک **دقیقه یکبار**، توئیت‌های جدید را دریافت کرده و همزمان با دریافت توییت‌ها، آنها را در مانگو هم ذخیره کند.(دقت کنید که هر توئیت باید جداگانه ذخیره شود و توییت‌های تکراری بر اساس فیلد id هم باید حذف شوند که البته می‌توانید upsert کنید)

[1] https://bit.ly/2XWSqM7
[2] https://www.mongodb.com/products/compass
[3] https://openmymind.net/mongodb.pdf

کد نوشته شده را تا زمانی اجرا کنید که حداقل هزار توئیت منحصر بفرد در مانگو ذخیره شده باشند. با دستور count، مطمئن شوید که هزار توییت ذخیره شده باشد.

خروجی گام اول

نحوه ورود دستی داده‌ها در مانگو و فیلدهای اضافه شده، کدهای نوشته شده برای درج اطلاعات و نحوه اطمینان از درج هزار توییت در گزارش آورده شود.

گام دوم - پیش پردازش داده

در این گام با استفاده از **Regex** هشتگ های استفاده شده کاربر در فیلد **content** را پیدا کرده و سپس با استفاده از دستور **update** در فیلدی به نام **hashtags** به صورت **Array** ذخیره کنید.

حروف ک و ی عربی موجود در فیلد **content** را با حروف معادل فارسی جایگزین کنید.

خروجی گام دوم

دستور نوشته شده، خروجی و زمان اجرا

گام سوم - دستورات اصلی

1. نام کاربرانی که **mediaContentType** توییت آنها **image/jpeg** هستند و **parentId** آنها مقدار دارد را بیابید.
2. قصد داریم فیلد **gov** با مقدار **true** را به توئیت‌های حاوی هشتگ شبندر، شستا و فولاد اضافه کنیم. این کار را چگونه انجام می‌دهید ؟
3. قصد داریم به کسانی که در بازه ساعت نه تا ده صبح، توییت کردند جایزه بدهیم **senderName** و **senderProfileImage** این کاربران را بیابید.(در صورت نبود توییت در این بازه، بازه دلخواه دیگری انتخاب کنید.)

خروجی گام سوم

دستور نوشته شده، خروجی و زمان اجرا

گام چهارم - دستورات تجمعی و آماری (Aggregate Functions)

1. میخواهیم کاربران را بر اساس فعالیتشان دسته بندی کنیم. کاربران را به سه دسته به صورت زیر تقسیم کنید:
   کاربرانی با یک توییت، کاربرانی با دو تا سه توییت، کاربرانی با بیش از سه توییت
   دستوری بنویسید که تعداد هر گروه را بر گرداند.
2. تعداد توییت های هر هشتگ را بشمارید و به صورت نزولی رتبه بندی کنید.

[1]https://pymongo.readthedocs.io/en/stable/tutorial.html

3. برای توییت‌هایی که **parentId** دارند، فیلد type را حذف کنید.
4. پرتکرارترین و کم‌تکرارترین هشتگ را بیابید.
5. ده هشتگ پر استفاده هر روز را بیابید. (بازه زمانی جزء ورودی های کوئری خواهد بود).
6. فعالترین کاربر هر روز را به همراه تعداد توئیت‌های انجام شده، پیدا کنید.

خروجی گام چهارم

دستور نوشته شده، خروجی و زمان اجرا

گام پنجم - بررسی کارآیی شاخص‌ها

با توجه به ساختار انعطاف پذیر مانگو، استفاده از شاخص‌ها در فیلدهایی که در جستجوها، به کرات استفاده میشوند نقش مهمی در کارآیی برنامه ما خواهد داشت. برای برخی از سوالات فوق که در زیر تعیین شده است ابتدا مشخص کنید چه شاخصی باید ایجاد شود (روی چه فیلدهایی / صعودی یا نزولی) و بعد از ایجاد شاخص میزان افزایش سرعت پاسخگویی به همان سوال را گزارش کنید.

سوالات:

گام سوم : سوال ۱ و ۲

گام چهارم: سوال ۱ و ۵ و ۶

گام پنجم - تفکیک توئیت‌ها / ریتوئیت‌ها

براساس فیلد type ، ریتوئیت ها را به یک کالکشن جدید منتقل کرده ، از کالکشن فعلی حذف کنید.

