CAR RENTAL SYSTEM

Project Detail:

- **Team: Database Geeks**

- **Team members:**

  - **Samar Pratap Singh UB ID: samarpra ,**

  - **Darryl Vas Prabhu UB ID: dvasprab,**

  - **Aditya Singh UB ID : singh24**

Problem Statement:

Due to the advancement in technology, businessmen look for new ways to incorporate it within to increase their profit. While this might be true, a business cannot sustain without true demand. There are many areas where a demand gives rise to a solution. One such case is in online booking system. Time is of utmost importance and to avoid wasting time, online systems give you the benefit of performing various tasks with a click of a few buttons.

Due to the increase in the cost of automobiles, people generally prefer to rent a car rather than buy one if they wish to travel for trips or go grocery shopping or just to head out to visit family/friends. The business owners charge a small fee for renting the car while the customer is benefited by having the luxury of driving around in the car as his own. However, in order to book a car, customers most often need to visit the car rental location to check if the cars are available, and details such as brand, mileage etc . All this is time consuming and hence an automated system which shows all the available cars, with its details for booking is made available through this project.

The target users of the database would be students, elders possessing a driving license.

The car rental system is administered and managed by the car rental companies.

Excel file systems are spreadsheet data which can become quite large and difficult to manage owing to duplicate data. A database on the other hand would be easier to manage since the database contains relations between data tables through which we can fetch data, update data in a quick and efficient manner.

Target User:
- The users:
  Anyone who wants to rent a car i.e. Students, tourists, people in general.
- The administrator:
  Car rental companies who are using this rental automobile system.
- Real-life scenario description:

SUNY Buffalo is a world renowned university and hence students come from all over the globe. Students find it hard to travel places since they are new to the city. Time is of the essence due to busy University schedules and waiting in line to physically check cars for rent is a tedious task. In order to help students and the like, this vehicle management system is implemented.

E/R Diagram:
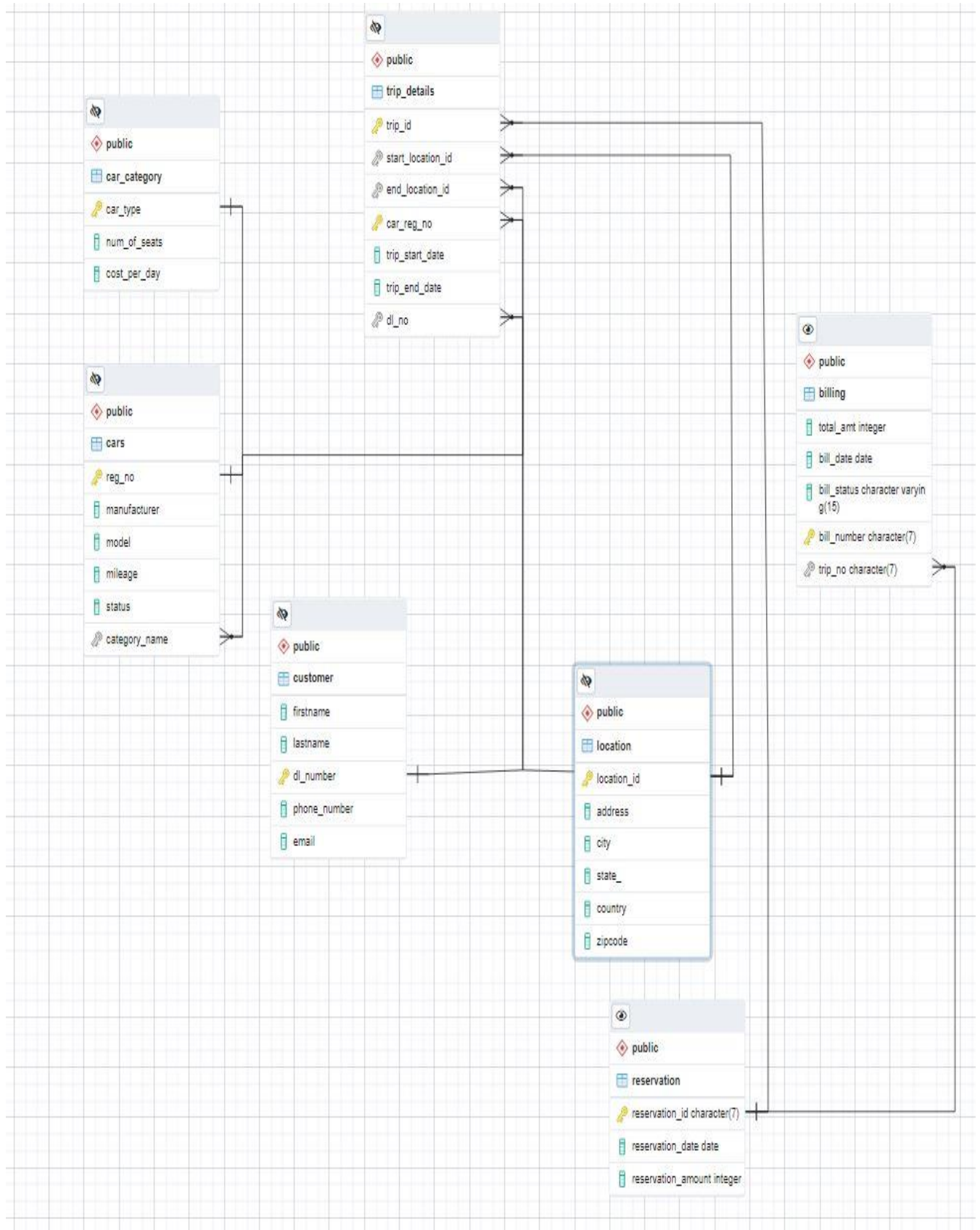
Database Implementation:

Entities/Tables and Attributes :

1. Customer:

The person who makes a car reservation through the car rental system will be the customer. Details like the client's firstname varchar, lastname varchar, dl_number varchar, and phone number varchar, email varchar will be stored in the customer table where Primary key to uniquely identify each customer is his/her driving license number dl_number.

2. Cars:

The system's cars will be listed in the car table. Each car will belong to a certain category and include the car's details such as car registration number denoted by reg_no varchar, the manufacturer varchar (15), the model varchar(10), Mileage numeric(8,1), Status varchar(10)i.e. if the car is Available for booking or if it's Not Available, and category_name varchar(20) denoting which class the car belongs to be it Sedan or SUV. The primary key is car registration number reg_no since it'unique for each car and the foreign key is category_name which references table Car_Category.

3. Car_Category:

Each type of car has a category which mainly denotes if the capacity is a 4 seater or a 6 seater. The cost is based on a per day usage which is based on the car category. car_type varchar(20), num_of_seats int, cost_per_day numeric(5,2) are the attributes of the table. The primary key (car_type) here is car_type.

4. Location:

The location table indicates the various locations of the car rental business. Location will have characteristics such as Location_ID varchar (6), Address varchar (50), city varchar (20), state_ char(4) , country char (3), and zipcode integer. The primary key is the (Location_ID) which is assumed to be unique for a given car rental location.

5. Trip_Details:

The Trip_details includes the details of the journey the car has taken while it was rented. Customer has the option to pick up the vehicle from a certain place and drop it off at the same or a different location. Trip_details will have attributes such as start_location_id varchar (6), end_location_id char(6), trip_id char (7), car_reg_no varchar(10), dl_no varchar(20). trip_id + car_reg_no) together make up the primary key since no 2 tuples will have the same trip_id and car_reg_no. The tables related to multiple other tables through foreign keys . Foreign keys start_location_id and end_location_id REFERENCES locations(Location_id). Foreign key (car_reg_no) REFERENCES Cars(RegNo), Foreign key (dl_no) REFERENCES Customer(dl_number). trip_start_date and trip_end_date are used to indicate how many days the customer rented the car and calculate the associated rental cost.

6. Reservation:

The entity named reservations will keep track of each customer's reservation. The properties of a reservation include the reservation_id char(20) which is same as the trip_id of the trip_details table, reservation_date date to indicate when the reservation was made, and reservation_amt int which denotes how much amount was paid upfront to reserve the car. The primary key here is (reservation_id).

