



UTD

THE UNIVERSITY OF TEXAS AT DALLAS

PROJECT – 3

GROUP - 18

(BUAN 6320: Database Foundations for Business Analytics)

Abhishek Arya	axa220149@utdallas.edu
Manan Mistry	m xm220108@utdallas.edu
Tanishka Patel	txp210040@utdallas.edu
Het Shah	hrs220002@utdallas.edu
Shekhar Vashist	sxv220008@utdallas.edu

Project made under the guidance of: Farzad Kamalzadeh



SECTION 1: INDEXING AND QUERY TIMING

1.1 List all the current indexes in your database and the columns they are associated with along with the index type.

We have 5 tables in our database and below are the indexes that we have for each table along with the column names and index type.

Table	Key	Type	Column
hierarchy	Primary	BTREE	hierarchy5_id
productdetails	Primary	BTREE	product_id, product_length
	hierarchy5_id_idx	BTREE	hierarchy5_id
promotion	Primary	BTREE	promo_id
	prmo_id_idx	BTREE	promo_id
sales	Primary	BTREE	sales_id
	store_id_idx	BTREE	store_id
	product_id_idx	BTREE	product_id
	promo_id_idx	BTREE	promo_id
storedetails	Primary	BTREE	store_id
	store_id_index	BTREE	store_id

1.2 Explain what is in common between these columns (why these columns are indexed automatically by the database management system)

These indexes are the primary keys and the foreign keys which we incorporated while creating our schema and that is why the database management system has automatically added them.

1.3 Make a copy of your database and delete all the indexes there (you might need to delete foreign keys before you can delete some of the indexes) – now you have two databases: database A with indexes and database B without any indexes.

We have created a duplicate database and named it "mydb_new". So, we have 2 databases "mydb" and "mydb_new". As mentioned, we have removed the indexes from the mydb_new and to do that we had to remove the foreign key constraints which we did by using the ALTER query.

Query used to remove the foreign keys:

```
ALTER TABLE productdetails DROP FOREIGN KEY hierarchy5_id;
ALTER TABLE sales DROP FOREIGN KEY store_id;
ALTER TABLE sales DROP FOREIGN KEY product_id;
ALTER TABLE sales DROP FOREIGN KEY promo_id;
```



Screen shot for the tables we have in our mydb_new database.

The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'Schemas' tree, with 'mydb_new' selected. Under 'Tables', there are five entries: hierarchy, productdetails, promotion, sales, and storedetails. The 'Hierarchy' table is currently expanded. The main workspace contains a SQL editor with the following code:

```
11  from promotions;
12
13  • show index
14    from storedetails;
15
16  • ALTER TABLE productdetails DROP FOREIGN KEY hierarchy5_id;
17  • ALTER TABLE sales DROP FOREIGN KEY store_id;
18  • ALTER TABLE sales DROP FOREIGN KEY product_id;
19  • ALTER TABLE sales DROP FOREIGN KEY promo_id;
20
21  • show tables from mydb_new;
```

Below the SQL editor is a 'Result Grid' showing the results of the 'show tables from mydb_new;' query:

Tables_in_mydb_new
hierarchy
productdetails
promotion
sales
storedetails

At the bottom, the 'Output' pane shows the execution log:

#	Time	Action	Message	Duration / Fetch
30	19:40:21	DROP INDEX 'PRIMARY' ON 'mydb_new'.`storedetails`	OK	0.000 sec
31	19:40:26	DROP INDEX 'PRIMARY' ON 'mydb_new'.`promotion`	OK	0.000 sec
32	19:40:30	DROP INDEX 'PRIMARY' ON 'mydb_new'.`productdetails`	OK	0.000 sec
33	20:04:27	show tables from mydb_new	5 row(s) returned	0.016 sec / 0.000 sec

Screenshots after checking the indexes from our new database:

Sales Table:

The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'Schemas' tree, with 'mydb_new' selected. Under 'Tables', there are five entries: hierarchy, productdetails, promotion, sales, and storedetails. The 'Sales' table is currently expanded. The main workspace contains a SQL editor with the following code:

```
15
16  • ALTER TABLE productdetails DROP FOREIGN KEY hierarchy5_id;
17  • ALTER TABLE sales DROP FOREIGN KEY store_id;
18  • ALTER TABLE sales DROP FOREIGN KEY product_id;
19  • ALTER TABLE sales DROP FOREIGN KEY promo_id;
20
21  • show tables from mydb_new;
22
23
24  • show index
25    from sales;
26
```

Below the SQL editor is a 'Result Grid' showing the results of the 'show index from sales' query:

Table	Non_unique	Key_name	Seq_in_index	Column_name	Collation	Cardinality	Sub_part	Packed	Null	Index_type	Comment	Index_comment	Visible	Expression
sales	0	PRIMARY	1	id	latin1_swedish_ci	1			NO	BTREE			YES	

At the bottom, the 'Output' pane shows the execution log:

#	Time	Action	Message	Duration / Fetch
32	19:40:30	DROP INDEX 'PRIMARY' ON 'mydb_new'.`productdetails`	OK	0.000 sec
33	20:04:27	show tables from mydb_new	5 row(s) returned	0.016 sec / 0.000 sec
34	20:10:16	show index from sales	4 row(s) returned	0.016 sec / 0.000 sec
35	20:10:23	show index from sales	0 row(s) returned	0.000 sec / 0.000 sec



Hierarchy table:

The screenshot shows the MySQL Workbench interface with the 'Hierarchy' table selected. The 'Schemas' tree on the left shows the 'mydb_new' schema containing the 'hierarchy' table. The 'Result Grid' tab displays the table structure with columns: hierarchy1_id, hierarchy2_id, hierarchy3_id, hierarchy4_id, and hierarchy5_id (PK). The 'Output' tab shows the history of actions taken on the database, including 'show tables from mydb_new', 'show index from sales', and 'show index from hierarchy'. The status bar at the bottom right indicates 'Read Only' mode.

```
File Edit View Query Database Server Tools Scripting Help
Navigator Schemas Administration Information
SCHEMAS Filter objects
assignment mydb mydb_new
Tables Views Stored Procedures Functions project6320 sales sys
mydb_new
Tables hierarchy productdetails promotion sales storedetails
Result Grid Filter Rows: Export: Wrap Cell Contents: Table Non_unique Key_name Seq_in_Index Column_name Collation Cardinality Sub_part Packed Null Index_type Comment Index_comment Visible Expression
Table: hierarchy
Columns:
hierarchy1_id varchar(2)
hierarchy2_id varchar(2)
hierarchy3_id varchar(2)
hierarchy4_id varchar(2)
hierarchy5_id varchar(2) PK
Result 10 x
Output:
Action Output
# Time Action Message Duration / Fetch
33 20:04:27 show tables from mydb_new 5 row(s) returned 0.016 sec / 0.000 sec
34 20:10:16 show index from sales 4 row(s) returned 0.000 sec / 0.000 sec
35 20:10:23 show index from sales 0 row(s) returned 0.000 sec / 0.000 sec
36 20:11:29 show index from hierarchy 0 row(s) returned 0.000 sec / 0.000 sec
SQLAdditions < | > | Jump to
Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.
Result Grid Form Editor
Read Only Context Help Snippets
```

Productdetails table:

The screenshot shows the MySQL Workbench interface with the 'Productdetails' table selected. The 'Schemas' tree on the left shows the 'mydb_new' schema containing the 'productdetails' table. The 'Result Grid' tab displays the table structure with columns: hierarchy1_id, hierarchy2_id, hierarchy3_id, hierarchy4_id, and hierarchy5_id (PK). The 'Output' tab shows the history of actions taken on the database, including 'show tables from mydb_new', 'show index from sales', and 'show index from productdetails'. The status bar at the bottom right indicates 'Read Only' mode.

```
File Edit View Query Database Server Tools Scripting Help
Navigator Schemas Administration Information
SCHEMAS Filter objects
assignment mydb mydb_new
Tables Views Stored Procedures Functions project6320 sales sys
mydb_new
Tables hierarchy productdetails promotion sales storedetails
Result Grid Filter Rows: Export: Wrap Cell Contents: Table Non_unique Key_name Seq_in_Index Column_name Collation Cardinality Sub_part Packed Null Index_type Comment Index_comment Visible Expression
Table: productdetails
Columns:
hierarchy1_id varchar(2)
hierarchy2_id varchar(2)
hierarchy3_id varchar(2)
hierarchy4_id varchar(2)
hierarchy5_id varchar(2) PK
Result 11 x
Output:
Action Output
# Time Action Message Duration / Fetch
34 20:10:16 show index from sales 4 row(s) returned 0.016 sec / 0.000 sec
35 20:10:23 show index from sales 0 row(s) returned 0.000 sec / 0.000 sec
36 20:11:29 show index from hierarchy 0 row(s) returned 0.000 sec / 0.000 sec
37 20:12:21 show index from productdetails 0 row(s) returned 0.016 sec / 0.000 sec
SQLAdditions < | > | Jump to
Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.
Result Grid Form Editor
Read Only Context Help Snippets
```



Promotion Table:

The screenshot shows the MySQL Workbench interface with the 'Project 3 SQL Queries' tab selected. The left pane displays the 'mydb_new' schema with its tables: hierarchy, productdetails, promotion, sales, and storedetails. The right pane shows the results of the following SQL query:

```

16 • ALTER TABLE productdetails DROP FOREIGN KEY hierarchy5_id;
17 • ALTER TABLE sales DROP FOREIGN KEY store_id;
18 • ALTER TABLE sales DROP FOREIGN KEY product_id;
19 • ALTER TABLE sales DROP FOREIGN KEY promo_id;
20
21 • show tables from mydb_new;
22
23
24 • show index
25   from promotion;
26

```

The results grid shows the columns: Table, Non_unique, Key_name, Seq_in_Index, Column_name, Collation, Cardinality, Sub_part, Packed, Null, Index_type, Comment, Index_comment, Visible, and Expression. The output section shows the execution log with four entries:

#	Time	Action	Message	Duration / Fetch
35	20:10:23	show index from sales	0 row(s) returned	0.000 sec / 0.000 sec
36	20:11:29	show index from hierarchy	0 row(s) returned	0.000 sec / 0.000 sec
37	20:12:21	show index from productdetails	0 row(s) returned	0.016 sec / 0.000 sec
38	20:12:54	show index from promotion	0 row(s) returned	0.000 sec / 0.000 sec