(اختیاری - نمره اضافی)

با استفاده از امکان **Map/Reduce**[1] میانگین تعداد هشتگ را برای هر کاربر بدست آورید . پر تکرارترین کلمه فارسی در توئیت را هم بیابید و اگر می‌توانید، ایست‌واژه‌ها را هم حذف کنید.

[1] https://api.mongodb.com/python/2.0/examples/map_reduce.html

بخش دوم - مدلسازی داده‌ها با گراف / کار با Neo4j

هدف از این سوال، آشنایی و کار با بانک اطلاعاتی گراف‌محور **Neo4j** می‌باشد.

برای این سوال، نیاز به نصب هیچ نرم‌افزاری نداریم. تنها ابزارهای ما، گوگل کولب (صرفا برای وارد کردن دیتا و نه کار با دیتابیس) و سندباکس سایت **Neo4j** می‌باشند. البته می‌توانید برای یادگیری عمیق‌تر، با مراجعه به آموزش‌های موجود در سایت **Neo4j**، تمامی موارد را بر روی سیستم خود نصب کنید که در اینجا به آن پرداخته نمی‌شود.

پس از ثبت‌نام در سایت، از قسمت **New Project**، می‌توانیم به دیتابیس‌های مختلف دسترسی داشته باشیم. اکیدا توصیه می‌شود قبل از اقدام به حل سوالات، از آموزش‌های موجود، مخصوصا دیتابیس Movies، بهره ببرید. برای این تمرین، Blank Sandbox را اجرا می‌کنیم.

گام اول - آشنایی با زبان سایفر

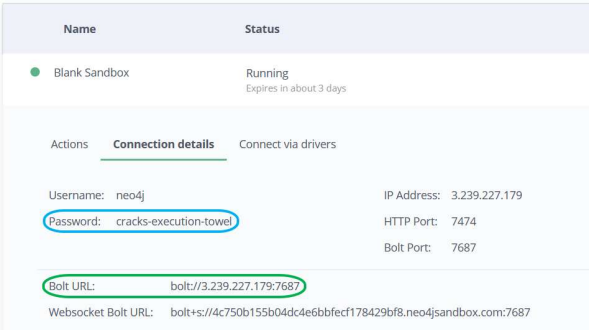
برای کار با نئوفورجی، نیاز به آشنایی با زبان سایفر خواهید داشت.برای این که به سرعت با این زبان ساده اما کارآمد برای کار با گراف‌ها آشنا شوید، آموزش موجود در این آدرس

https://guides.neo4j.com/sandbox/recommendations

را به کمک سندباکس نئوفورجی انجام دهید تا با اصول پایه‌ای زبان سایفر آشنا شوید. خروجی شما در این مرحله، نحوه بارگذاری داده‌ها و توضیح مختصر کوئری ها موجود در آدرس فوق به همراه تصویری از خروجی تولید شده خواهد بود.

گام دوم - بارگذاری داده‌ها و آماده سازی محیط کار با کولب

در این گام، قصد پردازش اطلاعات مجموعه تلویزیونی ارباب حلقه‌ها و روابط بین بازیکنان آنرا از طریق مدلسازی با گراف داریم. پس از ایجاد یک سندباکس جدید، با کلیک بر روی نام سندباکس ایجاد شده، مشاهده می‌کنیم که اطلاعات مورد نیاز برای دسترسی به دیتابیس (یوزر، پسورد و یک آدرس) در تب Connection details قرار گرفته است.



کافیست این اطلاعات را در فایل نوت‌بوکی که در اختیار شما قرار داده شده است، وارد و آن را اجرا کنید. به علت محدودیت‌های سرور، وارد کردن اطلاعات به دیتابیس کمی طول می‌کشد و بابت تاخیر در لود اولیه داده‌ها، نگران نباشید.

تقریبا در تمامی سوالات، از شما خواسته‌ایم خروجی را چاپ کنید. علاوه بر آن، لطفا از خروجی گراف ایجاد شده نیز اسکرین‌شات گرفته و در گزارش قرار دهید. همچنین در هر بخش، توضیح دهید چه فرآیندی در حال انجام می‌باشد(توضیح مختصر کوئری انجام شده). هر سوال، ممکن است یک یا چند خروجی داشته باشد و لزوما مورد خواسته شده، روی تمامی دیتاها کار نخواهد کرد (یعنی ممکن است خروجی تولید نشود) و نیازی به تغییر دیتا (بجز دو قسمت ذکرشده در سوال‌های 5 و 6) نمی‌باشد.

