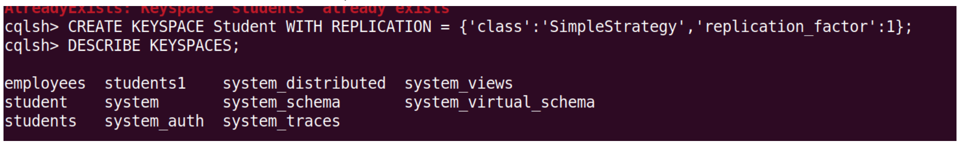
Working with Cassandra

Create KeySpace :

CREATE KEYSPACE Student WITH REPLICATION =
{'class':'SimpleStrategy','replication_factor':1};

Describe the existing Keyspaces:

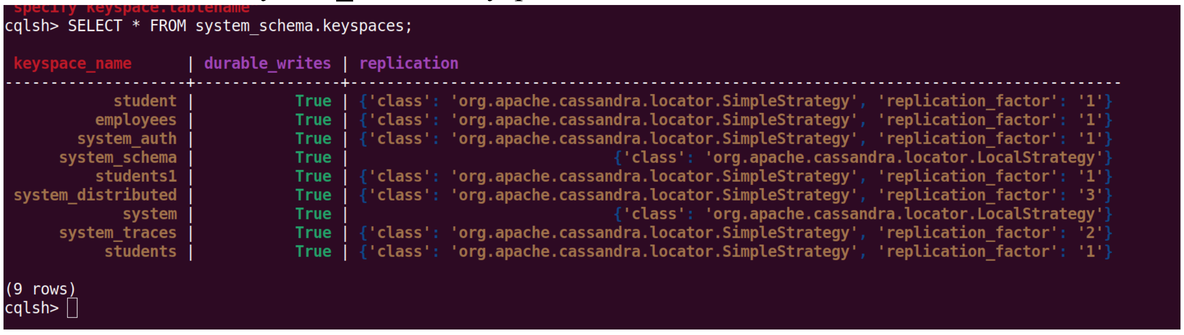
DESCRIBE KEYSPACES;



```
AlreadyExists: Keyspace 'students' already exists
cqlsh> CREATE KEYSPACE Student WITH REPLICATION = {'class':'SimpleStrategy','replication_factor':1};
cqlsh> DESCRIBE KEYSPACES;

employees  students1    system_distributed  system_views
student    system       system_schema       system_virtual_schema
students   system_auth  system_traces
```

For More details on existing keyspaces:

SELECT * FROM system_schema.keyspaces;



```
 specify keyspace.tablename
cqlsh> SELECT * FROM system_schema.keyspaces;

 keyspace_name      | durable_writes | replication
--------------------+----------------+-------------------------------------------------------------------------------------
            student |           True | {'class': 'org.apache.cassandra.locator.SimpleStrategy', 'replication_factor': '1'}
          employees |           True | {'class': 'org.apache.cassandra.locator.SimpleStrategy', 'replication_factor': '1'}
        system_auth |           True | {'class': 'org.apache.cassandra.locator.SimpleStrategy', 'replication_factor': '1'}
      system_schema |           True |                              {'class': 'org.apache.cassandra.locator.LocalStrategy'}
          students1 |           True | {'class': 'org.apache.cassandra.locator.SimpleStrategy', 'replication_factor': '1'}
 system_distributed |           True | {'class': 'org.apache.cassandra.locator.SimpleStrategy', 'replication_factor': '3'}
             system |           True |                              {'class': 'org.apache.cassandra.locator.LocalStrategy'}
      system_traces |           True | {'class': 'org.apache.cassandra.locator.SimpleStrategy', 'replication_factor': '2'}
           students |           True | {'class': 'org.apache.cassandra.locator.SimpleStrategy', 'replication_factor': '1'}

(9 rows)
cqlsh>
```

use the keyspace "Student":

USE Student;

To create table (column family) by name Student_Info:

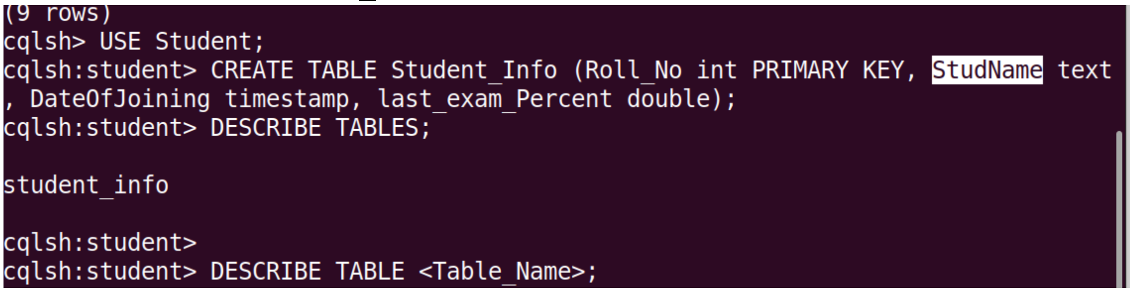
CREATE TABLE Student_Info (Roll_No int PRIMARY KEY, StudName text,
DateOfJoining timestamp, last_exam_Percent double);

Lookup the names of all tables in the current keyspaces

DESCRIBE TABLES;

Describe the table information

DESCRIBE TABLE <Table_Name>;



```
(9 rows)
cqlsh> USE Student;
cqlsh:student> CREATE TABLE Student_Info (Roll_No int PRIMARY KEY, StudName text
, DateOfJoining timestamp, last_exam_Percent double);
cqlsh:student> DESCRIBE TABLES;

student_info

cqlsh:student>
cqlsh:student> DESCRIBE TABLE <Table_Name>;
```

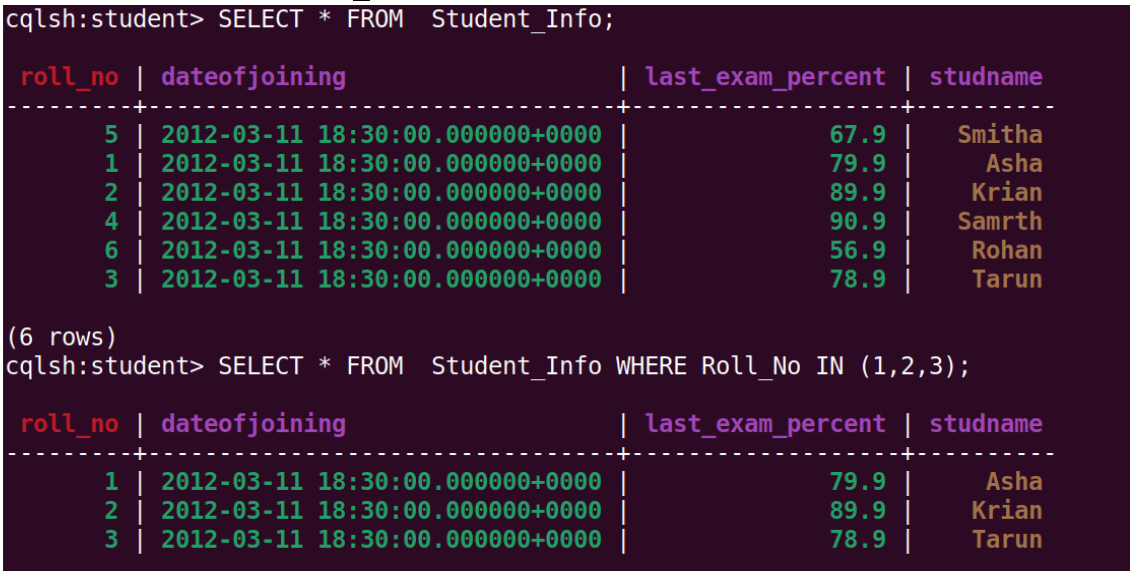
CRUD

Insert :

BEGIN BATCH
INSERT INTO Student_Info(Roll_No, StudName, DateOfJoining, last_exam_Percent)
VALUES (1,'Asha','2012-03-12',79.9)
INSERT INTO Student_Info(Roll_No, StudName, DateOfJoining, last_exam_Percent)
VALUES (2,'Krian','2012-03-12',89.9)
INSERT INTO Student_Info(Roll_No, StudName, DateOfJoining, last_exam_Percent)
VALUES (3,'Tarun','2012-03-12',78.9)
INSERT INTO Student_Info(Roll_No, StudName, DateOfJoining, last_exam_Percent)
VALUES (4,'Samrth','2012-03-12',90.9)
INSERT INTO Student_Info(Roll_No, StudName, DateOfJoining, last_exam_Percent)
VALUES (5,'Smitha','2012-03-12',67.9)
INSERT INTO Student_Info(Roll_No, StudName, DateOfJoining, last_exam_Percent)
VALUES (6,'Rohan','2012-03-12',56.9)
APPLY BATCH;

