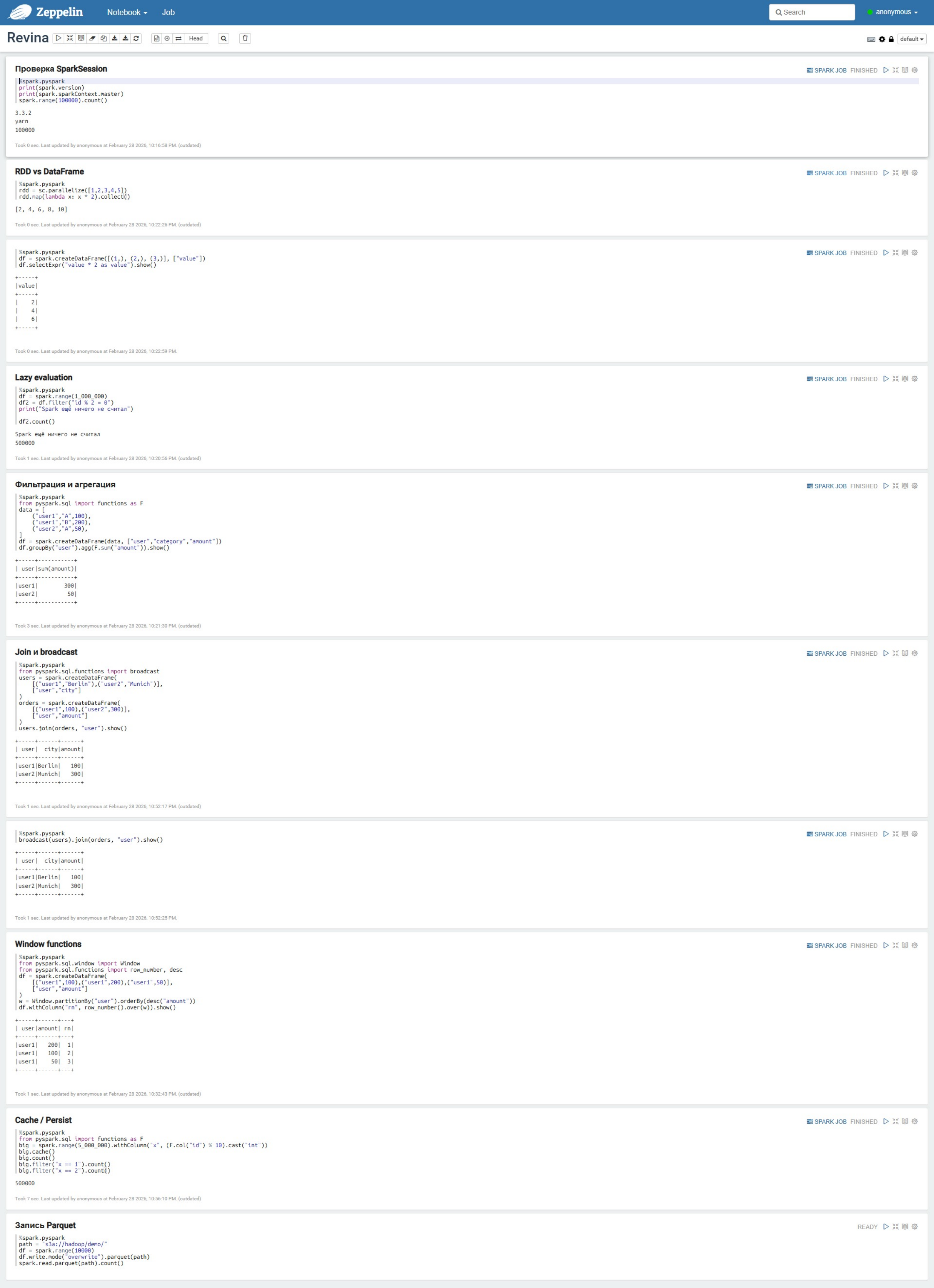

Revina

Проверка SparkSession

```
%spark.pyspark
print(spark.version)
print(spark.sparkContext.master)
spark.range(100000).count()
```

```
3.3.2
yarn
100000
```

