Project 1: PayRental - Property Renting Marketplace

Akshay Neema (2016CS10323), Riya Singh (2016CS50388), Siddhant Shingi (2016CS10310)

Due date: March 1, 2020, 11:55pm IST

1 Description

1.1 Project Description

Airbnb is an online-based marketing company that connects people looking for accommodation (Airbnb guests) to people looking to rent their properties (Airbnb hosts) on a short-term or long-term basis. As a rental ecosystem, Airbnb generates tons of data including but not limited to the density of rentals across regions (cities and neighborhoods), price variations across rentals, host-guest interactions in the form of reviews, and so forth.

Our idea is to create PayRental, a property renting marketplace which allows the user to match with the hosts who have put up their properties on rent. We will perform an exploratory data analysis of the dataset to understand the rental landscape through various visualisations. The complete system is as follows :-

- Allows users to create a *secured user account* which opens the PayRental homescreen. Users can choose from the property listings based on metrics like price and location. Once, chosen there is an option for booking the property and a calendar which let's the users know the availability of the listings.
- Allows the hosts to create a *secured host account* which opens the PayRental homescreen. Hosts are given the option to check out their listings - the bookings, the availability etc. and add or remove listings.

1.2 Entities and Attributes

Entity	Attributes
Property	id, name, property type, root type, price, city, latitude, longitude, min nights, max nights
Reviews	id, date
Calender	date, available
Users	user id, name, password
Hosts	host id, name, password
Bookings	id, check-in date, check-out date, adults, children

Table 1: List of Entities and Attributes

Attribute	Permitted Values
City	Chicago, New York, Los Angeles
Room type	Entire home/apt, Private room, Shared room, Hotel room
Property type	Aparthotel, Apartment, Barn, Bed and breakfast, Boat, Boutique hotel, Bungalow, Bus, Cabin, Camper/RV, Campsite, Casa particular (Cuba), Castle, Cave, Chalet, Condohotel, Condominium, Cottage, Dome house, Dorm, Earth house, Farm stay, Guesthouse, Guest suite, Hostel, Hotel, House, Houseboat, Hut, Igloo, In-law, Island, Lighthouse, Loft, Minsu (Taiwan), Nature lodge, Other, Pension (South Korea), Plane, Resort, Serviced apartment, Tent, Timeshare, Tiny house, Tipi, Townhouse, Train, Treehouse, Vacation home, Villa
Dates	Jan, 2019 - Dec, 2019

Table 2: Structure of downloaded raw data

2 Data Source and Statistics

2.1 Data Source

The data is taken from officially available Airbnb dataset curated in late 2019. The Airbnb data consists mainly of three tables :-
1. **Listings** : This table contains the information of all the property listings on the site of Airbnb, with 106 attributes some of which are *price (double precision)*, *listing type, host name, latitude, review per month, scores* and *ratings*.
2. **Reviews** : This table consists of the reviews given by the guests and has the following attributes - *date, listing id, reviewer id* and *comment (textual)*.
3. **Calendar** : Provide the detail of booking by each listing. It has the attributes - *date, listing id, available* and *price*.

Table Name	Number of Attributes	Key Attributes
Listings	106	id, host id, price, review per month, is superhost, score
Reviews	6	listing id, id, date, reviewer id, reviewer name, comments
Calendar	7	listing id, date, available, price, min nights, max nights

Table 3: Structure of raw data

2.2 Data Retrieval and Cleanup

Airbnb online data has property, review and calendar data for 120 cities all over the world. Out of these 120 cities, 28 cities are located in United States. We choose 3 most dense cities (w.r.t the Airbnb properties) in the United States due to extremely large size restrictions especially on strings like comment. No scraping was required as the data is available in .csv format.
Mainly the cleanup was done manually by discarding the redundant or unnecessary columns). Further all the '|' characters in string type columns were replaced by ',' and '|' was set as delimeter in the csv files. After the cleanup, data was inserted to the table using bulk copy.
After the cleanup the attributes and the tables are :-