در این گام، تنها به آماده سازی نوت بوک لازم برای کار با داده‌ها می پردازیم .

گام سوم - کار عملی با داده‌ها

1- یک نمای کلی از انواع نودها و رابطه‌ها (به صورت گرافی) نمایش دهید (از دستور **Visualization** استفاده کنید.).

2- الف) چند شخصیت[1] و ب) چند شخصیت با نام یکتا وجود دارند؟

3- تعداد افراد هر سرزمین[2] را به همراه نام سرزمین به ترتیب حروف الفبا (سرزمین) نمایش دهید.

4- نام شخصیت‌های تک فرزند را بیابید. آیا می‌توانید برای این افراد تک فرزند، نام پدربزرگشان را هم نمایش دهید ؟

5- رابطه[3] دشمنی[4] بین اعضای خانواده را (درصورت امکان) حذف کنید.

6- الف) تعداد شمشیرزن[5]های هر سرزمین را بیابید و به صورت نزولی نمایش دهید (نمایش سرزمین‌هایی که شمشیرزن ندارند، لازم نیست).

ب) برای شمشیرزن‌های بدون سرزمین، در صورت امکان، از طریق سرزمین برادر/خواهر، سرزمین وی را تعیین کنید.

7- چند درصد شهروندان **Shire**، در رخداد[6] **the Council of Elrond** شرکت کردند؟

8- به ازای هر سرزمین، در صورت وجود، نام و تعداد افرادی را که حداقل در یک رخداد شرکت داشته اند را به همراه نام تمامی رخدادها (که حداقل یک نفر از آن سرزمین در آن شرکت کرده است)، بیابید (نام اشخاص و رخدادها برای هر کشور فقط یکبار ذکر شوند). برای سرزمینهایی که افراد آن در هیچ رخدادی شرکت نداشته اند، میتوان مقدار 0 و لیست خالی بازگرداند.

---

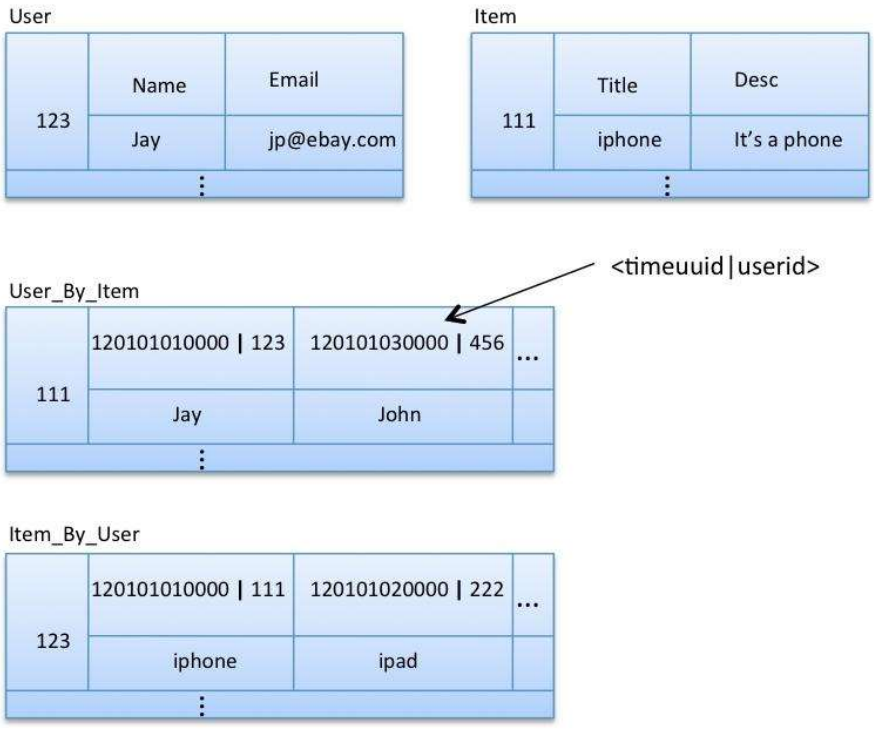
[1] Character
[2] Country
[3] Relationship
[4] ENEMY
[5] swordfighter
[6] Event

بخش سوم - دیتابیس‌های سطرگسترده / کار با کاساندرا

دیتابیس‌های سطرگسترده مانند کاساندرا یا **HBase** برای ذخیره داده‌هایی ساخته شده‌اند که ساختار *کلید/مقدار* دارند به نحوی که با داشتن *کلید*، بتوان تمام اطلاعات موجود در بخش *مقدار* را با سرعت بسیار بالا خواند و پردازش کرد.

