Linnéuniversitetet
Kalmar Växjö

Linnaeus University

1DV503 – Database Technology
Assignment 1

Student: Ryustem Shaban
Personal number: 19990104-2334
Student ID: rs223fx

Contents

Task 1

1. Maybe (Likely) – because it is many to many relationship and there may be actors with no roles

2. Maybe (Likely) – there is no specific number of how many actors have acted in how many movies.

3. Maybe (Very likely) – There is a possibility for this statement to be true while the relationship is many-to-one by one being the actors' side and many the movies' side, however no specific number was specified for lead roles of the actors.

4. False – because each movie can have a maximum of one lead actor, while the relation between them is many-to-one by actors being the one side and movies many, however the actor can perform lead role in multiple different movies

5. Maybe (Likely) – Because there may be directors who were not actors in a movie.

6. Maybe (Very likely) – Because there is a clearly stated relationship between them so at least one producer was an actor if not more than one, however there is a possibility of having empty slot within one of the sets which may make the statement to be "False".

7. False – The E/R diagram clearly states that there is a possibility for producer to be an actor

8. Maybe (Likely) – No specific numbers were given so we cannot make "True" or "False" assumptions (dozen stands for 12 actors)

9. Maybe (Likely) – There is a possibility for this statement to be either true or false, however we need more precise information to identify it as "True" or "False" statement

10. Maybe (Very likely) – There is a high possibility for this statement to be true, because movies have many-to-one relationship with directors meaning that each movie will have one director, but one director can have multiple movies, on the other side producers have many-to-many relationship with movies, but there may be that instance that one producer may produced only one movie.

11. True – At least one of the relationship instances given will respect the conditions given in this question since the relations between them are not violating the requirements of the question, since "Some" stands for

unspecified amount which can be 1 or 2, but in overall it is possible for these conditions to be true

12. Maybe (Very unlikely) – It is possible for this statement to be true since Actor have relationship with all the entities stated above, however no specific binary answer for this question can be given since there is lack of information.

13. Maybe (Unlikely) – There is a possibility for this statement to be both True and False it, because there is a possibility for director to be an actor and since actor can act in a movie it is possible, the other way around is that it can be False since maybe there will be no directors acted in a movie they directed as well, in overall there is a possibility for both, however no specification regarding this type of information was given.

