개 발 환 경 설 정

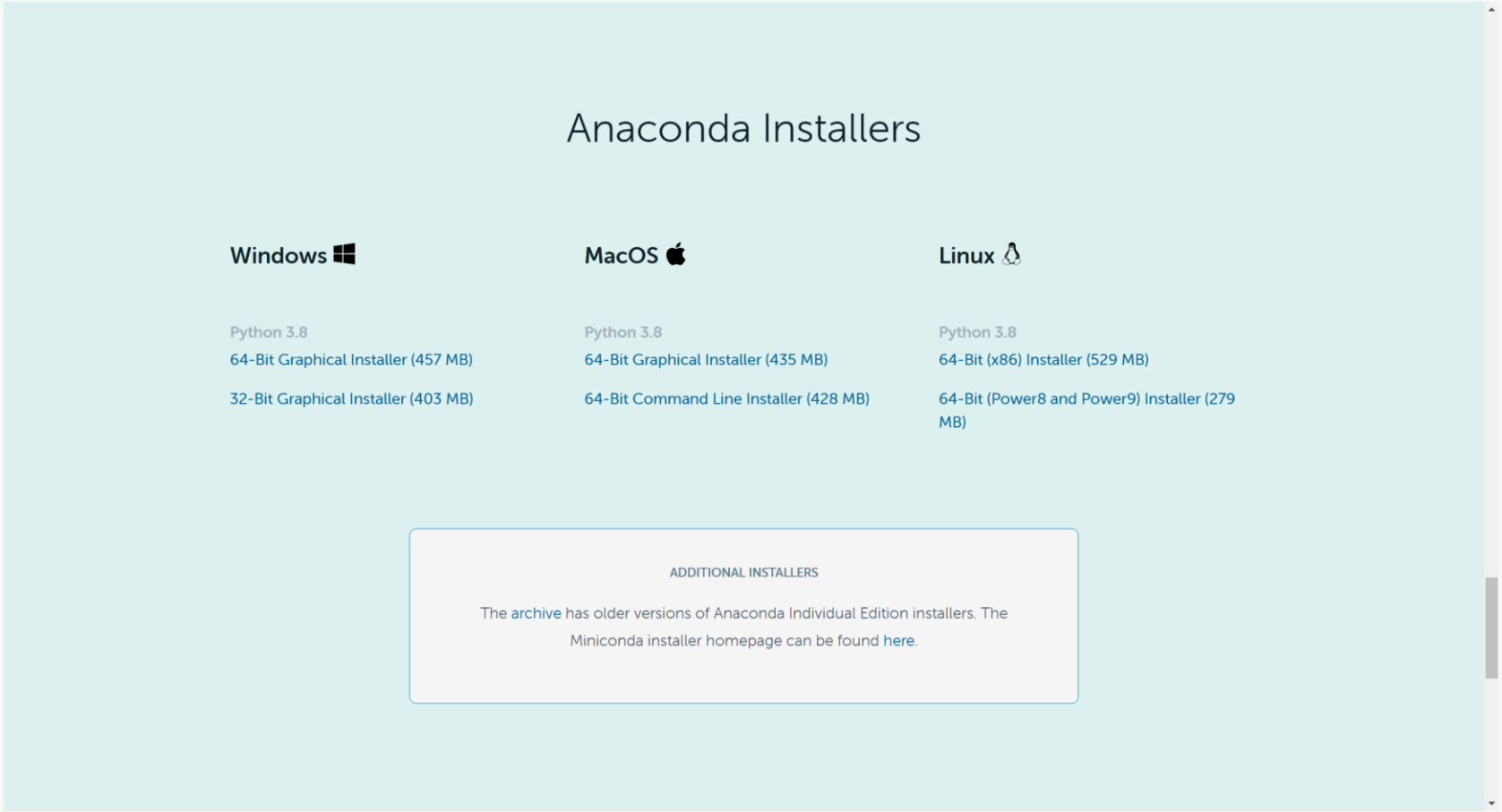
with  python™

00. 아나콘다 – Windows 10

(2020년 12월 기준)

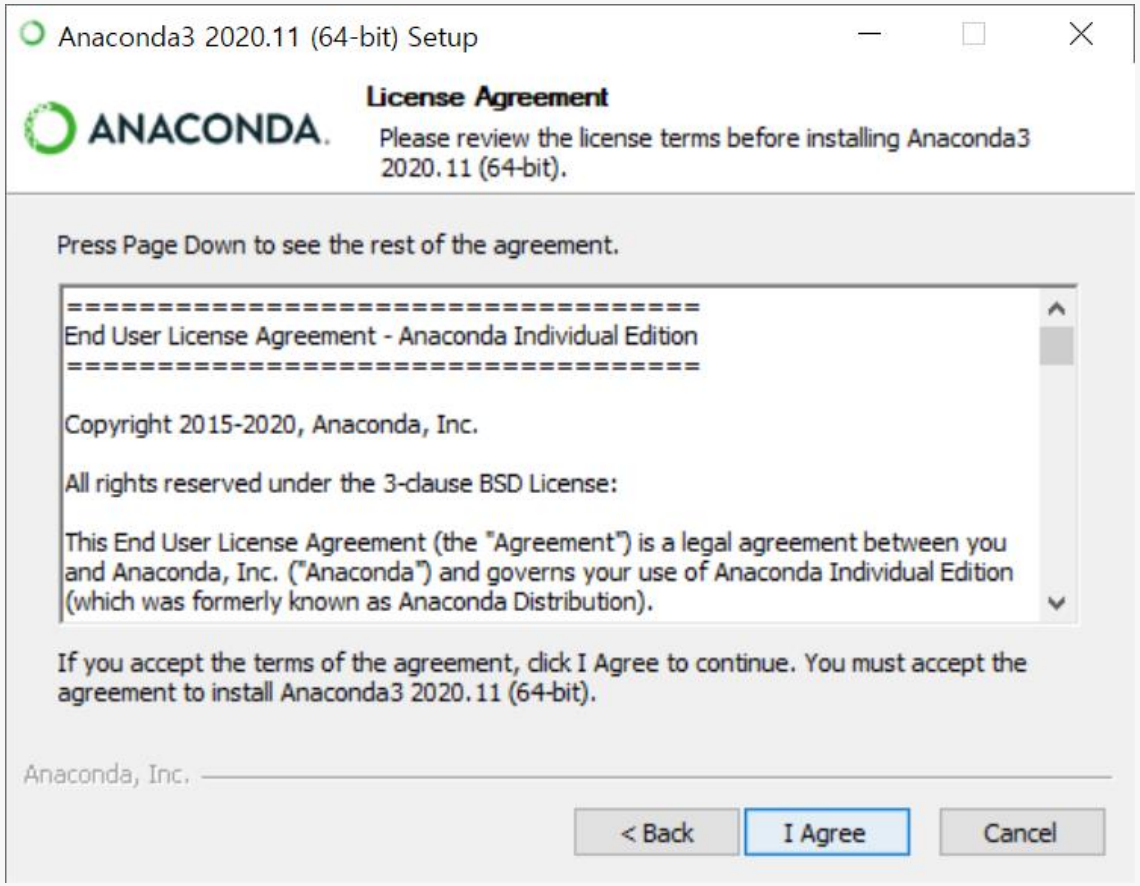
OS별 머신러닝 & 딥러닝을 구현할 수 있도록 도와주는 패키지

(웹사이트 주소: https://www.anaconda.com/products/individual)



