МИНОБРНАУКИ РОССИИ
САНКТПЕТЕРБУРГ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра информационной безопасности

ОТЧЕТ

по лабораторной работе №1
по дисциплине «Распределенные системы обработки данных»
Тема: Сравнение Spark API и Spark SQL

Студентка гр. 1361 _____ Токарева У.В.
Преподаватель _____ Троценко В.В.

Санкт-Петербург
2024

ЦЕЛЬ РАБОТЫ

Необходимо запустить и изучить готовое SparkSQL решение. Затем завершить начатое SparkAPI решение.

ИСПОЛЬЗОВАННЫЕ ФУНКЦИИ

1. `F.avg()` – Расчет средних значений kills, deaths, assists и gold.

   Использовано: Создание kda (kills, deaths, assists) для каждого игрока.

2. `F.floor()` – Округление средних значений kda и gold.

   Использовано: Обеспечение точности данных для того, чтобы избежать нецелых значений в выводе.

3. `F.when()`, `F.then()`, `F.otherwise()` – Вычисление winrate неосновной победы.

   Использовано: Приведение процента победы к общему количеству сыгранных матчей.

4. `F.count()` – Подсчет общего количества матчей для каждого игрока.

   Использовано: Определение рейтинга игрока на основе его опыта.

5. `F.max()` – Нахождение максимального счета позиций для каждого игрока.

   Использовано: Выбор лучших позиций для каждого игрока.

6. `F.min()` – Нахождение минимальной позиции для игроков.

   Использовано: Выбор наиболее эффективной позиции для игрока.

7. `F.row_number()` – Присвоение рангов игрокам на основе их статистики.

   Использовано: Сортировка игроков по их производительности и названию.

8. `F.lit()` – Разделение данных по группам.

   Использовано: Группировка игроков.

9. `F.desc()` – Сортировка результатов по убыванию количества матчей.

   Использовано: Представление, в первую очередь, игроков с наибольшим опытом.

10. `F.asc()` – Сортировка результатов по возрастанию имени.

    Использовано: Стандартизация порядка вывода игроков.

11. `F.format_string():` – Форматирование вывода kda, gold и winrate.

    Использовано: Представление данных в удобном для чтения виде.

ПРИЛОЖЕНИЕ 1. ИСХОДНЫЙ КОД ПРОГРАММЫ

```
from pyspark.sql import DataFrame, Window
from pyspark.sql import functions as F
from common import read_csv


### Работа студентки группы 1361 - Токаревой Ульяны Владимировны


def solve() -> DataFrame:
match = read_csv('match')
player = read_csv('player')
player_result = read_csv('player_result')

kda = player_result.groupBy('player_id').agg(
F.floor(F.avg('kill')).alias("avg_kills"),
F.floor(F.avg('death')).alias("avg_death"),
F.floor(F.avg('assist')).alias("avg_assists"),
F.floor(F.avg('gold') / 1000).alias("avg_gold_k")
)

winrate = player_result.join(match, "match_id",
how="left").groupBy("player_id").agg(
F.floor(F.avg(F.when(player_result["is_radiant"] ==
match["radiant_won"], 1).otherwise(0)) * 100).alias("winrate"),
F.count('player_id').alias("matches_cnt")
)

pos_cnt = player_result.groupBy("player_id",
"pos").agg(F.count("pos").alias("pos_cnt"))

player_window = Window.partitionBy("player_id")
maxpos = pos_cnt.withColumn(
"max_pos_cnt", F.max("pos_cnt").over(player_window)
).filter(
```

```
F.col("max_pos_cnt") == F.col("pos_cnt")
).groupBy("player_id").agg(F.min("pos").alias("pos"))

res = player.join(kda, on="player_id") \
.join(winrate, on="player_id") \
.join(maxpos, on="player_id") \
.select(
F.row_number().over(Window.partitionBy(F.lit("Костыль от
warning`ов про
партиции")).orderBy(F.desc("matches_cnt"),F.asc("name"))).alias(
"N"),
player.name,
F.col("pos"),
F.format_string("%d/%d/%d", kda.avg_kills, kda.avg_death,
kda.avg_assists).alias("kda"),
F.format_string("%.0fk",
F.col("avg_gold_k").cast("float")).alias("avg_gold"),
F.format_string("%d%%/%d", F.col("winrate").cast("int"),
winrate.matches_cnt).alias("winrate")
) \
.sort("N", ascending=False)

return res
```