

Among Us - MongoDB Analysis

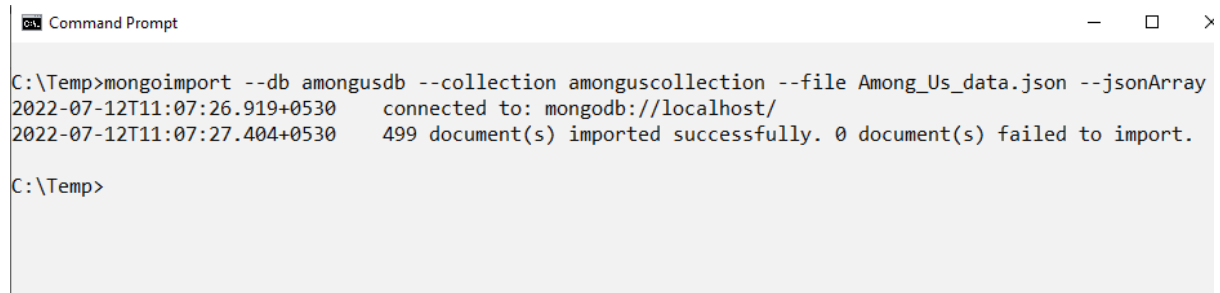
Saravanan Thiyagarajan

1.1: Read in the data using the mongoimport command.

Query: (Text)

mongoimport --db amongusdb --collection amonguscollection --file Among_Us_data.json --jsonArray

Screenshot:



```
cmd Command Prompt
C:\Temp>mongoimport --db amongusdb --collection amonguscollection --file Among_Us_data.json --jsonArray
2022-07-12T11:07:26.919+0530    connected to: mongodb://localhost/
2022-07-12T11:07:27.404+0530    499 document(s) imported successfully. 0 document(s) failed to import.

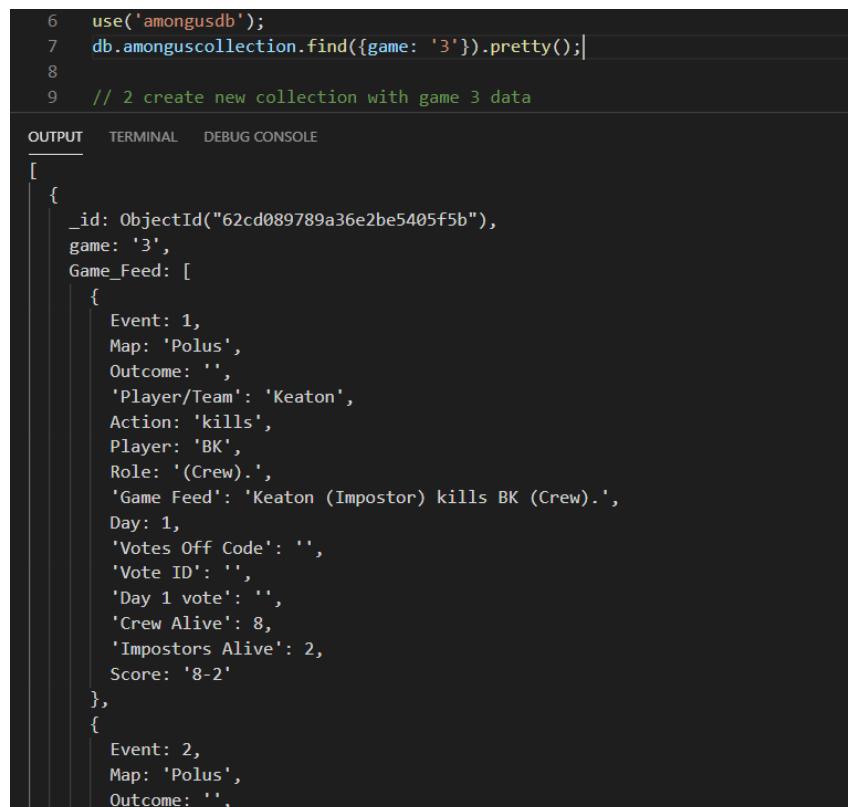
C:\Temp>
```

