Patient and Services Management for a Large Hospital

Yves Nieto ykn2@njit.edu

Daniel Helmer dlh4@njit.edu

CS331-004

GROUP 1

2- This report should include the materials from Phases I & II (with modifications made over the course of the project):

Phase I a) summary of the system requirements

The Hospital is organized into DEPARTMENTs:

- Each department has a unique name,unique number, department specializtion and a doctor who *manages* the department
- A department may have several services provided
- Departments are subset into INPATIENT and OUTPATIENT
- INPATIENT should show the arrival and departure date
- OUTPATIENT only has departure date

Each Department *handles* a number of PATIENTs

- Each patient has a ssn, name, address, sex, and birthdate
- A statement gets *raised on* a patient for services provided
- Each patient gets treated by one or many doctors

The database will track the Hospitals DOCTORs

- Each doctor has a ssn, name,phone#, and specialty
- Each doctor *works for* one department, but can *treat* many patients
- Each doctor *provides* services to patients
- Doctors can *prescribe* many or no medicine to patients
- Each *prescription* has a quantity and date associated with it

The database will track the patients VISIT_HISTORY

- Each visit has the date and purpose of visit.

The database will track the hospitals PHARMACY

- Each pharmacy has a unique number, and phone#
- The pharmacy *provides* medicine to patient

The database will track MEDICINE

- Medicine has a cost and unique name.

The database will track STATEMENT

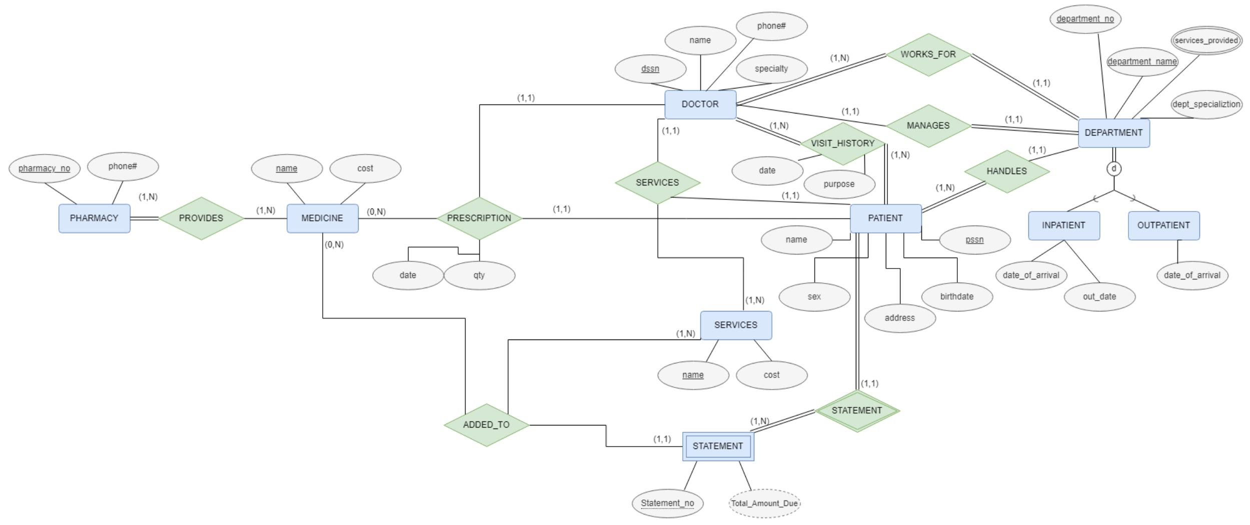
- Statement has a unique statement#, and the total amount due

