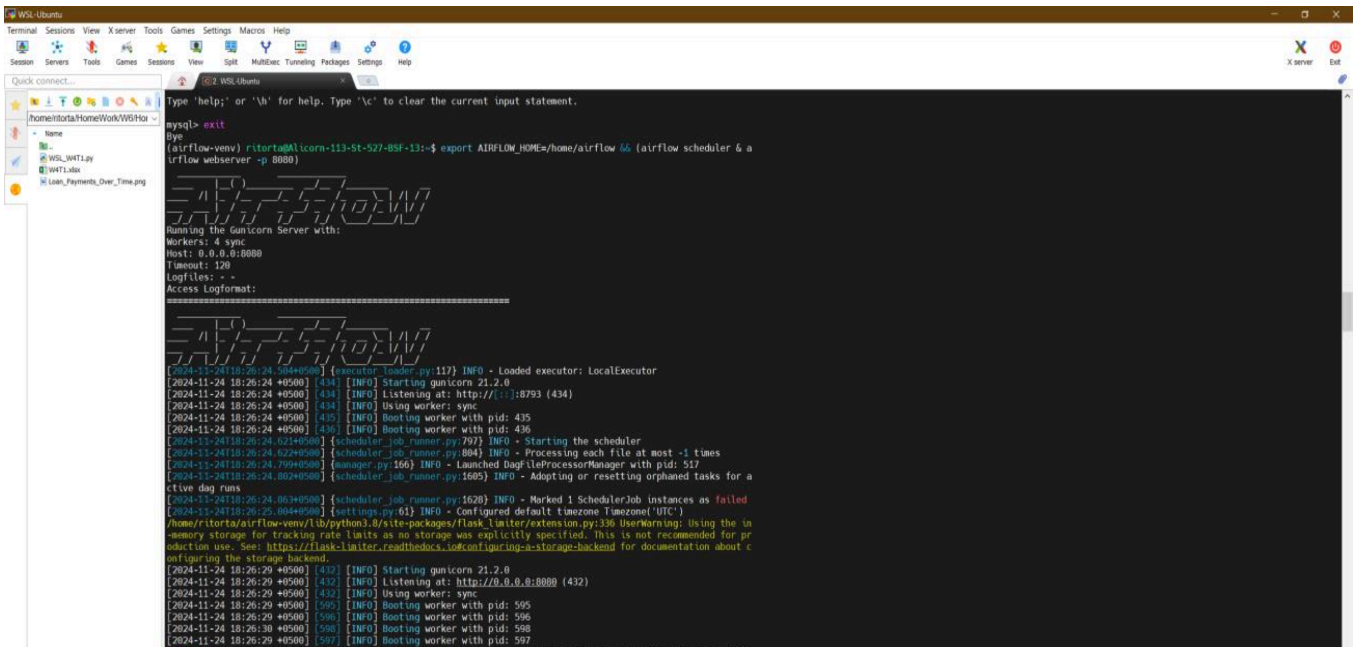
### Урок 6. Операторы в Airflow и их применение для ETL

1. Установить спарк как показано на семинаре:
- Для этого переместите папку spark в home.
- Дайте права командой chmod -R 777 ./
- nano ~/.bashrc
- export SPARK_HOME=/home/spark && export PATH=$PATH:$SPARK_HOME/bin:$SPARK_HOME/sbin
- source ~/.bashrc
- sudo apt-get install openjdk-8-jdkpip
- Указанные библиотеки нужно также установить и в виртуальную среду: python3 -m venv airflow venv && source airflow venv /bin/activate
- pip install pyspark==3.2.4
- pip install pandas==1.5.3
- pip install SQLAlchemy==1.4.46

Используйте ДЗ которые вы мне высылали для 3-4 семинара. Запустите данные задачи ПОСЛЕДОВАТЕЛЬНО, одну за другой в аирфлоу. Пришлите мне скриншоты выполненных задач в аирфлоу, логов аирфлоу, скриншоты что у вас записались таблицы в БД mysql на WSL. По возможности доработайте код чтобы изображение с линии платежей генерировалось в указанную директорию. Скриншоты соберите в pdf.


WSL-Ubuntu
Terminal Sessions View X server Tools Games Settings Macros Help
Session Servers Tools Games Sessions View Split MultiExec Tunneling Packages Settings Help
X server Exit
Quick connect...
2. WSL-Ubuntu
/home/ritorta/HomeWork/W6/Hor
Name
..
