ER Diagram

Grupo: RedPush

Hector Gabriel Aponte Santiago - hector.aponte8@upr.edu

Leonardo Torres De La Rosa - leonardo.torres5@upr.edu

Roberto Carlos Hernandez Martinez - roberto.hernandez13@upr.edu

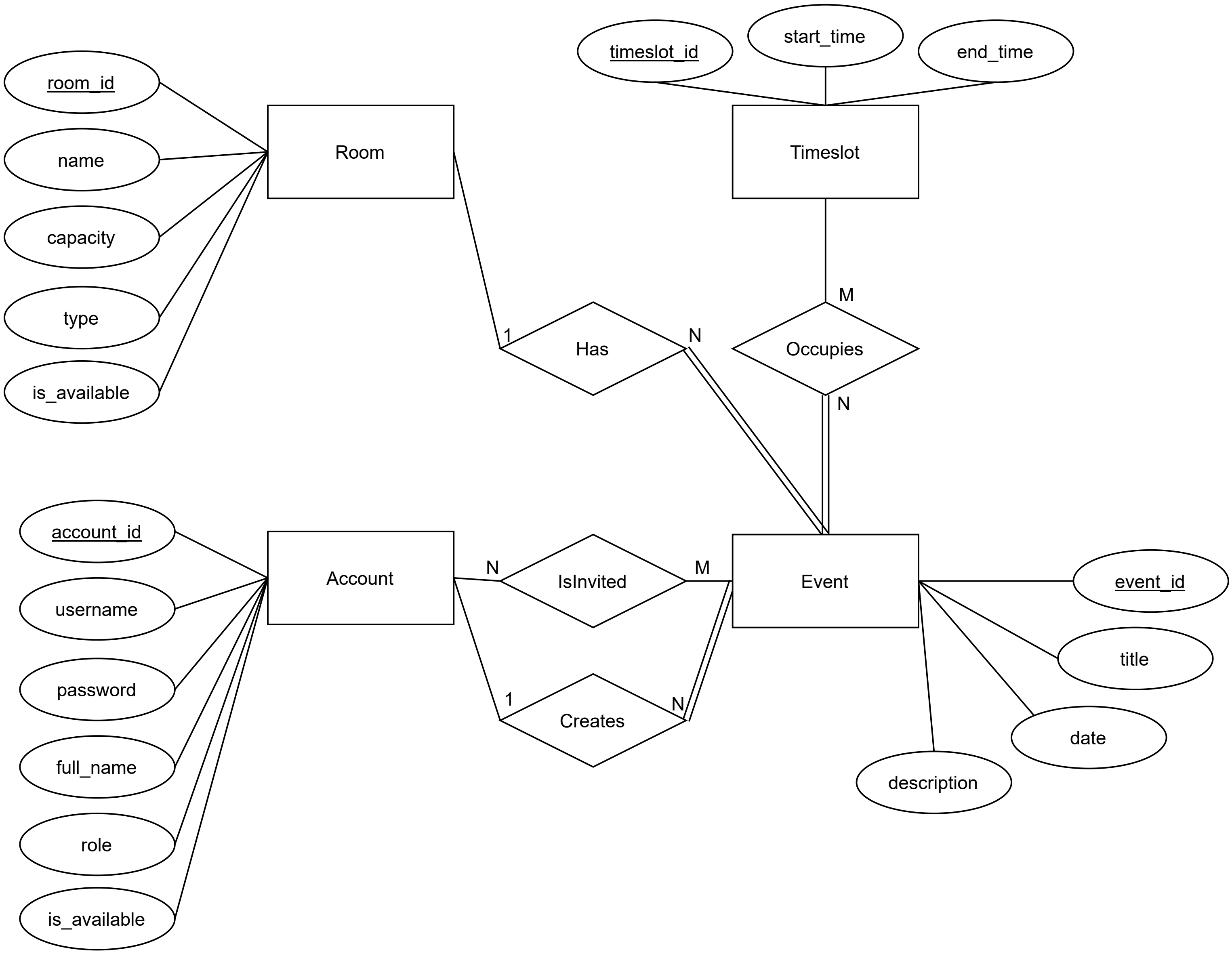
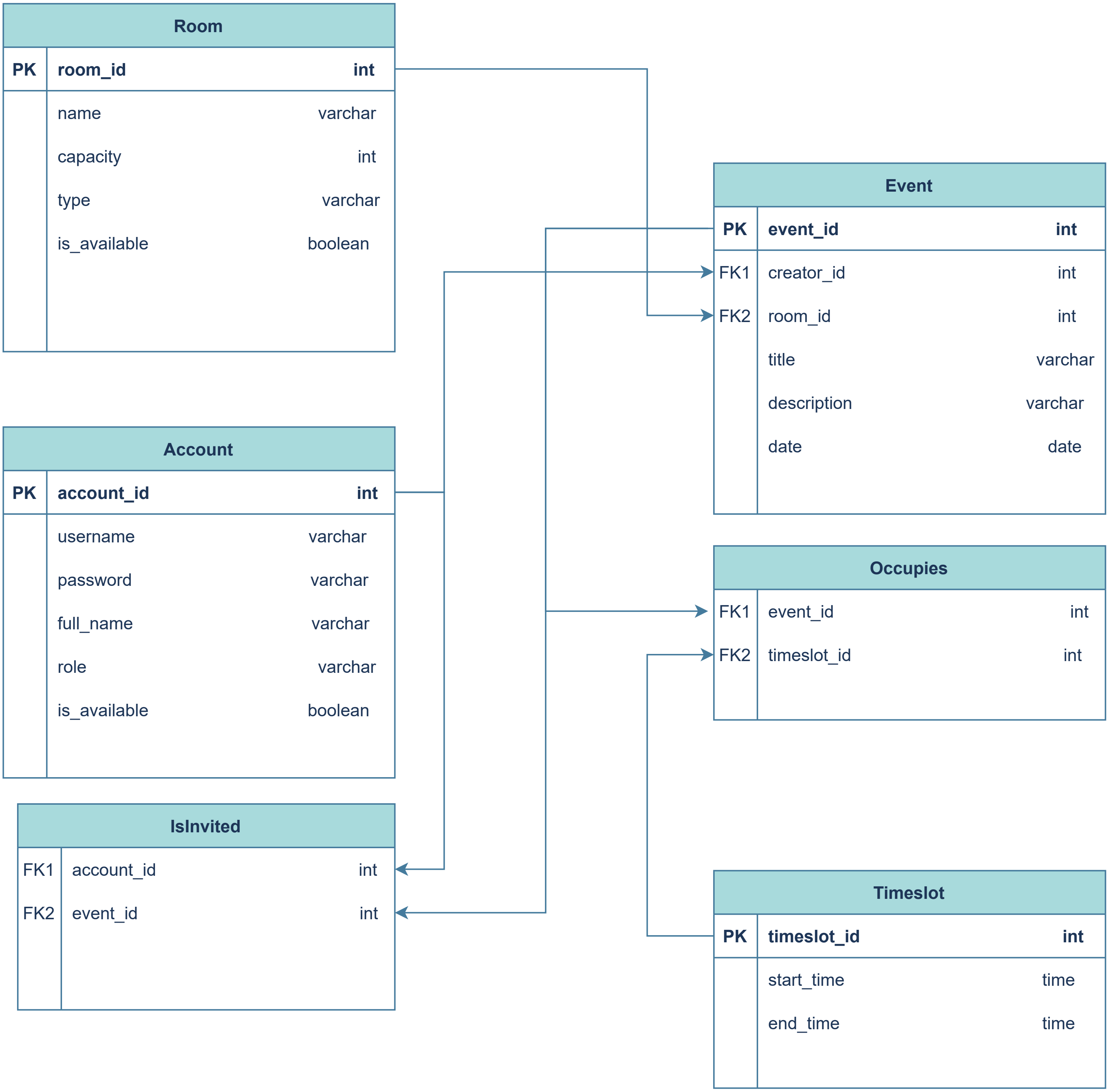


