Data Engineering: Assignment 4

MongoDB:

1. List total number of customers living in California?

```
use sakila;

db.customers.find( {} )

/* 1. List total number of customers living in california? */
db.customers.find( { District: "California" } ).pretty()
```

customers 0.090 s 9 Docs 50 p. 1 1 - 9 Tree

Key	Value	Type
(1) 2	{ "First Name" : "PATRICIA", "Last Name" : "JOHNSON" } (9 fields)	Document
(2) 14	{ "First Name" : "BETTY", "Last Name" : "WHITE" } (9 fields)	Document
(3) 51	{ "First Name" : "ALICE", "Last Name" : "STEWART" } (9 fields)	Document
(4) 112	{ "First Name" : "ROSA", "Last Name" : "REYNOLDS" } (9 fields)	Document
(5) 182	{ "First Name" : "RENEE", "Last Name" : "LANE" } (9 fields)	Document
(6) 214	{ "First Name" : "KRISTIN", "Last Name" : "JOHNSTON" } (9 fields)	Document
(7) 269	{ "First Name" : "CASSANDRA", "Last Name" : "WALTERS" } (9 fields)	Document
(8) 420	{ "First Name" : "JACOB", "Last Name" : "LANCE" } (9 fields)	Document

2. List all movies that are rated NC-17

```
/* 2. List all movies that are rated NC-17 */
db.films.find( { Rating: "NC-17" } ).pretty()

/* 3. List the count of movies by category */
db.films.aggregate(
    [
```

films 0.021 s 210 Docs 50 p. 1 1 - 50 Tree

Key	Value	Type
(1) 3	{ Category : "Documentary", Title : "ADAPTATION HOLES" } (10 fields)	Document
(2) 10	{ Category : "Sports", Title : "ALADDIN CALENDAR" } (10 fields)	Document
(3) 14	{ Category : "Classics", Title : "ALICE FANTASIA" } (10 fields)	Document
(4) 16	{ Category : "Foreign", Title : "ALLEY EVOLUTION" } (10 fields)	Document
(5) 15	{ Category : "Foreign", Title : "ALIEN CENTER" } (10 fields)	Document
(6) 27	{ Category : "Sports", Title : "ANONYMOUS HUMAN" } (10 fields)	Document
(7) 31	{ Category : "Family", Title : "APACHE DIVINE" } (10 fields)	Document
(8) 29	{ Category : "Action", Title : "ANTITRUST TOMATOES" } (10 fields)	Document

3. List the count of movies by category

```
/* 3. List the count of movies by category */
db.films.aggregate(
    [
        { '$group': { '_id': "$Category", "Count": { '$sum': 1 } } },
        { '$sort' : {'Count' : -1} }
    ]
)
```

films 0.021 s 210 Docs 50 p. 1 1 - 50 Tree

Key	Value	Type
(1) 3	{ Category : "Documentary", Title : "ADAPTATION HOLES" } (10 fields)	Document
(2) 10	{ Category : "Sports", Title : "ALADDIN CALENDAR" } (10 fields)	Document
(3) 14	{ Category : "Classics", Title : "ALICE FANTASIA" } (10 fields)	Document
(4) 16	{ Category : "Foreign", Title : "ALLEY EVOLUTION" } (10 fields)	Document
(5) 15	{ Category : "Foreign", Title : "ALIEN CENTER" } (10 fields)	Document
(6) 27	{ Category : "Sports", Title : "ANONYMOUS HUMAN" } (10 fields)	Document
(7) 31	{ Category : "Family", Title : "APACHE DIVINE" } (10 fields)	Document
(8) 29	{ Category : "Action", Title : "ANTITRUST TOMATOES" } (10 fields)	Document

4. Find the top 2 movies with movie length greater than 25mins OR which has commentaries as a special feature

```
/* 4. Find the top 2 movies with movie length greater than 25mins OR which has
commentaries as a special feature */
db.films.find( { $or: [ { "Special Features": /Commentaries/}, { Length: { $gt: 25 } } ] } )
.sort({"Length":-1}).limit(2).pretty()
```

films 0.052 s 2 Docs 50 p. 1 1 - 2 Tree

Key	Value	Type
(1) 674	{ Category : "Foreign", Title : "PET HAUNTING" } (10 fields)	Document
(2) 663	{ Category : "Classics", Title : "PATIENT SISTER" } (10 fields)	Document

