DATABASE FOUNDATIONS FOR BUSINESS ANALYTICS
(BUAN 6320)

GROUP 14

Project 2 Report

Submitted By: -
Sahejbir Singh Kumar
Indrajeet S Thakare
Bala Krishna Bobbili
Mano Snigdha Devara
Shreya Shamarthi
Harshitha Reddy Nandikonda

Part 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.

In primary indexing, the index gets stored into B-Trees. Hence, the index type is B-Trees.

```
use sqlproject;
show index from Station;
```

Table	Non_unique	Key_name	Seq_in_index	Column_name	Collation	Cardinality	Sub_part	Packed	Null	Index_type	Comment	Index_comment	Visible	Expression
station	0	PRIMARY	1	Station_id	A	70	NULL	NULL		BTREE			YES	NULL

```
use sqlproject;
show index from status;
```

Table	Non_unique	Key_name	Seq_in_index	Column_name	Collation	Cardinality	Sub_part	Packed	Null	Index_type	Comment	Index_comment	Visible	Expression
status	0	PRIMARY	1	status_id	A	70006066	NULL	NULL		BTREE			YES	NULL
status	1	station_id	1	station_id	A	28190	NULL	NULL	YES	BTREE			YES	NULL

```
use sqlproject;
show index from trip;
```

Table	Non_unique	Key_name	Seq_in_index	Column_name	Collation	Cardinality	Sub_part	Packed	Null	Index_type	Comment	Index_comment	Visible	Expression
trip	0	PRIMARY	1	Trip_id	A	648426	NULL	NULL		BTREE			YES	NULL
trip	1	start_station_id	1	start_station_id	A	72	NULL	NULL	YES	BTREE			YES	NULL
trip	1	end_station_id	1	end_station_id	A	70	NULL	NULL	YES	BTREE			YES	NULL

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

Database management system follows Primary Indexing. Indexes have been automatically created for all the columns in out database. The primary key is automatically indexed because the primary key, index gets stored into B-Trees. Since the primary keys are unique, database management system manages uniqueness by Primary Key Indexing.

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 created a new database (Database B) and copied the data.

```
create table status as (select * from database_a.status);
create table station as (select * from database_a.station);
create table trip as (select * from database_a.trip);
```

We dropped the indexes from the newly created database (Database B) using:

DROP INDEX ‘PRIMARY’ ON STATUS.

Dropped indexes similarly in trip and station table.

Below is the snapshot of database B relations having no indexes.

```
use Database_B;

show index from status;
show index from station;
show index from trip;
```

Filter Rows: Export: Wrap Cell Content:

Table	Non_unique	Key_name	Seq_in_index	Column_name	Collation	Cardinality	Sub_part	Packed	Null	Index_type	Comment	Index_comment	Visible	Expression

Now we have two same databases – database A (with indexes) and database B (without indexes).

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

Query I: Identifying the most popular routes.

```
SELECT count(*) AS "num Trips"
,start_station_name ,end_station_name
FROM trip
group by start_station_name,end_station_name
order by "num Trips" desc;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

num Trips	start_station_name	end_station_name
182	2nd at Folsom	2nd at Folsom
701	2nd at Folsom	2nd at South Park
814	2nd at Folsom	2nd at Townsend
312	2nd at Folsom	5th at Howard
225	2nd at Folsom	Beale at Market
101	2nd at Folsom	Broadway St at Battery St
174	2nd at Folsom	Civic Center BART (7th at Market)
470	2nd at Folsom	Clay at Battery
210	2nd at Folsom	Commercial at Montgomery
115	2nd at Folsom	Davis at Jackson
239	2nd at Folsom	Embarcadero at Bryant
249	2nd at Folsom	Embarcadero at Folsom
206	2nd at Folsom	Embarcadero at Sansome
82	2nd at Folsom	Embarcadero at Vallejo
36	2nd at Folsom	Golden Gate at Polk
189	2nd at Folsom	Grant Avenue at Columbus Avenue
1009	2nd at Folsom	Harry Bridges Plaza (Ferry Building)
377	2nd at Folsom	Howard at 2nd
225	2nd at Folsom	Market at 10th
441	2nd at Folsom	Market at 4th
2459	2nd at Folsom	Market at Sansome
133	2nd at Folsom	Mechanics Plaza (Market at Batte...
92	2nd at Folsom	Post at Kearney

Query II: Identifying the most popular routes in each city.

```
select count(*) AS "num Trips",start_station_name ,end_station_name
from Station left join trip on station.STATION_ID=trip.START_STATION_ID
group by city,start_station_name,end_station_name
order by "num Trips" desc limit 10;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

num Trips	start_station_name	end_station_name
517	San Jose Diridon Caltrain Station	San Jose City Hall
1005	San Jose Diridon Caltrain Station	Paseo de San Antonio
185	San Jose Diridon Caltrain Station	SJSU 4th at San Carlos
2200	San Jose Diridon Caltrain Station	Santa Clara at Almaden
1260	San Jose Diridon Caltrain Station	San Pedro Square
175	San Jose Diridon Caltrain Station	San Jose Diridon Caltrain Station
227	San Jose Diridon Caltrain Station	San Salvador at 1st
805	San Jose Diridon Caltrain Station	MLK Library
634	San Jose Diridon Caltrain Station	Japantown
434	San Jose Diridon Caltrain Station	San Jose Civic Center