1.2: Display data for the match with “game” = “3”

Query:

```
db.amonguscollection.find({game: '3'}).pretty();
```

Screenshot:



```
6 use('amongusdb');
7 db.amonguscollection.find({game: '3'}).pretty();|
8
9 // 2 create new collection with game 3 data

OUTPUT  TERMINAL  DEBUG CONSOLE
[
  {
    _id: ObjectId("62cd089789a36e2be5405f5b"),
    game: '3',
    Game_Feed: [
      {
        Event: 1,
        Map: 'Polus',
        Outcome: '',
        'Player/Team': 'Keaton',
        Action: 'kills',
        Player: 'BK',
        Role: '(Crew).',
        'Game Feed': 'Keaton (Impostor) kills BK (Crew).',
        Day: 1,
        'Votes Off Code': '',
        'Vote ID': '',
        'Day 1 vote': '',
        'Crew Alive': 8,
        'Impostors Alive': 2,
        Score: '8-2'
      },
      {
        Event: 2,
        Map: 'Polus',
        Outcome: ''
      }
    ]
  }
]
```

2: Explore Game 3. Create a new collection with only the document relating to game 3

Query:

```
db.createCollection('game3');
var g3 = db.amonguscollection.find({game: '3'});
db.game3.insert(g3.toArray());
```

Screenshot:

```
10 use('amongusdb');
11 db.createCollection('game3');
12 var g3 = db.amonguscollection.find({game: '3'});
13 db.game3.insert(g3.toArray());
14
```

OUTPUT TERMINAL DEBUG CONSOLE

```
{
  acknowledged: 1,
  insertedIds: {
    '0': ObjectId("62cd089789a36e2be5405f5b")
  }
}
```

2.1: Display the Game Feed data for the game in the new collection.

Query:

```
db.game3.aggregate([
  {$unwind: '$Game_Feed'},
  {$project: {'Game_Feed': 1}}]);
```

Screenshot:

```
17 db.game3.aggregate([
18   {$unwind: '$Game_Feed'},
19   {$project: {'Game_Feed': 1}}]);
```

