Entity-Relationship Diagram report

Overall assumptions and justification:

- That to keep track of a student's 'learning' we use assessments. Justification – this is how our learning is measured at university.
- That research is performed within groups. Justification – this is how research is conducted within the university

Entities

Element(s)	Explanation / Justification
Faculty	The primary key of this entity is 'Faculty_Name'. An assumption I've made here is that each faculty has a unique name which seems justifiable as it would not make sense to have multiple faculties with the same name, and thus this attribute can be used as the primary key. An attribute I included in this entity is the 'Faculty_Leader' because it holds reference to the id of the leader of a faculty.
Department	The primary key of the Department entity is the 'Department_Name'. Here I made the assumption that name is unique allowing it to be used as the primary key. This seems justifiable because there should be no departments with the same name as I indicated on the Exeter website. Attributes I included are: - 'Facililites_Description', to describe the facilities within the department - 'Contact_number', so that the department can be contacted regarding any questions.
Staff	Staff is represented as an entity as it has relationships and attributes. The primary key is' Staff_ID' as it will be unique to all staff members; generated when they join a department for the first time. Attributes I deemed staff required include 'Staff_Title', 'Salary', 'Address', 'Email', 'Year_Of_Hire', and 'department_Name'. Justifying these attributes, each staff member would be paid a regular salary, and Year_Of_Hire can be used to deduce the length of time a staff has been in the department. The staff_Title and email were included as research on the Exeter website (provided in the specification) showed this information was often included. The address and department name were also added as they fit the purpose of the database; which is keeping track of its academic staff; therefore I believe data should be stored about where they live and the department they work in.
Research staff	Research staff is represented as an entity in my design because it has both relationships and unique and shared attributes. The primary key of this entity is the shared attribute 'Staff_ID'. The unique attribute I've included here is 'Staff_Bio'. My justification for including this attribute would be my analysis of existing research staff who often include information in their profiles. An assumption I'm making here is that all research staff will include this information, in the case I'm wrong then NULL information would be stored in these attributes.
Research Group	Under the assumption that each group is given a unique title, I've assigned 'Group_Title' as the primary key. My assumption can be justified as there would be no reason to have multiple research groups with identical names as indicated on the university website. Moreover, using this existing information as the primary key makes sense rather than generating a new unique Id. Another attribute I've included is the 'Group_Lead'; providing reference to the foreign key 'Staff_ID' that is assigned as the leader of the group. Justifying this, each research group on the university website has a group leader.
Project	I have created an entity called 'Project' to represent a research group project. The attribute 'Project_Name' is the primary key of this entity however; this is under the assumption that no other projects exist with an identical name. I believe I can justify this as an identical project name would be an act of plagiarism and should therefore not occur within the university. Moreover, the attributes 'Start_Date' and 'Aim' are included as these provide information on the goal of the project and the ability to derive how long the project has existed.
Academic staff	Academic staff is represented as an entity because it has unique attributes, shared attributes, and many relationships. The primary key of this entity is the shared attribute 'Staff_ID'. This can be used as the primary key because it is unique for all staff members. This entity also contains 'Staff_Bio' with the same assumptions and justification as the research entity.
Module	The module entity represents a module a student would take as part of a programme. The primary key of this entity is the 'Module_Code' as it provides a unique reference to each module; justifying its use. Further attributes I believe belong in the entity are: - 'Title', which provides a human-readable name for the module. - 'Compulsory', which indicates whether a module is required to pass the programme - 'Credits', which provide the value of the module in credits - 'Noncondonable', which indicates whether a module is condonable - 'ModuleLeader', which is a foreign key referring to the 'Staff_ID' of the leader of the module. To further justify these attributes, modules on the university website also contain this information and all have a designated module leader.
Programme	The 'Programme' entity has a composite primary key using the 'Programme_Name' and the 'Awarded' attributes. I choose to design my ER in this way as the list of programmes provided on the university website can all be uniquely identified using these two attributes. Some programmes had the same programme name, and for that reason, I choose to use a composite primary key. - 'Programme_Name' provides an identifier for the program - 'Admission_criteria' describes what is required for a student to enter a program. - 'UCAS_Code' links the program to its UCAS information. I cannot use this code as the primary key for the programme as some programmes do not have a UCAS code (as seen in the programmes list provided on the university website). - 'Description' attribute provides an overview of the programme contents. - 'Awarded' provides details about the degree awarded on completion of the program. To further justify these attributes, programmes on the university website all contain this information.
Student	The primary key of Student is 'Student_ID'. To justify its use, it will be uniquely generated for each new student that joins the university. Attribute 'Name' is included as it provides a reference to the student's name, 'Current_Address', 'Phone_Number', and 'Email' are included as they provide a way to contact the student. 'Birth_Date' and 'Gender' are included as they provide information that should be held about a student; justifiably as we provided this information on our enrolment. The student entity also contains the attribute 'Tutor_ID' which provides a reference to the primary key of 'academic staff', 'Staff_ID' which is itself a shared attribute with

