ISYS2095 – Assessment 2

Adam Mutimer (S3875753)

Part A: SQL Programming

Task 1: Non-Nested Queries

Question 1.1:

Total rows loaded: 31

	DEPTNAME	DeptNameCount
1	Computer Science	11
2	Department of Computer Science	3
3	School of Information Technology	3
4	CRC for Distributed Systems Technology	2
5	Applied Computing and Mathematics	1
6	Basser Department of Computer Science	1
7	Centre for Information Technology Research	1
8	Computer Science Department	1
9	Computer Science Laboratory	1
10	Computer Science and Computer Engineering	1

Figure 1 - Question 1.1 - Query Results

```
SELECT department.deptname,
       count(department.deptname) AS DeptNameCount
  FROM department
 WHERE department.deptname IS NOT NULL AND
       'DeptNameCoun' != 0
 GROUP BY department.deptname
 ORDER BY DeptNameCount DESC,
          department.deptname ASC;
```

Question 1.2:

Total rows loaded: 10

	title	givename	famname	PaperCount
1	Dr	Gerard	Shetty	6
2	Dr	Ron	Das	3
3	Dr	Lee	Reichelt	2
4	Dr	Xuming	Rooney	2
5	Dr	Glenn	Fitzgerald	1
6	Dr	Gerald	Grav	1
7	Dr	Jack	Hlynka	1
8	Dr	Richard	Nappi	1
9	Dr	Gilles	Preece	1
10	Dr	Alan	Santhanakrishnan	1

Figure 2 - Question 1.2 - Query Results

```
SELECT   academic.title,
         academic.givename,
         academic.famname,
         Count(paper.title) AS PaperCount
     FROM academic, interest, paper, author
     WHERE academic.title like 'dr'
         AND interest.descrip like '%database%'
         AND interest.acnum = academic.acnum
         AND paper.title like '%database%'
         AND paper.panum = author.panum
         AND author.acnum = academic.acnum
     GROUP BY academic.acnum
     ORDER BY PaperCount DESC, academic.famname ASC,
academic.givename ASC;
```

Task 2: Nested Queries

Question 2.1:

Total rows loaded: 2

	deptnum	deptname	state
1	131	CRC for Distributed Systems Technology	Qld
2	143	Gippsland School of Computing and I.T.	VIC

Figure 3 - Question 2.1 Query Results

```
SELECT  department.deptnum,
        department.deptname,
        department.state
    FROM department
    WHERE UPPER(department.state) IN ("VIC", "QLD")
        AND department.deptnum NOT IN ( SELECT academic.deptnum FROM
academic );
```

Question 2.2:

Total rows loaded: 2

	deptnum	deptname	state
1	131	CRC for Distributed Systems Technology	Qld
2	143	Gippsland School of Computing and I.T.	VIC

Figure 4 - Question 2.2 Query Results

```
SELECT  department.deptnum,
        department.deptname,
        department.state
    FROM department
    WHERE NOT EXISTS( SELECT academic.deptnum FROM academic WHERE
academic.deptnum = department.deptnum GROUP BY academic.deptnum )
    AND UPPER(department.state) IN ("VIC", "QLD");
```

Question 2.3:

Total rows loaded: 11

	acnum	Full Name
1	290	AProf. Jonathan Garber
2	232	Prof. Pete Grohol
3	264	Dr. Eduardo Harding
4	200	Prof. Thomas Hedges
5	199	Mr. Svante Karr
6	201	Mrs. Hans Kellner
7	282	Dr. Roger Khadye
8	203	Dr. Iain Mitchell
9	206	Dr. Edward Robinson
10	204	Dr. Rob Schaeffer
11	205	Mr. Will Zakel

Figure 5 - Question 2.3 Query Results

NOTE: Question specified only to show the full name of the academics, It did not specify if we needed to concatenate the 3 columns or display them individually. I assumed it required them to be concatenated. *Used: Title + Given Name + Family Name*

```
SELECT academic.acnum,
       academic.title || '. ' || academic.givename || ' ' ||
academic.famname AS [Full Name]
  FROM academic
 WHERE academic.acnum IN (
           SELECT author.acnum
             FROM author
            WHERE academic.acnum IN (SELECT acnum from author WHERE
panum IN (SELECT panum from author WHERE ACNUM=202) GROUP BY ACNUM)
       )
       AND academic.acnum != 202
   GROUP BY acnum
   ORDER BY academic.FAMNAME, academic.givename
```