OUTPUT TERMINAL DEBUG CONSOLE

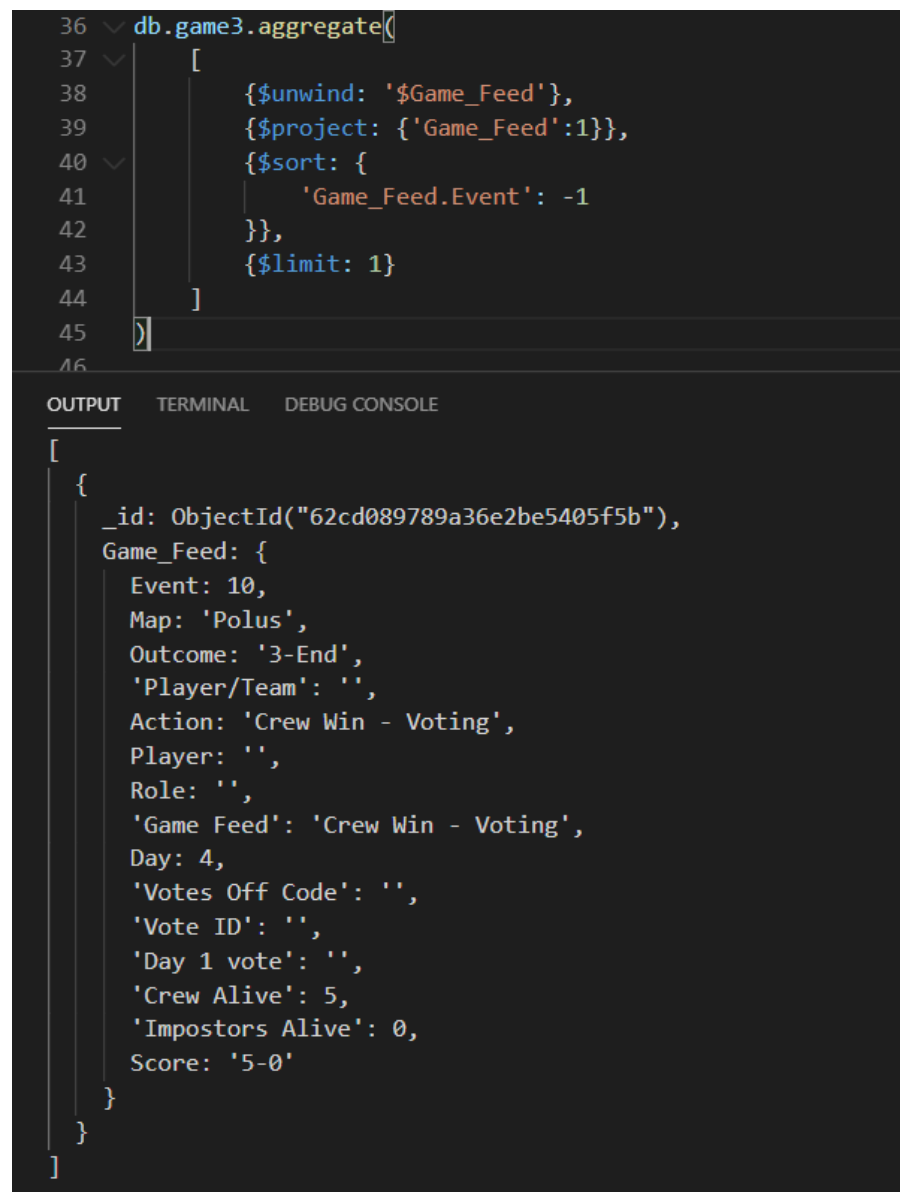
```
[
  {
    _id: ObjectId("62cd089789a36e2be5405f5b"),
    Game_Feed: {
      Event: 1,
      Map: 'Polus',
      Outcome: '',
      'Player/Team': 'Keaton',
      Action: 'kills',
      Player: 'BK',
      Role: '(Crew).',
      'Game Feed': 'Keaton (Impostor) kills BK (Crew).',
      Day: 1,
      'Votes Off Code': '',
      'Vote ID': '',
      'Day 1 vote': '',
      'Crew Alive': 8,
      'Impostors Alive': 2,
      Score: '8-2'
    }
  },
  {
    _id: ObjectId("62cd089789a36e2be5405f5b"),
    Game_Feed: {
      Event: 2,
```

2.2: Display the last event in game 3.

Query:

```
db.game3.aggregate([
  {
    $unwind: '$Game_Feed',
    $project: {'Game_Feed':1}},
  {$sort: {
    'Game_Feed.Event': -1
  }},
  {$limit: 1}
])
```

Screenshot:



The screenshot shows a MongoDB query being executed in a code editor. The query is an aggregation pipeline that unwinds the 'Game_Feed' array, sorts the events in descending order of their 'Event' value, and limits the result to one document. The output of the query is displayed in the 'OUTPUT' tab, showing a single document with the details of the last event in game 3.

```
36 db.game3.aggregate([
37   [
38     {$unwind: '$Game_Feed'},
39     {$project: {'Game_Feed':1}},
40     {$sort: {
41       'Game_Feed.Event': -1
42     }},
43     {$limit: 1}
44   ]
45 ])
```

OUTPUT **TERMINAL** **DEBUG CONSOLE**

```
[
  {
    _id: ObjectId("62cd089789a36e2be5405f5b"),
    Game_Feed: {
      Event: 10,
      Map: 'Polus',
      Outcome: '3-End',
      'Player/Team': '',
      Action: 'Crew Win - Voting',
      Player: '',
      Role: '',
      'Game Feed': 'Crew Win - Voting',
      Day: 4,
      'Votes Off Code': '',
      'Vote ID': '',
      'Day 1 vote': '',
      'Crew Alive': 5,
      'Impostors Alive': 0,
      Score: '5-0'
    }
  }
]
```

2.3: Who won game 3, imposters or crew?

