Normalization

Normalization is a process of organizing data in the database.

It is used to minimize the redundancy from relations or a set of relations.

➢ *1st Normal Form:*

It says that a relation is in 1NF if and only if:

- The domain of each attribute contains only atomic value.
- The value of each attribute contains only a single value from that domain.
- There should not be repeating groups.

➢ *2nd Normal Form:*

It says that a relation is in second normal form if and only if:

- The relation should be in 1NF.
- Every non key attribute should be fully functionally dependent on each candidate key.

➢ *3rd Normal Form:*

It says that a relation is in 3rd normal form if and only if:

- It is in 2NF.
- Transitive functional dependency of a non-prime attribute on any candidate key should be removed. Or

All Fields can be determined only by the key in the table and no other column.

Theatre (th_id, loc, city, state)

Th id	loc	city	state
100	Akota	Vadodara	Gujarat
101	Sarkhej	Ahmedabad	Gujarat

Each cell is single valued

The table is in 1NF.

The table is in 2NF as there are no candidate keys.

th_id --> loc and loc --> city hence th_id --> city

Also, city --> state so th_id --> state.

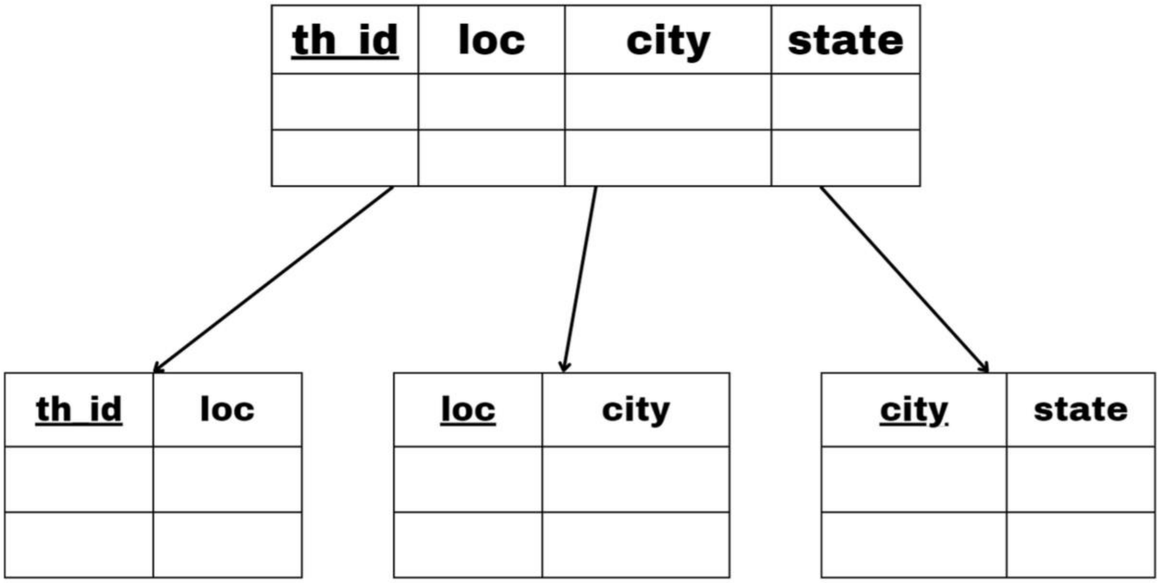
Since there is transitivity, the table is not in 3NF.

The table can be divided as:

Th_main (th_id, loc)

Th_loc (loc, city)

Th_st (city,_state)



Screen (screen_id, th_id, silver_seats, gold_seats, screen_name)

