

# Programming for Big Data

## Lecture 1

### Data Processing with Spark

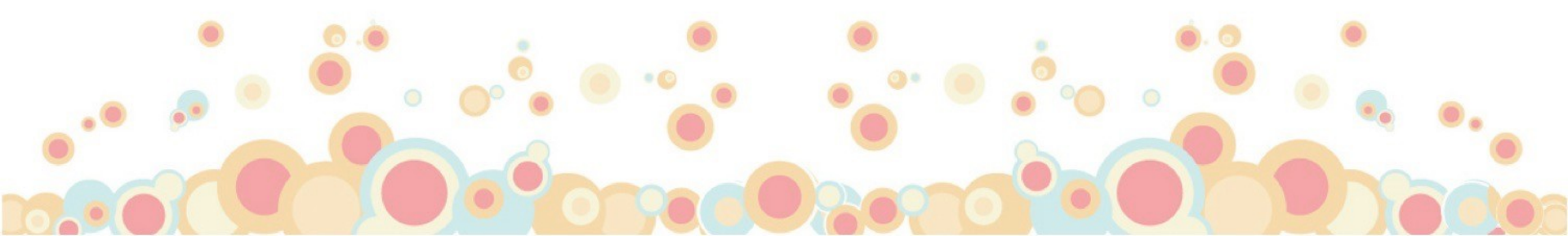
Dr. Bojan Božić

Dublin institute of Technology



## Preliminaries

1. Data Warehousing
2. Big Data Architectures
  - Batch Processing - Data Lakes
  - Batch and Stream Processing - Lambda
  - Stream Processing - Kappa
3. Apache Spark
  - What is Spark
  - What is Spark used for?
  - Spark Architecture
4. Summary



# PRELIMINARIES

# My Details

Name: Dr. Bojan Božić

Email: [bojan.bozic@dit.ie](mailto:bojan.bozic@dit.ie)

Details on SPARK labs:

<https://ceadar.dit.ie/bojan.bozic/SPARK/>

Final assignment will be  
published on webcourses.

Any Questions, Please email me!



# Books and Resources

## Mastering Spark

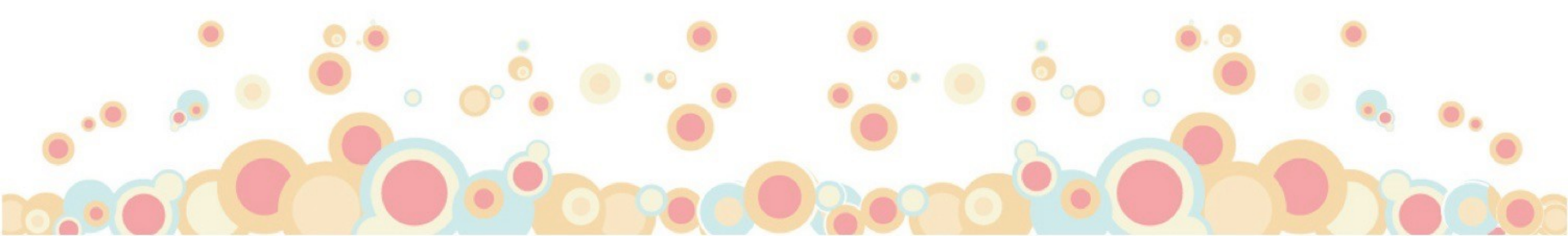
[www.packtpub.com/big-data-and-business-intelligence/mastering-apache-spark](http://www.packtpub.com/big-data-and-business-intelligence/mastering-apache-spark)

## Apache Spark From Inception to Production

[http://info.mapr.com/rs/mapr/images/Getting\\_Started\\_With\\_Apache\\_Spark.pdf](http://info.mapr.com/rs/mapr/images/Getting_Started_With_Apache_Spark.pdf)

## Spark Programming Guide

<http://spark.apache.org/docs/latest/programming-guide.html>

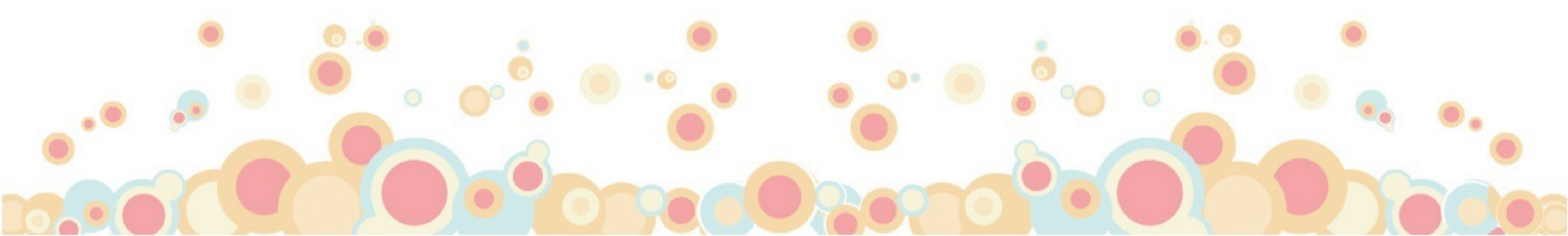


# Course Details

Classes: Wednesday 18:30 - 21:30

- 4 labs with 3 lab assignments (voluntary)
- Final assignment starts in last (4<sup>th</sup>) lab

