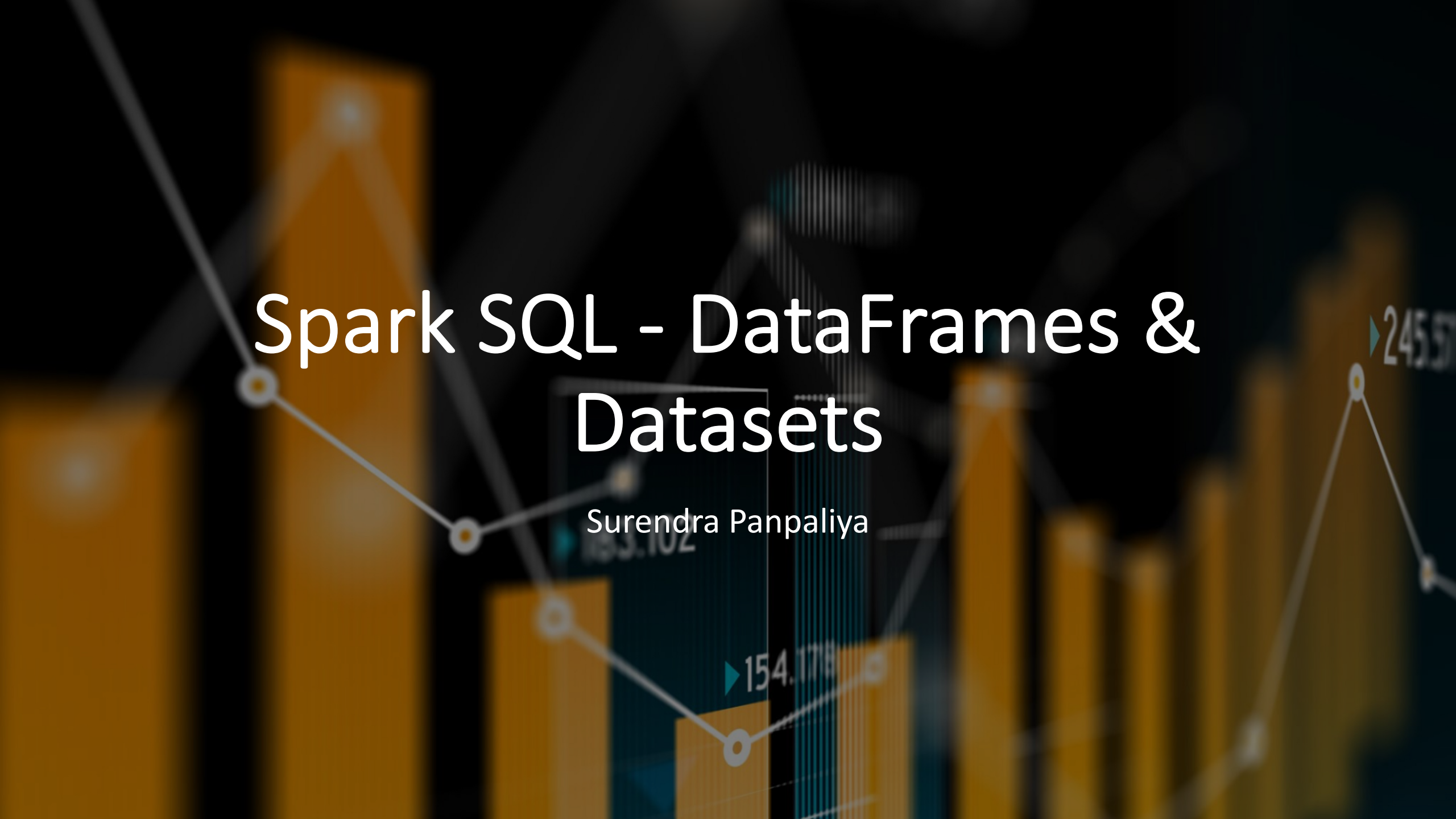


Spark SQL - DataFrames & Datasets

Surendra Panpaliya

Spark SQL

Spark SQL lets you query