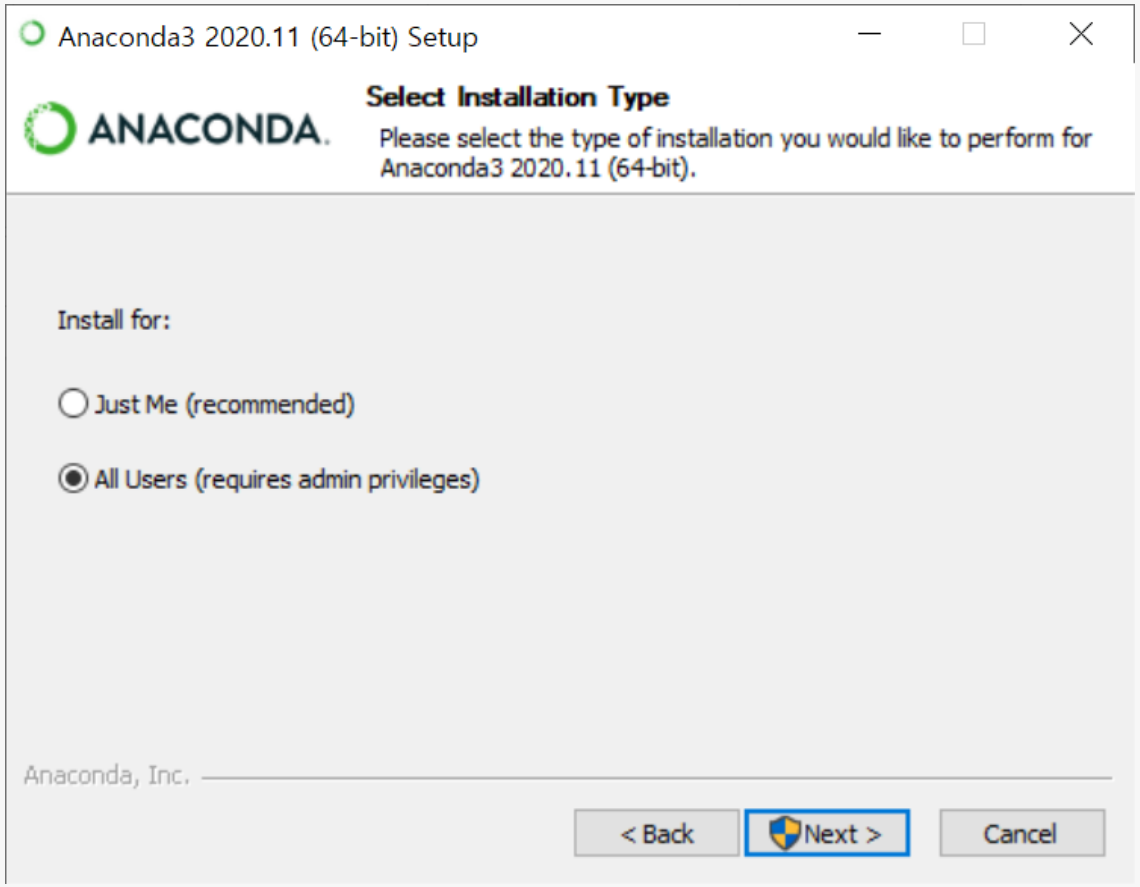
00. 아나콘다 – Windows 10

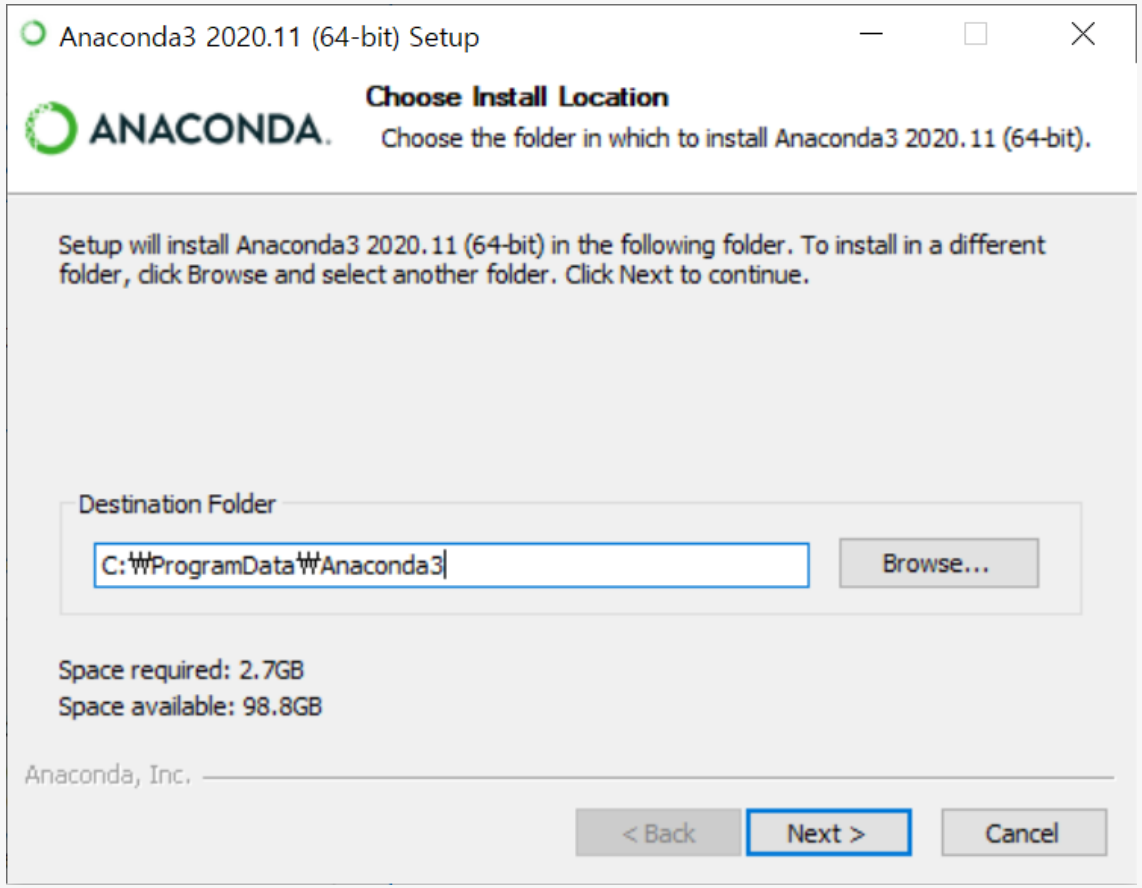
(2020년 12월 기준)



00. 아나콘다 – Windows 10

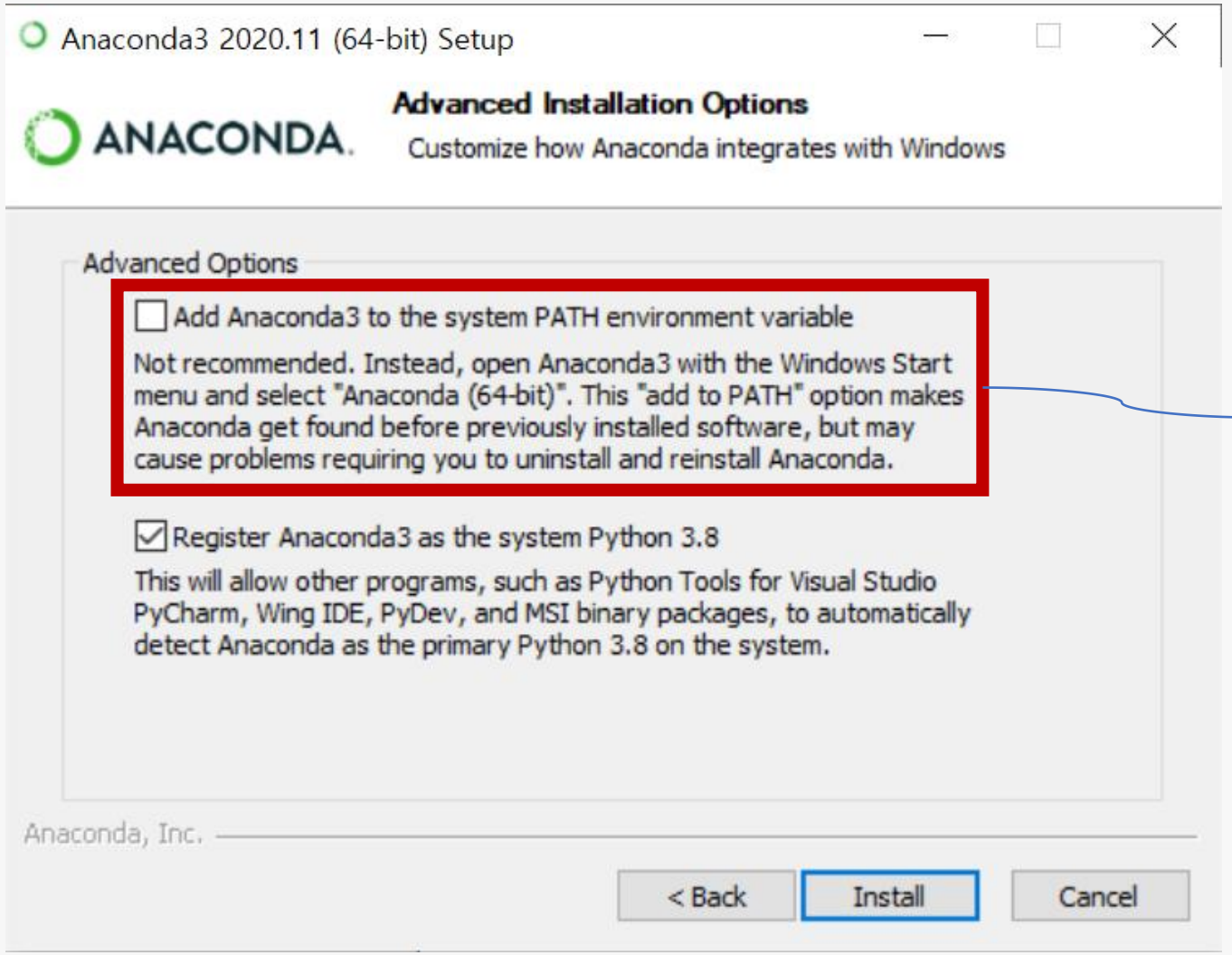
(2020년 12월 기준)





00. 아나콘다 – Windows 10

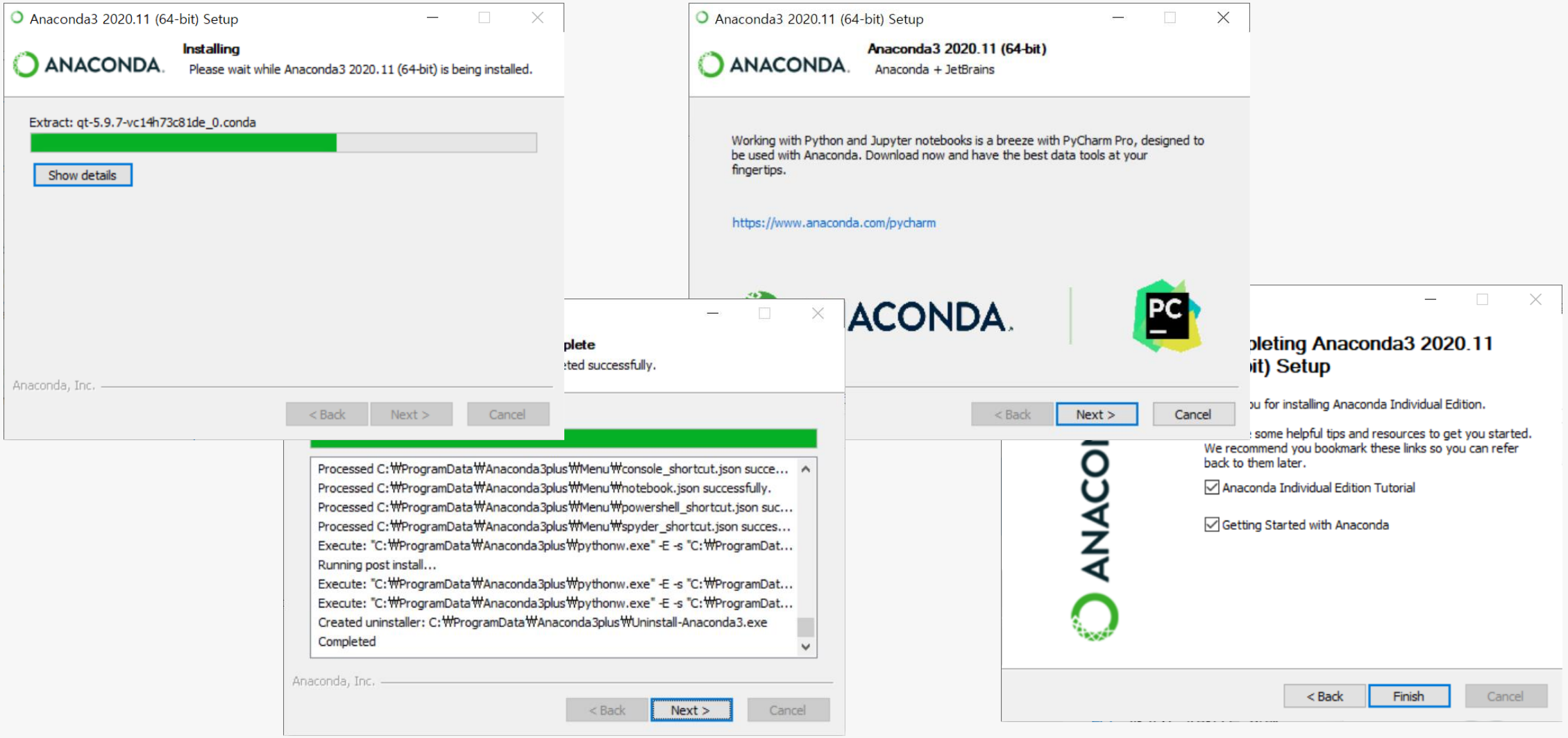
(2020년 12월 기준)



- 환경 변수의 개념을 잘 모른다면 추가하지 않는다.
- 환경 변수는 언제든지 추가가 가능하다.