The database will track SERVICES

- Services have a unique name, and cost

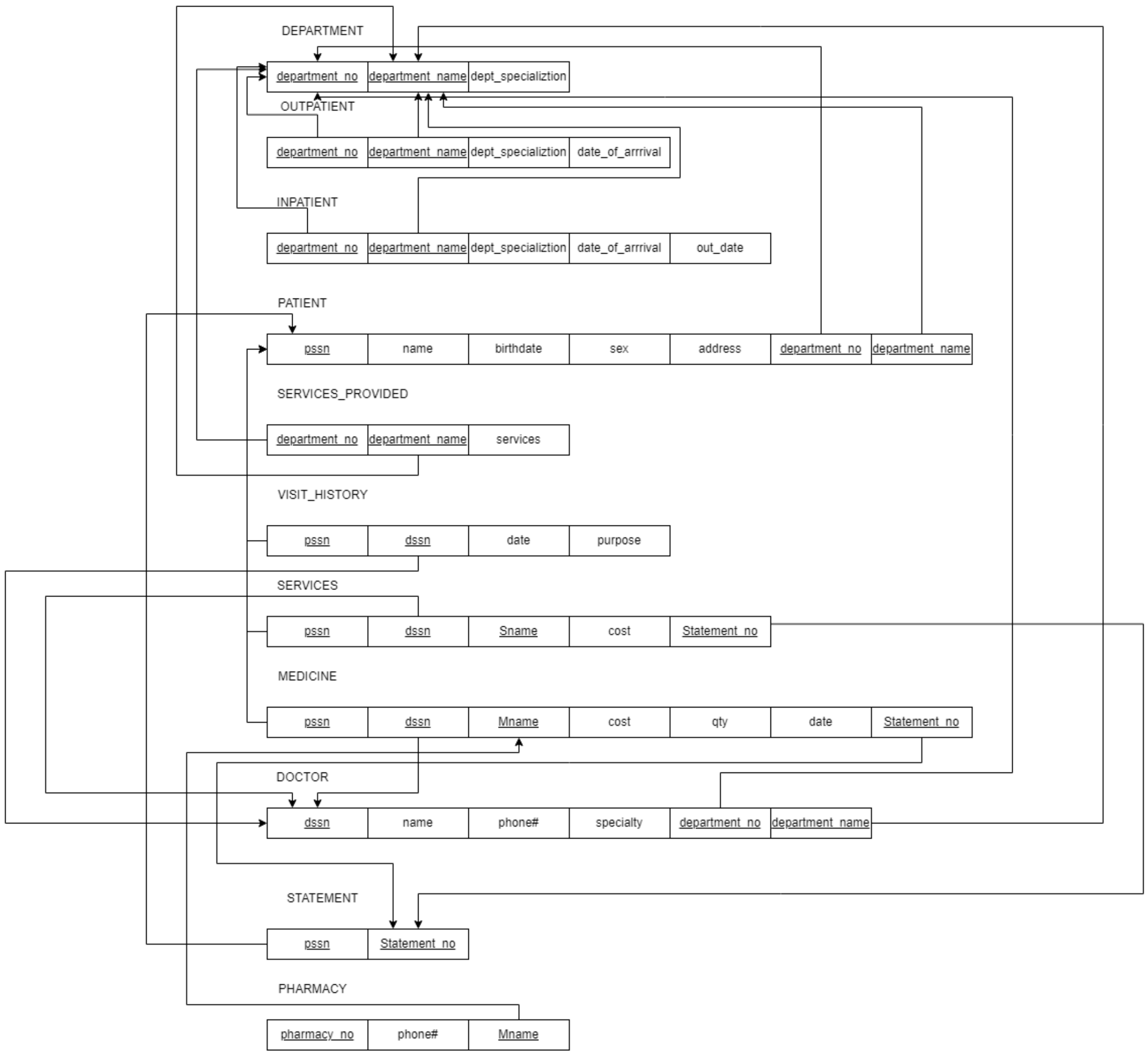
At the beginning of the project we struggled in trying to figure out how we can determine whether a patient is an inpatient or out patient. We decided to incorporate a superclass department and subclasses outpatient and inpatient.

Phase I b) the entity-relationship design



Here at the ER diagram we incorporated some previous examples into the model. The prescription relation is a ternary between doctor patient medicine. We thought that the services relationship could also be a ternary but this caused confusion and we decided to implement something else in phase 2.

Phase I c) the (relational) logical database design.


DEPARTMENT
department_no | department_name | dept_specializtion
OUTPATIENT
department_no | department_name | dept_specializtion | date_of_arrrival
INPATIENT
department_no | department_name | dept_specializtion | date_of_arrrival | out_date
PATIENT
pssn | name | birthdate | sex | address | department_no | department_name
SERVICES_PROVIDED
department_no | department_name | services
VISIT_HISTORY
pssn | dssn | date | purpose
SERVICES
pssn | dssn | Sname | cost | Statement_no
MEDICINE
pssn | dssn | Mname | cost | qty | date | Statement_no
DOCTOR
dssn | name | phone# | specialty | department_no | department_name
STATEMENT
pssn | Statement_no
PHARMACY
pharmacy_no | phone# | Mname

Here or the relational model there are more than one primary key which we later fixed. The foreign keys were not properly represented. We decided on a table for services provided as it would be the composite attribute to determine the services in each department (such as locations in the company relational model).

Phase II a) summary of the system requirements

The Hospital is organized into DEPARTMENTs:

- Each department has a unique name,unique number, department specializtion and a doctor who *manages* the department
- We keep track of the start date of the department manager
- A department may have several services provided

Each Department *handles* a number of PATIENTs

- Each patient has a ssn, name, address, sex,in/outpatient, and birthdate
- A statement gets *raised on* a patient for services provided
- Each patient gets treated by one or many doctors

The database will track the Hospitals DOCTORs

- Each doctor has a ssn, name,phone#, and specialty
- Each doctor *works for* one department, but can *treat* many patients
- Each doctor *provides* services to patients
- Doctors can *prescribe* many or no medicine to patients
- Each *prescription* has a quantity and date associated with it

The database will track the hospitals PHARMACY

- Each pharmacy has a unique number, and phone#
- The pharmacy *provides* medicine to patient

The database will track MEDICINE

- Medicine has a cost and unique name.

