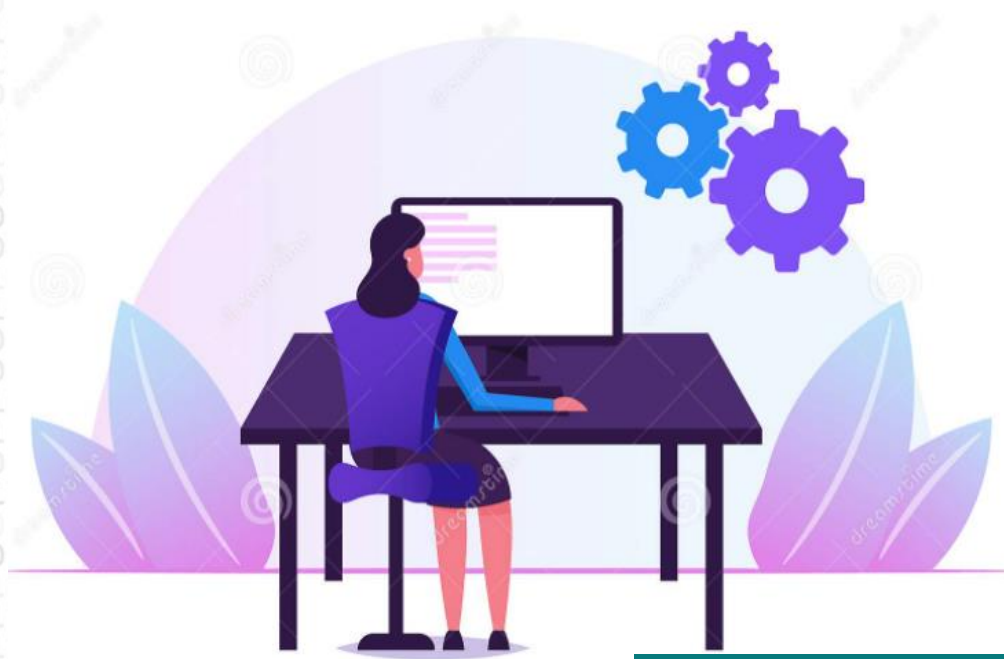


# Welcome!

- We'll start in a moment :)
- We are recording tonight's event. We may plan to take screenshots for social media.
  - ***If you want to remain anonymous***, change your name & keep video off.
- We'll introduce the hosts and break in-between for Q/A.
- We will make some time for Q&A at the end of the presentation as well.
- You can come prepared with questions. And, feel free to take notes.
- Online event best practices:
  - Don't multitask. Distractions reduce your ability to remember concepts.
  - Mute yourself when you aren't talking.
  - We want the session to be interactive.
  - Feel free to unmute and ask questions in the middle of the presentation.
  - Turn on your video if you feel comfortable.
  - Disclaimer: Speaker doesn't know everything!

## Check out:

- [Technical Tracks](#) and [Digital Events](#)
- Get updates – join the [Digital mailing list](#)
- Give us your feedback – take the [Survey](#)



# WWCode Digital + **Backend** **Backend Study Group**

July 15, 2021

Copyright © 2021 by Madhurima Nath

WOMEN WHO  
**CODE**

# Introduction & Agenda

- Welcome from WWCode!
- Our mission: Inspiring women to excel in technology careers.
- Our vision: A world where women are representative as technical executives, founders, VCs, board members and software engineers.
- What is Backend Engineering?
- **Insights into data engineering, data science and machine learning engineering**
  - **Data engineering [Part 1 of 2]**
    - + Introduction
    - + Similarities/Differences
    - + Day in a life of data engineer
    - + Tech stack
  - Data science and machine learning engineering [Part 2 of 2]



Prachi Shah  
**Senior Software  
Engineer @ Metromile**



Madhurima Nath  
**Data Scientist @  
Slalom**



# Backend Engineering

- What is Backend Engineering?
- Design, build and maintain server-side web applications.
- Concepts: Client-server architecture, API, micro-service, database engineering, distributed systems, storage, performance, deployment, availability, monitoring, etc.

## Software Design

- Defining the architecture, modules, interfaces and data.
- Solve a problem or build a product.
- Define the input, output, business rules, data schema.
- Design patterns solve common problems.
- 3 Types:
  - UI design: Data visualization and presentation.
  - Data design: Data representation and storage.
  - Process design: Validation, manipulation and storage of data.

# What is data engineering?

## Data engineers

- **design and build pipelines** to transform and transport data into a format readily useable by the data scientists or other end users.
- pipelines take data from many disparate sources and collect them into a single warehouse that is the unified data source for others.
- work closely with **DevOps and data science/machine learning** teams

# Data Engineer (DE) vs Data Scientist (DS) vs Machine Learning Engineer (MLE)

## **Data engineer:**

builds and develops pipelines, and maintains of data infrastructure, either on-premises or in the cloud (or hybrid or multi-cloud), comprising of databases or data warehouses

## **Data scientist:**

builds and develops mathematical and statistical models -- called machine learning models, to find patterns and gain more insights from the data

## **Machine learning engineer:**

design architecture and pipelines (or software) to integrate and automate the process of running the models developed by data scientists with the entire infrastructure

