MongoDB
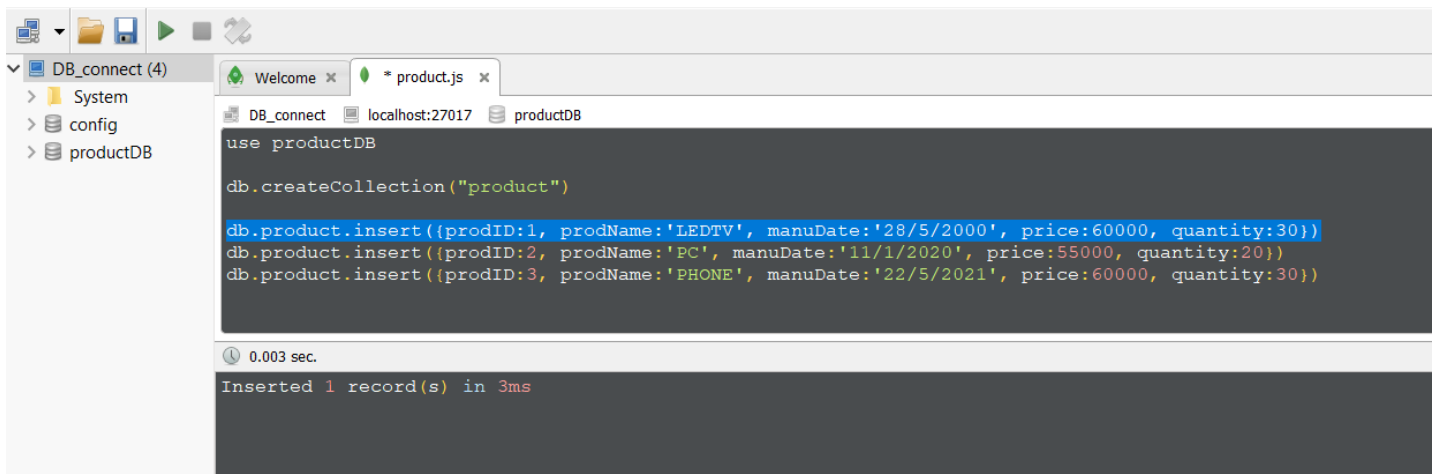
1

2

```
use productDB

db.createCollection("product")

db.product.insert({prodID:1, prodName:'LEDTV', manuDate:'28/5/2000', price:60000, quantity:30})
db.product.insert({prodID:2, prodName:'PC', manuDate:'11/1/2020', price:55000, quantity:20})
db.product.insert({prodID:3, prodName:'PHONE', manuDate:'22/5/2021', price:60000, quantity:30})
```

0.002 sec.

Inserted 1 record(s) in 2ms

0.001 sec.

Inserted 1 record(s) in 1ms

3


```
use productDB

db.createCollection("product")

//inserting
db.product.insert({prodID:1, prodName:'LEDTV', manuDate:'28/5/2000', price:60000, quantity:30})
db.product.insert({prodID:2, prodName:'PC', manuDate:'11/1/2020', price:55000, quantity:20})
db.product.insert({prodID:3, prodName:'PHONE', manuDate:'22/5/2021', price:60000, quantity:30})

//print
db.product.find().pretty()
```

product   0.044 sec.

Key	Value	Type
(1) ObjectId("60a8ab2d2281ea6af093e6ec")	{ 6 fields }	Object
_id	ObjectId("60a8ab2d2281ea6af093e6ec")	ObjectId
prodID	1.0	Double
prodName	LEDTV	String
manuDate	28/5/2000	String
price	60000.0	Double
quantity	30.0	Double
(2) ObjectId("60a8ab462281ea6af093e6ed")	{ 6 fields }	Object
_id	ObjectId("60a8ab462281ea6af093e6ed")	ObjectId
prodID	2.0	Double
prodName	PC	String
manuDate	11/1/2020	String
price	55000.0	Double
quantity	20.0	Double
(3) ObjectId("60a8ab462281ea6af093e6ee")	{ 6 fields }	Object
_id	ObjectId("60a8ab462281ea6af093e6ee")	ObjectId
prodID	3.0	Double
prodName	PHONE	String
manuDate	22/5/2021	String
price	60000.0	Double
quantity	30.0	Double
```

4

DB_connect (4)
  System
  config
  productDB

Welcome  |  * product.js

DB_connect  |  localhost:27017  |  productDB

