SQLD 알짜 요약

1. SQL 연산순서★★★

From
Where
Group by
Having
Select
Order by

DML

- Select, insert, update, delete

DDL

- Alter, create, modify, drop

TCL

- Rollback, commit

DCL

- Grant, revoke

2. Distinct

어떤 컬럼값들의 중복을 제거 한 결과를 출력한다.

- Select distinct col from table;
- Select distinct col1, col2 from table; 의 경우엔 col1 과 col2의 값이 모두 같지 않은 것 만 출력한다. <주의>
-

3. Alias★★

Select 절에서 사용가능, where 절에서는 사용 불가!!!
Select col as **name** from table; = select col **name** from table;

4. concat

Select col1 + col2 + col3 from table; (SQL Server)
Select col1 || col2 || col3 from table; (oracle)
Select concat(col1, col2) from table; *연산자가 2개!! 기억!!

5. 논리연산자

1) NOT - ~가 아니다
2) AND – A 그리고 B (둘다 만족)
3) OR – A 또는 B (둘중 하나만 만족해도 OK!)

6. SQL 연산자★

A between B and C – **B <= A <= C**

A in (1,2,3) – **A = 1 or A = 2 or A = 3**

A like '_ble*' – **A의 값중 2,3,4번째 값이 ble 인 모든 데이터 출력**

7. escape

and email like '@_%' escape '@'

*아무 문자나 가능

8. rownum, top

oracle에선 where절 옆에 rownum

SQL server의 경우 select 옆에 top

9. null의 정의★★★

모르는값, 정의되지 않은값 (공백이나 0 과는 다르다)

산술연산에서 null이 들어가게 되면 null이 출력된다.

*null + 2, null * 4, null + null 모두 결과는 null

조건절에 null이 들어가게되면 false를 반환 함.

*null=null, null=2

집계함수(sum, count, min, max…) 에서 null은 데이터 대상에서 제외 된다.

정렬시에는 오라클에서는 가장큰 것이 되고, SQL Server에서는 가장 작은 값이 된다.

Nvl(col, 0) – col이 널이면 0 반환 아니면 col 반환

Nvl2(col,1,0) – col이 null이면 0 반환, 아니면 1 반환

Isnull(col,0) – col이 널이면 0 반환 아니면 col 반환

Nullif(col,0) – col이 0이면 null 반환, 아니면 col 반환

Coalesce(col1, col2, col3..) – null 아닌 첫번째 값 반환

10. 정렬★★

- 느려질 수 있다.
- 가장 마지막에 실행
- null이 어디에 오는지..

컬럼명으로 정렬, 앞의 기준이 같을 때 그 다음 컬럼으로 정렬

기본값은 asc(오름차순), desc는 내림차순

Order by col1, col2 desc

출력순서(번호)로 정렬, select 절의 출력 순서로 정렬 순서를 지정
Order by 2, 1 desc

11. 숫자함수

Round(222.45, 1) 소수점 둘째자리에서 반올림하여 첫째자리까지 출력
Round(225.67, 0) 소수점 첫째자리에서 반올림하여 정수만 출력
-1 파라미터는 1의 자리에서 반올림하여 정수를 출력

Ceil(oracle) / ceiling(SQL Server) 올림함수, 파라미터 사용법은 round와 같음
Floor 버림 함수, 파라미터 사용법은 round와 같음

12. 문자함수★

Lower, upper – 소문자로, 대문자로
Trim, ltrim, rtrim – 양쪽공백제거, 왼쪽, 오른쪽 공백제거
Lpad, rpad – 특정 자리를 정하고, 왼쪽 / 오른쪽 의 공백을 채워주는 함수

- Select lpad('A', 5, '*') from dual;
- ****A, rpad면 A****

Substr – SELECT SUBSTR('korea', 2, 2) FROM DUAL; or 이 출력
Instr - SELECT INSTR('CORPORATE FLOOR','PO') AS idx FROM DUAL; 4 가 출력

13. 날짜함수★

To_char – 날짜형 데이터를 문자로 출력

- Select to_char(sysdate, 'YYYY-MM-DD') from dual;

To_date – 문자형 데이터를 날짜형으로 출력

- select to_date('2022-09-22') from dual;

sysdate (oracle), getdate() (SQL Server)

