Solución

1)Para este punto se plantearon 2 soluciones

1

```sql
with protocols_organized (protocol,client)
  as (
  SELECT
    protocol, client
    FROM traffic
    group by client, protocol
  )



SELECT
  distinct t.client,
  t.protocol
FROM traffic
INNER JOIN (
    SELECT
    STRING_AGG(protocol, ',') AS protocol, client
    FROM protocols_organized
    group by client
  ) AS t
ON traffic.client = t.client
ORDER BY client
```

	client	protocol
1	19-58-33-40-6E-66	BGP,DNS,POP,SNP
2	9E-43-EA-54-0A-E7	BGP,DNS,HTTP,HTTPS
3	A6-B6-94-1E-07-FE	BGP,DHCP,DNS,HTTPS,TCP
4	BB-0B-0C-1D-24-F4	IMAP,POP,SNP,TCP
5	E4-00-CE-46-3F-26	DNS,IMAP

2)

```
SELECT
  client,
  STUFF(
    (SELECT  distinct protocol +', '
     FROM traffic
     WHERE client = a.client
     FOR XML PATH ('')),
    1,
    0,
    '') AS [protocol]
FROM traffic AS a
GROUP BY client
ORDER BY client
```

	client	protocol
1	19-58-33-40-6E-66	BGP, DNS, POP, SNP,
2	9E-43-EA-54-0A-E7	BGP, DNS, HTTP, HTTPS,
3	A6-B6-94-1E-07-FE	BGP, DHCP, DNS, HTTPS, TCP,
4	BB-0B-0C-1D-24-F4	IMAP, POP, SNP, TCP,
5	E4-00-CE-46-3F-26	DNS, IMAP,

2) se plantea la siguiente solución

```python
equipoB = [3, 1, 7, 8]
equipoA = [2, 10, 5, 4, 8]


def counts(equipoA, equipoB):

    resultado=[]

    for valor_partidoB in  equipoB:

        gol_menor_partido=0

        for valor_partidoA in equipoA:

            if  valor_partidoA <= valor_partidoB:

                gol_menor_partido = gol_menor_partido+1

        resultado.append(gol_menor_partido)

    return resultado

print(counts(equipoA,equipoB))
```

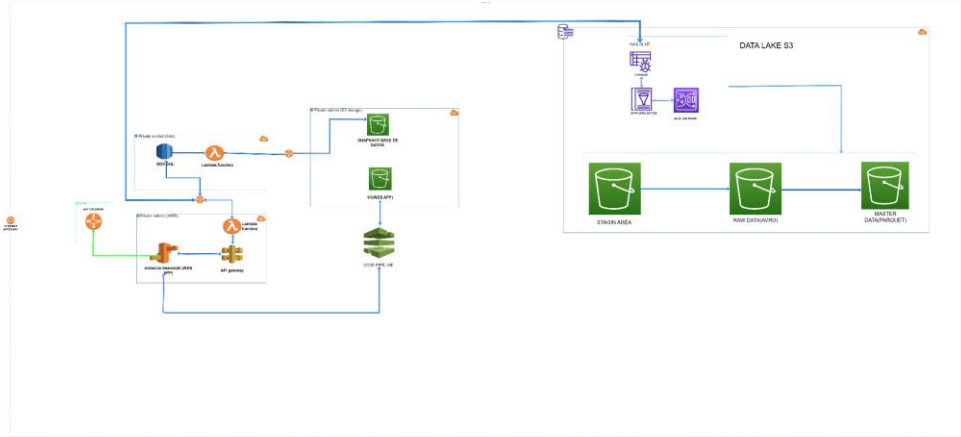
Este código cuenta con prueba unitaria:

```python
import unittest

import app

class TestCounts(unittest.TestCase):

  def test_counts_with_equal_lists(self):
    teamA = [2, 10, 5, 4, 8]
    teamB = [2, 10, 5, 4, 8]
    expected_counts = [1, 5, 3, 2, 4]
    actual_counts = app.counts(teamA, teamB)
    self.assertEqual(expected_counts, actual_counts)

  def test_counts_with_different_lists(self):
    teamA = [2, 10, 5, 4, 8]
    teamB = [3, 1, 7, 8]
    expected_counts = [1, 0, 3, 4]
    actual_counts = app.counts(teamA, teamB)
    self.assertEqual(expected_counts, actual_counts)

if __name__ == "__main__":
  unittest.main()
```

La cual termina ok para ambos casos

```
[1, 5, 3, 2, 4]
..
----------------------------------------------------------------------
Ran 2 tests in 0.000s

OK
```

3) se plantea el siguiente MSA para las necesidades planteadas por el punto 3


DATA LAKE S3
STAGIN AREA
RAW DATA(AVRO)
MASTER DATA(PARQUET)
API gateway