Question1

To connect to this MongoDB, you need to either on the campus network or connect via UCDavis VPN.

In this question, do not download more than enough resources from the server. Let the server to do all the calculations if possible. (Limit the results to the first 10 rows if necessary.)

The following code connects to a sample airbnb database. A sample of a document can be found at https://docs.atlas.mongodb.com/sample-data/sample-airbnb (https://docs.atlas.mongodb.com/sample-data/sample-airbnb)

The collection contains documents that represent the vacation home listing details and reviews of customers about the listing. These documents reflect a randomized subset of the original publicly available source, from several different cities around the globe.

```
library(tidyverse)
```

```
## ── Attaching packages ──────────────────────────────────────────────── ti
dyverse 1.3.0 ──
```

```
## ✓ ggplot2 3.3.0     ✓ purrr   0.3.3
## ✓ tibble  3.0.0     ✓ dplyr   0.8.5
## ✓ tidyr   1.0.2     ✓ stringr 1.4.0
## ✓ readr   1.3.1     ✓ forcats 0.5.0
```

```
## ── Conflicts ───────────────────────────────────────────────────── tidyvers
e_conflicts() ──
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(mongolite)

m <- mongo("airbnb", db = "data", url = "mongodb://mongouser:secret@alan.ucdavis.edu/dat
a")
```

a. How many properties are of `room_type == "Entire home/apt"` and number of beds >= 3.

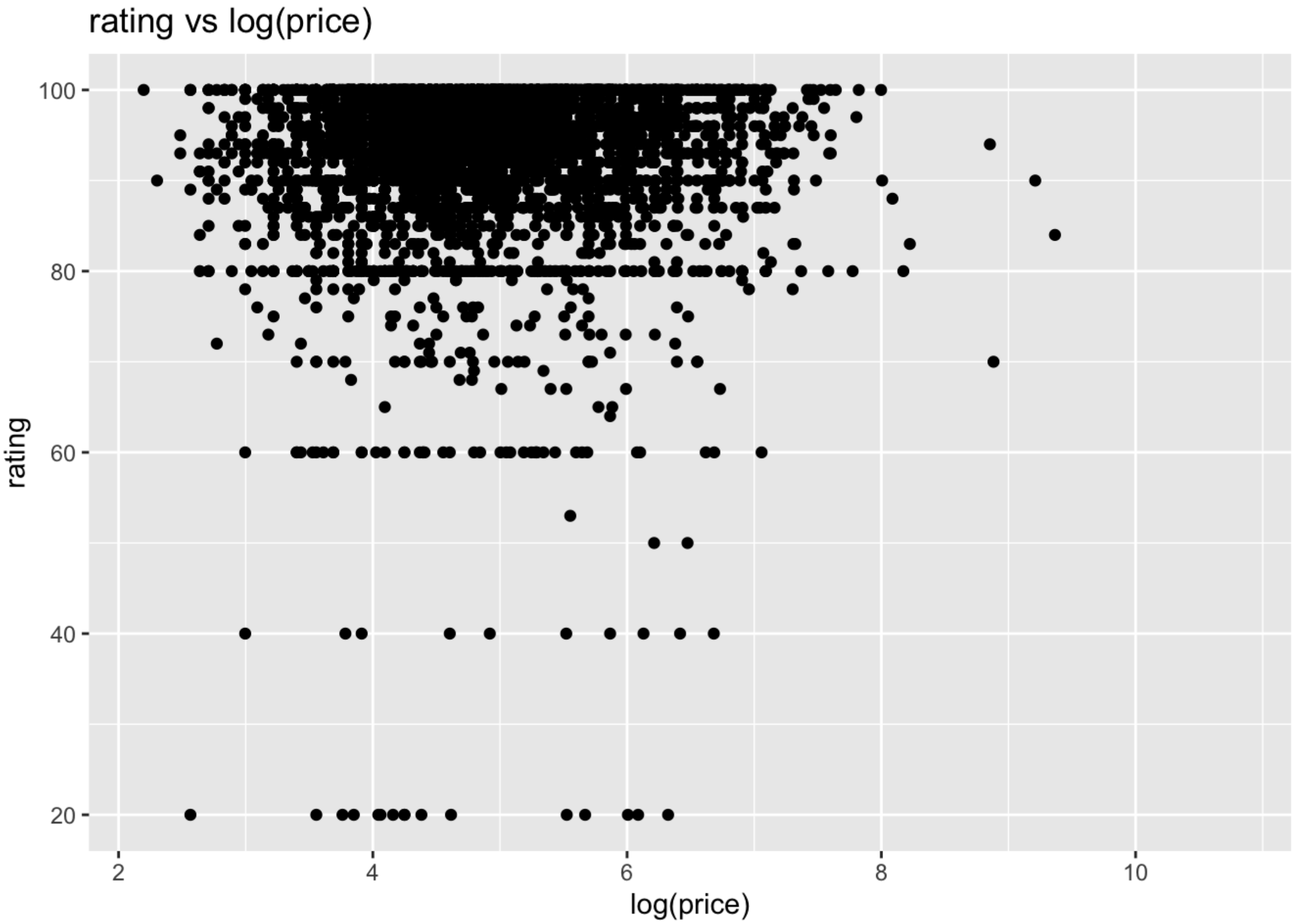
```
m$count('{"room_type": "Entire home/apt",
        "bedrooms": {"$gte": 3}}')
```

```
## [1] 631
```

