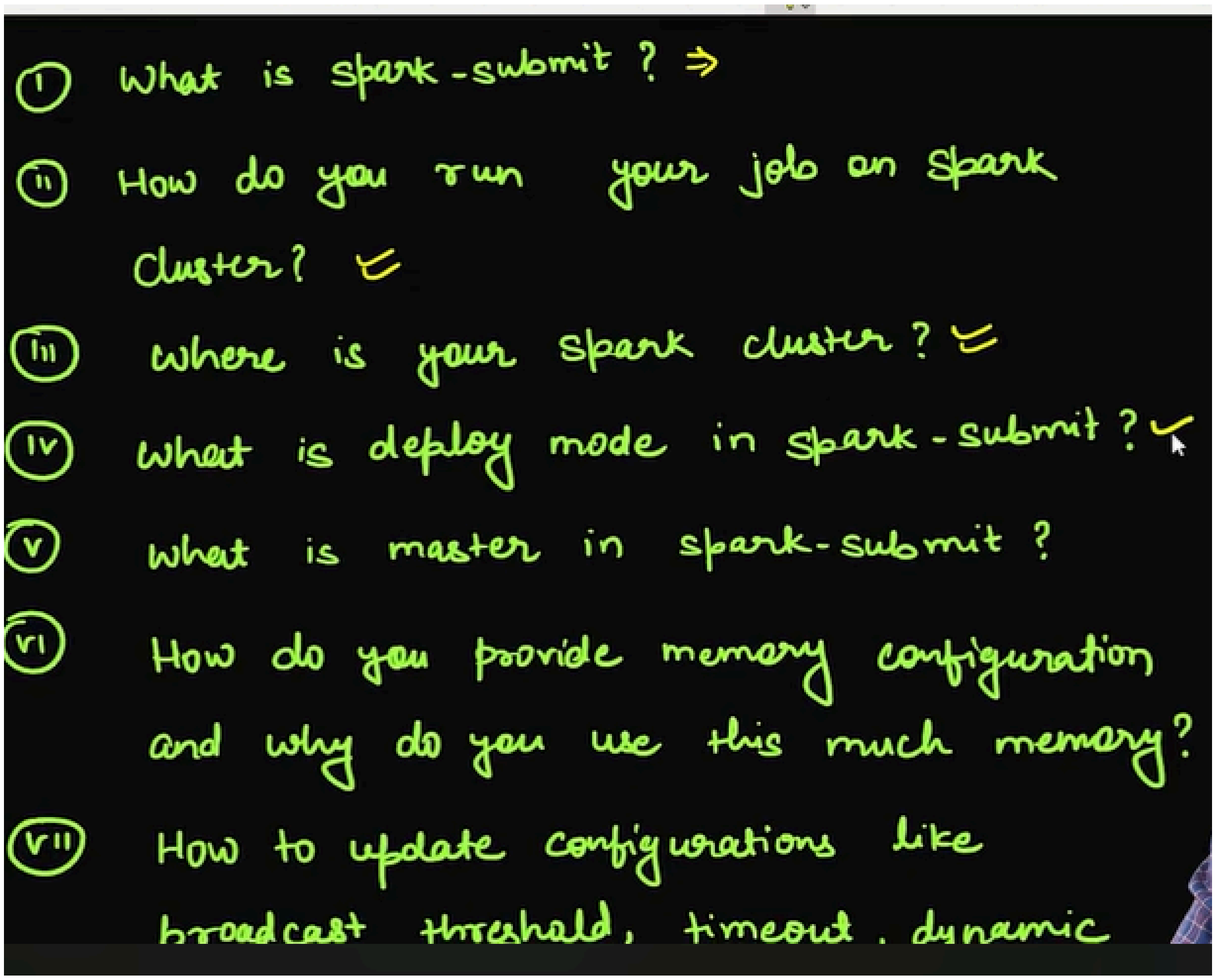
Spark Submit

1. Potential interview questions



(i) What is spark-submit ? ⇒

(ii) How do you run your job on spark Cluster? ✓

(iii) where is your spark cluster ? ✓

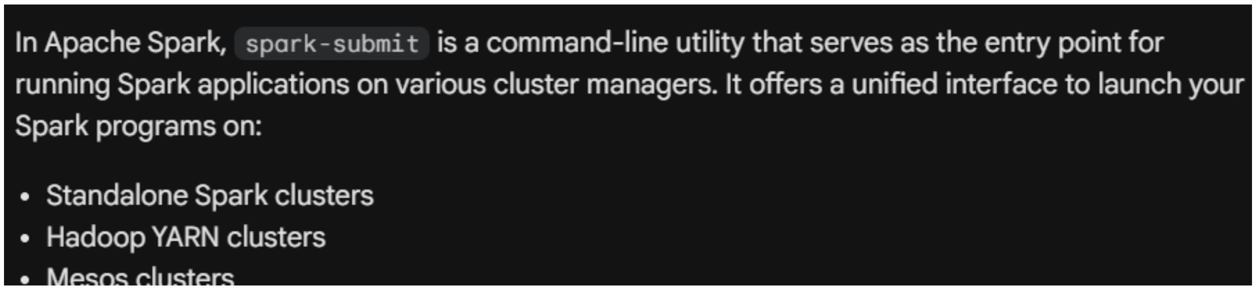
(iv) what is deploy mode in spark-submit ? ✓

(v) what is master in spark-submit ?

(vi) How do you provide memory configuration and why do you use this much memory?

