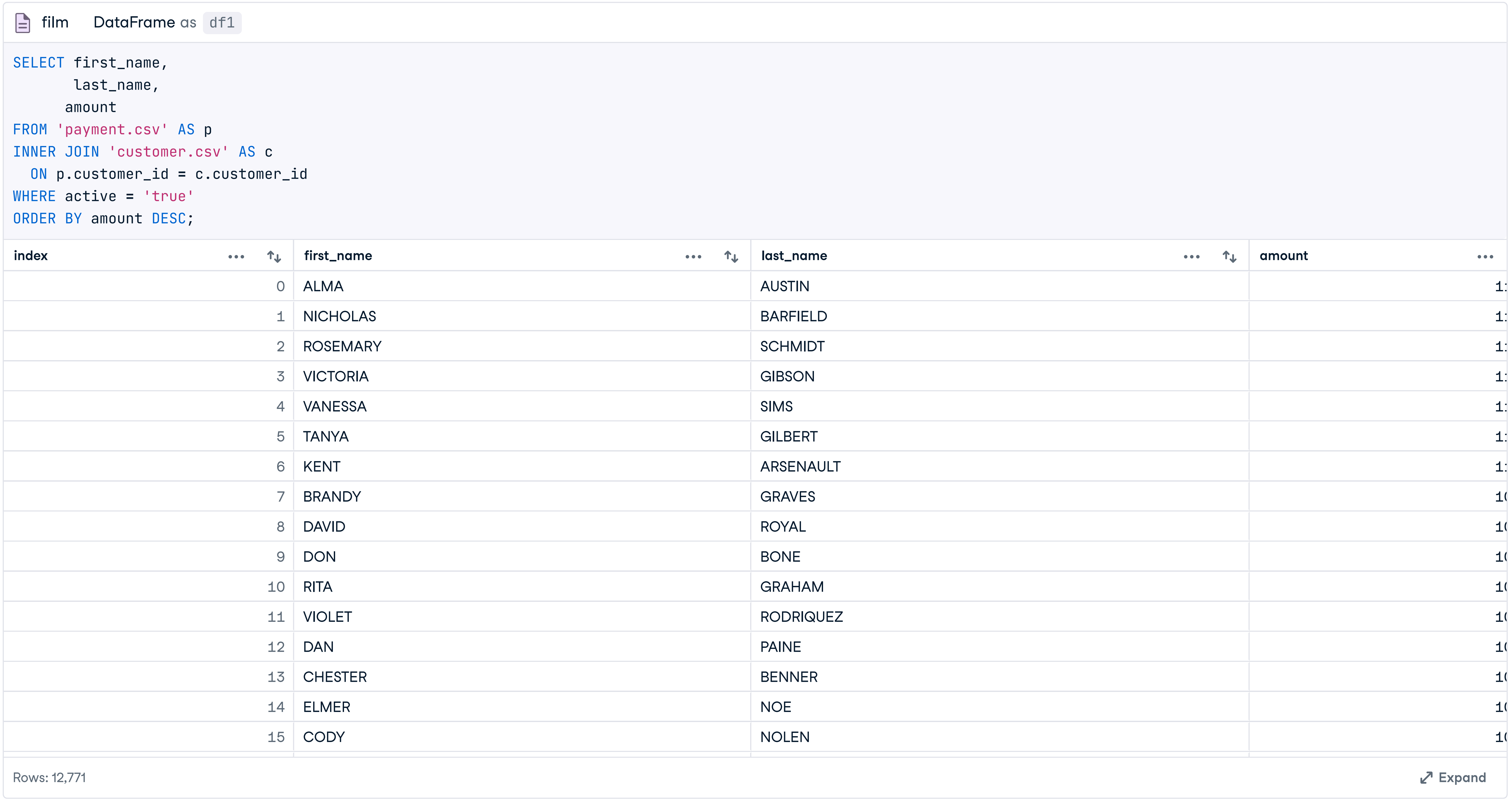
film DataFrame as df

```
SELECT title, description
FROM 'film.csv' AS f
INNER JOIN 'language.csv' AS l
  ON f.language_id = l.language_id
WHERE l.name IN ('Italian', 'French')
  AND f.release_year = 2005;
```