The database will track STATEMENT

- Statement has a unique statement#, and the total amount due

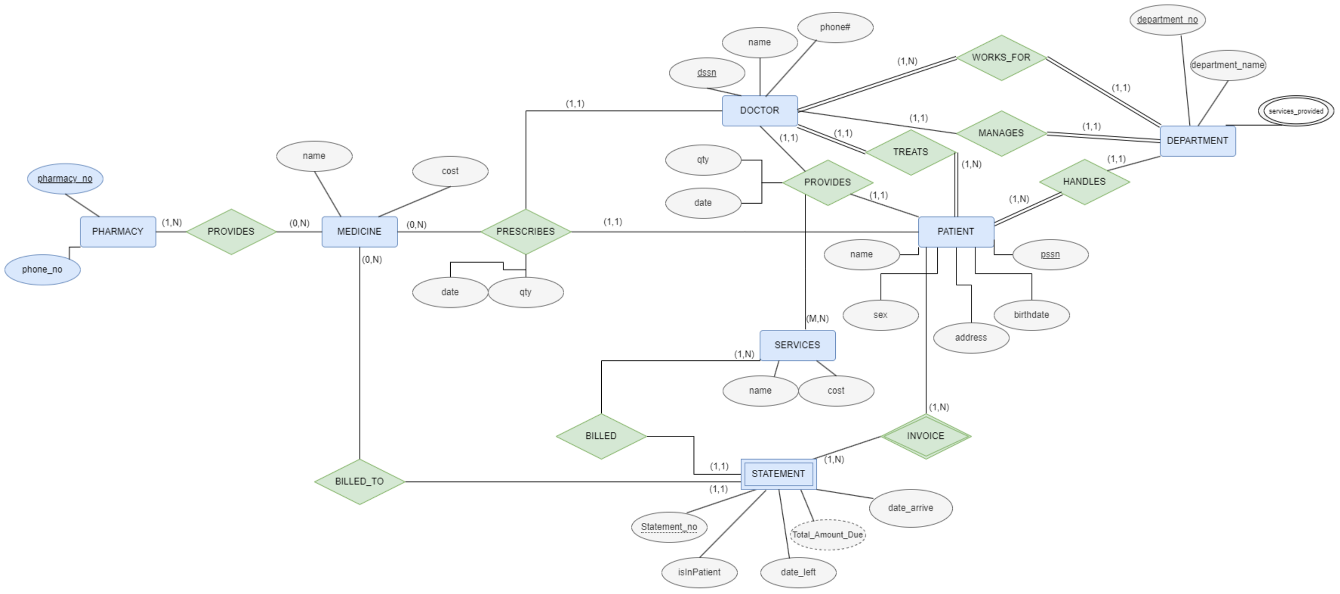
The database will track SERVICES

- Services have a unique name, and cost

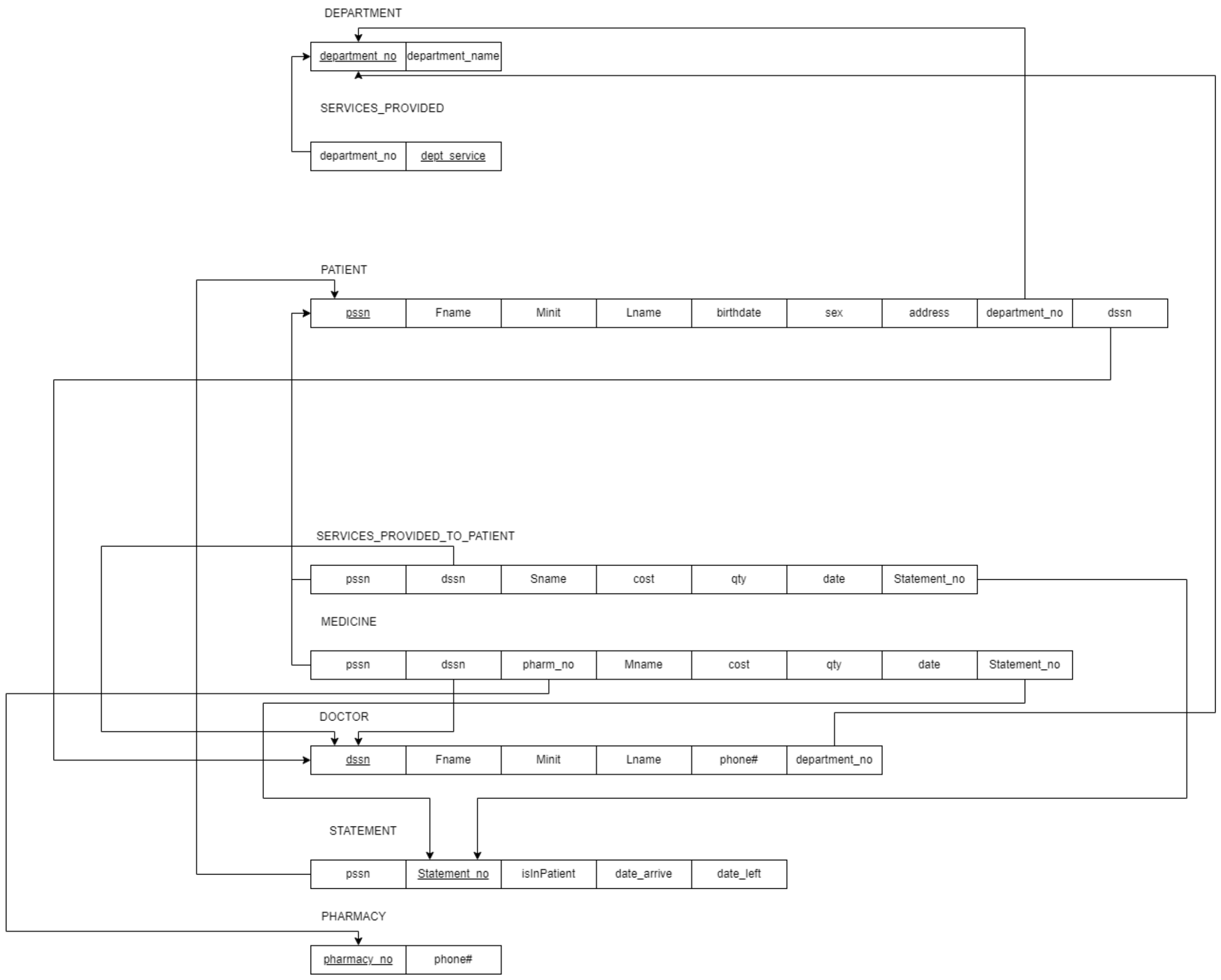
In phase II we decided to remove the subclasses inpatient and outpatient in return using datein and dateout attributes to determine whether the patient was an in/outpatient. We removed visit_history as this would be determined by queries rather than being included in the database itself. But the overal functionality and purpose remains the same.

Phase II b) the entity-relationship design

In the ER diagram we changed names of the statement relation to invoice to remove confusion. No more subclass in/outpatient in department. In the statement we introduced a datein



Phase II c) the (relational) logical database design.


DEPARTMENT
department_no | department_name
SERVICES_PROVIDED
department_no | dept_service
PATIENT
pssn | Fname | Minit | Lname | birthdate | sex | address | department_no | dssn
SERVICES_PROVIDED_TO_PATIENT
pssn | dssn | Sname | cost | qty | date | Statement_no
MEDICINE
pssn | dssn | pharm_no | Mname | cost | qty | date | Statement_no
DOCTOR
dssn | Fname | Minit | Lname | phone# | department_no
STATEMENT
pssn | Statement_no | isInPatient | date_arrive | date_left
PHARMACY
pharmacy_no | phone#