1. **Table: payrental**
   city_data , state_data , id , listing_url , scrape_id , last_scraped , name , summary , space , experiences_offered, neighborhood_overview , notes , transit , access , interaction , house_rules , thumbnail_url , medium_url , picture_url , xl_picture_url , host_id , host_name , host_since , host_location , host_about , host_response_time, host_response_rate , host_acceptance_rate , host_is_superhost , host_listings_count , host_identity_verified, street , neighbourhood , neighbourhood_cleansed , neighbourhood_group_cleansed , city , state , zipcode , market , smart_location , country_code , country , latitude , longitude , is_location_exact , property_type , room_type , accommodates , bathrooms , bedrooms , beds , amenities , square_feet , price, weekly_price , monthly_price , security_deposit , cleaning_fee , guests_included

, extra_people , minimum_nights , maximum_nights , has_availability , availability_30 , availability_60 , availability_90 , availability_365 and so on.

2. **Table: reviews**
   city , state , listing_id , id , date , reviewer_id
3. **Table: calender**
   city , state , listing_id , date , available , price , adjusted_price , minimum_nights , maximum_nights
4. **Table: users**
   user_id , name , username , password
5. **Table: hosts**
   host_id , host_name , host_username , password
6. **Table: bookings**
   booking_id , property_id , host_id , user_id , check_in_date , check_out_date , number_adults , number_children

2.3 Statistics

Table Name	Table Size	No.of Tuples	Time to Load
Calender	266 MB	3078751	11116.739 ms
Reviews	217 MB	2974950	26839.228 ms
Payrental	170 MB	98259	51242.337 ms
Users	132 MB	2423928	3764.284 ms
Hosts	3952 kB	65356	176.990 ms

Table 4: Data Statistics

We have chosen three cities for a close exploratory data analysis from the Airbnb data. The cities were chosen on the basis of popularity and usage of Airbnb in and with a reasonable number of listings. The cities are **New York, Los Angeles and Chicago** and we will mostly be analysing the given data of these cities.

City	Number of Listings
New York	50036
Los Angeles	39690
Chicago	8533

Table 5: Number of Total Property Listings in NY, LA and Chicago

City	Number of Reviews
New York	1255302
Los Angeles	1368989
Chicago	350659

Table 6: Number of Total Reviews in NY, LA and Chicago

3 Functionality

3.1 User's view of System

We have given two options to the guest - **USER** or **HOST**. This gives us the flexibility to change the users, hosts and bookings making all the tables we have dynamic. The properties are explained as follows :

1. **User :**

   (a) **User Signup** : For any user, if it is the first time the user is using PayRental, the user needs to sign up with a unique userid and password with character length greater than or equal to 6.

   (b) **User Login** : Once the user is signed up, the user data is entered to the **users** table and the user can now log in to the PayRental portal with his/her credentials.

   (c) **Home** : The first page on signing in the portal shows a home page with options of the users to search for properties, book etc. and description of the features of the site and a logo of PayRental.

   (d) **Property Listings** : Property listings are a core feature of PayRental. In this tab, the users can filter the properties according to their requirements. This runs queries on the **PayRental** table and gives the solution to what the user wishes to have in his/her property. The queries in this part can take variable time depending on what the user demands. The filters are as follows :
   i. *City* :
   This filter gives us the option to view all the property listings of **Chicago, NY and LA**. We had earlier added each state and city given in the data but retrieval and clean up of all three was not feasible. Hence, we had to remove the State, City filter and restrict ourselves to these three cities.
   ii. *Property* type :
   This filter lets us choose the kind of property we are looking for i.e. apartment, house, villa, etc.
   iii. *Room type* :
   This filter chooses the kind of rooms we are looking for i.e. private room, entire apartment etc.
   iv. *Price* :
   This filter is perhaps the most used while looking for property listings and has a range 0-1500 dollars per night. It is implemented using a drag to change bar with the default minimum value set to 0.
   v. *Distance* :
   This filter used while looking for property listings which are at a specific distance from the centre of the city. It has a range 0-200 kms and like price, is implemented using a drag to change bar with the default minimum value set to 0.
   vi. *Rating* :
   This filter is used to get the highest (preferably) rated listings from the range 0-100.
   Once you choose your desired property, choosing the **Book** button takes you to the booking page. You get finer details which will be explained in the next section.

