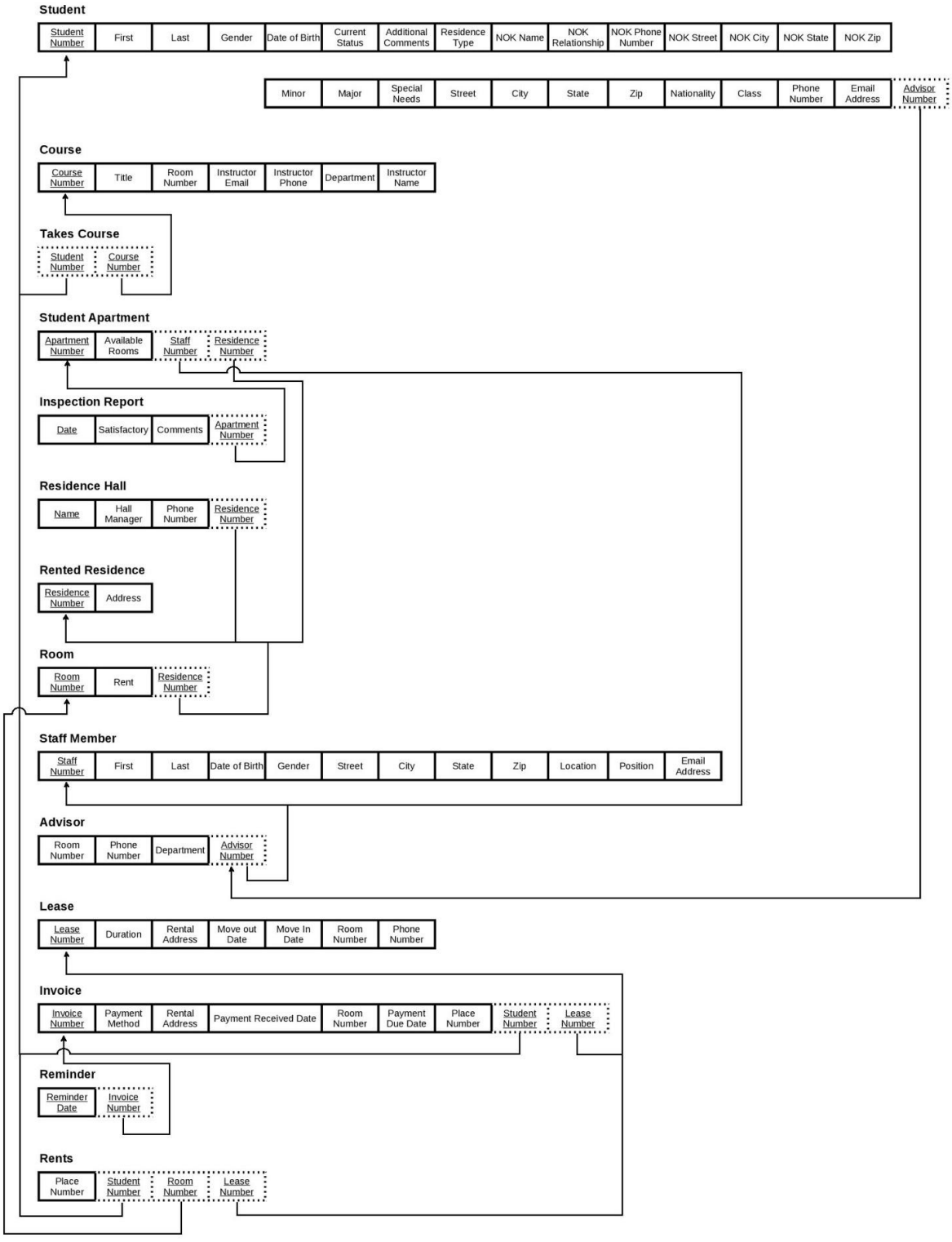
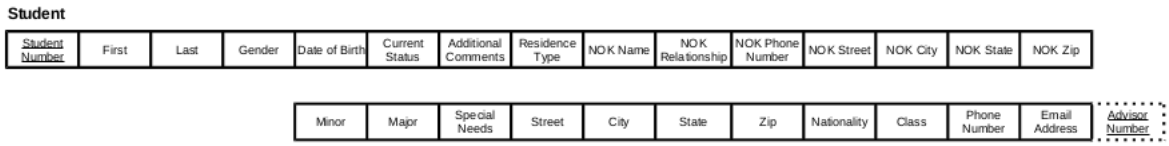
Revised Relational Model



