

Partitioning in Oracle Database 11g

오라클 백서

2007년 6월

참고 :

본 문서는 일반적인 제품 방침을 간략하게 설명하는 것을 목적으로 합니다. 정보 제공만을 목적으로 하므로 어떤 계약에도 본 문서를 포함시킬 수 없습니다. 자료나 코드 또는 기능을 제공할 것이라는 약속이 아니므로 구매 결정 시 본 문서에 의존해서는 안 됩니다. 오라클의 제품에 대해 기술한 사양이나 기능의 개발이나 출시 또는 시기는 오라클의 단독 재량에 따릅니다.

참고	2
파티셔닝개념	4
도입	4
파티셔닝의 혜택	4
파티셔닝의 기초	4
관리 가능성을 위한 파티셔닝	6
성능을 위한 파티셔닝	6
가용성을 위한 파티셔닝	7
파티셔닝	8
파티셔닝 기본 전략	8
파티셔닝 확장자	9
Partition Advisor	10
파티셔닝 전략 및 확장자 개요	11
파티셔닝의 정보 라이프사이클 관리	11
결론	12

파티셔닝 개념

도입

1997년 Oracle 8.0에서 처음 도입된 Oracle Partitioning은 오라클 데이터베이스의 가장 중요하고 성공적인 기능의 하나로, 수 많은 애플리케이션의 관리 가능성, 성능, 및 가용성을 향상시킵니다. Oracle Database 11g는 획기적인 기능 향상을 달성한 8세대 파티셔닝 제품을 도입했습니다. 이 새로운 기술을 보다 다양한 비즈니스 시나리오 모델링을 가능하게 하는 한편, 파티셔닝 어드바이스 및 자동화라는 완벽한 새로운 프레임워크를 통해 모든 기업에서 Oracle Partitioning을 사용할 수 있도록 합니다. Oracle Database 11g는 파티셔닝이 처음 도입된 이래 최대의 신제품으로 평가되며, 지난 10여 년 간 파티셔닝에 대한 고객 투자를 지속적으로 보호할 것입니다.

파티셔닝의 혜택

파티셔닝은 관리 가능성, 성능, 가용성 증대를 통해 수 많은 애플리케이션에 엄청난 혜택을 제공합니다. 파티셔닝은 중요도에 따라 특정 쿼리나 유지 활동의 성능을 향상시킬 수 있습니다. 파티셔닝은 기존 관련 정보를 저렴한 스토리지에 저장하여 지속적으로 온라인화하는 “티어드 아카이빙” 접근 방식을 사용하여 TCO를 획기적으로 줄여줍니다. 따라서 Oracle Partitioning은 대규모 환경을 위한 정보 라이프사이클 관리 (Information Lifecycle Management)에 효율적이고 간편하면서 강력한 접근을 가능하게 합니다.

파티셔닝은 또한 DB 디자이너 및 관리자들이 첨단 애플리케이션에서 제기되는 가장 어려운 문제들도 해결할 수 있도록 지원합니다. 파티셔닝은 멀티 테라바이트 시스템이나 극단적 고가용성을 요구하는 시스템 구축을 위한 핵심 툴입니다.

파티셔닝의 기초

파티셔닝은 테이블, 인덱스 및 인덱스 정렬 테이블을 보다 작은 단위로 분할합니다. 이들 데이터베이스 객체의 작은 단위들은 각각 파티션으로 불립니다. 각각의 파티션은 이름을 가지고 있으며 저장 특성을 갖기도 합니다. 데이터베이스 관리자 입장에서 보면, 파티션 객체는 개별적 혹은 집합적으로 관리될 수 있는 몇 개의 조각을 갖고 있습니다. 때문에 관리자는 파티션 객체를 관리하는데 있어 상당한 유연성을 발휘할 수 있습니다.

