

Title: **Aggregation Pipeline**

Aim: **Applying Aggregation pipeline using stages**

Name: Krish Agarwal

Register Number: 21112016

Class: 4BSc DS

NoSQL Lab 8

10/05/23

Reusing previously imported data

```
employee> db.spotify_data.find()
{
  "_id": ObjectId("6423b5968bafc3f1b9991695"),
  "": 0,
  "Artist": "Gorillaz",
  "Track": "Feel Good Inc.",
  "Album": "Demon Days",
  "Album_type": "album",
  "Danceability": 0.82,
  "Energy": 0.71,
  "Key": 6,
  "Loudness": -6.68,
  "Speechiness": 0.18,
  "Acousticness": 0.01,
  "Instrumentalness": 0,
  "Liveness": 0.01,
  "Valence": 0.77,
  "Tempo": 138.56,
  "Duration_ms": 222640,
  "Total": {
    "$sum": [
      "$Danceability",
      "$Energy",
      "$Key",
      "$Loudness",
      "$Speechiness",
      "$Acousticness",
      "$Instrumentalness",
      "$Liveness",
      "$Valence",
      "$Tempo",
      "$Acousticness"
    ]
  }
},
}
```

\$match stage: The first stage to select only the matching documents from a collection. It is equivalent to the .find() method.

Syntax: `db.spotify_data.aggregate([{$match: {$and: [{Artist: 'Eminem'}, {Album: 'Recovery'}]}}])`

Retrieving the music data where the artist is 'Eminem' and the album is 'Recovery'.

```
employee> db.spotify_data.aggregate([{$match: {$and: [{Artist: 'Eminem'}, {Album: 'Recovery'}]}}])
{
  "_id": ObjectId("6423b5968bafc3f1b9991713"),
  "": 10,
  "Artist": "Eminem",
  "Track": "Love The Way You Lie",
  "Album": "Recovery",
  "Album_type": "album",
  "Danceability": 0.7,
  "Energy": 0.9,
  "Key": 10,
  "Loudness": -5.8,
  "Speechiness": 0.23,
  "Acousticness": 0.24,
  "Instrumentalness": 0,
  "Liveness": 0.52,
  "Valence": 0.64,
  "Tempo": 86.99,
  "Duration_ms": 203073,
  "Total": {
    "$sum": [
      "$Danceability",
      "$Energy",
      "$Key",
      "$Loudness",
      "$Speechiness",
      "$Acousticness",
      "$Instrumentalness",
      "$Liveness",
      "$Valence",
      "$Tempo",
      "$Acousticness"
    ]
  }
},
{
  "_id": ObjectId("6423b5968bafc3f1b9991716"),
  "": 10,
  "Artist": "Eminem",
  "Track": "Not Afraid",
  "Album": "Recovery",
  "Album_type": "album",
  "Danceability": 0.86,
  "Energy": 0.99,
  "Key": 9,
  "Loudness": -1.19,
  "Speechiness": 0.26,
  "Acousticness": 0.55,
  "Instrumentalness": 0,
  "Liveness": 0.24,
  "Valence": 0.67,
  "Tempo": 114.64,
  "Duration_ms": 208111,
  "Total": {
    "$sum": [
      "$Danceability",
      "$Energy",
      "$Key",
      "$Loudness",
      "$Speechiness",
      "$Acousticness",
      "$Instrumentalness",
      "$Liveness",
      "$Valence",
      "$Tempo",
      "$Acousticness"
    ]
  }
},
}
```

\$group stage: Groups the input documents by the specified `_id` expression and returns a single document containing the accumulated values for each distinct group.

Syntax: `db.spotify_data.aggregate([{$group: {_id: '$Artist'}}])`

Grouping the Artist names with `_id` gives out a list of all the Artist present in the dataset.

```
employee> db.spotify_data.aggregate([{$group: {_id: '$Artist'}}])
{
  "_id": "Pouya",
  "_id": "Tami Terrell",
  "_id": "White Noise Baby Sleep",
  "_id": "Paulo Londra",
  "_id": "Alex Zurdo",
  "_id": "The Gramps",
  "_id": "Ana Bárbara",
  "_id": "Gaab",
  "_id": "Maes",
  "_id": "The Cure",
  "_id": "Javed Ali",
  "_id": "dÁvd",
  "_id": "Rolf Zuckowski",
  "_id": "José Alfredo Jiménez",
  "_id": "Tazzy",
  "_id": "The Script",
  "_id": "Ramirez",
  "_id": "Standly",
  "_id": "Elefante",
  "_id": "Chris Lake"
}
```

Accumulation using \$sum: It sums up all the values which were accumulated by the `$group` stage.

