CREATE CONSTRAINT on (S:Student) ASSERT S.Score IS UNIQUE;

CREATE INDEX ON :Test(Score);

Q1: Read in each record in this file as a node without a label in Neo4j with two properties called ‘stu’ and ‘score’?

lOAD CSV WITH HEADERS FROM 'file:/c:/skr/students18.csv' AS line CREATE(students{stu:line.Name, score:toInt(line.Score)})

ROWS
TEXT
CODE

```
╒════════════════════════╤═════════╕
│n                       │labels(n)│
╞════════════════════════╪═════════╡
│{score: 59, stu: Saeed} │[]       │
├────────────────────────┼─────────┤
│{score: 81, stu: Sam}   │[]       │
├────────────────────────┼─────────┤
│{score: 77, stu: Sammy} │[]       │
├────────────────────────┼─────────┤
│{score: 77, stu: Sammy} │[]       │
├────────────────────────┼─────────┤
│{score: 76, stu: Saman} │[]       │
├────────────────────────┼─────────┤
│{score: 34, stu: Saman} │[]       │
├────────────────────────┼─────────┤
│{score: 19, stu: Shah}  │[]       │
├────────────────────────┼─────────┤
│{score: 94, stu: Saeed} │[]       │
├────────────────────────┼─────────┤
│{score: 81, stu: Saeed} │[]       │
├────────────────────────┼─────────┤
│{score: 60, stu: Sam}   │[]       │
├────────────────────────┼─────────┤
│{score: 77, stu: Sam}   │[]       │
├────────────────────────┼─────────┤
│{score: 61, stu: Shah}  │[]       │
├────────────────────────┼─────────┤
│{score: 72, stu: Shah}  │[]       │
├────────────────────────┼─────────┤
│{score: 34, stu: Joe}   │[]       │
├────────────────────────┼─────────┤
│{score: 39, stu: Joseph}│[]       │
├────────────────────────┼─────────┤
│{score: 60, stu: Joseph}│[]       │
```

```
├────────────────────────┼──────────┤
│{score: 56, stu: Jordan}│[]        │
├────────────────────────┼──────────┤
│{score: 66, stu: Jordan}│[]        │
├────────────────────────┼──────────┤
│{score: 77, stu: Sammy} │[]        │
├────────────────────────┼──────────┤
│{score: 76, stu: Saman} │[]        │
├────────────────────────┼──────────┤
│{score: 25, stu: Jordan}│[]        │
├────────────────────────┼──────────┤
│{score: 94, stu: Josh}  │[]        │
├────────────────────────┼──────────┤
│{score: 21, stu: Josh}  │[]        │
├────────────────────────┼──────────┤
│{score: 35, stu: Joe}   │[]        │
├────────────────────────┼──────────┤
│{score: 90, stu: Joe}   │[]        │
├────────────────────────┼──────────┤
│{score: 52, stu: Joseph}│[]        │
├────────────────────────┼──────────┤
│{score: 67, stu: Josh}  │[]        │
├────────────────────────┼──────────┤
│{score: 67, stu: April} │[]        │
├────────────────────────┼──────────┤
│{score: 18, stu: April} │[]        │
├────────────────────────┼──────────┤
│{score: 66, stu: Art}   │[]        │
├────────────────────────┼──────────┤
│{score: 66, stu: Art}   │[]        │
├────────────────────────┼──────────┤
│{score: 66, stu: Art}   │[]        │
├────────────────────────┼──────────┤
│{score: 18, stu: Lynn}  │[]        │
├────────────────────────┼──────────┤
│{score: 92, stu: April} │[]        │
├────────────────────────┼──────────┤
│{score: 69, stu: Lynn}  │[]        │
├────────────────────────┼──────────┤
│{score: 76, stu: Lynn}  │[]        │
├────────────────────────┼──────────┤
│{score: 71, stu: LeeAnn}│[]        │
├────────────────────────┼──────────┤
│{score: 37, stu: Mo}    │[]        │
├────────────────────────┼──────────┤
│{score: 55, stu: Mo}    │[]        │
```

