Tip:

다음 단계(Next Steps)

자세한 내용 보기

oracle.com/technetwork/developer-tools/adf/overview

Oracle Application Development Framework 11g Release 1 (11.1.1.4.0)을 위한 Oracle Fusion Middleware Fusion 개발자가이드

Oracle Fusion 개발자 가이드: Oracle ADF Business Component 및 Oracle ADF Face을 이용한 대용량 인터넷 애플리케이션의 개발 (McGraw-Hill, 2010년)

Oracle ADF Insider 세미나 시리즈

Nimphius의 다른 글 읽기

Oracle ADF Code Corner

Oracle Technology Network Harvest 블로그

Oracle JDeveloper 및 Oracle ADF에 대한 논의

Oracle JDeveloper 포럼

테크놀러지 : Oracle Exadata

Smart Scan과 스토리지 인덱스(Storage Index)의 만남

저자 - **Arup Nanda**
감수 - **김윤호** 수석 아키텍트, Oracle APAC(yun-ho.kim@oracle.com)

Oracle Exadata에서 스토리지 인덱스를 사용하여 I/O 속도를 높이는 방법에 대한 이해

Oracle Exadata Database Machine은 데이터베이스 성능을 완전히 새로운 차원으로 끌어 올렸다. 그렇다면, 이와 같은 성능 향상을 실현하는 요인은 정확하게 무엇일까? Oracle Database 11g Release 2, Oracle Exadata의 Smart Flash Cache, Hybrid Columnar Compression, SmartScan 기능 그리고 InfiniBand interconnect 등과 같은 여러 Oracle Exadata Database Machine 구성 요소들이 고성능을 실현하는데 기여하고 있다. 이와 같은 성능을 지원하는 핵심 기술 중 하나는 스토리지 인덱스다. 이는 일반적인 데이터베이스 인덱스가 아니다. 스토리지 인덱스는 스토리지 서버의 메모리 – 또는 스토리지 셀(Storage cell) 내에 상주하며, 스토리지 셀에서 연관성 없는 데이터베이스 블록을 제외시키는 방식으로 불필요한 I/O를 대폭 줄인다. 여기에서는 스토리지 인덱스가 어떻게 실행되고 효율적으로 활용되도록 하기 위해서는 어떻게 해야 하는지에 대해 설명하고 있다.

전통적인 데이터베이스 I/O

CPU 용량과 메모리 형태의 컴퓨팅 파워는 지난 수십 년 동안 가히 기하급수적으로 증가했지만, 스토리지 서브시스템의 파워는 그와 같은 수준으로 증가하지는 못했다. 스토리지 성능이 CPU와 메모리 성능처럼 향상되지 못한 데는 여러 많은 이유가 있으며 물리적인 요소도 그 중 하나다. 단일 디스크의 회전 속도는 한계가 있으며 디스크 어레이의 각 플래터(Platter)는 제한된 용량의 정보만 보관할 수 있다. 따라서 프로세싱 체인의 취약한 링크는 계속해서 스토리지가 될 것이며 컴퓨팅 응답 시간 중 가장 큰 비중을 차지하게 될 것이다.

사용자가 Oracle Exadata 상에서 실행되지 않는 Oracle Database 인스턴스에 대해 다음과 같은 쿼리를 발행하면 어떻게 되는지 생각해 보겠다.

```
select avg(amt) from sales where
cust_level = 3.
```

cust_level 컬럼 상에 인덱스가 없는 경우, 이는 옵티마이저(Optimizer)가 이 쿼리를 위해 전체 테이블 스캔을 선택해야 한다는 것을 의미한다. 이 세션에 해당하는 Oracle Database 프로세스는 스토리지 내 세일

즈 테이블의 전체(All) 데이터베이스 블록을 가져와 데이터를 검사한 다음, 해당 쿼리를 충족하지 않는 열(Row)을 삭제하라는 요청을 발행한다. (Oracle Database가 열을 요청한 경우, 전체 데이터베이스 블록—일반적으로 8KB의 크기—을 디스크에서 메모리로 가져와야 한다.)
스토리지 서브시스템은 오직 비트 및 바이트 단위의 데이터 용량만 처리할 뿐 데이터베이스 블록 내에 저장된 실제 데이터에 대해서는 그 어떤 정보도 가지고 있지 않다. 데이터베이스 서버만이 어떤 블록 내에 어떤 데이터가 있는 지 판단할 수 있다.