WSL_W4T1.py
W4T1.xlsx
Loan_Payments_Over_Time.png

```
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> exit
Bye
(airflow-venv) ritorta@Alicorn-113-St-527-BSF-13:~$ export AIRFLOW_HOME=/home/airflow && (airflow scheduler & a
irflow webserver -p 8080)
  ____________       _____________
 ____    |__( )_________  __/__  /________      __
____  /| |_  /__  ___/_  /_ __  /_  __ \_ | /| / /
___  ___ |  / _  /   _  __/ _  / / /_/ /_ |/ |/ /
 _/_/  |_/_/  /_/    /_/    /_/  \____/____/|__/
Running the Gunicorn Server with:
Workers: 4 sync
Host: 0.0.0.0:8080
Timeout: 120
Logfiles: - -
Access Logformat:
=================================================================
  ____________       _____________
 ____    |__( )_________  __/__  /________      __
____  /| |_  /__  ___/_  /_ __  /_  __ \_ | /| / /
___  ___ |  / _  /   _  __/ _  / / /_/ /_ |/ |/ /
 _/_/  |_/_/  /_/    /_/    /_/  \____/____/|__/
[2024-11-24T18:26:24.504+0500] {executor_loader.py:117} INFO - Loaded executor: LocalExecutor
[2024-11-24 18:26:24 +0500] [434] [INFO] Starting gunicorn 21.2.0
[2024-11-24 18:26:24 +0500] [434] [INFO] Listening at: http://[::]:8793 (434)
[2024-11-24 18:26:24 +0500] [434] [INFO] Using worker: sync
[2024-11-24 18:26:24 +0500] [435] [INFO] Booting worker with pid: 435
[2024-11-24 18:26:24 +0500] [436] [INFO] Booting worker with pid: 436
[2024-11-24T18:26:24.621+0500] {scheduler_job_runner.py:797} INFO - Starting the scheduler
[2024-11-24T18:26:24.622+0500] {scheduler_job_runner.py:804} INFO - Processing each file at most -1 times
[2024-11-24T18:26:24.799+0500] {manager.py:166} INFO - Launched DagFileProcessorManager with pid: 517
[2024-11-24T18:26:24.802+0500] {scheduler_job_runner.py:1605} INFO - Adopting or resetting orphaned tasks for a
ctive dag runs
[2024-11-24T18:26:24.863+0500] {scheduler_job_runner.py:1628} INFO - Marked 1 SchedulerJob instances as failed
[2024-11-24T18:26:25.004+0500] {settings.py:61} INFO - Configured default timezone Timezone('UTC')
/home/ritorta/airflow-venv/lib/python3.8/site-packages/flask_limiter/extension.py:336 UserWarning: Using the in
-memory storage for tracking rate limits as no storage was explicitly specified. This is not recommended for pr
oduction use. See: https://flask-limiter.readthedocs.io#configuring-a-storage-backend for documentation about c
onfiguring the storage backend.
[2024-11-24 18:26:29 +0500] [432] [INFO] Starting gunicorn 21.2.0
[2024-11-24 18:26:29 +0500] [432] [INFO] Listening at: http://0.0.0.0:8080 (432)
[2024-11-24 18:26:29 +0500] [432] [INFO] Using worker: sync
[2024-11-24 18:26:29 +0500] [595] [INFO] Booting worker with pid: 595
[2024-11-24 18:26:29 +0500] [596] [INFO] Booting worker with pid: 596
[2024-11-24 18:26:30 +0500] [598] [INFO] Booting worker with pid: 598
[2024-11-24 18:26:29 +0500] [597] [INFO] Booting worker with pid: 597
```

```python
# Файл Task1 и файлы вместе с папками Home_3 и Home_4 надо скинуть
в WSL для того, чтобы они работали.

from airflow import DAG
from airflow.operators.bash import BashOperator
from datetime import timedelta
```

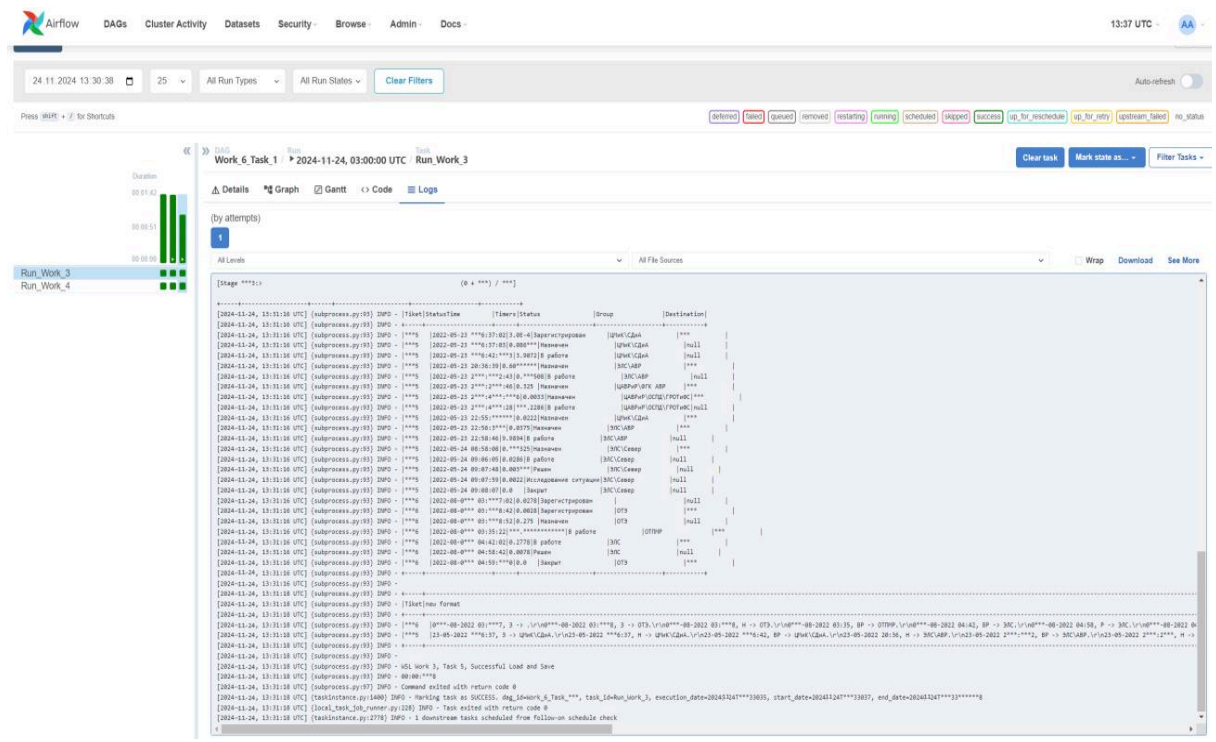
```
import pendulum

default_args = {
'owner': 'Ritorta',
'depends_on_past': False,
'start_date': pendulum.datetime(year=2024, month=4,
day=23).in_timezone('Europe/Moscow'),
'email': ['meddesu@yandex.ru'],
'email_on_failure': False,
'email_on_retry': False,
'retries': 0,
'retry_delay': timedelta(minutes=5)
}

dag1 = DAG('Work_6_Task_1',
default_args=default_args,
description="Home_Work_6",
catchup=False,
schedule_interval='0 6 * * *')

WSL_Home_3 = BashOperator(
task_id='Run_Work_3',
