P146216_STQD6324_Assignment3   ▷ ✕ 📖 ✎ ⎘ ⬆   📄 ⊖ Head   🗑   ⌨ ⚙ 🔒 default ▼

```
//function to parse the data file into HDFS

hadoop fs -put ml-100k/u.user /user/maria_dev/ml-100k/u.user
hadoop fs -put ml-100k/u.data /user/maria_dev/ml-100k/u.data
hadoop fs -put ml-100k/u.item /user/maria_dev/ml-100k/u.item
```
READY ▷ ✕ 📖 ⚙

```
%spark
import org.apache.spark.sql.SparkSession
import org.apache.spark.sql.functions._

val spark = SparkSession.builder().appName("MovieLens100k").getOrCreate()
spark.sparkContext.setLogLevel("WARN")

import org.apache.spark.sql.SparkSession
import org.apache.spark.sql.functions._
spark: org.apache.spark.sql.SparkSession = org.apache.spark.sql.SparkSession@24409154
```
FINISHED ▷ ✕ 📖 ⚙

Took 3 sec. Last updated by anonymous at July 23 2025, 9:40:07 PM. (outdated)

```
%spark
val userRDD = spark.sparkContext.textFile("hdfs:///user/maria_dev/ml-100k/u.user")

val usersDF = userRDD.map(_.split("\\|")).map(f =>
  (f(0).toInt, f(1).toInt, f(2), f(3), f(4))
).toDF("user_id", "age", "gender", "occupation", "zip")

usersDF.createOrReplaceTempView("users")

usersDF.show(5)

userRDD: org.apache.spark.rdd.RDD[String] = hdfs:///user/maria_dev/ml-100k/u.user MapPartitionsRDD[9] at textFile at <console>:33
usersDF: org.apache.spark.sql.DataFrame = [user_id: int, age: int ... 3 more fields]
+-------+---+------+----------+-----+
|user_id|age|gender|occupation|  zip|
+-------+---+------+----------+-----+
|      1| 24|     M|technician|85711|
|      2| 53|     F|     other|94043|
|      3| 23|     M|    writer|32067|
|      4| 24|     M|technician|43537|
|      5| 33|     F|     other|15213|
+-------+---+------+----------+-----+
only showing top 5 rows
```
FINISHED ▷ ✕ 📖 ⚙

Took 20 sec. Last updated by anonymous at July 23 2025, 9:41:08 PM. (outdated)

```
%spark
val ratingsRDD = spark.sparkContext.textFile("hdfs:///user/maria_dev/ml-100k/u.data")

val ratingsDF = ratingsRDD.map(_.split("\t")).map(f =>
  (f(0).toInt, f(1).toInt, f(2).toInt)
).toDF("user_id", "movie_id", "rating")

ratingsDF.createOrReplaceTempView("ratings")

ratingsDF.show(5)

ratingsRDD: org.apache.spark.rdd.RDD[String] = hdfs:///user/maria_dev/ml-100k/u.data MapPartitionsRDD[17] at textFile at <console>:33
ratingsDF: org.apache.spark.sql.DataFrame = [user_id: int, movie_id: int ... 1 more field]
+-------+--------+------+
|user_id|movie_id|rating|
+-------+--------+------+
|    196|     242|     3|
|    186|     302|     3|
|     22|     377|     1|
|    244|      51|     2|
|    166|     346|     1|
+-------+--------+------+
only showing top 5 rows
```
FINISHED ▷ ✕ 📖 ⚙

Took 3 sec. Last updated by anonymous at July 23 2025, 9:41:23 PM. (outdated)

```
%spark
val moviesRDD = spark.sparkContext.textFile("hdfs:///user/maria_dev/ml-100k/u.item")

val moviesDF = moviesRDD.map(_.split("\\|")).map(f =>
  (f(0).toInt, f(1))
).toDF("movie_id", "title")

moviesDF.createOrReplaceTempView("movies")

moviesDF.show(5)

moviesRDD: org.apache.spark.rdd.RDD[String] = hdfs:///user/maria_dev/ml-100k/u.item MapPartitionsRDD[25] at textFile at <console>:33
moviesDF: org.apache.spark.sql.DataFrame = [movie_id: int, title: string]
+--------+-----------------+
|movie_id|            title|
+--------+-----------------+
|       1| Toy Story (1995)|
|       2| GoldenEye (1995)|
|       3|Four Rooms (1995)|
|       4|Get Shorty (1995)|
|       5|   Copycat (1995)|
+--------+-----------------+
only showing top 5 rows
```
FINISHED ▷ ✕ 📖 ⚙

