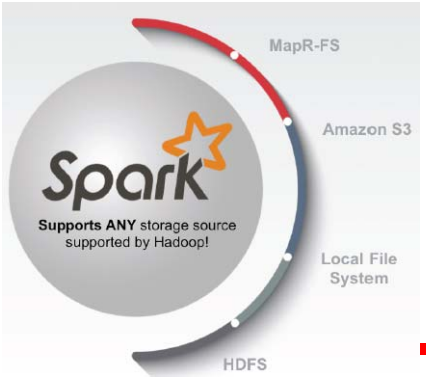


스파크 응용 구축

순천향대학교 컴퓨터공학과

이 상 정

학습 내용

1. 스파크 프로그램 수명주기
2. 스파크 응용 실행 환경
3. SFPD 스파크 응용

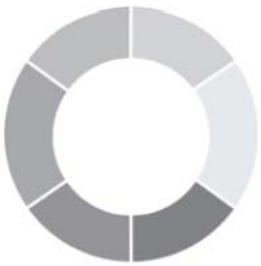
1. 스파크 프로그램 수명주기 (Spark Program Lifecycle)

스파크 응용 구축

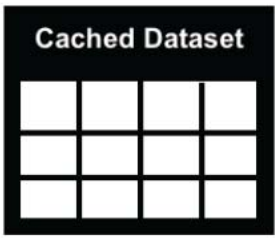
복습 – 스파크 프로그램 수명 주기 개요

1. Create input Dataset in your driver program

2. Use lazy transformations to define new Datasets



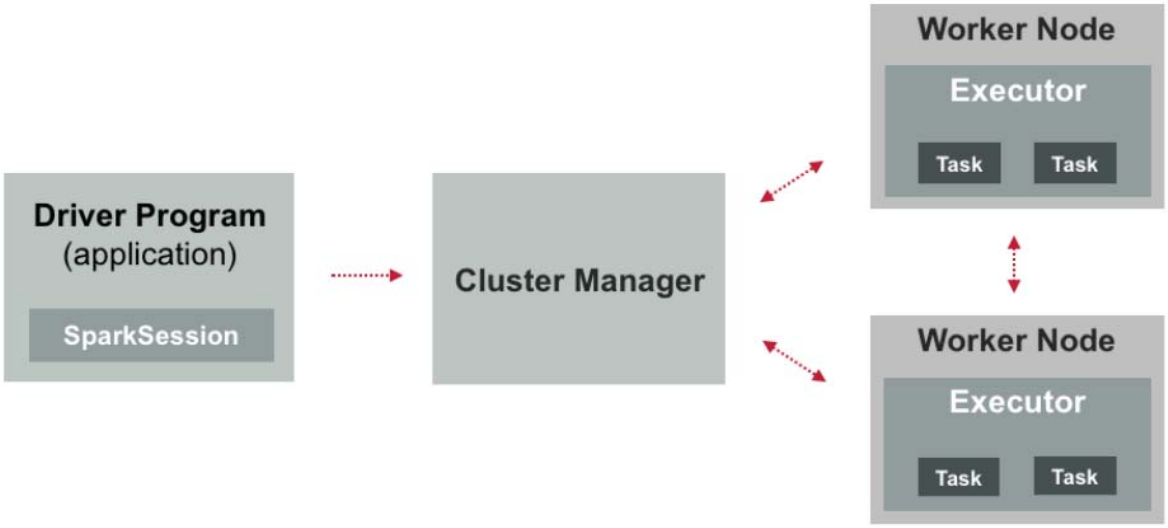
3. `Cache` any Datasets that are reused



4. Kick off computations using actions

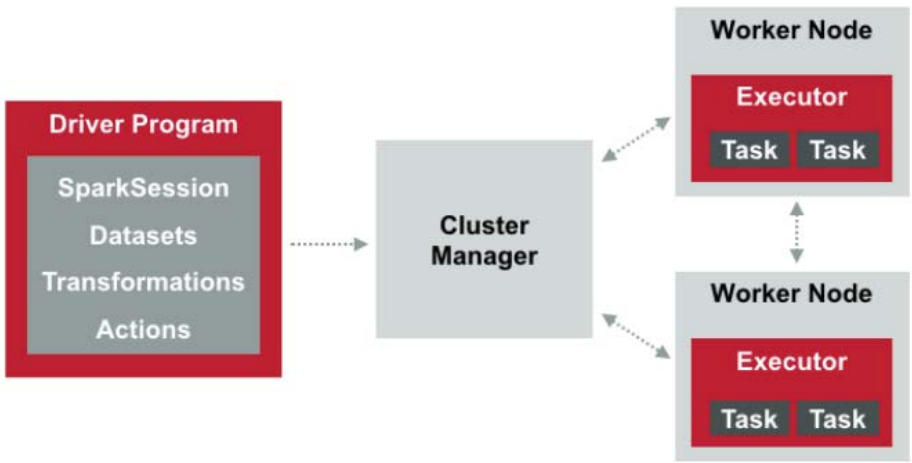
복습 – 스파크 컴포넌트

- 스파크 클러스터는 2개의 프로세스들로 구성
  - 드라이버 프로그램
  - 실행자(Executor) 프로세스를 실행하는 작업자 노드 (Worker Node)



