```
!dlt --version
```

```
dlt 1.6.1
```

```
import dlt
from dlt.sources.helpers.rest_client import RESTClient
from dlt.sources.helpers.rest_client.paginators import PageNumberPaginator


# your code is here
@dlt.resource(name="rides")   # <--- The name of the resource (will be used as the table name)
def ny_taxi():
    client = RESTClient(
        base_url="https://us-central1-dlthub-analytics.cloudfunctions.net",
        paginator=PageNumberPaginator(
            base_page=1,
            total_path=None
        )
    )

    for page in client.paginate("data_engineering_zoomcamp_api"):    # <--- API endpoint for retrieving taxi ride data
        yield page   #

pipeline = dlt.pipeline(
    pipeline_name="ny_taxi_pipeline",
    destination="duckdb",
    dataset_name="ny_taxi_data"
)


load_info = pipeline.run(ny_taxi)
print(load_info)
```

```
Pipeline ny_taxi_pipeline load step completed in 2.51 seconds
1 load package(s) were loaded to destination duckdb and into dataset ny_taxi_data
The duckdb destination used duckdb:////content/ny_taxi_pipeline.duckdb location to store data
Load package 1739611613.0544734 is LOADED and contains no failed jobs
```

```
import duckdb
from google.colab import data_table
data_table.enable_dataframe_formatter()

# A database '<pipeline_name>.duckdb' was created in working directory so just connect to it

# Connect to the DuckDB database
conn = duckdb.connect(f"{pipeline.pipeline_name}.duckdb")

# Set search path to the dataset
conn.sql(f"SET search_path = '{pipeline.dataset_name}'")

# Describe the dataset
conn.sql("DESCRIBE").df()
```

1 to 4 of 4 entries Filter

index	database	schema	name	column_names	column_types	temporary
0	ny_taxi_pipeline	ny_taxi_data	_dlt_loads	['load_id' 'schema_name' 'status' 'inserted_at' 'schema_version_hash']	['VARCHAR' 'VARCHAR' 'BIGINT' 'TIMESTAMP WITH TIME ZONE' 'VARCHAR']	false
1	ny_taxi_pipeline	ny_taxi_data	_dlt_pipeline_state	['version' 'engine_version' 'pipeline_name' 'state' 'created_at' 'version_hash' '_dlt_load_id' '_dlt_id']	['BIGINT' 'BIGINT' 'VARCHAR' 'VARCHAR' 'TIMESTAMP WITH TIME ZONE' 'VARCHAR' 'VARCHAR' 'VARCHAR']	false
2	ny_taxi_pipeline	ny_taxi_data	_dlt_version	['version' 'engine_version' 'inserted_at' 'schema_name' 'version_hash' 'schema']	['BIGINT' 'BIGINT' 'TIMESTAMP WITH TIME ZONE' 'VARCHAR' 'VARCHAR' 'VARCHAR']	false
3	ny_taxi_pipeline	ny_taxi_data	rides	['end_lat' 'end_lon' 'fare_amt' 'passenger_count' 'payment_type' 'start_lat' 'start_lon' 'tip_amt' 'tolls_amt' 'total_amt' 'trip_distance'	['DOUBLE' 'DOUBLE' 'DOUBLE' 'BIGINT' 'VARCHAR' 'DOUBLE' 'DOUBLE' 'DOUBLE' 'DOUBLE' 'DOUBLE' 'DOUBLE' 'TIMESTAMP WITH TIME ZONE'	false

```
df = pipeline.dataset(dataset_type="default").rides.df()
df.shape
```

```
(10000, 18)
```

```
with pipeline.sql_client() as client:
    res = client.execute_sql(
            """
            SELECT
```

```
        AVG(date_diff('minute', trip_pickup_date_time, trip_dropoff_date_time))
        FROM rides;
        """
    )
# Prints column values of the first row
print(res)
```

```
[(12.3049,)]
```

Start coding or generate with AI.