Lec 21 - Demo

Setup

```python
import numpy as np
import pandas as pd
import seaborn as sns

import pyarrow as pa

import polars as pl
```

Data

```python
df_lazy  = pl.scan_parquet("~/Scratch/nyc_taxi/yellow_tripdata_2022-*.parquet")
df_eager = pl.read_parquet("~/Scratch/nyc_taxi/yellow_tripdata_2022-*.parquet")
```

```python
df_eager.schema
```

```
{'VendorID': Int64,
 'tpep_pickup_datetime': Datetime(time_unit='ns', time_zone=None),
 'tpep_dropoff_datetime': Datetime(time_unit='ns', time_zone=None),
 'passenger_count': Float64,
 'trip_distance': Float64,
 'RatecodeID': Float64,
 'store_and_fwd_flag': Utf8,
 'PULocationID': Int64,
 'DOLocationID': Int64,
 'payment_type': Int64,
 'fare_amount': Float64,
 'extra': Float64,
 'mta_tax': Float64,
 'tip_amount': Float64,
 'tolls_amount': Float64,
 'improvement_surcharge': Float64,
 'total_amount': Float64,
 'congestion_surcharge': Float64,
 'airport_fee': Float64}
```

```python
df_eager.columns
```

```
['VendorID',
 'tpep_pickup_datetime',
 'tpep_dropoff_datetime',
 'passenger_count',
 'trip_distance',
 'RatecodeID',
 'store_and_fwd_flag',
 'PULocationID',
 'DOLocationID',
 'payment_type',
 'fare_amount',
 'extra',
 'mta_tax',
 'tip_amount',
 'tolls_amount',
 'improvement_surcharge',
 'total_amount',
 'congestion_surcharge',
 'airport_fee']
```

Tipping rates

```
df_eager.select([
  "tpep_pickup_datetime", "tip_amount", "fare_amount", "total_amount",
  (pl.col("tip_amount") / pl.col("fare_amount")).alias("tip_perc")
])
```

shape: (36256549, 5)

tpep_pickup_datetime	tip_amount	fare_amount	total_amount	tip_perc
datetime[ns]	f64	f64	f64	f64
2022-01-01 00:35:40	3.65	14.5	21.95	0.251724
2022-01-01 00:33:43	4.0	8.0	13.3	0.5
2022-01-01 00:53:21	1.76	7.5	10.56	0.234667
2022-01-01 00:25:21	0.0	8.0	11.8	0.0
2022-01-01 00:36:48	3.0	23.5	30.3	0.12766
2022-01-01 00:40:15	13.0	33.0	56.35	0.393939
2022-01-01 00:20:50	5.2	17.0	26.0	0.305882
2022-01-01 00:13:04	0.0	9.0	12.8	0.0
2022-01-01 00:30:02	2.25	12.0	18.05	0.1875
2022-01-01 00:48:52	0.0	5.0	8.8	0.0
2022-01-01 00:55:03	0.0	8.5	12.3	0.0
2022-01-01 00:31:06	0.0	4.5	8.3	0.0

tpep_pickup_datetime	tip_amount	fare_amount	total_amount	tip_perc
datetime[ns]	f64	f64	f64	f64
…	…	…	…	…
2022-11-30 23:22:21	5.84	25.91	35.05	0.225396
2022-11-30 23:30:00	5.61	24.47	33.38	0.22926
2022-11-30 23:15:12	0.0	25.88	29.18	0.0
2022-11-30 23:40:17	2.63	12.33	15.76	0.213301
2022-11-30 23:21:00	4.93	23.87	29.6	0.206535
2022-11-30 23:22:40	0.0	20.22	23.52	0.0
2022-11-30 23:58:12	0.0	33.79	37.09	0.0
2022-11-30 23:17:09	0.0	13.46	16.76	0.0
2022-11-30 23:48:48	2.0	13.59	18.89	0.147167
2022-11-30 23:04:36	1.18	8.0	12.98	0.1475
2022-11-30 23:18:37	2.15	10.5	16.45	0.204762
2022-11-30 23:30:50	0.0	24.97	28.27	0.0

```
df_eager.select([
  "tpep_pickup_datetime", "tip_amount", "fare_amount", "total_amount",
  (pl.col("tip_amount") / pl.col("fare_amount")).alias("tip_perc")
]).select([
  pl.min("tip_perc").alias("min"),
  pl.mean("tip_perc").alias("mean"),
  pl.median("tip_perc").alias("median"),
  pl.max("tip_perc").alias("max")
])
```

