Phase 2 of NYC Taxi dataset Analysis

Group 35
Abhishek Gawali
Akhil Shetty
Deepika Kini
Shounak Desai

Phase II

1. Propose a document-oriented model for your dataset and compare it with your relational model.You should also provide code to load your data into this model.

2. Provide a program that issues at least five interesting SQL queries over the previous relational model and propose indexes to speed up query execution (report your timings).

3. Discover and explain functional dependencies and discuss normalization with respect to the relational model you provided in Phase I.

4. Documentation

Code files for the project:

- Mongo:
  mongoimport : *mongo_import_commands.txt*
  Cleaning tables in mongo: *mongo_cleaning_commands.txt*
- SQL :
  Queries:
  index:

1. Document oriented model:

We are using Mongo for this project phase as a document based nosql database

A) Comparison between mongo and sql database:

Mongo Advantages:

- unrequired fields can be removed for some objects (reduces storage)
- Different datatypes allowed for same column
- Useful if data is in json format and has subdocument (objects within objects)
- Primary key already

Issue using mongo for our dataset:

Ratecode id and are wrongly classified as float but are integers throughout (LocationIDs are correctly classified as Int)

SQL advantage: can set datatype and get an error if data quality issue in datatype

We have no major advantage of using mongo(nosql) since our data is structured

B) Steps to load data:

Step 1: created database in the mongo : NYC_Taxi

Step 2: imported the csv datasets (converted from parquet in phase1) of green, yellow and LV_Hire using mongoimport commands

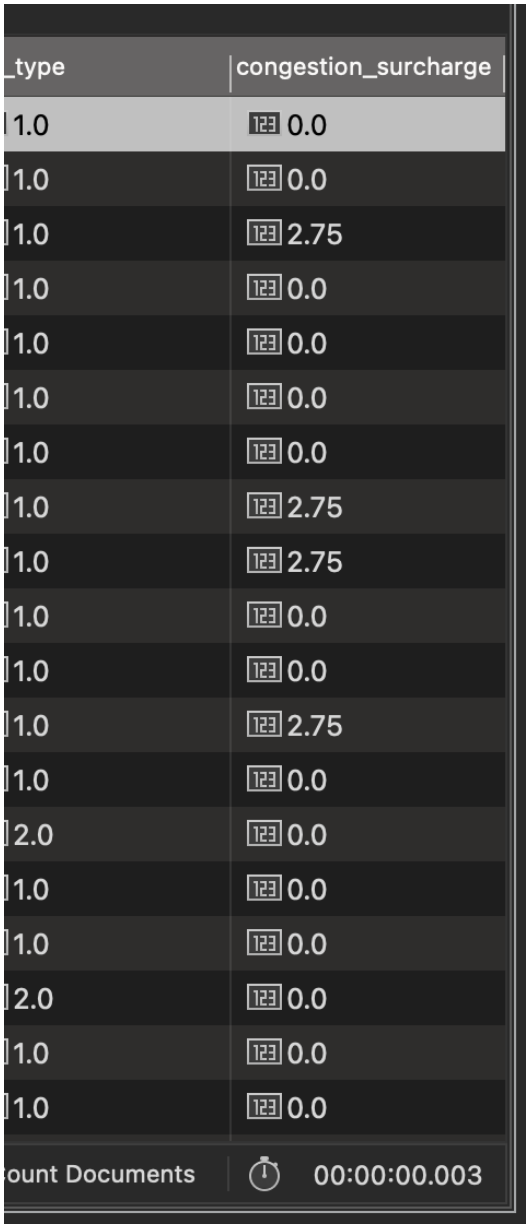
Step 3: imported the static files: ratecode, payment, trip type, vendor,location, HL Vendor and Base file using mongoimport commands

Step 4: cleaned data:

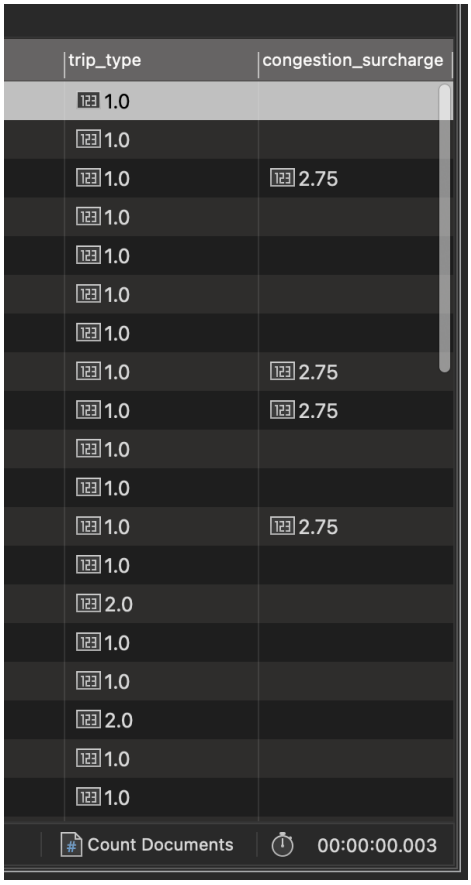
Used below query to remove data for rows in columns with null/unnecessary values

Removed quotes from some columns in location_lookup

```
db.green_cab.updateMany({congestion_surcharge: 0.0}, { $unset : {
congestion_surcharge: 0 }})
```