하지만, 애플리케이션 측면에서 보면, 파티션 테이블은 파티션되지 않은 테이블과 다를 바 없습니다. 다시 말해, SQL DML 명령어를 사용하여 파티션 테이블에 접근할 때 변경 필요성이 없다는 것입니다.

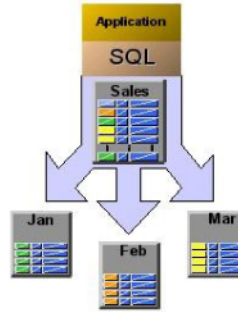


그림 1: 애플리케이션 및 DBA 관점에서의 파티션 테이블

테이블, 인덱스 및 인덱스 정렬 테이블 등 데이터베이스 객체들은 주어진 로 (row)가 위치할 파티션을 결정하는 칼럼 세트인 ‘파티셔닝 키 (partitioning key)’를 사용하여 파티셔닝됩니다. 예를 들어, 그림 1의 판매 테이블은 월간 파티셔닝 전략을 사용하여 판매 날짜 기준으로 range 파티셔닝되었습니다. 이 테이블은 모든 애플리케이션에 단일 ‘정상’ 테이블로 보입니다. 하지만 DBA는 서로 다른 스토리지 tier를 사용하거나 기존 데이터에 테이블 압축을 적용하거나 읽기 전용 테이블스페이스에 있는 기존 데이터의 전체 범위를 저장하거나 하는 등의 방법을 사용하여 각각의 월간 파티션을 개별적으로 저장, 관리할 수 있습니다.

선택된 인덱스 파티셔닝 전략과 무관하게, 인덱스는 기본 테이블의 기본 파티셔닝 전략과 연계되기도 하고 연계되지 않기도 합니다. 적절한 인덱스 파티셔닝 전략은 비즈니스 요구에 기반을 두고, 모든 애플리케이션에 파티셔닝이 적합하도록 지원합니다. Oracle Database 11g는 3가지 형식의 파티션 인덱스를 가지고 있습니다.

- Local Indexes: 로컬 인덱스는 기본 파티션 테이블과 연계된 파티셔닝된 테이블에 존재하는 인덱스로, 테이블의 파티셔닝 전략을 전수 받습니다. 따라서, 로컬 인덱스의 각각의 파티션은 기본 테이블의 오직 하나의 파티션과 대응합니다. 이같은 연계는 파티션 유지의 최적화를 가능하게 합니다. 예를 들어, 테이블 파티션이 드롭되면, 오라클도 대응 인덱스 파티션만을 드롭시키면 됩니다. 고가의 별도 인덱스 유지 작업이 요구되지 않습니다. 로컬 인덱스는 데이터 웨어하우스 환경에서 가장 일반화되어 있습니다.
- Global Partitioned Indexes: 글로벌 파티션 인덱스는 서로 다른 파티셔닝 키 혹은 파티셔닝 전략을 사용하여 파티셔닝 된 파티션 혹은 비파티션 테이블에 존재하는 인덱스입니다. 글로벌 파티션 인덱스는 range 혹은 hash 파티셔닝을 사용하여 파티셔닝되며, 기본 테이블과 연계되지 않습니다. 예를 들어, 테이블은 월별로 range 파티션되어 12개의 파티션을 갖지만, 이 테이블의 인덱스는 서로 다른 파티셔닝 키를 사용하여 range 파티셔닝되어 다른 수의 파티션을 가질 수 있습니다. 글로벌 파티션 인덱스는 데이터 웨어하우징 환경보다 OLTP 환경에서 널리 사용됩니다.

- Global Non-Partitioned Indexes: Global Non-Partitioned Indexes: 글로벌 논파티션 인덱스는 비파티션 테이블에 존재하는 인덱스와 본질적으로 동일합니다. 인덱스 구조는 파티션되지 않고 기본 테이블과 연계되지 않습니다. 데이터 웨어하우징 환경에서, 글로벌 논파티션 인덱스의 가장 일반적 사용 방법은 프라이머리 키 제한을 실행하는 것입니다. 반면, OLTP 환경은 주로 글로벌 논파티션 인덱스를 주로 사용합니다.