   (e) **Booking :** The booking in user login is where the user books a property. It shows user the amenities, the finer property details from table **payrental**, and all the relevant fill ups, pricing and availability to finally book the property.

   (f) **My Bookings :** This shows all the current and past bookings of the user, showing the dates, properties and hosts of all bookings done with this id.

   (g) **Statistics :** This shows all the queries on the data of the tables. It has queries like *cleanest city*, *city with maximum listings* and visual statistical representations of these queries.

   (h) **Reset Password** : This is to reset the password as and when required for the user login. This again enforces the constraint of 6 or more characters.

   (i) **Logout** : This is for the user to log out and go back to the login/signup page. You can shift to either user or host login now.

   (j) **About Us** : Some details about the creators - trivia, github IDs and what we do! This is how we have structured the User's view of system as user.

2. **Hosts :**

   (a) **Host Signup** : For any host, if it is the first time the host is using PayRental, the user needs to sign up with a unique host userid and password with character length greater than or equal to 6.

   (b) **Host Login** : Once the host is signed up, the host data is entered to the **hosts** table and the host can now log in to the PayRental portal with his/her credentials.

   (c) **Add Property** : For any user to add a property as a host, they need to fill a form so that it can be added to the **hosts** table and subsequently change the required tables.

   (d) **My Bookings** : This shows a list of all the property listings by this host and respective booking histories of each of them. Booking history consists of the date of booking, pricing etc.

(e) **Reset Password** : This is to reset the password as and when required for the host login. This again enforces the constraint of 6 or more characters.

(f) **Logout** : This is for the host to log out and go back to the login/signup page. You can shift to either user or host login now.

(g) **About Us** : Some details about the creators - trivia, github IDs and what we do! This is how we have structured the User's view of system as host.

3.2 System View

1. **Trigger**
We implemented the following trigger to ensure the validity of any new booking by checking the availability of the property from check-in to check-out dates. And if the booking is valid then we update the availability of the property.

```
CREATE OR REPLACE FUNCTION check_availability() RETURNS trigger AS
$BODY$
BEGIN
   if (select count(*) from calender
        where listing_id=new.property_id and date>=new.check_in_date
        and date<=new.check_out_date and available='f') > 0 then
    DELETE FROM bookings where bookings.booking_id = new.booking_id;
   ELSE
    update calender set available='f'
        where listing_id=new.property_id and date>=new.check_in_date
        and date<=new.check_out_date;
   END IF;
   RETURN NEW;
END;
$BODY$ language 'plpgsql';


CREATE TRIGGER book
AFTER INSERT
ON bookings
FOR EACH ROW
EXECUTE PROCEDURE check_availability();
```

2. **Constraints**
We have set primary key for every table. They are listed below in table : primary key format.

   - Payrental: listing_id
   - Reviews: review_id
   - Calendar: listing_id, Date
   - user: user_id
   - hosts: host_id
   - bookings: booking_id

   There are foreign key constraints on some tables.

   - host_id column in payrental is mapped to host_id column in hosts table with foreign key contraint
   - host_id, user_id and property_id columns in bookings table is mapped to host_id column in hosts table, user_id column in users table, listing_id column in payrental table respectively with foreign key contraint

3. **Sequence**
We have several sequences which keep track of the next booking_id, user_id, host_id, property_id and review_id that is to be handed out whenever there is a new new entry in their respective tables. Using

```
SELECT nextval (' sequence name ');
```

at the PHP front-end, we extract a new unique value that is next in the sequence for each insert query. This ensures uniqueness in the ids for each table.

