CSI 2132 Course Project Deliverable B - Emma Power - 300294808

Video Lookup

Requirement	Timestamp
1	0:08
2	0:45
3	1:00
4	1:30
5	2:03
6	3:40
7	4:43
8	5:20
9	6:35

Notes on Deliverable Requirements

- As SQL queries are dependent on input, the SQL code for functionalities is contained within the client/web app, as untying it from the web structure would cause it to stop making sense
- All code is included in the zip file that contains this document. The main code for the backend is in Server/index.js and the main code for the frontend is in Client/src/app.js
- Due to the insane compression requirements, the video may not open in some players. The video is tested to work with the latest version of VLC media player.

Technologies

For this application, I used a PostgreSQL database, and pgadmin4 for a frontend for database management. For the backend, I used Node.JS, that interfaced with the database using the "npm pg" extension. For the frontend, i used React, which interfaced with the database using HTML GET queries.

To run this application

- Start your PostgreSQL instance, and initialize it according to the DDL given later in this document.
- From the Server directory, run "npm start" to start the backend.
- From the Client directory, run "npm run dev" to start the frontend server. Navigate to the address given in console in your web browser.

DDLs

- The following are all the DDLs used in this simple application. Running all of these sequentially should give the desired output.

```
CREATE TABLE IF NOT EXISTS public."Booked"
(
    "Room_ID" bigint NOT NULL,
    "Booking_ID" bigint NOT NULL,
    CONSTRAINT "Booked_pkey" PRIMARY KEY ("Room_ID", "Booking_ID")
)

CREATE TABLE IF NOT EXISTS public."Booking"
(
    "Start_Date" date,
    "End_Date" date,
    "Booking_ID" bigint NOT NULL,
    CONSTRAINT "Booking_pkey" PRIMARY KEY ("Booking_ID")
)

CREATE TABLE IF NOT EXISTS public."Chain"
(
    "Name" character varying(64) COLLATE pg_catalog."default" NOT NULL,
    "Address" character varying(64) COLLATE pg_catalog."default",
    "Email" character varying(64) COLLATE pg_catalog."default",
    "Phone_Number" character varying(10) COLLATE pg_catalog."default",
    CONSTRAINT "Chain_pkey" PRIMARY KEY ("Name")
)

CREATE TABLE IF NOT EXISTS public."Creates"
(
    "Cust_ID" bigint NOT NULL,
    "Booking_ID" bigint NOT NULL,
    CONSTRAINT "Creates_pkey" PRIMARY KEY ("Cust_ID", "Booking_ID")
)

CREATE TABLE IF NOT EXISTS public."Customer"
(
    "Cust_ID" bigint NOT NULL,
    "Name" character varying(64) COLLATE pg_catalog."default",
    "Address" character varying(64) COLLATE pg_catalog."default",
    "Register_Date" date,
    CONSTRAINT "Customer_pkey" PRIMARY KEY ("Cust_ID")
)

CREATE TABLE IF NOT EXISTS public."Employee"
```

```
(
    "SIN" integer NOT NULL,
    "Name" character varying(64) COLLATE pg_catalog."default",
    "Address" character varying(64) COLLATE pg_catalog."default",
    "Position" character varying(64) COLLATE pg_catalog."default",
    CONSTRAINT "Employee_pkey" PRIMARY KEY ("SIN")
)

CREATE TABLE IF NOT EXISTS public."Employs"
(
    "Hotel_ID" bigint NOT NULL,
    "SIN" integer NOT NULL,