screen id	th id	silver_seats	Gold_seats	screen_name

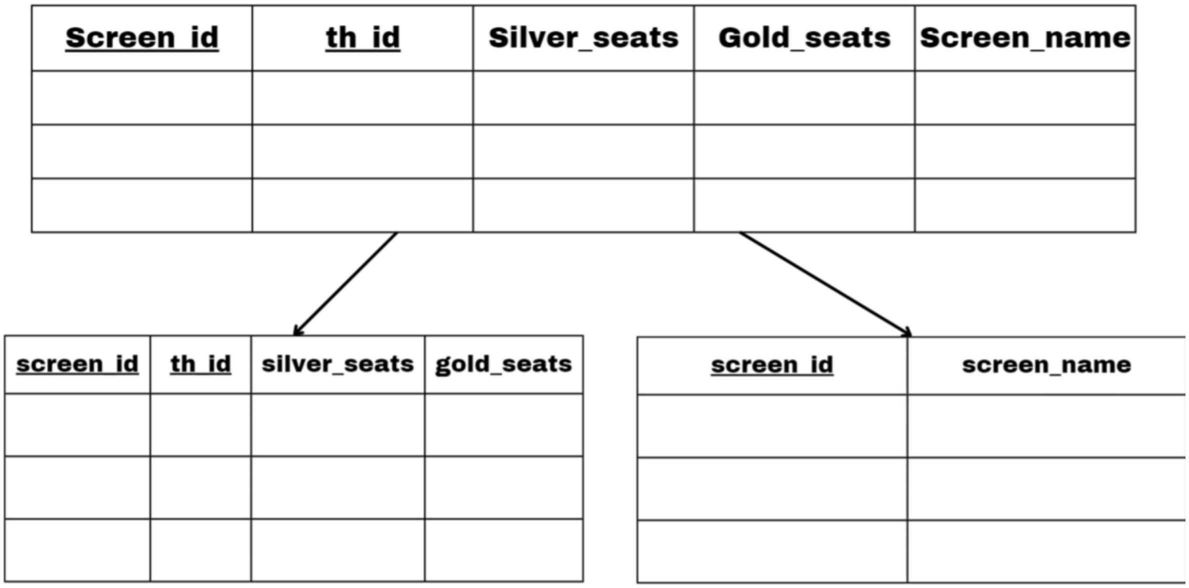
Each attribute is single valued. Therefore, it is in 1NF.

The number of silver seats and gold seats a screen will have is dependent on screen_id as well th_id. However, screen names depend on only screen_id.

So, table can be divided as:

Screen (screen_id, th_id, silver_seats, gold_seats)

Screen_name (screen_id, screen_name)



Screen id	th id	Silver_seats	Gold_seats	Screen_name

screen id	th id	silver_seats	gold_seats

screen id	screen_name

Both the tables are in 2NF.

Since Screen table has no transitivity, it is in 3NF.

Projection_Room (screen_id, th_id, server, type)

