MongoDB

Task 1 (import)

Load the EE5178 student CSV file, studentE.csv into collection “students” in database “hw6” in MongoDB, and write a MongoDB query to return the information (the document) about yourself. (10%)

```
lacostegege@DESKTOP-BCFOA41:~/NTUCSIE/112-2/Database_Management_System/hw6/hw6_supplements$ mongoimport --db hw6 --collection students --type csv --headerline
 --file studentE.csv
2024-06-19T11:39:26.778+0800    connected to: mongodb://localhost/
2024-06-19T11:39:26.796+0800    103 document(s) imported successfully. 0 document(s) failed to import.
lacostegege@DESKTOP-BCFOA41:~/NTUCSIE/112-2/Database_Management_System/hw6/hw6_supplements$ mongosh
Current Mongosh Log ID: 667252fbba2ededd80a26a12
Connecting to:          mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000&appName=mongosh+2.2.6
Using MongoDB:          6.0.15
Using Mongosh:          2.2.6
```

```
test> use hw6
switched to db hw6
hw6> db.students.find({ _id: "r12922116" })
[
  {
    _id: 'r12922116',
    position: '學生',
    dept: '資工所',
    year: 1,
    name: '王偉力 (WANG, WEI-LI)',
    email: 'r12922116@ntu.edu.tw',
    class: '資料庫系統-從SQL到NoSQL (EE5178)'
  }
]
```

Task 2 (aggregation pipeline)

Write a query to report the top 10 “dept" with the most number of 學生 in the “students” collection (不算旁聽生). (10%)

```
hw6> db.students.aggregate([
...     { $match: { position: "學生" }},
...     { $group: { _id: "$dept", studentCount: { $sum: 1 }}},
...     { $sort: { studentCount: -1 }},
...     { $limit: 10 }
... ])
[
  { _id: '資工所', studentCount: 14 },
  { _id: '電機所', studentCount: 11 },
  { _id: '生物機電所', studentCount: 8 },
  { _id: '電機系', studentCount: 6 },
  { _id: '經濟系', studentCount: 6 },
  { _id: '電信所', studentCount: 5 },
  { _id: '資訊管理所', studentCount: 5 },
  { _id: '生物機電系', studentCount: 4 },
  { _id: '資工系', studentCount: 4 },
  { _id: '工科海洋所', studentCount: 3 }
]
```

Task 3 (adding fields)

Import the students from “new_studentE.csv” into your “students” collection. For each student, add a new field “updated", and set it to “2024-05-23”. Write a query to return students of your department to make sure your updated is effective. (5%)

```
lacostegege@DESKTOP-BCFOA41:~/NTUCSIE/112-2/Database_Management_System/hw6/hw6_supplements$ mongoimport --db hw6 --collection students --type csv --headerline
 --file new_studentE.csv
2024-06-19T11:56:43.656+0800    connected to: mongodb://localhost/
2024-06-19T11:56:43.659+0800    3 document(s) imported successfully. 0 document(s) failed to import.
lacostegege@DESKTOP-BCFOA41:~/NTUCSIE/112-2/Database_Management_System/hw6/hw6_supplements$ mongosh
Current Mongosh Log ID: 66725700c4199df990a26a12
Connecting to:          mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000&appName=mongosh+2.2.6
Using MongoDB:          6.0.15
Using Mongosh:          2.2.6
```

```
test> use hw6
switched to db hw6
hw6> db.students.updateMany(
...     {},
...     { $set: { updated: ISODate("2024-05-23") }}
... )
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 106,
  modifiedCount: 106,
  upsertedCount: 0
}
hw6> db.students.find(
...     { dept: "資工所" },
...     { name: 1, dept: 1, updated: 1 }
... )
[
  {
    _id: 'q1990371',
    dept: '資工所',
    name: '謝十 (XIE SHI)',
    updated: ISODate('2024-05-23T00:00:00.000Z')
  },
  {
    _id: 'q1990374',
    dept: '資工所',
    name: '范三 (FAN SAN)',
    updated: ISODate('2024-05-23T00:00:00.000Z')
  },
```

```
{
  _id: 'q1990367',
  dept: '資工所',
  name: '鄭七 (ZHENG QI)',
  updated: ISODate('2024-05-23T00:00:00.000Z')
},
{
  _id: 'q1990372',
  dept: '資工所',
  name: '田四 (TIAN SI)',
  updated: ISODate('2024-05-23T00:00:00.000Z')
},
{
  _id: 'q1990368',
  dept: '資工所',
  name: '徐三 (XU SAN)',
  updated: ISODate('2024-05-23T00:00:00.000Z')
},
{
  _id: 'q1990373',
  dept: '資工所',
  name: '楊四 (YANG SI)',
  updated: ISODate('2024-05-23T00:00:00.000Z')
},
{
  _id: 'q1990369',
  dept: '資工所',
  name: '嶽六 (YUE LIU)',
  updated: ISODate('2024-05-23T00:00:00.000Z')
},
{
  _id: 'q1990376',
  dept: '資工所',
  name: '朱三 (ZHU SAN)',
  updated: ISODate('2024-05-23T00:00:00.000Z')
},
{
  _id: 'q1990370',
  dept: '資工所',
  name: '章六 (ZHANG LIU)',
  updated: ISODate('2024-05-23T00:00:00.000Z')
},
```

```
  {
    _id: 'q1990375',
    dept: '資工所',
    name: '楊三 (YANG SAN)',
    updated: ISODate('2024-05-23T00:00:00.000Z')
  },
  {
    _id: 'r12922116',
    dept: '資工所',
    name: '王偉力 (WANG, WEI-LI)',
    updated: ISODate('2024-05-23T00:00:00.000Z')
  },
  {
    _id: 'q1990377',
    dept: '資工所',
    name: '唐十 (TANG SHI)',
    updated: ISODate('2024-05-23T00:00:00.000Z')
  },
  {
    _id: 'q1990378',
    dept: '資工所',
    name: '宋五 (SONG WU)',
    updated: ISODate('2024-05-23T00:00:00.000Z')
  },
  {
    _id: 'q1990379',
    dept: '資工所',
    name: '許三 (XU SAN)',
    updated: ISODate('2024-05-23T00:00:00.000Z')
  },
  {
    _id: 'q1990398',
    dept: '資工所',
    name: '施十 (SHI SHI)',
    updated: ISODate('2024-05-23T00:00:00.000Z')
  }
]
```