index	title	description
0	ALI FOREVER	A Action-Packed Drama of a Dentist And a Crocodile who must Battle a Feminist in The Canadian Rockies
1	BEHAVIOR RUNAWAY	A Unbelieveable Drama of a Student And a Husband who must Outrace a Sumo Wrestler in Berlin
2	BIRCH ANTITRUST	A Fanciful Panorama of a Husband And a Pioneer who must Outgun a Dog in A Baloon
3	BOWFINGER GABLES	A Fast-Paced Yarn of a Waitress And a Composer who must Outgun a Dentist in California
4	BROTHERHOOD BLANKET	A Fateful Character Study of a Butler And a Technical Writer who must Sink a Astronaut in Ancient Japan
5	CHEAPER CLYDE	A Emotional Character Study of a Pioneer And a Girl who must Discover a Dog in Ancient Japan
6	COLDBLOODED DARLING	A Brilliant Panorama of a Dentist And a Moose who must Find a Student in The Gulf of Mexico
7	CONVERSATION DOWNHILL	A Taut Character Study of a Husband And a Waitress who must Sink a Squirrel in A MySQL Convention
8	DARES PLUTO	A Fateful Story of a Robot And a Dentist who must Defeat a Astronaut in New Orleans
9	DARKNESS WAR	A Touching Documentary of a Husband And a Hunter who must Escape a Boy in The Sahara Desert
10	DOZEN LION	A Taut Drama of a Cat And a Girl who must Defeat a Frisbee in The Canadian Rockies
11	DREAM PICKUP	A Epic Display of a Car And a Composer who must Overcome a Forensic Psychologist in The Gulf of Mexico
12	DRIFTER COMMANDMENTS	A Epic Reflection of a Womanizer And a Squirrel who must Discover a Husband in A Jet Boat
13	ENDING CROWDS	A Unbelieveable Display of a Dentist And a Madman who must Vanquish a Squirrel in Berlin
14	FACTORY DRAGON	A Action-Packed Saga of a Teacher And a Frisbee who must Escape a Lumberjack in The Sahara Desert
15	GHOST GROUNDHOG	A Brilliant Panorama of a Madman And a Composer who must Succumb a Car in Ancient India

Rows: 44

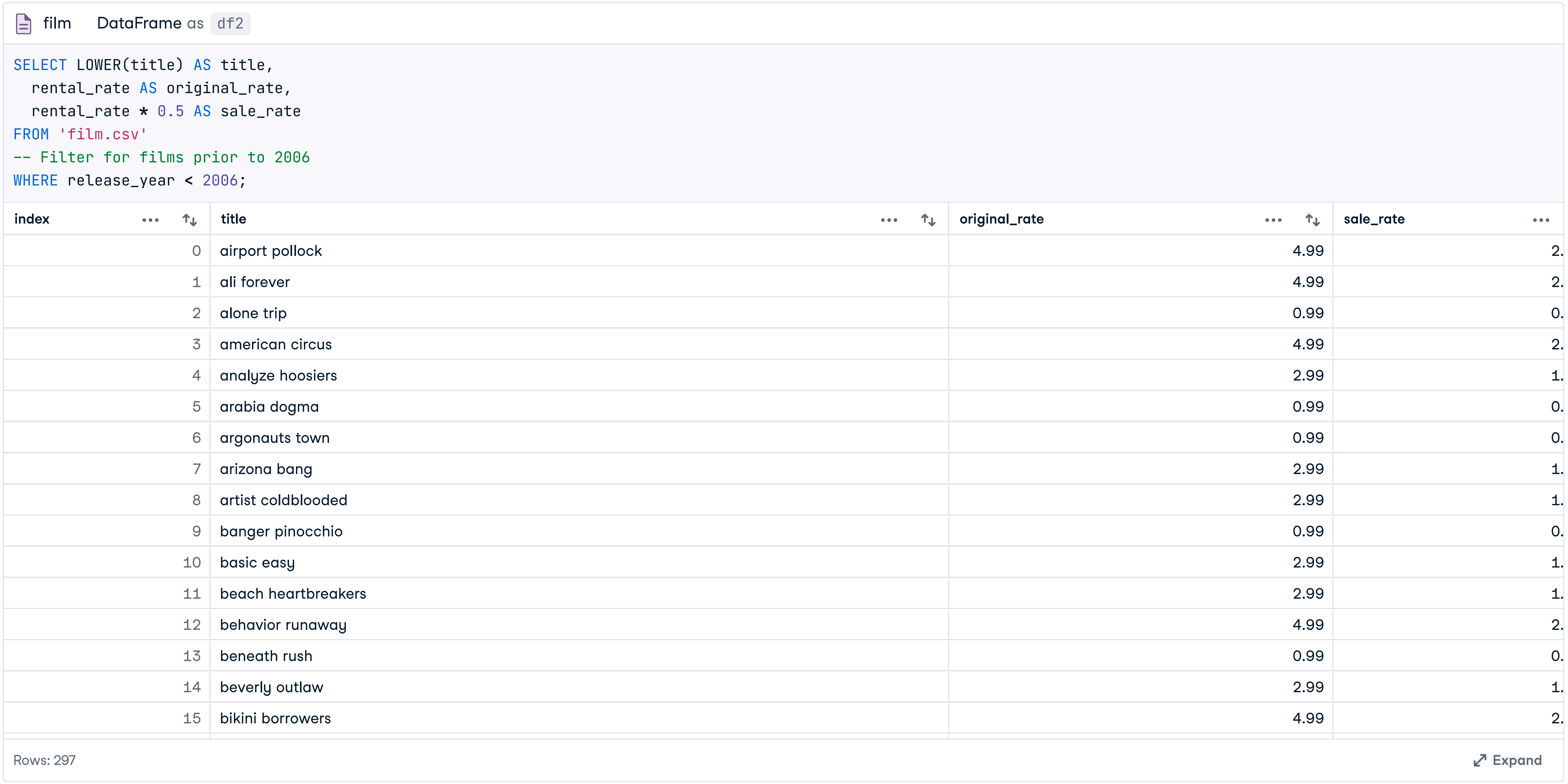
Expand



```
SELECT first_name,
       last_name,
      amount
FROM 'payment.csv' AS p
INNER JOIN 'customer.csv' AS c
  ON p.customer_id = c.customer_id
WHERE active = 'true'
ORDER BY amount DESC;
```