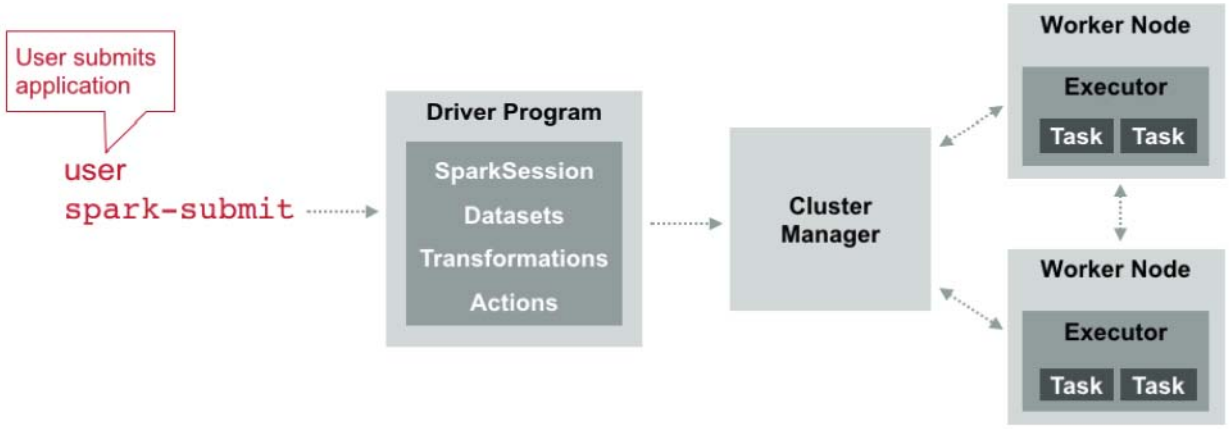
분산 스파크 응용의 컴포넌트

- 스파크는 마스터-슬레이브 구조로 구성
  - 하나의 중앙 중재자인 드라이버
  - 많은 분산 작업자들(distributed workers)
  - 드라이버와 작업자들은 각각 자신의 자바 프로세스를 실행
  - 드라이버와 실행자를 묶어서 스파크 응용이라고 함



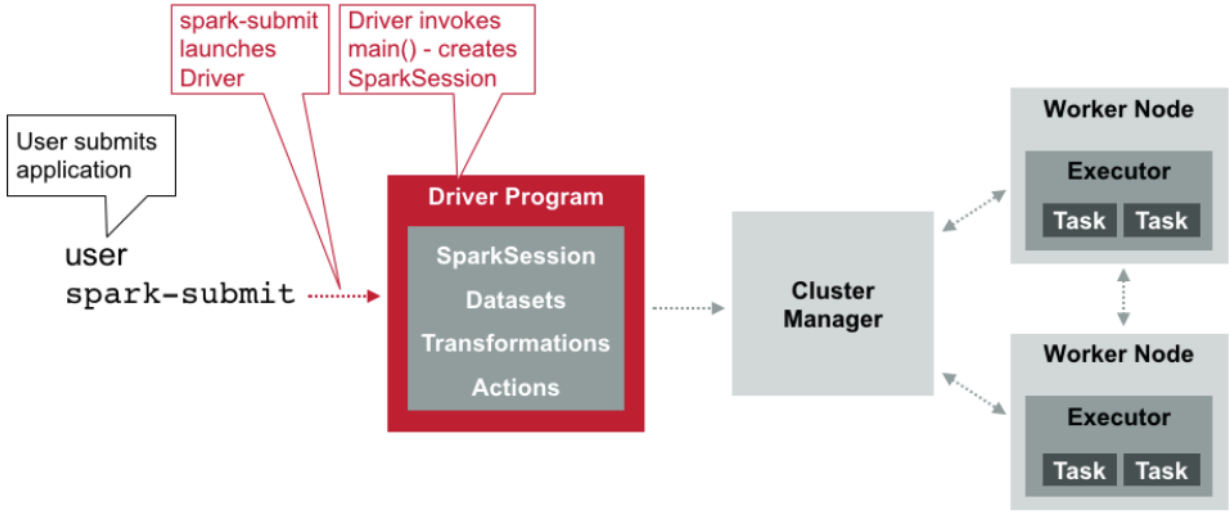
분산 스파크 응용의 컴포넌트 – 응용 제출

- 사용자는 spark-submit 명령으로 스파크 응용을 제출 (submit)