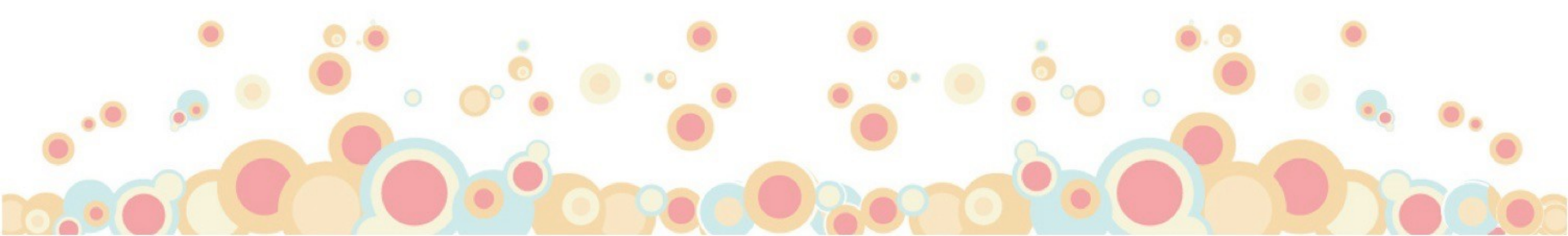
Grading: Only final assignment is graded  
(100%)





<http://spark.apache.org/>

<https://community.cloud.databricks.com>

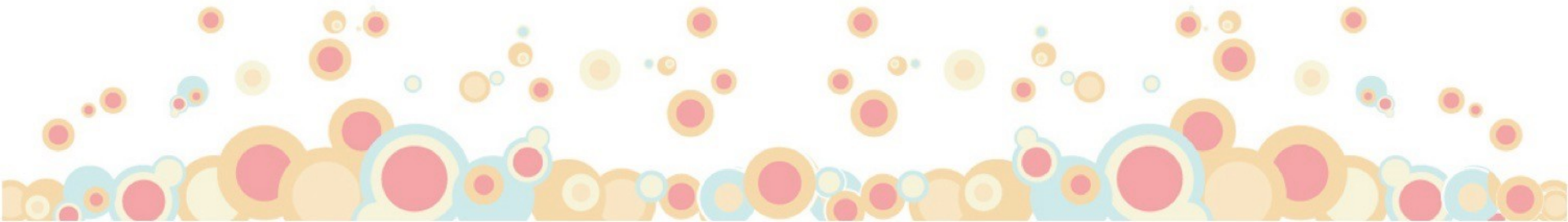


# DATA WAREHOUSING



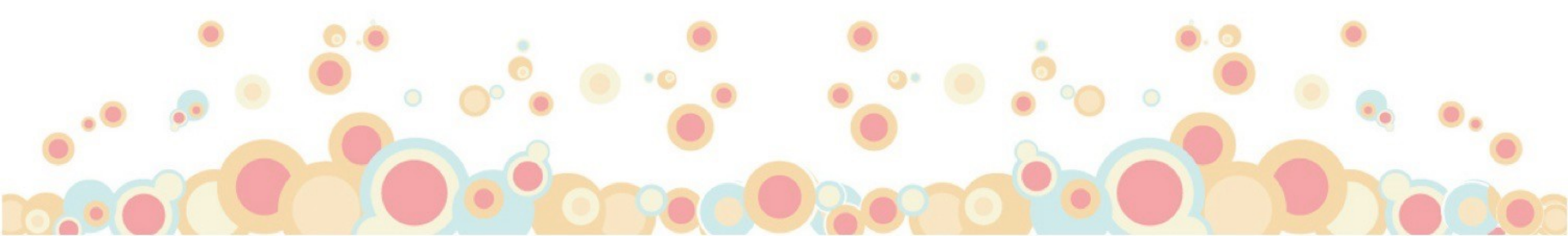
# Data Warehousing

- Since the 1970s, organisations have gained competitive advantage by **automating** business processes to offer more efficient and **cost-effective** services
- This resulted in accumulation of growing amounts of data in **operational** databases
- Organisations now focus on using operational data to support decision-making
- However, **operational systems were never designed to support such business activities**
- Behold the data warehouse was born!



# Data Warehousing

- A data warehouse is a **relational database** that is designed for query and analysis rather than for transaction processing
- It usually contains **historical data** derived from transaction data, but it can include data from other sources
- It separates **analysis workload** from **transaction workload** and enables an organisation to consolidate data from several sources to business users



# Data Warehousing

*“A copy of transaction data, specifically structured for query and analysis”*

—Ralph Kimball

*“A data warehouse is a **simple, complete and consistent** store of data obtained from a **variety of sources** and made available to end users in a way they can **understand and use it in a business context**”*

—IBM

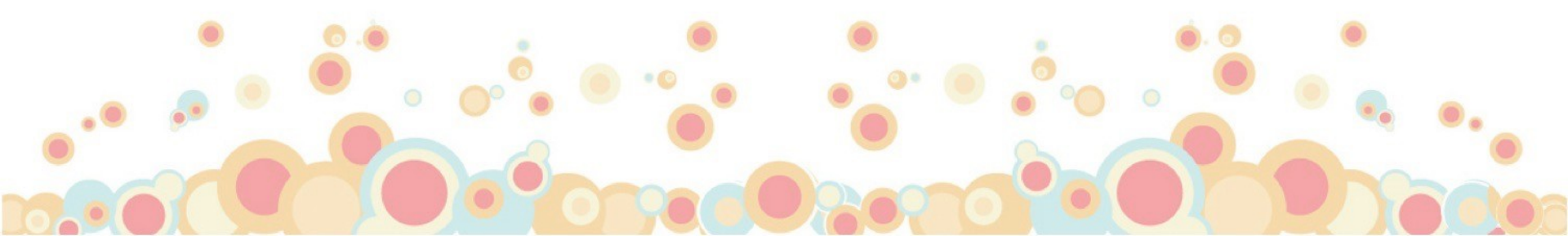
*“A data warehouse is a **subject-oriented, integrated, time-variant, and non-volatile** collection of data in support of management’s decision-making process”*