Task 4 (update)

Update the “updated" field of the new students and yourself to “2024-06-01”.

Then write one query to return yourself and these new students. (5%)

```
hw6> db.students.updateMany(
...     { _id: { $in: ["b19303008", "b19303129", "r09303019", "r12922116"] } },
...     { $set: { updated: ISODate("2024-06-01") } }
... )
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 4,
  modifiedCount: 4,
  upsertedCount: 0
}
hw6> db.students.find(
...     { updated: ISODate("2024-06-01") }
... )
[
  {
    _id: 'r12922116',
    position: '學生',
    dept: '資工所',
    year: 1,
    name: '王偉力 (WANG, WEI-LI)',
    email: 'r12922116@ntu.edu.tw',
    class: '資料庫系統-從SQL到NoSQL (EE5178)',
    updated: ISODate('2024-06-01T00:00:00.000Z')
  },
  {
    _id: 'b19303008',
    position: '學生',
    dept: '太空系',
    year: 4,
    name: '劉海王 (LIU, NEPTUNE)',
    email: 'b19303008@ntu.edu.tw',
    class: '資料庫系統-從SQL到NoSQL (EE5178)',
    updated: ISODate('2024-06-01T00:00:00.000Z')
  },
  {
    _id: 'b19303129',
    position: '學生',
    dept: '太空所',
    year: 2,
    name: '吳火箭 (WU, ROCKET)',
    email: 'r19303129@ntu.edu.tw',
    class: '資料庫系統-從SQL到NoSQL (EE5178)',
    updated: ISODate('2024-06-01T00:00:00.000Z')
  },
  {
    _id: 'r09303019',
    position: '學生',
    dept: '太空所',
    year: 1,
    name: '黃人馬 (HUANG, CENTAURUS)',
    email: 'r09303019@ntu.edu.tw',
    class: '資料庫系統-從SQL到NoSQL (EE5178)',
    updated: ISODate('2024-06-01T00:00:00.000Z')
  }
]
```

Task 5 (Incremental pipeline)

Design an incremental aggregation pipeline to calculate the number of student for each “dept" for the “updated” field from some starting date to some ending date. Run the pipeline for students with the value of “modified” field from 2024-01-01 to 2024-05-31. Store your result in a “tally” collection, and print out the result.