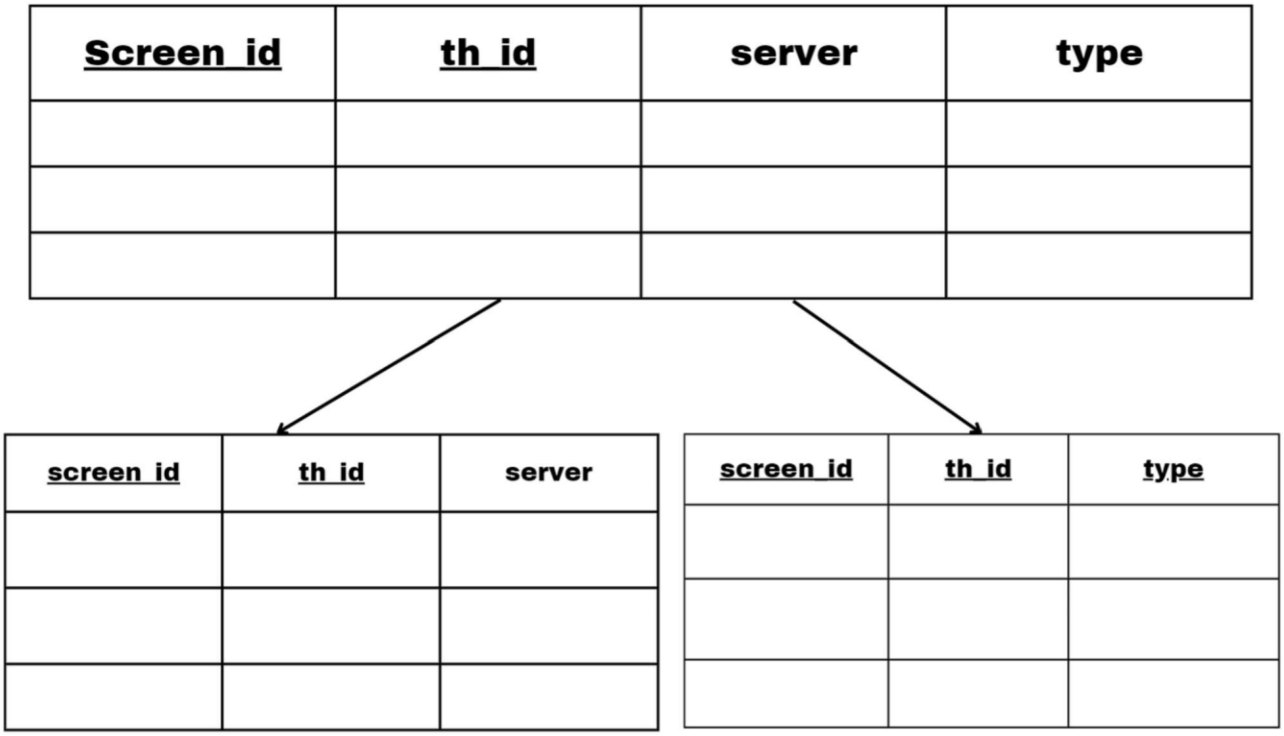
Screen id	th id	server	type

Since there can be multivalve for type attribute (2D, 3D), the table is not in 1NF.

The table can be divided as:

Proection_Room (screen_id, th_id, server)

Projection_type (screen_id, th_id, type)



Screen id	th id	server	type

screen id	th id	server

screen id	th id	type

As server depends on screen_id and th_id together the table Pro_Room is in 2NF. There is no more than 1 non-prime attribute

So, it is also in 3NF.

Pr_type has no non-prime attribute, so it is also in 2NF and in 3NF.

Employee (eid, th_id, ename, email, phone_no, job, hiredate, sal)

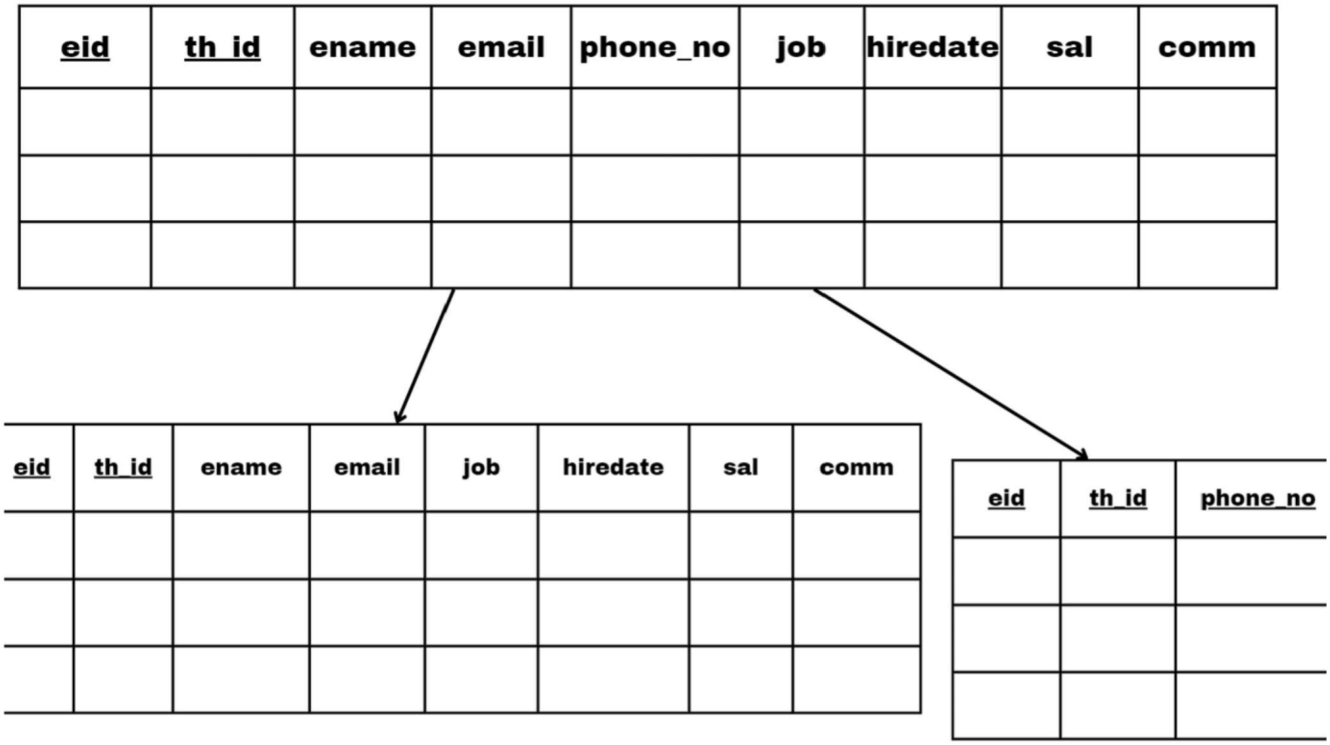
e_id	th_id	ename	email	Phone_no	job	hiredate	sal	comm

One employee can have more than one contact number, so the table is not in 1NF.