index	first_name	last_name	amount
0	ALMA	AUSTIN	11
1	NICHOLAS	BARFIELD	11
2	ROSEMARY	SCHMIDT	11
3	VICTORIA	GIBSON	11
4	VANESSA	SIMS	11
5	TANYA	GILBERT	11
6	KENT	ARSENAULT	11
7	BRANDY	GRAVES	10
8	DAVID	ROYAL	10
9	DON	BONE	10
10	RITA	GRAHAM	10
11	VIOLET	RODRIQUEZ	10
12	DAN	PAINE	10
13	CHESTER	BENNER	10
14	ELMER	NOE	10
15	CODY	NOLEN	10

Rows: 12,771

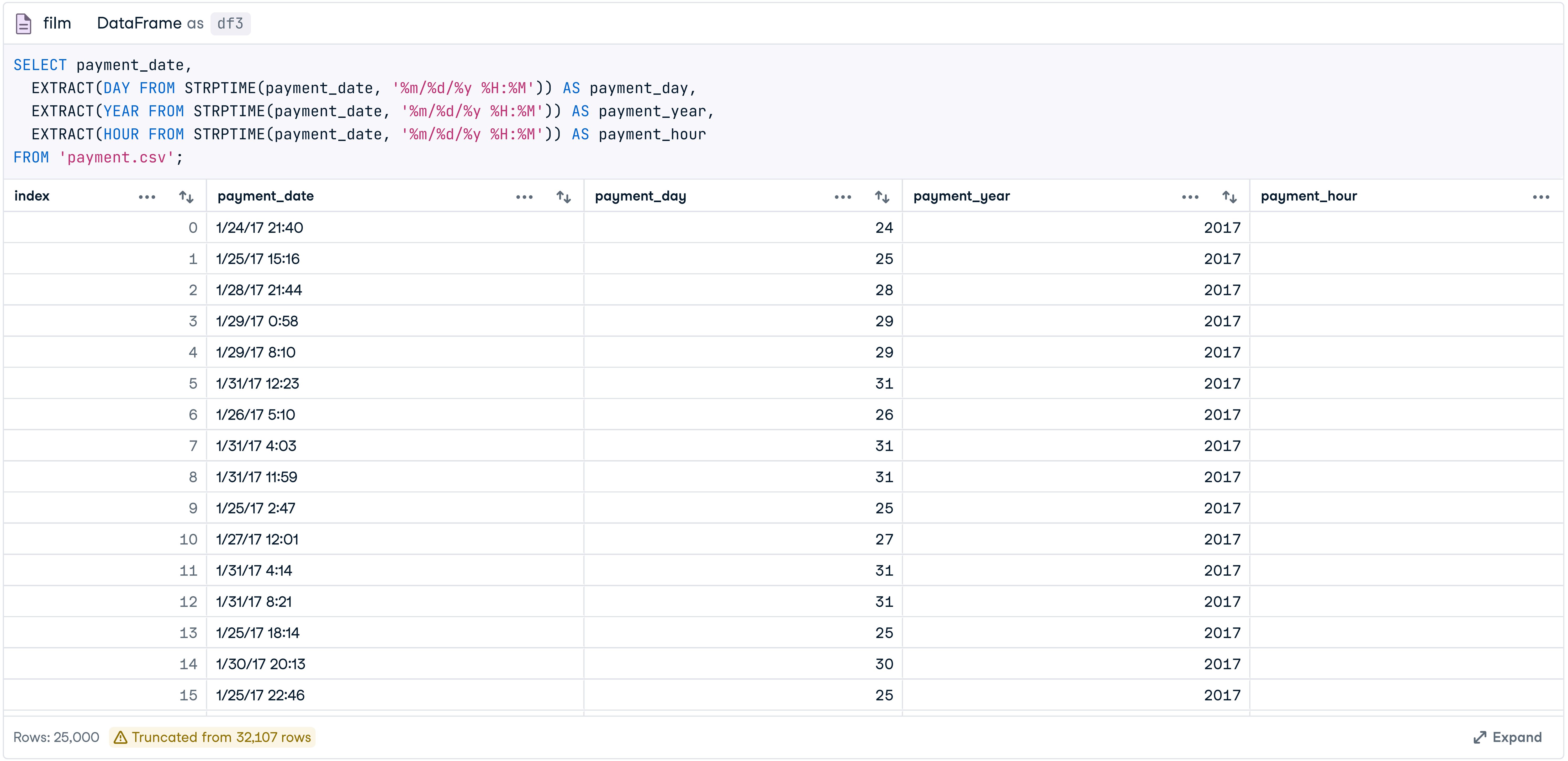

film DataFrame as df2

```
SELECT LOWER(title) AS title,
  rental_rate AS original_rate,
  rental_rate * 0.5 AS sale_rate
FROM 'film.csv'
-- Filter for films prior to 2006
WHERE release_year < 2006;
```