‘Student’ Relation



Relational Diagram for ‘Student’

Primary Key: Student Number
Prime Attributes: Student Number
Non-Prime Attributes: First, Last, Gender, Date of Birth, Current Status, Additional Comments, Residence Type, NOK Name, NOK Relationship, NOK Phone, NOK Street, NOK City, NOK State, NOK Zip, Minor, Major, Special Needs, Street, City, State, Zip, Nationality, Class, Phone Number, Email, **Advisor Number**

Functional Dependencies for ‘Student’ Relation

FD1a-s: Student Number First, Last, Gender, Date of Birth, Current Status, Additional Comments, Residence Type, NOK Name, Minor, Major, Special Needs, Street, City, State, Zip, Nationality, Class, Phone Number, Email

FD1a-s Explanation: The project description states that the following data is stored on every ‘Student’: ‘First’, ‘Last’, ‘Gender’, ‘Date of Birth’, ‘Current Status’, ‘Additional Comments’, ‘Residence Type’, ‘NOK Name’, ‘Minor’, ‘Major’, ‘Special Needs’, ‘Street’, ‘City’, ‘State’, ‘Zip’, ‘Nationality’, ‘Class’, ‘Phone Number’ and ‘Email’. Derived from intuition, ‘Student Number’ will be unique amongst all ‘Student’ tuples in the relation.

FD2: Student Number Advisor Number

FD2 Explanation: From the EER Diagram it can be shown that there is a one to many relationship between ‘Student’ and ‘Advisor’. The project description also states that all students registered in the university will be assigned an ‘Advisor’.

FD3a-f: NOK Name NOK Relationship, NOK Phone Number, NOK Street, NOK City, NOK State, NOK Zip

FD3a-f Explanation: The project description states that a student’s next-of-kin data will be stored, which includes the ‘NOK Relationship’, ‘NOK Phone Number’, ‘NOK Street’, ‘NOK City’, ‘NOK State’, and ‘NOK Zip’.

‘Student’ Relation Continued

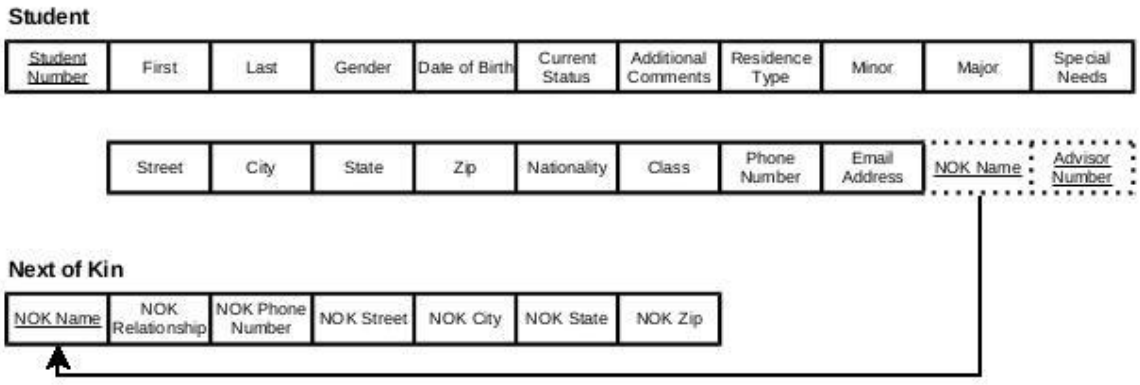
Normalization Explanation for 'Student'

1NF: The 'Student' relation is in 1NF because there are no multi valued or composite attributes in the relation. In our original EER Diagram, there was a composite attribute but it was removed when we converted it to the Relational Model so we are already in 1NF.

2NF: The 'Student' relation is in 2NF because it is confirmed to be in 1NF and there are no non-prime attributes dependent on a partial key. Because there is only one key, it is automatically in 2NF.

3NF: The 'Student' relation is not in 3NF. It satisfies the first part of being in 3NF which is that it be in 2NF, but there are non-prime attributes that functionally determine other non-prime attributes, thus it is not in 3NF. According to FD1h, Student Number NOK Name and according to FD3a-f, NOK Name NOK Relationship, NOK Phone Number, NOK Street, NOK City, NOK State, NOK Zip. Therefore a transitive dependency exists with 'NOK Name' being in the middle. 'NOK Name' is a non-prime attribute and it functionally determines six other non-prime attributes.