Task 2

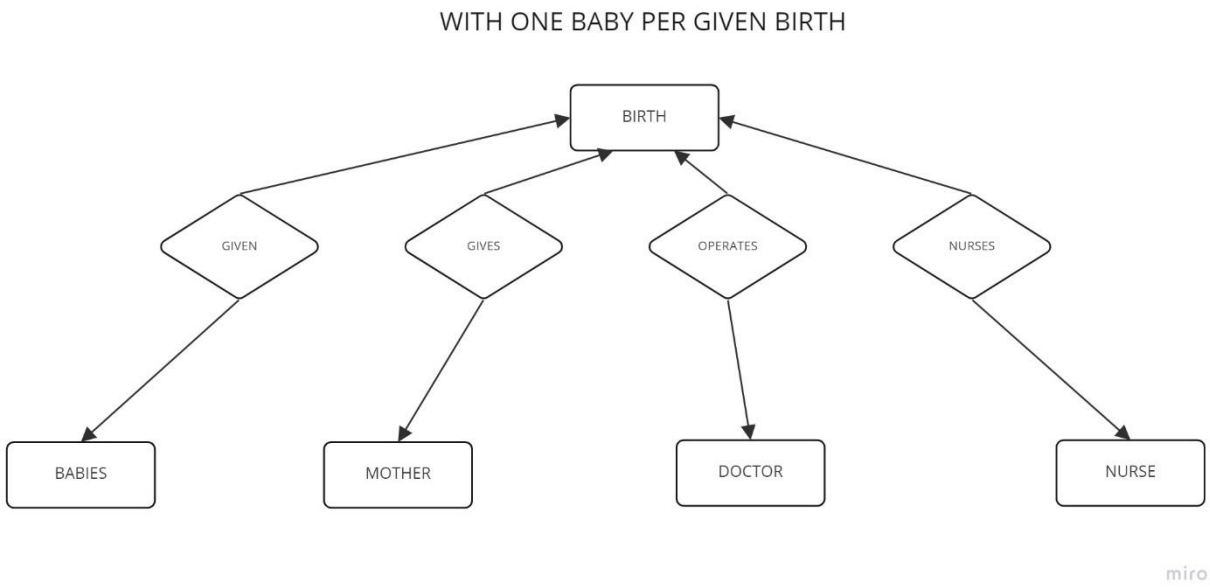

Figure 1

For the second exercise we were asked to use multiplicity to represent set of entities and their relationships in Entity Relationship diagram.

1. Starting with the BIRTH and BABIES with relationship under the title "GIVEN", it is a one-to-one relationship, because for each birth given as stated there is a unique baby, which eliminates "many" type relationship possibility.
   ➔ BABIES ARE GIVEN BIRTH, → BIRTH GIVES BABIES
2. Secondly, we have MOTHER with relationship GIVES BIRTH. Here again we have one-to-one relationship, because as stated one mother one birth at a time.
   ➔ MOTHER GIVES BIRTH, → BIRTH IS GIVEN BY MOTHER
3. As it was stated for the third relationship, we have DOCTOR under the relationship OPERATES connected to BIRTH with one-to-one relationship. In that specific case the one side is the BIRTH entity, and the other one side is the DOCTOR entity regarding the given information, because given doctor can operate only in one birth at a time.
   ➔ DOCTOR OPERATES BIRTH, → BIRTH IS OPERATED BY THE DOCTOR
4. And the last relationship is for NURSE, which is with relationship NURSES connected to BIRTH entity, this relationship is one-to-one, because one nurse can do nursing for one birth at given time. We do assume that entity set NURSE corresponds for one nurse at a given time.
   ➔ NURSE NURSES BIRTH, → BIRTH IS NURSED BY NURSE

Flaws are as follows:

1. The first flaw of this design is the limitation that for one birth we can have only one baby, while there is a possibility for more than one baby that can be born from one birth.
2. Secondly there was not asked to specify any relationship with the baby and the mother while there definitely should be.
3. Was stated that only one doctor can work on one birth, but in reality, there can be a case where more than one doctor can operate per birth.

P.S – Did made the DOCTOR and BIRTH relationship one-to-one, because DOCTOR can work on one BIRTH at a time a given time.