bash_command='export SPARK_HOME=/home/spark && export
PATH=$PATH:$SPARK_HOME/bin:$SPARK_HOME/sbin && spark-shell -i
/home/ritorta/HomeWork/W6/Home_3/WSL_W3Task_1_v2.scala',
dag=dag1)

WSL_Home_4  = BashOperator(
task_id='Run_Work_4',
bash_command='export SPARK_HOME=/home/spark && export
PATH=$PATH:$SPARK_HOME/bin:$SPARK_HOME/sbin && python3
/home/ritorta/HomeWork/W6/Home_4/WSL_W4T1.py',
dag=dag1)

WSL_Home_3 >> WSL_Home_4
```

```
/*
chcp 65001 && spark-shell -i
/home/ritorta/HomeWork/W6/WSL_W6T1dag.py --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.expressions.Window
import scala.io.Source
```

```
sc.setLogLevel("ERROR")

var sqlCoun =
s"jdbc:mysql://localhost:33061/spark?user=root&password=1"
var driver = "com.mysql.cj.jdbc.Driver"

val t1 = System.currentTimeMillis()

if(1==1){
   var df = spark.read.option("delimiter",",")
      .option("inferSchema", "true")
      .option("header", "true")
      .format("excel")
      .load("/home/ritorta/HomeWork/W6/Home_3/s3.xlsx")

   val df1 = df
      df1.write.format("jdbc").option("url", sqlCoun)
         .option("driver", driver).option("dbtable", "wsl_w3t5v2")
         .mode("overwrite").save()
   df1.show(400, truncate = false)

   val df2 = spark.read.format("jdbc").option("url", sqlCoun)
       .option("driver", driver)
       .option("dbtable", "wsl_w3t5v2")
       .load()

   val df_group = df2.distinct().where(col("fieldname") ===
"GNAME2")
       .select("objectid", "restime", "fieldvalue")
       .withColumnRenamed("fieldvalue", "Group")
       .withColumn("Destination", lit("1").cast("integer"))

   val df_status = df2.distinct().where(col("fieldname") ===
"Status")
       .select("objectid", "restime", "fieldvalue")
       .withColumnRenamed("fieldvalue", "Status")

   val df_sg = df2.filter((col("fieldname") isin ("status",
"GNAME2")))
       .select("objectid", "restime").distinct()

   val df_inner = df_sg.as("a")
       .join(df_status.as("a1"),col("a.objectid") ===
col("a1.objectid") && col("a.restime") ===
col("a1.restime"),"left")
```

```
       .join(df_group.as("a2"),col("a.objectid") ===
col("a2.objectid") && col("a.restime") ===
col("a2.restime"),"left")

.select(col("a.objectid"),col("a.restime"),col("a1.Status"),col("a
2.Group"),col("a2.Destination"))
       .withColumnRenamed("objectid", "Tiket")
       .withColumnRenamed("restime", "StatusTime")
       .distinct()

   val df_outer =
df_inner.select(col("Tiket"),col("StatusTime"),col("Status"),when(
row_number().over(Window.partitionBy(col("Tiket"))
       .orderBy(col("StatusTime"))) === 1 &&
col("Destination").isNull,"").otherwise(col("Group")).alias("Group
"),col("Destination"))

   val df_result =
df_outer.select(col("Tiket"),from_unixtime(col("StatusTime")).alia
s("StatusTime"),((lead(col("StatusTime"), 1)

.over(Window.partitionBy(col("Tiket")).orderBy(col("StatusTime")))
- col("StatusTime")) / 3600).alias("Timers"),last(col("Status"),
true)

.over(Window.partitionBy(col("Tiket")).orderBy(col("StatusTime")))
       .alias("Status"),last(col("Group"),
true).over(Window.partitionBy(col("Tiket")).orderBy(col("StatusTim
e")))
       .alias("Group"),col("Destination"))
       .withColumn("Timers", coalesce(col("Timers"), lit(0)))
       .withColumn("Timers", round(col("Timers"), 4))

   df_result.write.format("jdbc").option("url", sqlCoun)
       .option("driver", driver).option("dbtable", "wsl_w3t5v2a")
      .mode("overwrite").save()
   df_result.show(400, truncate = false)

   val df3 = spark.read.format("jdbc").option("url", sqlCoun)
       .option("driver", driver)