View data from the table "Student_Info"

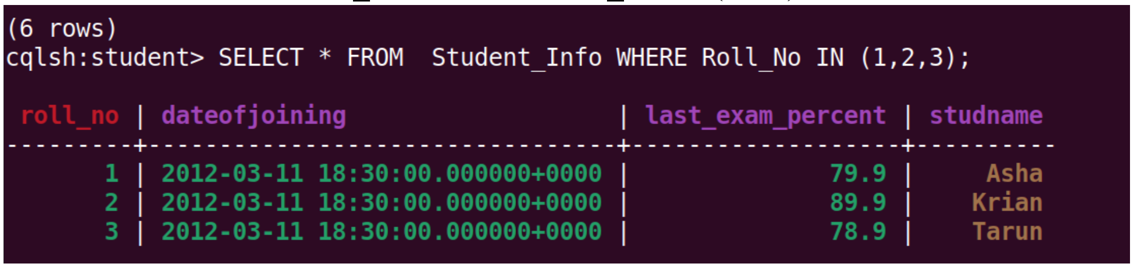
SELECT * FROM Student_Info;



```
cqlsh:student> SELECT * FROM  Student_Info;

 roll_no | dateofjoining                   | last_exam_percent | studname
---------+---------------------------------+-------------------+----------
       5 | 2012-03-11 18:30:00.000000+0000 |              67.9 |   Smitha
       1 | 2012-03-11 18:30:00.000000+0000 |              79.9 |     Asha
       2 | 2012-03-11 18:30:00.000000+0000 |              89.9 |    Krian
       4 | 2012-03-11 18:30:00.000000+0000 |              90.9 |   Samrth
       6 | 2012-03-11 18:30:00.000000+0000 |              56.9 |    Rohan
       3 | 2012-03-11 18:30:00.000000+0000 |              78.9 |    Tarun

(6 rows)
cqlsh:student> SELECT * FROM  Student_Info WHERE Roll_No IN (1,2,3);

 roll_no | dateofjoining                   | last_exam_percent | studname
---------+---------------------------------+-------------------+----------
       1 | 2012-03-11 18:30:00.000000+0000 |              79.9 |     Asha
       2 | 2012-03-11 18:30:00.000000+0000 |              89.9 |    Krian
       3 | 2012-03-11 18:30:00.000000+0000 |              78.9 |    Tarun
```

View data from the table "Student_Info" where Rollo column either has a value 1 or 2 or 3

SELECT * FROM Student_Info WHERE Roll_No IN (1,2,3);



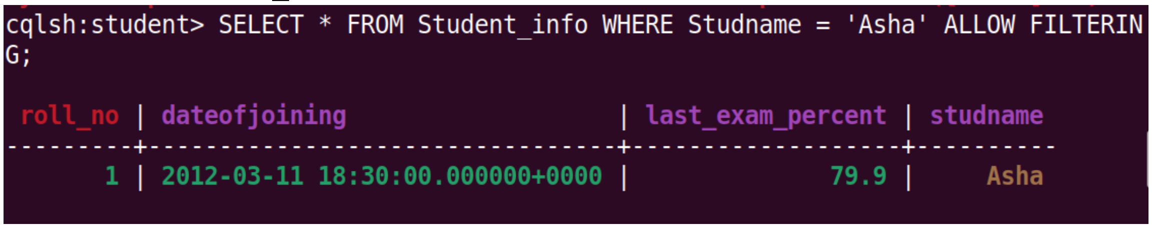
```
(6 rows)
cqlsh:student> SELECT * FROM  Student_Info WHERE Roll_No IN (1,2,3);

 roll_no | dateofjoining                   | last_exam_percent | studname
---------+---------------------------------+-------------------+----------
       1 | 2012-03-11 18:30:00.000000+0000 |              79.9 |     Asha
       2 | 2012-03-11 18:30:00.000000+0000 |              89.9 |    Krian
       3 | 2012-03-11 18:30:00.000000+0000 |              78.9 |    Tarun
```

To execute a non primary key - will throw an error
select * from Student_info where Studname= 'Asha';

So create an INDEX on the Column as below:
To create an INDEX on StudName Column of the Student_Info column family