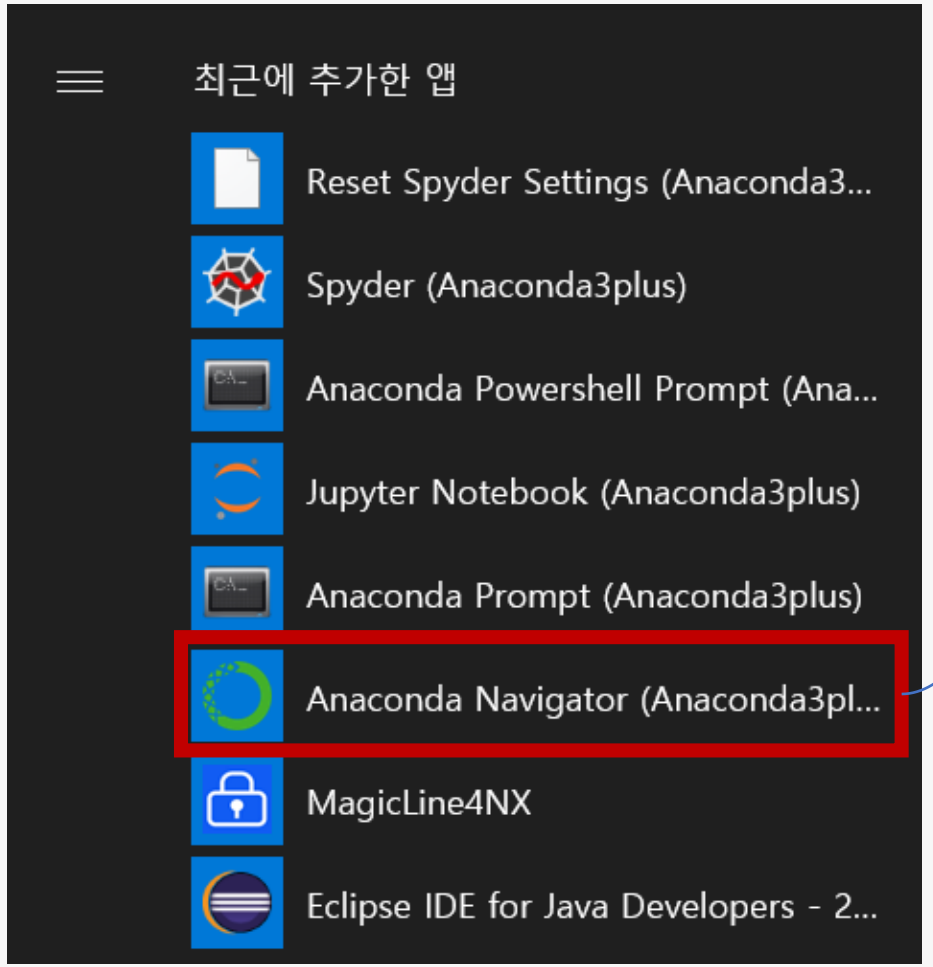
00. 아나콘다 – Windows 10

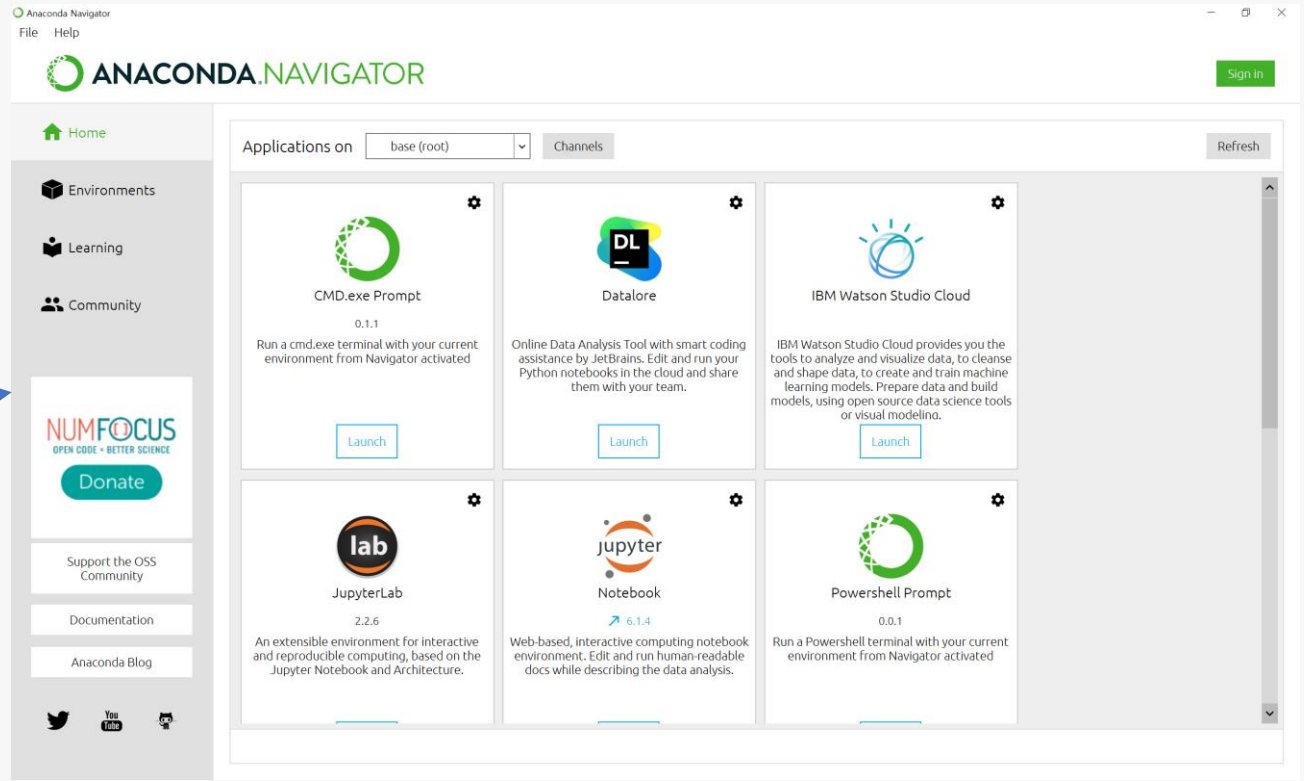
(2020년 12월 기준)



00. 아나콘다 – Windows 10

(2020년 12월 기준)

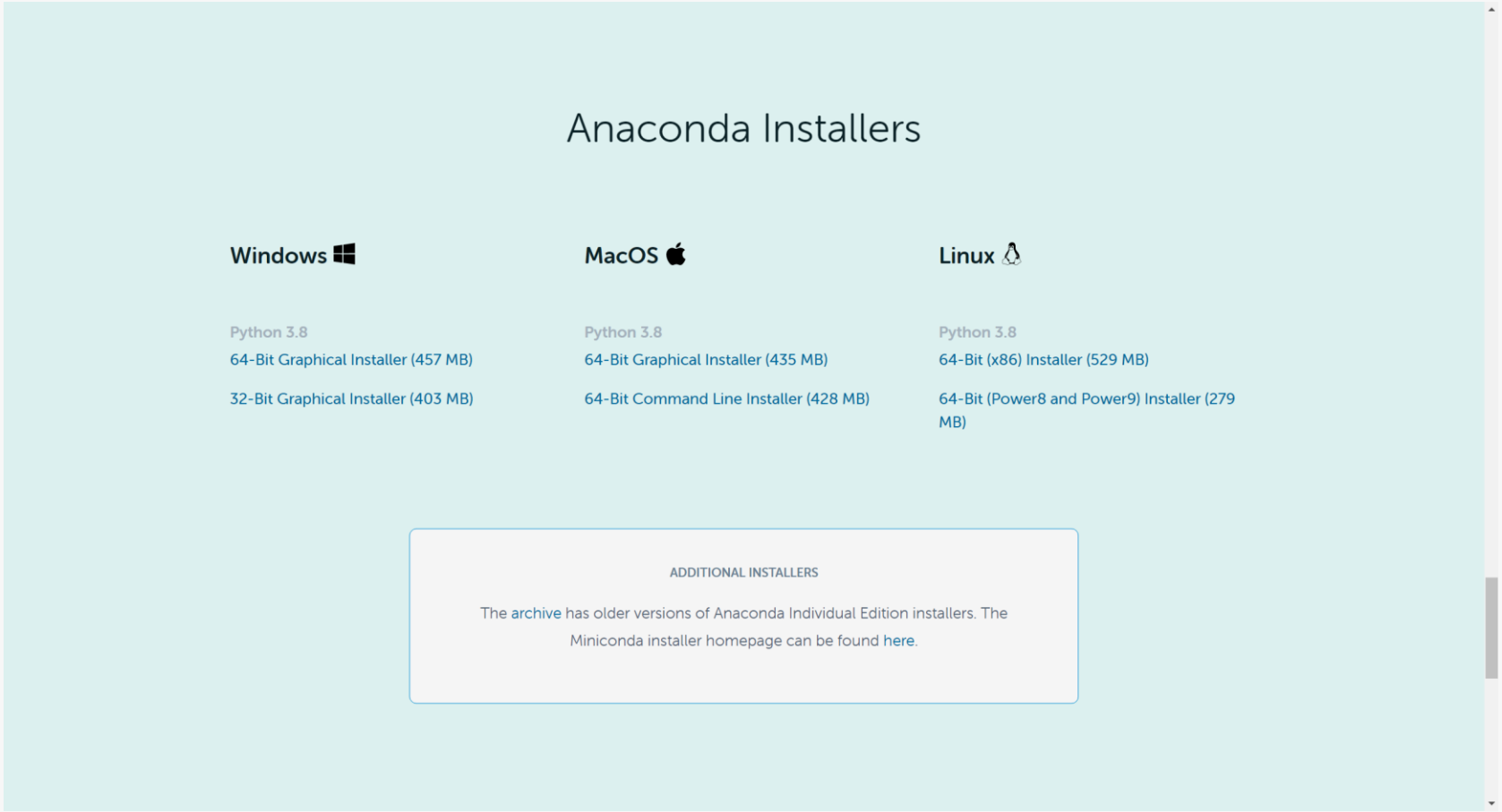




01. 아나콘다 – MacOS

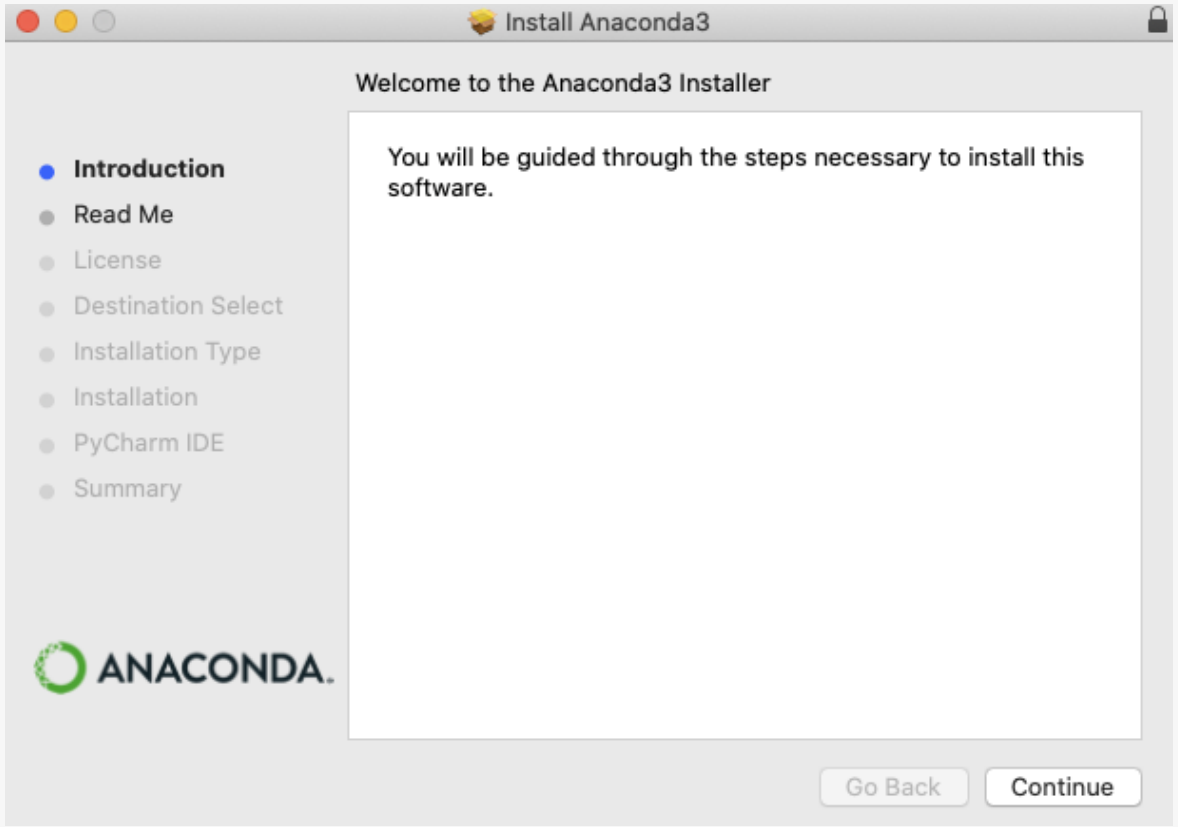
(2020년 12월 기준)