	the 'Staff' entity. I've given the 'student' entity the 'Tutor_ID' as it can be updated from NULL to an id when a student enrolls in a programme.
Research / taught postgraduate	The primary key of these entities is the 'Student_ID' attribute inherited from the student superclass. These entities are types of students; therefore, I can justify assigning them this attribute as their primary key. These two entities also have the attributes 'Previous_Study' which contains information on the course (or courses) they completed at an undergraduate level. To justify including this attribute, as they are postgraduates, they will have completed an undergraduate program, and information about this programme should be stored in the database. I included the foreign key attribute 'Research_Group' that is a reference to the primary key 'Group_Title' of the Research Group entity. I made the assumption that all research postgraduates are in research groups to conduct their research. To justify this, on the university website I located a Ph.D. student who was a member of the cybersecurity research group and was completing her project within this group.
Assessment	The 'Assessment' entity represents an assessment that will be taken as part of the module. The primary key of this entity is 'Assessment_ID' which will be uniquely generated for each assessment, justifying its use. Other attributes include the 'Grade_Weight' and the 'Author'. The 'Grade_Weight' should be included in this entity as it indicates what percentage of the final grade is decided by the mark achieved in this assessment. An assumption I made is that the 'Author' will be the module leader and therefore this attribute will provide a reference to the foreign key of 'Module_Leader' contained within the 'Module' entity (which is a reference to a Staff_ID). My justification for this is, the specification states the role of the module leader is to organize assessment activities.
Coursework	The primary key of the coursework entity is the shared attribute 'Assessment_ID'. The unique attributes of this entity are 'Date_Set' and 'Date_Due'. These attributes belong in this entity as they provide details about the time span of the coursework.
Exam	The primary key of the exam entity is the shared attribute 'Assessment_ID'. The unique attributes here are: - 'In_Person', which defines whether the exam is onsite or online. - 'Exam_Date', the date the exam is taking place. - 'Location', where it is taking place. Containing these attributes within my entity is justifiable because students are required to know this information before an exam.
Question	This entity represents a question in an assessment and has a composite primary key. It has the attributes 'QuestionNumber', 'Details', and 'Assessment'. The 'Assessment' attribute is a foreign key and provides a reference to the primary key of the 'Assessment' entity; 'Assessment_Id'. It is required along with the 'Question_Number' to create a candidate key that allows each Question to be uniquely identified. The Question entity also contains the 'Details' attribute which stores information about the question.