Table Diagram



Room

PK	**room_id**	**int**
	name	varchar
	capacity	int
	type	varchar
	is_available	boolean

Account

PK	**account_id**	**int**
	username	varchar
	password	varchar
	full_name	varchar
	role	varchar
	is_available	boolean

IsInvited

FK1	account_id	int
FK2	event_id	int

Event

PK	**event_id**	**int**
FK1	creator_id	int
FK2	room_id	int
	title	varchar
	description	varchar
	date	date

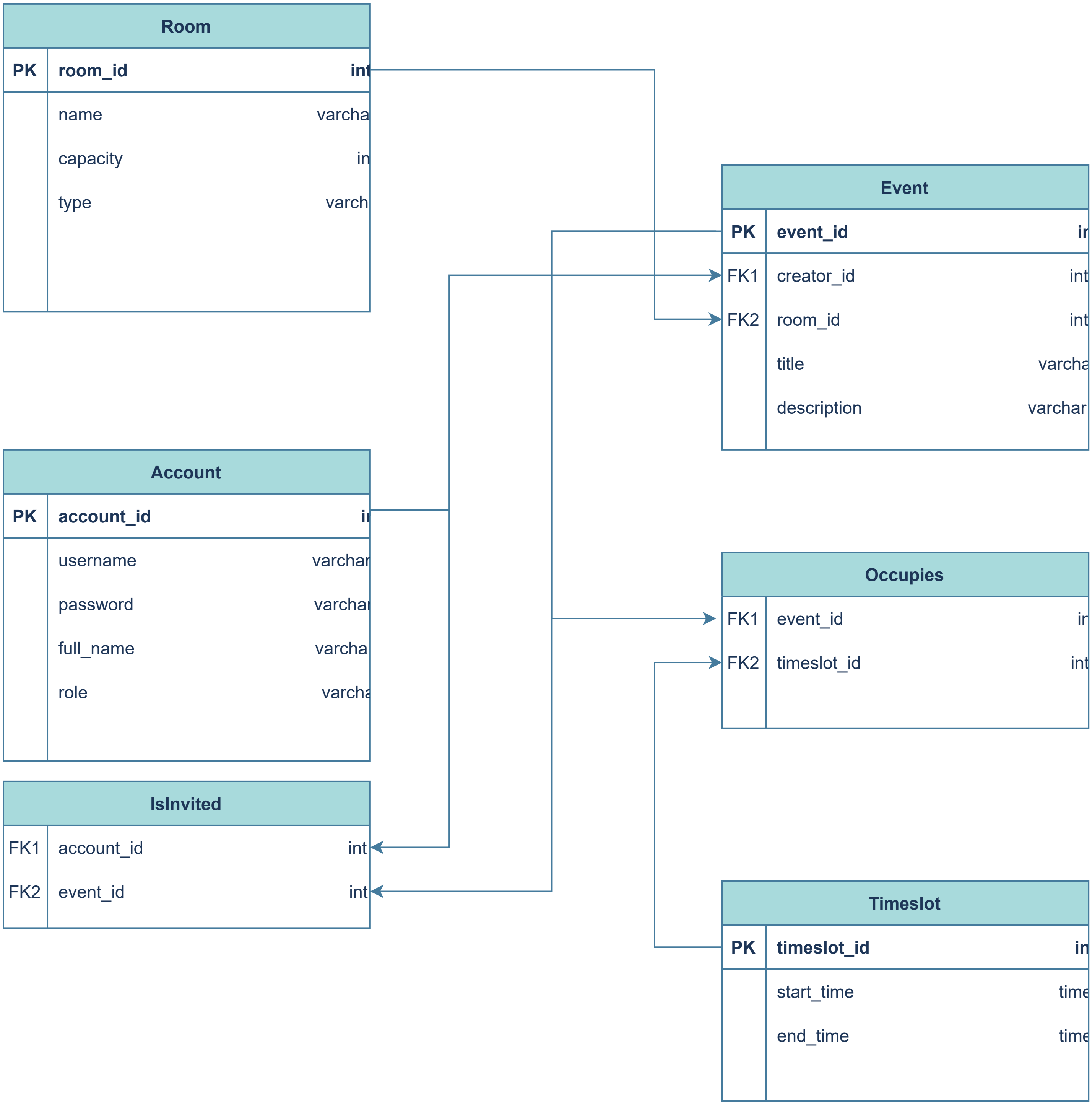
Occupies

FK1	event_id	int
FK2	timeslot_id	int

Timeslot

PK	**timeslot_id**	**int**
	start_time	time
	end_time	time

Table Diagram



Room

PK	**room_id**	**int**
	name	varcha
	capacity	in
	type	varch

Account

PK	**account_id**	**i**
	username	varchar
	password	varchar
	full_name	varcha
	role	varcha

IsInvited

FK1	account_id	int
FK2	event_id	int

Event

PK	**event_id**	**i**
FK1	creator_id	int
FK2	room_id	int
	title	varcha
	description	varchar

Occupies

FK1	event_id	i
FK2	timeslot_id	int

Timeslot

PK	**timeslot_id**	**in**
	start_time	time
	end_time	time

SAMPLE QUERIES

1. Register a new user

```
INSERT INTO account (username, password, full_name, role)
VALUES ('admin', 'password', 'Administrator', 'Department Staff');
```

2. Find an available room (lab, classroom, study space, etc.) at a time frame

```
SELECT r.id, r.name FROM room r WHERE r.id NOT IN
    (SELECT r.id FROM room r INNER JOIN event e on r.id = e.room_id
INNER JOIN occupies o on e.id = o.event_id INNER JOIN timeslot t on t.id
= o.timeslot_id
        WHERE (start_time BETWEEN '2021-01-10 10:00:00'::timestamp AND
'2021-09-19 19:00:00'::timestamp)
       AND (end_time BETWEEN '2021-01-10 10:00:00'::timestamp AND '2021-
09-19 19:00:00'::timestamp));
```

3. Find who appointed a room at a certain time

```
SELECT a.full_name, t.start_time, t.end_time FROM account a INNER JOIN
event e on a.id = e.creator_id INNER JOIN room r on e.room_id = r.id
INNER JOIN occupies o on e.id = o.event_id INNER JOIN timeslot t on
o.timeslot_id = t.id
WHERE r.id = 12 AND (t.start_time BETWEEN '2021-01-10
10:00:00'::timestamp AND '2021-09-19 19:30:00'::timestamp)
       AND (t.end_time BETWEEN '2021-01-10 10:00:00'::timestamp AND
'2021-09-19 19:30:00'::timestamp);
```

4. Give all day schedule for a room

```
SELECT e.title, e.description, t.start_time, t.end_time
FROM room r INNER JOIN event e on r.id = e.room_id INNER JOIN occupies o on
e.id = o.event_id INNER JOIN timeslot t on t.id = o.timeslot_id WHERE r.id = 2
ORDER BY t.start_time;
```