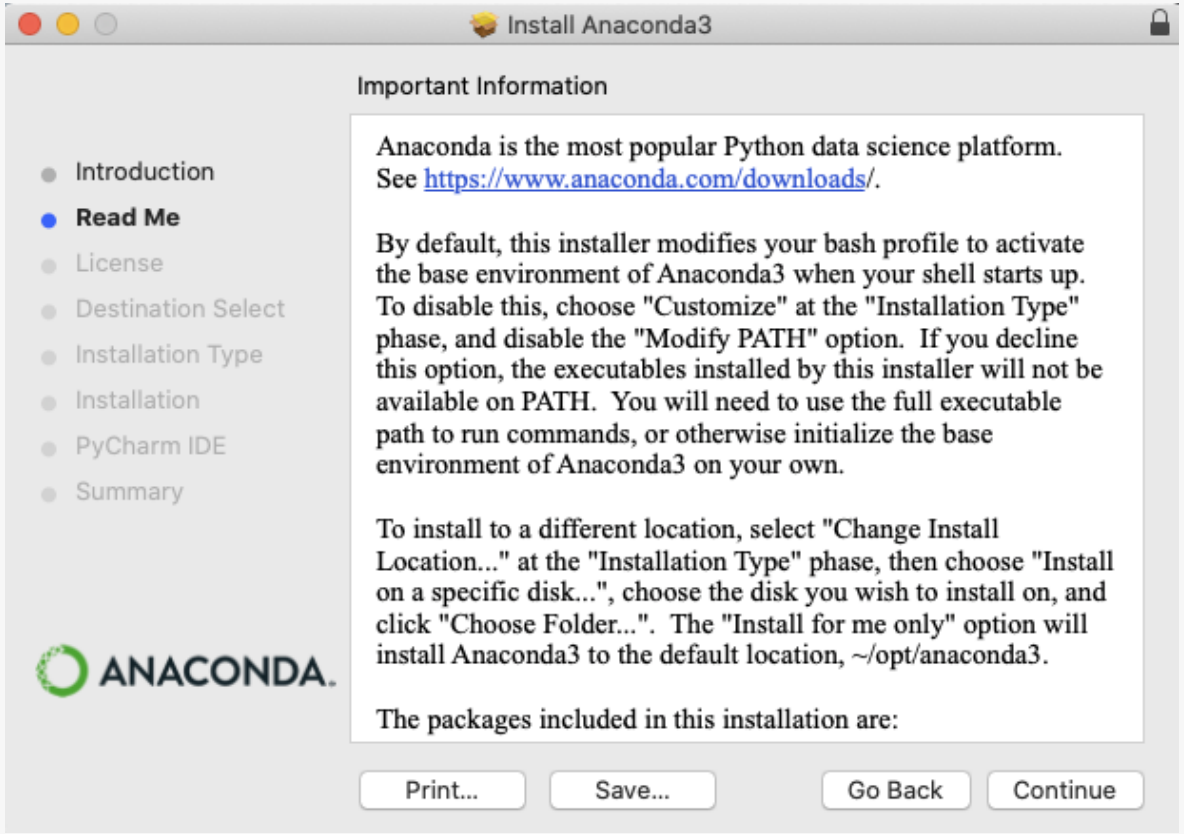
Shell Script 실행 방법 모른다면, 64-Bit Graphical Installer 선택
Shell Script 실행 방법 알고 있다면, 64-Bit Command Line Installer 선택



01. 아나콘다 – MacOS

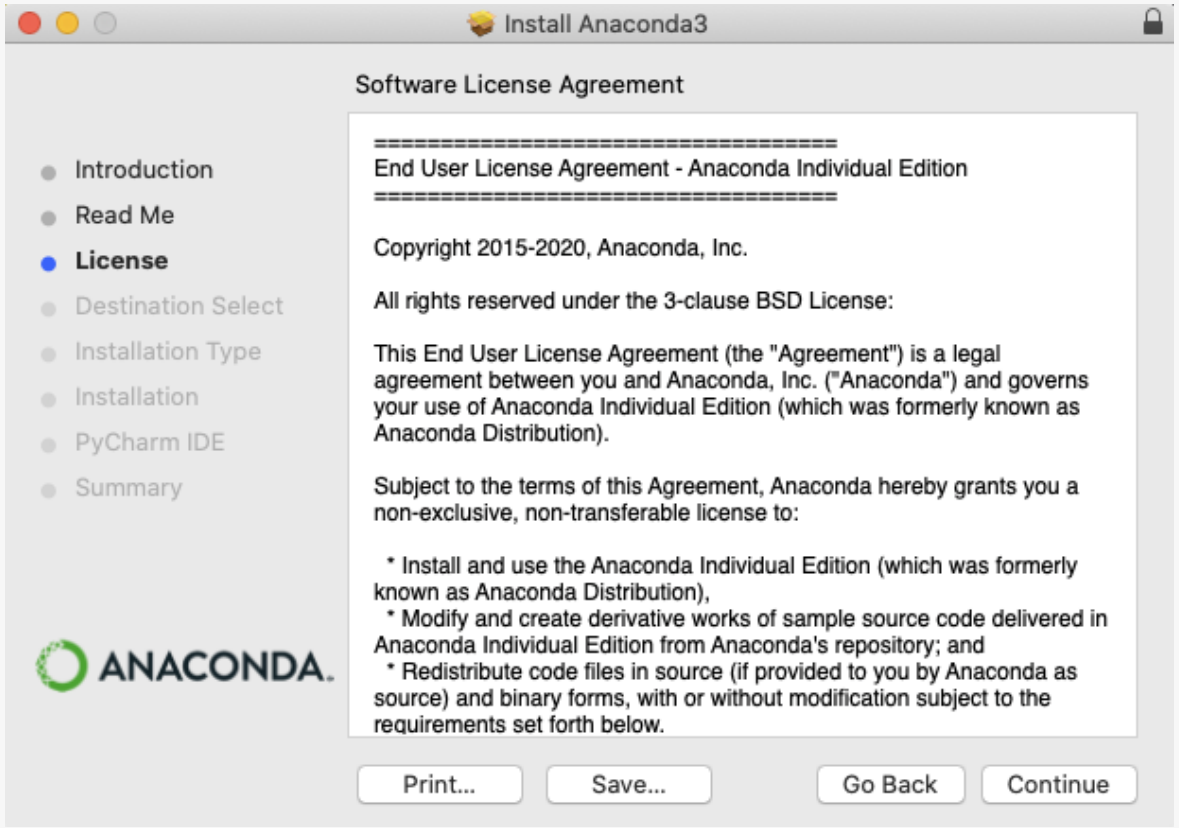
(2020년 12월 기준)

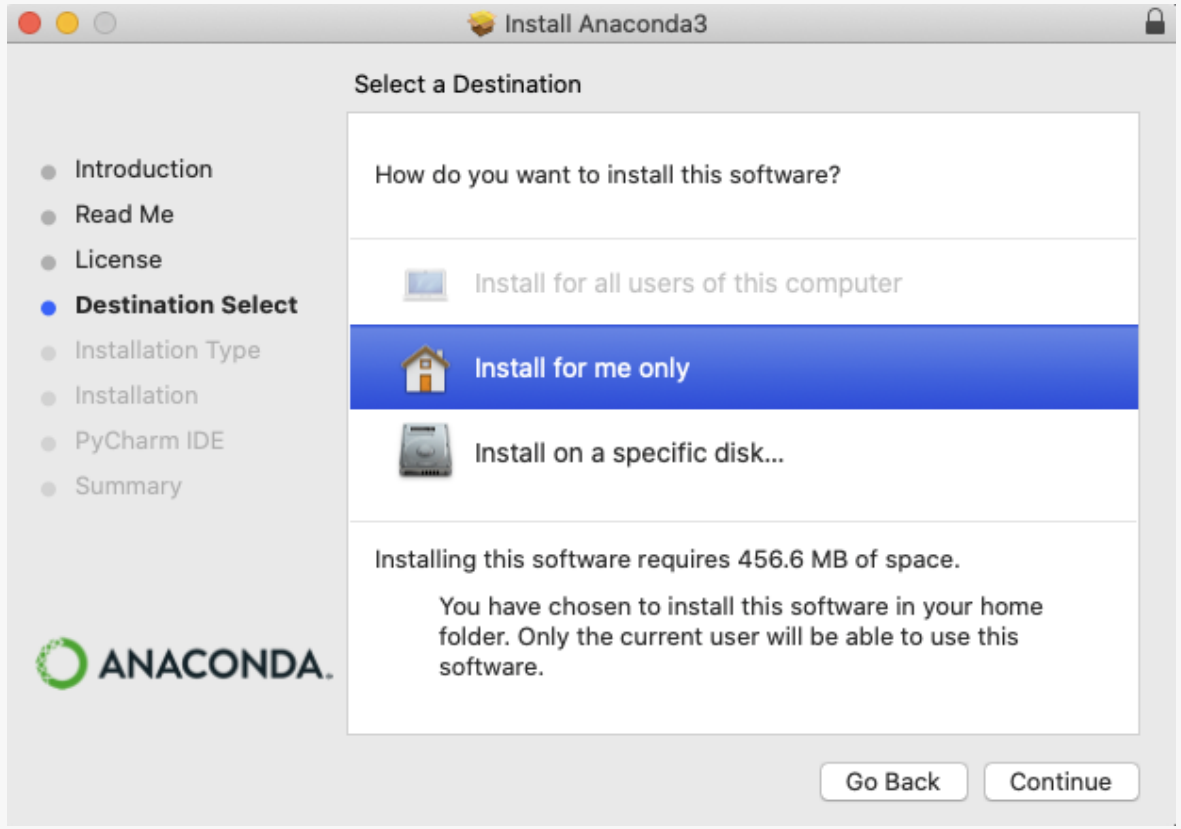




01. 아나콘다 – MacOS

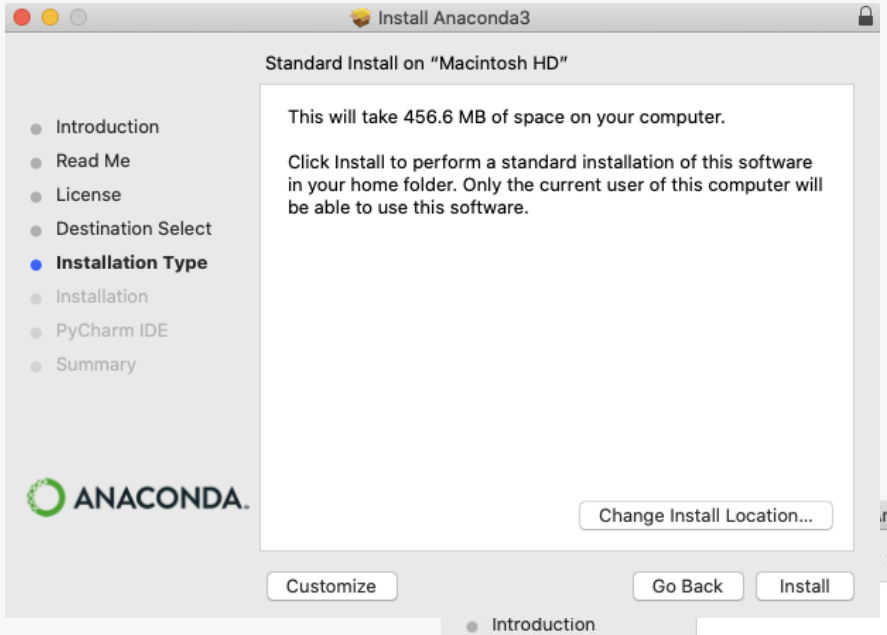
(2020년 12월 기준)