—Bill Inmon

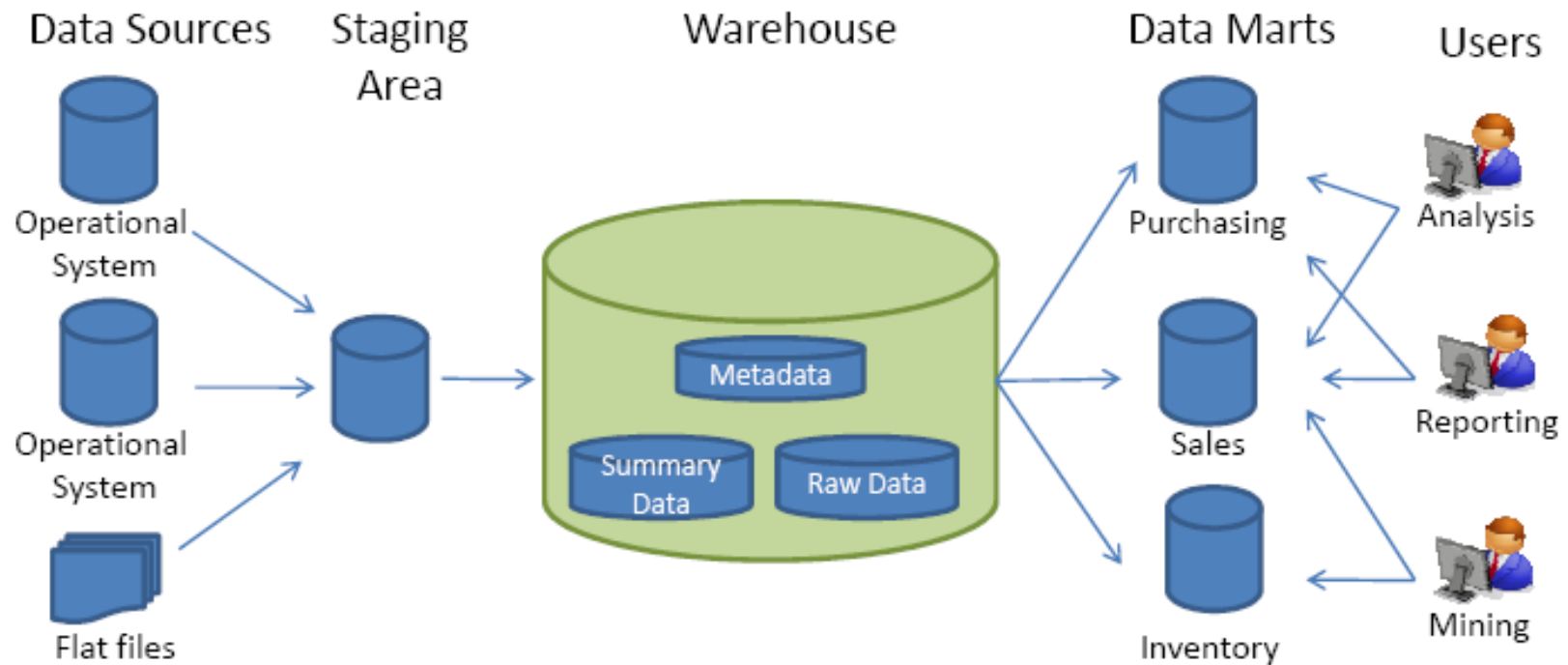
A decorative footer at the bottom of the slide featuring a collection of overlapping circles in various colors including orange, yellow, pink, and light blue, creating a bubbly or cellular effect.

# Data Warehousing

- Data Warehousing has two aspects
  - Data Warehouse Data:
    - An integrated decision support database
  - Data Warehouse Process (includes Governance):
    - **Software** and **Methodologies** used to gather, cleans, transform, and store data from a variety of different databases



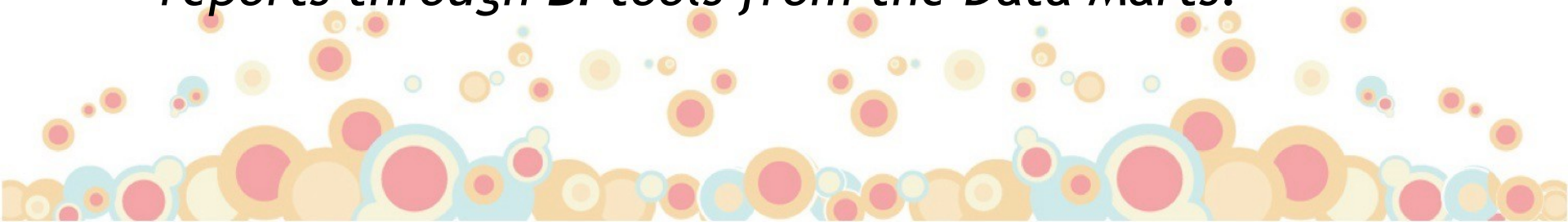
# Data Warehousing



# Data Warehousing


*A data warehouse architecture typically consists of:*

- *The **Data Sources** area contains raw transactional data that is regularly updated.*
- *The **Data Staging area** is all cleansing and integration processes.*
- *The **Data Warehouse** contains a complete **integrated database** set generated from Data Sources.*
- *The **Data Marts** are subsets of the Data Warehouse which provide a domain specific access layer to the Data Warehouse.*
- *The **user interfaces** allow users to generate analysis and reports through **BI** tools from the Data Marts.*





