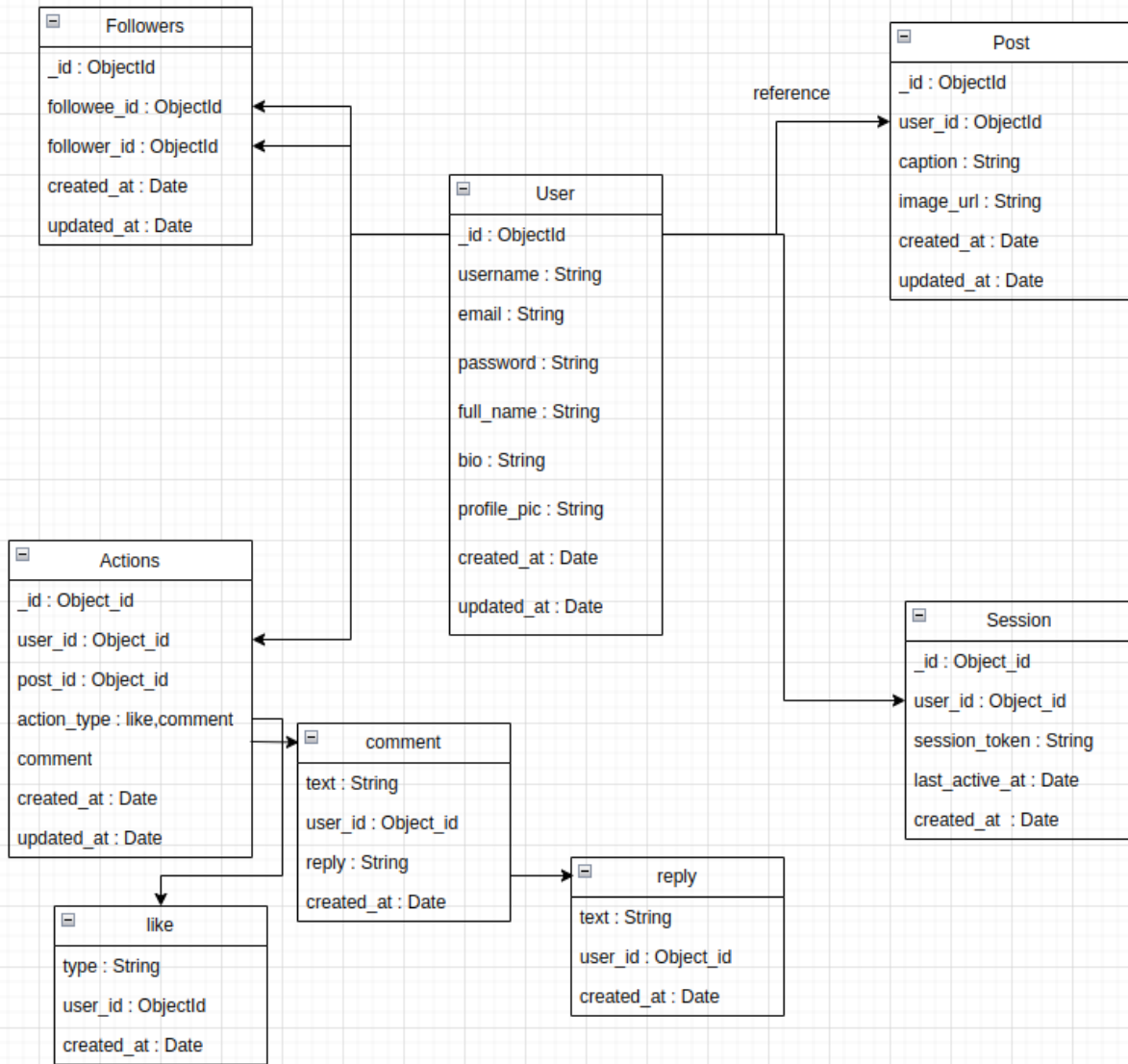


Instagram schema design based on nosql



Queries Screenshots

//InsertMany

es Tools Window Help

Run Debug Stop Import Export Monitoring Tasks DataGen Schema

localhost (2) x users_query.js (1) x * instagramdb:users@localhost (3) x users_query.js (2) x