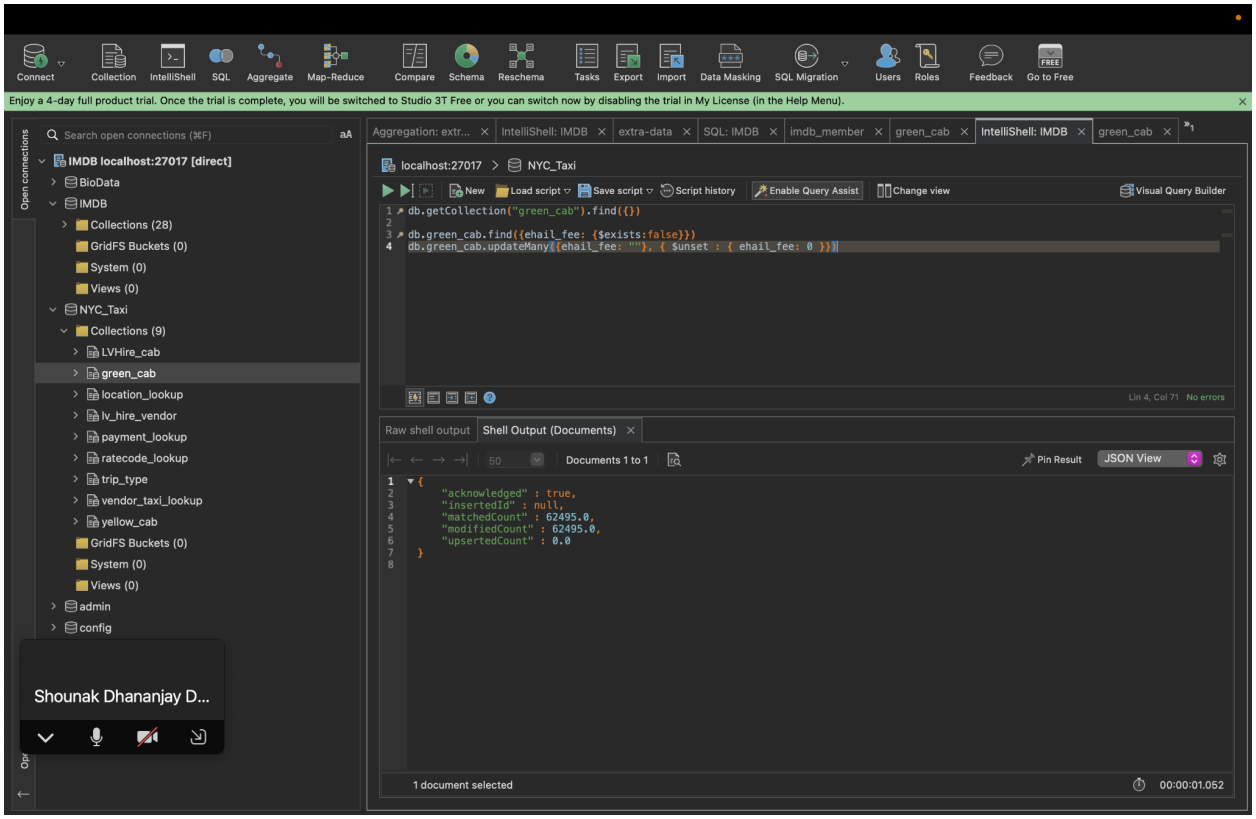


Before



After

Similarly done for many other columns

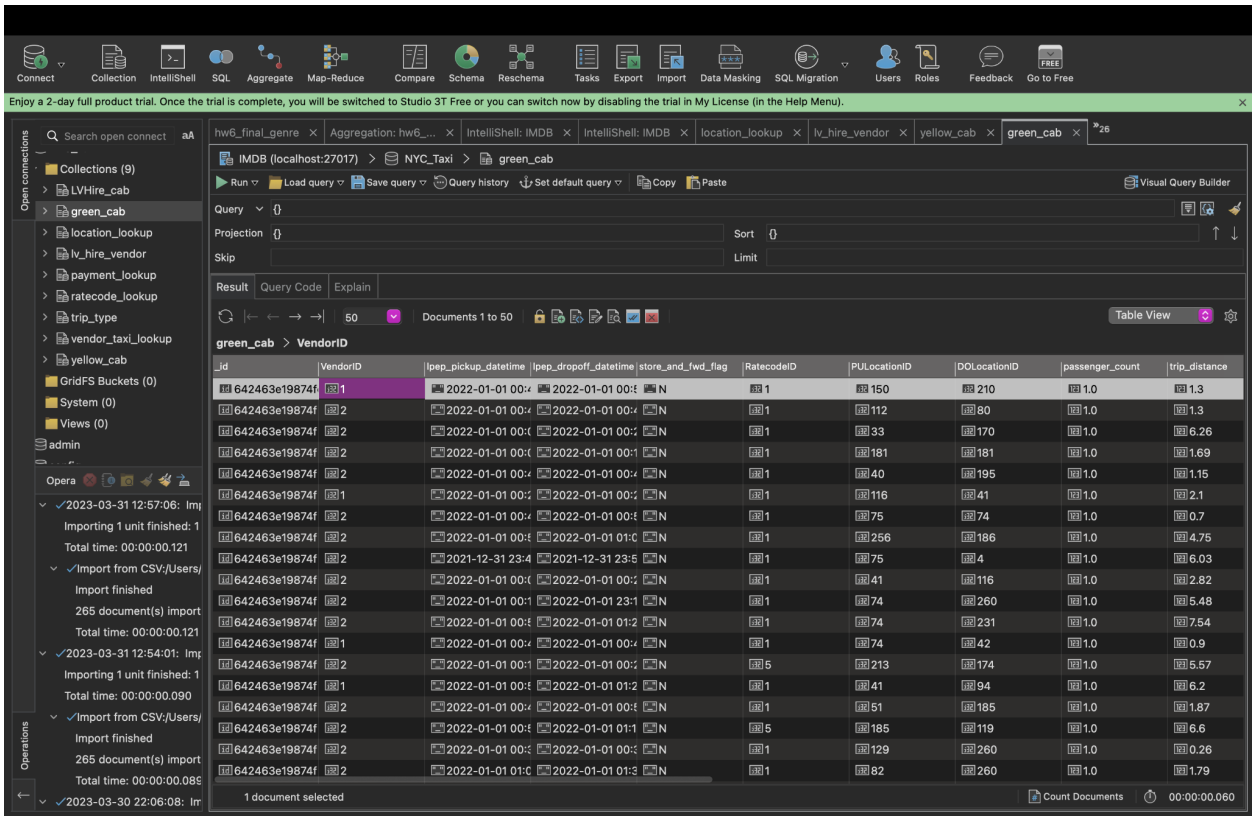


Step 5: changed datatype for some fields that were wrongly classified as decimal (foreign key fields)
Used the command:
db.green_cab.updateMany({trip_type : { $type: 1 }},[{**$set: { trip_type: { $toInt:** "$trip_type" }}}])