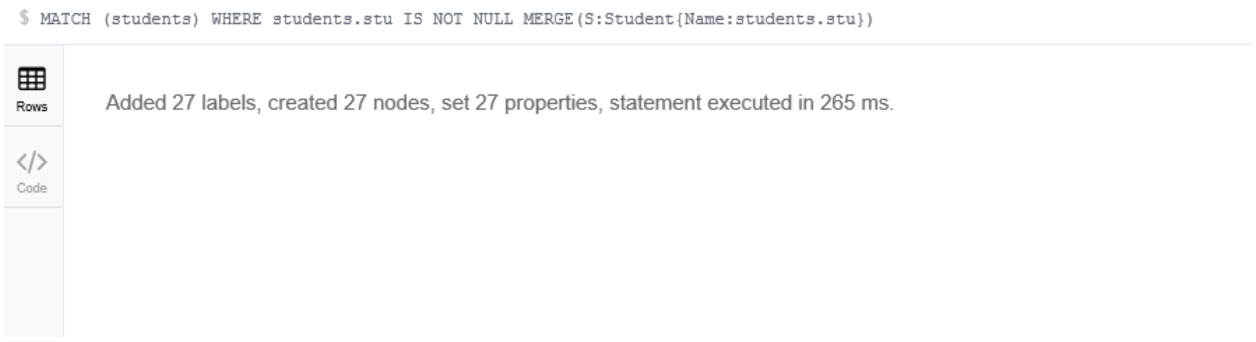
```
├────────────────────────┼─────────┤
│{score: 86, stu: Farah} │[]       │
├────────────────────────┼─────────┤
│{score: 99, stu: Farah} │[]       │
├────────────────────────┼─────────┤
│{score: 99, stu: Farah} │[]       │
├────────────────────────┼─────────┤
│{score: 86, stu: Renee} │[]       │
├────────────────────────┼─────────┤
│{score: 30, stu: LeeAnn}│[]       │
├────────────────────────┼─────────┤
│{score: 21, stu: LeeAnn}│[]       │
├────────────────────────┼─────────┤
│{score: 17, stu: Mo}    │[]       │
├────────────────────────┼─────────┤
│{score: 44, stu: Renee} │[]       │
├────────────────────────┼─────────┤
│{score: 86, stu: Renee} │[]       │
├────────────────────────┼─────────┤
│{score: 64, stu: Kurt}  │[]       │
├────────────────────────┼─────────┤
│{score: 25, stu: Kurt}  │[]       │
├────────────────────────┼─────────┤
│{score: 76, stu: Kurt}  │[]       │
├────────────────────────┼─────────┤
│{score: 86, stu: Tammy} │[]       │
├────────────────────────┼─────────┤
│{score: 35, stu: Tammy} │[]       │
├────────────────────────┼─────────┤
│{score: 90, stu: Tammy} │[]       │
├────────────────────────┼─────────┤
│{score: 51, stu: Roger} │[]       │
├────────────────────────┼─────────┤
│{score: 91, stu: Roger} │[]       │
├────────────────────────┼─────────┤
│{score: 95, stu: Elaine}│[]       │
├────────────────────────┼─────────┤
│{score: 83, stu: Elaine}│[]       │
├────────────────────────┼─────────┤
│{score: 78, stu: Elaine}│[]       │
├────────────────────────┼─────────┤
│{score: 38, stu: Susi}  │[]       │
├────────────────────────┼─────────┤
│{score: 40, stu: Susi}  │[]       │
├────────────────────────┼─────────┤
│{score: 65, stu: Roger} │[]       │
```

```
├────────────────────────┼─────────┤
│{score: 81, stu: Ed}    │[]       │
├────────────────────────┼─────────┤
│{score: 40, stu: Ed}    │[]       │
├────────────────────────┼─────────┤
│{score: 88, stu: Ed}    │[]       │
├────────────────────────┼─────────┤
│{score: 93, stu: Susi}  │[]       │
├────────────────────────┼─────────┤
│{score: 33, stu: Poneh} │[]       │
├────────────────────────┼─────────┤
│{score: 76, stu: Poneh} │[]       │
├────────────────────────┼─────────┤
│{score: 77, stu: Golpar}│[]       │
├────────────────────────┼─────────┤
│{score: 87, stu: Golpar}│[]       │
├────────────────────────┼─────────┤
│{score: 50, stu: Farouq}│[]       │
├────────────────────────┼─────────┤
│{score: 100, stu: Poneh}│[]       │
├────────────────────────┼─────────┤
│{score: 87, stu: Golpar}│[]       │
├────────────────────────┼─────────┤
│{score: 52, stu: Farouq}│[]       │
├────────────────────────┼─────────┤
│{score: 75, stu: Farouq}│[]       │
├────────────────────────┼─────────┤
│{score: 95, stu: Monir} │[]       │
├────────────────────────┼─────────┤
│{score: 68, stu: Monir} │[]       │
├────────────────────────┼─────────┤
│{score: 59, stu: Monir} │[]       │
├────────────────────────┼─────────┤
│{score: 86, stu: Mike}  │[]       │
├────────────────────────┼─────────┤
│{score: 99, stu: Mike}  │[]       │
├────────────────────────┼─────────┤
│{score: 99, stu: Mike}  │[]       │
└────────────────────────┴─────────┘
```