Query III: Identifying the stations from where most trips started.

```sql
select s.station_name, count(*) from trip as t join station as s on s.station_id = t.start_station_id
group by start_station_id, end_station_id order by count(*) desc limit 10;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

station_name	count(*)
San Francisco Caltrain 2 (330 Townsend)	6201
Harry Bridges Plaza (Ferry Building)	5660
Townsend at 7th	4994
2nd at Townsend	4768
Harry Bridges Plaza (Ferry Building)	4297
Embarcadero at Sansome	4183
Embarcadero at Folsom	3952
Steuart at Market	3889
2nd at South Park	3613
San Francisco Caltrain (Townsend at 4th)	3571

Query IV: Identifying the stations that have been preferred the least number of times by customers

```sql
select station.station_name, count(*) as count from station
inner join trip on station.station_id = trip.start_station_id
group by station.station_name order by count;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

station_name	count
Redwood City Public Library	198
Franklin at Maple	219
San Mateo County Center	286
Redwood City Medical Center	309
Mezes Park	337
Stanford in Redwood City	482
Park at Olive	710
Santa Clara County Civic Center	848
California Ave Caltrain Station	970
Rengstorff Avenue / California Street	1084
SJSU 4th at San Carlos	1139
Adobe on Almaden	1225
Cowper at University	1337
University and Emerson	1359
Arena Green / SAP Center	1439
SJSU - San Salvador at 9th	1442
San Jose Civic Center	1449
Redwood City Caltrain Station	1521
Evelyn Park and Ride	1644
St James Park	1657
San Salvador at 1st	1660
Ryland Park	1722
San Antonio Shopping Center	1801
Japantown	1854
San Antonio Caltrain Station	1940
Castro Street and El Camino Real	1969
MLK Library	1981
Palo Alto Caltrain Station	2000
Paseo de San Antonio	2180
San Jose City Hall	2358
San Pedro Square	2853

Query V: Identifying the stations that have been preferred the most number of times by customers.

```
select station.station_name, count(*) as count from station
inner join trip on station.station_id = trip.start_station_id
group by station.station_name order by count desc;
```

station_name	count
San Francisco Caltrain (Townsend at 4th)	48583
San Francisco Caltrain 2 (330 Townsend)	33494
Harry Bridges Plaza (Ferry Building)	31248
Temporary Transbay Terminal (Howard at Beale)	25933
Embarcadero at Sansome	25728
2nd at Townsend	25350
Steuart at Market	24220
Market at Sansome	23643
Townsend at 7th	23500
Market at 10th	19815
Market at 4th	19294
2nd at South Park	18308
Powell Street BART	17563
Grant Avenue at Columbus Avenue	15795
2nd at Folsom	15727
Beale at Market	15475
Embarcadero at Bryant	14392
Embarcadero at Folsom	13811
Civic Center BART (7th at Market)	13624
5th at Howard	13210
Howard at 2nd	12972
South Van Ness at Market	12574
Mechanics Plaza (Market at Battery)	11832
Spear at Folsom	11816
Powell at Post (Union Square)	11732
Commercial at Montgomery	11716
Yerba Buena Center of the Arts (3rd @ Howard)	10879
Broadway St at Battery St	10167

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).

Queries have been timed as per the below command:

```
set profiling =1;
select s.station_name, count(*) from trip as t join station as s on s.station_id = t.start_station_id
group by start_station_id, end_station_id order by count(*) desc limit 10;
show profiles;
```

Query_ID	Duration	Query
41	0.00012150	SHOW WARNINGS
42	2.66054625	select s.station_name, count(*) from trip as t jo...
43	0.00031350	set profiling =1
44	0.00009975	SHOW WARNINGS
45	4.25318150	select s.station_name, count(*) from trip as t jo...
46	0.00021975	set profiling =1
47	0.00010475	SHOW WARNINGS
48	3.30922525	select s.station_name, count(*) from trip as t jo...
49	0.00013150	USE `database_a`
50	0.00014800	set profiling =1
51	0.00007900	SHOW WARNINGS
52	3.21731150	select s.station_name, count(*) from trip as t jo...
53	0.00015975	set profiling =1
54	0.00009325	SHOW WARNINGS
55	2.67859525	select s.station_name, count(*) from trip as t jo...

On running these on all queries we get the below average query runtimes:

DATABASES	A	B
QUERY I	8.54602050	11.35684654
QUERY II	3.76957150	5.71239100
QUERY III	4.00908500	4.45617350
QUERY IV	4.21591450	5.00045725
QUERY V	3.57542004	4.14751100

We can clearly see that the database having indexes has faster query runtime.

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

We have created indexes in Database A. Snapshots have been provided below:

```
create index city_idx on station (city);
create index bikes_available_idx on status (bikes_available);
create index subscription_type_idx on trip (subscription_type);
```

city has been indexed from station table.
bikes_available has been indexed from status table.
subscription_type has been indexed from trip table.

