Bitwise Operator

1. $bitsAllSet: Matches if all of the bits are set in a given integer field.

Example:

db.collection.find({ field: { $bitsAllSet: 5 } })

2. $bitsAnySet: Matches if any of the bits are set in a given integer field.

Example:

db.collection.find({ field: { $bitsAnySet: 5 } })

3. $bitsAllClear: Matches if all of the bits are clear in a given integer field.

Example:

db.collection.find({ field: { $bitsAllClear: 5 } })

4. $bitsAnyClear: Matches if any of the bits are clear in a given integer field.

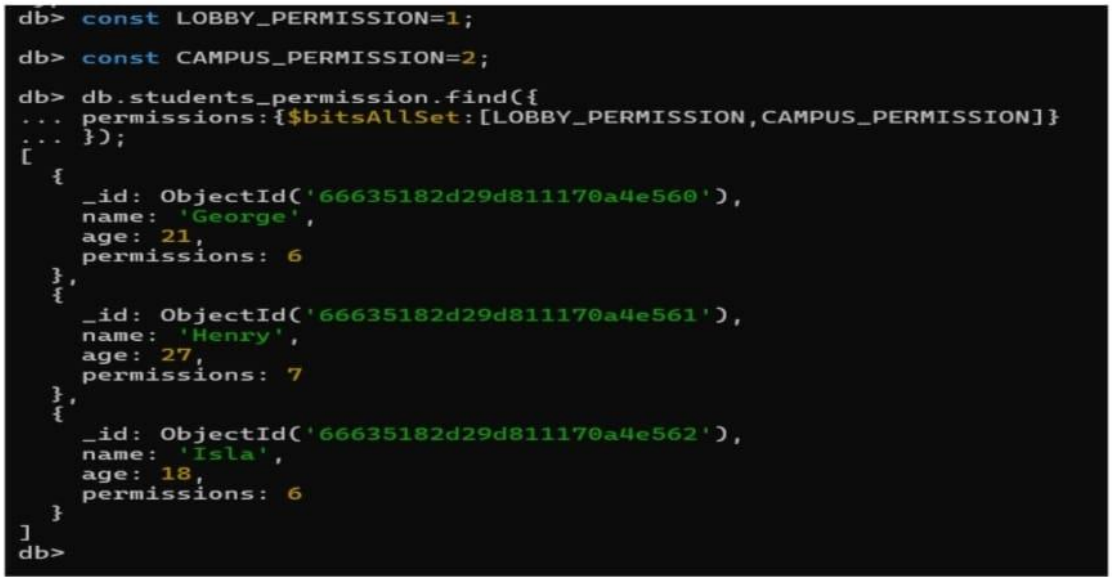
Example:

db.collection.find({ field: { $bitsAnyClear: 5 } })

QUERY:

Queries are used in MongoDB for several reasons:

1. **Data Retrieval**: Queries are used to retrieve specific data from a MongoDB database. You can use queries to fetch specific documents, fields, or values from a collection.
2. **Data Filtering**: Queries allow you to filter data based on specific conditions, such as equality, inequality, or range-based conditions. This helps to narrow down the results to only the relevant data.
3. **Data Aggregation**: Queries can be used to perform aggregation operations, such as grouping, sorting, and aggregating data. This helps to analyze and process large datasets.
4. **Data Validation**: Queries can be used to validate data against a set of rules or constraints, ensuring that the data meets certain criteria.
5. **Data Transformation**: Queries can be used to transform data from one format to another, such as converting data types or performing calculations.



For George and Isla, the `permissions` value of 6 (binary `110`) has both `LOBBY_PERMISSION` and `CAMPUS_PERMISSION` bits set. For Henry, the `permissions` value of 7 (binary `111`) also satisfies the condition as both required bits are set along with an additional bit. This query effectively filters the documents to find students who have both the lobby and campus permissions enabled.

GEOSPATIAL :

Name	Description
$geoIntersects	Selects geometries that intersect with a GeoJSON geometry. The 2dsphere index supports $geoIntersects.
$geoWithin	Selects geometries within a bounding GeoJSON geometry. The 2dsphere and 2d indexes support $geoWithin.
$near	Returns geospatial objects in proximity to a point. Requires a geospatial index. The 2dsphere and 2d indexes support $near.
$nearSphere	Returns geospatial objects in proximity to a point on a sphere. Requires a geospatial index. The 2dsphere and 2d indexes support $nearSphere.

MongoDB provides two types of geospatial indexes:

-2dsphere Indexes, which support queries that interpret geometry on a sphere.

-2d Indexes, which support queries that interpret geometry on a flat surface.

- and specify the string literal `"2dsphere"` as the index type:

db.collection.createIndex({ <location field> : "2dsphere" })

- To create a `2d` index, use the `db.collection.createIndex()` method, specifying the location field as the key and the string literal `"2d"` as the index type:

db.collection.createIndex({ <location field> : "2d" })

GEOSPATIAL QUERY:

1.Location-Based Services(LBS)
2.Logistics and Supply Chain Management
3.Real Estate and Property Management
4.Ride-Sharing and Transportation
5.Retail and Marketing

These queries utilize special operators like $geoNear, $geoWithin, and $near to find documents near a specific point, within a specified area, or based on proximity.

```
db> db.locations.find({
... location:{
... $geoWithin:{
... $centerSphere:[[-74.005,40.712],0.00621376]
... }
... }
... });
[
  {
    _id: 1,
    name: 'Coffee Shop A',
    location: { type: 'Point', coordinates: [ -73.985, 40.748 ] }
  },
  {
    _id: 2,
    name: 'Restaurant B',
    location: { type: 'Point', coordinates: [ -74.009, 40.712 ] }
  },
  {
    _id: 5,
    name: 'Park E',
    location: { type: 'Point', coordinates: [ -74.006, 40.705 ] }
  }
]
db>
```

Let's break down each part of the query:

db.locations.find(...):
This part of the command indicates that we are performing a find query on the locations collection.

location: { $geoWithin: { $centerSphere: [[-74.005, 40.712], 0.00621376]}:
The query is looking for documents where the location field contains geospatial data that is within a certain radius of a center point.

$geoWithin: This operator selects documents with geospatial data within a specified geometry. $centerSphere: This operator specifies a circle for a geospatial query. The circle is defined by a center point and a radius measured in radians.

• [[-74.005, 40.712], 0.00621376]: This array specifies the center of the circle and its radius.

• [-74.005, 40.712]: This array specifies the coordinates of the center point (longitude, latitude).

• 0.00621376: This value specifies the radius of the circle in radians. The radius is calculated based on the Earth's radius in the specified unit (usually miles or kilometers). In this case, it represents a distance of approximately 25 miles (0.00621376 radians)