13. 조건문★

Decode

- select decode(col1,'A',1,'B',2,3) from dual;
- col이 A면 1, B면 2, 아니면 3

case

```
case when col = 'A' then 1
     when col = 'B' then 2
     else 3 end;                                    서로 같다
case col when 'A' then 1
     when 'B' then 2
     else 3 end;
```

14. 집계함수★★

Count, min, sum, max 등

- null은 포함되지 않는다
- (1, null, 2, 3, null) 의 데이터를 기준으로 결과는 다음과 같다.
  - Count() – 3
  - Sum() – 6
  - Avg() – 2
  - Min() – 1
  - Max() – 3

Col1	Col2	Col3
null	null	1
2	3	2
1	null	null

Select sum(col1 + col2 + col3) from dual;

여기에서 먼저 sum을 생각하지말고 col1 + col2 + col3 을 먼저 생각해보면 첫번째 행은 null + null + 1이기에 null이 반환되고, 마지막 세번째 행도 마찬가지다.

그러므로, 두번째 행의 2 + 3 + 2의 값인 7이 결과가 된다.

반대로, sum(col1) + sum(col2) + sum(col3)의 값은 3 + 3 + 3 이므로 9가 출력이 된다.

이 차이를 알아야 한다.

15. 그룹바이 group by

집약기능을 가지고 있음 (다수의 행을 하나로 합침)

Group by 절에 온 컬럼만 select 절에 올 수 있음

16. join★★

Natural join

- 반드시 두 테이블 간의 동일한 이름, 타입을 가진 컬럼이 필요하다.
- 조인에 이용되는 컬럼은 명시하지 않아도 자동으로 조인에 사용된다.
- 동일한 이름을 갖는 컬럼이 있지만 데이터 타입이 다르면 에러가 발생한다.
- 조인하는 테이블 간의 동일 컬럼이 SELECT 절에 기술 되도 테이블 이름을 생략해야 한다.
- select department_id 부서, department_name 부서이름, location_id 지역번호, city 도시
  from departments
  natural join locations
  where city = 'Seattle';

Using

- USING 절은 조인에 사용될 컬럼을 지정한다.
- NATURAL 절과 USING 절은 함께 사용할 수 없다.
- 조인에 이용되지 않은 동일 이름을 가진 컬럼은 컬럼명 앞에 테이블명을 기술한다.

- 조인 컬럼은 괄호로 묶어서 기술해야 한다.
- select department_id 부서번호, department_name 부서, location_id 지역번호, city 도시
  from departments
  join locations using (location_id);

left outer join

- from table a left outer join table b
  on a.col = b.col 이것과 같은 오라클 sql 문법은
- from table a, table b
  where a.col = b.col(+)

join 순서

- from a,b,c
  a와 b가 join 되고, 그리고 c와 join 된다.

17. 서브쿼리★★★

Select – 스칼라 서브쿼리
From – 인라인뷰 (메인 쿼리의 컬럼 사용 가능)
Where – 중첩 서브쿼리
Group by – 사용 불가
Having – 중첩 서브쿼리
Order by – 스칼라 서브쿼리

In – 서브쿼리 출력값들 or 조건
Any/some – 서브쿼리 출력값들중 가장 작거나 큰 값과 비교
All – any/some과 반대 개념
Exists – 서브쿼리내 select 절엔 뭐가 와도 상관 없다. Row가 있으면 true, 없으면 false

18. 집합연산자★★

Union 정렬O 중복제거O 느리다
Intersect 정렬O 교집합 느리다
Minus (except) 정렬O 차집합 느리다
Union all 정렬X 중복제거X 빠르다

19. DDL★★

Truncate – drop & create, 테이블 내부 구조는 남아 있으나 데이터가 모두 삭제 됨
Drop – 테이블 자체가 없어짐 (당연 데이터도 없음)
Delete – 데이터만 삭제
Rollback, commit 이랑 항상 같이 나옴

20. DML★

Insert – 데이터 넣는 명령, insert into 테이블 (col1, col2, col3..) values ('11', '22', '33'..);

- values를 기준으로 좌우의 괄호속 개수가 맞는지

update – 데이터의 특정 행의 값을 변경 (delete & insert)

- update 테이블 set col = '값' where col1 = '조건';

delete – 데이터의 특정 행을 삭제

- delete from 테이블 where col = '조건';

merge – 특정 데이터를 넣을 때 해당 테이블 키값을 기준으로 있으면 update, 없으면 insert를 한다. (최근 기출)

위 문제 모두 commit, rollback, savepoint 와 주로 함께 출제 된다.

