

INSTITUT SUPÉRIEUR D'ÉLECTRONIQUE DE PARIS (ISEP)



II.2414 ADVANCED DATABASES AND BIG DATA

BIG DATA PROJECT (Data Lake and Data Pipeline)

BUHARI ALIYU

PROF. THIBAUT DE BROCA

JUNE 2023

Synopsis

The main goal of the 'Big Data project' is to create a simple end-to-end data architecture that spans the entire data lifecycle, encompassing data ingestion,

transformation, and exposition. Within this project, you have the freedom to choose the specific data you wish to retrieve and determine the desired output that will generate value from the input data.

However, it is essential to adhere to a Datalake architecture, as it plays a crucial role in organizing the data effectively, ensuring a streamlined data pipeline, and facilitating data sharing among stakeholders. By following the Datalake architecture principles, we can achieve a well-structured and efficient data ecosystem

Theme for the Data

The central focus of this project revolves around gathering movies, weather data, and forecast information from multiple sources. The primary goal is to conduct a comprehensive analysis of the current top-rated, upcoming, and trending movies, considering their popularity, vote counts, release dates, and weather conditions. By incorporating weather forecast data, this analysis aims to provide valuable insights for individuals planning to visit a cinema.

By visualizing the cinema data alongside weather forecasts, stakeholders such as consumers, researchers, statisticians, and data scientists can make more informed decisions about their moviegoing experiences. The integration of weather information allows users to consider factors like temperature, precipitation, and other weather-related conditions that can impact their cinema experience.

For example, if someone prefers to enjoy outdoor activities before or after watching a movie, they can check the weather forecast to determine the best time to visit the cinema. If it's expected to rain heavily, they might opt for an indoor movie theater instead of an outdoor cinema. On the other hand, if the weather forecast indicates clear skies and pleasant temperatures, they may choose an outdoor cinema for a unique and enjoyable experience.

Moreover, the combination of cinema data and weather forecasts enables users to apply filters and preferences based on their specific moviegoing desires. They can filter movies based on genre, release date, popularity, and user ratings, and further refine their choices by considering weather conditions. For instance, individuals who prefer to watch horror movies during rainy days can easily identify suitable options by cross-referencing the cinema data with the weather forecast.

Furthermore, this enhanced visualization empowers individuals to plan their cinema outings more effectively by taking into account both movie-related factors and weather conditions. It provides a comprehensive understanding of the movie landscape and allows users to make informed decisions, ensuring an enjoyable and tailored cinema experience

Data Sources

The retrieved data would be gathered in the data lake as raw data. Data sources include:

1. The Movie Database TMDB API
2. Weather Api

A community-built movie and television database is called The Movie Database (TMDB) API. The documentation for the API is available at <https://developer.themoviedb.org/docs/getting-started>, and it includes instructions on how to connect to our application and fetch data via the API. We have fetched 3 different dataset from the api, the top rated movies, the upcoming movies and trending movies in the cinema. We saved the data as a json file. While the weather api documentation can be access via <https://www.weatherapi.com/docs>, we fetched the current weather and forecast weather of Paris

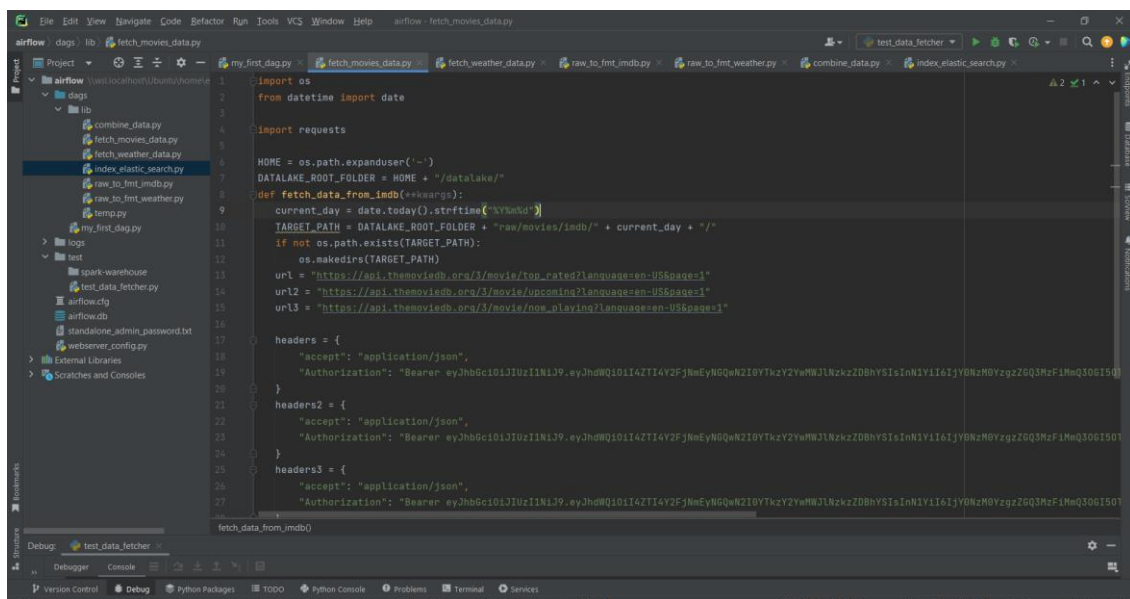
The image is a screenshot of a code editor, likely VS Code, showing a Python script named 'fetch_movies_data.py'. The script is part of an Airflow DAG. The code imports 'os', 'datetime', and 'requests'. It defines a 'HOME' variable and a 'DATA_LAKE_ROOT_FOLDER' path. A function 'fetch_data_from_tmdb' is defined, which takes 'current_day' as an argument. It constructs a 'TARGET_PATH' and checks if it exists. If not, it creates the directory. It then defines three URLs for TMDB API endpoints: 'https://api.themoviedb.org/3/movie/top-rated?language=en-US&page=1', 'https://api.themoviedb.org/3/movie/upcoming?language=en-US&page=1', and 'https://api.themoviedb.org/3/movie/now-playing?language=en-US&page=1'. It sets headers for 'accept' and 'authorization'. The script is part of an Airflow DAG, as indicated by the 'airflow' prefix in the file name and the 'fetch_data_from_tmdb' function call at the bottom. The left sidebar shows the project structure with files like 'combine_data.py', 'fetch_movies_data.py', 'fetch_weather_data.py', 'raw_to_fmri_weather.py', 'raw_to_fmri_weather.py', 'temp.py', 'my_first_dag.py', 'logs', 'test', 'spark-warehouse', 'test_data_fetcher.py', 'airflow.cfg', 'airflow.db', 'standalone_admin_password.txt', 'webserver_config.py', and 'External Libraries'. The bottom status bar shows 'Debug', 'Python Packages', 'TODO', 'Python Console', 'Problems', 'Terminal', and 'Services'.

Figure 1 shows how to access the movie database API.

The Weather API serves as a valuable source of raw data for this project, providing real-time weather forecasts (for the next 7 days) and current weather information for Paris. By integrating this data into the project's data lake, stakeholders can explore the relationship between weather conditions and cinema trends, gaining insights into moviegoer behavior and enabling informed decision-making in Paris.