Run the pipeline again with “updated” field from “2024-06-01” to 2024-06-30, and print our the “tally” document again. (5%)

```
hw6> db.students.aggregate([
...     { $match: { updated: { $gte: ISODate("2024-01-01 00:00:00"), $lte: ISODate("2024-05-31 23:59:59") } } },
...     { $group: { _id: "$dept", count: { $sum: 1 } } },
...     { $merge: {
...             into: "tally",
...             whenMatched: [ { $set: { count: { $add: [ "$$new.count", "$count" ] } } } ],
...             whenNotMatched: "insert"
...     } }
... ])

hw6> db.tally.find()
[
  { _id: '經濟系', count: 6 },
  { _id: '', count: 2 },
  { _id: '政治所', count: 3 },
  { _id: '醫工系', count: 2 },
  { _id: '機械所固力組', count: 2 },
  { _id: '資訊管理所', count: 5 },
  { _id: '流預所', count: 1 },
  { _id: '材料所應材班', count: 1 },
  { _id: '財金系', count: 1 },
  { _id: '基因學位學程', count: 1 },
  { _id: '網媒所', count: 3 },
  { _id: '生物機電所', count: 8 },
  { _id: '機械所製造組', count: 2 },
  { _id: '資工所', count: 14 },
  { _id: '土木所CAE組', count: 1 },
  { _id: '工科海洋所', count: 3 },
  { _id: '機械系', count: 1 },
  { _id: '工管系企管組', count: 1 },
  { _id: '材料系', count: 1 },
  { _id: '工管系科管組', count: 1 }
]
Type "it" for more
hw6> it
[
  { _id: '電信所', count: 6 },
  { _id: '動科系', count: 2 },
  { _id: '資工系', count: 5 },
  { _id: '電機所', count: 12 },
  { _id: '土木所營管組', count: 2 },
  { _id: '生醫電資所', count: 1 },
  { _id: '土木所水利組', count: 1 },
  { _id: '生物機電系', count: 4 },
  { _id: '森林環資所', count: 1 },
  { _id: '電機系', count: 8 },
  { _id: '材料所', count: 1 }
]
```

```
hw6> db.students.aggregate([
...     { $match: { updated: { $gte: ISODate("2024-01-01 00:00:00"), $lte: ISODate("2024-05-31 23:59:59") } } },
...     { $group: { _id: "$dept", count: { $sum: 1 } } },
...     { $merge: {
...             into: "tally",
...             whenMatched: [ { $set: { count: { $add: [ "$$new.count", "$count" ] } } } ],
...             whenNotMatched: "insert"
...     } }
... ])

hw6> db.students.aggregate([
...     { $match: { updated: { $gte: ISODate("2024-06-01 00:00:00"), $lte: ISODate("2024-06-30 23:59:59") } } },
...     { $group: { _id: "$dept", count: { $sum: 1 } } },
...     { $merge: {
...             into: "tally",
...             whenMatched: [ { $set: { count: { $add: [ "$$new.count", "$count" ] } } } ],
...             whenNotMatched: "insert"
...     } }
... ])

hw6> db.tally.find()
[
  { _id: '經濟系', count: 6 },
  { _id: '', count: 2 },
  { _id: '政治所', count: 3 },
  { _id: '醫工系', count: 2 },
  { _id: '機械所固力組', count: 2 },
  { _id: '資訊管理所', count: 5 },
  { _id: '流預所', count: 1 },
  { _id: '材料所應材班', count: 1 },
  { _id: '財金系', count: 1 },
  { _id: '基因學位學程', count: 1 },
  { _id: '網媒所', count: 3 },
  { _id: '生物機電所', count: 8 },
  { _id: '機械所製造組', count: 2 },
  { _id: '資工所', count: 15 },
  { _id: '土木所CAE組', count: 1 },
  { _id: '工科海洋所', count: 3 },
  { _id: '機械系', count: 1 },
  { _id: '工管系企管組', count: 1 },
  { _id: '材料系', count: 1 },
  { _id: '工管系科管組', count: 1 }
]
Type "it" for more
hw6> it
[
  { _id: '電信所', count: 6 },
  { _id: '動科系', count: 2 },
  { _id: '資工系', count: 5 },
  { _id: '電機所', count: 12 },
  { _id: '土木所營管組', count: 2 },
  { _id: '生醫電資所', count: 1 },
  { _id: '土木所水利組', count: 1 },
  { _id: '生物機電系', count: 4 },
  { _id: '森林環資所', count: 1 },
  { _id: '電機系', count: 8 },
  { _id: '材料所', count: 1 },
  { _id: '太空所', count: 2 },
  { _id: '太空系', count: 1 }
]
```