    CONSTRAINT "Employs_pkey" PRIMARY KEY ("Hotel_ID", "SIN")
)

CREATE TABLE IF NOT EXISTS public."Has"
(
    "Hotel_ID" bigint NOT NULL,
    "Room_ID" bigint NOT NULL,
    CONSTRAINT "Has_pkey" PRIMARY KEY ("Hotel_ID", "Room_ID")
)

CREATE TABLE IF NOT EXISTS public."Hotel"
(
    "Hotel_ID" bigint NOT NULL,
    "Address" character varying(64) COLLATE pg_catalog."default",
    "Email" character varying(64) COLLATE pg_catalog."default",
    "Phone_Number" character varying(10) COLLATE pg_catalog."default",
    "Stars" bigint,
    "Room_Count" bigint,
    "Area" character varying(64) COLLATE pg_catalog."default",
    CONSTRAINT "Hotel_pkey" PRIMARY KEY ("Hotel_ID")
)

CREATE TABLE IF NOT EXISTS public."Occupies"
(
    "Room_ID" bigint NOT NULL,
    "Cust_ID" bigint NOT NULL,
    CONSTRAINT "Occupies_pkey" PRIMARY KEY ("Room_ID", "Cust_ID")
)

CREATE TABLE IF NOT EXISTS public."Owns"
(
    "Name" character varying COLLATE pg_catalog."default" NOT NULL,
    "Hotel_ID" bigint NOT NULL,
    CONSTRAINT "Owns_pkey" PRIMARY KEY ("Name", "Hotel_ID")
)

CREATE TABLE IF NOT EXISTS public."Rents"
(
    "Start_Date" date NOT NULL,
    "End_Date" date NOT NULL,
    "Price" numeric,
    "Card_No" bigint,
    "Booking_ID" bigint,
    CONSTRAINT "Rents_pkey" PRIMARY KEY ("Start_Date", "End_Date")
)

CREATE TABLE IF NOT EXISTS public."Room"
(
    "Room_ID" bigint NOT NULL,
    "Amenities" character varying(64) COLLATE pg_catalog."default",
    "Price" numeric,
    "Capacity" integer,
    "View" character varying(64) COLLATE pg_catalog."default",
    "Problems" character varying(64) COLLATE pg_catalog."default",
    CONSTRAINT "Room_pkey" PRIMARY KEY ("Room_ID")
)


CREATE OR REPLACE FUNCTION delete_rooms()
RETURNS TRIGGER AS $$
BEGIN
        DELETE FROM "Room"
        USING "Has"
        WHERE "Room"."Room_ID" = "Has"."Room_ID"
        AND "Has"."Hotel_ID" = NEW."Hotel_ID";
        RETURN NEW;
END;
$$ LANGUAGE plpgsql;

CREATE OR REPLACE FUNCTION delete_has()
RETURNS TRIGGER AS $$
BEGIN
        DELETE FROM "Has"
        WHERE "Has"."Hotel_ID" = OLD."Hotel_ID";
        RETURN NEW;
END;
```

```
$$ LANGUAGE plpgsql;

CREATE OR REPLACE FUNCTION delete_owns()
RETURNS TRIGGER AS $$
BEGIN
        DELETE FROM "Owns"
        WHERE "Owns"."Name" = OLD."Name";
        RETURN NEW;
END;
$$ LANGUAGE plpgsql;

CREATE OR REPLACE FUNCTION delete_hotels()
RETURNS TRIGGER AS $$
BEGIN
        DELETE FROM "Hotel"
        USING "Owns"
        WHERE "Hotel"."Hotel_ID" = "Owns"."Hotel_ID"
        AND "Owns"."Name" = NEW."Name";
        RETURN NEW;
END;
$$ LANGUAGE plpgsql;


CREATE OR REPLACE VIEW public."CityRoom"
 AS
 WITH a AS (
         SELECT "Room"."Room_ID"
           FROM "Room",
            "Booked",
            "Booking"
          WHERE "Room"."Room_ID" = "Booked"."Room_ID" AND "Booked"."Booking_ID" =
"Booking"."Booking_ID" AND "Booking"."Start_Date" <= CURRENT_DATE AND
"Booking"."End_Date" >= CURRENT_DATE
        ), b AS (
         SELECT "Room"."Room_ID" AS room_id
           FROM "Room"
             LEFT JOIN a ON "Room"."Room_ID" = a."Room_ID"
          WHERE a."Room_ID" IS NULL
        ), c AS (
         SELECT b.room_id,
            "Hotel"."Hotel_ID",