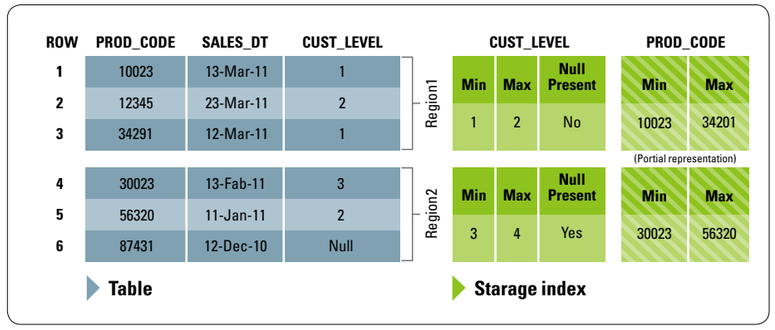
Oracle Exadata I/O 및 Smart Scan

Oracle Exadata의 스토리지는 어떤 열이 쿼리를 충족하는지 확인하기 위해 해당 서버의 데이터베이스 서버로 모든 블록을 가져오지는 않도록 쿼리 프로세싱을 변경했다. Oracle Exadata의 Smart Scan 기능은 스토리지 셀 내에서 특정 유형의 쿼리 프로세싱이 실행되도록 한다. Smart Scan 기술을 통해 데이터베이스 노드는 iDB (Intelligent Database)로 불리는 프로토콜을 통해 쿼리 세부 사항을 스토리지 셀로 전달한다. 이 정보를 이용하여 스토리지 셀은 데이터 집약적인 쿼리 프로세싱을 상당 부분을 분산시킨다. Oracle Exadata 스토리지 셀은 쿼리에 대한 추가 인텔리전스를 이용하여 스토리지 디스크를 검색하고 모든 데이터베이스 블록이 아니라 오직 관련된 바이트만 데이터베이스 노드로 전송한다. 이 때문에 스마트 스캔(Smart scan)이라고 부르는 것이다.
Oracle Exadata에서는 선택된 함수와 술어 내 =, 〉 등과 같은 연산자를 이용하는 전체 테이블 스캔(Full table scan)을 실행하면서 Smart Scan을 활용할 수 있다. Smart Scan을 통해 성능 향상이 가능한 함수를 확인하기 위해 다음과 같은 쿼리를 실행할 수 있다.

```
select name from v$sqlfn_metadata
```

스토리지 인덱스(Storage Index)