4. **Access Privileges**
We implemented 2 access modes: user host. For accessing the application in each mode respective authentications are required. For each of these 2 modes, we created the following access privileges:

   (a) User Mode: Any query through this mode do not affect the payrental or host table. It can only affect user table(creating new account or updating existing account) or booking table/calendar table(while booking a property).

   (b) Host Mode: Any query through this mode do not affect the user or booking table. It can only affect host table(creating new account or updating existing account) or payrental table/calendar table(while add a new property).

3.3 Queries and Runtime

3.3.1 List of Queries

1. **Account**:

   (a) Signup:

```
 INSERT INTO users (username, password) VALUES ($1, $2)
OR
INSERT INTO hosts (host_name, host_username, password) VALUES ($1, $2, $3)
```

   (b) Login:

```
 SELECT id, username, password FROM users WHERE username = $1
OR
SELECT host_id, host_username, password FROM hosts WHERE host_username = $1
```

   (c) Reset Password:

```
UPDATE users SET password = $2 WHERE id = $1
OR
UPDATE hosts SET password = $2 WHERE host_id = $1
```

2. **Property Listings**:

   (a) Select Property (for users) :

```
 SELECT payrental.id, property_type, room_type, cast(price as integer), payrental.city,
number_of_reviews as rcount, cast(review_scores_rating as integer) as rating,
round(distance(latitude::decimal, longitude::decimal, lat, lng)::numeric, 2) as distance,
picture_url FROM payrental, cityinfo WHERE room_type =  $1 AND property_type = $2
AND cityinfo.city = $3 AND payrental.price <= $4 AND distance<$5 ORDER BY $6
```

   (b) Add Property (for hosts) :

```
 INSERT INTO payrental * VALUES ($<complete tuple>)
```

3. **Booking**:

```
 INSERT INTO bookings(*) VALUES (next(booking_sequence), property_id, host_id, user_id,
check_in_date, check_out_date, number_adults, number_children)
```

4. **My Bookings**

   (a) User Bookings:

```
 SELECT * FROM bookings WHERE user_id = $1
```

   (b) Host Bookings:

```
SELECT * FROM bookings WHERE host_id = $1
```

5. **Statistics**: Using these queries, we wish to do an **exploratory data analysis of Airbnb for the cities of NY, LA and Chicago**. We have plotted a lot of graphs using this data to show direct comparison between the three cities closely regarding the property dealings and run these queries to infer information from them. These can be found under **statistics** section in our user view. Since, we have tried to be as exhaustive in the writing the queries and form inference, we are writing only a select few queries, some graphs (if available) and query timings along with these inference i.e. **understanding the real estate scene closely in these cities.**
aim to answer atleast these questions with our data :

   (a) How does the supply and demand of the properties vary in these cities with respect to each other.

   (b) How popular is a city? What does it depend on and which city wins in cleanliness, listing ratings, pricing, better hosts etc.

   (c) What type of places people prefer? What type of rooms they prefer? How does the pricing of the same varies in each city?

   (d) Seasonality in Demand and Prices (with respect to days) i.e. how the demand and prices varies with weekends, holidays etc.

   (e) Many more insights like correlation between prices and type of listings, effect of review counts, and touch on topics like what it takes to be a superhost?

   Queries : All the data is for the cities of NY, Chicago and LA.

