집합 연산자 개념서

이 문서는 SQLD 시험을 위한 집합 연산자 개념서입니다. 집합 연산자의 기본 개념, 종류, 사용 규칙, 실전 활용 예제, 성능 고려사항, 주의사항 및 제약사항, 그리고 SQLD 시험 대비 팁을 다룹니다. 집합 연산자는 두 개 이상의 쿼리 결과를 하나로 통합하는 중요한 SQL 기능입니다.

1. 집합 연산자의 기본 개념

1.1 정의

- 두 개 이상의 쿼리 결과를 하나의 결과로 통합하는 연산자
- 각각의 SELECT 문을 통해 생성된 결과를 세로 방향으로 결합
- 기본적인 집합 이론의 개념을 SQL에 적용

1.2 주요 특징

1. **SELECT문의 컬럼 개수 일치**
   - 비교되는 두 SELECT문의 컬럼 개수가 동일해야 함
2. **데이터 타입 일치**
   - 대응되는 컬럼의 데이터 타입이 동일하거나 변환 가능해야 함
3. **컬럼 이름**
   - 첫 번째 SELECT문의 컬럼 이름이 결과의 컬럼 이름이 됨

2. 집합 연산자의 종류

2.1 UNION

- 두 집합을 통합하고 중복을 제거
  자동으로 정렬 수행

```
SELECT 컬럼1, 컬럼2 FROM 테이블1
UNION
SELECT 컬럼1, 컬럼2 FROM 테이블2;
```

2.2 UNION ALL

- 두 집합을 통합하고 중복을 허용
  정렬을 수행하지 않음 (성능상 유리)

```
SELECT 컬럼1, 컬럼2 FROM 테이블1
UNION ALL
SELECT 컬럼1, 컬럼2 FROM 테이블2;
```

2.3 INTERSECT

- 두 집합의 교집합을 추출

```
SELECT 컬럼1, 컬럼2 FROM 테이블1
INTERSECT
SELECT 컬럼1, 컬럼2 FROM 테이블2;
```

2.4 MINUS (Oracle) / EXCEPT (SQL Server)

- 첫 번째 집합에서 두 번째 집합을 뺀 차집합

```
SELECT 컬럼1, 컬럼2 FROM 테이블1
MINUS
SELECT 컬럼1, 컬럼2 FROM 테이블2;
```

3. 집합 연산자 사용 규칙

3.1 기본 규칙

1. **컬럼 개수 일치**

```
-- 올바른 사용
SELECT 사원번호, 이름 FROM 사원1
UNION
SELECT 사원번호, 이름 FROM 사원2;

-- 잘못된 사용
SELECT 사원번호, 이름, 부서 FROM 사원1
UNION
SELECT 사원번호, 이름 FROM 사원2;
```

2. **데이터 타입 일치**

```
-- 올바른 사용
SELECT 사원번호, 이름 FROM 사원1
UNION
SELECT 사원번호, 이름 FROM 사원2;

-- 잘못된 사용
SELECT 사원번호, 입사일 FROM 사원1
UNION
SELECT 사원번호, 이름 FROM 사원2;
```

3. **ORDER BY 절 사용**

```
-- 전체 결과에 대한 정렬
SELECT 사원번호, 이름 FROM 사원1
UNION
SELECT 사원번호, 이름 FROM 사원2
ORDER BY 사원번호;
```

3.2 NULL 처리

- NULL 값도 하나의 데이터로 취급
- 집합 연산 시 NULL 값도 비교 대상에 포함

```
SELECT 부서번호, 매니저 FROM 부서1
UNION
SELECT 부서번호, NULL AS 매니저 FROM 부서2;
```

4. 실전 활용 예제

4.1 UNION 활용

```
-- 현재 사원과 퇴사한 사원의 전체 목록
SELECT 사원번호, 이름, '재직' AS 상태 FROM 현재사원
UNION
SELECT 사원번호, 이름, '퇴사' AS 상태 FROM 퇴사사원
ORDER BY 사원번호;
```

4.2 UNION ALL 활용

```
-- 월별 매출 데이터 통합 (중복 허용)
SELECT 월, 매출액 FROM 매출_2023
UNION ALL
SELECT 월, 매출액 FROM 매출_2024
ORDER BY 월;
```

4.3 INTERSECT 활용

```
-- 양쪽 테이블에 모두 존재하는 고객 찾기
SELECT 고객번호 FROM 온라인고객
INTERSECT
SELECT 고객번호 FROM 오프라인고객;
```

4.4 MINUS 활용

```
-- 온라인 전용 고객 찾기
SELECT 고객번호 FROM 온라인고객
MINUS
SELECT 고객번호 FROM 오프라인고객;
```

5. 성능 고려사항

5.1 UNION vs UNION ALL

UNION

- 중복 제거 작업 수행
- 정렬 작업 발생
- 상대적으로 느린 성능

UNION ALL

- 중복 제거 작업 없음
- 정렬 작업 없음
- 빠른 성능
- 중복 데이터가 없음이 보장될 때 사용 권장

5.2 성능 최적화 방안

1. **불필요한 컬럼 제거**

```
-- 필요한 컬럼만 선택
SELECT 사원번호, 이름 FROM 사원1
UNION ALL
SELECT 사원번호, 이름 FROM 사원2;
```

2. **WHERE 절을 통한 데이터 축소**

```
SELECT 사원번호, 이름 FROM 사원1 WHERE 부서 = '영업'
UNION ALL
SELECT 사원번호, 이름 FROM 사원2 WHERE 부서 = '영업';
```

6. 주의사항 및 제약사항

6.1 정렬 관련

- ORDER BY는 마지막 SELECT문에만 사용 가능
- 각각의 SELECT문에서는 ORDER BY 사용 불가

6.2 데이터 타입 변환

- 암시적 데이터 타입 변환이 가능한 경우도 있음
- 명시적 변환을 사용하는 것이 안전

6.3 집합 연산자 우선순위

- 괄호를 사용하여 연산 순서 명시 가능

```
(SELECT * FROM A UNION SELECT * FROM B)
INTERSECT
SELECT * FROM C;
```

SQLD 시험 대비 TIP

주요 출제 포인트

1. **집합 연산자의 종류와 특징**
   - UNION, UNION ALL, INTERSECT, MINUS의 차이
   - 각 연산자의 결과 예측
2. **제약사항 이해**
   - 컬럼 개수와 데이터 타입 일치 규칙
   - ORDER BY 사용 규칙
3. **성능 관련 특징**
   - UNION과 UNION ALL의 성능 차이
   - 최적화 방안

학습 전략

1. 각 집합 연산자의 기능과 차이점 이해
2. 실제 예제를 통한 결과 확인
3. 제약사항 숙지
4. 성능 관련 내용 이해