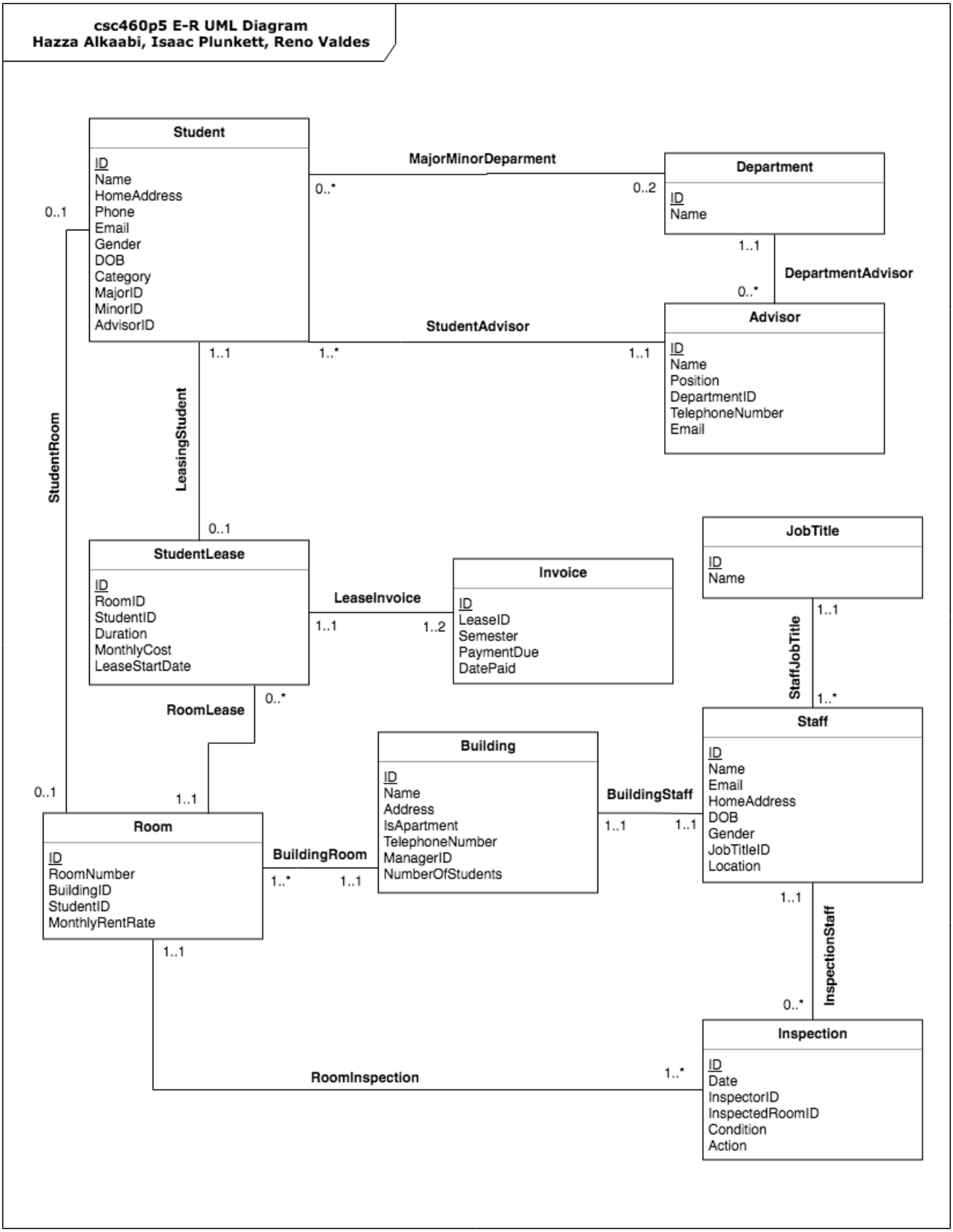
April 30th, 2018
Program 4

Hazza Alkaabi, Isaac Plunkett, Reno Valdes

Conceptual Database Design (ER UML Diagram on next page)

Notes:

- Student relation
  - Gender constraint: (“M”, “F”, “O”).
  - Category constraint: (“freshman”, “sophomore”, “junior”, “senior”, “graduate”).
  - MajorID is a foreign key to the ID in the Department relation.
  - MinorID is a foreign key to the ID in the Department relation.
  - AdvisorID is a foreign key to the ID in the Advisor relation.
- Advisor relation
  - DepartmentID is a foreign key to the ID in the Department relation.
- Room relation
  - BuildingID is a foreign key to the ID in the Building relation.
  - StudentID is a foreign key to the ID in the Student relation.
- Building relation
  - ManagerID is a foreign key to the ID in the Staff relation.
- StudentLease relation
  - RoomID is a foreign key to the ID in the Room relation.
  - StudentID is a foreign key to the ID in the Student relation.
- Invoice relation
  - LeaseID is a foreign key to the ID in the StudentLease relation.
- Inspection relation
  - InspectorID is a foreign key to the ID in the Staff relation.
  - InspectedRoomID is a foreign key to the ID in the Room relation.
- Staff relation
  - JobTitleID is a foreign key to the ID in the JobTitle relation.