فرض کنید می‌خواهیم اطلاعات امتیازهای داده شده به یک فیلم را ذخیره کنیم. در اینجا، *کلید* همان نام یا شناسه فیلم (**RowKey**) و *مقدار*، امتیازهای مختلفی است که کاربران به آن فیلم داده‌اند . یعنی خود بخش «*مقدار*»، حاوی ستون‌ها یا بخش‌هایی (**Column Family**) است که تعداد نامشخصی دارند. یک فیلم ممکن است تنها یک نفر امتیازدهنده داشته باشد و فیلمی دیگر، هزاران امتیاز داشته باشد(جدول **ratings** که در آن،کلید هر سطر آن، نام فیلم و ستون‌های آن، هزاران امتیاز–شامل شناسه امتیاز دهنده، زمان، امتیاز– خواهد بود). به همین دلیل به این دیتابیس‌ها، **Wide Row**یا **Column Family** می‌گوییم.

از طرفی اگر بخواهیم بدانیم یک کاربر چه فیلم‌هایی را لایک کرده است و یا به چه فیلم‌هایی امتیاز داده است، یک جدول **UserLikes** و **UserRatings** در نظر میگیریم که در هر دوی آنها، کلید جدول، کدکاربر و ستون‌های آن ، نام فیلم‌ها خواهند بود. **بنابراین در این نوع از دیتابیس‌ها، جدول طراحی نمی کنیم بلکه دنبال یافتن کلید/مقدارهایی هستیم که هر کدام بتواند به یک کوئری مورد نیاز ما پاسخ دهد.** (هر چند ممکن است برخی اطلاعات مانند اطلاعات خود یوزر و فیلم ظاهراً به صورت جدول عادی ذخیره شوند - این جداول هم پشت صحنه با قالب کلید مقدار ذخیره می‌شوند– اما سایر جداول، ساختاری کاملا مطابق به قالب کلید/مقدار خواهند داشت– شکل زیر)



در طراحی دیتابیس‌های سطر گسترده ، باید دید رابطه‌ای را کنار بگذارید و بسته به نیاز اطلاعاتی و جستجوهایی که انجام خواهید داد، به طراحی جداول بپردازید. نگران افزونگی و تکرار داده‌ها نباشید چون برای بالابردن سرعت جستجو در بین میلیون‌ها رکوردی که در بین ده‌ها نود شبکه پخش شده‌اند، مجبوریم فضای دیسک را بیشتر از حالت نرمال، مصرف کنیم.

توصیه می‌کنیم قبل از شروع کار با بخش از تمرین ، دو مقاله **ebay** با عنوان **Cassandra Data Modeling Best Practices** را که به صورت عملی و با ذکر یک مثال، به بررسی نحوه مدلسازی داده‌ها در کاساندرا پرداخته است را حتما مطالعه کنید. در ادامه، توضیحاتی مختصر راجع به این دیتابیس و مفاهیم پایه آن ذکر می کنیم و سپس به بیان خود تمرین این بخش خواهیم پرداخت.

مقدمه‌ای بر کاساندرا

پایگاه داده سطر گسترده کاساندرا یکی از محبوب‌ترین دیتابیس‌های **NoSQL** است. در کاساندرا جداول در **keyspace**ها(معادل دیتابیس در بانک‌های اطلاعاتی رابطه‌ای) قرار می‌گیرند و هر نود می‌تواند شامل یک یا چند **keyspace** باشد و هر **keyspace** دارای استراتژی تکرار(**Replication**) و توزیع(**Partitioning**) مخصوص به خودش است. سپس با تعریف جداول با ستون‌های مشخص می‌توان اطلاعات را در سطرها ذخیره کرد.