7. Billing:

A charge for the specific booking will be produced when a client returns a car. The following billing details are included: total_amt int, bill_date date, bill_status varchar(15), bill_number char(7), trip_no char (7). The primary key here is the (bill_no), and FOREIGN KEY (trip_no) REFERENCES reservation (reservation_id).

Relation Between Tables:

1. Cars to Car_Category:

Each vehicle belongs to a certain automotive category. When a customer chooses a vehicle, the cost per day is derived from the vehicle category to which the chosen vehicle belongs. "Belongs to" is the name of the relation.

2. Cars to Location:

The customer will pick up or drop off the vehicle in a certain area. The automobile can be picked up or returned by the customer at that specific spot. So, a site will have automobiles present.

3. Reservation to Billing:

A vehicle charge will be produced for each reservation after the consumer returns. In the event that a reservation is canceled, no bill will be connected to the reservation. Gives is the name of the connection.

4. Trips_Details to Location:

Rental car pick-up/drop-off locations are available for customers. Drop off place is the name of the relationship.

5. Customer to Cars to Reservation:

The customer will pick the rental vehicle. Therefore, the consumer will be involved with both the automobile and the reservation. These three things have a ternary relationship, and that relationship's name is "Rents."

Records in the table:

The data in the tables was generated using a Python strings and faker package. Certain attribute values were fetched from sites such as Kaggle , data.io world which are available in the reference section.

Car table data:

Screenshot of car table data insertion in the database:

```
9    SELECT * FROM cars;
```

Data output Messages Notifications

	reg_no [PK] character varying (7)	manufacturer character varying (15)	model character varying (10)	mileage numeric (8,1)	status character varying (15)	category_name character varying (15)
1	6HPQ627	chevrolet	door	7880.0	Available	Premium SUV
2	4LOA572	ford	door	9752.6	Available	SUV
3	8NWD510	jeep	mpv	1203.8	Available	SUV
4	3IIN511	ford	fusion	3417.3	Available	SUV
5	6TMI968	ford	f-150	1905.8	Available	SUV
6	3TAQ294	ford	f-150	3674.2	Available	Premium SUV
7	2BJK268	nissan	pathfinder	3413.8	Not Available	Luxury Sedan
8	7MVR143	nissan	door	1773.8	Available	SUV
9	7XIT412	chevrolet	sonic	6607.9	Available	SUV
10	2WZA507	gmc	door	14033.9	Available	Luxury Sedan
11	5GVN722	chrysler	van	3222.4	Available	Premium SUV
12	5ZRH792	gmc	door	2744.2	Available	Luxury Sedan
13	5QDQ933	ford	srw	13215.0	Available	Luxury Sedan
14	6OAL188	dodge	journey	3700.9	Not Available	Luxury Sedan
15	0ADH828	nissan	maxima	3495.1	Not Available	Sedan
16	1IUX359	nissan	versa	2761.1	Available	Premium SUV
17	7NZE317	dodge	van	10722.4	Not Available	SUV
18	3DFG429	dodge	charger	3509.8	Not Available	Luxury Sedan
19	7PSN267	ford	expedition	1493.9	Available	Premium SUV

Total rows: 200 of 200   Query complete 00:00:00.087

Location table data insertion:

Screenshot of Location table data insertion in the database:

```
9   SELECT * FROM location;
```