Returned 81 rows in 54 ms.

Q2: se Neo4j Cypher that creates a node labeled “Student” for each student in the database?

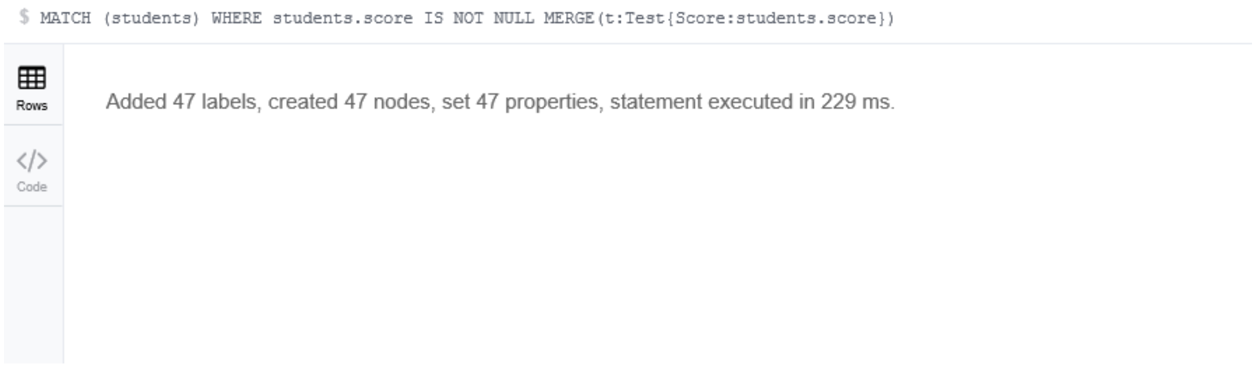
MATCH (students) WHERE students.stu IS NOT NULL MERGE(S:Student{Name:students.stu})


```
$ MATCH (students) WHERE students.stu IS NOT NULL MERGE(S:Student{Name:students.stu})
```

```
Added 27 labels, created 27 nodes, set 27 properties, statement executed in 265 ms.
```

