
Data Modeling and Relational Database Design(한글판)

학생용 – 볼륨 1

20000KR13
Edition 1.3
2002년 8월
D38039

ORACLE®

만든 이

Patrice Daux
Jeff Gallus
Jan Speelpenning

기술 제공자 및 검토자

Kate Heap
Simmie Kastner
Joni Lounsberry
Satyajit Ranganathan
Sunshine Salmon
Stijn Vanbrabant
Gabriella Varga

발행인

Christine Markusic

Copyright © Oracle Corporation, 2002. All rights reserved.

이 문서는 Oracle International Corp.의 독점적 정보를 포함하고 있습니다. 이 정보는 사용 제한 및 기밀 유지 규정을 포함하는 사용권 계약에 따라 제공되며 저작권법에 의해 보호됩니다. 이 소프트웨어를 리버스 엔지니어링하는 것은 금지되어 있습니다. 이 문서를 미국 국방성 내의 정부 기관에 제공할 때는 제한된 권리 규정이 적용되며 다음 범례를 적용할 수 있습니다.

Restricted Rights Legend

Use, duplication or disclosure by the Government is subject to restrictions for commercial computer software and shall be deemed to be Restricted Rights software under Federal law, as set forth in subparagraph (c) (1) (ii) of DFARS 252.227-7013, Rights in Technical Data and Computer Software (October 1988).

This material or any portion of it may not be copied in any form or by any means without the express prior written permission of Oracle International Corporation. Any other copying is a violation of copyright law and may result in civil and/or criminal penalties.

If this documentation is delivered to a U.S. Government Agency not within the Department of Defense, then it is delivered with 'Restricted Rights,' as defined in FAR 52.227-14, Rights in Data-General, including Alternate III (June 1987).

The information in this document is subject to change without notice. If you find any problems in the documentation, please report them in writing to Education Products, Oracle Corporation, 500 Oracle Parkway, Box, Redwood Shores, CA 94065. Oracle International Corporation does not warrant that this document is error-free.

Oracle is a registered trademark and Oracle, SQL*Plus, SQL*Net, Oracle Developer, Oracle8i, Oracle9i, Oracle9i Designer and PL/SQL are trademarks or registered trademarks of Oracle Corporation.

인용된 모든 다른 회사명 또는 제품명은 명시된 목적으로만 사용되었으며 각 소유 회사들의 상표일 수 있습니다.

Oracle Internal & OAI Use Only

목차

1 엔티티, 속성 및 관계 소개

- 개요 1-2
- 왜 개념적 모델을 생성합니까? 1-3
- 이상과 실제... 1-5
- 엔티티 관계 모델링 1-7
- 엔티티 관계 모델링의 목표 1-8
- 클러스터 유형 1-9
- 엔티티 1-10
- 엔티티 및 인스턴스 1-11
- 엔티티 및 세트 1-12
- 속성 1-13
- 속성 예제 1-14
- 관계 1-15
- 관계 예제 1-16
- Employee는 Job을 가짐 1-17
- 도표 상의 엔티티 표시 1-18
- 도표의 속성 1-20
- 도표 상의 관계 1-21
- 도표를 통한 용이한 의사 전달 1-22
- 관계 라인의 특성 1-23
- 두 가지 관점 1-24
- 한쪽 방향 1-25
- 다른 방향 1-26
- 관계 끝 읽기 1-27
- 기능이 데이터를 구동함 1-34
- 일기 예보 1-35
- 일기 예보, 솔루션 1-38
- ER 도표의 그래픽 요소 1-39
- 요약 1-40
- 연습 1-41
- 연습: 인스턴스인가 아니면 엔티티인가? 1-42
- 연습: 투숙객 1-43
- 연습: 읽기 1-44
- 연습: 읽기 및 주석 달기 1-45
- 연습: 호텔 1-46

2 엔티티 및 속성 세부 사항

개요 2-2

정보에 데이터 비교 2-3

데이터 2-4

엔티티 2-6

1: 필수 m 3-9

일부 배경 정보 2-9

일부 요구되는 기능 2-13

엔티티 정의의 변천 과정 2-15

업무 요구 사항 2-17

속성... 2-19

명사, 엔티티, 속성 2-20

EM 엔티티 및 속성 2-22

속성 및 엔티티 2-23

중복성 2-24

하위 유형... 2-25

하위 유형 예제 2-26

하위 유형: 규칙 2-27

하위 유형: 세 레벨 2-28

하위 유형에 대한 자세한 내용 2-29

요약 2-30

연습 2-31

요약 2-34

판매점 목록 2-36

하위 유형 2-38

연습: 주소(1/2) 2-40

연습: 주소(2/2) 2-41

3 관계 세부 설명

개요 3-2

관계 설정 3-3

관계 이름 3-5

관계 명명 3-6

선택 가능성 3-7

선택 가능성 3-8

필수

정도 3-10
비전이성 3-12
관계 유형 1:m 3-13
관계 유형 m:1 3-15
관계 유형 m:m 3-16
관계 유형 1:1 3-18
1:1 관계 룰 3-19
1:1 관계 프로세스 3-20
중복 관계 3-21
관계 및 속성 3-22
관계에 속성 비교 3-23
속성 또는 엔티티 3-24
관계에 속성 비교 3-25
관계에 속성 비교 3-26
m:m 관계로 특정 사항을 숨길 수 있음 3-27
Quantity는 ...의 속성 3-28
관계의 속성 3-29
새로운 ORDER 엔티티 3-30
한 주문에 대한 다수의 제품 3-31
또 다른 새 엔티티: ORDER ITEM 3-32
테이블 3-33
m:m 관계 분석 3-34
m:1 관계 분석 3-37
정규화 규칙 3-39
데이터 모델링의 첫번째 정규 폼 3-40
데이터 모델링의 두번째 정규 폼 3-41
데이터 모델링의 세번째 정규 폼 3-42
요약 3-43
연습 3-44
연습: 관계 읽기 3-45
컨텍스트 찾기(1) 3-46
컨텍스트 찾기(2) 3-47
컨텍스트 찾기(3) 3-48
컨텍스트 찾기(4) 3-49
연습: 교차 엔티티 명명 3-50
연습: 영수증 3-51
연습: Moonlight P&O 3-52
연습: 가격 목록 3-54
연습: 전자 메일 3-55
연습: 휴일 3-56
연습: ER 모델 정규화 3-57

4 제약 조건

개요 4-2

렘브란트 4-3

신원 증명과 표시 4-5

고유 식별자 예제 4-6

고유 식별자 4-7

고유 식별자 4-8

복수 관계 UID 4-10

올바르게 정의된 고유 식별자 4-12

잘못된 고유 식별자 4-13

정보를 포함하는 코드 4-14

호 4-15

배타적 호 4-16

가능한 호 구성 4-17

몇 가지 잘못된 호 구성 4-18

호 또는 하위 유형 4-20

호 및 하위 유형 4-21

하위 유형은 호 내의 관계를 숨김 4-22

값 집합 4-23

기타 제약 조건: 범위 검사 4-24

기타 제약 조건: 상태 값 변환 4-25

조건부 관계 4-26

경계 4-28

요약 4-29

연습 4-30

연습: 식별 방법 4-31

연습: 신원 증명 1 4-32

연습: 신원 증명 2 4-33

연습: 신원 증명 3 4-34

연습: 신원 증명 4 4-35

연습: 신원 증명 5 4-36

연습: 신원 증명 6 4-37

연습: Moonlight UID 4-38

연습: 테이블 14-39

연습: 테이블 2 4-41

연습: 제약 조건 1 4-42

연습: 제약 조건 2 4-43

연습: 제약 조건 3 4-44

5 모델링 변화

- 개요 5-2
- 변경 및 시간 5-3
- DAY 엔티티 5-4
- 모델링 변경 5-6
- 국가에도 수명 주기가 있습니다. 5-8
- 제품 및 가격 5-10
- 지불할 가격은 얼마입니까? 5-11
- 가격 목록 검색 5-12
- Priced Product에 대한 주문 5-13
- 협상 가격 5-14
- 현재 가격 5-16
- 저널링 5-17
- 요약 5-19
- 연습 5-20
- 연습: 근무 교대 5-21
- 연습: 딸기 와퍼 5-22
- 연습: 번들(1) 5-24
- 연습: 번들(2) 5-25
- 연습: 제품 구조 5-26

6 고급 모델링 항목

- 개요 6-2
- 패턴 6-3
- 패턴: 마스터-디테일 6-5
- 패턴: 바구니 6-6
- 패턴: 분류 6-7
- 패턴: 계층 구조 6-8
- 패턴: 체인 6-10
- 패턴: 네트워크 6-11
- BOM 6-13
- BOM-예제 6-14
- 대칭 관계 6-15
- 패턴: 롤 6-16
- 롤 6-17
- Fan Trap 6-18
- 분석된 Fan Trap 6-19
- 패턴: 데이터 웨어하우스 6-20
- 그리기 규칙 6-21
- 적절한 규칙 사용 6-22
- 모델 가독성 6-23
- 일반 모델링 6-24
- 일반적인 모델 6-26

일반적인 모델 6-27
보다 일반적인 모델 6-28
보다 일반적인 것 이상의 모델 6-29
가장 일반적인 모델 6-30
두 분야의 최고 6-31
요약 6-32
연습 6-33
연습 6-34
연습: 데이터 웨어하우스 6-35
연습: Argo 및 Erat 6-38
연습: 동의어 6-39

7 엔티티 모델 매핑

개요 7-2
왜 데이터 디자인 모델을 생성합니까? 7-3
테이블 표시 7-4
변환 프로세스 7-6
용어 매핑 7-7
일반 이름 지정 항목 7-8
Oracle의 이름 지정 제한 7-11
엔티티에 대한 기본 매핑 7-12
속성에 대한 기본 매핑 7-13
기본 매핑 7-14
관계에 대한 규칙 7-16
1:m 매핑 관계 7-17
세로줄 표시가 있는 관계와 비전이적 관계의 매핑 7-19
연쇄적인 세로줄 표시가 있는 관계 매핑 7-20
m:m 관계 매핑 7-21
1:1 관계 매핑 7-22
호 매핑 7-23
하위 유형 매핑 7-25
상위 유형 구현 7-27
하위 유형 구현 7-29
상위 유형 및 하위 유형(호) 구현 7-31
저장 영역의 의미 7-33
저장 영역의 의미 상위 유형 구현 7-34
저장 영역의 의미 하위 유형 구현 7-35
저장 영역의 의미 상위 유형과 하위 유형(호) 구현 7-36

요약 7-37
연습 7-38
연습: 기본 엔티티, 속성 및 관계 매핑 7-39
연습: 상위 유형 매핑 7-40
Moonlight ER 모델 일부 7-41
연습: 품질 검사 하위 유형 구현 7-42
연습: 품질 검사 호 구현 7-43
연습: 기본 키와 열 매핑 7-44

8 비정규화된 데이터

개요 8-2
비정규화 개요 8-3
비정규화 기법 8-4
파생 가능 값 저장 8-5
파생 가능 값을 저장하는 전자 메일 예제 8-6
테이블 사전 조인 8-7
테이블 사전 조인의 전자 메일 예제 8-8
하드 코딩된 값 8-9
하드 코딩된 값의 전자 메일 예제 8-10
마스터로 디테일 유지 8-11
마스터로 디테일을 보관하는 전자 메일 예제 8-12
마스터에 최근 디테일 반복 8-13
마스터에 단일 디테일을 반복하는 전자 메일 예제 8-14
단락 키 8-15
단락 키의 전자 메일 예제 8-16
종료일 열 8-17
종료 날짜 열 예제 8-18
현재 지시자 열 8-19
현재 지시자 열 예제 8-20
계층 레벨 지시자 예제 8-21
계층 레벨 지시자 예제 8-22
비정규화 요약 8-23
연습 8-24
연습: 비정규화 명명(1/3) 8-25
연습: 비정규화 명명(2/3) 8-26
연습: 비정규화 명명(3/3) 8-27
연습: 트리거(1/6) 8-28
연습: 트리거(2/6) 8-29
연습: 트리거(3/6) 8-30
연습: 트리거(4/6) 8-31
연습: 트리거(5/6) 8-32
연습: 트리거(6/6) 8-33
연습: 가격 목록 비정규화 8-34
연습: 전역 이름 지정 8-35

9 데이터베이스 설계 고려 사항

개요 9-2

왜 데이터 설계를 수정해야 합니까? 9-3

Oracle 데이터 유형 9-4

제안된 열 순서 9-6

기본 키 9-7

인공 키 9-11

시퀀스 9-13

외래 키 기능 9-14

인덱스 9-16

인덱스 선택 9-17

어떤 열을 인덱스화해야 합니까? 9-19

인덱스는 언제 사용할 수 있습니까? 9-21

테이블과 인덱스 분할 9-22

뷰 9-23

뷰를 사용하는 이유 9-24

기존 방식의 설계 9-25

일반 호 구현 9-26

분산 데이터베이스 9-27

분산 데이터베이스의 이점 9-28

데이터베이스 구조 9-29

요약 9-31

연습 9-32

데이터 유형(1) 9-34

데이터 유형(2) 9-35

부록 A

부록 B

Oracle Internal & OAI Use Only

1

엔티티, 속성 및 관계 소개

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

Oracle Internal & OAI Use Only

개요

왜 개념적 모델링인가?

주요 롤 항목 소개:

- 엔티티
- 속성
- 관계

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

소개

이 단원은 개념적 모델링이 필요한 이유를 설명하고 주요 롤 항목인 엔티티, 속성 및 관계를 소개합니다.

목표

이 단원을 마치면 다음을 수행할 수 있습니다.

- 개념적 모델링이 중요한 이유 설명
- 엔티티의 개념 설명 및 예시
- 속성의 개념 설명 및 예시
- 관계의 개념 설명 및 예시
- 간단한 도표 그리기
- 간단한 도표 읽기

왜 개념적 모델을 생성합니까?

- 업무의 정보 요구를 정확히 설명합니다.
- 논의를 원활하게 합니다.
- 실수를 방지하고 이해를 도와줍니다.
- 중요한 "이상적 시스템"을 문서화할 수 있게 합니다.
- 물리적 데이터베이스 설계에 적합한 기반을 형성합니다.
- 많은 관련 작업자에게 매우 유용한 작업 방법입니다.

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

왜 개념적 모델링인가?

이 과정은 개념적 데이터 모델링 및 물리적 데이터 모델링에 대해 다룹니다. 이 과정을 학습해야 하는 이유는 무엇일까요? 테이블이 필요한 경우 엔티티 모델을 생성하는 데 시간을 투자해야 하는 이유는 무엇일까요? 프로그램이 필요한 경우 업무 기능과 인터뷰 및 피드백 세션을 살펴 보아야 하는 이유는 무엇일까요? 이 과정을 통해 이러한 이유를 이해하게 될 것입니다. 왜 모델링에 시간을 할애하는 것이 현명한 결정이며 적절한 투자인지 알게 됩니다. 모델에서 테이블 및 키 정의를 파생시키는 방법뿐 아니라 모델을 생성하고 읽고 이해하는 방법, 모델을 검사하는 방법 등 여러 정보를 얻을 수 있습니다.

이 목록은 개념적 모델을 생성하는 이유를 보여줍니다. 가장 중요한 이유는 개념적 모델을 사용하면 향후 시스템의 형태에 대한 논의를 원활하게 진행할 수 있다는 것입니다. 사용자와 동료 간 뿐만 아니라 사용자와 후원자 간의 의사 전달에도 도움이 됩니다. 또한 모델은 물리적 데이터베이스의 기본 설계를 위한 기반을 형성합니다. 마지막으로 중요한 사항은 모델의 생성 및 변경 비용이 매우 저렴하다는 것입니다.

학습 내용

이 과정에서는 업무 요구 사항을 분석하는 방법, 찾은 내용을 엔티티 관계 도표로 나타내는 방법, 해당 모델에서 테이블 및 다양한 기타 데이터베이스 객체를 정의하고 세분화하는 방법을 배웁니다.

요약하자면 이 과정을 통해 다음과 같은 정보를 얻게 됩니다.

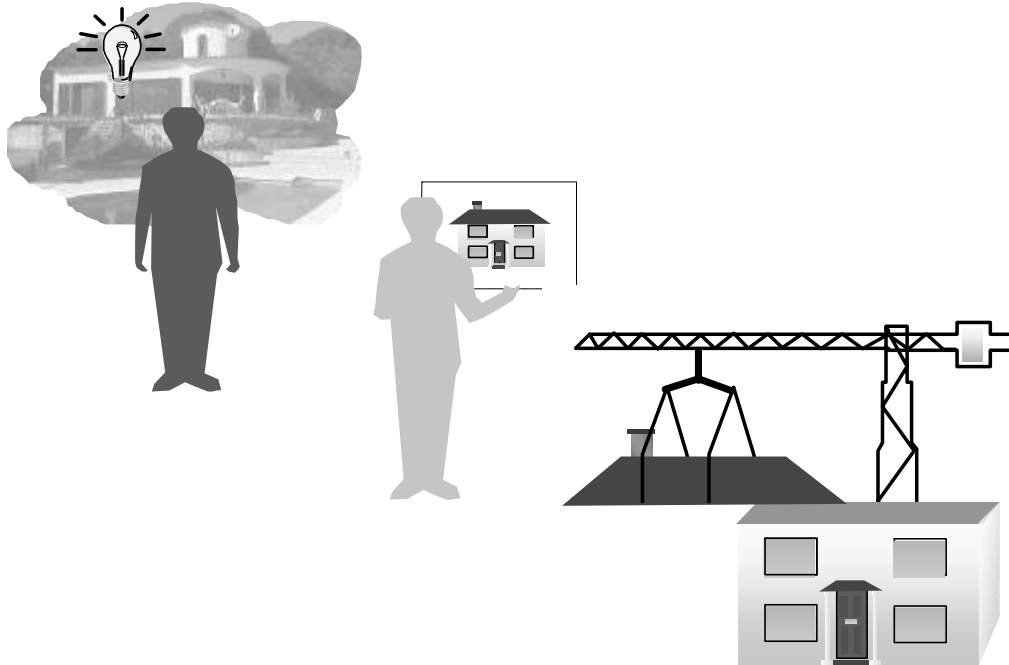
- 업무에 요구되는 정보 및 적용되는 규칙을 모델링하는 방법
- 데이터베이스에 필요한 테이블과 그 이유
- 테이블에 필요한 열과 그 이유
- 필요한 제약 조건 및 기타 데이터베이스 객체

또한 다음 대상자에게 이러한 사항을 설명하는 방법도 알게 됩니다.

- 후원자
- 개발자
- 동료 설계자

Oracle Internal & OAI Use Only

이상과 실제...



ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

집 짓기에 비유

누군가 집을 짓고자 하는 경우를 상상해 보십시오. 처음에 이 집은 미래 집 주인의 마음에 아이디어나 다양한 꿈의 조각들로만 존재합니다. 경우에 따라 미래의 주거자가 자신이 어떤 집을 원하는지 모르거나 자신이 원하는 집이 실현 가능한 것인지 모를 수도 있습니다. 꿈은 내부적인 모순과 불가능성으로 짝 차 있을 수 있습니다. 이러한 사실이 꿈의 세계에서는 문제가 되지 않지만, 실제로 집을 지으려면 먼저 이러한 불일치와 장애물을 극복해야 합니다.

건축업자는 사용할 자재, 지붕 들보의 크기, 배관의 용량 및 기타 여러 사항에 대한 설명을 포함하는 구체적인 집의 청사진을 필요로 합니다. 건축업자는 계획에 따라 작업하며 청사진에 그려진 대로 건축할 수 있는 지식을 보유하고 있습니다. 그러나 집 주인의 아이디어가 어떻게 건축업자를 위한 청사진이 될 수 있을까요? 바로 이러한 작업에 설계사가 관여하는 것입니다.

설계사

설계사는 집주인과 건축업자 사이에서 중개인 역할을 합니다. 설계사는 아이디어를 모델로 변환하는 기술을 훈련 받습니다. 설계사는 아이디어에 대한 설명을 듣고 갖가지 질문에 대답합니다. 아이디어를 추출하여 논의 및 분석할 수 있는 형식으로 만들고, 조언을 주고, 어울리는 옵션에 대해 설명하고, 문서화하고, 집 주인의 승인을 얻어 내는 설계사의 기술은 미래의 집 주인에게 원하는 집을 계획할 수 있도록 해주는 토대가 됩니다.

집 짓기에 비유

스케치

설계자가 이해한 꿈은 새로운 주택의 스케치로 변형됩니다. 오직 스케치로 말입니다. 이것은 평면도와 몇몇 설계자의 느낌으로 구성되며 건축의 세부 사항이 아니라 주택의 기능적 요구 사항을 보여줍니다. 이것이 개념적 모델의 첫번째 버전입니다.

쉬운 변경

모델의 부분이 만족스럽지 않거나 잘못 이해된 경우 모델을 쉽게 변경할 수 있습니다. 이러한 변경 작업에는 약간의 시간과 지우개 또는 새 용지만 있으면 됩니다. 이것은 모델만 변경하는 것입니다. 이 단계에서 소요되는 변경 비용은 아주 작습니다. 건설이 시작된 후에 평면도나 지붕 치수를 변경하는 것보다 훨씬 더 저렴할 것입니다. 그런 후 주택 모델을 다시 검토하고 추가 변경을 수행합니다. 설계자는 논쟁의 소지가 있는 모든 문제가 해결되고 모델이 안정화되고 집주인의 최종 허가를 받을 상태가 될 때까지 계속해서 꿈을 탐색하고 명확하게 하며 대안을 제시합니다.

기술적 설계도

그런 다음 설계자는 건축업자가 집을 건설하는 데 사용할 수 있는 계획안인 기술적 설계도로 모델을 변환합니다. 예를 들어, 문의 수, 벽 및 대들보의 두께, 배관 치수 및 지붕의 정확한 구성을 결정하기 위한 계산이 수행됩니다. 이러한 사항은 고객이 개입될 필요가 없는 기술적 문제입니다.

"어떻게"가 아닌 "무엇을" 고려

개념적 모델은 프로세스에서 "무엇을" 단계를 해결해주지만 설계는 "어떻게" 건설해야 하는가라는 문제를 해결합니다.

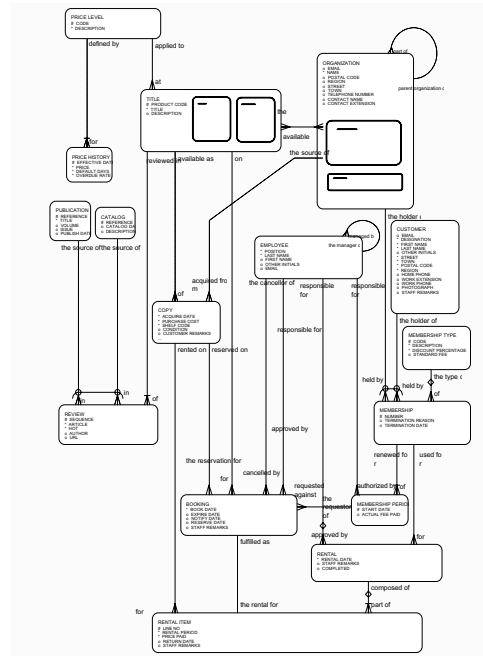
개념적 모델링은 설계자의 작업과 유사합니다. 즉, 사람들의 마음 속에만 있는 것을 실제로 충분히 생성할 수 있는 설계로 변환하는 일입니다.

Oracle Internal & OAI Use Only

엔티티 관계 모델링

- 업무 구현이 아닌
업무 모델링
- 잘 설정된
기법
- 강력한 구문 보유
- 읽기 쉬운 도표
형성...

...처음에는
다소 복잡하게
보일 수 있음



ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

모델링에 관련된 것은 무엇입니까?

엔티티 관계 모델링은 업무의 모델링에 대한 것입니다. 좀더 정확히 말하면 현재 시스템의 기능 또는 향후 제공될 시스템에 요구되는 기능을 기준으로 업무에 대한 데이터 요구사항을 모델링하는 것입니다.

업무를 모델링하려면 업무의 대상이 무엇인지를 정확히 이해해야 합니다.

엔티티 관계 모델링은 업무의 정보 요구에 대한 상호 이해를 기술하는 데 사용되는 기법입니다. 이 모델링은 읽기 쉬우며 따라서 확인하기도 쉬운 도표를 생성하는 안정된 기법입니다.

엔티티 관계 모델링의 목표

- **모든 필수 정보 포착**
- **정보는 오직 한 번 나타남**
- **이미 모델링된 다른 정보에서 파생 가능한 정보는 모델링하지 *않음***
- **정보는 예측 가능하며 논리적인 장소에 보관함**

ORACLE

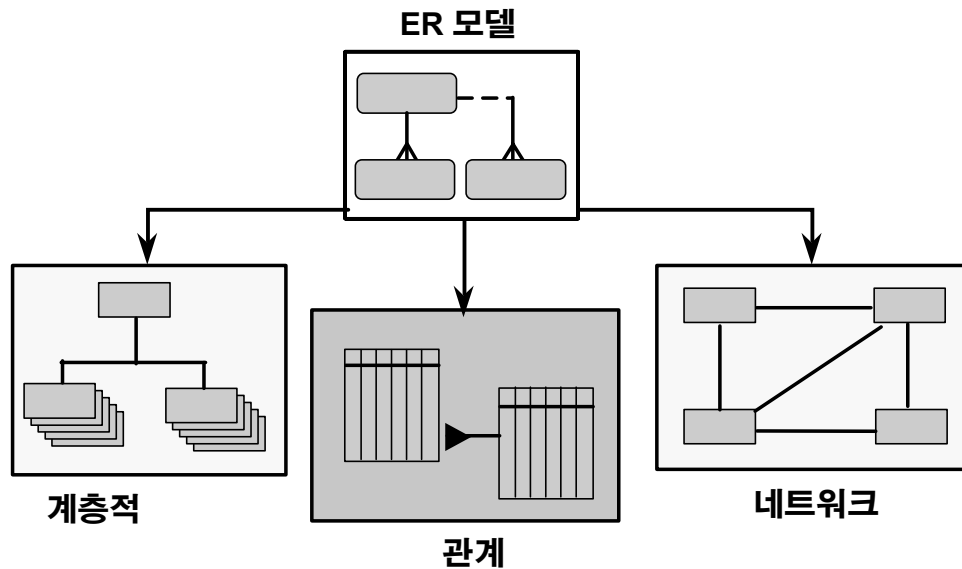
Copyright © Oracle Corporation, 2002. All rights reserved.

엔티티 관계 모델링의 목표

개념적 데이터 모델링의 목표는 다음을 보장하는 것입니다.

