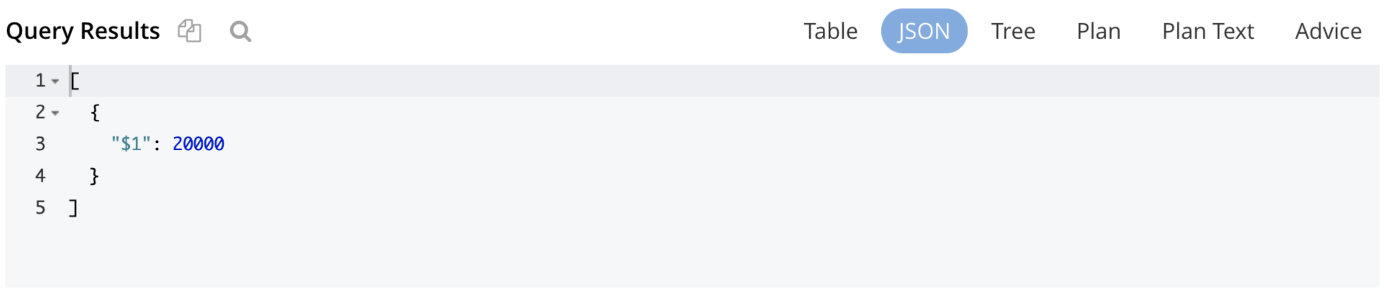
Last Name: Jengjirapas First Name: Natcha Student ID: 85939811

1.

A) The data type for the field “categories” is an array of strings.
B) The data type for the field “time_placed” is a string.
C) The data type for the field “phones” is an array of objects with subtypes including kind (string), and number (string).
D) The data type for the field “year” is number.

2.

A) Results of query after creating index:



```
[
  {
    "$1": 20000
  }
]
```

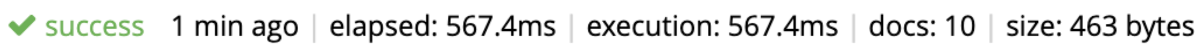
B)

a. Query:

```
SELECT Value `name`
FROM ShopALot
WHERE `type` = "products" and `category` = "Bread & Bakery" and
`list_price` > 3 Limit 10;
```

b. Time to execute without index:

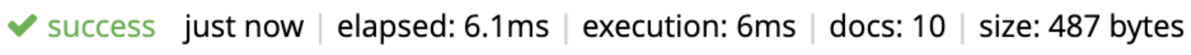
567.4ms



```
success 1 min ago | elapsed: 567.4ms | execution: 567.4ms | docs: 10 | size: 463 bytes
```

c. Time to execute with index:

6ms

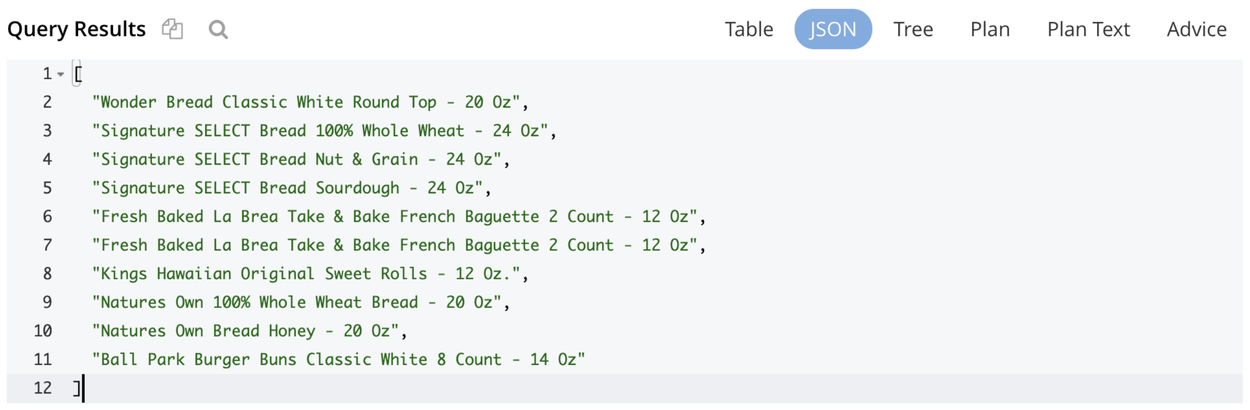


```
success just now | elapsed: 6.1ms | execution: 6ms | docs: 10 | size: 487 bytes
```

d. Create index statement:

```
CREATE INDEX adv_category_list_price_type_name ON
`ShopALot`(`category`,`list_price`,`name`) WHERE (`type` = 'products')
```

e. Results of query:



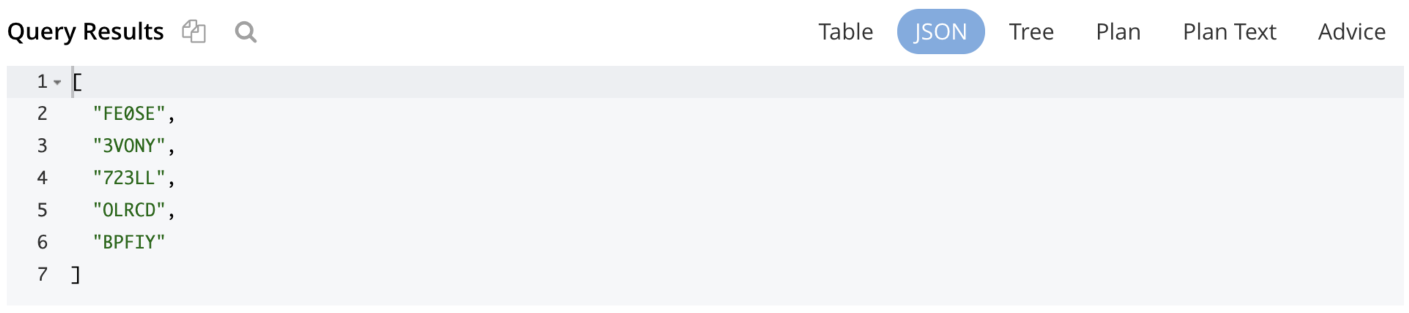
```
[
  "Wonder Bread Classic White Round Top - 20 Oz",
  "Signature SELECT Bread 100% Whole Wheat - 24 Oz",
  "Signature SELECT Bread Nut & Grain - 24 Oz",
  "Signature SELECT Bread Sourdough - 24 Oz",
  "Fresh Baked La Brea Take & Bake French Baguette 2 Count - 12 Oz",
  "Fresh Baked La Brea Take & Bake French Baguette 2 Count - 12 Oz",
  "Kings Hawaiian Original Sweet Rolls - 12 Oz.",
  "Natures Own 100% Whole Wheat Bread - 20 Oz",
  "Natures Own Bread Honey - 20 Oz",
  "Ball Park Burger Buns Classic White 8 Count - 14 Oz"
]
```

C)

a. Query:

```
SELECT Value `customer_id`
FROM ShopALot
WHERE `type` = "orders" and `total_price` > 600 and array_count(`items`) > 1;
```

b. Results of query:



```
[
  "FE0SE",
  "3VONY",
  "723LL",
  "OLRCD",
  "BPFIY"
]
```

✓ success just now | elapsed: 1.5s | execution: 1.5s | docs: 5 | size: 35 bytes

D)

a. Query:

```
select s.user_id, s.name
from ShopALot as s
where `type` = "users" and "SHOPPER" in s.kind and any phone in phones
satisfies phone.kind = "HOME" END and capacity is not missing
order by name.`last` ASC
limit 5;
```

b. Results of query:

```
[
  {
    "name": {
      "first": "Ashley",
      "last": "Aguilar"
    },
    "user_id": "YZNCV"
  },
  {
    "name": {
      "first": "Kyle",
      "last": "Arellano"
    },
    "user_id": "Q7W81"
  },
  {
    "name": {
      "first": "Joh",
      "last": "Arnold"
    },
    "user_id": "27PTC"
  },
  {
    "name": {
      "first": "Lor",
      "last": "Arnold"
    },
    "user_id": "5MWNO"
  },
  {
    "name": {
      "first": "Bri",
      "last": "Brown"
    },
    "user_id": "3C98L"
  }
]
```