5. Find the top 10 customers based on number of rentals

```
/* 5. Find the top 10 customers based on number of rentals */
db.customers.aggregate(
      [ { '$unwind': '$Rentals'},
        { '$group': { '_id': "$_id", 'number of rentals': { '$sum': 1 }, 'First Name': {'$first': '$First Name
        { '$sort' : {'number of rentals' : -1} },
        { '$limit' : 10},
      ]).pretty()
```

customers 0.136 s 10 Docs 50 p. 1 1 - 10 Tree

Key	Value	Type
(1) 148	{ "number of rentals" : 46, "First Name" : "ELEANOR", "Last Name" : "HUNT" } (	Document
(2) 526	{ "number of rentals" : 45, "First Name" : "KARL", "Last Name" : "SEAL" } (4 fielc	Document
(3) 144	{ "number of rentals" : 42, "First Name" : "CLARA", "Last Name" : "SHAW" } (4 f	Document
(4) 236	{ "number of rentals" : 42, "First Name" : "MARCIA", "Last Name" : "DEAN" } (4	Document
(5) 75	{ "number of rentals" : 41, "First Name" : "TAMMY", "Last Name" : "SANDERS"	Document
(6) 197	{ "number of rentals" : 40, "First Name" : "SUE", "Last Name" : "PETERS" } (4 fie	Document
(7) 469	{ "number of rentals" : 40, "First Name" : "WESLEY", "Last Name" : "BULL" } (4 f	Document
(8) 137	{ "number of rentals" : 39, "First Name" : "RHONDA", "Last Name" : "KENNEDY	Document

6. Output each stores' inventory counts for comparing. Store1 and store2 are about the same inventory.

```
/* (1) Output each stores's inventory counts for comparing. Store1 and store2 are about the same inventory. */
db.stores.aggregate(
      [{'$unwind':'$Inventory'},
       {'$group':{'_id': '$_id','count':{'$sum':1}}}]).pretty()
```

stores 0.051 s 2 Docs 50 p. 1 1 - 2 Tree

Key	Value	Type
(1) 1	{ count : 2270 }	Document
(2) 2	{ count : 2311 }	Document

7. Outputs number of customers by country. Business will know India and China's market are the most potential.

```
/* (2). Outputs number of customers by country. Business will know India and China's market are the most
potential. */
db.customers.aggregate(
    [
        { '$group': { '_id': "$Country", "Count": { '$sum': 1 } } },
        { '$sort' : {'Count' : -1}},
        { '$match': {'Count': {$gt:30}}}
    ]
).pretty()
```

customers 0.026 s 4 Docs 50 p. 1 1 - 4 Tree

Key	Value	Type
(1) India	{ Count : 60 }	Document
(2) China	{ Count : 53 }	Document
(3) United States	{ Count : 36 }	Document
(4) Japan	{ Count : 31 }	Document

8. Business will find top ten customers spent on film rental. Business can interview them to get some insights.

```
/* (3). Business will find top ten customers spent on film rental. Business can interview them to get some
insights */
db.customers.aggregate(
      [ { '$unwind': '$Rentals'},
        { '$unwind': '$Rentals.Payments'},
        { '$group': { '_id': "$_id", 'payment amounts': { '$sum': '$Rentals.Payments.Amount' }, 'First Name':
        { '$sort' : {'payment amounts' : -1} },
        { '$limit' : 10},
      ]).pretty()
```

customers 0.109 s 10 Docs 50 p. 1 1 - 10 Tree

Key	Value	Type
(2) 148	{ "payment amounts" : 216.5399944782257, "First Name" : "ELEANOR", "Last N	Document
(3) 144	{ "payment amounts" : 195.57999539375305, "First Name" : "CLARA", "Last Na	Document
(4) 137	{ "payment amounts" : 194.60999488830566, "First Name" : "RHONDA", "Last	Document
(5) 178	{ "payment amounts" : 194.60999464988708, "First Name" : "MARION", "Last N	Document
(6) 459	{ "payment amounts" : 186.61999535560608, "First Name" : "TOMMY", "Last N	Document
(7) 469	{ "payment amounts" : 177.59999537467957, "First Name" : "WESLEY", "Last N	Document
(8) 468	{ "payment amounts" : 175.6099956035614, "First Name" : "TIM", "Last Name"	Document
(9) 236	{ "payment amounts" : 175.57999563217163, "First Name" : "MARCIA", "Last N	Document
(10) 181	{ "payment amounts" : 174.65999460220337, "First Name" : "ANA", "Last Nam	Document