- 업무를 제대로 실행하는 데 필요한 모든 정보 부분이 인식됩니다.
- 모델은 완벽해야 합니다. 구현에 앞서 요구 사항을 파악해야 합니다. 종속 관계는 명확해야 합니다.
- 필수 정보의 각 부분은 모델에서 한 번만 나타나야 합니다.
- 이것은 중요한 목표입니다. 시스템이 특정 정보를 두 번 저장하면 바로 이 정보가 두 위치에서 동일하지 않을 가능성이 발생하는 것입니다. 정보 시스템을 사용하고 있으며 데이터가 일관되지 못하다는 사실을 발견한다면 어느 정보를 신뢰할 것입니까?
- 이 목표는 이상적인 시스템에는 파생 가능한 정보가 없다는 것을 의미합니다.
- 향후 제공될 시스템에서는 정보를 예측 가능한 논리적 장소에서 찾을 수 있으며 관련 정보는 함께 보관될 것입니다.
- 적절한 엔티티 관계 모델은 논리적으로 일관된 테이블 세트가 됩니다.

데이터베이스 유형



ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

데이터베이스 유형

엔티티 관계 모델링은 구현에 사용되는 하드웨어 또는 소프트웨어에 독립적입니다. 계층적 데이터베이스, 네트워크 데이터베이스 및 관계형 데이터베이스에 대한 기반으로 엔티티 관계 모델을 사용할 수 있지만 관계형 데이터베이스에 밀접한 관계가 있습니다.

엔티티

- **엔티티:**
 - 관련 데이터를 필요로 하는 업무에 중요한 "어떤 것"
 - 나열할 수 있는 것에 대한 이름
 - 일반적으로 명사임
- **예제: 객체, 이벤트**
- **엔티티는 인스턴스를 가짐**

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

엔티티의 정의

엔티티에 대한 많은 정의와 설명이 있습니다. 여기서는 일부만 소개하며 이 중에는 유용한 정보도 있고 매우 정확한 것도 있을 것입니다.

- 엔티티는 관련성이 있는 어떤 것입니다.
- 엔티티는 관련 정보를 유지해야 하는 어떤 업무에 중요한 사항들의 한 범주입니다.
- 엔티티는 목록을 만들 수 있으며 업무에 중요한 것입니다.
- 엔티티는 어떤 것들의 클래스 또는 유형입니다.
- 엔티티는 명명된 대상으로 대개 명사입니다.

엔티티의 두 가지 중요한 측면은 인스턴스를 가진다는 것과 엔티티의 인스턴스가 어떤 식으로든 업무에 관계가 있다는 것입니다.

엔티티와 엔티티 인스턴스 간의 차이점을 알아 두십시오.

엔티티 및 인스턴스

PERSON	Mahatma Gandhi
PRODUCT	2.5 x 35 mm copper nail
PRODUCT TYPE	nail
EMPLOYMENT CONTRACT	my previous contract
JOB	violinist
SKILL LEVEL	fluent
TICKET RESERVATION	tonight: Hamlet in the Royal
PURCHASE	the CD I bought yesterday
ELECTION	for parliament next fall
PRINTER PREFERENCE	...
DOCUMENT VERSION	...

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

엔티티에 대한 추가 정보

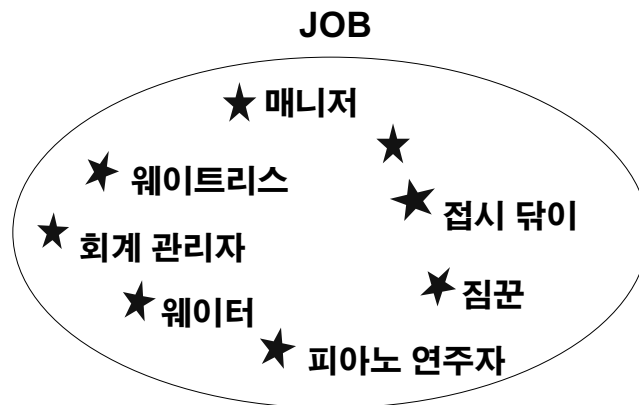
이 그림은 엔티티의 예제 및 해당 엔티티의 인스턴스 예제를 보여줍니다.

주:

- 많은 엔티티가 있습니다.
- 일부 엔티티는 많은 인스턴스를 가지며 일부는 적은 인스턴스만 가집니다.
- 엔티티는 다음과 같을 수 있습니다.
 - PERSON, PRODUCT와 같은 유형의 것
 - REQUIRED SKILL LEVEL과 같은 무형의 것
 - ELECTION과 같은 이벤트
- 한 엔티티의 인스턴스는 그 자체로 엔티티가 될 수 있습니다. JOB 엔티티의 "violinist" 인스턴스는 "David Oistrach", "Kyung-Wha Chung"과 같은 인스턴스를 가지는 다른 엔티티의 이름이 될 수 있습니다.

엔티티 및 세트

엔티티는 특정 업무에 관계가 있는 인스턴스 세트를 나타냅니다.



ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

엔티티 및 세트

엔티티를 세트로 간주할 수 있습니다. 이 그림은 JOB 세트를 보여주며 해당 세트는 해당 인스턴스의 일부를 보여줍니다. 엔티티 모델링 프로세스 맨 마지막에 엔티티는 테이블로 변환됩니다. 해당 테이블의 행은 개별 인스턴스를 나타냅니다.

엔티티 모델링 중에 전체 세트에 적용되는 속성과 규칙을 볼 수 있습니다. 종종 예제 인스턴스를 고려하여 규칙을 결정할 수 있습니다. 다음 단원은 이러한 예를 포함합니다.

세트 이론

엔티티 관계 모델링과 관계형 데이터베이스 이론은 모두 적절한 수학적 이론 즉, 세트 이론을 기반으로 합니다.

속성

- 업무에 중요한 어떤 것을 나타냄
- 엔티티의 **단일 값**으로 된 속성 세부 사항임
- 다음 기능을 수행하는 정보의 특정 부분임
 - 설명
 - 정량 분석
 - 한정
 - 분류
 - 엔티티 지정

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

속성이란?

속성은 특정 방식으로 엔티티를 설명하는 정보의 부분입니다. 속성은 엔티티의 속성 (property)이며 엔티티의 아주 작은 부분입니다.

엔티티는 속성을 가짐

당분간 모든 엔티티가 하나 이상의 속성을 가진다고 가정해 봅시다. 나중에 이 가정에 대한 예외를 살펴 보겠습니다. 속성은 엔티티를 설명, 정량 분석, 한정, 분류 및 지정합니다. 일반적으로 엔티티에 대한 많은 속성이 있지만 여기서는 업무와 관련이 있는 속성을 집중적으로 다루겠습니다.

값 및 데이터 유형

속성은 값을 가집니다. 속성값은 숫자, 문자열, 날짜, 이미지, 소리 등이 될 수 있습니다. 이것을 데이터 유형 또는 형식이라고 합니다. 일반적으로 엔티티의 인스턴스가 갖는 특정 속성값은 모두 동일한 데이터 유형을 갖습니다. 모든 속성은 데이터 유형을 갖습니다.

속성은 단일 값을 가짐

엔티티의 속성은 단일 값을 가져야 합니다. 좀더 정확히 말하자면 엔티티 인스턴스는 특정 시점에 해당 속성에 대해 하나의 값만 가질 수 있습니다. 이것은 속성의 가장 중요한 특징입니다.

그러나 속성값은 시간 경과에 따라 변경될 수 있습니다.

속성 예제

엔티티	속성
EMPLOYEE	Family Name, Age, Shoe Size, Town of Residence, Email, ...
CAR	Model, Weight, Catalog Price, ...
ORDER	Order Date, Ship Date, ...
JOB	Title, Description, ...
TRANSACTION	Amount, Transaction Date, ...
EMPLOYMENT	Start Date, Salary, ...
CONTRACT	

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

속성 예제

주:

- EMPLOYEE에 대한 Town of Residence 속성은 변경하기 쉽지만 특정 시점에 단일 값을 갖는 속성의 예입니다.
- Shoe Size 속성은 중요하지 않은 것처럼 보이지만 업무에 의존합니다. 즉, 업무에 따라 직원에게 업무용 의복이 제공될 경우 이것은 적용될 속성에 매우 중요한 영향을 줄 수 있습니다.
- Family Name 속성은 두 개의 이름을 갖는 사람에 대해서는 단일 값이 아닌 것처럼 보일 수 있습니다. 그러나 이 이중 이름은 하나의 이름을 형성하는 단일 문자열로 간주될 수 있습니다.

휘발성 속성

일부 속성은 휘발성(불안정 상태)입니다. 예로 속성 Age를 들 수 있습니다. 항상 비휘발성인 안정된 속성을 찾아 보십시오. 여러 개 중에 선택해야 하는 경우 비휘발성 속성을 사용하십시오. 예를 들어, Age 대신 Birth Date 속성을 사용하십시오.

관계

- 역시 업무에 중요한 어떤 것을 나타냄
- 엔티티가 상호 *관련되는* 양상을 나타냄
- 항상 두 엔티티(또는 한 엔티티의 경우 첫번째와 *두번째*) 사이에 존재함
- 항상 두 개의 관점을 가짐
- 양쪽에서 명명됨

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

Oracle Internal & OAI Use Only

관계 예제

EMPLOYEES는 JOBS를 가짐
JOBS는 EMPLOYEES에 의해 소유됨

PRODUCTS는 PRODUCT TYPE으로 분류됨
PRODUCT TYPE은 PRODUCT에 대한 분류임

PEOPLE은 TICKET RESERVATION을 만듦
TICKET RESERVATION은 PEOPLE에 의해 만들어짐

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

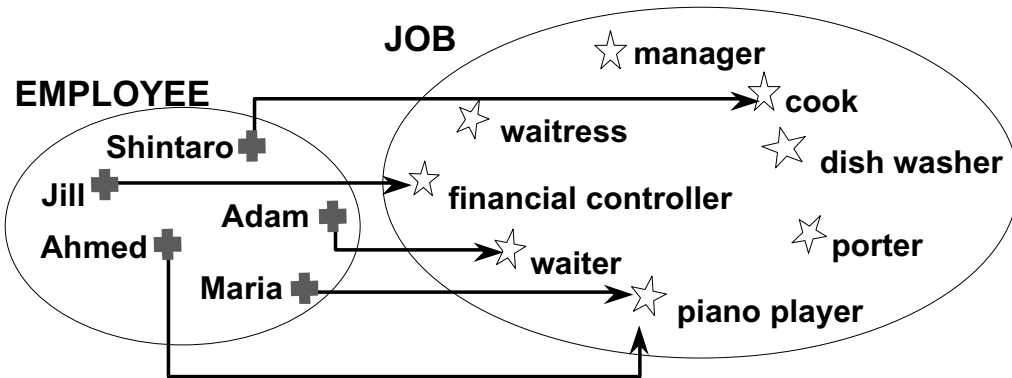
관계

관계는 두 엔티티를 연결합니다. 관계는 항상 두 엔티티의 중요한 종속 관계를 나타냅니다.

특정 관계를 여러 방식으로 나타낼 수 있습니다. 즉, EMPLOYEE는 JOB을 가짐 또는 EMPLOYEE는 JOB을 수행함 또는 EMPLOYEE는 JOB을 소유함과 같이 나타낼 수 있습니다.

JOB에 대해 적용되는 EMPLOYEE는 다른 관계를 나타냅니다. 이 예제는 두 엔티티가 둘 이상의 관계를 가질 수 있음을 보여줍니다.

Employee는 Job을 가짐



수치적 관찰:

- 모든 EMPLOYEE가 하나의 JOB을 가짐
- 어떠한 EMPLOYEE도 둘 이상의 JOB을 갖지 않음
- 모든 JOB이 하나의 EMPLOYEE에 의해 소유되지는 *않음*
- 일부 JOB은 둘 이상의 EMPLOYEE에 의해 소유됨

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

Employee는 Job을 가짐

엔티티 인스턴스에 대한 지식을 기반으로 다음의 네 가지 질문에 대해 결론을 내릴 수 있습니다.

- 모든 직원이 하나의 직무를 가져야 합니까?
- 즉, 이것은 직원에 대한 필수 관계입니까 선택 관계입니까?
- 직원은 둘 이상의 직무를 가질 수 있습니까?

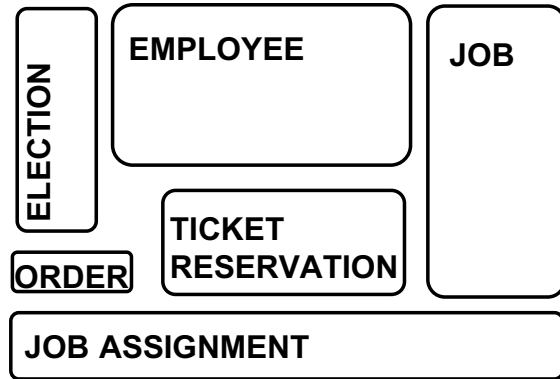
그리고

- 모든 직무가 한 직원에 의해 수행되어야 합니까?
- 즉, 이것은 직무에 대한 필수 관계입니까 선택 관계입니까?
- 둘 이상의 직원이 하나의 직무를 수행할 수 있습니까?

나중에 이러한 질문 사항이 왜 중요한지와 그 대답이 테이블 설계에 어떻게, 어떤 이유로 영향을 주는지 살펴 볼 것입니다.

도표 상의 엔티티 표시

- "모서리가 둥근 상자 (softbox)"로 그려짐
- 이름이 단수형임
- 이름이 안쪽에 표시됨
- 크기나 위치에는 특별한 의미가 없음



설계 중에 엔티티는 보통 테이블이 됩니다.

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

엔티티 관계 모델 및 도표

ER 모델(엔티티 관계 모델)은 중요한 엔티티 간의 모든 관계뿐 아니라 모든 엔티티 및 속성의 목록입니다. 또한 이 모델은 엔티티 설명, 데이터 유형 및 제약 조건과 같은 배경 정보를 제공합니다. 이 모델에 반드시 그림이 포함되지는 않지만 일반적으로 모델의 도표는 매우 유용합니다.

ER 도표(엔티티 관계 도표)는 모델 또는 모델의 일부를 나타내는 그림입니다. 일반적으로 하나의 모델을 여러 도표로 나타내어 여러 다른 업무 관점을 나타냅니다.

그래픽 요소

엔티티 관계 도표에서는 많은 수의 그래픽 요소를 사용합니다. 이 내용은 다음 페이지에서 다룹니다.

불행히도 ER 도표를 나타내는 ISO 표준이 없습니다. Oracle은 자체의 규칙을 보유하고 있습니다. 이 과정에서는 Oracle Designer 도구에 내장된 Oracle 도표 작성 기법을 사용합니다.

ER 도표에서 엔티티는 내부에 엔티티 이름을 포함하는 모서리가 둥근 상자 형태로 그려집니다. 엔티티 상자의 테두리는 서로 겹쳐지지 않습니다. 엔티티 상자는 항상 똑바르게 그려집니다.

이 책 전체에서 엔티티 이름은 대문자로 표시됩니다. 엔티티 이름은 단수형이 선호되며 이러한 방식으로 인해 더욱 쉽게 도표를 읽을 수 있습니다.

상자 크기

엔티티의 크기나 위치는 특별한 의미가 없습니다. 그러나 읽는 사람이 큰 엔티티를 작은 엔티티보다 중요하게 해석할 수도 있습니다.

엔티티로 인해 도출되는 결과

관계형 데이터베이스 설계 중에 엔티티가 보통 테이블을 이룹니다.

Oracle Internal & OAI Use Only

도표의 속성

모든 인스턴스에 알려져 있으며 모든 인스턴스에서 사용할 수 있는 필수 속성입니다.

일부 인스턴스에 알려져 있지 않거나 일부 인스턴스에서 중요하지 않게 인식되는 선택 속성입니다.

EMPLOYEE
* Family Name
* Address
o Birth Date
o Shoe Size
o Email

JOB
* Title
o Description

설계 중에 속성은 열이 됩니다.

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

속성 표시

속성은 엔티티 상자 내에 나열됩니다. 속성 앞에는 * 또는 O가 올 수 있습니다. 이러한 기호는 속성이 각각 필수 또는 선택임을 의미합니다. 이 책 전체에서 속성은 첫 문자가 대문자로 표시됩니다.

필수:

엔티티의 모든 인스턴스에 대해 엔티티 인스턴스가 기록될 때 해당 속성값을 알 수 있고 사용할 수 있으며 업무상 해당 값을 기록할 필요가 있다고 가정할 수 있습니다.

선택:

엔티티의 인스턴스를 기록할 때 해당 속성값을 알 수 없거나 사용 불가능할 수 있으며 또는 값을 알 수 있지만 중요하지 않을 수 있습니다.

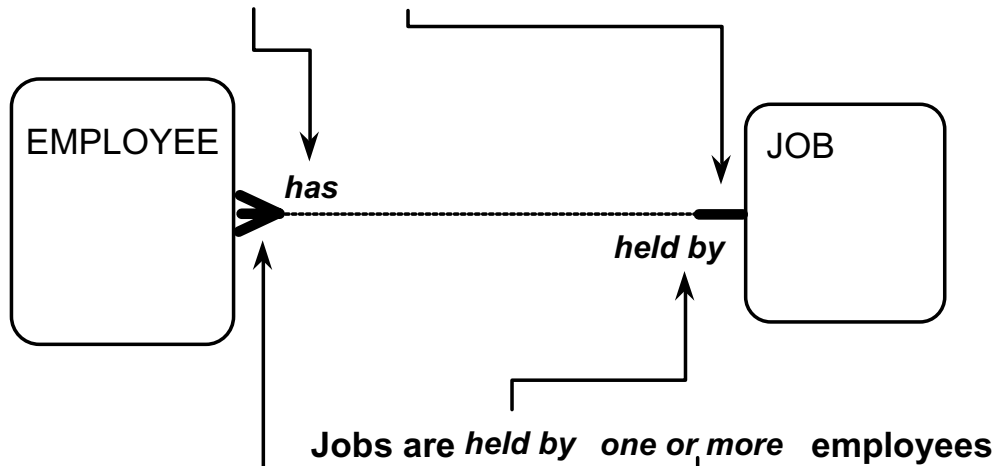
엔티티의 모든 속성을 도표에 표시할 필요는 없지만 테이블을 설계하기 전에 모든 속성을 알아야 합니다. 명확성과 가독성을 위해 도표에는 주로 소수의 속성만 표시됩니다. 일반적으로 엔티티가 무엇에 대한 것이며 어떤 속성이 엔티티를 자세하게 또는 간단하게 "정의"하는지 쉽게 파악할 수 있도록 해주는 속성을 선택하게 됩니다.

속성으로 도출되는 결과

설계 중에 속성을 일반적으로 열이 됩니다. 필수 속성은 null이 아닌 열이 됩니다.

도표 상의 관계

An employee *has* exactly one job.



설계 중에 관계는 외래 키가 됩니다.

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

관계 표시

관계는 엔티티를 연결하는 라인으로 표시됩니다. 한쪽 관점에 본 관계의 이름은 관계 라인의 시작점 부분에 표시됩니다.

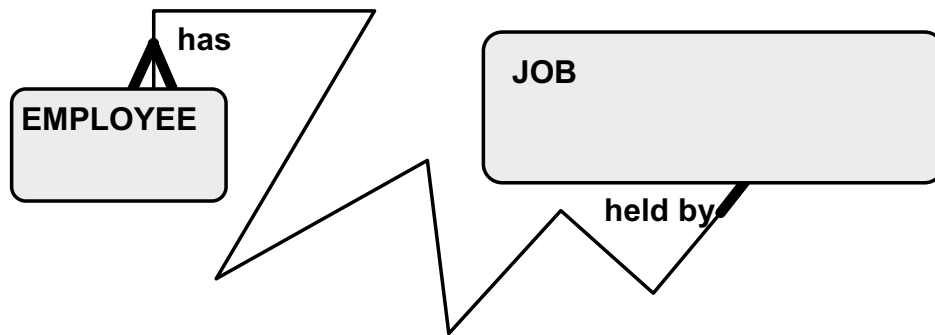
관계 라인 끝의 모양은 관계의 정도를 나타냅니다. 이것은 하나 또는 여럿을 나타냅니다. 하나는 정확히 하나를 의미하고 여럿은 하나 이상의 의미합니다.

위의 예제에서는 JOB이 하나 이상의 EMPLOYEE에 의해 소유된다고 가정합니다. 이것은 EMPLOYEE에서 삼발이 모양(또는 크로우풋)으로 표시됩니다.

한편 EMPLOYEE는 여기에서 정확히 하나의 JOB을 갖는다고 가정됩니다. 이것은 JOB에서 단일 라인으로 표시됩니다.

관계 라인은 직선이지만 곡선이 될 수도 있습니다. 곡선은 특수한 의미를 갖지 않으며 관계 라인의 시작점 위치도 마찬가지입니다. 아래 도표는 정확히 동일하지만 다소 명확성이 떨어지는 모델을 나타냅니다.

도표를 통한 용이한 의사 전달



ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

Oracle Internal & OAI Use Only

관계 라인의 특성

필수:



선택:



ORACLE

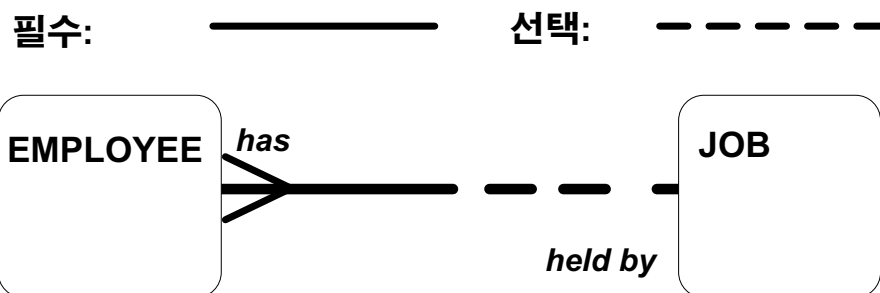
Copyright © Oracle Corporation, 2002. All rights reserved.

필수 또는 선택 관계

관계는 속성과 같은 방식으로 필수이거나 선택일 수 있습니다. 필수 관계는 실선으로 그려지고 선택 관계는 점선으로 그려집니다.

Oracle Internal & OAI Use Only

두 가지 관점



ORACLE

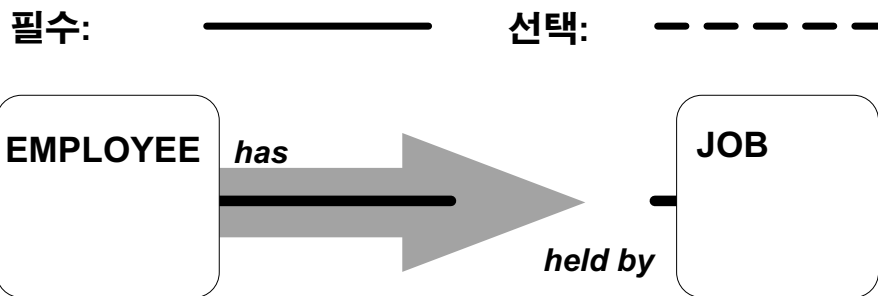
Copyright © Oracle Corporation, 2002. All rights reserved.

관계 및 관계 끝

여기서 EMPLOYEE와 JOB 간의 관계는 선택 관계 끝과 필수 관계 끝을 사용하여 모델링됩니다.

Oracle Internal & OAI Use Only

한쪽 방향



모든 EMPLOYEE는 정확히 하나의 JOB을 가짐

ORACLE

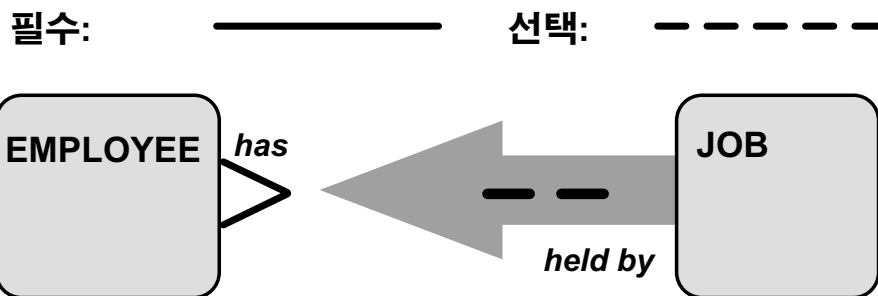
Copyright © Oracle Corporation, 2002. All rights reserved.

관계 및 관계 끝

관계를 읽을 때는 두 개의 관점으로 분리된다고 가정해 보십시오.

Oracle Internal & OAI Use Only

다른 방향



하나의 JOB은 하나 이상의 EMPLOYEE에 의해 소유될 수 있음

ORACLE

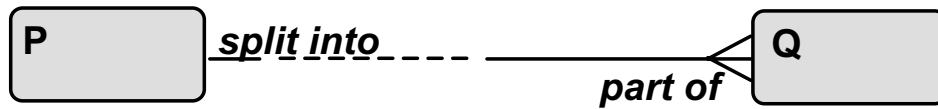
Copyright © Oracle Corporation, 2002. All rights reserved.

관계 및 관계 끝(계속)

- 모든 EMPLOYEE는 정확히 하나의 JOB을 가집니다. 또는
- 한 명의 EMPLOYEE는 정확히 하나의 JOB을 가져야 합니다.

Oracle Internal & OAI Use Only

관계 끝 읽기



ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

관계 끝 읽기

entity1에서 **entity2**로의 관계는 다음과 같이 읽어야 합니다.

Each **entity1** {must be | may be}

relationship_name

{one or more | exactly one} **entity2**

관계로 도출되는 결과

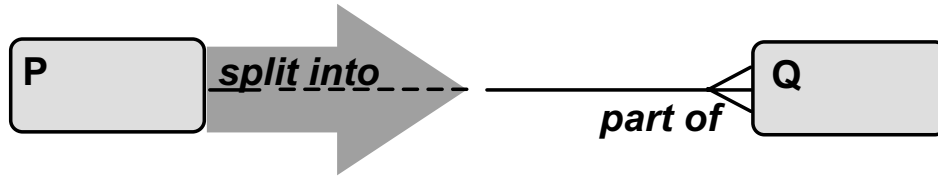
설계 중에 관계는 외래 키 및 외래 키 열이 됩니다. 선택적 관계는 필수가 아닌 외래 키 열을 생성합니다.

도표 상의 관계 이름

이 책 전체에서 도표 상의 관계 이름은 소문자 기울임꼴로 표시됩니다.

이 책에 제공되는 도표의 공간 절약과 가독성을 위해 경우에 따라 관계 이름을 매우 간략하게 표시하거나 전치사만 표시할 수도 있습니다.

