Université Constantine 2
جامعة قسنطينة 2

TDSAI

Chapitre 2 : SPARK
- Partie 3-

Dr. S.ZERABI
Faculté des NTIC
Soumeya.zerabi@univ-constantine2.dz

Etudiants concernés

Faculté/Institut	Département	Niveau	Spécialité
Nouvelles technologies	IFA	M1	SDIA

Opérations sur les Dataframes

- **Transformations**
- **Actions**
- **SQL**

select()-Transformation

Sélectionner une ou plusieurs colonnes du DF en passant les noms des colonnes à la fonction **select()**.

```
rdd = sc.sparkContext.parallelize([
    (1, 1., 'string1', date(2000, 1, 1)),
    (2, 3., 'string2', date(2000, 2, 1)),
    (3, 5., 'string3', date(2000, 3, 1))
])
df = sc.createDataFrame(rdd, schema=['a', 'b', 'c', 'd'])
df.select('a', 'b', 'c').show()
```

```
+---+---+-------+
|  a|  b|      c|
+---+---+-------+
|  1|1.0|string1|
|  2|3.0|string2|
|  3|5.0|string3|
+---+---+-------+
```

```
df.select(df.a, df.b, df.c).show()
```

```
+---+---+-------+
|  a|  b|      c|
+---+---+-------+
|  1|1.0|string1|
|  2|3.0|string2|
|  3|5.0|string3|
+---+---+-------+
```

select()-with condition-

Sélectionner des colonnes a partir d'un DF selon une condition spécifiée.

```
df.select(df.firstname, df.lastname, df.gender=='M').show()
```

```
+---------+--------+------------+
|firstname|lastname|(gender = M)|
+---------+--------+------------+
|    James|   Smith|        true|
|  Michael|        |        true|
|   Robert|Williams|        true|
|    Maria|   Jones|       false|
|      Jen|   Brown|       false|
+---------+--------+------------+
```

filter()

Filtrer les lignes du DF sur la base d'une condition spécifiée.

```
df.filter(df.gender=='M').select(df.firstname, df.lastname).show()
```

```
+---------+--------+
|firstname|lastname|
+---------+--------+
|    James|   Smith|
|  Michael|        |
|   Robert|Williams|
+---------+--------+
```

orderBy()

Trier les éléments du DF par ordre ascendant (par défaut) ou descendant.

```
rdd = sc.sparkContext.parallelize([
    ('Symbol','blanc','150M'),
    ('Ibisa','bleu','200M'),
    ('Symbol','blanc','150M'),
    ('Fiesta','bleu','180M'),
    ('I20','vert','110M'),
    ('picanto','blanc','140M'),
    ('Ibisa','noir','200M'),
    ('308','noir','250M')])
col=['marque','couleur','prix']
df=sc.createDataFrame(rdd, col)
df.orderBy('prix').show(5)
```

```
+-------+-------+----+
| marque|couleur|prix|
+-------+-------+----+
|    I20|   vert|110M|
|picanto|  blanc|140M|
| Symbol|  blanc|150M|
| Symbol|  blanc|150M|
| Fiesta|   bleu|180M|
```

orderBy() DESC()

Trier les éléments du DF par ordre descendant.

```
df.orderBy(df['prix'].desc()).show()
```

```
+-------+-------+----+
| marque|couleur|prix|
+-------+-------+----+
|    308|   noir|250M|
|  Ibisa|   bleu|200M|
|  Ibisa|   noir|200M|
| Fiesta|   bleu|180M|
| Symbol|  blanc|150M|
| Symbol|  blanc|150M|
|picanto|  blanc|140M|
|    I20|   vert|110M|
+-------+-------+----+
```

```
df.orderBy(df.prix.desc()).show()
```

withColumn()

maj des valeurs d'une colonne d'un DF.

```
df.withColumn("salary",df["salary"]*2).show()
```

```
+---------+----------+--------+----------+------+------+
|firstname|middlename|lastname|       dob|gender|salary|
+---------+----------+--------+----------+------+------+
|    James|          |   Smith|1991-04-01|     M|  6000|
|  Michael|      Rose|        |2000-05-19|     M|  8000|
|   Robert|          |Williams|1978-09-05|     M|  8000|
|    Maria|      Anne|   Jones|1967-12-01|     F|  8000|
|      Jen|      Mary|   Brown|1980-02-17|     F|    -2|
+---------+----------+--------+----------+------+------+
```

```
+---------+----------+--------+----------+------+------+
|firstname|middlename|lastname|       dob|gender|salary|
+---------+----------+--------+----------+------+------+
|    James|          |   Smith|1991-04-01|     M|  3000|
|  Michael|      Rose|        |2000-05-19|     M|  4000|
|   Robert|          |Williams|1978-09-05|     M|  4000|
|    Maria|      Anne|   Jones|1967-12-01|     F|  4000|
|      Jen|      Mary|   Brown|1980-02-17|     F|    -1|
+---------+----------+--------+----------+------+------+
```

withColumn()

maj du type d'une colonne d'un DF.

```
df2 = df.withColumn("salary",df.salary.cast("double"))
df2.printSchema()
```

```
root
 |-- firstname: string (nullable = true)
 |-- middlename: string (nullable = true)
 |-- lastname: string (nullable = true)
 |-- dob: string (nullable = true)
 |-- gender: string (nullable = true)
 |-- salary: double (nullable = true)
```

```
root
 |-- firstname: string (nullable = true)
 |-- middlename: string (nullable = true)
 |-- lastname: string (nullable = true)
 |-- dob: string (nullable = true)
 |-- gender: string (nullable = true)
 |-- salary: long (nullable = true)
```

withColumn()