9. Find number of actors of each category so that film company knows the demand and supply of actors of each category.

```
/* (4). Find number of actors of each category so that film company knows the demand and supply of actors
of each category */
db.films.aggregate(
     [ { '$unwind': '$Actors'},
       { '$unwind': '$Actors.actorId'},
       { '$group': { '_id': "$Category", 'count': { '$sum': 1 } } },
       { '$sort' : {'count' : -1} },
     ]).pretty()
```

films 0.029 s 16 Docs 50 p. 1 1 - 16 Tree

Key	Value	Type
(1) Sports	{ count : 441 }	Document
(2) Foreign	{ count : 397 }	Document
(3) Documentary	{ count : 385 }	Document
(4) Action	{ count : 363 }	Document
(5) Animation	{ count : 361 }	Document
(6) Drama	{ count : 350 }	Document
(7) Family	{ count : 347 }	Document
(8) Children	{ count : 344 }	Document
(9) New	{ count : 343 }	Document

10. Business will find the top three most popular actors of comedy films.

```
/* (5) Business will find the top three most popular actors of comedy films. */
db.customers.aggregate(
     [ { '$unwind': '$Rentals'},
       { '$lookup': {from: 'films', localField: 'Rentals.filmId', foreignField: '_id', as: 'customers_films'}
       { '$unwind': '$customers_films'},
       { '$unwind': '$customers_films.Actors'},
       { '$match': {'customers_films.Category': 'Comedy'}},
       { '$group': { '_id': "$customers_films.Actors.actorId",
       'First Name': {'$first': '$customers_films.Actors.First name'},
       'Last Name': {'$first': '$customers_films.Actors.Last name'},
       'number of rentals': { '$sum': 1 } } },
       { '$sort' : {'number of rentals' : -1} },
       { '$limit' : 3}
     ]  ).pretty()
```

customers 2.433 s 3 Docs 50 p. 1 1 - 3 Tree

Key	Value	Type
(1) 129	{ "First Name" : "DARYL", "Last Name" : "CRAWFORD", "number of rentals" : 87 } (4 fields)	Document
(2) 58	{ "First Name" : "CHRISTIAN", "Last Name" : "AKROYD", "number of rentals" : 87 } (4 fields)	Document
(3) 37	{ "First Name" : "VAL", "Last Name" : "BOLGER", "number of rentals" : 85 } (4 fields)	Document

Neo4j:

1. Find all Producers that produced the movie When Harry Met Sally

```
MATCH (movie:Movie{title:"When Harry Met Sally"})<-[:PRODUCED]-(producer:Person)
RETURN producer.name
```

	producer.name
1	"Nora Ephron"
2	"Rob Reiner"

Started streaming 2 records after 1 ms and completed after 1 ms.

2. Find directors who have directed more than 2 movies

```

MATCH (director:Person)-[:DIRECTED]->(movie:Movie)
WITH  director, count(movie) AS num_of_movies
WHERE num_of_movies > 2
RETURN director.name as director_name, num_of_movies
```

	director_name	num_of_movies
1	"Lana Wachowski"	5
2	"Andy Wachowski"	5
3	"Rob Reiner"	3
4	"Ron Howard"	3

Started streaming 4 records after 8 ms and completed after 10 ms.