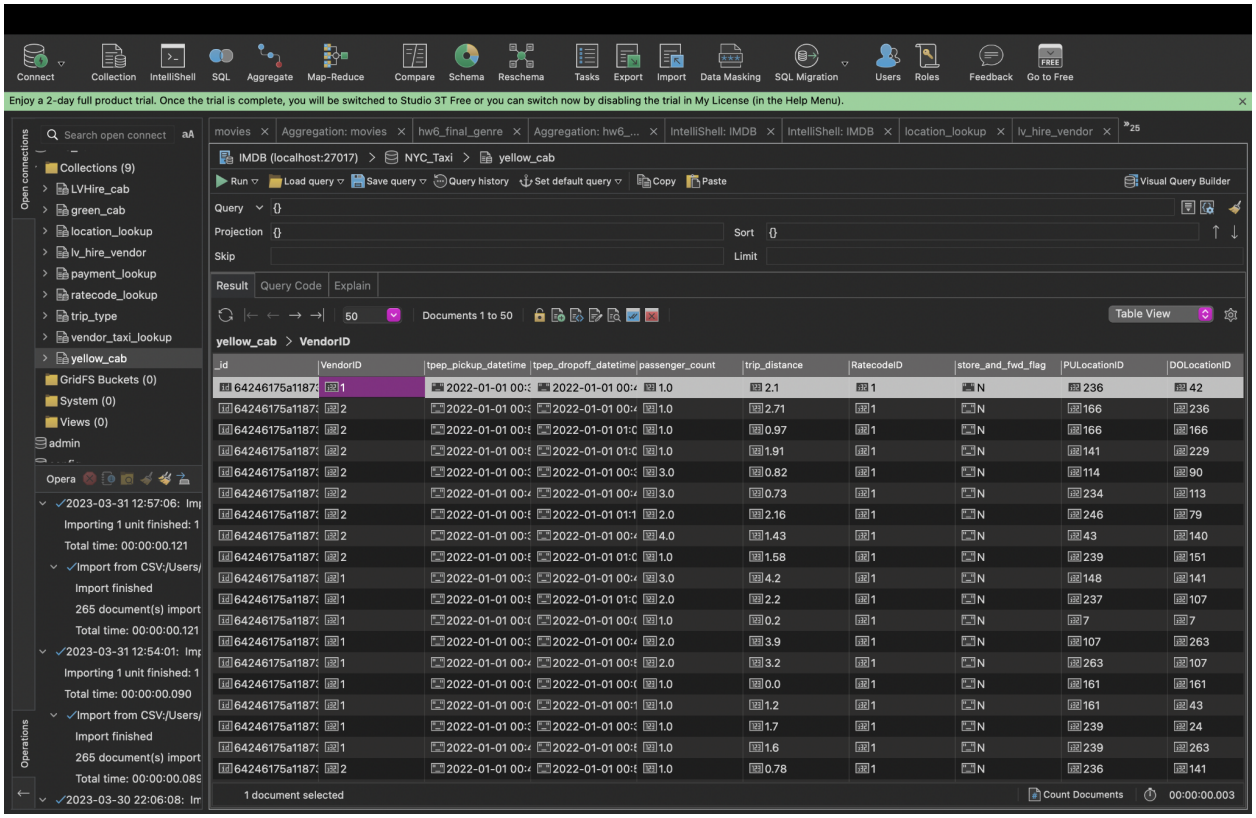
Please check the txt files for all commands

After importing and cleaning, the files look as follows:

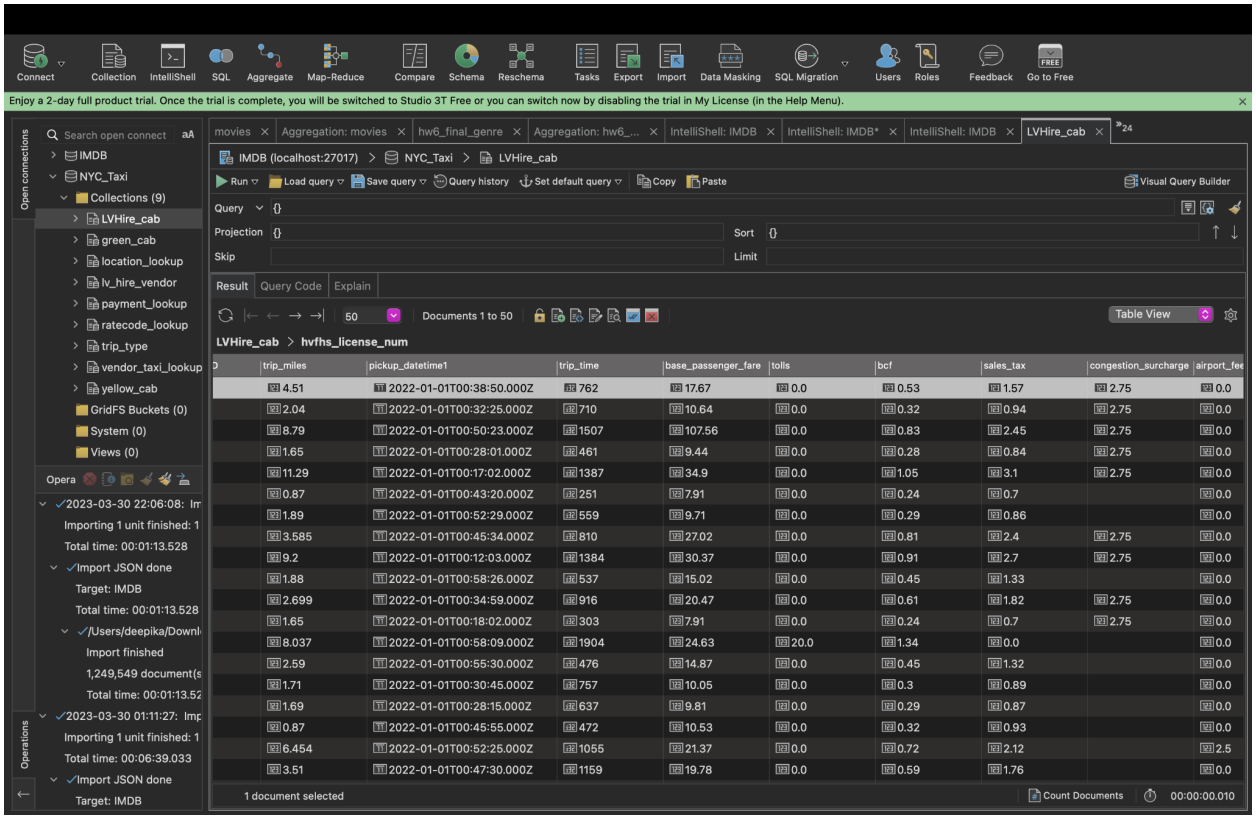
Green_cab



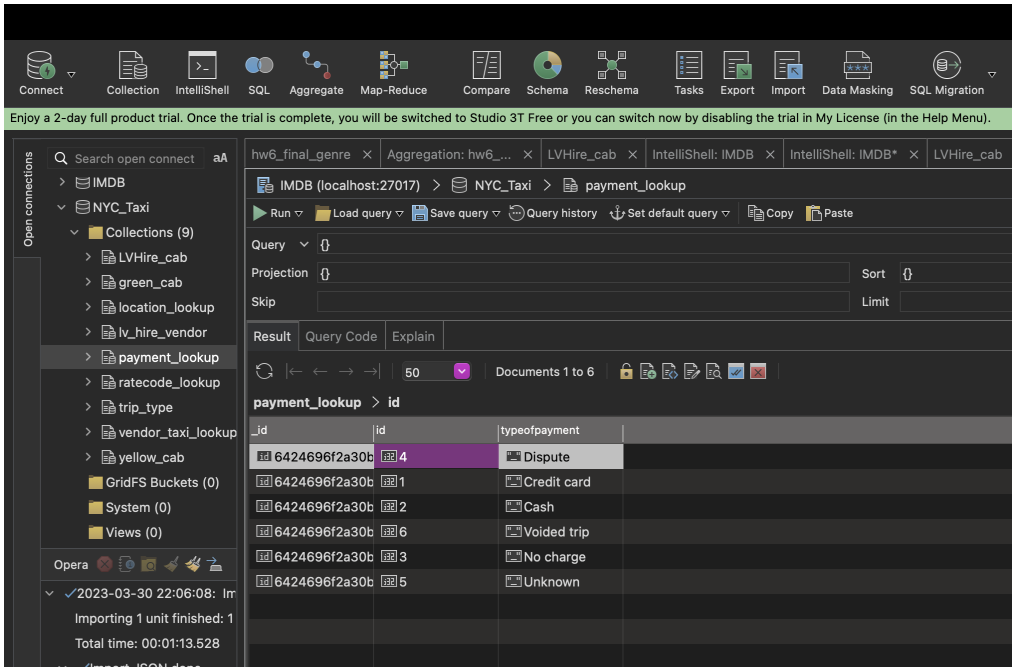
Yellow_cab


Connect
Collection
IntelliShell
SQL
Aggregate
Map-Reduce
Compare
Schema
Reschema
Tasks
Export
Import
Data Masking
SQL Migration
Users
Roles
Feedback
Go to Free
Enjoy a 2-day full product trial. Once the trial is complete, you will be switched to Studio 3T Free or you can switch now by disabling the trial in My License (in the Help Menu).
movies
Aggregation: movies
hw6_final_genre
Aggregation: hw6_...
IntelliShell: IMDB
IntelliShell: IMDB
location_lookup
lv_hire_vendor
IMDB (localhost:27017)
NYC_Taxi
yellow_cab
Run
Load query
Save query
Query history
Set default query
Copy
Paste
Visual Query Builder
Query {}
Projection {}
Sort {}
Skip
Limit
Result
Query Code
Explain
Documents 1 to 50
Table View
yellow_cab > VendorID
_id
VendorID
tpep_pickup_datetime
tpep_dropoff_datetime
passenger_count
trip_distance
RatecodeID
store_and_fwd_flag
PULocationID
DOLocationID
1 document selected
Count Documents
00:00:00.003

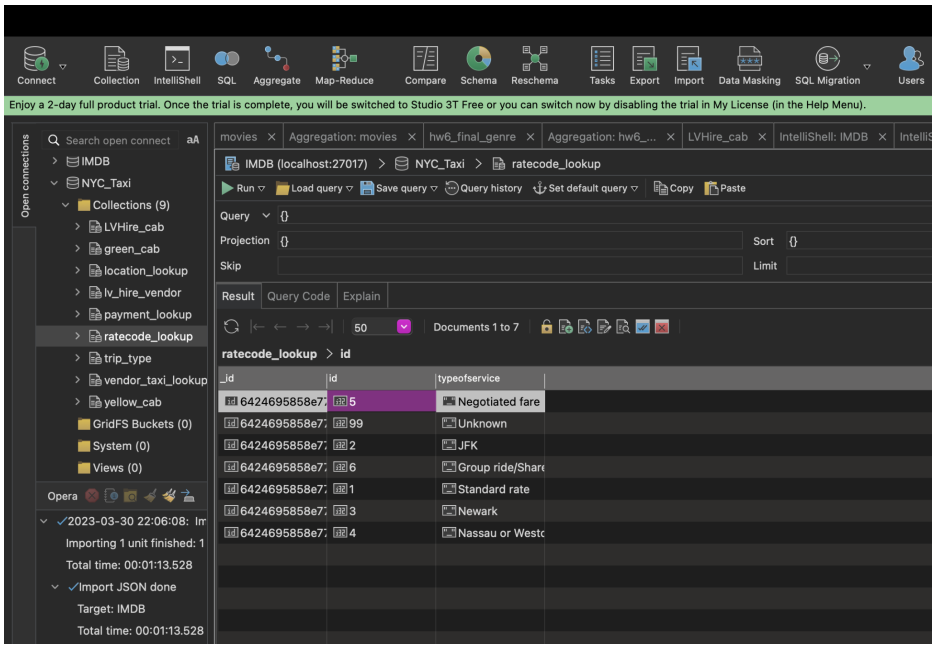
LvHire_cab


Connect
Collection
IntelliShell
SQL
Aggregate
Map-Reduce
Compare
Schema
Reschema
Tasks
Export
Import
Data Masking
SQL Migration
Users
Roles
Feedback
Go to Free
Enjoy a 2-day full product trial. Once the trial is complete, you will be switched to Studio 3T Free or you can switch now by disabling the trial in My License (in the Help Menu).
movies
Aggregation: movies
hw6_final_genre
Aggregation: hw6_...
IntelliShell: IMDB
IntelliShell: IMDB*
IntelliShell: IMDB
LVHire_cab
IMDB (localhost:27017)
NYC_Taxi
LVHire_cab
Run
Load query
Save query
Query history
Set default query
Copy
Paste
Visual Query Builder
Query {}
Projection {}
Sort {}
Skip
Limit
Result
Query Code
Explain
Documents 1 to 50
Table View
LVHire_cab > hvfhs_license_num
trip_miles
pickup_datetime1
trip_time
base_passenger_fare
tolls
bcf
sales_tax
congestion_surcharge
airport_fee
1 document selected
Count Documents
00:00:00.010

