In [2]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import sqlite3 ## using MySQL database
```

In [3]:
```
conn = sqlite3.connect('sakila.db')

df = pd.read_sql('''
    SELECT
        rental.rental_id, rental.rental_date, rental.return_date,
        customer.last_name AS customer_lastname,
        store.store_id,
        city.city AS rental_store_city,
        film.title AS film_title, film.rental_duration AS film_rental_duration,
        film.rental_rate AS film_rental_rate, film.replacement_cost AS film_replacement
        film.rating AS film_rating
    FROM rental
    INNER JOIN customer ON rental.customer_id == customer.customer_id
    INNER JOIN inventory ON rental.inventory_id == inventory.inventory_id
    INNER JOIN store ON inventory.store_id == store.store_id
    INNER JOIN address ON store.address_id == address.address_id
    INNER JOIN city ON address.city_id == city.city_id
    INNER JOIN film ON inventory.film_id == film.film_id
    ;
''', conn, index_col='rental_id', parse_dates=['rental_date', 'return_date'])
```

In [4]:
```
df.head()
```

Out[4]:

rental_id	rental_date	return_date	customer_lastname	store_id	rental_store_city	film_title	film_renta
1	2005-05-24 22:53:30	2005-05-26 22:04:30	HUNTER	1	Lethbridge	BLANKET BEVERLY	
2	2005-05-24 22:54:33	2005-05-28 19:40:33	COLLAZO	2	Woodridge	FREAKY POCUS	
3	2005-05-24 23:03:39	2005-06-01 22:12:39	MURRELL	2	Woodridge	GRADUATE LORD	
4	2005-05-24 23:04:41	2005-06-03 01:43:41	PURDY	1	Lethbridge	LOVE SUICIDES	
5	2005-05-24 23:05:21	2005-06-02 04:33:21	HANSEN	2	Woodridge	IDOLS SNATCHERS	

In [7]:
```
df.shape ## HOW MANY ROWS AND COLUMNS
```

Out[7]:
```
(16044, 10)
```

In [8]:
```
df.describe() #Statistical value and understanding
```

Out[8]:

	store_id	film_rental_duration	film_rental_rate	film_replacement_cost
