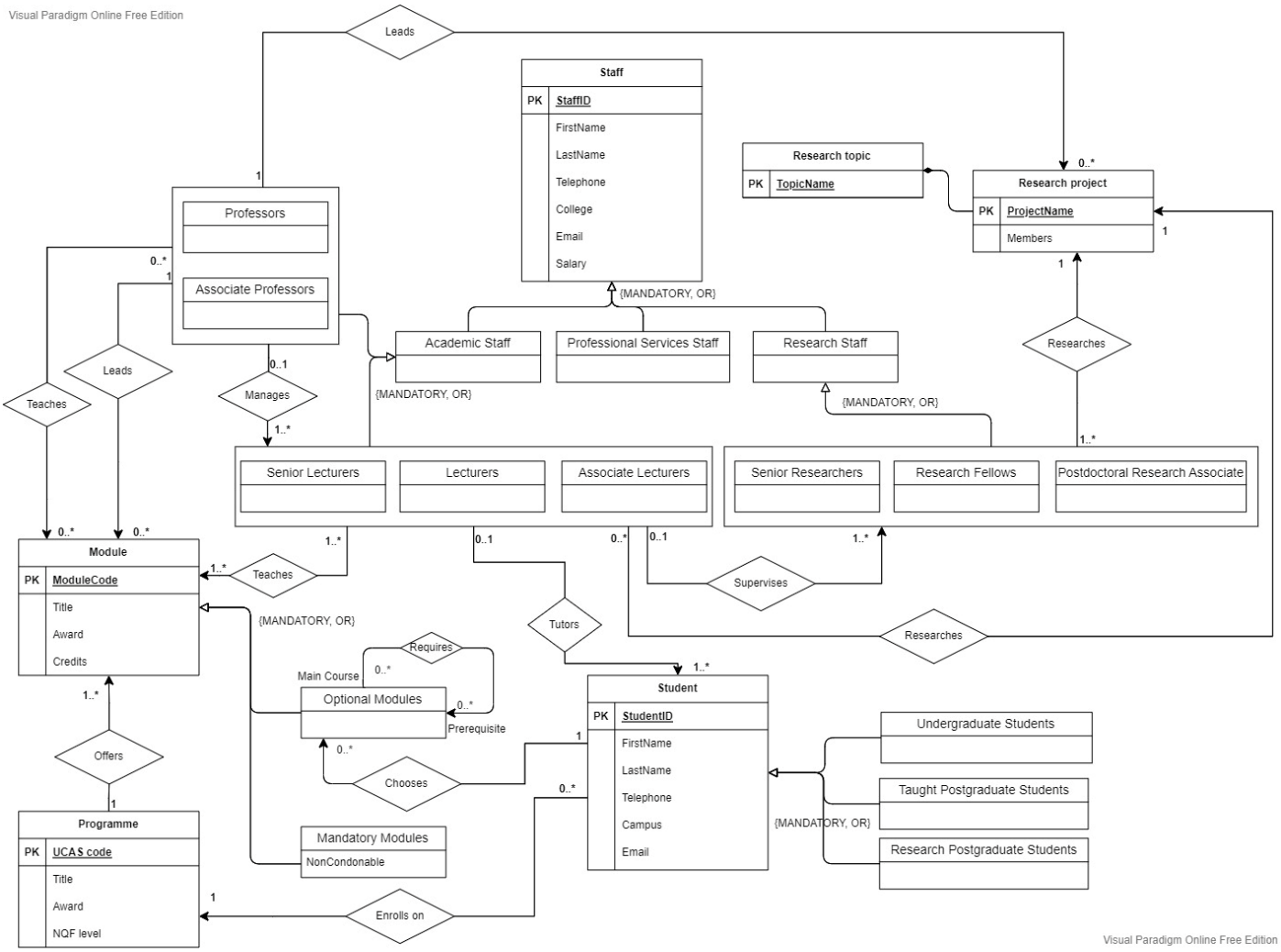
1.1 ERD design of the database



1.2 ERD explanation

The above ERD shows a high-level design for the department of computer science at the University of Exeter. The main components of said ERD revolve around the relationships between staff, students, programmes, and research.