스토리지를 스캐닝하는 동안, Oracle Exadata 스토리지 셀은 디스크 스토리지 내 어떤 영역이 해당 쿼리가 관심을 원하는 값을 명확하게 포함하고 있지 않는 지 규명하고 이러한 영역이 읽혀지는 것을 방지할 수 있다.
스토리지 셀은 쿼리가 원하는 데이터를 포함하지 않은 디스크 스토리지 영역이 읽혀지는 것을 방지하는 방법을 어떻게 알고 있을까? 그 해답은 스토리지 인덱스를 사용하는 것이다. 스토리지 인덱스는 물리적 스토리지의 특정 영역 내 데이터에 대한 일부 정보를 포함하고 있는 메모리 내 구조다. 이 정보는 스토리지 셀에 어떤 디스크 영역이 쿼리에서 원하는 값을 포함하고 있지 않으며, 따라서 스캔 중 해당 영역에는 액세스하지 않아야 하는지 알려 준다.
<그림 1>은 스토리지 인덱스 내 데이터가 어떻게 유지되는지 보여주고 있다. PROD_CODE, SALES_DT, CUST_LEVEL 등 여러 컬럼에 대한 스토리지 인덱스 입력 항목이 있으며, 각 스토리지 인덱스 입력 항목은 테이블의 물리적 영역(Region)을 차지하며 해당 영역 내 최소 및 최대값을 포함하는 것은 물론, 해당 영역 내 어떤 열이 널(null)을 포함하고 있는지 여부를 표시한다. 이 예제에서 스토리지 인덱스의 영역 1은 테이블의 1~3열을 나타내며 스토리지 인덱스 영역 2는 테이블의 4~6열을 나타낸다. <그림 1>의 영역 1에 대한 스토리지 인덱스 입력 항목은 영역 내 cust_level 컬럼의 최소 및 최대값이 각각 1과 2가 되어야 한다는 것을 보여준다. 이 입력 항목은 또한 영역 1의 테이블 열 내에는 널(null) 값이 없다는 것을 보여 준다. 영역 2의 스토리지 인덱스 입력 항목은 cust_level 컬럼의 최소 및 최대 값은 각각 3과 4가 되어야 한다는 것을 보여준다. 영역 2 내 테이블의 1개 열에는 널(null) 값이 포함되어 있으며 따라서 영역 2 내 스토리지 인덱스 입력 항목의 널 인디케이터(null indicator)는 Yes로 나타난다. 간단히 설명하기 위해 <그림 1>은 스토리지 인덱스 내부의 오직 단 1개 컬럼(CUST_LEVEL)의 전체 세부 사항을 보여 주고 있으며 다음 컬럼(PROD_CODE)은 부분적으로만 표시되어 있다.



<그림 1> 스토리지 인덱스의 개념 제시

앞서 예제 쿼리에서 사용자가 발행한 명령문은 다음과 같다.

```
select avg(amt) from sales where
cust_level = 3
```

이 쿼리가 Oracle Exadata 스토리지에 대해 실행되면, <그림 1>의 CUST_LEVEL 컬럼에 있는 스토리지 인덱스 입력 항목은 영역 1이 1과 2 사이의 값을 가지고 있으며, 따라서 해당 스토리지 인덱스 영역에서 cust_level = 3 결과는 발견되지 않을 것이다. 그러므로, 스토리지 셀은 디스크 내 해당 영역에 접근하지 않는다. 스토리지 인덱스의 영역 2는 그 값이 3과 4 사이가 되며 따라서 cust_level =3은 최소한 1개의 열에서 발견된다. 스토리지 셀은 디스크에서 영역 2를 읽어 온다.
마찬가지로, 다음과 같은 쿼리를 가정해 보겠다.

```
select avg(amt) from sales where
cust_level is null
```

이 경우, 스토리지 인덱스는 테이블(영역 1)의 열 1, 2 및 3이 명확하게 WHERE cust_level IS NULL 조건을 충족하지 않는 데 반해, 열 4, 5 및 6(영역 2) 중 1개 이상이 해당 조건을 충족할 수 있다.

보다시피, Oracle Exadata 스토리지 인덱스는 사용자가 원하는 값을 포함하고 있는 테이블의 영역을 찾는 것이 아니라, 명확하게 값을 포함하고 있지 않는 영역을 식별하여 I/O 프로세싱에서 이를 제거한다. 말하자면, 전통적인 데이터베이스 인덱스의 정반대인 네거티브(negative) 인덱스의 기능을 수행한다. 이는 정보를 포함하고 있을 수 있는 데이터베이스 블록을 발견하기 위한 것(삭제하는 것이 아니라)이다.

스토리지 인덱스는 디스크상에 저장되지 않으며 스토리지 셀 서버의 메모리 내에 상주한다. 이는 스토리지 셀이 반복 쿼리를 수신한 이후 자동으로 생성된다. 스토리지 인덱스를 생성 또는 유지하기 위해 사용자가 개입할 필요는 없다. 또한, 메모리 상주 구조이기 때문에 스토리지 셀이 재부팅되면 사라진다.

