```
%sql
USE sakila_dlh;
SHOW TABLES
```

Table

	database	tableName	isTemporary
1	sakila_dlh	customer	false
2	sakila_dlh	dim_date	false
3	sakila_dlh	fact_orders	false
4	sakila_dlh	film	false
5	sakila_dlh	inventory	false
6	sakila_dlh	rental	false

9 rows

```
%sql
USE DATABASE sakila_dlh;

CREATE OR REPLACE TABLE sakila_dlh.inventory
COMMENT "Payment Table"
LOCATION "dbfs:/FileStore/ds2002-capstone/sakila_dlh/payment"
AS SELECT * FROM view_inventory
```

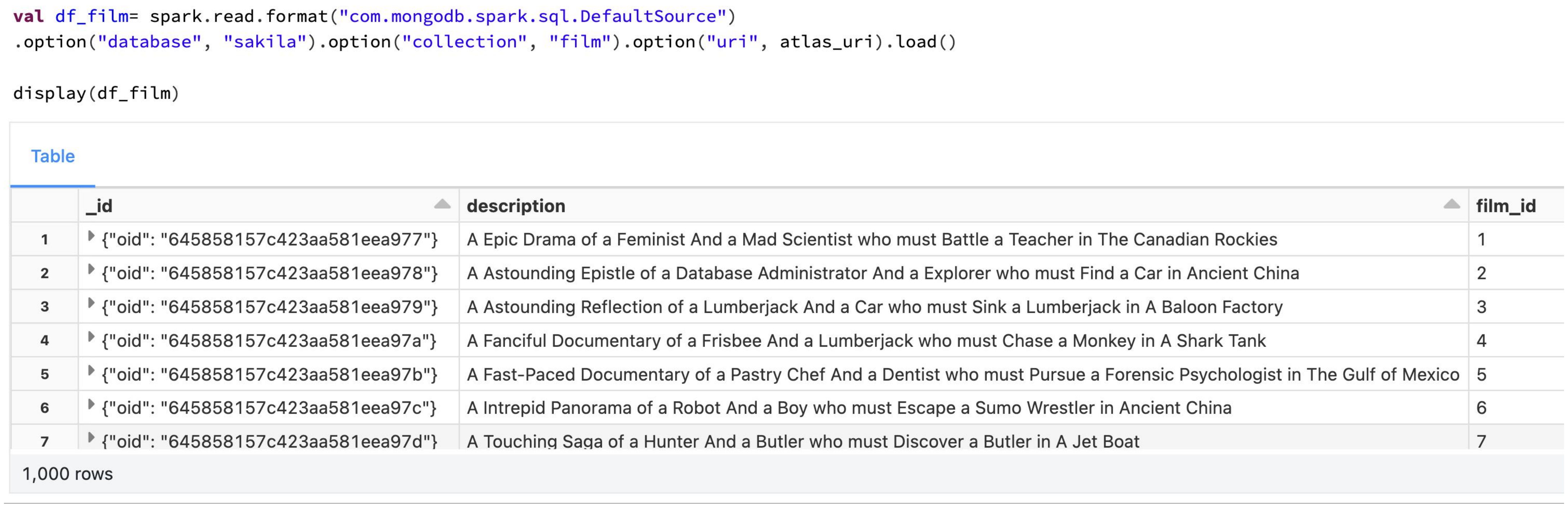
```
Query returned no results
```

```
%sql
SELECT * FROM sakila_dlh.inventory LIMIT 5
```

Table

	inventory_id	film_id	store_id	last_update
1	1	1	1	2006-02-15T05:09:17.000+0000
2	2	1	1	2006-02-15T05:09:17.000+0000
3	3	1	1	2006-02-15T05:09:17.000+0000
4	4	1	1	2006-02-15T05:09:17.000+0000
5	5	1	2	2006-02-15T05:09:17.000+0000

5 rows



```
val df_film= spark.read.format("com.mongodb.spark.sql.DefaultSource")
.option("database", "sakila").option("collection", "film").option("uri", atlas_uri).load()

display(df_film)
```