Query:

```
db.game3.aggregate(  
  [  
    {$unwind: '$Game_Feed'},  
    {$sort: {  
      'Game_Feed.Event': -1  
    }},  
    {$limit: 1},  
    {$project: {'Game_Feed.Game Feed':1}}  
  ]  
)
```

Screenshot:

```
58 db.game3.aggregate([  
59   [  
60     {$unwind: '$Game_Feed'},  
61     {$sort: {  
62       'Game_Feed.Event': -1  
63     }},  
64     {$limit: 1},  
65     {$project: {'Game_Feed.Game Feed':1}}  
66   ]  
67 ]
```

OUTPUT

TERMINAL

DEBUG CONSOLE

```
[  
  {  
    _id: ObjectId("62cd089789a36e2be5405f5b"),  
    Game_Feed: {  
      'Game Feed': 'Crew Win - Voting'  
    }  
  }  
]
```

2.4: Who picked the black color in game 3? Was that player crew or imposter?

Query:

```
db.game3.aggregate([
  {
    $unwind: '$player_data',
    {
      $project: {
        'player_data.name':1,
        'player_data.Role':1,
        'player_data.Color':1
      }
    },
    {
      $match: {
        'player_data.Color': 'Black'
      }
    }
  }
]);
```

Screenshot:

The screenshot shows a MongoDB query in a code editor and its output in a terminal window. The query is an aggregation pipeline that unwinds the 'player_data' array and filters for players with the color 'Black'.

```
72 db.game3.aggregate(
73   [
74     { $unwind: '$player_data' },
75     {
76       $project: {
77         'player_data.name':1,
78         'player_data.Role':1,
79         'player_data.Color':1
80       }
81     },
82     {
83       $match: {
84         'player_data.Color': 'Black'
85       }
86     }
87   ]
88 );
```

The output shows a single document with the following structure:

```
{
  _id: ObjectId("62cd089789a36e2be5405f5b"),
  player_data: {
    name: 'Keaton',
    Role: '(Impostor)',
    Color: 'Black'
  }
}
```

2.5: How many voting events happened in game 3?

Query:

```
db.game3.aggregate([
  { $unwind: '$voting_data' },
  { $project: {
    'voting_data': 1
  } },
  { $group: {
    _id: '$voting_data.Vote_Event',
    Votes_count: {
      $sum: 1
    }
  } },
  { $group: {
    _id: null,
    Total_Voting_Events: {
      $sum: 1
    }
  } }
]);
```

Screenshot:

```
91  ▾ db.game3.aggregate(
92  ▾  [ { $unwind: '$voting_data'},
93  ▾    { $project: {
94      |     'voting_data': 1
95      |   }
96      | },
97  ▾    { $group: {
98      |     _id: '$voting_data.Vote_Event',
99  ▾      |     Votes_count: {
100     |       $sum: 1
101     |     }
102     |   }
103     | },
104  ▾    { $group:{
105     |     _id: null,
106  ▾     |     Total_Voting_Events: {}
107     |     $sum: 1
108     |   }
109     | }
110     | ]]);
```

OUTPUT TERMINAL DEBUG CONSOLE

```
[
  {
    _id: null,
    Total_Voting_Events: 3
  }
]
```

2.6:

Database re-design suggestions and justifications as follows:

Suggestion 1:

Instead of embedding the Game_Feed, player_data and voting_data in the same collection, they can be separated into 3 different collections and the references can be added into the game collection.