Storedetails table:

The screenshot shows the MySQL Workbench interface with the 'Project 3 SQL Queries' tab selected. The left pane displays the 'mydb_new' schema with its tables: hierarchy, productdetails, promotion, sales, and storedetails. The right pane shows the results of the following SQL query:

```

16 • ALTER TABLE productdetails DROP FOREIGN KEY hierarchy5_id;
17 • ALTER TABLE sales DROP FOREIGN KEY store_id;
18 • ALTER TABLE sales DROP FOREIGN KEY product_id;
19 • ALTER TABLE sales DROP FOREIGN KEY promo_id;
20
21 • show tables from mydb_new;
22
23
24 • show index
25   from storedetails;
26

```

The results grid shows the columns: Table, Non_unique, Key_name, Seq_in_Index, Column_name, Collation, Cardinality, Sub_part, Packed, Null, Index_type, Comment, Index_comment, Visible, and Expression. The output section shows the execution log with four entries:

#	Time	Action	Message	Duration / Fetch
36	20:11:29	show index from hierarchy	0 row(s) returned	0.000 sec / 0.000 sec
37	20:12:21	show index from productdetails	0 row(s) returned	0.016 sec / 0.000 sec
38	20:12:54	show index from promotion	0 row(s) returned	0.000 sec / 0.000 sec
39	20:13:30	show index from storedetails	0 row(s) returned	0.000 sec / 0.000 sec

1.4 Write at least 5 queries (with JOINs between your tables)

```

SELECT productdetails.hierarchy5_id , hierarchy4_id, cluster_id
FROM hierarchy
JOIN productdetails
ON hierarchy.hierarchy5_id = productdetails.hierarchy5_id;

```

```

SELECT T1.product_id , T2.store_id, T2.store_size
FROM sales as T1
JOIN storedetails as T2

```



```
on T1.store_id = T2.store_id;
```

```
SELECT *
FROM sales as T1
JOIN productdetails as T2
on T1.product_id = T2.product_id;
```

```
SELECT *
FROM productdetails p
LEFT JOIN hierarchy h
ON p.hierarchy5_id = h.hierarchy5_id;
```

```
SELECT *
FROM sales as T1
JOIN promotion as T2
on T1.promo_id = T2.promo_id;
```

**1.5 Execute and time these queries on both databases and report your findings
(repeat timing for each query at least 10 times and average the times)**

As described, we performed the above-mentioned 5 joins on both the databases and repeated them 10 times. Following are the durations for these queries along with the average durations.

Join 1		
S. No.	Duration	
	mydb	mydb_new
1	0.0092590	0.0143370
2	0.0042458	0.0048398
3	0.0047048	0.0029120
4	0.0047063	0.0029545
5	0.0061550	0.0034040
6	0.0049965	0.0029810
7	0.0047528	0.0027848
8	0.0056295	0.0035785
9	0.0044935	0.0026965
10	0.0043695	0.0018758
Average Time	0.0053313	0.0042364



Join 2		
S. No.	Duration	
	mydb	mydb_new
1	0.0648010	0.0066825
2	0.0232860	0.0029335
3	0.0254210	0.0032828
4	0.0121758	0.0031370
5	0.0032605	0.0032295
6	0.0028740	0.0030093
7	0.0081933	0.0019200
8	0.0052275	0.0030133
9	0.0034510	0.0020208
10	0.0031825	0.0028923
Average Time	0.0151873	0.0032121

Join 3		
S. No.	Duration	
	mydb	mydb_new
1	0.0077500	0.0069988
2	0.0079078	0.0058010
3	0.0042440	0.0050835
4	0.0049578	0.0056370
5	0.0046858	0.0042435
6	0.0044440	0.0055780
7	0.0048793	0.0054120
8	0.0047220	0.0050660
9	0.0048728	0.0060500
10	0.0057980	0.0052830
Average Time	0.0054261	0.0055153

Join 4		
S. No.	Duration	
	mydb	mydb_new
1	0.0045880	0.0033020
2	0.0041988	0.0036558
3	0.0045960	0.0037380
4	0.0045670	0.0035850
5	0.0044900	0.0040505
6	0.0039505	0.0043750
7	0.0049200	0.0039093
8	0.0044860	0.0037168
9	0.0036445	0.0028508
10	0.0043068	0.0039043
Average Time	0.0043748	0.0037087



Join 5		
S. No.	Duration	
	mydb	mydb_new
1	0.0047143	0.0060633
2	0.0043740	0.0037553
3	0.0044875	0.0033308
4	0.0039323	0.0040460
5	0.0042280	0.0038518
6	0.0036055	0.0040115
7	0.0040215	0.0035223
8	0.0041393	0.0040298
9	0.0035778	0.0040843
10	0.0038728	0.0030980
Average Time	0.0040953	0.0039793

Findings: As we can see, the average duration of the new database has given smaller durations which means that the queries are taking less time to perform in the database with no indexes as compared to the database with indexes. Out of 5 joins there was only 1 join query (i.e., join 3) where mydb_new's duration was a little higher than the mydb other wise mydb_new has lesser duration in rest of the join queries.

1.6 Select some columns from database A (columns that are not already indexed) and create an index on them.

We have added the following indexes for the columns:

Table	Index	Column
hierarchy	idx_hierarchy_hierarchy4_id	hierarchy4_id
productdetails	idx_productdetails_cluster_id	cluster_id
sales	idx_sales_product_id	product_id
storedetails	idx_storedetails_storetype_id	storetype_id

1.7 Write a query for each column – the query should include the column in the WHERE clause in a condition.

Table: hierarchy

```
SELECT *
FROM hierarchy
WHERE hierarchy4_id = "H00000400";
```



Table: productdetails

```
SELECT *  
FROM productdetails  
WHERE cluster_id = "cluster_4";
```

Table: sales

```
SELECT *  
FROM sales  
WHERE product_id = "P0001";
```

Table: storedetails

```
SELECT *  
FROM storedetails  
WHERE storetype_id = "ST04";
```

1.8 Execute and time these queries on both databases and report your findings (repeat timing for each query at least 10 times and average the times)

Table: hierarchy		
S. No.	Duration	
	mydb	mydb_new
1	0.0011288	0.0055335
2	0.0019793	0.0005200
3	0.0005063	0.0005088
4	0.0003670	0.0006133
5	0.0003380	0.0004685
6	0.0003433	0.0006643
7	0.0004320	0.0004875
8	0.0003295	0.0005448
9	0.0003798	0.0005650
10	0.0007000	0.0005310
Average Time	0.0006504	0.0010437



Table: productdetails		
S. No.	Duration	
	mydb	mydb_new
1	0.0009948	0.0054110
2	0.0004148	0.0007455
3	0.0004050	0.0007740
4	0.0004728	0.0007400
5	0.0005270	0.0011115
6	0.0004880	0.0008088
7	0.0003865	0.0007418
8	0.0004785	0.0006983
9	0.0004108	0.0007315
10	0.0004175	0.0010785
Average Time	0.0004996	0.0012841

Table: sales		
S. No.	Duration	
	mydb	mydb_new
1	0.0030898	0.5782740
2	0.0021630	0.3881418
3	0.0030580	0.4212490
4	0.0018688	0.4032643
5	0.0018480	0.3667640
6	0.0019165	0.3753288
7	0.0018225	0.4124388
8	0.0026118	0.4310423
9	0.0018078	0.3882908
10	0.0019483	0.4552420
Average Time	0.0022134	0.4220036

Table: storedetails		
S. No.	Duration	
	mydb	mydb_new
1	0.0024603	0.0044748
2	0.0005505	0.0006833
3	0.0005988	0.0004403
4	0.0005240	0.0003680
5	0.0006048	0.0005415
6	0.0004660	0.0005228
7	0.0005288	0.0004973
8	0.0004798	0.0004260
9	0.0004520	0.0003800
10	0.0005983	0.0005263
Average Time	0.0007263	0.0008860



Findings: The results show that the database with indexes i.e., "mydb" has very less average duration for queries as compared to the database without indexes i.e., "mydb_new".

This implies that creating indexes will make the queries run more efficiently, especially for the tables with large data like we can see for the sales table in our data which has large data and the difference in the average duration is quite visible.

1.9 Make a conclusion based on your findings in this part.

After comparing the results of JOINS and WHERE queries we can conclude that creating indexes will make queries run more efficiently.

Even though the average durations for JOIN queries were not that significantly less for the database with index but the difference is quite visible for the WHERE queries in database with indexes.

Fast data retrieval is one of the most important benefits of adding indexes hence, with indexes in place, MySQL can locate and retrieve data much faster, which can lead to significant improvements in query performance.

SECTION 2: MONGODB AND MQL

2.1. Explore your dataset and familiarize yourself with the dataset and its content

We have taken Pakistan's Largest PakWheels Automobiles Listings dataset to answer the following questions. It was in a json format and while importing in the no sql booster for MongoDB it automatically changes into table format. Looking into the unstructured data it is large in size, total documents count = 55675; size = 74.6 MB and average object size = 1405(1.4 KB). We can find many attribute/fields like {@type, brand (@type, name)}, model, description , item condition, modelDate, manufacturer, FuelType, name, image, vehicleTransmission, color, bodyType, vehicleEngine, mileageFromOdometer, sellerLocation, postedFrom, keywords, extraFeatures , Features , price , priceCurrency .

The screenshots below shows the statistics of dataset that we have taken in this study.