localhost instagramdb

12

13 db.users.insertMany([

14 {

15 username: "user1",

16 email: "user1@example.com",

17 password: "password123",

18 full_name: "John Doe",

19 bio: "Hello, I am User 1!",

20 profile_pic: "user1.jpg"

21 }

22 ,

23 {

24 username: "user2",

25 email: "user2@example.com",

26 password: "pass987",

27 full_name: "Jane Smith",

28 bio: "Hi, I am User 2!",

29 profile_pic: "user2.jpg"

30 },

31 {

32 username: "user3",

33 email: "user3@example.com",

34 password: "qwerty123",

35 full_name: "Alex Johnson",

36 bio: "I'm User 3!",

37 profile_pic: "user3.jpg"

38 },

39]

0.024 s

Key	Value	Type
(1)	{ } (2 fields)	Object
acknowledged	true	Bool
insertedIds	Array[7]	Array
0	64b645ca58e08e051475cc74	ObjectId
1	64b645ca58e08e051475cc75	ObjectId
2	64b645ca58e08e051475cc76	ObjectId
3	64b645ca58e08e051475cc77	ObjectId
4	64b645ca58e08e051475cc78	ObjectId
5	64b645ca58e08e051475cc79	ObjectId
6	64b645ca58e08e051475cc7a	ObjectId

RunDebugStopImportExportMonitoringTasksDataGenSchema

localhostinstagramdb:system.profile@localhostusers_query.jsinstagramdb:users@localhostinstagramdb:users@localhost (1)instagramdb:users@localhost (2)users_

localhostinstagramdb

```
77
78 db.Post.insertMany([
79 {
80   user_id: ObjectId('64b645b058e08e05f475cc6d'),
81   caption: 'Post 1 caption...',
82   image_url: 'post1.jpg',
83   likes: [
84     ObjectId('64b645b058e08e05f475cc6e'),
85     ObjectId('64b645b058e08e05f475cc6f'),
86   ],
87   comments: [
88     {
89       user_id: ObjectId('64b645b058e08e05f475cc6e'),
90       text: 'Comment 1 on post 1.',
91       created_at: ISODate('2023-07-18T12:30:00Z'),
92     },
93     {
94       user_id: ObjectId('64b645b058e08e05f475cc6f'),
95       text: 'Comment 2 on post 1.',
96       created_at: ISODate('2023-07-18T12:45:00Z'),
97     },
98   ],
99 }
100 ])
```

Line78 db.Post.insertMany[...]Line239 db.users.find();

Go to line 239users14 Docs

Key	Value	Type
(1) 64b645b058e08e05f475cc6d	{ email: "user1@example.com" } (7 fields)	Document
_id	64b645b058e08e05f475cc6d	ObjectId
username	user1	String
email	user1@example.com	String
password	password123	String
full_name	John Doe	String
bio	Hello, I am User 1!	String
profile_pic	user1.jpg	String
(2) 64b645b058e08e05f475cc6e	{ email: "user2@example.com" } (7 fields)	Document
(3) 64b645b058e08e05f475cc6f	{ email: "user3@example.com" } (7 fields)	Document
(4) 64b645b058e08e05f475cc70	{ email: "user4@example.com" } (7 fields)	Document
(5) 64b645b058e08e05f475cc71	{ email: "user5@example.com" } (7 fields)	Document
(6) 64b645b058e08e05f475cc72	{ email: "user6@example.com" } (7 fields)	Document
(7) 64b645b058e08e05f475cc73	{ email: "user7@example.com" } (7 fields)	Document
(8) 64b645ca58e08e05f475cc74	{ email: "user1@example.com" } (7 fields)	Document
(9) 64b645ca58e08e05f475cc75	{ email: "user2@example.com" } (7 fields)	Document