(vii) How to update configurations like broadcast threshold, timeout, dynamic

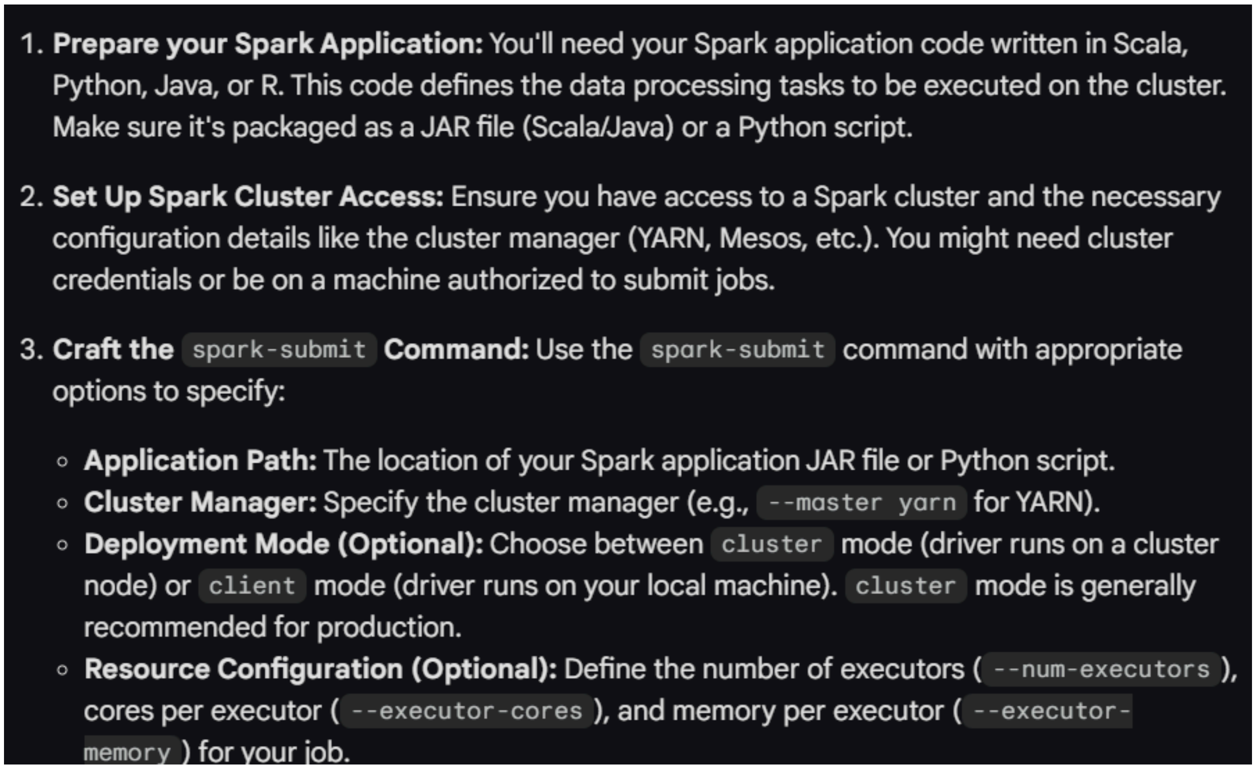
   -dynamic memory allocation
2. What is spark submit?
3. It is a command line tool which runs our spark application..it will take all the jar files and all the other files of application and runs the application



In Apache Spark, `spark-submit` is a command-line utility that serves as the entry point for running Spark applications on various cluster managers. It offers a unified interface to launch your Spark programs on:

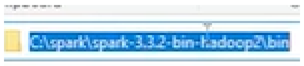
- Standalone Spark clusters
- Hadoop YARN clusters
- Mesos clusters

4. How do you run your job on a spark cluster?



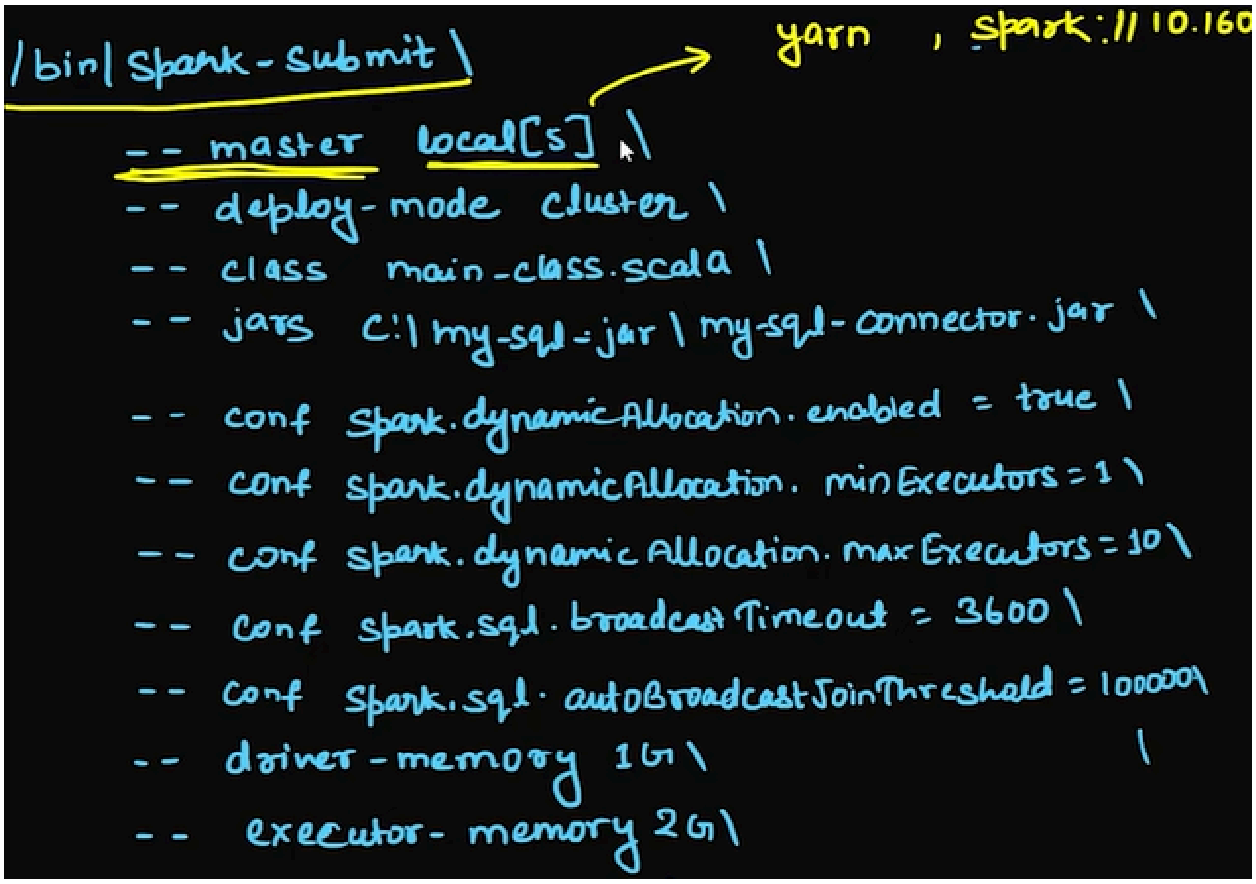
1. **Prepare your Spark Application:** You'll need your Spark application code written in Scala, Python, Java, or R. This code defines the data processing tasks to be executed on the cluster. Make sure it's packaged as a JAR file (Scala/Java) or a Python script.
2. **Set Up Spark Cluster Access:** Ensure you have access to a Spark cluster and the necessary configuration details like the cluster manager (YARN, Mesos, etc.). You might need cluster credentials or be on a machine authorized to submit jobs.
3. **Craft the `spark-submit` Command:** Use the `spark-submit` command with appropriate options to specify:
    - **Application Path:** The location of your Spark application JAR file or Python script.
    - **Cluster Manager:** Specify the cluster manager (e.g., `--master yarn` for YARN).
    - **Deployment Mode (Optional):** Choose between `cluster` mode (driver runs on a cluster node) or `client` mode (driver runs on your local machine). `cluster` mode is generally recommended for production.
    - **Resource Configuration (Optional):** Define the number of executors (`--num-executors`), cores per executor (`--executor-cores`), and memory per executor (`--executor-memory`) for your job.