The Realtion schema has removed a lot of tables. We have many attributes in Medicine and Services_provided_to_patient in order to link the doctor, patient, and statement_no. Our foreign keys are established correctly and only one primary key per table. The department table now only holds Deptartment_no and department_name, the subclasses in/outpatient are gone. Now in the statement table we have the attribute is inpatient for a true/false to determine whether they are inpatient or outpatient. The dates on date_arrive and date_left would be the same if they are an outpatient.

Phase III

3- Normalize the Relations: The steps to follow for each relation are:

a) Write out the relation (schema) including all attribute names. Indicate keys and foreign

DEPARTMENT(department_no, department_name)

Keys

{department_no}

{department_name}

b) Provide some sample data for the relation (5 rows).

INSERT INTO DEPARTMENT VALUES(5,'Oncology');

INSERT INTO DEPARTMENT VALUES(3,'Dermatology');

INSERT INTO DEPARTMENT VALUES(2,'Family Medicine');

INSERT INTO DEPARTMENT VALUES(4,'Surgery');

INSERT INTO DEPARTMENT VALUES(1,'Emergency');

c) State the Key for the relation and write down Functional Dependencies.

{department_no}-> department_name Full Functional

d) State that this relation is in 3NF.

This relation is 3NF

a) Write out the relation (schema) including all attribute names. Indicate keys and foreign

SERVICES_PROVIDED(*dept_no,* dept_service)

Keys

{dept_service}

FK

Dept_no

b) Provide some sample data for the relation (5 rows).