Q3

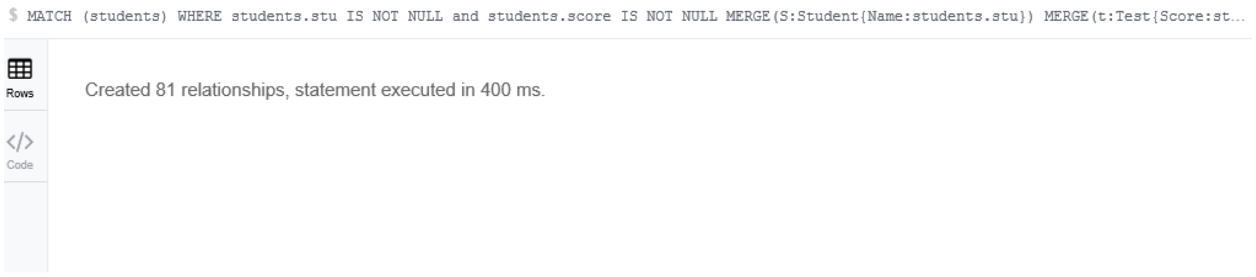
MATCH (students) WHERE students.score IS NOT NULL MERGE(t:Test{Score:students.score})


```
$ MATCH (students) WHERE students.score IS NOT NULL MERGE(t:Test{Score:students.score})
```

```
Added 47 labels, created 47 nodes, set 47 properties, statement executed in 229 ms.
```

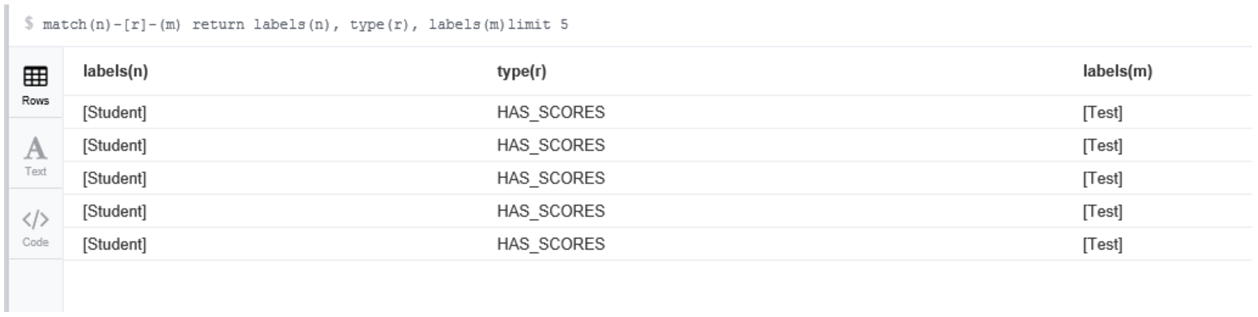
Q4

MATCH (students) WHERE students.stu IS NOT NULL and students.score IS NOT NULL MERGE(S:Student{Name:students.stu}) MERGE(t:Test{Score:students.score}) CREATE(S)-[r:HAS_SCORES]->(t);


```
$ MATCH (students) WHERE students.stu IS NOT NULL and students.score IS NOT NULL MERGE(S:Student{Name:students.stu}) MERGE(t:Test{Score:st...
```

```
Created 81 relationships, statement executed in 400 ms.
```

Check



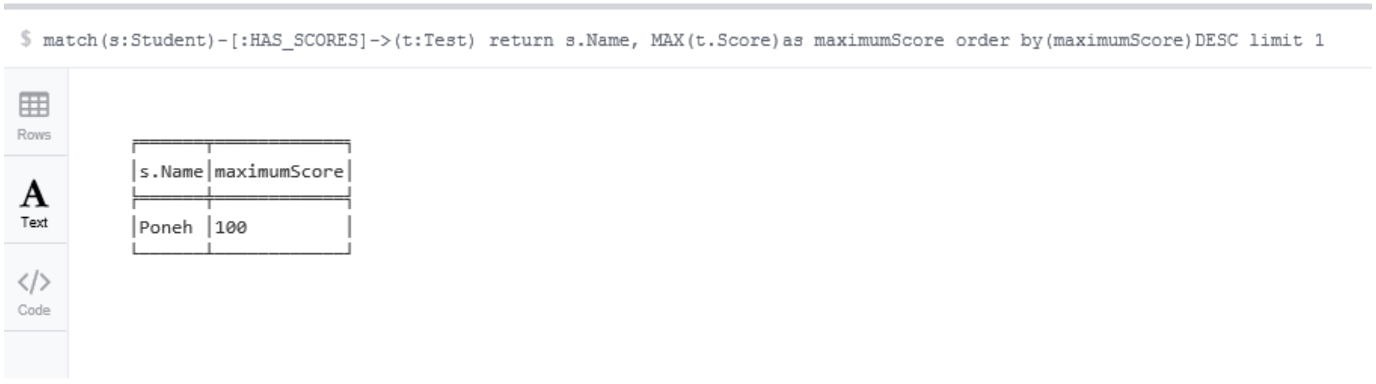
```
$ match(n)-[r]-(m) return labels(n), type(r), labels(m)limit 5
```