   - Number of hosts in each city:

```
SELECT city_data, COUNT(DISTINCT(host_id))
FROM payrental GROUP BY city_data ORDER BY city_data;
```

   - Top 5 hosts with the highest listings :

```
CREATE VIEW listingsperhost as
SELECT hostid, count(*) as countlistingsperhost
FROM payrental
GROUP by hostid
ORDER by count(*) asc;

SELECT listingsperhost.hostid, hostname, countlistingsperhost as highestlistings
FROM listingsperhost, hosts
where listingsperhost.hostid = hosts.hostid
ORDER by countlistingsperhost desc
LIMIT 5;


Time: 168.504 ms
```

   - City with highest listings :

```
SELECT citydata, count(*) as listingcount
FROM payrental
GROUP BY citydata;

Time: 79.763 ms
```

   - Host with highest reviews :

```
SELECT id, hostid, sum(cast(numberofreviews as integer)) as numreviews
FROM payrental
GROUP BY id, hostid;
```

```
Time: 197.978 ms

SELECT id, hostname, citydata, sum(cast(numberofreviews as integer)) as numreviews
FROM payrental
GROUP BY id, citydata, hostname
ORDER BY sum(cast(numberofreviews as integer)) desc
LIMIT 50;

Time: 189.294 ms
```

- City with highest reviews :

```
CREATE VIEW maxreviewsoflistings as
SELECT citydata, sum(cast(numberofreviews as integer))
FROM payrental
GROUP BY citydata;

Time: 79.763 ms
```

- High Demand Listings :

```
SELECT *
FROM maxreviewlistings
WHERE numreviews $>$ 500

Time: 187.532 ms
```

- Less Demand Listings :

```
SELECT *
FROM maxreviewlistings
WHERE numreviews $<$ 50

Time : 191.780 ms
```

- Most Expensive City :

```
SELECT citydata, avg(price) as averageprice
FROM payrental
GROUP by citydata
ORDER by avg(price) desc ;

Time: 117.070 ms
```

- Highest average price of listing by hosts :

```
SELECT hostid, hostname, avg(price) as  averageprice
FROM payrental
GROUP by hostid, hostname
ORDER by avg(price) desc;

Time: 224.977 ms
```

- City with maximum number of superhosts :

```
CREATE VIEW superhosts as
SELECT hostid, hostname, city
```

```
FROM payrental
WHERE hostissuperhost = 't';

Time: 3.470 ms

SELECT city, count(*) as numsuperhosts
FROM superhosts
GROUP by city
ORDER by count(*) desc;

Time: 102.398 ms
```

- Famous cities - cities with maximum reviews per month :

```
SELECT avg(cast(reviewspermonth as double precision)) as averagereviewspermonth, city
FROM payrental
GROUP by citydata
ORDER by avg(cast(reviewspermonth as double precision))) desc ;

Time: 74.485 ms
```

- Clean cities :

```
SELECT avg(cast(reviewscorescleanliness as double precision)) as cleanlinessscores, city
FROM payrental
GROUP by citydata
ORDER by avg(cast(reviewscorescleanliness as double precision)) desc ;

Time: 66.381 ms
```

- Property Types :

```
SELECT propertytype, hostissuperhost, count(*) as propertcount
FROM payrental
GROUP by propertytype, hostissuperhost
ORDER by count(*) desc ;

Time: 113.555 ms
```

- Room Types :

```
SELECT roomtype, hostissuperhost, count(*) as propertcount
FROM payrental
GROUP by roomtype, hostissuperhost
ORDER by count(*) desc ;

Time: 57.898 ms
```

Since the bookings are our own, we use reviews as a measure of number of bookings assuming half bookings result in reviews from general trend in the actual Airbnb data. Some inferences are :

(a) In terms of listings, Chicago > New York > Los Angeles. Hence the supply follows this trend in comparison of these three cities.

(b) In terms of reviews, that is the demand, Los Angeles > New York > Chicago

(c) Almost 50 hosts, mostly in LA, have number of reviews greater than 500. There are almost 0 listings with less than 50 reviews.

(d) Average price of listings are the highest in LA at 226 dollars followed by Chicago (180 dollars) and New York (158 dollars). LA has the highest number of superhosts while New York wins in cleanliness.

(e) Apartment wins as the preferred type of listings followed by House and Apartments. Room type is generally entire home/apartment followed by private rooms.