Normalization Procedure to put it in 3NF: The violation of 3NF occurs on FD3a-f, so we remove the non-prime attributes 'NOK Relationship', 'NOK Phone Number', 'NOK Street', 'NOK City', 'NOK State', and 'NOK Zip' from the Student relation and create a new relation called 'Next of Kin' with the aforementioned attributes. The attributes in question are removed completely from the 'Student' relation when added to the 'Next of Kin' relation. Besides these non-prime attributes, the 'NOK Name' attribute is also added to the new relation. 'NOK Name' will act as the primary key in the 'Next of Kin' relation and also will act as a foreign key in the Student relation.



Relational Diagram for 'Student' and 'Next of Kin'

'Student' Relation Continued

Now that we've created a 'Next of Kin' relation above, we need to check that it's normalized to 3NF as well:

Primary Key: NOK Name
Prime Attributes: NOK Name
Non-Prime Attributes: NOK Relationship, NOK Phone Number, NOK Street, NOK City, NOK State, NOK Zip

Functional Dependencies for 'Next of Kin' Relation

FD1: NOK Name NOK Relationship, NOK Phone Number, NOK Street, NOK City, NOK State, NOK Zip

FD1 Explanation: The project description states that a student's next-of-kin data will be stored, which includes the 'NOK Relationship', 'NOK Phone Number', 'NOK Street', 'NOK City', 'NOK State', and 'NOK Zip'.

Normalization Explanation for 'Next of Kin'

1NF: The 'Next of Kin' relation is in 1NF because there are no multi-valued or composite attributes in the relation.

2NF: The 'Next of Kin' relation is in 2NF because it is confirmed to be in 1NF and there are no non-prime attributes dependent on a partial key. Because the primary key is atomic, it is automatically in 2NF.

3NF: The 'Next of Kin' relation is in 3NF because it satisfies the first part of being in 3NF which is that it is in 2NF and there are no non-prime attributes that functionally determine other non-prime attributes so this relation is in 3NF.

'Course' Relation

Course

Course Number	Title	Room Number	Instructor Email	Instructor Phone	Department	Instructor Name

Relational Diagram for 'Course'

Primary Key: Course Number
Prime Attribute: Course Number
Non-prime Attributes: Title, Room Number, Instructor Email, Instructor Phone, Department, Instructor Name

Functional Dependencies for 'Course' Relation

FD1a-d: Course Number Title, Room Number, Department, Instructor Name

FD1a-d Explanation: From the project description, the 'Course Number', 'Course Title', 'Room Number', 'Instructor Name' and 'Department' will be stored in the database for each course taken by a student in the database.

FD2: Instructor Name Instructor Email, Instructor Phone

FD2 Explanation: Without an 'Instructor Name', there can intuitively be no corresponding 'Instructor Email' or 'Instructor Phone' because each instructor has their own phone number and email address.

Normalization Explanation for 'Course'

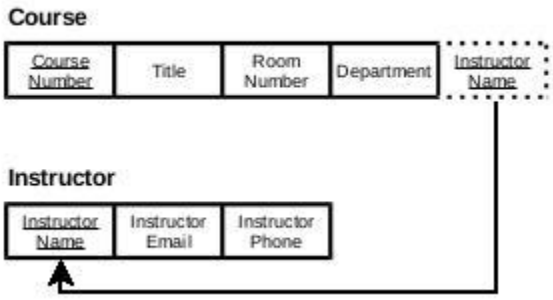
1NF: 'Course' is in 1NF because it contains only single-valued, atomic attributes.

2NF: The 'Course' relation is in 2NF due to the fact that it is in 1NF and the key is atomic, thus it is not possible for any functional dependency to be partial.

3NF: This relation is not in 3NF because although it is in 2NF, FD2 is a transitive functional dependency, that is the 'Instructor Email' and 'Instructor Phone' attributes, which are non-prime, are functionally dependent on the non-prime attribute 'Instructor Name'.

Normalization Procedure to put 'Course' in 3NF: FD2 violates 3NF. Thus we create an 'Instructor' relation whose PK is 'Instructor Name' and contains the non-prime attributes 'Instructor Email' and 'Instructor Phone' which were removed from the 'Course' relation. 'Instructor Name' was left in the 'Course' relation as a foreign key. Below is the resulting relational schema for 'Course' and 'Instructor'.

