ISYS2095 – Assessment 2

Adam Mutimer (S3875753)

# Part A: SQL Programming

## Task 1: Non-Nested Queries

### Question 1.1:



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:



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:



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:



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:



Figure - 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:



Figure - 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:



Figure - 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:



Figure - 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

Diagram

Description automatically generated

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:

This honestly still makes no sense to me?!?

|  |
| --- |
| 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

|  |
| --- |
| 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: SNo 🡪 CNumber  
FD2: 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**This is First Normal Form (1NF) as the “Code” is unique and “Name” is a single attribute column and has no other dependant relations

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

**CCode, OCode 🡪 Start Date, Weeks, Has Break**I belive this is a 2NF as the table itself contains 1NF atribuites but is reliant on the ***Course*** table its “CCode” meaning it has two relations

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

**CNumber 🡪 Start Date, End Date, Salary, Is Full Time, Is Casual**This is 1NF as it has no external relationships, contains only single attribute columns and has no dependant relationships

|  |
| --- |
| Signed(CNumber\*, SNo\*, ContractSigned) |

FD1: SNo 🡪 CNumber  
FD2: CNumber, SNo 🡪 ContractSigned

This is a 2NF, as it has two dependant relationships to form its composite key “CNumber” from ***Contract*** table and “SNo” from ***Staff*** table

|  |
| --- |
| Staff(SNo,StaffName,Academic Level) |

**SNo 🡪 StaffName, Academic Level**This is 1NF as it has no external relationships, contains only single attribute columns and has no dependant relationships

|  |
| --- |
| Lecture(CCode\*,OCode\*,SNo\*) |

This junction table is belive is a 2NF as it has relations

|  |
| --- |
| Tutor(CCode\*,OCode\*,SNo\*,Hours,Rate) |

FD1: CCode, OCode, SNo 🡪 Hours, Rate

|  |
| --- |
| Coordinate(CCode\*,OCode\*,SNo\*,Hours) |

FD1: CCode, OCode, OCode, SNo 🡪 Hours

### Question 4.2.2:

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) |