# Data Warehousing

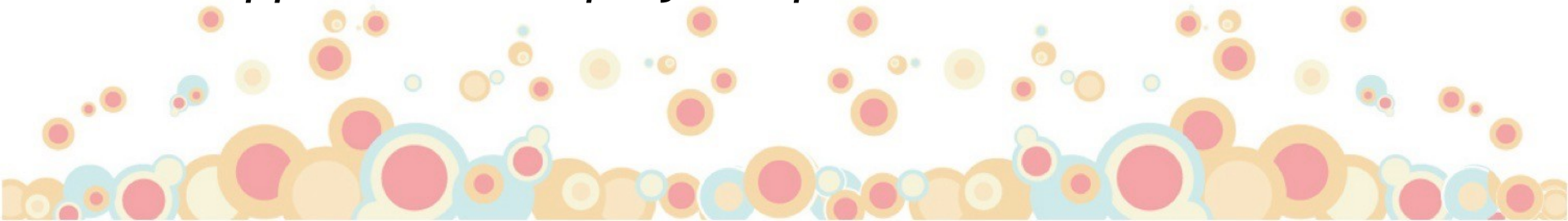
1. **Difficult to develop** - need to predefine questions, involve stakeholders, define scope and restricted exploration, lengthy duration of project, homogenisation of data
  2. **Difficult to maintain** - need to define an ETL process, set up a Schema change process, define methods of governance
  3. **Downstream Transformation** - early transformation can be immutable and result in lost or leaked data
  4. **Difficult to scale** - Newer technology such as **MPP** enables scaling but this can be an issue with legacy and cost
  5. **Always Limited** - can only answer predefined questions and not developer for explorative or interactive analysis
  6. **Restricted Schemas** - use dimensional models such as star or snowflake schema
  7. **Need for more technology to apply stats and modelling**
- 

# BIG DATA ARCHITECTURES



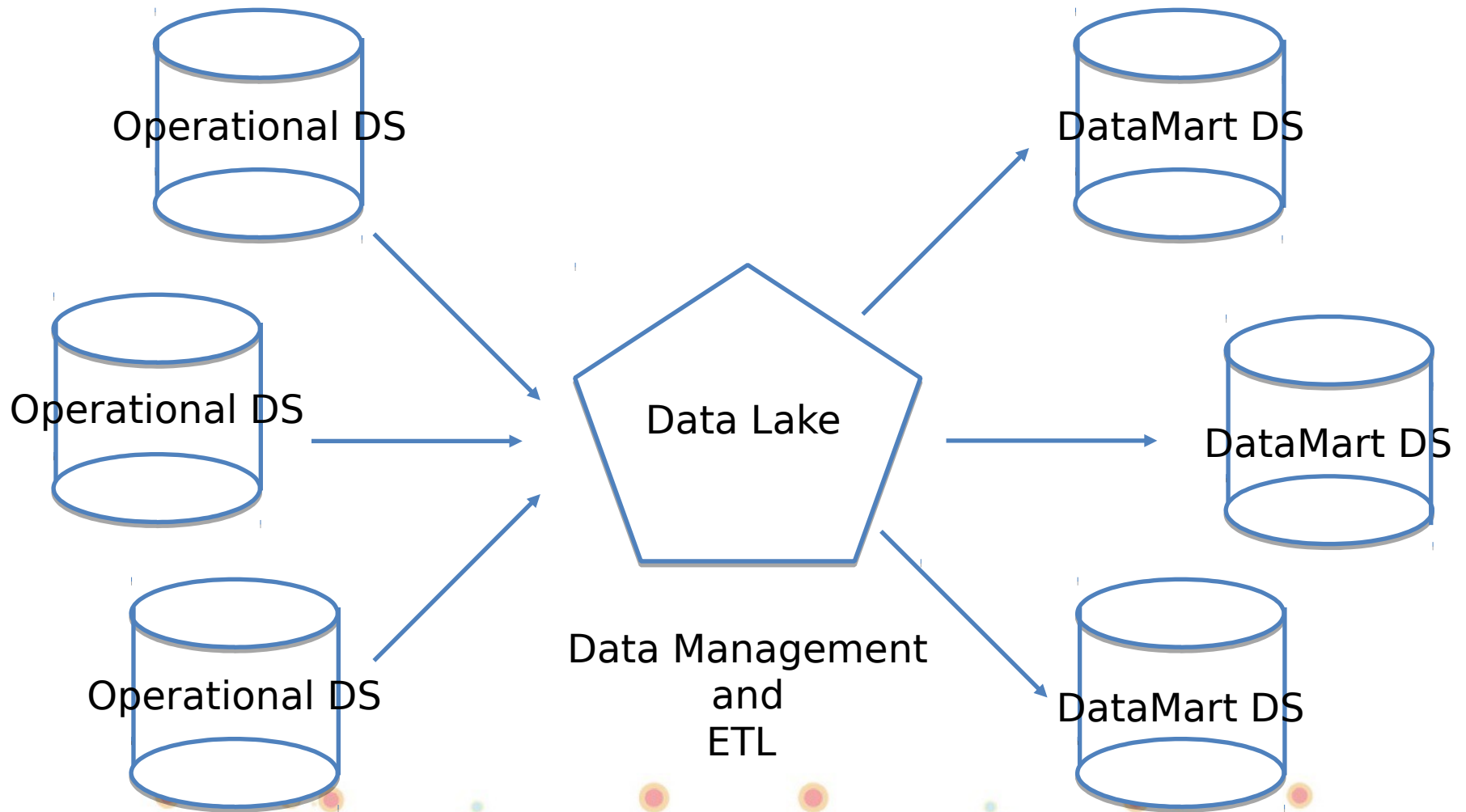
# Big Data Architectures

1. *Addresses the 4 Vs*
  - *Velocity - Streaming data*
  - *Veracity - Gaps in the data, dirty data, etc*
  - *Volume - Massive scale*
  - *Variety - Unstructured and structured*
2. *Run on commodity hardware*
3. *Address batch and stream processing*
4. *Supports advanced statistical modelling*
5. *Supports model deployment*
6. *Supports Data replay / reprocess*