            "Hotel"."Area"
           FROM b,
            "Has",
           "Hotel"
         WHERE b.room_id = "Has"."Room_ID" AND "Hotel"."Hotel_ID" = "Has"."Hotel_ID"
        )
 SELECT count(room_id) AS count,
    "Area"
   FROM c
  GROUP BY "Area";

CREATE OR REPLACE VIEW public."HotelRooms"
 AS
 WITH a AS (
        SELECT "Room"."Room_ID"
          FROM "Room",
           "Booked",
           "Booking"
         WHERE "Room"."Room_ID" = "Booked"."Room_ID" AND "Booked"."Booking_ID" =
"Booking"."Booking_ID" AND "Booking"."Start_Date" <= CURRENT_DATE AND
"Booking"."End_Date" >= CURRENT_DATE
        ), b AS (
        SELECT "Room"."Room_ID" AS room_id
          FROM "Room"
            LEFT JOIN a ON "Room"."Room_ID" = a."Room_ID"
         WHERE a."Room_ID" IS NULL
        ), c AS (
        SELECT b.room_id,
           "Hotel"."Hotel_ID" AS hotel_id,
           "Hotel"."Area"
          FROM b,
           "Has",
           "Hotel"
         WHERE b.room_id = "Has"."Room_ID" AND "Hotel"."Hotel_ID" = "Has"."Hotel_ID"
        )
 SELECT count(room_id) AS count,
    hotel_id
   FROM c
  GROUP BY hotel_id
  ORDER BY hotel_id;


CREATE OR REPLACE VIEW public."SearchRoomView"
 AS
 WITH a AS (
        SELECT "Room"."Room_ID" AS room_id,
           "Hotel"."Hotel_ID" AS hotel_id,
           "Hotel"."Area",
           "Room"."Price",
           "Room"."Capacity",
           "Hotel"."Stars"
          FROM "Room",
           "Has",
           "Hotel"
         WHERE "Room"."Room_ID" = "Has"."Room_ID" AND "Hotel"."Hotel_ID" =
"Has"."Hotel_ID"
        ), b AS (
        SELECT a.room_id,
           a.hotel_id,
           a."Area",
           a."Price",
           a."Capacity",
           a."Stars",
           "Owns"."Name",
           "Owns"."Hotel_ID",
           "Chain"."Name",
           "Chain"."Address",
           "Chain"."Email",
           "Chain"."Phone_Number",
           "Chain"."Name" AS name
          FROM a,
           "Owns",
           "Chain"
         WHERE a.hotel_id = "Owns"."Hotel_ID" AND "Owns"."Name"::text = "Chain"."Name"::text
        )
 SELECT b.room_id,
    b.hotel_id,
    b."Area",
    b."Price",
    b."Capacity",
    b."Stars",
    b.name,
    "HotelRooms".count AS "Hotel Rooms"
   FROM b b(room_id, hotel_id, "Area", "Price", "Capacity", "Stars", "Name", "Hotel_ID",
"Name_1", "Address", "Email", "Phone_Number", name)
     LEFT JOIN "HotelRooms" ON "HotelRooms".hotel_id = b.hotel_id;
```

```
CREATE OR REPLACE TRIGGER hotel_delete_trigger
   BEFORE DELETE
   ON public."Chain"
   FOR EACH ROW
   EXECUTE FUNCTION public.delete_hotels();


CREATE OR REPLACE TRIGGER owns_delete_trigger
   BEFORE DELETE
   ON public."Chain"
   FOR EACH ROW
   EXECUTE FUNCTION public.delete_owns();

CREATE OR REPLACE TRIGGER has_delete_trigger
   BEFORE DELETE
   ON public."Hotel"
   FOR EACH ROW
   EXECUTE FUNCTION public.delete_has();

CREATE OR REPLACE TRIGGER room_delete_trigger
   BEFORE DELETE
   ON public."Hotel"
   FOR EACH ROW
   EXECUTE FUNCTION public.delete_rooms();


CREATE INDEX idx_hotel_id ON "Hotel"("Hotel_ID");
CREATE INDEX idx_room_id ON "Room"("Room_ID");
CREATE INDEX idx_booking_date ON "Booking"("Start_Date", "End_Date");
```