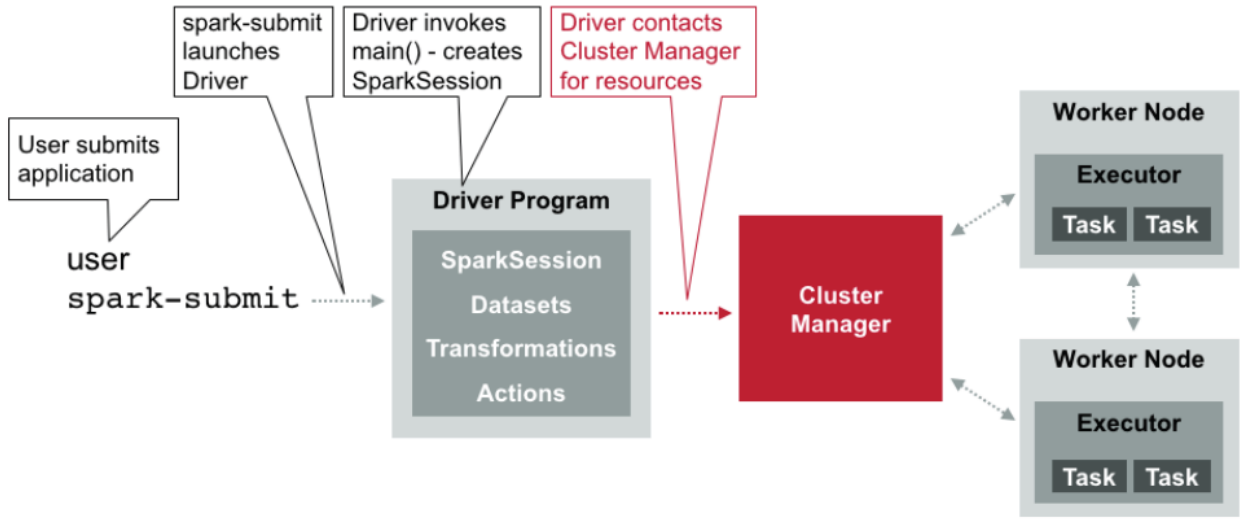
분산 스파크 응용의 컴포넌트 – 드라이버 프로그램

- spark-submit 명령으로 드라이버 프로그램의 실행 시작
  - main() 메서드 호출
  - main() 메서드에서 SparkSession 생성
  - SparkSession은 클러스터 관리자의 위치를 드라이버에게 알려줌