b. Query the overall experience ratings (`review_scores_rating`) and prices for all properties and plot a scatter plot of `rating` vs `log(price)` .

```
review_price_df <- m$find(fields = '{"name": true, "review_scores.review_scores_rating":
true, "price": true}')
review_price_df %>%
  mutate(review_scores_rating = review_scores$review_scores_rating,
         review_scores = NULL,
         log_price = log(price)) %>%
  ggplot(aes(y = review_scores_rating, x = log(price))) +
  geom_point() +
  labs(title = "rating vs log(price)", x = "log(price)", y = "rating")
```

```
## Warning: Removed 1474 rows containing missing values (geom_point).
```



c. Find all property names that have “Washer” and “Kitchen”.

```
m$find('{"amenities": {"$all": ["Washer", "Kitchen"]}}',
       fields = '{"name": true}',
       limit = 10)
```

```
##         _id                                             name
## 1  10006546                          Ribeira Charming Duplex
## 2  10009999                     Horto flat with small garden
## 3   1001265                 Ocean View Waikiki Marina w/prkg
## 4  10030955                      Apt Linda Vista Lagoa - Rio
## 5   1003530              New York City - Upper West Side Apt
## 6  10038496                     Copacabana Apartment Posto 6
## 7  10047964                   Charming Flat in Downtown Moda
## 8  10057447                   Modern Spacious 1 Bedroom Loft
## 9  10057826                                Deluxe Loft Suite
## 10 10059244 Ligne verte - à 15 min de métro du centre ville.
```

d. What are the name, price and number of bedrooms for the property with the largest number of reviews has?

```
m$find(fields = '{"name": true, "price": true, "bedrooms": true, "number_of_reviews": tr
ue}',
       sort = '{"number_of_reviews": -1}',
       limit = 1)
```

```
##       _id                            name bedrooms number_of_reviews price
## 1 4069429 #Private Studio - Waikiki Dream        0               533   124
```

e. Consider all properties which have more than 100 reviews, what is their average price grouped by property type?

```
m$aggregate('[
  {"$match": {"number_of_reviews": {"$gt": 100}}},
  {"$group": {
    "_id": "$property_type",
    "price": { "$sum": "$price" }}
  }
]')
```

```
##                   _id price
## 1      Boutique hotel   968
## 2            Bungalow   275
## 3  Serviced apartment   286
## 4               Other    65
## 5          Aparthotel   109
## 6   Bed and breakfast   362
## 7             Cottage   470
## 8               Hotel    87
## 9          Guesthouse  1419
## 10              House  8626
## 11          Apartment 49312
## 12              Cabin   388
## 13          Treehouse   185
## 14        Guest suite  1241
## 15               Loft  1418
## 16        Condominium  7697
## 17          Townhouse  1448
## 18             Hostel   447
```

Question2

To connect to this MongoDB, you need to either on the campus network or connect via UCDavis VPN.

In this question, do not download more than enough resources from the server. Let the server to do all the calculations if possible. (Limit the results to the first 10 rows if necessary.)

