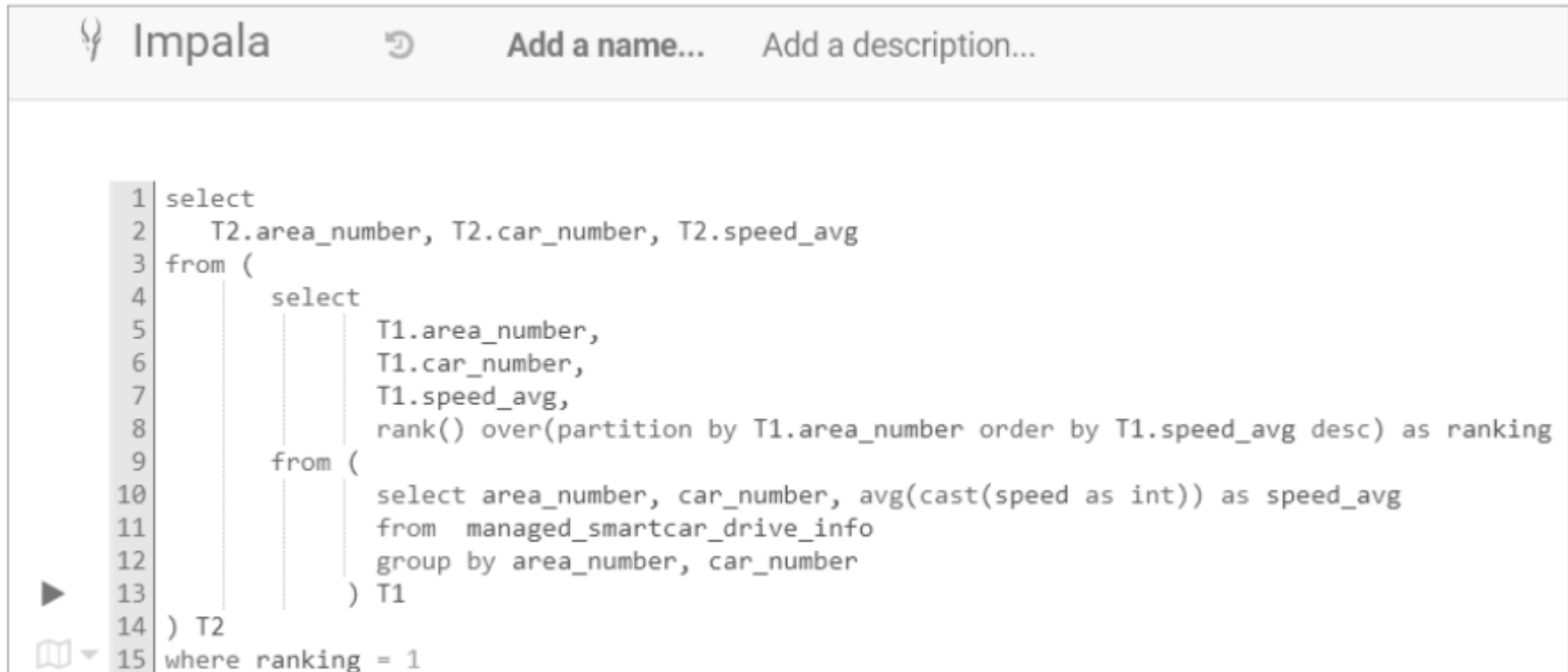


7.5 분석 파일럿 실행 3단계-임팔라로 실시간 분석

임팔라를 이용한 운행 지역 분석

01. Impala Editor에 그림 7.35의 쿼리를 입력하고 실행한다. 쿼리를 자세히 보면 Rank() 함수 등이 들어 있어 하이브에서 실행하면 오버헤드가 큰 쿼리지만 임팔라를 이용하면 빠른 응답 속도로 결과가 출력된다. 실행 쿼리는 C://예제소스/bigdata2nd-master/CH07/ImpalaSQL/ 경로에 있으니 참고한다.



```
1 select
2     T2.area_number, T2.car_number, T2.speed_avg
3 from (
4     select
5         T1.area_number,
6         T1.car_number,
7         T1.speed_avg,
8         rank() over(partition by T1.area_number order by T1.speed_avg desc) as ranking
9     from (
10        select area_number, car_number, avg(cast(speed as int)) as speed_avg
11        from managed_smartcar_drive_info
12        group by area_number, car_number
13    ) T1
14 ) T2
15 where ranking = 1
```