Relationships

Element	Explanation / Justification
hasDepartment	The relationship between 'Faculty' and 'Department'. The chosen cardinality of one-to-many seemed suitable as a faculty is made up of many departments but a department only belongs to a single faculty. I made participation in this relationship mandatory (for the department) because a new faculty would be created to group together a number of departments and therefore should always have at least a single department.
hasStaff	The relationship between 'Department' and 'Staff'. One-to-many cardinality seemed suitable as the computer science department would contain several staff members. I did not make participation here optional (for having staff) as I believe for a department to function there must be a minimum number of staff members.
offersProgram	The relationship between the 'Department' and the 'Program' entities. The chosen cardinality is one-to-many as a single department offers several program options to students, but a program is only part of one department. I did not make the participation here optional because I believe for a department to complete its function it must provide at least a single program.
isGroupLead	The relationship between 'research group' and 'research staff' represents a research staff being a group leader. This is a one-to-one binary relationship with optional participation on one side, as each 'research group' will have a single 'research staff' member as their group leader, however not every research staff is a group leader. An assumption I've made here is that only a 'research staff' member can be the group leader. This makes sense as a non-research staff member should not be leading a research group.
hasProjects	The relationship between 'research group' and 'project'. Here the cardinality is one-to-many because each group can have multiple research projects as indicated by information I found on the Exeter website.
isModuleLeader	This is a relationship between 'academic staff' and 'Module'. This is a one-to-one binary relationship with optional participation on one side. Each module has an 'academic staff' as its module leader but an 'academic staff' is not necessarily a module leader therefore, participation is not mandatory for an academic staff entity.
taughtBy	This is a relationship between 'Academic staff' and 'Module'. Here the cardinality is one-to-many as a single module can be taught by one or more academic staff as stated in the specification. An assumption I've made here is that all academic staff teach at least one module and thus participation is mandatory on both sides. To justify this, all academic staff should be teaching to fulfil their role or they could be classified as research staff.
Requires	This is a recursive relationship on the module entity. The relationship represents zero to many modules potentially having zero to many prerequisites that other modules have to be passed for a module to be taken. The cardinality is therefore many-to-many. To justify this relation, some modules on the Exeter website require other modules to be taken first for an individual to take a module.
hasModule	This is a relationship between 'Programme' and 'Module'. The cardinality is many-to-many as each programme has several modules, and each module can be in several different programmes; as indicated on the Exeter website. I've chosen to make the module entity have optional participation because if a student is a 'research postgraduate' they are enrolled in a programme but do not take any modules because they are conducting research.
Tutors	This is a relationship between the entity 'Student' and 'Academic staff'. The 'Tutors' relationship is required as it represents an academic staff member being a tutor to any number of students. The cardinality is one-to-Many with optional participation on one side as a student is assigned a single academic staff as their tutor, however, academic staff may tutor zero to many students.
Enroll	This is a relationship between 'Student' and 'Programme'. The purpose is to indicate that a student enrolls in a Programme. I made the assumption that cardinality is many-to-one, under a conflicted decision. To explain my decision, I'll begin with the assumption that for certain a programme there will be many students enrolled. On one side, once a student has completed their undergraduate

	programme they may then enroll in another programme for their postgraduate course. Hence, suggesting a cardinality of many-to-many. However, the specification states they are only enrolled in a single programme at a time, and thus I choose a cardinality of many-to-one to reflect this. I also made the assumption that to be a student they must be enrolled in a programme and thus participation is mandatory on both sides. To justify this, research on the Exeter website indicated that all students were enrolled in a programme even if they were postgraduates.
hasAssessment	This Is a relationship between the 'Module' entity and the 'Assessment' entity. It represents the majority of modules being assessed through both an examination and coursework. However, a small percentage of modules are only assessed through one of the other. Hence cardinality is a one-to-many binary relationship. The relationship is a single module having one or two assessments (or one-to-many).
hasQuestions	This is a relationship between the 'Assessment' and 'Question' entities. An assessment is made up of one or more questions, hence the cardinality of one-to-many.