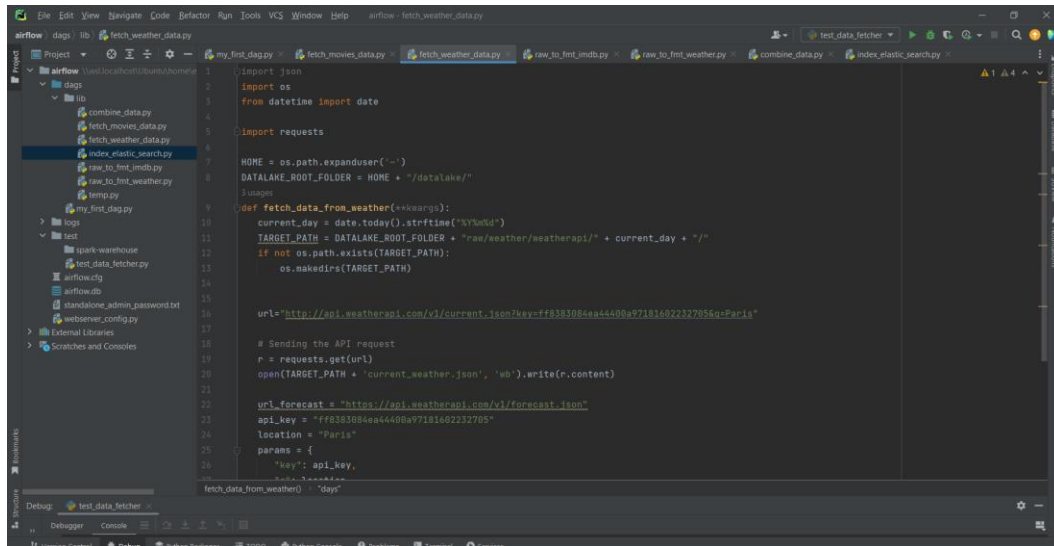


Figure 2: Connection to weather API

Data Lake Organization

We designed our data lake with a structure that adheres to the following naming convention: Each route will look like this:

`/[{layer}]/[{group}]/[{TableName}]/[{date}]/filename`

Layers can include: - **raw** which is the first phase of the data and is straight from the source system, the data is prepared for use by the Datalake in the second stage, which is *formatted*, and *usage* is the final stage before

Group: we use *movies* to store movies file and *weather* to save weather files

Table Name: An item in this folder will always have this name and a similar schema, in raw we use *imdb* for the movies and *weatherapi* for weather.

FileName: top-rated_movies.json, upcoming_movies.json, trending_movies.json, current_weather.json and forecast.json are the file names for the raw data.

Date: should be formatted as YYYYMMDD, for instance, 20230611 (11th June 2022).

For formatted file: top-rated_movies.snappy.parquet, trending_movies.snappy.parquet, upcoming_movies.snappy.parquet, current_weather.snappy.parquet and forecast.snappy.parquet.