'Course' Relation Continued



Relational Diagram for 'Course' and 'Instructor'

Next we ensure that the newly created 'Instructor' relation is also in 3NF:

Primary Key: Instructor Name
Prime Attributes: Instructor Name
Non-prime Attributes: Instructor Email, Instructor Phone

Functional Dependencies for 'Instructor' Relation

FD1a-b: Instructor Name Instructor Email, Instructor Phone

FD1a-b Explanation: This functional dependency is intuitive because for each 'Instructor' (determined by their unique name), there is a corresponding phone number and email address, as previously mentioned.

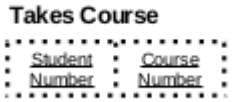
Normalization Explanation for 'Instructor' Relation

1NF: 'Instructor' is in 1NF due to the lack of multivalued and composite attributes.

2NF: This relation is in 2NF because it is in 1NF and the primary key is atomic, thus there can be no partial functional dependencies.

3NF: The 'Instructor' relation is in 3NF because it is in 2NF and there are no transitive functional dependencies to speak of.

'Takes Course' Relation



Relational Diagram for 'Takes Course'

Primary Key: composite (Student Number, Course Number)
Prime Attributes: Student Number, Course Number
Non-prime Attributes: none

Functional Dependencies for 'Takes Course' Relation

FD1a-b: Student Number, Course Number Student Number, Course Number

FD1a-b Explanation: Because the 'Takes' relationship from the EER diagram between the 'Student' and 'Course' entities is many to many, there will be only the trivial functional dependencies from the primary key to each of the prime attributes.

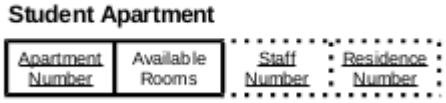
Normalization Explanation for 'Takes Course' Relation

1NF: 'Takes Course' is in 1NF because it contains no multivalued or composite attributes.

2NF: The 'Takes Course' relation is in 2NF because it is in 1NF and none of the functional dependencies are partial as there are no non-prime attributes.

3NF: This relation is in 3NF due to the fact that it is in 2NF and there are no transitive functional dependencies.

'Student Apartment' Relation



Relational Diagram for 'Student Apartment'

Primary Key: Apartment Number
Prime Attributes: Apartment Number
Non-Prime Attributes: Available Rooms, **Staff Number, Residence Number**

Functional Dependencies for 'Student Apartment' Relation

FD1: Apartment Number Available Rooms

FD1 Explanation: From the project description, an 'Apartment Number' uniquely identifies a 'Student Apartment' and the number of 'Available Rooms' is also stored per apartment.

FD2: Apartment Number Staff Number

FD2 Explanation: In the EER Diagram there is a one to many relationship between 'Student Apartment' and 'Staff Member'. The project description also states that a 'Student Apartment' is inspected by a 'Staff Member'.

FD3: Apartment Number Residence Number

FD3 Explanation: From the EER Diagram, 'Student Apartment' is part of a union with 'Residence Hall'. The union of these two is 'Rented Residence', so the name of the 'Student Apartment' determines the 'Residence Number'.

Normalization Explanation for 'Student Apartment' Relation

1NF: The 'Student Apartment' relation is in 1NF because there are no multi valued or composite attributes in the relation.

2NF: The 'Student Apartment' relation is in 2NF because it is confirmed to be in 1NF and there are no non-prime attributes dependent on a partial key. Because there is only one key the relation is automatically in 2NF.

3NF: The 'Student Apartment' relation is in 3NF because it is confirmed to be in 2NF and there are no non-prime attributes dependent on another non-prime attribute. Thus there are no transitive dependencies and the relation in 3NF.

'Inspection Report' Relation

Relational Diagram for 'Inspection Report'

Primary Key: composite (Date, Apartment Number)
Prime Attributes: Date, Apartment Number
Non-Prime Attributes: Satisfactory, Comments

Functional Dependencies for 'Inspection Report' Relation

FD1a-b: Date, Apartment Number Satisfactory, Comments

FD1a-b Explanation: In the project description, the information in an 'Inspection Report' includes an indication of whether the property's condition was 'Satisfactory' and any additional 'Comments' about the property. This 'Inspection Report' is uniquely identified by 'Date' of the inspection and the 'Apartment Number' being inspected because in the EER Diagram, the 'Inspection Report' is an attribute of the 'Inspects' relationship between 'Student Apartment' and 'Staff Member'. Because this relationship is one to many, the relationship attributes are moved over to the many side, in this case, the 'Student Apartment'. Because the 'Inspection Report' attribute was multivalued, it was made into its own relation with a composite key.