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

Query I: Identifying which day of the week has the highest number of trips and the start and end station associated with it.

```
select count(*) as trips_count, dayname(start_date) as day_of_week,
(select station_name from station where station_id = start_station_id) as start_station,
(select station_name from station where station_id = end_station_id) as end_station
from trip where subscription_type = 'Subscriber' group by start_station_id, end_station_id order by
count(*) desc limit 10;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

trips_count	day_of_week	start_station	end_station
6050	Thursday	San Francisco Caltrain 2 (330 Townsend)	Townsend at 7th
4864	Thursday	Townsend at 7th	San Francisco Caltrain (Townsend at 4th)
4319	Friday	2nd at Townsend	Harry Bridges Plaza (Ferry Building)
3926	NULL	Harry Bridges Plaza (Ferry Building)	2nd at Townsend
3815	Thursday	Embarcadero at Folsom	San Francisco Caltrain (Townsend at 4th)
3801	Friday	Embarcadero at Sansome	Steuart at Market
3718	Friday	Steuart at Market	2nd at Townsend
3516	Friday	2nd at South Park	Market at Sansome
3455	Friday	Harry Bridg Harry Bridges Plaza (Ferry Building)	Embarcadero at Sansome
3445	NULL	Temporary Transbay Terminal (Howard at Beale)	San Francisco Caltrain (Townsend at 4th)

Query II: Identifying the busiest station id for the whole week.

```
select station_id, count(*) as trips_count, dayname(times) as day
from status where bikes_available = 0 group by dayname(times) order by count(*);
```

station_id	trips_count	day
4	20873	Sunday
6	29515	Saturday
23	78078	Monday
4	93753	Tuesday
2	96401	Wednesday
6	98028	Friday
7	112356	Thursday

Query III: Getting the details of stations for San Jose city.

```
select * from station where city = "San Jose";
```

Station_id	station_name	lat	long	dock_count	city	installation_date
2	San Jose Diridon Caltrain Station	37.329732	-121.90178200000001	27	San Jose	8/6/2013
3	San Jose Civic Center	37.330698	-121.888979	15	San Jose	8/5/2013
4	Santa Clara at Almaden	37.333988	-121.894902	11	San Jose	8/6/2013
5	Adobe on Almaden	37.331415	-121.8932	19	San Jose	8/5/2013
6	San Pedro Square	37.336721000000004	-121.894074	15	San Jose	8/7/2013
7	Paseo de San Antonio	37.333798	-121.88694299999999	15	San Jose	8/7/2013
8	San Salvador at 1st	37.330165	-121.88583100000001	15	San Jose	8/5/2013
9	Japantown	37.348742	-121.89471499999999	15	San Jose	8/5/2013
10	San Jose City Hall	37.337391	-121.886995	15	San Jose	8/6/2013
11	MLK Library	37.335885	-121.88566000000002	19	San Jose	8/6/2013
12	SJSU 4th at San Carlos	37.332808	-121.88389099999999	19	San Jose	8/7/2013
13	St James Park	37.339301	-121.88993700000002	15	San Jose	8/6/2013
14	Arena Green / SAP Center	37.332692	-121.900084	19	San Jose	8/5/2013
16	SJSU - San Salvador at 9th	37.333954999999996	-121.877349	15	San Jose	8/7/2013
80	Santa Clara County Civic Center	37.352601	-121.90573300000001	15	San Jose	12/31/2013
84	Ryland Park	37.342725	-121.89561699999999	15	San Jose	4/9/2014
NULL	NULL	NULL	NULL	NULL	NULL	NULL

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).

Following the steps done in 1.5, we got the runtime of the “where” queries. Below are the results:

DATABASE	A	B
QUERY I	0.78401250	1.28210207

QUERY II	29.21545304	55.73942405
QUERY III	0.00351800	0.00853052

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

We can clearly see the advantage of indexes in as per our observations from the two databases – one with indexes and vice-versa. Using indexes results in query optimization i.e indexes make execution of queries much faster. Runtime for queries for database A is much less than database B. This is due to the absence of indexes in database B. Indexes expedited the query execution in database A.

Part 2: MongoDB and MQL

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

We have chosen the dataset – PakWheels. The dataset contains Pakistan's largest PakWheels automobile listings. We have explored the dataset and its content and got familiar with it.

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).

Non-relational databases are better to use for unstructured data, they are suitable for both operational and transactional data. This is because non-relational databases eliminated the limitations that come with relational databases like scalability, flexibility, and performance issues. MongoDB is a non-relational database that offers scalability, high performance, reliability, and flexibility. Data is stored in the form of documents. There are no tables (rows and columns), the data varies from record to record.

In our dataset, let us consider the "features" variable which is an array. One record has contained an array of 4 features, and one has an array of 12 features. This is the flexibility provided in the non-relational databases. Also, variable "bodyType" is not present in all records. This is not possible in relational databases.

```
features: Array
    0: "AM/FM Radio"
    1: "Alloy Rims"
    2: "Cassette Player"
    3: "Immobilizer Key"
```

```
features: Array
    0: "AM/FM Radio"
    1: "Air Bags"
    2: "Air Conditioning"
    3: "Alloy Rims"
    4: "Cassette Player"
    5: "Keyless Entry"
    6: "Power Locks"
    7: "Power Mirrors"
    8: "Power Steering"
    9: "Power Windows"
    10: "Sun Roof"
```

Another advantage of non-relational databases is they allow sub-documents. In our dataset, myriad variables like “brand”, “vehicleEngine”, “extraFeatures” have sub-documents which contain their own information (like nested documents). This is not possible in relational databases.

2.3. Import your dataset into MongoDB using the following process (do not use MongoDB Compass as it has a 16MB limit for JSON files).

The instructions have been followed to import the dataset.

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

Some attributes which are common in all documents – “itemCondition , “model”, “manufacturer”, “fuelType”, “sellerLocation” “name”, “decription”, modelDate”, “color” etc.

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