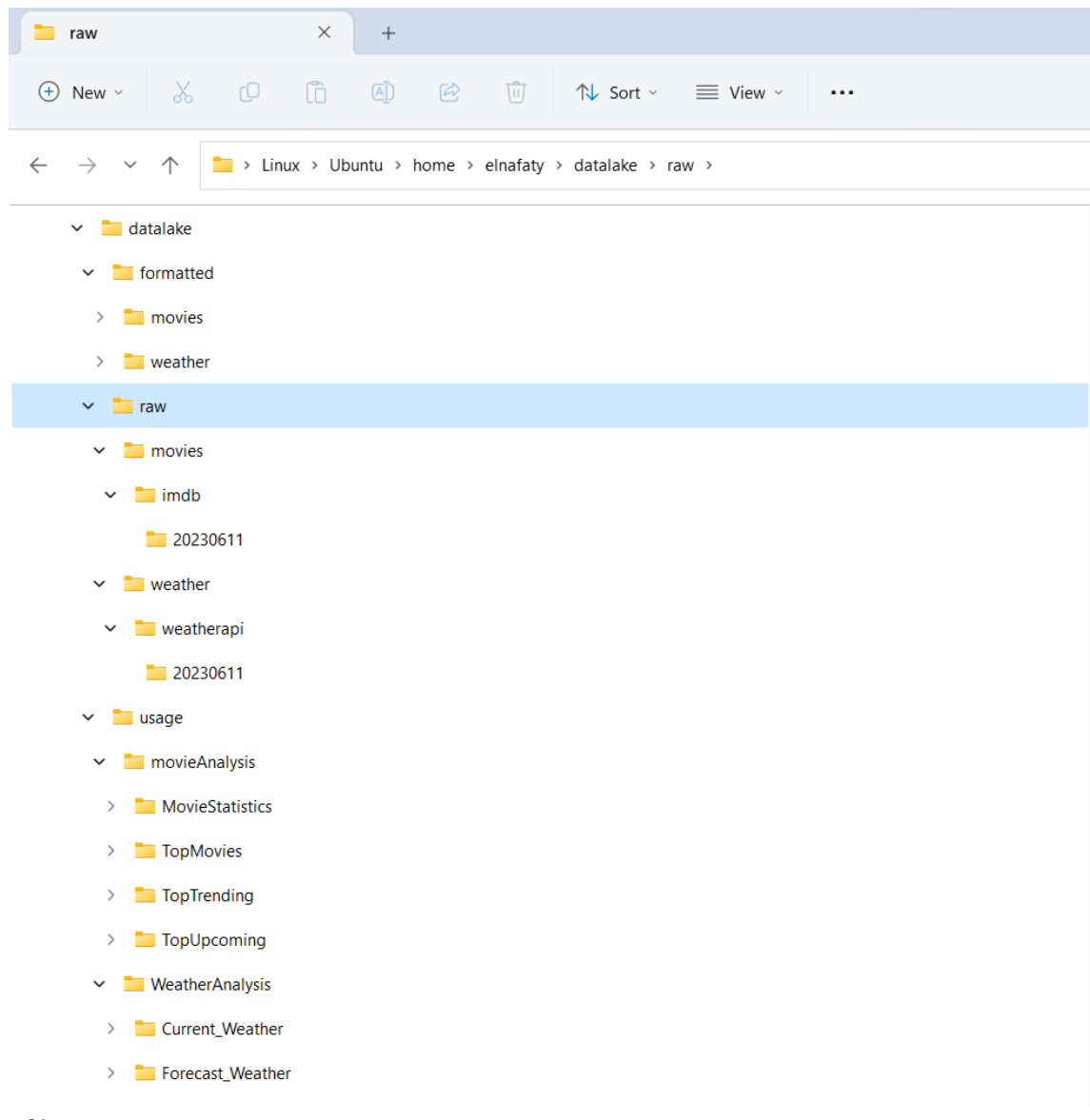


Figure 3: Datalake folder structure

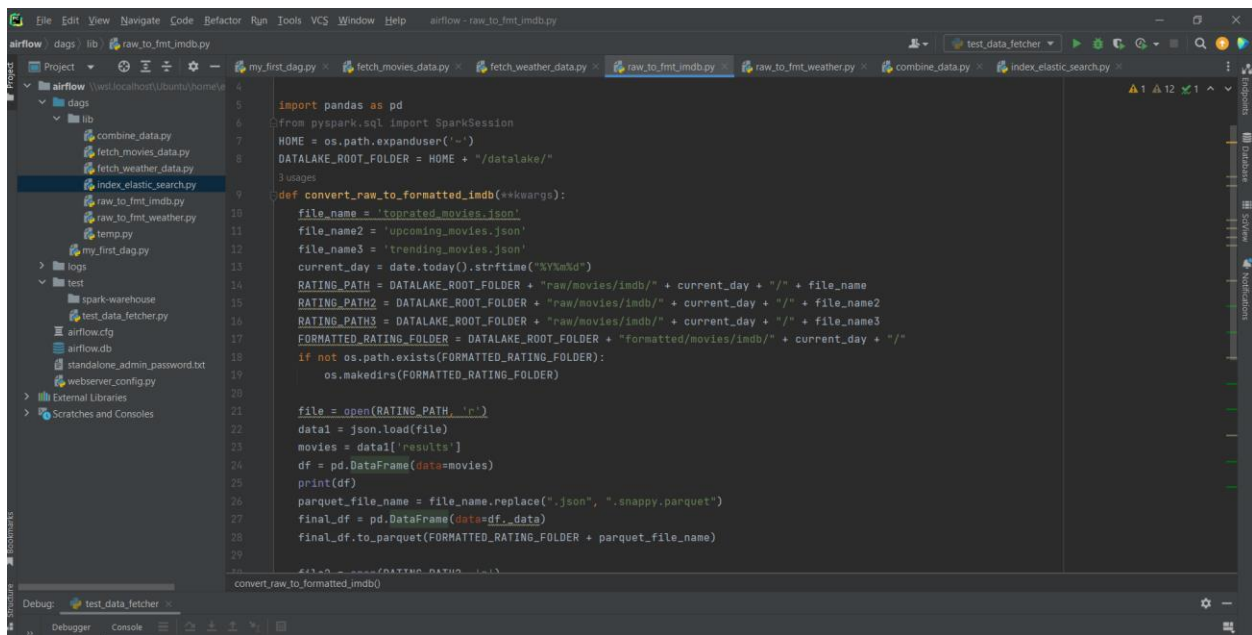
Data Pipeline

The data pipeline plays a crucial role in transferring data from its source to a designated destination, such as a data warehouse. Throughout this journey, the data undergoes transformations and optimizations, ultimately arriving in a state that allows for analysis and the development of valuable business insights. The data pipeline encompasses the essential steps of aggregating, organizing, and moving data effectively.

Data Pipeline Architecture

1. **Data Ingestion:** The source data is obtained through APIs. The data sources and ingestion process are detailed in Figure 1 and Figure 2 in the data source section. The collected data is then transferred to the raw folder in the data lake, where it can be further prepared for analysis. Python programming language is used for data ingestion in this project.
2. **Data Formatting:** In this step, the data is aggregated, cleansed, and manipulated to bring it to a standardized format suitable for analysis. Tools like Pandas are utilized to format the data like location, latitude and longitude for the weather. Figure 5 and Figure 6 illustrate the formatting process applied to data from different sources. The formatted data is then converted to the Parquet data format and compressed using snappy, which offers better processing and analysis capabilities.

By following these steps, the data is transformed and prepared in a way that optimizes its usability for subsequent analysis and visualization.



```
4 import pandas as pd
5 from pyspark.sql import SparkSession
6 HOME = os.path.expanduser('~')
7 DATALAKE_ROOT_FOLDER = HOME + "/datalake/"
8
9 def convert_raw_to_formatted_imdb(**kwargs):
10     file_name = 'toprated_movies.json'
11     file_name2 = 'upcoming_movies.json'
12     file_name3 = 'trending_movies.json'
13     current_day = date.today().strftime("%Y%m%d")
14     RATING_PATH = DATALAKE_ROOT_FOLDER + "raw/movies/imdb/" + current_day + "/" + file_name
15     RATING_PATH2 = DATALAKE_ROOT_FOLDER + "raw/movies/imdb/" + current_day + "/" + file_name2
16     RATING_PATH3 = DATALAKE_ROOT_FOLDER + "raw/movies/imdb/" + current_day + "/" + file_name3
17     FORMATTED_RATING_FOLDER = DATALAKE_ROOT_FOLDER + "formatted/movies/imdb/" + current_day + "/"
18     if not os.path.exists(FORMATTED_RATING_FOLDER):
19         os.makedirs(FORMATTED_RATING_FOLDER)
20
21     file = open(RATING_PATH, 'r')
22     data1 = json.load(file)
23     movies = data1['results']
24     df = pd.DataFrame(data=movies)
25     print(df)
26     parquet_file_name = file_name.replace(".json", ".snappy.parquet")
27     final_df = pd.DataFrame(data=df, data)
28     final_df.to_parquet(FORMATTED_RATING_FOLDER + parquet_file_name)
29
30 convert_raw_to_formatted_imdb()
```

Figure 5: Formatting data from imdb with Panda

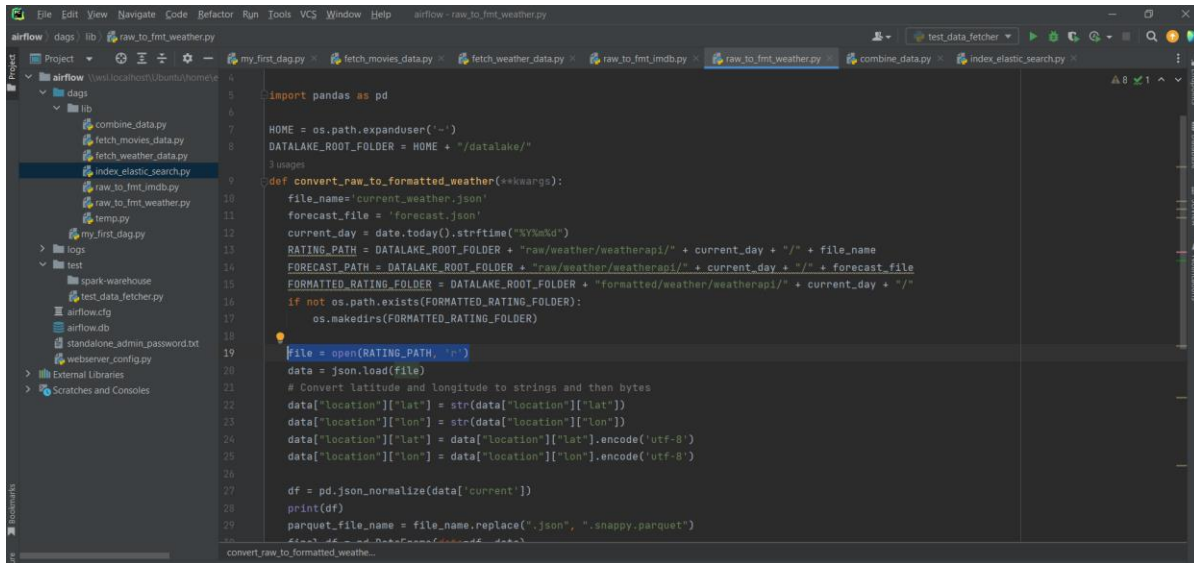


Figure 6: Formatting weather data with Panda

3. **Data Combination:** Apache Spark is utilized in this step to combine the data from different data sources and performed sql analysis. This combined data becomes the foundation for further analysis. It is indexed and made accessible through the Elasticsearch and Kibana to visualize the data, we use the cloud version of the elasticsearch and kibana