5. Give all day schedule for a user

```
SELECT e.title, e.description, ii.is_available, t.start_time, t.end_time
FROM account a INNER JOIN is_invited ii on a.id = ii.account_id INNER
JOIN event e on e.id = ii.event_id
INNER JOIN occupies o on e.id = o.event_id INNER JOIN timeslot t on t.id
= o.timeslot_id WHERE a.id = 20 ORDER BY t.start_time;
```

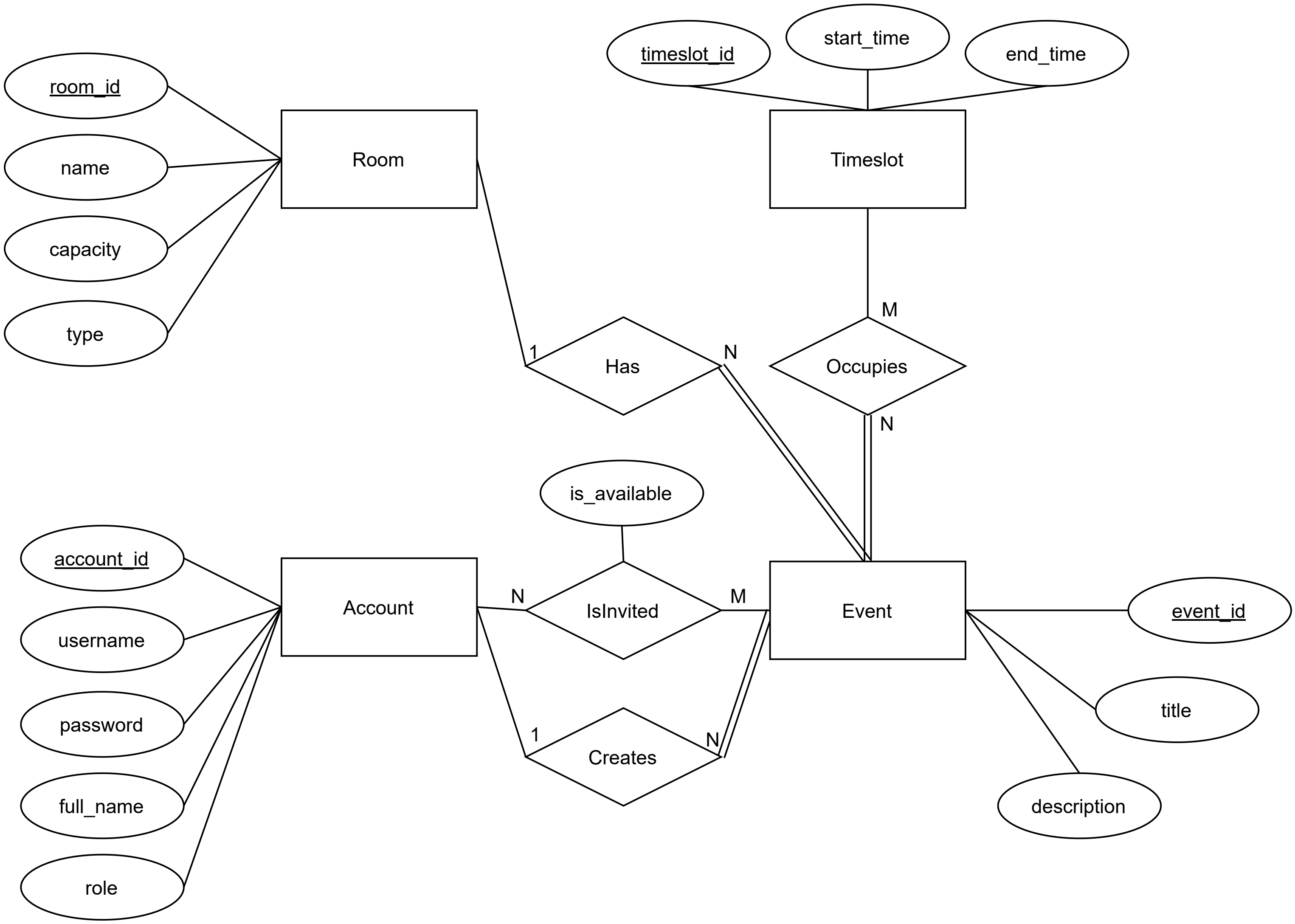
6. Create a meeting with 2+ people in a room

```
--   IMPORTANT: THESE QUERIES NEEDS BACKEND LOGIC TO RUN PROPERLY
--              DO NOT RUN AS IS
--              ONLY WRITTEN AS EXAMPLES

INSERT INTO event (title, description, creator_id, room_id) VALUES
('TITLE', 'DESCRIPTION', '#ID OF USER CREATING THE EVENT', '#ID OF
SELECTED ROOM');

-- BACKEND: GET ID OF CREATED EVENT
```

ER Diagram


room_id
name
capacity
type
Room
timeslot_id
start_time
end_time
Timeslot
M
1
Has
N
Occupies
N
is_available
account_id
username
password
full_name
role
Account
N
IsInvited
M
Event
event_id
title
description
1
Creates
N

Notes

8. Find a time that is free for everyone in the meeting.

Timeslot - Events. Si le restamos a Timeslots todos los Eventos de los users que se quieren invitar para el meeting, debe resultar en una tabla con solo los timeslots disponibles/que no conflijen.

* Considerar usar UNIX time en la entidad evento. Así se puede convertir y obtener la fecha y la duración del evento.

Review con Kristalys (1):

1. Remover meetings

1.1. Hacer mandatory participation (dos lineas de room a eventos).

2. Si no existe el espacio en la base de datos, esta disponible]

3. CanAccess -> A una tabla que tenga todos los timestamps y le restamos todos los records (eventos) que tenga los mismos slots. Va resultar una tabla de todos los slots disponibles.

4. Si hay que buscar tiempo disponible para una semana hay que hacer un query 7 veces (un query para cada 7 dias).

5. Cambiar Attends a Invites. Con un booleano attended.

6. Solamente el room tiene seguridad.

```
INSERT INTO occupies (event_id, timeslot_id) VALUES ('#ID OF CREATED
EVENT', '#ID OF SELECTED TIMESLOT');

-- BACKEND: FOR EACH USER IN INVITED_USERS DO:
INSERT INTO is_invited (account_id, event_id) VALUES ('#ID OF SELECTED
USERS TO INVITE', '#ID OF CREATED EVENT');
```

7. Limit the access to rooms appointment and information according to person’s
authorization (Professor, Student, Department Staff)

```
-- BACKEND: IF (USER.ROLE != 'DEPARTMENT STAFF')
--               # Limit access
```

8. Find a time that is free for everyone in the meeting.

```
SELECT t.start_time, t.end_time FROM timeslot t
WHERE t.id NOT IN (SELECT t.id FROM timeslot t INNER JOIN occupies o on
t.id = o.timeslot_id INNER JOIN event e on e.id = o.event_id
    INNER JOIN is_invited ii on e.id = ii.event_id INNER JOIN account a
on a.id = ii.account_id WHERE a.id IN (4,2,11,15,20)) ORDER BY
t.start_time;
```