또한, 오라클은 포괄적 파티셔닝 테이블 관리를 위한 SQL 명령어 세트를 제공하는데, 여기엔 새로운 파티션 추가, 드롭핑, 제거, 분할, 병합, 이동 및 압축을 위한 명령어가 포함됩니다.

관리 가능성을 위한 파티셔닝

Oracle Partitioning 옵션은 테이블 및 인덱스가 보다 작고 관리 가능한 단위로 파티셔닝되어, 데이터베이스 관리자가 데이터 관리를 위한 “분할 통치” 접근 방식을 적용할 수 있도록 합니다.

파티셔닝을 하면, 유지 관리 활동은 테이블의 특정 부분에 초점을 두고 진행될 수 있습니다. 예를 들면, 데이터베이스 관리자는 전체 테이블이 아닌 테이블중 2006년 데이터 등 특정 데이터를 포함하는 단일 부분만을 압축할 수 있습니다. 데이터베이스 객체 전반에 걸친 유지 관리 활동과 관련해서, 파티션 단위의 유지 활동이 가능해져, 유지 프로세스를 보다 관리 가능한 작은 규모로 분할할 수 있습니다.

관리 가능성을 위한 파티셔닝의 전형적 사용방법은 데이터 웨어하우스에서의 ‘롤링 윈도우 (rolling window)’ 로드 프로세스를 지원하는 것입니다. DBA가 새로운 데이터를 주간 단위로 로딩한다고 가정해 봅시다. 테이블은 range 파티셔닝되어 각각의 파티션이 1주일 분량의 데이터를 갖게 될 것입니다. 이 때 로딩 프로세스는 단순히 새로운 파티션을 추가하는 것입니다. 하나의 파티션을 추가하는 것은 DBA가 다른 어떤 파티션도 변경할 필요가 없기 때문에, 전체 테이블을 변경하는 것보다 훨씬 효율적입니다.

파티셔닝의 또다른 혜택은 데이터를 삭제할 때 나타나는데, 전체 파티션을 드롭시키는 방법으로 row 하나 하나를 삭제하는 것과 비교해 훨씬 효율적이고 빠릅니다.

성능을 위한 파티셔닝

파티셔닝은 검증하거나 운영할 데이터의 양을 제한함으로써 다음을 포함하여 많은 성능상의 혜택을 제공합니다:

- Partitioning Pruning: Partitioning pruning (일명 파티션 제거) 은 파티셔닝을 사용하여 성능을 향상시키는 가장 간단하면서 중요한 방법입니다. Partition pruning은 몇 가지 중요도 순에 따라 쿼리 성능을 향상시킵니다. 예를 들어, 애플리케이션이 과거 주문 기록을 갖고 있는 ORDERS 테이블을 포함하고 있고, 이 테이블이 주별 기준으로 파티셔닝되어 있다고 합시다.

특정한 어느 주의 주문 기록을 요청하는 쿼리는 ORDERS 테이블의 특정 파티션에만 접근할 것입니다. 만약 ORDERS 테이블이 2년간의 기록을 가지고 있다면, 이 쿼리는 104개의 파티션이 아니라 하나의 파티션에 접근합니다, 따라서 이 쿼리는 partition-pruning 덕분에 잠재적으로 약 100배 빠른 실행 속도를 낼 수 있게 되는 것입니다. Partition pruning은 오라클의 다른 기능들과 함께 사용됩니다. 오라클은 모든 인덱싱 기술, 결합 기술 또는 병렬 접근 방법과 함께 partition pruning을 활용합니다.