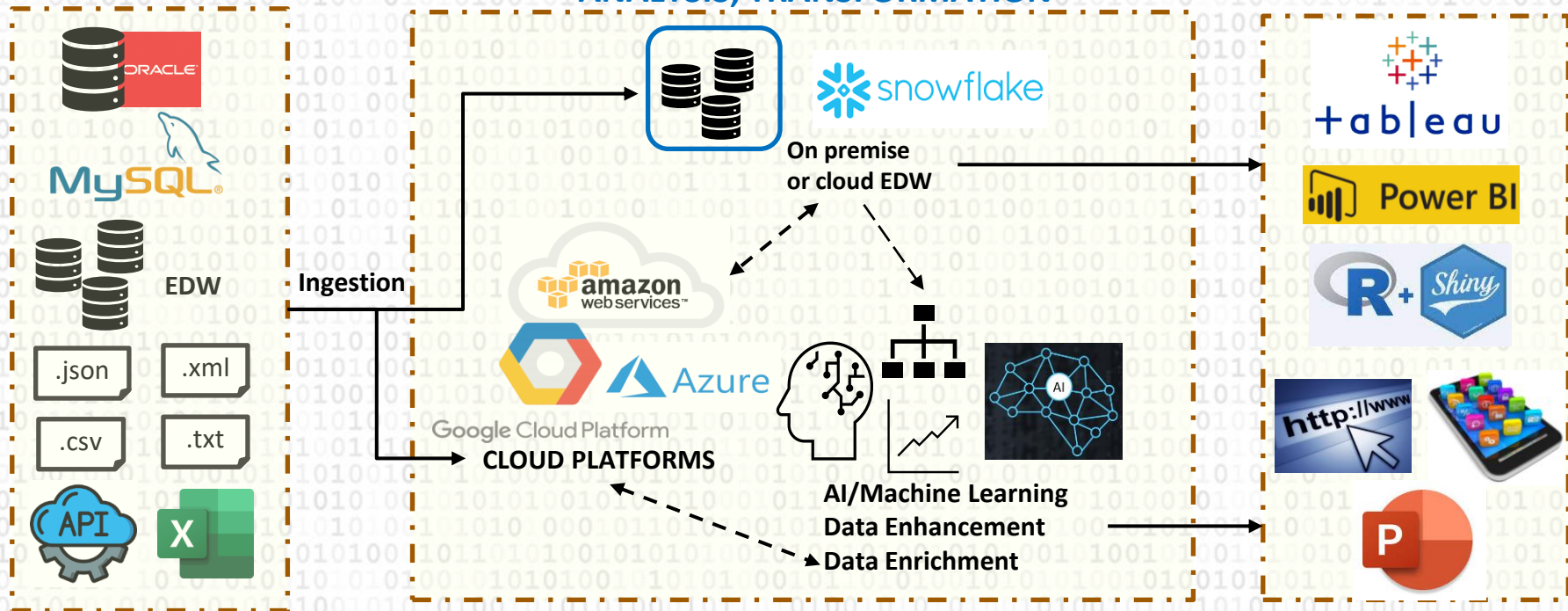


# Data Architecture Diagram

## DATA SOURCES

## DATA CLEANING, PROCESSING, ANALYSIS, TRANSFORMATION

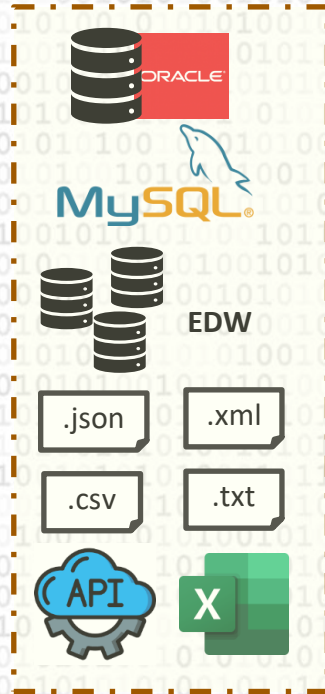
## USER INTERFACE



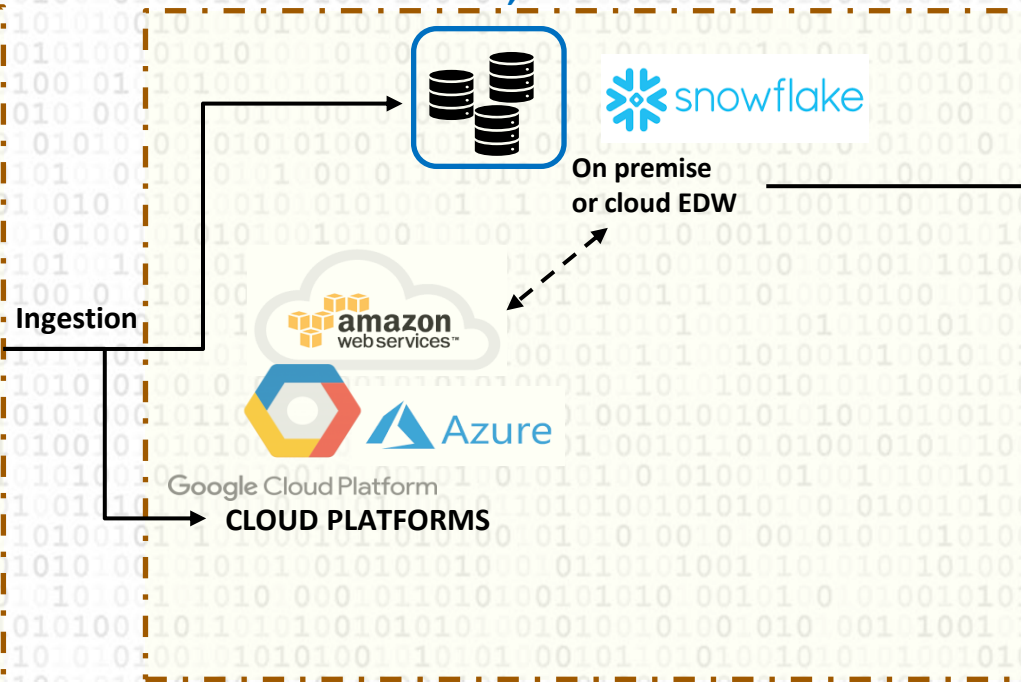
Disclaimer: These can change based on companies/industry  
Copyright © 2021 by Madhurima Nath