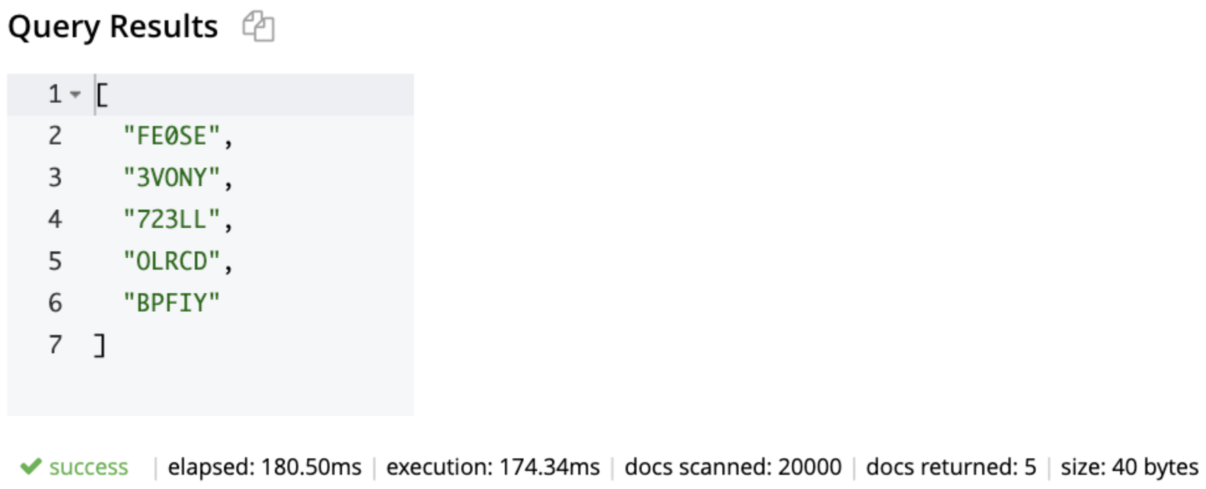
✔ success   just now | elapsed: 1.4s | execution: 1.4s | docs: 5 | size: 610 bytes

E)

a. Query C:

```
SELECT Value o.customer_id
FROM orders o
WHERE o.total_price > 600 and array_count(o.items) > 1;
```

b. Results of query:



Query Results

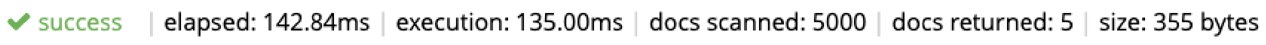
```
[
  "FE0SE",
  "3VONY",
  "723LL",
  "OLRCD",
  "BPFIY"
]
```

✓ success | elapsed: 180.50ms | execution: 174.34ms | docs scanned: 20000 | docs returned: 5 | size: 40 bytes

c. Query D:

```
select distinct s.user_id, s.name
from users as s
unnest s.phones as phone
where "SHOPPER" in s.kind and phone.kind = "HOME" and s.capacity is not
missing
order by s.name.`last` ASC
limit 5;
```

d. Results of query:



✓ success | elapsed: 142.84ms | execution: 135.00ms | docs scanned: 5000 | docs returned: 5 | size: 355 bytes

```
[
  {
    "user_id": "YZNCV",
    "name": {
      "first": "Ashley",
      "last": "Aguilar"
    }
  },
  {
    "user_id": "Q7W81",
    "name": {
      "first": "Kyle",
      "last": "Arellano"
    }
  },
  {
    "user_id": "27PTC",
    "name": {
      "first": "Joh",
      "last": "Arnold"
    }
  },
  {
    "user_id": "5MWNO",
    "name": {
      "first": "Lor",
      "last": "Arnold"
    }
  },
  {
    "user_id": "EGXWH",
    "name": {
      "first": "Kenneth",
      "last": "Brown"
    }
  }
]
```

e. Time of execution of both queries in Query vs Analytics

For query C, it took 1.5s in query service and 174.34ms in analytics service. For query D, it took 1.4s in query service and 135ms in analytics service. Running queries in analytics service is faster than query service.

