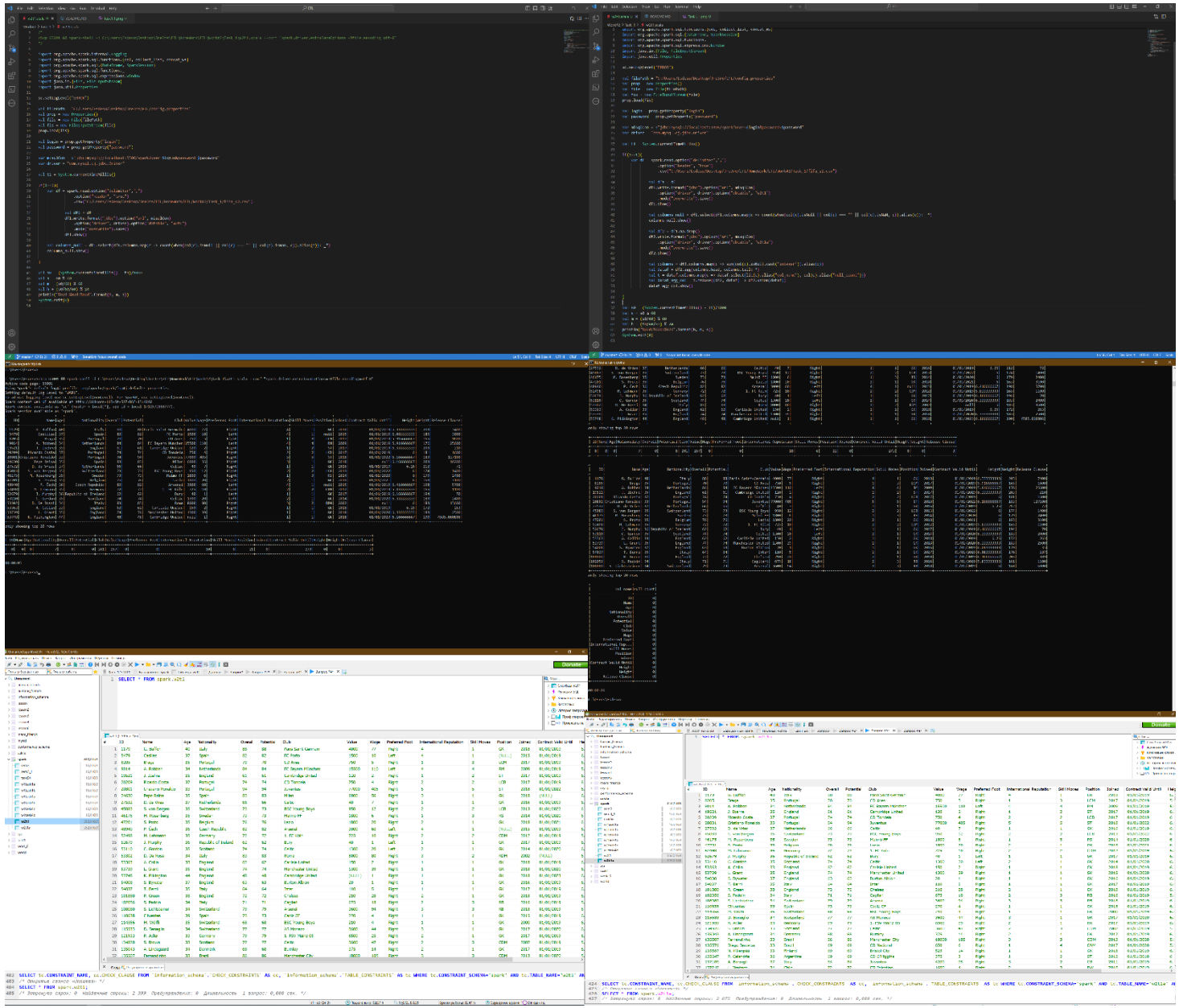
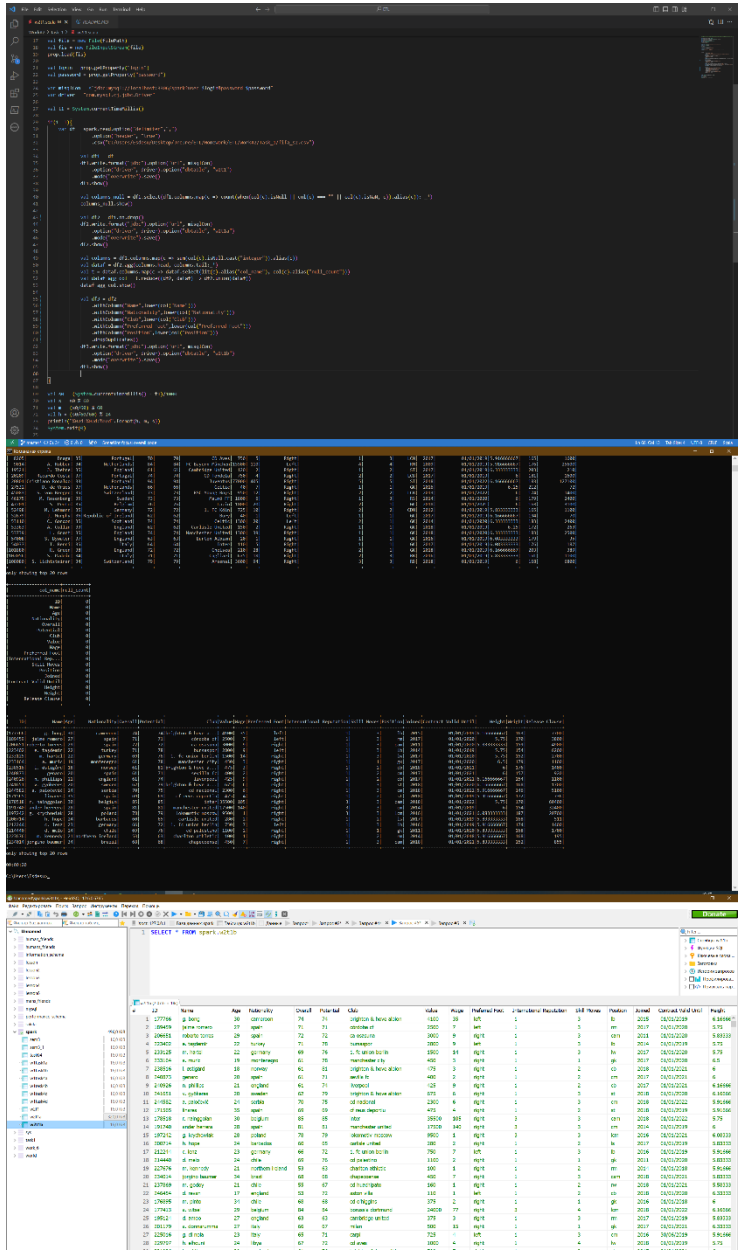
Урок 2. Введение в подготовку данных для аналитиков. Таблицы фактов и таблицы измерений