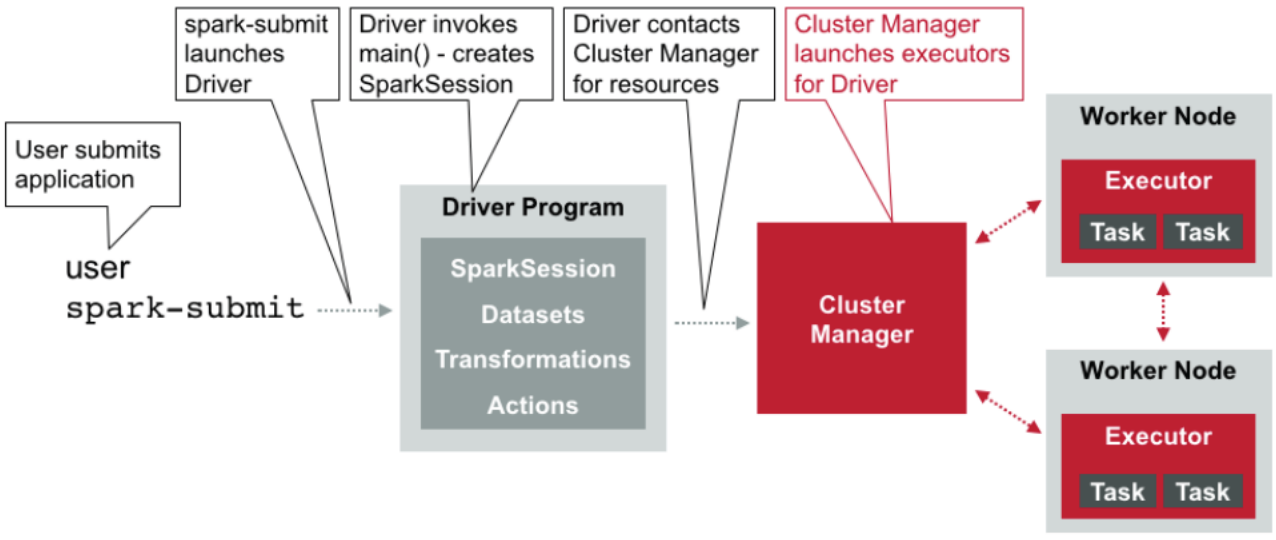
분산 스파크 응용의 컴포넌트 – 클러스터 관리자

- 드라이버 프로그램은 자원을 요청하고 실행자(executor)의 실행을 위해 드라이버 클러스터 관리자에 연결



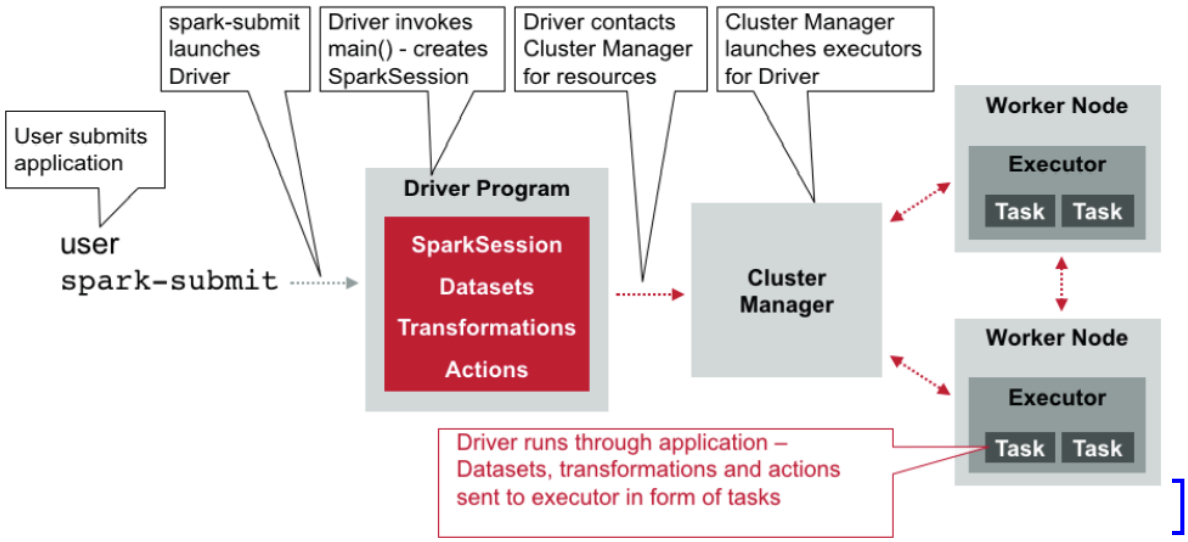
분산 스파크 응용의 컴포넌트 – 실행자

- 클러스터 관리자는 드라이버 프로그램을 위한 실행자를 실행



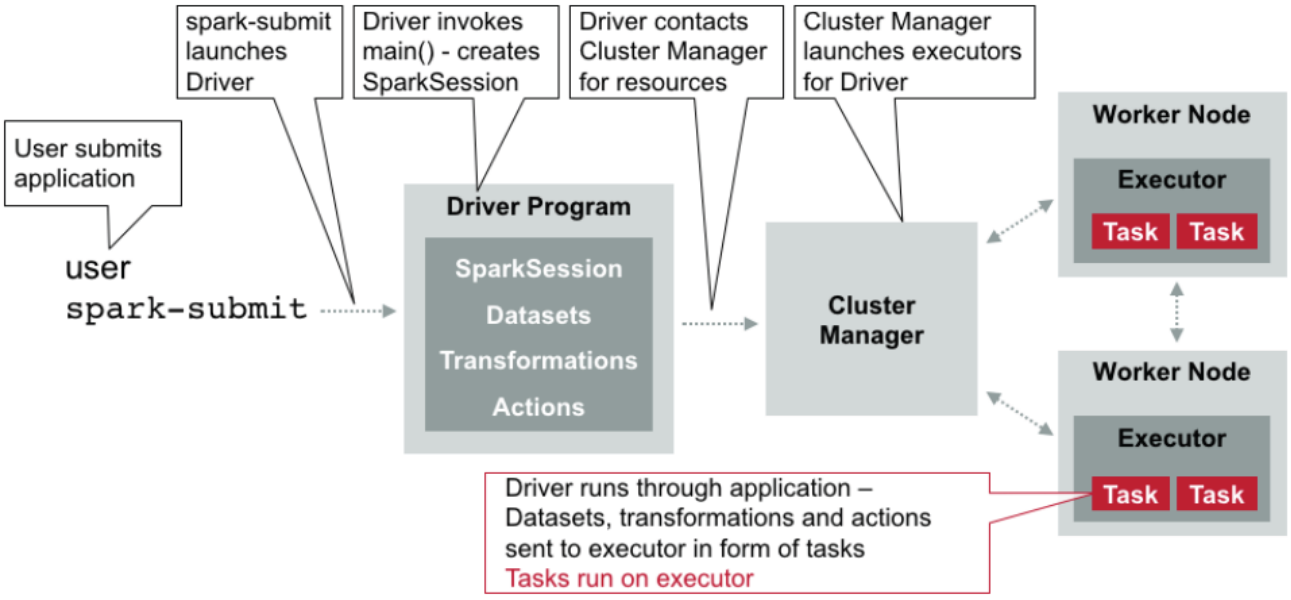
분산 스파크 응용의 컴포넌트 – 드라이버 응용 실행

- 드라이버는 응용을 실행
  - SparkSession을 생성
  - 데이터세트를 정의하고, 변환을 수행, 액션을 적용
  - Jar 또는 파이썬 파일 형태의 작업을 태스크 형식으로 실행자에 전송