Ajout d'une colonne au DF à partir d'une colonne existante.

```
df.withColumn("CopiedColumn",df.salary*100).show()
```

```
+---------+----------+--------+----------+------+------+-------------+
|firstname|middlename|lastname|       dob|gender|salary|CopiedColumn|
+---------+----------+--------+----------+------+------+-------------+
|    James|          |   Smith|1991-04-01|     M|  3000|       300000|
|  Michael|      Rose|        |2000-05-19|     M|  4000|       400000|
|   Robert|          |Williams|1978-09-05|     M|  4000|       400000|
|    Maria|      Anne|   Jones|1967-12-01|     F|  4000|       400000|
|      Jen|      Mary|   Brown|1980-02-17|     F|    -1|         -100|
+---------+----------+--------+----------+------+------+-------------+
```

Ajouter d'autres colonnes au DF.

```
from pyspark.sql.functions import lit
df.withColumn("Country", lit("Canada")).withColumn("anotherColumn",lit("anotherValue")).show()
```

```
+---------+----------+--------+----------+------+------+-------+-------------+
|firstname|middlename|lastname|       dob|gender|salary|Country|anotherColumn|
+---------+----------+--------+----------+------+------+-------+-------------+
|    James|          |   Smith|1991-04-01|     M|  3000| Canada| anotherValue|
|  Michael|      Rose|        |2000-05-19|     M|  4000| Canada| anotherValue|
|   Robert|          |Williams|1978-09-05|     M|  4000| Canada| anotherValue|
|    Maria|      Anne|   Jones|1967-12-01|     F|  4000| Canada| anotherValue|
|      Jen|      Mary|   Brown|1980-02-17|     F|    -1| Canada| anotherValue|
+---------+----------+--------+----------+------+------+-------+-------------+
```

withColumnRenamed()

Renommer une colonne dans un DF.

```
df.withColumnRenamed('dob', 'DateBirthday').show()
```

```
+---------+----------+--------+-------------+------+------+
|firstname|middlename|lastname|DateBirthday|gender|salary|
+---------+----------+--------+-------------+------+------+
|    James|          |   Smith|  1991-04-01|     M|  3000|
|  Michael|      Rose|        |  2000-05-19|     M|  4000|
|   Robert|          |Williams|  1978-09-05|     M|  4000|
|    Maria|      Anne|   Jones|  1967-12-01|     F|  4000|
|      Jen|      Mary|   Brown|  1980-02-17|     F|    -1|
+---------+----------+--------+-------------+------+------+
```

groupBy()

Regroupe les éléments du DF puis effectue une agrégation.

```
rdd = sc.sparkContext.parallelize([
    ('Symbol','blanc','150M'),
    ('Ibisa','bleu','200M'),
    ('Symbol','blanc','150M'),
    ('Fiesta','bleu','180M'),
    ('I20','vert','110M'),
    ('picanto','blanc','140M'),
    ('Ibisa','noir','200M'),
    ('308','noir','250M'),
])
df=sc.createDataFrame(rdd, schema=['marque','couleur','prix'])
df.groupBy(df.marque).count().show()
```

```
+-------+-----+
| marque|count|
+-------+-----+
| Symbol|    2|
|  Ibisa|    2|
| Fiesta|    1|
|    308|    1|
|    I20|    1|
|picanto|    1|
+-------+-----+
```

drop()

Supprimer une colonne dans un DF.

```
df.drop('gender').show()
```

```
+---------+----------+--------+----------+------+
|firstname|middlename|lastname|       dob|salary|
+---------+----------+--------+----------+------+
|    James|          |   Smith|1991-04-01|  3000|
|  Michael|      Rose|        |2000-05-19|  4000|
|   Robert|          |Williams|1978-09-05|  4000|
|    Maria|      Anne|   Jones|1967-12-01|  4000|
|      Jen|      Mary|   Brown|1980-02-17|    -1|
+---------+----------+--------+----------+------+
```

limit(*n*)

Limiter le résultat au nombre d'éléments spécifié (*n*).

```
df.limit(3).show()
```

```
+------+-------+----+
|marque|couleur|prix|
+------+-------+----+
|Symbol|  blanc|150M|
| Ibisa|   bleu|200M|
|Symbol|  blanc|150M|
+------+-------+----+
```

distinct()

Elimine les duplications des lignes dans le DF.

```
data=[('Ahmed',20,'Alg'),
      ('Mohamed',30,'Alg'),
      ('Ahmed',20,'Alg'),
      ('Sami',15,'Alg')]
col=['nom', 'age', 'pays']
df=sc.createDataFrame(data,col)
df.distinct().show()
```

```
+-------+---+----+
|    nom|age|pays|
+-------+---+----+
|Mohamed| 30| Alg|
|  Ahmed| 20| Alg|
|   Sami| 15| Alg|
+-------+---+----+
```

collect()-Action

Récupère tous les éléments de l'ensemble de données (de tous les nœuds) vers le nœud drive.