Justification:

This helps organize the data better and the game collection will look easier to read. It also helps in better data analytics as the query can be run for the respective collection instead of unwinding every time. Also, some of the common fields and overall status can be maintained in the game collection instead of looking into the Event data every time.

Suggestion 2:

Instead of repeating the Map in all the events, it can be common and single field for a game.

Justification:

Avoid repeating/duplicating the common data. 'Game_Feed.Map' is repeated field and can be represented as given in the below example.

Suggestion 3:

Add the winner of the game (Crew or Imposter) as a new field in the game collection.

Justification:

We can refer to the Game_Winner field to find the winner of the game instead of checking all the events and looking into the Outcome and 'Game Feed' fields.

Example document:

```
{
  _id: ObjectId("62cd089789a36e2be5405f5b"),
  game: '3',
  Map: 'Polus',
  Game_Winner: 'Crew',
  Game_Feed: ['G3-E1', 'G3-E2', ...],
  Player_Data: ['G3-P1', 'G3-P2', ...],
  Voting_Data: ['G3-V1', 'G3-V2', ...]
}
```

3.1: How many events in total do you have data for, in this collection (across all games)?

Query:

```
db.amonguscollection.aggregate([
  {
    $unwind: '$Game_Feed'
  },
  {
    $group: {
      _id: null,
      Total_Events_in_Collection: {
        $sum: 1
      }
    }
  }
]);
```

Screenshot:

The screenshot shows a MongoDB query in a code editor and its output in a terminal window. The query is an aggregation pipeline that unwinds the 'Game_Feed' array and groups the results to calculate the total number of events across all games.

```
114 db.amonguscollection.aggregate([
115   [
116     {
117       $unwind: '$Game_Feed'
118     },
119     {
120       $group: {
121         _id: null,
122         Total_Events_in_Collection: {
123           $sum: 1
124         }
125       }
126     }
127   ]
128 ]]);
```

The output window shows the result of the query, which is an array containing one document with the total number of events.

```
[
  {
    _id: null,
    Total_Events_in_Collection: 5889
  }
]
```


3.2: How many matches did the crew win versus how many matches did the impostors win?

Query:

```
db.amonguscollection.aggregate([
  { $unwind: '$Game_Feed' },
  { $project: {
    'Game_Feed.Outcome': 1,
    'Game_Feed.Game Feed': 1 }},
  { $match: {
    'Game_Feed.Outcome': { $ne: '' } }},
  { $group: {
    _id: '$Game_Feed.Game Feed',
    Count: { $sum: 1 } }},
  { $match: {
    $or: [ { '_id': { $regex: '^Impostor Win' } },
    { '_id': { $regex: '^Crew Win' } } ] }},
]);
```

Screenshot:

```
204 db.amonguscollection.aggregate(
205   [ { $unwind: '$Game_Feed',
206     { $project: {
207       'Game_Feed.Outcome': 1,
208       'Game_Feed.Game Feed': 1 }},
209     { $match: {
210       'Game_Feed.Outcome': { $ne: '' } }},
211     { $group: {
212       _id: '$Game_Feed.Game Feed',
213       Count: { $sum: 1 } }},
214     { $match: {
215       $or: [ { '_id': { $regex: '^Impostor Win' } },
216       { '_id': { $regex: '^Crew Win' } } ] }},
217   ]);
```

OUTPUT TERMINAL DEBUG CONSOLE

```
[
  {
    _id: 'Crew Win - Tasks',
    Count: 44
  },
  {
    _id: 'Crew Win - Voting',
    Count: 279
  },
  {
    _id: 'Impostor Win - Kills',
    Count: 174
  },
  {
    _id: 'Impostor Win - Sabotage',
    Count: 2
  }
]
```

3.3: Find out how many matches were played on each map.