F)

a. Query:

```sql
SELECT u.name, u.email, o.order_id, o.total_price
FROM orders o, users u
WHERE o.total_price > 600 and o.customer_id = u.user_id
order by o.total_price DESC;
```

b. Results of query:

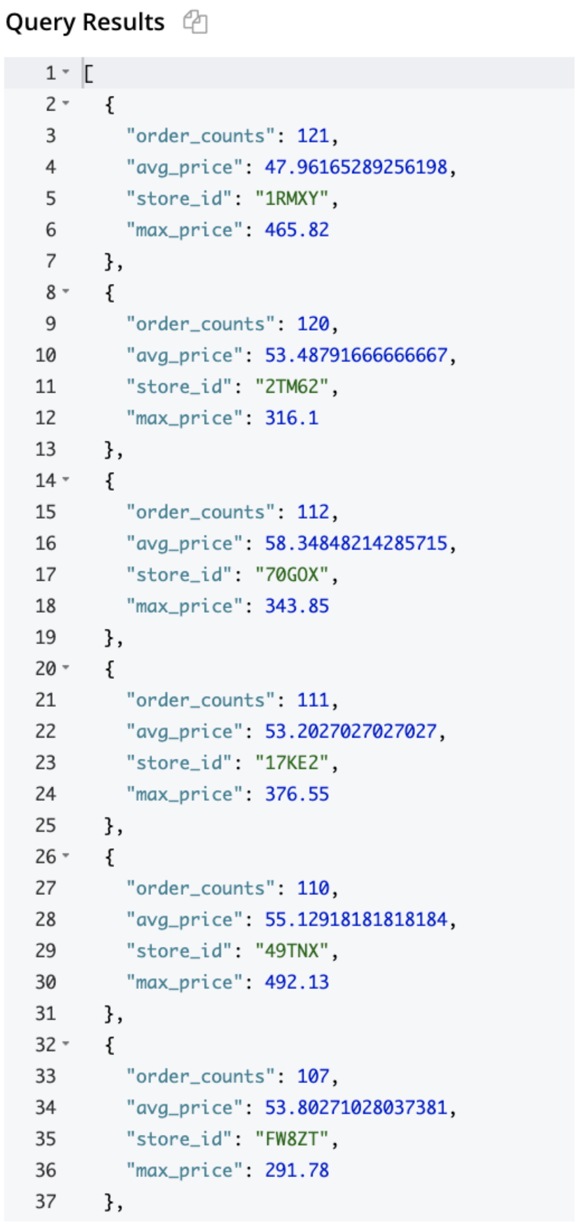
```json
[
  {
    "name": {
      "first": "Christine",
      "last": "Thomas"
    },
    "email": "thomas89979@hotmail.com",
    "order_id": "G6BT1",
    "total_price": 716.8
  },
  {
    "name": {
      "first": "Ter",
      "last": "Kirk"
    },
    "email": "kirk.ter478@gmail.com",
    "order_id": "JVQ1M",
    "total_price": 701.5
  },
  {
    "name": {
      "first": "Joseph",
      "last": "Gonzalez"
    },
    "email": "gonzalez855@yahoo.com",
    "order_id": "ZFGRA",
    "total_price": 700.82
  },
  {
    "name": {
      "first": "Brooke",
      "last": "Perez"
    },
    "email": "perezbrooke2373@aol.com",
    "order_id": "IKTI7",
    "total_price": 670.21
  },
  {
    "name": {
      "first": "Kei",
      "last": "Fields"
    },
    "email": "Fields_kei@aol.com",
    "order_id": "SW6PI",
    "total_price": 624.56
  },
  {
    "name": {
      "first": "Rya",
      "last": "Wright"
    },
    "email": "wrightrya@gmail.com",
    "order_id": "DNQ51",
    "total_price": 624.14
  },
  {
    "name": {
      "first": "Brittany",
      "last": "Weaver"
    },
    "email": "brittanyWeaver@gmail.com",
    "order_id": "9SBRP",
    "total_price": 600.14
  }
]
```

G)

a. Query:

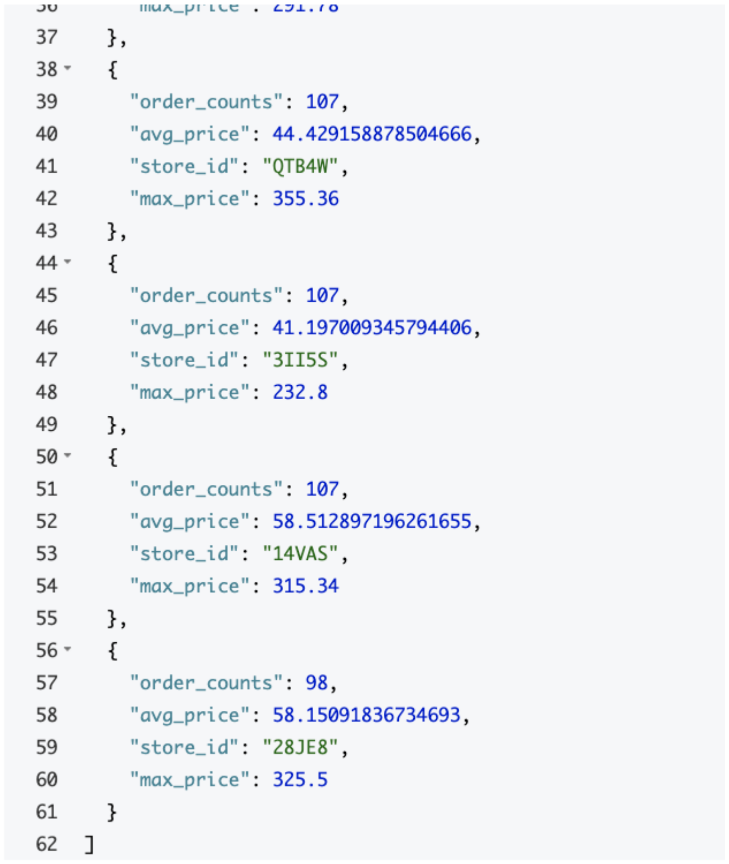
```
select o.store_id, count(o.order_id) as order_counts, avg(o.total_price) as
avg_price, max(o.total_price) as max_price
from orders o
group by o.store_id
order by order_counts DESC
limit 10;
```

b. Results of query:



**Query Results**

```
[
  {
    "order_counts": 121,
    "avg_price": 47.96165289256198,
    "store_id": "1RMXY",
    "max_price": 465.82
  },
  {
    "order_counts": 120,
    "avg_price": 53.48791666666667,
    "store_id": "2TM62",
    "max_price": 316.1
  },
  {
    "order_counts": 112,
    "avg_price": 58.34848214285715,
    "store_id": "70GOX",
    "max_price": 343.85
  },
  {
    "order_counts": 111,
    "avg_price": 53.2027027027027,
    "store_id": "17KE2",
    "max_price": 376.55
  },
  {
    "order_counts": 110,
    "avg_price": 55.12918181818184,
    "store_id": "49TNX",
    "max_price": 492.13
  },
  {
    "order_counts": 107,
    "avg_price": 53.80271028037381,
    "store_id": "FW8ZT",
    "max_price": 291.78
  },
```