Looking initially at Staff we see a generalised entity using StaffID as the primary key. This use of a primary key was an assumption made by me. I deduced the existence of a StaffID when considering one exists for students. Staff can be specialised into 3 different types, notably academic staff, professional services staff, and research staff. The tag {MANDATORY, OR} shows that all staff must belong to one of these three sub-categories. Furthermore, research staff can be further specialised as senior researchers, research fellows or postdoctoral research associates. Once again, the same {MANDATORY, OR} tag is used to denote that all research staff must belong to one of these sub-types. This specialisation is also present for academic staff, where they can be professors, associate professors, senior lectures, lecturers, or associate lecturers. These specialisations also have the {MANDATORY, OR} tag.

Continuing to the Student entity I have used the StudentID as the primary key. This value was used as the primary key as it is a unique digit given to all students, where all other attributes could be dependant on this one. Like staff the student entity is a generalisation, depicted by the hollow arrow of three sub-entities. These include undergraduate students, taught postgraduate students and research postgraduate students. These specialisations also include the {MANDATORY, OR} tag.

The research project entity is a composite entity of the research topic entity. This was justified by looking at the current research undergoing in the computer science department and noticing that all the individual projects could

be classified under a research topic. As primary keys I used topic name for research topic and project name for research project. I used these as browsing through the research pages on the department pages lead to a fruitless search for other attributes or details on the individual research projects.

Finally looking under the programme entity we can see I used the UCAS code as the primary key. This made sense as it was a good unique identifier for each programme offered by the university. Linked to the programme is the module entity through the offers relationship. The cardinality of this relationship shows that every programme must have at least one module with no upper bound it can offer. Like the UCAS code I used the module code for the primary key of the module entity as it gave a concise unique identifier for every module. The module entity can also be specialised into two sub-entities, which include both optional modules and mandatory modules. This specialisation contains the {MANDATORY, OR} tag as all modules must be one of these two types. The optional modules entity has a relationship with itself showing that to take a given module it might require a certain unknown number of prerequisites. The cardinality of this relationship represents this unknown number of prerequisites.

Looking at relationships with the student entity we can see each student can choose a certain number of optional modules. The cardinality represents uncertainty in the number of optional modules a student can choose. Furthermore, students can also enrol onto programmes. The cardinality of this relationship shows zero to many students can enrol on any programme, but they can enrol onto at most one programme. Students are also affected by one relationship. Senior lecturers, lecturers, and associate lecturers, grouped up in the ERD can tutor students. The cardinality of this relationship shows that not all lecturers and co must tutor students, but when tutoring students, they must tutor at least one with an unspecified upper bound.

Further looking at this group of lecturers and co we see that they have a total of 5 relationships. The tutors relationship to students. A teaches relationship to modules, representing at least 1 lecturer can teach a minimum of 1 module. No upper bound is specified to represent multiple lecturers being able to teach one module, and lecturers being able to teach multiple modules. In addition, lecturers and co can also research at most one research project. This was an assumption made by me. Browsing through the research being conducted in the computer science I noticed lecturers would only ever be involved in one project at once. The cardinality also shows that not all lecturers must be involved in a research project, but there is no upper bound in terms of lecturers involved in one. Senior researchers, research fellows and postdoctoral research associates also have this relationship with research project, with one slight caveat. The cardinality shows that researchers must be involved in one research project, whereas lectures were not. Finally, lecturers and co also have a relationship with researchers and co where an unspecified number of lecturers can supervise 1 or more researchers. This was an assumption found on the staff page, where researchers would be stated to be supervised by certain lecturers or professors. This was further found when looking at outlook, clicking on researcher profiles showed they were supervised by certain lecturers and professors.