It can be divided as:

Employee (eid, th_id, ename, email, job, hiredate, sal, comm)

Em_no (eid, th_id, phone_no)



eid	th_id	ename	email	phone_no	job	hiredate	sal	comm

eid	th_id	ename	email	job	hiredate	sal	comm

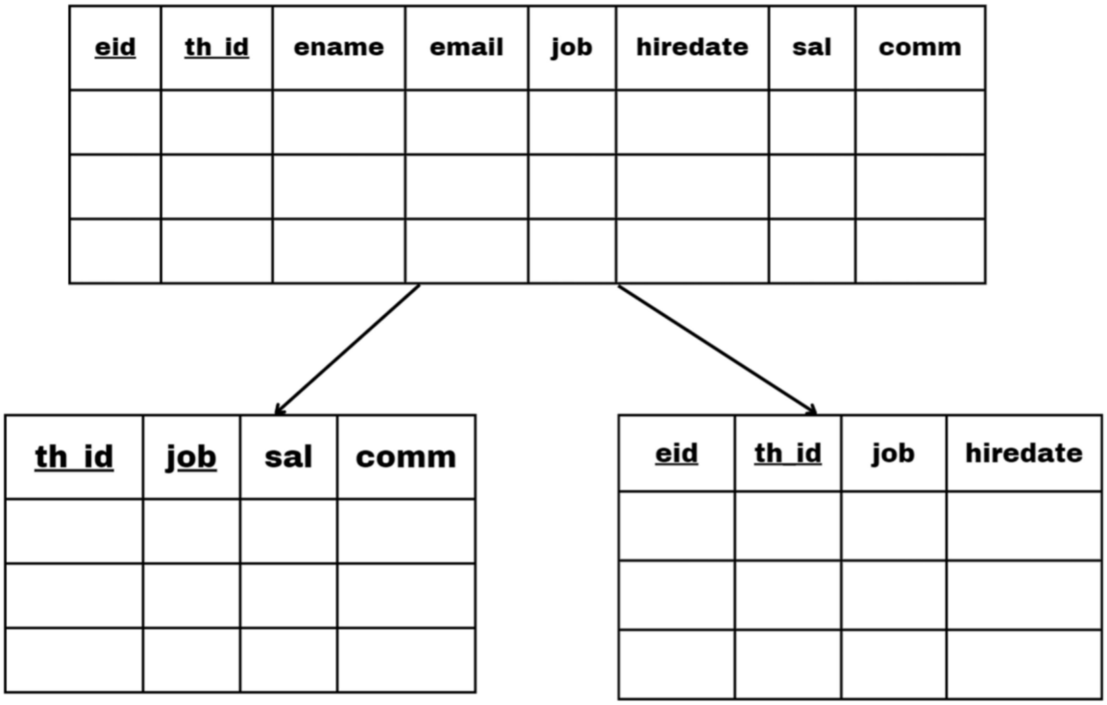
eid	th_id	phone_no

As ename depend only on eid and not on th_id so the table Employee is not in 2NF and sal depend on job and th_id not on eid. So table is divided as:

Employee_n (eid, ename)

Employee_jsal(th_id, job, sal, comm)

Employee (eid, th_id, email, job, hiredate)



eid	th id	ename	email	job	hiredate	sal	comm

th id	job	sal	comm

eid	th id	job	hiredate

Projectionist (pid, th_id, shift)

pid	th_id	shift

Each projectionist will have only one shift, so it is in 1NF.

As shift depends on pid as well as th_id, it is in 2NF.

Since there is no more than 1 non-prime attribute, it is in 3NF.

Movie (mov_name, release_dt, duration, lang, genre, budget)

Mov name	Release_dt	duration	language	genre	budget	M_share

As one movie has many genres and it can be in more than 1 language, the table is not in 1NF.