       .option("dbtable", "wsl_w3t5v2a")
       .load()

val df3_concat = df3.groupBy("Tiket")
   .agg(concat_ws(".\r\n", collect_list(concat_ws(",
",when(date_format(col("StatusTime"), "yyyy-MM-dd") ===
current_date(),date_format(col("StatusTime"), "yyyy-MM-dd
HH:mm:ss"))
```

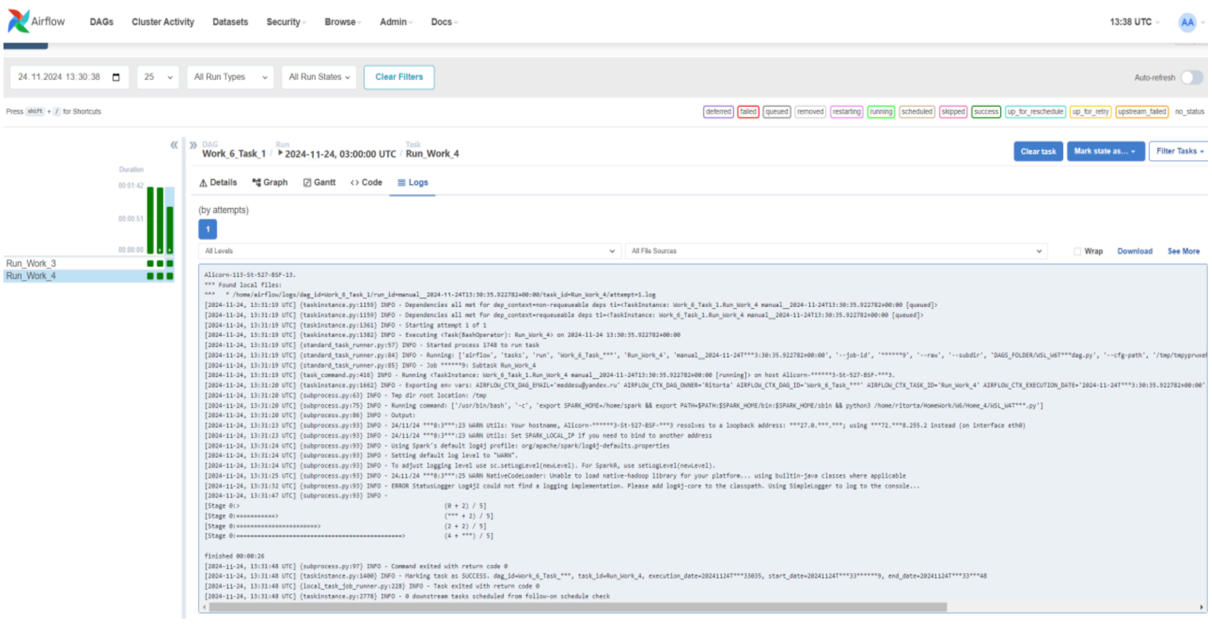
```
    .otherwise(date_format(col("StatusTime"), "dd-MM-yyyy HH:mm")),
    concat_ws(" -> ",when(col("Status") === "Зарегистрирован", "З")
    .when(col("Status") === "Назначен", "Н")
    .when(col("Status") === "В работе", "ВР")
    .when(col("Status") === "Закрыт", "ЗТ")
    .when(col("Status") === "Исследование ситуации", "ИС")
    .when(col("Status") === "Решен", "Р"),col("Group")))))
    .alias("new format"))
    .withColumn("new format", concat(col("new format"),lit(".")))
    .withColumn("Tiket",col("Tiket"))

    df3_concat.write.format("jdbc").option("url", sqlCoun)
        .option("driver", driver).option("dbtable", "wsl_w3t5v2b")
        .mode("overwrite").save()
    df3_concat.show(20, 700)

    println("WSL Work 3, Task 5, Successful Load and Save")

}

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)
```



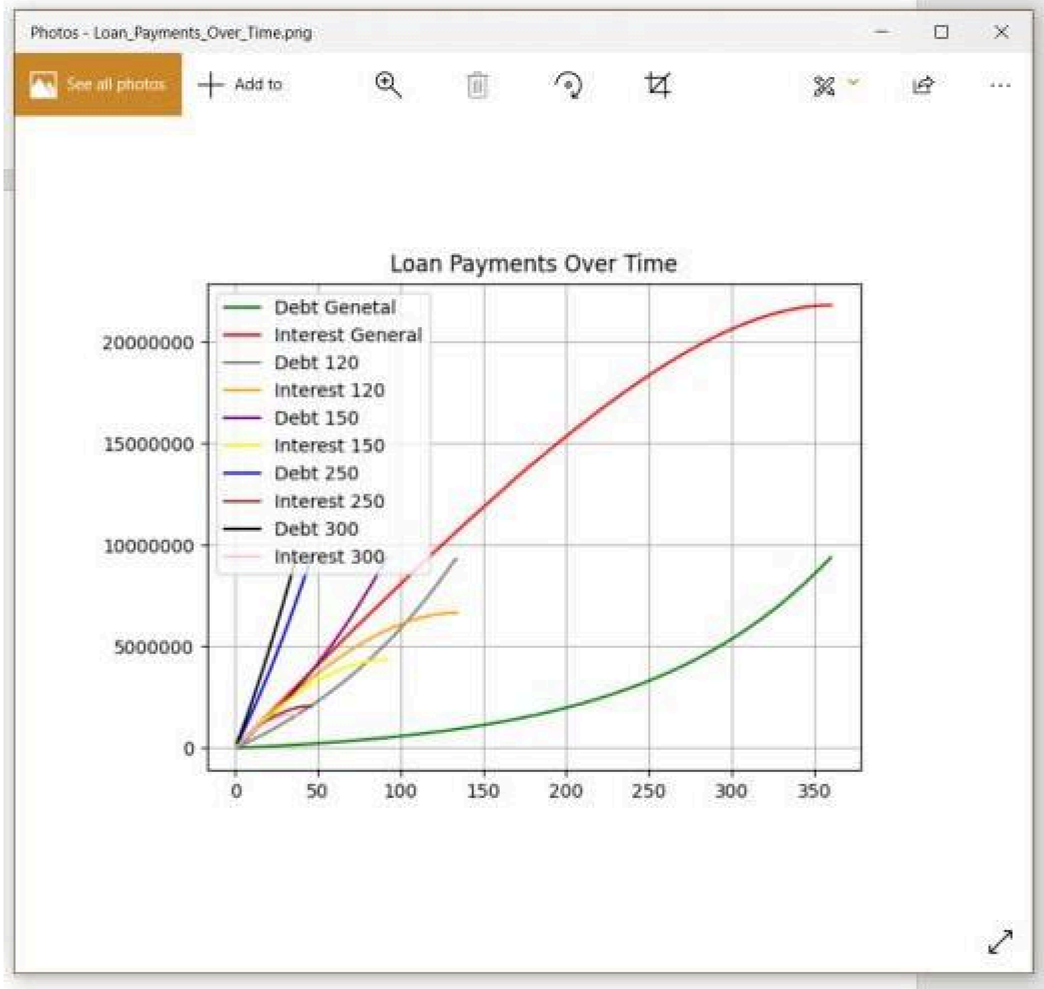
```
[Stage ***3:>                                                   (0 + ***) / ***]

+-----+--------------------+------+--------------------+--------------------+-----------+
[2024-11-24, 13:31:16 UTC] {subprocess.py:93} INFO - |Tiket|StatusTime          |Timers|Status              |Group               |Destination|
[2024-11-24, 13:31:16 UTC] {subprocess.py:93} INFO - +-----+--------------------+------+--------------------+--------------------+-----------+
[2024-11-24, 13:31:16 UTC] {subprocess.py:93} INFO - |***5   |2022-05-23 ***6:37:02|3.0E-4|Зарегистрирован      |ЦМиК\СДиА          |***        |
[2024-11-24, 13:31:16 UTC] {subprocess.py:93} INFO - |***5   |2022-05-23 ***6:37:03|0.000***|Назначен           |ЦМиК\СДиА          |null       |
[2024-11-24, 13:31:16 UTC] {subprocess.py:93} INFO - |***5   |2022-05-23 ***6:42:***3|3.9072|В работе           |ЦМиК\СДиА          |null       |
[2024-11-24, 13:31:16 UTC] {subprocess.py:93} INFO - |***5   |2022-05-23 20:36:39|0.60******|Назначен           |ЭЛС\АВР            |***         |
[2024-11-24, 13:31:16 UTC] {subprocess.py:93} INFO - |***5   |2022-05-23 2***:***2:43|0.***508|В работе           |ЭЛС\АВР            |null        |
[2024-11-24, 13:31:16 UTC] {subprocess.py:93} INFO - |***5   |2022-05-23 2***:2***:46|0.325 |Назначен           |ЦАВРиР\ОГК АВР     |***         |
[2024-11-24, 13:31:16 UTC] {subprocess.py:93} INFO - |***5   |2022-05-23 2***:4***:***6|0.0033|Назначен           |ЦАВРиР\ОСПД\ГРОТиФС|***         |
[2024-11-24, 13:31:16 UTC] {subprocess.py:93} INFO - |***5   |2022-05-23 2***:4***:28|***.2286|В работе           |ЦАВРиР\ОСПД\ГРОТиФС|null       |
[2024-11-24, 13:31:16 UTC] {subprocess.py:93} INFO - |***5   |2022-05-23 22:55:******|0.0222|Назначен           |ЦМиК\СДиА          |***         |
[2024-11-24, 13:31:16 UTC] {subprocess.py:93} INFO - |***5   |2022-05-23 22:56:3***|0.0375|Назначен           |ЭЛС\АВР            |***        |
[2024-11-24, 13:31:16 UTC] {subprocess.py:93} INFO - |***5   |2022-05-23 22:58:46|9.9894|В работе           |ЭЛС\АВР            |null       |
[2024-11-24, 13:31:16 UTC] {subprocess.py:93} INFO - |***5   |2022-05-24 08:58:08|0.***325|Назначен           |ЭЛС\Север          |***        |
[2024-11-24, 13:31:16 UTC] {subprocess.py:93} INFO - |***5   |2022-05-24 09:06:05|0.0286|В работе           |ЭЛС\Север          |null       |
