

참고: 본 문서는 정보 제공만을 목적으로 제공됩니다. 본 문서는 일체의 자료, 코드, 또는 기능을 보장하지 않으며, 본 문서의 정보에 의존하여 구매 결정을 내려서도 안됩니다. 본 문서에서 설명된 오라클 제품 또는 기능의 개발, 출시, 기능 제공 시점 등은 전적으로 오라클의 재량에 의해 결정됩니다.

ORACLE PARTITIONING

ORACLE PARTITIONING

- 8 세대에 걸쳐 검증된 기능
- 업계에서 가장 광범위하고 포괄적인 솔루션
- 파티셔닝을 이용한 다양한 애플리케이션의 성능, 가용성, 관리성 개선
- 데이터베이스의 정보 생명 주기 관리를 위한 핵심 테크놀로지: 온라인 “티어 기반 아카이빙”을 통해 TCO 를 극적으로 절감
- 애플리케이션 변경 작업 없이 구현 가능

Oracle Partitioning은 다양한 애플리케이션의 관리성, 성능, 가용성을 개선하기 위한 Oracle Database 11g Enterprise Edition의 옵션입니다. 파티셔닝을 이용하여 테이블, 인덱스, IOT(index-organized table)을 보다 작은 단위로 분할하고, 한층 세분화된 레벨의 데이터베이스 오브젝트 관리 및 접근을 지원할 수 있습니다. 오라클은 비즈니스 요구 사항의 해결을 위한 다양한 파티셔닝 방식을 포괄적으로 제공하고 있습니다. 또 파티셔닝 기능이 SQL 구문의 관점에서 완전히 투명하게 구현되므로, OLTP에서 데이터 웨어하우징에 이르기까지 어떤 애플리케이션에도 파티셔닝을 적용할 수 있습니다.

Oracle Partitioning의 혜택

파티셔닝은 다양한 애플리케이션 환경에서 관리성, 성능, 가용성을 극적으로 개선하는 효과를 제공합니다. 8번째 세대로 진화한 Oracle Partitioning은 매우 엄격한 고 가용성 요구 사항을 갖는 테라바이트급 시스템 환경에서 그 유용성을 검증 받아 왔습니다.

실제로 파티셔닝을 이용하여 쿼리 또는 유지보수 작업의 성능을 몇 배로 개선하는 경우를 드물지 않게 확인할 수 있습니다. 또 파티셔닝은 오래된 정보를 저가형 스토리지 디바이스에 저장하는 접근법인 “티어 기반 아카이빙(tiered archiving)”을 통해 총소유비용을 극적으로 절감할 수 있게 합니다. Oracle Partitioning은 대규모 환경의 정보 생명주기 관리를 위한 효과적이면서도 단순하고, 동시에 매우 강력한 접근법으로 활용될 수 있습니다.

Oracle Partitioning의 기본 기능

파티셔닝을 이용하면 테이블, 인덱스 또는 IOT(index-organized table)을 더 작은 단위로 분할할 수 있습니다. 이렇게 분할된 데이터베이스 오브젝트의 조각들을 파티션이라 부릅니다. (또는 컴포지트 파티션이 사용되는 경우에는 서브파티션이라 불립니다.) 각각의 파티션은 별도의 이름을 가지며, 서로 다른 스토리지 특성을 가질 수도 있습니다. 예를 들어 특정 파티션의 테이블 압축 기능을 활성화하거나, 다른 테이블스페이스에 저장하거나, 다른 ASM 디스크 그룹에 배치할 수 있습니다. 데이터베이스 관리자의 관점에서 볼 때, 파티션 된 오브젝트는 일괄적으로 또는 개별적으로 관리가 가능한 여러 개의 조각으로 구성되며, 따라서 오브젝트에 대한 관리 유연성을 크게 개선할 수 있습니다. 하지만 애플리케이션의 관점에서 볼 때에는 파티션

된 테이블은 파티션 되지 않은 테이블과 아무런 차이가 없습니다. 그러므로 애플리케이션 변경 작업은 요구되지 않습니다.