21. 제약조건★★★

PK – not null + unique

- 테이블당 하나의 PK를 가질 수 있음 (하나라는게 컬럼이 아님, 복합키 가능)

Not null – 해당 컬럼에 null이 올 수 없음

Unique – 해당 컬럼에 중복값이 올 수 없음

22. DCL

Grant, revoke 문법

- GRANT 시스템 권한명 [, 시스템 권한명 ... | 롤명] TO 유저명 [, 유저명... | 롤명 ... |PUBLIC | [WITH ADMIN OPTION];
- REVOKE { 권한명 [, 권한명...] ALL} ON 객체명 FROM {유저명 [, 유저명...] | 롤명(ROLE) | PUBLIC} [CASCADE CONSTRAINTS];
- Role은 객체

23. VIEW

- 독립성, 편의성, 보안성
- SQL을 저장하는 개념

24. 그룹함수

- roll up
- cube
- groupingsets
- grouping

어떤 결과가 나오고 어떤 함수를 사용 했는지에 대한 문제 기출

원본테이블

분류	내용	개수	금액
학용품	연필	1	400
학용품	지우개	3	1200
학용품	샤프	2	800
음식	김밥	1	2000
음식	제육덮밥	1	4800
음식	제육덮밥	2	9600
기타	게임	1	100

ROLLUP

분류	내용	개수	금액
기타	게임	1	100
기타	NULL	1	100
음식	김밥	1	2000
음식	제육덮밥	3	14400
음식	NULL	4	16400
학용품	샤프	2	800
학용품	연필	1	400
학용품	지우개	3	1200
학용품	NULL	6	2400
NULL	NULL	11	18900

CUBE

분류	내용	개수	금액
기타	게임	1	100
NULL	게임	1	100
음식	김밥	1	2000
NULL	김밥	1	2000
학용품	샤프	2	800
NULL	샤프	2	800
학용품	연필	1	400
NULL	연필	1	400
음식	제육덮밥	3	14400
NULL	제육덮밥	3	14400
학용품	지우개	3	1200
NULL	지우개	3	1200
NULL	NULL	11	18900
기타	NULL	1	100
음식	NULL	4	16400
학용품	NULL	6	2400

ROLLUP

(GROUP BY에 있는 컬럼들을 오른쪽에서 왼쪽순으로 그룹 생성)

a, b 로 묶이는 그룹의 값

a 로 묶이는 그룹의 소계

전체합계

CUBE

(나올 수 있는 모든 경우의 수로 그룹 생성)

a, b 로 묶이는 그룹의 값

a 로 묶이는 그룹의 소계

b 로 묶이는 그룹의 소계

전체합계

*rollup(A,B) != rollup(B,A), cube(A,B) = cube(B,A)

25. TCL

Commit, rollback

- Auto commit, begin transaction (commit 기능 잠시 끄기) end

26. 윈도우 함수

- rows between and 값이 증가한다.

rows between UNBOUNDED PRECEDING and CURRENT ROW) as "직업별 합계"

rows between 1 PRECEDING and 1 FOLLOWING) as "위아래합계"

- range between and 값이 동일하다

1. UNBOUNDED PRECEDING: 최종 출력될 값의 맨 처음 row의 값(Partition by 고려)
2. CURRENT ROW: 현재 row의 값
3. UNBOUNDED FOLLOWING: 최종 출력될 값의 맨 마지막 row의 값(Partition by 고려)