[2024-11-24, 13:31:16 UTC] {subprocess.py:93} INFO - |***5   |2022-05-24 09:07:48|0.003***|Решен              |ЭЛС\Север          |null       |
[2024-11-24, 13:31:16 UTC] {subprocess.py:93} INFO - |***5   |2022-05-24 09:07:59|0.0022|Исследование ситуации|ЭЛС\Север         |null       |
[2024-11-24, 13:31:16 UTC] {subprocess.py:93} INFO - |***5   |2022-05-24 09:08:07|0.0   |Закрыт              |ЭЛС\Север          |null       |
[2024-11-24, 13:31:16 UTC] {subprocess.py:93} INFO - |***6   |2022-08-0*** 03:***7:02|0.0278|Зарегистрирован    |                    |null       |
[2024-11-24, 13:31:16 UTC] {subprocess.py:93} INFO - |***6   |2022-08-0*** 03:***8:42|0.0028|Зарегистрирован    |ОТЭ                 |***        |
[2024-11-24, 13:31:16 UTC] {subprocess.py:93} INFO - |***6   |2022-08-0*** 03:***8:52|0.275 |Назначен           |ОТЭ                 |null       |
[2024-11-24, 13:31:16 UTC] {subprocess.py:93} INFO - |***6   |2022-08-0*** 03:35:22|***.************|В работе           |ОТПМР               |***        |
[2024-11-24, 13:31:16 UTC] {subprocess.py:93} INFO - |***6   |2022-08-0*** 04:42:02|0.2778|В работе           |ЭЛС                 |***        |
[2024-11-24, 13:31:16 UTC] {subprocess.py:93} INFO - |***6   |2022-08-0*** 04:58:42|0.0078|Решен              |ЭЛС                 |null       |
[2024-11-24, 13:31:16 UTC] {subprocess.py:93} INFO - |***6   |2022-08-0*** 04:59:***0|0.0   |Закрыт              |ОТЭ                 |***        |
[2024-11-24, 13:31:16 UTC] {subprocess.py:93} INFO - +-----+--------------------+------+--------------------+--------------------+-----------+
[2024-11-24, 13:31:16 UTC] {subprocess.py:93} INFO -
[2024-11-24, 13:31:18 UTC] {subprocess.py:93} INFO - +-----+-----------------------------------------------------------------------------------------------------------------------------
[2024-11-24, 13:31:18 UTC] {subprocess.py:93} INFO - |Tiket|new format
[2024-11-24, 13:31:18 UTC] {subprocess.py:93} INFO - +-----+-----------------------------------------------------------------------------------------------------------------------------
[2024-11-24, 13:31:18 UTC] {subprocess.py:93} INFO - |***6   |0***-08-2022 03:***7, З -> .\r\n0***-08-2022 03:***8, З -> ОТЭ.\r\n0***-08-2022 03:***8, Н -> ОТЭ.\r\n0***-08-2022 03:35, ВР -> ОТПМР.\r\n0***-08-2022 04:42, ВР -> ЭЛС.\r\n0***-08-2022 04:58, Р -> ЭЛС.\r\n0***-08-2022 0
[2024-11-24, 13:31:18 UTC] {subprocess.py:93} INFO - |***5   |23-05-2022 ***6:37, З -> ЦМиК\СДиА.\r\n23-05-2022 ***6:37, Н -> ЦМиК\СДиА.\r\n23-05-2022 ***6:42, ВР -> ЦМиК\СДиА.\r\n23-05-2022 20:36, Н -> ЭЛС\АВР.\r\n23-05-2022 2***:***2, ВР -> ЭЛС\АВР.\r\n23-05-2022 2***:2***, Н ->
[2024-11-24, 13:31:18 UTC] {subprocess.py:93} INFO - +-----+-----------------------------------------------------------------------------------------------------------------------------
[2024-11-24, 13:31:18 UTC] {subprocess.py:93} INFO -
[2024-11-24, 13:31:18 UTC] {subprocess.py:93} INFO - WSL Work 3, Task 5, Successful Load and Save
[2024-11-24, 13:31:18 UTC] {subprocess.py:93} INFO - 00:00:***8
[2024-11-24, 13:31:18 UTC] {subprocess.py:97} INFO - Command exited with return code 0
[2024-11-24, 13:31:18 UTC] {taskinstance.py:1400} INFO - Marking task as SUCCESS. dag_id=Work_6_Task_***, task_id=Run_Work_3, execution_date=20241124T***33035, start_date=20241124T***33037, end_date=20241124T***33******8
[2024-11-24, 13:31:18 UTC] {local_task_job_runner.py:228} INFO - Task exited with return code 0
[2024-11-24, 13:31:18 UTC] {taskinstance.py:2778} INFO - 1 downstream tasks scheduled from follow-on schedule check
```

```
import time, sys, os