5.
6. Lets deep dive into spark submit
7. So if install the spark in our local setup..then we can access spark submit inside bin

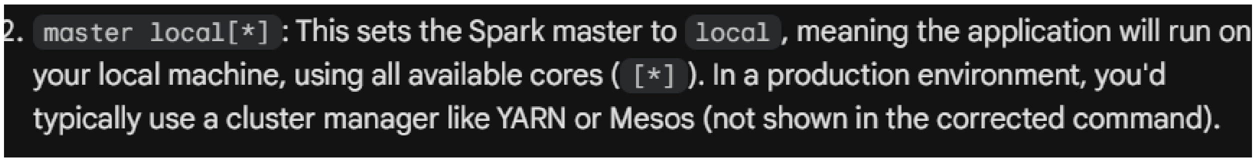


C:\spark\spark-3.3.2-bin-hadoop2\bin

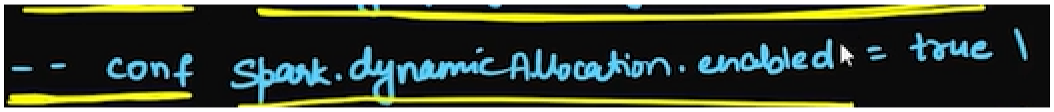
8. Lets see a sample spark submit



```
                                  yarn   , spark://10.160
/bin/spark-submit \
    -- master  local[5] \
    -- deploy-mode  cluster \
    -- class  main-class.scala \
    -- jars  C:\my-sql-jar\my-sql-connector.jar \
    -- conf  spark.dynamicAllocation.enabled = true \
    -- conf  spark.dynamicAllocation. minExecutors=1 \
    -- conf  spark.dynamicAllocation. maxExecutors=10 \
    -- conf  spark.sql.broadcastTimeout = 3600 \
    -- conf  spark.sql.autoBroadcastJoinThreshold = 100000 \
    -- driver-memory  1G \                                    \
    -- executor-memory 2G \
```

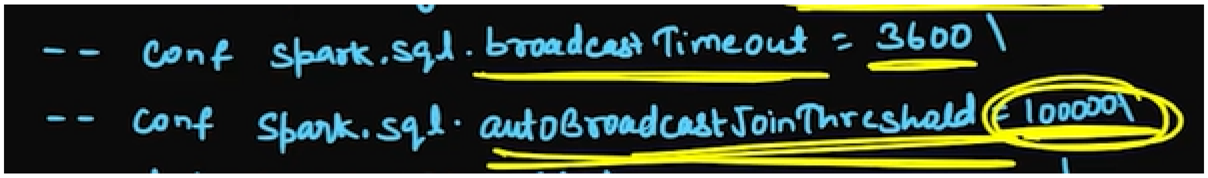
9. This commands explained : https://g.co/gemini/share/275f0f1399a9
10. 

2. `master local[*]`: This sets the Spark master to `local`, meaning the application will run on your local machine, using all available cores (`[*]`). In a production environment, you'd typically use a cluster manager like YARN or Mesos (not shown in the corrected command).