9. Allow user to mark time space as “Unavailable”/ “Available” (By default it is all marked as available)

```
-- SET AS AVAILABLE
UPDATE is_invited SET is_available = true WHERE account_id = '#ID OF
USER' AND event_id = '#ID OF EVENT';
-- SET AS UNAVAILABLE
UPDATE is_invited SET is_available = false WHERE account_id = '#ID OF
USER' AND event_id = '#ID OF EVENT';
```

10. Only Department Staff can mark a time space as “Unavailable”/ “Available” for any type of room (By default it is all marked as available)

```
--BACKEND: IF (USER.ROLE == 'DEPARTMENT STAFF')

INSERT INTO event (title, creator_id, room_id) VALUES ('UNAVAILABLE',
'#ID OF CREATOR', '#ID OF ROOM');

-- BACKEND: GET ID OF CREATED EVENT

INSERT INTO occupies (event_id, timeslot_id) VALUES ('#ID OF CREATED
EVENT', '#ID OF SELECTED TIMESLOT');
```

11. User Statistic
a. Most used Room

```
SELECT r.name, count(r.name) as times_used FROM room r INNER JOIN event
e on r.id = e.room_id
GROUP BY r.name ORDER BY times_used DESC LIMIT 1;
```

b. User logged in user has been most booked with

12. Global Statistic
a. Find busiest hours (Find top 5)

```
SELECT start_time, count(start_time) as times_scheduled FROM timeslot t
INNER JOIN occupies o on t.id = o.timeslot_id INNER JOIN event e on e.id
= o.event_id
GROUP BY t.id ORDER BY times_scheduled DESC LIMIT 5;
```

b. Find most booked users (Find top 10)

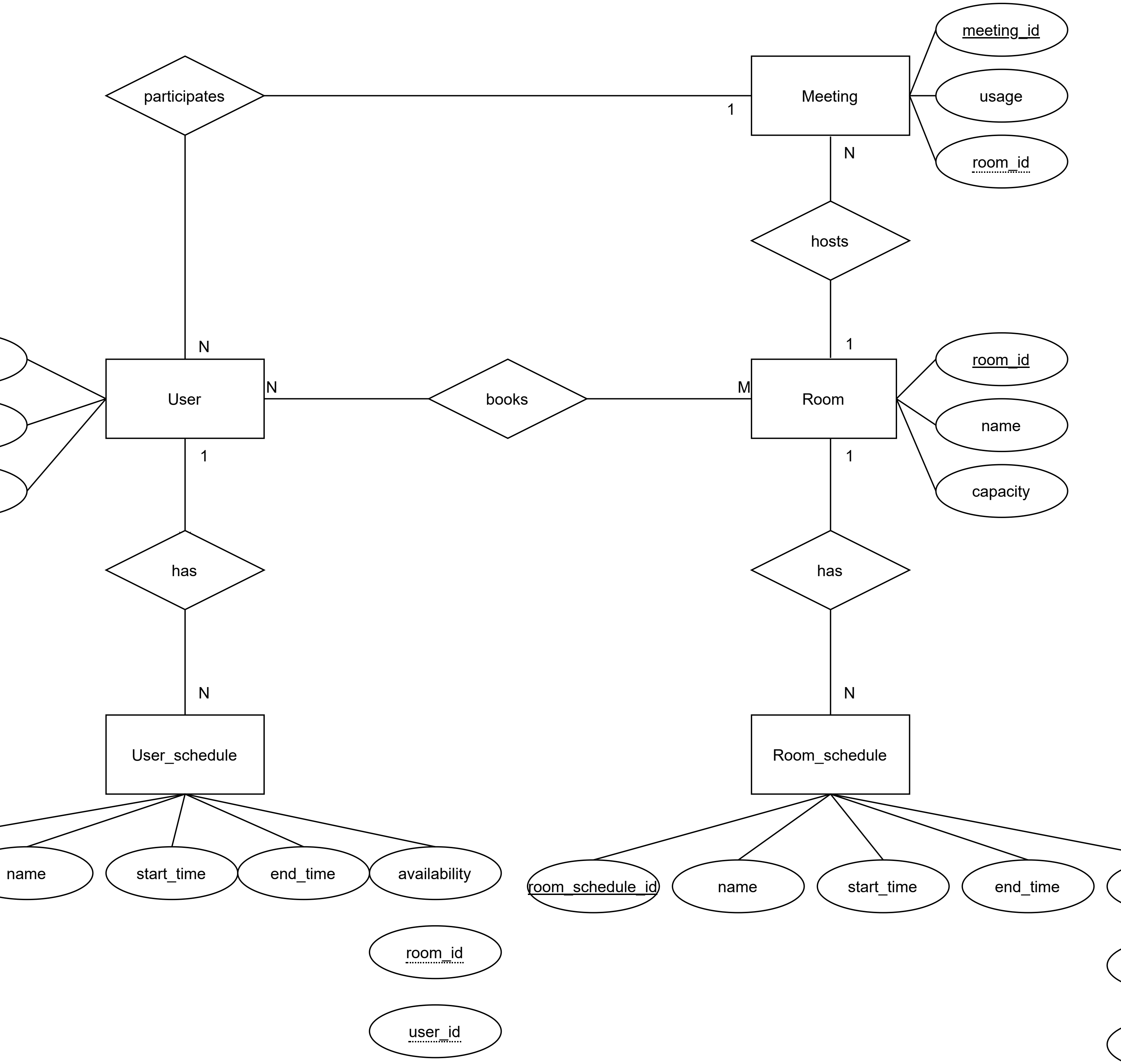
```
SELECT a.full_name, count(a.full_name) as books FROM account a INNER
JOIN is_invited ii on a.id = ii.account_id INNER JOIN event e on e.id =
ii.event_id
GROUP BY a.full_name ORDER BY books DESC LIMIT 10;
```

c. Find most booked rooms (Find top 10)

```
SELECT r.name, count(r.name) as times_booked FROM room r INNER JOIN
```

```
event e on r.id = e.room_id INNER JOIN occupies o on e.id = o.event_id
INNER JOIN timeslot t on t.id = o.timeslot_id
GROUP BY r.name ORDER BY  times_booked DESC LIMIT 10;
```

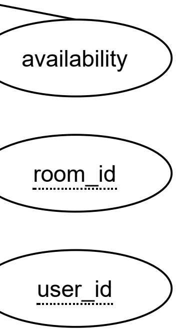
user_id
username
role
user_schedule_id


meeting_id
participates
1
Meeting
usage
N
room_id
hosts
1
N
User
N
books
M
Room
room_id
name
capacity
1
1
has
has
N
N
User_schedule
Room_schedule
name
start_time
end_time
availability
room_schedule_id
name
start_time
end_time
room_id
user_id

Notes

8. Find a time that is free for everyone in the meeting.

Room_schedule - User_schedules. Si le restamos a Room_schedule todos los User_schedule de los users que se quieren invitar para el meeting, debe resultar en una tabla con solo los room_schedules disponibles/que no conflijen.