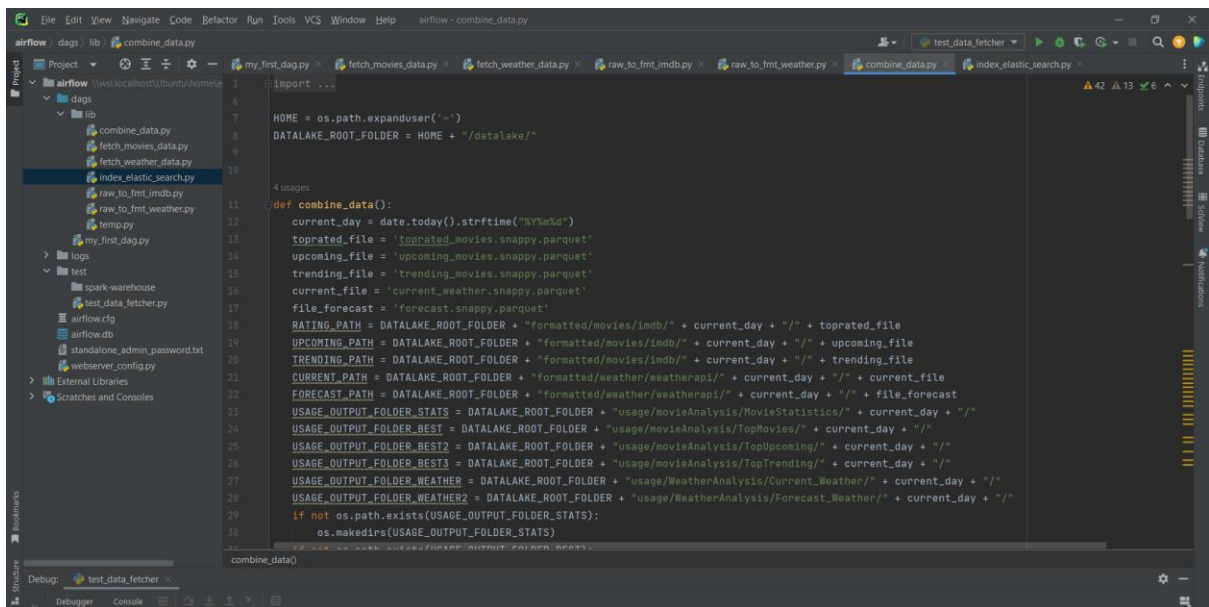
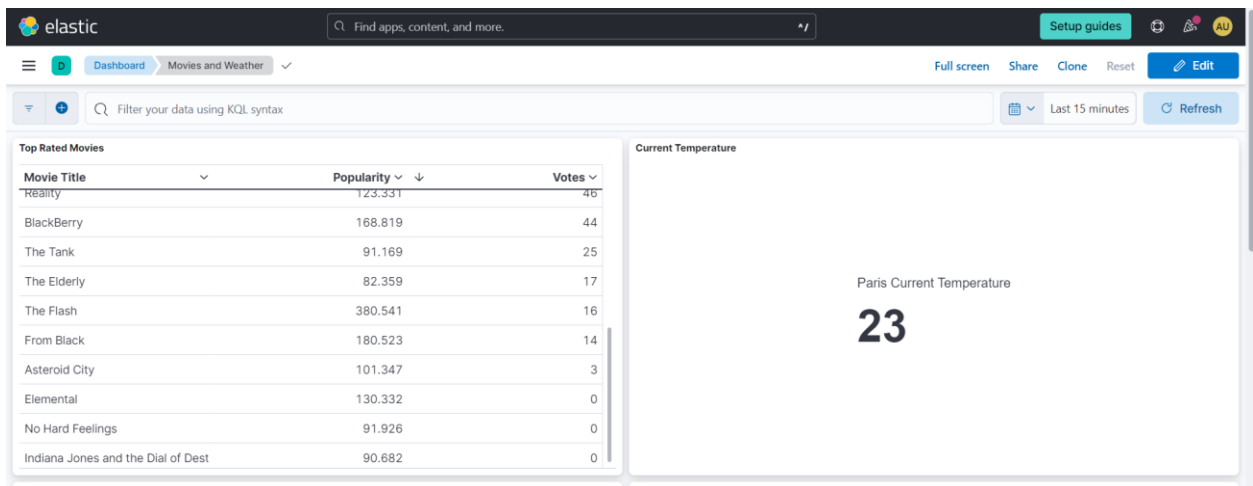