관계 끝 읽기

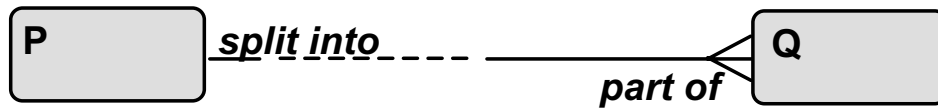


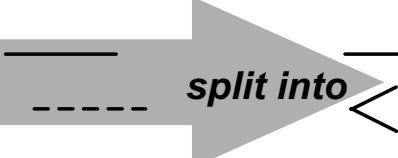

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

Oracle Internal & OAI Use Only

관계 끝 읽기



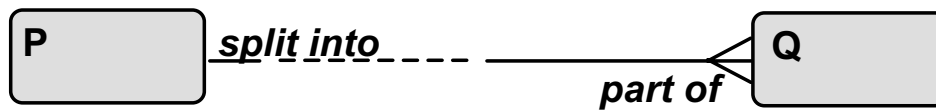
“Each P must be  exactly one Q”
may be  one or more Qs

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

Oracle Internal & OAI Use Only

관계 끝 읽기



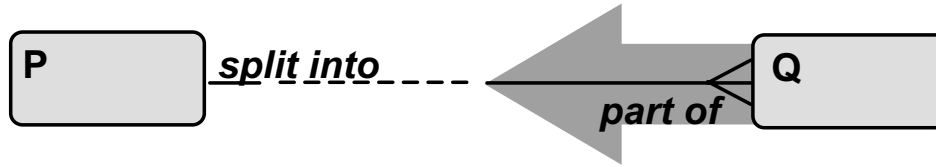
“Each P may be *split into* one or more Qs”

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

Oracle Internal & OAI Use Only

관계 끝 읽기



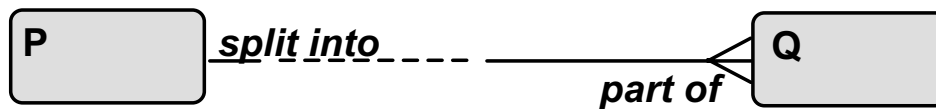
“Each P may be split into one or more Qs”

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

Oracle Internal & OAI Use Only

관계 끝 읽기



“Each P may be split into one or more Qs”

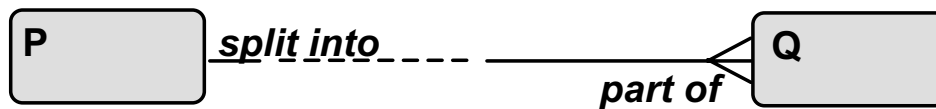
“Each Q must be _____ exactly one P ”
 may be ----- one or more Ps

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

Oracle Internal & OAI Use Only

관계 끝 읽기



“Each P may be split into one or more Qs”

“Each Q must be part of exactly one P”

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

Oracle Internal & OAI Use Only

기능이 데이터를 구동함

- **업무 기능은 항상 제시됩니다.**
 - 명시적
 - 가정됨
- **업무 기능은 데이터를 필요로 합니다.**
- **엔티티, 속성 또는 관계는 다음과 같은 이유로 모델링될 수 있습니다.**
 - 업무 기능에 사용됩니다.
 - 업무 요구는 가까운 미래에 발생할 수 있습니다.

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

기능이 개념적 데이터 모델을 구동함

이 과정에서는 기능 모델링 방법을 다루지 않지만 기능은 개념적 데이터 모델에 대한 임의의 논의에서 언제든지 제시될 수 있습니다. 향후 시스템에 요구되는 기능을 알거나 가정하지 않고는 개념적 데이터 모델에 대해 논의하거나 판단할 수 없습니다.

개념적 데이터 모델에 대한 논의는 주로 데이터 구조에 대한 것처럼 보이지만 실제로는 기능, 즉 일반적으로는 기능의 불분명한 부분 또는 미결정 부분에 대한 것입니다. 사용되는 언어는 개념적 데이터 모델의 언어이며 사용되는 표현은 엔티티 관계 도표의 표현이지만 실제로 논의되는 내용은 기능에 대한 것입니다.








기능은 개념적 데이터 모델을 구동합니다. "직원의 신발 치수를 알 필요가 있습니까?"라는 질문은 "이 정보를 필요로 하는 업무 기능이 있습니까?"라는 질문에 대답해야만 답변을 얻을 수 있습니다.

개념적 데이터 모델을 시스템 기능의 그림자로 간주할 수 있습니다.

이 과정에서 대부분은 기능 설명 페이지 뒤에 나오는 페이지를 읽지 않도록 하기 위해 기능을 간략하게 묘사하거나 가정하기만 합니다.

일기 예보

January 26

København		1/-5	➔ 3
Bremen		0/-3	↙ 4
Berlin		3/-1	↠ 3
München		5/-3	↠ 3
Amsterdam		8/3	↗ 4
Bruxelles		4/0	➔ 2
Paris		4/1	➔ 3
Bordeaux		7/2	↗ 3

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

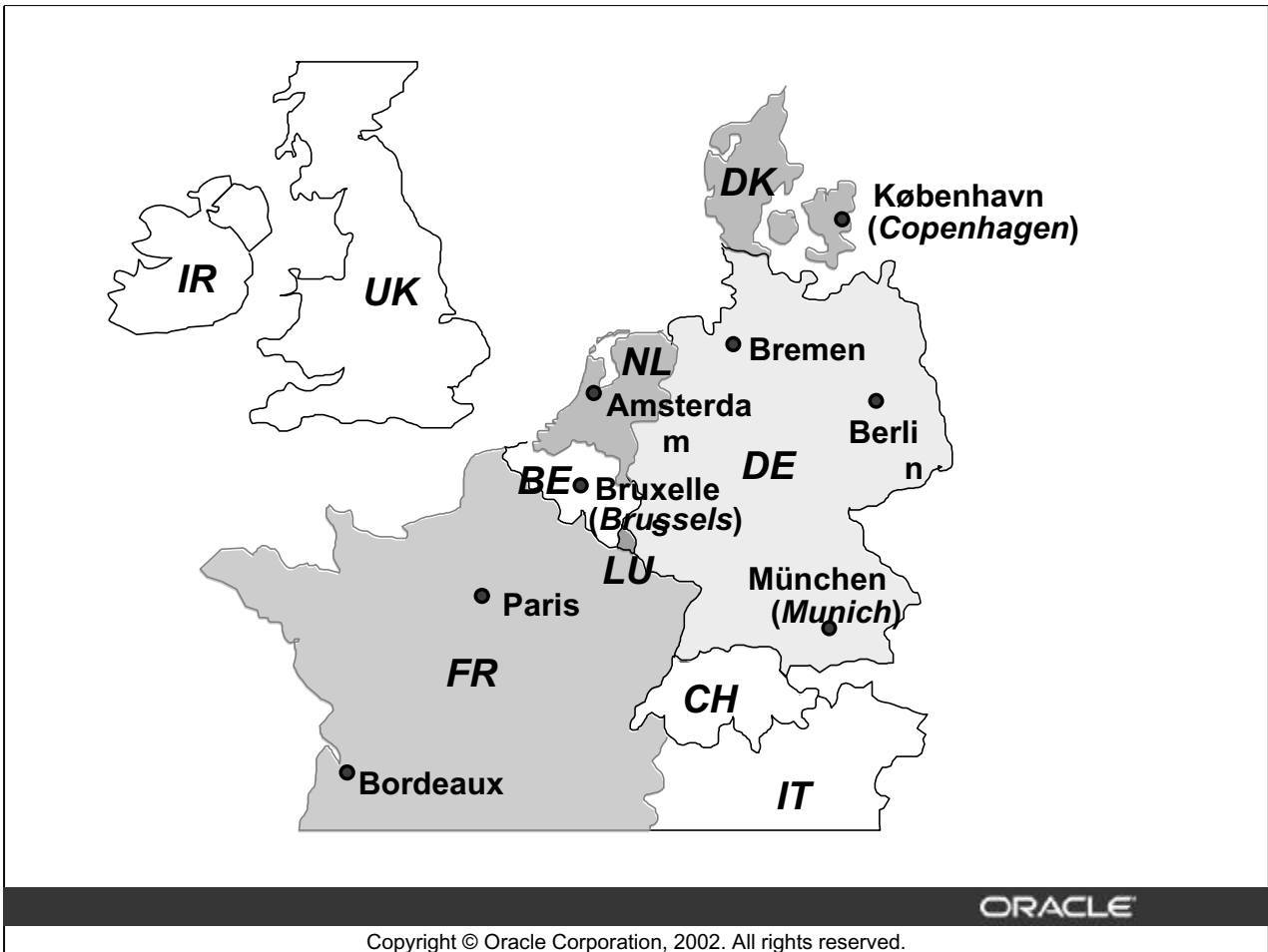
어떤 정보를 사용할 수 있습니까?

이 그림은 유럽 신문에서 발췌한 일기 예보의 일부로서, 다양한 정보 유형을 보여줍니다. 어떤 유형의 정보가 있습니까? 예를 들어, 첫번째로 보게 되는 내용은 "København", "Bremen"입니다. 이것은 도시, 좀더 정확하게 말하면 도시의 이름입니다. 작은 그림은 날씨 유형을 나타냅니다. 이 그림은 아이콘에 해당합니다. 다음 열은 최고 기온과 최저 기온입니다. 화살표는 풍향이며 그 옆의 숫자는 풍속을 나타냅니다. 그런 다음 맨 위에는 예측일이 표시됩니다. 따라서 다음과 같은 데이터를 갖게 됩니다.

- 도시
- 도시의 이름(예: "København")
- 날씨 유형(예: "cloudy with rain")
- 날씨 유형의 아이콘
- 최저 기온
- 최고 기온
- 풍향 화살표
- 풍속
- 예측일

이것이 전부입니까?

아닙니다. 추가 정보를 찾을 수 있습니다. 이를 위해서는 약간의 "업무" 지식이 있어야 합니다. 이 경우 지리적 지식이 필요합니다.



정보 유형

일기 예보 대상 도시가 무작위 순서로 표시되지 않는다는 것을 알 수 있습니다. 프랑스의 도시와 마찬가지로 독일의 도시(Bremen, Berlin 및 München)는 그룹화됩니다. 뿐만 아니라 도시는 이름의 알파벳 순서로 나열되지 않고 북쪽에서 남쪽으로 나열되는 것처럼 보입니다. 틀림없이 이 보고서는 그룹화 및 정렬을 용이하게 하기 위한 어떤 정보를 "알고 있습니다." 이러한 정보에는 다음이 포함될 수 있습니다.

- 도시의 국가
- 도시의 지리적 위치

그리고

- 국가의 지리적 위치

다음 단계

위의 정보 유형 중에서 엔티티, 속성 및 관계가 되는 정보 유형을 식별해 보십시오.

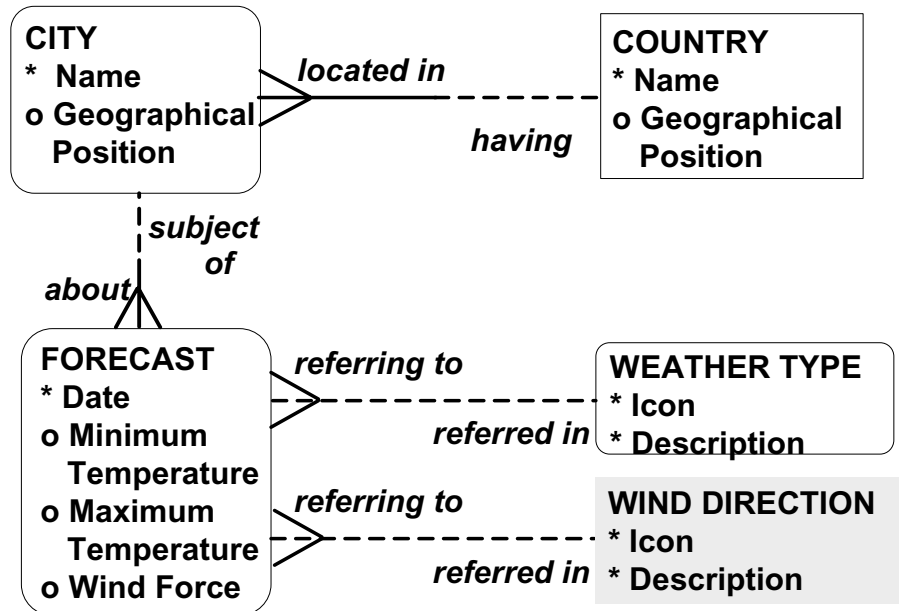
City와 Country는 파악하기 쉽습니다. 이것은 엔티티이며 둘 다 최소한 Name 및 Geographical Position 속성을 가집니다. Weather Type 또한 사용할 수 있는 속성 Icon이 있으므로 엔티티가 될 수 있습니다. 동일한 이유로 Wind Direction 엔티티가 있을 수 있습니다. 이제, 기온과 예측일은 어디에 해당할까요? 이 두 항목은 City의 속성이 될 수 없습니다. 예측일은 City에 대한 단일 값이 아닙니다. 한 도시에 대해 여러 예측일이 있을 수 있기 때문입니다. 이러한 방식으로 Forecast와 같이 Date, Minimum 및 Maximum Temperature, Wind Force 속성을 갖는 하나의 엔티티가 여전히 누락되었음을 알 수 있습니다.

다음 사이에는 관계가 있을 수 있습니다.

- COUNTRY 및 CITY
- CITY 및 FORECAST
- FORECAST 및 WEATHER TYPE
- FORECAST 및 WIND DIRECTION

Oracle Internal & OAI Use Only

일기 예보, 솔루션



ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

정보 유형

이 엔티티 관계 도표에서 일부 가정은 관계에 대한 것입니다.

- 모든 FORECAST는 하나의 CITY에 대한 것이어야 합니다. 그리고
- 모든 CITY가 하나의 FORECAST에 있어야 하는 것은 아니며 여러 FORECAST에 있을 수 있습니다.
- 모든 CITY는 하나의 COUNTRY에 위치합니다. 그리고
- 모든 COUNTRY는 하나 이상의 CITY를 가집니다.
- FORECAST가 항상 WEATHER TYPE을 포함해야 하는 것은 아닙니다. 그리고
- 모든 WEATHER TYPE이 하나의 FORECAST에 있는 것은 아니며 여러 FORECAST에 있을 수 있습니다.
- FORECAST가 항상 WIND DIRECTION을 포함해야 하는 것은 아닙니다. 그리고
- 모든 WIND DIRECTION이 하나의 FORECAST에 있는 것은 아니며 여러 FORECAST에 있을 수 있습니다.

이러한 가정에 바탕이 되는 논리는 우리가 FORECAST가 참조하는 날짜나 CITY를 알고 있는 한 불완전한 FORECAST를 여전히 FORECAST로 간주한다는 것입니다.

ER 도표의 그래픽 요소

☑ 엔티티

☑ 속성

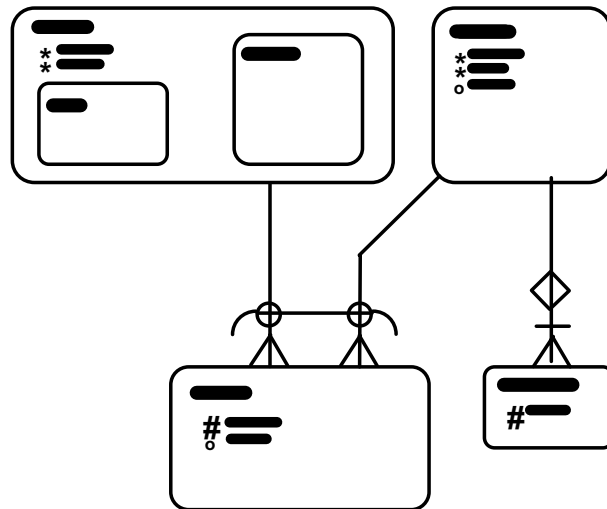
☑ 관계

하위 유형

고유한 식별자

호

비전이성



ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

기타 그래픽 요소

이 그림은 ER 도표에서 볼 수 있는 모든 그래픽 요소를 보여줍니다. 앞서 엔티티, 속성 및 관계를 나타내는 방식을 살펴 보았습니다.

후속 단원에서는 다음과 같은 나머지 네 가지 유형의 요소에 대해 논의합니다.

- 다른 엔티티 범위 내의 엔티티로 표현되는 하위 유형
- 고유 식별자 - 속성 앞의 # 또는 관계 라인 사이에서 표시줄로 나타남
- 호 - 두 개 이상의 관계 라인 사이에서 호 형태의 라인으로 나타남
- 비전이성 기호 - 관계 라인 사이에서 다이아몬드로 나타남

제한된 그래픽 요소 세트

아시다시피 ER 도표의 그래픽 요소 세트는 매우 제한적입니다. ER 모델링의 복잡성은 명확히 겉으로 드러나지 않습니다. ER 모델링의 기본 복잡성은 업무의 이해, 해당 업무에서 중요한 역할을 하는 엔티티, 엔티티를 설명하는 관련 속성 및 이들을 연결하는 관계의 인식에 달려 있습니다.

요약

- ER 모델링은 정보를 개념적으로 모델링함
- 기능적 업무 요구를 기반으로 함
- "어떻게"가 아니라 "무엇을"이 중요함
- 도표는 용이한 의사 전달 수단을 제공함
- 상세하지만 많은 데이터를 포함하지는 않음

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

요약

개념적 모델은 업무의 기능적 요구 및 정보 요구를 모델링하기 위해 생성됩니다. 이러한 모델은 현재 요구를 기반으로 할 수 있지만 미래의 요구도 반영할 수 있어야 합니다. 이 과정은 정보 요구의 모델링을 다룹니다. 기능적 요구는 데이터 모델에 적합한 유일한 기반을 형성하기 때문에 데이터 모델링 중에 무시될 수 없습니다. 이론적으로 개념적 모델을 생성할 때 구현 중 발생 가능한 기술적 문제를 고려하지 않습니다. 결과적으로 이 모델은 업무적으로 수행하는 것과 요구되는 것에만 중점을 두고 있으며 어떻게 실현되는가는 고려하지 않습니다.

엔티티 관계 모델링은 정보 요구를 포착하기 위해 제대로 구성된 기법입니다. ER 모델은 기술적 데이터 모델에 대한 기반을 형성합니다. 기술적 고려 사항은 해당 레벨에서 발생합니다.

엔티티 관계 도표는 읽기 쉽고 비교적 생성하기 쉬운 도표 형태로 ER 모델을 나타냅니다. 이러한 도표는 초기에는 업무 요구를 논의하기 위한 기반을 형성하며 차후에는 향후 시스템에 대한 최상의 가능한 맵을 제공합니다.

이 도표는 상당수의 세부 정보를 보여주지만 혼란스러울 만큼 상세히 나타내지는 않습니다.

연습

- 인스턴스 또는 엔티티
- 투숙객
- 읽기
- 호텔
- 조리법

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

Oracle Internal & OAI Use Only

연습: 인스턴스인가 엔티티인가?

개념	E/A/I?	인스턴스 또는 엔티티 예제
PRESIDENT		
ELLA FITZGERALD		
DOG		
ANIMAL		
HEIGHT		
	E	CAR
	A	CAR
	I	CAR

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

연습 1-1: 인스턴스 또는 엔티티

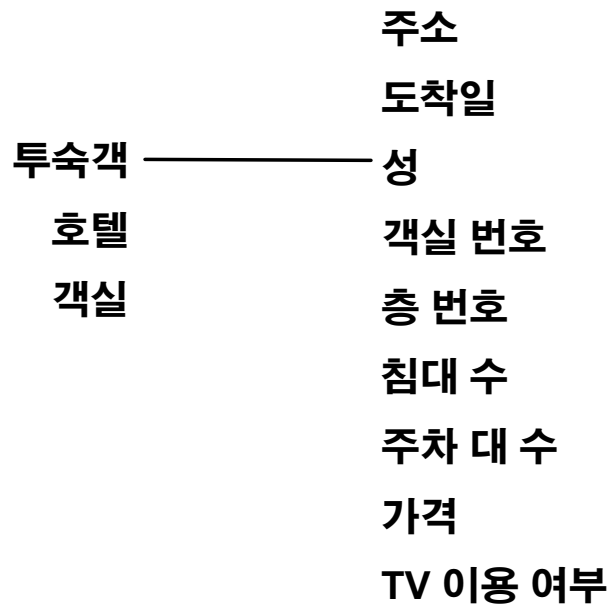
목표

이 연습의 목표는 엔티티, 속성 및 엔티티의 인스턴스를 구분하는 방법을 배우는 것입니다.

과제

다음 개념 중 엔티티, 속성, 인스턴스에 해당한다고 여기는 것을 각각 나열하십시오. 엔티티로 생각하는 경우 예제 인스턴스를 제시하십시오. 속성 또는 인스턴스로 생각하는 경우 엔티티를 제시하십시오. 마지막 세 행에 대해 적절한 개념을 찾으십시오.

연습: 투숙객



ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

연습 1-2: 투숙객

목표

이 연습의 목표는 엔티티에 대한 속성을 인식하는 것입니다.

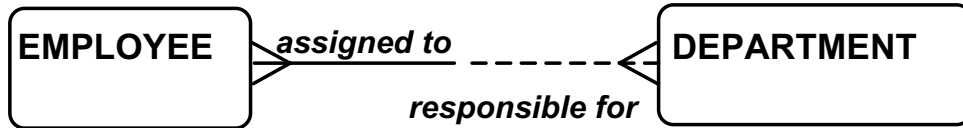
시나리오

그림 왼쪽에는 호텔 환경에서 중요한 역할을 하는 투숙객, 호텔, 객실이라는 엔티티 세 개가 있습니다. 오른쪽에는 선택할 수 있는 속성이 표시됩니다.

과제

속성과 해당 속성이 설명하는 엔티티 간에 선을 그립니다.

연습: 읽기



- A 각 EMPLOYEE는 하나 이상의 DEPARTMENT에 할당될 수 있습니다.
각 DEPARTMENT는 하나 이상의 EMPLOYEE를 책임져야 합니다.
- B 각 EMPLOYEE는 하나 이상의 DEPARTMENT에 할당되어야 합니다.
각 DEPARTMENT는 하나 이상의 EMPLOYEE를 책임질 수 있습니다.
- C 각 EMPLOYEE는 정확히 하나의 DEPARTMENT에 할당되어야 합니다.
각 DEPARTMENT는 정확히 하나의 EMPLOYEE를 책임질 수 있습니다.

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

연습 1-3: 읽기

목표

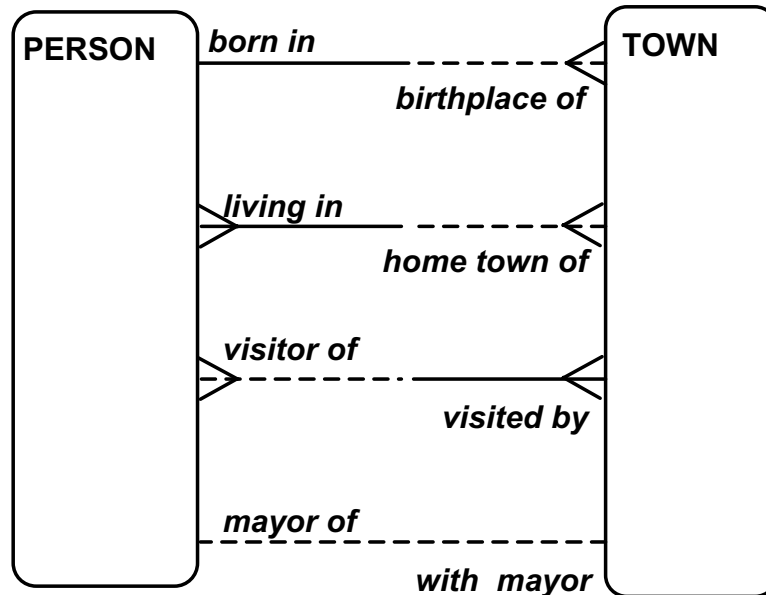
이 연습의 목표는 관계를 읽는 것입니다.

과제

어느 내용이 도표에 해당합니까?

Oracle Internal & OAI Use Only

연습: 읽기 및 주석 달기



ORACLE

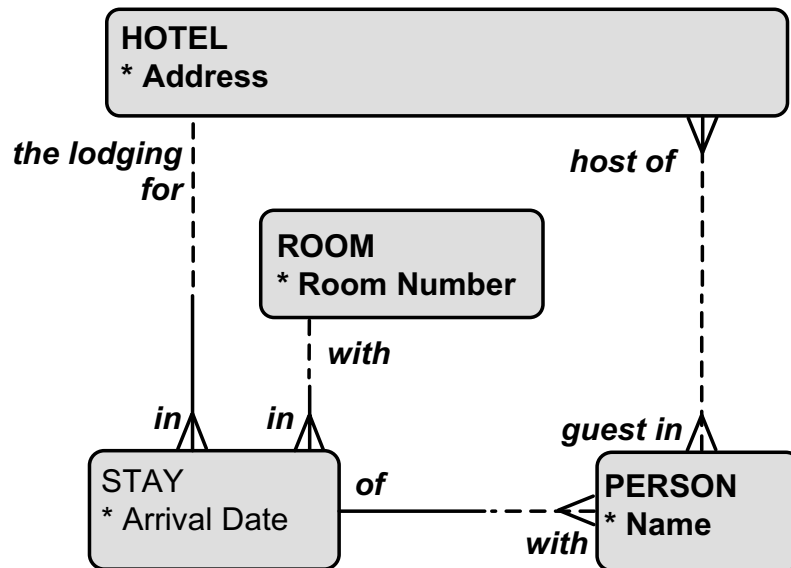
Copyright © Oracle Corporation, 2002. All rights reserved.

연습 1-4: 읽기 및 주석 달기

과제

1. 여기 제시된 모델에서 각 관계를 읽어 보십시오.
2. 다음으로 방금 읽은 관계에 주석을 달니다. 사람 및 도시에 대한 지식을 사용하십시오.

연습: 호텔



ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

연습 1-5: 호텔

과제

1. 여기 제시된 모델의 관계에 대해 주석을 달니다.
2. 호텔 업무에 다소 도움이 될 수 있는 PERSON과 HOTEL 간의 가능한 관계 두 가지를 추가로 구성합니다.

Ralph's Raving Recipes	
Soups	Açorda alentejana bread soup from Portugal
Vegetarian 15 min easy	For 4 persons: 1 onion 4 cloves of garlic 1 red pepper 1 liter of vegetable broth 4 tablespoons of olive oil 4 fresh eggs 1 handful of parsley or coriander salt, pepper 9-12 slices of (old) bread
Preparation	Cut the onion into small pieces and fry together with the garlic. Wash the red pepper, cut it in half, remove the seeds and fry it for at least 15
page 127	
ORACLE	
Copyright © Oracle Corporation, 2002. All rights reserved.	

연습 1-6: 조리법

목표

이 연습의 목표는 지정된 정보 소스에 제시된 다양한 정보 유형을 파악하는 것입니다.