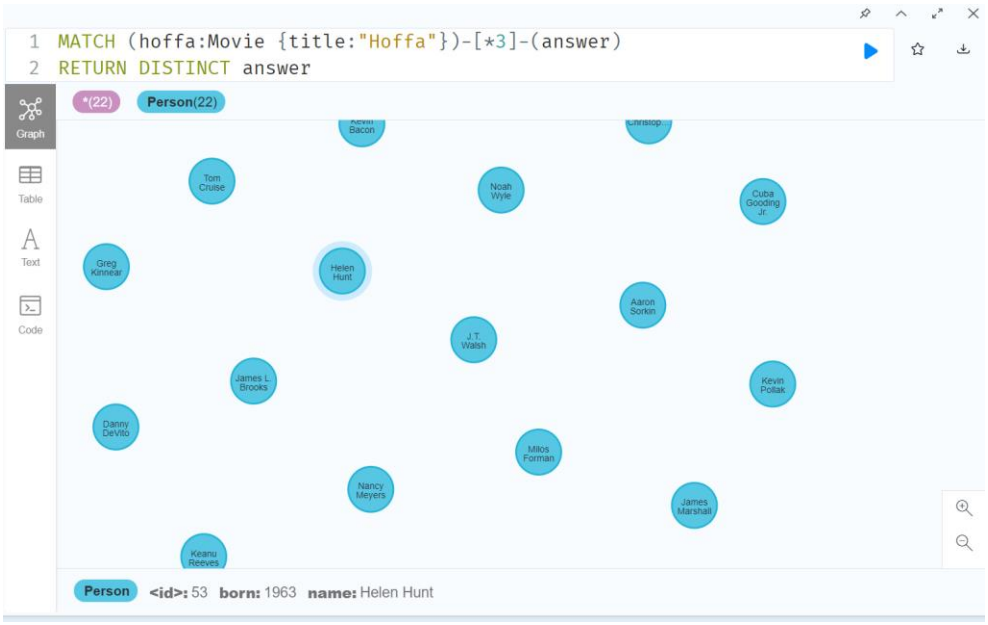
3. Find the actors with 5+ movies, and the movies in which they acted

```
MATCH (actor:Person)-[:ACTED_IN]->(movie:Movie)
WITH actor,count(movie) AS num_of_movies, collect(movie.title) as movies
WHERE num_of_movies > 5
RETURN actor.name, movies, num_of_movies
```

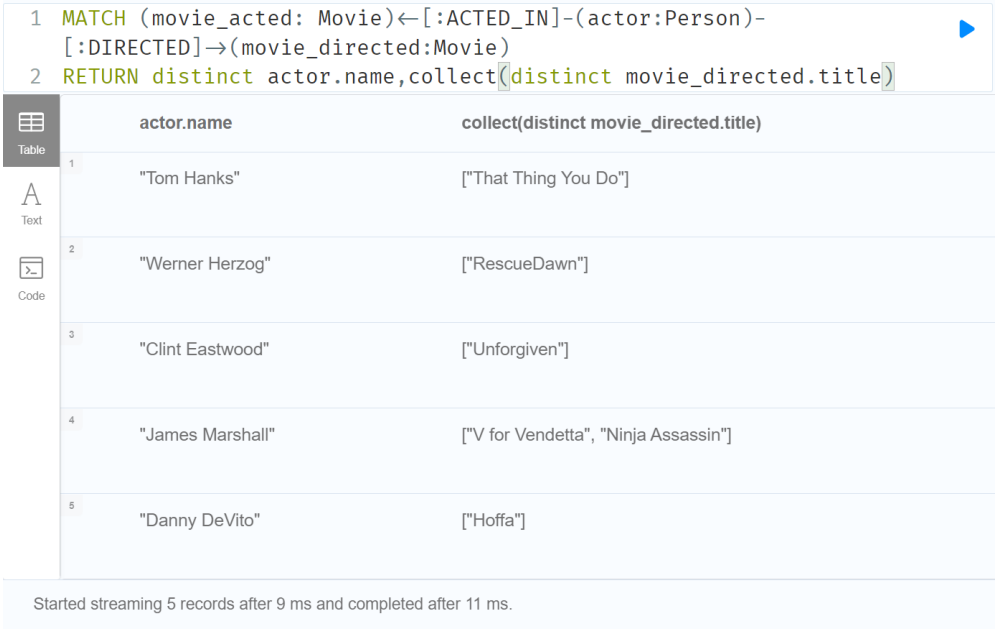
	actor.name	movies
1	"Keanu Reeves"	["The Matrix", "The Matrix Reloaded", "The Matrix Revolutions", "The Devil's Advocate", "The Replacem
2	"Tom Hanks"	["You've Got Mail", "Sleepless in Seattle", "Joe Versus the Volcano", "That Thing You Do", "Cloud Atlas"

Started streaming 2 records after 13 ms and completed after 16 ms.

