

# Базы данных

Малюгин Руслан

10 марта 2022 г.

## 1 Вставка данных

Вначале я начал работать с базой данных через MongoCompass. Нашел датасет и загрузил его через интерфейс.

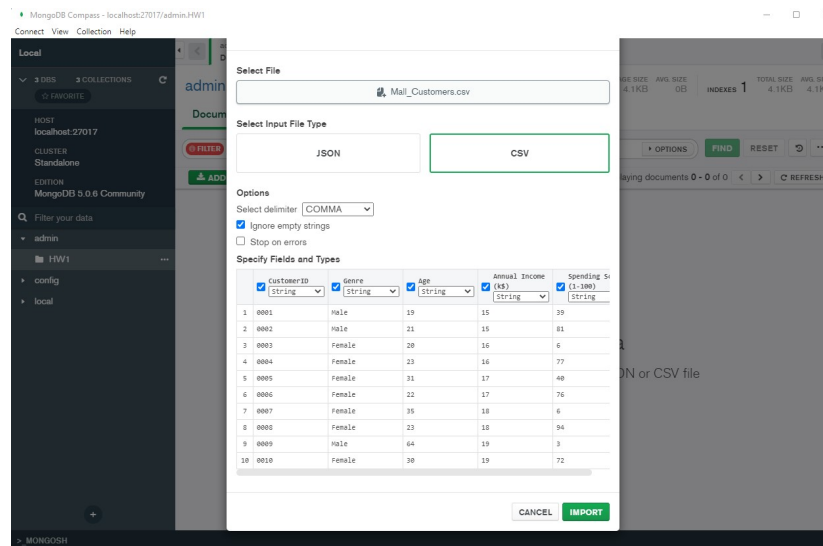


Рис. 1: Загрузка датасета.

После этого я так же через Compass, а именно MONGOSH консоль загрузил еще своих данных.

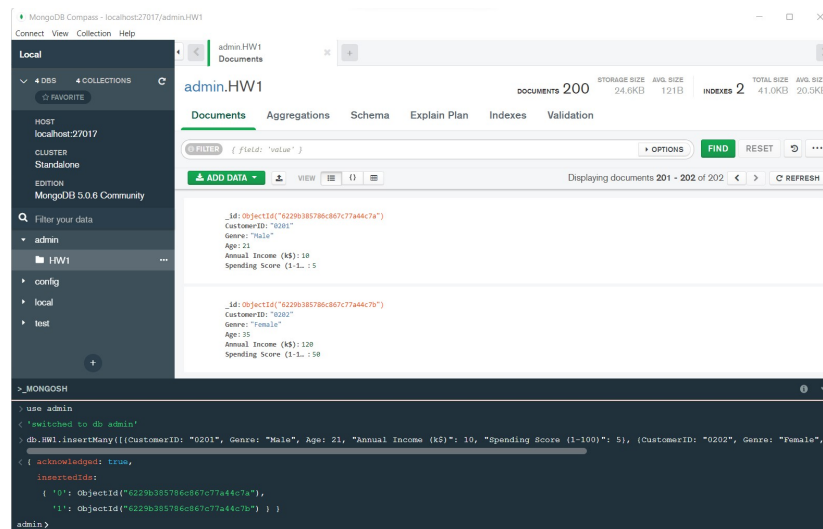


Рис. 2: Загрузка доп. данных.

## 2 Напишем несколько запросов

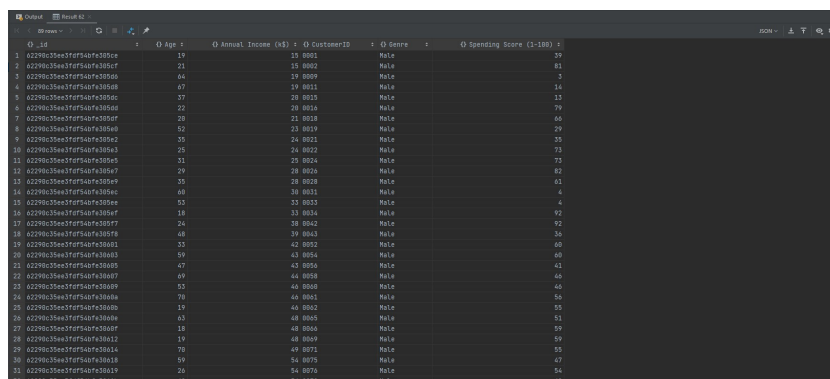
Здесь я решил, что удобнее будет пользоваться DataGrip.

### 2.1 Начало

Первый запрос, исполняет просто поиск ячеек с определенным полем.