Relationship with Entities

Element	Explanation / Justification
takesAssessment	This Is a relationship between 'Student' and 'Assessment'. It represents a student taking an assessment. This relation has a cardinality of many-To-Many with many students taking zero to two assessments. I can justify this because a student may miss the assessment reflected in the optional participation of the assessment entity. This relationship has a relational entity containing the attribute 'Grade_Achieved' reflecting the grade achieved in the exam. I can justify designing my ERD in this way as the 'Grade_Achieved' attribute neither belongs to a student entity nor the Assessment entity and thus is required to be an attribute of the relationship.

Enhanced ER concept (Generalisation/Specialization)

Entities	Explanation/ Justification
Professional service staff, Research staff, academic staff	These elements all make use of generalization/specialization with respect to their sharing of the attributes of the 'staff' entity. I designed my ERD in this way as the diagram appeared clearer and it seemed redundant to repeat the same information. The constraints here are {Optional, OR} as looking at the Exeter website indicated that staff members were assigned into one of these categories with no overlap that I could find. I've chosen optional here as the specification states "generally fall into one of three categories" which leads me to assume that some staff members may not fit these categories and therefore would not be classified as one of these options, justifying my use of optional.
Postdoctoral research associate, research fellows, senior research fellows	These elements all make use of generalization/specialization with respect to their sharing of the attributes of the 'research staff' entity. I designed my entity relationship diagram in this way to reduce the amount of repeated data and increase overall clarity. The constraints here are {Mandatory, Or} as each research staff may only have one of these positions. I've chosen mandatory for my constraint as I believe to be a research staff, they must be in one of these categories.
Lectures, senior lectures, associate professors, professors, associate lectures	These elements all make use of generalization/specialization concerning their sharing of attributes with academic staff, which shares attributes with staff. I designed my entity relationship diagram in this way as no unique attributes were in the generalized entities and thus repeating attributes would be redundant. The constraints here are {Mandatory, Or} as each academic staff must be assigned one and only one of these roles.
Research postgraduate, taught postgraduate, undergraduate	These elements all make use of generalization/specialization concerning their sharing of the attributes with the 'student' entity. The constraints here are {Mandatory, Or} as each student may only be classified as one of these subclass types. I've chosen mandatory for my constraint as I believe to be a student, they must be one of these subclasses.
Coursework, Exam	These elements make use of generalization/specialization. I designed my ERD in a way that highlights the differences between the two entities but maintains that they are still both forms of assessment. For this reason, they both have the shared attribute 'Assessment_ID' which is contained in the 'Assessment' entity as their primary key. The constraints here are {Mandatory, Or} because the assessment must either be an exam or a piece of coursework.

Complex Relationships

The complex relationship '**ModuleOutcome'** was included in my ERD because it represents the one-to-one relationship with a student, the one-to-one relationship with a module, and the one-to-many (two) relationship with the grade achieved in (either one or two) assessments taken by a student. I can justify using a complex relationship because I needed to show that the outcome of a module neither belonged to a 'module' entity nor a 'student' entity and thus requires an entity with its own attributes belonging to the relation. The attributes 'Final_Grade' and 'Passed' belong to this relationship and store information on the final grade achieved by a student in a module and whether they passed the module.

The complex relationship '**membersOfGroup'** represents one-to-many research postgraduate students, one-to-many research staff, and zero-to-many academic staff being part of one-to-many research groups. I included this complex relationship in my ERD because it effectively shows how both students, research staff, and academic staff make up the members of a research group. I made participation for academic staff optional because a group could only consist of research staff as indicated by looking through research members on the website. However, I made research postgraduate participation mandatory because I believe that to complete their program, they must complete a research project and therefore must be part of a group. I also made research staff mandatory because I believe If they're research staff then they must be conducting research and therefore be part of one of these groups. Finally, I made the participation in a 'research group' one to two because academic staff members and research staff of existing research groups (on the Exeter website) were sometimes in two groups.