QUERIES BASED ON UPPER ER DIAGRAM

1. Register a new user

1. Register a new user

```
INSERT INTO user (username, role)
VALUES('admin','DEPARTMENT STAFF');
```

2. Find an available room (lab, classroom, study space, etc.) at a time frame

```
SELECT name FROM room where room_id NOT IN (SELECT * FROM room NATURAL INNER JOIN
event WHERE NOT ((start_time < wanted_start_time AND end_time <= wanted_end_time) OR
(start_time > wanted_start AND start_time >= wanted_end_time));
```

3. Find who appointed a room at a certain time

```
SELECT username FROM user NATURAL INNER JOIN room NATURAL INNER JOIN event WHERE
room_id = # AND  ((start_time <= wanted_start_time && end_time >= wanted_end) OR (start_time >=
wanted_start_time && end_time <= wanted_end_time));
```

4. Give all day schedule for a room

```
SELECT title, start_time, end_time FROM room NATURAL INNER JOIN event WHERE room_id = #
```

5. Give all day schedule for a user

```
SELECT title, start_time, end_time FROM user NATURAL INNER JOIN event WHERE user_id = #
```

6. Create a meeting with 2+ people in a room

???

7. Limit the access to rooms appointment and information according to person's authorization (Professor, Student, Department Staff)

~~Back-end logic depending on user_role~~

8. Find a time that is free for everyone in the meeting.

```
SELECT event_id,
```

9. Allow user to mark time space as "Unavailable"/ "Available" (By default it is all marked as available)

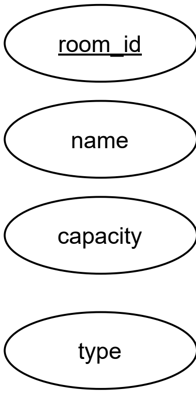
10. Only Department Staff can mark a time space as "Unavailable"/ "Available" for any type of room (By default it is all marked as available)

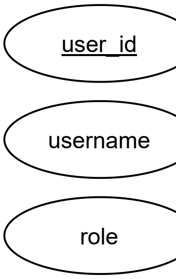
11. User Statistic

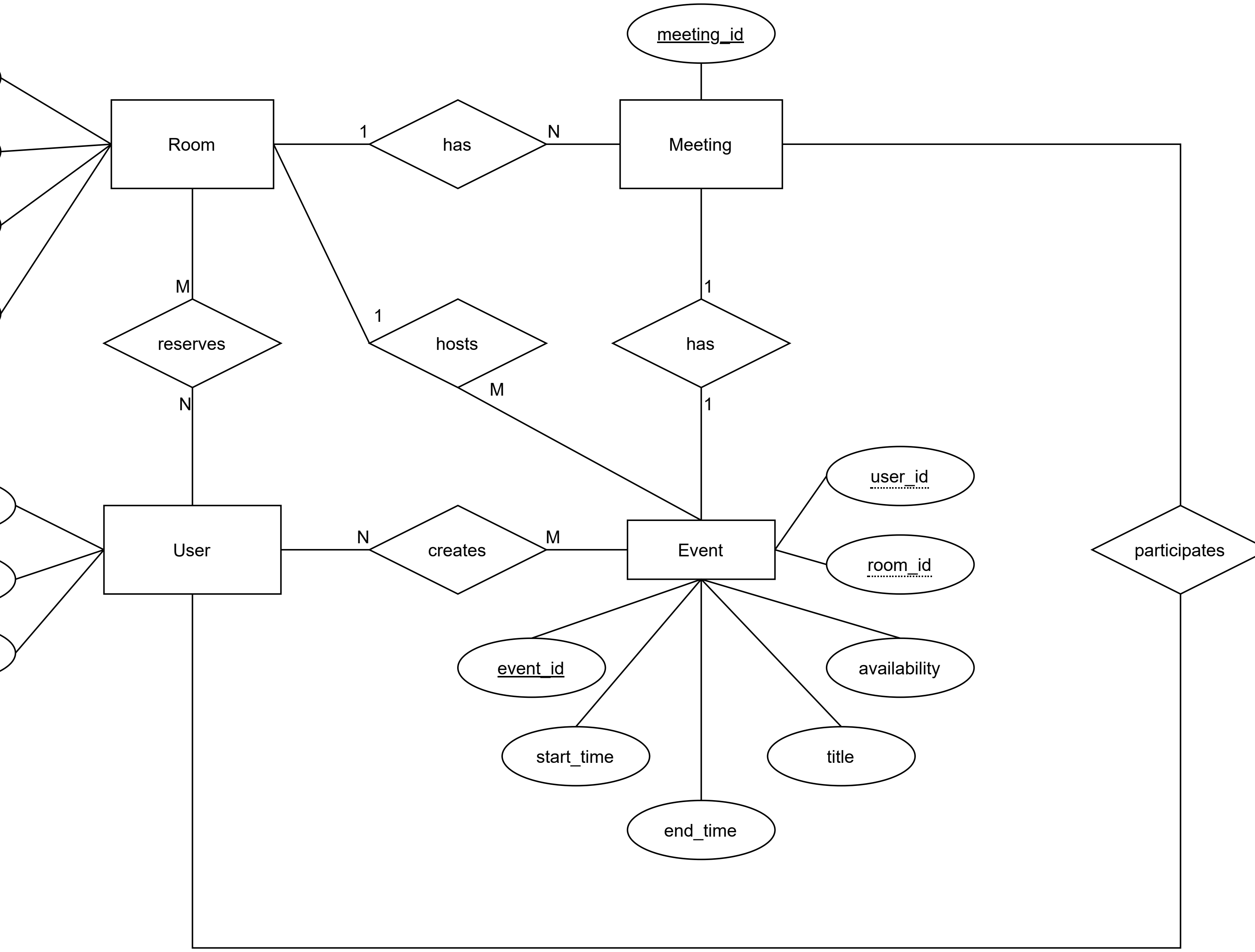
- **a. Most used Room**
- **b. User logged in user has been most booked with**

12. Global Statistic

- **a. Find busiest hours (Find top 5)**
- **b. Find most booked users (Find top 10)**
- **c. Find most booked rooms (Find top 10)**






meeting_id
Room
1
has
N
Meeting
M
reserves
N
1
hosts
M
1
has
1
user_id
User
N
creates
M
Event
room_id
participates
event_id
availability
start_time
title
end_time

ER Diag

ram

Notes

SAMPLE QUERIES

1. Register a new user

```
INSERT INTO user (username, role)
VALUES('admin','DEPARTMENT STAFF');
```

2. Find an available room (lab, classroom, study space, etc.) at a time frame

3. Find who appointed a room at a certain time

4. Give all day schedule for a room

5. Give all day schedule for a user

6. Create a meeting with 2+ people in a room

???

7. Limit the access to rooms appointment and information according to person's authorization (Professor, Student, Department Staff)

8. Find a time that is free for everyone in the meeting.

9. Allow user to mark time space as "Unavailable"/ "Available" (By default it is all marked as available)

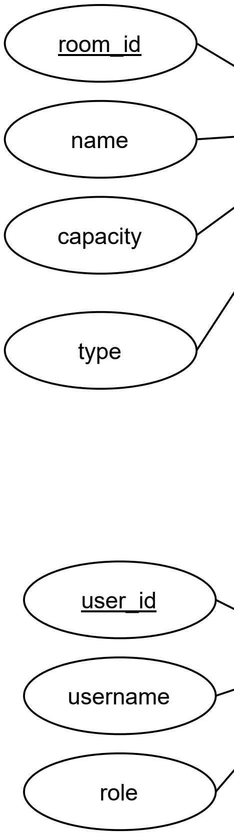
10. Only Department Staff can mark a time space as "Unavailable"/ "Available" for any type of room (By default it is all marked as available)