7.5 분석 파일럿 실행 3단계-임팔라로 실시간 분석

임팔라를 이용한 운행 지역 분석

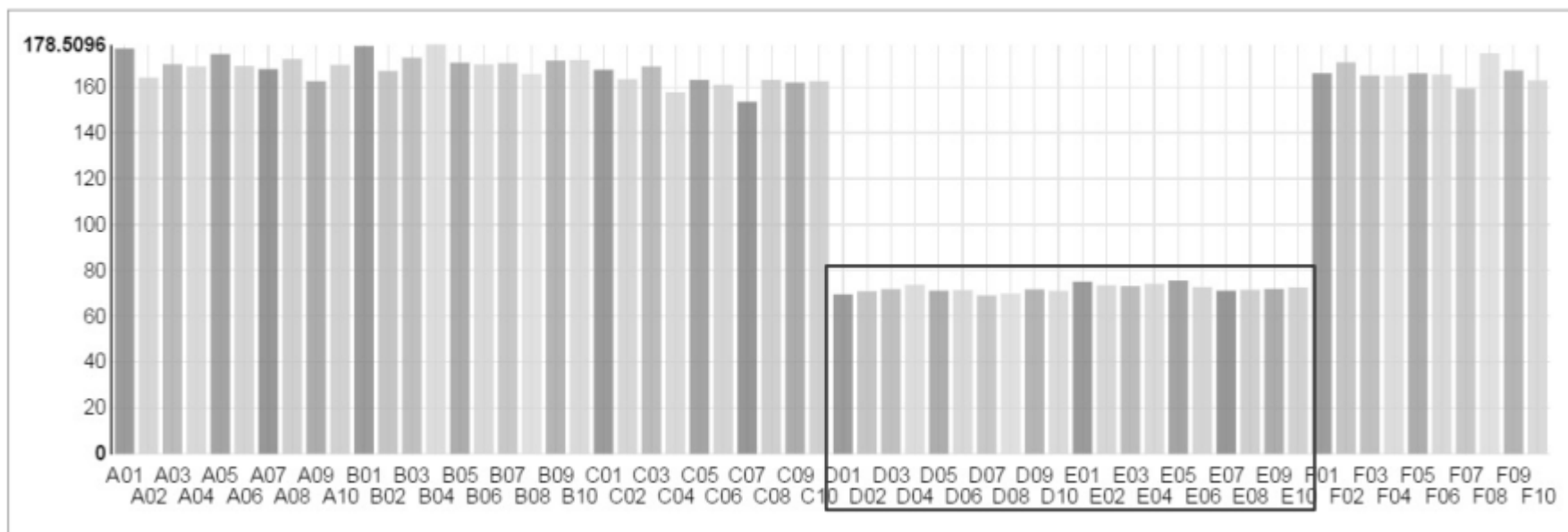
02. 정상적으로 실행됐다면 다음과 같은 결과가 출력될 것이다. 다음의 출력된 6개의 결과를 보면 A01~A06 지역에서 차량번호가 N0001, D0069인 차량이 160km/h 이상의 평균 속도로 과속하는 상습차량으로 보인다.

최근 쿼리 쿼리 로그 열 결과 차트			
	area_number	car_number	speed_avg
0	A01	N0001	176.77675840978594
1	A02	D0069	164.04942965779469
2	A03	D0069	169.88931297709922
3	A04	D0069	168.97069597069597
4	A05	N0001	174.33333333333334
5	A06	D0069	169.14576271186439

7.5 분석 파일럿 실행 3단계-임팔라로 실시간 분석

임팔라를 이용한 운행 지역 분석

03. 차트 보기로 해서 데이터 전체를 그림 7.36의 표에서 그림 7.37의 막대 그래프로 전환해서 한눈에 보이도록 시각화한다.



Tip _ 인메모리 기반 분석 엔진 사용

임팔라는 빠른 응답 속도를 보여 주지만 많은 리소스를 사용한다. 특히 메모리 사용률이 높는데, 분산 환경의 데이터를 처리할 때 발생하는 중간 데이터셋을 모두 메모리에 올려 놓고 작업하기 때문이다. 반면 하이브는 중간 데이터셋을 디스크에 저장해 느리지만 안정적이다. 빅데이터 하둡 환경에서는 임팔라와 하이브를 상황에 맞춰 선택적으로 사용해야 한다. 마트화된 소규모 데이터셋을 대상으로 빠른 애드혹 분석을 수행할 때는 임팔라를 사용하고, 대규모 데이터셋에서 긴 시간에 걸친 가공 및 추출 작업에는 반드시 하이브를 이용해야 한다. 후자의 경우 임팔라 사용이 빈번하면 하둡 클러스터에 메모리 오버헤드가 발생해 장애로 이어질 수 있다. 보통은 업무 마감 후 빅데이터 웨어하우스의 대규모 배치 작업에는 안정적인 하이브가 적합하고, 업무 시간 중에 마트 데이터를 대상으로 애드혹(Ad-Hoc)한 탐색적 분석에는 빠른 임팔라가 적합하다. 마트 데이터는 집계/요약된 작은 데이터셋으로, 임팔라 작업 시 메모리에 대한 부담이 적기 때문이다. 또한 이 같은 문제를 근본적으로 해결하기 위해 빅데이터 시스템을 배치계와 분석계로 분리 구성하고, 배치 처리된 데이터를 분석계에 주기적으로 복사하는 아키텍처를 만들기도 한다. 이때 DistCp라는 하둡 분산 카피 기술이 이용된다.

7.5 분석 파일럿 실행 3단계-임팔라로 실시간 분석

임팔라를 이용한 운영 지역 분석

실습