Task 2 process

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	Jenifer	Lecturer	2019	ECM3423	CS	Peter	Computer Graphics	1	2
11	Jenifer	Lectuerer	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 Profesor	2016	ECM3408	HPC	Jack	Enterprise Computing	2	1
36	Bob	Associate Profesor	2016	ECM2433	HPC	Jack	The C Family	2	3
36	Bob	Associate Profesor	2016	ECM3423	HPC	Jack	Computer Graphics	1	2
36	Bob	Associate Profesor	2016	ECM2418	HPC	Jack	Computer Languages	1	4

My process began with removing repeated elements to achieve the 1st normal form. This entailed separating all multi-valued columns; including 'Module Number', 'Module Name', 'Term', and 'Number of Times'. Resulting in each row of the table containing no multi-value attribute as shown above. Within this table, each row could be uniquely identified using a composite key consisting of the 'Staff Number' and 'Module Number'.

Relation schema

Table (Staff Number, Staff Name, Position, Year of Hire, Module Number, Research Group, Group Lead, Module Name, Term, Number of Times)
Primary key: *Staff Number, Module Number*

2nd Normal Form

PD1: Staff Number -> Staff Name, Position, Year of Hire, Research Group, Group Lead
PD2: Module Number -> Module Name, Term
FD3: Module Number, Staff Number -> Number of Times

The next step was to remove all partial dependencies to achieve 2nd normal form. I did this by making unique tables for the partial dependencies shown above.

Firstly, I created a staff table with the primary key 'Staff Number'. Within this table, I Included all attributes that were associated with a staff member, including: 'Staff Name', 'Position', 'Year of Hire', 'Research Group', and 'Research Lead'. These attributes were included because they were all partially dependent on the 'Staff Number' and could be uniquely identified using this primary key.

Secondly, I created a module table with the primary key 'Module Number'. Within this table, I included the attributes that were associated with each module, such as the 'Module Name' and 'Term'. These attributes were included because they were all partially dependent on the 'Module Number' and could be uniquely identified using this primary key.

Thirdly, I created a times taught table consisting of the 'Module Number', 'Staff Number', and 'Number of Times'. I recognized that the number of times a module was taught was not partially dependent on the module number, and staff number because by removing an attribute from A, where the dependency is A -> B, the dependency no longer holds and is instead a full functional dependency. Within this table, the attribute 'Number of Times' is stored as it can be uniquely identified using the composite primary key consisting of 'staff number' and 'Module number'.

Relation Schema

Staff (Staff Number, Staff Name, Position, Year of Hire, Research Group, Group Lead)
Primary Key: *Staff Number*

Module (Module Number, Module Name, Term)
Primary Key: *Module Number*

TimesTaught (Module Number, Staff Number, Number of Times)
Primary Key: *Module Number, Staff Number*
Foreign Key: Module Number references Module (Module Number)
Foreign Key: Staff Number references Staff (Staff Number)

Staff Table

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

Module Table

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

TimesTaught Table

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

3rd Normal Form

TD: Staff Number -> Research Group -> Group Lead

To achieve the 3rd normal form, I recognized the transitive dependency shown above. 'Staff Number' has a transitive dependency with 'Group Lead' and 'Research Group'. 'Group Lead' was not solely dependent on the primary key but also on the 'Research Group'. Therefore, I generated a new 'Research' table containing the 'Research Group' as the primary key with the 'Group Lead' as an attribute.

I then removed the attribute group lead from the Staff table but maintained the Research Group attribute because it acts as a foreign key to the new Research table.

Relation schema

Staff (Staff Number, Staff Name, Position, Year of Hire, Research Group)
Primary Key: *Staff Number*
Foreign Key: Research Group references Research (Research Group)

Module (Module Number, Module Name, Term)
Primary Key: *Module Number*

Research (Research Group, Group Lead)
Primary Key: *Research Group*

TimesTaught (Module Number, Staff Number, Number of Times)
Primary Key: *Module Number, Staff Number*
Foreign Key: Module Number references Module (Module Number)
Foreign Key: Staff Number references Staff (Staff Number)

Staff Table

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

Research Table

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

Module Table

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

TimesTaught Table

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