index	title	original_rate	sale_rate
0	airport pollock	4.99	2.
1	ali forever	4.99	2.
2	alone trip	0.99	0.
3	american circus	4.99	2.
4	analyze hoosiers	2.99	1.
5	arabia dogma	0.99	0.
6	argonauts town	0.99	0.
7	arizona bang	2.99	1.
8	artist coldblooded	2.99	1.
9	banger pinocchio	0.99	0.
10	basic easy	2.99	1.
11	beach heartbreakers	2.99	1.
12	behavior runaway	4.99	2.
13	beneath rush	0.99	0.
14	beverly outlaw	2.99	1.
15	bikini borrowers	4.99	2.

Rows: 297

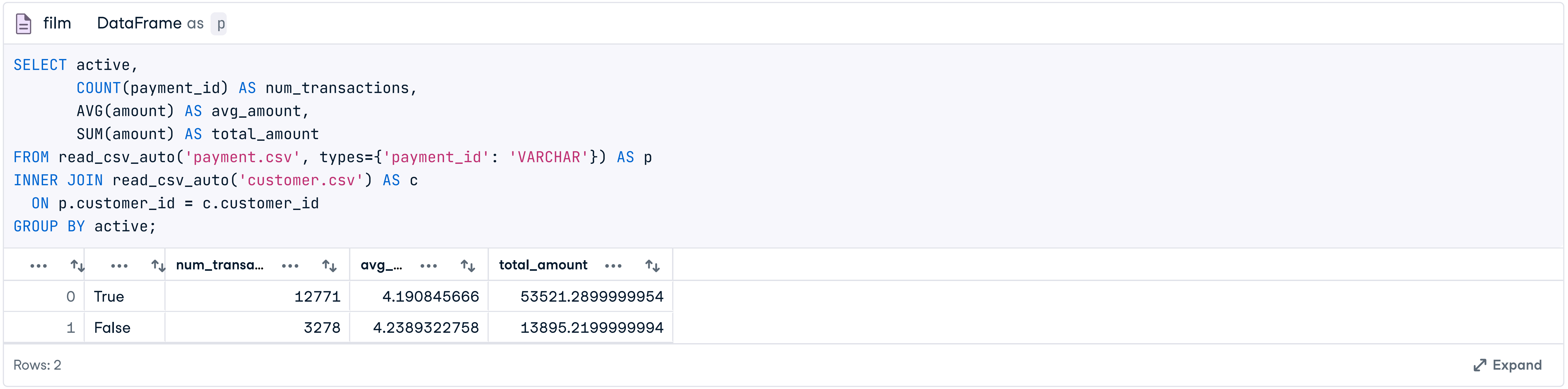
Expand


film DataFrame as df3

```sql
SELECT payment_date,
  EXTRACT(DAY FROM STRPTIME(payment_date, '%m/%d/%y %H:%M')) AS payment_day,
  EXTRACT(YEAR FROM STRPTIME(payment_date, '%m/%d/%y %H:%M')) AS payment_year,
  EXTRACT(HOUR FROM STRPTIME(payment_date, '%m/%d/%y %H:%M')) AS payment_hour
FROM 'payment.csv';
```

index	payment_date	payment_day	payment_year	payment_hour
0	1/24/17 21:40	24	2017	
1	1/25/17 15:16	25	2017	
2	1/28/17 21:44	28	2017	
3	1/29/17 0:58	29	2017	
4	1/29/17 8:10	29	2017	
5	1/31/17 12:23	31	2017	
6	1/26/17 5:10	26	2017	
7	1/31/17 4:03	31	2017	
8	1/31/17 11:59	31	2017	
9	1/25/17 2:47	25	2017	
10	1/27/17 12:01	27	2017	
11	1/31/17 4:14	31	2017	
12	1/31/17 8:21	31	2017	
13	1/25/17 18:14	25	2017	
14	1/30/17 20:13	30	2017	
15	1/25/17 22:46	25	2017	