- Partition-wise Joins: 파티셔닝은 또 partition-wise join이라 불리는 기술을 사용하여 멀티 테이블 결합 성능을 향상시킵니다. Partition-wise joins는 2개의 테이블이 결합되고, 이들 중 적어도 하나의 테이블이 결합 키로 파티셔닝될 때 적용할 수 있습니다. Partition-wise joins는 대규모 결합을 결합 테이블과 '동일한' 데이터 세트의 작은 결합으로 분할합니다. 여기서 '동일'하다는 것은 결합된 양 측 파티셔닝 키 값 세트의 정확한 일치율을 의미하는 것으로, 이같은 "동일" 데이터 세트의 결합만이 결과를 생성할 수 있으며 다른 데이터 세트는 고려할 필요가 없다는 것을 의미합니다. 오라클은 이미 (물리적) equi 파티셔닝된 데이터 세트의 팩트를 사용하여 결합을 하거나, 런타임에 테이블을 투명하게 재배포 ("repartitioning") 하여 다른 테이블의 파티셔닝과 매칭되는 equi 파티셔닝된 데이터를 생성함으로써 보다 짧은 시간에 결합을 전반적으로 완성합니다. 이는 직렬 및 병렬 실행 모두에 성능상의 혜택을 제공합니다.

가용성을 위한 파티셔닝

파티셔닝된 데이터베이스 객체들은 파티션 독립적으로 이는 고가용성 전략에서 중요한 의미를 갖습니다. 예를 들어, 테이블 내 한 파티션이 사용 불가하더라도, 나머지 다른 파티션은 온라인으로 남아 있어 사용 가능합니다. 따라서 애플리케이션은 해당 파티션에 상관 없이 계속해서 쿼리 및 트랜잭션을 실행할 수 있습니다. 데이터베이스 또한 사용 불가능한 파티션을 필요로 하는 경우가 아니라면 계속 운영될 것입니다.

데이터베이스 관리자는 각각의 파티션이 각각의 테이블스페이스에 저장되도록 지정할 수 있습니다, 이는 관리자가 테이블의 다른 파티션과 무관하게 각각의 파티션에 백업 및 복구 작업을 할 수 있다는 것을 의미합니다. 따라서 재난이 발생하는 경우, 데이터베이스는 활성 데이터만으로 구성된 파티션을 가지고 복구되고 다른 파티션에 있는 비활성 데이터는 추후 편리한 시간에 복구될 수 있습니다. 따라서 시스템 다운타임이 단축됩니다.

또한, 파티셔닝은 예상된 다운타임을 단축합니다. 파티셔닝이 제공하는 이같은 성능 향상을 통해 DBA는 상대적으로 작은 배치 (batch) 창에서 대규모 데이터베이스 객체에 대한 유지 활동을 수행할 수 있습니다.

파티셔닝 모델링

Oracle Database 11g는 가장 포괄적인 파티셔닝 전략 세트를 제공함으로써, 데이터 분할과 실제 비즈니스 요구를 적절하게 연계할 수 있도록 합니다. 모든 가능한 파티셔닝 전략은 단일 (onelevel) 또는 복합 파티션 테이블을 위해 사용될 수 있는 기본적인 데이터 배포에 의존합니다. 게다가, 오라클은 다양한 파티셔닝 확장자를 제공하여, 파티셔닝 키 선택의 유연성을 확대하고, 필요시 자동 파티션 생성을 가능하게 하고, 파티셔닝 되지 않은 객체에 대한 파티셔닝 전략을 조언합니다.

파티셔닝 기본 전략

Oracle Partitioning은 다음과 같이 실제로 어떻게 데이터를 다양한 개별 파티션에 위치하도록 할 것인가를 통제하는 3가지 기본적인 데이터 배포 방법을 제공합니다.