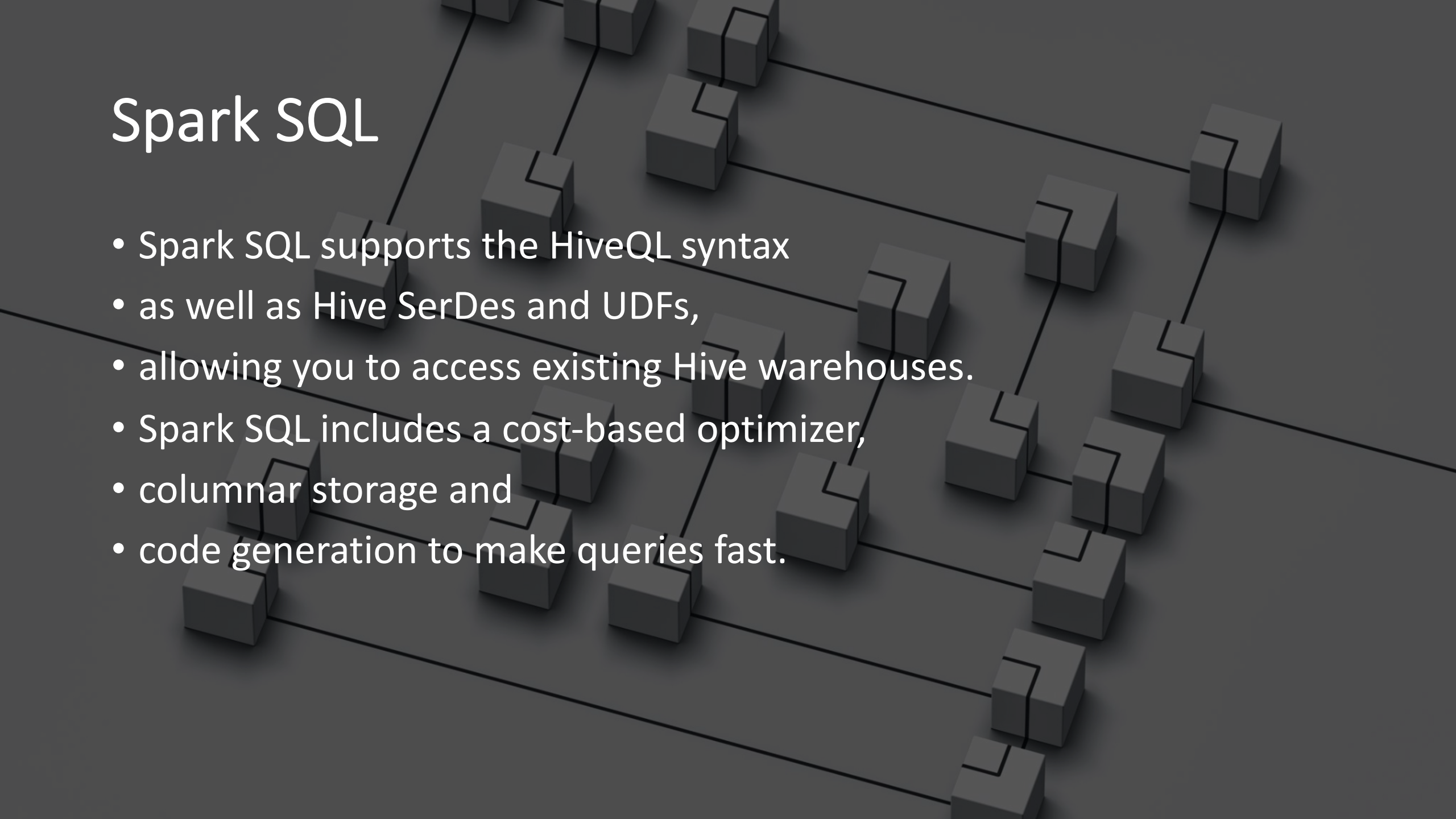
structured data inside Spark programs,

using either SQL or

a familiar DataFrame API.