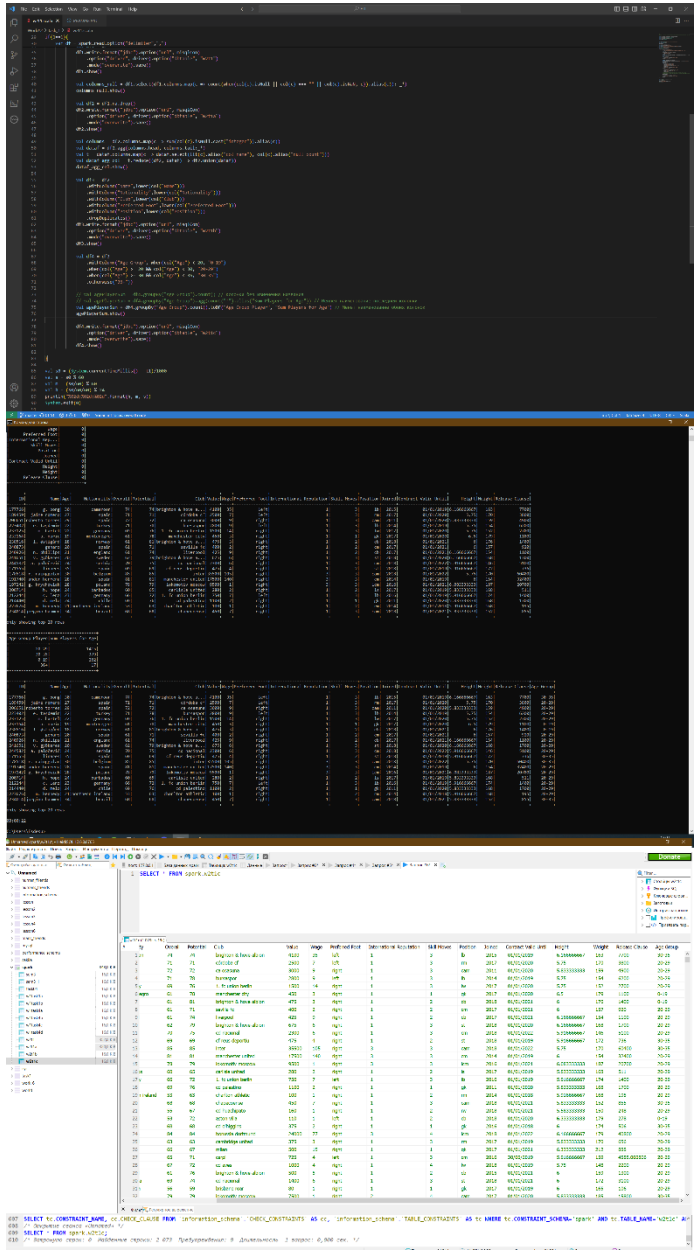
1. Скачайте датасет fifas2.csv. Проанализируйте его и определите, какие данные являются неполными. Удалите ненужные колонки и недостающие значения.