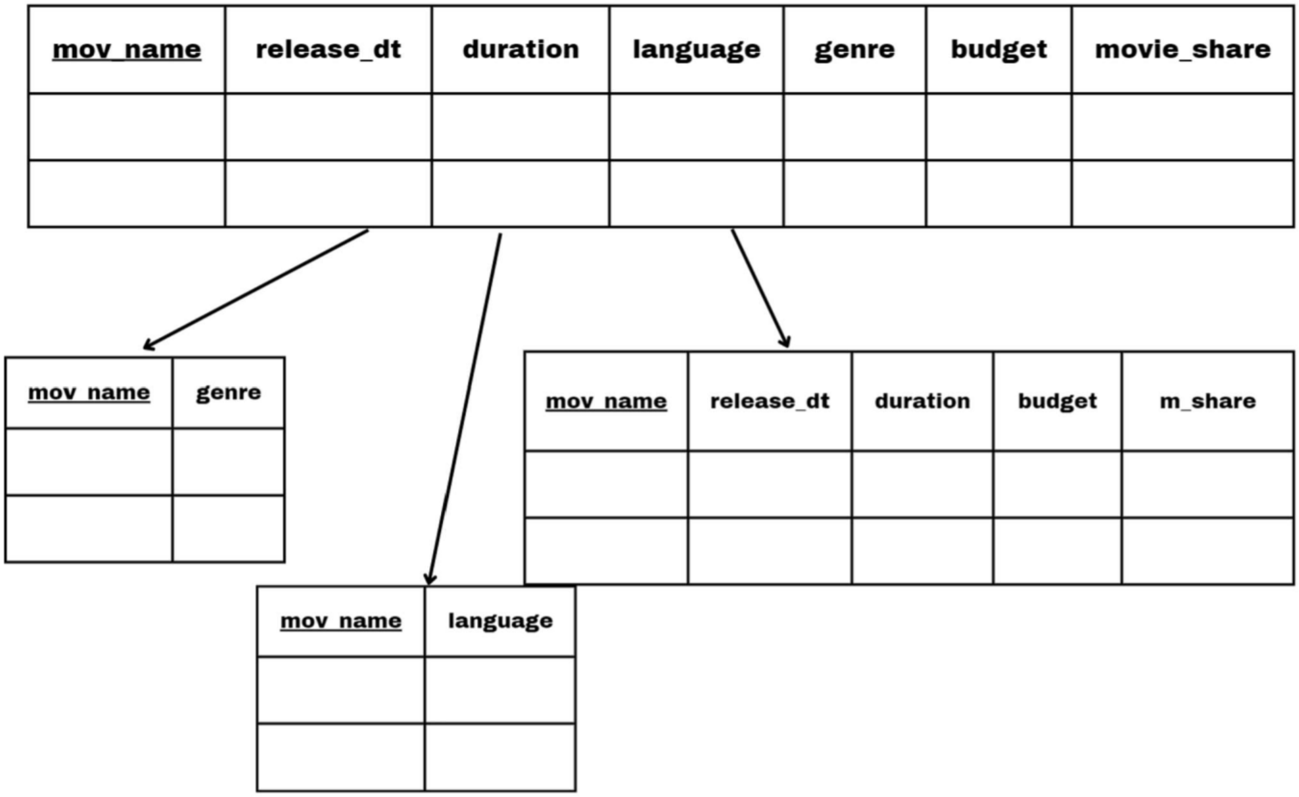
It can be divided as:

Movie (mov_name, release_dt, duration, budget, movie_share)

Movie_genre (mov_name, genre)

Movie_lang (mov_name, language)

As there is no non-prime attribute in Movie_genre and Movie_lang, they both are in 2NF and 3NF.