Syntax: `db.spotify_data.aggregate([{$group: {_id: '$Artist', totalSongs: {$sum: 1}}})`

Grouping all the Artists together and computing the number of songs by each artist.

```
employee> db.spotify_data.aggregate([{$group: {_id: '$Artist', totalSongs: {$sum: 1}}})
{
  "_id": "Sonu Nigam", totalSongs: 10 },
  "_id": "Gilberto Santa Rosa", totalSongs: 10 },
  "_id": "Hugh Jackman", totalSongs: 10 },
  "_id": "Djavan", totalSongs: 10 },
  "_id": "TV Girl", totalSongs: 10 },
  "_id": "Ovi", totalSongs: 10 },
  "_id": "José José", totalSongs: 10 },
  "_id": "Imanbek", totalSongs: 10 },
  "_id": "Camille Saint-Saëns", totalSongs: 10 },
  "_id": "Evanescence", totalSongs: 10 },
  "_id": "The Kid LAROI", totalSongs: 10 },
  "_id": "Kato", totalSongs: 10 },
  "_id": "Edgardo Ruíz", totalSongs: 10 },
  "_id": "Los Angeles Azules", totalSongs: 10 },
  "_id": "Cascada", totalSongs: 10 },
  "_id": "Parokya NI Edgar", totalSongs: 10 },
  "_id": "Rochak Kohli", totalSongs: 10 },
  "_id": "Lil Nas X", totalSongs: 10 },
  "_id": "Journey", totalSongs: 10 },
  "_id": "Aerosmith", totalSongs: 10 }
}
```

\$skip: It is used to skip n number of documents and passes the remaining documents

Syntax: `db.spotify_data.aggregate([{$group: {_id: '$Artist', totalSongs: {$sum: 1}}}, {$skip: 10}])`

Grouping all the Artists together and computing the number of songs by each artist wherein we skip the first 10 outputs.

```

employee> db.spotify_data.aggregate([{$group: {_id: '$Artist', totalSongs: {$sum: 1}}}, {$skip: 10}])
{
  "_id": "Taylor Swift", "totalSongs": 10 },
  { "_id": "RM", "totalSongs": 10 },
  { "_id": "Matisse", "totalSongs": 10 },
  { "_id": "Bibi and Tina", "totalSongs": 10 },
  { "_id": "Journey", "totalSongs": 10 },
  { "_id": "Rochak Kohli", "totalSongs": 10 },
  { "_id": "Lil Nas X", "totalSongs": 10 },
  { "_id": "Aerosmith", "totalSongs": 10 },
  { "_id": "Cascada", "totalSongs": 10 },
  { "_id": "Parokya MI Edgar", "totalSongs": 10 },
  { "_id": "Los Angeles Azules", "totalSongs": 10 },
  { "_id": "Camille Saint-Saëns", "totalSongs": 10 },
  { "_id": "Evanescence", "totalSongs": 10 },
  { "_id": "The Kid LAROI", "totalSongs": 10 },
  { "_id": "Edgardo Nuñez", "totalSongs": 10 },
  { "_id": "Kato", "totalSongs": 10 },
  { "_id": "Diwan", "totalSongs": 10 },
  { "_id": "Sono Nigam", "totalSongs": 10 },
  { "_id": "Gilberto Santa Rosa", "totalSongs": 10 },
  { "_id": "Hugh Jackman", "totalSongs": 10 }
}

```

\$limit: It is used to pass first n number of documents thus limiting them.

Syntax: `db.spotify_data.aggregate([{$group: {_id: '$Artist', totalSongs: {$sum: 1}}}, {$skip: 10}, {$limit: 10}])`

Grouping all the Artists together and computing the number of songs by each artist wherein we skip the first 10 outputs and limit the output to only 10 entries.

```

employee> db.spotify_data.aggregate([{$group: {_id: '$Artist', totalSongs: {$sum: 1}}}, {$skip: 10}, {$limit: 10}])
{
  "_id": "Henrique & Juliano", "totalSongs": 10 },
  { "_id": "A Day To Remember", "totalSongs": 10 },
  { "_id": "Katy Perry", "totalSongs": 10 },
  { "_id": "Matheus & Kauan", "totalSongs": 10 },
  { "_id": "Bread", "totalSongs": 10 },
  { "_id": "Zezé Di Camargo & Luciano", "totalSongs": 10 },
  { "_id": "Erik Satie", "totalSongs": 10 },
  { "_id": "Sting", "totalSongs": 10 },
  { "_id": "REO Speedwagon", "totalSongs": 10 },
  { "_id": "Christian Nodal", "totalSongs": 10 }
}

```

\$out: It is used to write resulting documents to a new collection

Syntax: `db.spotify_data.aggregate([{$group: {_id: '$Artist', totalSongs: {$sum: 1}}}, {$skip: 10}, {$limit: 10}, {$out: {db: 'employee', coll: 'spotify_data2'}}])`