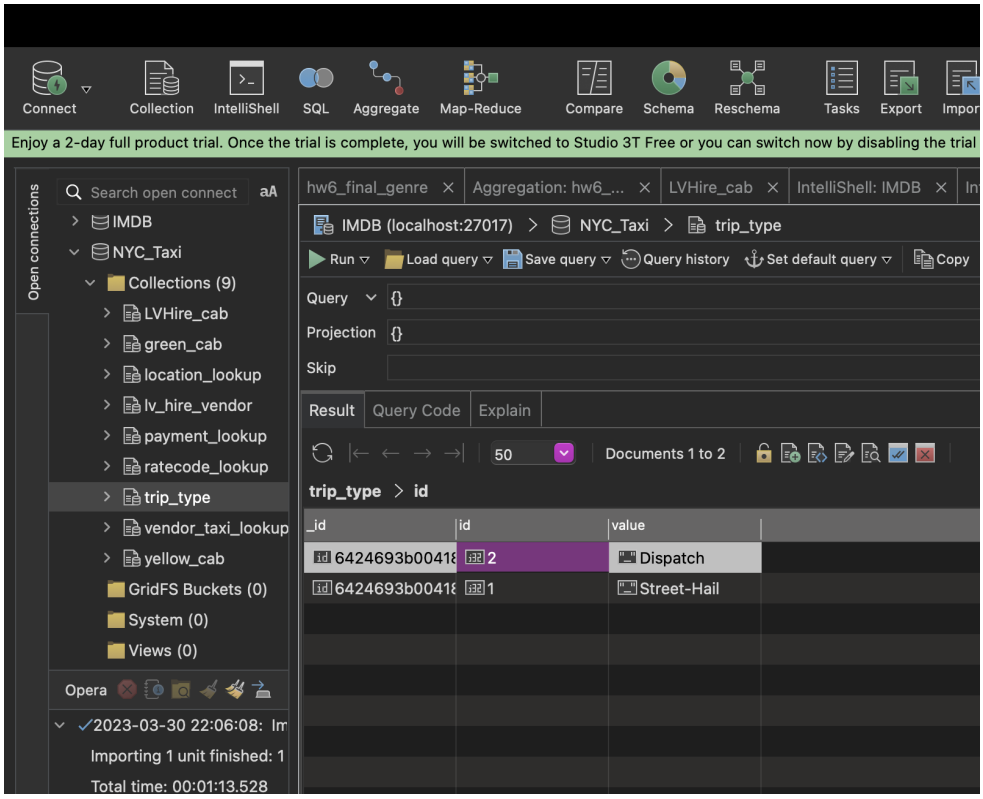
Payment_lookup



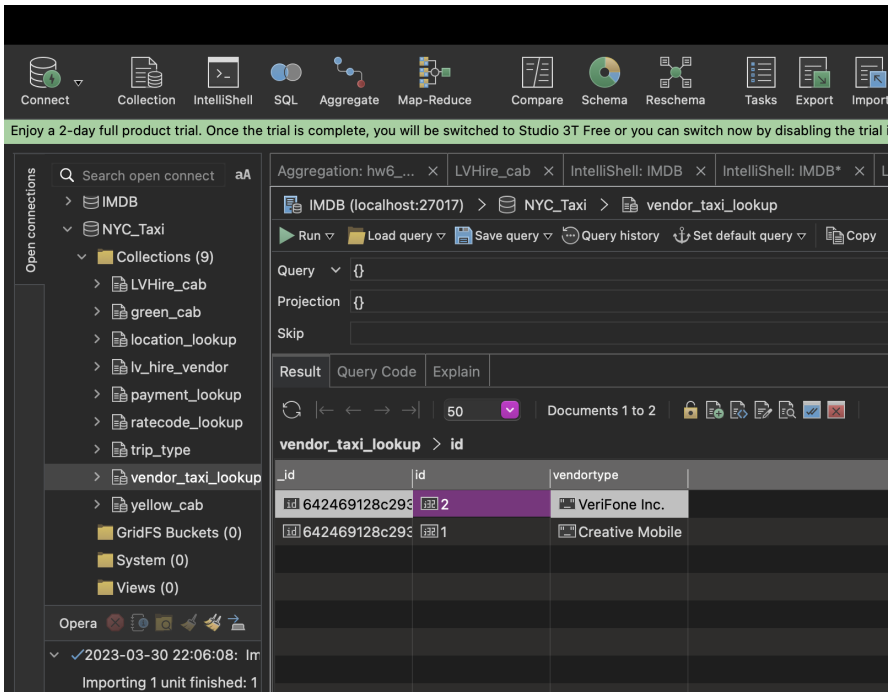
Ratecode_lookup



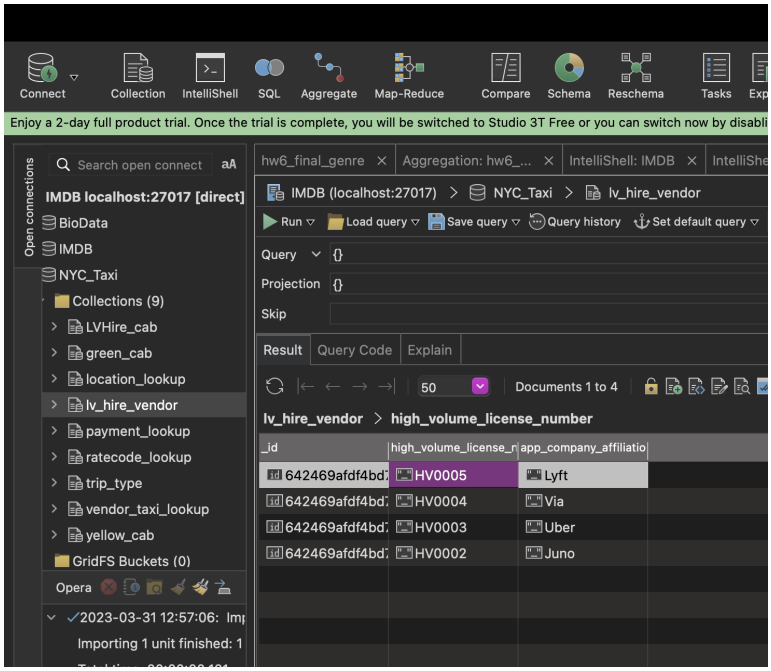
Trip_type



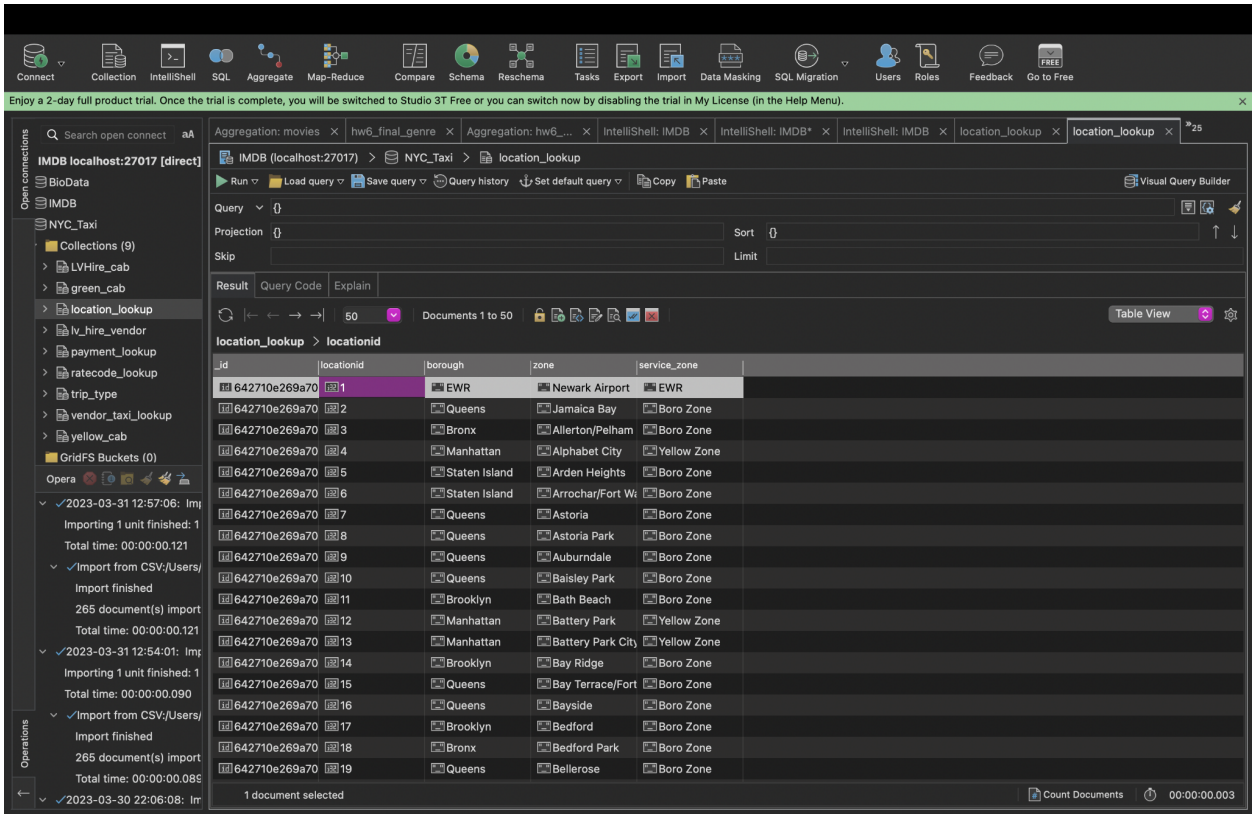
Vendor_taxi_lookup



Lv_hire_vendor



Location_lookup


Connect
Collection
IntelliShell
SQL
Aggregate
Map-Reduce
Compare
Schema
Reschema
Tasks
Export
Import
Data Masking
SQL Migration
Users
Roles
Feedback
Go to Free
Enjoy a 2-day full product trial. Once the trial is complete, you will be switched to Studio 3T Free or you can switch now by disabling the trial in My License (in the Help Menu).
Open connections
Search open connect
IMDB localhost:27017 [direct]
BioData
IMDB
NYC_Taxi
Collections (9)
LVHire_cab
green_cab
location_lookup
lv_hire_vendor
payment_lookup
ratecode_lookup
trip_type
vendor_taxi_lookup
yellow_cab
GridFS Buckets (0)
Operations
Aggregation: movies
hw6_final_genre
Aggregation: hw6_...
IntelliShell: IMDB
IntelliShell: IMDB*
IntelliShell: IMDB
location_lookup
location_lookup
IMDB (localhost:27017) > NYC_Taxi > location_lookup
Run
Load query
Save query
Query history
Set default query
Copy
Paste
Visual Query Builder
Query {}
Projection {}
Sort {}
Skip
Limit
Result
Query Code
Explain
50
Documents 1 to 50
Table View
location_lookup > locationid
_id
locationid
borough
zone
service_zone
642710e269a70 1 EWR Newark Airport EWR
642710e269a70 2 Queens Jamaica Bay Boro Zone
642710e269a70 3 Bronx Allerton/Pelham Boro Zone
642710e269a70 4 Manhattan Alphabet City Yellow Zone
642710e269a70 5 Staten Island Arden Heights Boro Zone
642710e269a70 6 Staten Island Arrochar/Fort W Boro Zone
642710e269a70 7 Queens Astoria Boro Zone
642710e269a70 8 Queens Astoria Park Boro Zone
642710e269a70 9 Queens Auburndale Boro Zone
642710e269a70 10 Queens Baisley Park Boro Zone
642710e269a70 11 Brooklyn Bath Beach Boro Zone
642710e269a70 12 Manhattan Battery Park Yellow Zone
642710e269a70 13 Manhattan Battery Park City Yellow Zone
642710e269a70 14 Brooklyn Bay Ridge Boro Zone
642710e269a70 15 Queens Bay Terrace/Fort Boro Zone
642710e269a70 16 Queens Bayside Boro Zone
642710e269a70 17 Brooklyn Bedford Boro Zone
642710e269a70 18 Bronx Bedford Park Boro Zone
642710e269a70 19 Queens Bellerose Boro Zone
1 document selected
Count Documents
00:00:00.003

2. SQL queries and optimization using Indices:

1) Payment Analysis:

We have analyzed the payment options, total payment and total number of payments for every cab.

```
select * from

(select count(1) as cnt,sum(y.total_amount+y.tip_amount) as amt, 'yellow' as cab ,'credit card' as typeofpayment
from yellowcab y join paymentlookup p on p.id=y.payment_type

and y.payment_type=1

union

select count(1) as cnt,sum(y.total_amount+y.tip_amount) as amt, 'yellow' as cab ,'cash' as typeofpayment from