스토리지 인덱스를 사용하기 위해, Oracle Exadata 쿼리는 smart scan을 사용해야 하며, 모든 유형의 애플리케이션이 스토리지 인덱스를 통해 효과를 거둘 수 있는 것은 아니다. 연산자를 가지고 있으며 많은 전체 테이블 스캔 또는 고속 전체 인덱스 스캔을 실행하는 쿼리를 포함한 애플리케이션 – 일반적으로 데이터 웨어하우징 환경에서 사용됨 – 은 스토리지 인덱스를 통해 상당한 효과를 거둘 수 있다. 반대로 OLTP(Online Transaction Processing) 애플리케이션은 대개 표준 인덱스를 통해 소수의 열에 접근하며 전체 테이블 스캔을 실행하지 않는다. 따라서 스토리지 인덱스로 개선 효과를 거둘 수 없다.

절감 효과 확인

스토리지 인덱스를 통해 얼마나 많은 I/O가 줄어드는지 확인해 보겠다. V$MYSTAT 뷰를 쿼리하여 Oracle Exadata 상의 I/O 절감 효과를 확인하겠다.

이 뷰는 현재 세션에 대한 여러 측정 지표의 값을 포함하고 있다. 관심을 가지고 있는 2가지의 측정 지표는 다음과 같다.

스토리지 인덱스를 통해 절감되는 셀 물리적 IO 바이트(Cell physical IO bytes). 이 측정 지표는 스토리지 셀 레벨에서 스토리지 인덱스를 활용함으로써 얼마나 많은 I/O가 제거되었는지 보여준다.

Smart Scan에 의해 반환된 셀 물리적 IO 인터커넥트 바이트.(Cell physical IO interconnect bytes). 이 측정 지표는 Smart Scan에 의해 데이터베이스 서버로 얼마나 많은 I/O가 반환되었는지 보여준다.

리스팅 1(Listing 1)의 코드는 이들 2개의 측정 지표에 해당하는 명칭과 값을 반환한다. Listing 1의 결과물에서 공간을 절약하기 위해, smart scan에 의해 반환된 셀 물리적 I/O 인터커넥트 바이트에서 "Smart Scan"과 스토리지 인덱스에 의해 절감된 셀 물리적 I/O 바이트에 대해 "SI Savings"라는 약칭으로 측정 지표를 나타낸다. 또한, 두 측정 지표에 대한 값을 메가바이트로 표시한다.

코드 리스팅 1: 스토리지 인덱스를 통한 I/O 절감 확인

```
col stat_value format 9,999.9999

select
  decode(name,
  'cell physical IO bytes saved by storage index',
  'SI Savings',
  'cell physical IO interconnect bytes returned by smart scan',
  'Smart Scan'
  ) as stat_name,
  value/1024/1024 as stat_value
from v$mystat s, v$statname n
where
  s.statistic# = n.statistic#
and
  n.name in (
  'cell physical IO bytes saved by storage index',
  'cell physical IO interconnect bytes returned by smart scan'
  )
/
```

Oracle Exadata에 대한 세션을 시작한 이후, 다음 명령을 발행한다.

```
select avg(amt) from sales where
cust_level is null
```

그런 다음, 리스팅 1의 쿼리를 실행하여 스토리지 인덱스 사용 효과를 확인할 수 있다. 그 결과물은 다음과 같다.

```
STAT_NAME   STAT_VALUE
----------- -----------
SI Savings     545.9234
Smart Scan     0.0012
```

결과물에서 볼 수 있듯이, 스토리지 인덱스를 통한 I/O 절감 효과는 약 546MB이며 스토리지 셀은 데이터베이스로 0.0012MB를 반환했다. 이들 수치들은 워크로드와 환경에 따라 크게 좌우되지만, 측정 개념은 여전히 동일하다. 이 쿼리를 이용하여 특정한 경우, 특히 스토리지 인덱스가 사용되지 않거나 아직 생성되

지 않는 경우, 그 절감 효과에 대해 조사할 수 있다.
스토리지 인덱스가 그처럼 대단하다면, 왜 Oracle Exadata는 항상 이를 사용하지 않을까? 그 대답은 한마디로 스토리지 인덱스가 효과가 있는 경우에만, 생성되고 사용되기 때문이라는 것이다.