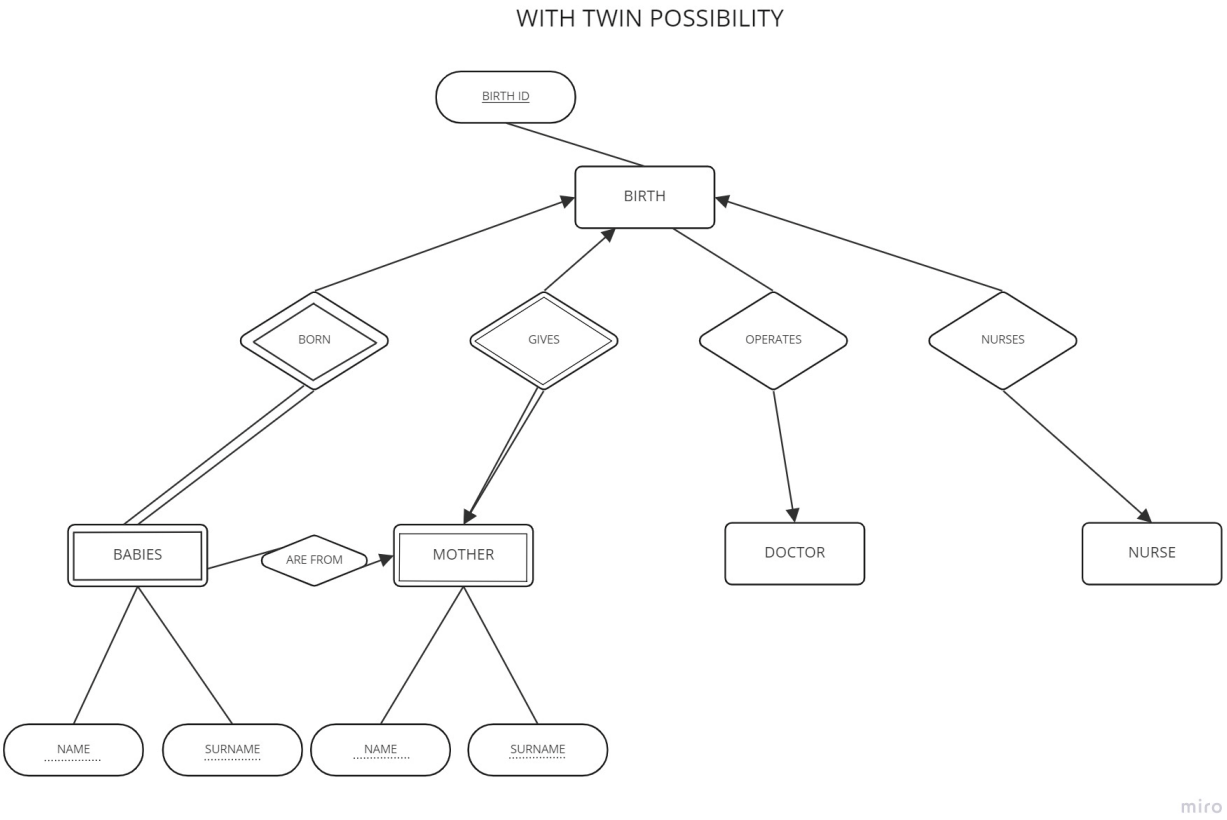

Figure 2

Change of the view from one unique baby for one birth to more than one baby for given birth is represented on *Figure 2* above. With this change of the view the possibility to have twins or three children for given birth is becoming a possibility so the relationship between BABIES and BIRTH is changed from one-to-one to many-to-one, many is the BABIES entity, and one is the BIRTH entity. Additionally considered to add derivable key attribute to the BIRTH entity as BIRTH_ID, to make it possible to identify each baby even if a birth involves more than one baby, also added derived key attributes to the babies as NAME and SURNAME, however NAME and SURNAME attributes by themselves can not uniquely identify BABIES entity, because there is a possibility even rarely to happen so that two babies can have the same name and surname, so for that case in order for us to identify BABIES entity uniquely we will be using foreign key BIRTH_ID, this will give us the possibility to uniquely identify the babies even if two babies are with same name and surname. Using this approach makes the BABIES entity a weak entity, because to identify BABIES entity uniquely we will always need the attributes of BIRTH entity. MOTHER entity will also be a weak entity, because the same problem as the BABIES entity can occur for the MOTHER entity, two mothers can have the same name, so to identify the MOTHER entity we will also use BIRTH_ID and lastly we decided to add relationship to BABIES and MOTHER ARE_FROM, which with the combination of the keys NAME and SURNAME attributes from BABIES and the foreign key attribute from BIRTH will make it possible to uniquely identify mother of each baby and vice versa.