Like lecturers supervising researchers, professors, (professors and associate professors), can manage lecturers. The manages relationship between these entities shows that not all professors must manage lecturers, but all lecturers must be managed by at least one professor. Furthermore, looking at the relationships of professors we see that every research project and module must be led by one professor. The cardinality shows that only one professor can lead a module and research project. This was once again deduced by looking at the department for computer science and looking in detail at the module and research pages. Analysing these pages revealed that every module and research project would only be led by one professor. Finally, professors can also teach modules. The cardinality on this relationship shows that not all professors must teach, but multiple can teach the same module. In addition, professors can also teach multiple modules.

The attributes for the staff entity were based around assumptions, i.e. the attributes FirstName, LastName and salary. As well as looking at individual profile pages for staff members which showed their email and telephones were also in the database. Attributes for the student entity were based by looking at the student registration services and seeing what information was stored there. For the programme and module entity I looked at the programme list as well as module lists for individual programmes and found which attributes were stored alongside the UCAS and module codes for each individual programme and module. One extra attribute was added for mandatory modules, the nonCondonable attribute was unnecessary for optional modules as no optional modules are non-condonable.

Finally, I added the members attribute for research project after having a look at what information was publicly displayed under each research project. This just contained the member list of staff involved in said research project.

2.1 and 2.2: Normalisation and Documentation

Staff Teaching Records Original Data

Staff Number	Staff Name	Position	Year of Hire	Module Number	Research Group	Group Lead	Module Name	Term	Number of Times
35	Mark	Lecturer	2018	ECM1400	ML&CV	Sarah	Programming	1	1
				ECM2418			Computer Languages	1	1
				ECM2433			The C Family	2	1
11	Jennifer	Lecturer	2019	ECM3423	CS	Peter	Computer Graphics	1	2
				ECM1400			Programming	1	1
23	Mat	Professor	2014	ECM3408	ML&CV	Sarah	Enterprise Computing	2	3
				ECM3423			Computer Graphics	1	2
				ECM1400			Programming	1	2
36	Bob	Associate Professor	2016	ECM3408	HPC	Jack	Enterprise Computing	2	1
				ECM2433			The C Family	2	3
				ECM3423			Computer Graphics	1	2
				ECM2418			Computer Languages	1	4

Staff Teaching Records 1st Normal Form

Staff Number	Staff Name	Position	Year of Hire	Module Number	Research Group	Group Lead	Module Name	Term	Number of Times
35	Mark	Lecturer	2018	ECM1400	ML&CV	Sarah	Programming	1	1
35	Mark	Lecturer	2018	ECM2418	ML&CV	Sarah	Computer Languages	1	1
35	Mark	Lecturer	2018	ECM2433	ML&CV	Sarah	The C Family	2	1
11	Jennifer	Lecturer	2019	ECM3423	CS	Peter	Computer Graphics	1	2
11	Jennifer	Lecturer	2019	ECM1400	CS	Peter	Programming	1	1
23	Mat	Professor	2014	ECM3408	ML&CV	Sarah	Enterprise Computing	2	3
23	Mat	Professor	2014	ECM3423	ML&CV	Sarah	Computer Graphics	1	2
23	Mat	Professor	2014	ECM1400	ML&CV	Sarah	Programming	1	2
36	Bob	Associate Professor	2016	ECM3408	HPC	Jack	Enterprise Computing	2	1
36	Bob	Associate Professor	2016	ECM2433	HPC	Jack	The C Family	2	3
36	Bob	Associate Professor	2016	ECM3423	HPC	Jack	Computer Graphics	1	2
36	Bob	Associate Professor	2016	ECM2418	HPC	Jack	Computer Languages	1	4

The first normal form can also be represented by the following relation schema:

StaffTeachingRecords (staffNumber, moduleNumber, staffName, position, yearOfHire, researchGroup, groupLead, moduleName, term, numberOfTimes)

I identified the primary keys to be the staffNumber and the moduleNumber. This was informed by looking at the data and finding that these were the only two pieces of data needed to distinguish every column from each other. I also assumed that multiple staff could have the same name or yearOfHire, as well as multiple modules possible having the same name. This would then mean that the only unique identifier would be the staffNumber and moduleNumber.