Data output   Messages   Notifications

	location_id [PK] character (6)	address character varying (50)	city character varying (20)	state_ character (4)	country character (3)	zipcode integer
1	LOC114	3005 Cranberry Hwy Rt 6 28	Wareham	MA	US	2538
2	LOC115	333 Main Street	Tewksbury	MA	US	1876
3	LOC122	200 Sunrise Mall	Massapequa	NY	US	11758
4	LOC123	101 South Beltline Highway	Mobile	AL	US	36606
5	LOC135	7155 State Rt 12 S	Lowville	NY	US	13367
6	LOC137	750 Academy Drive	Bessemer	AL	US	35022
7	LOC177	330 Sutton Rd	Huntsville	AL	US	35763
8	LOC224	11610 Memorial Pkwy South	Huntsville	AL	US	35803
9	LOC225	2800 Spring Avn SW	Decatur	AL	US	35603
10	LOC226	279 Troy Road	East Greenbush	NY	US	12061
11	LOC232	41301 US Hwy 280	Sylacauga	AL	US	35150
12	LOC242	6438 Basile Rowe	East Syracuse	NY	US	13057
13	LOC251	4975 Transit Rd	Lancaster	NY	US	14086
14	LOC254	495 Flatbush Ave	Hartford	CT	US	6106
15	LOC255	1600 Montclair Rd	Birmingham	AL	US	35210
16	LOC265	10040 County Road 48	Fairhope	AL	US	36533
17	LOC272	150 Barnum Avenue Cutoff	Stratford	CT	US	6614
18	LOC308	425 Route 31	Macedon	NY	US	14502
19	LOC309	41 Anawana Lake Road	Monticello	NY	US	12701

Total rows: 72 of 72   Query complete 00:00:00.060

Customer table data insertion:

Screenshot of Customer table data insertion in the database:

Query   Query History

```
1   SELECT * FROM Customer;
```

Data output   Messages   Notifications

	firstname character varying (15)	lastname character varying (15)	dl_number [PK] character varying (10)	phone_number character varying (15)	email character varying (30)
1	Jeffrey	Jackson	DLN102715	+1-796-703-5309	Jeffrey.Jackson@hot...
2	Margaret	Brown	DLN612207	+1-179-556-9317	Margaret.Brown@hot...
3	Jennifer	Smith	DLN864335	+1-766-634-7076	Jennifer.Smith@yaho...
4	Alicia	Smith	DLN131656	+1-886-385-7130	Alicia.Smith@outlook...
5	Willie	Bennett	DLN552172	+1-192-101-3402	Willie.Bennett@hotm...
6	Joseph	Anderson	DLN197518	+1-325-336-6313	Joseph.Anderson@ya...
7	Ruben	Morris	DLN493578	+1-334-525-1889	Ruben.Morris@yahoo...
8	Jessica	Burch	DLN607915	+1-471-779-8485	Jessica.Burch@yaho...
9	Steven	Smith	DLN239850	+1-154-984-3728	Steven.Smith@gmail...
10	Richard	Sims	DLN422547	+1-739-463-7605	Richard.Sims@outloo...
11	Kyle	Hinton	DLN773906	+1-524-862-8677	Kyle.Hinton@yahoo.c...
12	Sherry	Briggs	DLN721647	+1-765-107-7626	Sherry.Briggs@yahoo....
13	Shelly	Wright	DLN207961	+1-999-583-4017	Shelly.Wright@yahoo....
14	Jillian	Duncan	DLN740185	+1-631-243-8997	Jillian.Duncan@hotm...
15	Aaron	Wilson	DLN818308	+1-705-385-3181	Aaron.Wilson@outloo...
16	Kimberly	Hopkins	DLN810211	+1-696-442-9448	Kimberly.Hopkins@ya...
17	Mark	Garcia	DLN834696	+1-802-212-5475	Mark.Garcia@gmail.c...
18	Robert	Randall	DLN827846	+1-367-766-6176	Robert.Randall@yaho...
19	Timothy	Larson	DLN739272	+1-938-718-4141	Timothy.Larson@gma...
20	Rebecca	Jordan	DLN396237	+1-281-696-6540	Rebecca.Jordan@yah...
21	Martin	Petty	DLN702221	+1-405-809-1168	Martin.Petty@yahoo....
22	Karla	Watkins	DLN865436	+1-252-398-5094	Karla.Watkins@yahoo...
23	Eric	Carroll	DLN668670	+1-115-233-6380	Eric.Carroll@outlook....