3.4 ER Diagram

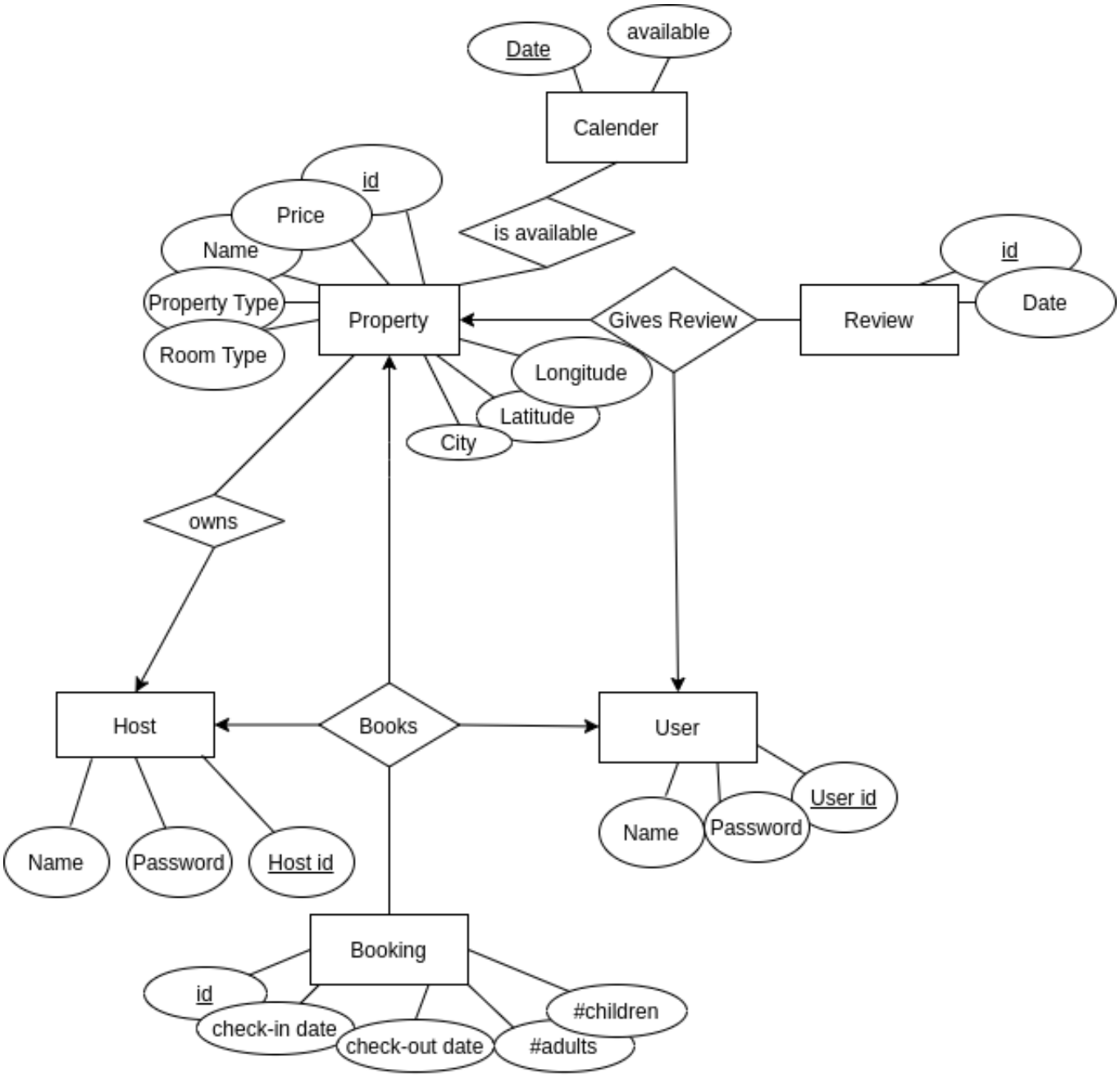

Figure 1: ER Diagram for PayRental Service