```
use productDB
//create collection
//1
db.createCollection("product")

//inserting
//2
db.product.insert({prodID:1, prodName:'LEDTV', manuDate:'28/5/2000', price:60000, quantity:30})
db.product.insert({prodID:2, prodName:'PC', manuDate:'11/1/2020', price:55000, quantity:20})
db.product.insert({prodID:3, prodName:'PHONE', manuDate:'22/5/2021', price:60000, quantity:30})

//print
//3
db.product.find().pretty()

//find
//4
db.product.find({'prodName':'LEDTV', 'quantity':30});
```

product   0.008 sec.

Key	Value	Type
(1) ObjectId("60a8ab2d2281ea6af093e6ec")	{ 6 fields }	Object
_id	ObjectId("60a8ab2d2281ea6af093e6ec")	ObjectId
prodID	1.0	Double
prodName	LEDTV	String
manuDate	28/5/2000	String
price	60000.0	Double
quantity	30.0	Double

Cassandra Cql

1

```
cqlsh> create keyspace company with replication = {'class':'SimpleStrategy', 'replication_factor':1}
   ... ;
cqlsh> use company;
cqlsh:company> create table employee(_id int, name text, emp_no int, salary int, dept text, designation text, marital_status text, hobbies set<text>, primary key(_id, name)
);
Invalid syntax at line 1, char 23
  create table employee(_id int, name text, emp_no int, salary int, dept text, designation text, marital_status text, hobbies set<text>, primary key(_id, name));
                        ^
cqlsh:company> create table employee(_id int, name text, emp_no int, salary int, dept text, designation text, marital_status text, hobbies set<text>, primary key (_id, name
));
Invalid syntax at line 1, char 23
  create table employee(_id int, name text, emp_no int, salary int, dept text, designation text, marital_status text, hobbies set<text>, primary key (_id, name));
                        ^
cqlsh:company> create table employee(_id int, name text, emp_no int, salary int, dept text, designation text, marital_status text, hobbies set<text>, primarykey (_id, name)
);
Invalid syntax at line 1, char 23
  create table employee(_id int, name text, emp_no int, salary int, dept text, designation text, marital_status text, hobbies set<text>, primarykey (_id, name));
                        ^
cqlsh:company>  create table Projectdetails (projectid int, projectname text, studName text, rating double, duration int, primary key (projectid, projectname));
cqlsh:company> drop table Projectdetails
   ... ;
cqlsh:company>  create table Employee (id int, name text, emp_no int, salary int, dept text, designation text, marital_status text, hobbies set<text>, primary key (id, name
));
cqlsh:company> describe Employee

CREATE TABLE company.employee (
    id int,
    name text,
    dept text,
    designation text,
    emp_no int,
    hobbies set<text>,
    marital_status text,
    salary int,
    PRIMARY KEY (id, name)
) WITH CLUSTERING ORDER BY (name ASC)
    AND bloom_filter_fp_chance = 0.01
    AND caching = {'keys': 'ALL', 'rows_per_partition': 'NONE'}
    AND comment = ''
    AND compaction = {'class': 'org.apache.cassandra.db.compaction.SizeTieredCompactionStrategy', 'max_threshold': '32', 'min_threshold': '4'}
    AND compression = {'chunk_length_in_kb': '64', 'class': 'org.apache.cassandra.io.compress.LZ4Compressor'}
    AND crc_check_chance = 1.0
    AND dclocal_read_repair_chance = 0.1
    AND default_time_to_live = 0
    AND gc_grace_seconds = 864000
    AND max_index_interval = 2048
    AND memtable_flush_period_in_ms = 0
    AND min_index_interval = 128
    AND read_repair_chance = 0.0
    AND speculative_retry = '99PERCENTILE';
```

2