시나리오

사용자는 웹에서 조리법을 이용할 수 있도록 하려는 출판사의 한 분석가입니다. 이 회사는 대중이 아주 쉬운 방법으로 조리법을 검색할 수 있게 하려 합니다. 쉬운 방법에 대한 아이디어는 높은 평가를 받고 있습니다.

과제

- Ralph의 저명한 Raving Recipe 책자의 예제 페이지를 분석하고 중요하다고 여기는 여러 다른 정보 유형을 나열합니다.
- 다양한 정보 유형을 엔티티 및 속성으로 그룹화합니다.
- 찾아낸 관계에 이름을 지정하고 도표를 그립니다.

Oracle Internal & OAI Use Only

엔티티 및 속성 세부 사항

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

Oracle Internal & OAI Use Only

개요

- 데이터와 정보의 비교
- 엔티티 및 엔티티 추적 방법
- 속성
- 하위 유형 및 상위 유형

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

소개

이 단원에서는 엔티티 및 속성에 대한 세부 정보를 제공하고 다양한 정보 소스에서 엔티티와 속성을 추적하는 방법을 설명합니다. 이 단원에서는 엔티티 정의의 변화 과정과 하위 유형 및 상위 유형 엔티티의 개념을 살펴봅니다. 또한 이 설명서의 여러 예제에서 사용되고 있는 가상의 회사인 ElectronicMail Inc.를 소개합니다.

목표

이 단원을 마치면 다음을 수행할 수 있습니다.

- 다양한 정보 소스에서 엔티티 추적
- 다양한 정보 소스에서 속성 추적
- 정보를 엔티티 또는 속성으로 모델링해야 하는 경우 결정
- 하위 유형 및 상위 유형 모델링

데이터와 정보의 비교

데이터

- 다른 사실을 유추할 수 있는 기존의 사실
- 원시 자료
- 예: 전화 번호부

정보

- 지식, 지능
- 예: 꽃가게 전화 번호

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

데이터와 정보의 비교

데이터와 정보, 이 두 단어는 자주 동의어처럼 사용됩니다. 하지만 의미는 서로 다릅니다.

데이터:

결론을 도출해 낼 수 있는 원시 자료입니다. 새로운 사실을 유추할 수 있는 사실입니다. 일반적인 예로 전화 번호부를 들 수 있습니다. 이것은 특정 내부 구조를 가지는 사실들의 거대한 집합체입니다.

정보:

지식, 지능 또는 특수한 의미나 기능을 갖는 특정 데이터입니다. 정보는 주로 데이터가 됩니다. 바꿔 말하면 정보는 주로 데이터에서 파생된 결과로서, 데이터의 특정 부분이 될 수 있습니다. 데이터가 특정 방식으로 구성되면 정보를 찾는 데 매우 유용합니다. 전화 번호부 데이터 예제를 확장해 보면 정보는 치과 의사의 전화 번호 또는 동료의 집 주소에 해당합니다.

데이터

- **개념적 모델링**
데이터 개념을 논리적이고 일관된 방식으로 상호 관련된 그룹으로 구조화
- **물리적 모델링**
(향후) 물리적 데이터베이스의 구조 모델링
- **기본**
일반적으로 종이 및 전자 기반과 같이 다양한 형식으로 된 일련의 데이터
- **웨어하우스**
거대한 정보 집합체

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

데이터

개념적 데이터 모델링

개념적 데이터 모델링은 업무 정보의 구조와 이러한 구조를 통제하는 규칙을 결정하기 위해 업무 및 업무 데이터를 검토하는 것입니다. 이 구조는 나중에 업무 데이터의 저장소를 정의하는 기반으로 사용될 수 있습니다. 개념적 데이터 모델링은 기술적 구현과는 별개입니다. 이러한 이유로, 업무가 일정 기간 동안 서서히 변경되더라도 개념적 데이터 모델은 장기간에 걸쳐 비교적 안정된 상태를 유지합니다. 개념적 데이터 모델링을 정보 엔지니어링이라고도 합니다.

물리적 데이터 모델링

물리적 데이터 모델링은 주어진 기술적 소프트웨어 및 하드웨어 환경에서의 구현과 관련이 있습니다. 물리적 구현은 기술의 현상태에 크게 의존하며 급속한 기술 변화에 좌우됩니다. 5년 전의 기술적 설계는 현재 상당히 구식이 되어 있을 것입니다.

개념적 모델과 물리적 모델을 구분하여 설계에서 비교적 안정적인 부분과 덜 안정적인 부분을 구분할 수 있습니다. 이것은 데이터 모델과 기능적 사양에 모두 적용됩니다.

데이터베이스

데이터베이스는 일련의 데이터입니다. 데이터의 다양한 부분은 일반적으로 종이 및 전자 기반과 같이 여러 형태로 사용할 수 있습니다. 예를 들어, 전자 기반의 데이터는 스프레드시트, 각종 파일 또는 일반 데이터베이스에 존재할 수 있습니다. 오늘날, 관계형 데이터베이스가 보편화되었지만 여전히 많은 구형 시스템이 사용되고 있습니다. 구형 시스템은 대개 계층적 데이터베이스와 네트워크 데이터베이스입니다. 좀더 최근의 시스템은 의미형 데이터베이스와 객체 지향 데이터베이스입니다.

데이터 웨어하우스

데이터 웨어하우스는 하나의 논리적 데이터베이스에 배치된 여러 소스의 데이터로 구성됩니다. 이 데이터 웨어하우스 데이터베이스(또는 좀더 정확하게 이 데이터베이스 구조)는

OLAP(Online Analytical Processing) 작업에 맞게 최적화되어 있습니다.

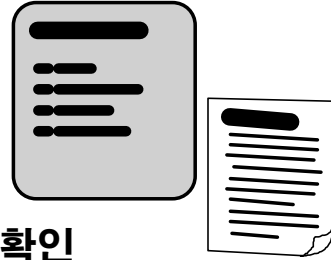
주로 데이터 웨어하우스는 다른 소스에서 입수되는 추가 정보와 일상 트랜잭션 시스템에서 제공되는 요약된 데이터를 포함합니다. 라우팅 시스템에 대한 트래픽 로드를 추적하는 전화 회사를 예로 들 수 있습니다. 이 시스템은 개별 전화 통화를 저장하지 않지만 시간별로 요약된 데이터를 저장합니다.

데이터 분석 관점에서 볼 때, 데이터 웨어하우스는 여타의 것들과 유사한 하나의 데이터베이스에 불과하지만 매우 고유하고 특징적인 기능상의 요구 사항을 가지고 있습니다.

Oracle Internal & OAI Use Only

엔티티

- 엔티티에 고유한 이름 부여
- 엔티티의 공식적 설명 생성
- 가능한 경우 몇몇 속성 추가
- 동음 이의어 인식
- 엔티티 이름과 설명을 정기적으로 확인
- 예약어 사용 피하기
- 엔티티 이름에서 관계 이름 제거



ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

엔티티 추적

예를 들어, 텍스트, 메모, 브로셔 및 화면에 포함된 업무와 관련된 명사는 주로 엔티티, 엔티티의 속성 또는 엔티티의 인스턴스를 나타냅니다.

엔티티에 고유한 이름 지정

먼저 개념을 정리하여 엔티티를 구분합니다. 다음으로 엔티티에 대한 고유하고 명확한 이름을 찾아봅니다. 명확한 이름보다 개념이 훨씬 더 많으므로 이름을 지정하는 것이 항상 쉬운 일은 아닙니다. 상상력을 동원하십시오. 사전을 사용하십시오. 단어의 조합을 사용하고 필요한 경우 'X'를 사용하십시오. 하지만 좋은 이름이 없다고 해서 모델링을 중단하지는 마십시오. 시간이 흐르면서 좋은 이름도 변화할 것입니다.

때때로 사용한 이름들을 확인합니다. 엔티티의 암시적 정의는 분석 중에, 예를 들면 속성을 추가하거나 관계의 선택 가능성을 변경할 경우에 달라질 수 있습니다.

공식적인 설명 생성

엔티티의 공식적인 설명을 생성합니다. 이 작업은 그다지 어렵지 않으며 직접 기술해 봄으로써 대상에 대한 개념을 명확히 할 수 있습니다. 이 설명을 정기적으로 확인합니다. 때때로 모델링 과정 중에 개념이 달라질 수 있습니다. 물론 이러한 변화에 따라 정의도 달라져야 합니다.

동의어 인식

동일한 개념도 다양한 업무적 상황에서 다른 이름으로 표현될 수 있습니다. 이런 경우에는 "...라고도 함"과 같이 동의어임을 표시해 두는 것이 좋습니다.

동음 이의어 피하기

업무상 한 단어가 여러 다른 개념으로 사용되곤 합니다. 다음 예제에서 볼 수 있듯이 동일한 사람도 때로 같은 단어를 여러 다른 의미로 사용할 수 있습니다.

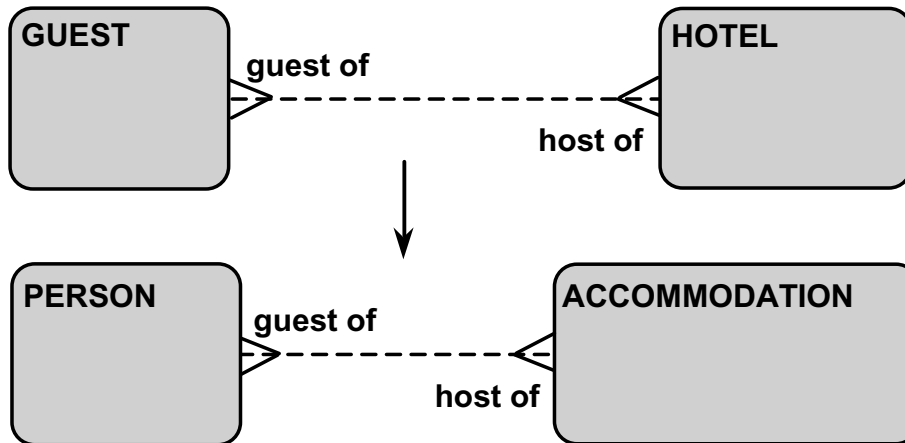
"여러분이 지금 참여하고 있는 데이터 모델링 과정(course)은 1999년에 작성되었으며 모델링 기술이 필요합니다." 이 문장에서 "과정(course)"이라는 단어는 실제 강의(오늘 참여하고 있는 것과 같은 형태), 학습 자료(1999년에 작성된) 및 과정 유형(특정 기술이 필요한)의 세 가지 다른 개념을 나타냅니다.

예약어 피하기

어떤 이름이든 엔티티 이름으로 사용할 수 있지만 가능하다면 엔티티 이름으로 데이터베이스 및 프로그래밍 용어를 사용하지 않아야 합니다. 이렇게 해야 설계 후반부에 이름 지정 문제나 혼란을 방지할 수 있습니다.

Oracle Internal & OAI Use Only

엔티티 이름의 관계 이름



ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

엔티티 이름에서 관계 이름 제거

다소 일반적인 방법으로 엔티티 이름을 선택할 수 있습니다. 이 예제에서 두 도표는 동일한 상황을 모델링합니다. 첫번째 모델에서 "guest" 부분은 관계 이름의 일부일 뿐 아니라 엔티티 이름의 일부이기도 합니다.

두번째 모델은 좀더 일반적인 방법으로 이름이 지정되어 있습니다. 여기서 guest는 guest의 역할을 하는 PERSON으로 표시됩니다.

일반적으로 선택의 여지가 있으면 좀더 일반적인 이름을 사용하십시오. 예를 들어, 동일한 엔티티 간에 person이 해당 숙박 시설의 직원이거나 주주임을 보여주는 두번째 관계를 추가할 수도 있습니다. 첫번째 모델을 사용할 경우에는 엔티티를 새로 만들어야 합니다.

이 주제는 하위 유형 및 롤의 개념과 밀접한 관계가 있습니다. 이 단원 뒷부분에서 패턴에 대해 논의할 때 이 내용을 좀더 자세히 확인할 수 있습니다.

일부 배경 정보

"ElectronicMail(EM)에서는 매력적이며 사용자에게 친숙한 웹 기반 전자 메일 시스템을 제공하려고 합니다. 중요한 개념은 사용자와 메시지입니다.

EM 사용자는 해당 EM 사용자를 설정한 사람이 제공한 30자 이내의 고유 주소와 암호를 가집니다. 이름, 국적, 생일, 업종 등과 같은 일부 추가 정보를 요청하는 경우에도 그 사람이 실제로 누구인지 알 수 없습니다.

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

전자 메일 예제

이 과정에서는 다양한 업무 상황을 조사해 봅니다. 한 가지 상황은 전자 메일 서비스를 제공하는 회사인 ElectronicMail에 대한 것입니다. 다음에 일부 배경 정보가 제공됩니다.

Oracle Internal & OAI Use Only

일부 배경 정보

사용자는 메일 메시지를 주고 받을 수 있어야 합니다. 메일 메시지는 일반적으로 단순한 텍스트로 이루어져 있지만 파일이 첨부되어 있을 수 있습니다. 첨부된 내용은 메시지와 함께 전송되고 보관되지만 당사의 소프트웨어로 생성되지는 않은 스프레드시트 같은 파일입니다.

메시지는 폴더에 보관됩니다. 모든 사용자는 처음에 Inbox, Outbox 및 Wastebasket의 세 폴더를 가집니다. 사용자가 추가 폴더를 생성할 수 있습니다."

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

Oracle Internal & OAI Use Only

EM logo

advertisement area1

Compose Template: default

Subject: test Send

To: bipi.giovanni.papini@yahoo.co Save Draft

Cc: myself Save Template

Bcc: Cancel

Message text: as well
arala
pompidom

Keep Copy ☐

☐ Add Signature

Attachments: Type:

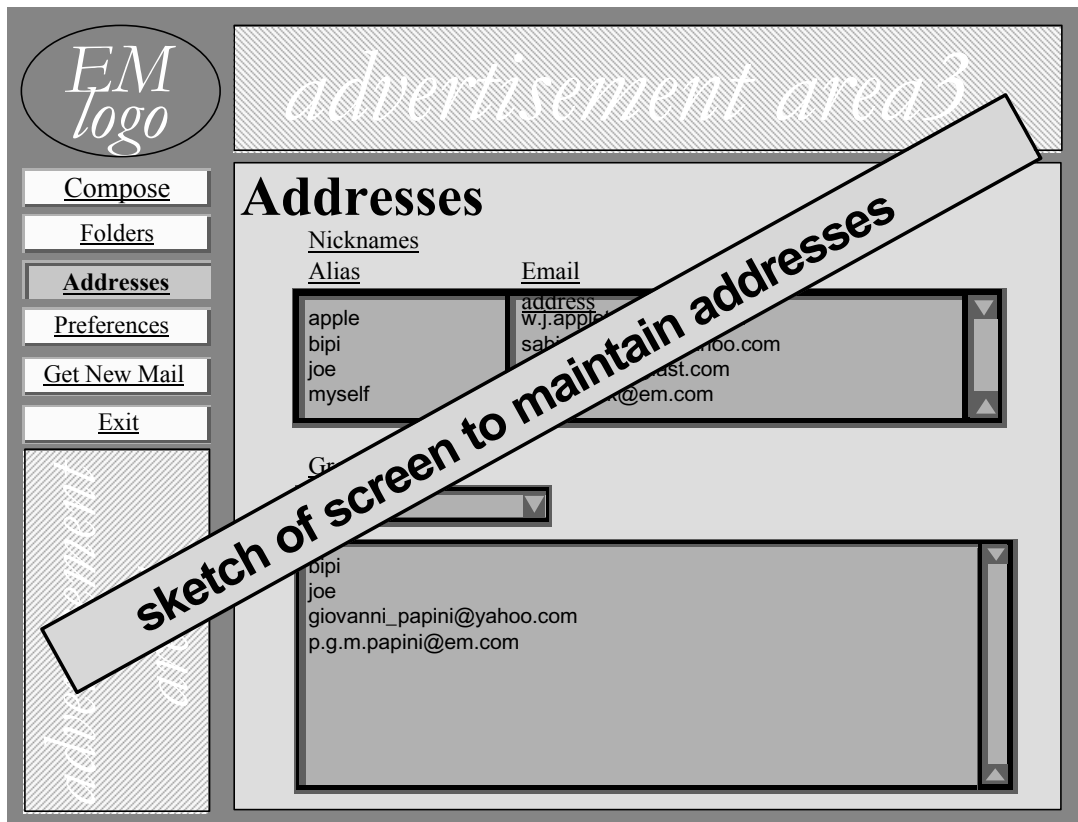
abc.html	Hypertext
xyz.doc	Word document

advertisement area2

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

Oracle Internal & OAI Use Only



ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

전자 메일 예제

이 스크린샷을 통해 Compose a Mail Message 화면과 Maintain Addresses 화면이 어떤 형태인지 이해할 수 있습니다.

일부 요구되는 기능

- "ElectronicMail 사용자는 전자 메일 주소 그룹과 같은 메일 목록으로 메시지의 주소를 지정할 수 있어야 합니다. 시스템은 데이터베이스 공간을 절약하기 위해 전송된 메시지의 단일 복사본과 메시지 수신자에 대한 정보만 보관해야 합니다.
- 사용자는 본인의 메시지에 대한 템플릿을 작성할 수 있어야 합니다. 템플릿은 이름이 지정되어야 하며 실제 메시지에 포함된 모든 내용을 포함할 수 있습니다. 템플릿은 새 메시지에 사용될 수 있습니다.

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

Oracle Internal & OAI Use Only

일부 요구되는 기능

- 사용자는 메시지에 회신할 수 있어야 합니다. 사용자는 회신함으로써 이전 메시지가 추가된 새 메시지를 생성합니다.
- 사용자는 기억하기 쉬운 애칭 뒤에 복잡한 주소를 숨기기 위해 전자 메일 주소에 대한 별칭을 생성할 수 있어야 합니다."

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

Oracle Internal & OAI Use Only

엔티티 정의의 변화

- 메시지는 사용자에게 의해 전송된 텍스트입니다.
- 메시지는 EM 사용자에게 의해 전송된 텍스트입니다.
- 메시지는 EM 사용자에게 의해 전송된 메모입니다. 메시지는 텍스트 또는 제목 등을 반드시 포함하지는 않습니다.
- 메시지는 EM 사용자에게 의해 전송되었거나 EM 사용자에게 의해 수신된 메모입니다. 메시지는 텍스트 또는 제목 등을 반드시 포함하지는 않습니다.
- 메시지는 EM 사용자에게 의해 수신된 메모입니다. 메시지는 텍스트 또는 제목 등을 반드시 포함하지는 않습니다.

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

엔티티 정의의 변화

개념의 변화를 설명하기 위해 ElectronicMail의 MESSAGE 엔티티를 고려해 보십시오. 이 엔티티에 대한 직관적인 설명은 다음과 같습니다.

누구나 사용할 수 있습니까? 아마 아닐 것입니다.

모든 메시지에 텍스트가 포함되어야 합니까? 텍스트 없이 첨부 파일만 운반하는 메시지는 전송할 수 있습니까?

EM 사용자가 외부 소스로부터 수신한 메시지의 경우는 어떻습니까? 보관할 필요가 있습니까?

이제 EM 사용자가 메시지를 외부 전자 메일 주소로만 전송하는 경우를 가정해 봅시다. EM 사용자는 메일 메시지 복사본을 보관하려고 하지 않습니다. 이 경우 해당 메시지를 필요로 하는 내부 EM 사용자가 없으므로 시스템에서 메시지를 보관할 필요가 없습니다. 사실, EM 사용자가 전송한 메시지를 보관하는 것은 중요하지 않으며 EM 사용자가 실제로 받은 메시지만 중요합니다.

여기에 표시된 사고 과정은 최초 아이디어에서 잘 다듬어진 상태로 정의가 변화하는 일반적인 과정입니다. 물론 잘 다듬어진 상태의 정의가 최종 정의라는 뜻은 아닙니다.

엔티티 수명 주기

엔티티의 수명 주기에 대해 생각할 경우 대상을 명확하게 파악하는 것이 도움이 됩니다. 수명 주기는 엔티티의 기능적 단계를 나타냅니다. 예를 들어, 엔티티 인스턴스가 어떻게 발생합니까? 어떻게 변경됩니까? 어떻게 사라집니까?

MESSAGE 엔티티의 경우 다음과 같은 질문을 할 수 있습니다.

- 언제 "어떤 것"이 메시지가 됩니까?
- 언제 메시지가 변경됩니까?
- 언제 메시지를 제거할 수 있습니까?

메시지 생성

compose 화면에 텍스트를 입력할 때 이 텍스트는 메시지가 됩니까? To 또는 Subject 필드와 같은 일부 필드를 완성하기 전에 이를 메시지로 고려하는 것이 타당하지 않다고 생각할 수 있습니다. send 키를 누르면 확인이 수행됩니다. 모든 확인이 수행되어야만 메시지가 탄생됩니다.

메시지 제거

시스템에서는 언제 메시지를 제거할 수 있습니까? 사용자는 언제 Delete 키를 누릅니까? 동일한 메시지의 다른 수신자가 있을 때 시스템에서는 어떤 작업을 수행해야 합니까? 메시지의 삭제는 사용자가 더 이상 메시지를 읽을 권한이 필요하지 않다는 것을 알리기 위해 시스템에 보내는 신호로 보는 것이 더 적절합니다. 동일한 메시지를 수신한 모든 사용자가 이 작업을 수행한 후에야 메시지를 삭제할 수 있습니다. 명확히 말해서, 메시지가 존재하려면 여전히 해당 메시지를 필요로 하는 수신자가 있어야 합니다.

메시지 변경

메시지를 변경하려고 합니까? 텍스트가 전송되지 않는 한, 텍스트는 메시지가 아닌 것으로 간주되기 때문에 아무 문제가 되지 않습니다. 메시지를 전송한 후에 변경하려고 합니까? 이전에 이미 전송했던 어떤 것을 변경하려고 합니까? 그렇게는 할 수 없습니다. 텍스트를 변경하면 새 메시지가 생성됩니다.

초안

아직 전송할 준비가 되지 않은 메시지의 경우는 어떻습니까? 사용자가 나중에 메시지를 완성하기 원하는 경우를 가정해 봅시다. 이러한 메시지를 보관할 장소가 있습니까? 시스템에 전송되지 않았거나 완성되지 않은 메시지를 보관하려고 합니까? DRAFT는 MESSAGE 엔티티의 특수한 경우입니까 아니면 DRAFT를 별도의 엔티티로 취급해야 합니까?

템플릿

템플릿의 경우는 어떻습니까? 템플릿은 메시지가 될 수 있는 모든 것에 대한 것이지만 템플릿은 메시지에 대한 일종의 스탬프로서만 사용됩니다. 템플릿은 이름이 있지만 메시지는 이름이 없습니다. TEMPLATE는 MESSAGE 엔티티의 특수한 경우입니까 아니면 별도의 엔티티로 간주해야 합니까?

업무 기능

- "ElectronicMail 사용자는 전자 메일 주소 그룹과 같은 메일 목록으로 메시지의 주소를 지정할 수 있어야 합니다. 시스템은 데이터베이스 공간을 절약하기 위해 전송된 메시지의 단일 복사본과 메시지 수신자에 대한 정보만 보관해야 합니다.
- 사용자는 본인의 메시지에 대한 템플릿을 작성할 수 있어야 합니다. 템플릿은 이름이 지정되어야 하며 실제 메시지에 포함된 모든 내용을 포함할 수 있습니다. 템플릿은 새 메시지에 사용될 수 있습니다.

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

기능

앞서 제시한 엔티티 정의의 변화에서, 정의는 메시지 관련 시스템의 기능을 고찰하는 과정에서 변화되었습니다. 이것은 앞서 소개한 "기능은 개념적 데이터 모델을 구동함"이라는 표현을 설명해 줍니다.

시스템 기능 또는 원하는 기능에 대한 초기 아이디어는 설명적 텍스트와 인터뷰 메모 등의 동사에서 추출할 수 있습니다. 위의 텍스트에서 기능은 자세한 설명 없이 상위 레벨에서 표현되었습니다. 그렇지만 좀더 자세한 기능을 상상해 볼 수 있습니다.

이 과정에서 기능은 주로 암시적으로 가정되며 때로는 자세히 설명되기도 합니다.

업무 기능

- 사용자는 메시지에 회신할 수 있어야 합니다. 사용자는 회신함으로써 이전 메시지가 추가된 새 메시지를 생성합니다.
- 사용자는 기억하기 쉬운 애칭 뒤에 복잡한 주소를 숨기기 위해 전자 메일 주소에 대한 별칭을 생성할 수 있어야 합니다."

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

Oracle Internal & OAI Use Only

속성...

- 항상 "of what(무엇의 속성)"인지 표시합니다.
- 관계가 아닌 엔티티의 속성(property)입니다.
- 단일 값을 가져야 합니다.
- 다음과 같은 형식을 가집니다.
 - 문자열
 - 숫자
 - 날짜
 - 그림
 - 사운드
- 데이터의 기본 요소입니다.

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

속성 추적

앞에서 논의한 것처럼, 텍스트, 메모, 브로셔 및 보고 있는 화면에 포함된 업무와 관련된 명사는 주로 엔티티, 엔티티의 속성 또는 엔티티의 인스턴스를 나타냅니다. 일반적으로 "Of what(무엇의 속성인가?)"과 "Of what format(무슨 형식의 속성인가?)"이라고 질문함으로써 쉽게 속성을 인식할 수 있습니다. 속성은 해당 속성이 속하는 엔티티를 설명, 정량 분석, 한정, 분류, 지정하거나 엔티티에 상태를 부여합니다. 속성은 엔티티의 속성(property)으로 정의되며 이것은 속성 자체만 있는 개념은 없다는 것을 의미합니다.

아래에 표시된 ElectronicMail에 대한 배경 정보 내용에서, 처음 나온 엔티티는 대문자로 표시되며 속성은 상자 안에 포함되고 인스턴스는 기울임꼴로 표시됩니다.

명사, 엔티티, 속성

- "ElectronicMail(EM)에서는 매력적이며 사용자에게 친숙한 웹 기반 전자 메일 시스템을 제공하려고 합니다. 중요한 개념은 사용자와 메시지입니다.
- *EM USER*는 해당 EM 사용자를 설정한 *PERSON01*이 제공한 30자 이내의 고유 주소와 암호를 가집니다. *name*, *COUNTRY*, *birth date*, *line of business* 등과 같은 일부 추가 정보를 요청하는 경우에도 그 사람이 실제로 누구인지 알 수 없습니다.

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

속성 추적

정보 유형을 나열하고, 가능한 엔티티 및 속성을 구분하고 그룹화합니다. 필요한 경우 예제에 속성(예: COUNTRY의 Name 속성)을 추가합니다. 인스턴스에서 하나 이상의 속성(예: FOLDER의 Name 속성)을 제거합니다.