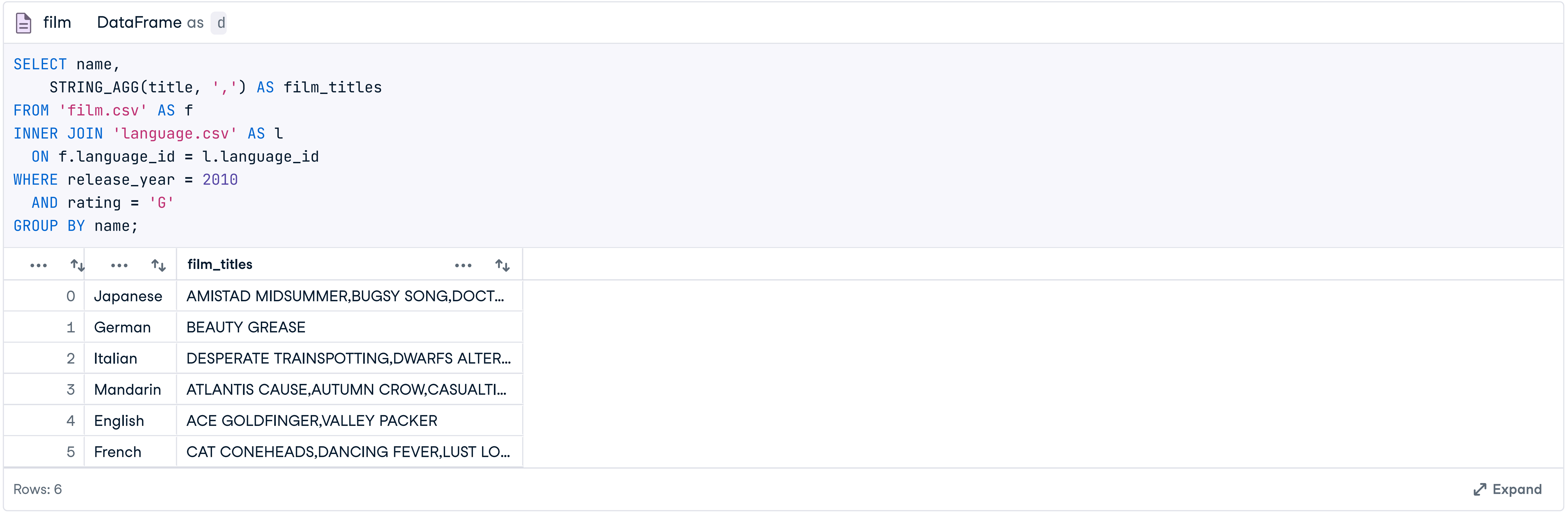
Rows: 25,000 ⚠ Truncated from 32,107 rows


film DataFrame as p

```sql
SELECT active,
       COUNT(payment_id) AS num_transactions,
       AVG(amount) AS avg_amount,
       SUM(amount) AS total_amount
FROM read_csv_auto('payment.csv', types={'payment_id': 'VARCHAR'}) AS p
INNER JOIN read_csv_auto('customer.csv') AS c
  ON p.customer_id = c.customer_id
GROUP BY active;
```

		num_transa…	avg_…	total_amount
0	True	12771	4.190845666	53521.2899999954
1	False	3278	4.2389322758	13895.2199999994

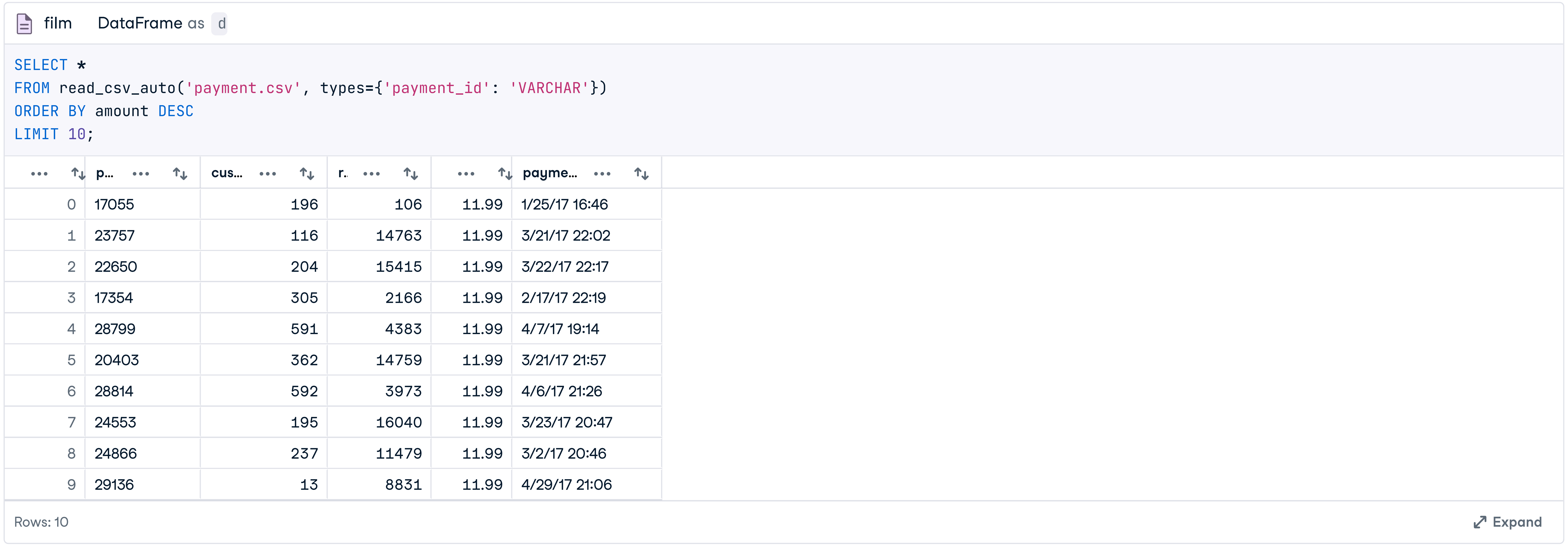
Rows: 2


film DataFrame as d

```sql
SELECT name,
    STRING_AGG(title, ',') AS film_titles
FROM 'film.csv' AS f
INNER JOIN 'language.csv' AS l
  ON f.language_id = l.language_id
WHERE release_year = 2010
  AND rating = 'G'
GROUP BY name;
```