در کاساندرا نظیر دیتابیس‌های دیگر هر سطر دارای یک کلید است اما مفهوم و کارکرد کلید در کاساندرا کمی با سایر دیتابیس‌ها متفاوت است. در کاساندرا نحوه توزیع داده‌ها بین نودها بر اساس **Partition Key** و نحوه مرتب‌سازی داده‌ها در هر پارتیشن، براساس **Clustering Key** انجام می‌شود. دقت کنید که داده‌ها در کاساندرا هنگام ذخیره سورت می‌شوند و هنگام بازیابی، نمی‌توانید دستور سورت داده‌ها را براساس فیلدی غیر از آنچه در کلاسترکی مشخص شده است بدهید. کلیدها در کاساندرا می‌تواند ساده یا ترکیبی باشند و با توجه به شرایط هر جدول می‌تواند **Partition Key** و **Clustering Key** چندمقداری داشته باشد.(برای آشنایی با این مفاهیم می‌توانید به این مقاله فارسی مراجعه کنید[1])

نکته‌ای که در کار کردن با دیتابیس‌های سطر گسترده شبیه به کاساندرا باید به آن توجه کنیم این است که تکرار داده در جداول مختلف امری طبیعی است و معمولاً نمی‌توان با طراحی یک جدول به تمام سؤالات پاسخ داد و بر اساس نیازمندی‌های سؤالات مختلف باید جدول مربوط به آنرا طراحی کنیم.

نصب و راه‌اندازی

برای کار با دیتابیس کاساندرا به **JDK-8** نیاز داریم. سپس کاساندرا را با استفاده از این راهنماها نصب می‌کنیم:

windows : https://phoenixnap.com/kb/install-cassandra-on-windows

ubuntu : https://phoenixnap.com/kb/install-cassandra-on-ubuntu

[1] http://yun.ir/hdww7c

و سپس در صورت نیاز درایور کاساندرا برای پایتون را نصب می‌کنیم.

گام اول

ابتدا دیتاست مورد نظر را از این لینک[1] دانلود کنید. این دیتاست بخشی از دیتاست بزرگ FMA است که برای تحلیل و بررسی متادیتا موزیک به همراه صوت آن می‌باشد. برای سادگی کار ۱۲ ستون از این دیتاست را انتخاب کرده‌ایم و داده فعلی شامل ۱۰۶۵۷۴ سطر است. ستون‌ها اطلاعات مربوط به خواننده، آلبوم، تاریخ انتشار، تعداد شنیده شدن آهنگ و آلبوم و سایر مشخصات موسیقیایی همچون ژانر موزیک را نیز دارند. توجه کنید که برای برخی از سطرها اطلاعات ممکن است ناقص باشند و مثلاً تاریخ انتشار یا ژانر موزیک خالی باشد.

بعد از دانلود کردن دیتاست و آشنایی اولیه با ستون‌های آن، یک **keyspace** ایجاد کنید و سعی کنید تا با استفاده از درایور پایتون کاساندرا یا با استفاده از شل کاساندرا **cqlsh** با نحوه ایجاد جدول و وارد کردن داده‌ها در سطرها آشنا شوید و با تعریف **primary key** های مختلف تفاوت و اهمیت نحوه تعریف کلیدها را بیشتر لمس کنید.

گام دوم

با توجه به سؤالاتی که در ادامه می‌آید جداولی را طراحی کنید و سپس با نوشتن **query** های مناسب پاسخ سؤال را بیابید. برای هر بخش نحوه ایجاد جدول، زمان لازم برای وارد کردن داده، نحوه نوشتن **query** و زمان اجرای آنرا در گزارش ذکر کنید.

1. لیست آهنگ‌های آلبوم **Rumble, Young Man, Rumble** .
2. آهنگ‌های آرتیستی با نام **RoccoW** که ژانری برای آن‌ها ثبت نشده است.
3. آهنگ‌های **Hip-Hop** که طولی کمتر از ۳ دقیقه دارند و در سال‌های ۲۰۱۵ و ۲۰۱۶ منتشر شده‌اند.
4. آهنگ‌های **Electronic** و **Pop** که در ماه آپریل منتشر شده‌اند و بیش از 300 بار شنیده شده‌اند.

گام سوم

5. حال می‌خواهیم با استفاده از دستورات **aggregation** و **group by** و **order by** می‌خواهیم اطلاعات مختلفی را از دیتاست مورد نظر استخراج کنیم. (کاساندرا از نسخه ۳.۱۰ گروه‌بندی را به امکانات خود اضافه کرده است و پشتیبانی از مرتب سازی هم بر روی فیلدهای غیر از کلاستر کی،ممکن است با شکست مواجه شود)
6. تعداد آهنگ‌های ژانر **Folk** در شش ماه اول سال ۲۰۰۸.

