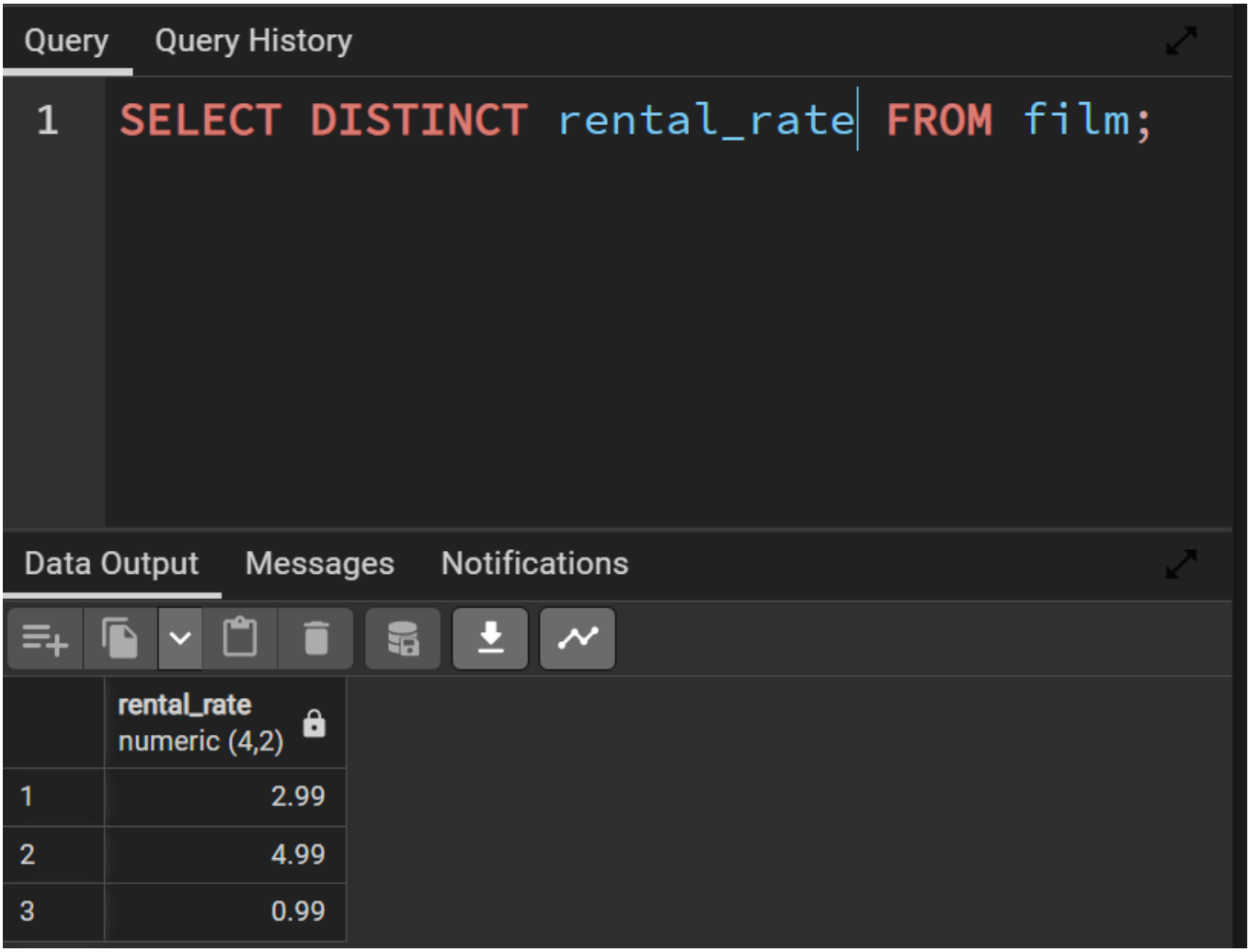
Notes 2 SELECT DISTINCT

> SELECT DISTINCT rental_rate FROM film

returns:



There's three tiers of rental rate. This is very similar to df.unique() in python since it gives all the possible options within the selected range. With df.unique you specify the range using:

> df['column name(s)'].unique()

```
In [10]: import numpy as np
         import pandas as pd
         df = pd.read_csv(r'C:\Users\User\Desktop\Udemy Data Science Course Resources\03-Pandas\Sales_Funnel_CRM.csv')

In [11]: df.head()
```

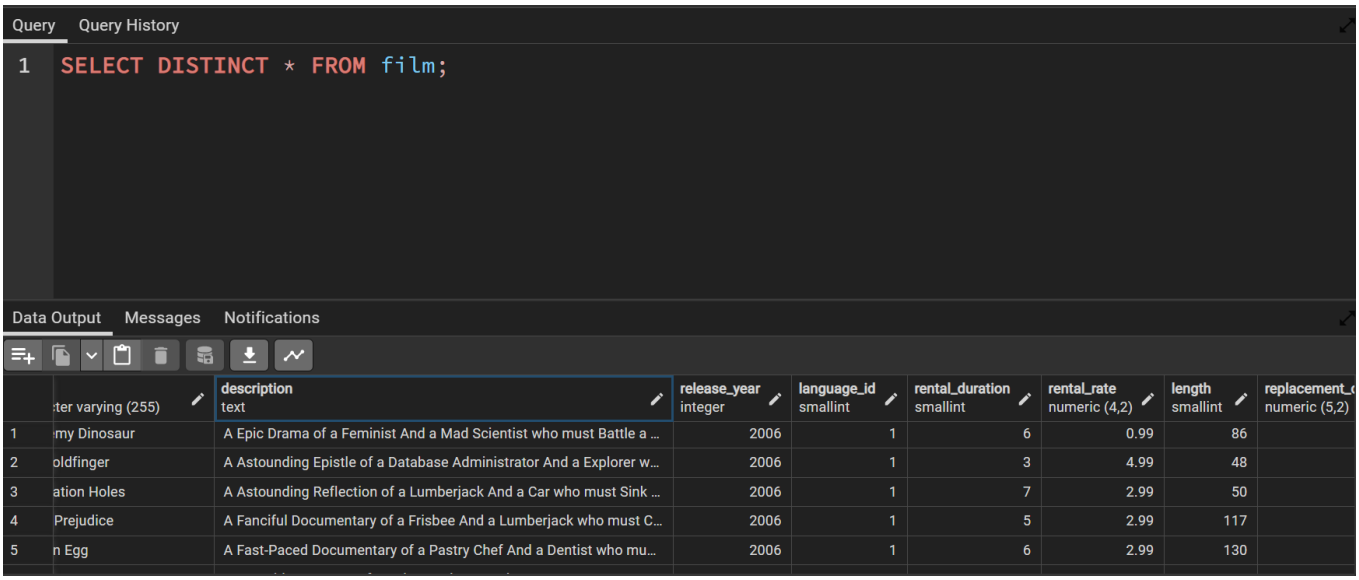
Out[11]:

	Account Number	Company	Contact	Account Manager	Product	Licenses	Sale Price	Status
0	2123398	Google	Larry Pager	Edward Thorp	Analytics	150	2100000	Presented
1	2123398	Google	Larry Pager	Edward Thorp	Prediction	150	700000	Presented
2	2123398	Google	Larry Pager	Edward Thorp	Tracking	300	350000	Under Review
3	2192650	BOBO	Larry Pager	Edward Thorp	Analytics	150	2450000	Lost
4	420496	IKEA	Elon Tusk	Edward Thorp	Analytics	300	4550000	Won

```
In [18]: df['Company'].unique()
```

Out[18]:
```
array([' Google', 'BOBO', 'IKEA', 'Tesla Inc.', 'Microsoft', 'Walmart',
       'Apple', 'Exxon Mobile', 'ATT', 'CVS Health', 'Salesforce',
       'Cisco'], dtype=object)
```

and with SQL, you just include the list of columns after SELECT DISTINCT



Selecting every distinct result from the entire table appears to basically accomplish the same thing as regular select, since the descriptions appear to be all unique to begin with. So, release year ends up repeating whereas if you select it as the only column:

Query | Query History

```
1  SELECT DISTINCT release_year FROM film;
```

Data Output | Messages | Notifications

	release_year integer
1	2006