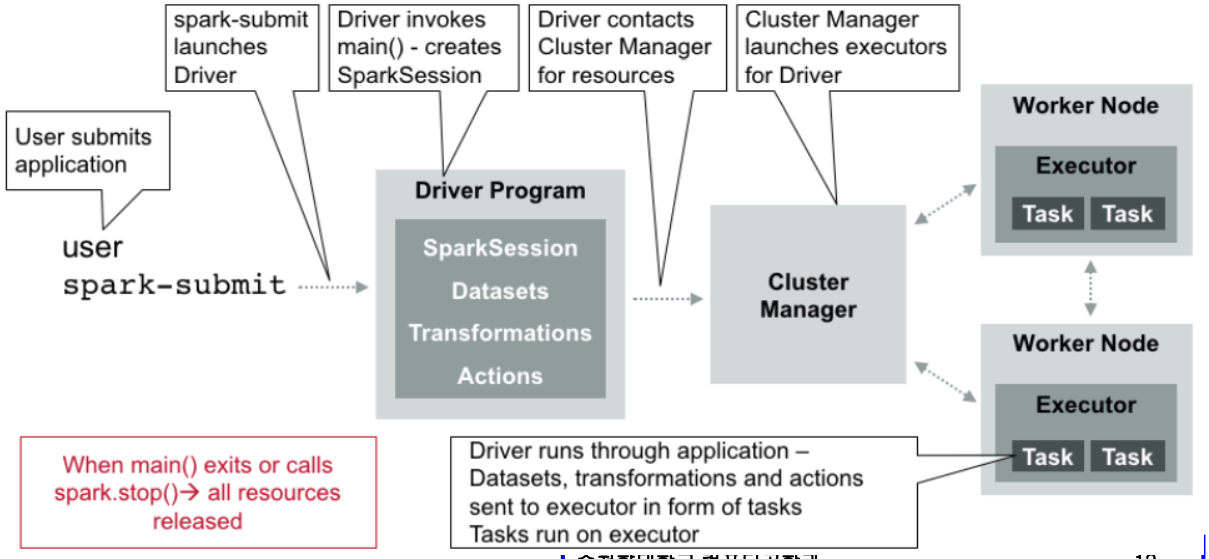
분산 스파크 응용의 컴포넌트 – 태스크 실행

- 실행자의 태스크 실행



분산 스파크 응용의 컴포넌트 – 응용 종료

- Main() 메서드가 SaprkSession.stop을 호출하여 응용 종료
  - 클러스터 관리자가 자원을 해제
  - 실행자 종료



2. 스파크 응용 실행 환경

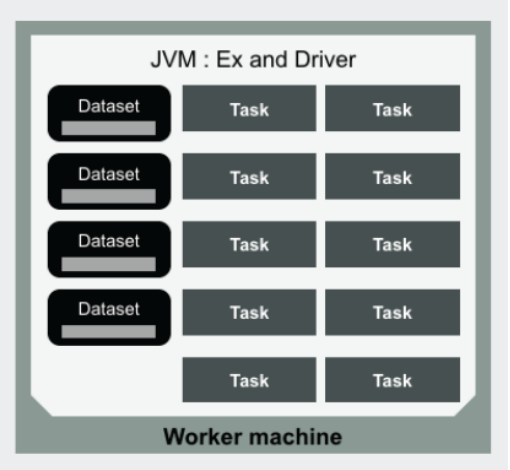
스파크 응용 실행 옵션

- 스파크는 클러스터 관리자를 통해 실행자를 실행
  - 클러스터 관리자는 스파크의 플러그가 가능한 컴포넌트(pluggable component)이어서 Hadoop YARN, Apache Mesos 등과 같은 다양한 자원들의 관리가 가능
  - 자체의 내장 클러스터 관리자도 제공

Launch Mode	Runtime Environment
1. Local	Runs in the same JVM
2. Standalone	Simple cluster manager
3. Hadoop YARN	Resource manager in Hadoop 2
4. Apache Mesos	General cluster manager

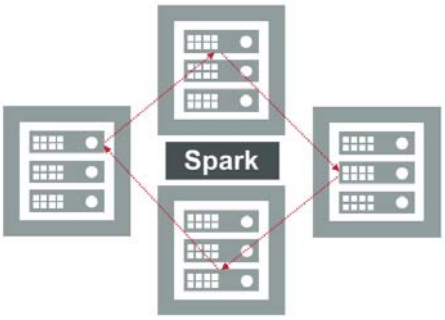
로컬 모드 (Local Mode)

- 로컬 모드는 하나의 JVM에서 드라이버와 작업자(worker)가 실행
  - 모든 데이터세트들이 같은 메모리 공간에 생성
  - 중앙의 마스터가 없고, 사용자가 실행을 시작
  - 개발 단계에서 프로토타이핑, 디버깅, 테스팅 용도에 적합



독립형 모드 (Standalone Mode)

- 스파크는 독립형 배치 모드(standalone deploy mode)를 제공
  - 스파크가 제공하는 독립적인 클러스터 모드
  - 마스터와 작업자들을 수작업으로(manually) 실행하거나,스파크에서 제공하는 스크립트를 사용하여 실행
  - 클러스터의 각 노드에 스파크의 컴파일된 버전을 배치
  - 마스터의 spark://IP:PORT URL을 SparkSession 생성자에 전달하여 스파크 클러스터에서의 응용을 실행

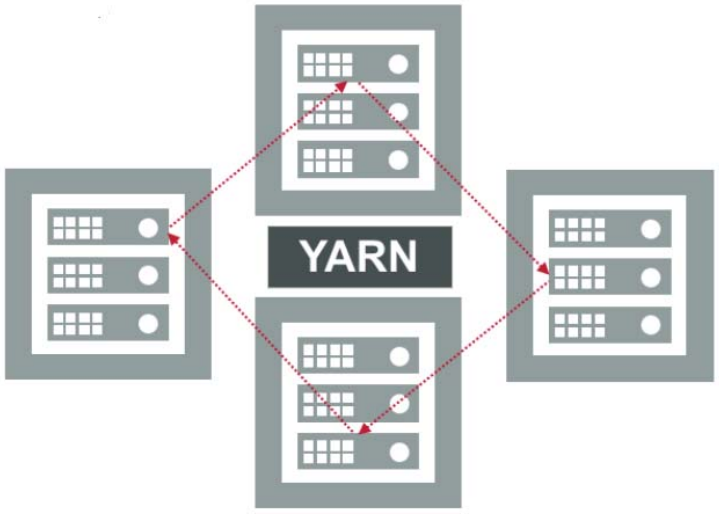



```
import org.apache.spark.sql.SparkSession

val spark =
SparkSession.builder.master("spark://<host_IP>:7077").appName(
"sfpdApp").getOrCreate()
```


하둡 YARN (1)

- 하둡 클러스터의 YARN에서 실행
- 2개의 동작 모드
  - 클러스터 모드
  - 클라이언트 모드



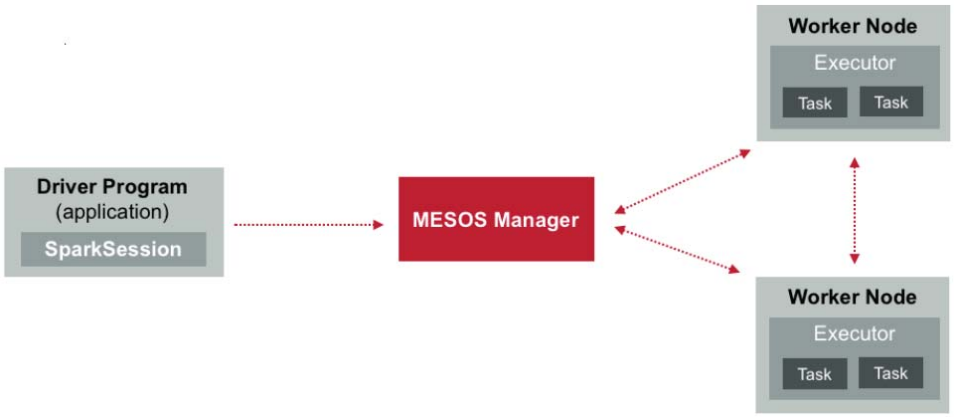
하둡 YARN (2)