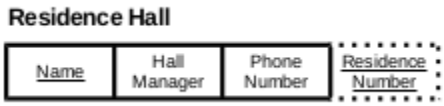
Normalization Explanation for 'Inspection Report' Relation

1NF: The 'Inspection Report' relation is in 1NF because there are no multivalued or composite attributes present in the relation.

2NF: The 'Inspection Report' relation is in 2NF because it is in 1NF and because the only two non-prime attributes, 'Satisfactory' and 'Comments' are both functionally dependent on the full key, there can be no partial dependencies, thus the relation is in 2NF.

3NF: The 'Inspection Report' relation is in 3NF because it is in 2NF and there is no non-prime attribute dependent on another non-prime attribute.

'Residence Hall' Relation



Relational Diagram for 'Residence Hall'

Primary Key: Name
Prime Attributes: Name
Non-Prime Attributes: Hall Manager, Phone Number, **Residence Number**

Functional Dependencies for 'Residence Hall' Relation

FD1a-b: Name Hall Manager, Phone Number

FD1a-b Explanation: From the business rules, the name of a 'Residence Hall' is unique among all residence halls. From the project description, each 'Residence Hall' has a 'Phone Number' and a 'Hall Manager'.

FD2: Name Residence Number

FD2 Explanation: From the EER Diagram, 'Residence Hall' is part of a union with 'Student Apartment'. The union of these two is 'Rented Residence' so the name of the 'Residence Hall' determines the 'Residence Number'.

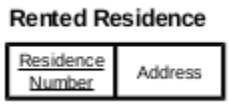
Normalization Explanation for 'Residence Hall' Relation

1NF: The 'Residence Hall' relation is in 1NF because there are no multi valued or composite attributes in the relation.

2NF: The 'Residence Hall' relation is in 2NF because it is confirmed to be in 1NF and there are no non-prime attributes dependent on a partial key. Because the primary key is atomic, the relation is automatically in 2NF.

3NF: The 'Residence Hall' relation is in 3NF because it is confirmed to be in 2NF and there are no non-prime attribute dependent on another non-prime attribute. This means there are no transitive dependencies and thus 'Residence Hall' is in 3NF.

'Rented Residence' Relation



Relational Diagram for 'Rented Residence'

Primary Key: Residence Number
Prime Attributes: Residence Number
Non-Prime Attributes: Address

Functional Dependencies for 'Rented Residence' Relation

FD1: Residence Number Address

FD1 Explanation: From the EER diagram, it can be seen that 'Rented Residence' is the result of the union between 'Student Apartment' and 'Residence Hall'. Both of these entities contain an 'Address' attribute, so the 'Residence Number' determines the 'Address'. The 'Address' will either be for a 'Student Apartment' or 'Residence Hall'.

Normalization Explanation for 'Rented Residence' Relation

1NF: 'Rented Residence' is in 1NF because it has no multivalued attributes or composite attributes.

2NF: 'Rented Residence' is in 2NF because it conforms to 1NF. In addition, "Rented Residence"'s PK is atomic. Hence, 'Address' fully depends on the whole PK; therefore, "Rented Residence" relation is in 2NF.

3NF: 'Rented Residence' is in 3NF because it is in 2NF and there are no transitive dependencies. This is because there are only two attributes and the non-prime attribute fully depends on the primary key.

'Room' Relation

Relational Diagram for Room

Primary Key: composite (Room Number, Residence Number)
Prime Attributes: Room Number, Residence Number
Non-Prime Attributes: Rent

Functional Dependencies for 'Room' Relation

FD1: Room Number, Residence Number Rent

FD1 Explanation: The project description specifies that a 'Room' has a 'Room Number' that uniquely identifies it. Because 'Room' is a weak entity, the primary key of its owner, namely 'Residence Number', helps uniquely identify the 'Room' making it a composite key. The only other piece of information that is stored about a 'Room' found in the project description is the monthly 'Rent' of the 'Room'.