```
> db.PakWheels.distinct("model")
< [
    '1 Series',          '1000',         '120 Y',
    '1200',              '1300',         '200 D',
    '200 T',             '205',          '240 Gd',
    '250 D',             '3 Series',     '300 C',
    '300 Series',        '323',          '350Z',
    '370Z',              '5 Series',     '6 Series',
    '626',               '7 Series',     '86',
    '929',               'A Class',      'A3',
    'A4',                'A5',           'A6',
    'A800',              'AD Van',       'APV',
    'Accent',            'Accord',       'Acty',
    'Aerio',             'Airwave',      'Allion',
    'Almera',            'Alpha',        'Alphard',
    'Alphard Hybrid',    'Alsvin',       'Altezza',
    'Alto',              'Alto Lapin',   'Anglia',
    'Aqua',              'Atenza Wagon', 'Atrai Wagon',
    'Autobiography',     'Avanza',       'Avensis',
    'Aveo',              'Axela',        'Aygo',
    'Azwagon',           'B2200',        'BJ40',
    'BR-V',              'Baleno',       'Beat',
    'Beetle',            'Bego',         'Belta',
    'Besturn',           'Bj212',        'Blue Bird',
    'Bluebird Sylphy',   'Bolan',        'Boltoro',
    'Bongo',             'Boon',         'Brabus ',
```

```
> db.PakWheels.distinct("modelDate")
< [
    1942, 1944, 1952, 1958, 1960, 1961, 1962,
    1963, 1964, 1965, 1966, 1967, 1968, 1969,
    1970, 1971, 1972, 1973, 1974, 1975, 1976,
    1977, 1978, 1979, 1980, 1981, 1982, 1983,
    1984, 1985, 1986, 1987, 1988, 1989, 1990,
    1991, 1992, 1993, 1994, 1995, 1996, 1997,
    1998, 1999, 2000, 2001, 2002, 2003, 2004,
    2005, 2006, 2007, 2008, 2009, 2010, 2011,
    2012, 2013, 2014, 2015, 2016, 2017, 2018,
    2019, 2020, 2021
  ]
```

```
> db.PakWheels.distinct("itemCondition")
< [ 'used' ]
> db.PakWheels.distinct("fuelType")
< [ 'CNG', 'Diesel', 'Electric', 'Hybrid', 'Lpg', 'Petrol' ]
> db.PakWheels.distinct("@type")
< [ 'Car' ]
> db.PakWheels.distinct("vehicleTransmission")
< [ 'Automatic', 'Manual' ]
```

```
> db.PakWheels.distinct("color")
< [
    ' Urban Titanium',
    '-',
    '...',
    '0',
    '14/ 16',
    '3 shaded german colour',
    'A.Silver',
    'AQ Jade',
    'AQ.JADE',
    'AQUA BLUE',
    'AS YOU CAN SEE',
    'AUTOMATIC ',
    'Ac installed working',
    'All colors',
    'Angori',
    'Any colour ',
    'Aqua Blue',
    'Aqua Marine',
    'Aqua blue',
    'Aqua green',
    'Aqua green ',
    'Army ',
    'Ash White',
    'Attitude Black',
```

```
> db.PakWheels.distinct("manufacturer")
< [
    'Adam',        'Audi',         'Austin',       'BAIC',
    'BMW',         'Bentley',      'Cadillac',     'Changan',
    'Chery',       'Chevrolet',    'Chrysler',     'Citroen',
    'Classic Cars', 'DFSK',        'Daehan',       'Daewoo',
    'Daihatsu',    'Datsun',       'Dodge',        'Dongfeng',
    'FAW',         'Fiat',         'Ford',         'GMC',
    'Geely',       'Golden Dragon', 'Haval',       'Hino',
    'Honda',       'Hummer',       'Hyundai',      'Isuzu',
    'JAC',         'JW Forland',   'Jeep',         'KIA',
    'Land Rover',  'Lexus',        'MG',           'MINI',
    'Master',      'Mazda',        'Mercedes Benz', 'Mitsubishi',
    'Nissan',      'Others',       'Peugeot',      'Plymouth',
    'Porsche',     'Prince',       'Proton',       'Range Rover',
    'Renault',     'Roma',         'Saab',         'Skoda',
    'Sogo',        'SsangYong',    'Subaru',       'Suzuki',
    'Tesla',       'Toyota',       'United',       'Vauxhall',
    'Volkswagen',  'Volvo',        'Willys',       'ZOTYE'
  ]
```

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

There are two variables which are not common in all the documents – “bodyType” and “features”. “bodyType” is present in only few documents and “features” has different number of elements stored as an array among different documents.

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

```
> db.PakWheels.distinct("bodyType")
< [
    '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',
    'Single Cabin',         'Station Wagon',
    'Subcompact hatchback', 'Truck',
    'Van'
  ]
```

```
> db.PakWheels.distinct("features")
< [
    'ABS',
    'AM/FM Radio',
    'Air Bags',
    'Air Conditioning',
    'Alloy Rims',
    'CD Player',
    'Cassette Player',
    'Climate Control',
    'CoolBox',
    'Cruise Control',
    'DVD Player',
    'Front Camera',
    'Front Speakers',
    'Heated Seats',
    'Immobilizer Key',
    'Keyless Entry',
    'Navigation System',
    'Power Locks',
    'Power Mirrors',
    'Power Steering',
    'Power Windows',
    'Rear AC Vents',
    'Rear Camera',
    'Rear Seat Entertainment',
```

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).

Query I: Black coloured automobiles with price greater than 1000000, and body type as SUV or Compact SUV.