- Range: 데이터가 파티셔닝 키 값의 범위에 따라 배포됩니다. (파티셔닝 키가 날짜 칼럼이라면, 'January2007' 파티션은 '01-JAN-2007' and '31-JAN-2007' 사이의 파티셔닝 키 값을 가진 row를 포함합니다) 데이터 배포는 빠짐 없이 연속적으로 이루어지며, 범위의 하부 경계는 앞선 범위의 상부 경계에 의해 정의됩니다.
- List: 데이터 배포는 파티셔닝 키 값의 목록에 의해 정의됩니다. (파티셔닝 키 값이 지역이라면, 'North America' 파티션은 'Canada', 'USA' 및 'Mexico'를 포함하게 될 것입니다) 특별 'DEFAULT' 파티션은 목록의 어떤 것에 의해서도 명시적으로 정의되지 않는 파티션 키의 모든 값을 정의하기 위해 사용될 수 있습니다.
- Hash: 해시 알고리즘은 주어진 row를 위한 파티션을 결정할 파티셔닝 키에 적용될 수 있습니다. 앞의 2가지 데이터 배포 방법과 달리, 해시는 데이터와 파티션 사이에 아무런 논리적 매핑도 제공하지 않습니다.

지금까지 말한 데이터 배포 방법을 사용하면, 테이블은 단일 또는 복합 파티션 테이블로 파티셔닝될 수 있습니다.

- 단일 (one-level) 파티셔닝: 테이블은 하나 이상의 칼럼을 파티셔닝 키로 사용하여, 데이터 배포 방법을 지정함으로써 정의됩니다. 예를 들면, 파티셔닝 키로 수 칼럼을 가진 테이블과 'less_than_five_hundred' 및 'less_than_thousand'의 2개의 파티션이 있다면, 'less_than_thousand' 파티션은 다음과 같은 조건이 참인 row들을 포함합니다: $500 \leq \text{Partitioning key} < 1000$.

Range, List 및 Hash 파티셔닝된 테이블을 지정할 수 있습니다.

- Composite Partitioning: 2가지 데이터 배포 방법의 결합은 복합 파티셔닝된 테이블을 정의하기 위하여 사용됩니다. 먼저, 테이블은 첫 번째 데이터 배포 방법에 의해 파티셔닝된 후 다시 각각의 파티션이 두 번째 배포 방법에 의해 서브파티션으로 분할됩니다.

주어진 파티션을 위한 모든 서브파티션은 데이터의 논리적 서브세트를 나타냅니다. 예를 들어, range-hash 복합 파티션 테이블은 먼저 range 파티셔닝 된 후, 다시 각각의 개별 range 파티션이 hash 파티셔닝을 통해 서브파티션으로 됩니다.

가용한 복합 파티셔닝 기술로는 range-hash, range-list, range-range, list-range, list-list, and list-hash 등이 있습니다. 인덱스 정렬 테이블 (IOTs)은 range, hash 및 list 파티셔닝을 사용하여 파티셔닝될 수 있습니다. 복합 파티셔닝은 IOT를 위해 지원되지 않습니다.

파티셔닝 확장자

파티셔닝 기본 전략 외에도, 오라클은 파티셔닝 확장자를 제공합니다. Oracle Database 11g에서의 확장자는 주로 2가지 목적에 초점을 두고 있습니다.

- (a) 파티션 테이블의 관리 가능성을 획기적으로 향상
- (b) 파티셔닝 키 정의의 유연성 확대

확장자는 다음과 같습니다.

Interval Partitioning: Oracle Database 11g의 새로운 파티셔닝 전략인 Interval 파티셔닝은 range 방법의 성능을 확장자, 인터벌 정의를 사용하여 equi 파티셔닝된 범위를 정의합니다. 오라클은 개별 범위를 명시적으로 정의하지 않고, 파티션을 위한 데이터가 최초에 입력될 때마다 필요에 따라 파티션을 자동 생성합니다. Interval

파티셔닝은 파티션 테이블의 관리 가능성을 획기적으로 향상시킵니다. 예를 들면, interval 파티셔닝된 테이블은 오라클이 매월 새로운 파티션을 생성할 수 있도록 정의될 수 있습니다. 그러면 2007년 9월의 최초 기록이 데이터베이스에 입력되는 즉시, 'September 2007' 파티션이 자동 생성됩니다.