# Data Architecture Diagram – Data Engineer

## DATA SOURCES



## DATA CLEANING, PROCESSING, ANALYSIS, TRANSFORMATION



Data available for  
both analytics and  
visualization

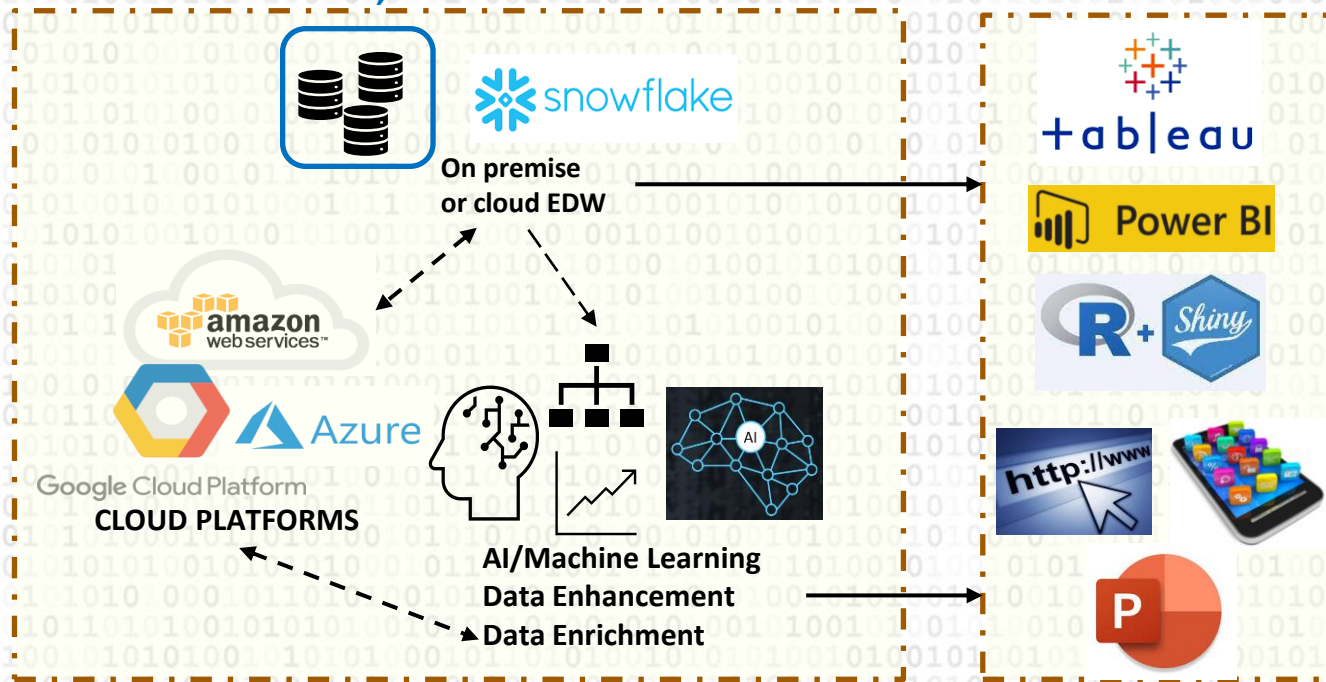
Disclaimer: These can change based on companies/industry  
Copyright © 2021 by Madhurima Nath



# Data Architecture Diagram – Data Scientist

**DATA CLEANING, PROCESSING,  
ANALYSIS, TRANSFORMATION**

**USER INTERFACE**



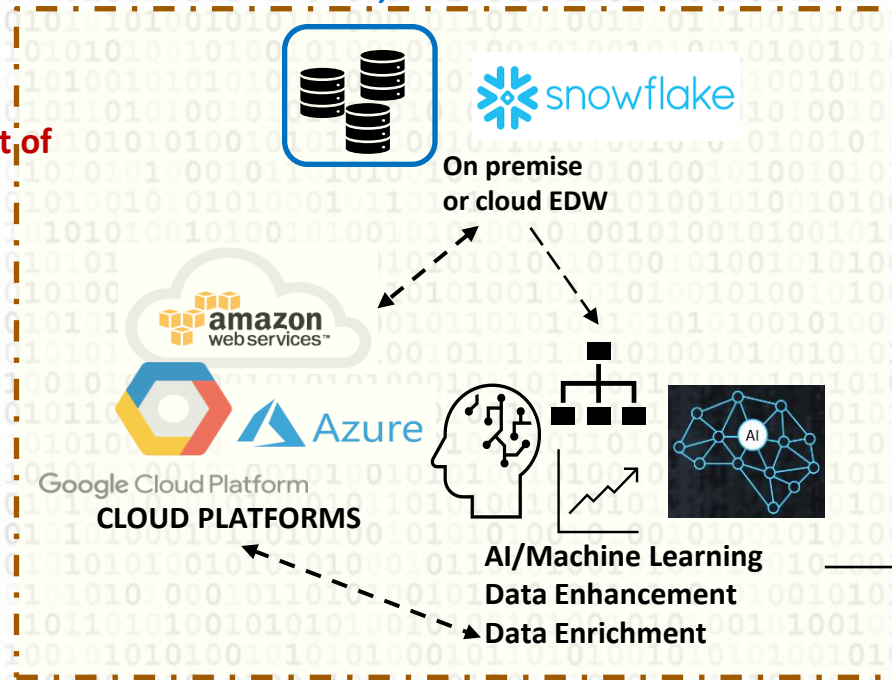
Disclaimer: These can change based on companies/industry  
Copyright © 2021 by Madhurima Nath

WOMEN WHO  
**CODE**

# Data Architecture Diagram – ML Engineer

**DATA CLEANING, PROCESSING,  
ANALYSIS, TRANSFORMATION**

**Automate and  
integrate with rest of  
the infrastructure**



**Data available for  
both analytics and  
visualization**

Disclaimer: These can change based on companies/industry  
Copyright © 2021 by Madhurima Nath

# Who are Data Engineers (DE), Data Scientists (DS) and Machine Learning Engineers (MLE)?

**You can unmute and talk or use the chat.**



# Data Engineer (DE) vs Data Scientist (DS) vs Machine Learning Engineer (MLE)

## **Data engineer:**

- Develops data ingestion pipelines
- Takes data from multiple sources and in multiple formats and combines to single source

## **Data scientist:**

- Finds patterns in the data to obtain insights
- Builds machine learning models to further enhance understanding of data

## **Machine learning engineer:**

- Develops scripts to integrate work of data scientists with the larger framework
- Automates the work of data scientist such that models can be triggered to run without a data scientist

# How do Data Engineers (DE), Data Scientists (DS) and Machine Learning Engineers (MLE) differ from each other?

**You can unmute and talk or use the chat.**

# How they are different/similar?

- Data quality analysis
- SQL queries

**DE**: Take data in a variety of formats, make it available in one queryable

- Integration of the pipelines with the overall architecture
- CI/CD\*

**DS**: Extensive data analysis, build ML models, perform statistical tests

**MLE**: Take ML models and deploy it using automated pipelines

Test different ML models

\*CI/CD: Continuous Integration  
Continuous Deployment

Disclaimer: These can change based on companies/industry  
Copyright © 2021 by Madhurima Nath



# NOTE

Data engineers, data scientists and machine learning engineers make use of existing/pre-built modules or libraries or functions.