from pyspark.sql.session import SparkSession
from pyspark.sql.functions import col, lit
import matplotlib.pyplot as plt
from sqlalchemy import create_engine
from pandas.io import sql
import warnings

warnings.filterwarnings("ignore")
t0=time.time()
con=create_engine("mysql://root:1@localhost:33061/spark")
os.environ['PYSPARK_PYTHON'] = sys.executable
os.environ['PYSPARK_DRIVER_PYTHON'] = sys.executable
spark=SparkSession.builder.appName("WSL Home Work
№4").getOrCreate()

sql.execute("""drop table if exists spark.`WSL_W6T1`""",con)
sql.execute("""CREATE TABLE if not exists spark.`WSL_W6T1` (
    `number` INT(10) NULL DEFAULT NULL,
    `Month` DATE NULL DEFAULT NULL,
    `Payment amount` FLOAT NULL DEFAULT NULL,
    `Payment of the principal debt` FLOAT NULL DEFAULT NULL,
    `Payment of interest` FLOAT NULL DEFAULT NULL,
    `Balance of debt` FLOAT NULL DEFAULT NULL,
    `interest` FLOAT NULL DEFAULT NULL,
    `debt` FLOAT NULL DEFAULT NULL
)
COLLATE='utf8mb4_general_ci'
ENGINE=InnoDB""",con)

from pyspark.sql.window import Window
from pyspark.sql.functions import sum as sum1
w =
Window.partitionBy(lit(1)).orderBy("number").rowsBetween(Window.un
boundedPreceding, Window.currentRow)
dfG = spark.read.format("com.crealytics.spark.excel")\
        .option("dataAddress", "'General'!A1:F361")\
        .option("useHeader", "false")\
        .option("treatEmptyValuesAsNulls", "false")\
        .option("inferSchema", "true").option("addColorColumns",
"true")\
    .option("usePlainNumberFormat","true")\
        .option("startColumn", 0)\
        .option("endColumn", 99)\
        .option("timestampFormat", "MM-dd-yyyy HH:mm:ss")\
        .option("maxRowsInMemory", 20)\
        .option("excerptSize", 10)\
        .option("header", "true")\
        .format("excel")\

.load("/home/ritorta/HomeWork/W6/Home_4/WSL_W6T1.xlsx").limit(1000
)\
        .withColumn("interest", sum1(col("Payment of
interest")).over(w))\
        .withColumn("debt", sum1(col("Payment of the principal
debt")).over(w))

df120 = spark.read.format("com.crealytics.spark.excel")\
        .option("dataAddress", "'120'!A1:F135")\
        .option("useHeader", "false")\
        .option("treatEmptyValuesAsNulls", "false")\
        .option("inferSchema", "true").option("addColorColumns",
"true")\
    .option("usePlainNumberFormat","true")\
        .option("startColumn", 0)\
        .option("endColumn", 99)\
        .option("timestampFormat", "MM-dd-yyyy HH:mm:ss")\
        .option("maxRowsInMemory", 20)\
        .option("excerptSize", 10)\
        .option("header", "true")\
        .format("excel")\

.load("/home/ritorta/HomeWork/W6/Home_4/WSL_W6T1.xlsx").limit(1000
)\
        .withColumn("interest", sum1(col("Payment of
interest")).over(w))\
        .withColumn("debt", sum1(col("Payment of the principal
debt")).over(w))

df150 = spark.read.format("com.crealytics.spark.excel")\
        .option("dataAddress", "'150'!A1:F93")\
        .option("useHeader", "false")\
        .option("treatEmptyValuesAsNulls", "false")\
        .option("inferSchema", "true").option("addColorColumns",
"true")\
    .option("usePlainNumberFormat","true")\
        .option("startColumn", 0)\
        .option("endColumn", 99)\
        .option("timestampFormat", "MM-dd-yyyy HH:mm:ss")\
        .option("maxRowsInMemory", 20)\
        .option("excerptSize", 10)\
        .option("header", "true")\
        .format("excel")\
 .load("/home/ritorta/HomeWork/W6/Home_4/WSL_W6T1.xlsx").limit(1000
 )\
        .withColumn("interest", sum1(col("Payment of
 interest")).over(w))\
        .withColumn("debt", sum1(col("Payment of the principal