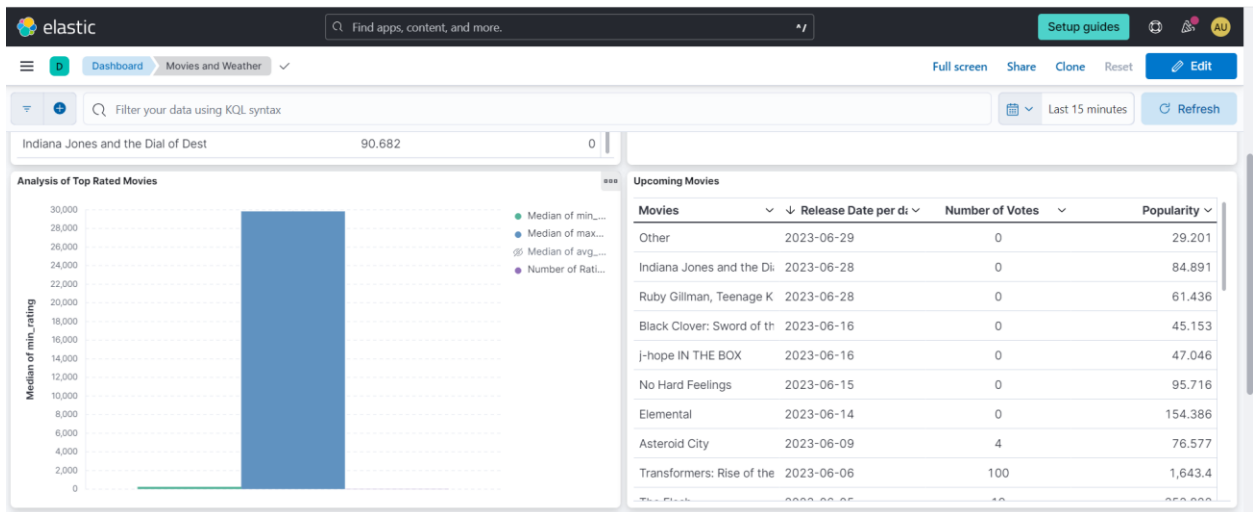
Figure 7: Data combination with Apache Spark


```

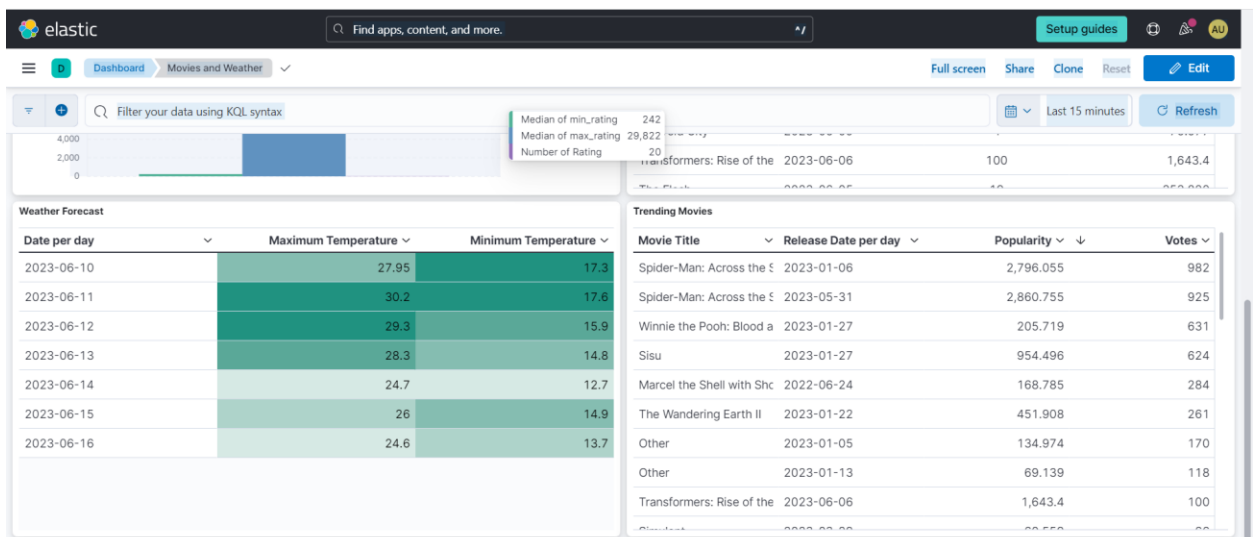
13 1
14 2
15 3
16 4
17 5
18 6
19 7
20 8
21 9
22 10
23 11
24 12
25 13
26 14
27 15
28 16
29 17
30 18
31 19
32 20
33 21
34 22
35 23
36 24
37 25
38 26
39 27
40 28
41 29
42 30
43 31
44 32
45 33
46 34
47 35
48 36
49 37
50 38
51 39
52 40
53 41
54 42
55 43
56 44
57 45
58 46
59 47
60 48
61 49
62 50
63 51
64 52
65 53
66 54
67 55
68 56
69 57
70 58
71 59
72 60
73 61
74 62
75 63
76 64
77 65
78 66
79 67
80 68
81 69
82 70
83 71
84 72
85 73
86 74
87 75
88 76
89 77
90 78
91 79
92 80
93 81
94 82
95 83
96 84
97 85
98 86
99 87
100 88
101 89
102 90
103 91
104 92
105 93
106 94
107 95
108 96
109 97
110 98
111 99
112 100
113 101
114 102
115 103
116 104
117 105
118 106
119 107
120 108
121 109
122 110
123 111
124 112
125 113
126 114
127 115
128 116
129 117
130 118
131 119
132 120
133 121
134 122
135 123
136 124
137 125
138 126
139 127
140 128
141 129
142 130
143 131
144 132
145 133
146 134
147 135
148 136
149 137
150 138
151 139
152 140
153 141
154 142
155 143
156 144
157 145
158 146
159 147
160 148
161 149
162 150
163 151
164 152
165 153
166 154
167 155
168 156
169 157
170 158
171 159
172 160
173 161
174 162
175 163
176 164
177 165
178 166
179 167
180 168
181 169
182 170
183 171
184 172
185 173
186 174
187 175
188 176
189 177
190 178
191 179
192 180
193 181
194 182
195 183
196 184
197 185
198 186
199 187
200 188
201 189
202 190
203 191
204 192
205 193
206 194
207 195
208 196
209 197
210 198
211 199
212 200
213 201
214 202
215 203
216 204
217 205
218 206
219 207
220 208
221 209
222 210
223 211
224 212
225 213
226 214
227 215
228 216
229 217
230 218
231 219
232 220
233 221
234 222
235 223
236 224
237 225
238 226
239 227
240 228
241 229
242 230
243 231
244 232
245 233
246 234
247 235
248 236
249 237
250 238
251 239
252 240
253 241
254 242
255 243
256 244
257 245
258 246
259 247
260 248
261 249
262 250
263 251
264 252
265 253
266 254
267 255
268 256
269 257
270 258
271 259
272 260
273 261
274 262
275 263
276 264
277 265
278 266
279 267
280 268
281 269
282 270
283 271
284 272
285 273
286 274
287 275
288 276
289 277
290 278
291 279
292 280
293 281
294 282
295 283
296 284
297 285
298 286
299 287
300 288
301 289
302 290
303 291
304 292
305 293
306 294
307 295
308 296
309 297
310 298
311 299
312 300
313 301
314 302
315 303
316 304
317 305
318 306
319 307
320 308
321 309
322 310
323 311
324 312
325 313
326 314
327 315
328 316
329 317
330 318
331 319
332 320
333 321
334 322
335 323
336 324
337 325
338 326
339 327
340 328
341 329
342 330
343 331
344 332
345 333
346 334
347 335
348 336
349 337
350 338
351 339
352 340
353 341
354 342
355 343
356 344
357 345
358 346
359 347
360 348
361 349
362 350
363 351
364 352
365 353
366 354
367 355
368 356
369 357
370 358
371 359
372 360
373 361
374 362
375 363
376 364
377 365
378 366
379 367
380 368
381 369
382 370
383 371
384 372
385 373
386 374
387 375
388 376
389 377
390 378
391 379
392 380
393 381
394 382
395 383
396 384
397 385
398 386
399 387
400 388
401 389
402 390
403 391
404 392
405 393
406 394
407 395
408 396
409 397
410 398
411 399
412 400
413 401
414 402
415 403
416 404
417 405
418 406
419 407
420 408
421 409
422 410
423 411
424 412
425 413
426 414
427 415
428 416
429 417
430 418
431 419
432 420
433 421
434 422
435 423
436 424
437 425
438 426
439 427
440 428
441 429
442 430
443 431
444 432
445 433
446 434
447 435
448 436
449 437
450 438
451 439
452 440
453 441
454 442
455 443
456 444
457 445
458 446
459 447
460 448
461 449
462 450
463 451
464 452
465 453
466 454
467 455
468 456
469 457
470 458
471 459
472 460
473 461
474 462
475 463
476 464
477 465
478 466
479 467
480 468
481 469
482 470
483 471
484 472
485 473
486 474
487 475
488 476
489 477
490 478
491 479
492 480
493 481
494 482
495 483
496 484
497 485
498 486
499 487
500 488
501 489
502 490
503 491
504 492
505 493
506 494
507 495
508 496
509 497
510 498
511 499
512 500
513 501
514 502
515 503
516 504
517 505
518 506
519 507
520 508
521 509
522 510
523 511
524 512
525 513
526 514
527 515
528 516
529 517
530 518
531 519
532 520
533 521
534 522
535 523
536 524
537 525
538 526
539 527
540 528
541 529
542 530
543 531
544 532
545 533
546 534
547 535
548 536
549 537
550 538
551 539
552 540
553 541
554 542
555 543
556 544
557 545
558 546
559 547
560 548
561 549
562 550
563 551
564 552
565 553
566 554
567 555
568 556
569 557
570 558
571 559
572 560
573 561
574 562
575 563
576 564
577 565
578 566
579 567
580 568
581 569
582 570
583 571
584 572
585 573
586 574
587 575
588 576
589 577
590 578
591 579
592 580
593 581
594 582
595 583
596 584
597 585
598 586
599 587
600 588
601 589
602 590
603 591
604 592
605 593
606 594
607 595
608 596
609 597
610 598
611 599
612 600
613 601
614 602
615 603
616 604
617 605
618 606
619 607
620 608
621 609
622 610
623 611
624 612
625 613
626 614
627 615
628 616
629 617
630 618
631 619
632 620
633 621
634 622
635 623
636 624
637 625
638 626
639 627
640 628
641 629
642 630
643 631
644 632
645 633
646 634
647 635
648 636
649 637
650 638
651 639
652 640
653 641
654 642
655 643
656 644
657 645
658 646
659 647
660 648
661 649
662 650
663 651
664 652
665 653
666 654
667 655
668 656
669 657
670 658
671 659
672 660
673 661
674 662
675 663
676 664
677 665
678 666
679 667
680 668
681 669
682 670
683 671
684 672
685 673
686 674
687 675
688 676
689 677
690 678
691 679
692 680
693 681
694 682
695 683
696 684
697 685
698 686
699 687
700 688
701 689
702 690
703 691
704 692
705 693
706 694
707 695
708 696
709 697
710 698
711 699
712 700
713 701
714 702
715 703
716 704
717 705
718 706
719 707
720 708
721 709
722 710
723 711
724 712
725 713
726 714
727 715
728 716
729 717
730 718
731 719
732 720
733 721
734 722
735 723
736 724
737 725
738 726
739 727
740 728
741 729
742 730
743 731
744 732
745 733
746 734
747 735
748 736
749 737
750 738
751 739
752 740
753 741
754 742
755 743
756 744
757 745
758 746
759 747
760 748
761 749
762 750
763 751
764 752
765 753
766 754
767 755
768 756
769 757
770 758
771 759
772 760
773 761
774 762
775 763
776 764
777 765
778 766
779 767
780 768
781 769
782 770
783 771
784 772
785 773
786 774
787 775
788 776
789 777
790 778
791 779
792 780
793 781
794 782
795 783
796 784
