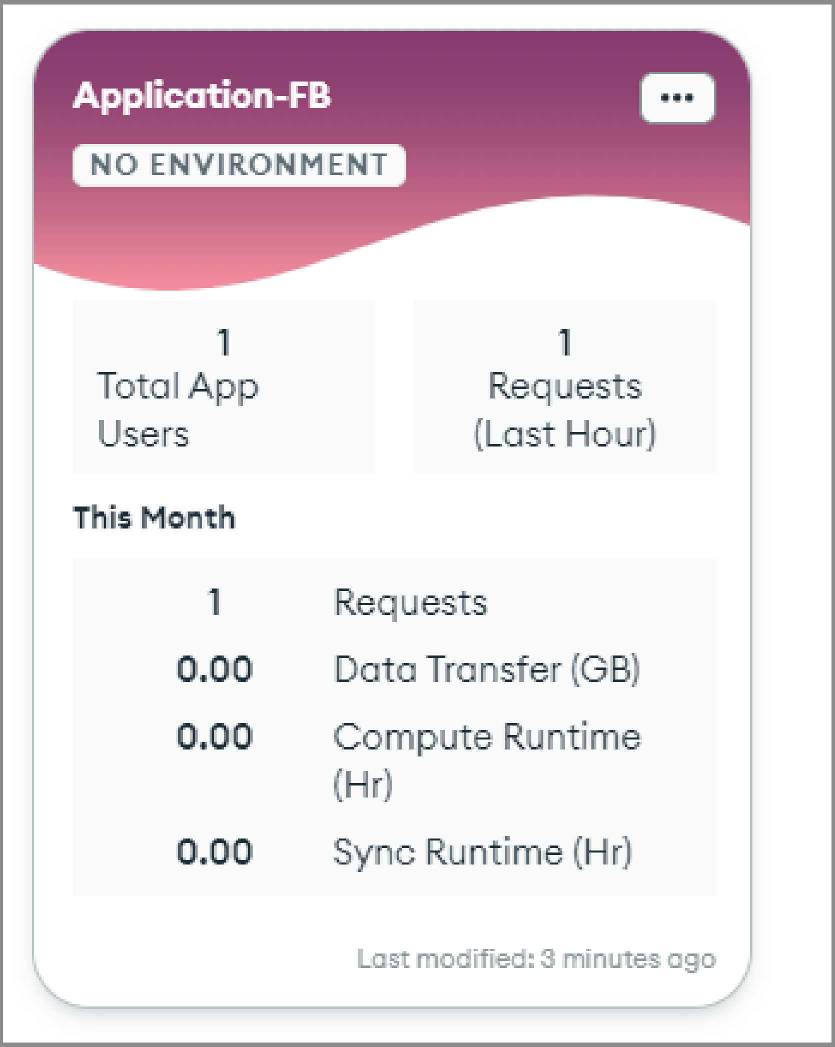
- **Create a new app**



Application-FB

NO ENVIRONMENT

1 Total App Users

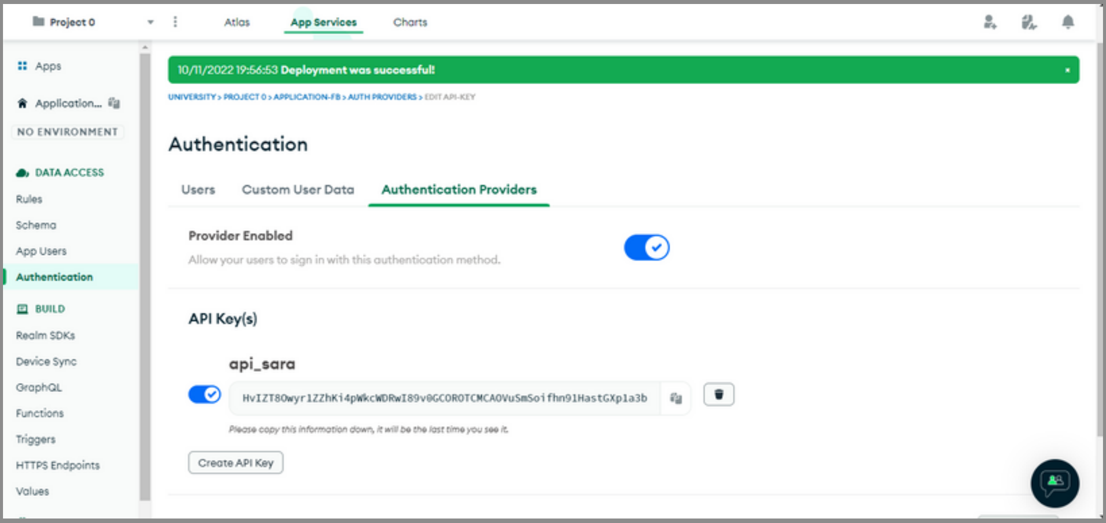
1 Requests (Last Hour)