We use the following query to obtain the stats:

```
db.PakWheels.stats();
```



pakwheels:PakWheels@localhost - NoSQLBooster for MongoDB

File Edit Options View Favorites Tools Window Help

Connect SQL Run Debug Import Export Monitoring Tasks DataGen Schema

Open Connections

localhost

admin (0.19GB)
config
local
users
pakwheels (2)

pkwheels:PakWheels@localhost

```
localhost > db.PakWheels.stats();
18   fromType: "file",
19   batchSize: 2000,
20   contents
21 })
22 db.PakWheels.stats();
```

Result (3) x Result (4) x Result (5) x Result (6) x Result (7) x Console (1) x Result (8) x Console (2) x Result (9) x Result (10) x Result (11) x Result (12) x

0.092 s

Key	Type
ns	String
size	Int32
count	Int32
avgObjSize	Int32
numOrphanDocs	Int32
storageSize	Int32
freeStorageSize	Int32
capped	Bool
wiredTiger	Object
nindexes	Int32
indexBuilds	Array
totalIndexSize	Int32

Copyright © nosqlbooster.com Version 8.0.9 Free Edition Line: 22, Column: 1 (21 selected) Show Log Feedback 11:18:09 PM

Windows Taskbar: 77°F Mostly cloudy ENG 11:18 PM IN 5/9/2023

pakwheels:PakWheels@localhost - NoSQLBooster for MongoDB

File Edit Options View Favorites Tools Window Help

Connect SQL Run Debug Import Export Monitoring Tasks DataGen Schema

Open Connections

localhost

admin (0.19GB)
config
local
users
pakwheels (2)

pkwheels:PakWheels@localhost

```
15 > mb.importContent({
16   connection: "localhost",
17   database: "pakwheels",
18   fromType: "file",
19   batchSize: 2000,
20   contents
21 })
22 db.PakWheels.stats();
```

Result (3) x Result (4) x Result (5) x Result (6) x Result (7) x Console (1) x Result (8) x Console (2) x Result (9) x Result (10) x Result (11) x Result (12) x

0.092 s

Key	Type
numOrphanDocs	Int32
storageSize	Int32
freeStorageSize	Int32
capped	Bool
wiredTiger	Object
nindexes	Int32
indexBuilds	Array
totalIndexSize	Int32
totalSize	Int32
indexSizes	Object
scaleFactor	Int32
ok	Int32

Copyright © nosqlbooster.com Version 8.0.9 Free Edition Line: 22, Column: 1 (21 selected) Show Log Feedback 11:21:07 PM

Windows Taskbar: 75°F Mostly cloudy ENG 11:21 PM IN 5/9/2023



2.2. Explain why it is better to use non-relational databases such as MongoDB to work with such a dataset (explain in the context of your dataset)

As we can see from the above screenshots the unstructured data is large in size around 75 MB with total document counts of 55675, one of the main advantages of non-relational databases is their ability to handle large volumes of unstructured or semi-structured data, which can be difficult to manage in a traditional relational database. This is because non-relational databases allow for flexible schema design and can store data in a variety of formats, such as JSON, BSON, or XML.

Another advantage of non-relational databases is their ability to scale horizontally, meaning that they can easily distribute data across multiple servers to handle high levels of traffic and ensure high availability. This is particularly useful for datasets that are constantly growing or that need to be accessed by many users at the same time. **Our dataset is of the same nature where new data can be added for each consecutive coming years whether related to car's bodytype, brand, mileage From Odometer or color.** So we can say that our dataset is lively and it would expand horizontally and for that non-relational database is a must.

As our dataset is large and unstructured, non-relational databases can provide faster read and write performance compared to relational databases, especially when dealing with large datasets. This is because non-relational databases can store data in a way that is optimized for fast retrieval and can use techniques such as sharing and indexing to speed up queries.

In summary, if your dataset is large and unstructured or semi-structured, needs to be accessed by many users simultaneously, or requires high performance and scalability, then a non-relational database such as MongoDB might be a better choice than a traditional relational database. However, the choice of database depends on many factors, including the nature of the data, the application requirements, and the available resources, so it's important to carefully evaluate the options before making a decision.

2.3. Import your dataset into MongoDB using either MongoDB Compass or MongoDB Database Tools*

NoSQLBooster is one of the most popular cross-platform GUI (Graphical User Interface) **tools for MongoDB that houses a built-in MongoDB Script Debugger, Comprehensive Server Monitoring Tools, Query Code Generator, and advanced IntelliSense support.**

After importing the JSON dataset into the No SQL Booster MongoDB we found out the following output



```
import into "pakwheels.PakWheels" start...
import into "pakwheels.PakWheels" 100%, 55675 docs inserted.
import into "pakwheels.PakWheels" finished.
```

A total of 55675 document(s) have been imported into 1 collection(s).

```
{
  "PakWheels": {
    "nInserted": 55675,
    "nModified": 0,
    "nSkipped": 0,
    "failed": 0
  }
}
```

The below screenshots shows the same:

The screenshot shows the NoSQLBooster interface for MongoDB. In the left sidebar, under 'Open Connections', 'localhost' is selected, and 'pakwheels' is highlighted. In the main pane, there's a code editor window titled 'pakwheels:PakWheels@localhost' containing the import script. Below it is a 'Console' window showing the command-line output of the import process. The console output matches the text shown above. At the bottom, the Windows taskbar is visible with various icons and system status information.

```
1 import into "pakwheels.PakWheels" start...
2 import into "pakwheels.PakWheels" 100%, 55675 docs inserted.
3 import into "pakwheels.PakWheels" finished.
4
5 A total of 55675 document(s) have been imported into 1 collection(s).
6 [
7   "PakWheels": {
8     "nInserted": 55675,
9     "nModified": 0,
10    "nSkipped": 0,
11    "failed": 0
12  }
13 ]
```



pakwheels:PakWheels@localhost - NoSQLBooster for MongoDB

File Edit Options View Favorites Tools Window Help

Connect SQL Run Debug Stop Import Export Monitoring Tasks DataGen Schema

Open Connections

localhost

- admin (0.19GB)
- config
- local
- users
- pakwheels (2)**

```

13 ]
14
15 mb.importContent({
16   connection: "localhost",
17   database: "pakwheels",
18   fromType: "file",
19   batchSize: 2000,
20   contents
21 }) (property) contents: {
22   content: string;
23   collection: string;
24   idPolicy: string;
} []
26.846 s
1 import into "pakwheels.PakWheels" start...
2 import into "pakwheels.PakWheels" 100%, 55675 docs inserted.
3 import into "pakwheels.PakWheels" finished.
4
5 A total of 55675 document(s) have been imported into 1 collection(s).
6 [
7   "PakWheels": {
8     "nInserted": 55675,
9     "nModified": 0,
10    "nSkipped": 0,
11    "failed": 0
12  }
13 ]

```

Copyright © nosqlbooster.com Version 8.0.9 Free Edition

Line: 21, Column: 3 (133 selected) Show Log Feedback 11:25:29 PM

Windows Taskbar: 75°F Mostly cloudy ENG 11:25 PM IN 5/9/2023

By using db.PakWheels.find() function we can find out all the fields, values in all the documents in a collection. Screenshot is below:

pakwheels:PakWheels@localhost - NoSQLBooster for MongoDB

File Edit Options View Favorites Tools Window Help

Connect SQL Run Debug Stop Import Export Monitoring Tasks DataGen Schema

Open Connections

localhost

- admin (0.19GB)
- config
- local
- users
- pakwheels (2)**

```

14
15 mb.importContent({
16   connection: "localhost",
17   database: "pakwheels",
18   fromType: "file",
19   batchSize: 2000,
20   contents
21 })
22 db.PakWheels.stats();
23
24 db.PakWheels.find()
25
26

```

Console x Find x

Key	Type
1 645b07071d4b20b940bfd8fd { "@type": "Car", "name": "Suzuki Mehran 2013 for sale in Rawalpindi" } (24 fields)	Document
_id	Objectid
@type	String
brand	Object
model	String
description	I'm 100% original. Alloy Rims. New tires installed recently. Driven on petrol throughout. Never been int.
itemCondition	String
modelDate	2013 (2.0K)
manufacturer	Suzuki
fuelType	Petrol

Copyright © nosqlbooster.com Version 8.0.9 Free Edition

Line: 24, Column: 1 (19 selected) Show Log Feedback 11:36:22 PM

Windows Taskbar: 75°F Mostly cloudy ENG 11:36 PM IN 5/9/2023

2.4. List some of the attributes (field/properties) of your database which are common among all documents.

For this we use the following query to find out the common fields among all the documents:



Query:

```
db.PakWheels.aggregate([{$project:{fields:{$objectToArray:"$$ROOT"}},{$unwind:"$fields"},{$group:{_id:"$fields.k",count:{$sum:1}}},{$match:{count:{$gte:db.PakWheels.count()}}},{$project:{_id:0, field:"$_id"}}}])
```

The below screenshots shows the output of the above query which shows that all the attributes are common except “bodyType”. 23 attributes are common among all the documents.

The screenshot shows the NoSQLBooster interface for MongoDB. The left sidebar lists connections, and the main area shows the command line and results. The command entered is the aggregate query provided in the text above. The results pane displays a table of 23 common attributes across 23 documents. The table has columns for Key, Value, and Type. The keys correspond to the fields listed in the table below.

Key	Type
{ field : "name" }	Object
field	String
{ field : "vehicleTransmission" }	Object
{ field : "extraFeatures" }	Object
{ field : "adLastUpdated" }	Object
{ field : "_id" }	Object
{ field : "modelDate" }	Object
{ field : "vehicleEngine" }	Object
{ field : "keywords" }	Object
{ field : "features" }	Object

The University of Texas at Austin
EST. 1969

Key	Value	Type
⑩ (9)	{ field : "features" }	Object
⑩ (10)	{ field : "sellerLocation" }	Object
⑩ (11)	{ field : "brand" }	Object
⑩ (12)	{ field : "manufacturer" }	Object
⑩ (13)	{ field : "@type" }	Object
⑩ (14)	{ field : "model" }	Object
⑩ (15)	{ field : "fuelType" }	Object
⑩ (16)	{ field : "color" }	Object
⑩ (17)	{ field : "priceCurrency" }	Object
⑩ (18)	{ field : "image" }	Object

Copyright © nosqlbooster.com Version 8.0.9 Free Edition Line: 1, Column: 1 Show Log Feedback 12:30:02 AM

Key	Value	Type
⑩ (14)	{ field : "model" }	Object
⑩ (15)	{ field : "fuelType" }	Object
⑩ (16)	{ field : "color" }	Object
⑩ (17)	{ field : "priceCurrency" }	Object
⑩ (18)	{ field : "image" }	Object
⑩ (19)	{ field : "price" }	Object
⑩ (20)	{ field : "mileageFromOdometer" }	Object
⑩ (21)	{ field : "itemCondition" }	Object
⑩ (22)	{ field : "postedFrom" }	Object
⑩ (23)	{ field : "description" }	Object

Copyright © nosqlbooster.com Version 8.0.9 Free Edition Line: 1, Column: 1 Show Log Feedback 12:30:06 AM