Query:

```
db.amonguscollection.aggregate([
  {
    $unwind: '$Game_Feed',
    {
      $project: {
        'game': 1,
        'Game_Feed.Map': 1,
        'Game_Feed.Outcome': 1 }
    },
    {
      $match: {
        'Game_Feed.Outcome': {
          $ne: ''
        }
      }
    },
    {
      $group: {
        _id: '$Game_Feed.Map',
        Number_of_games_played: {
          $sum: 1
        }
      }
    }
  ]]);
```

Screenshot:

```
206 db.amonguscollection.aggregate(
207   [
208     {
209       $unwind: '$Game_Feed',
210       {
211         $project: {
212           'game': 1,
213           'Game_Feed.Map': 1,
214           'Game_Feed.Outcome': 1 }
215       },
216       {
217         $match: {
218           'Game_Feed.Outcome': {
219             $ne: ''
220           }
221         }
222       },
223       {
224         $group: {
225           _id: '$Game_Feed.Map',
226           Number_of_games_played: {
227             $sum: 1
228           }
229         }
230       }
231     ]
232   ]]);
```

OUTPUT TERMINAL DEBUG CONSOLE

```
[
  {
    _id: 'Polus',
    Number_of_games_played: 409
  },
  {
    _id: 'The Skeld',
    Number_of_games_played: 88
  },
  {
    _id: 'MIRA HQ',
    Number_of_games_played: 7
  }
]
```

3.4: How many times in total across all games did the crew skip a vote?

Query:

```
db.amonguscollection.aggregate([
  { $unwind: '$Game_Feed' },
  { $project: { 'Game_Feed' : 1 } },
  { $match: {
    $and: [
      { 'Game_Feed.Action': 'skips voting.' },
      { 'Game_Feed.Player/Team': 'Crew' } ] }
  },
  { $group: {
    _id: null,
    Total_times_Crew_skipped_Vote: { $sum: 1 } }
  }
]);
```

Screenshot:

```
228 ✓ db.amonguscollection.aggregate(
229 ✓   [ { $unwind: '$Game_Feed'},
230     { $project: {'Game_Feed' : 1}},
231     { $match: {
232       $and: [
233         {'Game_Feed.Action': 'skips voting.'},
234         {'Game_Feed.Player/Team': 'Crew'} ] }
235     },
236     { $group: {
237       _id: null,
238       Total_times_Crew_skipped_Vote: { $sum: 1 } }
239     }
240   ] );
```

OUTPUT TERMINAL DEBUG CONSOLE

```
[
  {
    _id: null,
    Total_times_Crew_skipped_Vote: 693
  }
]
```

3.5: How many times in total across all matches does the crew vote against imposters?

Query:

```
db.amonguscollection.aggregate([
  {
    $unwind: '$Game_Feed',
    $project: {
      'Game_Feed' : 1}},
  {
    $match: {
      $and: [
        {'Game_Feed.Action': 'votes off'},
        {'Game_Feed.Player/Team': 'Crew'}]]},
  {
    $group: {
      _id: null,
      Total_times_Crew_Votes_off_Impostors: {$sum: 1}}}
]);
```

Screenshot:

```
245  db.amonguscollection.aggregate(
246  [
247    { $unwind: '$Game_Feed'},
248    { $project: {
249      'Game_Feed' : 1}},
250    { $match: {
251      $and: [
252        {'Game_Feed.Action': 'votes off'},
253        {'Game_Feed.Player/Team': 'Crew'}]]},
254    { $group: {
255      _id: null,
256      Total_times_Crew_Votes_off_Impostors: {$sum: 1}}}
257  ]
258  );
```

OUTPUT TERMINAL DEBUG CONSOLE

```
[
  {
    _id: null,
    Total_times_Crew_Votes_off_Impostors: 896
  }
]
```

3.6: Is the game more or less hard for imposters

Answer: **Yes**, the game is harder for the imposters.