- Rank 1,1,3,4….
- Dense_rank 1,1,2,3…

- Partition by, order by

  Row_number() over (partition by col1 order by col2)…

27. 계층형 함수★

prior 자식데이터 = 부모데이터

부모데이터에서 자식데이터로 가면 순방향

```
SELECT LEVEL,                    *순방향
          LPAD(' ', 4 * (LEVEL-1)) || 사원 사원,
          관리자,
          CONNECT_BY_ISLEAF ISLEAF
    FROM 사원
   START WITH 관리자 IS NULL
CONNECT BY PRIOR 사원 = 관리자;

SELECT LEVEL,                    *역방향
          LPAD(' ', 4 * (LEVEL-1)) || 사원 사원,
          관리자,
          CONNECT_BY_ISLEAF ISLEAF
    FROM 사원
   START WITH 사원           = 'D'
CONNECT BY PRIOR 관리자 = 사원;
```

28. PL/SQL

- exception (생략가능)
- procedule 반드시 값이 안나옴
- trigger 커밋, 롤백 안됨
  Before, after별로 Insert, update, delete 가 있음
- function 반드시 반환값이 있음

29. 엔터티★★

관리해야 할 대상이 엔터티가 될 수 있다.
인스턴스 2개이상
업무에서 사용해야 함 (프로세스)
관계를 하나이상 가져야 한다

유형 엔터티
개념 엔터티
사건 엔터티

기본 엔터티
중심 엔터티
행위 엔터티

30. 속성★★

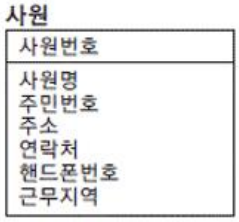
기본 속성
설계 속성
파생 속성

31. 도메인
데이터유형
크기
제약조건

- Check, primary key, Foreign key, not null, unique... 개념 정리

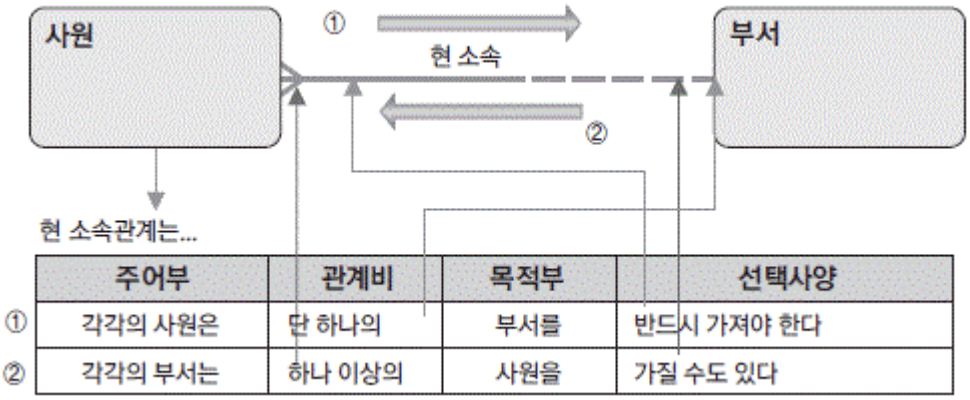
32. 관계

IE

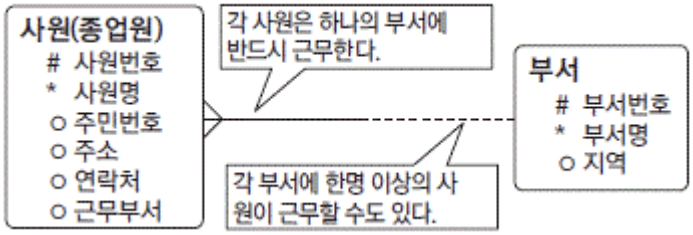


ER win의 부호 사용	Identifying	Non-identifying	의미
타원, 해쉬 마크 및 까마귀 발			0,1, 또는 그 이상의 개체 허용
까마귀 발이 있는 해쉬 마크			1 또는 그 이상의 개체 허용
해쉬 마크가 있는 타원			0 또는 1 개체 허용
해쉬 마크만 있음			정확히 1 개체 허용

Barker



	주어부	관계비	목적부	선택사양
①	각각의 사원은	단 하나의	부서를	반드시 가져야 한다
②	각각의 부서는	하나 이상의	사원을	가질 수도 있다



33. 식별자★★

유일성 – 유일하게 인스턴스를 구분

최소성 – 최소 컬럼으로

불변성 – 값이 바뀌지 않아야 함

존재성 – not null

- 위 4개를 만족하면 후보키가 될 수 있으며, 그중 하나, 대표하는 것이 기본키 이다.