		film_titles
0	Japanese	AMISTAD MIDSUMMER,BUGSY SONG,DOCT…
1	German	BEAUTY GREASE
2	Italian	DESPERATE TRAINSPOTTING,DWARFS ALTER…
3	Mandarin	ATLANTIS CAUSE,AUTUMN CROW,CASUALTI…
4	English	ACE GOLDFINGER,VALLEY PACKER
5	French	CAT CONEHEADS,DANCING FEVER,LUST LO…

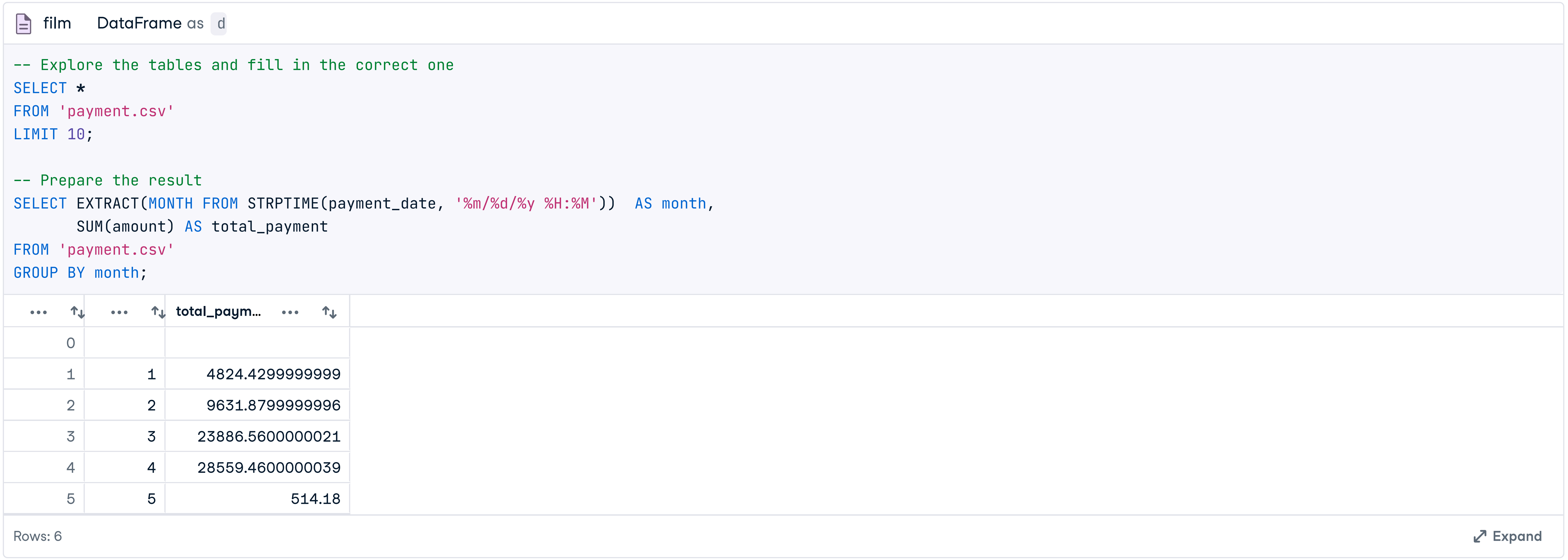
Rows: 6


film DataFrame as d

```
SELECT *
FROM read_csv_auto('payment.csv', types={'payment_id': 'VARCHAR'})
ORDER BY amount DESC
LIMIT 10;
```

	p...	cus...	r.		payme...
0	17055	196	106	11.99	1/25/17 16:46
1	23757	116	14763	11.99	3/21/17 22:02
2	22650	204	15415	11.99	3/22/17 22:17
3	17354	305	2166	11.99	2/17/17 22:19
4	28799	591	4383	11.99	4/7/17 19:14
5	20403	362	14759	11.99	3/21/17 21:57
6	28814	592	3973	11.99	4/6/17 21:26
7	24553	195	16040	11.99	3/23/17 20:47
8	24866	237	11479	11.99	3/2/17 20:46
9	29136	13	8831	11.99	4/29/17 21:06