Free Edition

//updateOne()

```
gramdb:sessions@localhost x  instagramdb:posts@localhost x  instagramdb:users@localhost (4) x  instagramdb:sessions@localhost (1) x  users_query.js (4) x  instagramdb:ses

localhost x  instagramdb x

241
242
243
244
245
246
247
248
249 // db.posts.find()
250
251 db.posts.updateOne(
252   {},
253   {
254     $set: {
255       caption: "Updated caption",
256       image_url: "updated_post.jpg",
257       updated_at: new Date(),
258     },
259   }
260 );
261
262
263 db.posts.find()
264
```

Line78 db.Post.insertMany(... x Line263 db.posts.find() x

Go to line 263 posts 2 Docs

```
1  /* 1 createdAt:18/07/2023, 14:17:01*/
2  {
3    "_id" : ObjectId("64b6518558e08e05f475cc85"),
4    "user_id" : ObjectId("64b645b058e08e05f475cc6d"),
5    "caption" : "Updated caption",
6    "image_url" : "updated_post.jpg",
7    "likes" : [
8      ObjectId("64b645b058e08e05f475cc6e"),
9      ObjectId("64b645b058e08e05f475cc6f")
10   ],
11   "comments" : [
12     {
13       "user_id" : ObjectId("64b645b058e08e05f475cc6e"),
14       "text" : "Comment 1 on post 1.",
15       "created_at" : ISODate("2023-07-18T18:00:00.000+05:30")
16     },
17     {
18       "user_id" : ObjectId("64b645b058e08e05f475cc6f"),
19       "text" : "Comment 2 on post 1.",
20       "created_at" : ISODate("2023-07-18T18:15:00.000+05:30")
21     }
22   ]
23 }
```

Free Edition

localhost x instagramdb:system.profile@localhost x users_query.js x instagramdb:users@localhost x instagramdb:users@localhost (1) x instagramdb:users@localhost (2) x users_