34. 식별자 & 비식별자★

식별자

- 강한관계,
- PK가 많아진다 (조인시)
- SQL이 복잡해짐

비식별자

- 약한관계
- SQL이 느려짐

35. ERD

그리는방법

- 좌상에서 우하로
- 관계명 반드시 표기 하지 않아도 됨
- UML은 객체지향에서만 쓰인다

36. 성능 데이터 모델링★

아키텍쳐 모델링 **(먼저)**

- 테이블, 파티션, 컬럼등의 정규화 및 반정규화

SQL 튜닝 **(그 다음)**

- Join 수행 원리

Hash join

- 등가 join만 사용 함
- Hash 함수를 사용하여 select, join 컬럼 저장 (선행 테이블)
- 선행 테이블이 작다
- Hash 처리를 위한 별도 공간 필요

NL join

- 랜덤 엑세스
- 대용량 sort 작업
- 선행 테이블이 작을수록 유리

Sort Merge

- Join 키를 기준으로 정렬
- 등가/비등가 join 가능

Optimizer

- CBO 제일 경제적인걸 정함
- RBO 규칙에 의해서 정함

- 실행계획 읽는 방법

Id	Operation	Name	Rows	Bytes	Cost(%CPU)	Time
0	select cstmer_no		14	1638	17(0)	00:00:01
1	nested loops		1			
2	nested loops		14	1638	17(0)	00:00:01
3	table access full	tbwcstmr	14	1218	3(0)	00:00:01
* 4	indes unique scan	pk_cstmr	1		0(0)	00:00:01
5	table accescc by index rowid	tbwcard	1	30	1(0)	00:00:01

실행계획을 읽는 순서는 다음과 같다.
ID 3 → 4 → 2 → 5 → 1 → 0

37. 정규화★★★

1차. 원자성

2차. 부분함수종속성 제거

3차. 이행함수종속성 제거

BCNF. 정의

정규화 절차	설명
제 1정규화	- 속성의 원자성을 확보 - **기본키를 설정**
제 2정규화	- 기본키가 2개 이상의 속성으로 이루어진 경우, **부분 함수 종속성**을 제거(분해)
제 3정규화	- 기본키를 제외한 칼럼 간에 종속성 제거 - **이행 함수 종속성**을 제거
BCNF	- 기본키를 제외하고 후보키가 있는 경우, 후보키가 기본키를 종속시키면 분해
제 4정규화	- 여러 칼럼들이 하나의 칼럼을 종속시키는 경우 분해하여 다중 값 종속성을 제거
제 5정규화	- 조인에 위햐서 종속성이 발생되는 경우 분해

Select시 join 때문에 느려질 수 있다. (테이블이 늘어나서)
Insert, update는 빨라질 수 있다. (테이블 사이즈가 작아져서)

38. 이상현상

1. 삽입 이상 (insertion anomaly) : 새 데이터를 삽입하기 위해서 불필요한 데이터도 함께 삽입해야 하는 이상 문제.
2. 갱신 이상(update anomaly) : 중복 튜플 중 일부만 변경하여 데이터가 불일치하게 되는 이상 문제.
3. 삭제 이상(delete anomaly) : 튜플을 삭제하면 필요한 데이터까지 함께 삭제되는 이상 문제.

39. 반정규화★★

데이터의 무결성을 해칠 수 있음

절차

- 대량범위처리 빈도수 조사
- 범위처리 빈도수
- 통계처리 여부

종류

- 테이블 병합 1:1/1:M
- 슈퍼/서브 타입 병합
- 부분테이블 분할
- 통계테이블 분할
- 중복테이블 분할
- 부분테이블 분할
- 이력 컬럼 추가
- 중복 컬럼 추가
- PK를 일반 컬럼으로 병합

- 파생 컬럼 추가
- 응용 시스템 오작동을 피하기 위한 임시값 컬럼 추가
- 중복 관계 추가

40. 데이터에 따른 성능

Row migration

[행 이전]