Oracle Internal & OAI Use Only

명사, 엔티티, 속성

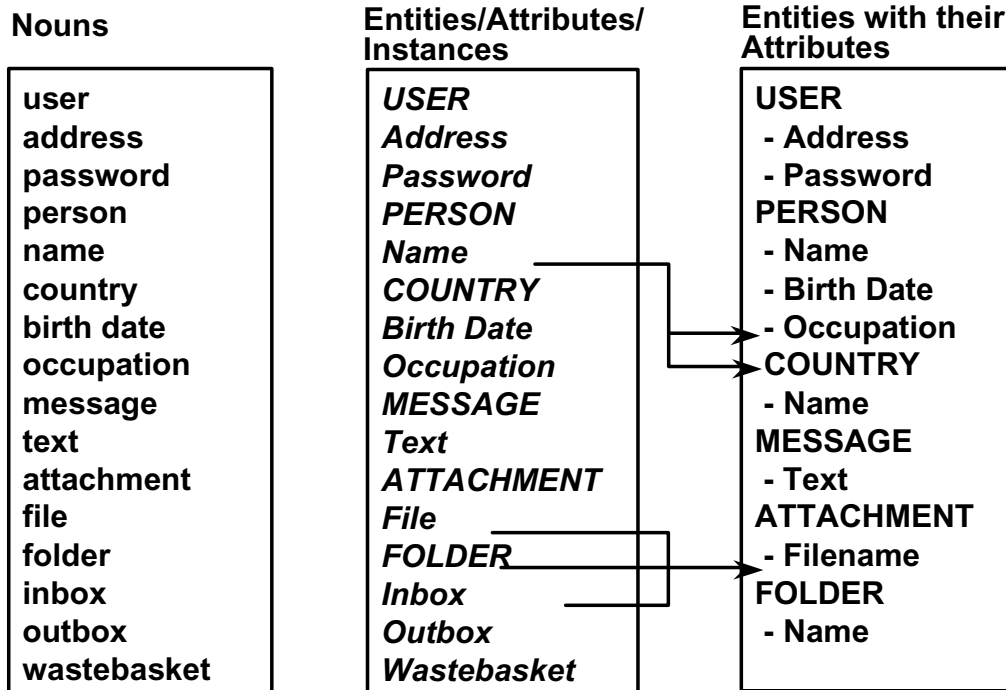
- 사용자는 메일 MESSAGE를 주고 받을 수 있어야 합니다. 메일 메시지는 일반적으로 단순한 텍스트로 이루어져 있지만 파일이 첨부되어 있을 수 있습니다. ATTACHMENT는 메시지와 함께 전송되고 보관되지만 우리의 소프트웨어로 생성되지는 않은 스프레드시트 같은 파일입니다.
- 메시지는 FOLDER에 보관됩니다. 모든 사용자는 처음에 Inbox, Outbox 및 Wastebasket의 세 폴더를 가집니다. 사용자가 추가 폴더를 생성할 수 있습니다.

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

Oracle Internal & OAI Use Only

EM 엔티티 및 속성



ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

속성 이름 지정

속성 이름은 이후 단계에서 열 이름의 후보가 됩니다. 열 이름은 규칙을 따라야 합니다. 예약어를 사용하지 않으면서 속성 이름을 지정해 보십시오.

미리 지정되지 않은 경우에는 약어를 사용하지 마십시오. 자주 사용되는 약어의 예로 Id, No, Descr, Ind(icator)를 들 수 있습니다.

Amount, Value, Number와 같은 속성 이름을 사용하지 마십시오. 다음과 같이 항상 속성 이름의 의미를 나타내는 설명을 추가하십시오. Amount Paid, Estimated Value, Licence No.

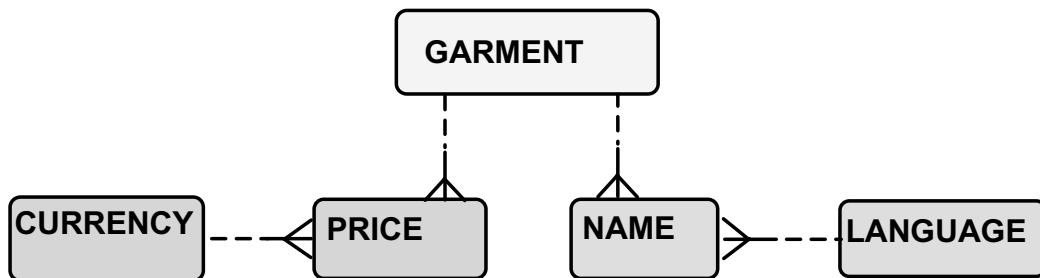
항상 동일한 위치에 "date" 또는 "indicator"와 같이 속성 이름의 자주 사용되는 이름 구성 요소를 추가합니다. 예를 들어, Start Date, Creation Date 및 Purchase Date는 끝에 Date가 표시된 경우입니다.

둘 이상의 단어로 구성되는 속성 이름에는 밑줄을 사용하지 마십시오. 엔티티 이름처럼 속성 이름은 최대한 명확하고 이해하기 쉬워야 합니다.

속성 및 엔티티



한 모델의 속성이 다른 모델에서 엔티티일 수 있습니다.



ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

속성과 엔티티 비교

경우에 따라, 한 상황에서 속성인 정보가 다른 상황에서는 엔티티일 수 있습니다. 이것은 순전히 업무와 관련되어 있습니다. Name과 같은 일반 속성을 엔티티로 모델링해야 할 수 있습니다. 이런 경우는 모델에서 언어와 같은 추가 차원이 필요한 경우에 발생합니다. 제품 이름을 여러 언어로 관리하고 제품 가격을 여러 통화 단위로 관리해야 하는 경우에는 한 제품이 여러 이름을 가진다는 사실을 알 수 있습니다. 예를 들어, "이 특정 의류 품목은 영문으로 'Acapulco swimming trunks'이고 독일어로는 'Akapulko Badehose'입니다."

일반적으로 볼 수 있는 차원은 시간입니다. 이것에 대해서는 이후에 설명합니다.

중복성

COMMODITY

- * Name
- * Price exclusive VAT
- * Price inclusive VAT
- * VAT %

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

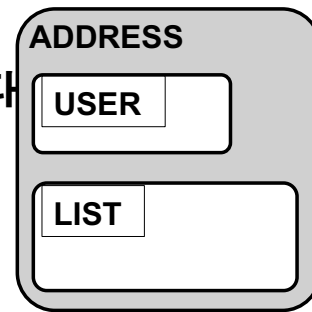
중복성

중복된 속성 즉, 다른 속성값에서 파생될 수 있는 속성값을 사용하지 않도록 특히 주의해야 합니다. 아래에 예제가 제공됩니다. 파생 가능한 정보를 사용하는 것은 일반적으로 물리적인 설계 결정에 따른 것입니다. Date Instance Created 및 User Who Modified와 같은 감사 유형 속성에 대해서도 마찬가지입니다.

Oracle Internal & OAI Use Only

하위 유형...

- 상위 유형의 모든 속성을 상속합니다.
- 상위 유형의 모든 관계를 상속합니다.
- 일반적으로 자체의 속성이나 관계 또는 업무 기능을 가집니다.
- 상위 유형 내에 그려집니다.
- 절대 단독으로 존재하지 않습니다.
- 자체의 하위 유형을 가질 수 있습니다.
- "하위 엔티티"라고도 합니다.



ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

하위 유형 및 상위 유형

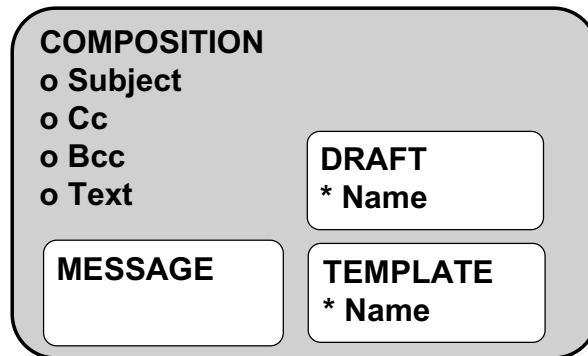
경우에 따라 X 엔티티를 하위 유형으로 세분하는 것이 적절할 수 있습니다. 인스턴스의 그룹이 해당 그룹에 대해서만 존재하는 속성(attribute)이나 관계와 같은 특수 속성(property)을 갖거나 속성 값 중 하나에 대해 고정 값을 가질 때 또는 그룹에만 적용되는 기능이 있을 때가 여기에 해당될 수 있습니다. 이러한 그룹을 X의 하위 유형이라고 하며, X 엔티티는 상위 유형이라고 합니다. 하위 유형은 특정 제약 조건이 해당 하위 유형에만 적용될 때도 모델링됩니다. 이 내용은 제약 조건에 대한 단원에서 좀더 자세히 다룹니다.

하위 유형은 X의 모든 속성(property)을 가지며 일반적으로 추가 속성을 가집니다. 이 예제에서, 상위 유형 ADDRESS는 두 개의 하위 유형인 USER와 LIST로 나뉘어집니다. USER와 LIST 하위 유형의 공통점은 NAME 속성을 가진다는 점과, 메시지를 작성할 때 To 필드에서 두 하위 유형이 모두 사용될 수 있다는 기능적인 사실입니다.

상속

다음 그림에서 MESSAGE, DRAFT 및 TEMPLATE의 상위 유형인 COMPOSITION은 새 엔티티입니다. 하위 유형은 일반적으로 여러 개의 공통 속성을 갖습니다. 이러한 공통 속성은 상위 유형 레벨에 나열됩니다. 관계에서도 이와 동일합니다. 하위 유형은 상위 유형 엔티티의 모든 속성과 관계를 상속합니다.

하위 유형: 예제



ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

하위 유형: 예제

도표를 다음과 같이 읽으십시오.

- 모든 MESSAGE(DRAFT 또는 TEMPLATE)는 COMPOSITION입니다.
- 따라서 Subject 및 Text와 같은 속성을 갖습니다. 반대로 하면 다음과 같습니다.
- 모든 COMPOSITION은 MESSAGE, DRAFT 또는 TEMPLATE입니다.

하위 유형: 규칙

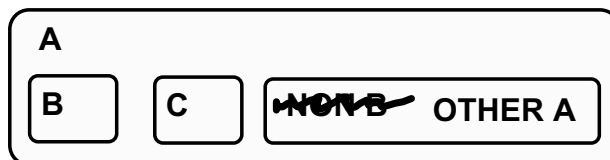
동일한 엔티티의 하위 유형은 다음과 같아야 합니다.

- **완전함:**
상위 유형의 모든 인스턴스 또한 하위 유형 중 하나의 인스턴스입니다.

그리고

- **상호 배타적임:**
상위 유형의 모든 인스턴스는 오직 하나의 하위 유형에 대한 것입니다.

하위 유형의
이름을 적절히
지정합니다.



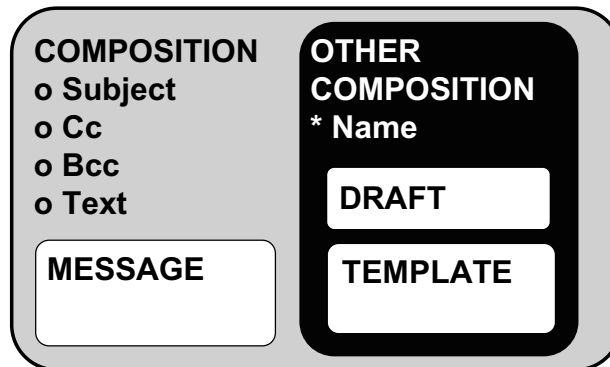
ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

항상 둘 이상의 하위 유형이 있음

엔티티 관계 모델링은 완전한 ER 모델에서 하위 유형은 절대 단독형이 아니라는 사실을 규정합니다. 즉, 엔티티에 하위 유형이 있는 경우 항상 적어도 하나 이상의 또 다른 하위 유형이 있어야 합니다. 이것은 타당합니다. 엔티티와 단일 하위 유형을 구분하는 이유가 무엇입니까? 이러한 생각이 두 하위 유형 규칙을 낳았습니다.

하위 유형: 세 레벨



ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

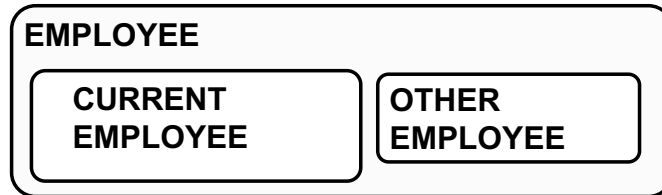
중첩된 하위 유형

하위 유형을 중첩할 수 있습니다. 읽기 쉽도록 하기 위해 일반적으로 둘 이상의 레벨로 하위 유형을 생성하지는 않지만 이렇게 하지 말아야 할 이유도 없습니다. 새 레벨을 생성한 후에 속성 및 관계의 배치를 다시 고려하십시오.

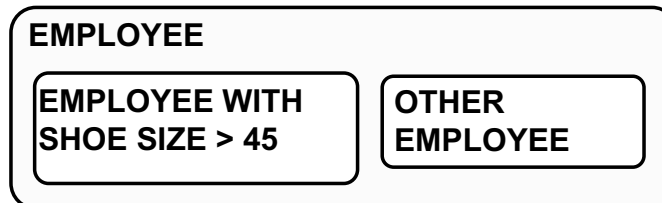
Oracle Internal & OAI Use Only

하위 유형에 대한 자세한 내용

하위 유형은 항상 존재합니다...



... 그러나 항상 타당한 것은 아닙니다.



ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

하위 유형은 항상 존재함

모든 엔티티는 항상 하위 유형이 있을 수 있습니다. 항상 인스턴스를 그룹으로 세분화하기 위한 규칙을 생성할 수 있지만 이것은 문제가 되지 않습니다. 하위 유형을 생성하는 이유는 동시에 유사성과 차이점을 나타내야 하는 경우가 항상 있기 때문입니다.

하위 유형 구현

다양한 방법으로 하위 유형 엔티티를 구현할 수 있습니다. 예를 들어, 상위 엔티티를 기반으로 별도의 테이블이나 단일 테이블로 구현할 수 있습니다.

요약

엔티티

- 텍스트의 명사
- 실체, 비실체, 이벤트

속성

- 엔티티의 단일 값을 갖는 수식자

하위 유형

- 상위 유형의 모든 속성 및 관계 상속
- 자체의 속성 및 관계를 가질 수 있음

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

요약

엔티티는 기능적으로 업무를 설명하는 텍스트에서 주로 명사로 나타날 수 있습니다. 엔티티는 실체, 비실체 및 이벤트일 수 있습니다. 엔티티의 하위 유형은 해당 엔티티의 모든 속성 및 관계를 공유하지만 추가 속성 및 관계를 가질 수 있습니다.

속성은 해당 속성이 속하는 엔티티를 설명, 정량 분석, 한정, 분류, 지정하거나 엔티티에 상태를 부여하는 정보의 기본 부분으로 단일 값을 갖습니다.

대부분의 엔티티는 속성을 갖습니다.

모든 속성은 속성이 처음에 속했던 엔티티와 관련된 별도의 엔티티로 승격될 수 있습니다. 이름을 여러 언어로 관리하거나 가격을 여러 통화로 관리해야 할 때와 같이 속성이 단일 값을 갖지 않을 때 이 작업을 수행해야 합니다.

연습

- 서적
- Moonlight Coffees
- 판매점
- 하위 유형
- 일정
- 주소

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

Oracle Internal & OAI Use Only



1. 방금 책 쓰는 일을 끝냈습니다. 이것은 정의와 권력에 대한 소설입니다.
2. 우리는 이 책을 방금 출판했습니다. 이제 양장본을 사용할 수 있습니다.
3. 피카소에 대한 새 서적을 읽어 보았습니까? 예. 아주 좋은 책이더군요!
4. 원한다면 제 책을 빌려 드리겠습니다.
5. 이제 막 이 책을 스페인어로 번역하는 작업을 시작했습니다. 16세기의 원본 스타일이 아니라 현대 영어 텍스트를 토대로 번역했습니다.
6. 부모님께 드리려고 그 책을 주문했습니다.

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

연습 2-1: 서적

목표

이 연습의 목적은 텍스트에 사용된 단어의 다양한 의미를 구분하는 것입니다.

과제

- 이 텍스트에서 "서적"이라는 단어는 여러 의미로 사용됩니다. 이러한 의미는 출판 회사나 도서 판매업체의 관점에서 볼 때 여러 다른 엔티티에 해당합니다. 모두 서적을 나타내는 다양한 엔티티를 구별해 보십시오. 이러한 엔티티에 대해 보다 적합한 이름을 부여하고 이러한 이름을 구분하기 위한 한 두 개의 속성을 구성하십시오.
- 텍스트에 준하여 ER 모델을 생성합니다. 페이지 맨 위에 가장 일반적인 엔티티를 표시하고 맨 아래에 가장 구체적인 엔티티를 표시하십시오. 중간에 다른 엔티티를 적절히 표시하십시오. 관계 이름에 대해서는 신경 쓸 필요가 없습니다.



7. 예. 그 책을 판매하고 있습니다. 그 책은 예술 분야 서적에서 찾아야 합니다.
8. "전쟁과 평화"는 2쇄 때 아주 적은 양이 출판되었습니다.
9. "My name is Asher Lev"가 지금까지 나온 책 중에서 가장 훌륭한 책 중 하나라고 생각합니다. 제 것은 자필 서명이 되어 있습니다.
10. 퇴직하면 엔티티 관계 모델링에 대한 책을 쓰고 싶습니다.

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

연습 2-1: 서적

목표

이 연습의 목적은 텍스트에 사용된 단어의 다양한 의미를 구분하는 것입니다.

과제

- 이 텍스트에서 "서적"이라는 단어는 여러 의미로 사용됩니다. 이러한 의미는 출판 회사나 도서 판매업체의 관점에서 볼 때 여러 다른 엔티티에 해당합니다. 모두 서적을 나타내는 다양한 엔티티를 구별해 보십시오. 이러한 엔티티에 대해 보다 적합한 이름을 부여하고 이러한 이름을 구분하기 위한 한 두 개의 속성을 구성하십시오.
- 텍스트에 준하여 ER 모델을 생성합니다. 페이지 맨 위에 가장 일반적인 엔티티를 표시하고 맨 아래에 가장 구체적인 엔티티를 표시하십시오. 중간에 다른 엔티티를 적절히 표시하십시오. 관계 이름에 대해서는 신경 쓸 필요가 없습니다.

요약

- Moonlight Coffees는 현재 세계 12개국에 500개 이상의 판매점을 보유하고 있으며 빠르게 성장하고 있는 고품질 커피 판매 체인입니다. 판매점은 주요 쇼핑 센터, 엔터테인먼트 및 회사 밀집 지역, 공항, 철도역, 박물관과 같은 최상급 위치에 있습니다. Moonlight Coffees는 약 9,000명의 직원이 있습니다.

제품

- 모든 판매점은 커피, 차, 청량음료 및 다양한 패스트리를 제공하고 있습니다. 대부분의 판매점에서는 업서 및 경우에 따라 극장표 같은 비식품을 판매합니다.

연습 2-2: Moonlight

시나리오

사용자는 Moonlight Coffees Inc.에서 계약직으로 일하고 있습니다. 업무 분석가인 동료 중 한 명이 몇 가지 문서를 준비했습니다. 다음은 요약 문서에서 발췌한 내용입니다.

과제

- Moonlight Coffees에 중요하다고 생각되는 엔티티 목록을 약 15가지 만드십시오. 상상력과 상식을 활용하여 아래 출력된 요약에서 얻은 정보를 사용하십시오.
- 다음을 나타내는 엔티티의 공식적인 정의를 작성하십시오.
 - 커피 판매점
 - Moonlight 직원

요약

재무

판매점 관리 부서에서는 본사에 매일 현지 통화로 판매 수치를 보고하고 있습니다. Moonlight에서는 매월 변경되는 내부 환율 목록을 사용합니다. 1999년 1월 1일 이래로 EU 국가들은 유로화로 보고해야 합니다.

주식

Moonlight Coffees는 주식회사로, 주식은 NASDAQ에서 거래되며 증권 시세 표시 기호는 MLTC입니다. 직원들은 스톡 옵션 계획에 참여할 수 있습니다.

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

연습 2-2: Moonlight

시나리오

사용자는 Moonlight Coffees Inc.에서 계약직으로 일하고 있습니다. 업무 분석가인 동료 중 한 명이 몇 가지 문서를 준비했습니다. 다음은 요약 문서에서 발췌한 내용입니다.

과제

- Moonlight Coffees에 중요하다고 생각되는 엔티티 목록을 약 15가지 만드십시오. 상상력과 상식을 활용하여 아래 출력된 요약에서 얻은 정보를 사용하십시오.
- 다음을 나타내는 엔티티의 공식적인 정의를 작성하십시오.
 - 커피 판매점
 - Moonlight 직원



판매점 목록

Shoplist, ordered to date opened page 4

**181 The Flight, JFK Airport terminal 2, New York, USA,
212.866.3410, Airport, 12-oct-97**

**182 Hara, Kita Shinagawa,Tokyo, JP, 3581.3603/4,
Museum, 25-oct-97**

**183 Phillis, 25 Phillis Rd, Atlanta, USA, 405.867.3345,
Shopping Centre, 1-nov-97**

**184 JFK, JFK Airport terminal 4, New York, USA,
212.866.3766, Airport, 1-nov-97**

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

연습 2-3: 판매점

시나리오

사용자는 Moonlight Coffees에서 계약직으로 일하고 있습니다. 수행해야 할 작업은 업무에 대한 개념적 데이터 모델을 생성하는 것입니다. Moonlight에 대한 모든 종류의 문서를 수집했습니다. 다음은 판매점 목록 페이지입니다.

과제

목록의 정보를 ER 모델의 기초로 사용하십시오. 매우 신중하게 속성을 모두 찾아 내십시오.

Moonlight Coffees



판매점 목록

**185 VanGogh, Museumplein 24, Amsterdam, NL,
76.87.345, Museum, 10-nov-97**

**186 The Queen, 60 Victoria Street, London, UK,
203.75.756, Railway Station, 25-nov-97**

**187 Wright Bros, JFK Airport terminal 1, New York,
USA, 212.866.9852, Airport, 6-jan-98**

**188 La Lune, 10 Mont Martre, Paris, FR, 445 145 20,
Entertainment, 2-feb-98**

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

연습 2-3: 판매점

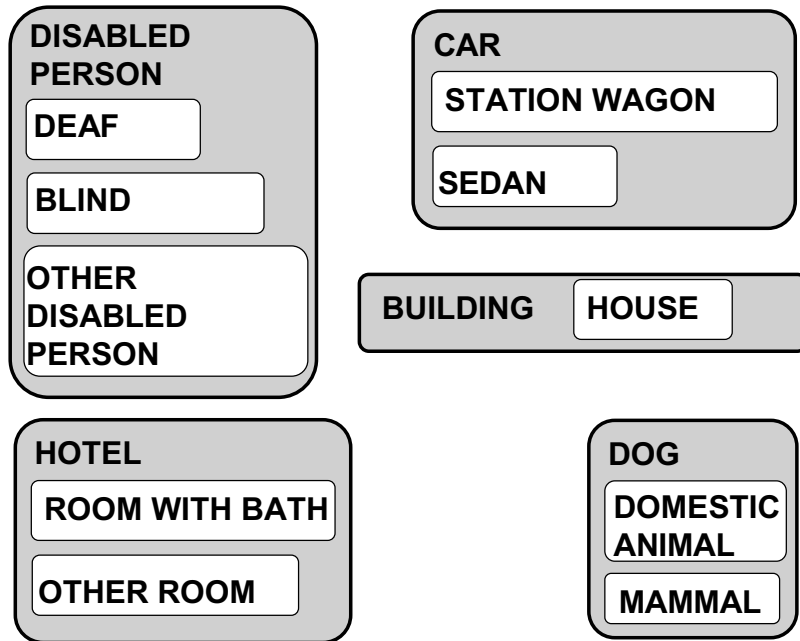
시나리오

사용자는 Moonlight Coffees에서 계약직으로 일하고 있습니다. 수행해야 할 작업은 업무에 대한 개념적 데이터 모델을 생성하는 것입니다. Moonlight에 대한 모든 종류의 문서를 수집했습니다. 다음은 판매점 목록 페이지입니다.

과제

목록의 정보를 ER 모델의 기초로 사용하십시오. 매우 신중하게 속성을 모두 찾아 내십시오.

하위 유형



ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

연습 2-4: 하위 유형

목표

이 연습의 목표는 올바른 하위 유형과 잘못된 하위 유형을 찾아내는 것입니다.

과제

그림에서 잘못된 하위 유형을 모두 찾습니다. 하위 유형이 잘못되었다고 생각하는 이유를 설명하십시오. 더 나은 형태로 모델을 조정하십시오.

van Gogh, Museumplein, Amsterdam

Schedule Oct 12 - Oct 18				prepared by Janet			
Shift	Mon	Tue	Wed	Thu	Fri	Sat	Sun
Annet S			2		2	2	1
Annet B	1				1	1	
Dennis	2	2	1	2	3		
J_rgen						5	4
Kiri		3			4	4	
Wil							

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

연습 2-5: 일정

시나리오

사용자는 Moonlight Coffees에서 계약직으로 일하고 있습니다.

과제

Amsterdam의 판매점 중 하나에서 사용되는 근무 일정을 엔티티 관계 모델에 대한 기반으로 사용하십시오. 예를 들어 일정에 따르면 10월 12-18일 주에 Annet B는 월요일, 금요일 및 토요일에 첫번째 교대 근무 일정이 잡혀 있습니다.

이 근무표를 보면 매일 한 사람이 한 번만 교대하는 것을 알 수 있습니다.

연습: 주소(1/2)

Rheingasse 123
53111 Bonn
Germany

34 Oxford Road
Reading
Berkshire RG1 8JS
UK

1020 Maple Drive
Kirkland WA 98234
USA

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

연습 2-6: 주소

목표

이 연습의 목표는 주소 모델링의 다양한 방법을 정렬하는 것입니다.

과제

엔티티 PERSON(또는 ADDRESS)은 아래 예제와 같이 주소를 설명하는 속성을 가질 수 있습니다.

1. 향후 시스템이 정확한 국제 메일 기능을 생성하도록 요구되는 경우 어떻게 주소 정보를 모델링할 수 있습니까?

연습: 주소(2/2)

**P.O. Box 66708
Nairobi
Kenya**

**c/o Mrs Smith
Maude Street
Sandton
Johannesburg 2144
South Africa**

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

연습 2-6: 주소

과제

1. 이전 연습에서 생성한 모델에서도 아래 주소를 사용할 수 있습니까?
2. 또한 시스템에 다음 범주의 주소를 검색하기 위한 기능이 요구되는 경우 모델이 달라지는지 확인합니다. 필요에 맞게 변경합니다.