They write their own functions or custom codes as well; however, not same as a software engineer or software developer.

Disclaimer: These can change based on companies/industry

Copyright © 2021 by Madhurima Nath

# A Day in a Life of Data Engineer

## Tasks:

- make available in one place datasets ingested from a variety of sources in a variety of formats
- build the data pipelines to ingest streaming and batch data from many sources
- pipelines perform extract, transform, and load (ETL) processes to make the data more usable
- convert from row-oriented formats to column-oriented formats to facilitate analytical queries, partition data, index it, tag it, govern it, etc.

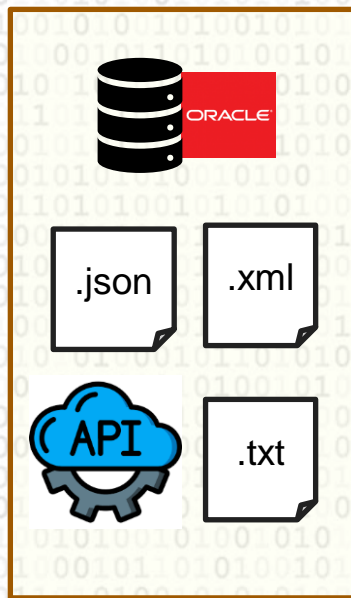
# A Day in a Life of Data Engineer

- make available in one place datasets ingested from a **variety of sources in a variety of formats**

## INTERNAL



## EXTERNAL



## INTERNAL



## manual file sources

\*API: Application Programming Interface: lets your product or service communicate with other products and services without having to know how they're implemented



# A Day in a Life of Data Engineer

- make available in one place datasets ingested from a **variety of sources in a variety of formats**



```
id name age\n01 ABCD 28\n02 XYZ 39\n...\n...
```

```
id,name,age\n01,ABCD,28\n02,XYZ,39
```

```
<log date=May 01, 2021>\n  <message id=1>\n    <person id=01></person>\n    <text> hi, how are you </text>\n    <time> May 01, 2021-16:09:22\n  </time>\n  </message>\n  ...\n  ...
```

```
{\n  "text": "RT@abcland: my best day ..",\n  "created_at": "Fri Apr 15 10:05:45\n    +0000 2020",\n  "retweet_count": 0,\n  "id_str": "561819028170009",\n  "entities": {\n    "user_mentions": [],\n    "hashtags": []\n  }, ...\n}
```



# A Day in a Life of Data Engineer

- make available in one place datasets ingested from a **variety of sources** in a **variety of formats**



```
id name age\n01 ABCD 28\n02 XYZ 39\n...\n...
```

```
id,name,age\n01,ABCD,28\n02,XYZ,39\n...\n...
```

```
<log date=May 01, 2021>\n  <message id=1>\n    <person id=01></person>\n    <text> hi, how are you </text>\n    <time> May 01, 2021-16:09:22\n    </time>\n  </message>\n  ...\n  ...
```

```
{\n  "text": "RT@abcland: my best day ...",\n  "created_at": "Fri Apr 15 10:05:45\n    +0000 2020",\n  "retweet_count": 0,\n  "id_str": "561819028170009",\n  "entities": {\n    "user_mentions": [],\n    "hashtags": []\n  }, ...
```



Can you identify which data formats are structured/unstructured/semi-structured?

You can unmute and talk or use the chat.

# A Day in a Life of Data Engineer

- make available in one place datasets ingested from a **variety of sources in a variety of formats**



```
id name age\n01 ABCD 28\n02 XYZ 39\n...\n...
```

```
id,name,age\n01,ABCD,28\n02,XYZ,39\n...\n...
```

```
<log date=May 01, 2021>\n  <message id=1>\n    <person id=01></person>\n    <text> hi, how are you </text>\n    <time> May 01, 2021-16:09:22\n  </time>\n</message>\n...\n...
```

```
{\n  "text": "RT@abcland: my best day ..",\n  "created_at": "Fri Apr 15 10:05:45\n    +0000 2020",\n  "retweet_count": 0,\n  "id_str": "561819028170009",\n  "entities": {\n    "user_mentions": [],\n    "hashtags": []\n  }, ...\n}
```



**UNSTRUCTURED**

**SEMI-STRUCTURED**

**STRUCTURED**



# A Day in a Life of Data Engineer

- make available in one place datasets ingested from a **variety of sources** in a **variety of formats**

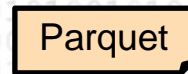


```
id name age\n01 ABCD 28\n02 XYZ 39\n...\n...
```

```
id,name,age\n01,ABCD,28\n02,XYZ,39\n...\n...
```

```
<log date=May 01, 2021>\n  <message id=1>\n    <person id=01></person>\n    <text> hi, how are you </text>\n    <time> May 01, 2021-16:09:22\n  </time>\n  </message>\n  ...\n  ...
```

```
{\n  "text": "RT@abcland: my best day ...",\n  "created_at": "Fri Apr 15 10:05:45\n    +0000 2020",\n  "retweet_count": 0,\n  "id_str": "561819028170009",\n  "entities": {\n    "user_mentions": [],\n    "hashtags": []\n  }, ...
```