Table

	_id	description	film_id
1	{"oid": "645858157c423aa581eea977"}	A Epic Drama of a Feminist And a Mad Scientist who must Battle a Teacher in The Canadian Rockies	1
2	{"oid": "645858157c423aa581eea978"}	A Astounding Epistle of a Database Administrator And a Explorer who must Find a Car in Ancient China	2
3	{"oid": "645858157c423aa581eea979"}	A Astounding Reflection of a Lumberjack And a Car who must Sink a Lumberjack in A Baloon Factory	3
4	{"oid": "645858157c423aa581eea97a"}	A Fanciful Documentary of a Frisbee And a Lumberjack who must Chase a Monkey in A Shark Tank	4
5	{"oid": "645858157c423aa581eea97b"}	A Fast-Paced Documentary of a Pastry Chef And a Dentist who must Pursue a Forensic Psychologist in The Gulf of Mexico	5
6	{"oid": "645858157c423aa581eea97c"}	A Intrepid Panorama of a Robot And a Boy who must Escape a Sumo Wrestler in Ancient China	6
7	{"oid": "645858157c423aa581eea97d"}	A Touching Saga of a Hunter And a Butler who must Discover a Butler in A Jet Boat	7

1,000 rows

```
#fetch the rentals table

rental_csv = f"{batch_dir}/sakila_rental.csv"

df_rental = spark.read.format('csv').options(header = 'true', inferSchema = 'true').load(rental_csv)
display(df_rental)
```

Table

	rental_id	rental_date	inventory_id	customer_id	return_date	staff_id	last_update
1	1	2005-05-24T22:53:30.000+0000	367	130	2005-05-26T22:04:30.000+0000	1	2006-02-15T21:30:53.000+0000
2	2	2005-05-24T22:54:33.000+0000	1525	459	2005-05-28T19:40:33.000+0000	1	2006-02-15T21:30:53.000+0000
3	3	2005-05-24T23:03:39.000+0000	1711	408	2005-06-01T22:12:39.000+0000	1	2006-02-15T21:30:53.000+0000
4	4	2005-05-24T23:04:41.000+0000	2452	333	2005-06-03T01:43:41.000+0000	2	2006-02-15T21:30:53.000+0000
5	5	2005-05-24T23:05:21.000+0000	2079	222	2005-06-02T04:33:21.000+0000	1	2006-02-15T21:30:53.000+0000
6	6	2005-05-24T23:08:07.000+0000	2792	549	2005-05-27T01:32:07.000+0000	1	2006-02-15T21:30:53.000+0000

10,000 rows | Truncated data

Table

	rental_id	rental_date	inventory_id	customer_id	return_date	staff_id	last_update
1	1	2005-05-24T22:53:30.000+0000	367	130	2005-05-26T22:04:30.000+0000	1	2006-02-15T21:30:53.000+0000
2	2	2005-05-24T22:54:33.000+0000	1525	459	2005-05-28T19:40:33.000+0000	1	2006-02-15T21:30:53.000+0000
3	3	2005-05-24T23:03:39.000+0000	1711	408	2005-06-01T22:12:39.000+0000	1	2006-02-15T21:30:53.000+0000
4	4	2005-05-24T23:04:41.000+0000	2452	333	2005-06-03T01:43:41.000+0000	2	2006-02-15T21:30:53.000+0000
5	5	2005-05-24T23:05:21.000+0000	2079	222	2005-06-02T04:33:21.000+0000	1	2006-02-15T21:30:53.000+0000

