```
/*
    Name: Milton Clary
    DTSC660: Data and Database Managment with SQL
    Module 8
    Assignment 7

WHY I  CHOSE THE AIR BNB DATA SET:
I chose the AirBnB data set because I have always had an interest in real
estate and the AirBnB business model. specifically in the short term
rental space.

*/

-------------------------------------------------------------------------------
/*                              Table Creation                               */
-------------------------------------------------------------------------------

CREATE TABLE airbnb (id varchar(70),
                     name text,
                     host_id varchar(70),
                     host_identity_verified varchar(70),
                     host_name varchar(70),
                     neighbourhood_group varchar(70),
                     neighbourhood varchar(70),
                     lat numeric(50,10),
                     long numeric(50,10),
                     country varchar(70),
                     country_code varchar(70),
                     instant_bookable bool,
                     cancelation_policy varchar(70),
                     room_type varchar(70),
                     construction_year varchar(70),
                     price money,
                     service_fee money,
                     minimum_nights int,
                     number_of_reviews int,
                     last_review varchar(70),
                     reviews_per_month numeric(50,10),
                     review_rate_number int,
                     calculated_host_listings_count int,
                     availability_365 int,
                      house_rules text,
                     license varchar(70));

-------------------------------------------------------------------------------
```

```sql
-------------------------------------------------------------------------------
/*                              Import Data                                  */
-------------------------------------------------------------------------------

COPY airbnb
FROM 'C:\Users\Public\SQL\Airbnb_Open_Data.csv'
WITH (FORMAT CSV, HEADER);

SELECT * FROM airbnb;

-------------------------------------------------------------------------------



-------------------------------------------------------------------------------
/*                              Question 1:                                  */
-------------------------------------------------------------------------------

CREATE TABLE airbnb_backup AS SELECT * FROM airbnb;

-------------------------------------------------------------------------------
/*                               Question 2                                  */
-------------------------------------------------------------------------------

ALTER TABLE airbnb ADD COLUMN host_name_copy varchar(70);
UPDATE airbnb SET host_name_copy = host_name;

-------------------------------------------------------------------------------
/*                               Question 3                                  */
-------------------------------------------------------------------------------

/* 13 rows of 'minimum_nights' were negative numbers. I chose to address
this using option a., and made these values NULL. I used UPDATE to update
the TABLE, SET to set the minimum_nights column to NULL when the condition
of less than zero. WHERE specifies that condition. */

UPDATE airbnb
SET minimum_nights = NULL
WHERE minimum_nights < 0

-------------------------------------------------------------------------------
/*                               Question 4                                  */
```

```sql
-------------------------------------------------------------------------------

/* Price had NULL values. I Used aproach b. on the price column. I took the mean
of the
price column, and filled in the null values. I used UPDATE to update
the TABLE, SET to set the price column to the average of price. In order
to get the average, I used a subquery where I used CAST to change the
data type to numeric so I could use the AVG aggregate function. I then
specified that the average should only look at values that were not NULL.
Also, I used WHERE to specify the NULL values needed to be replaced. */

UPDATE airbnb
SET price = (SELECT AVG(CAST(price AS numeric(70,10))) FROM airbnb WHERE price IS
NOT NULL)
WHERE price IS NULL;

-------------------------------------------------------------------------------
/*                                Question 5                                 */
-------------------------------------------------------------------------------
/* There were values in the neighbourhood_group column that were mispelled.
I chose to correct the Brooklyn value that was mispelled to be consistent.
I used UPDATE then the name of the table to be updated, SET the
neighbourhood_group to the value i wanted to be used. then the WHERE to
specify what value needed to be replaced.*/

UPDATE airbnb
SET neighbourhood_group = 'Brooklyn'
WHERE neighbourhood_group= 'brookln'

-------------------------------------------------------------------------------
/*                                Question 6                                 */
-------------------------------------------------------------------------------
/* In the host_name column, I found the name Cil mispelled as 'Cil. I used
UPDATE then the name of the table to be updated, SET the
host_name to the value i wanted to be used. then the WHERE to
specify what value needed to be replaced.*/
 */

UPDATE airbnb
SET host_name = 'Cil'
WHERE host_name = '''Cil';

-------------------------------------------------------------------------------
/*                                Question 7                                 */
-------------------------------------------------------------------------------
```

```
/* I chose to create a column from the house_rules column that specified
if the house allowed smoking. this would allow us to see if potentially
allowing smoking effected the price of the rental, as well as if the
properties that did or did not allow smoking were rented more frequently.
I started a transaction so that my changes would be reversible if needed.
I then used ALTER TABLE to create the new column to the table. I made
the data type a bool so that it was either it did allow smoking or did not.
I then created a case, I used ILIKE so that it would not be case
sensitive, and used the wild card so that no matter where it said no
smoking it would generate a TRUE. I then used an ELSE statment
so that it would return FALSE if it did not meet the condition.
I then used COMMIT to make the change permanent.

TROUBLESHOOTING STEPS:
I noticed there were other spellings of 'No Smoking' i.e. 'Non Smoking'
or 'no-smoking' I wanted to find a way to include those in my no-smoking column.
I added a regular expression to include n, dash, t, or space in-between no and
smoking
it does not catch all of the instances for example 'no pets or smoking' or
'smoking in the apartment not allowed'
However, it still provided enough data to give an idea on the impact "no smoking"
has on rentals.

*/
START TRANSACTION;

ALTER TABLE airbnb ADD COLUMN no_smoking bool;

UPDATE airbnb
SET no_smoking = CASE
WHEN house_rules ~* 'no[tn\- ]*smoking' THEN TRUE
ELSE FALSE
END
RETURNING no_smoking, house_rules;

COMMIT;
```