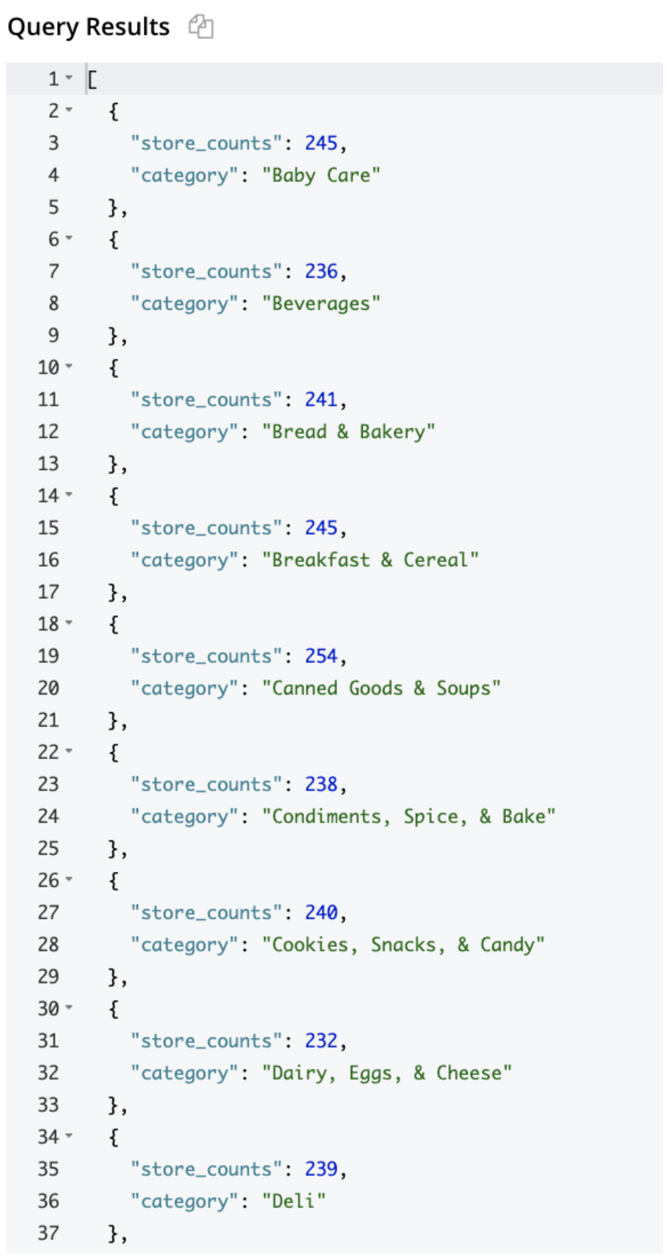
```
    "max_price": 291.78
  },
  {
    "order_counts": 107,
    "avg_price": 44.429158878504666,
    "store_id": "QTB4W",
    "max_price": 355.36
  },
  {
    "order_counts": 107,
    "avg_price": 41.197009345794406,
    "store_id": "3II5S",
    "max_price": 232.8
  },
  {
    "order_counts": 107,
    "avg_price": 58.512897196261655,
    "store_id": "14VAS",
    "max_price": 315.34
  },
  {
    "order_counts": 98,
    "avg_price": 58.15091836734693,
    "store_id": "28JE8",
    "max_price": 325.5
  }
]
```

H)

a. Query:

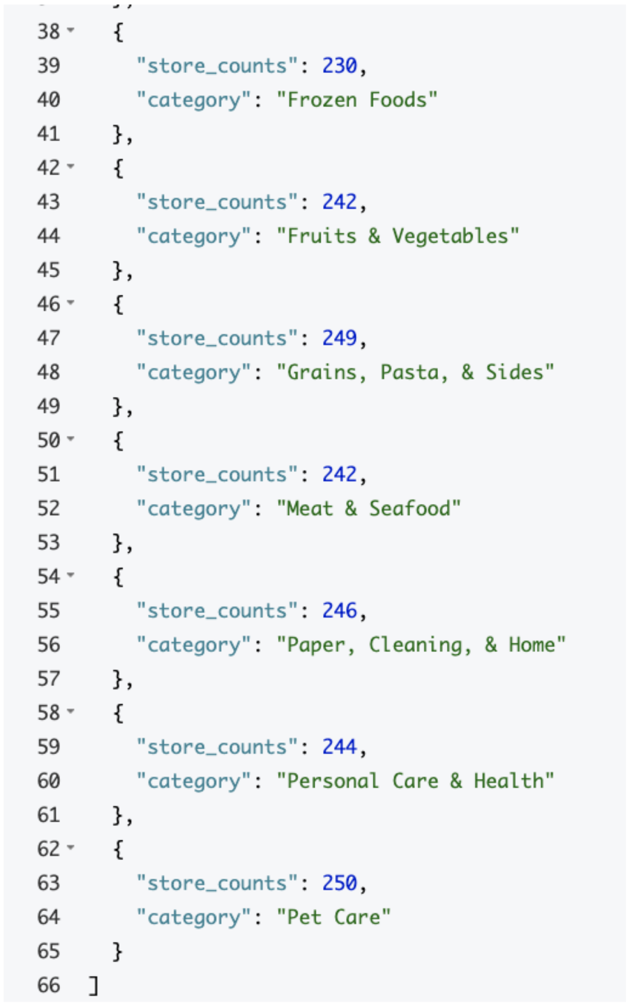
```
select c as category, count(s.store_id) as store_counts
from stores s
unnest s.categories as c
group by c;
```

b. Results of query:



Query Results

```
[
  {
    "store_counts": 245,
    "category": "Baby Care"
  },
  {
    "store_counts": 236,
    "category": "Beverages"
  },
  {
    "store_counts": 241,
    "category": "Bread & Bakery"
  },
  {
    "store_counts": 245,
    "category": "Breakfast & Cereal"
  },
  {
    "store_counts": 254,
    "category": "Canned Goods & Soups"
  },
  {
    "store_counts": 238,
    "category": "Condiments, Spice, & Bake"
  },
  {
    "store_counts": 240,
    "category": "Cookies, Snacks, & Candy"
  },
  {
    "store_counts": 232,
    "category": "Dairy, Eggs, & Cheese"
  },
  {
    "store_counts": 239,
    "category": "Deli"
  },
```



```json
  {
    "store_counts": 230,
    "category": "Frozen Foods"
  },
  {
    "store_counts": 242,
    "category": "Fruits & Vegetables"
  },
  {
    "store_counts": 249,
    "category": "Grains, Pasta, & Sides"
  },
  {
    "store_counts": 242,
    "category": "Meat & Seafood"
  },
  {
    "store_counts": 246,
    "category": "Paper, Cleaning, & Home"
  },
  {
    "store_counts": 244,
    "category": "Personal Care & Health"
  },
  {
    "store_counts": 250,
    "category": "Pet Care"
  }
]
```

I) Extra Credit:

a. Query:

```
WITH
mapco as (select distinct o.customer_id
          from orders o, stores s
          where o.store_id = s.store_id and s.name = "Mapco"),

countryM as (select distinct o.customer_id
             from orders o, stores s
             where o.store_id = s.store_id and s.name = "Country Market"),

finalCustomer as (select distinct mapco.customer_id
                  from mapco, countryM
                  where mapco.customer_id = countryM.customer_id and
                  mapco.customer_id not in (select distinct value o.customer_id
                             from orders o, stores s
                             where o.store_id = s.store_id and s.name = "7-Eleven")),

stat as (SELECT o.customer_id, count(o.order_id) as total_orders,
         min(o.total_price) as cheapest_order, max(o.total_price) as expensive_order
         FROM orders o
         group by o.customer_id)

SELECT u.user_id, u.name, stat.total_orders, stat.cheapest_order,
stat.expensive_order
FROM users u, stat, finalCustomer
WHERE u.user_id = stat.customer_id and
      stat.customer_id = finalCustomer.customer_id and
      u.email like "%aol.com" and
      stat.total_orders > 2
ORDER BY u.name.`last` ASC;
```

b. Results of query:

```
[
  {
    "total_orders": 7,
    "user_id": "EIHL9",
    "name": {
      "first": "Deborah",
      "last": "Bowman"
    },
    "cheapest_order": 10.94,
    "expensive_order": 214.34
  },
  {
    "total_orders": 7,
    "user_id": "5E8XN",
    "name": {
      "first": "Car",
      "last": "Gray"
    },
    "cheapest_order": 8.08,
    "expensive_order": 141.44
  },
  {
    "total_orders": 9,
    "user_id": "T85X9",
    "name": {
      "first": "Scott",
      "last": "Pugh"
    },
    "cheapest_order": 22.3,
    "expensive_order": 261.72
  },
  {
    "total_orders": 6,
    "user_id": "CMWO8",
    "name": {
      "first": "Catherine",
      "last": "Weber"
    },
    "cheapest_order": 5.32,
    "expensive_order": 111.45
  },
  {
    "total_orders": 4,
    "user_id": "NBGKB",
    "name": {
      "first": "Robert",
      "last": "Williams"
    },
    "cheapest_order": 24.4,
    "expensive_order": 192.72
  }
]
```