shape: (1, 4)

min	mean	median	max
f64	f64	f64	f64
-40.0	NaN	0.242581	inf

```
df_eager.filter(
  (pl.col("fare_amount") > 0) &
  (pl.col("tip_amount") > 0)
).select([
  "tpep_pickup_datetime", "tip_amount", "fare_amount", "total_amount",
  (pl.col("tip_amount") / (pl.col("total_amount") - pl.col("tip_amount"))).alias("tip_per
]).with_columns([
  pl.all().sort_by("tip_perc")
])
```

shape: (27513713, 5)

tpep_pickup_datetime	tip_amount	fare_amount	total_amount	tip_perc
datetime[ns]	f64	f64	f64	f64
2022-02-05 07:49:26	0.01	500.0	500.31	0.00002
2022-11-09 21:51:13	0.01	496.0	496.31	0.00002
2022-11-06 22:13:30	0.01	495.0	495.31	0.00002
2022-11-02 22:09:46	0.01	490.0	490.31	0.00002
2022-11-11 23:32:51	0.01	490.0	490.31	0.00002
2022-11-04 22:08:54	0.01	487.0	489.81	0.00002
2022-08-19 14:56:34	0.01	450.0	466.56	0.000021
2022-07-05 14:53:07	0.01	400.0	400.31	0.000025
2022-07-08 15:57:56	0.01	350.0	352.06	0.000028
2022-09-28 17:46:30	0.01	350.0	350.31	0.000029
2022-06-07 21:39:50	0.01	310.0	330.36	0.00003
2022-08-23 14:52:37	0.01	325.0	325.31	0.000031
…	…	…	…	…
2022-04-20 22:45:06	120.0	0.01	120.31	387.096774
2022-08-15 10:57:57	120.0	0.01	120.31	387.096774
2022-04-02 17:30:53	125.0	0.01	125.31	403.225806
2022-06-13 19:18:58	150.0	0.01	150.31	483.870968
2022-08-05 21:03:47	150.0	0.01	150.31	483.870968
2022-06-25 10:01:47	155.0	0.01	155.31	500.0
2022-10-29 19:28:41	155.0	0.01	155.31	500.0
2022-01-09 00:56:26	168.88	0.01	169.19	544.774194
2022-08-19 11:32:01	217.0	0.01	217.31	700.0
2022-04-12 11:44:10	225.0	0.01	225.31	725.806452
2022-07-25 13:32:00	250.0	0.01	250.31	806.451613
2022-10-15 05:23:22	500.0	0.01	500.31	1612.903226

```
df_eager.filter(
  (pl.col("fare_amount") > 0) &
  (pl.col("tip_amount") > 0)
).select([
  "tpep_pickup_datetime", "tip_amount", "fare_amount", "total_amount",
  (pl.col("tip_amount") / (pl.col("total_amount") - pl.col("tip_amount"))).alias("tip_per
]).select([
  pl.min("tip_perc").alias("min"),
  pl.mean("tip_perc").alias("mean"),
  pl.median("tip_perc").alias("median"),
  pl.max("tip_perc").alias("max")
])
```

shape: (1, 4)

min	mean	median	max
f64	f64	f64	f64
0.00002	0.195446	0.2	1612.903226

```
df_eager.filter(
  (pl.col("fare_amount") > 0) &
  (pl.col("tip_amount") > 0)
).select([
  "tpep_pickup_datetime", "tip_amount", "fare_amount", "total_amount",
  (pl.col("tip_amount") / (pl.col("total_amount") - pl.col("tip_amount"))).alias("tip_per
  pl.col("tpep_pickup_datetime").dt.hour().alias("hour"),
  pl.col("tpep_pickup_datetime").dt.weekday().alias("wday")
]).groupby(
  ["hour","wday"]
).agg([
  pl.mean("tip_perc").alias("mean_tip_perc")
]).with_columns([
  pl.all().sort_by("mean_tip_perc")
])
```

shape: (168, 3)

hour	wday	mean_tip_perc
u32	u32	f64
7	1	0.189804
6	1	0.190432
8	2	0.19054
20	6	0.190774
19	5	0.19091