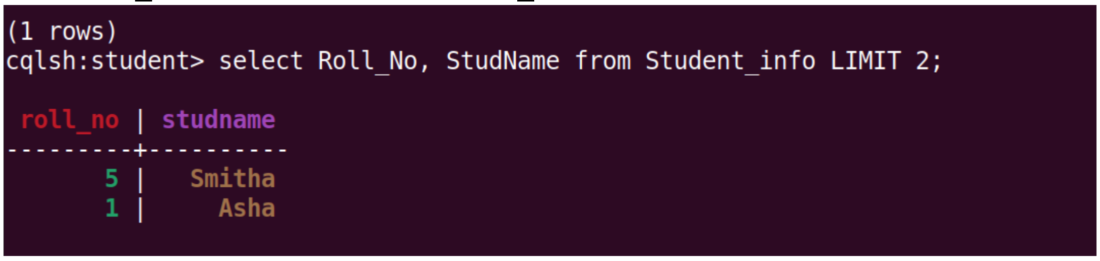
CREATE INDEX ON Student_Info (StudName);

Now execute the query based on the INDEXED Column:
select * from Student_info where Studname= 'Asha';



```
cqlsh:student> SELECT * FROM Student_info WHERE Studname = 'Asha' ALLOW FILTERIN
G;

 roll_no | dateofjoining                   | last_exam_percent | studname
---------+---------------------------------+-------------------+----------
       1 | 2012-03-11 18:30:00.000000+0000 |              79.9 |     Asha
```

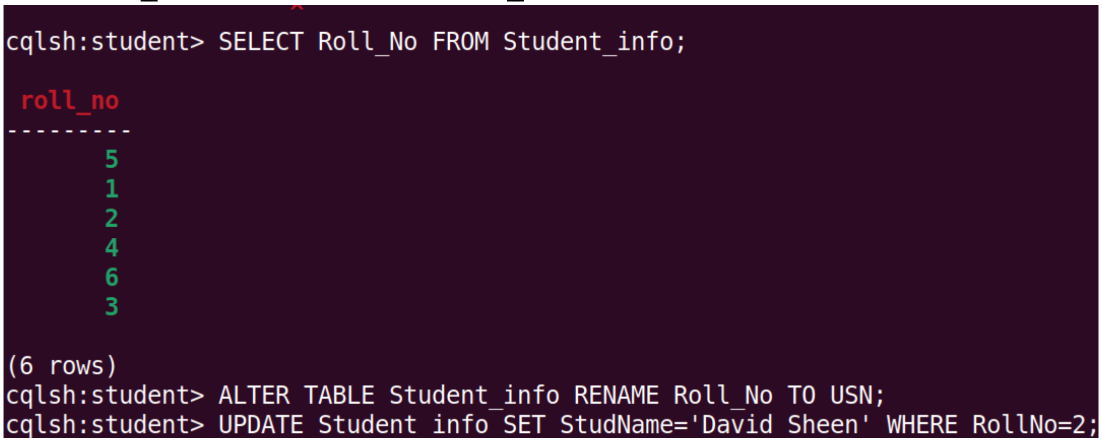
To specify the number of rows retured in the output
select Roll_No, StudName from Student_info LIMIT 2;



```
(1 rows)
cqlsh:student> select Roll_No, StudName from Student_info LIMIT 2;

 roll_no | studname
---------+----------
       5 |   Smitha
       1 |     Asha
```

Alias for Column:

Select Roll_No as “USN” from Student_info;



```
cqlsh:student> SELECT Roll_No FROM Student_info;

 roll_no
---------
       5
       1
       2
       4
       6
       3

(6 rows)
cqlsh:student> ALTER TABLE Student_info RENAME Roll_No TO USN;
cqlsh:student> UPDATE Student_info SET StudName='David Sheen' WHERE RollNo=2;
```

UPDATE

UPDATE Student_info SET StudName='David Sheen' WHERE RollNo=2;

Lets try to update the primary key

UPDATE Student_info SET Roll_No=6 WHERE Roll_No=3;

DELETE
DELETE LastExamPercent FROM Student_info WHERE USN=2;

Delete a Row
DELETE FROM student_info WHERE USN=2;

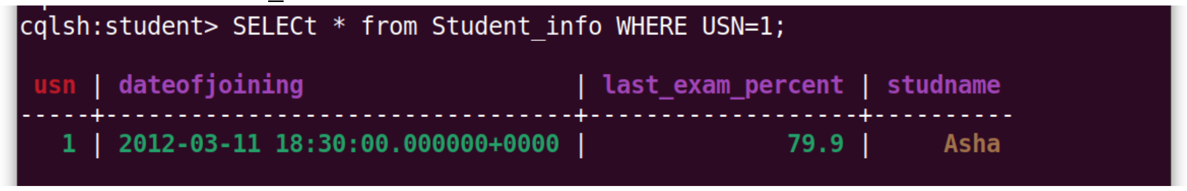
Set Collection
A column of type set consists of unordered unique values. However, when the column is queried, it returns, it returns the values in sorted order. For example, for text values, it sorts in alphabetical order.