Task 6 (Merge)

Merge information in the student_groupE.csv file you're your students collection. That is, import student_groupE.csv into the students collection with the "--mode=merge" option to add the group information into each student. (5%)

Or, you can also load the student_groupE CSV file into mongoDB, then use $lookup to merge it with the student collection. Either approach get the same credits.

```
tacostegege@DESKTOP-BCFOA4T:~/NTUCSIE/112-2/Database_Management_System/hw6/hw6_supplements$ mongoimport --db hw6 --collection students --type csv --headerline
 --file student_groupE.csv --mode=merge
2024-06-19T13:14:46.067+0800    connected to: mongodb://localhost/
2024-06-19T13:14:46.076+0800    93 document(s) imported successfully. 0 document(s) failed to import.
```

Task 7

Write a pipeline to output each group with names of group members in an array. Note: those who do not have a group number (i.e.，旁聽生) should not appear in the output. (5%)

```
hw6> db.students.aggregate([
...     { $match: { group: { $exists: true, $ne: null } } },
...     { $group: { _id: "$group", members: { $push: "$name" } } },
...     { $project: { _id: 0, group: "$_id", members: 1 } },
...     { $sort: { group: 1 } }
... ])
[
  {
    members: [
      '馮四 (FENG SI)',
      '王三 (WANG SAN)',
      '田七 (TIAN QI)',
      '趙九 (ZHAO JIU)'
    ],
    group: 1
  },
  {
    members: [ '彭八 (PENG BA)', '林六 (LIN LIU)', '蔡七 (CAI QI)', '田六 (TIAN LIU)' ],
    group: 2
  },
  {
    members: [ '周八 (ZHOU BA)', '王一 (WANG YI)', '孫八 (SUN BA)', '馬三 (MA SAN)' ],
    group: 3
  },
  {
    members: [ '陳七 (CHEN QI)', '宋七 (SONG QI)', '范六 (FAN LIU)', '鄭二 (ZHENG ER)' ],
    group: 4
  },
  {
    members: [ '魯七 (LU QI)', '薛八 (XUE BA)', '唐十 (TANG SHI)', '宋五 (SONG WU)' ],
    group: 5
  },
  {
    members: [ '韓五 (HAN WU)', '何九 (HE JIU)', '范三 (FAN SAN)', '許三 (XU SAN)' ],
    group: 6
  },
  {
    members: [
      '周哲璋 (CHOU,CHE-WEI)',
      '鄧雅文 (TENG, YA-WEN)',
      '吳吉加 (CHI-CHIA WU)',
      '王偉力 (WANG, WEI-LI)'
    ],
    group: 7
  },
  {
    members: [
      '林九 (LIN JIU)',
      '高九 (GAO JIU)',
      '朱三 (ZHU SAN)',
      '章六 (ZHANG LIU)'
    ],
    group: 8
  },
  {
    members: [ '韓二 (HAN ER)', '朱十 (ZHU SHI)', '胡三 (HU SAN)', '李三 (LI SAN)' ],
    group: 9
  },
  {
    members: [ '嶽四 (YUE SI)', '崔五 (CUI WU)', '楊四 (YANG SI)', '李九 (LI JIU)' ],
    group: 10
  },
```

```
  {
    members: [ '嶽二 (YUE ER)', '章九 (ZHANG JIU)', '劉一 (LIU YI)', '嶽八 (YUE BA)' ],
    group: 11
  },
  {
    members: [ '羅八 (LUO BA)', '吳八 (WU BA)', '林四 (LIN SI)', '鄧九 (DENG JIU)' ],
    group: 12
  },
  {
    members: [ '許十 (XU SHI)', '高十 (GAO SHI)', '魯五 (LU WU)', '高七 (GAO QI)' ],
    group: 13
  },
  {
    members: [ '張五 (ZHANG WU)', '何七 (HE QI)', '林十 (LIN SHI)', '蘇八 (SU BA)' ],
    group: 14
  },
  {
    members: [ '姜七 (JIANG QI)', '卓六 (ZHUO LIU)', '陳一 (CHEN YI)' ],
    group: 17
  },
  {
    members: [ '施四 (SHI SI)', '鄧八 (DENG BA)', '謝十 (XIE SHI)', '林五 (LIN WU)' ],
    group: 18
  },
  {
    members: [ '施一 (SHI YI)', '吳一 (WU YI)', '章四 (ZHANG SI)', '楊六 (YANG LIU)' ],
    group: 19
  },
  {
    members: [ '許八 (XU BA)', '何十 (HE SHI)', '郭九 (GUO JIU)', '鄭七 (ZHENG QI)' ],
    group: 20
  },
  {
    members: [ '梁三 (LIANG SAN)', '馬五 (MA WU)', '魯十 (LU SHI)', '羅六 (LUO LIU)' ],
    group: 21
  },
  {
    members: [ '彭七 (PENG QI)', '謝三 (XIE SAN)', '韓四 (HAN SI)' ],
    group: 22
  }
]
Type "it" for more
hw6> it
[
  {
    members: [ '董六 (DONG LIU)', '田一 (TIAN YI)', '楊三 (YANG SAN)' ],
    group: 23
  },
  {
    members: [ '徐六 (XU LIU)', '郭六 (GUO LIU)', '田四 (TIAN SI)' ],
    group: 24
  }
]
```