# BATCH PROCESSING - DATA LAKES

# Data Lakes

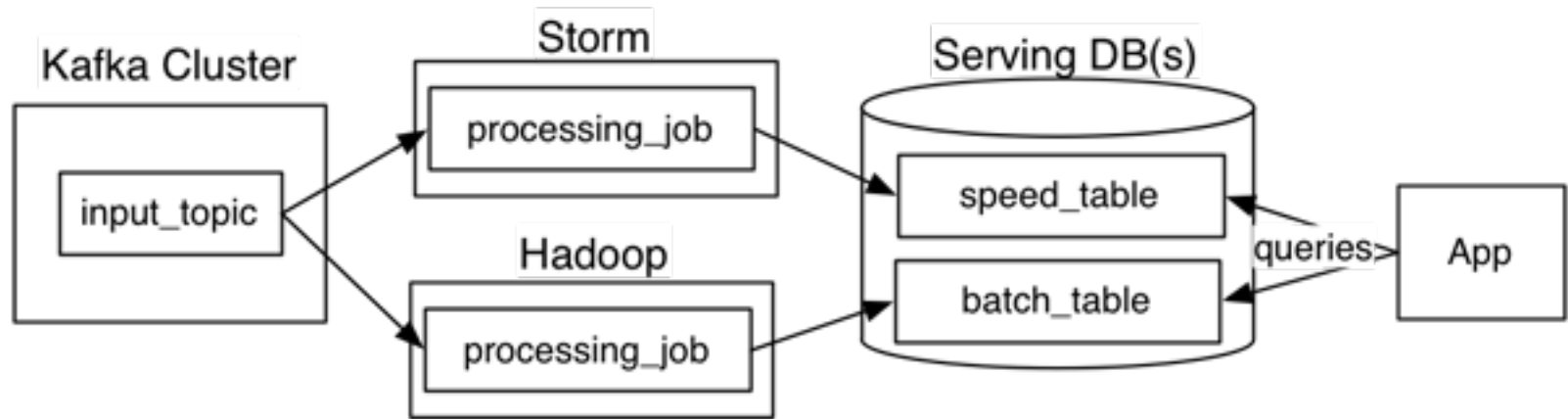


# Data Lakes

1. ***Retain all Data*** - don't throw away anything, everything is kept in the original form, no upstream transformation
2. ***Support all data types*** - Structured (csv, tsv, rdf, sql) and un-structured (text, numeric), no homogenisation required, use HDFS or Key-Value Store
3. ***Supports all users*** - data scientists, business users etc
4. ***Easily adapt to change***
5. ***Provide faster insights*** - Schema-on-read so there is no need to define complex ETL and homogenisation efforts
6. ***Requires strict governance and data engineering efforts***
7. ***Supports Batch Processing***

# BATCH AND STREAM PROCESSING - LAMBDA

# Lambda Architecture

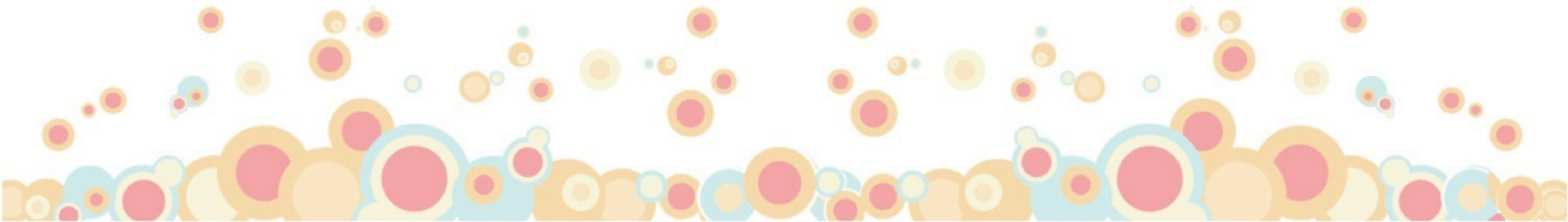


## Resources:

- <https://www.oreilly.com/ideas/questioning-the-lambda-architecture>
- <http://radar.oreilly.com/2015/02/improving-on-the-lambda-architecture-for-streaming-analysis.html>
- <http://lambda-architecture.net/>

# Lambda Architecture

*A complex architecture that requires two code bases to be maintained, each applying functions independently on data. Given the stress of software production, each code base can fall out of step resulting in different functions being applied to the same data at different stages.*