INSERT INTO SERVICES_PROVIDED VALUES(5,'Proton Therapy');

INSERT INTO SERVICES_PROVIDED VALUES(5,'Electron Therapy');

INSERT INTO SERVICES_PROVIDED VALUES(4,'Facial Reconstruction');

INSERT INTO SERVICES_PROVIDED VALUES(3,'Contact Allergy Testing');

INSERT INTO SERVICES_PROVIDED VALUES(2,'Women''s Health Checkup');

INSERT INTO SERVICES_PROVIDED VALUES(1,'Trauma care');

c) State the Key for the relation and write down Functional Dependencies.

{dept_service}->dept_no Full Functional

d) State that this relation is in 3NF.

This Realtion is 3NF

a) Write out the relation (schema) including all attribute names. Indicate keys and foreign

DOCTOR(<u>dssn,</u> Fname, Minit, Lname, phone_num, *dept_num*)

Keys

{dssn},{Fname},{Minit},{Lname},{phone_num}

FK

{dept_num}

b) Provide some sample data for the relation (5 rows).

INSERT INTO DOCTOR VALUES('555555555','John','A','Deer',1234567890,2);

INSERT INTO DOCTOR VALUES('123456789','Sheila','E','Nieto',1437778888,3);

INSERT INTO DOCTOR VALUES('987654321','Carlos','D','Glizzy',8888888888,1);

INSERT INTO DOCTOR VALUES('987456321','Beatrice','B','Ballin',1236547897,5);

INSERT INTO DOCTOR VALUES('123654789','Edwardo','F','Squidwardo',0123456789,4);

c) State the Key for the relation and write down Functional Dependencies.

{dssn}->Fname, Minit, Name, Phone_num, dept_num Full Functional

d) State that this relation is in 3NF.

Relation is 3NF

a) Write out the relation (schema) including all attribute names. Indicate keys and foreign

PATIENT(pssn,Fname,Minit, LName, Birth_date,sex,address,dept_num,dssn)

Keys

{pssn},{Fname},{Minit,{Lname},{Birth_date},{sex},{address}

FK

{dept_num}

{dssn}

b) Provide some sample data for the relation (5 rows).

INSERT INTO PATIENT VALUES('147258369','Yves','K','Nieto',TO_DATE('2001-09-10',
'yyyy-mm-dd'),'M','107 Pennsylvania Ave',3,'123456789');

INSERT INTO PATIENT VALUES('963852741','Micheal','','Myers',TO_DATE('1999-10-31',
'yyyy-mm-dd'),'M','666 Jeff ST',1,'987654321');

INSERT INTO PATIENT VALUES('852741963','Freddy','S','Kreuger',TO_DATE('1980-10-31', 'yyyy-mm-dd'),'M','666 John ST',4,'123654789');

INSERT INTO PATIENT VALUES('951789456','Davida','S','Pumpkins',TO_DATE('1999-10-31', 'yyyy-mm-dd'),'F','345 Saturday Lane',2,'555555555');

INSERT INTO PATIENT VALUES('951753852','Lavar','','Ball',TO_DATE('1967-09-01', 'yyyy-mm-dd'),'M','74 Ball st',5,'987456321');

c) State the Key for the relation and write down Functional Dependencies.

{pssn}->Fname,Minit, LName, Birth_date,sex,address,dept_num,dssn Full Functional

{dssn}->dept_num Full Functional

{pssn}->dssn Transitive

d) State that this relation is in 3NF.

Not 3NF because of trasnitive dependency.

a) Write out the relation (schema) including all attribute names. Indicate keys and foreign

PATIENT1(pssn,Fname,Minit, LName, Birth_date,sex,address,dept_num**)**

Keys

{pssn}{Fname}{Minit}{Lname}{Birth_date}{sex}{address}

FK

{dept_num}

PATIENT2(pssn,dssn)

Keys

{pssn}

FK

{dssn}

b) Provide some sample data for the relation (5 rows).

```
INSERT INTO PATIENT1 VALUES('147258369','Yves','K','Nieto',TO_DATE('2001-09-10',
'yyyy-mm-dd'),'M','107 Pennsylvania Ave',3);

INSERT INTO PATIENT1 VALUES('963852741','Micheal','','Myers',TO_DATE('1999-10-31',
'yyyy-mm-dd'),'M','666 Jeff ST',1);

INSERT INTO PATIENT1 VALUES('852741963','Freddy','S','Kreuger',TO_DATE('1980-10-31',
'yyyy-mm-dd'),'M','666 John ST',4);

INSERT INTO PATIENT1
VALUES('951789456','Davida','S','Pumpkins',TO_DATE('1999-10-31', 'yyyy-mm-dd'),'F','345
Saturday Lane',2);

INSERT INTO PATIENT1 VALUES('951753852','Lavar','','Ball',TO_DATE('1967-09-01',
'yyyy-mm-dd'),'M','74 Ball st',5);
```

PATIENT2

INSERT INTO PATIENT2 VALUES('147258369','123456789');

INSERT INTO PATIENT2 VALUES('963852741','987654321');

INSERT INTO PATIENT2 VALUES('852741963','123654789');

INSERT INTO PATIENT2 VALUES('951789456,'555555555');

INSERT INTO PATIENT2 VALUES('951753852','987456321');

c) State the Key for the relation and write down Functional Dependencies.