Neo4j

Task 1 (Load CSV)

Load the student CSV, studentE.csv, into Neo4J and create one node for each student. (5%)

```
LOAD CSV WITH HEADERS FROM 'file:///studentE.csv' AS row
CREATE (:Student {
    id: row._id,
    position: row.position,
    dept: row.dept,
    year: toInteger(row.year),
    name: row.name,
    email: row.email,
    class: row.class
});
```

Table

Code

Added 103 labels, created 103 nodes, set 719 properties, completed after 110 ms.

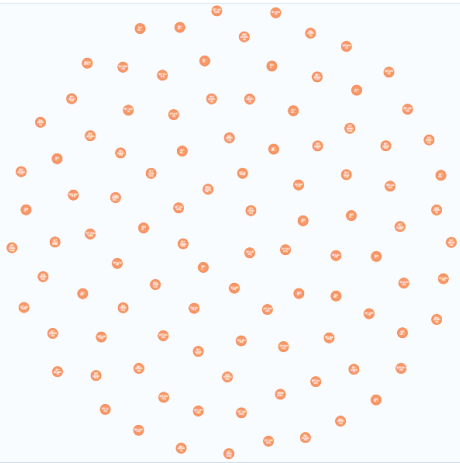
```
neo4j$ MATCH (n) return n;
```

Graph

Table

Text

Code



Overview

Node labels

* (103) Student (103)

Displaying 103 nodes, 0 relationships.

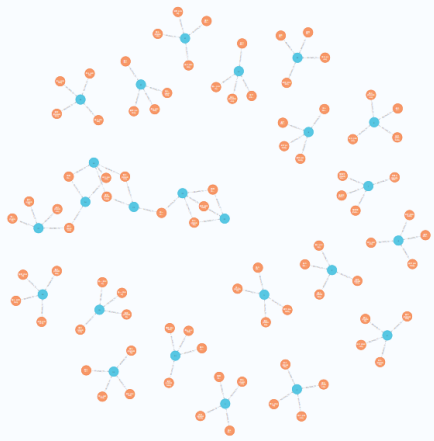
Task 2 (Load CSV and merge into graph)

Load student_groupE.csv into Neo4J, create one node for each group, and create a relationship between each student and his/her group, if the relationship exists. Accomplish all of these with one query. (10%)

```
LOAD CSV WITH HEADERS FROM 'file:///student_groupE.csv' AS row
MERGE (g:Group {name: row.group})
WITH row, g
MATCH (s:Student {id: row._id})
MERGE (s)-[:BELONGS_TO]→(g);
```

Added 24 labels, created 24 nodes, set 24 properties, created 93 relationships, completed after 60 ms.

```
neo4j$ MATCH (s:Student)-[:BELONGS_TO]→(g:Group) RETURN s, g;
```



Overview

Node labels

* (108) Student (84) Group (24)

Relationship Types

* (93) BELONGS_TO (93)

Displaying 108 nodes, 0 relationships.