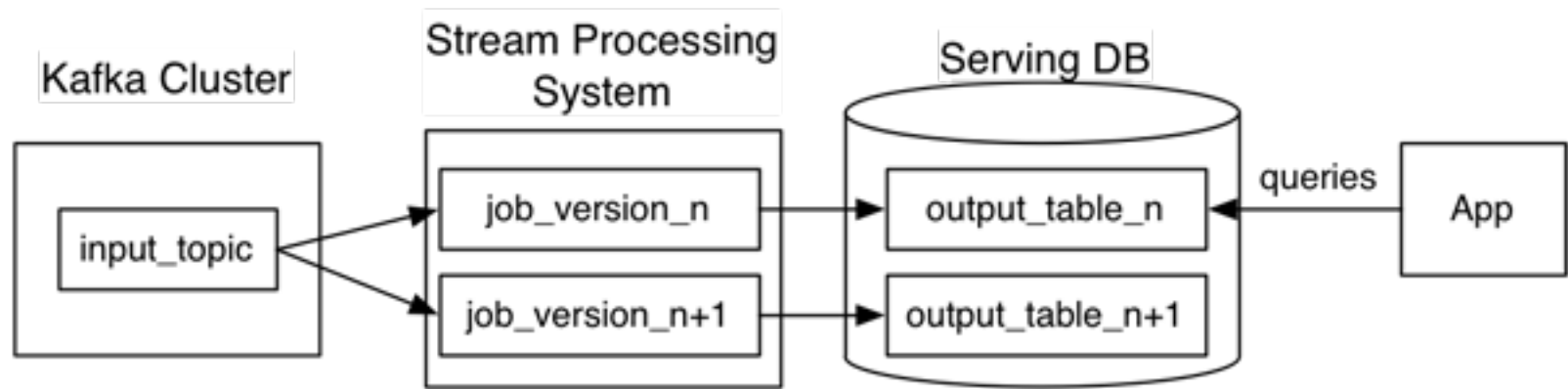


**STREAM PROCESSING -**

**KAPPA**



# Kappa Architecture



## Resources:

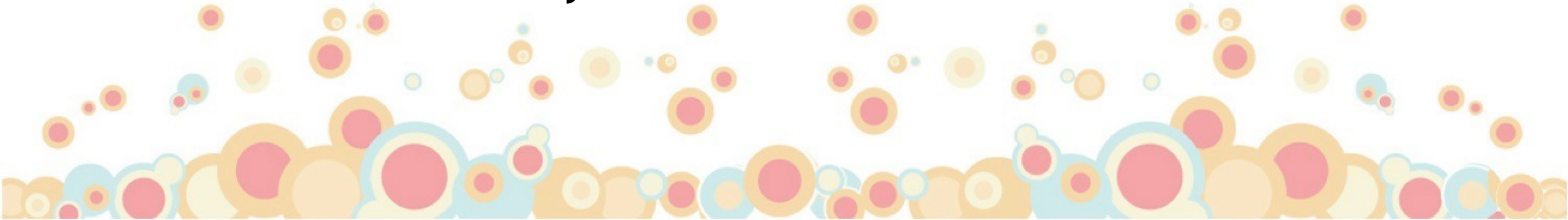
<http://milinda.pathirage.org/kappa-architecture.com/>

<https://community.mapr.com/community/exchange/blog/2017/04/04/the-trouble-with-kappa-architecture>

# Big Data Architecture

Data Processing has developed in a way that requires new systems that support a variety of mechanisms to stream, manage, process and interrogate data.

The ecosystem was divided between a variety of Apache Incubator projects until the arrival of Spark - a general purpose data processing engine that has gained momentum in the last several years.

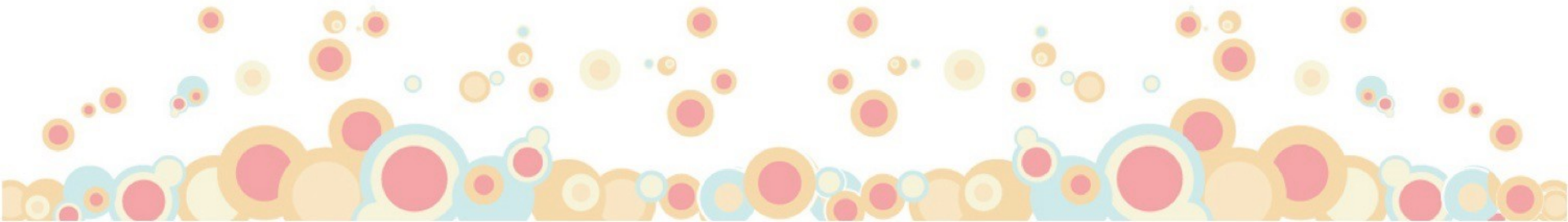


A decorative footer at the bottom of the slide featuring a horizontal row of overlapping circles in various colors including yellow, orange, pink, and light blue. Some circles are solid, while others have concentric outlines.

**APACHE SPARK**

# What is Spark?

Spark, an incubated project of the Apache Software Foundation, is a general-purpose data processing engine, suitable for use in a wide range of circumstances including, **interactive queries across large data sets, processing of streaming data, large scale machine learning and large-scale ETL.**




# What is Spark?

Spark has several core benefits:

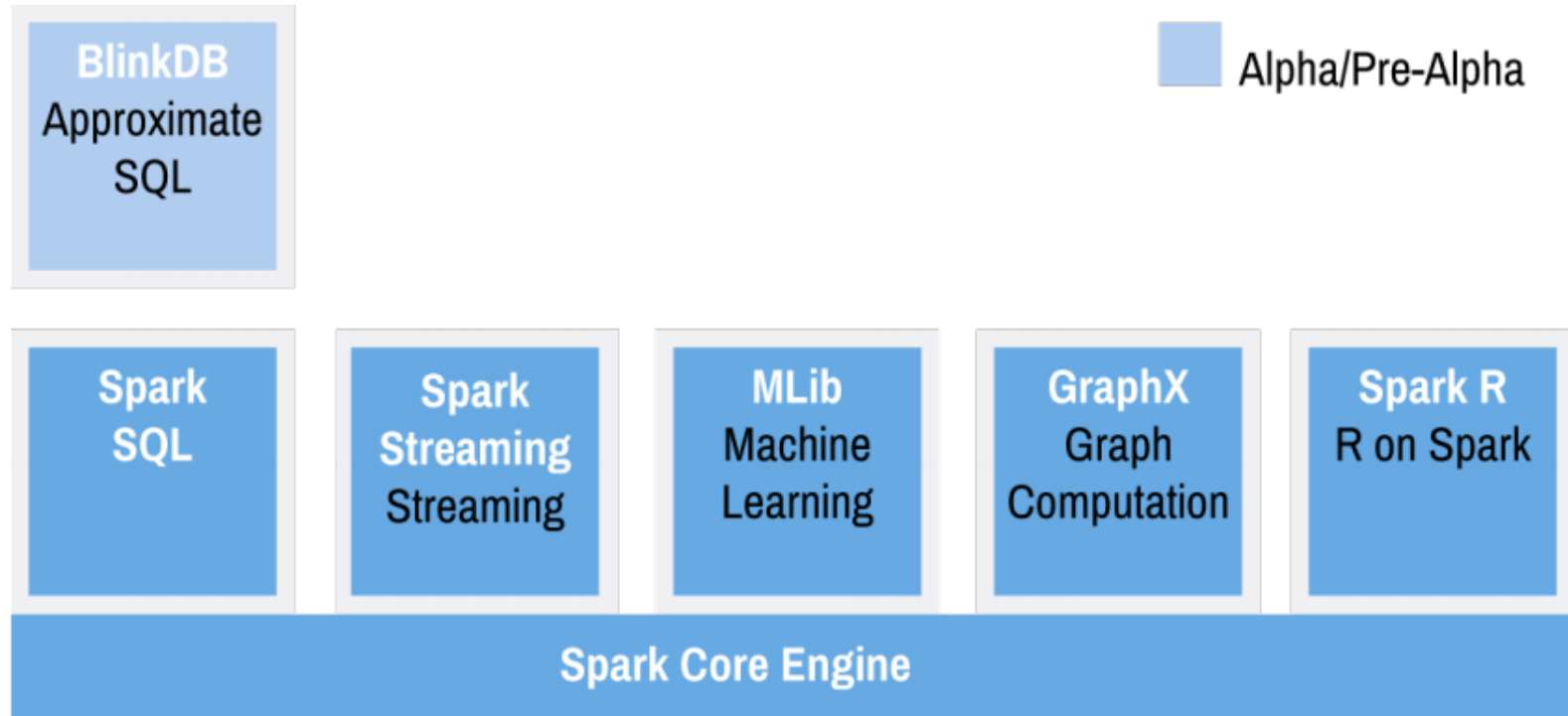
- 1) Simplicity** - its a single environment that can address a lot of tasks
- 2) Speed** - Spark is an in-memory Big Data solution and is thus blistering fast compare to standard on-disk Hadoop
- 3) Support** - Spark has APIs for Java, Python, R, Scala and is currently supported by a range of different commercial applications



# What is Spark used for?

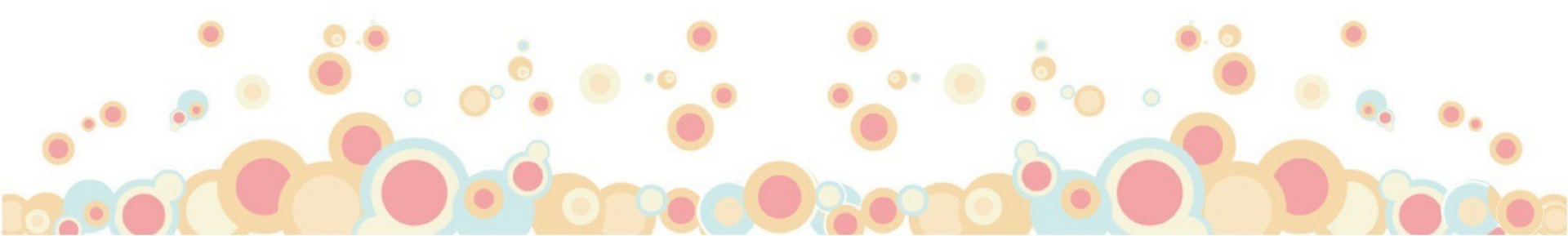
- 1) Stream Processing** - Spark, like Storm, can support streamed data
  - 2) Machine Learning** - Spark scales horizontally (is a cluster based system) and has an active library called Spark ML that provides a range of algorithms that work at scale
  - 3) Interactive Analysis** - Can be used to explore data as opposed to providing pre-built cubes and warehouses
  - 4) Data Integration** - Perfect solution for supporting large scale ETL efforts
- 
- A decorative footer at the bottom of the slide featuring a collection of overlapping circles in various colors including pink, orange, yellow, and light blue, creating a bubbly, abstract pattern.

# Spark Architecture



# Spark Architecture

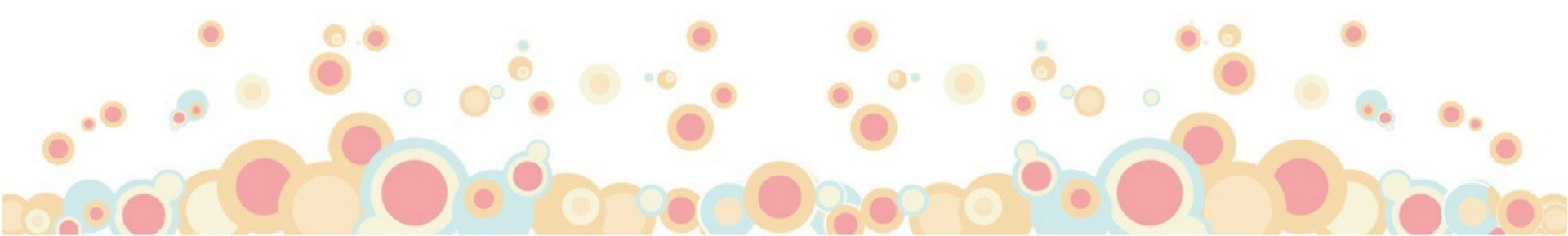
- 1) Spark Core:** Management functions of Spark, Scheduling, Resilient Distributed Datasets, etc.
- 2) Spark SQL** - Used to interact with structured data and can be used by different BI tools.
- 3) Spark Streaming** - Used to support the processing of streamed data but the code can be repurposed for batch as it is based on RDDs.