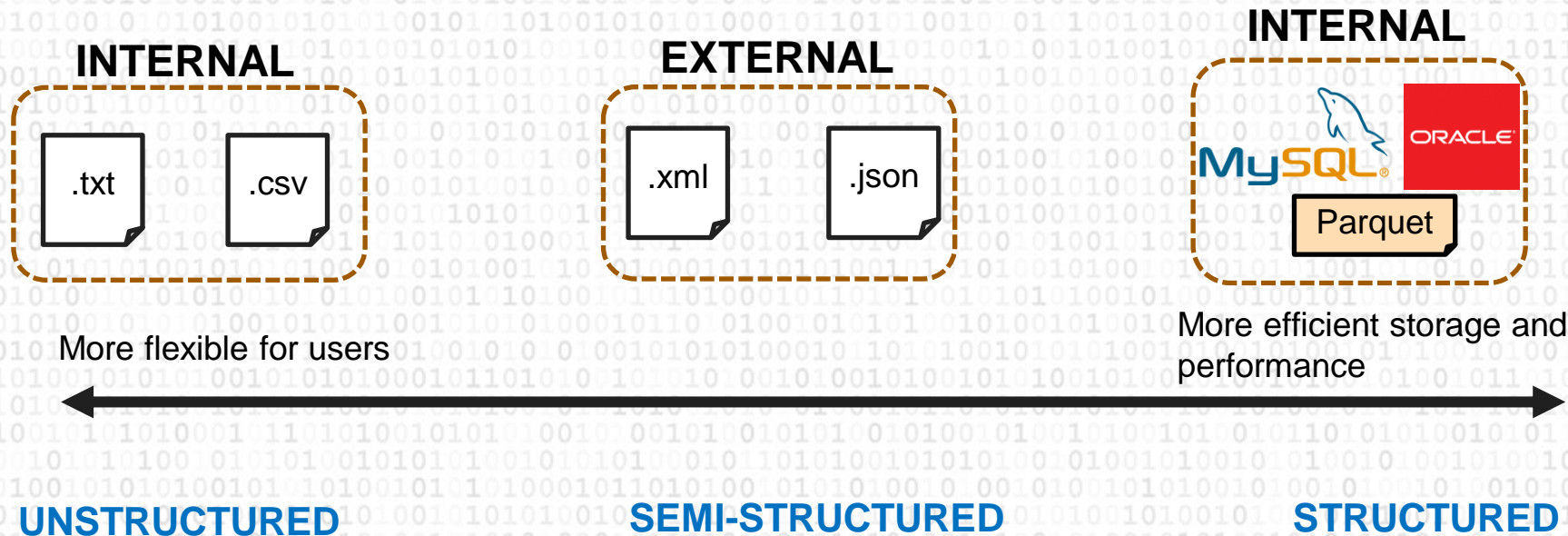


**Which data formats are more user-friendly?**  
**Which data formats are more storage efficient?**

**You can unmute and talk or use the chat.**

# A Day in a Life of Data Engineer

- make available in one place datasets ingested from a **variety of sources in a variety of formats**



Disclaimer: These can change based on companies/industry  
Copyright © 2021 by Madhurima Nath

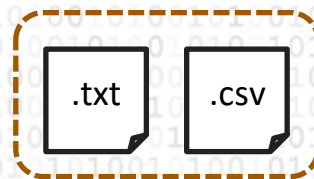
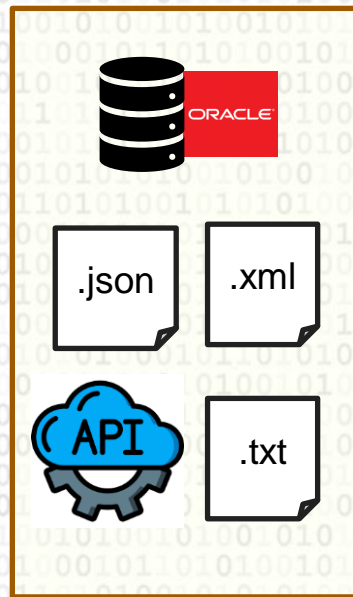
# A Day in a Life of Data Engineer

- Ingest data from **internal and external** sources – could be **unstructured, semi-structured or structured** and make it available at one place

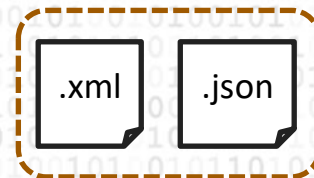
## INTERNAL



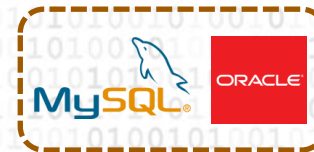
## EXTERNAL



UNSTRUCTURED



SEMI-STRUCTURED



STRUCTURED



# A Day in a Life of Data Engineer

## Tasks:

- build the data pipelines to ingest streaming and batch data from many sources

## **Streaming data – real-time –**

e.g.: log files generated by customers using your mobile or web applications, ecommerce purchases, in-game player activity, information from social networks, financial trading floors, or geospatial services.

# A Day in a Life of Data Engineer

## Tasks:

- build the data pipelines to ingest streaming and batch data from many sources

## Streaming data – real-time –

e.g.: log files generated by customers using your mobile or web applications, ecommerce purchases, in-game player activity, information from social networks, financial trading floors, or geospatial services.

```
from pyspark import SparkContext  
from pyspark.streaming import StreamingContext
```



Calling pre-built spark functions

```
sc = SparkContext(master, appName)  
ssc = StreamingContext(sc, 1)
```

“appName”: name for your application

“master”: Spark/local compute node

“1”: interval of 1 second



(<https://spark.apache.org/docs/latest/streaming-programming-guide.html>)

# A Day in a Life of Data Engineer

## Tasks:

- build the data pipelines to ingest streaming and batch data from many sources

## **Batch data – not real-time –**

e.g.: very large amounts of data when data sources are legacy systems,  
moving data from databases/data warehouse to cloud.



# A Day in a Life of Data Engineer

## Tasks:

- build the data pipelines to ingest streaming and batch data from many sources

## Batch data – not real-time –

e.g.: very large amounts of data when data sources are legacy systems, moving data from databases/data warehouse to cloud.