Task 3

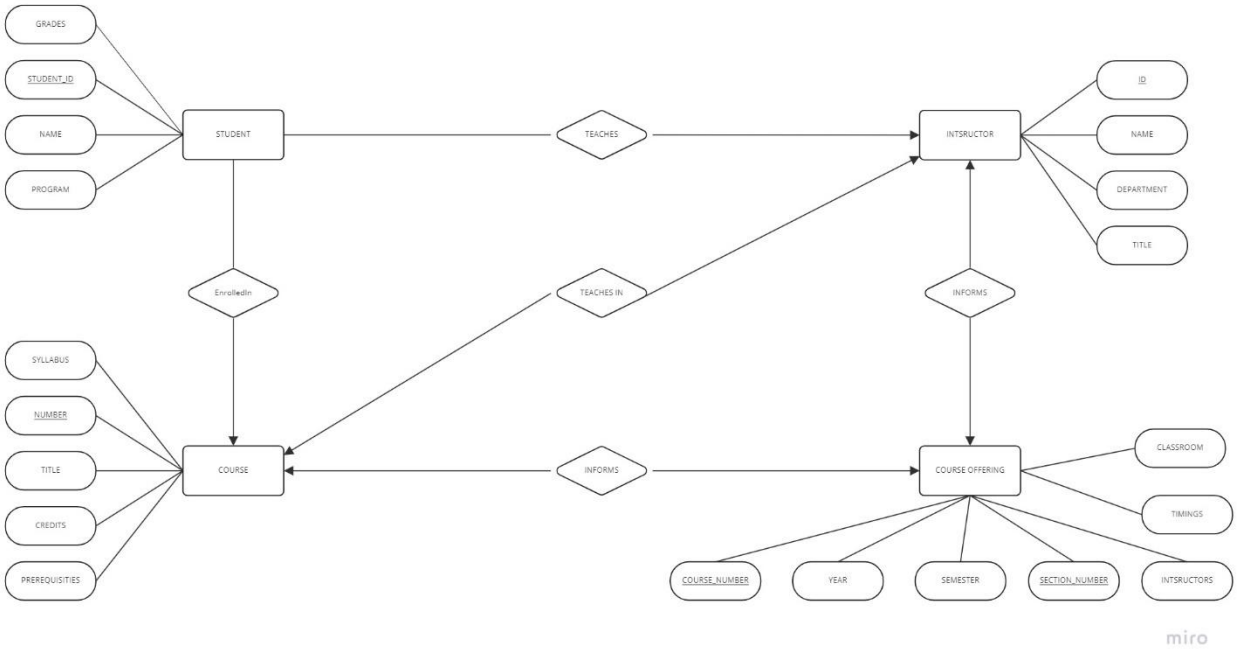

Figure 3

It was asked for us to construct an E/R diagram of given entities and their attributes. As it is visible from *Figure 3*, 5 entities were defined as.

1. → STUDENTS TAKES COURSE, → COURSE IS TAKEN BY THE STUDENTS
   STUDENT have many-to-one relationship with COURSE entity, because many students can take one course at a time.

2. → STUDENTS IS BEING TAUGHT BY INSTRUCTOR, → INSTRUCTOR TEACHES STUDENT
   STUDENT have many-to-one relationship with the INSTRUCTOR entity connected with the relationship TEACHES with INSTRUCTOR entity being the one side and STUDENT entity being the many side, it is because in our case many students can get taught by one instructor taking the course, but that same instructor can teach multiple students at the same time.

   P.S In that example was taken that the course is being led by one instructor only, however there can be cases that the course can be led by more than instructor, in that case the relationship will be many-to-many.

3. → COURSE HAVE COURSE OFFERING, COURSE OFFERING IS BEING MADE WITH COURSE
   COURSE have one-to-one relationship with COURSE OFFERING, because one course can have only one unique course offering and one course offering does belongs to only one course.

4. → INSTRUCTOR TEACHES IN COURSE, COURSE IS BEING TAUGHT BY THE INSTRUCTOR
   INSTRUCTOR entity have one-to-one relationship with COURSE with relationship TEACHES IN, because a unique instructor can teach in one course only, and we are assuming that for that diagram we are talking about one course per diagram so the instructor will be have one-to-one relationship, if that was not the case then the relationship between instructor and course will be many-to-one where COURSE will be the many side and INSTRUCTOR the one side which will state that the COURSE is taught by one instructor, however same instructor may be teaching many courses (the requirement here is that the two courses should not be at the same time), and there is one more possibility which will make the relation many-to-many this case will mean that one course can be taught by multiple instructors (as it was for Computer security course) and these instructors can be working on multiple courses (with the requirement that they don't have class at the same time for two courses again).

5. → COURSE OFFERING INFORMS INSTRUCTOR, INSTRUCTOR IS BEING INFORMED WITH THE COURSE OFFERING
   INSTRUCTOR have one-to-one relationship with COURSE OFFERING entity, because each course will have unique offering for an instructor, and since the instructor is only teaching on specific course from our assumption stated in the previous relation description instructor can have one course offering, however this relationship can be one-to-many and many-to-many as well depending to the exact same conditions stated on the previous relationship.