 debt")).over(w))

 df250 = spark.read.format("com.crealytics.spark.excel")\
        .option("dataAddress", "'250'!A1:F47")\
        .option("useHeader", "false")\
        .option("treatEmptyValuesAsNulls", "false")\
        .option("inferSchema", "true").option("addColorColumns",
 "true")\
    .option("usePlainNumberFormat","true")\
        .option("startColumn", 0)\
        .option("endColumn", 99)\
        .option("timestampFormat", "MM-dd-yyyy HH:mm:ss")\
        .option("maxRowsInMemory", 20)\
        .option("excerptSize", 10)\
        .option("header", "true")\
        .format("excel")\

 .load("/home/ritorta/HomeWork/W6/Home_4/WSL_W6T1.xlsx").limit(1000
 )\
        .withColumn("interest", sum1(col("Payment of
 interest")).over(w))\
        .withColumn("debt", sum1(col("Payment of the principal
 debt")).over(w))

 df300 = spark.read.format("com.crealytics.spark.excel")\
        .option("dataAddress", "'300'!A1:F38")\
        .option("useHeader", "false")\
        .option("treatEmptyValuesAsNulls", "false")\
        .option("inferSchema", "true").option("addColorColumns",
 "true")\
    .option("usePlainNumberFormat","true")\
        .option("startColumn", 0)\
        .option("endColumn", 99)\
        .option("timestampFormat", "MM-dd-yyyy HH:mm:ss")\
        .option("maxRowsInMemory", 20)\
        .option("excerptSize", 10)\
        .option("header", "true")\
        .format("excel")\

 .load("/home/ritorta/HomeWork/W6/Home_4/WSL_W6T1.xlsx").limit(1000
 )\
        .withColumn("interest", sum1(col("Payment of 
 interest")).over(w))\
        .withColumn("debt", sum1(col("Payment of the principal 
 debt")).over(w))

 df_combined = 
 dfG.union(df120).union(df150).union(df250).union(df300)

 df_combined.write.format("jdbc").option("url","jdbc:mysql://localh
 ost:33061/spark?user=root&password=1")\
        .option("driver", 
 "com.mysql.cj.jdbc.Driver").option("dbtable", "WSL_W6T1")\
        .mode("append").save()

 """df_pandas = df_combined.toPandas()"""

 df_pandas1 = dfG.toPandas()
 df_pandas2 = df120.toPandas()
 df_pandas3 = df150.toPandas()
 df_pandas4 = df250.toPandas()
 df_pandas5 = df300.toPandas()

 ax = plt.gca()
 ax.ticklabel_format(style='plain')

 df_pandas1.plot(kind='line', x='number', y='debt', color='green', 
 ax=ax, label='Debt Genetal')
 df_pandas1.plot(kind='line', x='number', y='interest', 
 color='red', ax=ax, label='Interest General')
 df_pandas2.plot(kind='line', x='number', y='debt', color='grey', 
 ax=ax, label='Debt 120')
 df_pandas2.plot(kind='line', x='number', y='interest', 
 color='orange', ax=ax, label='Interest 120')
 df_pandas3.plot(kind='line', x='number', y='debt', color='purple', 
 ax=ax, label='Debt 150')
 df_pandas3.plot(kind='line', x='number', y='interest', 
 color='yellow', ax=ax, label='Interest 150')
 df_pandas4.plot(kind='line', x='number', y='debt', color='blue', 
 ax=ax, label='Debt 250')
 df_pandas4.plot(kind='line', x='number', y='interest', 
 color='brown', ax=ax, label='Interest 250')
 df_pandas5.plot(kind='line', x='number', y='debt', color='black', 
 ax=ax, label='Debt 300')
 df_pandas5.plot(kind='line', x='number', y='interest', 
 color='pink', ax=ax, label='Interest 300')

 plt.title('Loan Payments Over Time')
 plt.grid ( True )
```

```
ax.set(xlabel=None)

plot_directory = "/home/ritorta/HomeWork/W6/Home_4/"
plot_filename = "Loan_Payments_Over_Time.png"
plt.savefig(plot_directory + plot_filename)

plt.show()
spark.stop()
t1=time.time()
print('finished',time.strftime('%H:%M:%S',time.gmtime(round(t1-t0)
)))
```



```
Alicorn-113-St-527-BSF-13.