모든 주소:

- Kirkland 내에 위치
- Bonn에서 우편 번호가 53111인 경우
- 사서함
- 다음 위치:
 - Oxford Road 또는
 - Oxford Rd 또는
 - OXFORD ROAD 또는
 - OXFORD RD
- 읽기

Oracle Internal & OAI Use Only

관계 세부 설명

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

Oracle Internal & OAI Use Only

개요

- 관계
- 10가지 관계 유형
- 비전이성(nontransferability)
- 속성을 가지는 것처럼 보이는 관계
- 정규화 규칙

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

소개

이 단원에서는 두 엔티티 간의 관계가 어떻게 설정되는가에 대해 자세히 설명합니다. 10개의 관계 유형과 이보다 덜 일반적인 유형의 예가 제시됩니다. 이 단원에서는 비전이적 관계를 살펴보고 관계와 속성 간의 차이와 유사성을 설명합니다. 또한 관계가 속성을 가지는 상황에 대한 해결책도 제시합니다. 끝으로 정규화 규칙이 개념 모델이라는 맥락에서 설명됩니다.

목표

이 단원을 마치면 다음을 수행할 수 있습니다.

- 명쾌하게 정의된 엔티티 간의 관계 생성
- 일반적인 관계 유형과 그렇지 않은 관계 유형의 식별
- 이례적인 관계 유형의 실제 예제 제시
- 특정 정보의 모델링을 위해 속성과 관계 중 무엇을 사용할지 선택
- m:m 관계를 하나의 교차 엔티티와 두 개의 관계로 분석
- 기타 관계 분석 및 분석 시기 파악
- 정규화 규칙

관계 설정

- **관계의 존재 여부 결정**
- **두 가지 관점에서 관계 이름 선택**
- **선택 가능성 결정**
- **정도 결정**
- **비전이성(nontransferability) 결정**

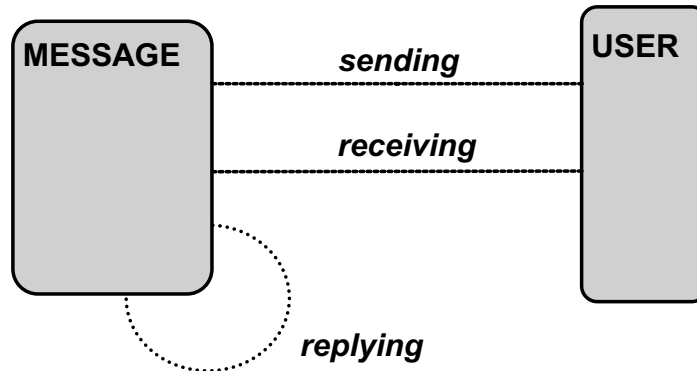
ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

관계의 존재 여부 결정

- 각 엔티티에 대해, 엔티티가 어떤 식으로든 모델 내 하나 이상의 엔티티와 관련이 있는지 질문하고, 관련이 있다면 "기본 구조" 관계를 점선으로 그립니다.
- 일반적으로 모델 내 모든 엔티티는 하나 이상의 다른 엔티티와 관련이 있습니다. 매우 드물지만 예외는 있습니다.
- 두 개의 엔티티는 두 번 이상 관련이 있을 수 있습니다. 예를 들어, Electronic Mail 시스템의 경우, MESSAGE 엔티티와 USER 엔티티 사이에는 두 개의 관계가 존재하는데, 하나는 MESSAGE를 보내는 사람에 대한 것이고 하나는 MESSAGE를 받는 사람에 대한 것입니다.
- 엔티티는 그 자체와도 관련이 있을 수 있습니다. 이를 순환 관계라고 합니다. 예를 들어, MESSAGE는 다른 MESSAGE에 대한 응답일 수 있습니다. 자세한 내용에 대해서는 순환 관계 관련 문단을 참조하십시오.

관계 설정

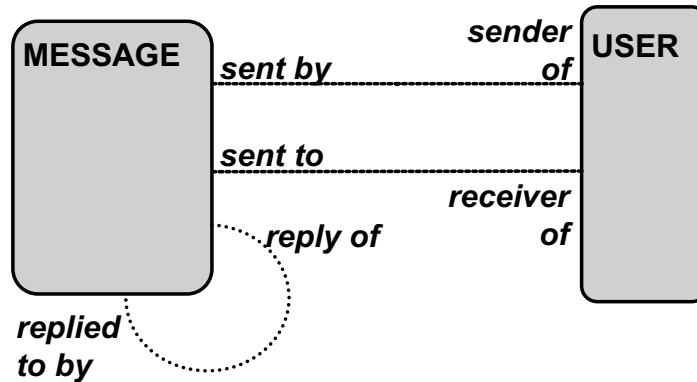


ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

Oracle Internal & OAI Use Only

관계 이름



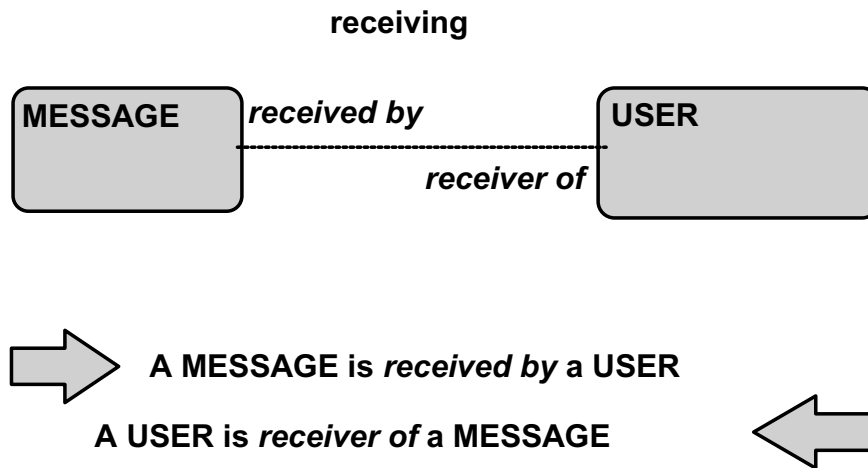
ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

관계 이름 선택

- 경우에 따라 두번째 관점에 대한 관계 이름이 *is owner of*와 *is owned by*와 같이 단순히 다른 관점의 수동형입니다. 어떤 경우에는 *parent of / child of* 또는 *composed of / part of*와 같이 두 개념이 대비되는 명칭이 존재합니다.
- 전치사로 끝나는 이름을 사용합니다.
- 이름을 찾을 수 없을 경우, 다음과 같은 관계 이름을 사용할 수 있습니다.
 - *Consists of / is part of*
 - *Is classified as / is classification for*
 - *Is assigned to / is assignment of*
 - *Is referred to / referring to*
 - *Responsible for / the responsibility of*
- 어떤 경우에는 *with, in, of, for, by, about, at, into*와 같이 매우 짧은 이름으로도 충분합니다.

관계 명명



ORACLE

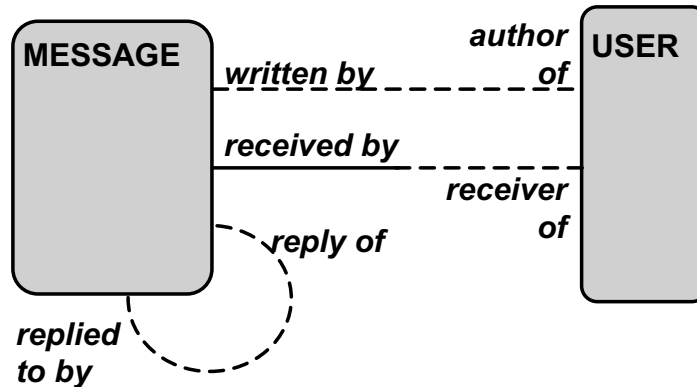
Copyright © Oracle Corporation, 2002. All rights reserved.

관계 명명

*sent to*와 *receiver of*는 정반대입니까? 그렇다면 MESSAGE가 USER에게 전송되는 경우 MESSAGE는 도착한다는 가정도 성립됩니다. 이 경우, 관계의 이름을 *received by* / *receiver of*...로 명명하는 것이 더 안전할 것입니다.

Oracle Internal & OAI Use Only

선택 가능성



ORACLE

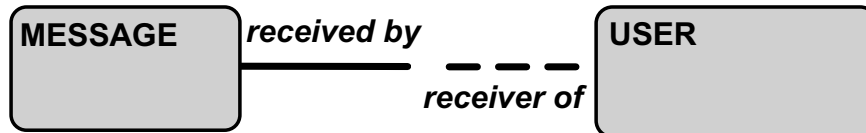
Copyright © Oracle Corporation, 2002. All rights reserved.

관계 양쪽 끝 모두의 선택 가능성 결정

- 다음 질문에 답하십시오.
 - 모든 MESSAGE는 USER에 의해 전송되어야 합니까?
 - 모든 USER는 MESSAGE의 송신자이어야 합니까?
 - 모든 MESSAGE는 USER에게 전송되어야 합니까?
 - 모든 USER는 MESSAGE에 수신자로 지정되어야 합니까?
- 대답이 "Yes"일 경우 관계 끝은 필수이고, 그렇지 않을 경우 선택입니다.
- 이 점에 주의하십시오. 종종 관계 끝이 필수인 것처럼 보이지만, 실제로는 그렇지 않습니다. ElectronicMail 예제에서, 모든 MESSAGE는 USER에 의해 전송되어야 하는 것처럼 보입니다. 그러나 외부 사용자에게 의해 내부 USER에게 전송된 MESSAGE는 시스템이 외부 사용자들도 유지하지 않는 한, USER와 관련이 없습니다.
- 어떤 경우에는 관계가 결국에는 필수이지만 처음에는 필수가 아닌 경우도 있습니다. 그러한 관계는 선택으로 모델링되어야 합니다.

선택 가능성

아니오: — — — — . 예: —————



- 모든 MESSAGE는 USER에 의해 수신되어야 합니까? **예**
- 모든 USER는 MESSAGE의 수신자이어야 합니까? **아니오**

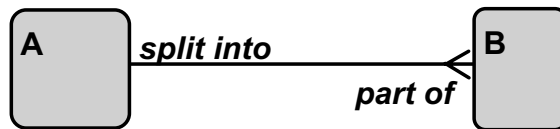
ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

관계 양쪽 끝의 정도 결정

- 다음 질문에 답하십시오.
 - MESSAGE는 둘 이상의 USER에 의해 작성될 수 있습니까?
 - USER는 둘 이상의 MESSAGE의 작성자일 수 있습니까?
- 대답이 No일 경우 정도는 "1"이라고 합니다.
- 대답이 Yes일 경우 정도는 "many" 또는 간단히 "m"이라고 합니다.
- 이것은 모든 관계 끝에 대해 결정되어야 합니다.
- A에서 B로의 필수 "many"(mandatory "many") 관계 끝은 반드시 A가 둘 이상으로 분할되어야 한다는 것을 의미하지 않습니다.

필수 1: 필수 m



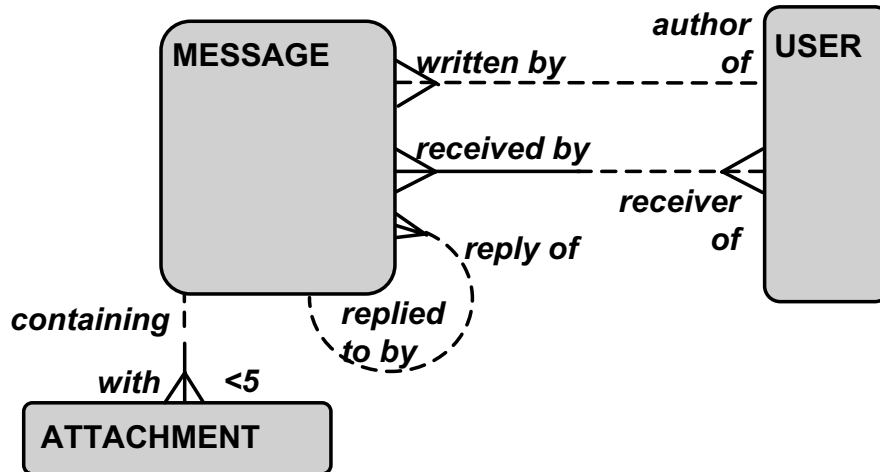
- 모든 A는 하나 이상의 B로 분할되어야 합니다.
- 모든 B는 정확히 하나의 A의 부분이어야 합니다.

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

Oracle Internal & OAI Use Only

정도



ORACLE

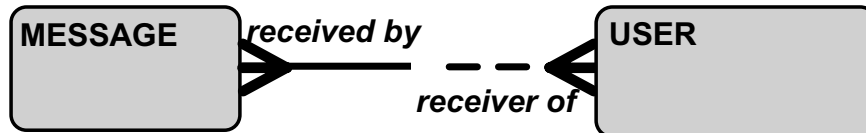
Copyright © Oracle Corporation, 2002. All rights reserved.

정도

- 선택적 "many" (optional many) 관계 끝은 0, 1 또는 그 이상을 의미합니다. 전자 메일 예제에서, 한 사람의 USER는 0개, 1개 또는 그 이상의 MESSAGE의 작성자일 수 있습니다.
- 정도는 고정 값이거나, 최대 수가 있습니다. 하나의 MESSAGE는 하나 이상의 ATTACHMENT를 포함할 수 있지만, 업무상의 이유로 인하여 MESSAGE 당 ATTACHMENT는 4개를 초과하지 않는다고 가정합니다. 이 경우 정도는 <5입니다. 그러나 그림에서는 크로우풋으로 표시되어 있습니다.

정도

하나: — 하나 이상: <



- 하나의 MESSAGE가 둘 이상의 USER에 의해 수신될 수 있습니까?
- 하나의 USER가 둘 이상의 MESSAGE의 수신자가 될 수 있습니까?

예

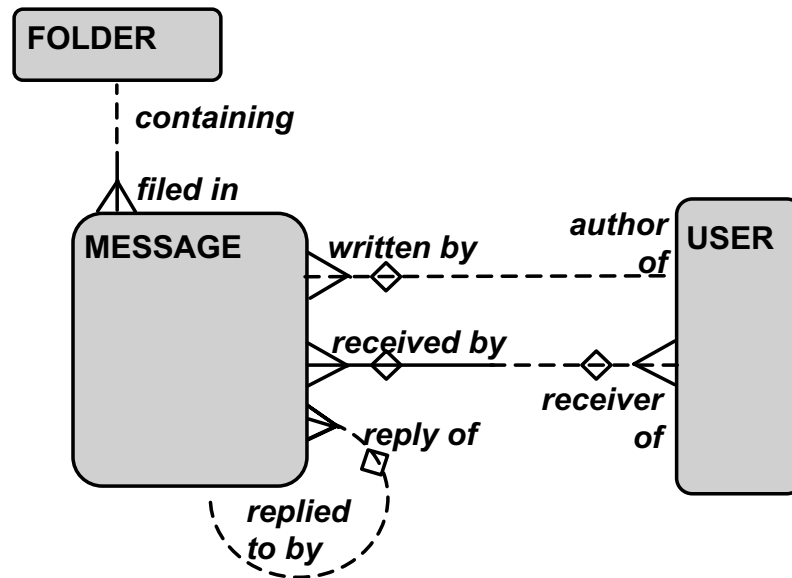
예

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

Oracle Internal & OAI Use Only

비전이성(nontransferability)



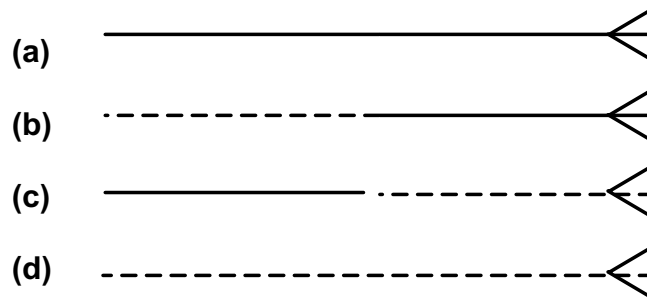
ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

관계 양쪽 끝의 비전이성(nontransferability) 결정

- MESSAGE가 생성될 때, MESSAGE의 작성자인 USER는 하나의 사실입니다. MESSAGE가 작성된 후 메일 시스템을 사용하여 작성자를 변경할 수 있다면 이상할 것입니다.
- 관계는 종종 한 번 연결이 이루어지면 연결을 변경할 수 없다는 속성을 가집니다. 그러한 속성을 비전이성(nontransferability)이라고 합니다. 비전이성(nontransferability)은 갱신 불가능한 외래 키로 이어집니다. 비전이성(nontransferability)은 그림에서 관계의 끝 사이를 잇는 선에 놓인 다이아몬드형 기호로 표시됩니다.
- 모든 관계가 비전이적 관계는 아닙니다. 메일 시스템에서 사용자가 FOLDER에 MESSAGE를 파일로 저장할 수 있다고 가정합니다. 이것은 사용자가 MESSAGE가 파일로 저장되는 FOLDER를 변경할 수 있도록 허용되는 경우에만 유용한 기능입니다.

관계 유형 1:m



ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

관계 유형

정도에 따라 명명된 다음과 같은 세 가지 기본 관계 그룹이 있습니다.

- One to many (1:m)
- Many to many (m:m)
- One to one (1:1)

이 문단에서는 다양한 유형에 대해 설명하고 그 변형 예제를 제시합니다.

관계—1:m

1:m 관계의 여러 유형은 ER 모델에서 가장 일반적입니다. 이미 몇 가지 예제가 제시되었습니다.

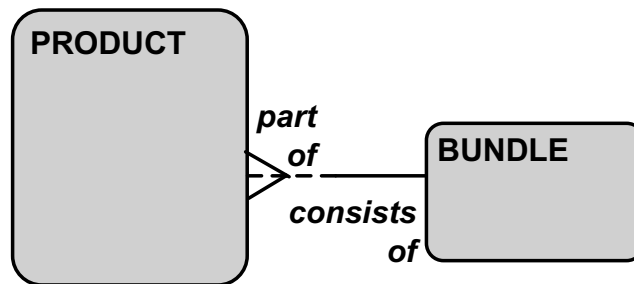
1. 양 끝에서 필수. 이 유형의 관계는 일반적으로 상대방이 없으면 존재할 수 없는 엔티티를 모델링합니다. 마스터에 대한 필수 디테일의 존재는 엄격한 업무 규칙이기보다는 희망 사항인 경우가 많습니다. 종종 이 관계는 하나의 엔티티가 항상 여러 개의 디테일로 분할되는 것을 나타냅니다. 다른 관점에서 보면, 이 관계는 항상 분류되고 지정되는 엔티티를 나타냅니다.

필수 1 대신 필수 m 사용

일반적으로 사람들은 주제에 대해 다른 관점을 취함으로써 (a) 관계 유형을 피하고 (b) 유형을 선호할 것입니다. 예를 들어, 주문이 적어도 하나의 주문 항목을 가진 어떤 것으로 정의된다고 가정합시다. 달리 말해서, 주문은 구성된 개념으로 간주됩니다. 사용자는 주문을 엔티티로서 모델링하지 않고, 대신 ORDER HEADER와 같은 약간 다른 개념으로 모델링할 수 있습니다. 그런 다음, ORDER HEADER가 0, 1 또는 그 이상의 ORDER ITEM을 가지도록 정의합니다. 그러면 주문은 두 개의 엔티티로 구성된 하나의 실체, 즉 하나 이상의 ORDER ITEM을 가진 ORDER HEADER가 될 것입니다. 빈 헤더는 주문으로 간주되지 않을 것입니다.

Oracle Internal & OAI Use Only

관계 유형 m:1



ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

관계 유형

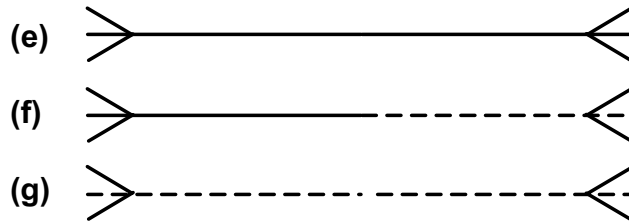
필수 1 대신 필수 m을 사용하는 이유

양쪽 끝 모두 필수인 1:m 관계를 구현하는 것은 기술적인 문제를 발생시킵니다.

특히 새로 생성된 레코드에 대해 디테일이 존재하도록 만들기가 어렵습니다. 대부분의 관계형 데이터베이스 환경에서는 더욱 불가능합니다.

2. 선택 1: 필수 m. 이것은 (d)와 더불어 가장 일반적인 관계 유형입니다. 일반적으로 전체 관계의 최소 90%는 (b) 유형과 (d) 유형입니다. 이 관계는 1 끝의 엔티티는 독립형일 수 있는 반면, many 끝의 엔티티는 상대 엔티티의 맥락에서만 존재할 수 있음을 나타냅니다.
3. 필수 1: 선택 m. 이 유형은 일반적인 유형이 아닙니다. 엔티티 인스턴스가 비어 있지 않은 세트일 경우 그리고 세트의 요소가 독립적으로 존재할 수 있는 경우에만 엔티티 인스턴스가 존재한다는 것을 나타내는 관계에 한해 이 유형을 볼 수 있습니다. 아래 예제에서 PRODUCT는 하나의 BUNDLE의 일부일 수 있습니다. 이 모델에 따르면, BUNDLE은 비어 있는지 여부는 중요하지 않습니다.
4. 양 끝에서 선택. (b) 유형에 대한 설명을 참조하십시오.

관계 유형 m:m



ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

관계 유형

관계—m:m

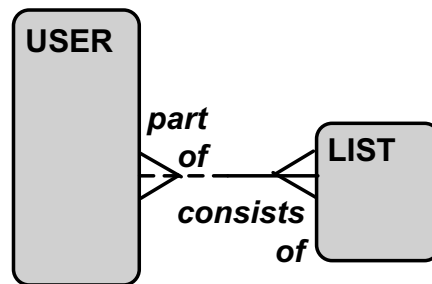
다양한 유형의 m:m 관계는 ER 모델의 첫 버전에서 일반적으로 나타납니다. ER 모델의 나중 단계에서는 대부분의 m:m 관계, 어쩌면 모든 m:m 관계가 사라집니다.

5. 양쪽에서 필수인 관계는 정상적인 환경에서는 흔하지 않습니다. 이 관계는 엔티티 인스턴스가 즉시 다른 엔티티의 인스턴스에 지정될 뿐 아니라 역으로도 지정될 경우에만 생성될 수 있다는 것을 의미하는 것처럼 보입니다. 그러나 어느 쪽 엔티티의 인스턴스도 없을 경우 어떻게 이런 일이 발생할 수 있습니까? 처음부터 필수 규칙을 시행하면 충돌이 일어납니다.

그러나 이 관계는 "하나의 선은 언제나 수많은 점으로 구성되고 하나의 점은 언제나 수많은 선의 일부이다"라는 수학적 특성과 같은 이론적 특성을 가지는 모델의 일부일 수 있습니다. 이것은 또한 "하나의 부서는 항상 여러 직원들을 가지고 있고 한 사람의 직원은 항상 한 부서에 할당된다"라는 기존의 상황을 설명할 수 있습니다. 여기에서, 상황이 항상 이렇게 전개된다는 보장이 있는가 하는 의문이 제기될 수 있습니다.

양쪽에서 필수인 m:m 관계는 관계가 호의 일부일 경우에 발생할 수 있습니다. 자세한 내용은 제약 조건 관련 단원을 참조하십시오.

관계 유형 m:m



ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

관계 유형

6. 일반적으로 나중 단계에서 사라지기는 하지만 모델의 초기 버전에서는 한 끝에서 필수인 관계가 드물지 않습니다.
7. 양 끝에서 선택인 관계는 선택은 모델의 초기 버전에서 일반적입니다. 이것도 일반적으로 나중 단계에서 사라집니다.

관계 유형 1:1

- (h) _____
- (i) _____
- (j) _____

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

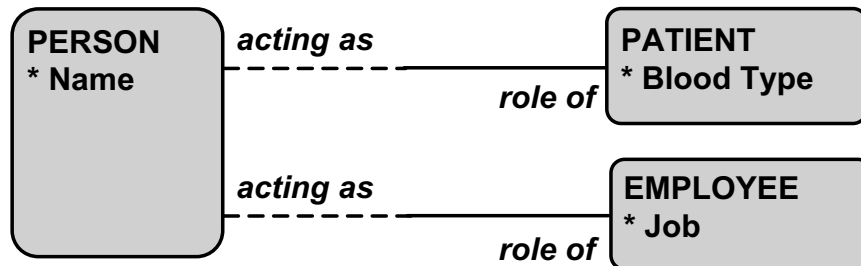
관계 유형

관계—1:1

일반적으로 모든 ER 모델에서 다양한 유형의 1:1 관계는 단지 몇 가지뿐입니다.

8. 양 끝에서 필수인 1:1 관계는 두 개의 엔티티를 긴밀히 연결합니다. 즉 PERSON 엔티티와 BIRTH 엔티티와 같이 엔티티 하나에 대해 인스턴스 하나를 생성하면 동시에 다른 엔티티에 대해서도 정확히 하나의 전용 인스턴스가 있어야 합니다. 그렇게 되면 굳이 두 개의 엔티티를 구분하고자 하는 이유가 무엇인지에 대한 의문이 제기됩니다. 수궁할 만한 유일한 대답은 기능상 필요성이 있을 경우뿐입니다. 모델이 이 관계를 가질 경우, 그것은 종종, 어찌면 항상, 호의 일부입니다.
9. 한 끝에서 필수인 관계는 롤이 모델링되어 있는 모델에서 나타납니다. 이 병원 모델이 그러한 예입니다.

1:1 관계 롤



ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

관계 롤

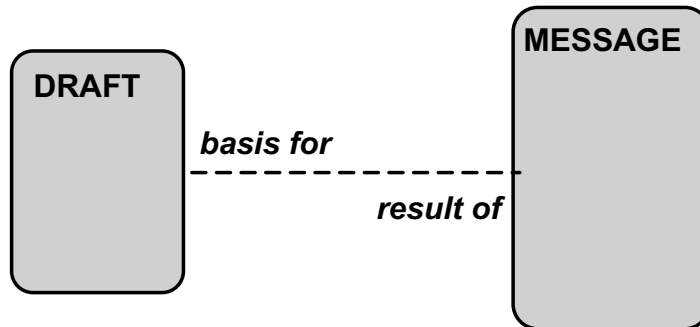
주: 이러한 롤 기준 관계는 종종 is/is type of 또는 단순히 is/is로 명명됩니다.

PATIENT와 EMPLOYEE는 모두 PERSON이 담당하는 롤입니다. 이 모델에 따르면, BLOOD TYPE 속성은 이 사람이 PATIENT일 경우에만 중요합니다. PERSON은 두 가지 롤을 모두 담당할 수도 있기 때문에 PATIENT와 EMPLOYEE는 PERSON의 하위 유형으로 모델링될 수 없습니다. 롤 개념에 대해서는 나중에 다시 설명합니다.