797 785
798 786
799 787
800 788
801 789
802 790
803 791
804 792
805 793
806 794
807 795
808 796
809 797
810 798
811 799
812 800
813 801
814 802
815 803
816 804
817 805
818 806
819 807
820 808
821 809
822 810
823 811
824 812
825 813
826 814
827 815
828 816
829 817
830 818
831 819
832 820
833 821
834 822
835 823
836 824
837 825
838 826
839 827
840 828
841 829
842 830
843 831
844 832
845 833
846 834
847 835
848 836
849 837
850 838
851 839
852 840
853 841
854 842
855 843
856 844
857 845
858 846
859 847
860 848
861 849
862 850
863 851
864 852
865 853
866 854
867 855
868 856
869 857
870 858
871 859
872 860
873 861
874 862
875 863
876 864
877 865
878 866
879 867
880 868
881 869
882 870
883 871
884 872
885 873
886 874
887 875
888 876
889 877
890 878
891 879
892 880
893 881
894 882
895 883
896 884
897 885
898 886
899 887
900 888
901 889
902 890
903 891
904 892
905 893
906 894
907 895
908 896
909 897
910 898
911 899
912 900
913 901
914 902
915 903
916 904
917 905
918 906
919 907
920 908
921 909
922 910
923 911
924 912
925 913
926 914
927 915
928 916
929 917
930 918
931 919
932 920
933 921
934 922
935 923
936 924
937 925
938 926
939 927
940 928
941 929
942 930
943 931
944 932
945 933
946 934
947 935
948 936
949 937
950 938
951 939
952 940
953 941
954 942
955 943
956 944
957 945
958 946
959 947
960 948
961 949
962 950
963 951
964 952
965 953
966 954
967 955
968 956
969 957
970 958
971 959
972 960
973 961
974 962
975 963
976 964
977 965
978 966
979 967
980 968
981 969
982 970
983 971
984 972
985 973
986 974
987 975
988 976
989 977
990 978
991 979
992 980
993 981
994 982
995 983
996 984
997 985
998 986
999 987
1000 988
1001 989
1002 990
1003 991
1004 992
1005 993
1006 994
1007 995
1008 996
1009 997
1010 998
1011 999
1012 1000
1013 1001
1014 1002
1015 1003
1016 1004
1017 1005
1018 1006
1019 1007
1020 1008
1021 1009
1022 1010
1023 1011
1024 1012
1025 1013
1026 1014
1027 1015
1028 1016
1029 1017
1030 1018
1031 1019
1032 1020
1033 1021
1034 1022
1035 1023
1036 1024
1037 1025
1038 1026
1039 1027
1040 1028
1041 1029
1042 1030
1043 1031
1044 1032
1045 1033
1046 1034
1047 1035
1048 1036
1049 1037
1050 1038
1051 1039
1052 1040
1053 1041
1054 1042
1055 1043
1056 1044
1057 1045
1058 1046
1059 1047
1060 1048
1061 1049
1062 1050
1063 1051
1064 1052
1065 1053
1066 1054
1067 1055
1068 1056
1069 1057
1070 1058
1071 1059
1072 1060
1073 1061
1074 1062
1075 1063
1076 1064
1077 1065
1078 1066
1079 1067
1080 1068
1081 1069
1082 1070
1083 1071
1084 1072
1085 1073
1086 1074
1087 1075
1088 1076
1089 1077
1090 1078
1091 1079
1092 1080
1093 1081
1094 1082
1095 1083
1096 1084
1097 1085
1098 1086
1099 1087
1100 1088
1101 1089
1102 1090
1103 1091
1104 1092
1105 1093
1106 1094
1107 1095
1108 1096
1109 1097
1110 1098
1111 1099
1112 1100
1113 1101
1114 1102
1115 1103
1116 1104
1117 1105
1118 1106
1119 1107
1120 1108
1121 1109
1122 1110
1123 1111
1124 1112
1125 1113
1126 1114
1127 1115
1128 1116
1129 1117
1130 1118
1131 1119
1132 1120
1133 1121
1134 1122
1135 1123
1136 1124
1137 1125
1138 1126
1139 1127
1140 1128
1141 1129
1142 1130
1143 1131
1144 1132
1145 1133
1146 1134
1147 1135
1148 1136
1149 1137
1150 1138
1151 1139
1152 1140
1153 1141
1154 1142
1155 1143
1156 1144
1157 1145
1158 1146
1159 1147
1160 1148
1161 1149
1162 1150
1163 1151
1164 1152
1165 1153
1166 1154
1167 1155
1168 1156
1169 1157
1170 1158
1171 1159
1172 1160
1173 1161
1174 1162
1175 1163
1176 1164
1177 1165
1178 1166
1179 1167
1180 1168
1181 1169
1182 1170
1183 1171
1184 1172
1185 1173
1186 1174
1187 1175
1188 1176
1189 1177
1190 1178
1191 1179
1192 1180
1193 1181
1194 1182
1195 1183
1196 1184
1197 1185
1198 1186
1199 1187
1200 1188
1201 1189
1202 1190
1203 1191
1204 1192
1205 1193
1206 1194
1207 1195
1208 1196
1209 1197
1210 1198
1211 1199
1212 1200
1213 1201
1214 1202
1215 1203
1216 1204
1217 1205
1218 1206
1219 1207
1220 1208
1221 1209
1222 1210
1223 1211
1224 1212
1225 1213
1226 1214
1227 1215
1228 1216
1229 1217
1230 1218
1231 1219
1232 1220
1233 1221
1234 1222
1235 1223
1236 1224
1237 1225
1238 1226
1239 1227
1240 1228
1241 1229
1242 1230
1243 1231
1244 1232
1245 1233
1246 1234
1247 1235
1248 1236
1249 1237
1250 1238
1251 1239
1252 1240
1253 1241
1254 1242
1255 1243
1256 1244
1257 1245
1258 1246
1259 1247
1260 1248
1261 1249
1262 1250
1263 1251
1264 1252
1265 1253
1266 1254
1267 1255
1268 1256
1269 1257
1270 1258
1271 1259
1272 1260
1273 1261
1274 1262
1275 1263
1276 1264
1277 1265
1278 1266
1279 1267
1280 1268
1281 1269
1282 1270
1283 1271
1284 1272
1285 1273
1286 1274
1287 1275
1288 1276
1289 1277
1290 1278
1291 1279
1292 1280
1293 1281
1294 1282
1295 1283
1296 1284
1297 1285
1298 1286
1299 1287
1300 1288
1301 1289
1302 1290
1303 1291
1304 1292
1305 1293
1306 1294
1307 1295
1308 1296
1309 1297
1310 1298
1311 1299
1312 1300
1313 1301
1314 1302
1315 1303
1316 1304
1317 1305
1318 1306
1319 1307
1320 1308
1321 1309
1322 1310
1323 1311
1324 1312
1325 1313
1326 1314
1327 1315
1328 1316
1329 1317
1330 1318
1331 1319
1332 1320
1333 1321
1334 1322
1335 1323
1336 1324
1337 1325
1338 1326
1339 1327
1340 1328
1341 1329
1342 1330
1343 1331
1344 1332
1345 1333
1346 1334
1347 1335
1348 1336
1349 1337
1350 1338
1351 1339
1352 1340
1353 1341
1354 1342
1355 1343
1356 1344
1357 1345
1358 1346
1359 1347
1360 1348
1361 1349
1362 1350
1363 1351
1364 1352
1365 1353
1366 1354
1367 1355
1368 1356
1369 1357
1370 1358
1371 1359
1372 1360
1373 1361
1374 1362
1375 1363
1376 1364
1377 1365
1378 1366
1379 1367
1380 1368
1381 1369
1382 1370
1383 1371
1384 1372
1385 1373
1386 1374
1387 1375
1388 1376
1389 1377
1390 1378
1391 1379
1392 1380
1393 1381
1394 1382
1395 1383
1396 1384
1397 1385
1398 1386
1399 1387
1400 1388
1401 1389
1402 1390
1403 1391
1404 1392
1405 1393
1406 1394
1407 1395
1408 1396
1409 1397
1410 1398
1411 1399
1412 1400
1413 1401
1414 1402
1415 1403
1416 1404
1417 1405
1418 1406
1419 1407
1420 1408
1421 1409
1422 1410
1423 1411
1424 1412
1425 1413
1426 1414
1427 1415
1428 1416
1429 1417
1430 1418
1431 1419
1432 1420
1433 1421
1434 1422
1435 1423
1436 1424
1437 1425
1438 1426
1439 1427
1440 1428
1441 1429
1442 1430
1443 1431
1444 1432
1445 1433
1446 1434
1447 1435
1448 1436
1449 1437
1450 1438
1451 1439
1452 1440
1453 1441
1454 1442
1455 1443
1456 1444
1457 1445
1458 1446
1459
```