Rows: 10

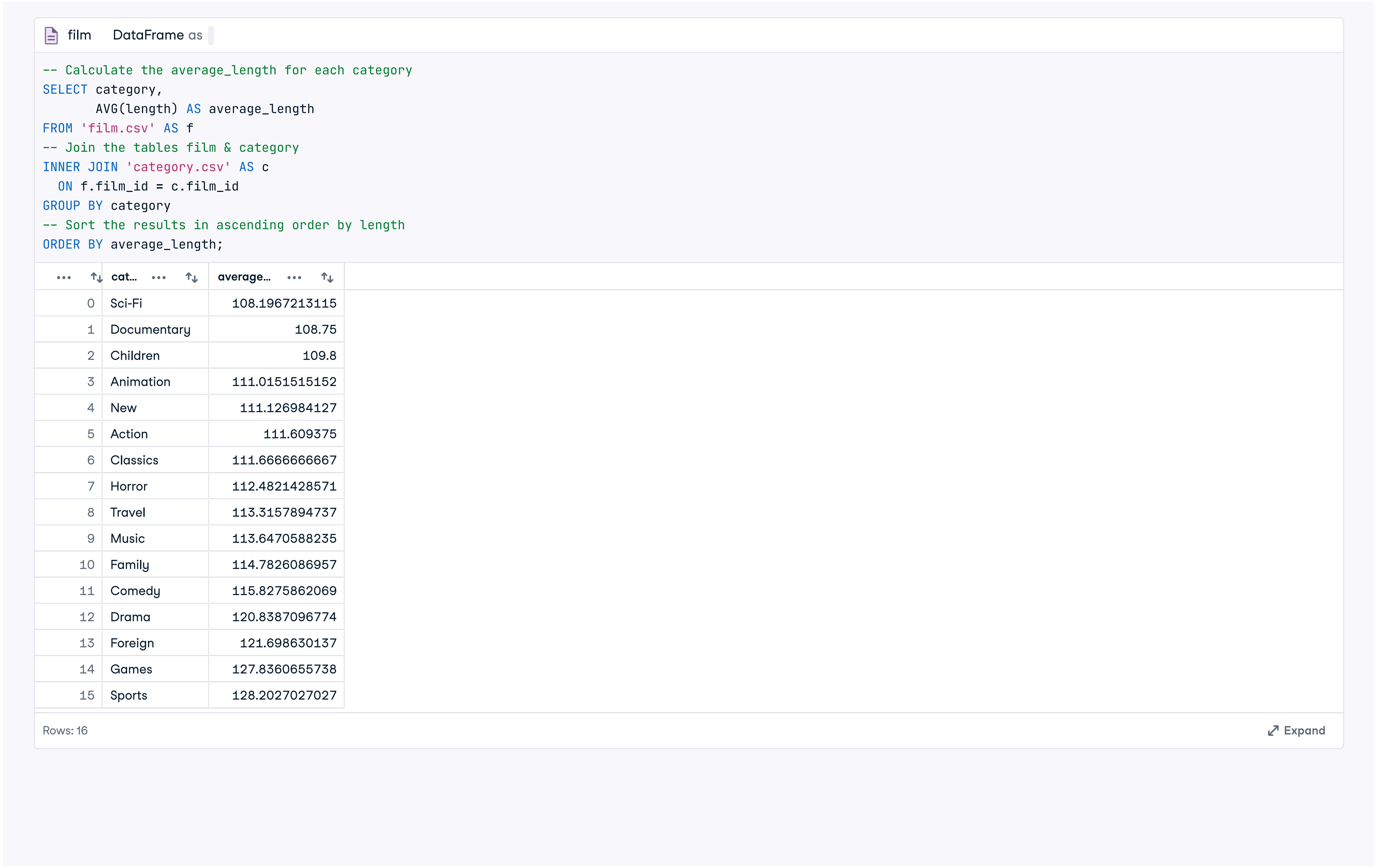

film DataFrame as d

```
-- Explore the tables and fill in the correct one
SELECT *
FROM 'payment.csv'
LIMIT 10;

-- Prepare the result
SELECT EXTRACT(MONTH FROM STRPTIME(payment_date, '%m/%d/%y %H:%M'))  AS month,
       SUM(amount) AS total_payment
FROM 'payment.csv'
GROUP BY month;
```

		total_paym...
0		
1	1	4824.4299999999
2	2	9631.8799999996
3	3	23886.5600000021
4	4	28559.4600000039
5	5	514.18

