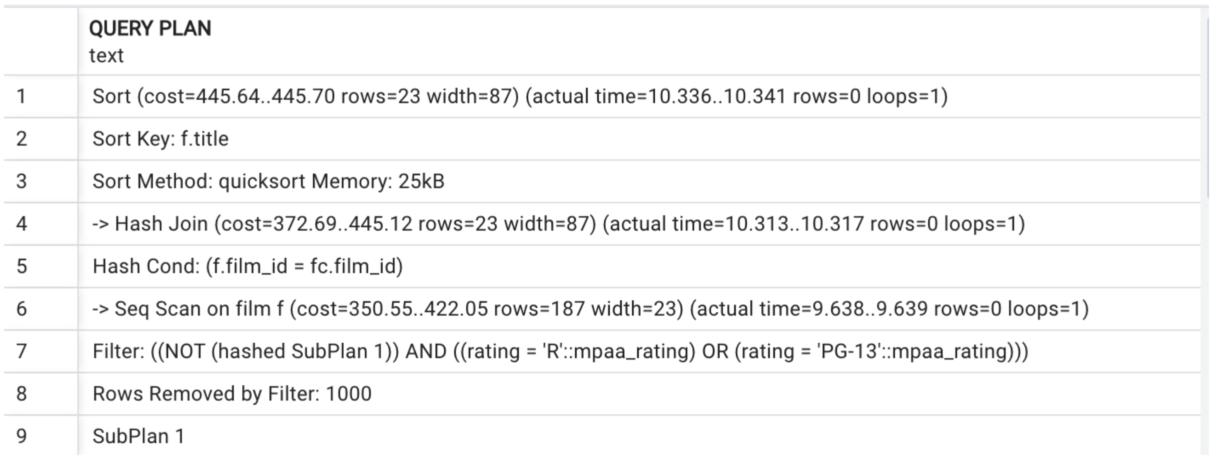
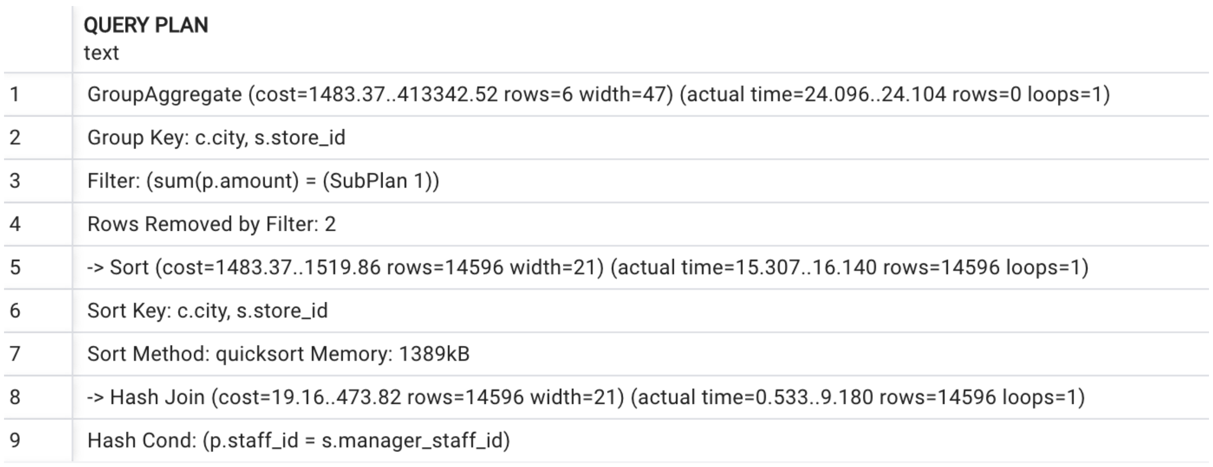


	QUERY PLAN text
1	Sort (cost=445.64..445.70 rows=23 width=87) (actual time=10.336..10.341 rows=0 loops=1)
2	Sort Key: f.title
3	Sort Method: quicksort Memory: 25kB
4	-> Hash Join (cost=372.69..445.12 rows=23 width=87) (actual time=10.313..10.317 rows=0 loops=1)
5	Hash Cond: (f.film_id = fc.film_id)
6	-> Seq Scan on film f (cost=350.55..422.05 rows=187 width=23) (actual time=9.638..9.639 rows=0 loops=1)
7	Filter: ((NOT (hashed SubPlan 1)) AND ((rating = 'R'::mpaa_rating) OR (rating = 'PG-13'::mpaa_rating)))
8	Rows Removed by Filter: 1000
9	SubPlan 1

The most expensive step of first queries is going through film table and selecting from it
Solution: using hash index on film title



	QUERY PLAN text
1	GroupAggregate (cost=1483.37..413342.52 rows=6 width=47) (actual time=24.096..24.104 rows=0 loops=1)
2	Group Key: c.city, s.store_id
3	Filter: (sum(p.amount) = (SubPlan 1))
4	Rows Removed by Filter: 2
5	-> Sort (cost=1483.37..1519.86 rows=14596 width=21) (actual time=15.307..16.140 rows=14596 loops=1)
6	Sort Key: c.city, s.store_id
7	Sort Method: quicksort Memory: 1389kB
8	-> Hash Join (cost=19.16..473.82 rows=14596 width=21) (actual time=0.533..9.180 rows=14596 loops=1)
9	Hash Cond: (p.staff_id = s.manager_staff_id)

The most expensive step of secod queries is matching and grouping city and store
Solution: use b-tree index on store_id