# read data in csv format

```
spark.read.format("csv")  
.option("mode", "FAILFAST")  
.option("inferSchema", "true")  
.option("path", "path/to/file(s)")  
.schema(someSchema)  
.load()
```

# write the data as parquet format

```
dataframe.write.format("parquet")  
.option("mode", "OVERWRITE")  
.option("dateFormat", "yyyy-MM-dd")  
.option("path", "path/to/file(s)")  
.save()
```

# A Day in a Life of Data Engineer

## Tasks:

- build the data pipelines to ingest streaming (i.e., real-time) and batch data from many sources

## Writing files/data

```
dataframe.write.format("csv")  
.option("mode", "OVERWRITE")  
.option("dateFormat", "yyyy-MM-dd")  
.option("path", "path/to/file(s)")  
.save()
```

Save modes –

append: add output to already existing file at the same location

overwrite: completely overwrite over existing data/file

errorIfExists: throw an error and fail if data/file already exists at the same location

ignore: do nothing, if data/file already exists

# A Day in a Life of Data Engineer

## Tasks:

- build the data pipelines to ingest streaming and batch data from many sources

## Streaming data – real-time –

e.g.: log files generated by customers using your mobile or web applications, ecommerce purchases, in-game player activity, information from social networks, financial trading floors, or geospatial services.

# read data continuously from AWS S3 location

```
val inputDF = spark.readStream.json("s3://logs")
```

# write the data into MySQL database

```
inputDF.groupBy($"action", window($"time", "1 hour")).count()  
  .writeStream.format("jdbc")  
  .start("jdbc:mysql://...")
```



databricks

(<https://databricks.com/blog/2016/07/28/structured-streaming-in-apache-spark.html>)



Disclaimer: These can change based on companies/industry

Copyright © 2021 by Madhurima Nath

WOMEN WHO  
**CODE**



# A Day in a Life of Data Engineer

## Tasks:

- build the data pipelines to ingest streaming and batch data from many sources

## Batch data – not real-time –

e.g.: very large amounts of data when data sources are legacy systems, moving data from databases/data warehouse to cloud.

# read data in JSON format from AWS S3 location

```
val inputDF = spark.read.json("s3://logs")
```

# write the data into MySQL database

```
inputDF.groupBy($"action", window($"time", "1 hour")).count()  
  .write.format("jdbc")  
  .start("jdbc:mysql://...")
```



databricks

<https://databricks.com/blog/2016/07/28/structured-streaming-in-apache-spark.html>



# A Day in a Life of Data Engineer

## Tasks:

- build the data pipelines to ingest streaming and batch data from many sources

**Can you give examples of streaming and batch data?**

**You can unmute and talk or use the chat.**

# A Day in a Life of Data Engineer

## Tasks:

- build the data pipelines to ingest streaming and batch data from many sources

## Examples of streaming data

- e-commerce data
- social media data
- financial trading data

## Examples of batch data

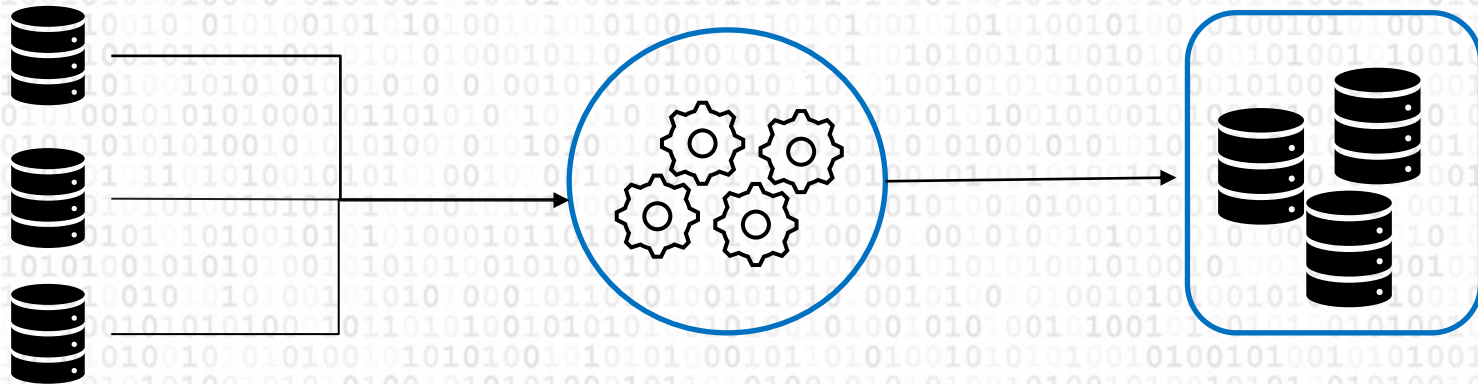
- data from Oracle/Postgres databases
- data from enterprise data warehouses (EDW)
- data from enterprise resource planning (ERP)



# A Day in a Life of Data Engineer

## Tasks:

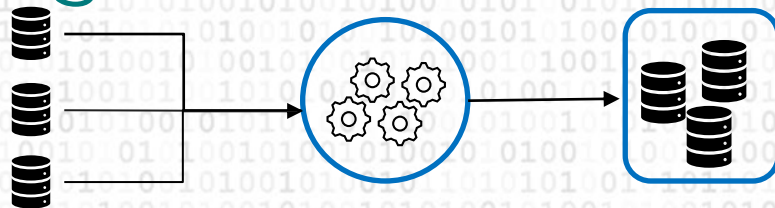
- pipelines perform **Extract, Transform, and Load (ETL)** processes to make the data more usable
- Extract (E) – data from sources (databases)
- Transform (T) – data to match certain common defined format (for business purposes)
- Load (L) – load re-formatted data to data warehouse



# A Day in a Life of Data Engineer

Tasks:

- pipelines perform Extract, Transform, and Load (ETL) processes



Example:

Suppose you want to extract data from multiple databases and put it in cloud data storage.

## How would you extract the data to load into cloud data storage?

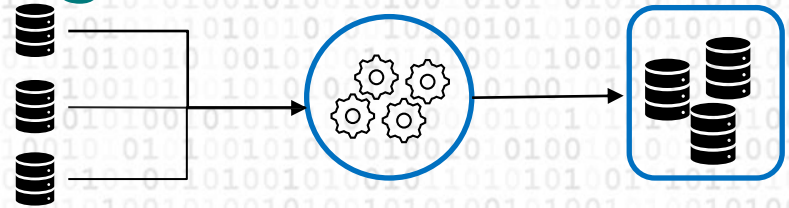
**You can unmute and talk or use the chat.**

Disclaimer: These can change based on companies/industry  
Copyright © 2021 by Madhurima Nath

# A Day in a Life of Data Engineer

Tasks:

- pipelines perform Extract, Transform, and Load (ETL) processes



Example:

Suppose you want to extract data from multiple databases and put it in cloud data storage.