mov_name	release_dt	duration	language	genre	budget	movie_share

mov_name	genre

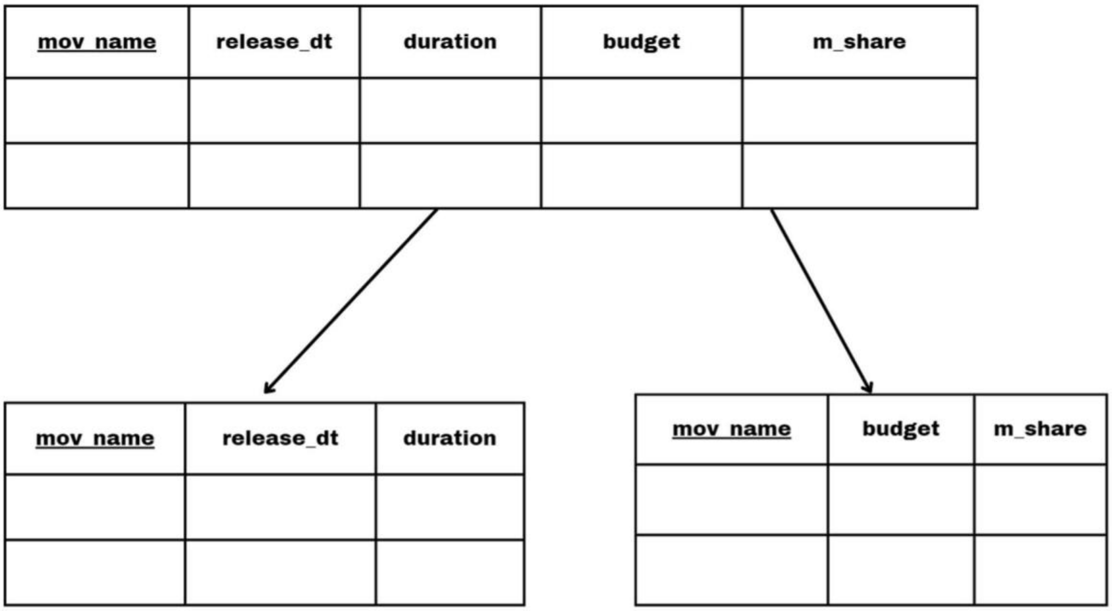
mov_name	release_dt	duration	budget	m_share

mov_name	language

M_share depend on budget and budget depend on mov-name. So there is transitivity. So it is not in 3NF. The table can be divided as :

Movie (mov_name, release_dt, duration)

Movie_share (mov_name, budget, m_share)



mov_name	release_dt	duration	budget	m_share

mov_name	release_dt	duration

mov_name	budget	m_share

Shows (screen_id, th_id, sdatetime, mov_name, runtime)

Screen id	Th id	sdatetime	Mov_name	runtime

Mov_name and runtime will be single valued for same screen_id, th_id and sdatetime together.

So, the table is in 1NF.

Mov_name and runtime both are totally dependent on the candidate key (screen_id, th_id, sdatetime). So, the table is in 2NF.
As there is no transitivity the table is in 3NF.

Silver_seat (silver_seatnum, screen_id, th_id, sdatetime, availability, price)