Justification: (based on the data analysis done before)

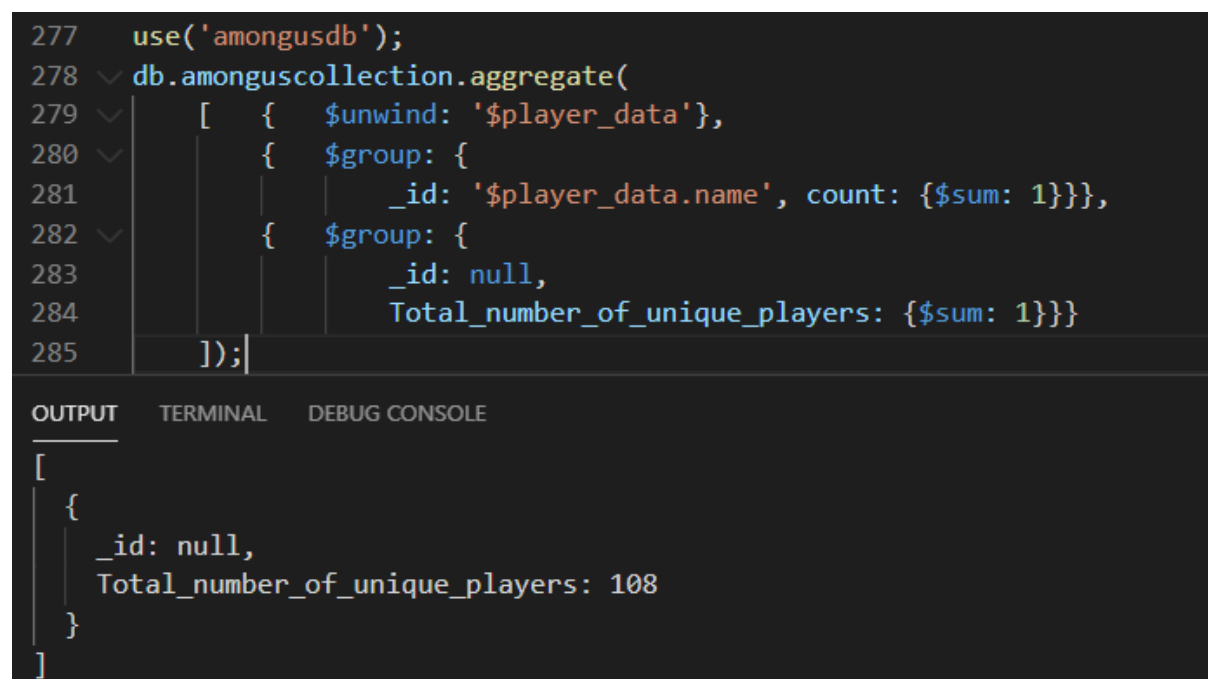
- Crew had the most chances of winning over Imposters. Crew – 65% vs Imposters – 35%
- Crew members were able to Vote off the Imposters most of the time instead of winning by completing the tasks. So, it looks like Voting off is easier.
- Most of the games were played in Polus map. Probably this is making easier for Crew members. Diversifying with different Map options might help Imposters improve their game.
- Imposters winning by Sabotage is almost close to zero. So, it looks like the Sabotage task is very hard and the Imposters need to work hard to kill the crew members. This could be one of the main reasons the imposters are getting caught and voted off by crew members.

4.1: Find the number of unique players in the data set.