[1] https://drive.google.com/file/d/12lyO-5YGDD75cxwL0SU_iFcXswrbeLXN/view?usp=sharing

7. متوسط طول آهنگ‌ها و ۱۰ آهنگی که بیشترین طول را دارند.

8. برترین آلبوم‌های **Rock** در سال ۲۰۱۶ بر اساس قرار گرفتن در لیست مورد آلبوم‌های مورد علاقه یا همان **favorites_album**. تعداد **favorites_artist** آرتیست‌های هر کدام از این آلبوم‌ها چقدر است؟

9. مجموعا آهنگ‌های هر کدام از ژانرها چند بار شنیده شده‌اند. خروجی بر اساس این عدد مرتب شود.

10. محبوب‌ترین آرتیست هر کدام از سال‌های ۲۰۱۵، ۲۰۱۶ و ۲۰۱۷ بر اساس **listens_album**.

بخش چهارم - کار با الستیک سرچ

یکی از دیتابیس های معروف در حوزه جستجوی متن با سرعت بالا در ذخیره انواع داده‌های متنی و پاسخگویی به انواع کوئری های کاربر بر روی آنها، الستیک سرچ است که در اکوسیستم استارتاپی ایران هم بسیار پرطرفدار است.

در این بخش هم هدف اصلی، آشنایی اولیه با این دیتابیس و نحوه کار با آن است که حداکثر با دوساعت صرف زمان، میتوانید به راحتی آنرا انجام دهید.

کافی است به آدرس زیر مراجعه کرده و تمامی مراحل آنرا انجام دهید :

yun.ir/fi9loe

تمام دستورات آنرا از ابتدا تا انتها در محیط کیبانا که محیط گرافیکی کار با الستیک‌سرچ است انجام داده، با گرفتن اسکرین شات از خروجی آنها، گزارش خود را آماده کنید. داده‌ها را طوری وارد کنید که هر کوئری حداقل دوجواب در خروجی برگرداند.

در انتهای کار، با صدا زدن **API** زیر در یک برنامه پایتون حداقل هزار توئیت را در الستیک سرچ ذخیره نمایید. دقت کنید که هشتگ‌ها که همان نمادهای بورسی هستند را به ازای هر توئیت، مشابه با کاری که در بخش مانگو انجام دادید استخراج کرده و در فیلدی جداگانه به صورت آرایه (لیست) به توئیت اضافه کرده و نهایتا این توئیت پردازش شده رادر الستیک سرچ ذخیره کنید که در ادامه بتوانید از این هشتگ‌ها برای رسم نمودارهای گام آخر، استفاده نمایید.

سپس سه کوئری به دلخواه بر روی این مجموعه اجرا کرده و خروجی آنها را مستند کنید.

درگام آخر نیز، یک داشبورد با سه ویژوالیزیشن **ابرهشتگ‌ها / تعداد نمادهای پرتکرار / تعداد کاربران با بیشترین توئیت** ایجاد نمایید .