This Month

1	Requests
0.00	Data Transfer (GB)
0.00	Compute Runtime (Hr)
0.00	Sync Runtime (Hr)

Last modified: 3 minutes ago

- **Enable Authentication provider (API)**
- **Create an API key**



10/11/2022 19:56:53 **Deployment was successful!**

UNIVERSITY > PROJECT 0 > APPLICATION-FB > AUTH PROVIDERS > EDIT API-KEY

Authentication

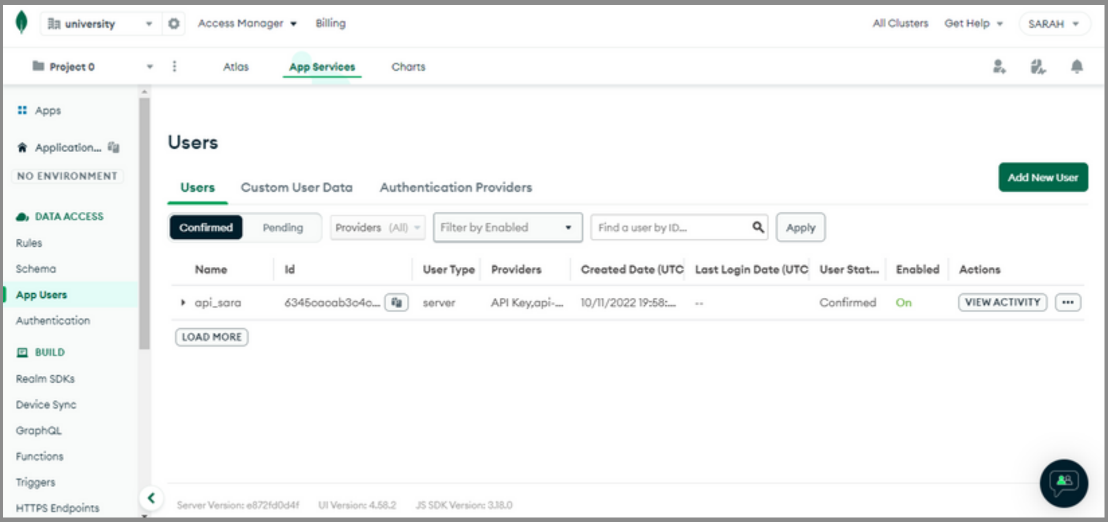
Provider Enabled ☑
Allow your users to sign in with this authentication method.

API Key(s)

api_sara ☑

```
HvIZT8Owyr1ZZhKi4pWkcWDRwI89v0GCOROTCMCAOVuSmSoifhn91HastGXp1a3b
```

Please copy this information down, it will be the last time you see it.