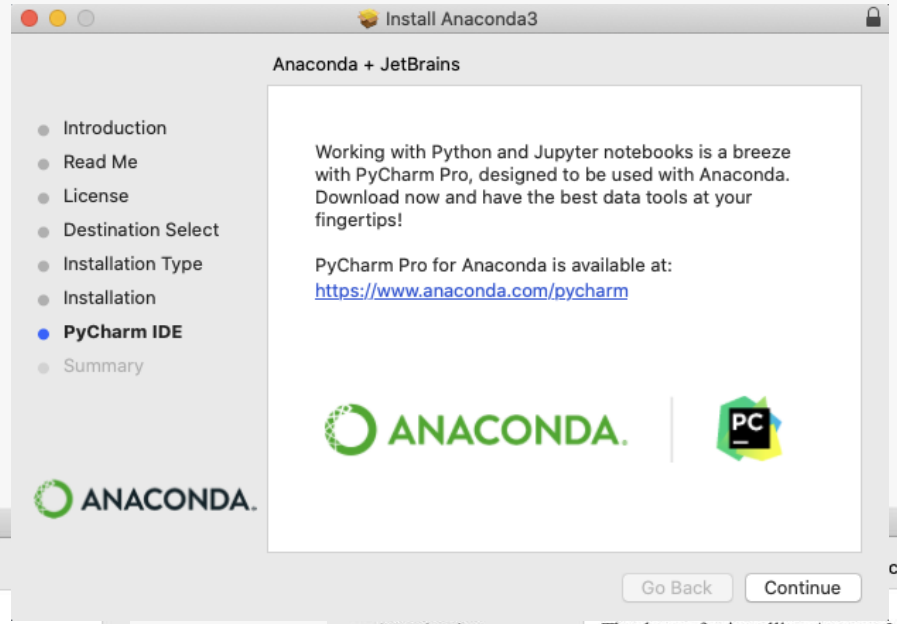


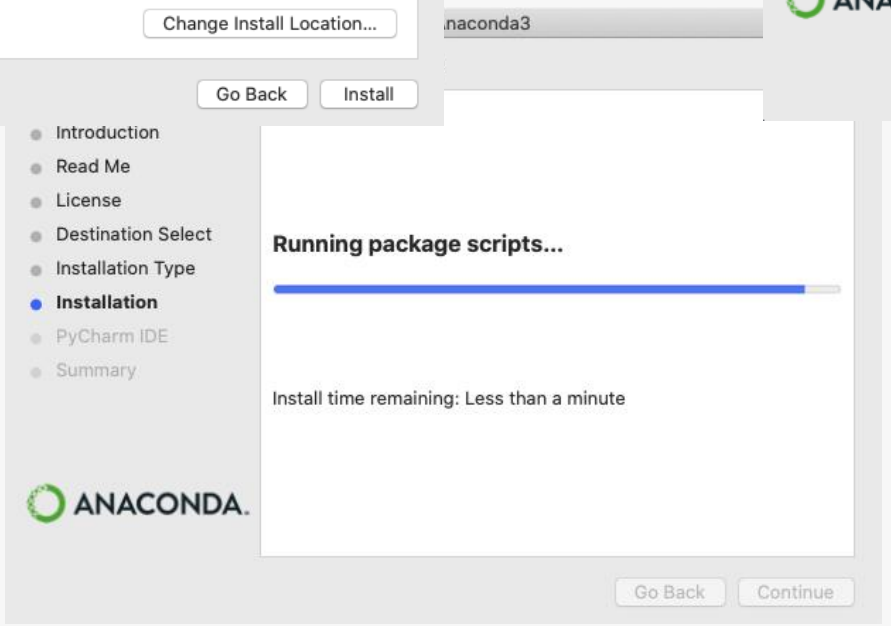


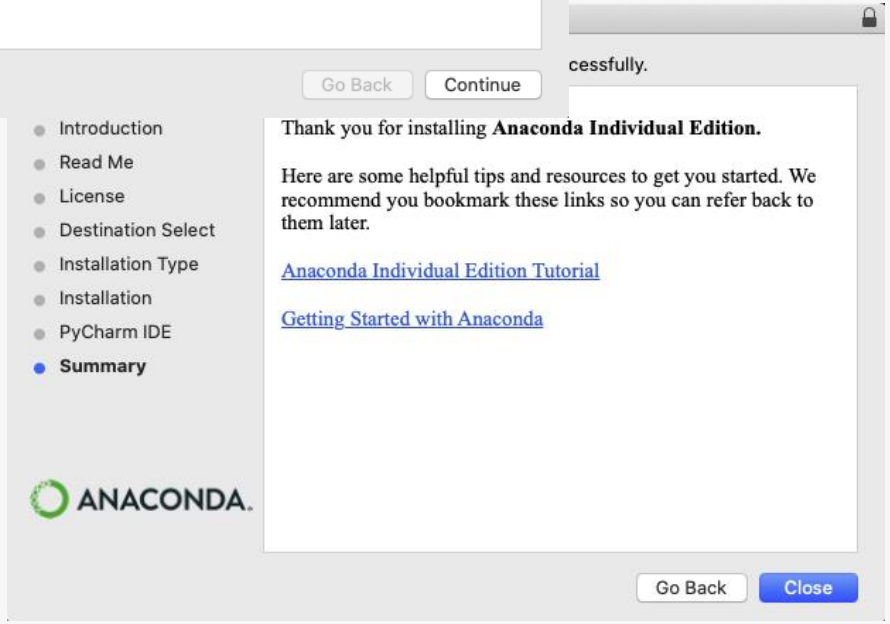
01. 아나콘다 – MacOS

(2020년 12월 기준)



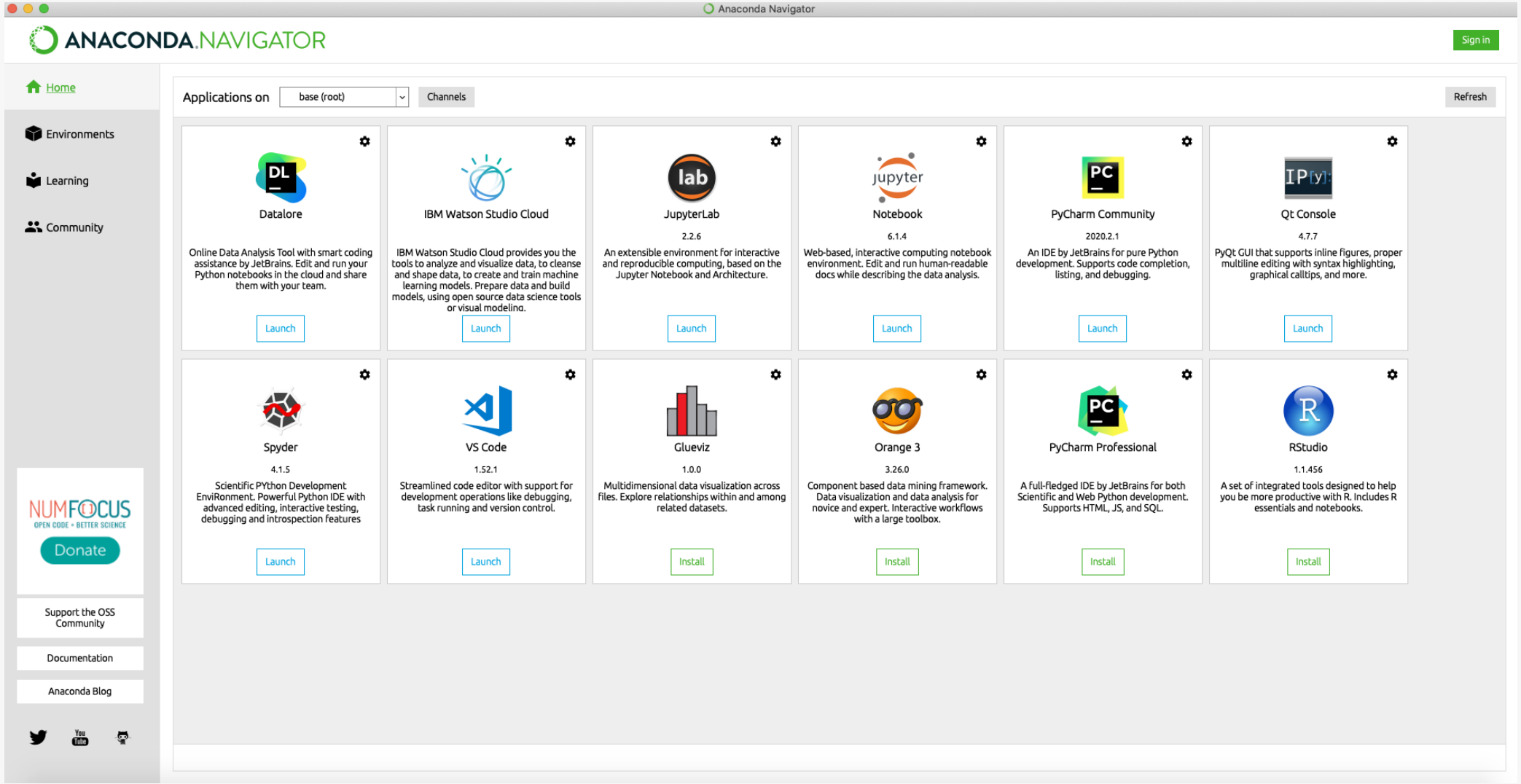






01. 아나콘다 – MacOS

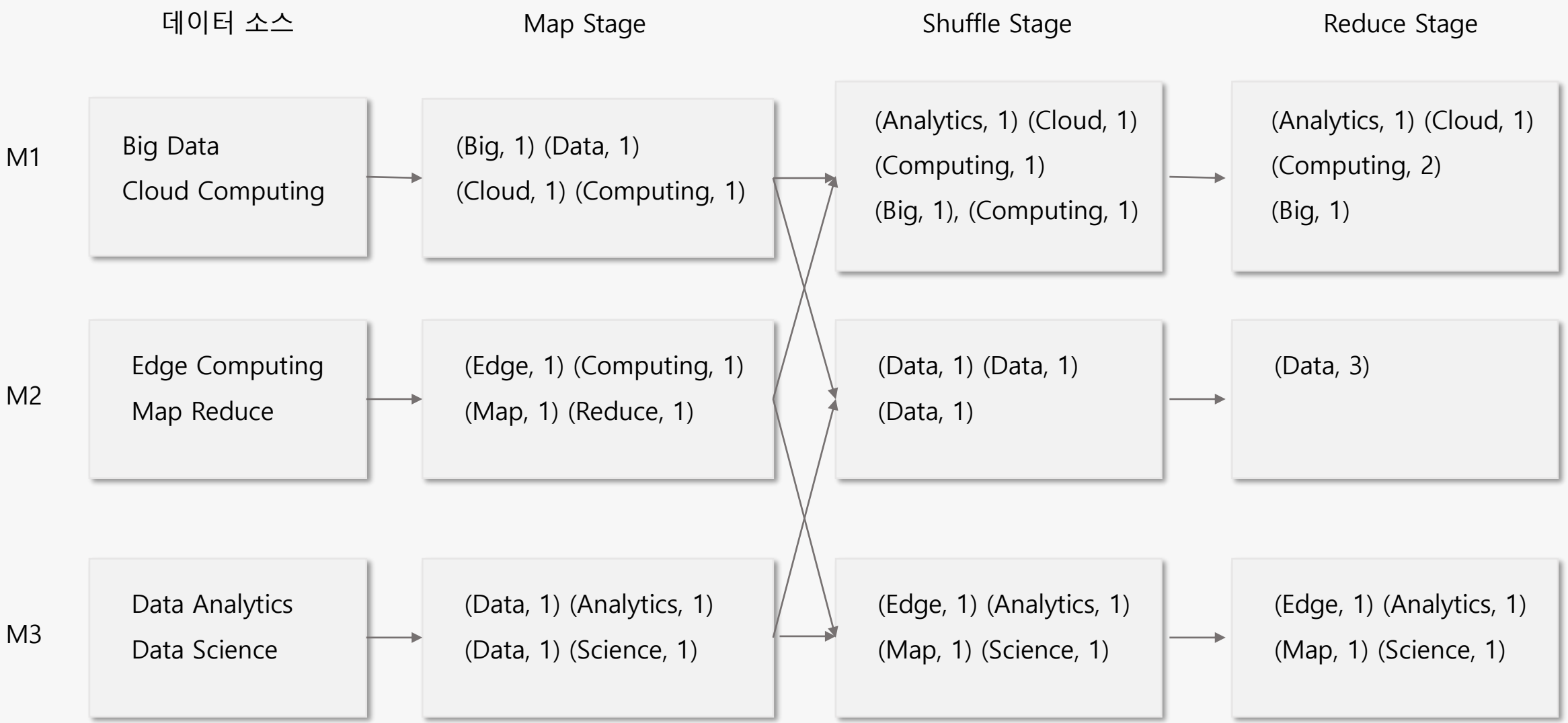
(2020년 12월 기준)