count	16044.000000	16044.00000	16044.000000	16044.000000
mean	1.506171	4.93549	2.942630	20.215443
std	0.499978	1.40169	1.649678	6.081771
min	1.000000	3.00000	0.990000	9.990000
25%	1.000000	4.00000	0.990000	14.990000
50%	2.000000	5.00000	2.990000	20.990000
75%	2.000000	6.00000	4.990000	25.990000
max	2.000000	7.00000	4.990000	29.990000

In [10]:
```
df['film_rental_rate'].mean() # SHOWING MEAN OF FILM RENTALS rates
```

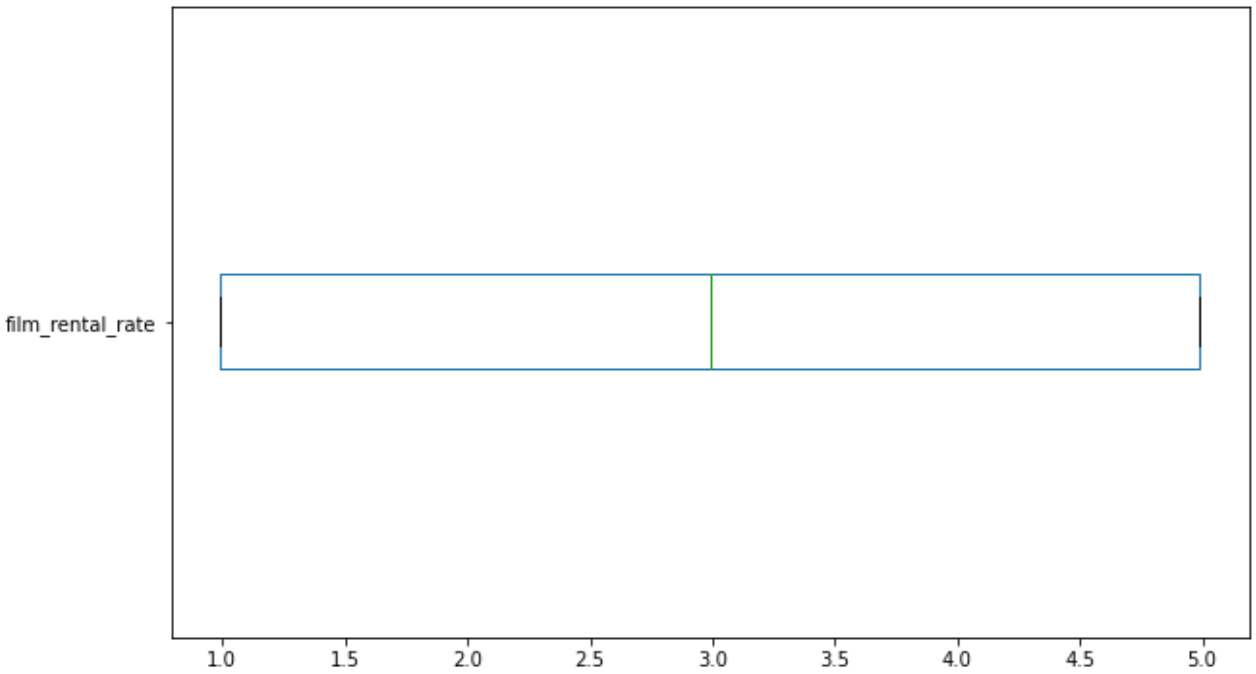
Out[10]: 2.9426302667662574

In [11]:
```
df['film_rental_rate'].median() #Showing  the median of rental rates
```

Out[11]: 2.99

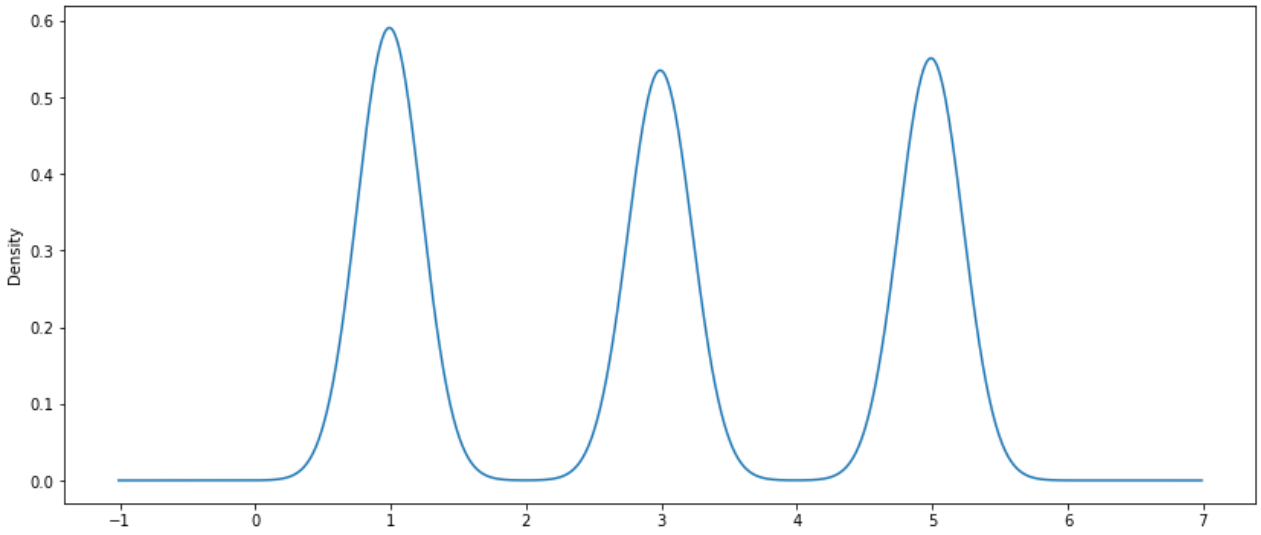
In [16]:
```
df['film_rental_rate'].plot(kind='box', vert=False, figsize=(10,6) )
```

Out[16]: <AxesSubplot:>



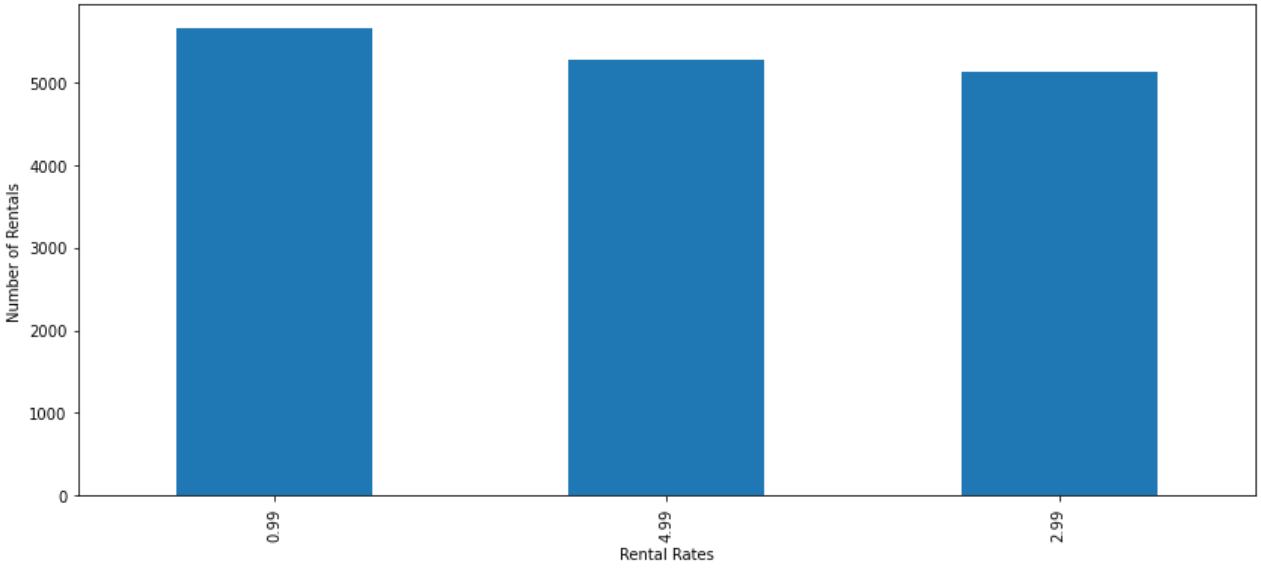
In [18]:
```
df['film_rental_rate'].plot(kind='density', figsize=(14,6))
```

