Databricks Autoloader - Session 1
====================================

Streaming Data

spark structured streaming is also there to handle streaming data.

Autoloader provides or additional capabilities when compared to structured streaming in spark..

Autoloader is a wrapper on top of spark structured streaming.

Batch data...

Copy into command plays a vital role

```
%sql
create table orders(
order_id int,
order_date string,
customer_id int,
order_status string
) using delta
```

```
%sql
describe detail orders
```

```
%fs
ls /FileStore/data
```

```
%sql
copy into orders
from (select order_id::int, order_date, customer_id::int, order_status from
'dbfs:/FileStore/data/*')
fileformat = CSV
format_options('header' = 'true')
```

```
%sql
select * from orders
```

copy into is a retriable and idempotent operation

copy into keeps a track of what files are already loaded

it works well when we have a static schema

copy into is best suited when you are dealing with upto thousand of files..

can be best suited for scheduled job that runs at periodic interval

Databricks Autoloader - Session 2
====================================

streaming data

spark structured streaming...

Autoloader brings few more capabilities

data will be kept in
dbfs:/FileStore/retail_data/orders_data

checkpoint info will be kept in
dbfs:/FileStore/retail_data/orders_checkpoint

1. either you manually define the schema

2. otherwise make sure that atlease one file is present so that schema can be inferred

Autoloader

is specially designed to handle cloud storages

Autoloader incrementally and efficiently processes new data files as they arrive in cloud storage..

=> No file state management

=> scale to millions and billions of file

=> autoloader is quite easy to use

=> it supports schema evolution and inference

=> checkpoint optimization (key value pairs in rocks db...)

=> autoloader optimized file listing

  => incremental file listing
  => uses cloud native API's to get lists of files
  => it uses file notification service to get to know the files which are coming new..
  => fewer api's

AWS S3
azure data lake storage
blob storage
Google cloud storage
databricks file system

Copy into vs Autoloader

```
landing_zone = "dbfs:/FileStore/retail_data"
orders_data = landing_zone + "/orders_data"
checkpoint_path = landing_zone + "/orders_checkpoint"

orders_df = spark.readStream \
.format("cloudFiles") \
.option("cloudFiles.format","csv") \
.option("cloudFiles.inferSchema","true") \
.option("cloudFiles.inferColumnTypes","true") \
.option("cloudFiles.schemaLocation",checkpoint_path) \
.load(orders_data)

orders_df.display()

orders_df.writeStream \
.format("delta") \
.option("checkpointLocation",checkpoint_path) \
.outputMode("append") \
.toTable("orderdelta")

%sql
show tables

%sql
select count(*) from orderdelta

%sql
describe orderdelta
```

Databricks Autoloader - Session 3
===================================

copy into - batch

autoloader - streaming workloads

csv, json

parquet, avro

schema inference works -

1000 files or 50 GB of data whatever happens first

1000 files each of 1 mb...

1 GB

files of 5 gb each

10 files..

cloudFiles.schemaInference.sampleSize.numBytes
cloudFiles.schemaInference.sampleSize.numFiles

Schema Inference
how to manually define the schema
Schema Hints
More information - Add new columns (filename,timestamp)
query name while writing the data

Databricks Autoloader - Session 4
====================================

Schema evolution modes

4 modes

1. none - disabling the schema evolution

2. fail on new columns - the job will fail on encountering new columns but if the datatype changes for existing column it will put in rescued data.

3. rescue - a new column or a change in datatype all of this goes in rescued data column.

4. add new columns (default) -

Databricks Autoloader - Session 5
====================================

DBFS
Azure Datalake Gen2
Amazon S3

Azure Datalake Gen2
======================

storage account name - autoloaderdemosa
container name - retail-data
orders-data
orders1.csv

Amazon S3
============

login to your aws console

create a new user and give the user access to s3 buckets

databricks
s3 full access
programmatic access
Access key ID - AKIA3IB4DE2JIZSZRRX7
Secret access key - IhWKg4FYgd82nAGyBGi2qUByuhBtaOzgLlaMpMwr

s3://trendytech-databricks-bucket/retail_data/orders_data/orders1.csv