labels(n)	type(r)	labels(m)
[Student]	HAS_SCORES	[Test]
[Student]	HAS_SCORES	[Test]
[Student]	HAS_SCORES	[Test]
[Student]	HAS_SCORES	[Test]
[Student]	HAS_SCORES	[Test]

Q5 Print the highest score in the class and the name of all students who have scored that?

Code

match(s:Student)-[:HAS_SCORES]->(t:Test) return s.Name, MAX(t.Score)as maximumScore order by(maximumScore)DESC limit 1



```
$ match(s:Student)-[:HAS_SCORES]->(t:Test) return s.Name, MAX(t.Score)as maximumScore order by(maximumScore)DESC limit 1
```

```
╒══════╤════════════╕
│s.Name│maximumScore│
╞══════╪════════════╡
│Poneh │100         │
└──────┴────────────┘
```

$match(n) return n, labels(n)

GRAPH

Q6: For each student, print the student name, the highest_score for student, the lowest_score for the student and the average score for the student as highest_score * .6 + lowest_score * .4?

Code:

match(s:Student)-[:HAS_SCORES]->(t:Test) with s, MAX(t.Score)as maxScore, MIN(t.Score)as MinScore, Max(t.Score *0.6) as maxavg, Min(t.Score*0.4)as minavg match(s:Student)-[:HAS_SCORES]->(t:Test) return s.Name, maxScore, MinScore, avg(maxavg+minavg) as Avergage

Result:

match(s:Student)-[:HAS_SCORES]->(t:Test) with s, MAX(t.Score)as maxScore, MIN(t.Score)as MinScore, Max(t.Score *0.6)

as maxavg, Min(t.Score*0.4)as minavg match(s:Student)-[:HAS_SCORES]->(t:Test) return s.Name, maxScore, MinScore,

avg(maxavg+minavg) as Avergage

ROWS
TEXT
CODE

s.Name	maxScore	MinScore	Avergage
Mike	99	86	93.8
Kurt	76	25	55.6
Saman	76	34	59.2
Poneh	100	33	73.2
Susi	93	38	71
Renee	86	44	69.2
LeeAnn	71	21	51
Farah	99	86	93.8
Mo	55	17	39.8
Saeed	94	59	80
Sammy	77	77	77
Shah	72	19	50.8
Ed	88	40	68.8
Joe	90	34	67.6
Jordan	66	25	49.6
Tammy	90	35	68
April	92	18	62.4
Art	66	66	66
Farouq	75	50	65
Roger	91	51	75
Josh	94	21	64.8
Elaine	95	78	88.2

```
├──────┼────────┼────────┼─────────────────┤
│Sam   │81      │60      │72.6             │
├──────┼────────┼────────┼─────────────────┤
│Golpar│87      │77      │83               │
├──────┼────────┼────────┼─────────────────┤
│Lynn  │76      │18      │52.800000000000004│
├──────┼────────┼────────┼─────────────────┤
│Monir │95      │59      │80.6             │
├──────┼────────┼────────┼─────────────────┤
│Joseph│60      │39      │51.6             │
└──────┴────────┴────────┴─────────────────┘
```

Returned 27 rows in 201 ms.