Interval 파티션 테이블을 위한 가용 기술은 Interval, Interval-List, Interval-Hash 및 Interval-Range 입니다.

REF 파티셔닝: Oracle Database 11g는 기존 모-자 관계를 활용하여 테이블을 파티셔닝하는 것을 허용합니다. 모테이블의 파티셔닝 전략은 모의 파티셔닝 키 칼럼을 자테이블로 저장하지 않고도, 자테이블로 전수됩니다. REF 파티셔닝을 사용하지 않고 모-자간 동일 파티셔닝 전략을 채택하려면, 모테이블의 모든 파티셔닝 키 칼럼을 자테이블로 복제해야만 합니다. 반면, REF 파티셔닝은 파티셔닝 키 칼럼을 저장하지 않고, 논리적 데이터 모델에 따라 자연스럽게 테이블을 파티셔닝하도록 지원함으로써, 비정규화 (denormalization) 및 공간 절약을 위한 비용을 절감할 수 있도록 합니다. REF 파티셔닝은 또한, 테이블의 논리적 형태를 변화시키는 모든 파티션 유지 활동을 투명하게 모테이블에서 자테이블로 전수합니다. 게다가, REF 파티셔닝은 모 및 자테이블의 equi 파티셔닝을 위한 partition-wise joins를 가능하게 함으로써, 유지 활동 성능을 향상시킵니다.

예를 들면, 모의 ORDERS테이블은 ORDER_DATE

칼럼에Range파티셔닝되었고, 자의 ORDER ITEMS 테이블은ORDER_DATE

칼럼을 가지고 있지만ORDERS 테이블을 참고하여 파티셔닝될 수 있습니다.
ORDERS

테이블이 월별로 파티셔닝되었다면, '2007년 1월' 주문의 내역은 ORDER ITEMS 테이블의 단일 파티션에 저장되고, 모의ORDERS 테이블에equi 파티셔닝됩니다. 만약, 'Feb-2007' 파티션이 ORDERS 테이블에 추가된다면, 오라클은 해당 파티션을 ORDER ITEMS 테이블에 투명하게 추가할 것입니다.

모든 파티셔닝 전략이 REF 파티셔닝을 위해 사용될 수 있습니다.

가상 칼럼 기반 파티셔닝: 예전 오라클 버전에서는, 파티셔닝 키가 물리적으로 테이블에 존재해야만 테이블이 파티셔닝될 수 있었습니다. Oracle Database 11g의 새로운 기능인 가상 칼럼은 이같은 한계를 극복하여 테이블에 존재하는 하나 이상의 기존 칼럼을 사용하여 파티셔닝 키가 expression에 의해 정의되도록 하고, expression을 메타데이터로만 저장할 수 있도록 하였습니다.

virtualcolumns에 파티셔닝 전략을 정의하는 것이 가능할 정도로 파티셔닝 성능이 강화되면서, 비즈니스 요구를 보다 포괄적으로 충족시키는 것이 가능해졌습니다. 칼럼이 정보로 과부하되는 것은 예외적인 것이 아닙니다. 예를 들어10자리 계정 ID는 앞의 3자리에 계정 브랜치 정보를 포함할 수 있습니다. 가상 칼럼 기반 파티셔닝의 확장자를 사용하면, ACCOUNT_ID 칼럼을 포함하는 ACCOUNTstable은 이 스테이블의 파티셔닝 키가 되는 ACCOUNT_ID의 최초 3자리로부터 파생된 가상 (파생) 칼럼ACCOUNT_BRANCH를 통해 확장될 수 있습니다.

가상 칼럼 기반 파티셔닝은 모든 파티셔닝 기본 전략에 지원됩니다.