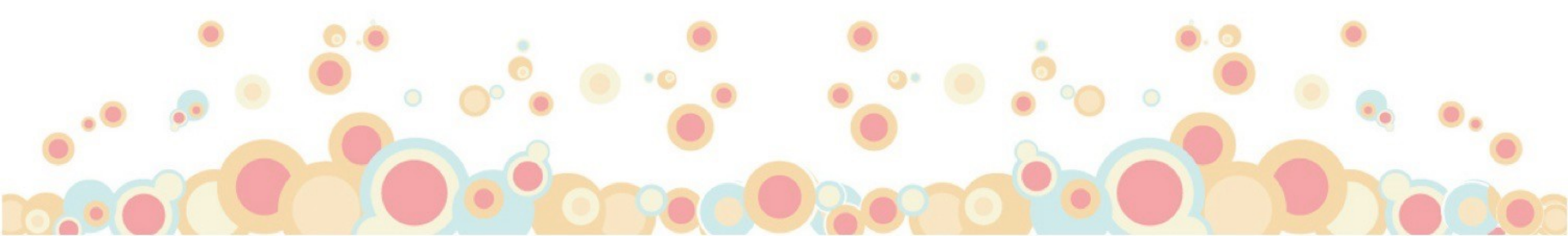
# Spark Architecture

- 4) MLlib** - Provides a set of common algorithms for addressing scalable ML problems.
- 5) GraphX** - GraphX supports analysis of and computation over graphs of data.
- 6) Spark R** - A lightweight interface for users who prefer to use R



# Spark Architecture

Resilient Distributed Dataset (RDD) sits as the principal abstraction at the core of Spark. RDDs provide in-memory data storage, distributed across a cluster in a manner that is demonstrably both fault-tolerant and efficient.



# Spark Architecture

Resilient Distributed Dataset (RDD) supports two basic types of operation:

**Transformation:** Creates a new RDD by changing the original RDD using things like map, filter, etc.

**Actions:** Such as counts that measure but do not change the original data

Operations can be chained together and are lazily evaluated - thus not compute until required

# Spark Architecture

## Transformations

Map

filter

flatMap

mapPartitions

mapPartitionsWithIndex

x

sample

## Actions

reduce

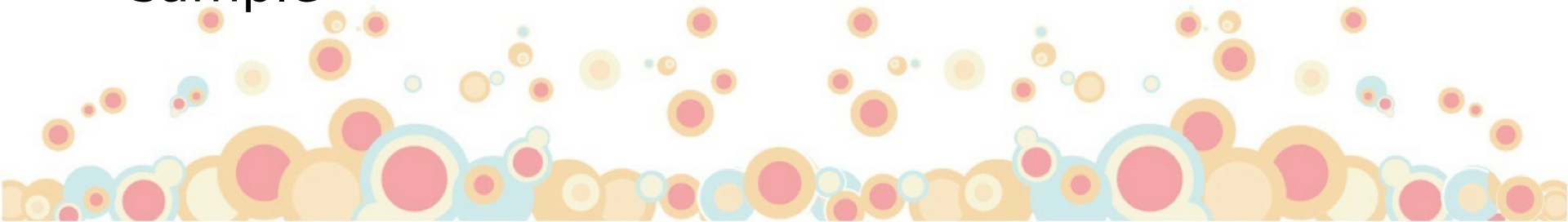
collect

count

first

take

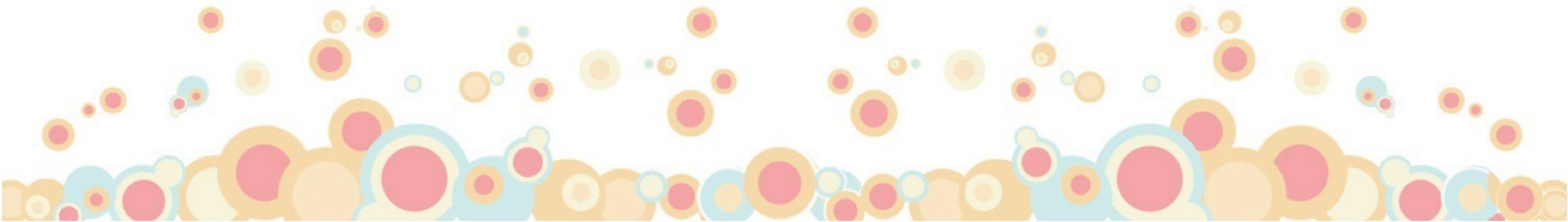
takeSample



# SUMMARY

## Preliminaries

1. Data Warehousing
2. Big Data Architectures
  - Batch Processing - Data Lakes
  - Batch and Stream Processing - Lambda
  - Stream Processing - Kappa
3. Apache Spark
  - What is Spark
  - What is Spark used for?
  - Spark Architecture
4. Summary



# Questions

?

