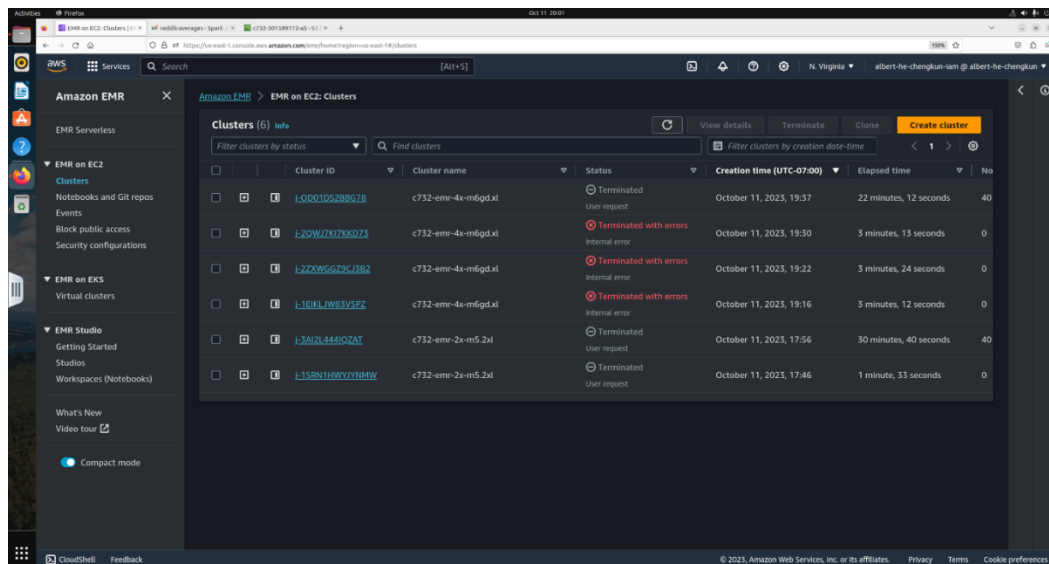


Name: He Chengkun

Student ID:

1. Take a screen shot of your list of EMR clusters (if more than one page, only the page with the most recent), showing that all have Terminated status.



2. For Section 2:

- a. What fraction of the input file was prefiltered by S3 before it was sent to Spark?

Stage Id	Description	Submitted	Duration	Tasks: Succeeded/Total	Input	Output
0	json at NativeMethodAccessorImpl.java:0	+ details 2023/10/12 01:13:44	5 s	4/4	2.6 MiB	27.2 KiB

Stage Id	Description	Submitted	Duration	Tasks: Succeeded/Total	Input	Output
0	json at NativeMethodAccessorImpl.java:0	+ details 2023/10/12 01:23:31	4 s	4/4	97.7 KiB	27.2 KiB

The original input is 2.4 MB, and the input prefiltered by S3 is 97.7 KB. Input size was reduced by ~2.30 MB.

- b. Comparing the different input numbers for the regular version versus the prefiltered one, what operations were performed by S3 and which ones performed in Spark?

```

number of output rows: 5,245
number of dynamic part: 0
union output: 0/5 6/6

▼ Details

== Parsed Logical Plan ==
InsertIntoHadoopFsRelationCommand s3://c732-301599772-a5/output/weather-2, false, JSON, [compression=gzip, path=s3://c732-301599772-a5/output/weather-2], Overwrite, [station, date, tmax]
+- Project [station#0, date#1, (cast(value#3 as double) / cast(10 as double)) AS tmax#17]
  +- Filter (observation#2 = TMAX)
    +- Filter StartsWith(station#0, CA)
      +- Filter isnull(qflag#5)
        +- Relation [station#0,date#1,observation#2,value#3,mflag#4,qflag#5,sflag#6,obstime#7] s3selectCSV

== Analyzed Logical Plan ==
InsertIntoHadoopFsRelationCommand s3://c732-301599772-a5/output/weather-2, false, JSON, [compression=gzip, path=s3://c732-301599772-a5/output/weather-2], Overwrite, [station, date, tmax]
+- Project [station#0, date#1, (cast(value#3 as double) / cast(10 as double)) AS tmax#17]
  +- Filter (observation#2 = TMAX)
    +- Filter StartsWith(station#0, CA)
      +- Filter isnull(qflag#5)
        +- Relation [station#0,date#1,observation#2,value#3,mflag#4,qflag#5,sflag#6,obstime#7] s3selectCSV

== Optimized Logical Plan ==
InsertIntoHadoopFsRelationCommand s3://c732-301599772-a5/output/weather-2, false, JSON, [compression=gzip, path=s3://c732-301599772-a5/output/weather-2], Overwrite, [station, date, tmax]
+- WriteFiles
  +- Project [station#0, date#1, (cast(value#3 as double) / 10.0) AS tmax#17]
    +- Filter (((isnotnull(station#0) AND isnotnull(observation#2)) AND isnull(qflag#5)) AND StartsWith(station#0, CA)) AND (observation#2 = TMAX)))
      +- FileScan s3selectCSV [station#0,date#1,observation#2,value#3,qflag#5] Batched: false, DataFilters: [isnotnull(station#0), isnotnull(observation#2), isnull(qflag#5), StartsWith(station#0, CA), (obs..., Format: CSVS3Select, Location: InMemoryFileIndex(1 paths)(s3://c732-301599772-a5/weather-1), PartitionFilters: [1], PushedFilters: [IsNotNull(station), IsNotNull(observation), IsNull(qflag), StringStartsWith(station,CA), EqualTo...], ReadSchema: struct:station:string,date:string,observation:string,value:int,qflag:string

== Physical Plan ==
Execute InsertIntoHadoopFsRelationCommand s3://c732-301599772-a5/output/weather-2, false, JSON, [compression=gzip, path=s3://c732-301599772-a5/output/weather-2], Overwrite, [station, date, tmax]
+- WriteFiles
  +- *(1) Project [station#0, date#1, (cast(value#3 as double) / 10.0) AS tmax#17]
    +- *(1) Filter (((isnotnull(station#0) AND isnotnull(observation#2)) AND isnull(qflag#5)) AND StartsWith(station#0, CA)) AND (observation#2 = TMAX)))
      +- FileScan s3selectCSV [station#0,date#1,observation#2,value#3,qflag#5] Batched: false, DataFilters: [isnotnull(station#0), isnotnull(observation#2), isnull(qflag#5), StartsWith(station#0, CA), (obs..., Format: CSVS3Select, Location: InMemoryFileIndex(1 paths)(s3://c732-301599772-a5/weather-1), PartitionFilters: [1], PushedFilters: [IsNotNull(station), IsNotNull(observation), IsNull(qflag), StringStartsWith(station,CA), EqualTo...], ReadSchema: struct:station:string,date:string,observation:string,value:int,qflag:string

```

Take a look of the Spark History of weather ETL S3 select. From Spark History -> SQL/DataFrame -> Details -> Physical Plan -> PushedFilters, we can know that all the filtering operations were pushed down to S3.

- For Section 3: Look up the hourly costs of the m7gd.xlarge instance on the EC2 On-Demand Pricing page. Estimate the cost of processing a dataset ten times as large as reddit-5 using just those 4 instances. If you wanted instead to process this larger dataset making full use of 16 instances, how would it have to be organized?

Viewing 671 of 671 available instances

Q m7gd.xlarge X 1 match < 1 >

Instance name ▲	On-Demand hourly rate ▼	vCPU ▼	Memory ▼	Storage ▼	Network performance ▼
m7gd.xlarge	\$0.2136	4	16 GiB	1 x 237 NVMe SSD	Up to 12500 Megabit