먼저, 데이터 분배(data distribution)가 중요한 역할을 담당한다. 예를 들어, Oracle Exadata상에서 세일즈 테이블 샘플에 대해 다음과 같은 쿼리를 실행해 보겠다.

```
select avg(amt) from sales where
sales_dt = '13-MAR-11';
```

그런 다음 리스팅 1의 쿼리를 실행하고 결과를 검토하여 스토리지 인덱스의 효과를 확인해 보겠다.

```
STAT_NAME        STAT_VALUE
---------------  ---------------
SI Savings       0.0000
Smart Scan       0.9035
```

그 결과가 다소 놀라울 수도 있다. 스토리지 인덱스를 통한 I/O 절감은 0이다. 다시 말해 스토리지 인덱스는 사용되지 않았다. 하지만, "Smart Scan" 통계에서 smart scan이 I/O를 크게 줄인 것 – 단 0.9MB (2~3백 기가바이트의 데이터를 포함하고 있는 테이블에서) – 을 확인할 수 있다.
하지만, 왜 스토리지 인덱스가 사용되지 않았을까? 세일즈 테이블의 sales_dt 컬럼의 값은 임의로 테이블 블록 상에 분배되며 거의 모든 블록은 다음을 충족하는 열을 포함하고 있다.

```
where sales_dt = '13-MAR-11'
```

스토리지 인덱스가 사용되는 경우에도, I/O를 줄이지 못할 수도 있다. 그 이유는 모든 블록이 잠재적으로 쿼리를 충족하며 이에 따라 I/O 절감 효과가 전혀 없기 때문이다.
이와 같은 상황을 해결하기 위해 소수의 데이터베이스 블록 내에서 더 많은 일치 열이 발견될 가능성을 높일 수 있으며, 따라서 Oracle Exadata 스토리지 인덱스는 보다 적은 수의 블록을 찾아 데이터베이스 서버 노드로 전송한다. sales_dt 컬럼이 명령한 테이블을 재로딩하여 이를 수행할 수 있다. SQL 명령문은 다음과 같다.

```
SQL> rename sales to sales_old;
SQL> create table sales nologging as select * from sales_old where 1=2;
SQL> insert /*+ APPEND */ into sales select * from sales_old order by
sales_dt;
```

이제 세일즈 테이블에 대한 쿼리를 실행한다.

```
select avg(amt) from sales
where sales_dt = '13-MAR-11';
```

리스팅 1의 코드를 실행하여 측정 지표를 확인해 보면, 스토리지 인덱스를 통해 434MB의 I/O가 절감된 것을 확인할 수 있다.

```
STAT_NAME        STAT_VALUE
---------------  ---------------
SI Savings       434.0045
Smart Scan       0.0012
```

결론

Oracle Exadata의 스토리지 인덱스는 스토리지 셀 내 데이터의 분배를 인식하고 있으며 smart scan을 실행하는 동안 스토리지 셀 내 물리적 I/O를 제거할 수 있도록 지원한다. 이는 I/O를 크게 줄여 Oracle Exadata Database Machine 내에서 데이터를 훨씬 빠르게 처리할 수 있도록 한다. Oracle Exadata 쿼리 내에서 스토리지 인덱스가 사용되는지 여부를 확인할 수 있으며 특정 쿼리에서 I/O 절감을 측정할 수 있다.

Tip:

다음 단계(Next Steps)

Oracle Exadata에 대한 보다 자세한 내용 보기
oracle.com/us/products/database/database-machine
Oracle Exadata Database Machine 및 Exadata Storage Server의 테크니컬 개요(pdf)

저자 : Arup Nanda (arup@proligence.com)는 14년 이상 Oracle DBA로 활동하면서 성능 튜닝에서 보안 및 재난 복구에 이르는 데이터베이스 관리의 모든 측면을 다루고 있다. 2003년에는 Oracle Magazine의 '올해의 DBA'에 선정되기도 했다.