1. Update로 인해 행 길이가 증가했을 때, 저장 공간이 부족한 경우 발생
2. 원래 정보를 기존 블록에 남겨두고 실제 데이터는 다른 블록에 저장
   → 검색 시, 원래 블록에서 주소를 먼저 읽고 다른 블록을 찾아야 하므로 성능 감소
3. 해결책 : PCTFREE 영역을 충분히 할당한다.
   → PCTFREE가 너무 큰 경우 데이터 저장 공간 부족으로 공간 효율성 감소

Chaing

[행 연결]

1. 데이터가 커서 여러 블록에 나누어 저장하는 현상
   → 2개 이상의 데이터 블록을 검색해야 하므로 성능 감소
2. Initial Row Piece(행 조각)와 Row Pointer로 블록 내에 저장
3. 해결책 : DB_BLOCK_SIZE를 크게 하여 최소화 가능
   → 사이즈 변경이 어렵고, 무조건 크게 할 수 없음

List partition

- 특정 값을 기준으로
- 관리 쉬움
- 데이터가 치우칠 수 있음

Range partition

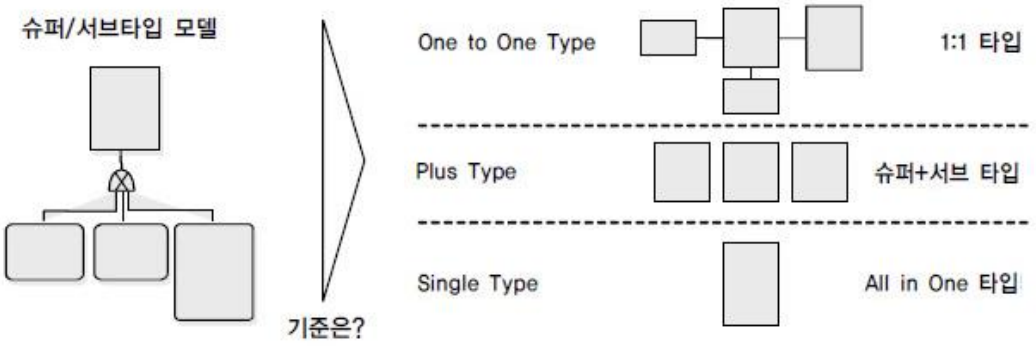
- 특정 값의 범위
- 관리 쉬움
- 가장 많이 씀

Hash partition

- 관리 어려움

41. 슈퍼/서브 타입

1. 1:1 타입(One to One Type)
2. 슈퍼 + 서브타입(Plus Type)
3. All in One타입(Single Type)



1) 트랜잭션은 항상 일괄로 처리하는데 테이블은 개별로 유지되어 Union 연산에 의해 성능이 저하될 수 있다.
2) 트랜잭션은 항상 서브타입 개별로 처리하는데 테이블은 하나로 통합되어 있어 불필요하게 많은 양의 데이터 때문에 성능이 저하된다.
3) 트랜잭션은 항상 슈퍼+서브타입을 공통으로 처리하는데 개별로 유지되어 있거나 하나의 테이블로 집약되어있어 성능이 저하된다.

42. 분산 데이터베이스

분할 투명성	사용자가 입력한 전역 질의를 여러 개의 단편 질의로 변환해 주기 때문에 사용자는 전역 스키마가 어떻 게 분할되어 있는지 알 필요가 없음
위치 투명성	어떤 작업을 수행하기 위해 분산 데이터베이스상에 존재하는 어떠한 데이터의 물리적인 위치도 알 필요가 없음
지역사상 투명성	지역DBMS와 물리적 DB사이의 Mapping 보장. 각 지역시스템 이름과 무관한 이름 사용 가능
중복 투명성	어떤 데이터가 중복되었는지, 또는 어디에 중복 데이터를 보관하고 있는지 사용자가 알 필요가 없음
장애 투명성	분산되어 있는 각 컴퓨터 시스템이나 네트워크에 장애가 발생하더라도 데이터의 무결성이 보장됨
병행 투명성	다수 Transaction 동시 수행시 결과의 일관성 유지, 잠금(Locking)과 타임스탬프(Timestamp)의 두 가지 방법을 주로 사용

단점 : 데이터 무결성을 해칠 수 있다.

43. 인덱스★

사용 못하는 경우

- 부정형
- Like
- 형변환(묵시적)

악영향

- DML 사용시 성능이 저하됨