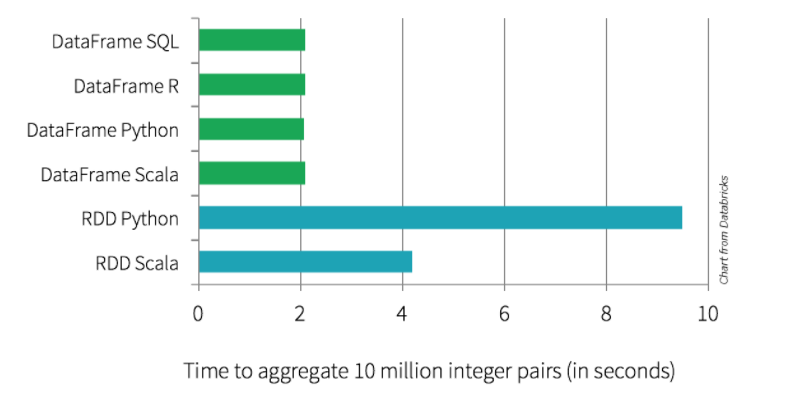
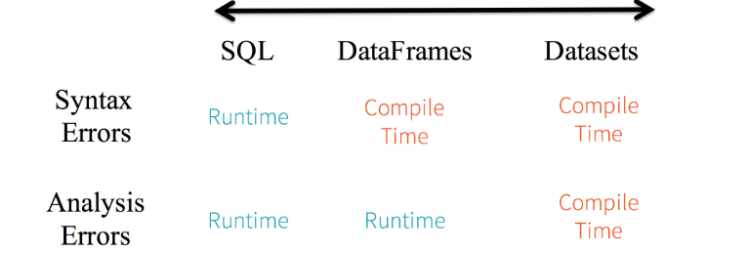
**Spark DataFrame**

SparkSQL chính là một trong năm [thành phần chính của Spark](http://itechseeker.com/tutorials/apache-spark/thanh-phan-cua-apache-spark/) được phát triển cho việc sử lý dữ liệu có cấu trúc (structured data processing). Chúng ta có thể tương tác với SparkSQL thông qua SQL, DataFrames API hoặc Datasets API. Để nắm được khái niệm DataFrames  và Datasets  trong Spark, trước hết chúng ta cần tìm hiểu về RDD (Resilient Distributed Datasets).

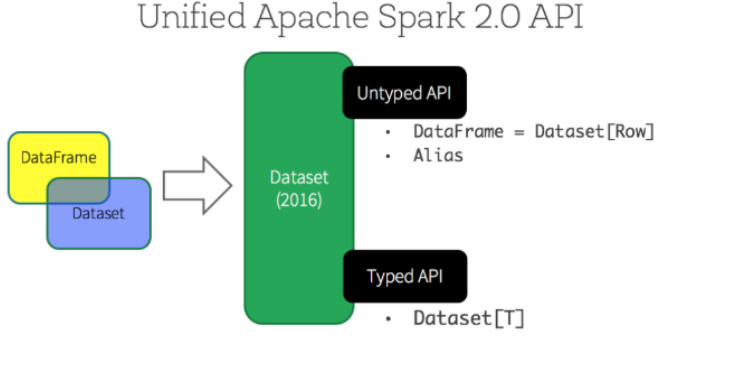
**RDD** là cấu trúc dữ liệu nền tảng của Spark, được sử dụng để phát triển Spark từ khi dự án này mới được ra đời. Resilient ở đây có thể hiểu là khả năng khôi phục dữ liệu khi dữ liệu xảy ra lỗi hoặc bị mất dữ liệu trong quá trình sử dụng. Distributed có nghĩa là các phần tử và các đối tượng (objects) trong Spark là không thể thay đổi (immutable) và được phân tán ra nhiều nodes khác nhau trong một cluster. Chính thuộc tính này của RDD cho phép Spark có thể thực hiện các thuật toán và tiến hành xử lý một cách song song, qua đó giúp tăng tốc độ và hiệu suất của hệ thống.