*** Found local files:
***   * /home/airflow/logs/dag_id=Work_6_Task_1/run_id=manual__2024-11-24T13:30:35.922782+00:00/task_id=Run_Work_4/attempt=1.log
[2024-11-24, 13:31:19 UTC] {taskinstance.py:1159} INFO - Dependencies all met for dep_context=non-requeueable deps ti=<TaskInstance: Work_6_Task_1.Run_Work_4 manual__2024-11-24T13:30:35.922782+00:00 [queued]>
[2024-11-24, 13:31:19 UTC] {taskinstance.py:1159} INFO - Dependencies all met for dep_context=requeueable deps ti=<TaskInstance: Work_6_Task_1.Run_Work_4 manual__2024-11-24T13:30:35.922782+00:00 [queued]>
[2024-11-24, 13:31:19 UTC] {taskinstance.py:1361} INFO - Starting attempt 1 of 1
[2024-11-24, 13:31:19 UTC] {taskinstance.py:1382} INFO - Executing <Task(BashOperator): Run_Work_4> on 2024-11-24 13:30:35.922782+00:00
[2024-11-24, 13:31:19 UTC] {standard_task_runner.py:57} INFO - Started process 1748 to run task
[2024-11-24, 13:31:19 UTC] {standard_task_runner.py:84} INFO - Running: ['airflow', 'tasks', 'run', 'Work_6_Task_***', 'Run_Work_4', 'manual__2024-11-24T***3:30:35.922782+00:00', '--job-id', '******9', '--raw', '--subdir', 'DAGS_FOLDER/WSL_W6T***dag.py', '--cfg-path', '/tmp/tmpyprwxel
[2024-11-24, 13:31:19 UTC] {standard_task_runner.py:85} INFO - Job ******9: Subtask Run_Work_4
[2024-11-24, 13:31:19 UTC] {task_command.py:416} INFO - Running <TaskInstance: Work_6_Task_1.Run_Work_4 manual__2024-11-24T13:30:35.922782+00:00 [running]> on host Alicorn-******3-St-527-BSF-***3.
[2024-11-24, 13:31:20 UTC] {taskinstance.py:1662} INFO - Exporting env vars: AIRFLOW_CTX_DAG_EMAIL='meddesu@yandex.ru' AIRFLOW_CTX_DAG_OWNER='Ritorta' AIRFLOW_CTX_DAG_ID='Work_6_Task_***' AIRFLOW_CTX_TASK_ID='Run_Work_4' AIRFLOW_CTX_EXECUTION_DATE='2024-11-24T***3:30:35.922782+00:00'
[2024-11-24, 13:31:20 UTC] {subprocess.py:63} INFO - Tmp dir root location: /tmp
[2024-11-24, 13:31:20 UTC] {subprocess.py:75} INFO - Running command: ['/usr/bin/bash', '-c', 'export SPARK_HOME=/home/spark && export PATH=$PATH:$SPARK_HOME/bin:$SPARK_HOME/sbin && python3 /home/ritorta/HomeWork/W6/Home_4/WSL_W4T***.py']
[2024-11-24, 13:31:20 UTC] {subprocess.py:86} INFO - Output:
[2024-11-24, 13:31:23 UTC] {subprocess.py:93} INFO - 24/11/24 ***8:3***:23 WARN Utils: Your hostname, Alicorn-******3-St-527-BSF-***3 resolves to a loopback address: ***27.0.***.***; using ***72.***8.255.2 instead (on interface eth0)
[2024-11-24, 13:31:23 UTC] {subprocess.py:93} INFO - 24/11/24 ***8:3***:23 WARN Utils: Set SPARK_LOCAL_IP if you need to bind to another address
[2024-11-24, 13:31:24 UTC] {subprocess.py:93} INFO - Using Spark's default log4j profile: org/apache/spark/log4j-defaults.properties
[2024-11-24, 13:31:24 UTC] {subprocess.py:93} INFO - Setting default log level to "WARN".
[2024-11-24, 13:31:24 UTC] {subprocess.py:93} INFO - To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
[2024-11-24, 13:31:25 UTC] {subprocess.py:93} INFO - 24/11/24 ***8:3***:25 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
[2024-11-24, 13:31:32 UTC] {subprocess.py:93} INFO - ERROR StatusLogger Log4j2 could not find a logging implementation. Please add log4j-core to the classpath. Using SimpleLogger to log to the console...
[2024-11-24, 13:31:47 UTC] {subprocess.py:93} INFO -
[Stage 0:>                                                          (0 + 2) / 5]
[Stage 0:===========>                                               (*** + 2) / 5]
[Stage 0:=======================>                                   (2 + 2) / 5]
[Stage 0:===============================================>           (4 + ***) / 5]

finished 00:00:26
[2024-11-24, 13:31:48 UTC] {subprocess.py:97} INFO - Command exited with return code 0
[2024-11-24, 13:31:48 UTC] {taskinstance.py:1400} INFO - Marking task as SUCCESS. dag_id=Work_6_Task_***, task_id=Run_Work_4, execution_date=20241124T***33035, start_date=20241124T***33******9, end_date=20241124T***33***48
[2024-11-24, 13:31:48 UTC] {local_task_job_runner.py:228} INFO - Task exited with return code 0
[2024-11-24, 13:31:48 UTC] {taskinstance.py:2778} INFO - 0 downstream tasks scheduled from follow-on schedule check
```


Photos - Loan_Payments_Over_Time.png
See all photos
Add to
Loan Payments Over Time
Debt Genetal
Interest General
Debt 120
Interest 120
Debt 150
Interest 150
Debt 250
Interest 250
Debt 300
Interest 300
20000000
15000000
10000000
5000000
0
0
50
100
150
200
250
300
350