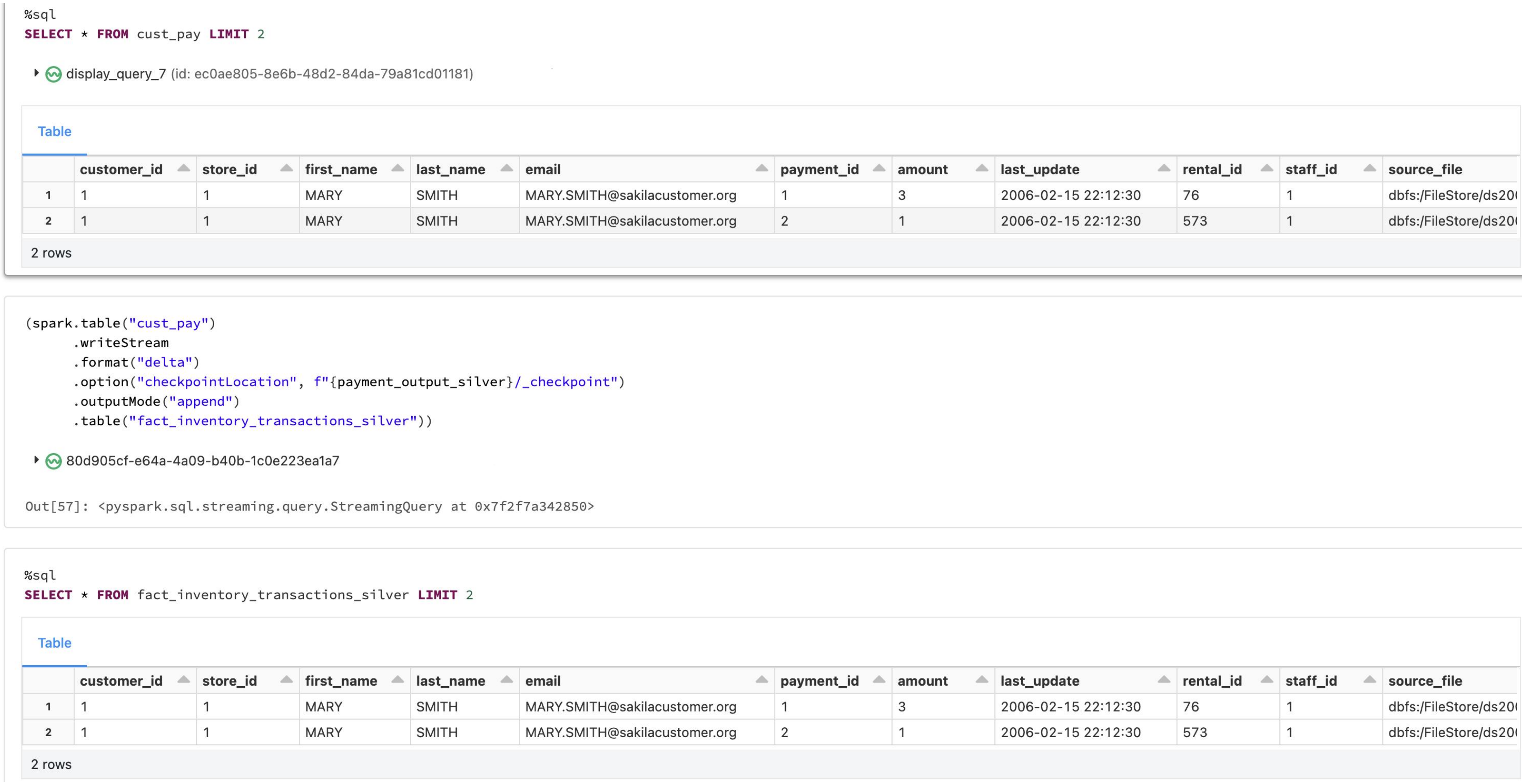
5 rows

```
#fetch the fact orders table

fact_orders_csv = f"{batch_dir}/sakila_fact_table.csv"

df_fact_orders = spark.read.format('csv').options(header = 'true', inferSchema = 'true').load(fact_orders_csv)
display(df_fact_orders)
```

Table

	order_key	customer_key	rental_key	film_key	rental_date_key	inventory_key	payment_date_key	first_name	last_name	address
1	1	1	1422	228	20050615	1021	20050615	MARY	SMITH	1913 Hanoi Way
2	2	362	1429	895	20050615	4116	20050615	NICHOLAS	BARFIELD	1163 London Parkway
3	3	341	1318	606	20050615	2760	20050615	PETER	MENARD	1217 Konotop Avenue
4	4	8	1305	42	20050615	187	20050615	SUSAN	WILSON	478 Joliet Way
5	5	410	1514	645	20050615	2937	20050615	CURTIS	IRBY	432 Garden Grove Street
6	6	14	1360	893	20050615	4107	20050615	BETTY	WHITE	770 Bydgoszcz Avenue

4,011 rows



```
%sql
SELECT * FROM cust_pay LIMIT 2
```

▸ display_query_7 (id: ec0ae805-8e6b-48d2-84da-79a81cd01181)

Table

	customer_id	store_id	first_name	last_name	email	payment_id	amount	last_update	rental_id	staff_id	source_file
1	1	1	MARY	SMITH	MARY.SMITH@sakilacustomer.org	1	3	2006-02-15 22:12:30	76	1	dbfs:/FileStore/ds20
2	1	1	MARY	SMITH	MARY.SMITH@sakilacustomer.org	2	1	2006-02-15 22:12:30	573	1	dbfs:/FileStore/ds20