```
db.PakWheels.find({'price': {$gt: 1000000},
$or: [{'extraFeatures.BodyType': 'SUV'}, {'extraFeatures.BodyType':
'Compact SUV'}],
'extraFeatures.Color': 'Black'}).limit(2).pretty()
```

Output:

```
> db.PakWheels.find({'price': {$gt: 1000000},
  $or: [{'extraFeatures.BodyType': 'SUV'}, {'extraFeatures.BodyType': 'Compact SUV'}],
  'extraFeatures.Color': 'Black'}).limit(2).pretty()
< { _id: ObjectId("6392dee85e6996759fa08993"),
    '@type': 'Car',
    brand: { '@type': 'Brand', name: 'Toyota' },
    model: 'Prado',
    description: 'TXL 2016 5 Seater Sunroof \r\nImport July 2021 \r\n4.5 grade with verifiable auction sheet ',
    itemCondition: 'used',
    modelDate: 2016,
    manufacturer: 'Toyota',
    fuelType: 'Petrol',
    name: 'Toyota Prado 2016 for sale in Lahore',
    image: 'https://cache2.pakwheels.com/ad_pictures/5371/toyota-prado-tx-limited-3-2016-53716621.jpg',
    vehicleTransmission: 'Automatic',
    color: 'Black',
    bodyType: 'SUV',
    vehicleEngine: { '@type': 'EngineSpecification', engineDisplacement: '2700cc' },
    mileageFromOdometer: '52,000 km',
    sellerLocation: ' DHA Defence, Lahore Punjab',
    postedFrom: ' Added via Phone',
    keywords: 'toyota prado 2016 for sale, toyota prado 2016, toyota prado 2016 lahore, 2016 toyota prado, toyota prado for sale, used toyota prado 2016',
    extraFeatures:
     { RegisteredIn: 'Un-Registered',
       Color: 'Black',
       Assembly: 'Imported',
```

Query II: Black or White coloured automobiles (cars) of either Suzuki or Toyota brand with seller location - Hyderabad.

```
db.PakWheels.find({'sellerLocation': {$regex: 'Hyderabad'},
$or: [{'extraFeatures.Color':'White'}, {'extraFeatures.Color':'Black'}],
$or: [{'brand.name':'Suzuki'}, {'brand.name':'Toyota'}]}).limit(3).pretty()
```

Output:

```
> db.PakWheels.find({'sellerLocation': {$regex: 'Hyderabad'},
  $or: [{'extraFeatures.Color':'White'}, {'extraFeatures.Color':'Black'}],
  $or: [{'brand.name':'Suzuki'}, {'brand.name':'Toyota'}]}).limit(3).pretty()
< { _id: ObjectId("6392dee85e6996759fa08a56"),
    '@type': 'Car',
    brand: { '@type': 'Brand', name: 'Toyota' },
    model: 'Corolla',
    description: 'Toyota Corolla Gli 2015 model Silver color 95000 original mileage minor touchups in Sides due to scratches but seal are in Geniune condition No major Or mi
    itemCondition: 'used',
    modelDate: 2015,
    manufacturer: 'Toyota',
    fuelType: 'Petrol',
    name: 'Toyota Corolla 2015 for sale in Hyderabad',
    image: 'https://cache2.pakwheels.com/ad_pictures/5661/toyota-corolla-gli-vvti-2015-56611654.jpg',
    vehicleTransmission: 'Manual',
    color: 'Silver',
    bodyType: 'Sedan',
    vehicleEngine: { '@type': 'EngineSpecification', engineDisplacement: '1300cc' },
    mileageFromOdometer: '95,000 km',
    sellerLocation: ' Latifabad, Hyderabad Sindh',
    postedFrom: ' Added via Phone',
    keywords: 'toyota corolla 2015 for sale, toyota corolla 2015, toyota corolla 2015 hyderabad, 2015 toyota corolla, toyota corolla for sale, used toyota corolla 2015',
    extraFeatures:
     { RegisteredIn: 'Sindh',
       Color: 'Silver',
       Assembly: 'Local',
```

Query III: Automobiles released after 2010, fuel type being either Electric or Hybrid.

```
db.PakWheels.find({'modelDate': {$gt: 2010},
$or: [{'fuelType': 'Hybrid'}, {'fuelType': 'Electric'}]}).limit(5).pretty()
```

Output:

```
> db.PakWheels.find({'modelDate': {$gt: 2010},
  $or: [{'fuelType': 'Hybrid'}, {'fuelType': 'Electric'}]}).limit(5).pretty()
< { _id: ObjectId("6392dee85e6996759fa08997"),
    '@type': 'Car',
    brand: { '@type': 'Brand', name: 'Toyota' },
    model: 'Corolla',
    description: 'Toyota Corolla touring ,pearl white,1800 cc hybrid , fresh import , verifiable auction sheet available, heated seats + steering,cruise control with no l
    itemCondition: 'used',
    modelDate: 2019,
    manufacturer: 'Toyota',
    fuelType: 'Hybrid',
    name: 'Toyota Corolla 2019 for sale in Rawalpindi',
    image: 'https://cache3.pakwheels.com/ad_pictures/5218/toyota-corolla-2019-52180762.jpg',
    vehicleTransmission: 'Automatic',
    color: 'White',
    vehicleEngine: { '@type': 'EngineSpecification', engineDisplacement: '1800cc' },
    mileageFromOdometer: '10,000 km',
    sellerLocation: ' Chaklala Scheme, Rawalpindi Punjab',
    postedFrom: ' Added via Phone',
    keywords: 'toyota corolla 2019 for sale, toyota corolla 2019, toyota corolla 2019 rawalpindi, 2019 toyota corolla, toyota corolla for sale, used toyota corolla 2019',
    extraFeatures:
     { RegisteredIn: 'Un-Registered',
       Color: 'White',
       Assembly: 'Imported',
       EngineCapacity: '1800 cc',
       BodyType: 'N/A',
```

Query IV: Unregistered automobiles which are posted from website and having the following extra features – unregistered, assembly: imported.

```
db.PakWheels.find({'postedFrom':'Added via Website',
'extraFeatures.RegisteredIn':'Un-
Registered','extraFeatures.Assembly':'Imported'}).limit(1).pretty()
```

Output:

```
> db.PakWheels.find({'postedFrom':'Added via Website',
  'extraFeatures.RegisteredIn':'Un-Registered','extraFeatures.Assembly':'Imported'}).limit(1).pretty()
< { _id: ObjectId("6392dee85e6996759fa089a7"),
    '@type': 'Car',
    brand: { '@type': 'Brand', name: 'Toyota' },
    model: 'Corolla Fielder',
    description: '4 grade car WXB model full option....  auction sheet available and verified fresh import 2021 bumper to bumper genuine pearl white  colour leather seats low mi
    itemCondition: 'used',
    modelDate: 2018,
    manufacturer: 'Toyota',
    fuelType: 'Petrol',
    name: 'Toyota Corolla Fielder 2018 for sale in Islamabad',
    image: 'https://cache3.pakwheels.com/ad_pictures/5404/toyota-corolla-fielder-hybrid-g-w-b-2018-54045267.jpg',
    vehicleTransmission: 'Automatic',
    color: 'White',
    bodyType: 'Station Wagon',
    vehicleEngine: { '@type': 'EngineSpecification', engineDisplacement: '1500cc' },
    mileageFromOdometer: '44,000 km',
    sellerLocation: ' Islamabad Islamabad',
    postedFrom: 'Added via Website',
    keywords: 'toyota corolla fielder 2018 for sale, toyota corolla fielder 2018, toyota corolla fielder 2018 islamabad, 2018 toyota corolla fielder, toyota corolla fielder for
    extraFeatures:
     { RegisteredIn: 'Un-Registered',
       Color: 'White',
       Assembly: 'Imported',
       EngineCapacity: '1500 cc',
```

Query V: Manual automobiles (cars) launched before 2010 with fuel type being one of them – Petrol/Diesel/LPG.

```
db.PakWheels.find({'modelDate': {$lt: 2010},
$or: [{'fuelType': 'Petrol'}, {'fuelType': 'Diesel'}, {'fuelType':
'Lpg'}]}).limit(2).pretty()
```

Output:

```
> db.PakWheels.find({'modelDate': {$lt: 2010},
  $or: [{'fuelType': 'Petrol'}, {'fuelType': 'Diesel'}, {'fuelType': 'Lpg'}]}).limit(2).pretty()
< { _id: ObjectId("6392dee85e6996759fa0897f"),
    '@type': 'Car',
    brand: { '@type': 'Brand', name: 'Mitsubishi' },
    model: 'Pajero',
    description: 'Read Full Discription First\n\nIts Super Select ( 4×4 ) Edition\n\n92 Model 2007 Custom Auction 2008 Registered\n\nSeal To Seal Nut To Nut Genuine\n\nNo Ru
    itemCondition: 'used',
    modelDate: 1992,
    manufacturer: 'Mitsubishi',
    fuelType: 'Petrol',
    name: 'Mitsubishi Pajero 1992 for sale in Islamabad',
    image: 'https://cache2.pakwheels.com/ad_pictures/5543/mitsubishi-pajero-exceed-3-5-1992-55434056.jpg',
    vehicleTransmission: 'Automatic',
    color: 'Silver',
    bodyType: 'SUV',
    vehicleEngine: { '@type': 'EngineSpecification', engineDisplacement: '3500cc' },
    mileageFromOdometer: '82,400 km',
    sellerLocation: ' Airport Enclave, Islamabad Islamabad',
    postedFrom: ' Added via Phone',
    keywords: 'mitsubishi pajero 1992 for sale, mitsubishi pajero 1992, mitsubishi pajero 1992 islamabad, 1992 mitsubishi pajero, mitsubishi pajero for sale, used mitsubishi
    extraFeatures:
     { RegisteredIn: 'Karachi',
       Color: 'Silver',
       Assembly: 'Imported',
       EngineCapacity: '3500 cc',
```

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).

Query I: Update one car color to Red, for Suzuki, models from one of Mehran/MR Wagon/Wagon R, having Grey color.