Task 3 (Basic)

Write a Cypher query to return your name, and the names of your group partners, in the format below (5%)

```
MATCH (self:Student {name: '王偉力 (WANG, WEI-LI)'})-[:BELONGS_TO]→(g:Group)←[:BELONGS_TO]-
(partner:Student)
WHERE self <> partner
RETURN self.name AS Self, collect(partner.name) AS Partners;
```

	Self	Partners
1	"王偉力 (WANG, WEI-LI)"	["周哲瑋 (CHOU,CHE-WEI)", "吳吉加 (CHI-CHIA WU)", "鄧雅文 (TENG, YA-WEN)"]

Task 4 (Create and merge)

Create a new node with appropriate label and properties for each "dept" in your database, and create a relationship for each student to his/her “dept”. Do all of these using one query. (10%)

```
MATCH (s:Student)
WHERE s.dept IS NOT NULL
MERGE (d:Department {name: s.dept})
MERGE (s)-[:STUDIED_AT]→(d);
```

Table

Code

Added 30 labels, created 30 nodes, set 30 properties, created 101 relationships, completed after 59 ms.

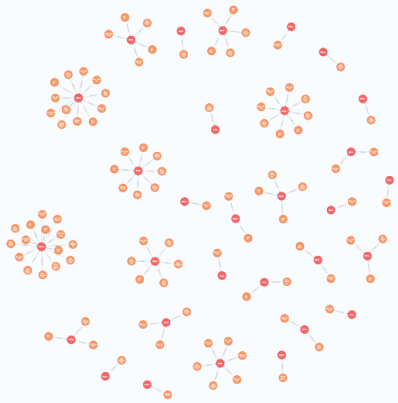
```
neo4j$ MATCH (s:Student)-[:STUDIED_AT]→(d:Department) return s, d;
```

Graph

Table

Text

Code



Overview

Node labels

* (131) Student (101) Department (30)

Relationship Types

* (101) STUDIED_AT (101)

Displaying 131 nodes, 0 relationships.

Task 5 (Aggregation)

Write a query to report the top 10 “dept” that have the most students in this graph database. (7%)

Note: you should only counting “學生", and excluding people such as “旁聽生".

```
MATCH (s:Student)
WHERE s.position = '學生'
MATCH (s)-[:STUDIED_AT]→(d:Department)
RETURN d.name AS Department, COUNT(s) AS StudentCount
ORDER BY StudentCount DESC
LIMIT 10;
```

	Department	StudentCount
1	"資工所"	14
2	"電機所"	11
3	"生物機電所"	8
4	"電機系"	6
5	"經濟系"	6
6	"電信所"	5
7	"資訊管理所"	5
8	"生物機電系"	4
9	"資工系"	4
10	"工科海洋所"	3

Started streaming 10 records after 11 ms and completed after 18 ms.

Task 6 (Advanced)

Write a query to return the group numbers (not the group nodes) of the top 5 most diversified groups (groups with the highest number of different departments), along with the names of these departments, ordered by group number in ascending order (10%).

```
MATCH (s:Student)-[:BELONGS_TO]→(g:Group)
WHERE s.dept IS NOT NULL
WITH g.name AS GroupNumber, collect(DISTINCT s.dept) AS Departments
RETURN GroupNumber, Departments
ORDER BY size(Departments) DESC, GroupNumber ASC
LIMIT 5;
```

	GroupNumber	Departments
1	"1"	["財金系", "生物機電所", "資工系", "醫工系"]
2	"10"	["基因學位學程", "機械所製造組", "資工所", "生物機電系"]
3	"11"	["生物機電所", "電機所", "經濟系", "電機系"]
4	"13"	["電機所", "土木所營管組", "動科系", "醫工系"]
5	"18"	["工管系科管組", "資工系", "資工所", "網媒所"]

Started streaming 5 records after 15 ms and completed after 17 ms.

Task 7 (Advanced: Potential partners)

Write a query to return the number of students attending this class in the department of each of your group partners (include yourself if you are in that department). The output format should be “parter_name, dept_name, dept_size” (8%)

```
MATCH (self:Student {name: '王偉力 (WANG, WEI-LI)'})-[:BELONGS_TO]→(g:Group)←[:BELONGS_TO]-
(partner:Student)
MATCH (partner)-[:STUDIED_AT]→(d:Department)
WITH partner, d
MATCH (s:Student)-[:STUDIED_AT]→(d)
RETURN partner.name AS partner_name, d.name AS dept_name, COUNT(s) AS dept_size;
```

	partner_name	dept_name	dept_size
1	"周哲瑋 (CHOU,CHE-WEI)"	"機械系"	1
2	"吳吉加 (CHI-CHIA WU)"	"電機所"	12
3	"鄧雅文 (TENG, YA-WEN)"	"電機所"	12

Started streaming 3 records after 11 ms and completed after 13 ms.