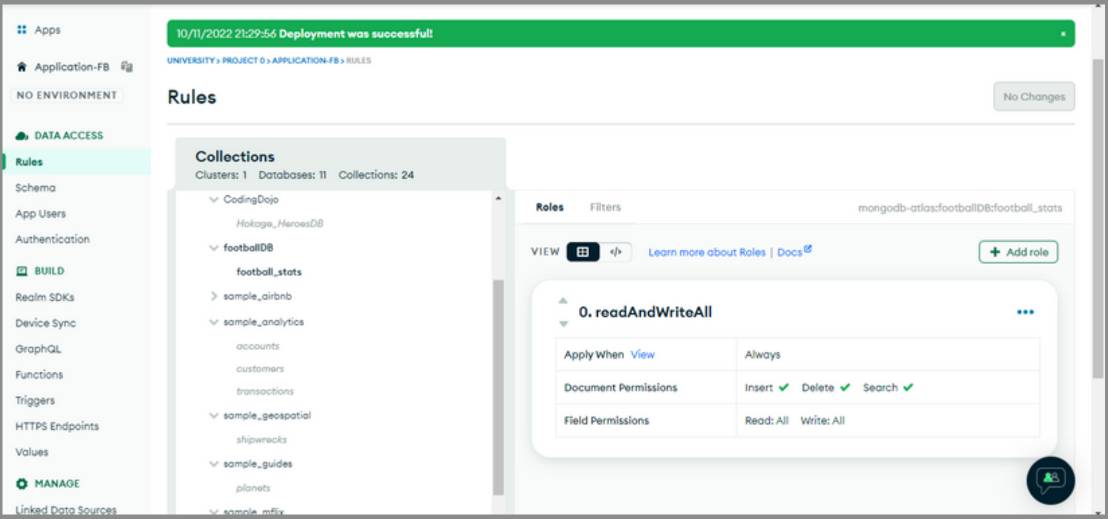


Users

Name	Id	User Type	Providers	Created Date (UTC	Last Login Date (UTC	User Stat...	Enabled	Actions
api_sara	6345aaab3c4c...	server	API Key,api-...	10/11/2022 19:58:...	--	Confirmed	On	VIEW ACTIVITY ...

Server Version: e872fd0d4f UI Version: 4.58.2 JS SDK Version: 3.18.0

- **Create a read/write rule on the chosen dataset**



10/11/2022 21:29:56 **Deployment was successful!**

UNIVERSITY > PROJECT 0 > APPLICATION-FB > RULES

Rules

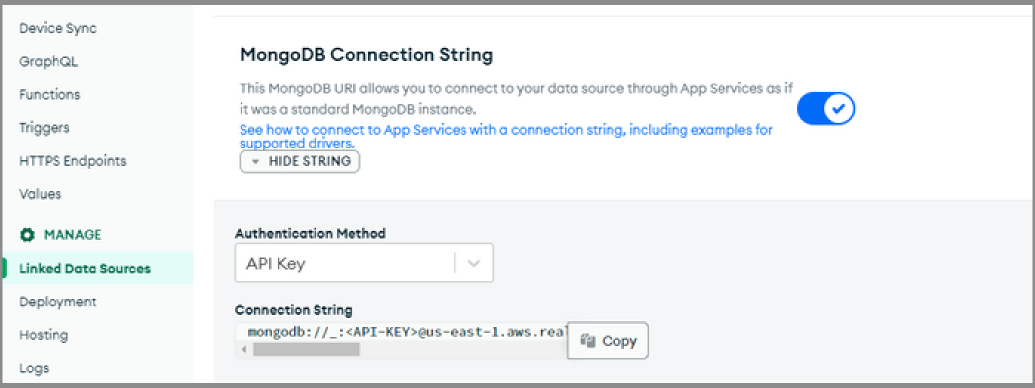
Collections
Clusters: 1 Databases: 11 Collections: 24

mongodb-atlas:footballDB:football_stats

0. readAndWriteAll

Apply When View	Always
Document Permissions	Insert ✓ Delete ✓ Search ✓
Field Permissions	Read: All Write: All

- **From the “Link Data Sources tab” activate the MongoDB connection string with the authentication method “API key".**