테이블은 '파티셔닝 키(partitioning key)' 을 통해 분할됩니다. 파티셔닝 키란 특정 로우가 어떤 파티션에 위치하는지 정의하는 일련의 컬럼을 말합니다. Oracle Database 11g는 업계에서 가장 포괄적인 파티셔닝 기능과 옵션, 그리고 최신 메커니즘을 제공하고 있습니다. 또 Partition Advisor를 이용하여 데이터 접근 방법에 따른 테이블 파티션 방안의 권고 사항을 확인할 수 있습니다.

아래 표는 Oracle Database 11g에서 제공되는 기본적인 파티셔닝 전략에 대해 설명하고 있습니다.

파티셔닝 전략	데이터 분할	샘플 비즈니스 케이스
Range Partitioning	연속적인 값의 영역을 기준으로 분할	· order_date를 기준으로 Orders 테이블을 파티셔닝
List Partitioning	일정한 순서가 없는 값을 기준으로 파티셔닝	· country를 기준으로 Orders 테이블을 파티셔닝
Hash Partitioning	해시 알고리즘 기반	· customer_id를 기준으로 Orders 테이블을 파티셔닝
Composite Partitioning o Range-Range o Range-List o Range-Hash o List-List o List-Range o List-Hash o Interval-Range * o Interval-List * o Interval-Hash *	위에서 설명된 Range, List, Hash, Interval Partitioning 중 두 가지 테크닉을 조합하여 사용	· Orders 테이블에 대해 order_date 기준으로 Range 파티셔닝을, Customer_id를 기준으로 Hash 서브파티셔닝을 수행 · Orders 테이블에 대해 order_date를 기준으로 Range 파티셔닝을, 다시 shipment_date를 기준으로 Range 서브파티셔닝 수행

위에서 설명된 파티셔닝 전략 이외에도, Oracle Database 11g는 다음과 같은 파티셔닝 익스텐션을 제공합니다.

파티셔닝 전략	데이터 분할	샘플 비즈니스 케이스
Interval Partitioning	동일한 크기의 간격을 기준으로 파티셔닝을 정의. 첫 번째 생성되는 파티션을 제외한 모든 파티션은 조건에 맞는 데이터가 입력되는 시점에 자동으로 생성됨. Range Partitioning의 확장	· Orders 테이블을 order_date 기준으로 '01-Jan-2007'로부터 시작해서 일정한 기간으로 파티셔닝

REF Partitioning	기본 키-외래 키 관계를 통해 자식 테이블 파티셔닝을 부모 테이블로부터 상속. 파티셔닝 키는 자식 테이블의 실제 컬럼에 저장되지 않음.	· (부모) Orders 테이블을 order_date 기준으로 Range Partitioning 수행. (자식) order_lines 테이블은 부모로부터 동일한 파티셔닝 방식을 상속. order_date 컬럼은 부모 테이블 (orders)에만 존재함.
Virtual column based Partitioning	위에서 설명된 파티셔닝テクニック 중 하나를 사용하되, 파티셔닝 키는 가상 컬럼 (virtual column)을 기반으로 설정. 가상 컬럼은 디스크에 저장되지 않으며 메타데이터의 형태로만 존재.	· Orders 테이블에 고객 계좌 번호를 기반으로 sales region을 유추하는 가상 컬럼을 설정. 그런 다음 sales region을 기준으로 orders 테이블의 List Partitioning을 수행.

IOT (index-organized table)에는 Range Partitioning, List Partitioning, Hash Partitioning이 적용될 수 있습니다. Oracle Database 11g은 또 세 가지 유형의 파티션된 인덱스를 지원합니다.

Local Index: Local Index란 파티션된 기저 테이블과 동일한 방법으로 파티션된 인덱스를 의미합니다. 각 로컬 인덱스의 파티션은 기저 테이블의 특정 파티션에 1대 1로 대응합니다.

Global Partitioned Index: Global Partitioned Index는 테이블의 다른 파티셔닝 키를 이용하여 파티션된 파티션된 테이블 또는 파티션되지 않은 테이블의 인덱스입니다. Global Partitioned Index는 Range Partitioning만을 지원합니다.

Global Non-Partitioned Index: Global Non-Partitioned Index는 기본적으로 파티션되지 않은 테이블의 인덱스와 동일합니다. 인덱스 구조는 파티션되지 않습니다.

오라클은 파티셔닝 테이블, 인덱스, IOT 등을 위한 다양하고 강력한 대안을 제공하며, 어떤 비즈니스 환경에서든 애플리케이션에 최적화된 파티셔닝 기술을 활용할 수 있습니다.