2.4.1. For these fields, provide some of the values they contain in the database

In case of “manufacturer” attribute total 50 values are there some of them are given below in the output console in the below screenshot

db.PakWheels.distinct("manufacturer");



```
pakwheels:PakWheels@localhost - NoSQLBooster for MongoDB
File Edit Options View Favorites Tools Window Help
Connect SQL Run Debug Stop Import Export Monitoring Tasks DataGen Schema
Open Connections * pakwheels:PakWheels@localhost x
localhost
  admin (0.19GB)
  config
  local
  users
  pakwheels (2)

localhost pakwheels
26 // 2.4
27
28 db.PakWheels.aggregate([
29   {$project: {fields: {$objectToArray: "$$ROOT"}, $unwind: "$fields"}, $group: {_id: "fields.k", count: {$sum: 1}}}, {
30   // 2.4.1
31   db.PakWheels.distinct("manufacturer")
32
33 // 2.5
34 db.PakWheels.aggregate([
35   {
36     $project: {
37       attributes: {
```

Key	Type
(1)	String
(2)	String
(3)	String
(4)	String
(5)	String
(6)	String
(7)	String
(8)	String
(9)	String
(10)	String

Copyright © nosqlbooster.com Version 8.0.9 Free Edition Line: 31, Column: 1 (37 selected) Show Log Feedback 12:40:52 AM

Windows Taskbar: 74°F Mostly cloudy ENG 12:40 AM IN 5/10/2023

```
Pakwheels:PakWheels@localhost - NoSQLBooster for MongoDB
File Edit Options View Favorites Tools Window Help
Connect SQL Run Debug Stop Import Export Monitoring Tasks DataGen Schema
Open Connections * pakwheels:PakWheels@localhost x
localhost
  admin (0.19GB)
  config
  local
  users
  pakwheels (2)

localhost pakwheels
26 // 2.4
27
28 db.PakWheels.aggregate([
29   {$project: {fields: {$objectToArray: "$$ROOT"}, $unwind: "$fields"}, $group: {_id: "fields.k", count: {$sum: 1}}}, {
30   // 2.4.1
31   db.PakWheels.distinct("manufacturer")
32
33 // 2.5
34 db.PakWheels.aggregate([
35   {
36     $project: {
37       attributes: {
```

Key	Type
(11)	String
(12)	String
(13)	String
(14)	String
(15)	String
(16)	String
(17)	String
(18)	String
(19)	String
(20)	String

Copyright © nosqlbooster.com Version 8.0.9 Free Edition Line: 31, Column: 1 (37 selected) Show Log Feedback 12:40:58 AM

Windows Taskbar: 74°F Mostly cloudy ENG 12:40 AM IN 5/10/2023

In case of “model” attribute total 50 values are there some of them are given below in the output console in the below screenshot

`db.PakWheels.distinct("model");`



pakwheels:PakWheels@localhost - NoSQLBooster for MongoDB

```
26 // 2.4
27
28 db.PakWheels.aggregate([
29   {
30     $project: {
31       attributes: {
32         $objectToArray: "$$ROOT"
33       }
34     }
35   },
36   {
37     $group: {
38       _id: "$fields.k",
39       count: {
40         $sum: 1
41       }
42     }
43   }
44 ],
45 {
46   $unwind: "$fields"
47 }
48 )
49
50
51 db.PakWheels.distinct("model");
```

Result (13) x Result (14) x Result (15) x Result (16) x Aggregate (2) x Console (1) x Aggregate (3) x Console (2) x Aggregate (4) x Result (18) x Result (17) x

Key	Value	Type
(1)	1 Series	String
(2)	1000	String
(3)	120 Y	String
(4)	1200	String
(5)	1300	String
(6)	200 D	String
(7)	200 T	String
(8)	205	String
(9)	240 Gd	String
(10)	250 D	String

Copyright © nosqlbooster.com Version 8.0.9 Free Edition Line: 31, Column: 1 (31 selected) Show Log Feedback 12:46:04 AM

pakwheels:PakWheels@localhost - NoSQLBooster for MongoDB

```
26 // 2.4
27
28 db.PakWheels.aggregate([
29   {
30     $project: {
31       attributes: {
32         $objectToArray: "$$ROOT"
33       }
34     }
35   },
36   {
37     $group: {
38       _id: "$fields.k",
39       count: {
40         $sum: 1
41       }
42     }
43   }
44 ],
45 {
46   $unwind: "$fields"
47 }
48 )
49
50
51 db.PakWheels.distinct("model");
```

Result (13) x Result (14) x Result (15) x Result (16) x Aggregate (2) x Console (1) x Aggregate (3) x Console (2) x Aggregate (4) x Result (18) x Result (17) x

Key	Value	Type
(10)	250 D	String
(11)	3 Series	String
(12)	300 C	String
(13)	300 Series	String
(14)	323	String
(15)	350Z	String
(16)	370Z	String
(17)	5 Series	String
(18)	6 Series	String
(19)	626	String

Copyright © nosqlbooster.com Version 8.0.9 Free Edition Line: 31, Column: 1 (31 selected) Show Log Feedback 12:46:09 AM

In case of “features” attribute total 28 values are there some of them are given below in the output console in the below screenshot

db.PakWheels.distinct("features");



pakwheels:PakWheels@localhost - NoSQLBooster for MongoDB

File Edit Options View Favorites Tools Window Help

Connect SQL Run Debug Stop Import Export Monitoring Tasks DataGen Schema

Open Connections

localhost

- admin (0.19GB)
- config
- local
- users
- pakwheels (2)

* pakwheels:PakWheels@localhost x

localhost pakwheels

```
26 // 2.4
27
28 db.PakWheels.aggregate([{$project:{fields:{$objectToArray:"$ROOT"}},{$unwind:"$fields"},{$group:{_id:"$fields.k",count:{$sum:1}}}},{
29
30 // 2.4.1
31 db.PakWheels.distinct("features");
32
33 // 2.5
34 db.PakWheels.aggregate([
35 {
36 $project: {
37 attributes: {
```

Result (15) x Result (16) x Aggregate (2) x Console (1) x Aggregate (3) x Console (2) x Aggregate (4) x Result (20) x Console (3) x Aggregate (5) x Result (17) x

0.509 s | 28 Docs

Key	Value	Type
(1)	ABS	String
(2)	AM/FM Radio	String
(3)	Air Bags	String
(4)	Air Conditioning	String
(5)	Alloy Rims	String
(6)	CD Player	String
(7)	Cassette Player	String
(8)	Climate Control	String
(9)	CoolBox	String
(10)	Cruise Control	String

Copyright © nosqlbooster.com Version 8.0.9 Free Edition

Line: 31, Column: 1 (34 selected) Show Log Feedback 12:48:52 AM

Windows Search Mail Camera Taskbar Icons

74°F Mostly cloudy ENG 12:48 AM IN 5/10/2023

pakwheels:PakWheels@localhost - NoSQLBooster for MongoDB

File Edit Options View Favorites Tools Window Help

Connect SQL Run Debug Stop Import Export Monitoring Tasks DataGen Schema

Open Connections

localhost

- admin (0.19GB)
- config
- local
- users
- pakwheels (2)

* pakwheels:PakWheels@localhost x

localhost pakwheels

```
26 // 2.4
27
28 db.PakWheels.aggregate([{$project:{fields:{$objectToArray:"$ROOT"}},{$unwind:"$fields"},{$group:{_id:"$fields.k",count:{$sum:1}}}},{
29
30 // 2.4.1
31 db.PakWheels.distinct("features");
32
33 // 2.5
34 db.PakWheels.aggregate([
35 {
36 $project: {
37 attributes: {
```

Result (15) x Result (16) x Aggregate (2) x Console (1) x Aggregate (3) x Console (2) x Aggregate (4) x Result (20) x Console (3) x Aggregate (5) x Result (17) x

0.509 s | 28 Docs

Key	Value	Type
(10)	Cruise Control	String
(11)	DVD Player	String
(12)	Front Camera	String
(13)	Front Speakers	String
(14)	Heated Seats	String
(15)	Immobilizer Key	String
(16)	Keyless Entry	String
(17)	Navigation System	String
(18)	Power Locks	String
(19)	Power Mirrors	String

Copyright © nosqlbooster.com Version 8.0.9 Free Edition

Line: 31, Column: 1 (34 selected) Show Log Feedback 12:48:56 AM

Windows Search Mail Camera Taskbar Icons

74°F Mostly cloudy ENG 12:48 AM IN 5/10/2023



The screenshot shows the NoSQLBooster interface for MongoDB. The top navigation bar includes File, Edit, Options, View, Favorites, Tools, Window, and Help. The main window displays an aggregation pipeline in the query editor:

```
db.PakWheels.aggregate([
  {
    $project: {
      attributes: {
        $objectToArray: "$$ROOT"
      }
    }
  },
  {
    $unwind: "$attributes"
  },
  {
    $group: {
      _id: "$attributes.k",
      count: {
        $sum: 1
      }
    }
  },
  {
    $match: {
      count: {
        $gt: 1
      }
    }
  }
])
```

The results pane shows a table with 28 documents, each containing a key and value pair:

Key	Type
(19)	String
(20)	String
(21)	String
(22)	String
(23)	String
(24)	String
(25)	String
(26)	String
(27)	String
(28)	String

Similarly, we can find out the rest of the values of these common attributes.

2.5. List some of the attributes (field/properties) of your database which are not common among all the documents

We used the following query to find out the not common attributes among all the documents in the database:

```
db.PakWheels.aggregate([
  {
    $project: {
      attributes: {
        $objectToArray: "$$ROOT"
      }
    }
  },
  {
    $unwind: "$attributes"
  },
  {
    $group: {
      _id: "$attributes.k",
      count: {
        $sum: 1
      }
    }
  },
  {
    $match: {
      count: {
        $lt: 2
      }
    }
  }
])
```



```
        count: {
          $ne: 55675
        }
      },
    {
      $project: {
        attribute: "$_id",
        _id: 0
      }
    }
  ]);

```

The below screenshot is showing the output of the above query:

The screenshot shows the NoSQLBooster interface for MongoDB. The left sidebar lists databases: admin (0.19GB), config, local, users, and pakwheels (2). The pakwheels database is selected. In the main pane, an aggregate query is being run against the pakwheels collection. The query code is as follows:

```
db.Pakwheels.aggregate([
  {
    $project: {
      attributes: {
        $objectToArray: "$$ROOT"
      }
    }
  },
  {
    $unwind: "$attributes"
  }
])
```

The results table shows one document with the key "attribute" and value "bodyType". The value is an object with the key "bodyType" and string value "bodyType".