```
cqlsh:company>  insert into Employee(id, name, dept, designation, emp_no, hobbies, marital_status, salary) values (1, 'Dhanraj','Software', 'junior dev', 101, {'football','
bloging','running'}, 'Not married', 90000);
cqlsh:company> select * form Employee;
SyntaxException: line 1:9 mismatched input 'form' expecting K_FROM (select * [form]...)
cqlsh:company> select * from Employee;

 id | name    | dept     | designation | emp_no | hobbies                         | marital_status | salary
----+---------+----------+-------------+--------+---------------------------------+----------------+--------
  1 | Dhanraj | Software | junior dev  |    101 | {'bloging', 'football', 'running'} |    Not married |  90000

(1 rows)
cqlsh:company>  insert into Employee(id, name, dept, designation, emp_no, hobbies, marital_status, salary) values (2, 'Raju','Ceo', 'Manager', 202, {'cooking', 'reading'},
'married', 88000);
cqlsh:company>  insert into Employee(id, name, dept, designation, emp_no, hobbies, marital_status, salary) values (4, 'Alex','Sales', 'Manager', 202, {'cooking', 'reading'}
, 'married', 111000);
cqlsh:company> select * from Employee;

 id | name    | dept     | designation | emp_no | hobbies                         | marital_status | salary
----+---------+----------+-------------+--------+---------------------------------+----------------+--------
  1 | Dhanraj | Software | junior dev  |    101 | {'bloging', 'football', 'running'} |    Not married |  90000
  2 |    Raju |      Ceo |     Manager |    202 |            {'cooking', 'reading'} |        married |  88000
  4 |    Alex |    Sales |     Manager |    202 |            {'cooking', 'reading'} |        married | 111000

(3 rows)
```

3

```
[...i you want to execute this query despite the performance unpredictability, use ALLOW FILTERING.]
cqlsh:company> select * from Employee where designation='Manager' ALLOW FILTERING;

 id | name | dept  | designation | emp_no | hobbies                | marital_status | previous_expirence | salary
----+------+-------+-------------+--------+------------------------+----------------+--------------------+--------
  2 | Raju |  Ceo  |     Manager |    202 | {'cooking', 'reading'} |        married  |               null |  88000
  4 | Alex | Sales |     Manager |    202 | {'cooking', 'reading'} |        married  |                 15 | 111000

(2 rows)
cqlsh:company> select * from Employee where designation='Manager'and salary>85000 ALLOW FILTERING;

 id | name | dept  | designation | emp_no | hobbies                | marital_status | previous_expirence | salary
----+------+-------+-------------+--------+------------------------+----------------+--------------------+--------
  2 | Raju |  Ceo  |     Manager |    202 | {'cooking', 'reading'} |        married  |               null |  88000
  4 | Alex | Sales |     Manager |    202 | {'cooking', 'reading'} |        married  |                 15 | 111000

(2 rows)
cqlsh:company> select * from Employee where designation='Manager'and salary>85000 Order by name desc ALLOW FILTERING;
InvalidRequest: Error from server: code=2200 [Invalid query] message="ORDER BY is only supported when the partition key is restricted by an EQ or an IN."
cqlsh:company> describe Employee
```

4

```
cqlsh:company> alter table Employee add previous_expirence int;
cqlsh:company>cqlsh:company> select * from Employee;

 id | name    id | name    | dept     | designation | emp_no | hobbies                         | marital_status | previous_expirence | salary
----+-------- ---+---------+----------+-------------+--------+---------------------------------+----------------+--------------------+--------
  1 | Dhanraj  1 | Dhanraj | Software | junior dev  |    101 | {'bloging', 'football', 'running'} |    Not married |               null |  90000
  2 |    Raju  2 |    Raju |      Ceo |     Manager |    202 |            {'cooking', 'reading'} |        married |               null |  88000
  4 |    Alex  4 |    Alex |    Sales |     Manager |    202 |            {'cooking', 'reading'} |        married |                 15 | 111000