- 클러스터 모드
  - 드라이버가 YARN 클러스터의 응용 마스터(application master)를 실행
  - 비동기 프로세스로 실행되어 작업의 종료를 기다리지 않음
  - 개발 완료 후 배포 버전에 적합
- 클라이언트 모드
  - 드라이버가 클라이언트 프로세스로 실행
  - 동기형 프로세스로 실행되어 작업의 종료를 기다림
  - 개발 진행 상태의 상호작용의 셸이나 디버깅 등에 적합

Cluster Mode	Client Mode
Driver launched in Application Master in cluster or worker	Driver launched in the client process that submitted the job
Can quit without waiting for job results (async)	Need to wait for result when job finishes (sync)
Suitable for production deployments	Useful for Spark interactive shell or debugging

아파치 Mesos

- 아파치 Mesos 클러스터 관리자를 사용
- 아파치 Mesos
  - 버클리대에서 개발한 클러스터 관리자
  - 데이터 센터 내의 자원을 공유/격리를 관리하는 기술로 개발



spark-submit

```
/spark-home/bin/spark-submit – used to run in any mode

./bin/spark-submit \
  --class <main-class>
  --master <master-url> \
  --deploy-mode <deploy-mode> \
  --conf <key>=<value> \
  ... # other options
  <application-jar> \
  [application-arguments]
```

- spark-submit은 응용을 실행하는 스크립트
  - --class 는 응용의 시작점인 main class 기술
  - --master 는 클러스터의 마스터 URL 또는 실행 모드
  - --deploy-mode는 동작 모드
  - --conf 는 스파크 설정 프로퍼티
  - <application-jar>는 응용의 jar 파일
  - [application-arguments]는 main() 메서드로 전달되는 인수

spark-submit 예

- 실행 모드 별 예
  - 로컬 모드
    - $SPARK_HOME/bin/spark-submit -class <class path> --master local[n] <application-jar>
  - 독립형 모드
    - $SPARK_HOME/bin/spark-submit -class <class path> --master spark:<master url> <application-jar>
  - YARN 클러스터
    - $SPARK_HOME/bin/spark-submit -class <class path> --master yarn --deploy-mode cluster <application-jar>
  - YARN 클라이언트
    - $SPARK_HOME/bin/spark-submit -class <class path> --master yarn --deploy-mode client <application-jar>

3. SFPD 스파크 응용

sfpdApp 응용

- SFPD 예를 독립형 버전의 응용으로 생성하고 실행
  - 스파크 셸/제플린이 아닌 SBT 툴로 빌드하여 jar 파일을 생성
  - spark-submit으로 실행
  - 프로그램 내용
    - 총 사건 수, 범죄유형 및 해당 건수 조사
  - 프로그램 디렉토리: ~/spark/sfpdApp
    - 스크립트: ~/spark/sfpdApp/sfpd.sbt
    - 프로그램 소스: ~/spark/sfpdApp/src/main/scala/sfpdApp.scala
    - 디렉토리 생성

```
bigdata@master:~$ cd ~/spark
bigdata@master:~/spark$ mkdir sfpdApp
bigdata@master:~/spark$ mkdir sfpdApp/src
bigdata@master:~/spark$ mkdir sfpdApp/src/main
bigdata@master:~/spark$ mkdir sfpdApp/src/main/scala
bigdata@master:~/spark$
```

```
bigdata@master:~/spark$ find sfpdApp
sfpdApp
sfpdApp/src
sfpdApp/src/main
sfpdApp/src/main/scala
bigdata@master:~/spark$
```

SFPDApp 응용 – 프로젝트 스크립트

❑ SBT 프로젝트 스크립트 작성

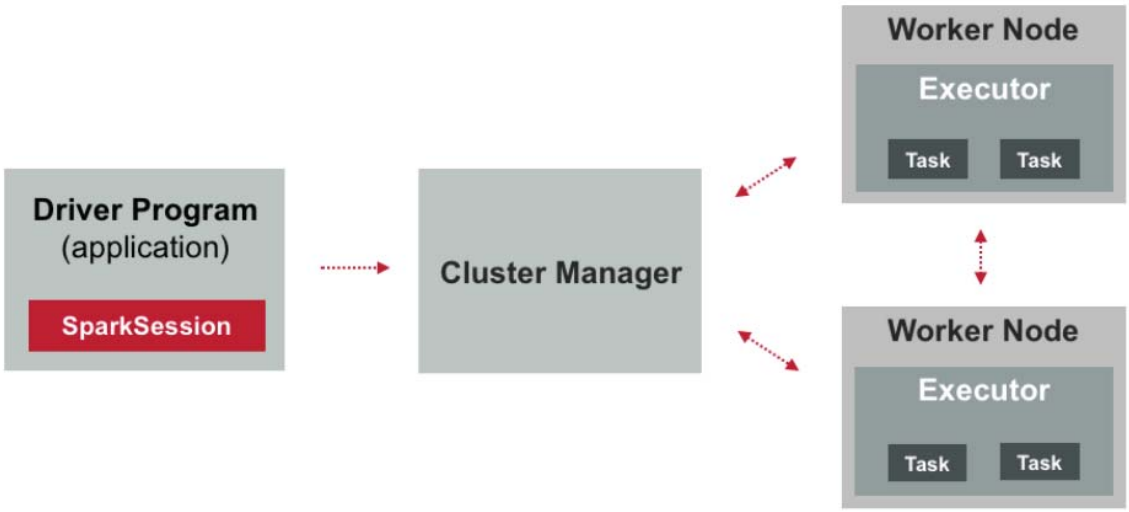
- 응용을 빌드하기 위한 라이브러리들의 버전 등 종속성 정보를 기술
- 프로젝트 스크립트
  - 메이븐 저장소(Maven Repository)의 Spark Project Core, SQL 참조
    - https://mvnrepository.com/artifact/org.apache.spark/spark-core
    - https://mvnrepository.com/artifact/org.apache.spark/spark-sql
    - 2.4.5 버전 참조
      - ≫ https://mvnrepository.com/artifact/org.apache.spark/spark-core_2.12/2.4.5
      - ≫ https://mvnrepository.com/artifact/org.apache.spark/spark-sql_2.12/2.4.5
  - ~/spark/sfpdApp/sfpd.sbt

```
name := "SFPD Project"
version := "1.0"
scalaVersion := "2.12.11"
libraryDependencies += "org.apache.spark" %% "spark-core" % "2.4.5"
libraryDependencies += "org.apache.spark" %% "spark-sql" % "2.4.5"
```