Task 4

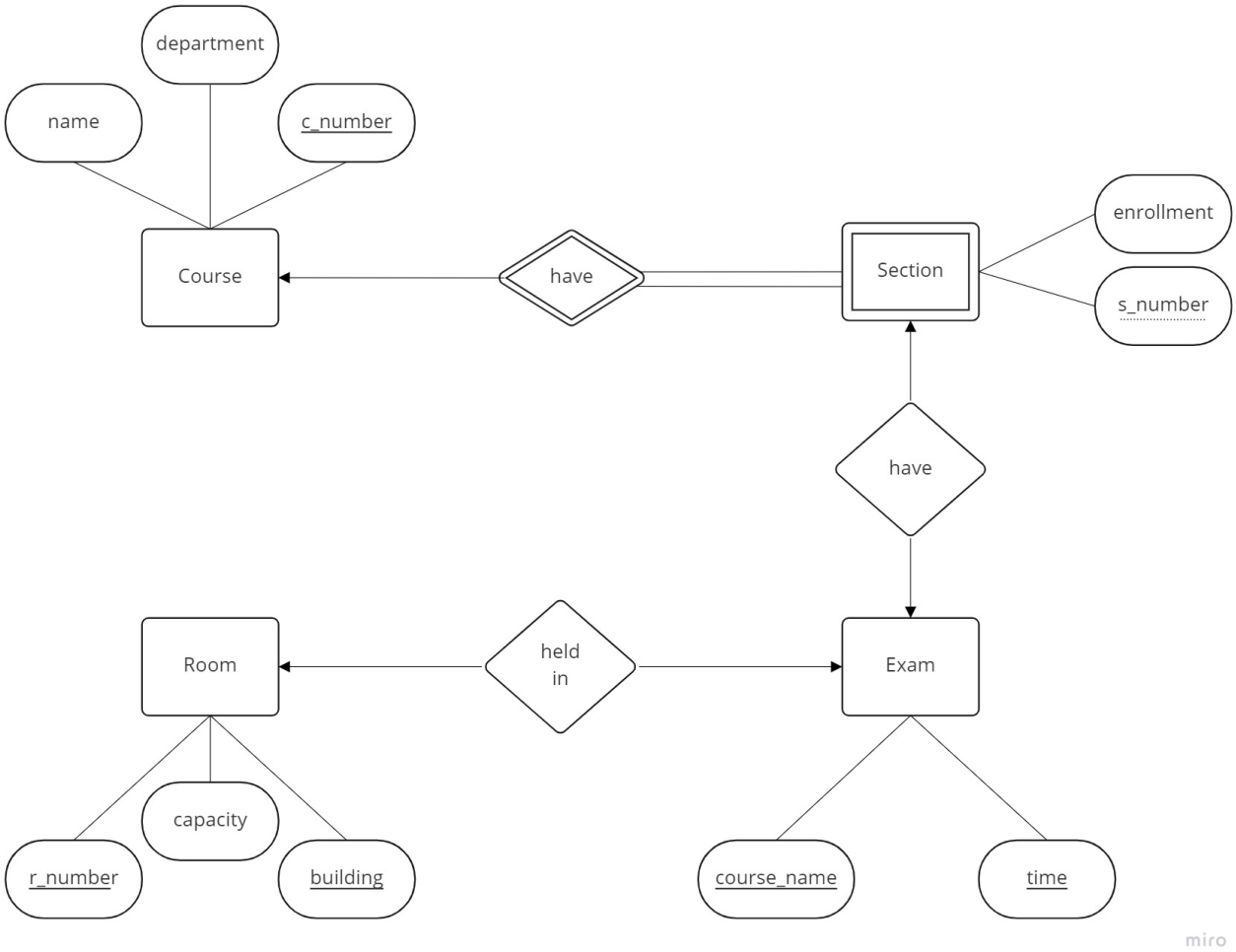

Figure 4

I would consider to include the Section entity, because even though it is a weak entity which can be defined uniquely by foreign key, there may (and should) be the case that different sections of one given course may have different exams at different times, for example if we take the Basic Mathematics course which was taken at Linnaeus university, for that course's exam there were two different variations (The English and the Swedish exam) which proves that we need that entity.