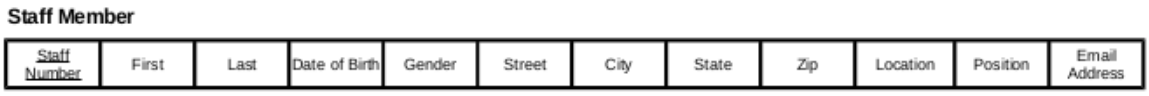
Normalization Explanation for 'Room' Relation

1NF: The 'Room' relation is in 1NF because there are no multivalued or composite attributes present in the relation.

2NF: The 'Room' relation is in 2NF because it is in 1NF and there is only one non-prime attribute, 'Rent' and this attribute is clearly dependent on the entire Primary Key, as seen in FD1.

3NF: The 'Room' relation is in 3NF because it is in 2NF and there is only one non-prime attribute, thus making any transitive dependency impossible.

'Staff Member' Relation



Relational Diagram for 'Staff Member'

Primary Key: Staff Number
Prime Attributes: Staff Number
Non-Prime Attributes: First, Last, Date of Birth, Gender, Street, City, State, Zip, Location, Position, Email Address

Functional Dependencies for 'Staff Member' Relation

FD1a-k: Staff Number First, Last, Date of Birth, Gender, Street, City, State, Zip, Location, Position, Email Address

FD1a-k Explanation: From the project description, the Staff Member's 'First' and 'Last' name, 'Date of Birth', 'Gender', 'Street', 'City', 'State', 'Zip', 'Location', 'Position' and 'Email Address' will be held in the database.

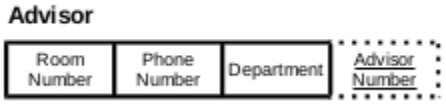
Normalization Explanation for 'Staff Member' Relation

1NF: 'Staff Member' is in 1NF because no multivalued or composite attributes exist in the relation.

2NF: The 'Staff Member' relation is in 2NF due to the fact that it is 1NF and the primary key is atomic, thus it is impossible for any non-prime attribute to be partially dependent on the primary key.

3NF: The 'Staff Member' relation is in 3NF because it is in 2NF and there are no transitive dependencies between one non-prime attribute and another, thus satisfying the definition of 3NF.

'Advisor' Relation



Relational Diagram for Advisor

Primary Key: Advisor Number
Prime Attributes: Advisor Number
Non-Prime Attributes: Room Number, Phone Number, Department

Functional Dependencies for 'Advisor' Relation

FD1a-c: Advisor Number Room Number, Phone Number, Department

FD1a-c Explanation: As specified in the project description, an 'Advisor' must have a 'Room Number', 'Phone Number', and 'Department'. The rest of the attributes specified in the project description about an 'Advisor' is inherited from it's parent relation, 'Staff Member'. An 'Advisor' must then be uniquely identified by a 'Advisor Number' because it is the primary key from it's parent relation. These attributes are reflected in the project's business rules and data dictionary.

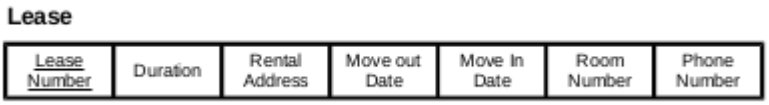
Normalization Explanation for 'Advisor' Relation

1NF: The 'Advisor' relation is in 1NF because there are no multivalued or composite attributes in the relation.

2NF: The 'Advisor' relation is in 2NF because it is in 1NF and there are no non-prime attributes dependent on a partial key.

3NF: The 'Advisor' relation is in 3NF because it is in 2NF and there are no non-prime attributes dependent on another non-prime attribute.

'Lease' Relation



Relational Diagram for Lease

Primary Key: Lease Number
Prime Attributes: Lease Number
Non-Prime Attributes: Duration, Rental Address, Move Out Date, Move In Date, Room Number, Phone Number

Functional Dependencies for 'Lease' Relation

FD1a-f: Lease Number ---> Duration, Rental Address, Move Out Date, Move In Date, Room Number, Phone Number

FD1a-f Explanation: This dependency is from the primary key constraint showing that 'Lease Number' uniquely identifies a 'Lease' and from the project description which shows that a 'Lease' contains a 'Duration', 'Rental Address', 'Move Out Date', 'Move In Date', 'Room Number' and 'Phone Number'.