Partition Advisor

Oracle Database 11g의 SQL Access Advisor는 성능 강화로 인덱스, 머티리얼라이즈드 뷰 및 머티리얼라이즈드 뷰 로그를 위해 이미 지원해 온 것 이외 파티셔닝 권장자를 생성합니다. SQL Access Advisor에 의해 생성된 권장자는, 파티셔닝만을 위한 것이든, 전체를 위한 것이든, 실행을 통해 나타날 향상된 성능을 보여줍니다. 생성된 스크립트는 수동으로 실행될 수도 있고, Oracle Enterprise Manager내부 큐로 제출될 수도 있습니다. 파티셔닝 디바이스 확장자를 통해, 특히 파티셔닝을 위한 권장자는 물론SQL Access Advisor의 보다 포괄적인 전체 권장자를 통하여 SQL 스테이트먼트 전반의 종합적 성능을 향상시킬 수 있습니다.

SQL Access Advisor에 통합된Partition Advisor는 라이선스 옵션인 Oracle Tuning Pack의 일부입니다. 이는 Enterprise Manager 내부 혹은 커맨드 라인 인터페이스를 통해 사용 가능합니다.

파티셔닝 전략 및 확장자 개요

다음의 테이블은 Oracle Database 11g에서 사용 가능한 모든 파티셔닝 기본 전략의 개념적 개요를 나타냅니다:

파티셔닝 전략	데이터 분류	비즈니스 사례
Range 파티셔닝	값의 연속 범위에 기반.	· 주문-날짜별 range 파티셔닝된 주문 테이블
List 파티셔닝	주문되지 않은 값 목록에 기반.	· 국가별 list 파티셔닝된 주문 테이블
Hash 파티셔닝	해시 알고리즘에 기반.	· 고객_id별 hash 파티셔닝된 주문 테이블
복합 파티셔닝 · Range-Range · Range-List · Range-Hash · List-List · List-Range · List-Hash · Interval-Range * · Interval-List * · Interval-Hash *	Range, List, Hash 및 Interval 파티셔닝의 두가지 조합에 기반	· 주문-날짜별 range 파티셔닝 및 고객-id별 hash 서브파티셔닝된 주문 테이블 · 주문-날짜별 range 파티셔닝 및 Range-Hash 선적-날짜별 range 서브파티셔닝된 주문 테이블

이같은 파티셔닝 전략에 더해, Oracle Database 11g는 다음의 확장 파티션을 제공합니다.

파티셔닝 전략	데이터 분류	비즈니스 사례
Interval 파티셔닝	동일 간격의 interval에 의한 정의. 최초 파티션이 예외적인 경우, 필요시 알맞은 데이터가 나타나면 모든 파티션이 자동으로 생성. Range 파티셔닝의 확장.	· 2007년 1월 1일부터 시작하여 미리 정의된 1일 간격의 interval을 갖는 주문-날짜별로 파티셔닝된 주문 테이블
REF 파티셔닝	primary-foreign 키 관계를 통하여 모테이블에서 파생된 자테이블을 위한 파티셔닝. 파티셔닝 키는 자테이블의 실제 칼럼에 저장되지 않음.	· 주문-날짜별로 range 파티셔닝된 (모)테이블과 파티셔닝 기술은 (자) 주문 라인 테이블 주문-날짜 칼럼은 모의 주문 테이블에서만 보임
가상 칼럼 기반 파티셔닝	파티셔닝은 앞에서 밝힌 기술 중 하나로 정의되고 파티셔닝 키는 가상 칼럼에 기반. 가상 칼럼은 디스크에 저장되지 않고 메타데이터로만 존재.	· 고객 번호를 기반으로 판매 지역을 분류하는 가상 칼럼을 가진 테이블. 주문 테이블은 판매지역별로 list 파티셔닝됨.