The following code connects to a sample airbnb database. A sample of a document can be found at https://docs.atlas.mongodb.com/sample-data/sample-supplies/ (https://docs.atlas.mongodb.com/sample-data/sample-supplies/)

Each document in the `sales` collection represents a single sale from a store run by the supply company. Each document contains the item(s) purchased, information on the customer who made the purchase, and several other details regarding the sale.

```
library(tidyverse)
```

```
## ── Attaching packages ─────────────────────────────────────────────────── ti
dyverse 1.3.0 ──
```

```
## ✓ ggplot2 3.3.0     ✓ purrr   0.3.3
## ✓ tibble  3.0.0     ✓ dplyr   0.8.5
## ✓ tidyr   1.0.2     ✓ stringr 1.4.0
## ✓ readr   1.3.1     ✓ forcats 0.5.0
```

```
## ── Conflicts ──────────────────────────────────────────────────────── tidyvers
e_conflicts() ──
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(mongolite)

m <- mongo("sales", db = "data", url = "mongodb://mongouser:secret@alan.ucdavis.edu/dat
a")
```

Hint: to handle the items, you will need to use a `$unwind` stage in `aggregate` .

The following unwinds the `items` array for a particular customer .

```
m$aggregate('[
  {"$match": {"customer.email": "cauho@witwuta.sv"}},
  {"$unwind": "$items"}
]')
```

The following gives a list of items for each transaction.

```
m$aggregate('[
  {"$unwind": "$items"},
  {"$group": {
    "_id": "$_id",
    "items": {"$addToSet" : "$items.name"},
    "purchaseMethod": {"$first": "$purchaseMethod"}
    }
  },
  {"$project": {"_id": false}},
  {"$limit": 10}
]')
```

(a) Find the number of items in each transaction.

```
query <- m$aggregate('[
    {"$unwind": "$items"},
    {"$group": {
      "_id": "$_id",
      "items": {"$addToSet" : "$items"},
      "count": {"$sum": "$items.quantity"}
      }
    },
    {"$limit": 10}
]')
query %>%
  select(`_id`, count)
```

```
##                          _id count
## 1  5bd761deae323e45a93ce2e5     9
## 2  5bd761deae323e45a93ce2e4    19
## 3  5bd761deae323e45a93ce2e3    25
## 4  5bd761deae323e45a93ce2e2    12
## 5  5bd761deae323e45a93ce2e1    37
## 6  5bd761deae323e45a93ce2e0    37
## 7  5bd761deae323e45a93ce2df    21
## 8  5bd761deae323e45a93ce2de     3
## 9  5bd761deae323e45a93ce1ff     4
## 10 5bd761deae323e45a93ce1fe    25
```