(3 rows)   (3 rows)
cqlsh:company> update Employee set previous_expirence=15 where id=4
         ... ;
InvalidRequest: Error from server: code=2200 [Invalid query] message="Some clustering keys are missing: name"
cqlsh:company> update Employee set previous_expirence=15 where id=4 and name='Alex';
cqlsh:company> select * from Employee;

 id | name    | dept     | designation | emp_no | hobbies                         | marital_status | previous_expirence | salary
----+---------+----------+-------------+--------+---------------------------------+----------------+--------------------+--------
  1 | Dhanraj | Software | junior dev  |    101 | {'bloging', 'football', 'running'} |    Not married |               null |  90000
  2 |    Raju |      Ceo |     Manager |    202 |            {'cooking', 'reading'} |        married |               null |  88000
  4 |    Alex |    Sales |     Manager |    202 |            {'cooking', 'reading'} |        married |                 15 | 111000

(3 rows)
cqlsh:company> copy Employee(id, name, dept, designation,emp_no,hobbies,marital_status,previous_expirence,salary) to 'C:\Users\dhanr\Desktop\LABTEST1\write.csv'
```

5

```
cqlsh:company> copy Employee(id, name, dept, designation,emp_no,hobbies,marital_status,previous_expirence,salary) to 'C:\Users\dhanr\Desktop\LABTEST1\write.csv'
         ... ;
Using 11 child processes

Starting copy of company.employee with columns [id, name, dept, designation, emp_no, hobbies, marital_status, previous_expirence, salary].
Processed: 3 rows; Rate:       1 rows/s; Avg. rate:       1 rows/s
3 rows exported to 1 files in 3.079 seconds.
cqlsh:company> truncate Employee;
cqlsh:company> select * from Employee;

 id | name | dept | designation | emp_no | hobbies | marital_status | previous_expirence | salary
----+------+------+-------------+--------+---------+----------------+--------------------+--------

(0 rows)
cqlsh:company>  copy Employee(id, name, dept, designation,emp_no,hobbies,marital_status,previous_expirence,salary) from 'C:\Users\dhanr\Desktop\LABTEST1\read.csv'
         ... ;
Using 11 child processes
```

export->truncate->import

```
cqlsh:company> select * from Employee;

 id | name    | dept     | designation | emp_no | hobbies                         | marital_status | previous_expirence | salary
----+---------+----------+-------------+--------+---------------------------------+----------------+--------------------+--------
  1 | Dhanraj | Software | junior dev  |    101 | {'bloging', 'football', 'running'} |    Not married |               null |  90000
  2 |    Raju |      Ceo |     Manager |    202 |            {'cooking', 'reading'} |        married |               null |  88000
  4 |    Alex |    Sales |     Manager |    202 |            {'cooking', 'reading'} |        married |                 15 | 111000

(3 rows)
```

csv file

	A	B	C	D	E	F	G	H	I	J	K
1	1	Dhanraj	Software	junior dev	101	{'bloging', 'f|	Not married		90000		
2	4	Alex	Sales	Manager	202	{'cooking', '|	married	15	111000		
3	2	Raju	Ceo	Manager	202	{'cooking', '|	married		88000		
4											
5											
6											
7											
8											
9											
10											

	A	B	C	D	E	F	G	H	I	J	K
1	1	Dhanraj	Software	junior dev	101	{'bloging', 'f|	Not married		90000		
2	4	Alex	Sales	Manager	202	{'cooking', '|	married	15	111000		
3	2	Raju	Ceo	Manager	202	{'cooking', '|	married		88000		