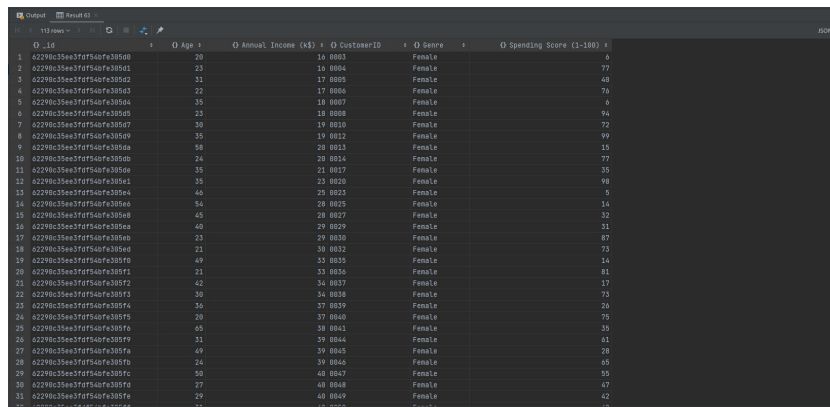
```
db.HW1.find({
  Genre: "Male",
})

db.HW1.find({
  Genre: "Female",
})
```



_id	Age	Annual Income (k\$)	CustomerID	Genre	Spending Score (1-100)
62298c35ea3f7d5aef3050a	19	15 0005	15 0005	Male	39
62298c35ea3f7d5aef3050f	21	15 0002	15 0002	Male	81
62298c35ea3f7d5aef3050d	64	19 0009	19 0009	Male	3
62298c35ea3f7d5aef3050e	67	19 0011	19 0011	Male	14
62298c35ea3f7d5aef3050c	27	20 0015	20 0015	Male	13
62298c35ea3f7d5aef3050a	22	20 0016	20 0016	Male	79
62298c35ea3f7d5aef3050e	20	21 0018	21 0018	Male	66
62298c35ea3f7d5aef3050d	52	23 0019	23 0019	Male	29
62298c35ea3f7d5aef3050c	20	24 0021	24 0021	Male	35
62298c35ea3f7d5aef3050e	25	24 0022	24 0022	Male	73
62298c35ea3f7d5aef3050d	31	25 0024	25 0024	Male	73
62298c35ea3f7d5aef3050e	20	26 0026	26 0026	Male	82
62298c35ea3f7d5aef3050e	35	28 0028	28 0028	Male	61
62298c35ea3f7d5aef3050c	60	30 0031	30 0031	Male	4
62298c35ea3f7d5aef3050e	53	32 0033	32 0033	Male	4
62298c35ea3f7d5aef3050f	18	33 0034	33 0034	Male	92
62298c35ea3f7d5aef3050f	24	38 0042	38 0042	Male	92
62298c35ea3f7d5aef3050e	40	39 0043	39 0043	Male	36
62298c35ea3f7d5aef30601	33	42 0052	42 0052	Male	60
62298c35ea3f7d5aef30603	59	43 0054	43 0054	Male	60
62298c35ea3f7d5aef3060e	47	43 0056	43 0056	Male	61
62298c35ea3f7d5aef3060f	49	44 0058	44 0058	Male	66
62298c35ea3f7d5aef30609	53	44 0068	44 0068	Male	66
62298c35ea3f7d5aef3060e	70	46 0061	46 0061	Male	56
62298c35ea3f7d5aef3060d	19	46 0062	46 0062	Male	55
62298c35ea3f7d5aef3060e	63	48 0065	48 0065	Male	51
62298c35ea3f7d5aef30609	18	48 0066	48 0066	Male	59
62298c35ea3f7d5aef30612	19	48 0069	48 0069	Male	59
62298c35ea3f7d5aef3061e	76	49 0071	49 0071	Male	55
62298c35ea3f7d5aef3061a	59	54 0075	54 0075	Male	67
62298c35ea3f7d5aef30619	26	54 0076	54 0076	Male	54

Рис. 3: Результат запроса 1.