개 발 환 경 설 정

Map Reduce Paradigm

01. MapReduce Paradigm



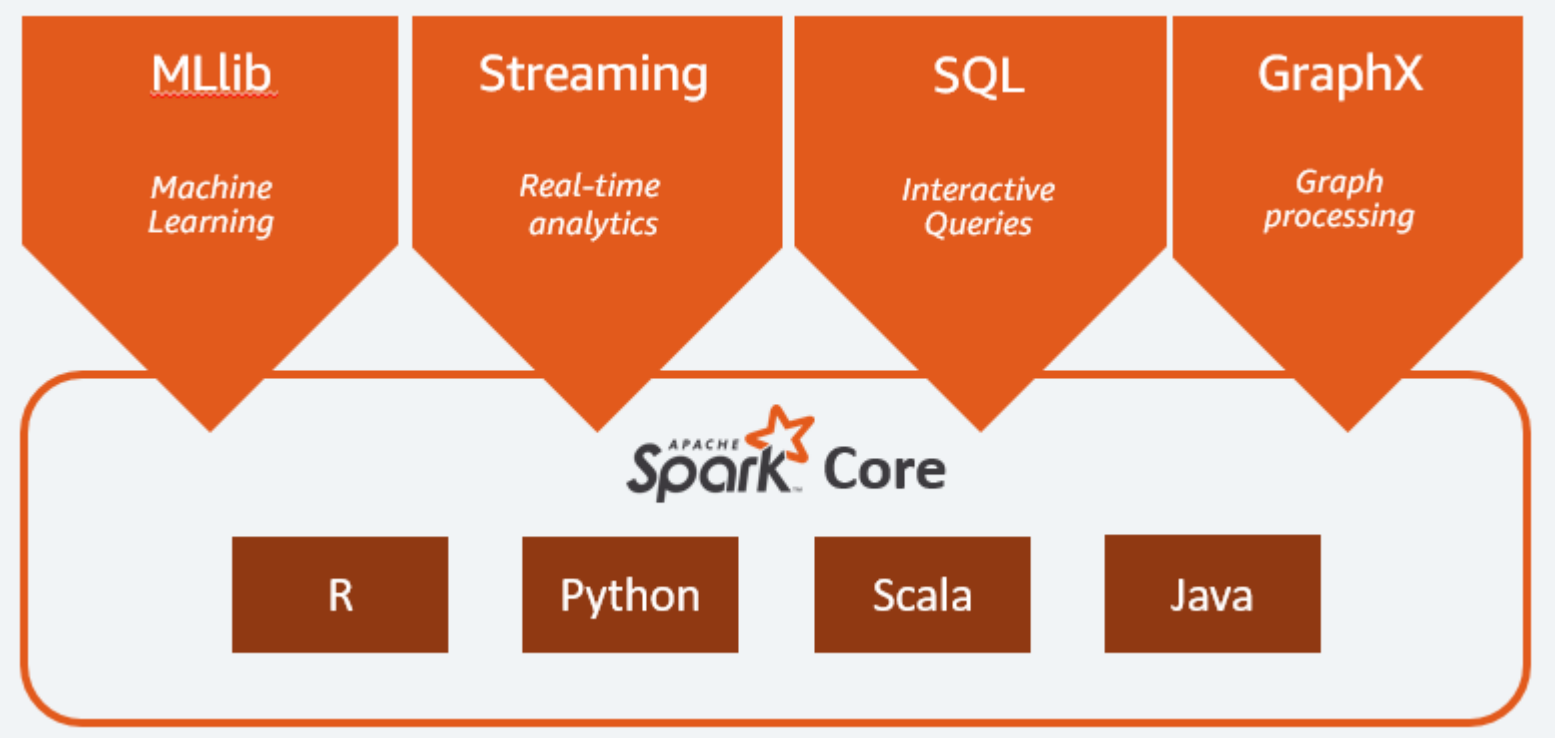
02. MapReduce Difficulty

- Programming 관점
  - ✓ 데이터 변환 위한 Low-Level API
  - ✓ Main Language: 자바
- Operation 관점
  - ✓ Designed to run on Commodity Hardware,
  - ✓ 데이터 회복력(Resiliency) → 각 Stage마다 데이터 저장 필요성 대두
  - ✓ 그런데, 각 단계마다 데이터 저장을 한다면, 파일 입출력 시, 상대적으로 데이터 처리 속도 감소

PySpark Basic

01. Spark 개요

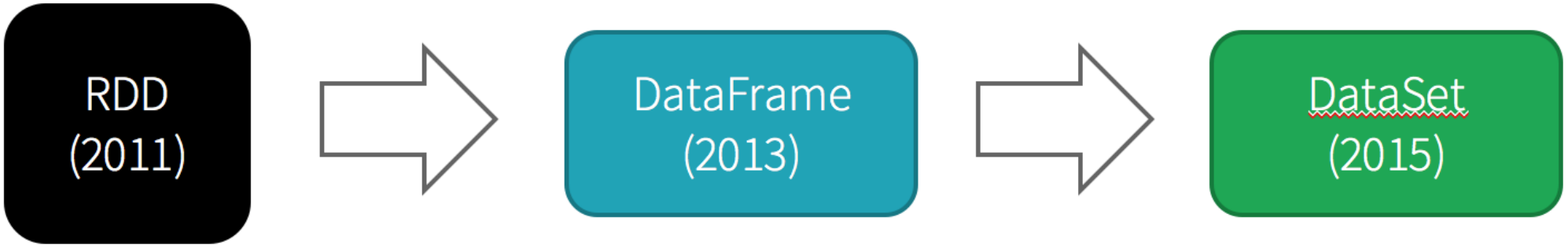
- Apache Spark는 2011년 버클리 대학의 AMPLab에서 개발됨
- 인메모리 기반의 대용량 데이터 고속 처리 엔진이며, 범용 분산 클러스터 컴퓨팅 프레임워크



01. Spark 개요

- Workloads
  - ✓ Batch Data Processing
  - ✓ Real-Time Data Processing
  - ✓ Machine Learning and Data Science
- Two Aspects
  - ✓ Processing Speed: Lazy Evaluation, Predicate Pushdown, Partition Pruning
  - ✓ Development Speed: High-Level API, Filtering, Grouping, Sorting, Joining, Aggregating

02. History of Spark APIs



Distribute collection of JVM objects

Functional Operators (map, filter, etc.)

Distribute collection of Row objects

Expression-based operations and UDFs

Logical plans and optimizer

Fast/efficient internal representations

Internally rows, externally JVM objects

Almost the "Best of both worlds": **type safe + fast**

But slower than DF
Not as good for interactive analysis, especially Python

databricks

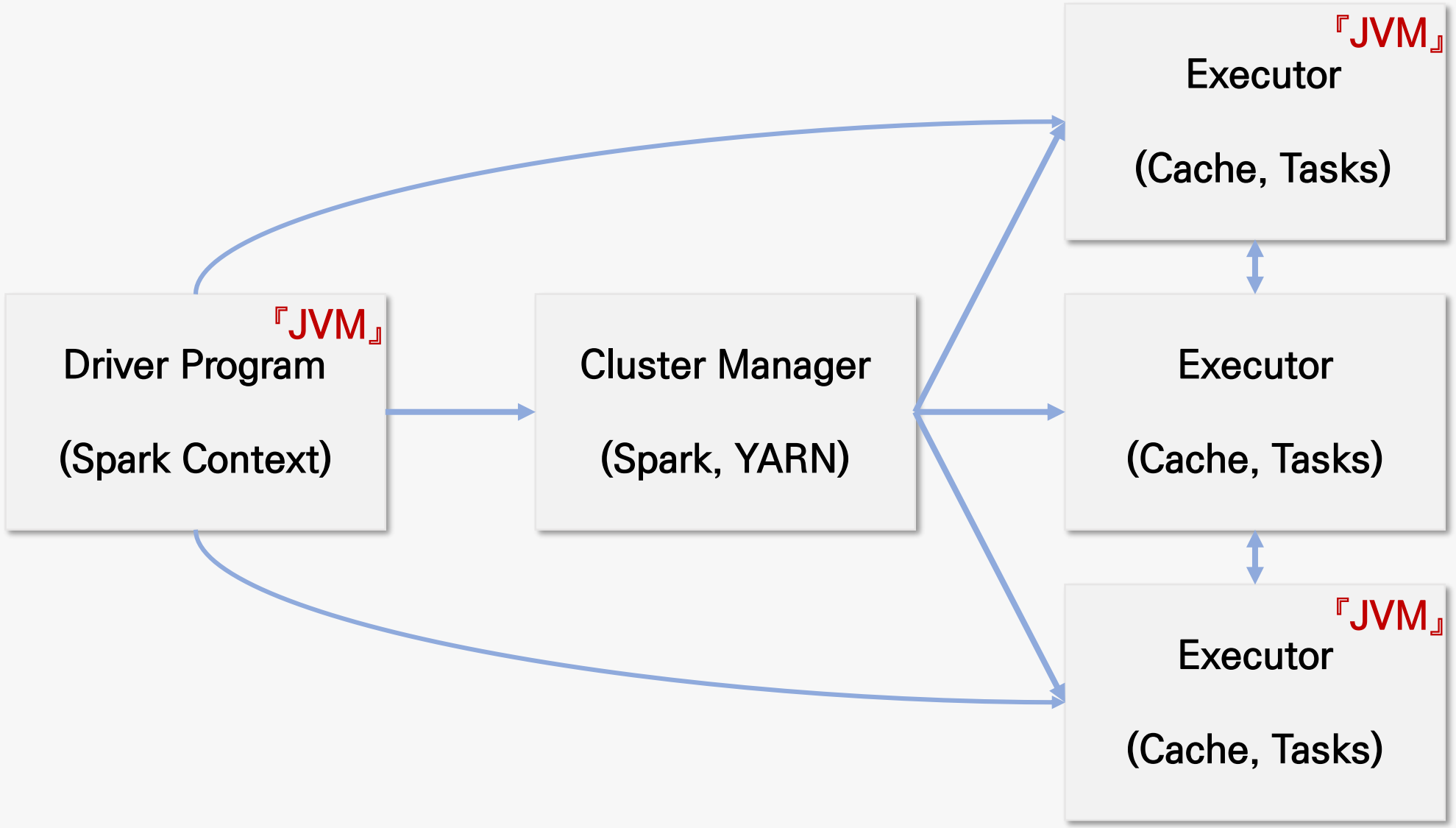
02. History of Spark APIs

RDD(Resilient Distributed Dataset)

Data Frame (DF)

Col 1	Col 2	...	Col N
1	1	1	1
2	2	2	2
3	3	3	3
4	4	4	4
5	5	5	5
.	.	.	.
.	.	.	.
N	N	N	N

02. Spark: Scalable



03. Spark: FAST & HOT

- FAST
  - ✓ Hadoop의 MapReduce 기반의 인메모리보다 약 100배 이상 처리 속도 빠름
  - ✓ DAG(Directed Acyclic Graph) Engine 기반으로 워크플로(WorkFlow) 최적화
- HOT
  - ✓ Amazon, Ebay, NASA, Yahoo 등에서 사용함
  - ✓ 국내 굴지의 대부분 기업에서 사용함

04. Spark: Development

- 개발 방법
  - ✓ 프로그래밍 언어: Python, JAVA, Scala
  - ✓ 기본 데이터 타입: RDD(Resilient Distributed Dataset)