PATIENT1

{pssn}->Fname,Minit, LName, Birth_date,sex,address,dept_num

PATIENT2

{pssn}->dssn

d) State that this relation is in 3NF.

Both are 3NF

a) Write out the relation (schema) including all attribute names. Indicate keys and foreign

STMT(pssn,stmt_num,is_in_patient,date_arrive,date_left)

Keys

{stmt_num}{is_in_patient}{date_arrive}{date_left}

FK

{pssn}

b) Provide some sample data for the relation (5 rows).

INSERT INTO STMT VALUES('147258369','000000001','0',TO_DATE('2022-09-10', 'yyyy-mm-dd'),TO_DATE('2022-09-10', 'yyyy-mm-dd'));

INSERT INTO STMT VALUES('963852741','000000002','1',TO_DATE('2000-10-31', 'yyyy-mm-dd'),TO_DATE('2022-10-31', 'yyyy-mm-dd'));

INSERT INTO STMT VALUES('852741963','000000003','0',TO_DATE('2022-05-10', 'yyyy-mm-dd'),TO_DATE('2022-05-10', 'yyyy-mm-dd'));

INSERT INTO STMT VALUES('951789456','000000004','1',TO_DATE('2000-01-01', 'yyyy-mm-dd'),TO_DATE('2000-01-02', 'yyyy-mm-dd'));

INSERT INTO STMT VALUES('951753852','000000005','1',TO_DATE('1989-06-22', 'yyyy-mm-dd'),TO_DATE('1989-06-25', 'yyyy-mm-dd'));

c) State the Key for the relation and write down Functional Dependencies.

{stmt_num}->is_in_patient,date_arrive,date_left,pssn Fully Functional

d) State that this relation is in 3NF.

This relation is 3NF

a) Write out the relation (schema) including all attribute names. Indicate keys and foreign

SERVICES_PROVIDED_TO_PATIENT(order_no,pssn,dssn,serv_name,serv_cost,serv_qty,stmt_num,serve_date)

Keys

{order_no}{stmt_num}{serv_name}{serv_cost}{serv_qty}{serv_date}

FK

{pssn}{dssn}

b) Provide some sample data for the relation (5 rows).

INSERT INTO SERVICES_PROVIDED_TO_PATIENT

VALUES(1,'147258369','123456789','Contact Allergy

Testing',500,1,'000000001',TO_DATE('2022-09-10', 'yyyy-mm-dd'));

INSERT INTO SERVICES_PROVIDED_TO_PATIENT

VALUES(2,'963852741','987654321','Trauma care',200,1,'000000002',TO_DATE('2022-10-31',

'yyyy-mm-dd'));

INSERT INTO SERVICES_PROVIDED_TO_PATIENT

VALUES(3',951789456','555555555','Women''s Health

Checkup',50,1,'000000003',TO_DATE('2005-10-31', 'yyyy-mm-dd'));

INSERT INTO SERVICES_PROVIDED_TO_PATIENT

VALUES(4,'951753852','987456321','Proton

Therapy',5000,1,'000000004',TO_DATE('2000-01-01', 'yyyy-mm-dd'));

INSERT INTO SERVICES_PROVIDED_TO_PATIENT

VALUES(5,'852741963','123654789','Facial

Reconstruction',19000,1,'000000005',TO_DATE('2022-05-10', 'yyyy-mm-dd'));

c) State the Key for the relation and write down Functional Dependencies.

order_no->,pssn,dssn,serv_name,serv_cost,serv_qty,stmt_num,serve_date

d) State that this relation is in 3NF.

Yes 3NF

a) Write out the relation (schema) including all attribute names. Indicate keys and foreign

PHARMACY(pharm_num,phone_num**)**

Key

{pharm_num}

{phone_num}

b) Provide some sample data for the relation (5 rows).

```
INSERT INTO PHARMACY VALUES(1,'6666666666');
INSERT INTO PHARMACY VALUES(2,'7777777777');
INSERT INTO PHARMACY VALUES(3,'0101010101');
INSERT INTO PHARMACY VALUES(4,'3333333333');
INSERT INTO PHARMACY VALUES(5,'6588888888');
```

c) State the Key for the relation and write down Functional Dependencies.