_id	Age	Annual Income (k\$)	CustomerID	Genre	Spending Score (1-100)
62298c35ea3f7d5aef3050d	20	16 0003	16 0003	Female	6
62298c35ea3f7d5aef3050d	23	16 0006	16 0006	Female	77
62298c35ea3f7d5aef3050d	31	17 0005	17 0005	Female	40
62298c35ea3f7d5aef3050d	22	17 0006	17 0006	Female	76
62298c35ea3f7d5aef3050e	35	18 0007	18 0007	Female	9
62298c35ea3f7d5aef3050d	23	18 0008	18 0008	Female	94
62298c35ea3f7d5aef3050f	30	19 0010	19 0010	Female	72
62298c35ea3f7d5aef3050e	35	19 0022	19 0022	Female	49
62298c35ea3f7d5aef3050e	38	20 0013	20 0013	Female	15
62298c35ea3f7d5aef3050d	24	20 0014	20 0014	Female	77
62298c35ea3f7d5aef3050e	35	21 0017	21 0017	Female	35
62298c35ea3f7d5aef3050e	35	23 0020	23 0020	Female	98
62298c35ea3f7d5aef3050e	46	25 0023	25 0023	Female	5
62298c35ea3f7d5aef3050e	54	28 0025	28 0025	Female	14
62298c35ea3f7d5aef3050e	45	28 0027	28 0027	Female	32
62298c35ea3f7d5aef3050e	40	29 0029	29 0029	Female	31
62298c35ea3f7d5aef3050e	23	29 0030	29 0030	Female	67
62298c35ea3f7d5aef3050e	21	30 0032	30 0032	Female	73
62298c35ea3f7d5aef3050f	49	33 0035	33 0035	Female	14
62298c35ea3f7d5aef3050f	21	33 0036	33 0036	Female	81
62298c35ea3f7d5aef3050f	42	34 0037	34 0037	Female	17
62298c35ea3f7d5aef3050f	30	34 0038	34 0038	Female	73
62298c35ea3f7d5aef3050f	36	37 0039	37 0039	Female	26
62298c35ea3f7d5aef3050f	20	37 0040	37 0040	Female	75
62298c35ea3f7d5aef3050f	65	38 0041	38 0041	Female	35
62298c35ea3f7d5aef3050f	31	39 0044	39 0044	Female	61
62298c35ea3f7d5aef3050f	49	39 0045	39 0045	Female	28
62298c35ea3f7d5aef3050f	24	39 0046	39 0046	Female	65
62298c35ea3f7d5aef3050f	60	40 0047	40 0047	Female	53
62298c35ea3f7d5aef3050f	27	40 0048	40 0048	Female	67
62298c35ea3f7d5aef3050f	29	40 0049	40 0049	Female	62

Рис. 4: Результат запроса 2.

### 2.2 Попробуем агрегацию

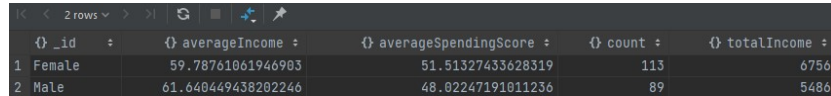
Воспользуемся \$group . Рассмотрим сколько в датасете мужчин, женщин, какая у них средняя зарплата и сколько они тратят.

```
db.HW1.aggregate({
  $group: {
```

```

    _id: "$Genre",
    totalIncome: { $sum: "$Annual_Income_(k$)" },
    averageIncome : { $avg: "$Annual_Income_(k$)" },
    count: { $sum : 1 },
    averageSpendingScore: { $avg: "$Spending_Score_(1-100)" }
  }
})

```



	_id :	averageIncome :	averageSpendingScore :	count :	totalIncome :
1	Female	59.78761061946983	51.51327433628319	113	6756
2	Male	61.640449438202246	48.02247191011236	89	5486

Рис. 5: Результат запроса 3.

Далее усовершенствуем запрос.

## 2.3 Совместим первые два пункта

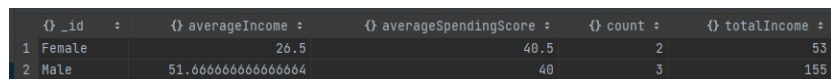
Теперь поставим ограничение, людям, которых мы рассматриваем должно быть 20 лет.

Для этого используем \$match .

```

db.HW1.aggregate([
  $match: {
    Age: 20
  },
  { $group: {
    _id: "$Genre",
    totalIncome: { $sum: "$Annual_Income_(k$)" },
    averageIncome : { $avg: "$Annual_Income_(k$)" },
    count: { $sum : 1 },
    averageSpendingScore: { $avg: "$Spending_Score_(1-100)" }
  } }
])

```



	_id :	averageIncome :	averageSpendingScore :	count :	totalIncome :
1	Female	26.5	40.5	2	53
2	Male	51.666666666666664	40	3	155

Рис. 6: Результат запроса 4.

## 2.4 Попробуем усложнить еще немного

Применим в начальном фильтре не точечное значение, а диапазон.

Это мы сделаем с помощью \$gt , \$lt .