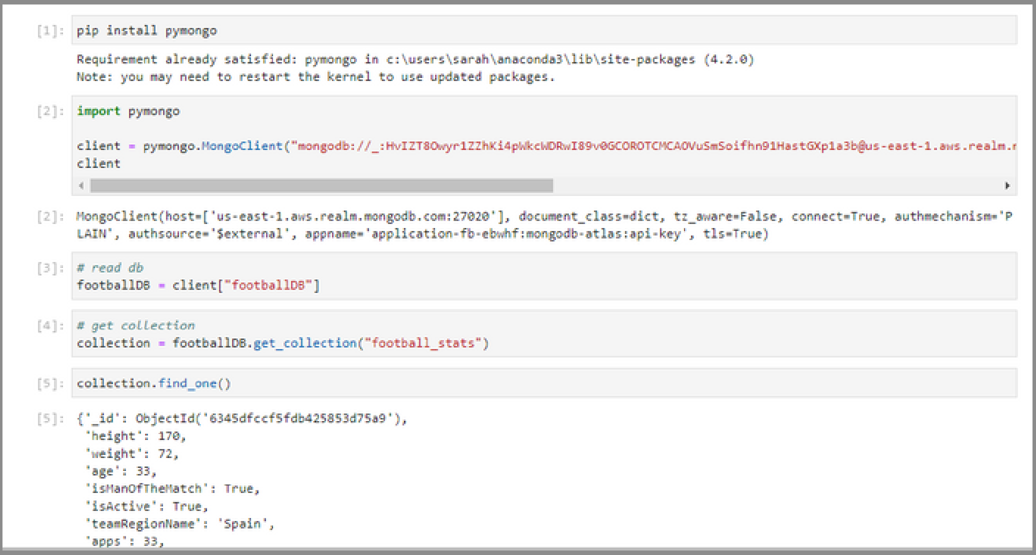
MongoDB Connection String

This MongoDB URI allows you to connect to your data source through App Services as if it was a standard MongoDB instance.

See how to connect to App Services with a connection string, including examples for supported drivers.

Authentication Method: API Key

Connection String: `mongodb://_:<API-KEY>@us-east-1.aws.rea`



```
[1]: pip install pymongo

     Requirement already satisfied: pymongo in c:\users\sarah\anaconda3\lib\site-packages (4.2.0)
     Note: you may need to restart the kernel to use updated packages.

[2]: import pymongo

     client = pymongo.MongoClient("mongodb://_:HvIZT8Owyr1ZZhKi4pWkcWDRwI89v0GCOROTCMCAOVuSmSoifhn91HastGXp1a3b@us-east-1.aws.realm.r
     client

[2]: MongoClient(host=['us-east-1.aws.realm.mongodb.com:27020'], document_class=dict, tz_aware=False, connect=True, authmechanism='P
     LAIN', authsource='$external', appname='application-fb-ebwhf:mongodb-atlas:api-key', tls=True)

[3]: # read db
     footballDB = client["footballDB"]

[4]: # get collection
     collection = footballDB.get_collection("football_stats")

[5]: collection.find_one()

[5]: {'_id': ObjectId('6345dfccf5fdb425853d75a9'),
      'height': 170,
      'weight': 72,
      'age': 33,
      'isManOfTheMatch': True,
      'isActive': True,
      'teamRegionName': 'Spain',
      'apps': 33,
```

FUNCTIONS

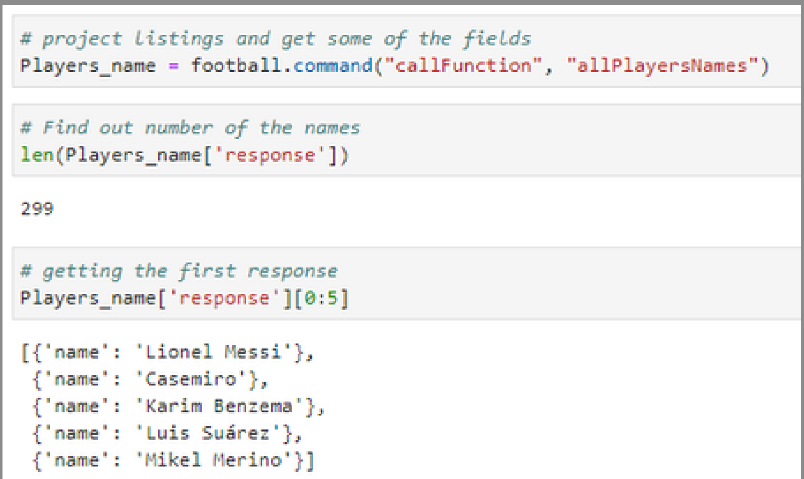
- Create a function that returns all players’ names from the dataset.