yellowcab y join paymentlookup p on p.id=y.payment_type

and y.payment_type=2

union

select count(1) as cnt,sum(y.total_amount+y.tip_amount) as amt, 'yellow' as cab ,'No charge' as typeofpayment
from yellowcab y join paymentlookup p on p.id=y.payment_type

and y.payment_type=3

union

select count(1) as cnt,sum(y.total_amount+y.tip_amount) as amt, 'yellow' as cab ,'dispute' as typeofpayment from
yellowcab y join paymentlookup p on p.id=y.payment_type

and y.payment_type=4

union

select count(1) as cnt,sum(y.total_amount+y.tip_amount) as amt, 'yellow' as cab ,'unknown' as typeofpayment from
yellowcab y join paymentlookup p on p.id=y.payment_type

and y.payment_type=5

union

select count(1) as cnt,sum(y.total_amount+y.tip_amount) as amt, 'yellow' as cab ,'void' as typeofpayment from
yellowcab y join paymentlookup p on p.id=y.payment_type

and y.payment_type=6

union

select count(1) as cnt,sum(y.total_amount+y.tip_amount) as amt, 'green' as cab ,'credit card' as typeofpayment
from greencab y join paymentlookup p on p.id=y.payment_type
```

```
and y.payment_type=1

union

select count(1) as cnt,sum(y.total_amount+y.tip_amount) as amt, 'green' as cab ,'cash' as typeofpayment from
greencab y join paymentlookup p on p.id=y.payment_type

and y.payment_type=2

union

select count(1) as cnt,sum(y.total_amount+y.tip_amount) as amt, 'green' as cab ,'No charge' as typeofpayment
from greencab y join paymentlookup p on p.id=y.payment_type

and y.payment_type=3

union

select count(1) as cnt,sum(y.total_amount+y.tip_amount) as amt, 'green' as cab ,'dispute' as typeofpayment from
greencab y join paymentlookup p on p.id=y.payment_type

and y.payment_type=4

union

select count(1) as cnt,sum(y.total_amount+y.tip_amount) as amt, 'green' as cab ,'Unknown' as typeofpayment from
greencab y join paymentlookup p on p.id=y.payment_type

and y.payment_type=5

union

select count(1) as cnt,sum(y.total_amount+y.tip_amount) as amt, 'green' as cab ,'void' as typeofpayment from
greencab y join paymentlookup p on p.id=y.payment_type

and y.payment_type=6

union

select count(1) ,sum(base_passenger_fare+tolls+bcf+sales_tax+congestion_surcharge+tips+airport_fee)as
amt,'forhire' as cab ,'credit' as typeofpayment from lvhiretaxi

union

select count(1) ,sum(base_passenger_fare+tolls+bcf+sales_tax+congestion_surcharge+tips+airport_fee)as
amt,'forhire_uber' as cab ,'credit' as typeofpayment from lvhiretaxi

where hvfhs_license_num ='HV0003'

union

select count(1) ,sum(base_passenger_fare+tolls+bcf+sales_tax+congestion_surcharge+tips+airport_fee)as
amt,'forhire_lyft' as cab ,'credit' as typeofpayment from lvhiretaxi

where hvfhs_license_num ='HV0005'

)x order by cab,typeofpayment
```