Проведем мини исследование и сравним зарплаты и траты молодежи и старшего поколения, в зависимости от пола.

```

db.HW1.aggregate([
  $match: {
    Age: { $gt: 20, $lt: 30 }
  },
  { $group: {
    _id: "$Genre",
    totalIncome: { $sum: "$Annual_Income_(k$)" },
    averageIncome : { $avg: "$Annual_Income_(k$)" },

```

```

    count: {$sum : 1},
    averageSpendingScore: {$avg: "$Spending_Score_(1-100)"}
  }}
})

```

{ _id :	{ averageIncome :	{ averageSpendingScore :	{ count :	{ totalIncome :
1 Male	57	67.5	14	798
2 Female	50.958333333333336	63.25	24	1223

Рис. 7: Результат запроса 5.

```

db.HW1.aggregate([
  $match: {
    Age: { $gt: 60}
  },
  {$group: {
    _id: "$Genre",
    totalIncome: { $sum: "$Annual_Income_(k$)" },
    averageIncome : {$avg: "$Annual_Income_(k$)" },
    count: {$sum : 1},
    averageSpendingScore: {$avg: "$Spending_Score_(1-100)"}
  }}
])

```

{ _id :	{ averageIncome :	{ averageSpendingScore :	{ count :	{ totalIncome :
1 Male	48.18181818181818	42.54545454545455	11	538
2 Female	53.333333333333336	47.166666666666664	6	320

Рис. 8: Результат запроса 6.

### 3 Удаление и вставка через консоль

Тут ничего сложного, напомним простые скрипты.

```
db.HW1.deleteOne({ CustomerID : "0202" })
```

```
db.HW1.deleteOne({ CustomerID : "0201" })
```

```

db.HW1.insertOne({
  Age: 25,
  "Annual_Income_(k$)" : 20,
  CustomerID: 0201,
  Genre: "Male",
  "Spending_Score_(1-100)": 45,
})

```

{ acknowledged :	{ deletedCount :	{ acknowledged :	{ deletedCount :
1 • true	1	1 • true	1

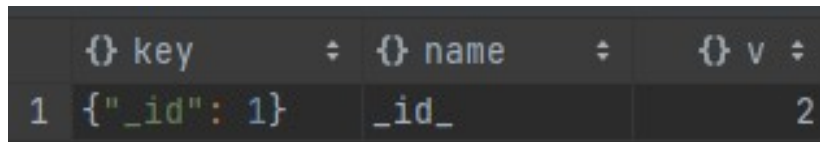
Рис. 9: Результат запросов.

## 4 Индексы

Для начала посмотрим какие у нас индексы есть.

Это делается следующей командой.

```
db.HW1.getIndexes()
```



	{ } key	÷	{ } name	÷	{ } v	÷
1	{ "_id": 1 }		_id_		2	

Рис. 10: Индексы.

По дефолту у нас всегда один индекс.

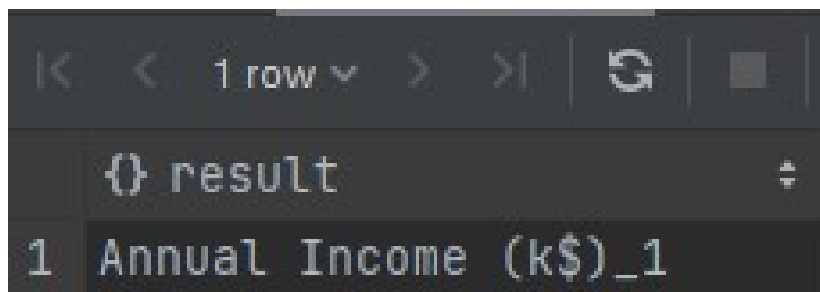
### 4.1 Создание и удаление индекса

Создадим и сразу удалим какой-нибудь индекс.

Создание и удаление осуществляется следующей командой

```
db.HW1.createIndex(  
{"Annual_Income_(k$)": 1}  
)
```

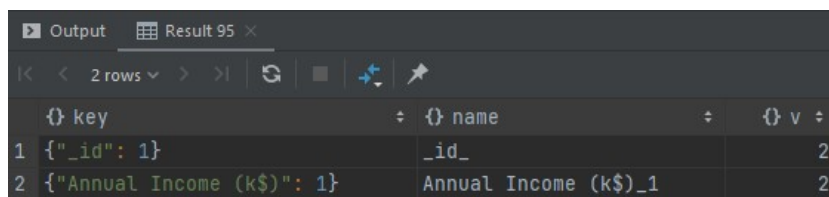
```
db.HW1.dropIndex(  
{"Annual_Income_(k$)": 1}  
)
```



	{ } result	÷
1	Annual Income (k\$)_1	

Рис. 11: Создание индекса.