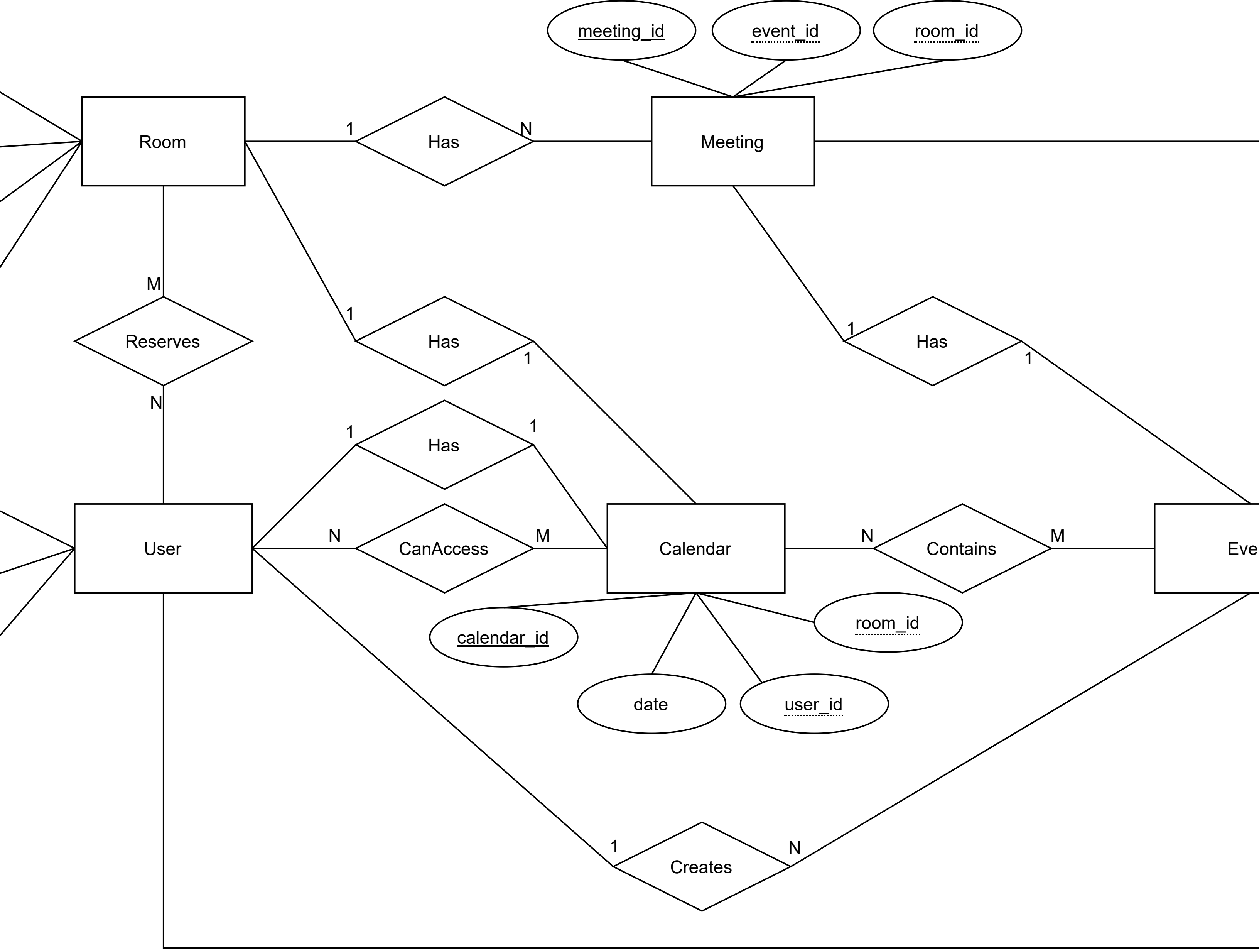
11. User Statistic

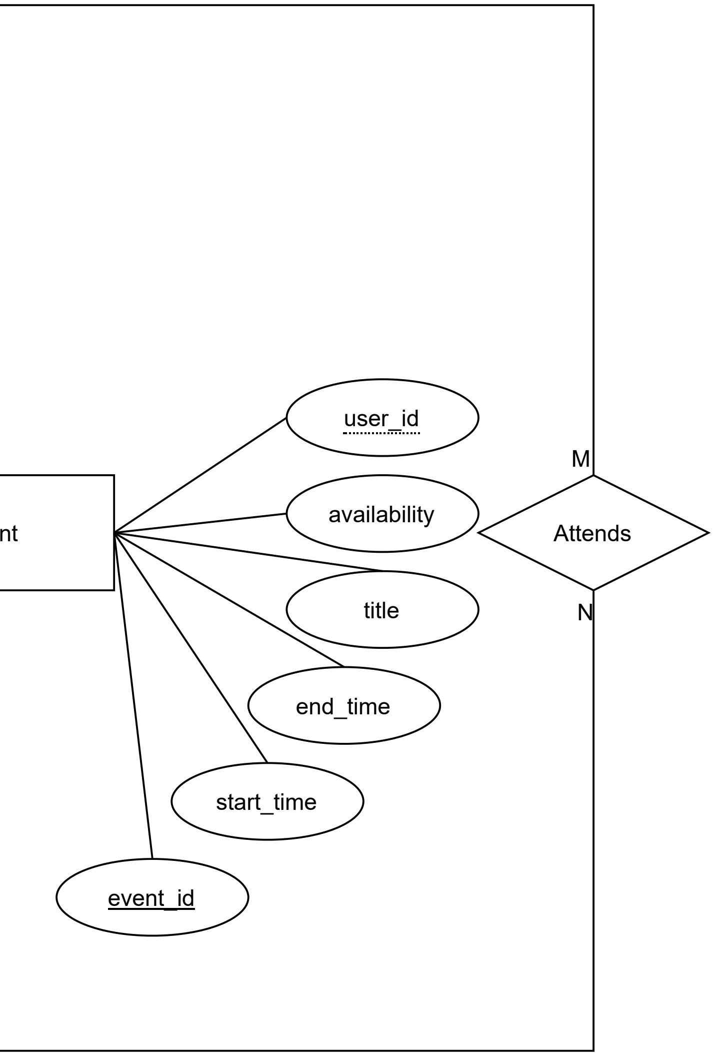
- **a. Most used Room**
- **b. User logged in user has been most booked with**

12. Global Statistic

- **a. Find busiest hours (Find top 5)**
- **b. Find most booked users (Find top 10)**
- **c. Find most booked rooms (Find top 10)**




meeting_id
event_id
room_id
Room
1
Has
N
Meeting
M
Reserves
N
1
Has
1
1
Has
1
1
Has
1
User
N
CanAccess
M
Calendar
N
Contains
M
Eve
calendar_id
room_id
date
user_id
1
Creates
N



8. Find a time that is free for everyone in the meeting.

Room_schedule - User_schedules. Si le restamos a Room_schedule todos los User_schedule de los users que se quieren invitar para el meeting, debe resultar en una tabla con solo los room_schedules disponibles/que no conflijen.

* Considerar usar UNIX time en la entidad evento. Así se puede convertir y obtener la fecha y la duración del evento.

Se puede considerar un meeting un evento?

Review con Kristalys (1):

1. Remover meetings

1.1. Hacer mandatory participation (dos lineas de room a eventos).

2. Si no existe el espacio en la base de datos, esta disponible]

3. CanAccess -> A una tabla que tenga todos los timestamps y le restamos todos los records (eventos) que tenga los mismos slots. Va resultar una tabla de todos los slots disponibles.