Question 2.4:

Total rows loaded: 1

	deptname
1	Computer Science

Figure 6 - Question 2.4 Query Result

```
SELECT deptname
  FROM (
        SELECT deptname,
               MAX(occur)
           FROM (
                 SELECT deptname,
                        Count(deptname) AS occur
                   FROM department
                   GROUP BY deptname
                 )
        );
```

Task 3: Set Operators

Question 3.1:

Total rows loaded: 2

	acnum
1	147
2	168

Figure 7 - Question 3.1 Query Results

```
SELECT academic.acnum
  FROM academic
EXCEPT
SELECT author.acnum
  FROM author
INTERSECT
SELECT acnum
  FROM (
           SELECT acnum,
                  Count(interest.acnum) AS count
             FROM INTEREST
            GROUP BY interest.acnum
           HAVING count >= 5
       );
```

Question 3.2:

Total rows loaded: 6

	acnum
1	108
2	117
3	322
4	352
5	363
6	444

Figure 8 - Question 3.2 Query Results

```
-- Select All academics:
SELECT academic.acnum
  FROM academic

-- Remove Academics that have not authored any papers AND Remove 114:
INTERSECT
SELECT interest.acnum
  FROM interest
 WHERE interest.acnum != 114

-- Remove Academics with out matching interest fields and total
matches matching 114 interest count
INTERSECT
SELECT interest.acnum
  FROM interest
 WHERE fieldnum IN (
           SELECT interest.fieldnum
             FROM interest
            WHERE interest.acnum = 114
       )
 GROUP BY interest.acnum
HAVING Count(interest.acnum) = (
                          SELECT Count(interest.fieldnum)
                            FROM interest
                           WHERE interest.acnum = 114
                      );
```

Part B: Normalisation

Task 4: Relational Database Design

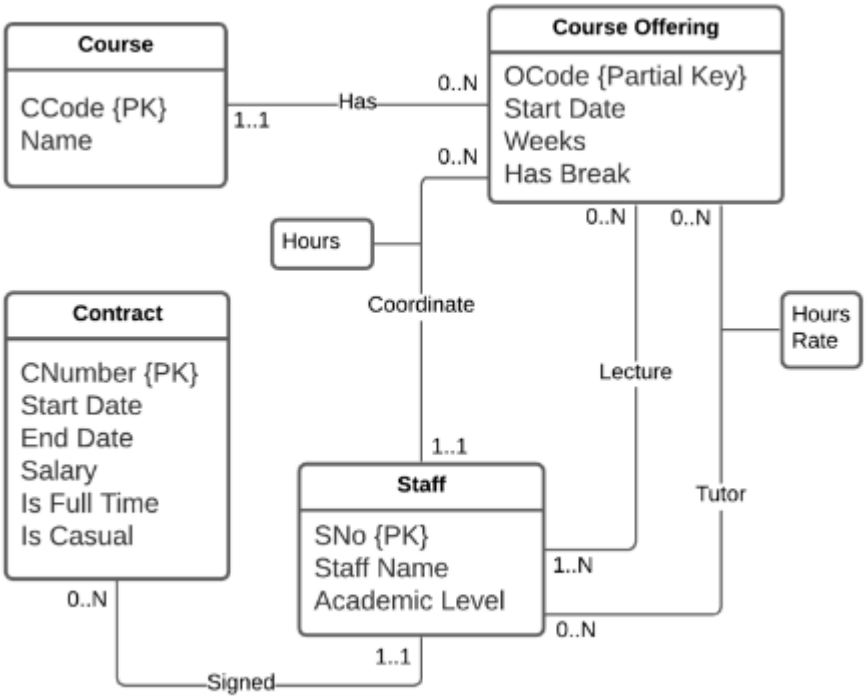

Figure 9 : University ER Diagram (Figure 2)

Course(CCode, Name)
CourseOffering(CCode*,OCode,Start Date,Weeks,Has Break)
Contract(CNumber, Start Date, End Date, Salary,Is Full Time, Is Casual,SNo*,Staff Name*)
Staff(SNo,StaffName,Academic Level)
Lecture(CCode*,OCode*,SNo*)
Tutor(CCode*,OCode*,SNo*,Hours,Rate)
Coordinate(CCode*,OCode*,SNo*,Hours)

Figure 10: Supplied Schema

Question 4.1.1:

Course(CCode, Name)

FD1: CCode → Name

CourseOffering(CCode*,OCode,Start Date,Weeks,Has Break)

FD1: CCode, OCode → Start Date, Weeks, Has Break

Contract(CNumber, Start Date, End Date, Salary,Is Full Time, Is Casual,SNo*,Staff Name*)

FD1: CNumber → Start Date, End Date, Salary, Is Full Time, Is Casual

FD2: SNo, Staff Name → CNumber

FD3: SNo → Staff Name

Staff(SNo,StaffName,Academic Level)

FD1: SNo → StaffName, Academic Level

Lecture(CCode*,OCode*,SNo*)

Trivial Functional Dependancey

Tutor(CCode*,OCode*,SNo*,Hours,Rate)

FD1: CCode, OCode, SNo → Hours, Rate

Coordinate(CCode*,OCode*,SNo*,Hours)

FD1: CCode, OCode, OCode, SNo → Hours

Question 4.1.2:

From my analysis and understanding of the database schema, ER diagram and the functional dependencies the table Contract is incorrect.

At first glance the solution would be to change the Functional dependency to this:

FD1: SNo, CNumber → Start Date, End Date, Salary, Is Full Time, Is Casual

However we are missing important information such as has the contract been signed or not, we are also storing "Staff Name" in the contract table this is a waste of space and has no use to us as we can lookup "Staff Name" using "SNo". So "Staff Name" will be removed from this table.

So my solution is the following:

Signed(CNumber*, SNo*, ContractSigned)

FD1: CNumber, SNo → ContractSigned

Contract(CNumber, Start Date, End Date, Salary,Is Full Time, Is Casual)

FD1: CNumber → Start Date, End Date, Salary, Is Full Time, Is Casual

Question 4.2.1:

Course(CCode, Name)

CCode → Name

1NF – I believe this holds the form of 1NF as both CCode and Name can only hold a single attribute making them atomic and each column contains the same data type. Does not use a composite key.

CourseOffering(CCode*,OCode,Start Date,Weeks,Has Break)

CCode, OCode → Start Date, Weeks, Has Break

2NF – I Belive this hold the form of 2NF as it is in the form of 1NF and has no partial dependancies. All non key attributes are dependant on the entire composite primary key.

Contract(CNumber, Start Date, End Date, Salary,Is Full Time, Is Casual)

CNumber → Start Date, End Date, Salary, Is Full Time, Is Casual

1NF – I believe this also hold the form of 1NF as CNumber, Start Date, End Date, Salary, Is Full Time and Is Casual can only hold single attributes making them atomic and each column contains the same data type. Does not use a composite key.

Signed(CNumber*, SNo*, ContractSigned)

FD1: CNumber, SNo → ContractSigned

2NF - I Belive this hold the form of 2NF as it is in the form of 1NF and has no partial dependancies. All non key attributes are dependant on the entire composite primary key.

Staff(SNo,StaffName,Academic Level)

SNo → StaffName, Academic Level

1NF – I believe this also hold the form of 1NF as SNo. Staff Name, Academic Level can only hold single attributes making them atomic and each column contains the same data type. Does not use a composite key.

Lecture(CCode*,OCode*,SNo*)

3NF – I think this is in 3NF as I don't belive it holds the form 1NF or 2NF but I may be incorrect on this however.

Tutor(CCode*,OCode*,SNo*,Hours,Rate)

CCode, OCode, SNo → Hours, Rate

2NF - I Belive this hold the form of 2NF as it is in the form of 1NF and has no partial dependancies. All non key attributes are dependant on the entire composite primary key.

Coordinate(CCode*,OCode*,SNo*,Hours)

CCode, OCode, OCode, SNo → Hours

2NF - I Belive this hold the form of 2NF as it is in the form of 1NF and has no partial dependancies. All non key attributes are dependant on the entire composite primary key.

Question 4.2.2:

The new database Schema is as follows:

Course(CCode, Name)
CourseOffering(CCode*,OCode,Start Date,Weeks,Has Break)
Staff(SNo,StaffName,Academic Level)
Contract(CNumber, Start Date, End Date, Salary,Is Full Time, Is Casual)
Signed(CNumber*, SNo*, ContractSigned)
Lecture(CCode*,OCode*,SNo*)
Tutor(CCode*,OCode*,SNo*,Hours,Rate)
Coordinate(CCode*,OCode*,SNo*,Hours)