그 밖에도, 오라클은 파티션의 추가, 드롭, 삭제, 분할, 병합, 이동, 압축 등 파티셔닝 테이블의 관리를 위한 포괄적인 SQL 커맨드 셋을 제공합니다.

관리성 개선을 위한 Oracle Partitioning의 활용

Oracle Partitioning 옵션을 이용하여 테이블과 인덱스를 보다 작은 단위로 분할함으로써 데이터 관리 환경을 개선하는 것이 가능합니다.

파티셔닝 환경에서는 테이블의 작은 조각들을 단위로 유지 보수 작업을 수행할 수 있습니다. 예를 들어, 전체 테이블을 백업하는 대신 테이블의 파티션 중 하나만을 백

업할 수 있을 것입니다. 전체 데이터베이스 오브젝트가 아닌 파티션 단위로 작업을 실행함으로써 유지보수 작업의 관리 편의성을 한층 개선하는 것이 가능합니다.

관리성 개선을 위해 파티셔닝을 활용하는 대표적인 경우로 데이터 웨어하우스의 '롤링 윈도우' 로드 프로세스를 예로 들 수 있습니다. DBA가 1 주일 간격으로 테이블에 새로운 데이터를 로드하는 상황을 가정해 봅시다. 이 테이블은 Range Partitioning이 적용되어 있으며 각 파티션은 1 주일 분의 데이터를 저장하고 있습니다. 따라서 로드 과정에서는 단순히 새로운 파티션을 추가하기만 하면 됩니다. 전체 테이블을 수정하는 것보다 새로운 파티션을 하나 추가하는 것이 DBA에게는 훨씬 편리할 것입니다. 파티션 된 테이블의 데이터를 삭제하는 경우도 마찬가지입니다. DELETE 커맨드를 사용하여 테이블 내의 데이터를 일일이 삭제하는 대신 파티션 하나를 드롭 처리함으로써 작업을 훨씬 쉽고 빠르게 완료할 수 있습니다.

성능 개선을 위한 Oracle Partitioning의 활용

데이터의 규모가 계속적으로 증가하면서 시스템 성능이 저하되는 문제가 자주 제기되곤 합니다. Oracle Partitioning을 이용하여 조회 대상 데이터의 수를 제한함으로써, 파티션 되지 않은 테이블에서는 성취하기 어려운 수준의 성능 개선 효과를 확인할 수 있습니다. Oracle Partitioning 옵션이 성능 측면에서 제공하는 효과가 아래와 같습니다.

Partitioning Pruning: Partitioning Pruning은 파티셔닝을 이용하여 성능을 개선하기 위한 가장 단순하면서도 가장 확실한 방법입니다. 애플리케이션이 사용하는 Shipments 테이블에 배송 히스토리 기록이 저장되어 있고, 이 테이블을 매일 단위로 파티셔닝 된다고 가정해 봅시다. 하루 동안의 배송 내역을 조회하는 쿼리는 Shipments 테이블 내에서 단 하나의 파티션에만 접근할 것입니다. Shipments 테이블에 2 년 간의 히스토리 데이터가 저장되어 있다면, 쿼리는 730 개의 파티션이 아닌 단 하나의 파티션만을 접근하는 셈입니다. 결과적으로, Partition Pruning을 이용함으로써 700 배 가량의 성능 개선 효과를 얻을 수 있습니다. Partition Pruning은 오라클의 다른 성능 관련 기능과 효과적으로 연동됩니다. 오라클은 인덱싱, 조인, 병렬 액세스 등의 테크닉에서 Partition Pruning을 활용할 수 있도록 지원할 예정입니다.

Partition-wise Join: Partition-wise Join을 이용하여 멀티-테이블 조인 작업의 성능을 개선할 수 있습니다. Partition-wise Join은 서로 다른 두 개의 테이블을 조인 처리함으로써 적용되며, 두 테이블은 조인 키(join key)를 기준으로 파티셔닝 됩니다. Partition-wise Join은 대규모 조인 작업을 보다 작은 크기로 분할함으로써 조인 실행 속도를 개선해 줍니다. 이 테크닉은 순차적, 병렬적 실행 모델에서 모두 뛰어난 효과를 제공합니다.