Out[18]: <AxesSubplot:ylabel='Density'>



In [22]:
```
ax = df['film_rental_rate'].value_counts().plot(kind='bar', figsize=(14,6))
ax.set_ylabel('Number of Rentals')
ax.set_xlabel('Rental Rates')
```

Out[22]:
```
Text(0.5, 0, 'Rental Rates')
```



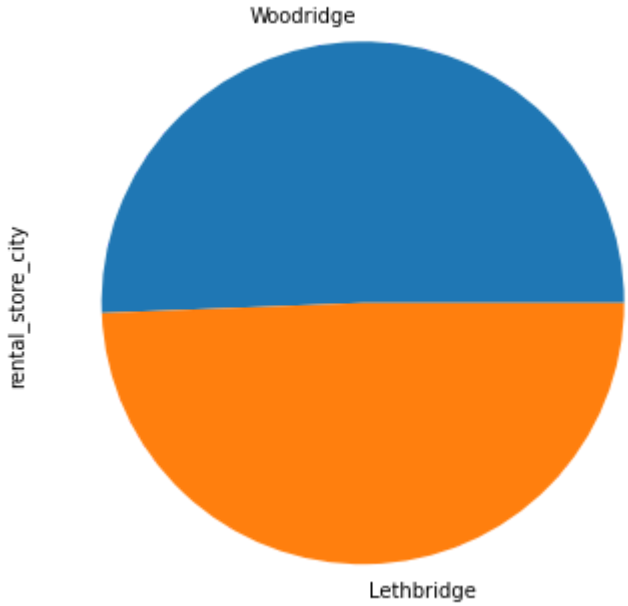
In [ ]:
```
#Categorical analysis and visualization
```

In [23]:
```
df['rental_store_city'].value_counts()
```

Out[23]:
```
Woodridge     8121
Lethbridge    7923
Name: rental_store_city, dtype: int64
```

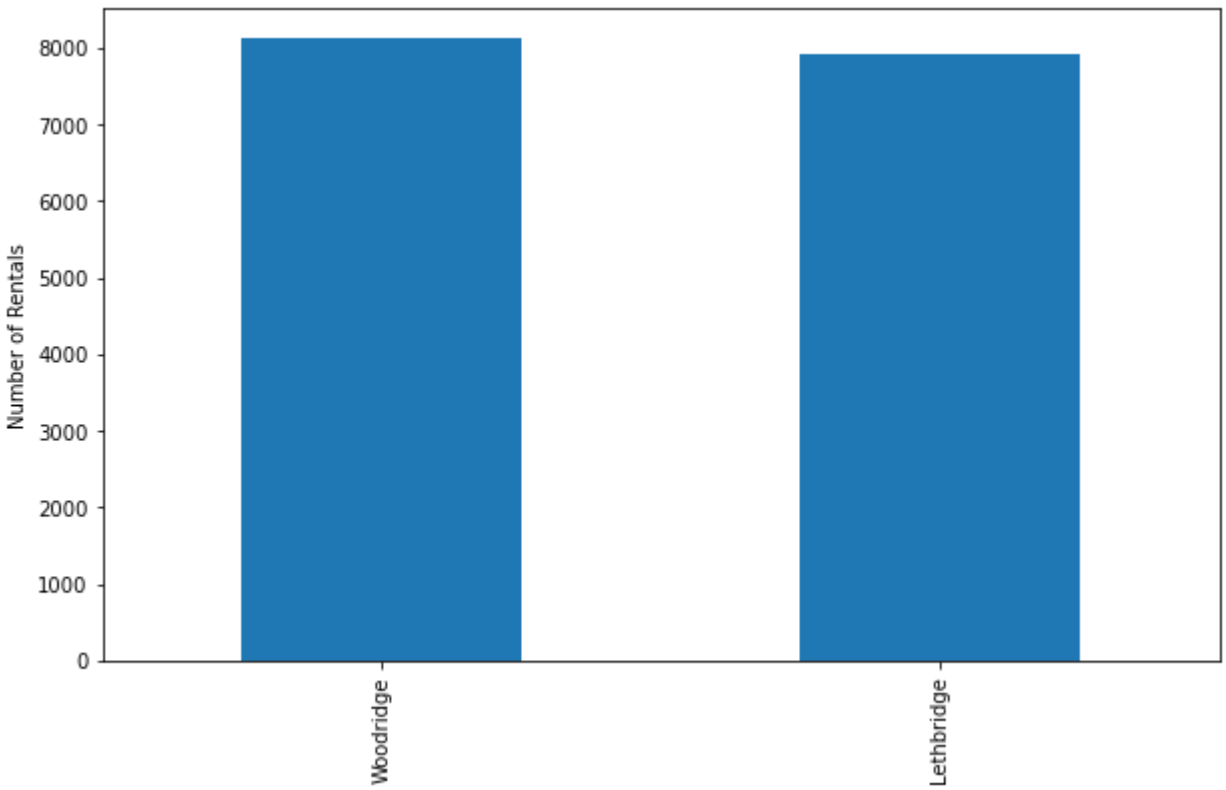
In [26]:
```
df['rental_store_city'].value_counts().plot(kind='pie', figsize=(10,6)) # plotting the
```

Out[26]:
```
<AxesSubplot:ylabel='rental_store_city'>
```



In [28]:
```
ax = df['rental_store_city'].value_counts().plot(kind='bar', figsize=(10,6))
ax.set_ylabel('Number of Rentals')
```

Out[28]:
```
Text(0, 0.5, 'Number of Rentals')
```



In [ ]:
```
# COLUMN WRANGLING
# We can also create new columns or modify existing ones.
```

In [29]:
```
df['rental_gain_return'] = df['film_rental_rate'] / df['film_replacement_cost'] * 100
df['rental_gain_return'].head()
```

Out[29]:
```
rental_id
1    13.597090
2    17.598587
3    19.946631
4     4.502046
5     9.969990
Name: rental_gain_return, dtype: float64
```

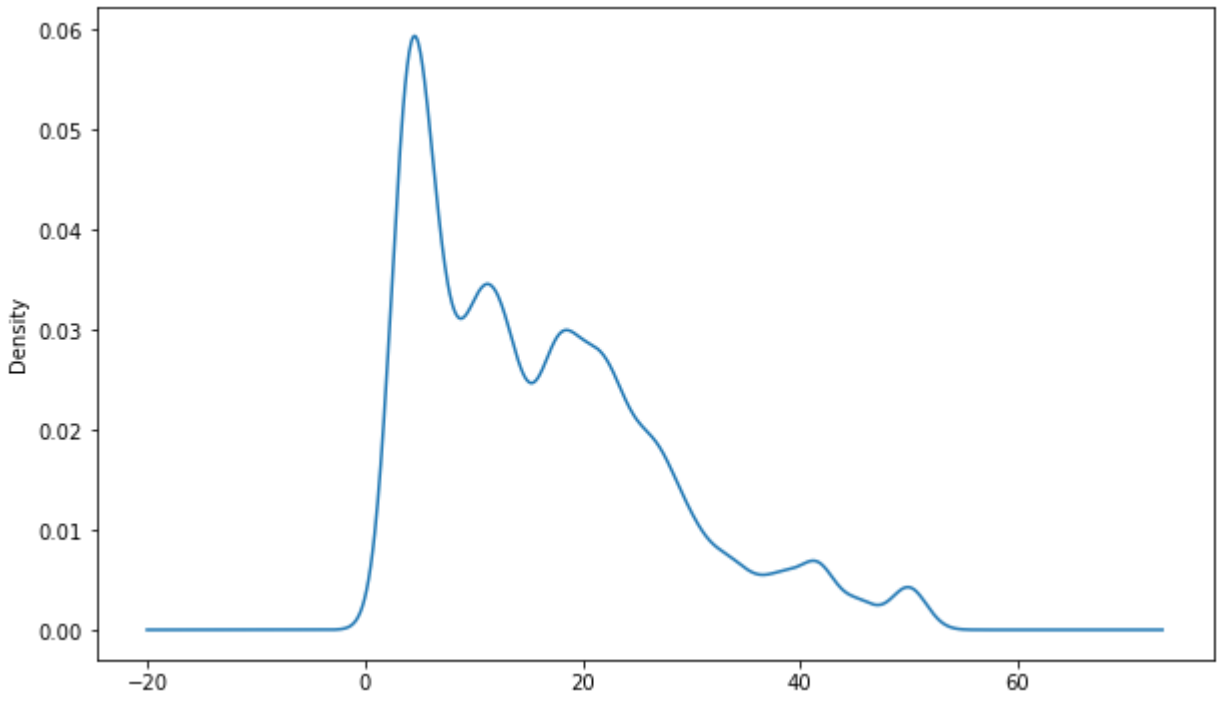
In [30]:
```
df.head()
```

Out[30]:

rental_id	rental_date	return_date	customer_lastname	store_id	rental_store_city	film_title	film_renta
1	2005-05-24 22:53:30	2005-05-26 22:04:30	HUNTER	1	Lethbridge	BLANKET BEVERLY	
2	2005-05-24 22:54:33	2005-05-28 19:40:33	COLLAZO	2	Woodridge	FREAKY POCUS	
3	2005-05-24 23:03:39	2005-06-01 22:12:39	MURRELL	2	Woodridge	GRADUATE LORD	
4	2005-05-24 23:04:41	2005-06-03 01:43:41	PURDY	1	Lethbridge	LOVE SUICIDES	
5	2005-05-24 23:05:21	2005-06-02 04:33:21	HANSEN	2	Woodridge	IDOLS SNATCHERS	

In [31]:
```
df['rental_gain_return'].plot(kind='density', figsize=(10,6))
```

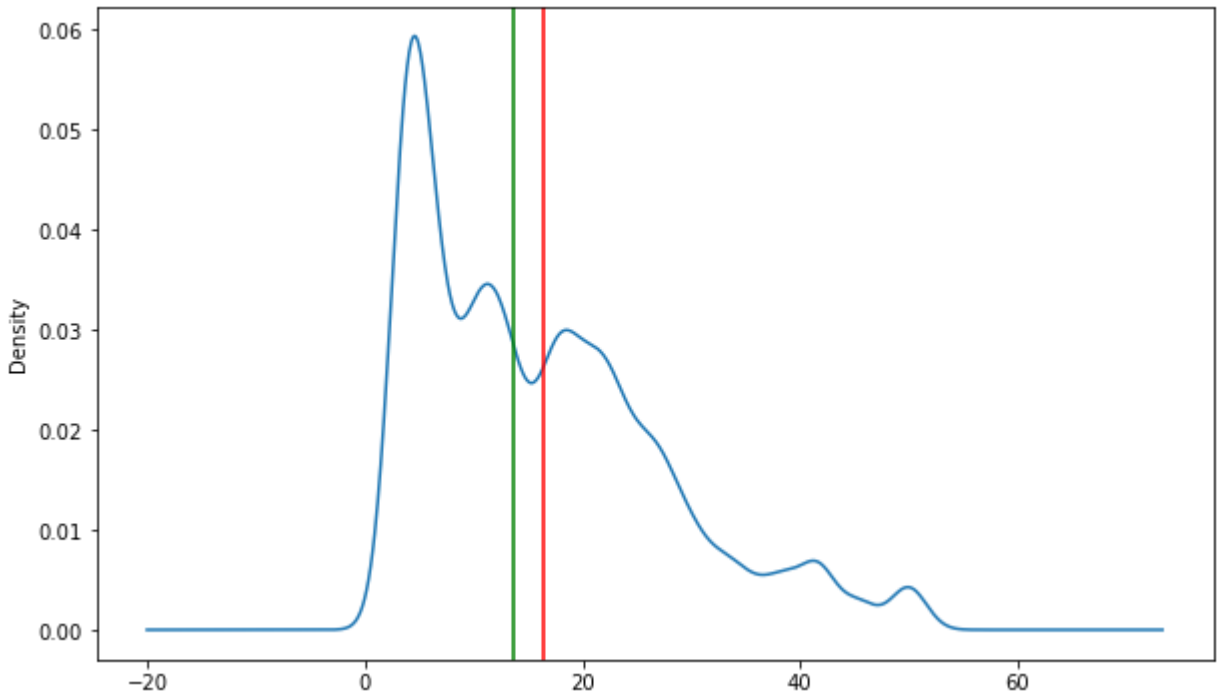
Out[31]:
```
<AxesSubplot:ylabel='Density'>
```



In [34]:

```
ax = df['rental_gain_return'].plot(kind='density', figsize=(10,6))
ax.axvline(df['rental_gain_return'].mean(), color='red') ## showing the mean of gain re
ax.axvline(df['rental_gain_return'].median(), color='green') ## showing the midean of g
```

Out[34]: <matplotlib.lines.Line2D at 0x207b804aeb0>



In [35]:
```
df['film_title'].value_counts().mean() ##While in average each film is rented 16.74 tim
```

Out[35]: 16.747390396659707

In [36]:
```
100 / 13.6 ##So 7.35 rentals are needed to recover film market price (film_replacement_
```

Out[36]: 7.352941176470589

In [ ]:
```
### SELECTION INDEXING
```

In [39]:
```
df.loc[df['customer_lastname'] == 'HANSEN'] ## GET THE RECORDS OF THE COSTUMER WITH THE
```

Out[39]:

rental_id	rental_date	return_date	customer_lastname	store_id	rental_store_city	film_title	film_rent
5	2005-05-24 23:05:21	2005-06-02 04:33:21	HANSEN	2	Woodridge	IDOLS SNATCHERS	
134	2005-05-25 21:48:41	2005-06-02 18:28:41	HANSEN	2	Woodridge	JUMPING WRATH	
416	2005-05-27 15:02:10	2005-05-29 10:34:10	HANSEN	2	Woodridge	LESSON CLEOPATRA	
809	2005-05-29 19:10:20	2005-06-05 19:05:20	HANSEN	2	Woodridge	INDIAN LOVE	

rental_id	rental_date	return_date	customer_lastname	store_id	rental_store_city	film_title	film_rent
1006	2005-05-31 00:57:08	2005-06-02 22:35:08	HANSEN	2	Woodridge	SALUTE APOLLO	
1368	2005-06-15 14:27:47	2005-06-23 18:07:47	HANSEN	1	Lethbridge	HUNCHBACK IMPOSSIBLE	
2603	2005-06-19 06:21:25	2005-06-26 03:19:25	HANSEN	2	Woodridge	CAT CONEHEADS	
5209	2005-07-09 11:22:39	2005-07-17 09:31:39	HANSEN	1	Lethbridge	WHALE BIKINI	
5266	2005-07-09 14:17:40	2005-07-16 10:42:40	HANSEN	2	Woodridge	LOATHING LEGALLY	
5592	2005-07-10 04:26:13	2005-07-19 02:32:13	HANSEN	2	Woodridge	LUKE MUMMY	
5635	2005-07-10 06:28:39	2005-07-17 08:35:39	HANSEN	2	Woodridge	FISH OPUS	
6129	2005-07-11 08:15:09	2005-07-18 13:00:09	HANSEN	2	Woodridge	STOCK GLASS	
6497	2005-07-12 03:04:29	2005-07-17 21:36:29	HANSEN	2	Woodridge	DANCING FEVER	
7786	2005-07-28 07:18:26	2005-07-29 03:00:26	HANSEN	2	Woodridge	KARATE MOON	
8300	2005-07-29 02:57:59	2005-08-05 01:12:59	HANSEN	2	Woodridge	VOYAGE LEGALLY	
8597	2005-07-29 12:55:55	2005-08-05 18:54:55	HANSEN	1	Lethbridge	TUXEDO MILE	
8787	2005-07-29 20:43:49	2005-07-31 15:15:49	HANSEN	2	Woodridge	LEGALLY SECRETARY	