Took 0 sec. Last updated by anonymous at February 28 2026, 10:16:58 PM. (outdated)

RDD vs DataFrame

```
%spark.pyspark
rdd = sc.parallelize([1,2,3,4,5])
rdd.map(lambda x: x * 2).collect()
```

```
[2, 4, 6, 8, 10]
```

Took 0 sec. Last updated by anonymous at February 28 2026, 10:22:26 PM. (outdated)

```
%spark.pyspark
df = spark.createDataFrame([(1,), (2,), (3,)], ["value"])
df.selectExpr("value * 2 as value").show()
```

```
+-----+
|value|
+-----+
|    2|
|    4|
|    6|
+-----+
```

Took 0 sec. Last updated by anonymous at February 28 2026, 10:22:59 PM.

Lazy evaluation

```
%spark.pyspark
df = spark.range(1_000_000)
df2 = df.filter("id % 2 = 0")
print("Spark ещё ничего не считал")

df2.count()
```

```
Spark ещё ничего не считал
500000
```

Took 1 sec. Last updated by anonymous at February 28 2026, 10:20:56 PM. (outdated)

Фильтрация и агрегация

```
%spark.pyspark
from pyspark.sql import functions as F
data = [
    ("user1","A",100),
    ("user1","B",200),
    ("user2","A",50),
]
df = spark.createDataFrame(data, ["user","category","amount"])
df.groupBy("user").agg(F.sum("amount")).show()
```

```
+-----+-----------+
| user|sum(amount)|
+-----+-----------+
|user1|        300|
|user2|         50|
+-----+-----------+
```

Took 3 sec. Last updated by anonymous at February 28 2026, 10:21:30 PM. (outdated)

Join и broadcast

```
%spark.pyspark
from pyspark.sql.functions import broadcast
users = spark.createDataFrame(
    [("user1","Berlin"),("user2","Munich")],
    ["user","city"]
)
orders = spark.createDataFrame(
    [("user1",100),("user2",300)],
    ["user","amount"]
)
users.join(orders, "user").show()
```

```
+-----+------+------+
| user|  city|amount|
+-----+------+------+
|user1|Berlin|   100|
|user2|Munich|   300|
+-----+------+------+
```

Took 1 sec. Last updated by anonymous at February 28 2026, 10:52:17 PM. (outdated)

```
%spark.pyspark
broadcast(users).join(orders, "user").show()
```

```
+-----+------+------+
| user|  city|amount|
+-----+------+------+
|user1|Berlin|   100|
|user2|Munich|   300|
+-----+------+------+
```

Took 1 sec. Last updated by anonymous at February 28 2026, 10:52:25 PM.

Window functions

```
%spark.pyspark
from pyspark.sql.window import Window
from pyspark.sql.functions import row_number, desc
df = spark.createDataFrame(
    [("user1",100),("user1",200),("user1",50)],
    ["user","amount"]
)
w = Window.partitionBy("user").orderBy(desc("amount"))
df.withColumn("rn", row_number().over(w)).show()
```

```
+-----+------+---+
| user|amount| rn|
+-----+------+---+
|user1|   200|  1|
|user1|   100|  2|
|user1|    50|  3|
+-----+------+---+
```

Took 1 sec. Last updated by anonymous at February 28 2026, 10:32:43 PM. (outdated)

Cache / Persist

```
%spark.pyspark
from pyspark.sql import functions as F
big = spark.range(5_000_000).withColumn("x", (F.col("id") % 10).cast("int"))
big.cache()
big.count()
big.filter("x == 1").count()
big.filter("x == 2").count()
```

```
500000
```

Took 7 sec. Last updated by anonymous at February 28 2026, 10:56:10 PM. (outdated)

Запись Parquet

```
%spark.pyspark
path = "s3a://hadoop/demo/"
df = spark.range(10000)
df.write.mode("overwrite").parquet(path)
spark.read.parquet(path).count()
```