Oracle Partitioning을 통한 정보 라이프사이클 관리

Oracle Partitioning을 사용하면, 최소한의 비용으로 최대한의 데이터를 저장해야 하는 오늘날의 도전 과제를 적절하게 해결할 수 있습니다. 데이터가 어떻게 접근되는지를 이해한다면, 개별 파티션의 독립성은 “티어드 아키아빙” 전략의 온라인 영역을 충족시키는 핵심 구동자임을 알게 될 것입니다. 특히 과거 데이터를 포함하고 있는 테이블에서, 파티셔닝은 개별 파티션 (혹은 파티션 그룹)들이 서로 다른 티어에 저장되도록 함으로써, 서로 다른 물리적 속성 및 가격 포인트를 제공합니다. 예를 들면, 2년간의 데이터를 갖고 있는 주문 테이블은 고가의 하이엔드 스토리지 티어에 최근 분기 데이터만을 저장하고, 나머지 테이블 (데이터의 약 90%)은 저렴한 스토리지 티어에 저장할 수 있습니다. Oracle Partitioning을 사용하면, 스토리지 비용은 최종 사용자 접근에 영향을 미치지 않고도, 대폭 절감 (50% 이상의 비용 절감도 가능)할 수 있어 저장 정보를 위한 TCO를 최적화합니다.

OTN 에서 무료로 다운받을 수 있는 툴인 Oracle ILM Assistant는 이같은 비용 절감을 보여주고, 테이블 파티셔닝 방법을 알려주며, 파티션을 다른 스토리지 티어로 이동할 시간을 옮길 조언합니다.

결론

Oracle Partitioning의 새롭고 향상된 기능을 고려한다면, Oracle Database 11g는 1997년 Oracle Partitioning 발표 이래 가장 중요한 제품입니다. 오라클은 매번 주요 제품을 내놓을 때마다, 새로운 파티셔닝 기술을 추가하든지, 확장성을 강화하든지 아니면 관리 가능성 및 관리 기능을 확장하든지 하는 방법으로 파티셔닝 기능을 강화해 왔습니다. 오라클은 새로운 파티셔닝 기술을 계속 추가하여 모든 비즈니스 요구를 충족시키는 최적의 파티셔닝 기술을 만들어 나갈 것입니다.

파티셔닝은 모두를 위한 것입니다. Oracle Partitioning은 거의 모든 데이터베이스 애플리케이션의 관리 가능성, 성능 및 사용성을 획기적으로 향상시킵니다. 파티셔닝은 첨단 애플리케이션에 적용될 수 있으며 이들 애플리케이션의 성공을 보장하는 핵심 기술입니다. 파티셔닝은 또한, 광범위한 일반 데이터베이스 애플리케이션에도 적용되어 이들 애플리케이션의 관리를 간소화하고 비용을 절감하도록 해줍니다. 파티셔닝은 애플리케이션에 투명하므로, 고가의 시간 소모적 애플리케이션 변경 없이 쉽게 구현할 수 있습니다.



2007년 6월

저자: Hermann Baer

공동 저자:

Oracle Corporation
World Headquarters
500 Oracle Parkway
Redwood Shores, CA 94065

U.S.A.

월드와이드 문의:

Phone: +1.650.506.7000

Fax: +1.650.506.7200

oracle.com

Copyright © 2007, Oracle. All rights reserved.

본 문서는 정보 제공만을 목적으로 제공되며, 문서의 내용은 별도 공지 없이 변경될 수 있습니다. 본 문서에는 오류가 포함되어 있을 수 있으며, 상업성 또는 특정 목적의 규제 준수에 대한 명시적, 암시적인 일체의 보장을 제시하지 않고 있습니다. 오라클은 본 문서에 관련하여 직접적/간접적으로 발생하는 일체의 법적 책임 또는 계약상의 의무를 거부합니다. 본 문서는 오라클의 사전 서면 승인 없이 어떤 목적, 어떤 방법으로도 전자적/기계적인 형태로 복제, 전송될 수 없습니다.

Oracle은 Oracle Corporation 및/또는 계열사의 등록 상표입니다. 다른 이름은 해당 업체의 상표일 수 있습니다.