10043	2005-07-31 19:02:07	2005-08-07 17:58:07	HANSEN	2	Woodridge	MARS ROMAN	
12179	2005-08-18 01:21:21	2005-08-19 00:59:21	HANSEN	2	Woodridge	FOREVER CANDIDATE	
13477	2005-08-20 01:07:00	2005-08-26 02:47:00	HANSEN	2	Woodridge	FINDING ANACONDA	
14350	2005-08-21 08:58:38	2005-08-30 03:29:38	HANSEN	1	Lethbridge	PRIMARY GLASS	

In [ ]:
```
## CREATE LIST OF THE FILM WITH HIGHEST REPLACEMENT COST
```

In [41]:
```
df['film_replacement_cost'].max()
```

Out[41]:
```
29.99
```

In [42]:
```
df.loc[df['film_replacement_cost']==df['film_replacement_cost'].max(), 'film_title']. u
```

Out[42]:
```
array(['IDOLS SNATCHERS', 'LAWLESS VISION', 'SONG HEDWIG',
       'LOATHING LEGALLY', 'PATIENT SISTER', 'RESERVOIR ADAPTATION',
       'JEEPERS WEDDING', 'GOLDFINGER SENSIBILITY', 'CHARIOTS CONSPIRACY',
       'HONEY TIES', 'GRAFFITI LOVE', 'SLACKER LIAISONS', 'DIRTY ACE',
       'BLINDNESS GUN', 'WYOMING STORM', 'FEUD FROGMEN', 'SALUTE APOLLO',
       'JINGLE SAGEBRUSH', 'HILLS NEIGHBORS', 'UNCUT SUICIDES',
       'EVERYONE CRAFT', 'FLATLINERS KILLER', 'BALLROOM MOCKINGBIRD',
       'RIVER OUTLAW', 'ARABIA DOGMA', 'VIRGIN DAISY', 'JERICHO MULAN',
       'SASSY PACKER', 'TRACY CIDER', 'LOVER TRUMAN', 'DOCTOR GRAIL',
       'GILMORE BOILED', 'PRINCESS GIANT', 'CRUELTY UNFORGIVEN',
       'REIGN GENTLEMEN', 'WEST LION', 'BONNIE HOLOCAUST', 'EARTH VISION',
       'RANDOM GO', 'CLOCKWORK PARADISE', 'FANTASIA PARK', 'RIGHT CRANES',
       'CUPBOARD SINNERS', 'OSCAR GOLD', 'SMILE EARRING',
       'HOLLYWOOD ANONYMOUS', 'POSEIDON FOREVER',
       'EXTRAORDINARY CONQUERER', 'QUEST MUSSOLINI', 'JAPANESE RUN',
       'CLYDE THEORY', 'DESPERATE TRAINSPOTTING'], dtype=object)
```

In [ ]:
```

```