Rooms Manager Database procedures

This document aims to provide the database manager with the syntax of procedures for displaying or modifying data. Note the syntax to using a procedure: `CALL name ( [ argument ] [, ...] )`.

Before you begin, some syntax-matters

1. Anything string-related has to be enclosed with `''` for psql.
2. PostgreSQL uses the `yyyy-mm-dd` format e.g., `2000-12-31`.

Procedures

Adding a department

```
CALL add_department(did INTEGER, IN dname VARCHAR(50));
```

Removing a department

```
CALL remove_department (target_did INTEGER);
```

Adding an employee

```
CALL add_employee(ename VARCHAR(50), hp_contact VARCHAR(50), kind VARCHAR(7), did INTEGER);
```

This generates a unique eid for the employee which follows an increasing sequence starting from 1, and also a unique email which concatenates their initials to their eid, followed by the company's email.

For instance, assume that the phone number and did are valid.

Then `CALL add_employee('Abraham Benedict Cumberbatch Donkey', '12345678', 'Senior', 69);` would result in an email `ABCD2@gsnail.com`, if the employee has eid 2.

Removing an employee

```
CALL remove_employee(IN eid INTEGER, resigned_date DATE);
```

This call simply tags a date in the resigned_date attribute of the employee with given eid.

Adding a room

```
CALL add_room(room_name VARCHAR(50),floor_num INTEGER, room_num INTEGER, did INTEGER)
```

Changing room capacity

```
CALL change_capacity (manager_eid INTEGER, floornum INTEGER , roomnum INTEGER , capacity INTEGER , effective_date DATE)
```

Checking non-compliance

```
SELECT * FROM non_compliance(sDate DATE, eDate DATE)
```

Declaring health

```
CALL declare_health (eid INTEGER, date DATE, temperature DECIMAL);
```

Contact Tracing

Returns list of employees in close contact with some eid. eid represents the employee with a fever. Note that this function should be implemented as a trigger with health declaration. `CALL SELECT * FROM contact_tracing(eid INTEGER)`

View future meeting

```
SELECT * FROM view_future_meeting(sDate DATE, eid INTEGER)
```

Booking a room

```
CALL book_room (floor integer, room integer, date date, start_hr integer, end_hr integer, booker_eid integer)
```

Unbooking a room

`CALL unbook_room (floor integer, room integer, date date, start_hr integer, end_hr integer, booker_eid integer)`

View manager report

`SELECT * FROM view_manager_report (start_date DATE, manager_eid INTEGER)`

Approve meeting

`CALL approve_meeting (floor_no INTEGER, room_no INTEGER, date DATE, start_hour INTEGER, end_hour INTEGER, eid INTEGER)`

Join meeting

`CALL join_meeting (floor_no INTEGER, room_no INTEGER, date DATE, start_hour INTEGER, end_hour INTEGER, eid INTEGER)`

Leave meeting

`CALL leave_meeting (floor_no INTEGER, room_no INTEGER, date DATE, start_hour INTEGER, end_hour INTEGER, eid INTEGER)`

Searching for a room

SELECT * FROM `search_room (min_cap int, meeting_date date, start_hr int, end_hr int)`

Functions that will activate with triggers

Remove_future_meetings_on_fever()

Deletes all sessions of employee with fever, as well as their close contacts, adds them to blacklist

assign_email()

Assigns an email to an employee (Initials+Eid+@gsnail.com)

Check_sessions_blacklist()

Checks if employee in blacklist. If so, delete meeting