TaSk-12

Mini ProJect

2.1 Normalization steps:-

First normal form!- Ensure atomic attribute values for each table.

second normal Form!- Remove partial dependenceics ;each non-key attributs must depend on the whole primary key.

Third normal Form!- Remove transitie dependencis; non-key attribs depend only on the primary key.

candidate keys examples

- Patient :- Patient id (unique)
- Policy :- Policy number (unique)
- claim!- claimid (unique)
- Hospital!- Hospital id (unique)

2.2. SQL Queries to process, approve and track insurance claims.

Insert new claim!-

```
INSERT INTO claim[colaimid, policy id, patienid, Hospital id,
 claim number, amount).
 VALUES [1001, 'PN123', 1, 101, 22, 8000]
```

Approve a claim

```
 update claim SET status =' Approval'.
Approval Date = '2023-02-01". WHERE cldmid = 1001,
```

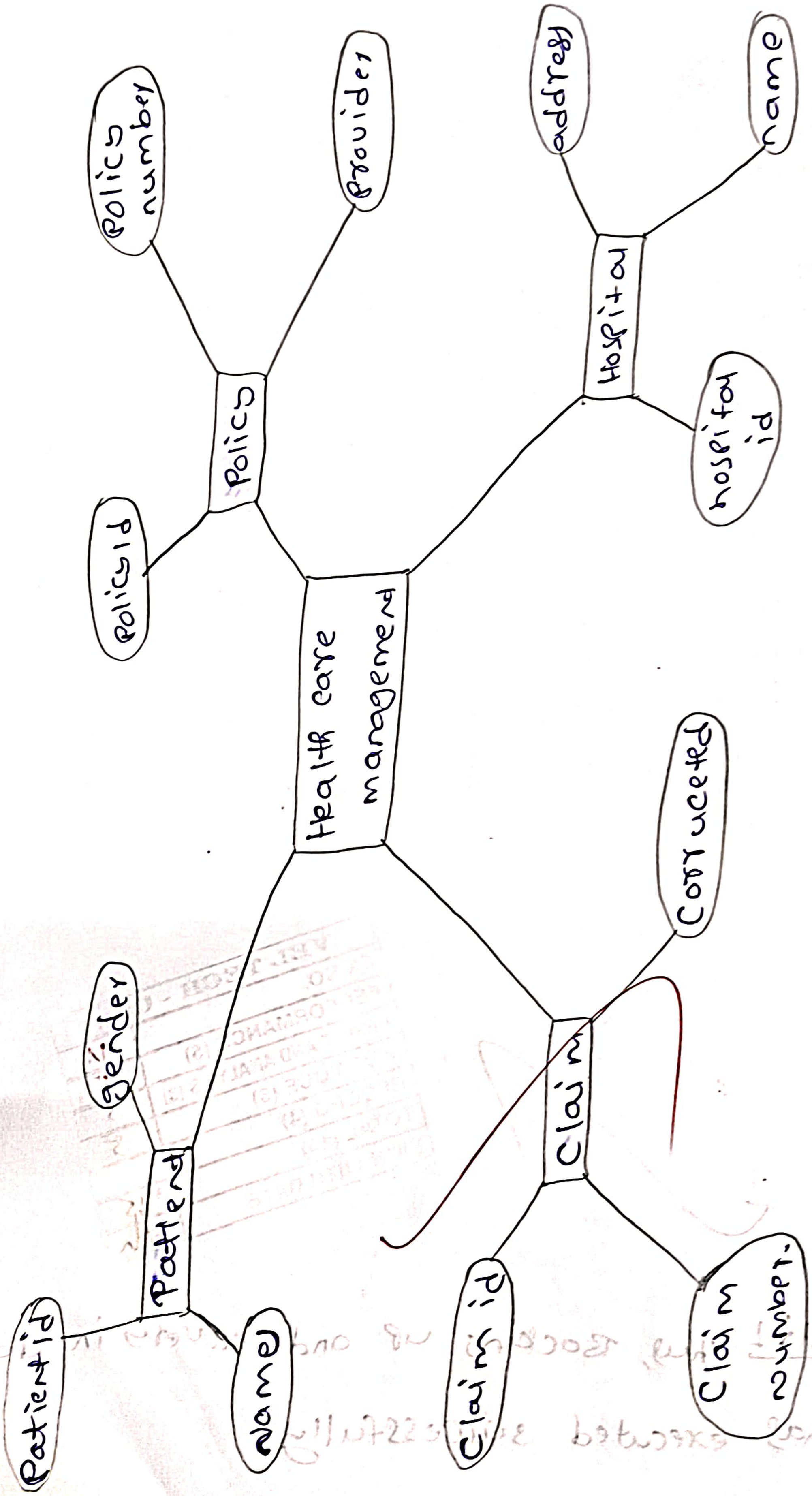
- Track claims for a given Policy:

```
SELECT ''FROM claim' WHERE Policy number ='PN123';
```

- calculate total approved claims per Patient!

```
SELECT patientid, sum(Amount) s TOTAl APPROVAl
FROM claim
WHERE. status = 'APProved'
GROUP BY_ Patient id;
```

E-R Diasram



Health care management
Polics
Policyid
Polics number
Provider
Hospital
address
name
hospital id
Claim
Corrucetd
Claim id
Claim number.
Patient
gender
Patient id
Name

2.3 Transaction management during concurrent claims approvals.

1. Optimistic concurrency. control :-

```
UPDATE Claim
  SET status =' Approved', version = version +1
WHERE Claim id ='uuid -1234' AND Status ='IN-Review'
AND.version = 3;
```

2. Pessimistic locking

```
BEGIN;
SELECT Status FROM Claim WHERE Claim-id =' uuid-1234'
FOR UPDATE; UPDATE claim SET status =' Approved' WHERE
Claim-id ='uuid-1234'; COMMIT;
```

3. Idempotency. & business-level. compensation.
make approval operations idempotent

SUBMITTED -> IN-Review -> Approved.

4. Isolation Levels.
use read commited typically, or REPEATABLE.READ
SERIALIZE if strict. serializibity. is required.

2.4. Performing CRUD operations in MongoDB to manage claims records.

- Basic mongoDB CRUD operations for claim records:-

- Create:-

```
db.claims.insertOne({
    claimid: 1001;
  Policy number = 'PN123';
    Patient id =1,
"timestamp" = "2025-01-15T08:30.003",
"sensors" :{
 "Temperature" = 23.5,
  "humidity" = 45.5,
};
  "status" = "active",
 "metadata" : {
    "manufacturp" =" AcmeDevices",
    "Firemaare version" : "1.2.5"
  }
}
```

How Indexing steps

```
db. devices bata.create index [{"device id" :: 1}]);
db.device Data.create index [{"Location.LocationId" :1}]);
db.device Data. create index [{"devis id:"1; "Location.
                                         Location Id" : 1}]);
```

Benefits

- Fetch all records of a device in different Location
- Query all devices in a location.
- Retrive temperature reading acros) all thermostats

VEL TECH - CSE	
EX NO.	12
PERFORMANCE (5)	5
RESULT AND ANALYSIS (3)	5
VIVA VOCE (3)	5
RECORD (4)	
TOTAL (15)	15
SIGN WITH DATE	

Result

Thus the user case is successfully verified and executed successfully