Key	Value	Type
attribute	{ attribute : "bodyType" }	Object
bodyType	bodyType	String

At the bottom, the status bar indicates: Copyright © nosqlbooster.com Version 8.0.9 Free Edition Line: 60, Column: 1 (393 selected) Show Log Feedback 11:40:26 PM 75°F Mostly cloudy ENG 11:40 PM IN 5/9/2023.

Using this we get "**bodyType**" as an attribute which is not common in all documents.

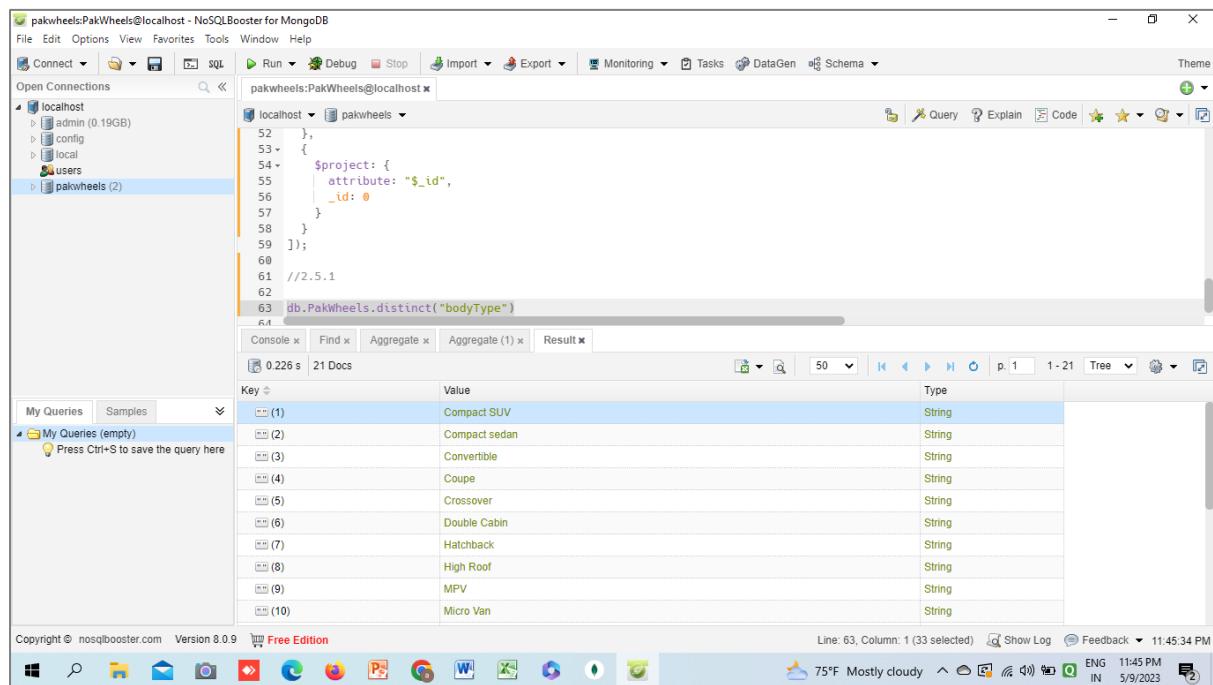
2.5.1. For these fields, provide some of the values they contain in the database

As we know that "bodyType" attribute is the uncommon one among all the documents, some of the values it contains in the database are as follows with screenshots attached:

Total 21 values are inside the "bodyType" attribute some of them are as follows: Compact SUV, Compact Sedan, Convertible, Coupe , Crossover , Double Cabin, Hatchback, High Roof, MPV , Micro Van, Mini-Van, Mini Vehicles, Off-Road Vehicles, Pick Up, SUV , Sedan, Van, Truck, and so on.

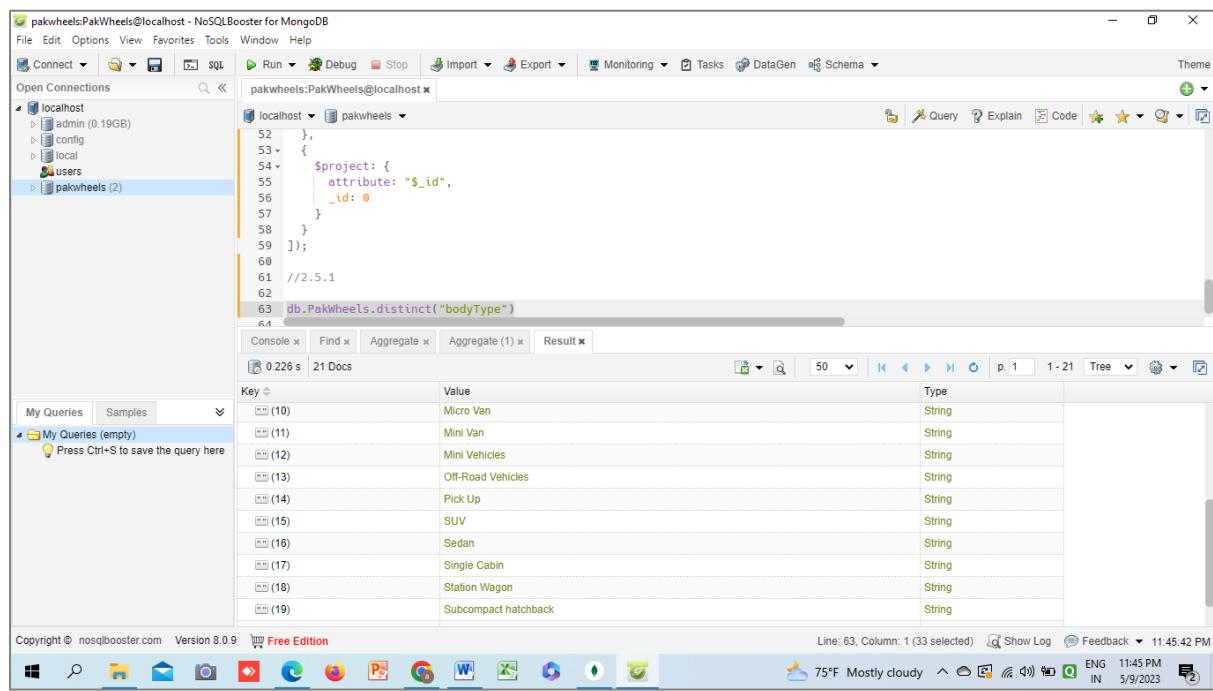


The below screenshots shows the output of query
db.PakWheels.distinct("bodyType")



Screenshot of NoSQLBooster for MongoDB showing the results of the query `db.PakWheels.distinct("bodyType")`. The results are displayed in a table:

Key	Value	Type
(1)	Compact SUV	String
(2)	Compact sedan	String
(3)	Convertible	String
(4)	Coupe	String
(5)	Crossover	String
(6)	Double Cabin	String
(7)	Hatchback	String
(8)	High Roof	String
(9)	MPV	String
(10)	Micro Van	String



Screenshot of NoSQLBooster for MongoDB showing the results of the query `db.PakWheels.distinct("bodyType")`. The results are displayed in a table:

Key	Value	Type
(10)	Micro Van	String
(11)	Mini Van	String
(12)	Mini Vehicles	String
(13)	Off-Road Vehicles	String
(14)	Pick Up	String
(15)	SUV	String
(16)	Sedan	String
(17)	Single Cabin	String
(18)	Station Wagon	String
(19)	Subcompact hatchback	String



pakwheels:PakWheels@localhost - NoSQLBooster for MongoDB

File Edit Options View Favorites Tools Window Help

Connect SQL Run Debug Stop Import Export Monitoring Tasks DataGen Schema Theme

Open Connections

localhost admin (0.19GB) config local users pakwheels (2)

```

52 },
53 {
54 $project: {
55   | attribute: "$_id",
56   | _id: 0
57 }
58 ];
59 //2.5.1
60
61 db.Pakwheels.distinct("bodyType")
62
63

```

Console x Find x Aggregate x Aggregate (1) x Result x

0.226 s 21 Docs

Key	Value	Type
(12)	Mini Vehicles	String
(13)	Off-Road Vehicles	String
(14)	Pick Up	String
(15)	SUV	String
(16)	Sedan	String
(17)	Single Cabin	String
(18)	Station Wagon	String
(19)	Subcompact hatchback	String
(20)	Truck	String
(21)	Van	String

Copyright © nosqlbooster.com Version 8.0.9 Free Edition

Line: 63, Column: 1 (33 selected) Show Log Feedback 11:45:44 PM

Windows Search Mail Camera Task View Start Taskbar Weather ENG 11:45 PM IN 5/9/2023

2.6. Write at least 5 queries using the key-value pairs you found in the previous steps to narrow down the result (provide the queries and results in your report)

For 2.6 we can use these 5 queries

```

db.PakWheels.find({manufacturer: "Honda"}).count()
db.PakWheels.find({price: {$lt: 700000}}).count()
db.PakWheels.find({bodyType: "Compact SUV"}).count()
db.PakWheels.find({"model": "Alpha", "price": {$gt: 1500000}}).count()
db.PakWheels.find({bodyType: "SUV", fuelType: "Petrol"}).count()

```

1. Count of Manufacturer as Honda is **11519**.
2. Count of price less than 700000 are **8346**.
3. Count of bodyType as “Compact SUV” is **98**.
4. Count of model as Alpha and price greater than 1500000 is **1**.
5. Count of bodyType as SUV and fuelType as Petrol is **2766**.