localhost x instagramdb x

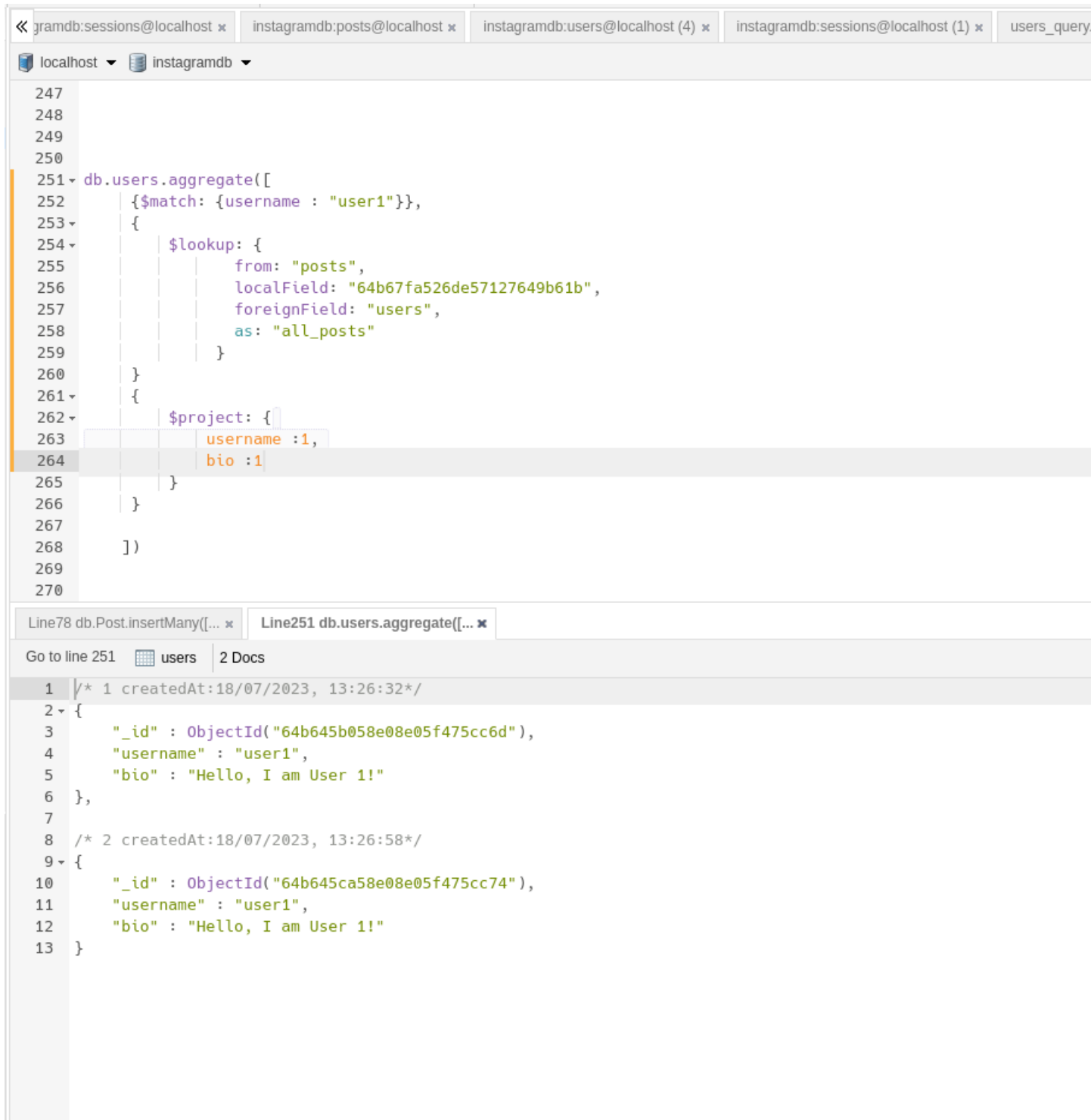
```
124 // inserting in actions collection
125 db.actions.insertMany([
126   {
127     user_id: ObjectId('64b645b058e08e05f475cc6d'),
128     action_type: 'like',
129     post_id: ObjectId('64b6518558e08e05f475cc85'),
130     created_at: ISODate('2023-07-18T12:00:00Z'),
131   },
132
133   {
134     user_id: ObjectId('64b645b058e08e05f475cc6e'),
135     action_type: 'comment',
136     post_id: ObjectId('64b6518558e08e05f475cc86'),
137     comment: {
138       user_id: ObjectId('64b645b058e08e05f475cc6e'),
139       text: 'This is a comment.',
140       created_at: ISODate('2023-07-18T12:30:00Z'),
141       replies: [
142         {
143           user_id: ObjectId('64b645b058e08e05f475cc6d'),
144           text: 'Reply 1 to the comment.',
145           created_at: ISODate('2023-07-18T12:32:00Z'),
146         },
147       ],
148     },
149   },
150 ]),
```

Line125 db.actions.insertMan... x

Go to line 125

```
1 {
2   "acknowledged" : true,
3   "insertedIds" : [
4     ObjectId("64b6756d26de57127649b610"),
5     ObjectId("64b6756d26de57127649b611"),
6     ObjectId("64b6756d26de57127649b612"),
7     ObjectId("64b6756d26de57127649b613")
8   ]
9 }
```

```
//aggregate , $match $lookup $project
```



