

```
public class Main {
    public static void main(String[] args) {
        SparkConf conf = new SparkConf()
            .setAppName("")
            .setMaster("local");

        JavaSparkContext sc = new JavaSparkContext(conf);

        SparkSession spark = SparkSession
            .builder()
            .sparkContext(sc.sc())
            .getOrCreate();

        Dataset<Row> orders = spark.read()
            .option("header", true)
            .option("inferSchema", true)
            .csv("instacart/orders.csv");

// 1) Policzenie liczby zamówień wg godziny ich wykonania
        Dataset x1 = orders.groupBy(col("order_hour_of_day"))
            .count()
            .orderBy("order_hour_of_day");
        x1.show(((int) x1.count()), false);

// 2) Policzenie zamówień wg dnia tygodnia ich wykonania (wyświetlenie w kolejności rosnącej).
        List<Row> list = new ArrayList<Row>();
        list.add(RowFactory.create("Sunday", "0"));
        list.add(RowFactory.create("Monday", "1"));
        list.add(RowFactory.create("Tuesday", "2"));
        list.add(RowFactory.create("Wednesday", "3"));
        list.add(RowFactory.create("Thursday", "4"));
        list.add(RowFactory.create("Friday", "5"));
        list.add(RowFactory.create("Saturday", "6"));

        List<org.apache.spark.sql.types.StructField> listOfStructField = new
        ArrayList<org.apache.spark.sql.types.StructField>();
        listOfStructField.add(DataTypes.createStructField("name", DataTypes.StringType, true));
        listOfStructField.add(DataTypes.createStructField("number", DataTypes.StringType,
true));
        StructType structType = DataTypes.createStructType(listOfStructField);
        Dataset<Row> days = spark.createDataFrame(list, structType);
        days.show();

        Dataset x21 = orders.groupBy(col("order_dow")).count();
        Dataset<Row> x22 = days
            .join(x21, orders.col("order_dow").equalTo(days.col("number")))
            .orderBy("count")
            .select("name", "count");
        x22.show(((int) x22.count()), false);

// 3) Policzenie liczby oraz procentowego rozkładu produktów wg działu sklepu (wyświetlenie w
kolejności rosnącej)
        Dataset<Row> products = spark.read()
            .option("header", true)
            .option("inferSchema", true)
            .csv("instacart/products.csv");
        Dataset<Row> departments = spark.read()
            .option("header", true)
```

```

        .option("inferSchema", true)
        .csv("instacart/departments.csv");

long sum = products.count();
Dataset x31 = products.groupBy("department_id")
    .count()
    .join(departments, "department_id")
    .select("department", "count");
Dataset x32 = x31.withColumn("percentage", col("count").divide(sum).multiply(100))
    .orderBy("percentage");
x32.show((int) x32.count(), false);
    }
}

```

Aplikacja została uruchomiona w środowisku IntelliJ i otrzymano następujące wyniki.

1) Policzenie liczby zamówień wg godziny ich wykonania

order_hour_of_day	count
0	22758
1	12398
2	7539
3	5474
4	5527
5	9569
6	30529
7	91868
8	178201
9	257812
10	288418
11	284728
12	272841
13	277999
14	283042
15	283639
16	272553
17	228795
18	182912
19	140569
20	104292
21	78109
22	61468
23	40043

2) Do wyświetlenia wyniku należy użyć nazw dni. W tym celu należy stworzyć kolekcję dataframe (Dataset<Row>) zawierającą numery i nazwy dni tygodnia, a następnie połączyć ją z wynikiem zliczania zamówień (proszę jako pierwszy dzień tygodnia przyjąć niedzielę).

```
+-----+-----+
|name    |number|
+-----+-----+
|Sunday  |0      |
|Monday  |1      |
|Tuesday |2      |
|Wednesday|3     |
|Thursday|4      |
|Friday  |5      |
|Saturday|6      |
+-----+-----+
```

Policzenie zamówień wg dnia tygodnia ich wykonania (wyświetlenie w kolejności rosnącej).

```
+-----+-----+
|name    |count |
+-----+-----+
|Thursday|426339|
|Wednesday|436972|
|Saturday|448761|
|Friday  |453368|
|Tuesday |467260|
|Monday  |587478|
|Sunday  |600905|
+-----+-----+
```

3) Policzenie liczby oraz procentowego rozkładu produktów wg działu sklepu (wyświetlenie w kolejności rosnącej)

department	count	percentage
bulk	38	0.07647721783931734
other	548	1.1028819835775239
meat seafood	907	1.8253904363226534
pets	972	1.9562067299951698
alcohol	1054	2.12123651585896
babies	1081	2.175575591692159
breakfast	1115	2.244002576074706
international	1139	2.2923039768153277
missing	1258	2.5317984221542424
deli	1322	2.6606021574625665
bakery	1516	3.0510384801159236
produce	1684	3.3891482853002737
dry goods pasta	1858	3.7393334406697796
canned goods	2092	4.210272097890839
household	3085	6.208742553534052
dairy eggs	3449	6.941313798100144
frozen	4007	8.064321365319595
beverages	4365	8.784817259700532
pantry	5371	10.809450974078247
snacks	6264	12.606665593302205
personal care	6563	13.208420544195782