The below screenshots shows the output of above 5 queries:



```
pakwheels:PakWheels@localhost ✘
localhost pakwheels
61 // 2.5.1
62
63 db.PakWheels.distinct("bodyType")
64
65 // 2.6
66
67 db.PakWheels.find({manufacturer: "Honda"}).count()
68 db.PakWheels.find({price: {$lt: 700000}}).count()
69 db.PakWheels.find({bodyType: "Compact SUV"}).count()
70 db.PakWheels.find({model: "Alpha", "price": {$gt: 1500000}}).count()
71 db.PakWheels.find({bodyType: "SUV", fuelType: "Petrol"}).count()
72
```

Result (1) ✘ Result (2) ✘ Result (3) ✘ Result (4) ✘ Result (5) ✘ Result (6) ✘ Result (7) ✘ Result (8) ✘ Result (9) ✘ Result (10) ✘ Result (11) ✘ Result (12) ✘ >>

0.144 s

1 |11519|


```
pakwheels:PakWheels@localhost ✘
localhost pakwheels
61 // 2.5.1
62
63 db.PakWheels.distinct("bodyType")
64
65 // 2.6
66
67 db.PakWheels.find({manufacturer: "Honda"}).count()
68 db.PakWheels.find({price: {$lt: 700000}}).count()
69 db.PakWheels.find({bodyType: "Compact SUV"}).count()
70 db.PakWheels.find({model: "Alpha", "price": {$gt: 1500000}}).count()
71 db.PakWheels.find({bodyType: "SUV", fuelType: "Petrol"}).count()
72
```

Result (2) ✘ Result (3) ✘ Result (4) ✘ Result (5) ✘ Result (6) ✘ Result (7) ✘ Result (8) ✘ Result (9) ✘ Result (10) ✘ Result (11) ✘ Result (12) ✘ Result (13) ✘ >>

0.213 s

1 |8346|


```
pakwheels:PakWheels@localhost ✘
localhost pakwheels
61 // 2.5.1
62
63 db.PakWheels.distinct("bodyType")
64
65 // 2.6
66
67 db.PakWheels.find({manufacturer: "Honda"}).count()
68 db.PakWheels.find({price: {$lt: 700000}}).count()
69 db.PakWheels.find({bodyType: "Compact SUV"}).count()
70 db.PakWheels.find({model: "Alpha", "price": {$gt: 1500000}}).count()
71 db.PakWheels.find({bodyType: "SUV", fuelType: "Petrol"}).count()
72
```

Result (3) ✘ Result (4) ✘ Result (5) ✘ Result (6) ✘ Result (7) ✘ Result (8) ✘ Result (9) ✘ Result (10) ✘ Result (11) ✘ Result (12) ✘ Result (13) ✘ Result (14) ✘ >>

0.164 s

1 |98|


```
pakwheels:PakWheels@localhost ✘
localhost pakwheels
61 // 2.5.1
62
63 db.PakWheels.distinct("bodyType")
64
65 // 2.6
66
67 db.PakWheels.find({manufacturer: "Honda"}).count()
68 db.PakWheels.find({price: {$lt: 700000}}).count()
69 db.PakWheels.find({bodyType: "Compact SUV"}).count()
70 db.PakWheels.find({model: "Alpha", "price": {$gt: 1500000}}).count()
71 db.PakWheels.find({bodyType: "SUV", fuelType: "Petrol"}).count()
72
```

Result (4) ✘ Result (5) ✘ Result (6) ✘ Result (7) ✘ Result (8) ✘ Result (9) ✘ Result (10) ✘ Result (11) ✘ Result (12) ✘ Result (13) ✘ Result (14) ✘ Result (15) ✘ >>

0.108 s

1 |1|



```
pakwheels:PakWheels@localhost ✘
localhost ✘ pakwheels ✘
61 // 2.5.1
62
63 db.PakWheels.distinct("bodyType")
64
65 // 2.6
66
67 db.PakWheels.find({manufacturer: "Honda"}).count()
68 db.PakWheels.find({price: {$lt: 700000}}).count()
69 db.PakWheels.find({bodyType: "Compact SUV"}).count()
70 db.PakWheels.find({model: "Alpha", "price": {$gt: 1500000}}).count()
71 db.PakWheels.find({bodyType: "SUV", fuelType: "Petrol"}).count()
72
Result (5) ✘ Result (6) ✘ Result (7) ✘ Result (8) ✘ Result (9) ✘ Result (10) ✘ Result (11) ✘ Result (12) ✘ Result (13) ✘ Result (14) ✘ Result (15) ✘ Result (16) ✘ >>
0.216 s
1 2766
```

2.7. Write at least 5 update queries to update some of the values in your database (provide the queries and results in your report)

Below are screenshots along with the update queries.

```
db.PakWheels.updateOne({ manufacturer:"Suzuki"},{$set:{colour:"White"}});
```

The screenshot shows the NoSQLBooster interface for MongoDB. The left sidebar lists databases: admin (0.19GB), config, local, users, and pakwheels (2). The main area displays a mongoDB script.js file with the following code:

```
77 db.PakWheels.find({"model": "Alpha", "price": {$gt: 1500000}}).count()
78 db.PakWheels.find({bodyType: "SUV", fuelType: "Petrol"}).count()
79
80 //2.7
81 //Update
82 db.PakWheels.updateOne({ manufacturer:"Suzuki"},{$set:{colour:"White"}});
83 db.PakWheels.updateOne({model:"Mehran"},{$set:{sellerLocation:"Lahore Punjab"}});
84 db.PakWheels.updateOne({name: "Mitsubishi Pajero 1992 for sale in Islamabad"}, {$set:{colour:"Green"}});
85 db.PakWheels.updateOne({bodyType:"Hatchback"}, {$set:{mileageFrom0Dometer:"2000 Km"}});
86 db.PakWheels.updateOne({fuelType:"Petrol"}, {$set:{itemCondition:"new"}});
87
88 //2.8
89
```

The results pane shows the output of the update query:

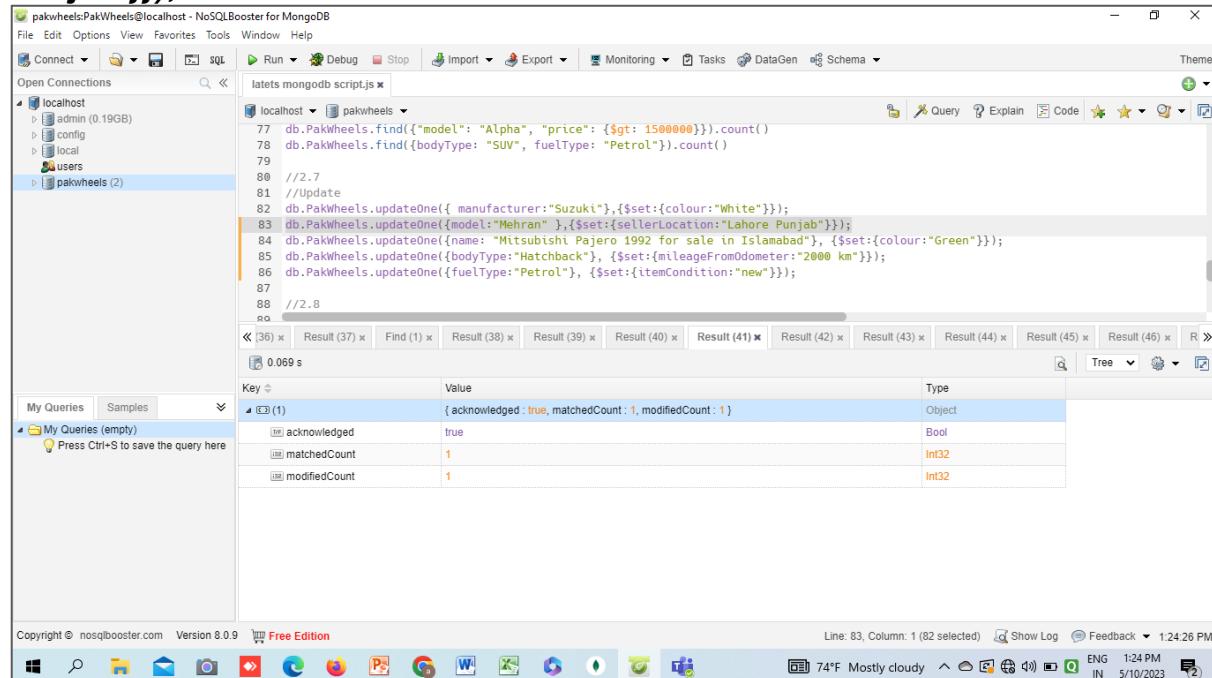
```
Result (37) ✘ Find (1) ✘ Result (38) ✘ Result (39) ✘ Result (40) ✘ Result (41) ✘ Result (42) ✘ Result (43) ✘ Result (44) ✘ Result (45) ✘ Result (46) ✘ >>
0.068 s
```

Key	Type
ACK(1)	Object
acknowledged	Bool
matchedCount	Int32
modifiedCount	Int32

Copyright © nosqlbooster.com Version 8.0.9 Free Edition



db.PakWheels.updateOne({model:"Mehran"},{\$set:{sellerLocation:"Lahore Punjab"}});



```
pkwheels@localhost - NoSQLBooster for MongoDB
File Edit Options View Favorites Tools Window Help
Connect SQL Run Debug Stop Import Export Monitoring Tasks DataGen Schema
Open Connections
localhost
  admin (0.19GB)
  config
  local
  users
  pakwheels (2)

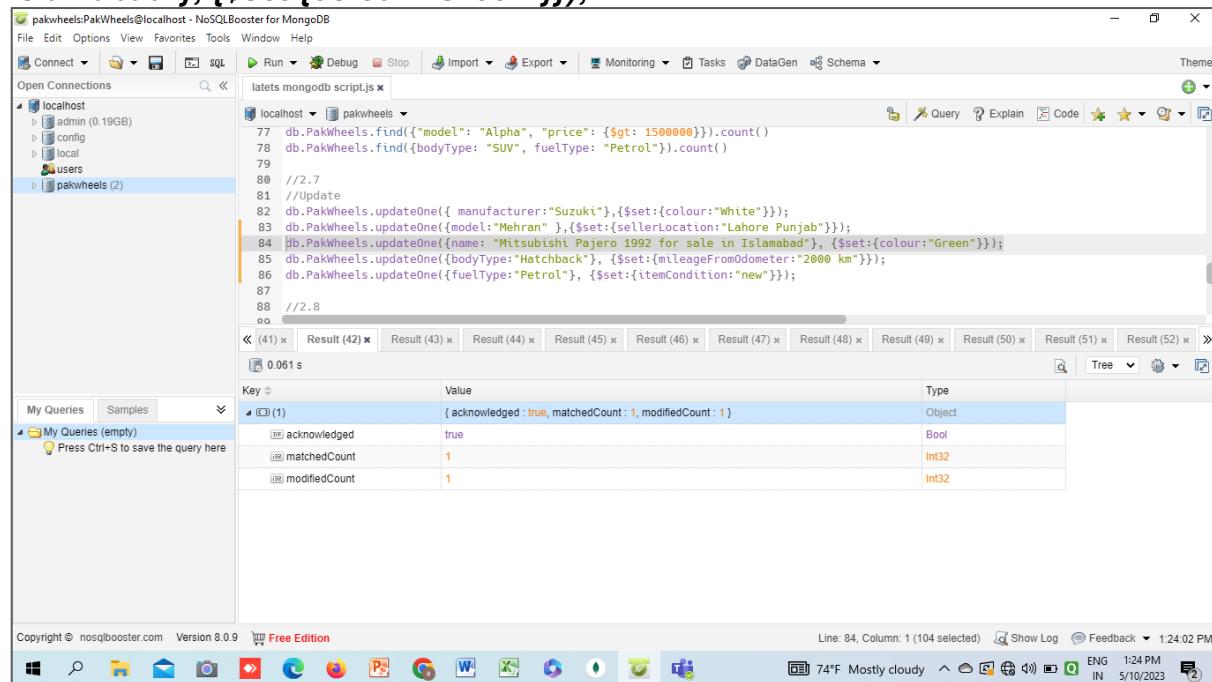
lates mongodb script.js x
77 db.PakWheels.find({ "model": "Alpha", "price": { $gt: 1500000 } }).count()
78 db.PakWheels.find({ bodyType: "SUV", fuelType: "Petrol" }).count()
79
80 //2.7
81 //Update
82 db.PakWheels.updateOne({ manufacturer: "Suzuki" }, { $set: { colour: "White" } });
83 db.PakWheels.updateOne({ model: "Mehran" }, { $set: { sellerLocation: "Lahore Punjab" } });
84 db.PakWheels.updateOne({ name: "Mitsubishi Pajero 1992 for sale in Islamabad" }, { $set: { colour: "Green" } });
85 db.PakWheels.updateOne({ bodyType: "Hatchback" }, { $set: { mileageFromOdometer: "2000 km" } });
86 db.PakWheels.updateOne({ fuelType: "Petrol" }, { $set: { itemCondition: "new" } });
87
88 //2.8
89
90
91 0.069 s
```