Output:

cab	paymentoption	totalpayment	countofpayment
forhire	credit	306,801,508.27	12,637,977
forhire_lyft	credit	82,285,840.68	3,394,738
forhire_uber	credit	224,515,667.59	9,243,239
green	cash	311,101.87	21,638
green	credit card	700,667.61	34,108
green	dispute	733.94	134
green	No charge	-84.63	319
green	Unknown	27.85	1
green	void	[NULL]	0
yellow	cash	3,872,255.87	233,407
yellow	credit card	18,648,504.68	804,905
yellow	dispute	369,933.07	4,875
yellow	No charge	44,037.39	5,388

Indexing:

The query initially ran for **53 seconds**. After applying the below index the query ran for **31 seconds**.

```
create index yellow_payment on yellowcab
(
total_amount,
tip_amount
);
create index green_payment on greencab
(
total_amount,
tip_amount
);
```

```
create index forhire_payment on lvhiretaxi
(
base_passenger_fare,
tolls,
bcf,
sales_tax,
congestion_surcharge,
tips,
airport_fee
);
```

2) Average time delay analysis:

We have analyzed the average time difference between the requested time and the pickup time for LVhire cabs for different locations.

```
select
case when Cab='HV0003' then 'Uber'
else 'Lyft'
end,
Borough,
Average_time_diff_in_minutes_bet_Req_and_Pickup
from
(select l.hvfhs_license_num as Cab ,l2.borough as Borough,avg(extract(minute from pickup_datetime::time
-request_datetime::time)) as Average_time_diff_in_minutes_bet_Req_and_Pickup from lvhiretaxi l
join locationlookup l2
on
l.pulocationid =l2.locationid
group by l.hvfhs_license_num ,l2.borough
)x;
```

Output:

cab	borough	average_time_diff_in_minutes_bet_req_and_pickup
Uber	"Bronx"	3.465647741
Uber	"Brooklyn"	3.3228500899
Uber	"Manhattan"	2.8668117711
Uber	"Queens"	3.626435703
Uber	"Staten Island"	4.7685436594
Uber	"Unknown"	6.4224137931
Lyft	"Bronx"	4.6021455779
Lyft	"Brooklyn"	4.0903635342
Lyft	"Manhattan"	3.4964322801
Lyft	"Queens"	4.4315833317
Lyft	"Staten Island"	5.7869834828
Lyft	"Unknown"	5.456445993

Indexing:

The query initially ran for **65 seconds**. After applying the below index the query ran for **45 seconds**.

```
create index lvhire_lookup on lvhiretaxi(
hvfhs_license_num,
pickup_datetime,
request_datetime,
pulocationid
)
```

3) Trip Distance analysis:

We have analyzed the total miles covered by all cabs.

```
select y.cab as cab ,y.dst as distance_in_miles from (select sum(trip_distance) as dst , 'yellow' as cab from yellowcab y
union
select sum(trip_distance) as dst,'green' as cab from greencab g
```

```
union

select sum(trip_miles) as dst,'forhire' as cab from lvhiretaxi l

union

select sum(trip_miles) as dst,'forhire_uber' as cab from lvhiretaxi l where hvfhs_license_num ='HV0003'

union

select sum(trip_miles) as dst,'forhire_lyft' as cab from lvhiretaxi l where hvfhs_license_num ='HV0005'

) y

order by dst desc,cab desc;
```

Output:

cab	distance_in_miles
forhire	58,797,246.035
forhire_uber	42,491,370.94
forhire_lyft	16,305,875.095
green	4,859,525.73
yellow	3,419,294.71

Indexing:

The query initially ran for **20 seconds**. After applying the below indexes the query ran for **16 seconds**.

```
create index trip_dist on greencab(trip_distance)

create index trip_dist_yellow on yellowcab(trip_distance)

create index trip_miles_lvhire on lvhiretaxi(trip_miles)
```

4) Pickup Location analysis:

We have analyzed the top 10 pickup locations of ‘Uber’ and ‘Lyft’ cabs.

```
select * from (select * from (select 'Uber' as CabType,pulocationid ,l2.borough,l2."zone" ,l2.service_zone ,count(1)
as cnt from lvhiretaxi l join locationlookup l2 on l2.locationid =l.pulocationid

where hvfhs_license_num='HV0003'

group by 2,3,4,5 order by cnt desc limit 10) x

union
```

```
select * from (select 'Lyft' as CabType,pulocationid ,l2.borough,l2."zone" ,l2.service_zone ,count(1) as cnt from
lvhiretaxi l join locationlookup l2 on l2.locationid =l.pulocationid

where hvfhs_license_num='HV0005'

group by 2,3,4,5 order by cnt desc limit 10) x

union

select * from (

select 'Juno' as CabType,pulocationid ,l2.borough,l2."zone" ,l2.service_zone ,count(1) as cnt from lvhiretaxi l join
locationlookup l2 on l2.locationid =l.pulocationid

where hvfhs_license_num='HV0002'

group by 2,3,4,5) x

) y order by cnt desc,cabtype desc
```

