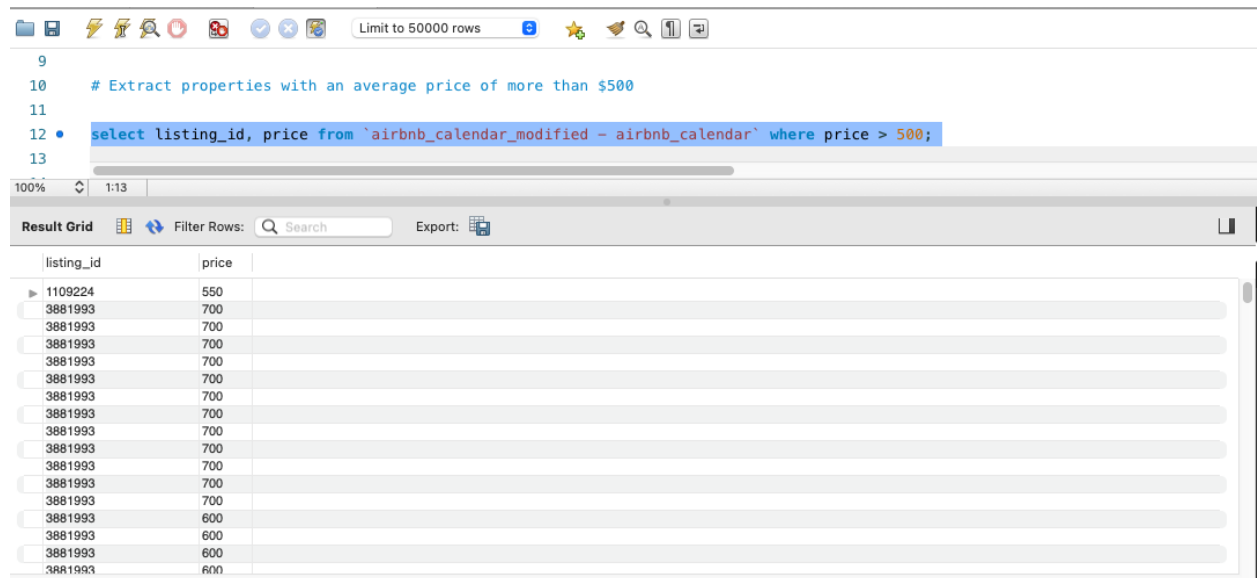


Name: Tathagata Roy

Employee code: TAS067

What is the time period used?



The screenshot shows a data analysis interface. At the top, there's a toolbar with various icons and a 'Limit to 50000 rows' button. Below the toolbar, a SQL query is displayed in a code editor. The query is as follows:

```
9
10 # Extract properties with an average price of more than $500
11
12 • select listing_id, price from `airbnb_calendar_modified` - `airbnb_calendar` where price > 500;
13
```

Below the query editor, there's a 'Result Grid' section. It includes a 'Filter Rows' search bar and an 'Export' button. The results are displayed in a table with two columns: 'listing_id' and 'price'.

listing_id	price
1109224	550
3881993	700
3881993	700
3881993	700
3881993	700
3881993	700
3881993	700
3881993	700
3881993	700
3881993	700
3881993	700
3881993	700
3881993	700
3881993	600
3881993	600
3881993	600
3881993	600

For each property, find out the number of days the property was available and not available (create a table with listing_id, available days, unavailable days and available days as a fraction of total days)

```
# For each property, find out the number of days the property was available and not available
# create a table with listing_id, available days, unavailable days and available days as a fraction of total days
create table available_property(
68     listing_id int,
69     available int
70 );
71
72 • insert into available_property(select listing_id, count(*) as available_count
73     from
74     `airbnb_calendar_modified` - `airbnb_calendar`
75     where available = 't'
76     group by listing_id);
77
78 • create table availability(
79     listing_id int,
80     available int,
81     not_available int,
82     fraction_of_dates_available FLOAT
83 );
84
85 • select listing_id, available, 365 - available as not_available, available/365 as fraction_of_dates_available
86     from available_property;
87
88 • insert into availability(select listing_id, available, 365 - available as not_available, available/365 as fraction_of_dates_available
89     from available_property);
90
91 • select * from availability;
92
```

How many properties were available on more than 50% of the days?

```
33
34
35 #Properties available more than 50% time
36
37 • select count(*) as available_more_than_50_percent_time from(select listing_id, count(*) as count_entry
38   from
39   `airbnb_calendar_modified` - `airbnb_calendar`
40   where available = 't'
41   group by listing_id
42   having count(*) > 182) t;
43
```

100% 20:41

Result Grid Filter Rows: Search Export:

available_more_than_50_percent_t...
▶ 1726

How many properties were available on more than 75% of the days?

