계층형 쿼리(Hierarchical Query)는 오라클에서만 지원하고 있는 아주 막강한 기능 중의 하나다.

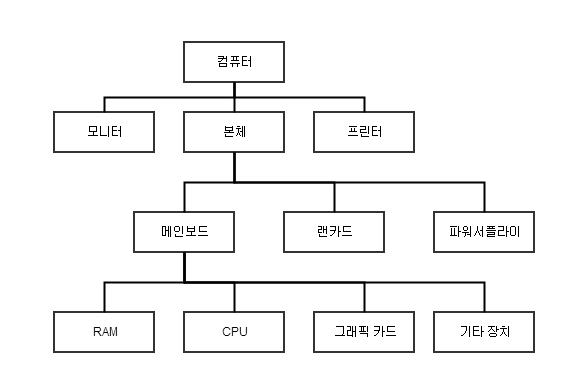
관계형(relational)이라는 의미가 서로 평등하고 수평적인 관계를 의미하는 반면 계층형(hirearchical) 구조는 이와는 다르게 평등한 관계가 아닌 계급적이고 수직적인 관계를 가진다. 한마디로 말해서 관게형이 평면적이라 한다면 계층형은 수직적 구조라 할수있다.

계층형 정보에는 사회적인 의미의 계급구조, 등이 있고

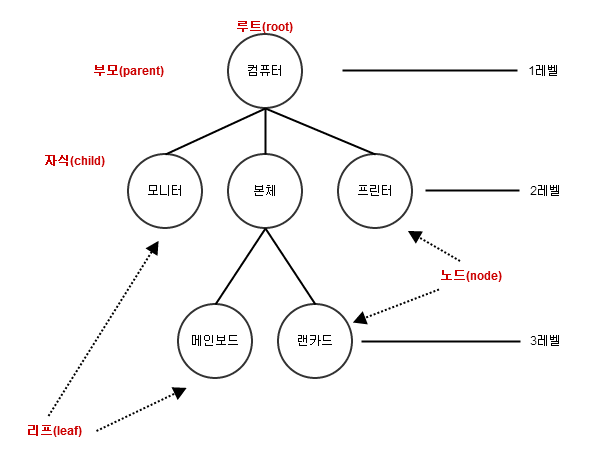
소프트웨어 관점에서 살펴보면, 웹사이트에서 사용하는 답변형 게시판이나 BOM(Bill Of Material)등이 계층형 정보의 전형적인 예이다.

\* BOM(Bill Of Material) 이란? 하나의 완제품이 만들어지는데 필요한 구성품목들의 내역 혹은 조합도를 말함.

  컴퓨터 구성도(BOM)



계층적 구조에서 사용되는 용어 정리



노드(node) : 위 그림에서 원모양으로 표시된 항목을 노드라고 한다. 각각의 품목이 하나의 노드가 되며 실제 테이블에서는 하나의 로우에 대응

부모(parent) : 부모노드라고도 한다. 즉 트리구조에서 상위에 있는 노드를 말한다.

자식(child) : 자식노드라고도 한다. 위 그림에서 '모니터'는 컴퓨터의 자식노드이며, 컴퓨터는 모니터와 본체의 부모노드가 된다.

리프(leaf) : 리프노드라고도 한다. 더이상 하위에 연결된 노드가 없는 항목을 말한다. 즉 자식노드가 없는 노드라고 할수있으며 모니터, 프린터, 랜카드가 이에 해당된다.

루트(root) : 계층형, 트리구조에서 최상위에 있는 노드를 말한다. 컴퓨터가 이에 해당

레벨(level) : 트리구조에서의 각각의 계층을 말한다. 루트에 해당되는 '컴퓨터'가 1레벨이 되며, 순차적으로 그 하위에 있는 '모니터', '본체' 등이 2레벨이 된다.

