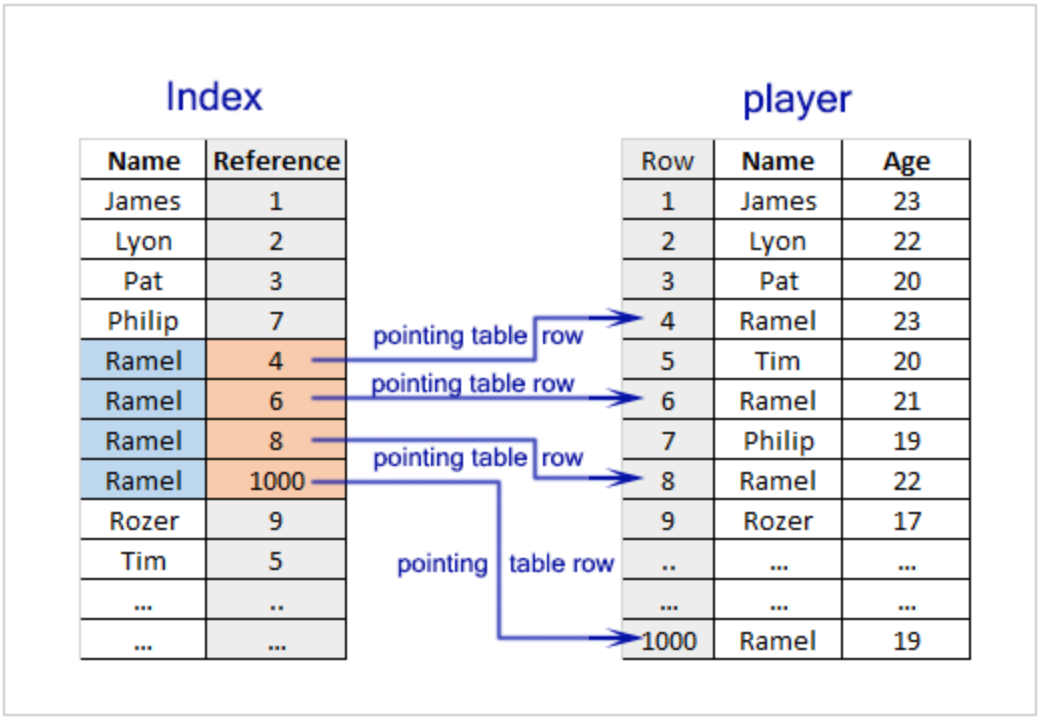
Mongodb: Indexing

The whole point of having an index is to speed up search queries by essentially cutting down the number of records/rows in a table that needs to be examined.

An index is a data structure (most commonly a B- tree) that stores the values for a specific column in a table. An index is created on a column of a table. So, the key points to remember that, an index consists of column values from one table, and that, those values are stored in a data structure.

Here is the pictorial presentation of an index table.



Here in the above example, suppose we have a table 'player' of two columns - 'name' and 'age' and we have shown the location of rows by 'row' column, but point to be noted that 'row' is not a created column in the table.

On the left, each index entry contains the index key (Name). Each entry also includes a reference (which points) to the rows of the table player, which share that particular value and from which we can retrieve the required information.

If we want to see those rows which contain the name 'Ramel', the pointer will point to the table player in location 4,6,8 and 1000 which is referring from the index table.

Index Types

MongoDB provides different types of an index to support specific types of data and queries.

Default _id

All the collections of MongoDB have an index on the _id field that exists by default. If no values specified for the _id the mongodb will create an _id field with an ObjectId value.

The _id index is unique, it restricts clients from inserting two documents with the same value for the _id field.

Single Field

MongoDB supports user-defined indexes on a single field of a document. Here is the example below considering a single column index.

Compound Index

MongoDB also supports user-defined indexes on multiple fields. The order of fields listed in a compound index has significance. For instance, if a compound index consists of {"education": 1, "password": -1 }, the index sorts first by education and then, within each education value, sort by password

Multikey Index

MongoDB uses multikey indexes to index the content stored in arrays. When you index on a column that holds an array value, MongoDB creates separate index entries for every element of the array. These multikey indexes allow queries to select documents that contain arrays by matching on element or elements of the arrays.

Geospatial Index

To support efficient queries of geospatial coordinate data, MongoDB provides two special indexes: 2d indexes and 2sphere indexes uses for planar geometry when returning results spherical geometry to return results.

Text Indexes

Another type of index that MongoDB provides is text index, that supports searching for string content in a collection. These text indexes do not store language-specific stop words (e.g. "the", "a", "or") and restrict the words in a collection to only store root words.

Hashed Indexes

To support hash-based sharding, MongoDB provides a hashed index type, which indexes the hash of the value of a field. These indexes have a more random distribution of values along with their range, but only support equality matches and cannot support range-based queries.

Index Properties

Unique Indexes

The unique property for an index causes reject duplicate values for the indexed field. To create a unique index on a field that already has duplicate values. Other than the unique constraint, unique indexes are functionally interchangeable with other MongoDB indexes.

Sparse Indexes

The sparse property of an index ensures that the index only contains entries for documents that have the indexed field and skips documents that do not have the indexed field.

The sparse index option can be combined with the unique index option to reject documents that have duplicate values for a field but ignore that do not have the indexed key.

TTL Indexes

Another special index is TTL indexes that can be used to automatically remove documents from a collection after a certain amount of time. This is ideal for certain types of information like machine generated event data, logs, and session information that only need to stay behind in a database for a finite amount of time.

How to create Index

Creating an Index in MongoDB is done by using the "**createIndex()**" method.

The following example shows how add index to collection. Let's assume that we have our same Employee collection which has the Field names of "Employeeid" and "EmployeeName".

```
db.Employee.createIndex({Employeeid:1})
```

1 indicates that they should be sorted in ascending order.

How to Find Indexes: getindexes()

Finding an Index in MongoDB is done by using the **"getIndexes"** method.

```
db.Employee.getIndexes()
```

ensureIndex() method also accepts list of options (which are optional). Following is the list −

Parameter	Type	Description
background	Boolean	Builds the index in the background so that building an index does not block other database activities. Specify true to build in the background. The default value is **false**.
unique	Boolean	Creates a unique index so that the collection will not accept insertion of documents where the index key or keys match an existing value in the index. Specify true to create a unique index. The default value is **false**.
name	string	The name of the index. If unspecified, MongoDB generates an index name by concatenating the names of the indexed fields and the sort order.
dropDups	Boolean	Creates a unique index on a field that may have duplicates. MongoDB indexes only the first occurrence of a key and removes all documents from the collection that contain subsequent occurrences of that key. Specify true to create unique index. The default value is **false**.
sparse	Boolean	If true, the index only references documents with the specified field. These indexes use less space but behave differently in some situations (particularly sorts). The default value is **false**.
expireAfterSeconds	integer	Specifies a value, in seconds, as a TTL to control how long MongoDB retains documents in this collection.
v	index version	The index version number. The default index version depends on the version of MongoDB running when creating the index.
weights	document	The weight is a number ranging from 1 to 99,999 and denotes the significance of the field relative to the other indexed fields in terms of the score.
default_language	string	For a text index, the language that determines the list of stop words and the rules for the stemmer and tokenizer. The default value is **english**.
language_override	string	For a text index, specify the name of the field in the document that contains, the language to override the default language. The default value is language.

Exercise

The dataset of students with 200 documents has been given.

Import the dataset in Mongodb with suitable Collection name. Create Index on name. Create another index on name in descending order. Try to fetch the student data on the name criteria.