To find the first normal form I removed any repeating groups found in the original data. These included staffNumber, staffName, position, yearOfHire, researchGroup and groupLead. Removing these repeating groups lead to the above table.

To find the second normal form I removed partial dependencies. Partial dependencies are a form of functional dependencies. Which state that for two attributes A and B of a relation R, if each value of A is associated with exactly one value of B, B is said to be functionally dependent on A, denoted as A -> B. Furthermore, for partial dependencies we analyse the overall dependency and see whether removing certain attribute(s) from A still makes the overall dependency A -> B hold. To achieve this I looked at which attributes linked to the keys staffNumber and moduleNumber, when removed still make the overall dependency hold.

Applying this strategy to the first normal form of the staff teaching records lead me to the following relation schema representing the 2nd norm form staff teaching records.

Staff Teaching Records 2nd Normal Form

StaffTeachingRecords (staffNumber, moduleNumber, groupLead, numberOfTimes)

Staff (staffNumber, staffName, position, yearOfHire, researchGroup)

Module (moduleNumber, moduleName, term)

The data tables for the 2nd normal form looked like the following:

Staff Teaching Records

Staff Number	Module Number	Group Lead	Number of Times
35	ECM1400	Sarah	1
35	ECM2418	Sarah	1
35	ECM2433	Sarah	1
11	ECM3423	Peter	2
11	ECM1400	Peter	1
23	ECM3408	Sarah	3
23	ECM3423	Sarah	2
23	ECM1400	Sarah	2
36	ECM3408	Jack	1
36	ECM2433	Jack	3
36	ECM3423	Jack	2
36	ECM2418	Jack	4

Staff

Staff Number	Staff Name	Position	Year of Hire	Research Group
35	Mark	Lecturer	2018	ML&CV
11	Jennifer	Lecturer	2019	CS
23	Mat	Professor	2014	ML&CV
36	Bob	Associate Professor	2016	HPC

Module

Module Number	Module Name	Term
ECM1400	Programming	1
ECM2418	Computer Languages	1
ECM2433	The C Family	2
ECM3423	Computer Graphics	1
ECM3408	Enterprise Computing	2

Finally, to find the 3rd normal form I had to remove all transitive dependencies. These dependencies are attributes not directly dependant on an attribute A, but dependant on its transitive dependency B.

Looking at the 2nd normal form of the staff teaching records I noticed that it included a transitive dependency between the staffNumber and the groupLead.

This transitive dependency can be denoted by:

staffNumber -> researchGroup -> groupLead

Taking this into consideration led me to finding the 3rd normal form for the staff teaching records.

Staff Teaching Records 3rd Normal Form

StaffTeachingRecords (staffNumber, moduleNumber, numberOfTimes)

Staff (staffNumber, staffName, position, yearOfHire, researchGroup)

Module (moduleNumber, moduleName, term)

Research (researchGroup, groupLead)

The data tables for the 3rd normal form looked like the following:

Staff Teaching Records

Staff Number	Module Number	Number of Times
35	ECM1400	1
35	ECM2418	1
35	ECM2433	1
11	ECM3423	2
11	ECM1400	1
23	ECM3408	3
23	ECM3423	2
23	ECM1400	2
36	ECM3408	1
36	ECM2433	3
36	ECM3423	2
36	ECM2418	4

Staff

Staff Number	Staff Name	Position	Year of Hire	Research Group
35	Mark	Lecturer	2018	ML&CV
11	Jennifer	Lecturer	2019	CS
23	Mat	Professor	2014	ML&CV
36	Bob	Associate Professor	2016	HPC

Module

Module Number	Module Name	Term
ECM1400	Programming	1
ECM2418	Computer Languages	1
ECM2433	The C Family	2
ECM3423	Computer Graphics	1
ECM3408	Enterprise Computing	2

Research

Research Group	Group Lead
ML&CV	Sarah
CS	Peter
HPC	Jack