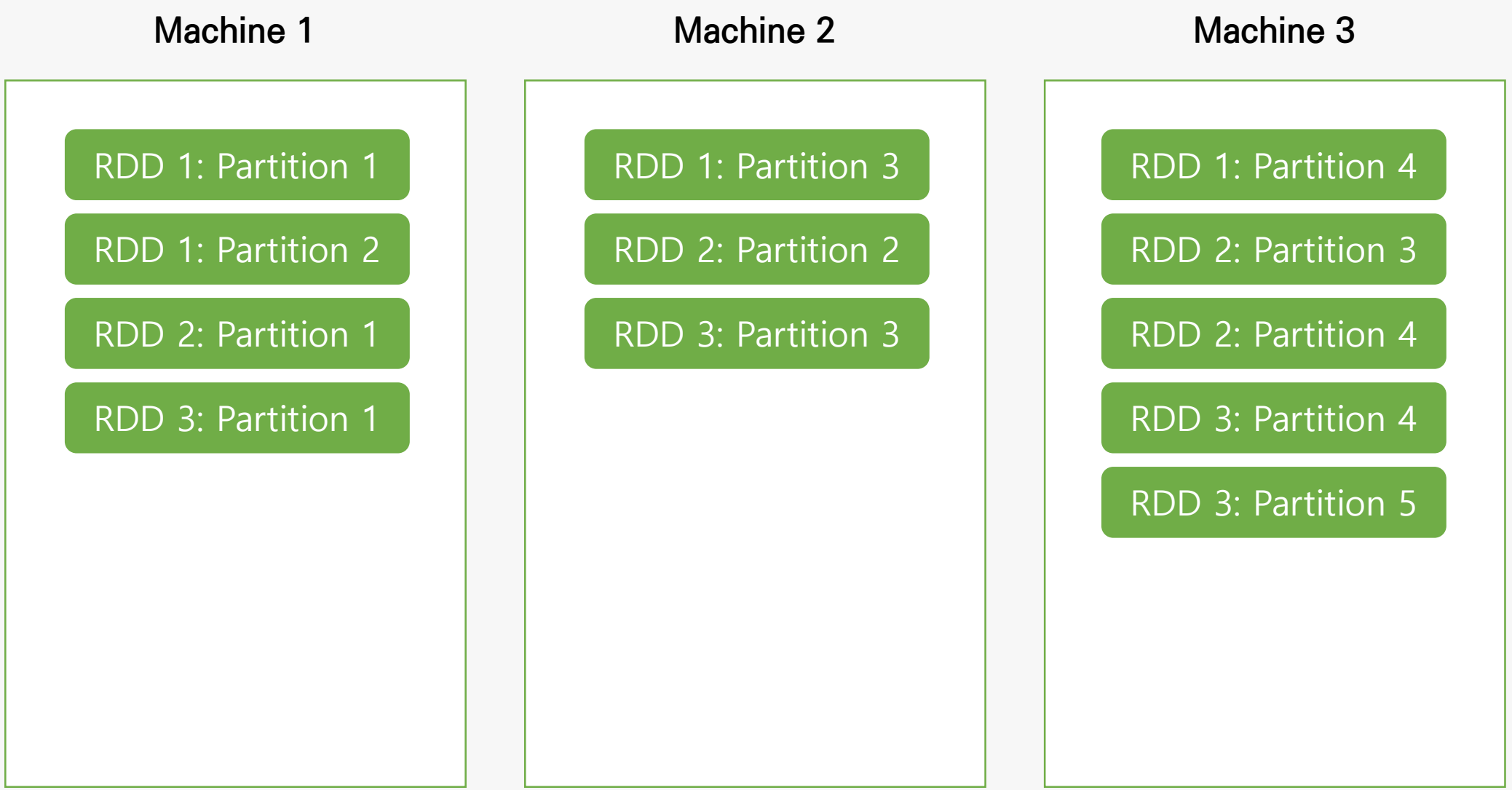
05. Spark: Python 언어 사용의 주요 이유

- Why Python
  - ✓ 컴파일(Compile), 라이브러리 의존성 부담 감소
  - ✓ 자바 및 Scala에 비해 상대적으로 쉬운 코딩
- If you want to be more,
  - ✓ Scala 언어를 배워야 한다.
  - ✓ Built in Scala

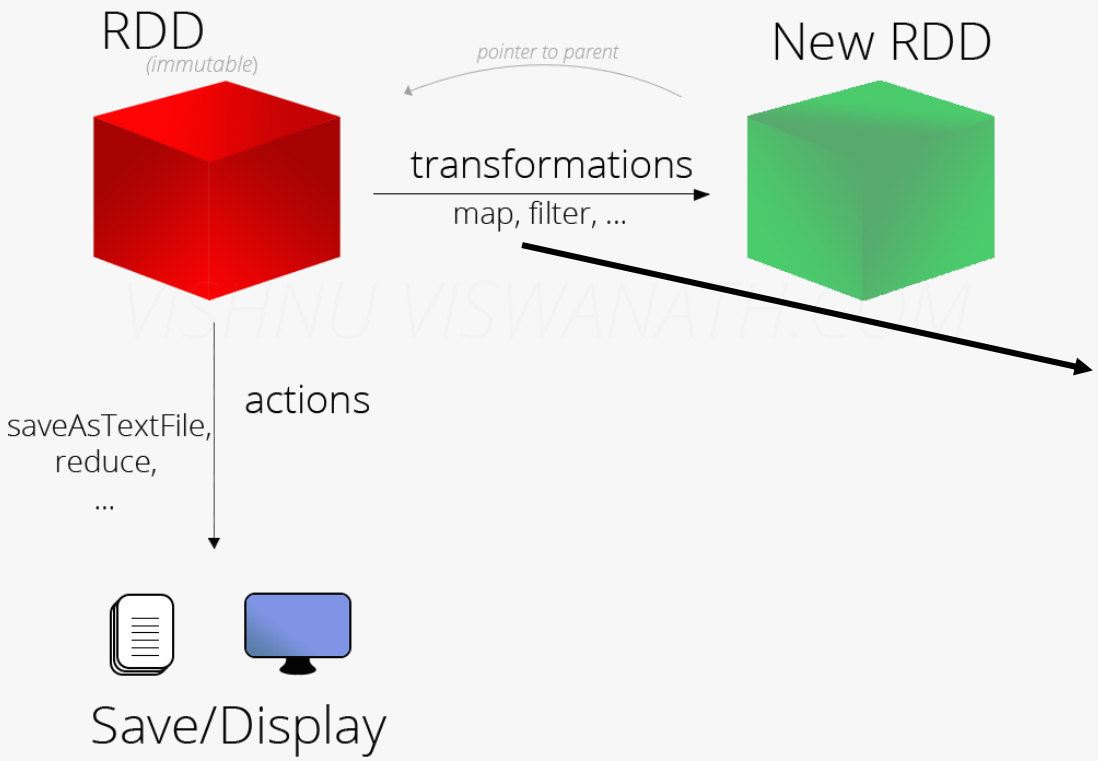
06. RDD

- *Resilient: 회복력 있는, 메모리 내부에서 데이터 유실 시, 복구 가능
- Distributed
- Dataset
- 주요 특징
  - ✓ Immutable, Fault-Tolerant (결함 허용), Parallel

06. RDD

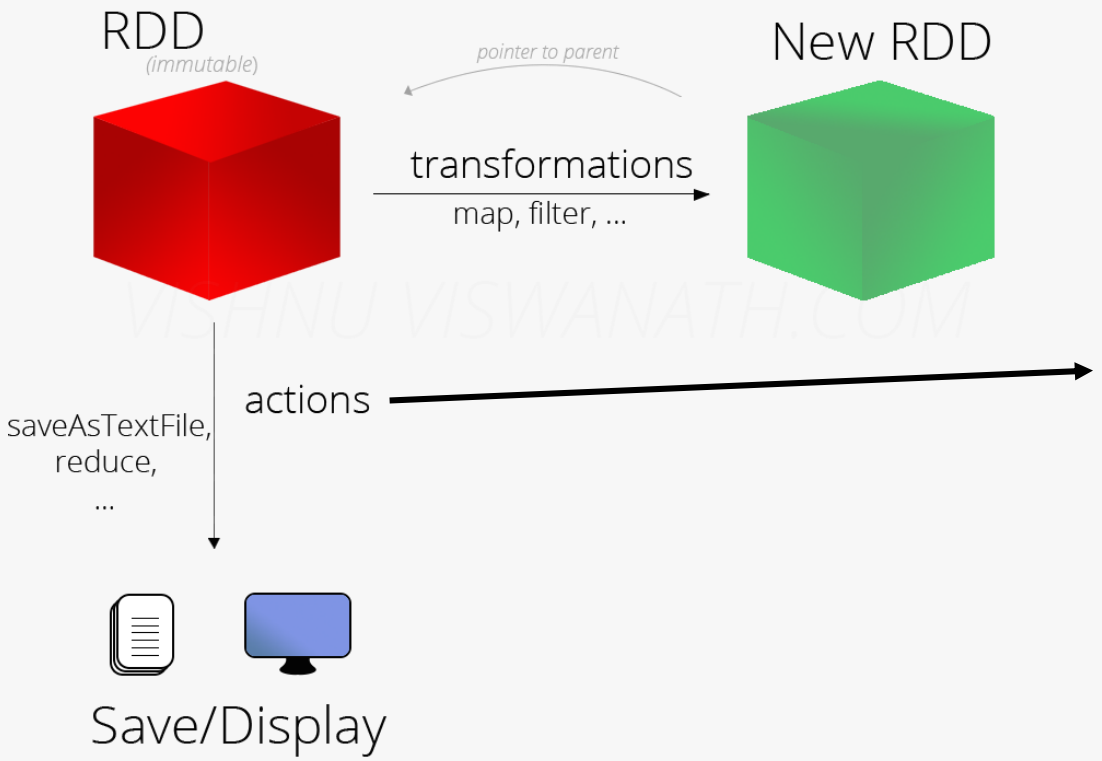


06. RDD



- map
- flatmap
- filter
- distinct
- sample
- union, intersection, subtract, cartesian