Would consider including course entity as well since it is the strong entity for the section and section can only be defined uniquely with the help of the course entity.

Would definitely consider adding room entity as well since it is providing information regarding capacity which will give the database to do more precise calculations and choose the most suitable room considering the enrolment attribute taken from the section entity's attributes. Do we need building name or not may vary from university to university since in Linnaeus we have only rooms with those unique room_numbers, the best case in that scenario may be also to have room_number attribute as key since if it happens to be including the building name and the room number we would not need the building name at all, all that information will be included to the room_number which will safe us memory and time and capacity as an ordinary non-key attribute. If this is not possible room_name and building both may be taken as keys.

Task 5

1. $R^1 := \pi_{NAME} (\sigma_{ID} (\pi_{ID} (\sigma_{code = 2DV513}(Enrolledin))) Students)$

   R^1 – Firstly selects ID where code is 2DV513 from (Enrolledin), takes that ID and selects the same ID instances from the (Students) and projects names which are containing the ID selected ID instances.

2. $R := \pi_{Name} (\sigma_{ID} (\pi_{ID} (\sigma_{(code =2DV513 \ AND \ 1DV513}(Enrolledin))) Students)$

   R – Selects ID where code instances are 2DV513 AND 1DV513, projects these ID instance and selects those IDs from (Enrolledin), then selects those same ID instances where code was equal to 2DV513 and 1DV513 and projects names of those ID instances

3. $R := \pi_{LECTURER} (\sigma_{code=2DV610} (Subject))$

   R – Projects selected lecturer from subject relation where code is equal to 2DV610

4. $R := \pi_{LECTURER} (\sigma_{(code=2DV513 \ AND \ code=1DV513} (Subject)))$

   R – Projects lecturer from subject relation where code is equal to 2DV513 and 1DV513

5. $R^1 := \sigma_{CODE} (lecturer = \text{“Ilir”}(Subject))$
   $R^2 := \sigma_{ID} (code = R^1 (Enrolledin))$
   $R := \pi_{NAME} (id != R^2(Students))$

   R^1 – Selects code from relation Subject where lecturer is equal to “Ilir”
   R^2 – Selects ID of those students from relation Enrolledin where code is equal to result of the query R^1 (in that case the subject which is taught by Ilir
   R – Projects the name of the students from relation Students whose ID is not equal to the IDs of students from the query R^2. (Will project name of all the students who are not Enrolledin the subject of Ilir).

Task 6

Relation → Interviews (manager, applicant, day, time, room).

1. Applicant and manager are functionally dependant from day, time and room to be identified uniquely. If we were taken only day, it would not help to identify the specific time same goes for the time attribute also, only with time we would not be able to identify the day uniquely.

   Day, time, room → applicant, manager
   Day, manager → room
   Day, time, manager → applicant, room

2. The candidate keys of the relationship are day, time and room since with their help it is possible to identify the rest of the attributes such as applicant and manager, pair of (day, time, manager) can also identify applicant, room also then relation (time, room -> manager) can also be given as additional functional dependency.

   $$\frac{\text{Day, time, room}^{+}}{\text{applicant, manager}}$$

3. For this relation to be on the third normal form we first need to prove that It is on the first and second normal form. Let's take for example the given parameters

Day	Time	Manager	Applicant	Room
Monday	09:00	Jonas	Gabriela	3rd

1NF –

- It is on the first normal form, because all the attributes within the relationship will be atomic.
- All column entities will be on the same kind respectfully
- No present composite attributes (attributes which can be divided to sub-parts)
- Each column have a unique name
- All rows are unique (tuples) with at least one attribute ((time will always, be unique even though there can be redundancy for the rest of the attributes, which will make the columns unique)

The relationship is on the first normal form, because it fulfils all the conditions.