allPlayersNames

```
exports = function(){

   var collection = context.services.get("mongodb-atlas").db("footballDB").collection("football_stats");
   return collection.find({ }, { name: 1, _id:0 })

};
```



```
# project listings and get some of the fields
Players_name = football.command("callFunction", "allPlayersNames")
```

```
# Find out number of the names
len(Players_name['response'])
```

```
299
```

```
# getting the first response
Players_name['response'][0:5]
```

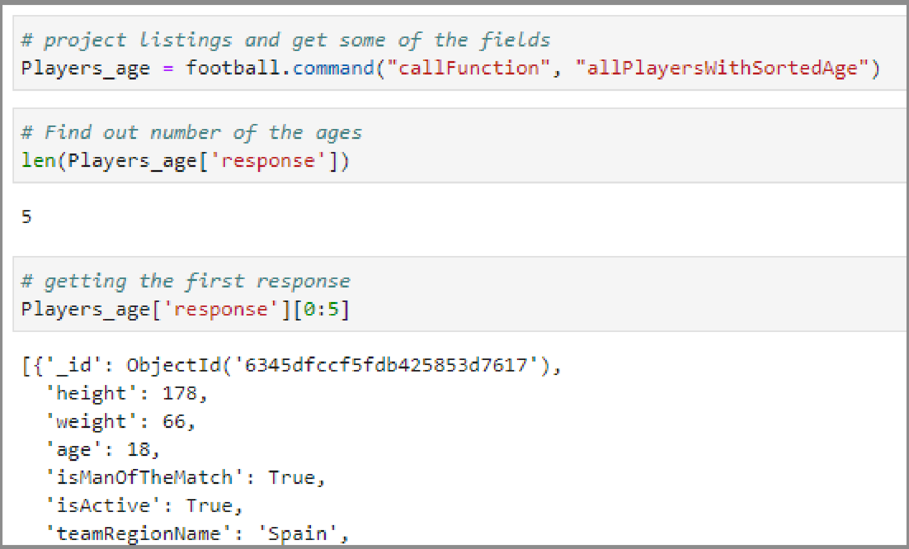
```
[{'name': 'Lionel Messi'},
 {'name': 'Casemiro'},
 {'name': 'Karim Benzema'},
 {'name': 'Luis Suárez'},
 {'name': 'Mikel Merino'}]
```

- Create a function that returns a sorted list of all players who are 20 years old or younger. (sort has to be from youngest to oldest)

allPlayersWithSortedAge

```
exports = function(){

   var collection = context.services.get("mongodb-atlas").db("footballDB").collection("football_stats");
   return collection.aggregate([{ "$match": {age: {"$lte":20} } }, {"$sort": {age: 1}} ])
};
```



```
# project listings and get some of the fields
Players_age = football.command("callFunction", "allPlayersWithSortedAge")
```

```
# Find out number of the ages
len(Players_age['response'])
```

```
5
```

```
# getting the first response
Players_age['response'][0:5]
```

```
[{'_id': ObjectId('6345dfccf5fdb425853d7617'),
  'height': 178,
  'weight': 66,
  'age': 18,
  'isManOfTheMatch': True,
  'isActive': True,
  'teamRegionName': 'Spain',
```