2 rows

```
(spark.table("cust_pay")
      .writeStream
      .format("delta")
      .option("checkpointLocation", f"{payment_output_silver}/_checkpoint")
      .outputMode("append")
      .table("fact_inventory_transactions_silver"))
```

▸ 80d905cf-e64a-4a09-b40b-1c0e223ea1a7

```
Out[57]: <pyspark.sql.streaming.query.StreamingQuery at 0x7f2f7a342850>
```

```
%sql
SELECT * FROM fact_inventory_transactions_silver LIMIT 2
```

Table

	customer_id	store_id	first_name	last_name	email	payment_id	amount	last_update	rental_id	staff_id	source_file
1	1	1	MARY	SMITH	MARY.SMITH@sakilacustomer.org	1	3	2006-02-15 22:12:30	76	1	dbfs:/FileStore/ds20
2	1	1	MARY	SMITH	MARY.SMITH@sakilacustomer.org	2	1	2006-02-15 22:12:30	573	1	dbfs:/FileStore/ds20

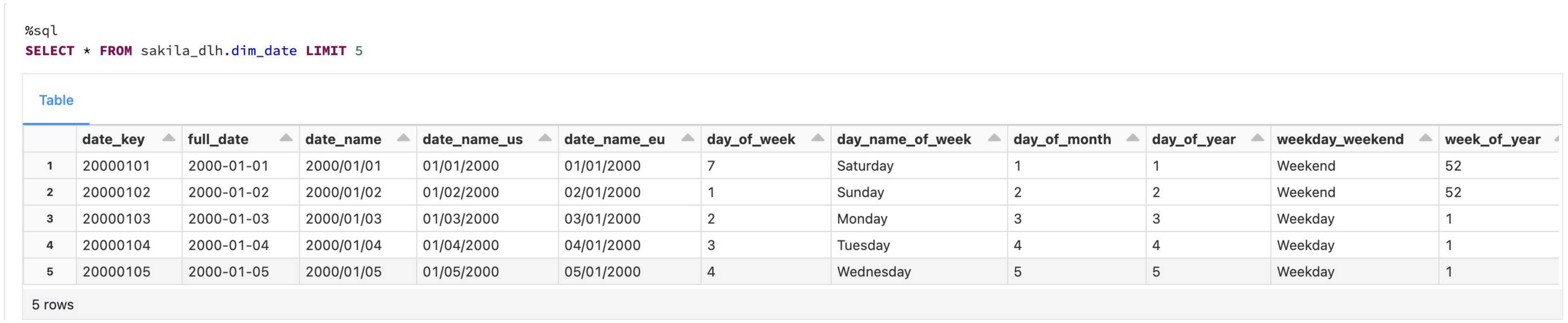
2 rows

```
%sql
SELECT release_year AS movie_release_year
    , rating AS movie_rating
    , AVG(rental_rate) AS average_rental_rate
FROM sakila_dlh.fact_inventory_transactions_silver
GROUP BY movie_release_year, movie_rating
ORDER BY movie_release_year DESC
```

Table

	movie_release_year	movie_rating	average_rental_rate
1	2006	G	2.846020942408229
2	2006	NC-17	3.054516129032097
3	2006	PG-13	3.0067616493461324
4	2006	R	2.8242749529188647
5	2006	PG	2.943418482343942

5 rows



```
%sql
SELECT * FROM sakila_dlh.dim_date LIMIT 5
```

Table

	date_key	full_date	date_name	date_name_us	date_name_eu	day_of_week	day_name_of_week	day_of_month	day_of_year	weekday_weekend	week_of_year
1	20000101	2000-01-01	2000/01/01	01/01/2000	01/01/2000	7	Saturday	1	1	Weekend	52
2	20000102	2000-01-02	2000/01/02	01/02/2000	02/01/2000	1	Sunday	2	2	Weekend	52
3	20000103	2000-01-03	2000/01/03	01/03/2000	03/01/2000	2	Monday	3	3	Weekday	1
4	20000104	2000-01-04	2000/01/04	01/04/2000	04/01/2000	3	Tuesday	4	4	Weekday	1
5	20000105	2000-01-05	2000/01/05	01/05/2000	05/01/2000	4	Wednesday	5	5	Weekday	1

5 rows