{pharm_num}->phone_num

d) State that this relation is in 3NF.

Yes 3NF

a) Write out the relation (schema) including all attribute names. Indicate keys and foreign

Medicine(med_name,med_cost)

Keys

{med_name}{med_cost}

b) Provide some sample data for the relation (5 rows)

INSERT INTO MEDICINE VALUES('Advil',10);

INSERT INTO MEDICINE VALUES('Pain Pills',5);

INSERT INTO MEDICINE VALUES('Sleeping Pills',5);

INSERT INTO MEDICINE VALUES('Ibeuprofen',30);

INSERT INTO MEDICINE VALUES('Anti-Acid',20);

c) State the Key for the relation and write down Functional Dependencies.

{med_name}->med_cost

d) State that this relation is in 3NF.

Yes 3NF

Phase III a) System Requirements

The Hospital is organized into DEPARTMENTs:

- Each department has a unique name,unique number, department specialization and a doctor who *manages* the department
- We keep track of the start date of the department manager
- A department may have several services provided

Each Department *handles* a number of PATIENTs

- Each patient has a ssn, name, address, sex,in/outpatient, and birthdate
- A statement gets *raised on* a patient for services provided
- Each patient gets treated by one or many doctors

The database will track the Hospitals DOCTORs

- Each doctor has a ssn, name,phone#, and specialty
- Each doctor *works for* one department, but can *treat* many patients
- Each doctor *provides* services to patients
- Doctors can *prescribe* many or no medicine to patients
- Each *prescription* has a quantity and date associated with it

The database will track the hospitals PHARMACY

- Each pharmacy has a unique number, and phone#
- The pharmacy *provides* medicine for a prescription

The database will track PRESCRIPTIONS

- Each prescription has a order number, patient ssn, doctor ssn, medicine name, quantity, prescription date, statement number and pharmacy number

The database will track MEDICINE

- Medicine has a cost and unique name.

The database will track STATEMENT

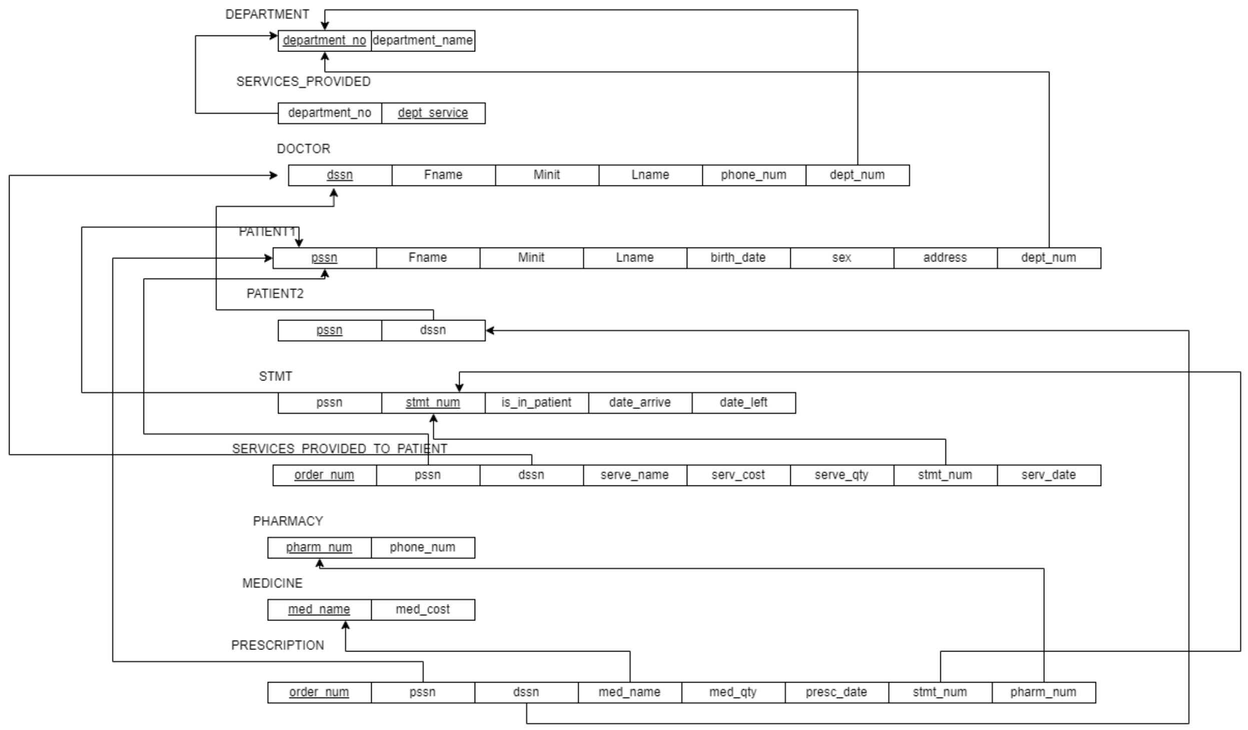
- Statement has a unique statement#, and the total amount due

The database will track SERVICES

- Services have a unique name, and cost

In the final system requirements we were required to make a change by renaming the medicine table to prescriptions and then making a new and simpler table for medicines, as previously a doctor would be provide medicine to a patient, but what makes more sense is to have a doctor provide a prescription to a patient with much more info such as cost, date, quantity. Prescription being the ternary relationship for Doctor Patient and medicine needed a table that was lacking in the previous version. It was called medicine which we repurposed to hold the attributes med_name and med_cost.

