Implementation in NoSql (MongoDB)

Used MongoDB online playground to run below queries (https://mongoplayground.net/)

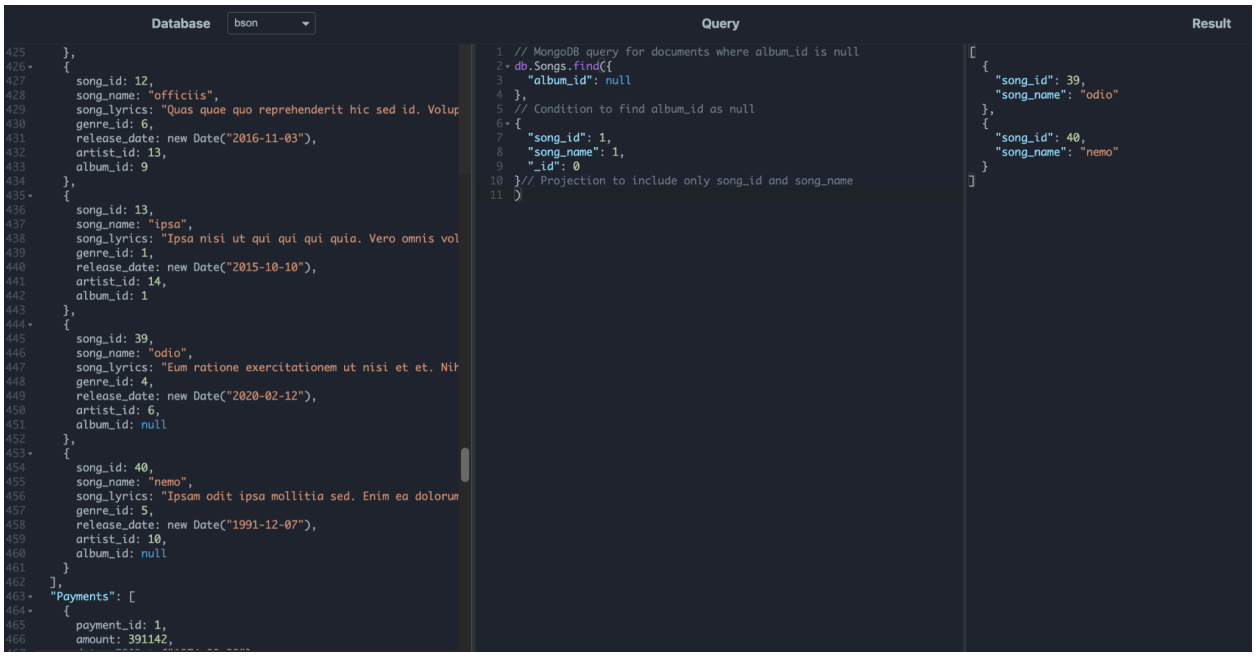
Query 1: Join query (Find albums with at least one song in the 'Pop' genre)

```
db.Album.aggregate([
  {
    $lookup: {
      from: "Songs",
      // Join with the Songs collection
      localField: "album_id",
      // Match album_id from Album collection
      foreignField: "album_id",
      // Match album_id from Songs collection
      as: "songs"// Alias for the resulting array of songs

    }
  },
  {
    $unwind: "$songs"// Unwind the songs array to filter songs individually

  },
  {
    $lookup: {
      from: "Genre",
      // Join with the Genre collection
      localField: "songs.genre_id",
      // Match genre_id from Songs collection
      foreignField: "genre_id",
      // Match genre_id from Genre collection
      as: "genre"// Alias for the resulting genre

    }
  },
  {
    $unwind: "$genre"// Unwind the genre array to filter genres individually

  },
  {
    $match: {
      "genre.genre_name": "Pop"// Filter albums having at least one song in the "Pop"
genre

    }
  },
  {
    $group: {
      _id: "$album_id",
      // Group by album_id
      album_name: {
```

```
        $first: "$album_name"
      },
      // Include album name in the result
      artist_id: {
        $first: "$artist_id"
      }// Include artist_id in the result

    }
  }
])
```

Query 2: SimpleQuery (List the song_id, song_name that does not belong to any album)

```
// MongoDB query for documents where album_id is null
db.Songs.find({
  "album_id": null
},
// Condition to find album_id as null
{
  "song_id": 1,
  "song_name": 1,
  "_id": 0
}// Projection to include only song_id and song_name
)
```



Query 3: Nested query (List the artist_name, artist_id having maximum song_count)

```
// Aggregation pipeline to find artist with maximum song_count
db.Songs.aggregate([
  {
    $group: {
      _id: "$artist_id",
      // Group by artist_id
      song_count: {
        $count: {}
      }// Count the number of songs for each artist

    }
  },
  {
    $lookup: {
      from: "Artist",
      // Join with the Artist collection
      localField: "_id",
      // Match artist_id from Songs with artist_id in Artist
      foreignField: "artist_id",
      // Match artist_id in Artist collection
      as: "artist_info"// Add artist information to the result

    }
  },
  {
    $unwind: "$artist_info"// Unwind the artist_info array to access artist_name
```

```
  },
  {
    $sort: {
      song_count: -1
    }// Sort by song_count in descending order

  },
  {
    $limit: 1// Limit to 1 document, the one with the maximum song_count

  },
  {
    $project: {
      _id: 0,
      // Exclude the _id field
      artist_id: "$_id",
      // Include artist_id
      artist_name: "$artist_info.artist_name",
      // Include artist_name from the Artist collection
      song_count: 1// Include song_count

    }
  }
])
```

Query 4: Join Query (List the artist_name of all artists who have payment amount < 1,000,000)

```
// Aggregation pipeline to join Artist and Payments collections, and filter by total payment amount
db.Payments.aggregate([
  {
    $lookup: {
      from: "Artist",
      // Join with the Artist collection
      localField: "artist_id",
      // Match artist_id in Payments with artist_id in Artist
      foreignField: "artist_id",
      // Match artist_id in Artist collection
      as: "artist_info"// Add artist information to the result

    }
  },
  {
    $unwind: "$artist_info"// Unwind the artist_info array to access artist_name

  },
  {
    $group: {
      _id: "$artist_id",
      // Group by artist_id
      total_payment: {
        $sum: "$amount"
      },
      // Sum the payment amounts per artist
      artist_name: {
        $first: "$artist_info.artist_name"
      }// Get the artist name

    }
  },
  {
    $match: {
      total_payment: {
        $lt: 1000000
      }// Filter for artists whose total payment is below 1000000

    }
  },
  {
    $project: {
      _id: 0,
      // Exclude the _id field
      artist_name: 1,
      // Include artist_name
      artist_id: "$_id",
      // Include artist_id from the group
      total_payment: 1// Include total payment
```

```
    }
  },
  {
    $sort: {
      artist_id: 1// Sort by artist_id in ascending order

    }
  }
])
```