Limit to 50000 rows

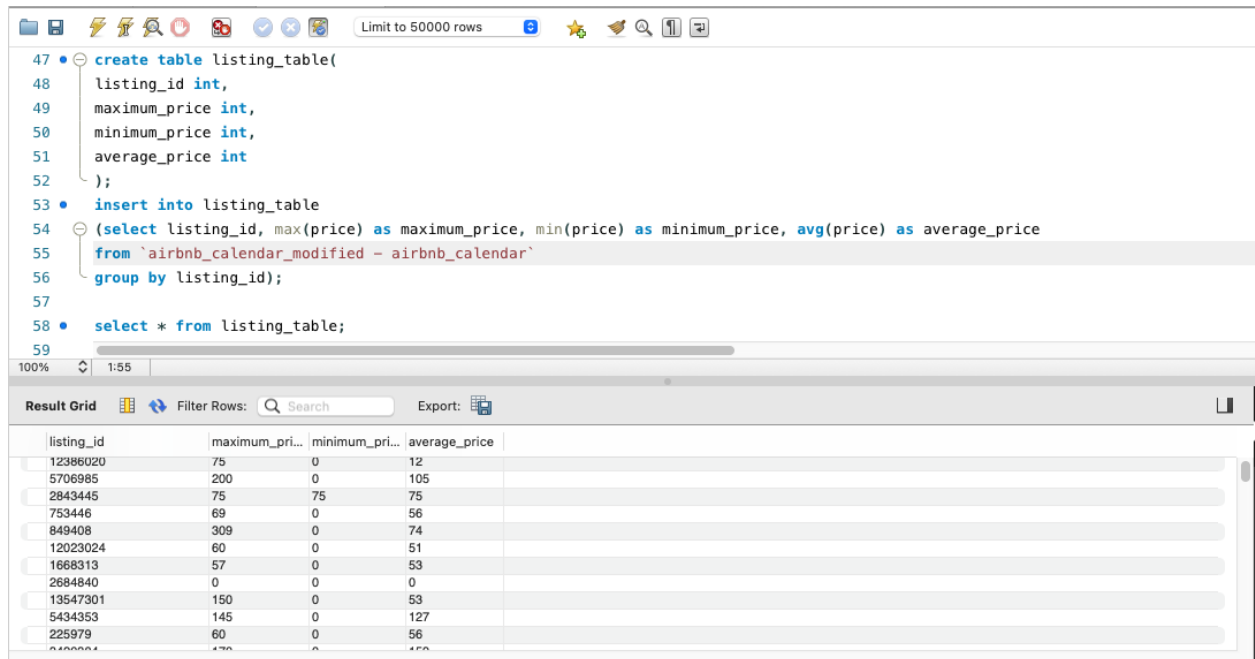
```
22
23
24
25 #Properties available more than 75% time
26
27 • select count(*) as available_more_than_75_percent_time from(select listing_id, count(*) as count_entry
28   from
29   `airbnb_calendar_modified` - `airbnb_calendar`
30   where available = 't'
31   group by listing_id
32   having count(*) > 273) t;
33
34
```

100% 26:32

Result Grid Filter Rows: Search Export:

available_more_than_75_percent_time
▶ 1423

Create a table with max, min and average price of each property



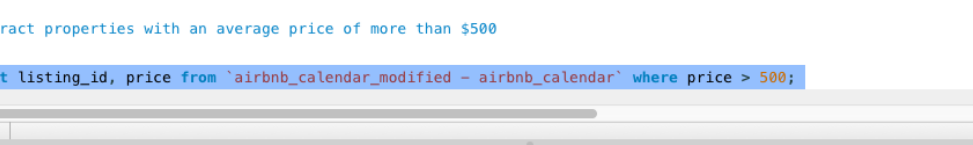
The screenshot shows a SQL IDE interface with a toolbar at the top containing icons for file operations, execution, and search. A status bar indicates "Limit to 50000 rows". The SQL editor contains the following code:

```
47 • create table listing_table(  
48     listing_id int,  
49     maximum_price int,  
50     minimum_price int,  
51     average_price int  
52 );  
53 • insert into listing_table  
54     (select listing_id, max(price) as maximum_price, min(price) as minimum_price, avg(price) as average_price  
55     from `airbnb_calendar_modified` - `airbnb_calendar`  
56     group by listing_id);  
57  
58 • select * from listing_table;  
59
```

Below the editor, a "Result Grid" tab is active, displaying the results of the query. It includes a "Filter Rows" search bar and an "Export" button. The data is presented in a table with the following columns: listing_id, maximum_price, minimum_price, and average_price.

listing_id	maximum_price	minimum_price	average_price
12386020	75	0	12
5706985	200	0	105
2843445	75	75	75
753446	69	0	56
849408	309	0	74
12023024	60	0	51
1668313	57	0	53
2684840	0	0	0
13547301	150	0	53
5434353	145	0	127
225979	60	0	56

Extract properties with an average price of more than \$500



The screenshot shows a Jupyter Notebook interface. The top toolbar includes icons for file operations, search, and execution. The code cell contains a SQL query to filter listings with an average price greater than \$500. The output area shows the 'Result Grid' with columns 'listing_id' and 'price'. The first row is expanded, showing a listing_id of 1109224 and a price of 550. Subsequent rows show listing_id 3881993 with a price of 700, and then listing_id 3881993 with a price of 600.

```
9  
10 # Extract properties with an average price of more than $500  
11  
12 • select listing_id, price from `airbnb_calendar_modified` - `airbnb_calendar` where price > 500;  
13
```

100% 1:13

Result Grid Filter Rows: Search Export:

listing_id	price
1109224	550
3881993	700
3881993	700
3881993	700
3881993	700
3881993	700
3881993	700
3881993	700
3881993	700
3881993	700
3881993	700
3881993	700
3881993	700
3881993	700
3881993	600
3881993	600
3881993	600
3881993	600