Проверим, что он существует, командой из прошлого пункта



	{ } key	÷	{ } name	÷	{ } v	÷
1	{ "_id": 1 }		_id_		2	
2	{ "Annual_Income_(k\$)": 1 }		Annual Income (k\$)_1		2	

Рис. 12: Индекс.

Теперь удалим его

Проверим, что его нет.

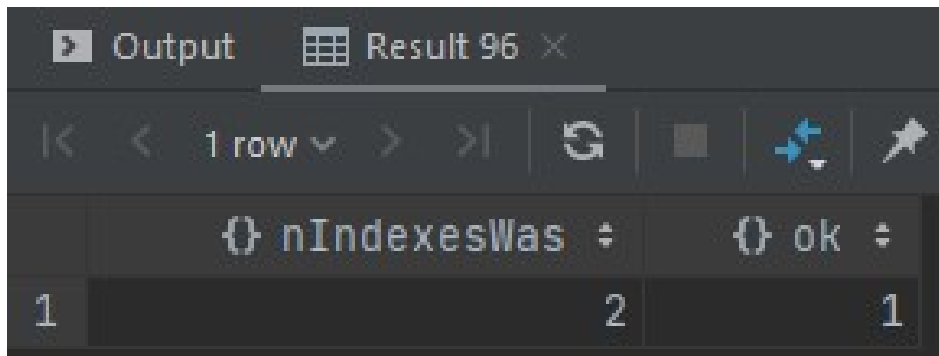


Рис. 13: Удаленный индекс.

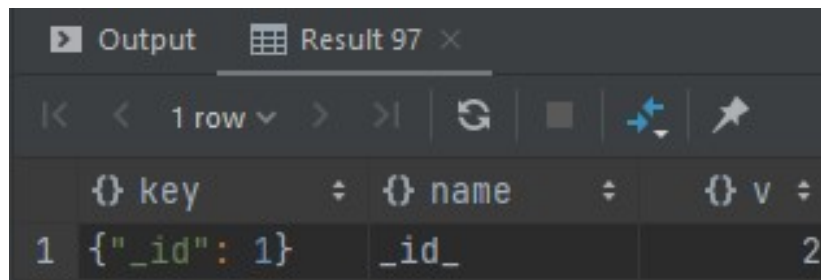


Рис. 14: Индексы.

## 4.2 Проверка производительности с индексами и без

Выполним предпоследний запрос из пункта про запросы и посмотрим на время работы без индекса.

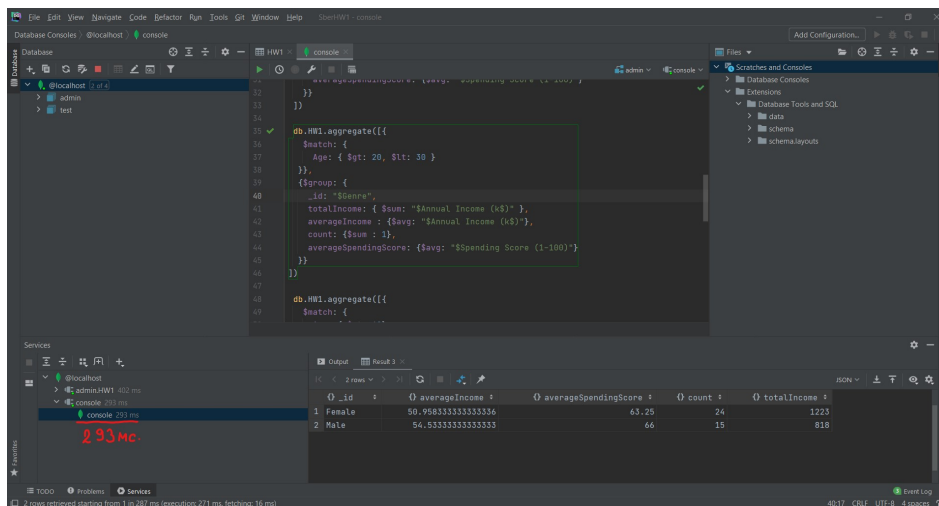


Рис. 15: Время работы запроса без индекса.

Как видим, он отработал примерно за 300мс.

Подумаем какой индекс тут может быть полезен.

Добавим индекса для первого фильтра.

Накинем еще индексов, посмотрим стало ли лучше

Индексы немного ускоряют работу, но совсем немного. Это случается из-за того, что у нас мало данных. Так же при больших данных надо иметь в виду, что индексы накладывают дополнительные расходы по памяти.

Рис. 16: Время работы с индексом.

Рис. 17: Время работы с индексом.