4. Movies and actors exactly 3 "hops" away from the movie Hoffa



5. Find all actors who have also directed movies and the movies that they directed



6. Find the writers and the taglines they write for the movies.

```
MATCH (writer:Person)-[:WROTE]→(movie:Movie)
RETURN writer.name AS writer_name, movie.tagline AS movie_tagline
```

Table | Text | Code

	writer_name	movie_tagline
1	"Aaron Sorkin"	"In the heart of the nation's capital, in a courthouse of the U.S. government, one man wi
2	"Jim Cash"	"I feel the need, the need for speed."
3	"Cameron Crowe"	"The rest of his life begins now."
4	"Nora Ephron"	"At odds in life... in love on-line."
5	"David Mitchell"	"Everything is connected"
6	"Andy Wachowski"	"Freedom! Forever!"

Started streaming 10 records in less than 1 ms and completed after 1 ms.

7. Find each movie's rating score and its reviewer.

```
MATCH (reviewer:Person)-[review:REVIEWED]→(movie:Movie)
RETURN movie.title AS title,review.rating AS rating,reviewer.name
AS reviewer
ORDER BY rating DESC

```

Table | Text | Code

	title	rating	reviewer
1	"The Replacements"	100	"James Thompson"
2	"Cloud Atlas"	95	"Jessica Thompson"
3	"Jerry Maguire"	92	"Jessica Thompson"
4	"Unforgiven"	85	"Jessica Thompson"
5	"The Da Vinci Code"	68	"Jessica Thompson"
6	"The Replacements"	65	"Jessica Thompson"
7	"The Da Vinci Code"	65	"James Thompson"

Started streaming 9 records after 1 ms and completed after 2 ms.

8. Find rating score of movies' ratings above 80 and its director.

```
MATCH (director:Person)-[:DIRECTED]→(movie_directed:Movie)
MATCH (reviewer:Person)-[review:REVIEWED]→(movie_reviewed:Movie)
WHERE review.rating > 80 AND movie_directed = movie_reviewed
RETURN collect(distinct director.name) AS director,
movie_reviewed.title AS title,review.rating AS rating
ORDER BY rating DESC
```

Table | Text | Code

	director	title	rating
1	["Howard Deutch"]	"The Replacements"	100
2	["Tom Tykwer", "Lana Wachowski", "Andy Wachowski"]	"Cloud Atlas"	95
3	["Cameron Crowe"]	"Jerry Maguire"	92
4	["Clint Eastwood"]	"Unforgiven"	85

Started streaming 4 records after 1 ms and completed after 3 ms.

9. Find movies which have 5 or more actors.

```
MATCH (actor:Person)-[:ACTED_IN]→(movie:Movie)
WITH count(actor) AS num_of_actors, collect(actor.name) as actors,
movie
WHERE num_of_actors ⩾ 5
RETURN  movie.title,num_of_actors,actors
ORDER BY num_of_actors DESC
```

Table | Text | Code

	movie.title	num_of_actors	actors
7	"You've Got Mail"	6	["Tom Hanks", "Parker Posey", "Greg Kinnear", "Meg Ryan",
8	"Sleepless in Seattle"	6	["Meg Ryan", "Victor Garber", "Tom Hanks", "Bill Pullman", "F
9	"A League of Their Own"	6	["Tom Hanks", "Madonna", "Rosie O'Donnell", "Geena Davis"
10	"Frost/Nixon"	5	["Sam Rockwell", "Michael Sheen", "Frank Langella", "Oliver
11	"Apollo 13"	5	["Tom Hanks", "Ed Harris", "Gary Sinise", "Kevin Bacon", "Bill
12	"The Matrix"	5	["Emil Eifrem", "Hugo Weaving", "Laurence Fishburne", "Carr

Started streaming 14 records in less than 1 ms and completed after 2 ms.

10. Find movies that are directed and produced by the same person.

```
1 MATCH (director:Person)-[:DIRECTED]→(movie:Movie)←[:PRODUCED]-
  (producer:Person)
2 WHERE director.name = producer.name
3 RETURN movie.title AS title,director.name AS
  director_name,producer.name AS producer_name
```

Table | Text | Code

	title	director_name	producer_name
1	"Jerry Maguire"	"Cameron Crowe"	"Cameron Crowe"
2	"When Harry Met Sally"	"Rob Reiner"	"Rob Reiner"
3	"Something's Gotta Give"	"Nancy Meyers"	"Nancy Meyers"

Started streaming 3 records in less than 1 ms and completed after 1 ms.