Spark SQL

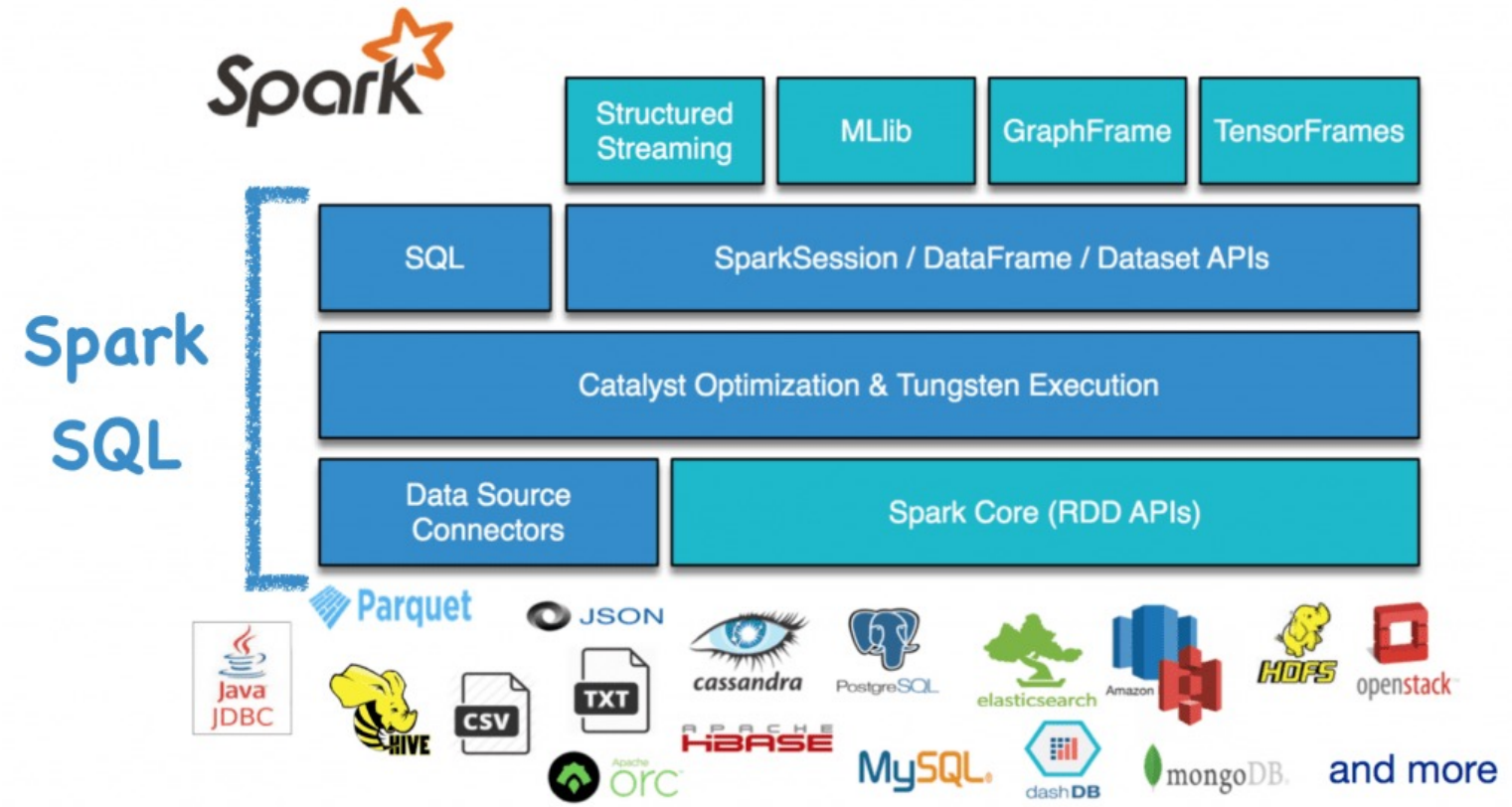
- DataFrames and SQL provide
- a common way to access
- a variety of data sources, including
- Hive, Avro, Parquet, ORC, JSON, and JDBC.
- You can even join data across these sources.



Spark SQL

- Spark SQL supports the HiveQL syntax
- as well as Hive SerDes and UDFs,
- allowing you to access existing Hive warehouses.
- Spark SQL includes a cost-based optimizer,
- columnar storage and
- code generation to make queries fast.

Spark SQL


Spark
Structured Streaming
MLlib
GraphFrame
TensorFrames
Spark SQL
SQL
SparkSession / DataFrame / Dataset APIs
Catalyst Optimization & Tungsten Execution
Data Source Connectors
Spark Core (RDD APIs)
Parquet
JSON
Java JDBC
HIVE
CSV
TXT
cassandra
PostgreSQL
elasticsearch
Amazon
HDFS
openstack
APACHE HBASE
Apache orc
MySQL
dashDB
mongoDB.
and more

DataFrames

A DataFrame is a Dataset organized

into named columns.

It is conceptually equivalent

to a table in a relational database or

a data frame in R/Python

but with richer optimizations

under the hood.

DataFrames

- DataFrame is represented by
- a Dataset of Rows.
- In the Scala API,
- DataFrame is simply a type alias of Dataset[Row].
- DataFrames are untyped and
- the elements within DataFrames are Rows.

DataFrames

- scala> df.show
- +--------------+--------+-----------+--------------------+--------+
- | capital|currency|independent| name| region|
- +--------------+--------+-----------+--------------------+--------+
- | Canberra| AUD| true| Australia| Oceania|
- | London| GBP| true| United Kingdom| Europe|
- |Washington D.C.| USD| true|United States of ...|Americas|
- | Paris| EUR| true| France| Europe|
- | Tokyo| JPY| true| Japan| Asia|
- | Dublin| EUR| true| Ireland| Europe|

Datasets

Dataset is a data structure in SparkSQL

which is strongly typed and

is a map to a relational schema.

It represents structured queries with encoders.

Datasets

It is an extension to data frame API.