csc460p5 E-R UML Diagram
Hazza Alkaabi, Isaac Plunkett, Reno Valdes
Student
ID
Name
HomeAddress
Phone
Email
Gender
DOB
Category
MajorID
MinorID
AdvisorID
MajorMinorDeparment
0..*
0..2
Department
ID
Name
1..1
DepartmentAdvisor
0..*
Advisor
ID
Name
Position
DepartmentID
TelephoneNumber
Email
StudentAdvisor
1..1
1..*
1..1
0..1
StudentRoom
LeasingStudent
0..1
StudentLease
ID
RoomID
StudentID
Duration
MonthlyCost
LeaseStartDate
LeaseInvoice
1..1
1..2
Invoice
ID
LeaseID
Semester
PaymentDue
DatePaid
JobTitle
ID
Name
1..1
StaffJobTitle
1..*
RoomLease
0..*
Staff
ID
Name
Email
HomeAddress
DOB
Gender
JobTitleID
Location
Building
ID
Name
Address
IsApartment
TelephoneNumber
ManagerID
NumberOfStudents
BuildingStaff
1..1
1..1
0..1
1..1
Room
ID
RoomNumber
BuildingID
StudentID
MonthlyRentRate
BuildingRoom
1..*
1..1
1..1
InspectionStaff
0..*
Inspection
ID
Date
InspectorID
InspectedRoomID
Condition
Action
1..1
RoomInspection
1..*

Logical Database Design

```
set autocommit off;

CREATE TABLE Department
(
   ID integer NOT NULL Primary Key,
   Name varchar2(256) NOT NULL
);

CREATE TABLE Advisor
(
   ID integer NOT NULL Primary Key,
   Name varchar2(256) NOT NULL,
   Position varchar2(256) NOT NULL,
   DepartmentID integer NOT NULL REFERENCES Department (ID),
   TelephoneNumber integer NOT NULL,
   Email varchar2(256) NOT NULL
);

CREATE TABLE Student
(
   ID integer NOT NULL Primary Key,
   Name varchar2(256) NOT NULL,
   Address varchar2(256) NOT NULL,
   PhoneNumber integer NOT NULL,
   Email varchar2(256) NOT NULL,
   Gender char(1) NOT NULL CHECK (Gender IN ('M', 'F', 'O')),
   DOB date NOT NULL,
   Category varchar2(256) NOT NULL CHECK (Category IN ('freshman', 'sophmore', 'junior',
'senior', 'graduate')),
   MajorID integer NULL REFERENCES Department (ID),
   MinorID integer NULL REFERENCES Department (ID),
   AdvisorID integer NOT NULL REFERENCES Advisor (ID)
);

CREATE TABLE JobTitle
(
   ID integer NOT NULL Primary Key,
   Name varchar2(256) NOT NULL
);

CREATE TABLE Staff
(
```

```
    ID integer NOT NULL Primary Key,
    Name varchar2(256) NOT NULL,
    Email varchar2(256) NOT NULL,
    HomeAddress varchar2(256) NOT NULL,
    DOB DATE NOT NULL,
    Gender varchar2(256) NOT NULL,
    JobTitleID integer NOT NULL REFERENCES JobTitle (ID),
    Location varchar2(256) NOT NULL
);

CREATE TABLE Building
(
    ID integer NOT NULL Primary Key,
    Name varchar2(256) NULL,
    Address varchar2(256) NOT NULL,
    IsApartment integer NOT NULL,
    TelephoneNumber integer NULL,
    ManagerID integer NOT NULL REFERENCES Staff (ID),
    NumberOfStudents integer NULL
);

CREATE TABLE Room
(
    ID integer NOT NULL Primary Key,
    RoomNumber integer NOT NULL,
    BuildingID integer NOT NULL REFERENCES Building (ID),
    StudentID integer NOT NULL REFERENCES Student (ID),
    MonthlyRentRate integer NOT NULL
);

CREATE TABLE StudentLease
(
    ID integer NOT NULL Primary Key,
    RoomID integer NOT NULL REFERENCES Room (ID),
    StudentID integer NOT NULL REFERENCES Student (ID),
    Duration integer NOT NULL,
    MonthlyCost integer NOT NULL,
    LeaseStartDate Date NOT NULL
);

CREATE TABLE Invoice
(
    ID integer NOT NULL Primary Key,
    LeaseID integer NOT NULL REFERENCES StudentLease (ID),
    Semester varchar2(256) NOT NULL,
    PaymentDue DATE NOT NULL,
```

```
    DatePaid DATE
);

CREATE TABLE Inspection
(
    ID integer NOT NULL Primary Key,
    InspectionDate DATE NOT NULL,
    InspectorID integer NOT NULL REFERENCES Staff (ID),
    InspectedRoomID integer NOT NULL REFERENCES Room (ID),
    Condition varchar2(256) NOT NULL,
    Action varchar2(256) NOT NULL
);

commit;
set autocommit on;
```

Normalization analysis

Every relation is in 4th Normal Form because every primary key in a relation determines everything else in that relation. If a relation is in 4NF, then it is also 3NF.

Additional queries descriptions

- Provide Lower Rents
    - The user can enter some Student ID to get all buildings/rooms that have lower monthly rent rate than the current monthly rent that the student pays.
    - This query is significant because it could provide the staff with knowledge they could provide inquiring students regarding alternatives to their rooms.
- Student categories Monthly Rent Average
    - Gets the average of the monthly rents for each category of students.
    - This query is useful because it provides good data