11. Here if 

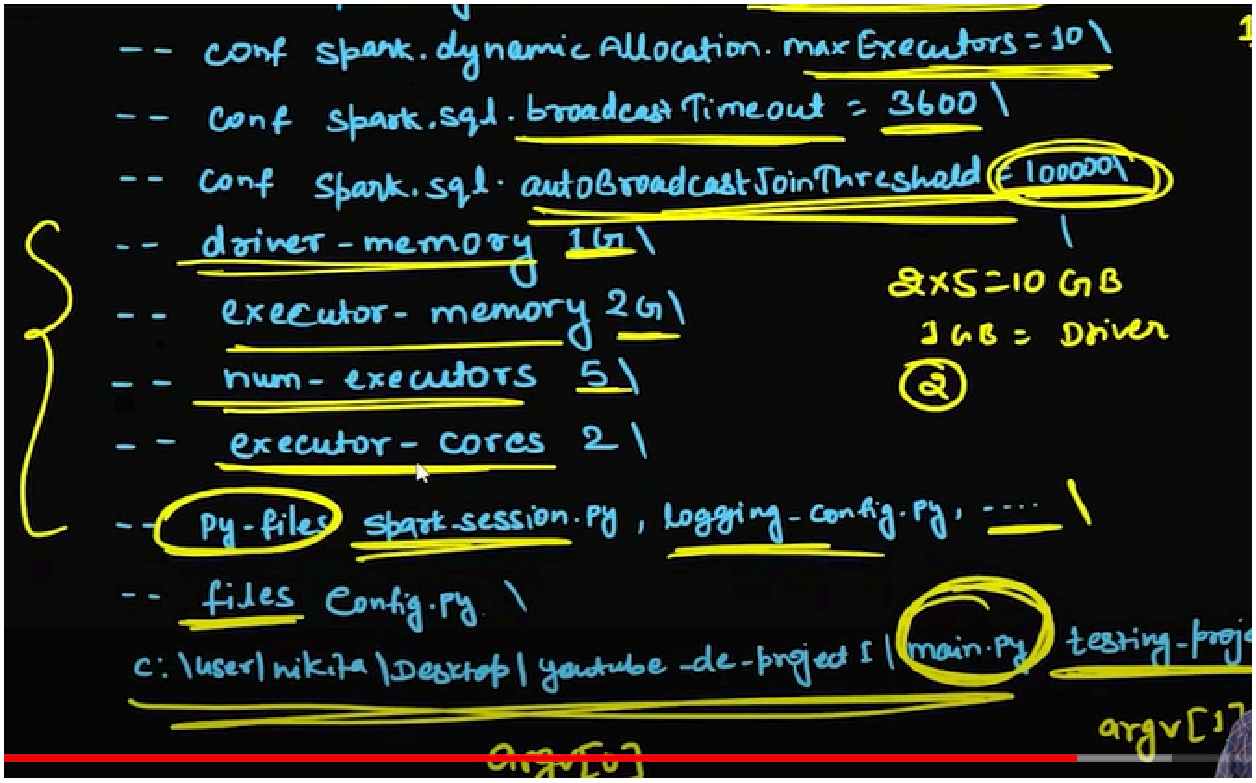
```
-- conf  spark.dynamicAllocation.enabled = true \
```

then if our application is not running or stays idle then…it dynamically decrease the memory and allocates to other resources/application

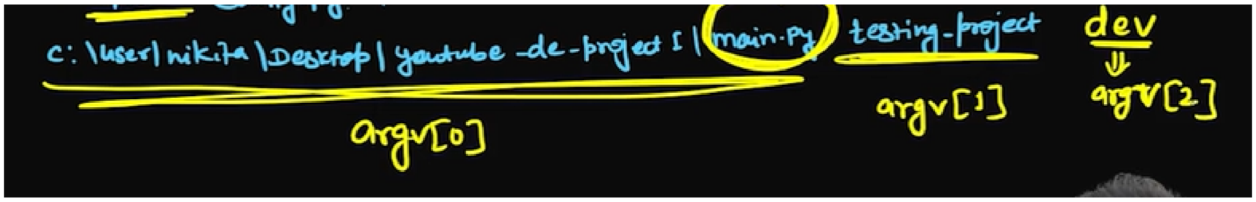
12. 

```
-- conf  spark.sql.broadcastTimeout = 3600 \
-- conf  spark.sql.autoBroadcastJoinThreshold = 100000 \
```

13. Here if the broadcasting does not completes by 3600s..then it will stop the broadcast
14. In the above spark submit commands…we are just giving our application jar file and all the configs required for our application and the memory required for our application

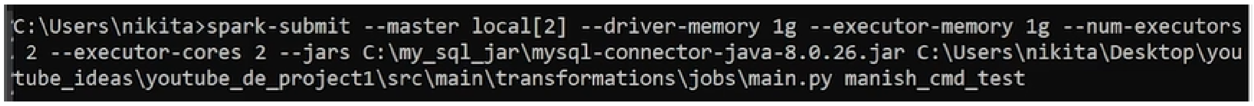
15. 

```
-- conf spark.dynamicAllocation.maxExecutors=10\
-- conf spark.sql.broadcastTimeout = 3600 \
-- conf spark.sql.autoBroadcastJoinThreshold=1000000\
-- driver-memory 1G\
-- executor-memory 2G\
-- num-executors 5\
-- executor-cores 2 \
-- py-files spark.session.py , logging-config.py, ---- \
-- files config.py \
c:\user\nikita\Desktop\youtube-de-project1\main.py testing-proj
```

16. 

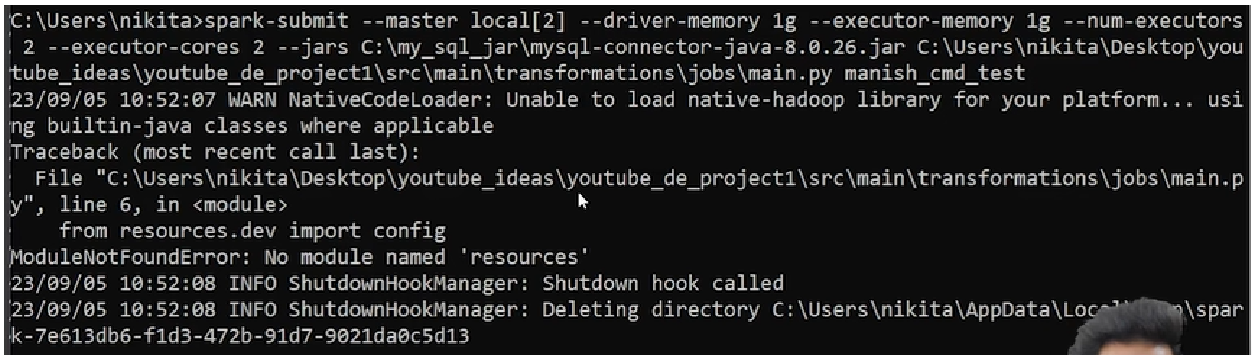
```
c:\user\nikita\Desktop\youtube-de-project1\main.py testing-project dev
```