06. RDD

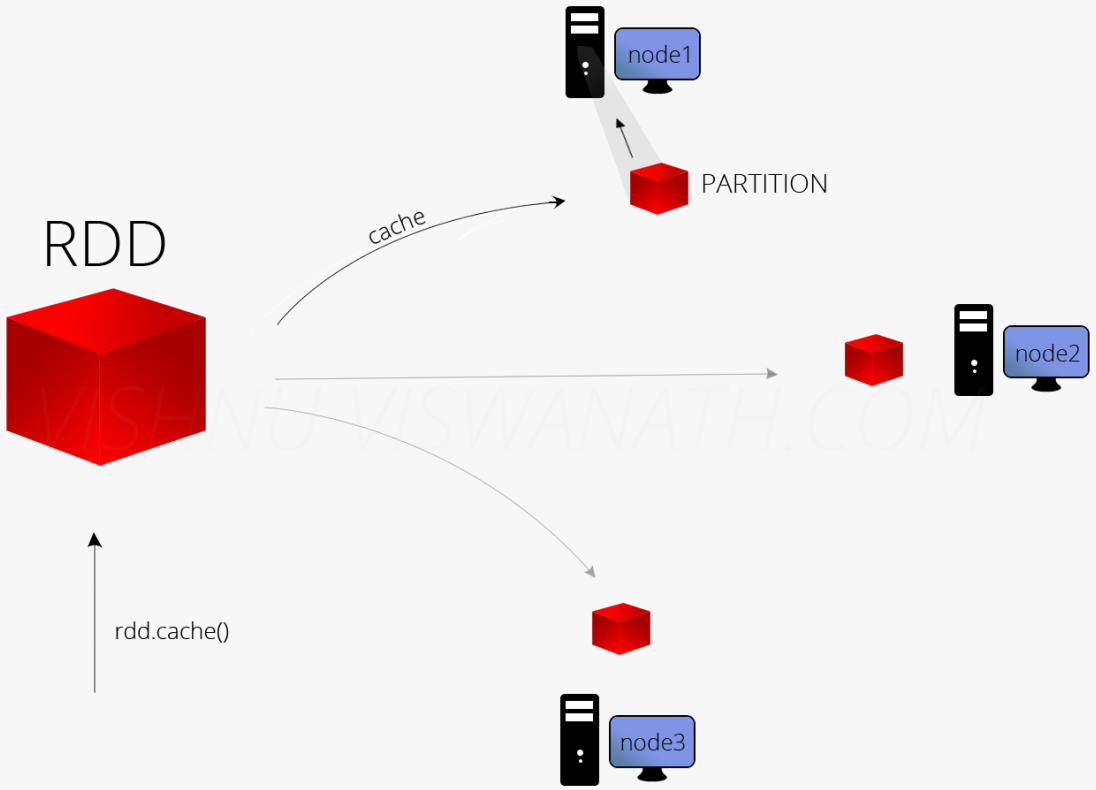


- collect
- count
- countByValue
- take
- top
- reduce
- … and more …

07. Lazy Evaluation

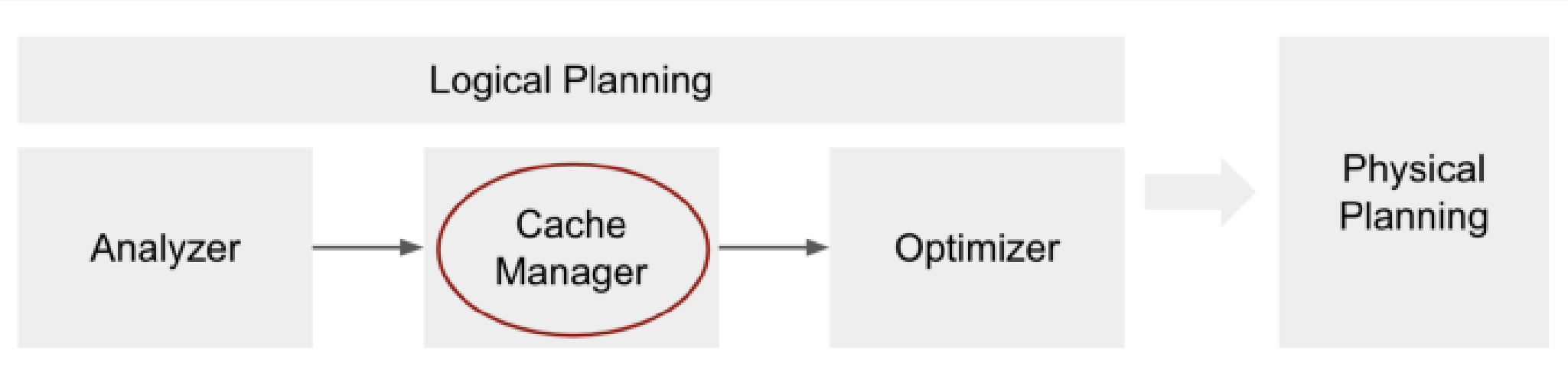
- Action이 일어나기 전까지 데이터 변환이 즉시 반영되는 것이 아님
- 즉, 데이터 변환 최적화 될 때까지 무한 반복해서 연산 수행

08. Caching

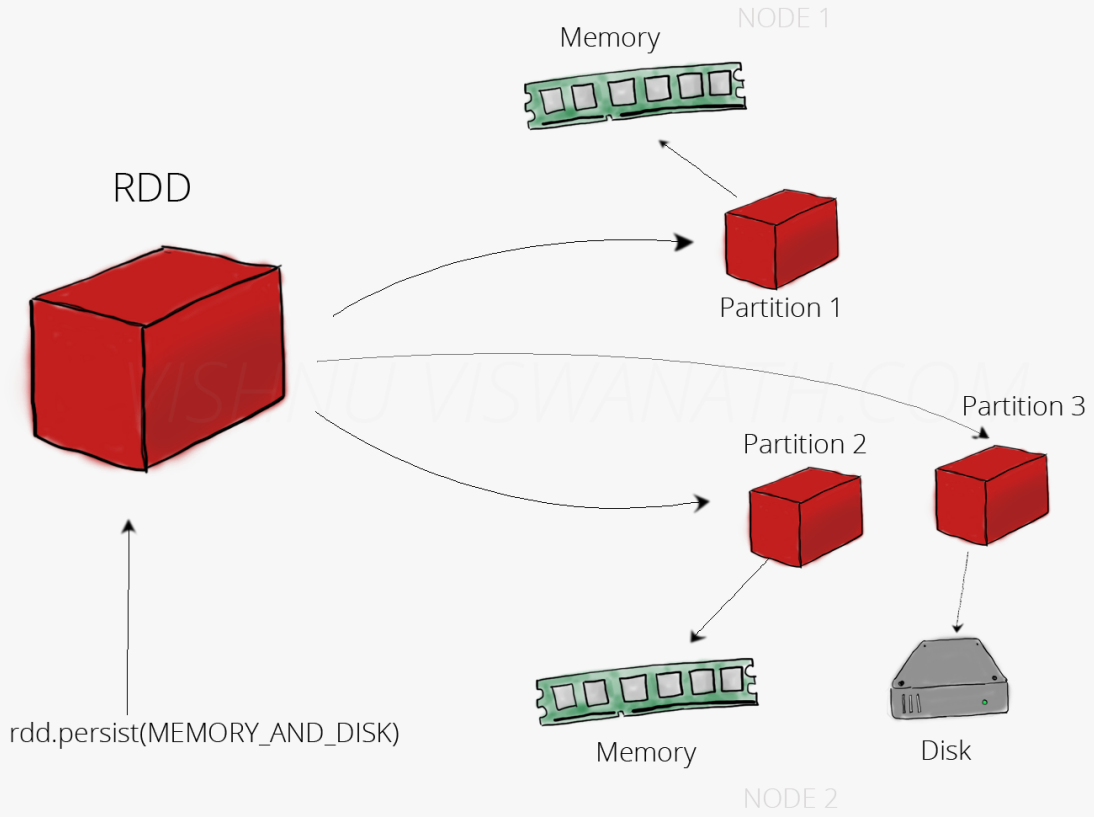


- Lazy Transformation
- Partition을 메모리에 적재 (Memory Only)
- RDD의 계산 결과를 메모리에 저장
  - 매번 다시 계산할 필요 없음
- Cache Manager에 의해 최적화

08. Caching



08. Persisting

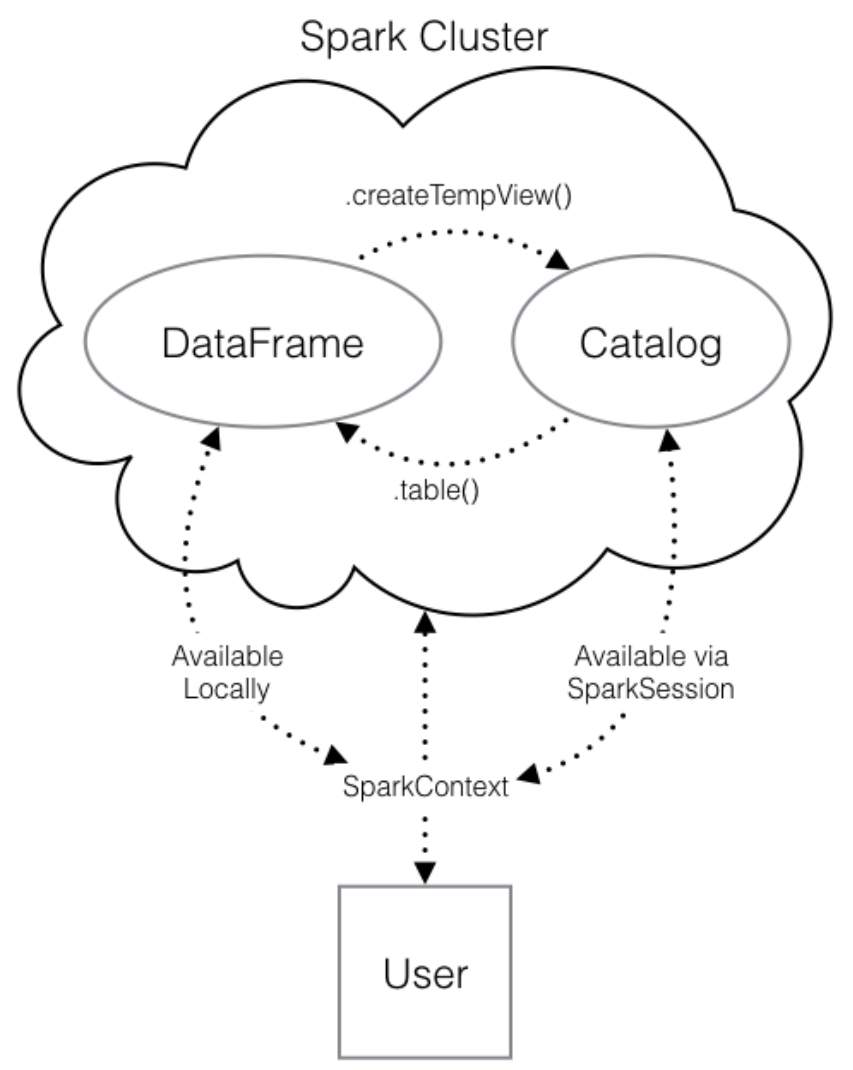


- Caching과 기본적으로 유사함
- 메모리 외에도 Disk와 같은 사용자 정의 스토리리에도 적재 가능

07. Creating RDD From

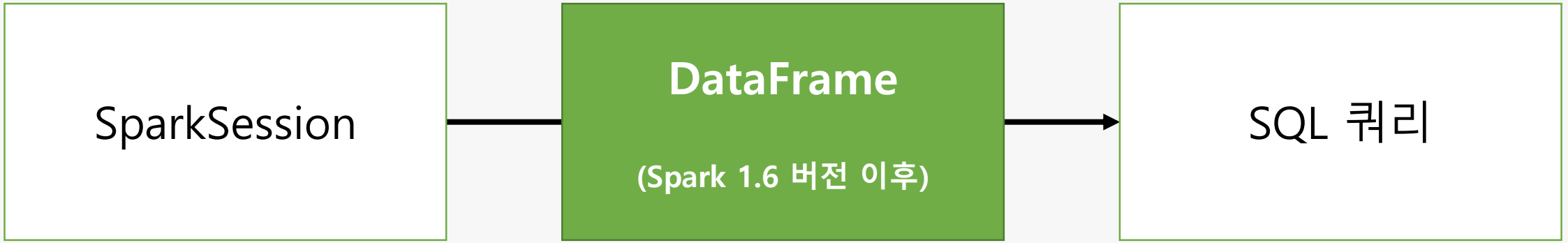
- JDBC
- Cassandra
- Hbase
- Elaticsearch
- JSON, CSV, etc

01. Spark Cluster 구조



PySpark SQL

01. Spark SQL 워크 플로우



02. DataFrame 객체

- Schema 가질 수 있음
- JSON, Hive, parquet, csv 파일 불러올 수 있음
- JDBC/ODBC, Tableau와 연동 가능
- 주요함수
  - ✓ printSchema, select, filter, etc.
- 주요 가이드라인
  - ✓ 참조: https://spark.apache.org/docs/latest/sql-programming-guide.html

02. RDD vs DataFrames

- RDD 사용하는 것은 점차 줄어듬
- MLLib, Spark Streaming도 점점 RDD 보다는 DataFrames를 기본 객체로 사용하려고 함
- DataFrame 객체에서 쿼리 작성으로 한 줄에 데이터 전처리를 끝낼 수 있음
- Hive가 지원된다면, JDBC/ODBC 서버와 직접 연동 가능

PySpark Data Pipelines

01. Data Pipeline 개요

- Input(s)
  - ✓ CSV, JSON, Web Services, Databases
- Transformations
  - ✓ withColumn( ), filter( ), drop( )
- Output(s)
  - ✓ CSV, Parquet, Database
- Validation
- Analysis

참 고 문 헌

- https://kks32-courses.gitbook.io/data-analytics/spark/rdd
- https://towardsdatascience.com/best-practices-for-caching-in-spark-sql-b22fb0f02d34