Creating a new collection which consists 10 values of Artists and their total number of songs in the original dataframe.

```

employee> db.spotify_data.aggregate([{$group: {_id: '$Artist', totalSongs: {$sum: 1}}}, {$skip: 10}, {$limit: 10}])
{
  "_id": "Henrique & Juliano", "totalSongs": 10 },
  { "_id": "A Day To Remember", "totalSongs": 10 },
  { "_id": "Katy Perry", "totalSongs": 10 },
  { "_id": "Matheus & Kauan", "totalSongs": 10 },
  { "_id": "Bread", "totalSongs": 10 },
  { "_id": "Zezé Di Camargo & Luciano", "totalSongs": 10 },
  { "_id": "Erik Satie", "totalSongs": 10 },
  { "_id": "Sting", "totalSongs": 10 },
  { "_id": "REO Speedwagon", "totalSongs": 10 },
  { "_id": "Christian Nodal", "totalSongs": 10 }
}

employee> db.spotify_data2.find()
{
  "_id": "24kGoldn", "totalSongs": 10 },
  { "_id": "Red Hot Chili Peppers", "totalSongs": 10 },
  { "_id": "Drake", "totalSongs": 10 },
  { "_id": "Black Eyed Peas", "totalSongs": 10 },
  { "_id": "Disturbed", "totalSongs": 10 },
  { "_id": "Kris Kristofferson", "totalSongs": 10 },
  { "_id": "Roddy Ricch", "totalSongs": 10 },
  { "_id": "Zac Efron", "totalSongs": 10 },
  { "_id": "MARINA", "totalSongs": 10 },
  { "_id": "I Prevail", "totalSongs": 10 }
}

```

Complex Queries: Setting up a condition to filter the data and then, sorting and accumulating the filtered data using \$sort, \$sum and \$group.

Syntax: `db.spotify_data.aggregate([{$match: {Duration_ms: {$gt: 150000}}}, {$group: {_id: '$Artist', totalDuration: {$sum: '$Duration_ms'}}}, {$sort: {$Artist: 1}}])`

The \$match stage filters the data with duration more than 150000ms, then \$group groups the artists with their total duration of all the songs combined which is sorted in ascending order by their first name.

```
employee> db.spotify_data.aggregate([
... {$match: {Duration_ms: {$gt: 150000}}},
... {$group: {_id: '$Artist', totalDuration: {$sum: '$Duration_ms'}}},
... {$sort: {$Artist: 1}}
... ])
[
  { _id: 'ROSALÍA', totalDuration: 1484701 },
  { _id: 'Stray Kids', totalDuration: 1931243 },
  { _id: 'Abel Pintos', totalDuration: 2267165 },
  { _id: 'Frank Sinatra', totalDuration: 1300893 },
  { _id: 'Ricky Montgomery', totalDuration: 2007006 },
  { _id: 'João Gilberto', totalDuration: 1399188 },
  { _id: 'Nekfeu', totalDuration: 2248760 },
  { _id: 'Michael Bublé', totalDuration: 1911016 },
  { _id: 'Pitbull', totalDuration: 2227880 },
  { _id: 'Metallica', totalDuration: 3544585 },
  { _id: 'Arcane', totalDuration: 1734626 },
  { _id: 'Marc Anthony', totalDuration: 2864087 },
  { _id: 'Cigarettes After Sex', totalDuration: 2692182 },
  { _id: 'Pritam', totalDuration: 2686438 },
  { _id: 'Niro', totalDuration: 2261574 },
  { _id: 'Bacilos', totalDuration: 2508354 },
  { _id: 'Trey Songz', totalDuration: 1966826 },
  { _id: 'Anderson .Paak', totalDuration: 2120984 },
  { _id: 'Gente De Zona', totalDuration: 2024062 },
  { _id: 'Kehlani', totalDuration: 1901218 }
]
```

Syntax: `db.spotify_data.aggregate([{$group: {_id: '$Artist', Songs: {$count: {}}, TotalDuration: {$sum: '$Duration_ms'}, Loudness: {$avg: '$Loudness'}}}, {$sort: {$Duration_ms: -1}}, {$out: {db: 'employee', coll: 'spotify_artist_info'}}])`

Creating a new collection where each artist's name, count of songs, total duration of songs and average of loudness in each song in the dataset.

```
employee> db.spotify_data.aggregate([
... {$group: {_id: '$Artist', Songs: {$count: {}}, TotalDuration: {$sum: '$Duration_ms'}, Loudness: {$avg: '$Loudness'}}},
... {$sort: {$Duration_ms: -1}},
... {$out: {db: 'employee', coll: 'spotify_artist_info'}}
... ])
employee> db.spotify_artist_info.find()
[
  {
    _id: 'Florence + The Machine',
    Songs: 10,
    TotalDuration: 2295408,
    Loudness: -5.181
  },
  {
    _id: 'Lil Tracy',
    Songs: 10,
    TotalDuration: 1570843,
    Loudness: -7.812
  },
]
```