17. So here if we want to run our application in dev..then we can just give dev
18. This is production level spark submit
19. Lets run a sample spark submit



```
C:\Users\nikita>spark-submit --master local[2] --driver-memory 1g --executor-memory 1g --num-executors
 2 --executor-cores 2 --jars C:\my_sql_jar\mysql-connector-java-8.0.26.jar C:\Users\nikita\Desktop\you
tube_ideas\youtube_de_project1\src\main\transformations\jobs\main.py manish_cmd_test
```

20. Here last line C:...is our argv[0] and manish_cmd_test is our argv[1]
21. After running

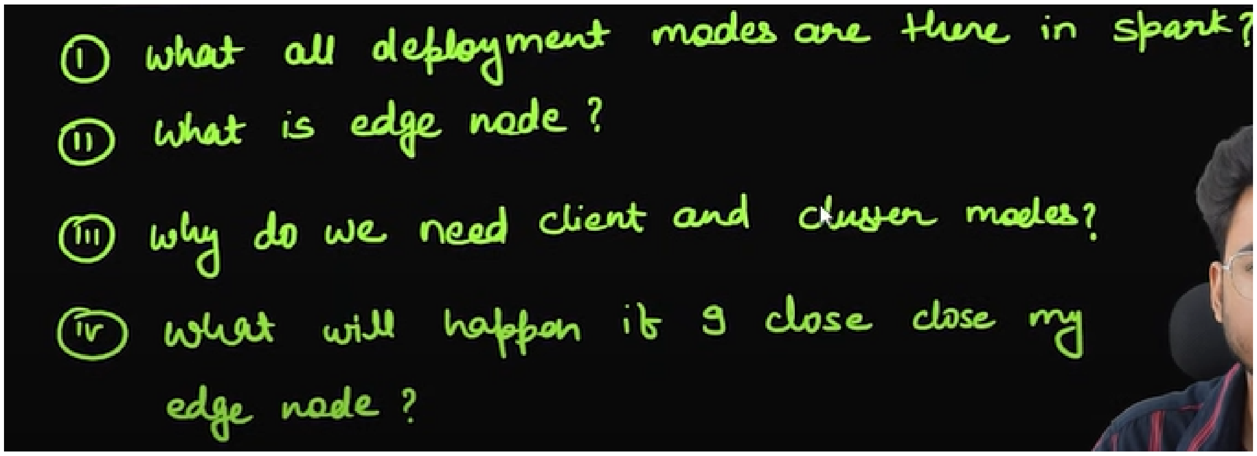


```
C:\Users\nikita>spark-submit --master local[2] --driver-memory 1g --executor-memory 1g --num-executors
 2 --executor-cores 2 --jars C:\my_sql_jar\mysql-connector-java-8.0.26.jar C:\Users\nikita\Desktop\you
tube_ideas\youtube_de_project1\src\main\transformations\jobs\main.py manish_cmd_test
23/09/05 10:52:07 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... usi
ng builtin-java classes where applicable
Traceback (most recent call last):
  File "C:\Users\nikita\Desktop\youtube_ideas\youtube_de_project1\src\main\transformations\jobs\main.p
y", line 6, in <module>
    from resources.dev import config
ModuleNotFoundError: No module named 'resources'
23/09/05 10:52:08 INFO ShutdownHookManager: Shutdown hook called
23/09/05 10:52:08 INFO ShutdownHookManager: Deleting directory C:\Users\nikita\AppData\Loc[illegible]\spar
k-7e613db6-f1d3-472b-91d7-9021da0c5d13
```

22. Will complete the spark project..and will understand more on spark submit

Deployment mode in spark

1. Potential interview questions



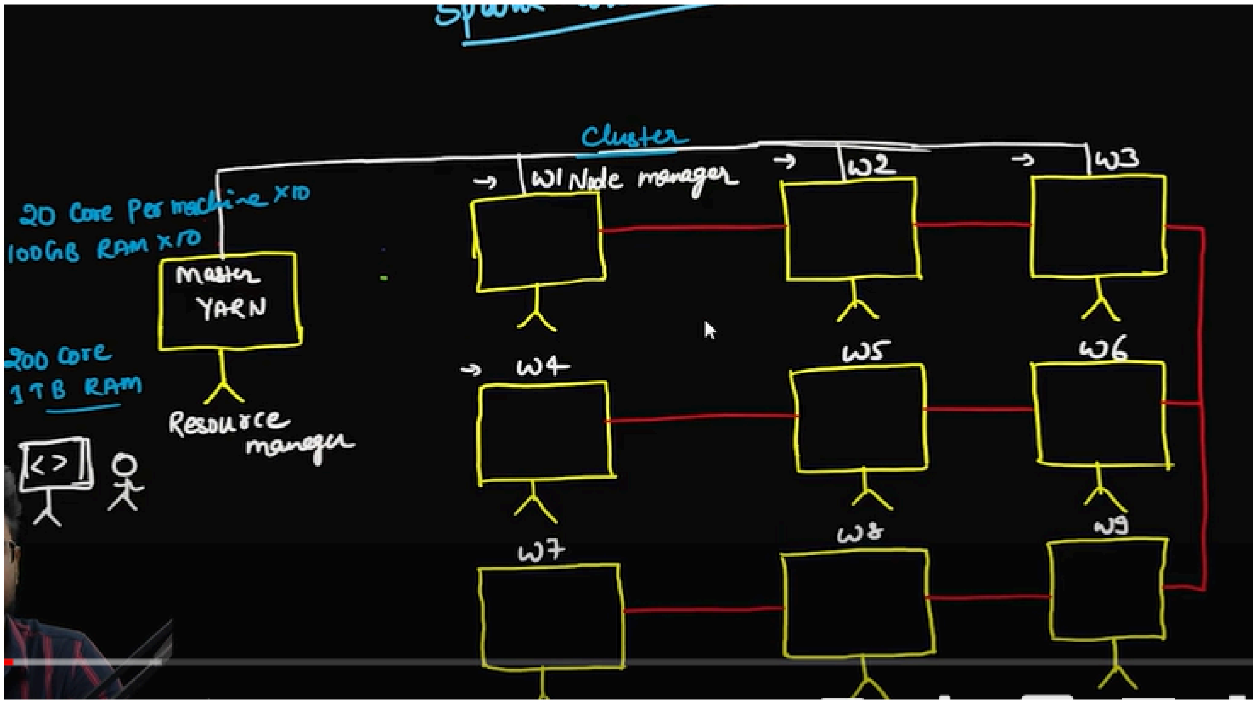
(i) what all deployment modes are there in spark?