2. Найдите в датафрейме полные дубликаты и удалите их. Значения могут быть одинаковыми, но написаны по-разному. Например, может отличаться размер регистра (заглавные и строчные буквы). Особое внимание уделить колонке с названиями команд.

3. Напишите функцию, которая добавит колонку с разбиением возраста по группам: до 20, от 20 до 30, от 30 до 36 и старше 36. Посчитайте количество футболистов в каждой категории.
ЗАДАНИЕ НУЖНО СДЕЛАТЬ С ПОМОЧЬЮ APACHE SPARK КАК ПОКАЗАНО НА СЕМИНАРЕ.
Соберите все в один pdf, код и скриншоты записанных таблиц с данными, скриншоты отработки лога спарка.







```scala
/*
chcp 65001 && spark-shell -i C:\Users\Esdesu\Desktop\JreJre\ETL\HomeWork\ETL\Work#2\Task_1\w2t1.scala --conf
"spark.driver.extraJavaOptions=-Dfile.encoding=utf-8"
*/

import org.apache.spark.internal.Logging
import org.apache.spark.sql.functions.{col, collect_list, concat_ws}
import org.apache.spark.sql.{DataFrame, SparkSession}
import org.apache.spark.sql.functions._
import org.apache.spark.sql.expressions.Window
import java.io.{File, FileInputStream}
import java.util.Properties

sc.setLogLevel("ERROR")

val filePath = "C:/Users/Esdesu/Desktop/JreJre/ETL/config.properties"
val prop = new Properties()
val file = new File(filePath)
val fis = new FileInputStream(file)
prop.load(fis)

val login = prop.getProperty("login")
val password = prop.getProperty("password")

var misqlCon = s"jdbc:mysql://localhost:3306/spark?user=$login&password=$password"
var driver = "com.mysql.cj.jdbc.Driver"

val t1 = System.currentTimeMillis()

if(1==1){
    var df = spark.read.option("delimiter",",")
                .option("header", "true")
```

```
            .csv("C:/Users/Esdesu/Desktop/JreJre/ETL/HomeWork/ETL/Work#2/Task_1/fifa_s2.csv")

        val df1 = df
        df1.write.format("jdbc").option("url", misqlCon)
            .option("driver", driver).option("dbtable", "w2t1")
            .mode("overwrite").save()
        df1.show()

        val columns_null = df1.select(df1.columns.map(c => count(when(col(c).isNull || col(c) === "" || col(c).isNaN,
c)).alias(c)): _*)
        columns_null.show()

        val df2 = df1.na.drop()
        df2.write.format("jdbc").option("url", misqlCon)
            .option("driver", driver).option("dbtable", "w2t1a")
            .mode("overwrite").save()
        df2.show()

        val columns = df2.columns.map(c => sum(col(c).isNull.cast("integer")).alias(c))
        val dataf = df2.agg(columns.head, columns.tail:_*)
        val t = dataf.columns.map(c => dataf.select(lit(c).alias("col_name"), col(c).alias("null_count")))
        val dataf_agg_col = t.reduce((df2, dataf) => df2.union(dataf))
        dataf_agg_col.show()

        val df3 = df2
            .withColumn("Name",lower(col("Name")))
            .withColumn("Nationality",lower(col("Nationality")))
            .withColumn("Club",lower(col("Club")))
            .withColumn("Preferred Foot",lower(col("Preferred Foot")))
            .withColumn("Position",lower(col("Position")))
            .dropDuplicates()
        df3.write.format("jdbc").option("url", misqlCon)
```

```
                .option("driver", driver).option("dbtable", "w2t1b")
                .mode("overwrite").save()
            df3.show()

            val df4 = df3
                .withColumn("Age Group", when(col("Age") < 20, "0-19")
                .when(col("Age") >= 20 && col("Age") < 30, "20-29")
                .when(col("Age") >= 30 && col("Age") < 36, "30-35")
                .otherwise("36+"))

            // val agePlayerSum = df4.groupBy("Age Group").count() // Колонки без изменения названия
            // val agePlayerSum = df4.groupBy("Age Group").agg(count("*").alias("Sum Players for Age")) // Меняет наименование
последней колонки
            val agePlayerSum = df4.groupBy("Age Group").count().toDF("Age Group Player", "Sum Players for Age") // Менет
наименование обоих колонок
            agePlayerSum.show()

            df4.write.format("jdbc").option("url", misqlCon)
                .option("driver", driver).option("dbtable", "w2t1c")
                .mode("overwrite").save()
            df4.show()

}

val s0 = (System.currentTimeMillis() - t1)/1000
val s = s0 % 60
val m = (s0/60) % 60
val h = (s0/60/60) % 24
println("%02d:%02d:%02d".format(h, m, s))
System.exit(0)
```