Step 1: How to handle day 0 load (everything that currently exist)?

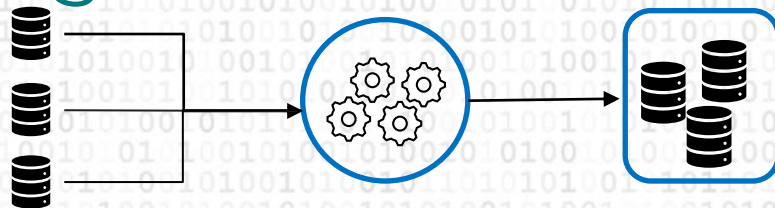
Step 2: How to handle new incremental data?



# A Day in a Life of Data Engineer

## Tasks:

- pipelines perform Extract, Transform, and Load (ETL) processes



## Example:

Suppose you want to extract data from multiple databases and put it in cloud data storage.

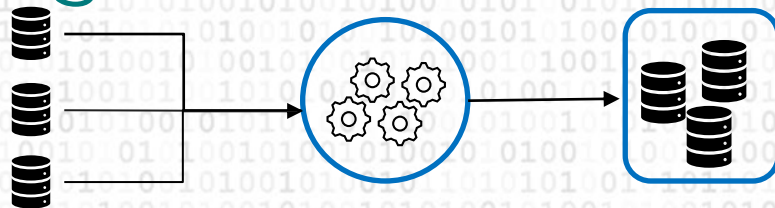
### Step 1: Day 0 (already existing data) load

- Partition the data
- Keep track of the partitions
- Load the partitioned data

# A Day in a Life of Data Engineer

## Tasks:

- pipelines perform Extract, Transform, and Load (ETL) processes



## Example:

Suppose you want to extract data from multiple databases and put it in cloud data storage.

Step 1: Day 0 (already existing data) load

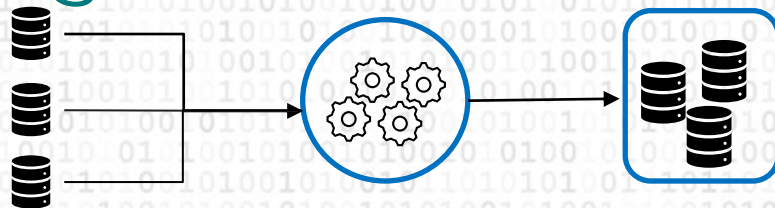
Step 2: New incremental data load

- decide on the frequency (daily/weekly/monthly)
- refresh the tables with new data

# A Day in a Life of Data Engineer

Tasks:

- pipelines perform Extract, Transform, and Load (ETL) processes



Example:

Suppose you want to extract data from multiple databases and put it in cloud data storage. (Code for Azure data storage on Databricks)

```
table_id = dbutils.widgets.get("table_id")
```

```
job_id = dbutils.widgets.get("job_id")
```

→ get source table id from source

→ get job id

```
storageCfg = DatalakeStorageConfig()
```

```
storageCfgPath = ConfigLoader(DatalakeStorageConfig().cfg_path)
```

→ load configuration of cloud storage

→ load configuration path of cloud storage

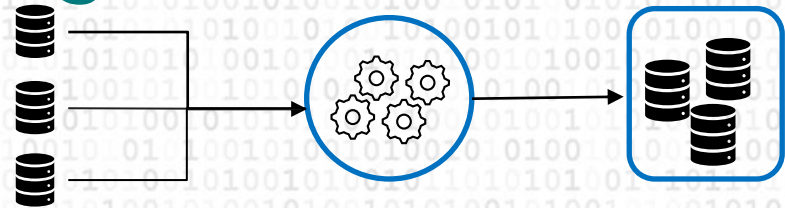
Disclaimer: These can change based on companies/industry  
Copyright © 2021 by Madhurima Nath



# A Day in a Life of Data Engineer

Tasks:

- pipelines perform Extract, Transform, and Load (ETL) processes



Example:

Suppose you want to extract data from multiple databases and put it in cloud data storage. (Code for Azure data storage on Databricks)

```
table_name = storageCfgPath.get_dataset_by_id(table_id).table_name
table_folder = storageCfgPath.get_dataset_by_id(table_id).folder_name
source_path = f'/{storageCfg.root_dir}/{table_name}'
dest_path = f'/{storageCfg.root_dir}/{table_folder}'
```

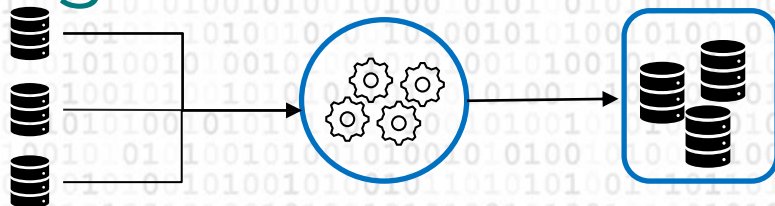
- set source table name
- set destination table folder
- set source table path as parametrized path
- set target table path as parametrized path

Disclaimer: These can change based on companies/industry  
Copyright © 2021 by Madhurima Nath

# A Day in a Life of Data Engineer

Tasks:

- pipelines perform Extract, Transform, and Load (ETL) processes



```
import json
from pyspark.sql.types import *
from pyspark.sql import functions as F
```