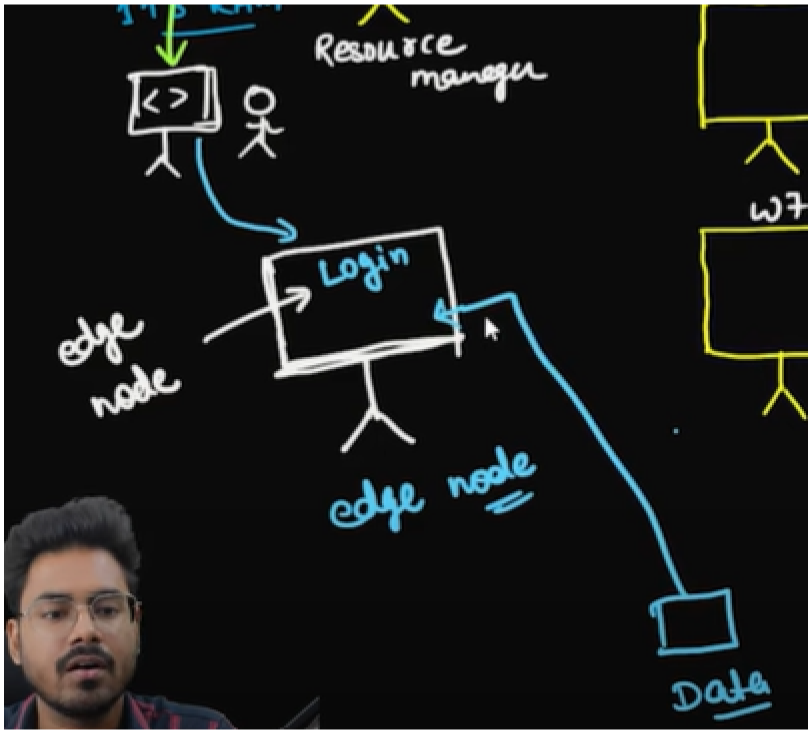
(ii) what is edge node ?

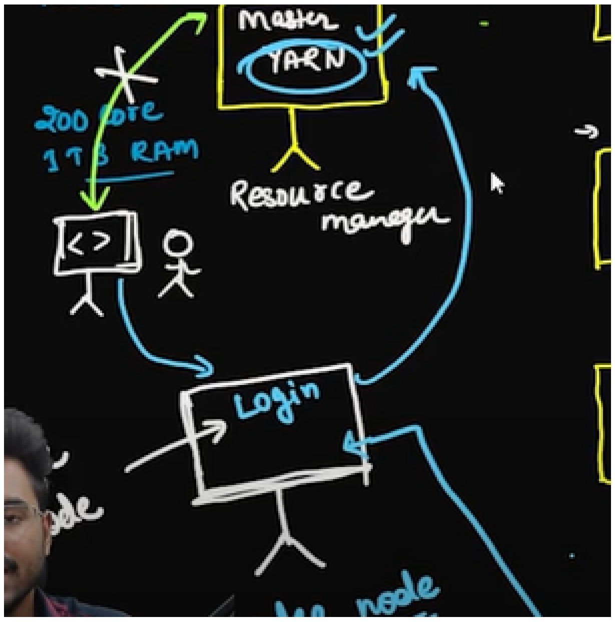
(iii) why do we need client and cluser modes?

(iv) what will happon if I close close my edge node ?

2. Lets learn this by using spark architecture

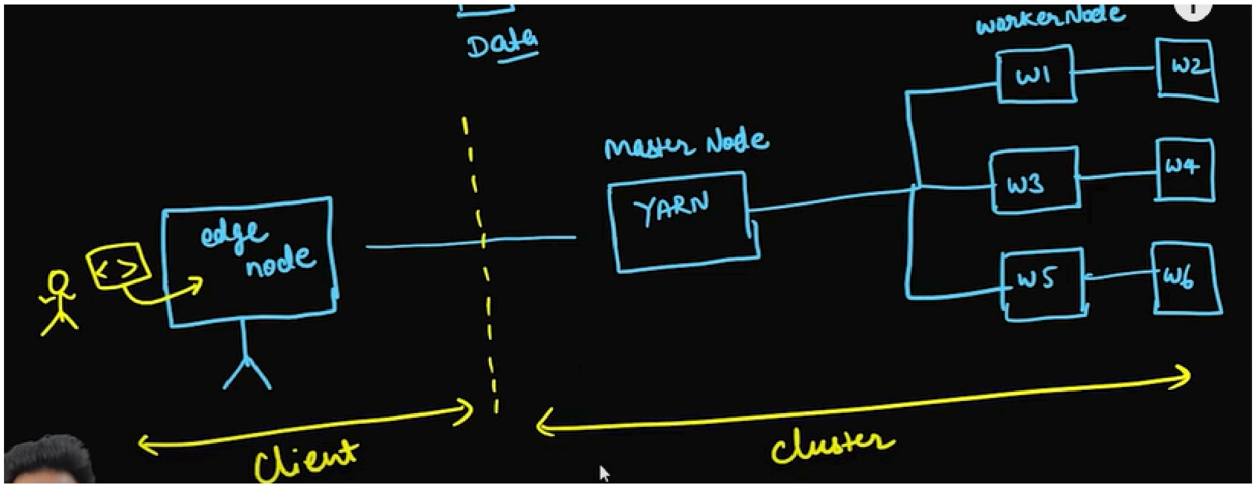


3. Here if a user is granted direct access to cluster…then he may go and manually copy a file in worker node..which is not a ideal thing to do