Took 2 sec. Last updated by anonymous at July 23 2025, 9:41:48 PM. (outdated)

```
%sql
--Calculate the average rating for each movie

SELECT movie_id, ROUND(AVG(rating),2) AS avg_rating
FROM ratings
GROUP BY movie_id
ORDER BY movie_id
LIMIT 10
```
FINISHED ▷ ✕ 📖 ⚙

⊞ 📊 🥧 📈 📉 📈 📉   ⬇ ▼

movie_id	avg_rating
1	3.88
2	3.21
3	3.03
4	3.55
5	3.3
6	3.58
7	3.8
8	4.0
9	3.9
10	3.83

Took 3 sec. Last updated by anonymous at July 23 2025, 10:16:36 PM.

```
%sql
--Identify the top ten movies with the highest average ratings
SELECT m.title, ROUND(AVG(r.rating),2) AS avg_rating
```
FINISHED ▷ ✕ 📖 ⚙

```sql
SELECT m.title, ROUND(AVG(r.rating),2) AS avg_rating
FROM ratings r
JOIN movies m ON r.movie_id = m.movie_id
GROUP BY m.title
ORDER BY avg_rating DESC
LIMIT 10
```

title	avg_rating
Santa with Muscles (1996)	5.0
Entertaining Angels: The Dorothy Day Story (1996)	5.0
They Made Me a Criminal (1939)	5.0
Star Kid (1997)	5.0
Great Day in Harlem, A (1994)	5.0
Saint of Fort Washington, The (1993)	5.0
Someone Else's America (1995)	5.0
Marlene Dietrich: Shadow and Light (1996)	5.0
Aiqing wansui (1994)	5.0
Prefontaine (1997)	5.0

Took 8 sec. Last updated by anonymous at July 23 2025, 10:16:47 PM.

```sql
%sql

--Find the user who have rated at least 50 movies and identify their favourite movie

WITH active_users AS (
  SELECT user_id, COUNT(*) AS num_ratings
  FROM ratings
  GROUP BY user_id
  HAVING num_ratings >= 50
),
active_users_top10 AS (
  SELECT user_id
  FROM active_users
  ORDER BY num_ratings DESC
  LIMIT 10
),
user_fav_movies AS (
  SELECT r.user_id, r.movie_id, r.rating
  FROM ratings r
  JOIN active_users_top10 a ON r.user_id = a.user_id
  WHERE r.rating = 5
),
user_fav_movies_with_title AS (
  SELECT ufm.user_id, m.title
  FROM user_fav_movies ufm
  JOIN movies m ON ufm.movie_id = m.movie_id
),
ranked_movies AS (
  SELECT *,
         ROW_NUMBER() OVER (PARTITION BY user_id ORDER BY title) AS rn
  FROM user_fav_movies_with_title
)
SELECT user_id, title AS favourite_movie
FROM ranked_movies
WHERE rn = 1
ORDER BY user_id
```

user_id	favourite_movie
13	2001: A Space Odyssey (1968)
234	12 Angry Men (1957)
276	2001: A Space Odyssey (1968)
303	Ace Ventura: Pet Detective (1994)
393	Apt Pupil (1998)
405	2001: A Space Odyssey (1968)
416	12 Angry Men (1957)
450	Age of Innocence, The (1993)
537	2001: A Space Odyssey (1968)
655	Angel Baby (1995)

Took 9 sec. Last updated by anonymous at July 23 2025, 10:16:55 PM.

```sql
%sql

--Find all the users who are less than 20 years old

SELECT *
FROM users
WHERE age < 20
LIMIT 10
```

user_id	age	gender	occupation	zip
30	7	M	student	55436
36	19	F	student	93117
52	18	F	student	55105
57	16	M	none	84010
67	17	M	student	60402
68	19	M	student	22904
101	15	M	student	05146
110	19	M	student	77840
142	13	M	other	48118
179	15	M	entertainment	20755

Took 4 sec. Last updated by anonymous at July 23 2025, 10:16:55 PM.

```sql
%sql

--Find all the users whose occupation is scientist and whose age is between 30 and 40 years old

SELECT *
FROM users
WHERE occupation = 'scientist' AND age BETWEEN 30 AND 40
LIMIT 10
```

user_id	age	gender	occupation	zip
40	38	M	scientist	27514
71	39	M	scientist	98034

74	39	M	scientist	T8H1N
107	39	M	scientist	60466
183	33	M	scientist	27708
272	33	M	scientist	53706
309	40	M	scientist	70802
337	37	M	scientist	10522
430	38	M	scientist	98199
538	31	M	scientist	21010

READY ▷ ⚔ 🏧 ⚙