10. 양 끝에서 선택인 관계는 혼하지 않습니다. 그러나, 개념적으로는 동일하지만 서로 다른 시스템에 존재하는 두 개의 엔티티 사이에 관계가 존재할 때 이러한 유형이 발생할 수 있습니다. 이것의 한 예는 한 시스템에 EMPLOYEE 엔티티가 있고 또 다른 시스템, 예를 들어 타사 시스템에 PERSON 엔티티가 있는 경우입니다.

많은 1:1 관계(세 가지 변형 중에서)는 다음 예제에서처럼 일부 엔티티가 하나의 프로세스 내의 다양한 단계를 나타낼 때 발생합니다. 이 경우 관계 이름은 항상 leads to / result of 또는 그와 유사한 것일 수 있습니다.

1:1 관계 프로세스



ORACLE

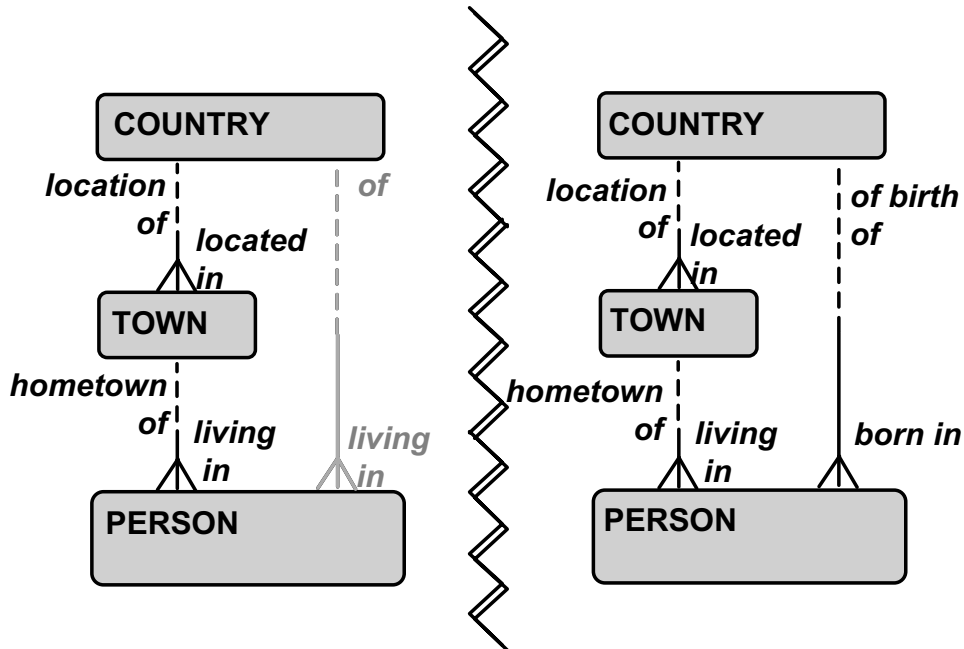
Copyright © Oracle Corporation, 2002. All rights reserved.

관계 롤

사람도 하나의 프로세스로 간주할 경우, 앞에서 제시한 BIRTH와 PERSON의 예는 이러한 일반적 개념에 그대로 적용됩니다.

Oracle Internal & OAI Use Only

중복 관계



ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

중복

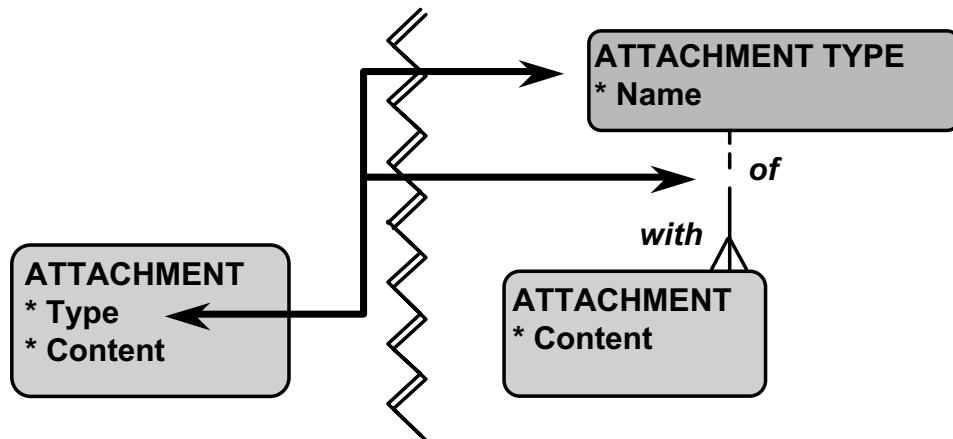
속성과 마찬가지로 관계는 중복될 수 있습니다.

예제 왼쪽 부분에서, 다른 두 개의 관계로부터 PERSON에서 COUNTRY로의 관계를 파생시킬 수 있으며 이것들은 모델에서 제거해야 합니다.

예제 오른쪽 부분에서 알 수 있듯이, 이것은 의미상의 문제이며 구조만으로는 결론지을 수 없습니다.

관계 및 속성

- 속성은 관계를 숨길 수 있습니다.
- 관계는 속성으로 "다운그레이드"될 수 있습니다.



ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

관계 및 속성

여러 속성은 하나의 관계를 숨길 수 있습니다. 사실, 모든 속성은 하나의 관계를 숨길 수 있습니다.

예제에서, ATTACHMENT 엔티티의 TYPE 속성은 ATTACHMENT TYPE 엔티티와 ATTACHMENT에서 ATTACHMENT TYPE으로의 관계로 대체될 수 있습니다.

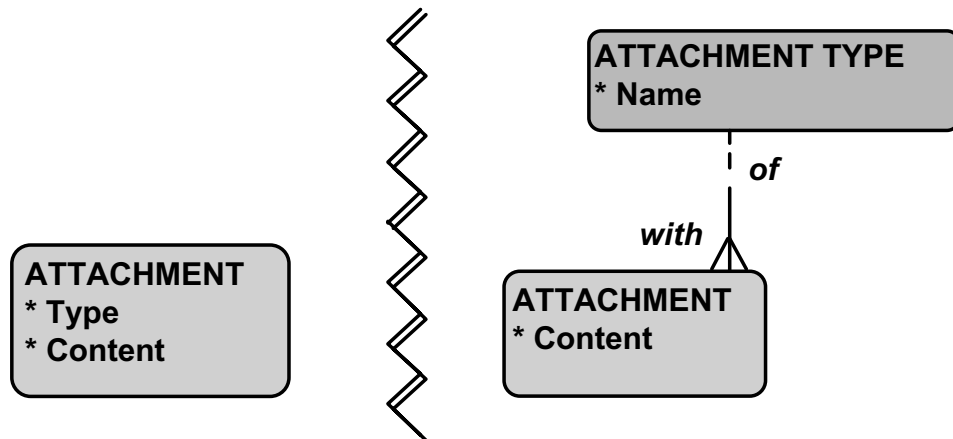
ATTACHMENT TYPE에 대한 추가 속성을 유지해야 할 필요가 발생하는 즉시 이런 식으로 모델링하는 외에 다른 대안은 없을 것입니다. ATTACHMENT TYPE이 유형의 Name 이외에 유지해야 할 중요한 속성이 없다면, 해당 엔티티의 제거를 고려하고 ATTACHMENT의 속성으로 Type을 취할 수 있을 것입니다.

또한 객실 유형이 싱글, 더블, 스위트, 세 가지뿐인 호텔 체인의 경우에서와 같이, 유형의 수가 고정된 소량일 경우 왼쪽 옵션의 사용을 고려할 수도 있을 것입니다.

관계에 속성 비교

- 쉬운 모델
- 테이블 감소
- 조인 없음

- 값 제어
- 값 목록
- 기타 관계



ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

관계에 속성 비교

ATTACHMENT 엔티티에 기준한 테이블은 두 상황 모두에서 동일한 열을 포함하지만, Attachment Type Name 열은 두번째 구현에서 외래 키 열이 됩니다. 이것은 ATTACHMENT에 대해 입력된 Attachment Type Name이 ATTACHMENT TYPE 엔티티에 기준한 테이블에 나열되는 유형에서만 취해질 수 있다는 것을 의미합니다. 목록은 선택 목록과 맞춤법 검사로 활용됩니다.

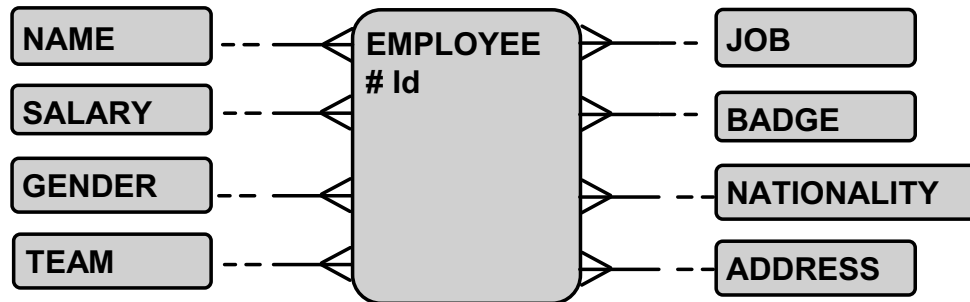
두 가지 모델에는 장단점이 있습니다.

한 개의 엔티티로 구성된 모델은 행이 적기 때문에 가독성이 좋습니다. 테이블 구현 시 필요한 정보를 얻기 위한 조인이 필요하지 않을 것입니다.

그러나 두 개의 엔티티로 구성된 모델은 대체로 유연성이 훨씬 더 뛰어납니다. 이것은 다른 엔티티에서 새 엔티티로의 관계를 생성할 수 있는 여지를 남깁니다. 주어진 세트를 기준으로 입력되는 값을 검사하므로 사용자가 제어력을 가지게 됩니다. 일반적으로, 두 개의 테이블로 구현하는 방식이 데이터베이스의 공간을 덜(때때로 훨씬 덜) 차지합니다.

속성과 엔티티는 일반적인 상식에 따라 선택하십시오.

속성 또는 엔티티



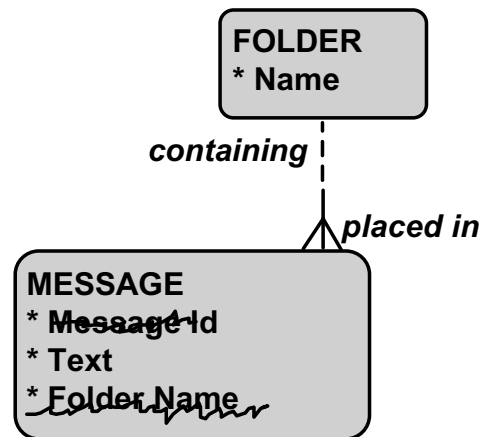
ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

Oracle Internal & OAI Use Only

관계에 속성 비교

- 외래 키 속성 같은 것은 없습니다.
- 일반적으로 속성 이름에 엔티티 이름이 포함되면 안됩니다.



ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

관계에 속성 비교

외래 키 속성이 존재하지 않음

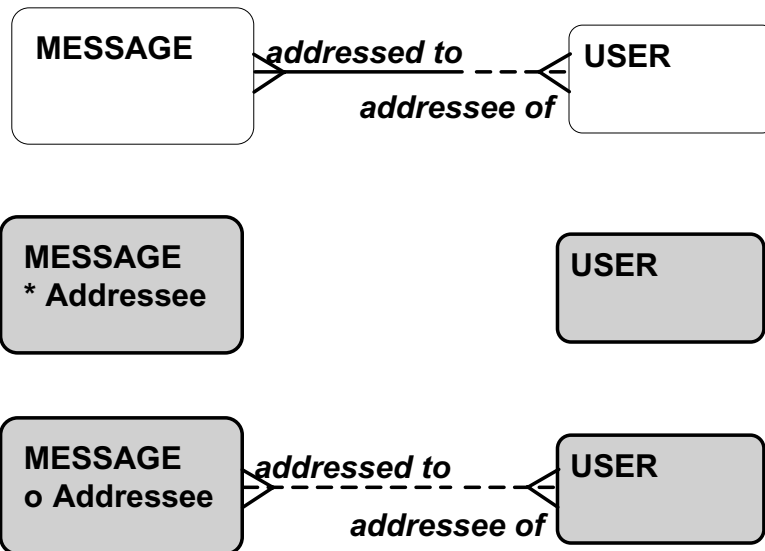
예제에서 MESSAGE 엔티티의 Folder Name 속성과 같은 외래 키 속성이 있음에 주의하십시오. ER 모델링에서는 외래 키 속성과 같은 것이 없습니다. 미래의 외래 키는 MESSAGE와 FOLDER의 관계로 나타납니다. 외래 키 열(들)은 FOLDER 엔티티의 고유한 기본 식별자에서 유래됩니다. 고유 식별자에 대한 자세한 내용은 제약 조건에 관한 단원을 참조하십시오.

속성 이름에 엔티티 이름 없음

속성 이름에 엔티티 이름이 포함되는 경우, 이것은 일반적으로 다음과 같은 상황에서 비롯됩니다.

- 속성이 위의 예제에서처럼 엔티티에 대한 관계를 숨기는 경우입니다. 두번째 엔티티는 아마도 나중 단계에서 추가되었을 것입니다.
- 속성이 엔티티를 숨기는 경우입니다. 전형적인 예로는 EMPLOYEE 엔티티의 Employment Date 속성을 들 수 있습니다. 직원은 동일한 회사에 의해 한 번만 채용될 수 있다는 규칙은 없을 것이기 때문에, Employment Date 속성이 EMPLOYMENT 엔티티를 숨길 수도 있습니다.
- 속성 이름 내의 엔티티 이름이 중복되는 경우입니다. 전형적인 예로는 MESSAGE 엔티티의 Message Id 속성을 들 수 있습니다. “Id”라는 이름으로 충분할 것입니다.
- 속성은 모델링되지 않은 1 대1 관계의 결과입니다. 예를 들어, EMPLOYEE 엔티티의 Birth Date 속성과 Birthplace 속성입니다. 이것들은 사실 모델링되지 않은(그리고 어쩌면 앞으로도 모델링되지 않을) BIRTH 엔티티의 속성들입니다.

속성에 관계 비교



ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

속성에 관계 비교

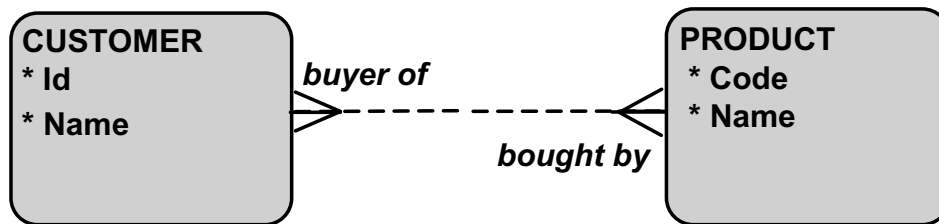
가끔 일부 정보는 엔티티들 간의 관계처럼 보이지만 실제로는 관계가 아닙니다.

ElectronicMail의 Compose Message 화면에는 사용자가 받는 사람들의 이름을 입력하도록 되어 있는 “To” 필드가 있습니다. 처음에는 이것을 MESSAGE와 USER 사이의 *addressed to / addressee of* 관계로 모델링하고 싶어할지 모르지만, 이러한 접근 방식은 문제가 있습니다. 메시지가 외부 사용자에게 전송될 경우, ElectronicMail이 단순히 관계 유지를 목적으로 과거에 메시지가 전송되곤 했던 모든 외부 사용자 주소를 추적하는 것이 타당할까요? 이것이 가능할까요?

이 경우에는 Addressee(받는 사람)를 MESSAGE 엔티티의 속성으로 보는 것이 더 나은 선택일 것입니다. 이 속성은 USER라고도 알려진 값을 포함할 수 있습니다. 달리 말하면, USER 엔티티에는 받는 사람에 대한 제안만 포함됩니다.

또 다른 가능성은 두 가지를 모두 수행하는 것입니다. 즉 받는 사람을 공동으로 처리하는 하나의 선택 관계와 하나의 선택 속성을 모델링하는 것입니다. 그러면 추가 제약 조건(그림에는 표시할 수 없음)은 최소한 하나의 속성과 관계에 실제로 MESSAGE에 대한 값이 부여되도록 해야 합니다.

m:m 관계로 특정 사항을 숨길 수 있음



ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

m:m 관계로 특정 사항을 숨길 수 있음

모델링 과정 중에는 많은 관계가 m:m 유형이 됩니다. 하지만 이런 현상은 대개 일시적일 뿐입니다. 모델에 더 많은 세부 정보를 추가할 수 있게 되면 많은 m:m 관계는 사라집니다. 이것은 심사 숙고한 결과 그러한 관계가 업무를 제대로 모델링하지 않기 때문입니다.

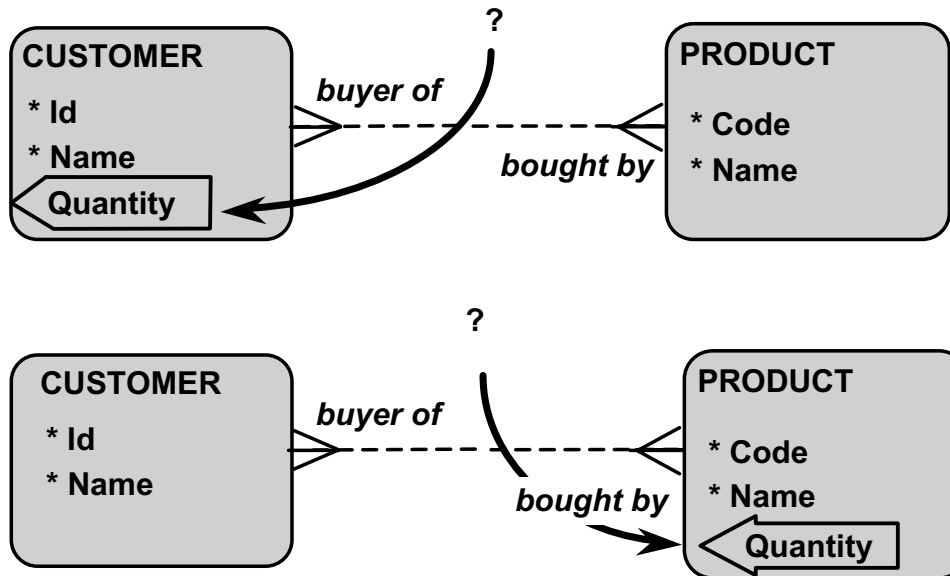
이에 대한 전형적인 예로는 CUSTOMER/PRODUCT 관계를 들 수 있습니다.

PRODUCT를 판매하는 어느 소매 회사에 대한 모델을 만든다고 합시다. CUSTOMER는 PRODUCTS를 구매합니다. 미래의 고객들도 시스템에 수용된다고 가정합니다. 이것은 다음을 의미합니다.

- CUSTOMER는 하나 이상의 PRODUCT를 구매할 가능성이 있습니다
- PRODUCT는 한 사람 이상의 CUSTOMER에 의해 구매될 가능성이 있습니다

이 회사의 일반적인 이벤트로서 Nick Sanchez라는 고객이 두 장의 셔츠를 구매한다고 가정합니다. "Nick Sanchez"는 CUSTOMER Name이고, "셔츠"는 PRODUCT Name입니다. 이때 수량 정보인 "두 장"은 어디에 놓아야 하는가 하는 문제가 남습니다.

Quantity는 ...의 속성



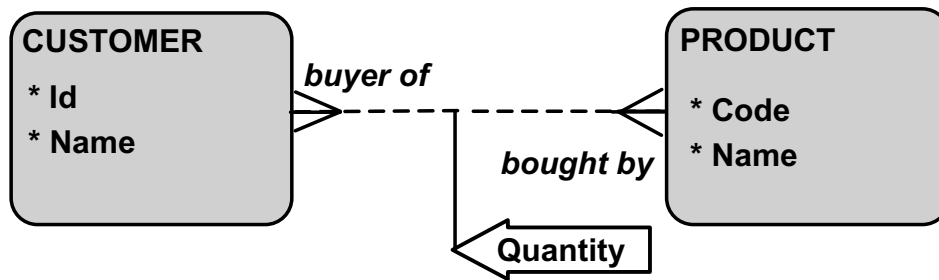
ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

Quantity

Quantity는 CUSTOMER나 PRODUCT의 속성이 아닌 것이 분명합니다. Quantity는 CUSTOMER와 PRODUCT 관계의 속성처럼 보입니다.

관계의 속성



ORACLE

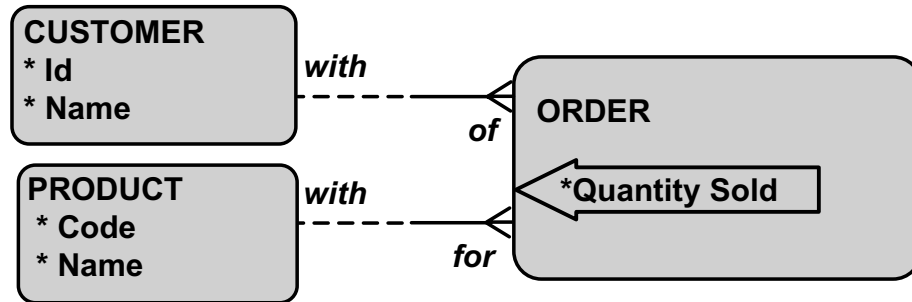
Copyright © Oracle Corporation, 2002. All rights reserved.

관계의 속성

관계는 속성을 가지지 않으며 가질 수도 없습니다. 분명히 수량이 속성인 엔티티가 빠져 있습니다. 따라서 모델을 변경해야 합니다. 여기에서 ORDER(또는 SALE이나 PURCHASE) 엔티티가 등장합니다.

Oracle Internal & OAI Use Only

새로운 ORDER 엔티티



CUSTOMERS		PRODUCTS		ORDERS		
Id	Name	Code	Name	Ctr_id	Pdt_code	Quantity_sold
1	Sanchez	1	Jeans	1	2	2
2	Lowitch	2	Shirt	1	3	2
3	Yomita	3	Tie	2	2	1
4				3		

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

새로운 Order 엔티티

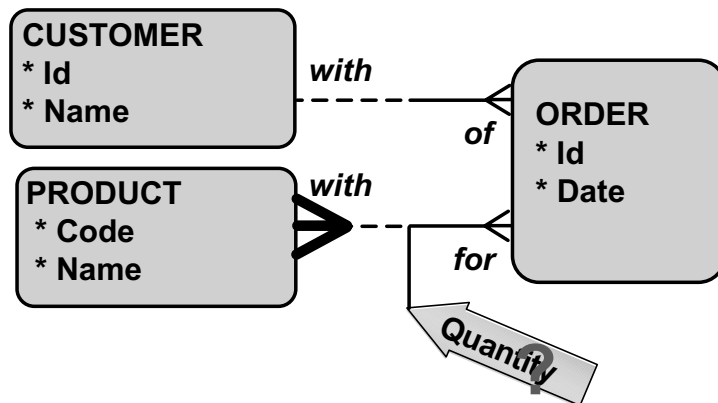
여기서의 테이블 설계는 모델 구현을 위한 기본 설계입니다. ORDERS 테이블에는 Ctr_id(CUSTOMER에 대한 외래 키)와 Pdt_code(PRODUCT에 대한 외래 키)의 두 개의 외래 키 열이 있습니다.

이제 Pepe Yomita가 상점에 들어와 바지 한 벌과 두 장의 셔츠와 실크 넥타이 하나를 구매한다고 합시다. 현재의 모델을 고려하면 이것은 Pepe가 바지, 셔츠 및 넥타이에 대해 각기 하나씩, 세 개의 주문을 한다는 것을 의미합니다. 세 개의 주문은 모두 동시에 한 사람의 동일한 고객으로부터 이루어집니다. 모델이 이것을 허용하기 때문에 지금까지는 별 문제가 없습니다.

이제 상점에서 주문에 대한 청구를 자동화하고자 한다고 합시다. (이것이 아마도 모델을 만드는 이유 중 하나일 것입니다.) 위 모델을 사용한다면, 시스템에서 이 세 개의 주문이 서로에게 속한다는 사실을 알 방법이 없기 때문에 이것은 세 가지 주문을 의미하고, 결국 세 개의 청구서를 의미합니다.

한 가지 주문이 둘 이상의 제품을 위한 주문이 될 수 있는 방식으로 모델을 변경하는 것이 더 나은 방법입니다. 그것은 곧 ORDER와 PRODUCT 사이에 m:m 관계를 설정해야 한다는 것을 의미하며, 이것에 대해서는 다음에 설명합니다.

한 주문에 대한 다수의 제품



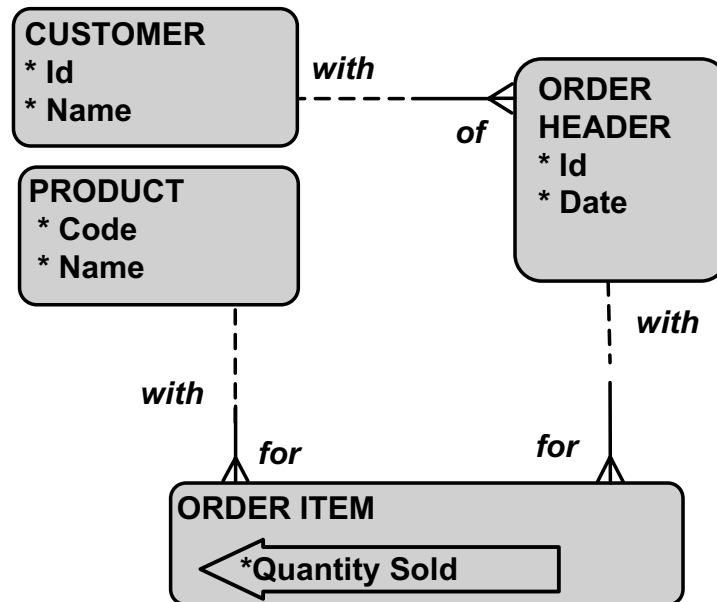
ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

한 주문에 대한 다수의 제품

그렇게 되면 다시 다음과 같은 문제가 제기됩니다. 수량은 어디에 두어야 하는가? 수량 속성은 단일 값이어야 하고 세 개의 값, 즉 1, 2 및 1을 동시에 포함할 수 없기 때문에 Quantity는 이제 더 이상 주문의 속성이 될 수 없습니다. Quantity는 이제 PRODUCT와 ORDER의 m:m 관계의 속성이 되었습니다.