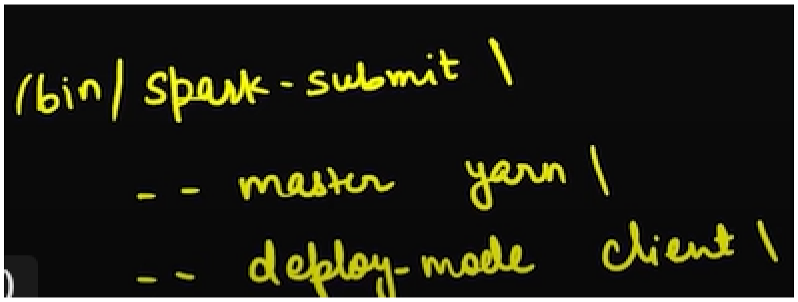
4. So to avoid this issue we use the help of edge node…which is a small commodity machine…which helps us to authenticate the user
5. Also now if we want to copy any data…first we send it to edge node..and now edge node will request master node…then master node replies back in which worker node shud the data must be copied


6. Actually edge nodes will be assigned by hadoop admin team in a company

7. Lets take a small cluster example

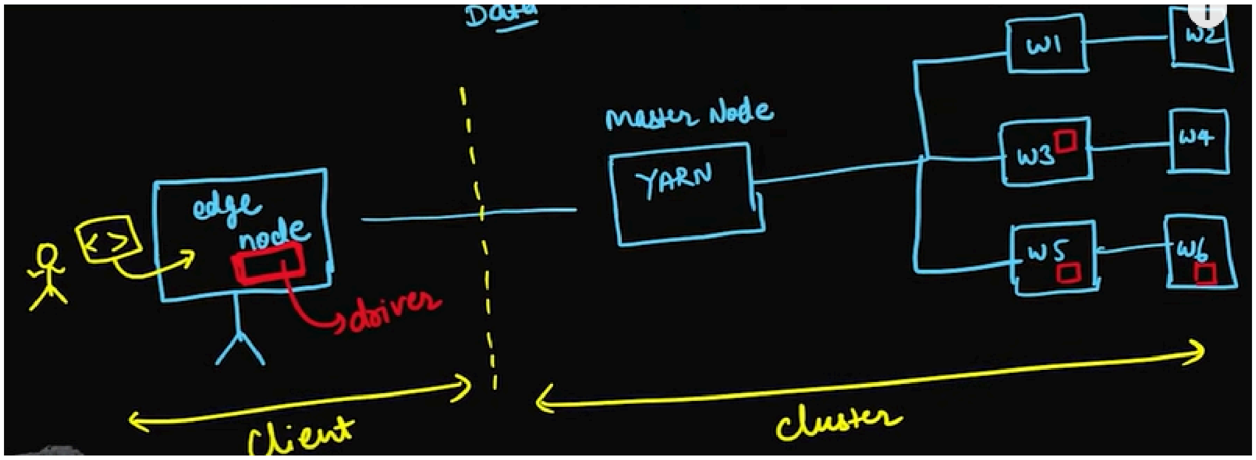


8. Here user has some code and sending it to edge node
9. Now if we run this in spark submit

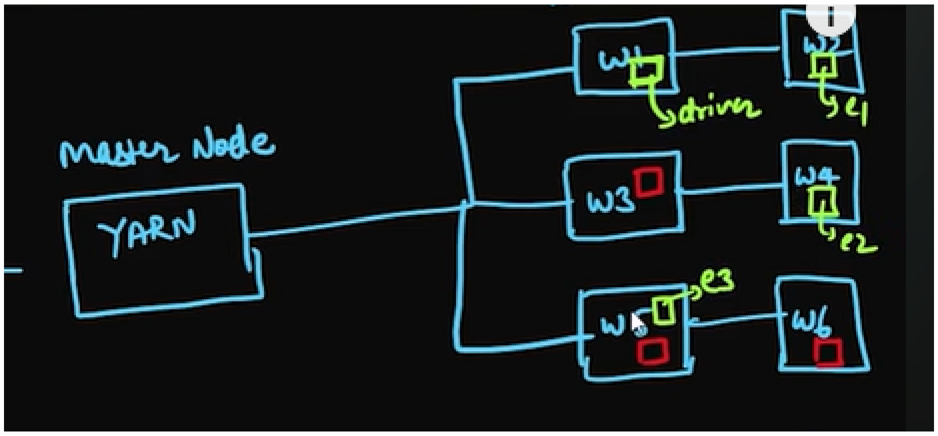


```
/bin/ spark-submit \
    -- master  yarn \
    -- deploy-mode  client \
```