4. Si hay que buscar tiempo disponible para una semana hay que hacer un query 7 veces (un query para cada 7 dias).

5. Cambiar Attends a Invites. Con un booleano attended.

6. Solamente el room tiene seguridad.

SAMPLE QUERIES

1. Register a new user

```
INSERT INTO user (username, role)
VALUES('admin','DEPARTMENT STAFF');
```

2. Find an available room (lab, classroom, study space, etc.) at a time frame

3. Find who appointed a room at a certain time

4. Give all day schedule for a room

5. Give all day schedule for a user

6. Create a meeting with 2+ people in a room

???

7. Limit the access to rooms appointment and information according to person's authorization (Professor, Student, Department Staff)

8. Find a time that is free for everyone in the meeting.

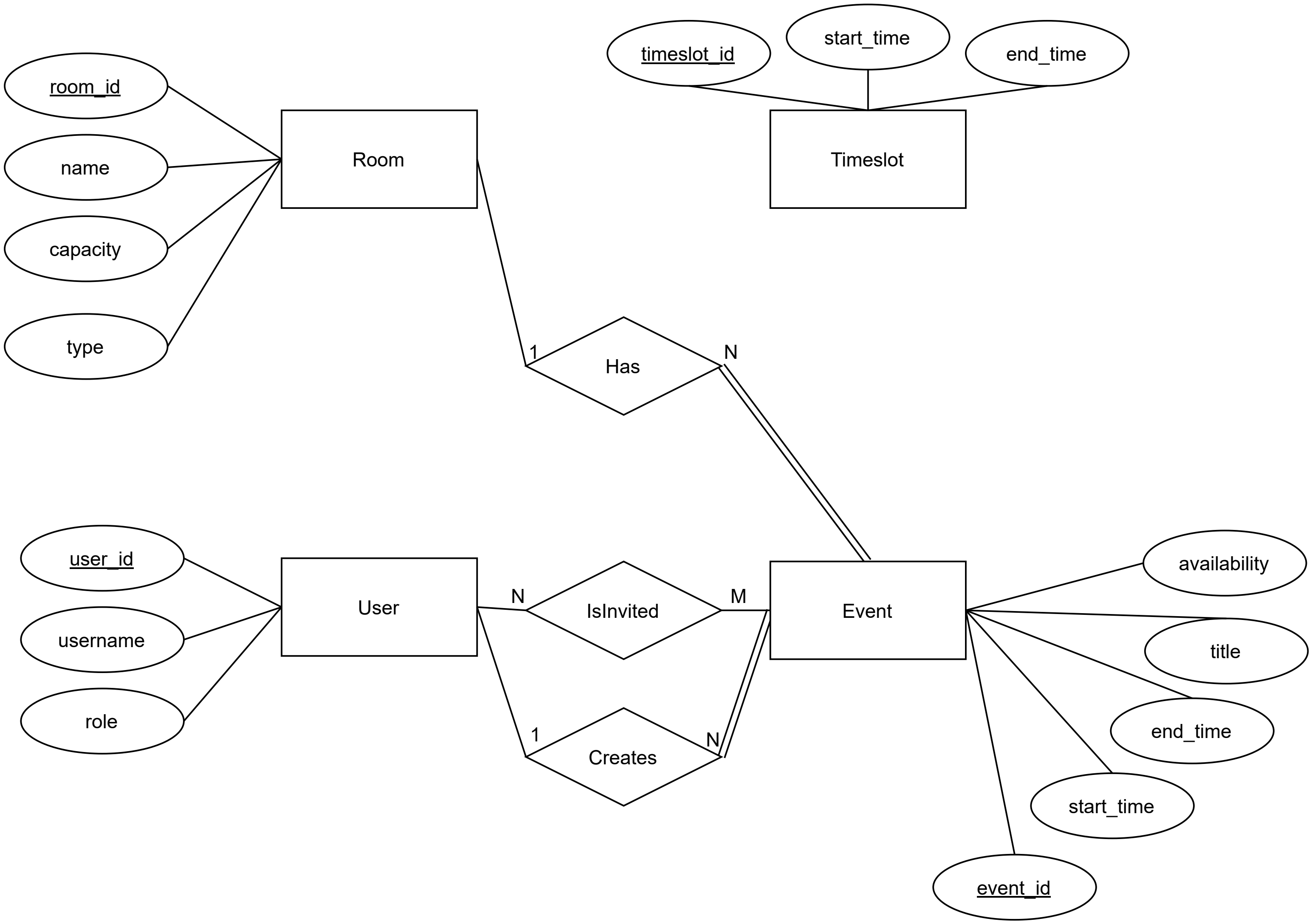
9. Allow user to mark time space as "Unavailable"/ "Available" (By default it is all marked as available)

10. Only Department Staff can mark a time space as "Unavailable"/ "Available" for any type of room (By default it is all marked as available)

11. User Statistic
- **a. Most used Room**
- **b. User logged in user has been most booked with**

12. Global Statistic
- **a. Find busiest hours (Find top 5)**
- **b. Find most booked users (Find top 10)**
- **c. Find most booked rooms (Find top 10)**

ER Diagram


timeslot_id
start_time
end_time
Timeslot
room_id
name
capacity
type
Room
1
Has
N
user_id
username
role
User
N
IsInvited
M
Event
1
Creates
N
availability
title
end_time
start_time
event_id

Notes

8. Find a time that is free for everyone in the meeting.

Timeslot - Events. Si le restamos a Timeslots todos los Eventos de los users que se quieren invitar para el meeting, debe resultar en una tabla con solo los timeslots disponibles/que no conflijen.

* Considerar usar UNIX time en la entidad evento. Así se puede convertir y obtener la fecha y la duración del evento.

Review con Kristalys (1):