Key	Value	Type
ACKNOWLEDGED	{ acknowledged: true, matchedCount: 1, modifiedCount: 1 }	Object
acknowledged	true	Bool
matchedCount	1	Int32
modifiedCount	1	Int32

Copyright © nosqlbooster.com Version 8.0.9 Free Edition Line: 83, Column: 1 (82 selected) Show Log Feedback 1:24:26 PM

Windows Search Mail Camera Internet Explorer Google Chrome Word Excel Powerpoint File Explorer Task View Start 74°F Mostly cloudy ENG 1:24 PM IN 5/10/2023

db.PakWheels.updateOne({name: "Mitsubishi Pajero 1992 for sale in Islamabad"}, {\$set:{colour:"Green"}});



```
pkwheels@localhost - NoSQLBooster for MongoDB
File Edit Options View Favorites Tools Window Help
Connect SQL Run Debug Stop Import Export Monitoring Tasks DataGen Schema
Open Connections
localhost
  admin (0.19GB)
  config
  local
  users
  pakwheels (2)

lates mongodb script.js x
77 db.PakWheels.find({ "model": "Alpha", "price": { $gt: 1500000 } }).count()
78 db.PakWheels.find({ bodyType: "SUV", fuelType: "Petrol" }).count()
79
80 //2.7
81 //Update
82 db.PakWheels.updateOne({ manufacturer: "Suzuki" }, { $set: { colour: "White" } });
83 db.PakWheels.updateOne({ model: "Mehran" }, { $set: { sellerLocation: "Lahore Punjab" } });
84 db.PakWheels.updateOne({ name: "Mitsubishi Pajero 1992 for sale in Islamabad" }, { $set: { colour: "Green" } });
85 db.PakWheels.updateOne({ bodyType: "Hatchback" }, { $set: { mileageFromOdometer: "2000 km" } });
86 db.PakWheels.updateOne({ fuelType: "Petrol" }, { $set: { itemCondition: "new" } });
87
88 //2.8
89
90
91 0.061 s
```

Key	Value	Type
ACKNOWLEDGED	{ acknowledged: true, matchedCount: 1, modifiedCount: 1 }	Object
acknowledged	true	Bool
matchedCount	1	Int32
modifiedCount	1	Int32

Copyright © nosqlbooster.com Version 8.0.9 Free Edition Line: 84, Column: 1 (104 selected) Show Log Feedback 1:24:02 PM

Windows Search Mail Camera Internet Explorer Google Chrome Word Excel Powerpoint File Explorer Task View Start 74°F Mostly cloudy ENG 1:24 PM IN 5/10/2023



**db.PakWheels.updateOne({bodyType:"Hatchback"},
{\$set:{mileageFromOdometer:"2000 km"});**

The screenshot shows the NoSQLBooster interface for MongoDB. The left sidebar lists connections to localhost, including the pakwheels database (selected). The main area contains a script named 'lates mongodb script.js' with the following code:

```
77 db.PakWheels.find({"model": "Alpha", "price": {$gt: 1500000}}).count()
78 db.PakWheels.find({bodyType: "SUV", fuelType: "Petrol"}).count()
79
80 //2.7
81 //Update
82 db.PakWheels.updateOne({ manufacturer: "Suzuki"},{$set:{colour:"White}});
83 db.PakWheels.updateOne({model:"Mehran"},{$set:{sellerLocation:"Lahore Punjab"}});
84 db.PakWheels.updateOne({name: "Mitsubishi Pajero 1992 for sale in Islamabad"}, {$set:{colour:"Green"}});
85 db.PakWheels.updateOne({bodyType:"Hatchback"}, {$set:{mileageFromOdometer:"2000 km"}});
86 db.PakWheels.updateOne({fuelType:"Petrol"}, {$set:{itemCondition:"new"}});
87
88 //2.8
89
```

Below the code, the results of the last query are displayed in a table:

Key	Value	Type
ACK(1)	{ acknowledged: true, matchedCount: 1, modifiedCount: 1 }	Object
acknowledged	true	Bool
matchedCount	1	Int32
modifiedCount	1	Int32

At the bottom of the interface, there is a toolbar with various icons and system status information.

db.PakWheels.updateOne({fuelType:"Petrol"}, {\$set:{itemCondition:"new"});

The screenshot shows the NoSQLBooster interface for MongoDB. The left sidebar lists connections to localhost, including the pakwheels database (selected). The main area contains a script named 'lates mongodb script.js' with the following code:

```
77 db.PakWheels.find({"model": "Alpha", "price": {$gt: 1500000}}).count()
78 db.PakWheels.find({bodyType: "SUV", fuelType: "Petrol"}).count()
79
80 //2.7
81 //Update
82 db.PakWheels.updateOne({ manufacturer: "Suzuki"},{$set:{colour:"White}});
83 db.PakWheels.updateOne({model:"Mehran"},{$set:{sellerLocation:"Lahore Punjab"}});
84 db.PakWheels.updateOne({name: "Mitsubishi Pajero 1992 for sale in Islamabad"}, {$set:{colour:"Green"}});
85 db.PakWheels.updateOne({bodyType:"Hatchback"}, {$set:{mileageFromOdometer:"2000 km"}});
86 db.PakWheels.updateOne({fuelType:"Petrol"}, {$set:{itemCondition:"new"}});
87
88 //2.8
89
```

Below the code, the results of the last query are displayed in a table:

Key	Value	Type
ACK(1)	{ acknowledged: true, matchedCount: 1, modifiedCount: 1 }	Object
acknowledged	true	Bool
matchedCount	1	Int32
modifiedCount	1	Int32

At the bottom of the interface, there is a toolbar with various icons and system status information.



2.8. Write at least 5 queries to insert new documents into your database (provide the queries and results in your report)

Below are screenshots along with the insert queries.

db.PakWheels.insertOne({ name: "Suzuki Cultus 2016 for sale in Lahore", {colour:"Red"}};

The screenshot shows the NoSQLBooster interface for MongoDB. The connection is set to 'localhost' and the database is 'pakwheels'. A script named 'lates mongodb script.js' is running, containing the following code:

```
89
90 //Insert
91 db.PakWheels.insertOne({ name: "Suzuki Cultus 2016 for sale in Lahore", {colour:"Red"}});
92 db.PakWheels.insertOne({model:"Toyota"},{bodyType:"Hatchback"});
93 db.PakWheels.insertOne({ modelDate : "2012" }, {sellerLocation:"Peshawar Punjab"});
94 db.PakWheels.insertOne({itemCondition:" used"}, {mileageFromOdometer:"100,000 km"});
95 db.PakWheels.insertOne({ vehicleTransmission: "Manual"}, {description:"New tires installed recently"});
96
97 //2.9
98 //Delete
99 db.PakWheels.deleteOne({manufacturer:"Honda"});
100 db.PakWheels.deleteOne({manufacturer:"Suzuki"});
```

The results pane shows a single document inserted with the following details:

Key	Type
acknowledged	Bool
insertedId	ObjectID

Copyright © nosqlbooster.com Version 8.0.9 Free Edition Line: 91, Column: 1 (89 selected) Show Log Feedback 1:43:43 PM

db.PakWheels.insertOne({model:"Toyota"},{bodyType:"Hatchback"});

The screenshot shows the NoSQLBooster interface for MongoDB. The connection is set to 'localhost' and the database is 'pakwheels'. A script named 'lates mongodb script.js' is running, containing the following code:

```
89
90 //Insert
91 db.PakWheels.insertOne({ name: "Suzuki Cultus 2016 for sale in Lahore", {colour:"Red"}});
92 db.PakWheels.insertOne({model:"Toyota"},{bodyType:"Hatchback"});
93 db.PakWheels.insertOne({ modelDate : "2012" }, {sellerLocation:"Peshawar Punjab"});
94 db.PakWheels.insertOne({itemCondition:" used"}, {mileageFromOdometer:"100,000 km"});
95 db.PakWheels.insertOne({ vehicleTransmission: "Manual"}, {description:"New tires installed recently"});
96
97 //2.9
98 //Delete
99 db.PakWheels.deleteOne({manufacturer:"Honda"});
100 db.PakWheels.deleteOne({manufacturer:"Suzuki"});
```

The results pane shows a single document inserted with the following details:

Key	Type
acknowledged	Bool
insertedId	ObjectID

Copyright © nosqlbooster.com Version 8.0.9 Free Edition Line: 92, Column: 1 (64 selected) Show Log Feedback 1:43:52 PM



db.PakWheels.insertOne({ modelDate : "2012"}, {sellerLocation:"Peshawar Punjab"});