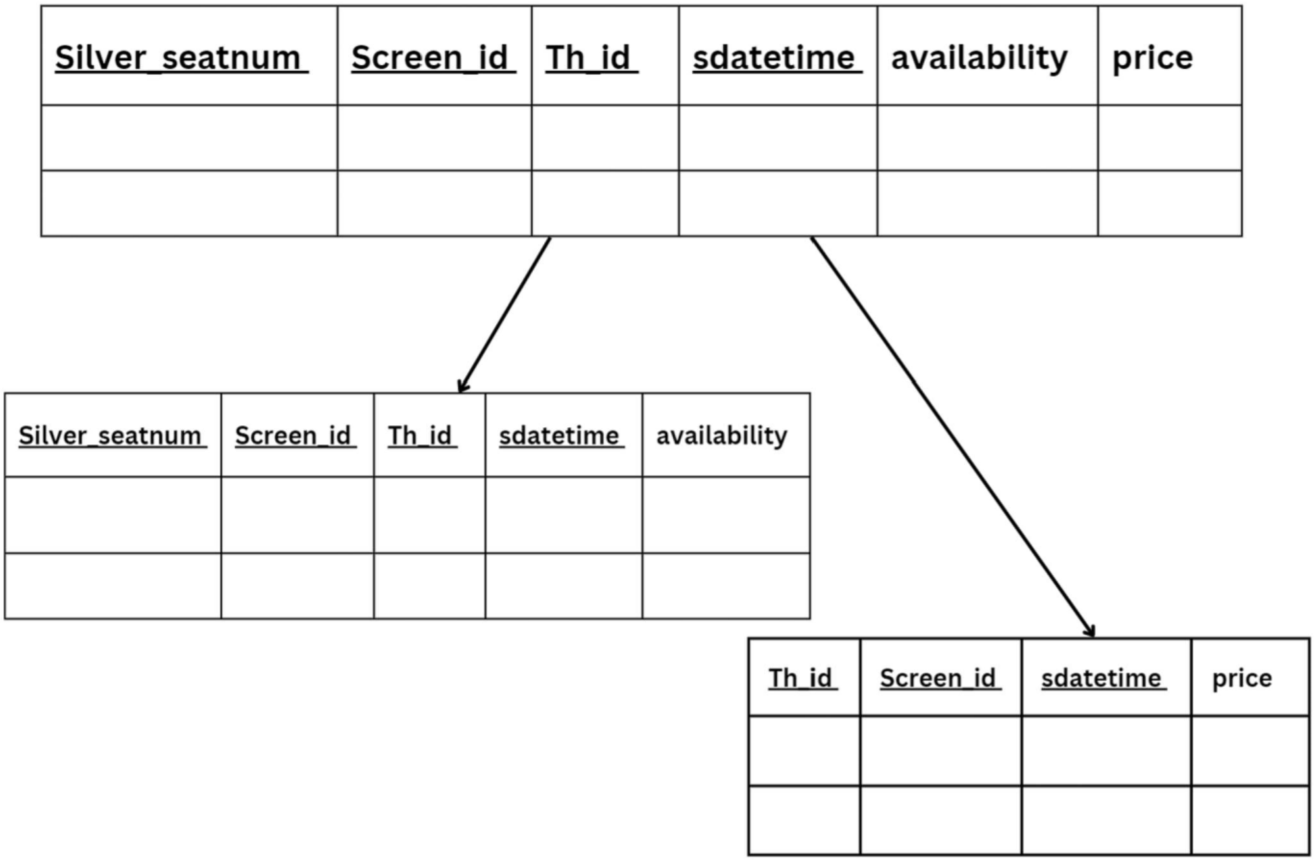
Silver_seatnum	Screen_id	Th_id	sdatetime	availability	price

Availability is a single value attribute, so the table is in 1NF.

Now price depends on screen_id, th_id and sdatetime only because each seat is silver. Therefore, the table is not in 2NF. The table is divided into:

Silver_seat (silver_seatnum, screen_id, th_id, sadtetime, availability)

Silver_price(th_id, screen_id, sdatetime, price)



Silver_seatnum	Screen_id	Th_id	sdatetime	availability	price

Silver_seatnum	Screen_id	Th_id	sdatetime	availability

Th_id	Screen_id	sdatetime	price

Availability depends on the candidate key totally so the silver_seat table is in 2NF. Price depends on th_id, screen_id and sdatetime together. So, table silver_price is also in 2NF.

As there is no more than 1 non-prime attribute the tables are in 3NF.

Gold_seat (gold_seatnum, screen_id, th_id, sdatetime, availability, price)

