Data science and SGBD: "server" VS "in-process"

Data Base lab session report - First year of a Master's degree in computer science

Florian EPAIN
florian.epain@irisa.fr

December 28, 2023

Contents

1 Introduction

To run the notebook

- postgreSQL
  - you will have to create a data base in postgreSQL;
  - change the psycopg2.connect() parameters.

1.1 Setup

All tests have been run in a laptop arch linux (distribution: manjaro).

processor	12 x AMD Ryzen 5 5600H with Radeon Graphics
memory	7,2 GiB of RAM
graphics processor	AMD Radeon Graphics

2 Queries

2.1 Restrictions

R.0: Les trajets effectués par le chauffeur dont le **medallion** est *0B57B9633A2FECD3D3B1944AFC7471CF*.

```
SELECT * FROM TripData WHERE medallion = '0B57B9633A2FECD3D3B1944AFC7471CF';
```

R.1: Les trajets pour lesquels **passenger_count** est supérieur ou égal à 2.

```
SELECT * FROM TripData WHERE passenger_count >= 2;
```

R.2: Les trajets pour lesquels **passenger_count** est égal à 1.

```
SELECT * FROM TripData WHERE passenger_count = 1;
```

2.2 Aggregates

A.0: La distance totale cumulée effectuée par l'ensemble des trajets en taxi.

```
SELECT SUM(trip_distance)
FROM TripData;
```

A.1: Pour chaque medallion, la distance moyenne effectuée pour les trajets avec un unique passager.

```
SELECT medallion, AVG(trip_distance)
FROM TripData
WHERE passenger_count = 1
GROUP BY medallion;
```

A.2: Pour chaque jour, le nombre de passagers moyen.

```
SELECT AVG(passenger_count)
FROM TripData
GROUP BY pickup_datetime;
```

A.3: Pour chaque chauffeur dont la distance moyenne parcourue est supérieur à la moyenne des distances, le nombre moyen de passagers par trajet

```
SELECT avg_pass_count
FROM (
    SELECT
        medallion,
        AVG(passenger_count) as avg_pass_count,
        AVG(trip_distance) as avg_trip_dist
    FROM TripData GROUP BY medallion
) as taxisStats
WHERE avg_trip_dist > (
    SELECT AVG(trip_distance) FROM TripData
);
```

3 Tests Optimisations

Index on medallion works well on R.0

4 Final Propositions

5 Comparison

	$Query0$	$Query1$	$Query2$	$Aggregate0$	$Aggregate1$	$Aggregate2$	Agg
postgreSQL no opti	$1.7572s$	$5.8131s$	$9.4128s$	$0.5364s$	$1.1292s$	$18.4276s$	1.
postgreSQL index on medallion	$0.0016s$	$5.6706s$	$10.6403s$		$1.5556s$	$19.6138s$	3.
postgreSQL index on passenger_count	$1.1665s$	$5.0622s$		$2.9944s$	$1.5556s$	$19.6138s$	3.
postgreSQL index + cluster							
duckDB no opti	$0.0003s$	$0.0004s$	$0.0003s$		$0.0885s$	$0.1041s$	0.