**DataFrame** là một API bậc cao hơn RDD được Spark giới thiệu vào năm 2013 (từ Apache Spark 1.3). Tương tự như RDD, dữ liệu trong DataFrame cũng được quản lý theo kiểu phân tán và không thể thay đổi (immutable distributed). Tuy nhiên dữ liệu này được sắp sếp theo các cột, tương tự như trong Relation Database.DataFrame được phát triển để giúp người dùng có thể dễ dàng thực hiện các thao tác xử lý dữ liệu cũng như làm tăng đáng kể hiệu quả sử lý của hệ thống.

Khi sử dụng DataFrame API, chúng ta gọi các hàm để trích xuất kết quả mong muốn và Spark sẽ tự động tiến hành các thuật toán xử lý. Tuy nhiên ở bước cuối cùng thì các thuật toán này vẫn được chạy trên RDD mặc dù người dùng chỉ tương tác với DataFrame. Bên cạnh các ưu điểm, thì nhược điểm lớn nhất của DataFrame là API này không hỗ trợ Compile-time type safety, do đó chúng ta khó có thể tiến hành thao tác trên dữ liệu. Ví dụ như khi chúng ta dùng DataFrame để truy xuất people(“age”), kết quả trả về không phải ở dạng Int mà ở dạng Column object. Vì vậy chúng ta không thể thực hiện các thao tác với kết quả này như đối với một Int object. Việc không hỗ trợ type safefy này làm người dùng không thể phát huy lợi thế của type system mà các ngôn ngữ lập trình như Scala, Java,.. hỗ trợ. Ngoài ra, nó còn làm tăng các lỗi runtime mà đáng ra đã được phát hiện tại compile time.



Để khắc phục nhược điểm của DataFrame , Spark đã phát triển và đưa vào sử dụng Datasets API từ năm 2015 (Apache Spark 1.6). Mục tiêu chính của Datasets API là nhằm kết hợp các ưu điểm của cả RDD và DataFrame. Khác với RDD (sử dụng Java serialization), Dataset sử dụng Encoder để serialize các objects (Serialization là quá trình chuyển một Object thành một dãy byte để tiến hành các thao tác xử lý hoặc để truyền Object qua mạng. De-serialization là quá trình ngược lại đẻ chuyển hóa một dãy byte thành một Object). Cơ chế làm việc của Encoder Serialization cho phép Spark có thể thực hiện nhiều thao tác (như filtering, sorting, hashing,…) mà không cần phải thực hiện De-serialization như các cơ chế Serialization thông thường. Kể từ Spark 2.0 thì DataFrame và Dataset được hợp nhất thành một và DataFrame là một type alias của DataSet[Row] (Row ở đây chính là một untyped object). Dataset với định nghĩa Dataset[T] là tập hợp của các strongly-typed objects với ‘T’ là một typed object.



PySpark giúp các nhà khoa học dữ liệu giao tiếp với RDD trong Apache Spark và Python thông qua thư viện **Py4j.**Có nhiều tính năng giúp PySpark trở thành một khung tốt hơn các tính năng khác:

* **Tốc độ:** Nó nhanh hơn 100 lần so với các khung xử lý dữ liệu quy mô lớn truyền thống.
* **Bộ nhớ đệm mạnh mẽ:** Lớp lập trình đơn giản cung cấp khả năng lưu trữ bộ nhớ cache và ổ đĩa mạnh mẽ.
* **Triển khai:** Có thể được triển khai thông qua Mesos, Hadoop thông qua Sợi hoặc trình quản lý cụm riêng của Spark.
* **Thời gian thực:** Tính toán thời gian thực và độ trễ thấp vì tính toán trong bộ nhớ.
* **Polyglot:**  Hỗ trợ lập trình trong Scala, Java, Python và R.