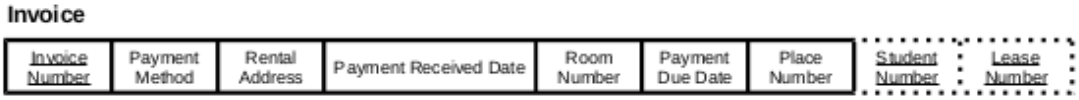
Normalization Explanation for 'Lease' Relation

1NF: The 'Lease' relation is in 1NF because there are no multi valued or composite attributes in the relation.

2NF: The 'Lease' relation is in 2NF because it is confirmed to be in 1NF and there are no non-prime attributes dependent on a partial key. Because there is only one key, the relation is automatically in 2NF.

3NF: The 'Lease' relation is in 3NF because it is confirmed to be in 2NF and there are no non-prime attributes dependent on another non-prime attribute. This means there are no transitive dependencies and thus the 'Lease' relation is in 3NF.

'Invoice' Relation



Relational Diagram for 'Invoice'

Primary Key: Invoice Number
Prime Attributes: Invoice Number
Non-Prime Attributes: Payment Method, Rental Address, Payment Received Date, Room Number, Payment Due Date, Place Number, **Student Number, Lease Number**

Functional Dependencies for 'Invoice' Relation

FD1 a-f: Invoice Number Payment Method, Rental Address, Payment Received Date, Room Number, Payment Due Date, Place Number

FD1 Explanation: From the project description, the 'Payment Method', 'Rental Address', 'Payment Received Date', 'Room Number', 'Payment Due Date', and 'Place Number' will be stored for each 'Invoice'. The 'Invoice Number' can intuitively be seen to be unique and is the primary key for the relation.

FD2: Invoice Number Student Number

FD2 Explanation: In the ER diagram there exists a one to many relationship between 'Invoice' and 'Student'.

FD3: Invoice Number Lease Number

FD3 Explanation: In the ER diagram there exists a one to many relationship between 'Invoice' and 'Lease'.

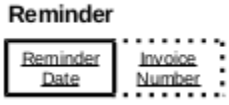
Normalization Explanation for 'Invoice' Relation

1NF: The 'Invoice' relation is in 1NF because there are no multivalued attributes or composite attributes.

2NF: The 'Invoice' relation is in 2NF because it is in 1NF and there are no non-prime attributes that partially depend on any part of the primary key. Moreover, 'Invoice''s PK is atomic. Therefore,by definition 'Invoice' is in 2NF.

3NF: The 'Invoice' relation is in 3NF because it conforms to 2NF and there are no transitive dependencies which means no non-prime attribute depends on another non-prime attribute.

'Reminder' Relation



Relational Diagram for Reminder

Primary Key: composite (Reminder Date, Invoice Number)
Prime Attributes: Reminder Date, Invoice Number
Non-Prime Attributes: None

Functional Dependencies for 'Reminder' Relation

FD1a-b: Reminder Date, Invoice Number Reminder Date, Invoice Number

FD1a-b Explanation: This trivial dependency is reflexive because 'Reminder Date' is a multi-valued attribute from 'Invoice', it is brought over into its own relation. Because there are no other attributes for this new relation besides the key, these are the only functional dependencies that can exist.

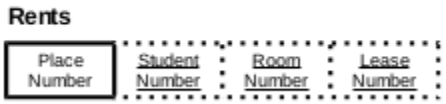
Normalization Explanation for 'Reminder' Relation

1NF: 'Reminder' is in 1NF because it contains no multivalued attributes or composite attributes.

2NF: 'Reminder' is in 2NF because it is already in 1NF and there are no partial dependencies. In addition, there are no non-prime attributes that depend on any part of the primary key because there are no non-prime attributes.

3NF: 'Reminder' is in 3NF because it is already in 2NF and no non-prime attributes depend on any other non-prime attribute.

'Rents' Relation



Relational Diagram for Rents

Primary keys: composite (Student Number, Room Number, Lease Number)
Prime Attributes: Student Number, Room Number, Lease Number
Non-prime Attributes: Place Number

Functional Dependencies for 'Rents' Relation

FD1a-c: Student Number, Room Number, Lease Number Student Number, Room Number, Lease Number

FDa-c Explanation: These are the trivial functional dependencies involving the foreign keys from the three entities involved in the ternary relationship Rents.