Top Rated Movies & Current Temperature



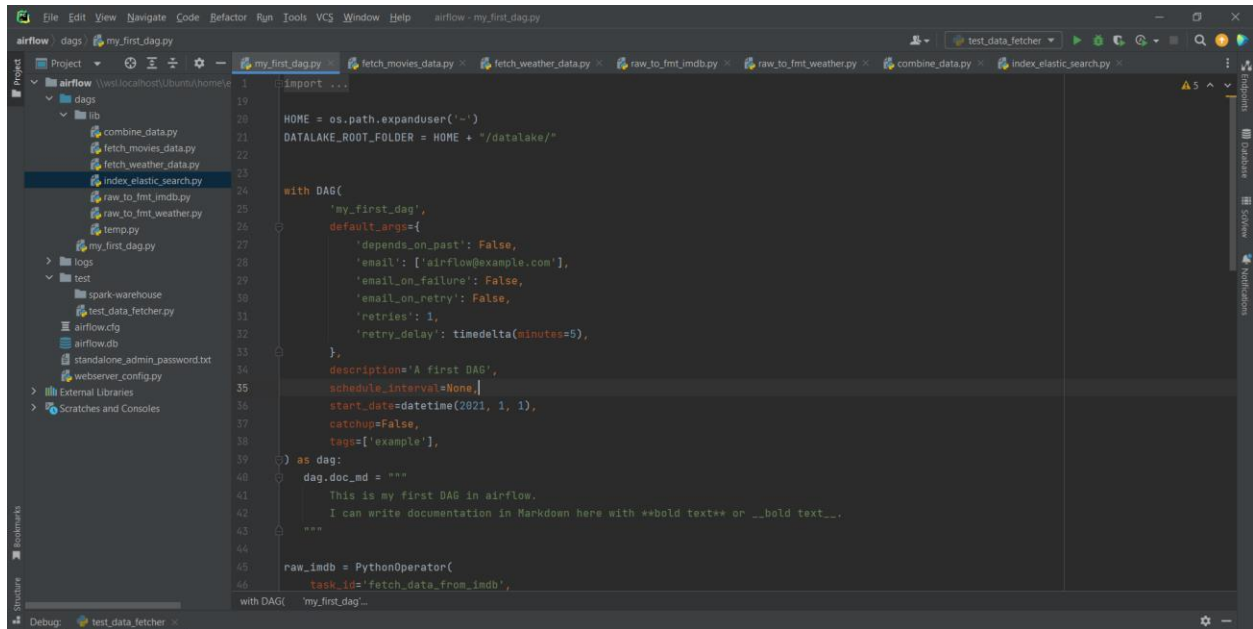
Analysis of Top Rated Movies & Upcoming Movies