Total rows: 200 of 200   Query complete 00:00:00.054

SQL Queries:

1.SELECT Clause:

selecting firstname, lastname, dl_number, trip_start_date from customer and trip_details table.

```
11  -- All customers who took trip:
12  SELECT firstname,lastname,dl_number,trip_start_date FROM Customer AS C, trip_details as td
13  WHERE C.dl_number = td.dl_no;
```

Data output   Messages   Notifications

	firstname character varying (15)	lastname character varying (15)	dl_number character varying (10)	trip_start_date timestamp without time zone
1	Daniel	Pitts	DLN893219	2022-10-04 09:51:57
2	Mark	Garcia	DLN834696	2022-09-20 12:11:51
3	Jacqueline	Smith	DLN353162	2022-09-20 06:06:49
4	Jaime	Smith	DLN901423	2022-09-25 15:12:49
5	Kathryn	Coleman	DLN199714	2022-09-09 06:05:40
6	Joseph	Anderson	DLN197518	2022-09-08 11:05:27
7	Christopher	Massey	DLN475618	2022-10-09 07:21:37
8	Robert	Randall	DLN827846	2022-10-05 09:44:21
9	Kenneth	Page	DLN964814	2022-09-24 14:19:17
10	Helen	Smith	DLN118462	2022-08-20 19:05:45
11	Samuel	Nichols	DLN601769	2022-10-16 22:17:53

Total rows: 50 of 50   Query complete 00:00:00.098

2. GROUP BY Clause:

Group by clause is used for getting status and category_name from the Cars table.

```
15  -- GROUP BY
16  SELECT category_name ,status,COUNT(*) FROM Cars
17  GROUP BY status,category_name ;
```

Data output   Messages   Notifications

	category_name character varying (15) 🔒	status character varying (15) 🔒	count bigint 🔒
1	Premium SUV	Not Available	10
2	Luxury Sedan	Available	40
3	SUV	Available	39
4	Sedan	Not Available	9
5	Premium SUV	Available	37
6	SUV	Not Available	18
7	Sedan	Available	34
8	Luxury Sedan	Not Available	13

Total rows: 8 of 8   Query complete 00:00:00.048

3. SUBQUERIES:

Subqueries are executed by using Select within From clause to get model, manufacturer and status details of cars with mileage < 1000;

```
8   -- Subquery:
9   SELECT model,manufacturer,status FROM (SELECT * FROM cars WHERE mileage
10                                          <1000) as h ;
11
12
13
14
```

Data output   Messages   Notifications

	model character varying (10)	manufacturer character varying (15)	status character varying (15)
1	doors	ford	Available
2	mustang	ford	Not Available
3	f-150	ford	Available
4	f-150	ford	Available
5	fusion	ford	Not Available
6	drw	ford	Not Available
7	cargo	nissan	Available
8	ranger	ford	Not Available
9	1500	chevrolet	Not Available

Total rows: 9 of 9   Query complete 00:00:00.051

4. JOIN:

Join using where clause to get the details of the firstname, lastname , customer preferred car manufacturer from customer, trip_details and cars tables.

```
3   -- Customer preferred car manufacturers
4   SELECT firstname, lastname, manufacturer FROM customer,trip_details,cars
5   WHERE trip_details.dl_no = customer.dl_number
6   AND trip_details.car_reg_no = cars.reg_no;
7
```

Data output   Messages   Notifications

	firstname character varying (15)	lastname character varying (15)	manufacturer character varying (15)
1	Daniel	Pitts	dodge
2	Mark	Garcia	ford
3	Jacqueline	Smith	dodge
4	Jaime	Smith	ford
5	Kathryn	Coleman	dodge
6	Joseph	Anderson	ford
7	Christopher	Massey	nissan
8	Robert	Randall	nissan
9	Kenneth	Page	chevrolet
10	Helen	Smith	dodge
11	Samuel	Nichols	dodge

Total rows: 50 of 50   Query complete 00:00:00.058

References

[1] faker.providers — Faker 15.1.1 documentation

[2] ellisbrown/us-apple-stores-by-states | Workspace | data.world

[3] US Cars Dataset | Kaggle

[4] Compare Rental Car Sizes and Classes | Enterprise Rent-A-Car

[5] List of Real USA Addresses | Kaggle

[6] pandas documentation — pandas 1.5.0 documentation (pydata.org)