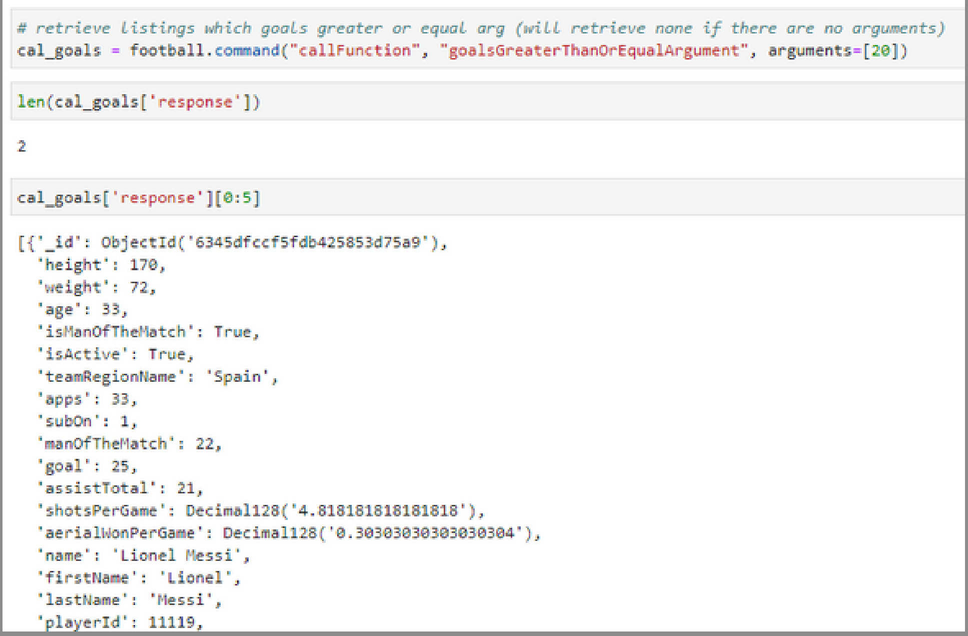
- Create a function that takes one argument (integer), and returns players' names, their team name, and the number of goals they scored. The number of goals is greater than or equal to the integer passed to the argument.

goalsGreaterThanOrEqualArgument

```
exports = function(arg){

    var collection = context.services.get("mongodb-
atlas").db("footballDB").collection("football_stats");
    return collection.find({ goal: {"$gte": arg}  })

};
```



```
# retrieve listings which goals greater or equal arg (will retrieve none if there are no arguments)
cal_goals = football.command("callFunction", "goalsGreaterThanOrEqualArgument", arguments=[20])
```

```
len(cal_goals['response'])
```

```
2
```

```
cal_goals['response'][0:5]
```

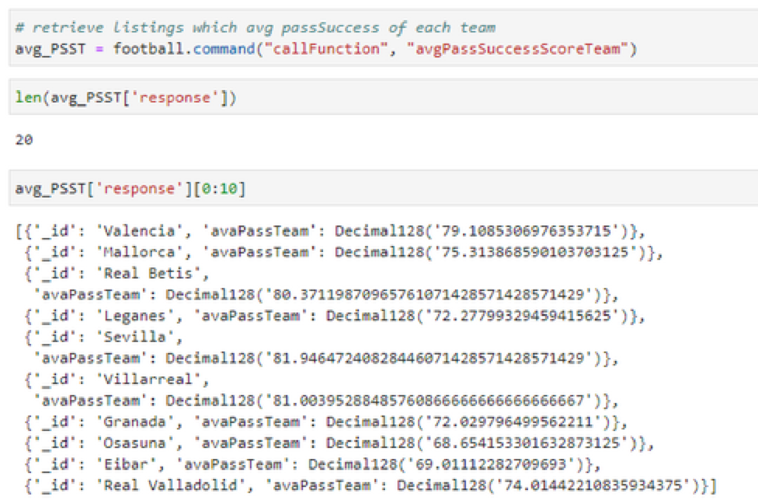
```
[{'_id': ObjectId('6345dfccf5fdb425853d75a9'),
  'height': 170,
  'weight': 72,
  'age': 33,
  'isManOfTheMatch': True,
  'isActive': True,
  'teamRegionName': 'Spain',
  'apps': 33,
  'subOn': 1,
  'manOfTheMatch': 22,
  'goal': 25,
  'assistTotal': 21,
  'shotsPerGame': Decimal128('4.818181818181818'),
  'aerialWonPerGame': Decimal128('0.30303030303030304'),
  'name': 'Lionel Messi',
  'firstName': 'Lionel',
  'lastName': 'Messi',
  'playerId': 11119,
```