Query:

```
db.amonguscollection.aggregate(  
  [  
    { $unwind: '$player_data' },  
    { $group: {  
      _id: '$player_data.name', count: {$sum: 1}} },  
    { $group: {  
      _id: null,  
      Total_number_of_unique_players: {$sum: 1}} }  
  ]  
);
```

Screenshot:



The screenshot shows a MongoDB query being executed in a terminal window. The query is an aggregation pipeline that first unwinds the 'player_data' array, then groups by 'player_data.name' to count each player, and finally groups by null to sum the counts and get the total number of unique players. The output shows a single document with the total count of 108 unique players.

```
277 use('amongusdb');  
278 db.amonguscollection.aggregate(  
279   [  
280     { $unwind: '$player_data' },  
281     { $group: {  
282       _id: '$player_data.name', count: {$sum: 1}} },  
283     { $group: {  
284       _id: null,  
285       Total_number_of_unique_players: {$sum: 1}} }  
286   ]  
);
```

OUTPUT TERMINAL DEBUG CONSOLE

```
[  
  {  
    _id: null,  
    Total_number_of_unique_players: 108  
  }  
]
```

4.2: Who is the best crew member? player who (as a crewmate) voted to remove the imposters the most number of times

Query:

```
db.amonguscollection.aggregate(  
  [  
    { $unwind: '$voting_data' },  
    { $project: {  
      'voting_data' : 1  
    } },  
    { $match: {  
      'voting_data.Vote' : {  
        $regex: '\\Impostor voted off'  
      } } },  
    { $group: {  
      _id: '$voting_data.name',  
      count: { $sum: 1 } } },  
    { $sort: { count: -1 } },  
    { $limit: 1 }  
  ] );
```

Screenshot:

```
318 db.amonguscollection.aggregate(  
319   [  
320     { $unwind: '$voting_data' },  
321     { $project: {  
322       'voting_data' : 1  
323     } },  
324     { $match: {  
325       'voting_data.Vote' : {  
326         $regex: '\\Impostor voted off'  
327       } } },  
328     { $group: {  
329       _id: '$voting_data.name',  
330       count: { $sum: 1 } } },  
331     { $sort: { count: -1 } },  
332     { $limit: 1 }  
333   ] );
```

OUTPUT	TERMINAL	DEBUG CONSOLE
[{ _id: 'BK', count: 178 }]		

4.3: Who is the worst crew member? player who (as a crewmate) voted to remove the other crew members the most number of times.

Query:

```
db.amonguscollection.aggregate([
  {
    $unwind: '$voting_data',
    {
      $project: {
        'voting_data' : 1
      }},
    {
      $match: {
        'voting_data.Vote' : {
          $regex: '\\Crew voted off'
        }},
    {
      $group: {
        _id: '$voting_data.name',
        count: {$sum: 1}},
    {
      $sort: {count: -1}},
    {
      $limit: 1}
]);
```

Screenshot:



The screenshot shows a MongoDB query being executed in a terminal window. The query is an aggregation pipeline that unwinds the 'voting_data' array, projects the 'voting_data' field, matches votes where the regex '\\Crew voted off' is found, groups by the voter's name, sorts by count in descending order, and limits the result to 1. The output shows a single document for 'Sam' with a count of 60.

```
338 db.amonguscollection.aggregate([
339   [
340     { $unwind: '$voting_data'},
341     { $project: {
342       'voting_data' : 1
343     }},
344     { $match: {
345       'voting_data.Vote' : {
346         $regex: '\\Crew voted off'
347       }},
348     { $group: {
349       _id: '$voting_data.name',
350       count: {$sum: 1}},
351     { $sort: {count: -1}},
352     { $limit: 1}
353   ]]);
```

OUTPUT **TERMINAL** **DEBUG CONSOLE**

```
[
  {
    _id: 'Sam',
    count: 60
  }
]
```

5:

New Statistics	Explanation
Chose to be a Crew member	Number of times a player chose to be a Crew member based on all the game data.
Chose to be an Imposter	Number of times a player chose to be an Imposter based on all the game data
Number of Kills	Number of kills the player has made as an imposter
Number of time player died	Number of times the player was killed by an imposter
Tasks completed by player	Number of tasks successfully completed by each player as a crew member
Number of Days game was played	How many number of days each game was played and which is the game that was played for most number of days
Who won the games completed just in Day 1	Did Crew or Imposters win the game most number of times when the game was over in Day 1 itself.