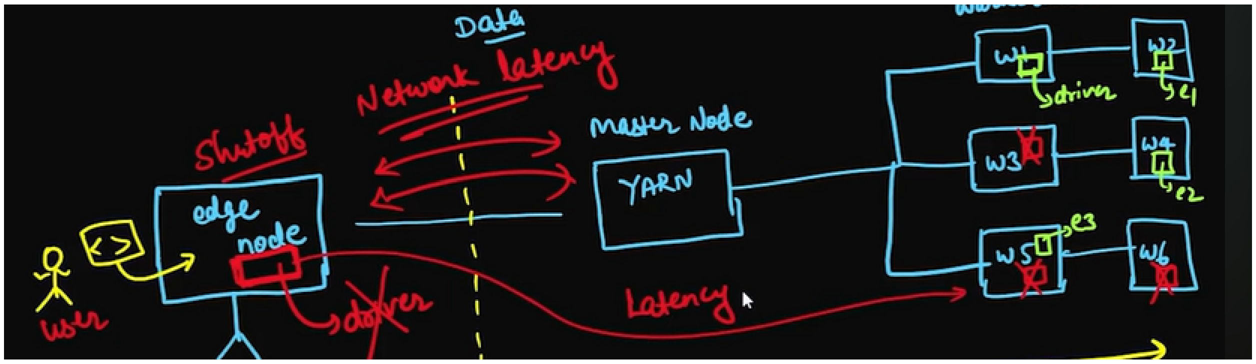
10. If the deployment mode is client …then the driver(application master container) will be made in edge node…and executors(3) will be in worker nodes



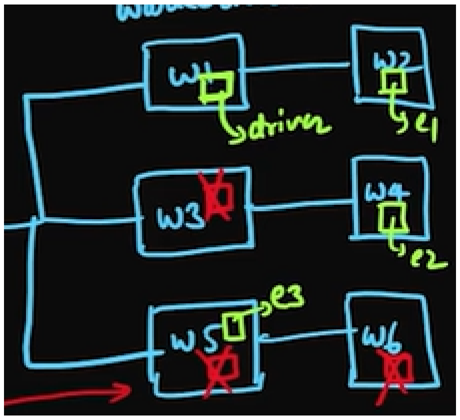
11. If the deploy-mode is cluster ..then driver and executors will be in worker nodes



12. Coming to the client node..
13. If a user by mistake shutoff's the driver node…then executors will also get shutdown…and our application get stopped..so this is one disadvantage
14. One more disadvantage is..there will be network latency in client …suppose if we are performing any broadcast action..then data must be travelled from edge node to worker nodes…which takes some time

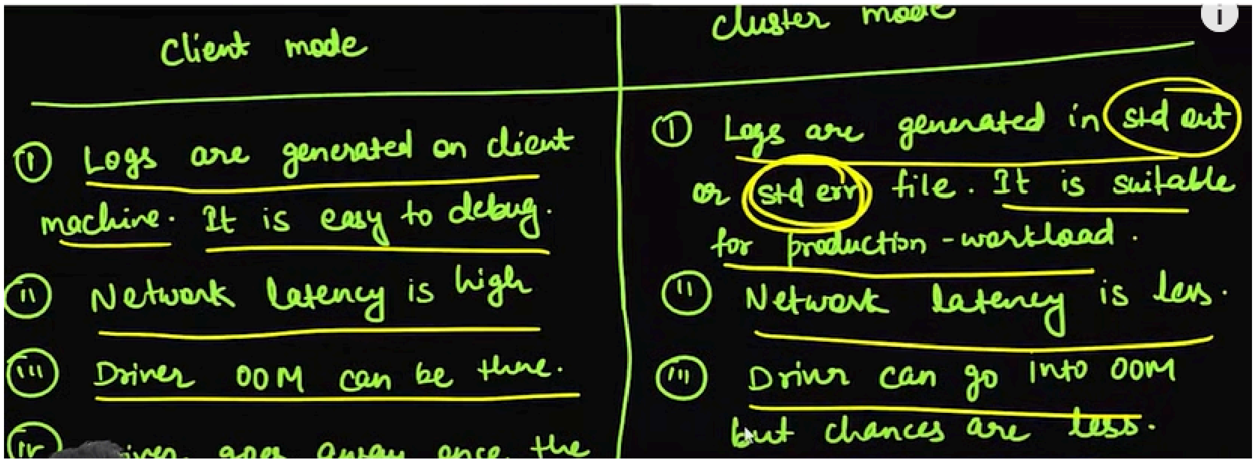


15. Advantage is that…we can see how our program is running in the backend in our display
16. Coming to cluster mode
17. There will be less latency..as driver is in the worker node itself

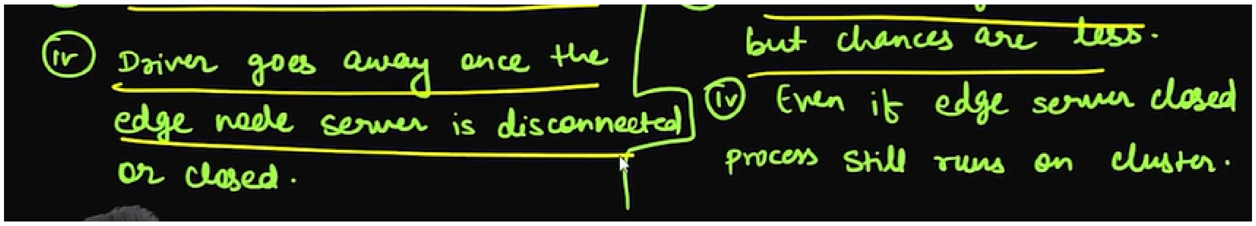


18. And even if the user shut's off the machine..the application will be run as the driver node has container

19. Now when ever we deploy through cluster mode..it gives us a application id…with this id…we can see whats happening in the cluster in webUI
20. Client vs Cluster



Client mode	cluster mode
(i) Logs are generated on client machine. It is easy to debug.	(i) Logs are generated in std out or std err file. It is suitable for production-workload.
(ii) Network latency is high	(ii) Network latency is less.
(iii) Driver OOM can be there.	(iii) Driver can go into OOM but chances are less.



but chances are less.

Client mode	cluster mode
(iv) Driver goes away once the edge node server is disconnected or closed.	(iv) Even if edge server closed process still runs on cluster.