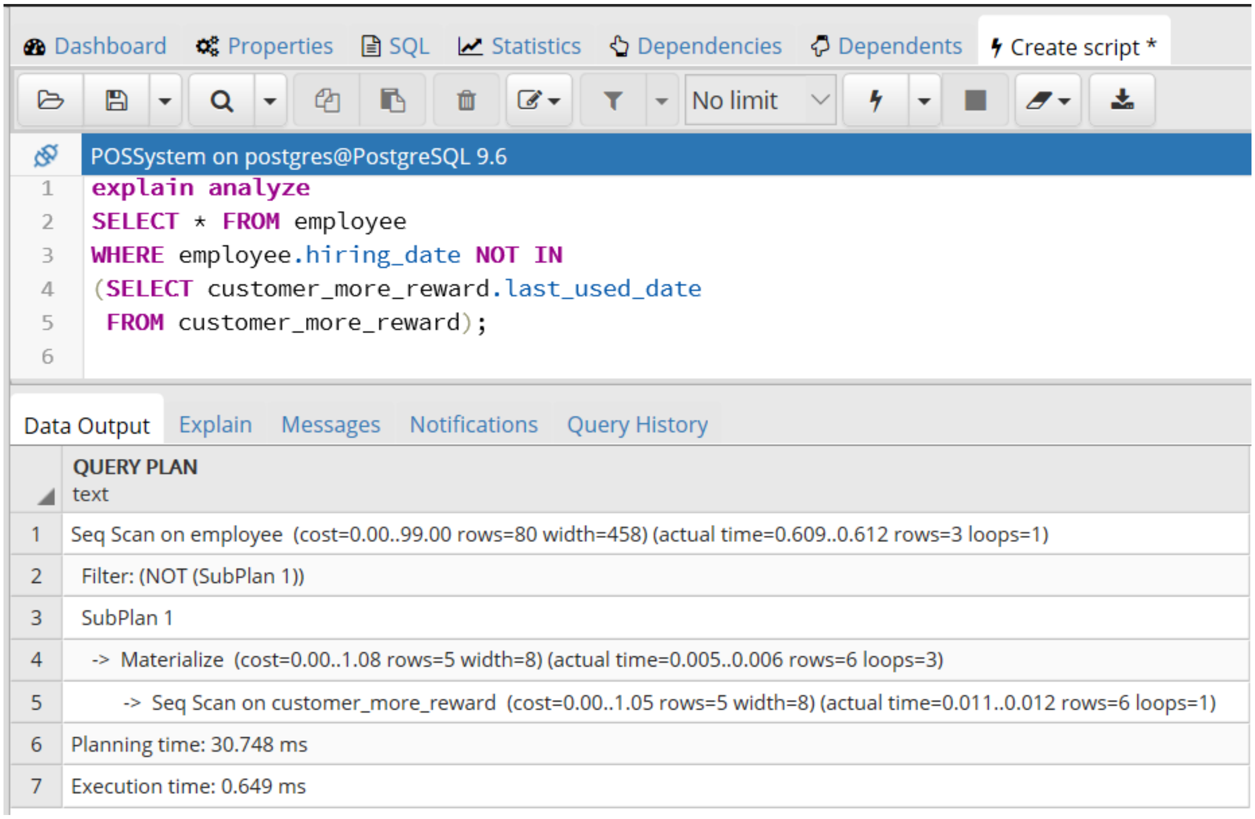
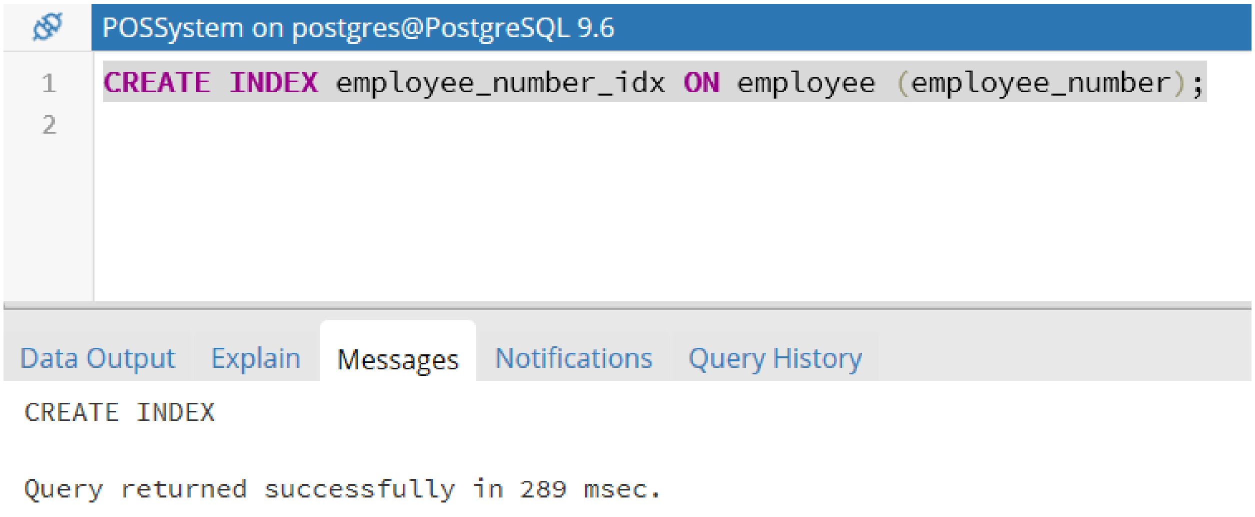
- Query without creating an index yet



```
explain analyze
SELECT * FROM employee
WHERE employee.hiring_date NOT IN
(SELECT customer_more_reward.last_used_date
 FROM customer_more_reward);
```