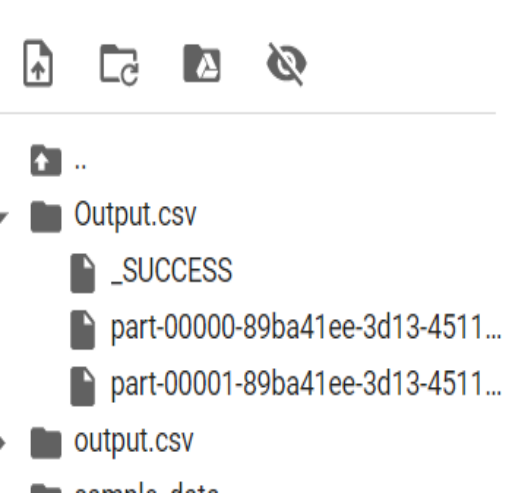
```
rdd = sc.sparkContext.parallelize([
    (1, 1., 'string1', date(2000, 1, 1)),
    (2, 3., 'string2', date(2000, 2, 1)),
    (3, 5., 'string3', date(2000, 3, 1))
])
df = sc.createDataFrame(rdd, schema=['a', 'b', 'c', 'd'])

df.collect()
```

```
[Row(a=1, b=1.0, c='string1', d=datetime.date(2000, 1, 1)),
 Row(a=2, b=3.0, c='string2', d=datetime.date(2000, 2, 1)),
 Row(a=3, b=5.0, c='string3', d=datetime.date(2000, 3, 1))]
```

Sauvegarder un DF sur disque

format() spécifie le format du DF créé.




```
data = [('James','','Smith','1991-04-01','M',3000),
  ('Michael','Rose','','2000-05-19','M',4000),
  ('Robert','','Williams','1978-09-05','M',4000),
  ('Maria','Anne','Jones','1967-12-01','F',4000),
  ('Jen','Mary','Brown','1980-02-17','F',-1)
]

columns = ["firstname","middlename","lastname","dob","gender","salary"]
df=sc.createDataFrame(data, columns)
df.write.format('csv').options(header='True').save('Output.csv')
```

SQL

… … … … … …

SparkSQL

- Permet d'effectuer des requêtes sur les données avec une syntaxe similaire à SQL.
- Pouvoir exécuter du SQL sur les dataframes sans la création d'une BD.
- **createOrReplaceTempView():** méthode permettant de créer une vue pour exécuter les requêtes SQLs.
- **Sql():** méthode permettant d'interroger la vue temporaire créé et retourner un dataframe.
- La vue temporaire créé est accessible tant que la session spark est disponible.

SparkSQL

```
data = [('James','','Smith','1991-04-01','M',3000),
  ('Michael','Rose','','2000-05-19','M',4000),
  ('Robert','','Williams','1978-09-05','M',4000),
  ('Maria','Anne','Jones','1967-12-01','F',4000),
  ('Jen','Mary','Brown','1980-02-17','F',-1)
]
columns = ["firstname","middlename","lastname","dob","gender","salary"]
df=sc.createDataFrame(data, columns)
```

```
df.filter(df.gender=='M').select(df.firstname, df.lastname).show()
```

```
df.createOrReplaceTempView('personne')
sc.sql('select firstname,lastname from personne where gender=="M"').show()
```

SparkSQL

```
rdd = sc.sparkContext.parallelize([
    ('Symbol','blanc','150M'),
    ('Ibisa','bleu','200M'),
    ('Symbol','blanc','150M'),
    ('Fiesta','bleu','180M'),
    ('I20','vert','110M'),
    ('picanto','blanc','140M'),
    ('Ibisa','noir','200M'),
    ('308','noir','250M')])
col=['marque','couleur','prix']
df=sc.createDataFrame(rdd, col)
```

```
df.orderBy(df.prix.desc()).show()
```

```
+-------+-------+----+
| marque|couleur|prix|
+-------+-------+----+
|    308|   noir|250M|
|  Ibisa|   bleu|200M|
|  Ibisa|   noir|200M|
| Fiesta|   bleu|180M|
| Symbol|  blanc|150M|
| Symbol|  blanc|150M|
|picanto|  blanc|140M|
|    I20|   vert|110M|
+-------+-------+----+
```

```
df.createOrReplaceTempView("vehicule")
sc.sql("select marque,prix from vehicule ORDER BY prix desc ").show()
```