**2.** Αρχικά, υλοποιήθηκε μια συνάρτηση που δημιουργεί ένα Spark Session και δέχεται ως όρισμα τον αριθμό των executors. <code create\_spark\_session>

Για το 2ο ζητούμενο δημιουργήσαμε ένα schema για το DataFrame του βασικού συνόλου δεδομένων. Τα αρχικά ονόματα των στηλών διατηρήθηκαν, ενώ οι τύποι δεδομένων προσαρμόστηκαν με βάση τα ζητούμενα. Στη συνέχεια δημιουργήθηκε ένα ενιαίο DataFrame για όλα τα δεδομένα, όπως φαίνεται στον παρακάτω κώδικα: <code>

Ο συνολικός αριθμός γραμμών του συνόλου δεδομένων, καθώς και ο τύπος κάθε στήλης είναι τα εξής: <output1>

**3.** Στο 3ο ζητούμενο υλοποιήθηκε μία συνάρτηση για το Query 1 χρησιμοποιώντας DataFrame API: <code query1\_df>

Στη συνέχεια το Query 1 υλοποιήθηκε με SQL API: <code query1\_sql>

Παρακάτω φαίνονται τα αποτελέσματα του Query 1 για τις δύο διαφορετικές υλοποιήσεις, καθώς και οι αντίστοιχοι χρόνοι εκτέλεσης. Παρατηρούμε ότι υπάρχει διαφορά στην επίδοση των δύο APIs. Συγκεκριμένα, το DataFrame API πραγματοποιήθηκε σε πολύ λιγότερο χρόνο σε σχέση με το SQL API. <output2>

**4.** Στο 4ο ζητούμενο υλοποιήθηκε μία συνάρτηση για το Query 2 χρησιμοποιώντας DataFrame/ SQL API: <code query2\_df>

Στη συνέχεια το Query 2 υλοποιήθηκε με RDD API: <code query2\_rdd>

Ακολουθούν τα αποτελέσματα του Query 2 για τις δύο διαφορετικές υλοποιήσεις, καθώς και οι αντίστοιχοι χρόνοι εκτέλεσης. Παρατηρούμε ότι υπάρχει διαφορά στην επίδοση των δύο APIs. Συγκεκριμένα, το DataFrame/ SQL API πραγματοποιήθηκε σε πολύ λιγότερο χρόνο σε σχέση με το RDD API. <output3>

**5.** Στο συγκεκριμένο ερώτημα χρησιμοποιήθηκαν τα δευτερεύοντα σύνολα δεδομένων “Estimated Median Income”, “revgecoding” και “LA\_income\_2015”. Επιπλέον, πραγματοποιήθηκε map για τα Vict Descent. <code από main>

Έπειτα υλοποιήθηκε το Query 3 χρησιμοποιώντας DataFrame/ SQL API. <code query3>

Το Query 3 εκτελέστηκε σε ένα for loop, δημιουργώντας διαδοχικά τρία Spark Sessions για 2, 3 και 4 executors. Παρακάτω φαίνονται τα αποτελέσματα και η διάρκεια της κάθε εκτέλεσης αντίστοιχα. Παρατηρούμε ότι ταχύτερα εκτελέστηκε το Query με 3 executors, έπειτα με 2 και τέλος με 4. <output4>

**6.** Για το ζητούμενο 6 χρησιμοποιήθηκαν επιπλέον τα δεδομένα “LAPD\_Police\_Stations”. Το Query 4 υλοποιήθηκε με DataFrame/SQL API και εκτελέστηκε με 4 executors. Αρχικά δημιουργήθηκε μια συνάρτηση για τον υπολογισμό της απόστασης μεταξύ 2 σημείων με συγκεκριμένες συντεταγμένες. Στη συνέχεια πραγματοποιήθηκαν από ένα join για τα δύο πρώτα ερωτήματα. Έπειτα πραγματοποιήθηκε ένα cross join μεταξύ των DataFrame των εγκλημάτων και του “LAPD\_Police\_Stations” για τον εντοπισμό του κοντινότερου τμήματος σε κάθε έγκλημα. Τέλος πραγματοποιήθηκαν άλλα δύο join για τα τελευταία δύο ερωτήματα. Ακολουθεί ο αντίστοιχος κώδικας: <code query4>

Τα αποτελέσματα της εκτέλεσης για τα 4 ερωτήματα είναι τα εξής: <output5>