Weather Forecast & Trending Movies

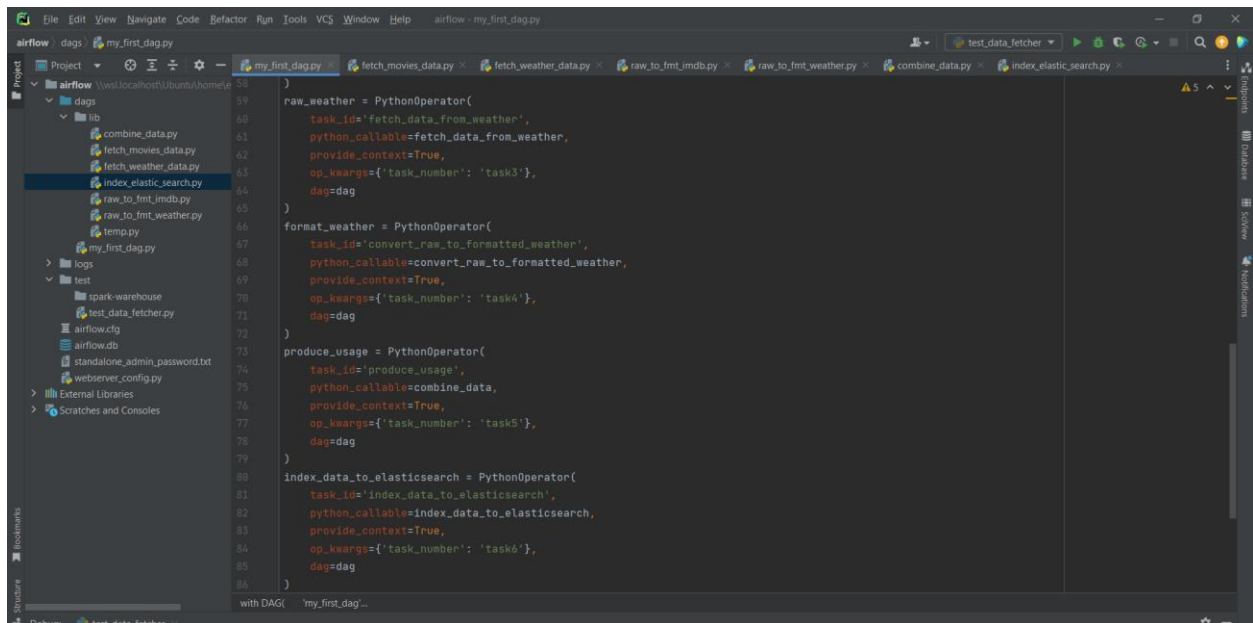
Directed Acyclic Graphs - DAGs

We use Apache Airflow to create DAGs (Directed Acyclic Graphs) to manage workflow orchestration of our data. The tasks and dependencies are defined in python (Figure 10 and 11) and Airflow manages the scheduling and execution. The DAG that reflects our pipeline architecture is shown in Figure 12.



```
1 import os
2
3 HOME = os.path.expanduser('~')
4 DATALAKE_ROOT_FOLDER = HOME + "/datalake/"
5
6
7 with DAG(
8     'my_first_dag',
9     default_args={
10         'depends_on_past': False,
11         'email': ['airflow@example.com'],
12         'email_on_failure': False,
13         'email_on_retry': False,
14         'retries': 1,
15         'retry_delay': timedelta(minutes=5),
16     },
17     description='A first DAG',
18     schedule_interval=None,
19     start_date=datetime(2021, 1, 1),
20     catchup=False,
21     tags=['example'],
22 ) as dag:
23     dag.doc_md = """
24     This is my first DAG in airflow.
25     I can write documentation in Markdown here with bold text or bold text...
26     """
27
28     raw_imdb = PythonOperator(
29         task_id='fetch_data_from_imdb',
30         python_callable=fetch_data_from_imdb,
31     )
32
33     with DAG(
34         'my_first_dag'...
```

Figure 10: DAG Python Code i



```
58
59
60 raw_weather = PythonOperator(
61     task_id='fetch_data_from_weather',
62     python_callable=fetch_data_from_weather,
63     provide_context=True,
64     op_kwargs={'task_number': 'task3'},
65     dag=dag
66 )
67
68 format_weather = PythonOperator(
69     task_id='convert_raw_to_formatted_weather',
70     python_callable=convert_raw_to_formatted_weather,
71     provide_context=True,
72     op_kwargs={'task_number': 'task4'},
73     dag=dag
74 )
75
76 produce_usage = PythonOperator(
77     task_id='produce_usage',
78     python_callable=combine_data,
79     provide_context=True,
80     op_kwargs={'task_number': 'task5'},
81     dag=dag
82 )
83
84 index_data_to_elasticsearch = PythonOperator(
85     task_id='index_data_to_elasticsearch',
86     python_callable=index_data_to_elasticsearch,
87     provide_context=True,
88     op_kwargs={'task_number': 'task6'},
89     dag=dag
90 )
91
92 with DAG(
93     'my_first_dag'...
```

Figure 11: DAG Python Code ii

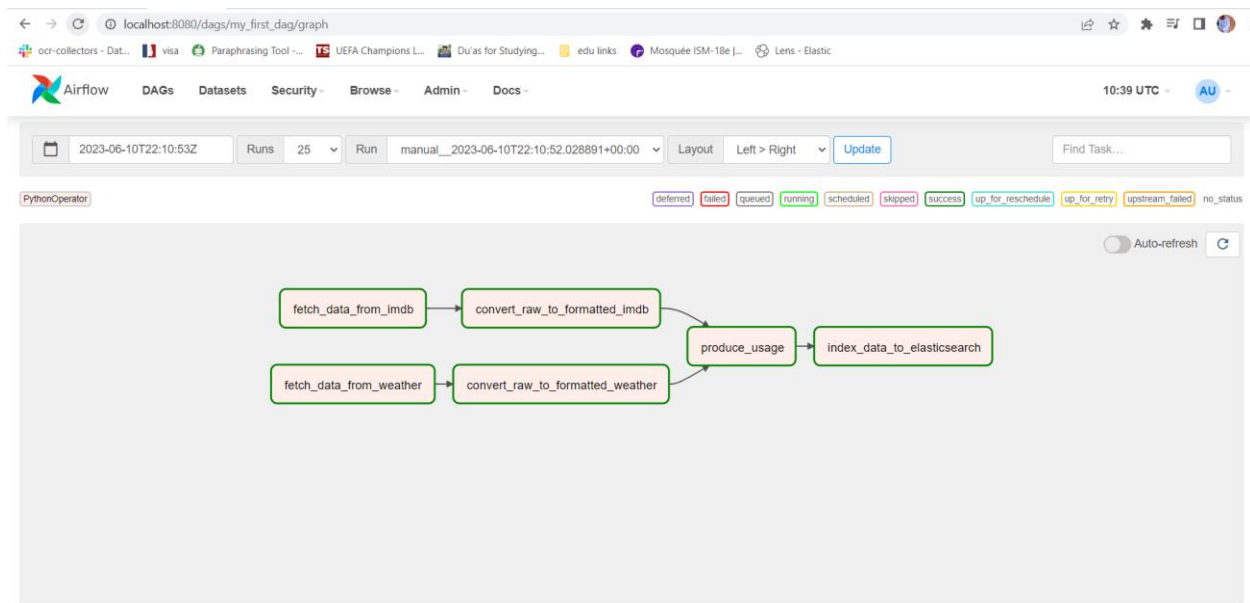


Figure 12 : DAG Airflow Graph View

Conclusion

Throughout the implementation of this project, we have gained extensive knowledge and valuable experience in various technologies. Despite facing challenges along the way, the outcome has been highly successful, providing us with in-depth learnings and honing our skills.

The dynamic and comprehensive visualization of cinema and weather data offers significant benefits to consumers, researchers, statisticians, data scientists, and others, empowering them to make well-informed decisions, perform comparisons, and apply filters based on their specific needs. By considering trending movies, upcoming releases, top-rated films, user ratings, and weather forecasts for the upcoming week, stakeholders can leverage these insights for powerful recommendations.

This project has allowed us to broaden our understanding of key concepts such as big data, data lakes, data transformation, visualization, streaming, data analytics, and workflow. We have become proficient in utilizing various tools and technologies, including Apache Airflow, Apache Spark, cloud-based Elasticsearch and Kibana, Python, Panda, Parquet, and JSON, among others.