The screenshot shows a MongoDB IDE interface with several tabs at the top: 'gramdb:sessions@localhost', 'instagramdb:posts@localhost', 'instagramdb:users@localhost (4)', 'instagramdb:sessions@localhost (1)', and 'users_query'. The main editor displays a JavaScript query for the 'users' collection in the 'instagramdb' database. The query uses the \$match, \$lookup, and \$project stages. It matches users with the username 'user1', looks up their posts from the 'posts' collection using a local field 'localField' and a foreign field 'foreignField' in the 'users' collection, and projects the 'username' and 'bio' fields. The query is as follows:

```
251 db.users.aggregate([
252   {$match: {username : "user1"}},
253   {
254     $lookup: {
255       from: "posts",
256       localField: "64b67fa526de57127649b61b",
257       foreignField: "users",
258       as: "all_posts"
259     }
260   }
261   {
262     $project: {
263       username :1,
264       bio :1
265     }
266   }
267 ])
```

Below the query, the IDE shows the results of the query. The results are displayed in a table with columns for 'Line', 'Document', and 'Index'. The results show two documents from the 'users' collection, both with the username 'user1' and the bio 'Hello, I am User 1!'. The documents are as follows:

```
1 /* 1 createdAt:18/07/2023, 13:26:32*/
2 {
3   "_id" : ObjectId("64b645b058e08e05f475cc6d"),
4   "username" : "user1",
5   "bio" : "Hello, I am User 1!"
6 },
7
8 /* 2 createdAt:18/07/2023, 13:26:58*/
9 {
10  "_id" : ObjectId("64b645ca58e08e05f475cc74"),
11  "username" : "user1",
12  "bio" : "Hello, I am User 1!"
13 }
```


localhost xinstagramdb:system.profile@localhost xusers_query.js xinstagramdb:users@localhost xinstagramdb:users@localhost (1) xinstagramdb:users@localhost

localhost xinstagramdb x

```
165
166
167
168 // InsertMany` method
169
170 db.followers.insertMany([
171   {
172     follower_id: ObjectId('64b645b058e08e05f475cc6e'), // Replace with the actual ObjectId of the follower
173     following_id: ObjectId('64b645b058e08e05f475cc6d'), // Replace with the actual ObjectId of the following user
174     created_at: ISODate('2023-07-18T12:00:00Z'),
175   },
176   {
177     follower_id: ObjectId('64b645b058e08e05f475cc6d'),
178     following_id: ObjectId('64b645b058e08e05f475cc6e'),
179     created_at: ISODate('2023-07-18T12:30:00Z'),
180   },
181 ]);
182
183
184
185
186 // db.users.find();
187
188
```

0.022 s

```
1 {
2   "acknowledged" : true,
3   "insertedIds" : [
4     ObjectId("64b678a426de57127649b614"),
5     ObjectId("64b678a426de57127649b615")
6   ]
7 }
```

3  Free Edition

\$ group \$ match

```
localhost ▾ instagrandb ▾
325 // profile_pic: "user14.jpg"
326 // }
327 // });
328
329
330 db.users.aggregate([
331   {
332     $group: {
333       _id: "$bio",
334       users: { $addToSet: "$_id" },
335     },
336   },
337   {
338     $match: {
339       _id: { $ne: null },
340       $expr: { $gt: [ { $size: "$users" }, 1 ] },
341     },
342   },
343 ]);
344
345
346
347
348
```

users 0.043 s 8 Docs

```
9
10 /* 2 */
11 {
12   "_id" : "Tea Lover!",
13   "users" : [
14     ObjectId("64b6c3c526de57127649b643"),
15     ObjectId("64b6c3c526de57127649b640"),
16     ObjectId("64b6c3c526de57127649b641"),
17     ObjectId("64b6c3c526de57127649b642")
18   ]
19 },
20
21 /* 3 */
22 {
23   "_id" : "Hello, I am User 1!",
24   "users" : [
25     ObjectId("64b645b058e08e05f475cc6d"),
26     ObjectId("64b645ca58e08e05f475cc74")
27   ]
28 },
29
30 /* 4 */
31 {
```