https://www.sahamyab.com/guest/twiter/list?v=0.1

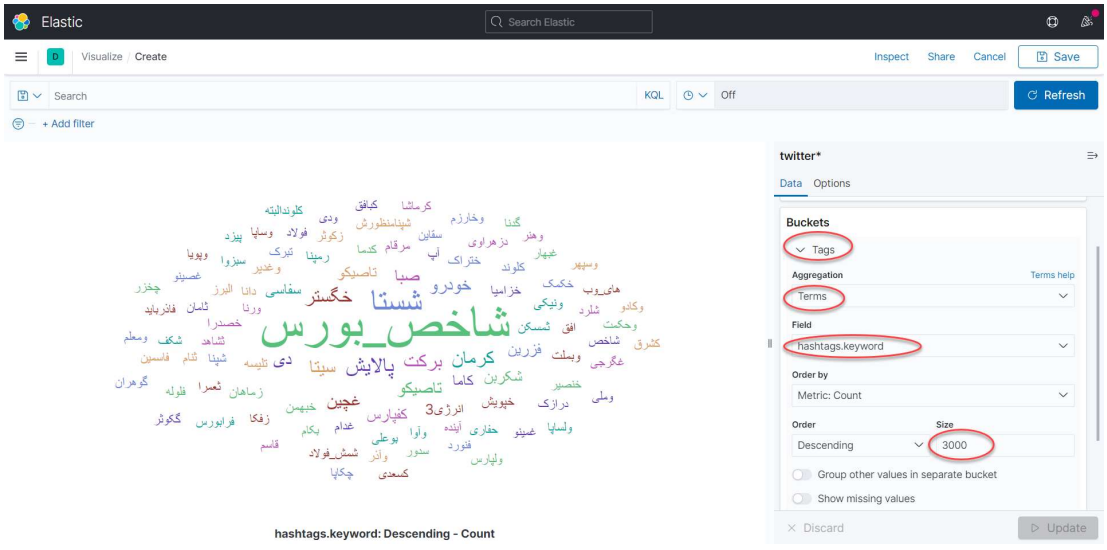
کد لازم برای این منظور در شکل صفحه بعد آمده است (قسمت ارسال درخواست را مشابه با کدهای بخش مانگو بنویسید - Header را وارد کنید تا خطای ۴۰۳ تولید نشود):

```python
import requests
from elasticsearch import Elasticsearch
import re

url = 'https://www.sahamyab.com/guest/twiter/list?v=0.1'
elasticSearch = Elasticsearch([{'host': 'localhost', 'port': 9200}])
total = 1000
fetched = 0
seenIds = set()
hashtags = list()

while fetched < total:
    response = requests.get(url=url)
    if response.status_code != 200:
        print('HTTP', response.status_code)
        continue
    data = response.json()["items"]  # Check the JSON Response Content documentation below
    for tweet in data:
        if tweet["id"] not in seenIds:
            try:
                tweet["hashtags"] = re.findall(r"#(\w+)", tweet["content"])
                elasticSearch.index(index="twitter", doc_type="twitter", body=tweet)
                seenIds.add(tweet["id"])
                fetched += 1
                print("tweet " + str(tweet["id"]) + "fetched,   total: " + str(fetched))
            except Exception as e:
                print(e)
```

ابرکلمات مشابه با زیر تولید خواهد شد که بعد از ذخیره به داشبورد اصلی منتقل خواهد شد :



نکات پیاده سازی

- در این تمرین فقط مجاز به استفاده از زبان برنامه نویسی Python خواهید بود.
- استفاده یا عدم استفاده از Docker اختیاری است

نکات تحویل

- مهلت ارسال این تمرین تا ۳۱ اردیبهشت ۱۴۰۰ خواهد بود.
- انجام این تمرین به صورت یک نفره می باشد.
- می توانید تمرین را حداکثر با یک هفته تاخیر ارسال نمائید ، نحوه محاسبه تاخیر نیز به این شکل خواهد بود که به ازای هر روز تاخیر ۵ درصد از نمره تمرین کسر خواهد شد.
- بعد از پایان مهلت ارسال تمرین، تمرین تحویل آنلاین نیز خواهد داشت ، که زمان آن متعاقبا از طریق سامانه مدیریت دروس اعلام خواهد شد.
- لطفا در روز تحویل آنلاین کدهای خود را آماده اجرا داشته باشید ،دقت نمائید که حق تغییر کدهای ارسالی را نخواهید داشت و همچنین افرادی که تمرین خود را تا قبل از تاریخ اعلام شده در سامانه آپلود نکرده باشند، مجاز به تحویل آنلاین تمرین نخواهند بود.
- می توانید برای پاسخ تمرین ها در اینترنت جستجو کنید اما وجود تشابه غیرمنطقی بین گزارش ها و کدهای ارسالی تقلب محسوب شده و نمره تمرین تمامی افراد شرکت کننده در آن صفر در نظر گرفته خواهد شد.
- گزارش شما در فرآیند تصحیح از اهمیت ویژه ای برخوردار است، لطفا تمامی مواردی که در شرح تمرین از شما خواسته شده را در گزارش ذکر نمائید.
- لطفا گزارش ، فایل کدها و سایر ضمائم مورد نیاز را با فرمت زیر در سامانه مدیریت دروس بارگذاری نمائید.

HW1_[Lastname]_[StudentNumber].zip

- در صورت نیاز به برقراری ارتباط با طراحان سوال هر قسمت، از ایمیل‌های زیر می‌توانید استفاده کنید :
  - **کاساندرا** : همایون مرادی ، homoradi@ut.ac.ir
  - **مانگودی‌بی** : علیرضا نیلگران : nilgaran@ut.ac.ir
  - **نئوفورجی** : s.taghizadeh@ut.ac.ir
  - **الستیک‌سرچ** : smbanaei@ut.ac.ir