2NF – The relationship is on the second normal form as well, because It does not have any non-prime attribute that is functionally dependant on proper subset of any candidate key of the relation.
In that case (day, manager) combination does define the room, however, since (day, manager) is not a proper subset of the candidate keys it does not violate the relation.

3NF – The relationship is on the third normal form as well, because the relation is on second normal form and there is no transitive dependency between non-prime attributes, (day, manager) does define the room, however since room is a prime attribute this does not break the rule.

BCNF – The relationship is not in Boyce Codd Normal Form, because there is a pair that is not a superkey defining a prime attribute where (day, manager) was the pair that was not entitled to be a superkey, however it does define a prime attribute room which is violating our BCNF conditions.

4. We will decompose the relation into BCNF by breaking the (day, time, room) and we will keep the relation Applicant→ time in a separate table. So, we will have the following:

   Day, time, room

   R^1(Day, time, manager, applicant) **R^2**(Day, manager, room)

In that case for R^2 Day, manager will be a primary key and we will refer day, manager to R^1 as a foreign key. So it will be as following.

R^1
Day, time, manager→ applicant

R^2
Day, manager → room

$$\mathbf{R^1}\left(\frac{\text{Day, time, manager}^{+}}{\text{applicant}}\right) \qquad \mathbf{R^2}\left(\frac{\text{Day, manager}^{+}}{\text{room}}\right)$$

Candidate keys for **R**[1] will be Day, time, manager and manager and non-prime attribute will be applicant and by this way the relation will be on BCNF.

Candidate key for **R**[2] will be the day, manager and non-prime attribute will be room and this relation will be respecting the BCNF conditions.

The proof for both relations is as it follows:

1NF –

- It is on the first normal form, because all the attributes within the relationship will be atomic.
- All column entities will be on the same kind respectfully
- No present composite attributes (attributes which can be divided to sub-parts)
- Each column have a unique name
- All rows are unique (tuples) with at least one attribute ((time will always, be unique even though there can be redundancy for the rest of the attributes, which will make the columns unique)

2NF –

The relations will be on the second normal form since they are on the first normal form and there are no present relations which are containing any proper upset of the candidate keys so it will not violate the second normal form.

3NF –

The relations will be on the third normal form since they are on the second normal form and there is no present transitive dependency between non-prime attributes.

BCNF –

The relations will be on the Boyce Codd normal form since relations do not violate the third normal form and left hand side of both of the dependencies are superkeys, which means that there is no present violation.

5.