Output:

	cabtype	pulocationid	borough	zone	service_zone	cnt
1	Uber	132	"Queens"	"JFK Airport"	"Airports"	139,968
2	Uber	79	"Manhattan"	"East Village"	"Yellow Zone"	137,397
3	Uber	61	"Brooklyn"	"Crown Heights North"	"Boro Zone"	127,363
4	Uber	138	"Queens"	"LaGuardia Airport"	"Airports"	115,764
5	Uber	231	"Manhattan"	"TriBeCa/Civic Center"	"Yellow Zone"	111,738
6	Uber	249	"Manhattan"	"West Village"	"Yellow Zone"	103,742
7	Uber	234	"Manhattan"	"Union Sq"	"Yellow Zone"	102,892
8	Uber	230	"Manhattan"	"Times Sq/Theatre District"	"Yellow Zone"	100,807
9	Uber	37	"Brooklyn"	"Bushwick South"	"Boro Zone"	98,707
10	Uber	68	"Manhattan"	"East Chelsea"	"Yellow Zone"	98,097
11	Lyft	61	"Brooklyn"	"Crown Heights North"	"Boro Zone"	59,312
12	Lyft	79	"Manhattan"	"East Village"	"Yellow Zone"	50,682
13	Lyft	132	"Queens"	"JFK Airport"	"Airports"	49,367
14	Lyft	37	"Brooklyn"	"Bushwick South"	"Boro Zone"	45,818
15	Lyft	138	"Queens"	"LaGuardia Airport"	"Airports"	43,905
16	Lyft	76	"Brooklyn"	"East New York"	"Boro Zone"	43,530
17	Lyft	7	"Queens"	"Astoria"	"Boro Zone"	40,375
18	Lyft	181	"Brooklyn"	"Park Slope"	"Boro Zone"	40,233
19	Lyft	42	"Manhattan"	"Central Harlem North"	"Boro Zone"	37,021

Indexing:

The query initially ran for **30 seconds**. After applying the below index the query ran for **11 seconds**.

```
create index location_index on locationlookup(locationid,borough);
```

5) Booking frequency analysis:

We have analyzed the numbers of cabs that are booked on weekdays compared to weekends.

```
select * from
(select 'Weekends_yellow Cab' cab, count(z.val_pickup::text ) from
(select to_Char(x.pickup,'Day')as val_pickup,to_char(x.dropoff,'Day') from
(select tpep_pickup_datetime::date as pickup,tpep_dropoff_datetime::date as dropoff from yellowcab y)x)z
where z.val_pickup like 'Sat%' or z.val_pickup ='Sun%'
union
select 'WeekDays_yellow Cab' cab,count(z.val_pickup::text )from
(select to_Char(x.pickup,'Day')as val_pickup,to_char(x.dropoff,'Day') from
(select tpep_pickup_datetime::date as pickup,tpep_dropoff_datetime::date as dropoff from yellowcab y)x)z
where z.val_pickup not like 'Sat%' and not z.val_pickup ='Sun%'
union
select 'Weekends_green Cab' cab, count(z.val_pickup::text ) from
(select to_Char(x.pickup,'Day')as val_pickup,to_char(x.dropoff,'Day') from
(select lpep_pickup_datetime::date as pickup,lpep_dropoff_datetime::date as dropoff from greencab y)x)z
where z.val_pickup like 'Sat%' or z.val_pickup ='Sun%'
union
select 'WeekDays_green_Cab' cab,count(z.val_pickup::text )from
(select to_Char(x.pickup,'Day')as val_pickup,to_char(x.dropoff,'Day') from
(select lpep_pickup_datetime::date as pickup,lpep_dropoff_datetime::date as dropoff from greencab y)x)z
where z.val_pickup not like 'Sat%' and not z.val_pickup ='Sun%'
union
select 'Weekends_for hire Cab' cab, count(z.val_pickup::text ) from
(select to_Char(x.pickup,'Day')as val_pickup,to_char(x.dropoff,'Day') from
(select pickup_datetime::date as pickup,dropoff_datetime::date as dropoff from lvhiretaxi y)x)z
where z.val_pickup like 'Sat%' or z.val_pickup ='Sun%'
union
select 'WeekDays_hire Cab' cab,count(z.val_pickup::text )from
```

```
(select to_Char(x.pickup,'Day')as val_pickup,to_char(x.dropoff,'Day') from
(select pickup_datetime::date as pickup,dropoff_datetime::date as dropoff from lvhiretaxi y)x)z
where z.val_pickup not like 'Sat%' and not z.val_pickup ='Sun%'
) outer_q
order by 1
```

Output:

cab	count
WeekDays_green_Cab	55,029
WeekDays_hire Cab	10,372,034
WeekDays_yellow Cab	885,385
Weekends_for hire Cab	2,265,943
Weekends_green Cab	7,466
Weekends_yellow Cab	163,190

Indexing:

The query initially ran for **170 seconds**. After applying the below indexes the query ran for **153 seconds**.

```
create index yellow_time on yellowcab(tpep_pickup_datetime,tpep_dropoff_datetime)
create index green_time on greencab(lpep_pickup_datetime,lpep_dropoff_datetime)
create index lvhire_time on lvhiretaxi(pickup_datetime,dropoff_datetime)
```

3. FDs and Normalization of Phase 1 data:

1. Payment Lookup

Id → typeofpayment
Typeofpayment → id

2. Vendortaxilookup

Id → vendortype

Vendortype → id

3. Ratecodelookup

Id → typeofservice
Typeofservice → id

4.locationlookup

Location, borough, zone → service_zone
Service_zone → Location, borough, zone

5. Lvhiretaxivendor

high_volume_license_number → app_company_affiliation
App_company_affiliation → high_volume_license_number

6. Triptype

Id → value
Value → id

7. Greencab

passenger_count, trip_distance, fare_amount, extra, mta_tax, tip_amount, tolls_amount, improvement_surcharge --> total_amount

Artificial_serial_key → rest of the table

8. Yellowcab

passenger_count, trip_distance, fare_amount, extra, mta_tax, tip_amount, tolls_amount, improvement_surcharge --> total_amount

Artificial_serial_key → rest of the table
9. LvHireTaxi

Artificial_serial_key → rest of the table

1NF -

The data provided in phase 1, clearly all the columns in all the tables had atomic values. Hence the whole data was already in 1NF.

2NF -

During phase 1, there were columns which were categorical in nature but had complete string values inside them. So we created a lookup table for those string values with ids being integers and then included those integers in the main tables. Hence, these lookup tables had the ids as primary keys and the string values were functionally dependent on the ids i.e. primary key. Hence, all these tables were in 2NF format.

3NF -

The tables had no transitivity present, all tables had all non-key columns dependent only on the primary key and since these tables were already in 1NF and 2NF, those tables were also in 3NF. Hence, the complete data is in normalized format.