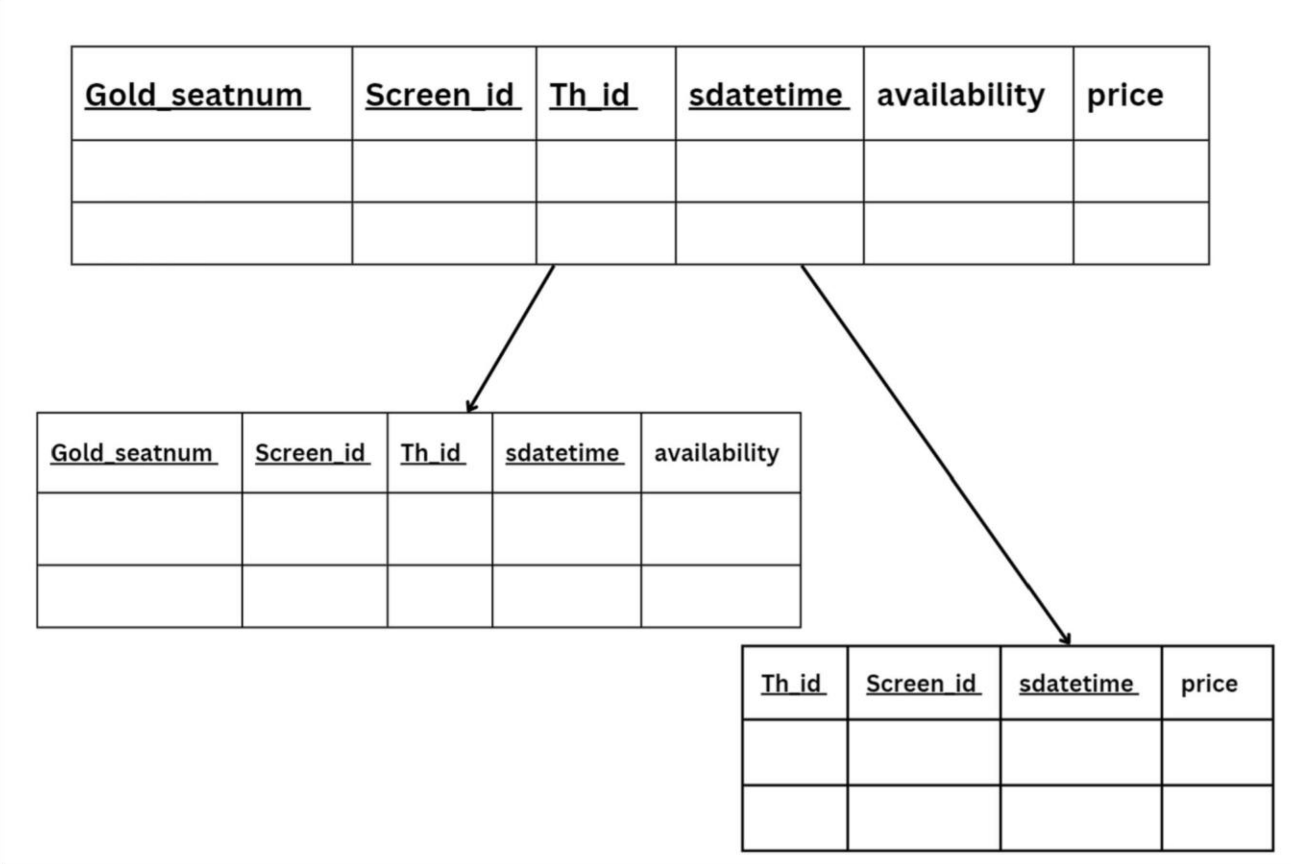
Gold_seatnum	Screen_id	Th_id	sdatetime	availability	price

Availability is a single value attribute, so the table is in 1NF.

Now price depends on screen_id, th_id and sdatetime only because each seat is gold. Therefore, the table is not in 2NF. The table is divided into:

Gold_seat (gold_seatnum, screen_id, th_id, sadtetime, availability)

Gold_price(th_id, screen_id, sdatetime, price)



Gold_seatnum	Screen_id	Th_id	sdatetime	availability	price

Gold_seatnum	Screen_id	Th_id	sdatetime	availability

Th_id	Screen_id	sdatetime	price

Availability depends on the candidate key totally so the gold_seat table is in 2NF. Price depends on th_id, screen_id and sdatetime together. So, table gold_price is also in 2NF.

As there is no more than 1 non-prime attribute the tables are in 3NF.

Customer (Cid, cname, phone_num, email, payment_method, th_id)

cid	Th_id	cname	Phone_num	email	Payment_method

As each attribute is a single valued the table is in 1NF.

It is in the 2NF as there are no candidate keys.

There is no transitivity, the table is in 3NF.

SilverTicket (ticket_id, th_id, nprice, bk_dt_time, bk_type, sdatetime, screen_id, silver_seatnum, cid, webname)

Ticket_is	Th_id	nprice	Bk_dt_time	Bk_type	sdatetime	Screen_id	Silver_seatnum	cid	webname

GoldTicket (ticket_id, th_id, nprice, bk_dt_time, bk_type, sdatetime, screen_id, gold_seatnum, cid, webname)

Ticket_is	Th_id	nprice	Bk_dt_time	Bk_type	sdatetime	Screen_id	gold_seatnum	cid	webname

Boxoffice (th_id, sdatetime, screen_id, mov_name)

Th_id	sdatetime	Screen_id	Mov_name

Each attribute is single valued, so the table is in 1NF.

Screen_id depends on th_id, sdatetime and mov_name together. So the is in 2NF.

As there are no more than 1 non-prime attribute the table is in 3NF.

Website (wid, sdatetime, screen_id, mov_name)

wid	sdatetime	Screen_id	Mov name

Each attribute is single valued, so the table is in 1NF.

Screen_id depends on wid, sdatetime and mov_name together. So the is in 2NF.

As there are no more than 1 non-prime attribute the table is in 3NF.