RELATIONAL MODEL



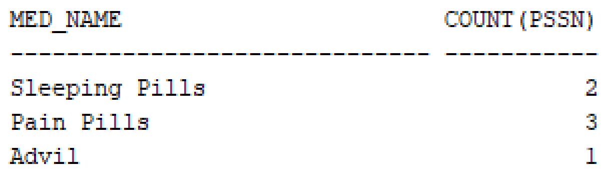
Due to Normalization of the ERR diagram it had changed alot with new inclusions of PATIENT1, PATIENT2, PRESCRIPTION and MEDICINE to fit the 3NF requirements.

4- Write four queries in English and answer in SQL code, show a screen shot or SNIP that shows the SQL code and the result (You may use the same queries from Phase II).

For the four SQL Queries:

1 contains GROUP BY

/*get the medicine name and how many times it was prescribed at the hospital*/
SELECT   MED_NAME, COUNT(pssn)
FROM   PRESCRIPTION
GROUP BY   MED_NAME;

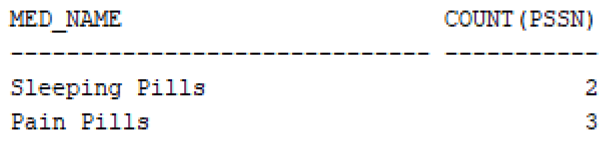


```
MED_NAME                       COUNT(PSSN)
------------------------------ -----------
Sleeping Pills                           2
Pain Pills                               3
Advil                                    1
```

1 contains GROUP BY and HAVING

/*get the medicine name and how many times it was prescribed at the hospital if it prescribed more than once*/

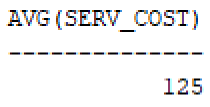
SELECT   MED_NAME, COUNT(pssn)
FROM   PRESCRIPTION
GROUP BY   MED_NAME
HAVING COUNT(pssn)>1;



```
MED_NAME                       COUNT(PSSN)
------------------------------ -----------
Sleeping Pills                           2
Pain Pills                               3
```

1 contains nested query with ALL

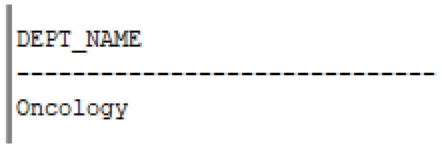
/*get average cost of a serv providd to patients in dept 1 and 2*/
SELECT AVG(serv_cost)
FROM SERVICES_PROVIDED_TO_PATIENT
WHERE dssn IN (
  SELECT dssn
  FROM DOCTOR
  WHERE dept_num = 1 OR dept_num = 2
) ;



```
AVG(SERV_COST)
--------------
           125
```

1 contains nested query with IN

```
/*return all dept names which have the proton therapy service*/
SELECT dept_name
FROM DEPARTMENT
WHERE dept_num = ALL(SELECT dept_num
FROM services_provided
WHERE DEPT_SERVICES='Proton Therapy');
```



```
DEPT_NAME
------------------------------
Oncology
```

5- A narrative conclusion section that describes:

a) the group's experience with the project (which steps were the most difficult? Which were the easiest? what did you learn that you did not imagine you would have? if you had to do it all over again, what would you have done differently?)

Overall the experience was not too tough. The most complicated parts were fixing and standardizing the entity relationship diagrams as they can be quite complicated to spot and simplify redundancy or unnecessary relationships. The relational schema is based off the ER so once you fix the ER, it's quite easy to make. The most difficult part might be creating the queries because with a somewhat abstract project, it can be hard to come up with meaningful/useful queries. On top of that, queries were the most complicated actual sql we did for the project but that itself was not too bad. I don't think we would have done anything differently overall, except adhere more to the 3NF standard if we knew from the beginning.

b) any final comments and conclusions.

Databases seem very viscous and hard to comprehend at first glance considering there can be so many different relations, but once broken down and studied a bit, its not all that bad. Especially once you get into the nitty gritty of it by making one yourself and inserting and updating as well as making queries that make sense/can be useful in the database. Our professor really helped out with the ER review because of that we were able to have a strong foundation for our project.