- Create a function that returns an average pass success score for each team. Rename field “_id” to “teamName” and round pass success average score to 2 decimal places. (Hint: Use $project keyword)

avgPassSuccessScoreTeam

```
exports = function(){

   var collection = context.services.get("mongodb-
atlas").db("footballDB").collection("football_stats");
   return collection.aggregate([{ "$group": { _id:
"$teamName", avaPassTeam : {"$avg":
"$passSuccess"} } }])
};
```



```
# retrieve listings which avg passSuccess of each team
avg_PSST = football.command("callFunction", "avgPassSuccessScoreTeam")
```

```
len(avg_PSST['response'])
```

```
20
```

```
avg_PSST['response'][0:10]
```

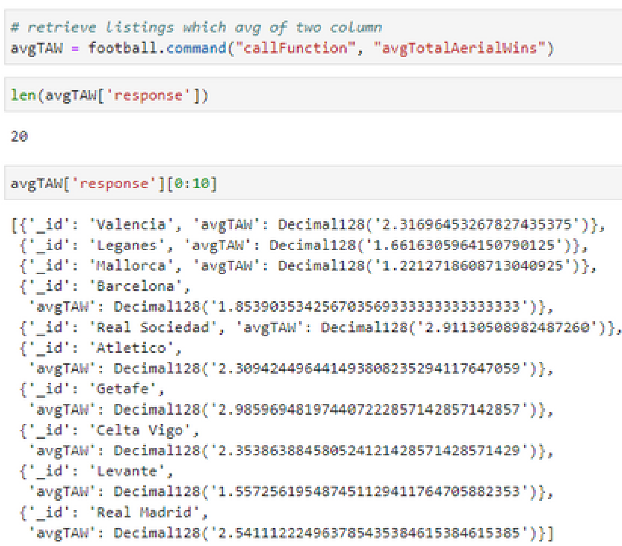
```
[{'_id': 'Valencia', 'avaPassTeam': Decimal128('79.1085306976353715')},
 {'_id': 'Mallorca', 'avaPassTeam': Decimal128('75.313868590103703125')},
 {'_id': 'Real Betis',
  'avaPassTeam': Decimal128('80.3711987096576107142857142857l429')},
 {'_id': 'Leganes', 'avaPassTeam': Decimal128('72.27799329459415625')},
 {'_id': 'Sevilla',
  'avaPassTeam': Decimal128('81.9464724082844607142857142857l429')},
 {'_id': 'Villarreal',
  'avaPassTeam': Decimal128('81.0039528848576086666666666666667')},
 {'_id': 'Granada', 'avaPassTeam': Decimal128('72.029796499562211')},
 {'_id': 'Osasuna', 'avaPassTeam': Decimal128('68.654153301632873125')},
 {'_id': 'Eibar', 'avaPassTeam': Decimal128('69.01112282709693')},
 {'_id': 'Real Valladolid', 'avaPassTeam': Decimal128('74.01442210835934375')}]
```

- **Create a function that returns total assists and aerial wins for each team (fields “assistTotal” and “aerialWonPerGame”). Rename field “_id” to “teamName” and round total aerial wins to 2 decimal places. (Hint: Use $proect keyword)**

avgTotalAerialWins

```
exports = function(){

   var collection = context.services.get("mongodb-atlas").db("footballDB").collection("football_stats");
   return collection.aggregate([{ "$group": { _id: "$teamName",  avgTAW: {"$avg": {"$multiply": ["$assistTotal","$aerialWonPerGame"]}} } }])
};
```



```
# retrieve listings which avg of two column
avgTAW = football.command("callFunction", "avgTotalAerialWins")
```

```
len(avgTAW['response'])
```

```
20
```

```
avgTAW['response'][0:10]
```