가용성 개선을 위한 Oracle Partitioning의 활용

파티션 된 데이터베이스 오브젝트는 파티션 독립성(partition independence)를 제공합니다. 파티션 독립성은고가용성 전략에서 매우 중요한 비중을 차지하는 개념입니다. 테이블의 특정 파티션을 사용할 수 없고 나머지 파티션은 모두 온라인 상태인 경우를 예로 들어 봅시다. 애플리케이션은 파티션 된 테이블에 여전히 쿼리와 트랜

관련 제품 및 서비스

- Oracle Partitioning은 Oracle Database 11g Enterprise Edition의 옵션으로 제공됩니다.

작업을 실행할 수 있습니다. 데이터베이스 작업이 사용 불가능한 파티션에 접근하지 않는 이상, 모든 작업은 성공적으로 실행됩니다.

또, 파티셔닝을 통해 예정된 다운타임을 단축할 수 있습니다. 파티셔닝이 제공하는 성능 효과로 인해, 데이터베이스 관리자가 수행하는 유지보수 작업은 상대적으로 짧은 시간 안에 완료될 수 있습니다.

Oracle Partitioning을 이용한 정보 생명주기 관리

Oracle Partitioning은 가능한 한 저렴한 비용으로 대량의 데이터를 관리해야 하는 오늘날의 환경에서 최적화된 대안을 제공합니다. 개별 파티셔닝의 독립성은 “계층화된 아카이빙(tired archiving)” 전략을 위한 핵심 기반으로 활용됩니다. 특히 히스토리 데이터를 저장한 테이블의 경우, 각각의 파티션(또는 파티션 그룹)을 서로 다른 스토리지 계층에 저장함으로써 각 파티션에 적합한 물리적 특성과 비용효율성을 제공할 수 있습니다. 예를 들어, 2년 간의 데이터를 저장한 Orders 테이블에서 최근 분기의 데이터만을 고가의 하이엔드 스토리지 tier에 저장하고 나머지 (약 90%)의 데이터는 저가형 스토리지 tier에 저장하는 방법을 사용할 수 있습니다. Oracle Partitioning을 활용함으로써, 엔드 유저에게 제공되는 서비스에 영향을 미치지 않은 상태에서 스토리지 비용을 극적으로 절감하고(50% 이상의 비용 절감 사례도 쉽게 찾아 볼 수 있습니다), 저장된 정보의 총소유비용을 최적화하는 것이 가능합니다.

Oracle Partitioning의 범용성

Oracle Partitioning을 이용하여 거의 모든 종류의 데이터베이스 애플리케이션의 관리성, 성능, 가용성을 혁신적으로 개선할 수 있습니다. 파티셔닝을 최신 애플리케이션에 적용하고, 애플리케이션의 성공적 구현을 보장하기 위한 핵심 테크놀로지로 활용하는 것이 가능합니다. 또 한편으로, 관리 환경을 단순화하고 비용을 절감하기 위한 목적에서 일반 데이터베이스 애플리케이션에 파티셔닝을 적용할 수도 있습니다.

파티셔닝 기능의 개선이라는 관점에서 볼 때, Oracle Database 11g는 1997년 Oracle Partitioning이 처음 소개된 이후 가장 의미 있는 버전으로서 출시되었습니다. 오라클은 오라클 데이터베이스의 새로운 메이저 버전이 발표될 때마다 새로운 파티셔닝 테크닉을 추가하고, 확장성을 개선하고, 관리성을 향상시키는 등 Oracle Partitioning의 기능을 개선해 왔습니다. 오라클은 향후에도 새로운 파티셔닝 테크닉을 추가적으로 구현함으로써 고객의 비즈니스 요구 사항에 최적화된 환경을 지원할 예정입니다.

Copyright 2007, Oracle. All Rights Reserved.

본 문서는 정보 제공만을 목적으로 제공되며, 문서의 내용은 사전 공지 없이 변경될 수 있습니다. 본 문서에는 오류가 포함되어 있을 수 있으며, 상업성 또는 특정 목적의 부합성에 대한 명시적, 묵시적인 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.

오라클은 Oracle Corporation 및 계열사의 공식등록상표입니다. 다른 이름은 해당 업체의 공식등록상표일 수 있습니다.