	QUERY PLAN text
1	Seq Scan on employee (cost=0.00..99.00 rows=80 width=458) (actual time=0.609..0.612 rows=3 loops=1)
2	Filter: (NOT (SubPlan 1))
3	SubPlan 1
4	-> Materialize (cost=0.00..1.08 rows=5 width=8) (actual time=0.005..0.006 rows=6 loops=3)
5	-> Seq Scan on customer_more_reward (cost=0.00..1.05 rows=5 width=8) (actual time=0.011..0.012 rows=6 loops=1)
6	Planning time: 30.748 ms
7	Execution time: 0.649 ms

- Create Index

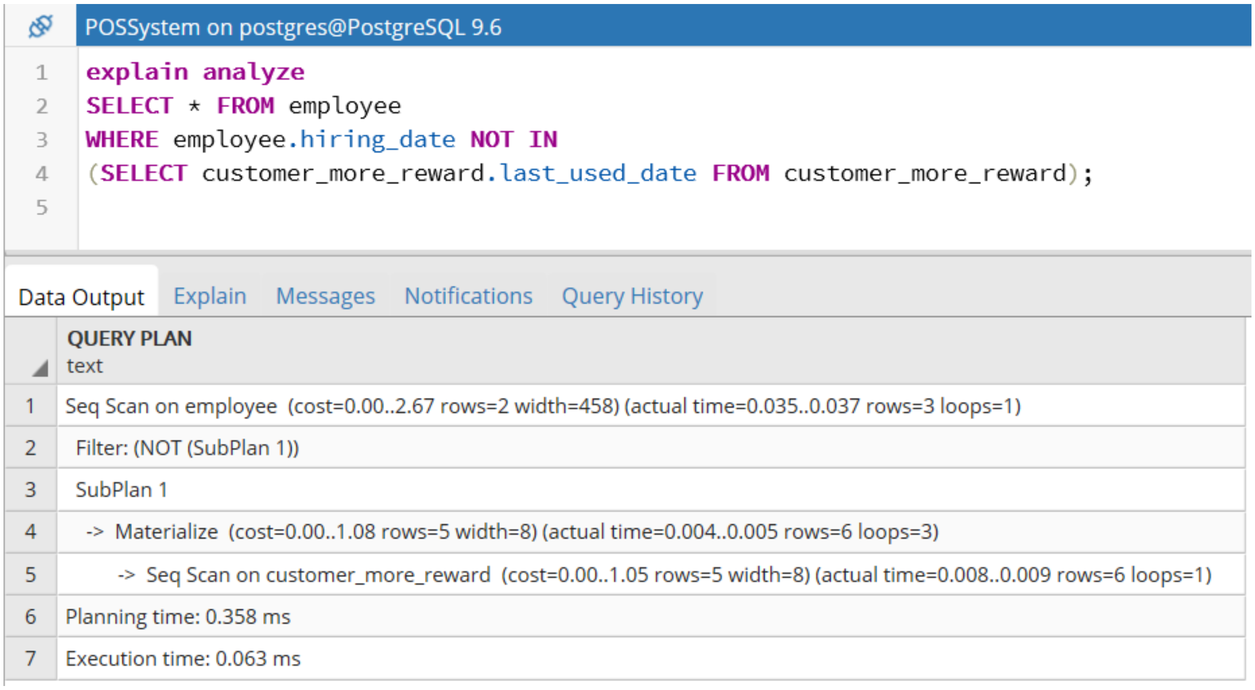


```
CREATE INDEX employee_number_idx ON employee (employee_number);
```

```
CREATE INDEX

Query returned successfully in 289 msec.
```

- After creating an index, as shown in figure below, with a same query, the performance, including planning time, executing time, and sequence scan in the table having an index are all much faster/ shorter than the one without an index.



```sql
explain analyze
SELECT * FROM employee
WHERE employee.hiring_date NOT IN
(SELECT customer_more_reward.last_used_date FROM customer_more_reward);

```

```
  QUERY PLAN
  text
1 Seq Scan on employee  (cost=0.00..2.67 rows=2 width=458) (actual time=0.035..0.037 rows=3 loops=1)
2   Filter: (NOT (SubPlan 1))
3   SubPlan 1
4     ->  Materialize  (cost=0.00..1.08 rows=5 width=8) (actual time=0.004..0.005 rows=6 loops=3)
5           ->  Seq Scan on customer_more_reward  (cost=0.00..1.05 rows=5 width=8) (actual time=0.008..0.009 rows=6 loops=1)
6 Planning time: 0.358 ms
7 Execution time: 0.063 ms
```