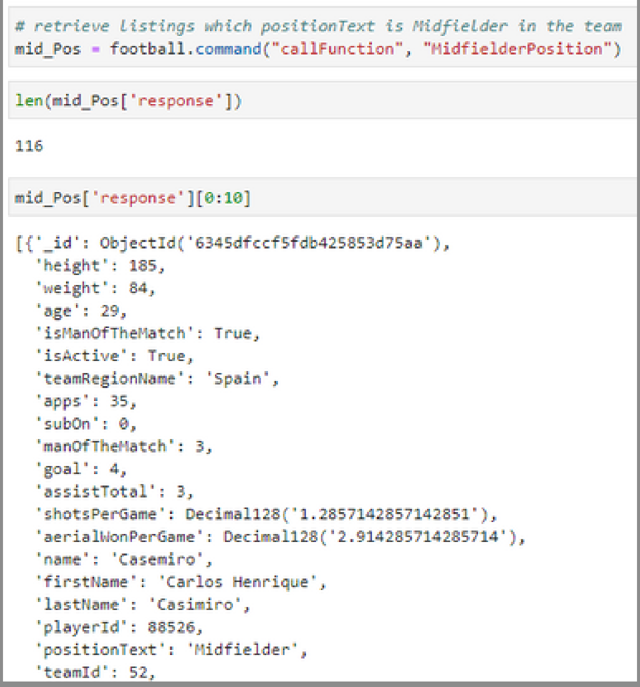
```
[{'_id': 'Valencia', 'avgTAW': Decimal128('2.31696453267827435375')},
 {'_id': 'Leganes', 'avgTAW': Decimal128('1.6616305964150790125')},
 {'_id': 'Mallorca', 'avgTAW': Decimal128('1.2212718608713040925')},
 {'_id': 'Barcelona',
  'avgTAW': Decimal128('1.853903534256703569333333333333333')},
 {'_id': 'Real Sociedad', 'avgTAW': Decimal128('2.91130508982487260')},
 {'_id': 'Atletico',
  'avgTAW': Decimal128('2.309424496441493808235294117647059')},
 {'_id': 'Getafe',
  'avgTAW': Decimal128('2.985969481974407222857142857142857')},
 {'_id': 'Celta Vigo',
  'avgTAW': Decimal128('2.353863884580524121428571428571429')},
 {'_id': 'Levante',
  'avgTAW': Decimal128('1.557256195487451129411764705882353')},
 {'_id': 'Real Madrid',
  'avgTAW': Decimal128('2.541112224963785435384615384615385')}]
```

- Create one function of your choice.

MidfielderPosition

```
exports = function(){

   var collection = context.services.get("mongodb-atlas").db("footballDB").collection("football_stats");
   var result = collection.find({positionText: "Midfielder"})
  return result
};
```



```
# retrieve listings which positionText is Midfielder in the team
mid_Pos = football.command("callFunction", "MidfielderPosition")
```

```
len(mid_Pos['response'])
```

```
116
```

```
mid_Pos['response'][0:10]
```

```
[{'_id': ObjectId('6345dfccf5fdb425853d75aa'),
  'height': 185,
  'weight': 84,
  'age': 29,
  'isManOfTheMatch': True,
  'isActive': True,
  'teamRegionName': 'Spain',
  'apps': 35,
  'subOn': 0,
  'manOfTheMatch': 3,
  'goal': 4,
  'assistTotal': 3,
  'shotsPerGame': Decimal128('1.2857142857142851'),
  'aerialWonPerGame': Decimal128('2.914285714285714'),
  'name': 'Casemiro',
  'firstName': 'Carlos Henrique',
  'lastName': 'Casimiro',
  'playerId': 88526,
  'positionText': 'Midfielder',
  'teamId': 52,
```