복습 – 스파크세션

❑ 스파크 프로그램은 SparkSession 생성으로 시작

- 클러스터 접근 위치 및 방법을 스파크에 알려줌
- 데이터세트를 생성하고 관리하기 위해 SparkSession의 메서드 사용



스파크 응용 구축 절차

- (스파크 셸 또는 제플린이 아닌) 독립적인 스파크 응용 구축 절차
  - 클래스 임포트
    - SparkSession 클래스 임포트
  - 클래스 정의
    - 프로그램 소스와 같은 이름의 클래스 정의
  - SparkSession 객체 생성
    - main() 메서드에서 생성
    - 응용의 이름 등의 파라미터 지정
  - SparkSession을 사용하여 spark.implicits 임포트
  - 데이터세트 생성 (read 메서드 적용)
  - 데이터세트 변환 및 액션

sfpdApp.scala

```
// ~/spark/sfpdApp/src/main/scala/sfpdApp.scala
// 클래스 임포트
import org.apache.spark.sql.SparkSession
// 케이스클래스 정의
case class Incidents(incidentnum:String, category:String, description:String,
    dayofweek:String,  date:String, time:String, pddistrict:String, resolution:String,
    address:String, X:Double, Y:Double, pdid:String)

// sfpdApp 클래스 정의
object sfpdApp {
   def main(args: Array[String]) {
    // SparkSession 객체 생성
     val spark = SparkSession.builder.appName("sfpdApp").getOrCreate()
    // spark.implicits 임포트
    import spark.implicits._

    // 콘솔 출력  메시지의 수준 조정
    val sc = spark.sparkContext
    sc.setLogLevel("WARN")
```

```
// 데이터세트 정의
val sfpdDS = spark.read.option("inferSchema",true).csv("/sparkdata/sfpd/sfpd.csv")
 .toDF("incidentnum", "category", "description", "dayofweek", "date", "time",
 "pddistrict", "resolution", "address", "X", "Y", "pdid").as[Incidents]
// 데이터세트 캐싱
sfpdDS.cache()

// 총 사건 수
  val sfpdCount = sfpdDS.count()
// 범죄 유형
  val sfpdCategory = sfpdDS.select("Category").distinct()
// 범죄 유형 당 사건 수
  val sfpdCategoryCount = sfpdDS.groupBy("Category").count()

// 콘솔에 출력
 println("Total number of incidents: %s".format(sfpdCount))
 println("Distinct categories of incidents:" )
 sfpdCategory.show(50)
 println("Number of incidents in each category:")
 sfpdCategoryCount.show(50)
 }
}
```

스파크 응용 구축

sfpdApp 빌드 스크립트 및 응용 작성

❑ SBT 스크립트 작성

```
bigdata@master:~/spark$ cd sfpdApp
bigdata@master:~/spark/sfpdApp$ nano sfpd.sbt
```

```
  GNU nano 2.9.3                                                    sfpd.sbt

name := "SFPD Project"
version := "1.0"
scalaVersion := "2.12.11"
libraryDependencies += "org.apache.spark" %% "spark-core" % "2.4.5"
libraryDependencies += "org.apache.spark" %% "spark-sql" % "2.4.5"
```

❑ 응용 작성

```
bigdata@master:~/spark/sfpdApp$
bigdata@master:~/spark/sfpdApp$ nano src/main/scala/sfpdApp.scala
```

```
  GNU nano 2.9.3                                        src/main/scala/sfpdApp.s

// ~/spark/sfpdApp/src/main/scala/sfpdApp.scala
// 클래스 임포트
import org.apache.spark.sql.SparkSession
// 케이스클래스 정의
case class Incidents(incidentnum:String, category:String, description:Strin

// sfpdApp 클래스 정의
object sfpdApp {
    def main(args: Array[String]) {
      // SparkSession 객체 생성
```

sfpdApp 응용의 빌드

- 응용 빌드
  - $ sbt package
    - target/scala-2.12/sfpd-project_2.12-1.0.jar 파일 생성

```
bigdata@master:~/spark/sfpdApp$ sbt package
[info] Updated file /home/bigdata/spark/sfpdApp/project/build.properties: set sbt.version to 1.3.8
[info] Loading project definition from /home/bigdata/spark/sfpdApp/project
[info] Loading settings for project sfpdapp from sfpd.sbt ...
[info] Set current project to SFPD Project (in build file:/home/bigdata/spark/sfpdApp/)
[info] Updating
https://repo1.maven.org/maven2/org/scala-lang/scala-library/2.12.11/scala-library-2.12.11.pom
  100.0% [##########] 1.6 KiB (690 B / s)


[info]   Compilation completed in 24.637s.
[success] Total time: 75 s (01:15), completed Jun 3, 2020 3:21:27 AM
bigdata@master:~/spark/sfpdApp$
bigdata@master:~/spark/sfpdApp$ ls
project  sfpd.sbt  src  target
bigdata@master:~/spark/sfpdApp$ ls target
scala-2.12  streams
bigdata@master:~/spark/sfpdApp$ ls target/scala-2.12
classes  sfpd-project_2.12-1.0.jar  update
bigdata@master:~/spark/sfpdApp$
```


sfpdApp 응용 실행 (1)