```
db.PakWheels.updateOne({'manufacturer': 'Suzuki','color': 'Grey', $or:
[{'model': 'Mehran'}, {'model': 'MR Wagon'}, {'model': 'Wagon R'}]
}, {$set: {'color': 'Red'}})
```

Output:

```
> db.PakWheels.updateOne({'manufacturer': 'Suzuki','color': 'Grey', $or: [{'model': 'Mehran'}, {'model': 'MR Wagon'}, {'model': 'Wagon R'}]
  }, {$set: {'color': 'Red'}})
< { acknowledged: true,
    insertedId: null,
    matchedCount: 1,
    modifiedCount: 1,
    upsertedCount: 0 }
```

Query II: Update the “Posted From” information from NA/NULL to “Added via Phone” for Suzuki or Toyota cars.

```
db.PakWheels.updateMany({'postedFrom': null, $or: [{'brand.name':'Suzuki'},
{'brand.name':'Toyota'}]},
{$set: {'postedFrom': 'Added via Phone'}})
```

Output:

```
> db.PakWheels.updateMany({'postedFrom': null, $or: [{'brand.name':'Suzuki'}, {'brand.name':'Toyota'}]},
  {$set: {'postedFrom': 'Added via Phone'}})
< { acknowledged: true,
    insertedId: null,
    matchedCount: 2,
    modifiedCount: 2,
    upsertedCount: 0 }
```

Query III: For cars sold in Islamabad with imported assembly and engine displacement of 3500cc, update the assembly to Local.

```
db.PakWheels.updateOne({'sellerLocation': {$regex: 'Islamabad'},
'vehicleEngine.engineDisplacement':'3500cc',
'extraFeatures.Assembly':'Imported'}, {$set:
{'extraFeatures.Assembly': 'Local'}})
```

Output:

```
> db.PakWheels.updateOne({'sellerLocation': {$regex: 'Islamabad'},
  'vehicleEngine.engineDisplacement':'3500cc', 'extraFeatures.Assembly':'Imported'}, {$set:
  {'extraFeatures.Assembly': 'Local'}})
< { acknowledged: true,
    insertedId: null,
    matchedCount: 1,
    modifiedCount: 1,
    upsertedCount: 0 }
```

Query IV: Unset the body type for cars having fuel type - CNG, Engine Displacement – 1500cc, prices – greater than 1000000

```
db.PakWheels.updateMany({'fuelType': 'CNG',
'vehicleEngine.engineDisplacement': '1500cc', 'price': {$gt:
1000000}}, {$unset: {'bodyType': 1}})
```

Output:

```
> db.PakWheels.updateMany({'fuelType': 'CNG', 'vehicleEngine.engineDisplacement': '1500cc', 'price': {$gt:
  1000000}}, {$unset: {'bodyType': 1}})
< { acknowledged: true,
    insertedId: null,
    matchedCount: 9,
    modifiedCount: 9,
    upsertedCount: 0 }
```

Query V: For Toyota cars released before 2010 having automatic transmission, set the vehicle transmission to manual.

```
db.PakWheels.updateMany({'modelDate': {$lt: 2010}, 'vehicleTransmission':
'Automatic', 'brand.name':
'Toyota'}, {$set: {'vehicleTransmission': 'Manual'}})
```

Output:

```
> db.PakWheels.updateMany({'modelDate': {$lt: 2010}, 'vehicleTransmission': 'Automatic', 'brand.name':
  'Toyota'}, {$set: {'vehicleTransmission': 'Manual'}})
< { acknowledged: true,
    insertedId: null,
    matchedCount: 3389,
    modifiedCount: 3389,
    upsertedCount: 0 }
```

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

Query I:

```
> db.PakWheels.insertOne({
   "@type" : "Car",
   "brand": {"@type":"Brand", "name": "Audi"},
   "model":"A5",
   "itemCondition": "used",
   "modelDate": 2021,
   "manufacturer":"Audi",
   "fuelType":"Diesel",
   "vehicleTransmission":"Automatic",
   "color":"Blue",
   "bodyType":"SUV",
   "vehicleEngine":{"@type":"EngineSpecification", "engineDisplacement":"1500cc"}, "mileageFromOdometer":"48,500 km",
   "postedFrom":"Added via Website",
   "extraFeatures":{"RegisteredIn":"Islamabad", "Color":"Blue", "Assembly":"Imported", "EngineCapacity":"1500 cc", "BodyType":"SUV", "LastUpdated:":"Feb 27, 2022", "AdRef#":"5269998"},
   "features":["Air Conditioning", "Alloy Rims", "CD Player"],
   "price":9548000,
   "priceCurrency":"PKR"
 })
< { acknowledged: true,
   insertedId: ObjectId("63970f7993649e47e756b60e") }
```

Query II:

```
> db.PakWheels.insertOne({
   "@type" : "Car",
   "brand": {"@type":"Brand", "name": "Honda"},
   "model":"Civic",
   "itemCondition": "used",
   "modelDate": 2016,
   "manufacturer":"Honda",
   "fuelType":"Diesel",
   "vehicleTransmission":"Automatic",
   "color":"White",
   "bodyType":"Sedan",
   "vehicleEngine":{"@type":"EngineSpecification", "engineDisplacement":"1300cc"}, "mileageFromOdometer":"40,530 km",
   "postedFrom":"Added via Phone",
   "extraFeatures":{"RegisteredIn":"Karachi", "Color":"White", "Assembly":"Imported", "EngineCapacity":"1300 cc", "BodyType":"Sedan", "LastUpdated:":"Mar 16, 2017", "AdRef#":"5216978"},
   "features":["Power Locks", "Alloy Rims", "Keyless Entry"],
   "price":6540000,
   "priceCurrency":"PKR"
 })
< { acknowledged: true,
   insertedId: ObjectId("639711eb93649e47e756b60f") }
```

Query III:

```
> db.PakWheels.insertOne({
    "@type" : "Car",
    "brand": {"@type":"Brand", "name": "Mazda"},
    "model":"CX 5",
    "itemCondition": "used",
    "modelDate": 2017,
    "manufacturer":"Mazda",
    "fuelType":"CNG",
    "vehicleTransmission":"Manual",
    "color":"Grey",
    "bodyType":"Compact Sedan",
    "vehicleEngine":{"@type":"EngineSpecification", "engineDisplacement":"1000cc"}, "mileageFromOdometer":"36,000 km",
    "postedFrom":"Added via Website",
    "extraFeatures":{"RegisteredIn":"Jauharabad", "Color":"Grey", "Assembly":"Imported", "EngineCapacity":"1300 cc", "BodyType":"Compact Sedan", "LastUpdated:":"Mar 16, 2017", "AdRef#":"4937978"
    "features":["Air Bags", "ABS", "AM/FM Radio"],
    "price":3210000,
    "priceCurrency":"PKR"
  })
< { acknowledged: true,
    insertedId: ObjectId("639714bb93649e47e756b610") }
```

Query IV:

```
> db.PakWheels.insertOne({
    "@type" : "Car",
    "brand": {"@type":"Brand", "name": "Nissan"},
    "model":"Sunny",
    "itemCondition": "used",
    "modelDate": 2012,
    "manufacturer":"Nissan",
    "fuelType":"Petrol",
    "vehicleTransmission":"Manual",
    "color":"Black",
    "bodyType":"Sedan",
    "vehicleEngine":{"@type":"EngineSpecification", "engineDisplacement":"1498cc"},
    "extraFeatures":{"RegisteredIn":"Hyderabad", "Color":"Black", "Assembly":"Imported", "EngineCapacity":"1498 cc", "AdRef#":"8136978"},
    "features":["Automatic Climate Control","Air Bags", "Anti Lock Braking System"],
    "price":5914000,
    "priceCurrency":"PKR"
  })
< { acknowledged: true,
    insertedId: ObjectId("639717c493649e47e756b611") }
```

Query V:

```
> db.PakWheels.insertMany(
    [{
    "@type" : "Car",
    "brand": {"@type":"Brand", "name": "Chevrolet"},
    "model":"Spark",
    "itemCondition": "used",
    "modelDate": 2020,
    "manufacturer":"Chevrolet",
    "fuelType":"Diesel",
    "vehicleTransmission":"Automatic",
    "color":"Red",
    "bodyType":"SUV",
    "vehicleEngine":{"@type":"EngineSpecification", "engineDisplacement":"1400cc", "mileageFromOdometer":"39,000 km"},
    "extraFeatures":{"RegisteredIn":"Kahuta", "Color":"Red", "Assembly":"Imported", "EngineCapacity":"1400 cc", "LastUpdated:":"Dec 12, 2021", "AdRef#":"1574268"},
    "features":["Alloy Rims","Air Bags", "Accent-color mirror caps"],
    "price":2540000,
    "priceCurrency":"PKR"
  },
  {"@type" : "Car",
    "brand": {"@type":"Brand", "name": "Mitsubishi"},
    "model":"Outlander",
    "itemCondition": "used",
    "modelDate": 2015,
    "manufacturer":"Mitsubishi",
    "fuelType":"Petrol",
    "vehicleTransmission":"Manual",
  "color":"White",
  "bodyType":"SUV",
  "vehicleEngine":{"@type":"EngineSpecification", "engineDisplacement":"2500cc"}, "mileageFromOdometer":"60,000 km",
  "postedFrom":"Added via Phone",
  "extraFeatures":{"RegisteredIn":"Islamabad", "Color":"White", "Assembly":"Imported", "EngineCapacity":"2500 cc", "BodyType":"SUV", "LastUpdated:":"Sep 15, 2019", "AdRef#":"5215998"},
  "features":["Assist Grips"],
  "price":8120000,
  "priceCurrency":"PKR"
  }
  ]

)
{ acknowledged: true,
  insertedIds:
   { '0': ObjectId("63971dba93649e47e756b612"),
     '1': ObjectId("63971dba93649e47e756b613") } }
```

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

Query I:

```
> db.PakWheels.deleteOne({"color":"Grey"})
< { acknowledged: true, deletedCount: 1 }
```

Query II:

```
> db.PakWheels.deleteMany({$or:[{"manufacturer":"Audi"},{"manufacturer":"Nissan"}]})
< { acknowledged: true, deletedCount: 1256 }
```

Query III:

```
> db.PakWheels.deleteOne({'modelDate': {$lt:2010},'vehicleTransmission':'Manual', 'brand.name': 'Suzuki'})
< { acknowledged: true, deletedCount: 1 }
```

Query IV:

```
> db.PakWheels.deleteMany({"bodyType":"Sedan"})
< { acknowledged: true, deletedCount: 22123 }
```

Query V:

```
> db.PakWheels.deleteMany({'postedFrom': 'Added via Phone',$or: [{'brand.name':'Suzuki'}, {'brand.name': 'Toyota'}], 'features':[]})
< { acknowledged: true, deletedCount: 2 }
```