hour	wday	mean_tip_perc
u32	u32	f64
8	1	0.191063
7	5	0.191065
8	3	0.191141
18	3	0.191353
7	3	0.191355
19	3	0.191364
7	4	0.191526
…	…	…
2	3	0.210654
2	4	0.210673
4	2	0.21145
5	7	0.211656
3	2	0.212481
4	5	0.213729
3	4	0.213903
4	7	0.214736
3	3	0.214736
3	1	0.215657
2	1	0.217996
5	6	0.297631

```
df_eager.filter(
  (pl.col("fare_amount") > 0) &
  (pl.col("tip_amount") > 0)
).select([
  "tpep_pickup_datetime", "tip_amount", "fare_amount", "total_amount",
  (pl.col("tip_amount") / (pl.col("total_amount") - pl.col("tip_amount"))).alias("tip_per
  pl.col("tpep_pickup_datetime").dt.hour().alias("hour"),
  pl.col("tpep_pickup_datetime").dt.weekday().alias("wday")
]).groupby(
  ["hour","wday"]
).agg([
  pl.mean("tip_perc").alias("mean_tip_perc")
]).with_columns([
  pl.col("mean_tip_perc").round(3)
]).with_columns([
  pl.all().sort_by(["wday", "hour"])
]).pivot(
  values="mean_tip_perc", index="wday", columns="hour"
)
```

shape: (7, 25)

wday	0	1	2	3	4	5	6	7	8	9	10	11	12	13	
u32	f64	f64	f64	f64	f64	f64	f64	f64	f64	f64	f64	f64	f64	f64	f6
1	0.199	0.202	0.218	0.216	0.206	0.196	0.19	0.19	0.191	0.195	0.202	0.198	0.199	0.204	0.
2	0.199	0.203	0.208	0.212	0.211	0.196	0.193	0.193	0.191	0.195	0.199	0.201	0.198	0.199	0.
3	0.2	0.203	0.211	0.215	0.208	0.2	0.194	0.191	0.191	0.195	0.197	0.198	0.197	0.199	0.
4	0.203	0.205	0.211	0.214	0.209	0.199	0.192	0.192	0.192	0.195	0.197	0.197	0.198	0.199	0.
5	0.195	0.201	0.204	0.208	0.214	0.201	0.194	0.191	0.193	0.196	0.197	0.201	0.198	0.198	0.
6	0.192	0.194	0.197	0.201	0.207	0.298	0.201	0.201	0.2	0.199	0.202	0.197	0.196	0.196	0.
7	0.195	0.195	0.195	0.198	0.215	0.212	0.199	0.2	0.2	0.198	0.198	0.197	0.197	0.196	0.

```
df_lazy.filter(
  (pl.col("fare_amount") > 0) &
  (pl.col("tip_amount") > 0)
).select([
  "tpep_pickup_datetime", "tip_amount", "fare_amount", "total_amount",
  (pl.col("tip_amount") / (pl.col("total_amount") - pl.col("tip_amount"))).alias("tip_per
  pl.col("tpep_pickup_datetime").dt.hour().alias("hour"),
  pl.col("tpep_pickup_datetime").dt.weekday().alias("wday")
]).groupby(
  ["hour","wday"]
).agg([
  pl.mean("tip_perc").alias("mean_tip_perc")
]).with_columns([
  pl.col("mean_tip_perc").round(3)
]).with_columns([
  pl.all().sort_by(["wday", "hour"])
]).collect(
).pivot(
  values="mean_tip_perc", index="wday", columns="hour"
)
```

```
PARTITIONED DS
```

shape: (7, 25)

wday	0	1	2	3	4	5	6	7	8	9	10	11	12	13	
u32	f64	f64	f64	f64	f64	f64	f64	f64	f64	f64	f64	f64	f64	f64	f6
1	0.199	0.202	0.218	0.216	0.206	0.196	0.19	0.19	0.191	0.195	0.202	0.198	0.199	0.204	0.
2	0.199	0.203	0.208	0.212	0.211	0.196	0.193	0.193	0.191	0.195	0.199	0.201	0.198	0.199	0.
3	0.2	0.203	0.211	0.215	0.208	0.2	0.194	0.191	0.191	0.195	0.197	0.198	0.197	0.199	0.
4	0.203	0.205	0.211	0.214	0.209	0.199	0.192	0.192	0.192	0.195	0.197	0.197	0.198	0.199	0.
5	0.195	0.201	0.204	0.208	0.214	0.201	0.194	0.191	0.193	0.196	0.197	0.201	0.198	0.198	0.
6	0.192	0.194	0.197	0.201	0.207	0.298	0.201	0.201	0.2	0.199	0.202	0.197	0.196	0.196	0.
7	0.195	0.195	0.195	0.198	0.215	0.212	0.199	0.2	0.2	0.198	0.198	0.197	0.197	0.196	0.