The screenshot shows the NoSQLBooster interface for MongoDB. The left sidebar lists databases: localhost, admin (0.19GB), config, local, users, and pakwheels (2). The pakwheels database is selected. The main area contains a code editor with the following script:

```
89
90 //Insert
91 db.PakWheels.insertOne({ name: "Suzuki Cultus 2016 for sale in Lahore", {colour:"Red"}});
92 db.PakWheels.insertOne({model:"Toyota"},{bodyType:"Hatchback"});
93 db.PakWheels.insertOne({ modelDate : "2012"}, {sellerLocation:"Peshawar Punjab"});
94 db.PakWheels.insertOne({itemCondition:" used"}, {mileageFromOdometer:"100,000 km"});
95 db.PakWheels.insertOne({ vehicleTransmission: "Manual"}, {description:"New tires installed recently"});
96
97 //2.9
98 //Delete
99 db.PakWheels.deleteOne({manufacturer:"Honda"});
100 db.PakWheels.deleteOne({manufacturer:"Suzuki"});
```

The results pane shows one document inserted:

Key	Type
②(1)	Object
↳ acknowledged	Bool
↳ insertedId	Objectid

Details of the inserted document:

Key	Type
②(1)	Object
↳ acknowledged	Bool
↳ insertedId	Objectid

Copyright © nosqlbooster.com Version 8.0.9 Free Edition Line: 93, Column: 1 (82 selected) Show Log Feedback 1:44:00 PM

db.PakWheels.insertOne({itemCondition:" used"}, {mileageFromOdometer:"100,000 km"});

The screenshot shows the NoSQLBooster interface for MongoDB. The left sidebar lists databases: localhost, admin (0.19GB), config, local, users, and pakwheels (2). The pakwheels database is selected. The main area contains a code editor with the following script:

```
89
90 //Insert
91 db.PakWheels.insertOne({ name: "Suzuki Cultus 2016 for sale in Lahore", {colour:"Red"}});
92 db.PakWheels.insertOne({model:"Toyota"},{bodyType:"Hatchback"});
93 db.PakWheels.insertOne({ modelDate : "2012"}, {sellerLocation:"Peshawar Punjab"});
94 db.PakWheels.insertOne({itemCondition:" used"}, {mileageFromOdometer:"100,000 km"});
95 db.PakWheels.insertOne({ vehicleTransmission: "Manual"}, {description:"New tires installed recently"});
96
97 //2.9
98 //Delete
99 db.PakWheels.deleteOne({manufacturer:"Honda"});
100 db.PakWheels.deleteOne({manufacturer:"Suzuki"});
```

The results pane shows one document inserted:

Key	Type
②(1)	Object
↳ acknowledged	Bool
↳ insertedId	Objectid

Details of the inserted document:

Key	Type
②(1)	Object
↳ acknowledged	Bool
↳ insertedId	Objectid

Copyright © nosqlbooster.com Version 8.0.9 Free Edition Line: 94, Column: 1 (84 selected) Show Log Feedback 1:44:08 PM



db.PakWheels.insertOne({ vehicleTransmission: "Manual"}, {description:"New tires installed recently"});

The screenshot shows the NoSQLBooster interface for MongoDB. The left sidebar lists databases: admin (0.19GB), config, local, users, and pakwheels (2). The main area shows a script named 'latests mongodb script.js' with the following code:

```
89
90 //Insert
91 db.PakWheels.insertOne({ name: "Suzuki Cultus 2016 for sale in Lahore", {colour:"Red"} );
92 db.PakWheels.insertOne({model:"Toyota"},{bodyType:"Hatchback"});
93 db.PakWheels.insertOne({ modelDate : "2012"}, {sellerLocation:"Peshawar Punjab"});
94 db.PakWheels.insertOne({itemCondition:" used"}, {mileageFromOdometer:"100,000 km"});
95 db.PakWheels.insertOne({ vehicleTransmission: "Manual"}, {description:"New tires installed recently"});
```

Below the code, the results of the insert operation are displayed in a table:

Key	Value	Type
ACK(1)	{ acknowledged : true, insertedId : ObjectId('645be5ff1d4b20b940c0b281') }	Object
acknowledged	true	Bool
insertedId	645be5ff1d4b20b940c0b281	ObjectId

At the bottom, the status bar shows 'Copyright © nosqlbooster.com Version 8.0.9 Free Edition' and 'Line: 95, Column: 1 (103 selected) Show Log Feedback 1:44:17 PM'.

2.9. Write at least 5 delete queries to remove documents from your database (provide the queries and results in your report)

Below are screenshots along with the delete queries.

db.PakWheels.deleteMany({manufacturer:"Honda"});

The screenshot shows the NoSQLBooster interface for MongoDB. The left sidebar lists databases: admin (0.19GB), config, local, users, and pakwheels (2). The main area shows a script named 'final script mongodb.js' with the following code:

```
94 db.PakWheels.insertOne({itemCondition:" used"}, {mileageFromOdometer:"100,000 km"});
95 db.PakWheels.insertOne({ vehicleTransmission: "Manual"}, {description:"New tires installed recently"});
96
97 //2.9
98 //Delete
99 db.PakWheels.deleteMany({manufacturer: "Honda"});
100 db.PakWheels.deleteMany({itemCondition: " used"});
101 db.PakWheels.deleteOne({mileageFromOdometer:"100,000 km"});
102 db.PakWheels.deleteOne({bodyType: "SUV"});
103 db.PakWheels.deleteOne({color: "White"});
104
105
```

Below the code, the results of the delete operation are displayed in a table:

Key	Value	Type
ACK(1)	{ acknowledged : true, deletedCount : 11513 }	Object
acknowledged	true	Bool
deletedCount	11,513 (11.5K)	Int32

At the bottom, the status bar shows 'Copyright © nosqlbooster.com Version 8.0.9 Free Edition' and 'Line: 99, Column: 1 (48 selected) Show Log Feedback 3:26:16 PM'.



db.PakWheels.deleteMany({vehicleTransmission:"Manual"});

```
* final script mongodb.js x
localhost pakwheels
94 db.PakWheels.insertOne({itemCondition: "used"}, {mileageFromOdometer: "100,000 km"});
95 db.PakWheels.insertOne({ vehicleTransmission: "Manual"}, {description:"New tires installed recently"});
96
97 //2.9
98 //Delete
99 db.PakWheels.deleteMany({manufacturer:"Honda"});
100 db.PakWheels.deleteMany({vehicleTransmission:"Manual"});
101 db.PakWheels.deleteMany({mileageFromOdometer:"100,000 km"});
102 db.PakWheels.deleteMany({bodyType: "SUV"});
103 db.PakWheels.deleteMany({color: "White"});
104
105
```

Key	Value	Type
ACK(1)	{ acknowledged : true, deletedCount : 12659 }	Object
__ acknowledged	true	Bool
__ deletedCount	12,659 (12.7K)	Int32

Copyright © nosqlbooster.com Version 8.0.9 [Free Edition](#) Line: 100, Column: 1 (56 selected) Show Log Feedback 3:31:59 PM
Windows Taskbar: 76°F Rain showers ENG 3:31 PM IN 5/10/2023

db.PakWheels.deleteMany({mileageFromOdometer:"100,000 km"});

```
* final script mongodb.js x
localhost pakwheels
94 db.PakWheels.insertOne({itemCondition: "used"}, {mileageFromOdometer: "100,000 km"});
95 db.PakWheels.insertOne({ vehicleTransmission: "Manual"}, {description:"New tires installed recently"});
96
97 //2.9
98 //Delete
99 db.PakWheels.deleteMany({manufacturer:"Honda"});
100 db.PakWheels.deleteMany({itemCondition:" used"});
101 db.PakWheels.deleteMany({mileageFromOdometer:"100,000 km"});
102 db.PakWheels.deleteOne({bodyType: "SUV"});
103 db.PakWheels.deleteOne({color: "White"});
104
105
```

Key	Value	Type
ACK(1)	{ acknowledged : true, deletedCount : 2882 }	Object
__ acknowledged	true	Bool
__ deletedCount	2,882 (2.9K)	Int32

Copyright © nosqlbooster.com Version 8.0.9 [Free Edition](#) Line: 101, Column: 1 (60 selected) Show Log Feedback 3:26:59 PM
Windows Taskbar: 76°F Rain showers ENG 3:26 PM IN 5/10/2023



db.PakWheels.deleteMany({bodyType: "SUV"});

```
* final script mongojs.js x
localhost pakwheels
94 db.PakWheels.insertOne({itemCondition: "used"}, {mileageFromOdometer: "100,000 km"});
95 db.PakWheels.insertOne({ vehicleTransmission: "Manual"}, {description:"New tires installed recently"});
96
97 //2.9
98 //Delete
99 db.PakWheels.deleteMany({itemCondition: " used"});
100 db.PakWheels.deleteMany({itemCondition: " used"});
101 db.PakWheels.deleteMany({mileageFromOdometer: "100,000 km"});
102 db.PakWheels.deleteMany({bodyType: "SUV"});
103 db.PakWheels.deleteOne({color: "White"});
104
105
```

Key	Value	Type
ACK(1)	{ acknowledged : true, deletedCount : 3221 }	Object
__ acknowledged	true	Bool
__ deletedCount	3,221 (3.2K)	Int32

db.PakWheels.deleteMany({color: "White"});

```
* final script mongojs.js x
localhost pakwheels
94 db.PakWheels.insertOne({itemCondition: "used"}, {mileageFromOdometer: "100,000 km"});
95 db.PakWheels.insertOne({ vehicleTransmission: "Manual"}, {description:"New tires installed recently"});
96
97 //2.9
98 //Delete
99 db.PakWheels.deleteMany({manufacturer: "Honda"});
100 db.PakWheels.deleteMany({itemCondition: " used"});
101 db.PakWheels.deleteMany({mileageFromOdometer: "100,000 km"});
102 db.PakWheels.deleteMany({bodyType: "SUV"});
103 db.PakWheels.deleteMany({color: "White"});
104
105
```

Key	Value	Type
ACK(1)	{ acknowledged : true, deletedCount : 15912 }	Object
__ acknowledged	true	Bool
__ deletedCount	15,912 (15.9K)	Int32

****End of Document****