1.

erae97
sakila.actor

Table Details

Engine:	**InnoDB**
Row format:	**Dynamic**
Column count:	**4**
Table rows:	**200**
AVG row length:	**81**
Data length:	**16.0 KiB**
Index length:	**0.0 bytes**
Max data length:	**0.0 bytes**
Data free:	**0.0 bytes**
Table size (estimate):	**16.0 KiB**
File format:	
Data path:	**C:\ProgramData\MySQL\MySQL Server 8.0\Data\sakila\actor.ibd**
Update time:	**2019-10-29 12:03:37**
Create time:	**2019-10-29 12:02:13**
Auto increment:	**201**
Table collation:	**utf8_general_ci**
Create options:	
Comment:	

2.

```sql
1 • SELECT
2     CONCAT(city.city, ', ', country.country) store,
3     CONCAT(staff.first_name, ' ', staff.last_name) manager,
4     SUM(payment.amount) total_sales
5     FROM country
6     INNER JOIN city ON city.country_id = country.country_id
7     INNER JOIN address ON address.city_id = city.city_id
8     INNER JOIN store ON store.address_id = address.address_id
9     INNER JOIN staff ON staff.staff_id = store.manager_staff_id
10    INNER JOIN inventory ON inventory.store_id = store.store_id
11    INNER JOIN rental ON rental.inventory_id = inventory.inventory_id
12    INNER JOIN payment ON payment.staff_id = staff.staff_id AND payment.rental_id = rental.rental_id
13    WHERE MONTH(rental.rental_date) = 2
14
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: IA

store	manager	total_sales
Lethbridge, Canada	Mike Hillyer	238.13

3.

```sql
1 •  SELECT
2        category.name, SUM(payment.amount) total_sales
3        From category
4        INNER JOIN film_category ON film_category.category_id = category.category_id
5        INNER JOIN film ON film_category.film_id = film.film_id
6        INNER JOIN inventory ON inventory.film_id = film.film_id
7        INNER JOIN rental ON rental.inventory_id = inventory.inventory_id
8        INNER JOIN payment ON payment.rental_id = rental.rental_id
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

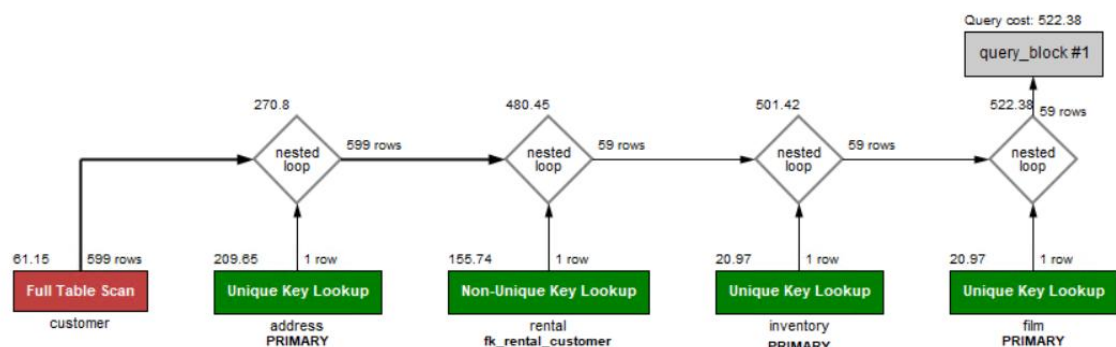| name	total_sales
Action	67406.56

5.

Timing (as measured at client side):
Execution time: 0:00:0.03100000

Timing (as measured by the server):
Execution time: 0:00:0.03682250
Table lock wait time: 0:00:0.00032100

Result Grid | Filter Rows:

@@innodb_buffer_pool_size/1024/1024/1024
▸ 0.007812500000

Query cost: 522.38
query_block #1

Indexes Used: return date is null and the current date is greater than the rental date + rental duration
Join Order: customer -> address -> rental -> inventory -> film
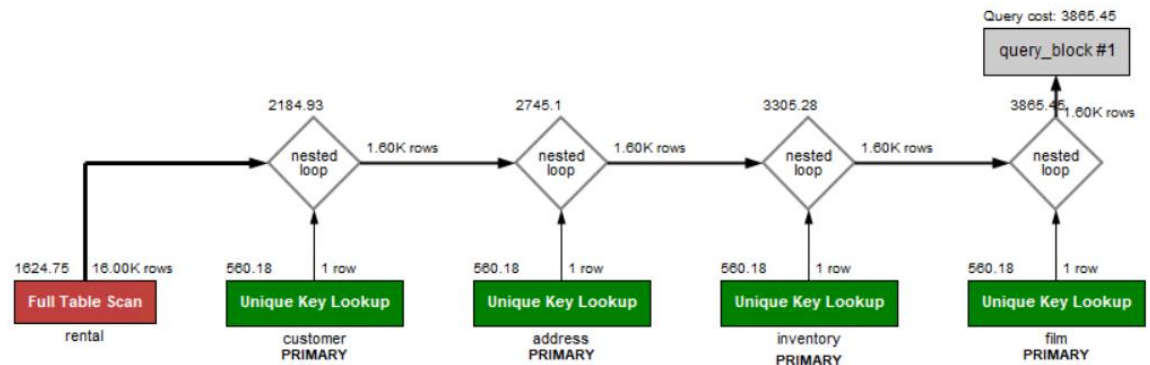
6.

Query cost: 3865.45

query_block #1

2184.93	2745.1	3305.28	3865.45
nested loop	nested loop	nested loop	nested loop

1.60K rows   1.60K rows   1.60K rows   1.60K rows

1624.75	16.00K rows	560.18	1 row	560.18	1 row	560.18	1 row	560.18	1 row
Full Table Scan		Unique Key Lookup		Unique Key Lookup		Unique Key Lookup		Unique Key Lookup	

rental

customer
PRIMARY

address
PRIMARY

inventory
PRIMARY

film
PRIMARY

7.

Query cost: 103.00

query_block #1

103.0   1000 rows

Full Table Scan

film

SELECT * FROM film WHERE film_id < 40 AND title="APOLLO TEEN";

Query cost: 8.07

query_block #1

8.07   39 rows

Index Range Scan

film
PRIMARY

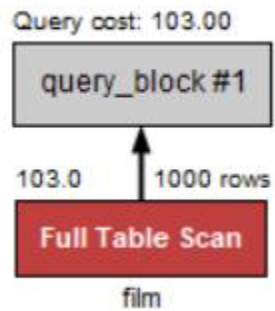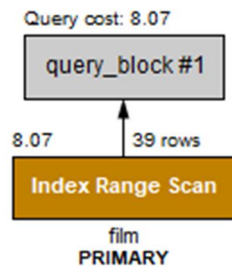Yes the index that I created gets used

8.

Timing (as measured at client side):
Execution time: 0:00:0.00000000

Timing (as measured by the server):
Execution time: 0:00:0.00305380
Table lock wait time: 0:00:0.00028400