—————> Call pre-built python and spark functions

```
source_df = spark.read.format('parquet')
               .load(source_path).filter(F.col('job_id') == job_id)

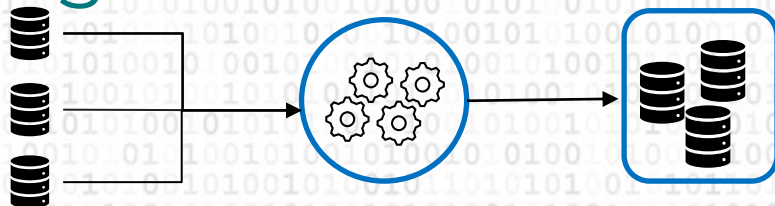
if source_df.count() > 0:
    (source_df.write.format('format').mode('merge').partitionBy('job_id').save(source_path) )
    try:
        sqlCmd = 'CREATE TABLE IF NOT EXISTS table' + table_name + " USING LOCATION '{}'"
               .format(dest__path)
        spark.sql(sqlCmd)
    except:
        OSError as e
```

Disclaimer: These can change based on companies/industry  
Copyright © 2021 by Madhurima Nath

# A Day in a Life of Data Engineer

Tasks:

- pipelines perform Extract, Transform, and Load (ETL) processes



```
import json
```

```
from pyspark.sql.types import *
```

```
from pyspark.sql import functions as F
```

→ Call pre-built python and spark functions

```
source_df = spark.read.format('parquet')
```

```
.load(source_path).filter(F.col('job_id') == job_id)
```

→ Read batch files (format – parquet)  
Filter these files with job id

```
if source_df.count() > 0:
```

```
(source_df.write.format('format').mode('merge').partitionBy('job_id').save(source_path) )
```

```
try:
```

```
sqlCmd = 'CREATE TABLE IF NOT EXISTS table' + table_name + " USING LOCATION '{ }'
```

```
.format(dest__path)
```

```
spark.sql(sqlCmd)
```

```
except:
```

```
    OSError as e
```

Disclaimer: These can change based on companies/industry

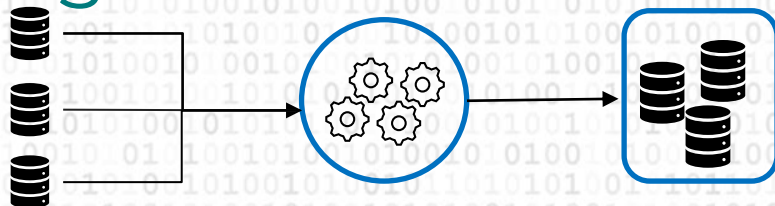
Copyright © 2021 by Madhurima Nath



# A Day in a Life of Data Engineer

Tasks:

- pipelines perform Extract, Transform, and Load (ETL) processes



```
import json
```

```
from pyspark.sql.types import *
```

```
from pyspark.sql import functions as F
```

Call pre-built python and  
spark functions

```
source_df = spark.read.format('parquet')
```

```
.load(source_path).filter(F.col('job_id') == job_id)
```

Read batch files (format – parquet)  
Filter these files with job id

```
if source_df.count() > 0:
```

```
( source_df.write.format('format').mode('merge').partitionBy('job_id').save(source_path) )
```

```
try:
```

```
sqlCmd = 'CREATE TABLE IF NOT EXISTS table' + table_name + " USING LOCATION '{}'"
```

```
.format(dest__path)
```

```
spark.sql(sqlCmd)
```

```
except:
```

```
    OSError as e
```

Save data to  
defined path  
Create table if table  
doesn't exist at  
defined location

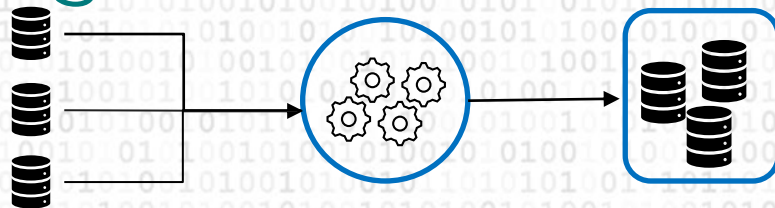
Disclaimer: These can change based on companies/industry  
Copyright © 2021 by Madhurima Nath

WOMEN WHO  
**CODE**

# A Day in a Life of Data Engineer

Tasks:

- pipelines perform Extract, Transform, and Load (ETL) processes



**How would you handle streaming data?**  
**What would be the steps?**

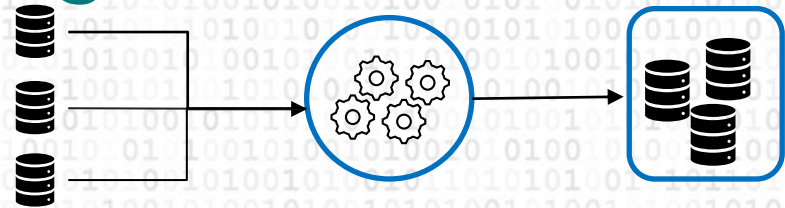
**You can unmute and talk or use the chat.**

Disclaimer: These can change based on companies/industry  
Copyright © 2021 by Madhurima Nath

# A Day in a Life of Data Engineer

## Tasks:

- pipelines perform Extract, Transform, and Load (ETL) processes



## Streaming data

### Step 1: Day 0 load – historical data load

- partition the data
- load the partitioned data

### Step 2: Real-time incremental data

- decide on the frequency of refresh – seconds/minutes/hours
- refresh data in the tables



# Tech stack for data engineers

Languages: **SQL**, **NoSQL**, **Python**, **Java**, **Scala**  
**Spark**, **Databricks**, **Kafka**, **Hadoop**, **CI/CD\*** framework



**nosql**



**databricks**



\*CI/CD: Continuous Integration Continuous Deployment

Disclaimer: These can change based on companies/industry

# Backend Study Group

- WWCode [Presentation](#) and [Demo](#)
- [WWCode YouTube channel](#):
  - March 25, 2021 session recording: [Backend Engineering](#)
  - April 8, 2021 session recording: [Java Microservice and REST API Demo](#)
  - April 22, 2021 session recording: [Creational Design Patterns](#)
  - May 20, 2021 session recording: [Structural Design Patterns](#)
  - June 3, 2021 session recording: [Behavioral Design Patterns](#)
- **Resources:**
  - [spark documentation](#)
  - [databricks documentation](#)
  - [snowflake documentation](#)
  - [azure data engineer certification course](#)
  - [google cloud data engineer certification course](#)
  - [aws big data certification](#)
  - [Udacity data engineering interview questions](#)