**계층형 쿼리의 작성**

[**DDL**]

CREATE TABLE BOM (

     ITEM\_ID INTEGER NOT NULL, -- 품목식별자

     PARENT\_ID INTEGER, -- 상위품목 식별자

     ITEM\_NAME VARCHAR2(20) NOT NULL, -- 품목이름

     ITEM\_QTY INTEGER, -- 품목 개수

     PRIMARY KEY (ITEM\_ID)

);

INSERT INTO BOM VALUES ( 1001, NULL, '컴퓨터', 1);

INSERT INTO BOM VALUES ( 1002, 1001, '본체', 1);

INSERT INTO BOM VALUES ( 1003, 1001, '모니터', 1);

INSERT INTO BOM VALUES ( 1004, 1001, '프린터', 1);

INSERT INTO BOM VALUES ( 1005, 1002, '메인보드', 1);

INSERT INTO BOM VALUES ( 1006, 1002, '랜카드', 1);

INSERT INTO BOM VALUES ( 1007, 1002, '파워서플라이', 1);

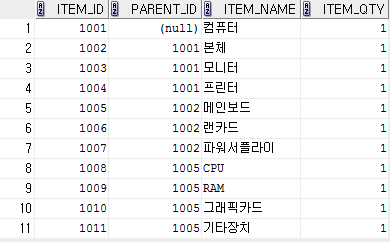
INSERT INTO BOM VALUES ( 1008, 1005, 'CPU', 1);

INSERT INTO BOM VALUES ( 1009, 1005, 'RAM', 1);

INSERT INTO BOM VALUES ( 1010, 1005, '그래픽카드', 1);

INSERT INTO BOM VALUES ( 1011, 1005, '기타장치', 1);

          select \* from bom 질의어 실행결과 :



**1) 일반 조인을 사용한 계층형 쿼리**

테이블 개수가 하나이며 각각의 품목 정보들은 서로 부모와 자식 관계로 연결되어 있으므로 조인을 사용해야하는데, 자기 참조 성격이 있으므로 셀프조인을 사용.

루트노드에 해당하는 컴퓨터의 경우 가장 상위의 1레벨에 속하기 때문에 PARENT\_ID컬럼값이 NULL이다. 그래서 최종 제품인 컴퓨터를 위해 외부 조인도 사용

SELECT bom1.item\_name,

     bom1.item\_id,

     bom2.item\_name parent\_item

**FROM bom bom1, bom bom2**

**WHERE bom1.parent\_id = bom2.item\_id(+)**

ORDER BY bom1.item\_id;



=> 동일한 테이블인 BOM을 각각 bom1, bom2라는 별칭(alias)을 주어 셀프조인을 사용하였고, WHERE절을 보면 알수있듯이 bom1이 하위 품목을 , bom2가 상위품목을 나타냄을 알수있다. 가장 상위 품목인 컴퓨터의 경우, 자신이 최상위 계층이고 PARENT\_ID컬럼에 NULL이 들어가므로 외부조인을 사용하였다.

=> 계층적인 순서로 결과가 나오긴했지만 계층형구조(트리구조)와는 조금 다른 결과이다.

     예를 들어 본체의 하위 품목인 메인보드, 랜카드, 파워서플라이 품목이 프린터 바로 밑에 조회되어 마치 프린터의 하위품목인것처럼 보여진다.

=> 이문제를 해결하기위해 오라클에서 지원하는 계층형 쿼리를 살펴보자.

**2) START WITH ... CONNECT BY 절을 사용한 계층형 쿼리**

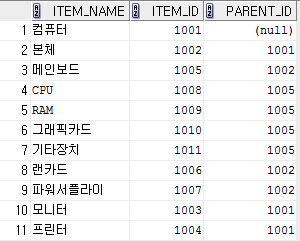
START WITH ... CONNECT BY 절은 계층형 정보를 표현하기 위한 목적으로 오라클 8i부터 지원되기 시작.

SELECT item\_name, item\_id, parent\_id

FROM bom

**START WITH** parent\_id IS NULL --루트노드를 지정,

**CONNECT BY PRIOR** item\_id = parent\_id;--부모와 자식노드들간의 관계를 연결



좀더 보기좋게 레벨별로 들여쓰기를 한후 결과를 다시 확인해보자.