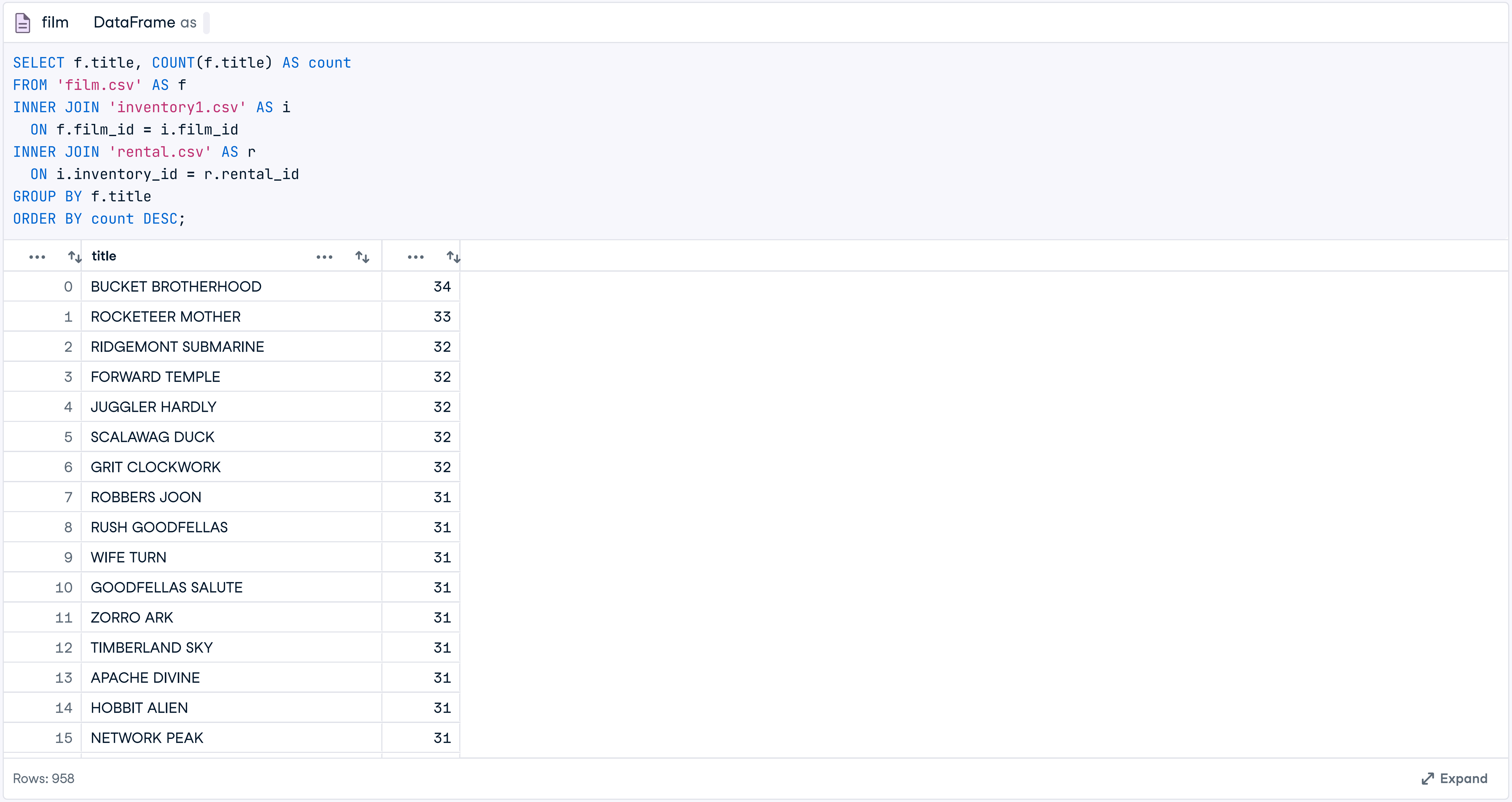
Rows: 6


film DataFrame as

```
-- Calculate the average_length for each category
SELECT category,
       AVG(length) AS average_length
FROM 'film.csv' AS f
-- Join the tables film & category
INNER JOIN 'category.csv' AS c
  ON f.film_id = c.film_id
GROUP BY category
-- Sort the results in ascending order by length
ORDER BY average_length;
```

	cat...	average...
0	Sci-Fi	108.1967213115
1	Documentary	108.75
2	Children	109.8
3	Animation	111.0151515152
4	New	111.126984127
5	Action	111.609375
6	Classics	111.6666666667
7	Horror	112.4821428571
8	Travel	113.3157894737
9	Music	113.6470588235
10	Family	114.7826086957
11	Comedy	115.8275862069
12	Drama	120.8387096774
13	Foreign	121.698630137
14	Games	127.8360655738
15	Sports	128.2027027027

Rows: 16


film DataFrame as

```
SELECT f.title, COUNT(f.title) AS count
FROM 'film.csv' AS f
INNER JOIN 'inventory1.csv' AS i
  ON f.film_id = i.film_id
INNER JOIN 'rental.csv' AS r
  ON i.inventory_id = r.rental_id
GROUP BY f.title
ORDER BY count DESC;
```

	title	
0	BUCKET BROTHERHOOD	34
1	ROCKETEER MOTHER	33
2	RIDGEMONT SUBMARINE	32
3	FORWARD TEMPLE	32
4	JUGGLER HARDLY	32
5	SCALAWAG DUCK	32
6	GRIT CLOCKWORK	32
7	ROBBERS JOON	31
8	RUSH GOODFELLAS	31
9	WIFE TURN	31
10	GOODFELLAS SALUTE	31
11	ZORRO ARK	31
12	TIMBERLAND SKY	31
13	APACHE DIVINE	31
14	HOBBIT ALIEN	31
15	NETWORK PEAK	31

Rows: 958

Expand