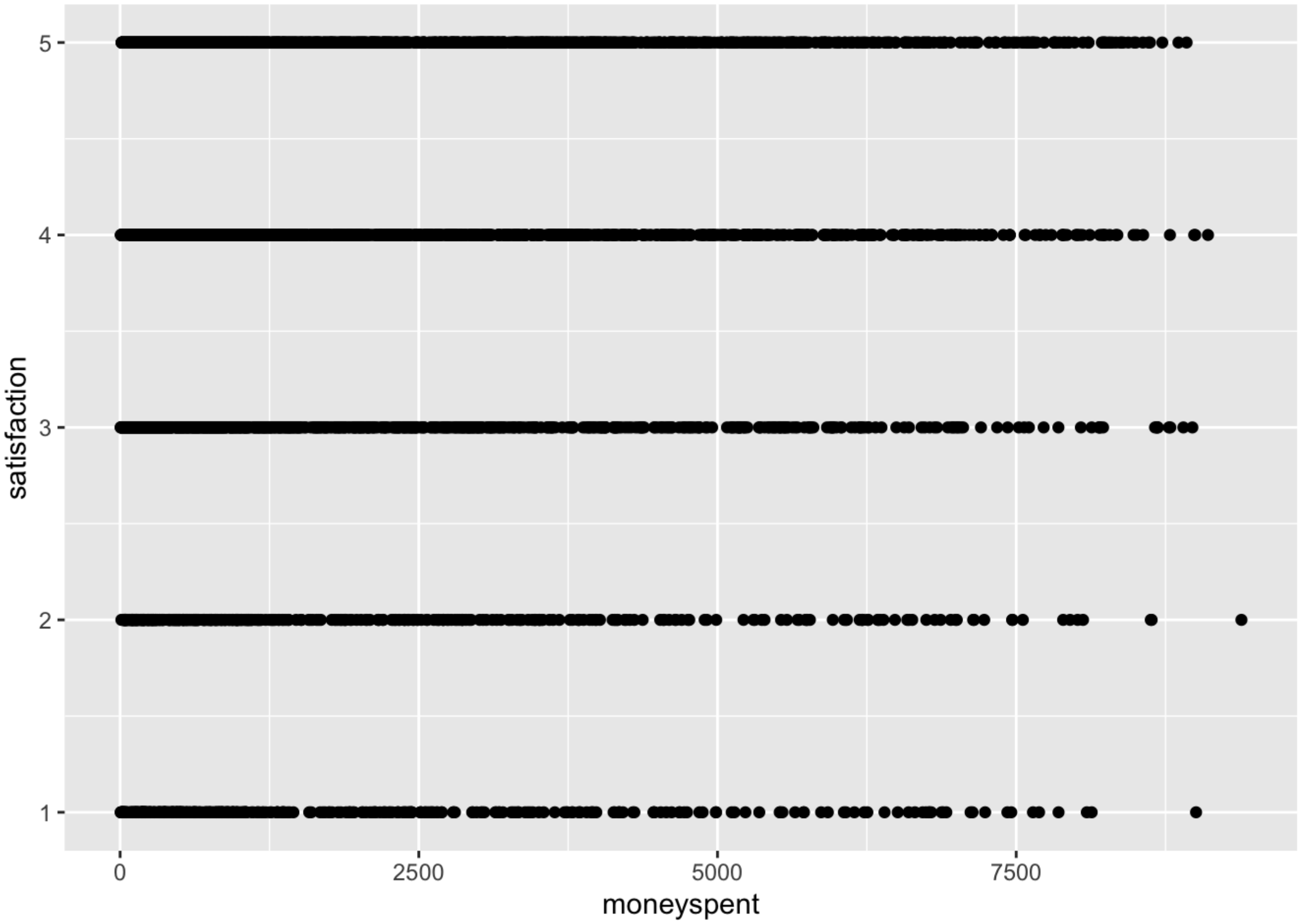
(b) Find the amount of money spent in each transaction. (Don’t forget multiple the `quantity` of each item)

```
m$aggregate('[
    {"$unwind": "$items"},
    {"$project": {
      "subtotal": {"$multiply": ["$items.price", "$items.quantity"]}
      }
    },
    {"$group": {
      "_id": "$_id",
      "moneyspent": {"$sum": "$subtotal"}
        }
    },
    {"$limit": 10}
]')
```

```
##                         _id moneyspent
## 1  5bd761deae323e45a93ce2e5     317.95
## 2  5bd761deae323e45a93ce2e4    5736.57
## 3  5bd761deae323e45a93ce2e3    5904.47
## 4  5bd761deae323e45a93ce2e2     394.87
## 5  5bd761deae323e45a93ce2e1     791.95
## 6  5bd761deae323e45a93ce2e0    2278.50
## 7  5bd761deae323e45a93ce2df     539.93
## 8  5bd761deae323e45a93ce2de      29.37
## 9  5bd761deae323e45a93ce1ff      44.04
## 10 5bd761deae323e45a93ce1fe     582.94
```

(c) Compute each customer satisfaction and plot it against the transction amount (you could reuse the result from (b)).

```
# inner join on b
x <- m$aggregate('[
    {"$unwind": "$items"},
    {"$project": {
      "subtotal": {"$multiply": ["$items.price", "$items.quantity"]}
      }
    },
    {"$group": {
      "_id": "$_id",
      "moneyspent": {"$sum": "$subtotal"}
        }
    }
]')
y <- m$find(fields = '{"customer.satisfaction": true}')
y <- y %>%
  mutate(satisfaction = customer$satisfaction, customer = NULL)
inner_join(
  x, y,
  by = "_id"
) %>%
  ggplot(aes(x = moneyspent, y = satisfaction)) +
  geom_point()
```



(d) Find the total sum of the transactions for each store.

```
# similar to a
m$aggregate('[
    {"$unwind": "$items"},
    {"$group": {
      "_id": "$storeLocation",
      "sum": {"$sum" : 1}
      }
    }
]')
```

```
##         _id  sum
## 1 San Diego 1891
## 2   Seattle 6121
## 3    Denver 8446
## 4    London 4395
## 5  New York 2758
## 6    Austin 3827
```

(e) How many notepad were sold in total?

```
m$aggregate('[
  {"$unwind": "$items"},
  {"$match": {"items.name": "notepad"}},
  {"$group": {
    "_id": null,
    "notepads": {"$sum": "$items.quantity"}
  }}
]')
```

```
##   _id notepads
## 1  NA    20727
```