- spark-submit 명령을 사용하여 실행
  - $ $SPARK_HOME/bin/spark-submit --class sfpdApp --master yarn target/scala-2.12/sfpd-project_2.12-1.0.jar

```
bigdata@master:~/spark/sfpdApp$
bigdata@master:~/spark/sfpdApp$ $SPARK_HOME/bin/spark-submit  --class sfpdApp --master yarn
  target/scala-2.12/sfpd-project_2.12-1.0.jar
20/06/03 03:48:31 INFO spark.SparkContext: Running Spark version 2.4.5
20/06/03 03:48:31 INFO spark.SparkContext: Submitted application: sfpdApp
20/06/03 03:48:31 INFO spark.SecurityManager: Changing view acls to: bigdata
20/06/03 03:48:31 INFO spark.SecurityManager: Changing modify acls to: bigdata
20/06/03 03:48:31 INFO spark.SecurityManager: Changing view acls groups to:
20/06/03 03:48:31 INFO spark.SecurityManager: Changing modify acls groups to:
20/06/03 03:48:31 INFO spark.SecurityManager: SecurityManager: authentication disabled; ui
acls disabled; users  with view permissions: Set(bigdata); groups with view permissions: Se
t(); users  with modify permissions: Set(bigdata); groups with modify permissions: Set()
20/06/03 03:48:31 INFO util.Utils: Successfully started service 'sparkDriver' on port 33257
.
20/06/03 03:48:31 INFO spark.SparkEnv: Registering MapOutputTracker
20/06/03 03:48:31 INFO spark.SparkEnv: Registering BlockManagerMaster
20/06/03 03:48:31 INFO storage.BlockManagerMasterEndpoint: Using org.apache.spark.storage.D
efaultTopologyMapper for getting topology information
20/06/03 03:48:31 INFO storage.BlockManagerMasterEndpoint: BlockManagerMasterEndpoint up
20/06/03 03:48:31 INFO storage.DiskBlockManager: Created local directory at /tmp/blockmgr-6
d56fd3e-01fd-4ae9-9c72-b7232c59e293
```

sfpdApp 응용 실행 (2)

```
20/06/03 03:49:02 INFO cluster.YarnCl
heduling beginning after waiting maxR
Total number of incidents: 383775
Distinct categories of incidents:
+--------------------+
|            Category|
+--------------------+
|               FRAUD|
|             SUICIDE|
|     FAMILY_OFFENSES|
|      VEHICLE_THEFT|
|  DISORDERLY_CONDUCT|
|            WARRANTS|
|         LIQUOR_LAWS|
|               ARSON|
|FORGERY/COUNTERFE...|
|            GAMBLING|
|      MISSING_PERSON|
|             BRIBERY|
|             ASSAULT|
|SEX_OFFENSES/FORC...|
|         DRUNKENNESS|
|   RECOVERED_VEHICLE|
|      OTHER_OFFENSES|
|     STOLEN_PROPERTY|
|     SECONDARY_CODES|
|           EXTORTION|
|                TREA|
|SEX_OFFENSES/NON_...|
|           LOITERING|

Number of incidents in each category:
+--------------------+-----+
|            Category|count|
+--------------------+-----+
|               FRAUD| 7416|
|             SUICIDE|  182|
|     FAMILY_OFFENSES|  201|
|      VEHICLE_THEFT|17581|
|  DISORDERLY_CONDUCT| 1052|
|            WARRANTS|17508|
|         LIQUOR_LAWS|  494|
|               ARSON|  690|
|FORGERY/COUNTERFE...| 2025|
|            GAMBLING|   46|
|      MISSING_PERSON|11560|
|             BRIBERY|  159|
|             ASSAULT|31843|
|SEX_OFFENSES/FORC...| 2043|
|         DRUNKENNESS| 1870|
|   RECOVERED_VEHICLE|  760|
|      OTHER_OFFENSES|50611|
|     STOLEN_PROPERTY| 2803|
|     SECONDARY_CODES| 4972|
|            VANDALISM|17987|
|       DRUG/NARCOTIC|14300|
|PORNOGRAPHY/OBSCE...|   10|
|            TRESPASS| 2930|
|        NON-CRIMINAL|50269|
|       LARCENY/THEFT|96955|
|          KIDNAPPING| 1268|
|            BURGLARY|15374|
+--------------------+-----+

bigdata@master:~/spark/sfpdApp$
```


과제

- 강의 시간의 실습 내용을 정리하여 제출
  - 스파크 실행 (spark-submit) 및 동작 확인
- 텀 프로젝트 과제
  - 텀 프로젝트 데이터를 사용하여 앞에서 배운 스파크 응용 구축을 적용하고 실행

참고 자료

- MapR Academy, http://learn.mapr.com/
  - Build a Simple Apache Spark Application
    - https://learn.mapr.com/series/sparkv2/dev-361-build-and-monitor-apache-spark-applications-spark-v21
      - Lesson 4: Build a Simple Apache Spark Application