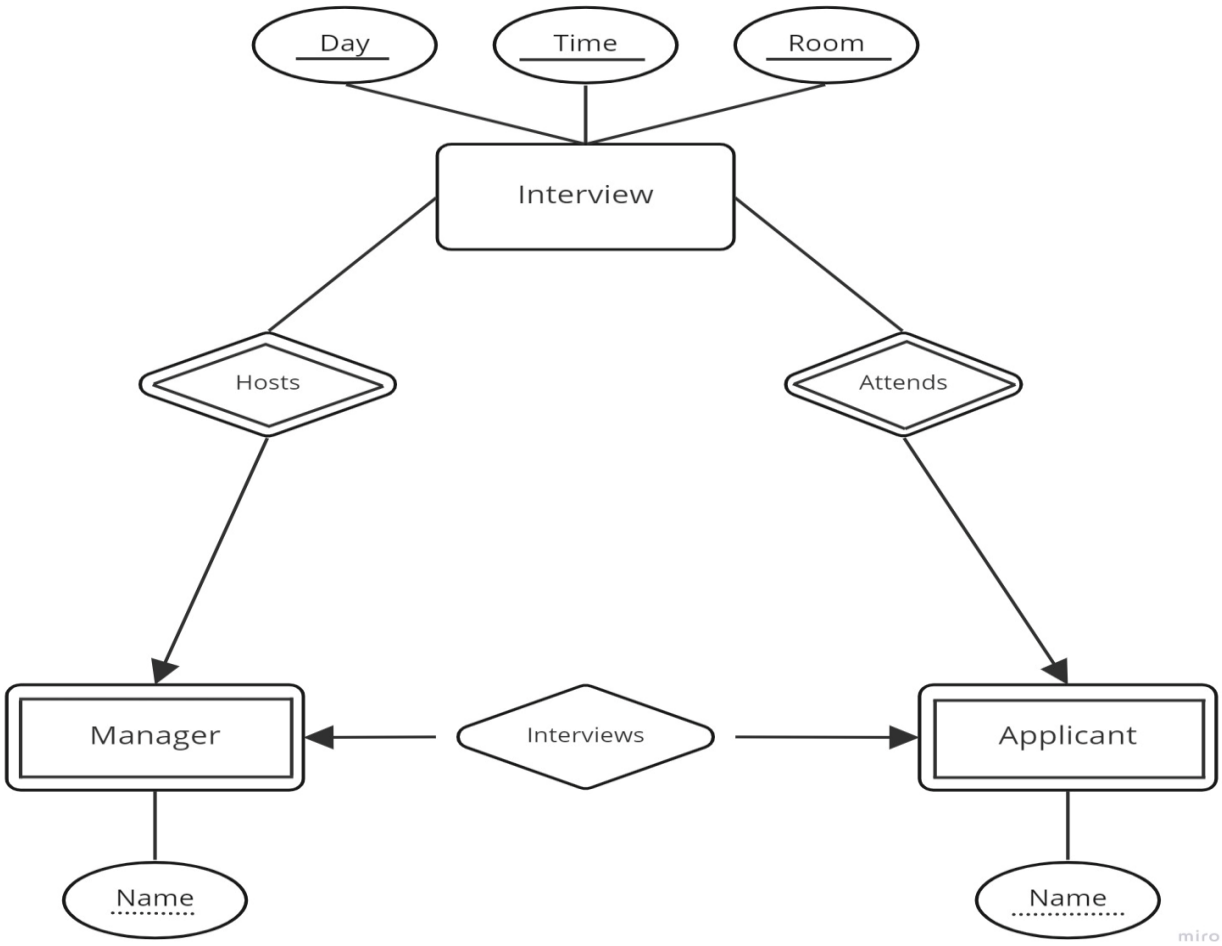

Figure 5

Figure 5 shows E/R diagram for the given relation (Day, time, room, manager, applicant) and as it is visible decided to add an entirely new entity under the name “Interview” which contains three key attributes (**Day, time, room**) instead of giving all of them as entity sets with no attributes. Additionally, entity sets **Manager** and **Applicant** contains attribute **name** which will help to identify them, however the attribute **name** itself was not enough to identify them uniquely since there may be two **Managers** or two **Applicants** with same **name**, for that reason we will need the foreign keys of the entity Interview to identify them. In that case both **Manager** and **Applicant** can be defined by the attribute **name** and combination of foreign keys from the **Interview**, which is making them weak entities. Could have added additional strong key attributes to the **Manager** and **Applicant** as “**ID_number**” and in that case they would no longer need to be weak entities and could have just removed the Entity **Interview** and instead added all the attributes to the *relationship Interviews*, however, was not sure if it is possible to add additional key attributes to the entities or not so followed that path (The path with stating attributes in relationship could be much more simpler in that case, however will be bad decision since it was asked to find possibly all dependencies).

Multiplicity of the relations:

→ Manager hosts interviews, → Interviews are being hosted by the manager

Manager and interviews have many-to-one relation with the relationship Hosts, the schema states that Manager can host multiple interviews during a day, however each interview is being hosted by one manager only.

→ Applicant attends interviews, → Interviews are attended by the applicant

Applicant and interviews does also have the same kind of relation under the relationship Attends since an applicant can attend multiple interviews (as long as they are on different days), however interview can not be attended by multiple applicants.

→ Manager interviews applicant, → Applicant is being interviewed by the manager
Manager and Applicant have one-to-one relation with relationship Interviews, the reason for that is because a manager can interview only one applicant at a given time and applicant can be interviewed by only one manager.