1. Remover meetings

1.1. Hacer mandatory participation (dos lineas de room a eventos).

2. Si no existe el espacio en la base de datos, esta disponible]

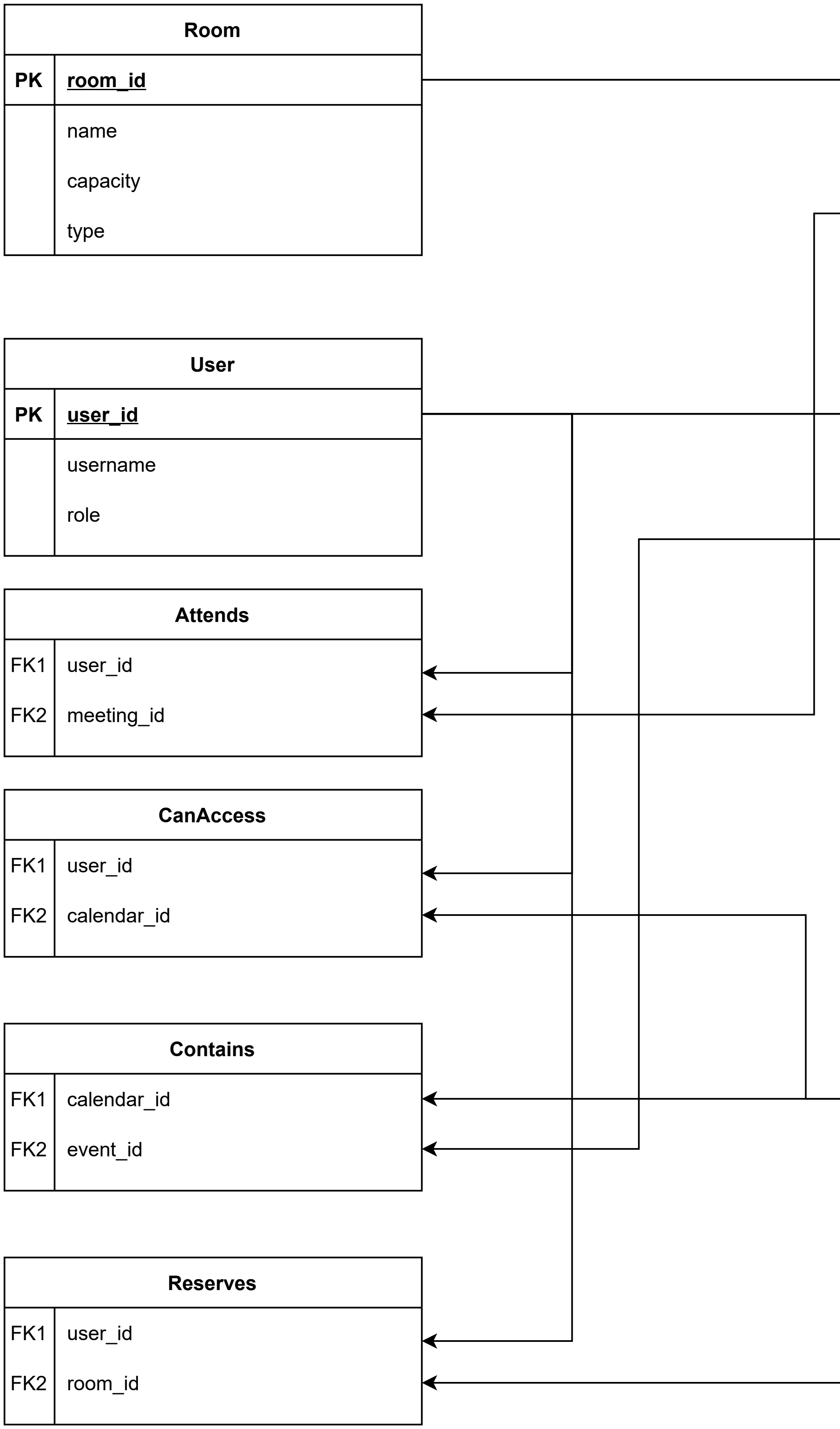
3. CanAccess -> A una tabla que tenga todos los timestamps y le restamos todos los records (eventos) que tenga los mismos slots. Va resultar una tabla de todos los slots disponibles.

4. Si hay que buscar tiempo disponible para una semana hay que hacer un query 7 veces (un query para cada 7 dias).

5. Cambiar Attends a Invites. Con un booleano attended.

6. Solamente el room tiene seguridad.

Table Diagram



Room	
PK	**room_id**
	name
	capacity
	type

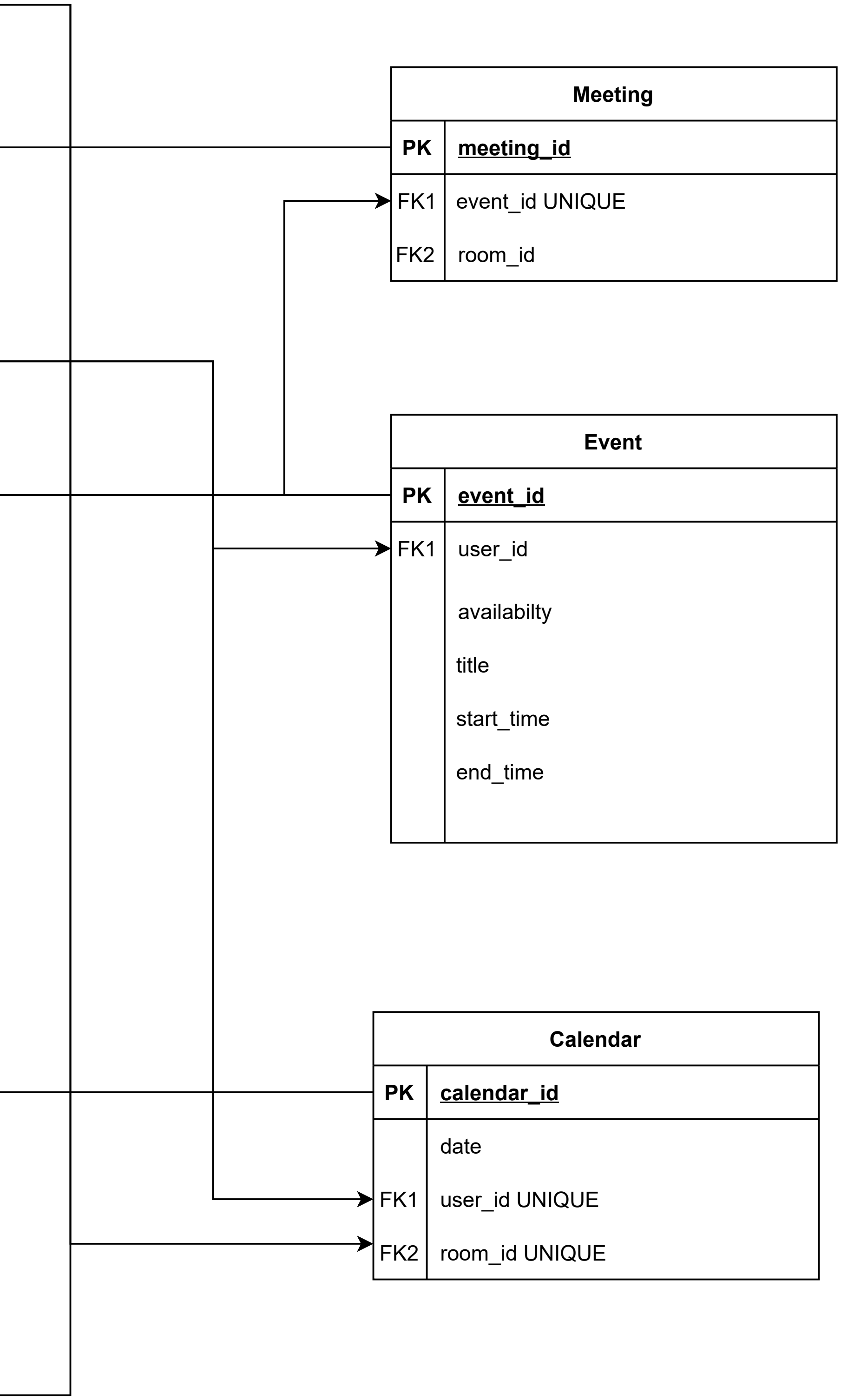
User	
PK	**user_id**
	username
	role

Attends	
FK1	user_id
FK2	meeting_id

CanAccess	
FK1	user_id
FK2	calendar_id

Contains	
FK1	calendar_id
FK2	event_id

Reserves	
FK1	user_id
FK2	room_id



Meeting	
PK	**meeting_id**
FK1	event_id UNIQUE
FK2	room_id

Event	
PK	**event_id**
FK1	user_id
	availabilty
	title
	start_time
	end_time

Calendar	
PK	**calendar_id**
	date
FK1	user_id UNIQUE
FK2	room_id UNIQUE