SELECT LPAD(' ', 2\*(**LEVEL**-1)) || item\_name item\_names,

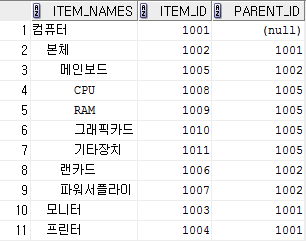
        item\_id,

        parent\_id

FROM bom

**START WITH** parent\_id IS NULL

**CONNECT BY** **PRIOR** item\_id = parent\_id;



\* PRIOR 키워드...

  본체의 PARENT\_ID는 부모노드인 컴퓨터의 ITEM\_ID와 연결되므로 PRIOR키워드는 PARENT가 아닌 ITEM\_ID앞에 붙어야한다.

**3) START WITH *조건1* ... CONNECT BY *조건2***

|  |
| --- |
| [[**START WITH** *조건1*] [**CONNECT BY** *조건2*] |

\* START WITH 조건1 : 루트노드를 식별한다. 조건1을 만족하는 모든 ROW 들은 루트노드가 된다.

  START WITH절을 생략할수도 있는데 이러한 경우 모든 ROW들을 루트노드로 간주한다. 조건1에는 서브쿼리도 올수있다.

\* CONNECT BY 조건2 : 부모와 자식노드들 간의 관계를 명시하는 부분이다.

                                  조건2에는 반드시 PRIOR연산자를 포함시켜야하며, 이는 부모노드의 컬럼을 식별하는데 사용된다.

                                  START WITH과는 달리 조건2에서는 서브쿼리가 올수 없다.

\* PRIOR연산자

   PRIOR키워드는 워직 **계층형 쿼리에서만 사용하는 오라클 SQL연산자**이다. 키워드라고 표현했지만 실제로는 CONNECT BY절에서 등호(=)와 동등한 레벨로 사용되는 연산자이며 CONNCET BY절에서 해당 컬럼의 부모로우를 식별하는데 사용된다.

앞의 예에서 본체의 PARENT\_ID 컬럼에는 컴퓨터의 ITEM\_ID값을 가지고 있으므로 PRIOR연산자는 ITEM\_ID앞에 붙게 된다.

\* 레벨 의사컬럼(LEVEL Pseudocolumn)

  계층형 정보를 표현할 때 레벨을 나타낸다.

  LEVEL도 일반적인 컬럼 처럼 SELECT, WHERE, ORDER BY 절에서 사용할수있다.



**[문제]**정리 퀴즈

샘플 테이블 (BOM)에서  본체를 포함해서 본체의 하위 품목들만 계층구조로 조회하시오

SELECT LEVEL,

           LPAD(' ', 2\*(LEVEL-1)) || item\_name item\_names,

       item\_id, parent\_id

FROM bom

START WITH **ITEM\_ID = '1002'**

CONNECT BY PRIOR item\_id = parent\_id;



**계층형 쿼리의 확장**

=> 오라클에서는 계층형 쿼리에서 추가적인 정보를 제공하기 위해 몇가지 키워드를 더 제공하고 있다.

CONNECT\_BY\_ROOT (루트노드 찾기)

**CONNECT\_BY\_ROOT***컬럼*

*\**​단독으로 사용되지 못하고 일반컬럼과 같이 사용해야함.

CONNECT\_BY\_ISCYCLE (중복 참조값 찾기)

CONNECT\_BY\_ISCYCLE은 반드시 CONNECT BY절에 NOCYCLE이 명시되어 있어야 사용이 가능하다.

CONNECT\_BY\_ISLEAF (리프노드 찾기)

계층형 쿼리에서 해당 로우가 리프노드인지(지삭노드가 없는 노드인지) 여부를 체크하여, 리프노드에 해당할경우 1을 그렇지 않을 경우 0을 반환

SYS\_CONNECT\_BY\_PATH (루트 찾아가기)

SYS\_CONNECT\_BY\_PATH ( *column*, *char* )