ALTER TABLE Student_info ADD hobbies set<text>

List Collection
When the order of elements matter, one should go for a list collection.
ALTER TABLE Student_info ADD language list<text>;

```
UPDATE Student_info
        SET hobbies=hobbies+{'Chess,Table Tennis'}
              WHERE USN=1;
```

SELECt * from Student_info WHERE USN=1;



```
cqlsh:student> SELECt * from Student_info WHERE USN=1;

 usn | dateofjoining                    | last_exam_percent | studname
-----+----------------------------------+-------------------+----------
   1 | 2012-03-11 18:30:00.000000+0000 |              79.9 |     Asha
```

```
UPDATE Student_info
         SET langusge=language+['Hindi,English']
              WHERE USN=1;
```

Note: You can remove an element from a set using the subtraction(-) operator.

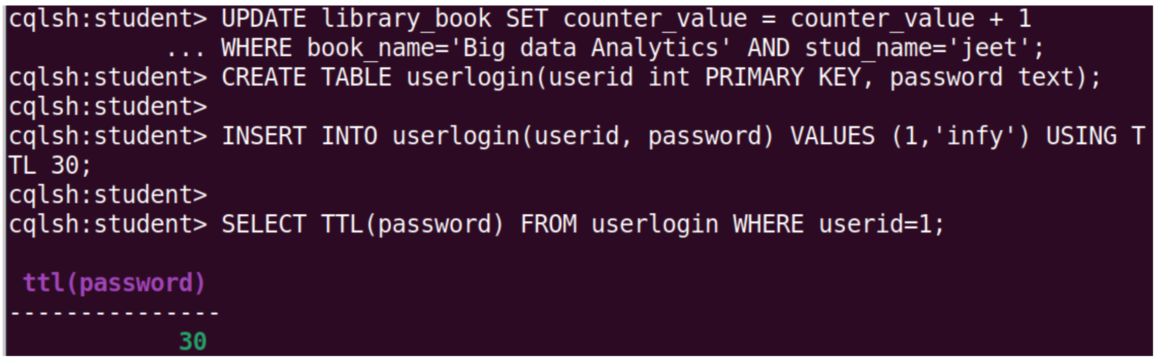
USING A COUNTER

A counter is a special column that is changed in increments. For example, we may need a counter column to count the number of times a particular book is issued from the library bythe student.

CREATE TABLE library_book(counter_value counter, book_name varchar, stud_name varchar, PRIMARY KEY(book_name,stud_name));

Load data into the counter column

UPDATE library_book SET counetr value=couner_vale+1 WHERE book_name='Big data Analytics' AND stud_name='jeet';



```
cqlsh:student> UPDATE library_book SET counter_value = counter_value + 1
           ... WHERE book_name='Big data Analytics' AND stud_name='jeet';
cqlsh:student> CREATE TABLE userlogin(userid int PRIMARY KEY, password text);
cqlsh:student>
cqlsh:student> INSERT INTO userlogin(userid, password) VALUES (1,'infy') USING T
TL 30;
cqlsh:student>
cqlsh:student> SELECT TTL(password) FROM userlogin WHERE userid=1;

 ttl(password)
---------------
            30
```

TIME TO LIVE

CREATE TABLE userlogin(userid int PRIMARY KEY, password text);

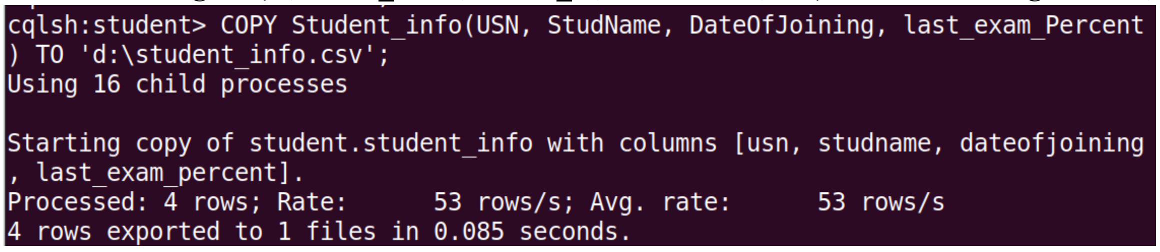
INSERT INTO userlogin(userid, password) VALUES (1,'infy') USING TTL 30;