FD2: Student Number, Room Number, Lease Number Place Number

FD2 Explanation: This functional dependency comes from the fact that 'Place Number' is an attribute of the ternary relationship between 'Student', 'Room' and 'Lease', thus intuitively the foreign keys from each of those relations would determine the 'Place Number'.

Normalization Explanation for 'Rents' Relation

1NF: This relation is in 1NF because none of the three attributes are multi-valued, nor are they composite.

2NF: The 'Rents' relation is in 2NF due to the fact that it is in 1NF and because there are no non-prime attributes, thus it would be impossible for there to be any partial functional dependency.

3NF: 'Rents' is in 3NF because it is in 2NF and there are no transitive functional dependencies present.

Comments on Project - Normalization Phase

A. Difficulties you faced in doing this implementation phase and how they were resolved?

Probably the biggest difficulty we faced during this phase of the project was making sure we had found all the functional dependencies for a relation and then making sure we could explain them. Some of the dependencies that existed because of items in the EER Diagram were difficult to explain. Using the sheet provided to us in class helped make things easier.

B. Likes and dislikes about this part of the project?

Carbajal, Enrique: What I liked about this phase of the project was learning to normalize. When we had lecture and the normalization concepts were given to us, I didn't fully understand them. I didn't really even understand them when they were explained by James the first, or fifth time. It was only when I did my part of the project that I came to understand what normalization was and what it is used for. In addition, I also liked this phase of the project because it went much smoother than the other parts. Since we did a good job in the first preceding parts, this part of the project was much easier.

Small, James: I liked using the normalization process on our design and getting it all into 3NF. It makes things very clear when looking at it now. I disliked having to create full page summaries of the relations already in 3NF. I think a small description on why they are in 3NF would have worked instead of a full explanation.

Takahashi, Mark: I enjoyed this phase of normalizing our relations because it was straightforward and easy to understand. I disliked having to explain why each relation was in each normal form. I understand though that although it was tedious it ensured that we correctly made them in 3NF and helped eliminate potential mistakes.

Urbach, Daniel: I liked applying the normalization process that we went over extensively in class on our project, even though most of our relations were already in 3NF. I disliked the fact that the three examples given were syntactically different, which made it hard to know exactly what was expected in the document.

Comments on Project - Normalization Phase Continued

C. What was the most challenging aspect of this part of the design?

Carbajal, Enrique: The most challenging part of the design was having to find precisely where the functional dependencies came from. I could simply make up the functional dependencies. This wouldn't be following the methodologies taught to us in class. So, I had to investigate and come up with concrete evidence for why I we had the functional dependencies we had.

Small, James: I thought the most challenging aspect involved determining the prime functional dependencies from the EER Diagram. We really had to make sure we understood the relationships before attempting to create and describe the functional

dependency from it.

Takahashi, Mark: What i found to be challenging is if there are more than one key how to explain this in the normalization explanation and the functional dependencies explanation. I was unsure if i could combine them in a single instance of a dependency or to create them on their own.

Urbach, Daniel: The most challenging aspect in this part of the design was understanding the normalization process enough to know which relations were in the three normal forms and more if they were not, to understand the underlying reasons why they were not.

D. Suggestions on how to improve this part of project?

Carbajal, Enrique: I don't have any suggestions for this part of the project. I say this because everything is straight forward and was easy to understand.

Small, James: A way to improve would be to make sure that three three examples given in the project 2b specs were written the same way. Each had their own way of showing the normalization process and it made it difficult because there wasn't any uniformity between them.

Takahashi, Mark: I believe if i would have asked more questions in class i would have had improved explanations for the functional dependencies and normalization explanations.

Comments on Project - Normalization Phase Continued

Urbach, Daniel: I would change the need to include full documentation on relations which are already in 3NF. Instead simply list the relations which are in 3NF and a brief description of why, rather than requiring a full page for each.

E. Did you make changes to your communication methods? If yes, why?

Carbajal, Enrique: As a group, we didn't make any changes to the way we communicated. We used the same methods as the previous parts of the project.

Small, James: No changes to communication. Just used google drive and chat when outside of school and face to face while at school.

Takahashi, Mark: No we did not make any changes to our communication methods. We continued to meet up outside of class and use google drive to collaborate our work.

Urbach, Daniel: We did not make any significant changes in how we communicate, because so far our methods have been working effectively.