Spark Dataset provides both

type safety and object-oriented programming interface.

Encounter the release of the dataset in Spark 1.6.

Datasets Encoder

The **encoder** is primary concept
in *serialization* and *deserialization* (**SerDes**)

framework in Spark SQL.

Encoders translate between

JVM objects and Spark’s internal binary format.

Spark has built-in encoders which are very advanced.

Datasets

They generate bytecode

to interact with **off-heap** data.

An encoder provides

on-demand access to individual attributes

without having to de-serialize an entire object.

Datasets

Spark Dataset is structured and lazy query expression that triggers the **action**.
Internally dataset represents a logical plan.

The **logical plan** tells the computational query that we need to produce the data.

the logical plan is a base catalyst query plan for the logical operator to form a logical query plan.

Datasets

It provides:

- The convenience of RDD.
- Performance optimization of DataFrame.
- Static type-safety of **Scala**.

Datasets

DATASETS PROVIDES A MORE

FUNCTIONAL PROGRAMMING INTERFACE

TO WORK WITH STRUCTURED DATA.

Need of Dataset in Spark

To overcome the limitations of RDD and Dataframe,

Dataset emerged.

In DataFrame, there was no provision

for **compile-time type safety**.

Data cannot be altered

without knowing its structure.

Need of Dataset in Spark

In RDD there was

no automatic optimization.

So for optimization,

we do it manually

when needed.

Features of Dataset in Spark

a. Optimized Query

b. Analysis at compile time

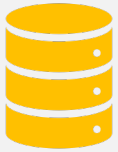
c. Persistent Storage

d. Inter-convertible

Features of Dataset in Spark

e. Faster Computation

f. Less Memory Consumption



g. Single API for Java and Scala

a. Optimized Query

Dataset in Spark provides Optimized query

using **Catalyst Query Optimizer** and **Tungsten**.

Catalyst Query Optimizer is an

execution-agnostic framework.

It represents and manipulates a data-flow graph.

a. Optimized Query

Data flow graph is a tree of expressions and

relational operators.

By optimizing the Spark job

Tungsten improves the execution.

Tungsten emphasizes the hardware architecture

of the platform on which Apache Spark runs.

b. Analysis at compile time

Using Dataset we can check syntax and

analysis at compile time.

It is not possible using Dataframe,

RDDs or regular SQL queries.

c. Persistent Storage

Spark Datasets are both

serializable and Queryable.

Can save it to persistent storage.

d. Inter-convertible

We can convert the Type-safe dataset to an “untyped” DataFrame.

To do this task Dataset holder provide

three methods for conversion from

Seq[T] or RDD[T] types to Dataset[T]

d. Inter-convertible

toDS(): Dataset[T]

toDF(): DataFrame

toDF(colNames: String*): DataFrame

e. Faster Computation

The implementation of the

Dataset is much faster

than the RDD implementation.

Increases the performance of the system.

e. Faster Computation

For the same performance

using the RDD,

the user manually considers

how to express computation

that parallelizes optimally.

f. Less Memory Consumption

While caching,

it creates a more optimal layout.

Spark knows the structure of data

in the dataset.

g. Single API for Java and Scala

It provides a single interface

for **Java** and **Scala**.

This unification ensures

we can use Scala interface,

code examples from both languages.

RDD vs DataFrames vs Datasets

RDD	DataFrames	Datasets
Functional transformations	Relational	Functional & Relational transformations
No type safety	No type safety	Type-safe
In-memory JVM objects resulting in GC & java serialization overheads	Tungsten execution engine - Off-Heap Memory Management using binary in-memory data representation	Tungsten execution engine - Off-Heap Memory Management using binary in-memory data representation
	JIT code generation	JIT code generation

RDD vs DataFrames vs Datasets

RDD	DataFrames	Datasets
	Sorting/suffling without deserialization	Sorting/suffling without deserialization
	No encoders	Encoders - generate byte code and provide on-demand access to attributes without the need for deserialization

RDD vs DataFrames vs Datasets

DataFrames	Datasets
DataFrames are untyped – No type check by the compiler	Datasets are typed distributed collections of data - compile- - time type safety
Transformations on dataFrames are untyped	Datasets include both untyped and typed transformations

Datasets Summary

- A Dataset is a distributed collection of data.
- Dataset is a new interface added in Spark 1.6
- that provides the benefits of RDDs
- (strong typing, ability to use powerful lambda functions)
- with the benefits of
- Spark SQL’s optimized execution engine.

Datasets Summary

- Datasets are typed distributed collections
- of data and it unifies DataFrame and RDD APIs.
- Datasets require structured/semi-structured data.
- Schemas and Encoders are part of Datasets.

Hands On

// vertically

spark.createDataset(Seq(1, 2))