SELECT TTL(password) FROM userlogin WHERE userid=1;

IMPORT and EXPORT

Export to CSV

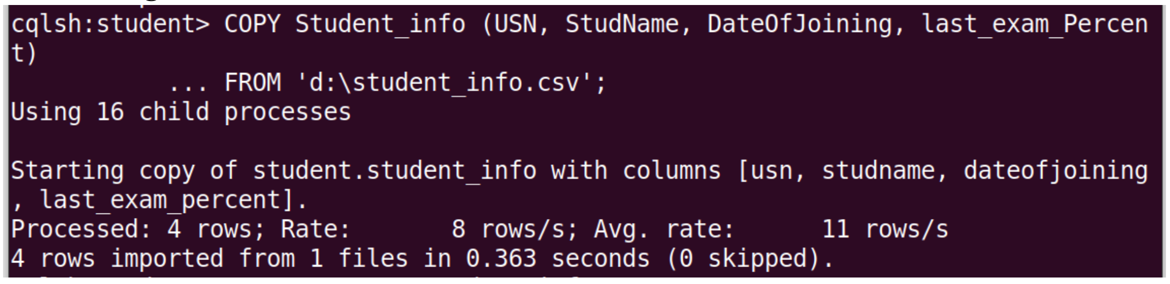
COPY elearninglists(id,course_order, course_id,courseowner,title) TO 'd:\elearninglists.csv';



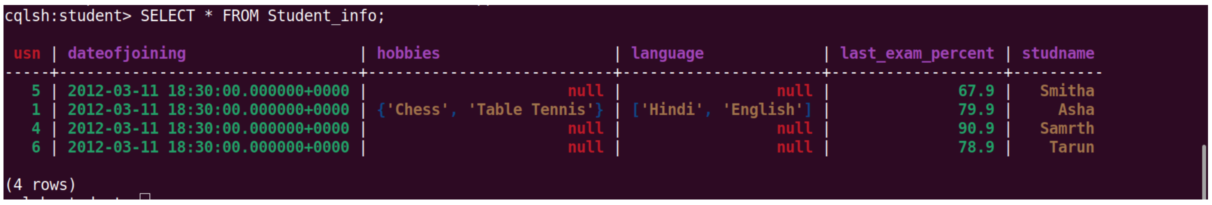
```
cqlsh:student> COPY Student_info(USN, StudName, DateOfJoining, last_exam_Percent
) TO 'd:\student_info.csv';
Using 16 child processes

Starting copy of student.student_info with columns [usn, studname, dateofjoining
, last_exam_percent].
Processed: 4 rows; Rate:      53 rows/s; Avg. rate:      53 rows/s
4 rows exported to 1 files in 0.085 seconds.
```

Import from CSV

COPY elearninglists(id,course_order, course_id,courseowner,title) FROM 'd:\elearninglists.csv';



```
cqlsh:student> COPY Student_info (USN, StudName, DateOfJoining, last_exam_Percen
t)
           ... FROM 'd:\student_info.csv';
Using 16 child processes

Starting copy of student.student_info with columns [usn, studname, dateofjoining
, last_exam_percent].
Processed: 4 rows; Rate:       8 rows/s; Avg. rate:      11 rows/s
4 rows imported from 1 files in 0.363 seconds (0 skipped).
```



```
cqlsh:student> SELECT * FROM Student_info;

 usn | dateofjoining                   | hobbies                   | language             | last_exam_percent | studname
-----+---------------------------------+---------------------------+----------------------+-------------------+----------
   5 | 2012-03-11 18:30:00.000000+0000 |                      null |                 null |              67.9 |   Smitha
   1 | 2012-03-11 18:30:00.000000+0000 | {'Chess', 'Table Tennis'} | ['Hindi', 'English'] |              79.9 |     Asha
   4 | 2012-03-11 18:30:00.000000+0000 |                      null |                 null |              90.9 |   Samrth
   6 | 2012-03-11 18:30:00.000000+0000 |                      null |                 null |              78.9 |    Tarun

(4 rows)
```

Import FROM STDIN

```
COPY persons(id,fname,lnmae)FROM STDIN;
```

Export to STDOUT

```
COPY elearninglists(id,course_order, course_id,courseowner,title) TO STDOUT;
```