또 다른 새 엔티티: ORDER ITEM



ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

또 다른 새 엔티티: ORDER ITEM

양 끝에서 필수인 1:m 관계를 피하기 위해 이름을 ORDER에서 ORDER_HEADER로 변경한 것에 주의하십시오. 이 모델에 대한 테이블 세트는 다음과 같습니다.

테이블

CUSTOMERS

Id	Name
1	Sanchez
2	Lowitch
	Yomita

ORDER_HEADERS

Id	Ctr_id	Date_ordered
1	1	25-MAY-1999
		25-MAY-1999
2	2	25-MAY-1999

ORDER_ITEMS

Ohd_id	Pdt_code	Quantity_sold
1	2	2
		2
2	2	1

PRODUCTS

Code	Name
1	Jeans
2	Shirt
3	Tie

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

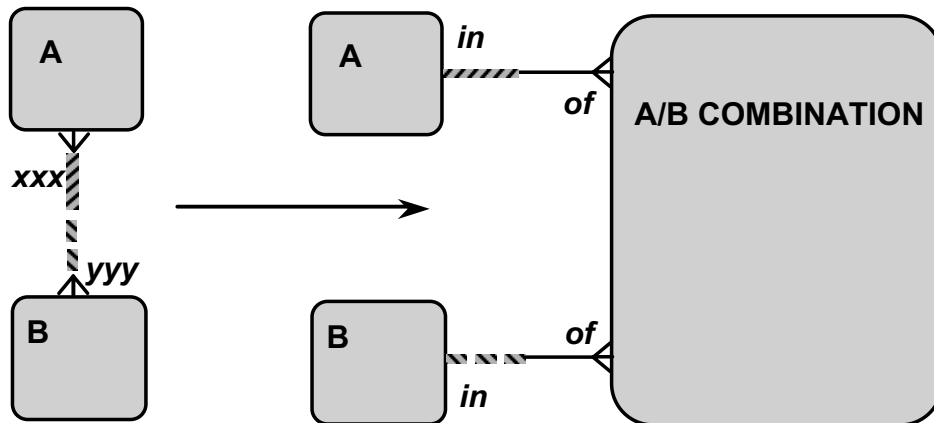
테이블

양 끝에서 필수인 1:m 관계를 피하기 위해 이름을 ORDER에서 ORDER_HEADER로 변경한 것에 주의하십시오. 이 모델에 대한 테이블 세트는 다음과 같습니다.

Oracle Internal & OAI Use Only

m:m 관계 분석

- 새로운 교차 엔티티 생성
- 두 개의 m:1 관계를 생성하고 선택 가능성 유도
- m:m 관계 제거



ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

관계 분석

관계 및 교차 엔티티

이 단원의 앞에서 속성을 가지는 것처럼 보이는 관계의 전형적인 예를 보았습니다. 예제에 제시된 관계는 many-to-many 관계였습니다. 지금 상황에서는 관계를 대체하고 속성을 보유할 수 있는 새로운 엔티티, 즉 교차 엔티티를 생성함으로써 문제를 해결합니다.

여기서는 다음과 같은 문제가 있을 수 있습니다.

- 일반적으로 관계를 분석하는 단계는 무엇입니까?
- 모든 m:m 관계는 분석되어야 합니까?
- m:m 이외의 다른 관계는 분석될 수 있습니까?

관계 분석

A 엔티티와 B 엔티티 사이의 m:m 관계를 분석한다고 합시다.

1. 먼저 새 교차 엔티티를 생성합니다. 이따금 모델링 중인 개념에 사용할 적합한 단어를 찾지 못하는 경우가 있을 것입니다. 새 엔티티는 항상 "A/B COMBINATION"처럼 새로운 단어를 만들거나 원래의 m:m 관계 이름에서 도출된 이름으로 지정할 수 있습니다. 엔티티에 적합한 이름을 사용할 수 없다고 해서 엔티티 모델링을 멈추지는 마십시오.

2. 그 다음 A/B COMBINATION 엔티티로부터 하나는 A에 대한 관계이고 하나는 B에 대한 관계인 두 개의 새로운 m:1 관계를 생성합니다. 처음에는 완벽한 A/B 쌍에만 관심이 있을 것이므로 A/B COMBINATION에서 이것들을 필수로 그립니다. 원래의 m:m 관계가 A 측에서 선택(또는 필수)이었다면, A에서 A/B COMBINATION으로의 새로운 관계도 선택(또는 필수)입니다.
3. 관계에 이름을 지정합니다. 두 개의 관계의 이름을 "in / of"로 지정할 수 있습니다.
4. 다음 단계는 시작할 때 생성한 m:m 관계를 제거하는 것입니다.
5. 마지막으로 새로 그려진 관계를 검토합니다. 이들 관계는 A/ B COMBINATION 쪽에서 볼 때 선택될 가능성이 있습니다. 또한 이들 관계가 m:m 유형으로 판명되고, 앞의 제품을 구매하는 고객의 예에서 보았던 것처럼 분석이 필요할 수도 있습니다.

모든 m:m 관계가 분석되어야 하는가?

대답은 여러 요인에 따라 달라집니다.

보통의 시나리오를 감안하면, ER 모델 생성을 시작할 때 많은 관계가 m:m 유형이라는 것을 알 수 있습니다. 이러한 관계의 대부분은 특정 속성을 위치시킬 장소가 있어야 하기 때문에 나중 단계에서 필요한 엔티티를 숨기는 것처럼 보입니다. 마지막에는 소수의 "순수 m:m" 관계만 남게 됩니다.

분석 안함

순전히 개념적인 데이터 모델링 관점에서 볼 때, 이러한 순수 m:m 관계를 분석할 필요는 없습니다. 이 모델은 테이블 설계의 기준이 되기에 충분히 다채롭습니다. m:m 관계는 이진 테이블로 변환되며, 이렇게 변환된 이진 테이블은 두 개의 외래 키의 열로만 구성된 테이블입니다. 이것은 m:m 관계를 분석했을 때 교차 엔티티에서 산출된 것과 동일한 테이블입니다.

개념적인 데이터 도표에서 m:m 관계는 별도의 엔티티 하나와 두 개의 관계를 더한 것보다 적은 공간을 필요로 합니다. 이러한 이유 때문에, 분석되지 않은 m:m 관계를 가진 도표가 더 명백하고 읽기도 더 쉽습니다.

분석함

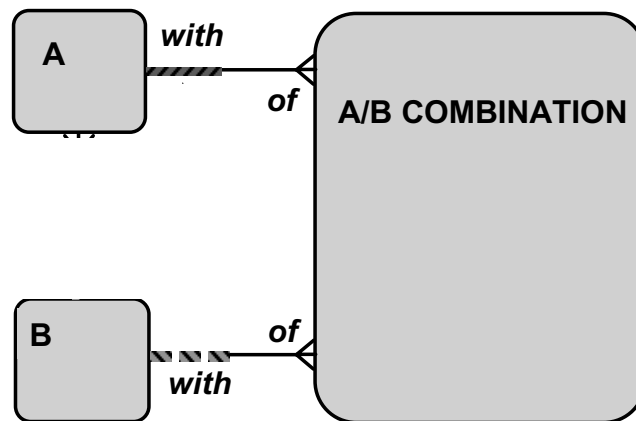
기능 모델링 관점에서 보면, 대답은 달라집니다. 모델에 순수 m:m 관계가 포함될 경우, 예를 들어 A 엔티티와 B 엔티티의 조합에 관한 정보를 보유해야 하는 업무상의 필요성이 분명히 존재합니다. 달리 말하면, 시스템에는 관계를 생성하는 최소한 하나의 업무 기능이 포함됩니다. 일반적으로 설계 도구들이 기능 모델에 대해 관계 생성을 필요로 하지만, 이러한 "관계 생성"은 엔티티의 속성 사용이라고 말할 수 없습니다. Oracle Designer도 예외는 아닙니다. 즉, Oracle Designer에서 ER 모델을 생성할 경우에는 모든 데이터 사용이 포함된 완전하게 정의된 기능 모델을 생성하기 위해 항상 m:m 관계를 분석해야 합니다.

기타 관계 분석

m:m 이외의 관계가 분석될 수 있습니까? 예. 모든 관계는 심지어 1:1 관계도 m:m 관계와 마찬가지로 하나의 교차 엔티티와 두 개의 관계로 분석될 수 있습니다. 그렇다면 언제 이러한 분석을 해야 할까요? 이러한 분석을 해야 할 상황은 쉽게 발견되지 않습니다. m:m 외의 관계를 분석해야 하는 일반적인 상황은 엔티티가 타사 패키지의 일부인 경우처럼 하나의 엔티티가 시스템 외부의 어떤 것을 나타내는 경우입니다.

m:m 관계 분석

- 새로운 교차 엔티티 생성
- 두 개의 m:1 관계를 생성하고 선택 가능성 유도
- m:m 관계 제거

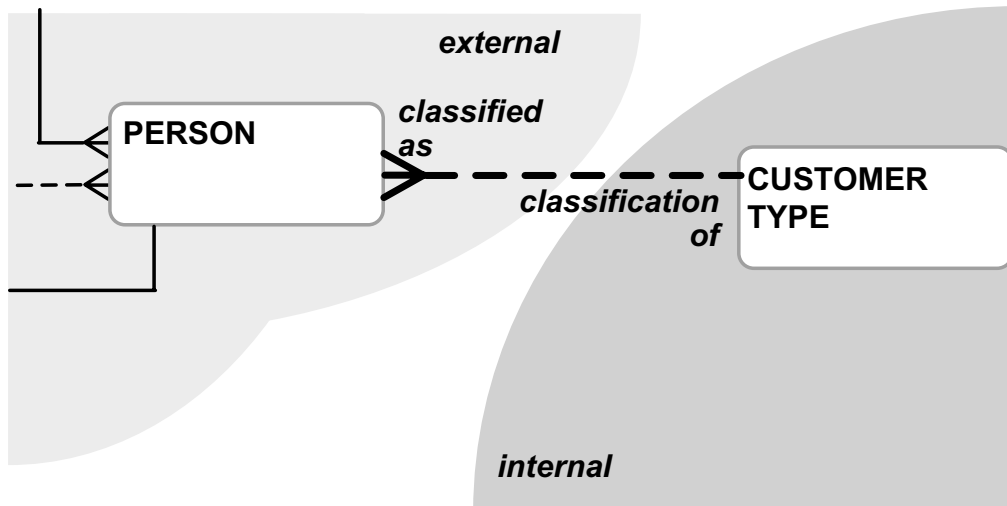


ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

Oracle Internal & OAI Use Only

m:1 관계 분석



ORACLE

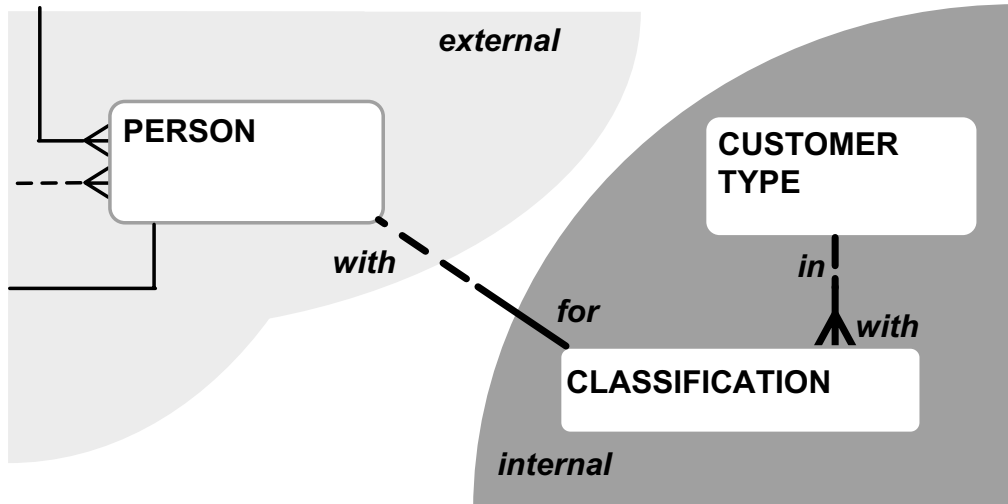
Copyright © Oracle Corporation, 2002. All rights reserved.

m:1 관계 분석

시스템이 외부의 **PERSON** 엔티티로부터 내부 엔티티 중 하나인 **CUSTOMER TYPE**으로 m:1 관계를 생성해야 한다고 가정 합니다.

Oracle Internal & OAI Use Only

m:1 관계 분석



ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

m:1 관계 분석

이렇게 하면 나중에 타사 PERSONS 테이블의 테이블 구조의 변화를 초래하게 됩니다. 이것은 바람직하지 않으며(타사는 종종 그러한 행위를 금지하는 계약을 체결할 것을 요청합니다) 해당 테이블에 대한 권한이 없을 경우에는 때때로 불가능하기조차 합니다.

위의 모델은 외부 엔티티 PERSON을 그대로 두고 내부로부터 참조를 수행합니다. m:1 관계는 하나의 CLASSIFICATION 엔티티와 두 개의 관계로 대체됩니다.

정규화 규칙

정규 폼 규칙	설명
첫번째 정규 폼	모든 속성은 단일 값입니다.
두번째 정규 폼(2NF)	속성은 엔티티의 전체 고유한 식별자에 종속되어야 합니다.
세번째 정규 폼	비-UID 속성은 다른 비-UID 속성에 종속될 수 없습니다.

"정규화된 엔티티-관계 데이터 모델은 자동으로 정규화된 관계형 데이터베이스 설계로 변환됩니다".

"세번째 정규 폼은 중복성을 제거한 데이터베이스 설계를 위해 일반적으로 수용되는 목표입니다".

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

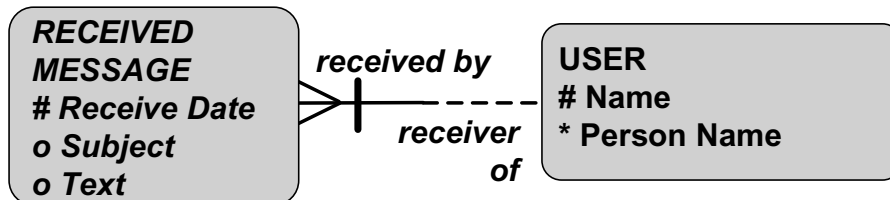
데이터 모델링 시 정규화

정규화는 관계형 데이터베이스 개념입니다. 그러나, 정확한 엔티티 모델을 생성했다면, 설계 시 생성된 테이블은 정규화 규칙을 따를 것입니다. 관계형 데이터베이스 설계로부터 비롯된 각각의 공식적인 정규화 규칙은 해당 데이터 모델 분석을 가집니다. ER 모델에서 속성 배치를 검증하는 데 사용될 수 있는 분석은 다음과 같습니다.

데이터 모델링의 첫번째 정규 폼

USER
Name
* Person Name
* Message Receive Date
o Message Subject
o MessageText

모든 속성은 단일 값을 가져야 합니다.



ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

데이터 모델링의 첫번째 정규 폼

모든 속성은 단일 값을 가져야 합니다.

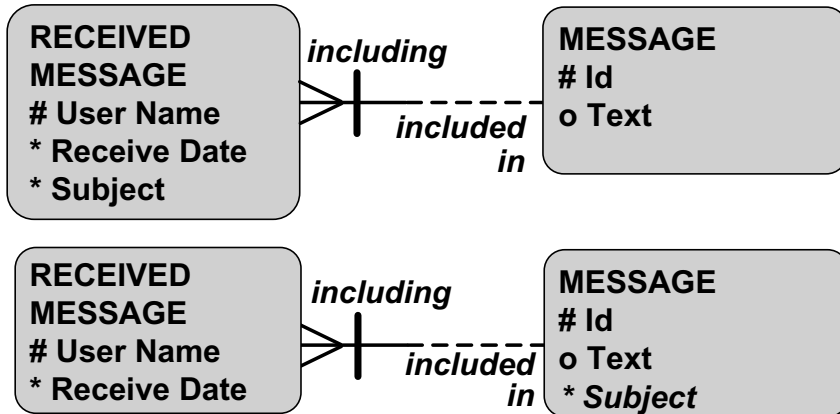
각 속성이 각 엔티티 발생에 대하여 단일 값을 가지는지 검증합니다. 속성에 반복 값이 있어서는 안됩니다.

종종 속성 이름에 *Message Subject* 및 *Message Text*와 같이 동일한 (엔티티) 이름이 있다는 사실로 잘못 배치된 속성을 인식할 수 있습니다.

속성에 여러 값이 있을 경우에는, 추가 엔티티를 생성하고 이것을 m:1 관계로 원래의 엔티티와 연관시킵니다.

데이터 모델링의 두번째 정규 폼

속성은 해당 엔티티 전체의 고유 식별자에 종속되어야 합니다.



ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

데이터 모델링의 두번째 정규 폼

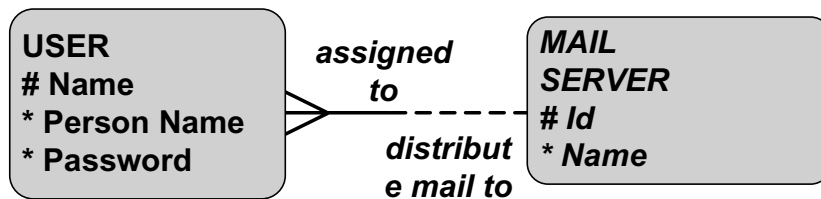
속성은 해당 엔티티 전체의 고유 식별자에 종속되어야 합니다.

각 속성이 해당 엔티티 전체의 고유 식별자에 종속되어 있는지 검증합니다. 각각의 특정한 UID 인스턴스는 각 속성의 단일 인스턴스를 결정해야 합니다. 속성이 해당 엔티티의 UID의 일부에만 종속되지 않는지 검증합니다. 만일 그렇다면, 그 속성은 잘못 위치된 것이며 이동시켜야 합니다.

데이터 모델링의 세번째 정규 폼

USER
Name
* Person Name
* Password
* Server Id
* Server Name

비 UID 속성은 다른 비 UID 속성에 종속될 수 없습니다.



ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

데이터 모델링의 세번째 정규 폼

비 UID 속성은 다른 비 UID 속성에 종속될 수 없습니다. 속성이 비-UID 속성에 종속되어 있을 경우, 종속하는 속성과 그것이 종속되어 있는 속성을 모두 새로운 관련된 엔티티로 이동합니다.

요약

- 관계는 엔티티가 연결되는 방식을 나타냅니다.
- 처음에 관계는 종종 m:m 유형인 것처럼 보입니다.
- 결국 관계는 m:1 유형이 가장 일반적입니다.
- 관계는 다음으로 분석될 수 있습니다.
 - 두 개의 새로운 관계
 - 하나의 교차 엔티티
- 세번째 정규 폼은 일반적으로 표준으로 받아들여 집니다.

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

요약

관계는 엔티티들을 연결하고 그것들이 연결되는 방식을 표현합니다. 열 가지 유형의 관계가 있으며, 4가지는 1:m 유형, 3가지는 m:m 유형, 그리고 3가지는 1:1 유형입니다.

1 측에서 선택인 m:1 관계는 완료된 ER 모델에서 가장 일반적인 유형입니다. 이 유형은 관계형 데이터베이스에서 매우 쉽게 구현할 수 있습니다.

ER 모델 생성 과정 중 초기에는 m:m 관계가 많습니다. 이들 중 대부분은 면밀한 조사를 거치고 나면 사라집니다.

관계는 속성을 가질 수 없습니다. 이 경우, 관계를 하나의 교체 엔티티와 두 개의 관계로 분석해야 합니다.

다른 유형들은 덜 일반적입니다—일부 유형은 양 끝에서 필수인 m:1 관계와 같은 실제보다 바람직한 상황을 표현합니다.

정규화된 데이터 모델은 정규화된 관계형 데이터베이스 설계로 이어집니다. 세번째 정규 폼은 일반적으로 받아들여지는 표준입니다.

연습

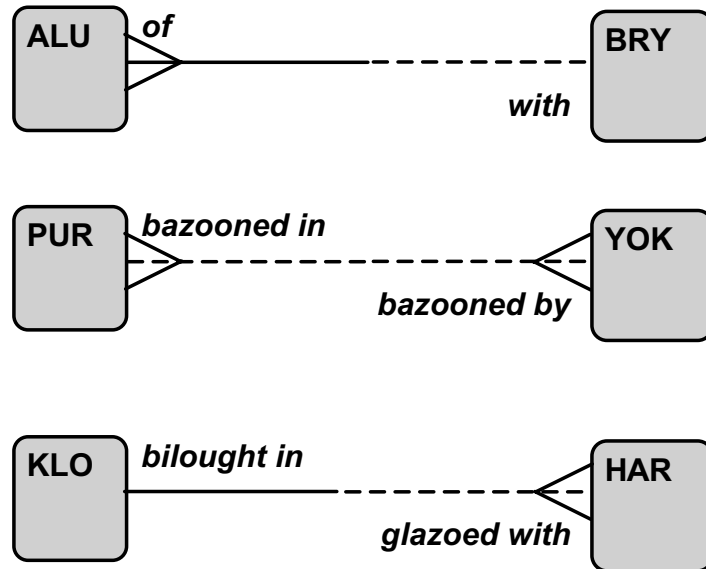
- 관계 읽기
- 컨텍스트 찾기
- 교차 엔티티 명명
- 영수증
- Moonlight P&O
- 가격 목록
- 전자 메일
- 휴일

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

Oracle Internal & OAI Use Only

연습: 관계 읽기



ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

연습 3-1: 관계 읽기

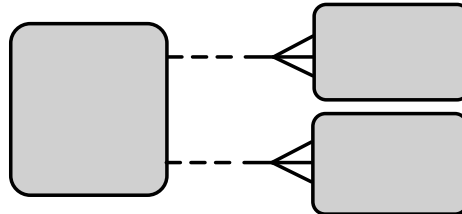
목표

이 연습의 목표는 ER 도표에서 관계를 읽는 방법을 배우는 것입니다.

과제

도표를 두 시각에서 읽어 보십시오. 해당 업무 영역을 파악하고 있으나 ER 모델을 읽는 방법을 모르는 사람들이 이해하고 확인할 수 있는 문장을 만드십시오.

컨텍스트 찾기(1)



ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

연습 3-2: 컨텍스트 찾기

목표

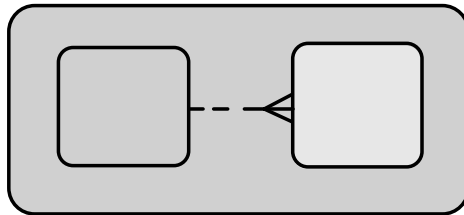
이 연습의 목적은 모델링 기술을 사용하는 것입니다.

과제

다음 ER 도표에 따라 모델에 맞는 컨텍스트를 찾으십시오.

Oracle Internal & OAI Use Only

컨텍스트 찾기(2)

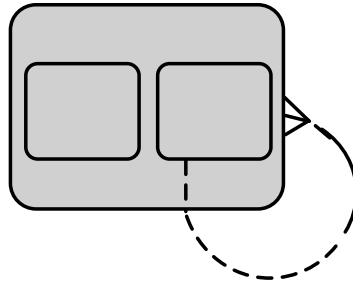


ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

Oracle Internal & OAI Use Only

컨텍스트 찾기(3)

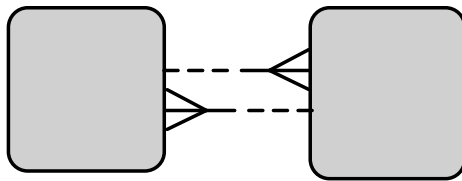


ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

Oracle Internal & OAI Use Only

컨텍스트 찾기(4)

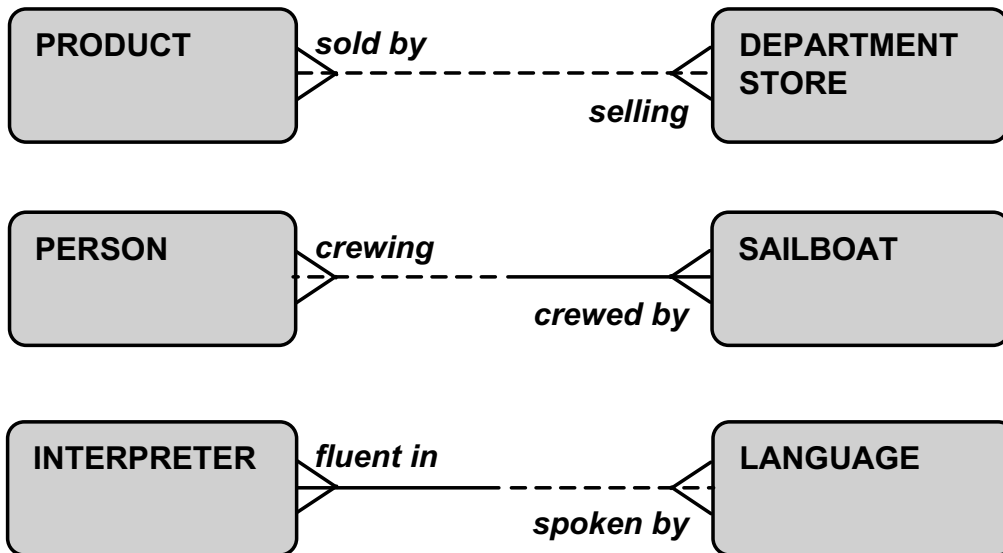


ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

Oracle Internal & OAI Use Only

연습: 교차 엔티티 명명



ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

연습 3-3: 교차 엔티티 명명

목표

이 연습의 목표는 m:m 관계를 분석한 후 교차 엔티티에 대한 적합한 이름을 찾는 것입니다.

과제

1. 다음의 m:m 관계를 분석하십시오. 교차 엔티티에 대해 적용 가능한 이름을 찾으십시오.
2. 교차 엔티티별로 다소 중요한 업무 컨텍스트에 적합한 하나 이상의 속성을 만드십시오. 해당 속성에 명확한 이름을 지정하십시오.

연습: 영수증

Served by: Dennis

Till: 3 Dec 8, 4:35 pm

```
-----
CAPPUCC M    3.60
              * 2    7.20
CREAM         .75
              * 2    1.50
APPLE PIE          3.50
BLACKB MUF       4.50
<SUB>           16.70
tax 12%          2.00
<TOTAL>        18.70
=====
CASH           20.00
RETURN         1.30
-----
```

Hope to serve you again
@MOONLIGHT COFFEES
25 Phillis Rd, Atlanta

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

연습 3-4: 영수증

목표

이 연습의 목적은 개념적 데이터 모델에 대한 기초로 실무에 사용되는 데이터의 간단한 소스를 사용하는 것입니다.

시나리오

당신은 Moonlight Coffees에서 계약직으로 일하고 있습니다. 수행해야 할 작업은 업무에 대한 개념적 데이터 모델을 생성하는 것입니다. Moonlight에 대한 모든 종류의 문서를 수집했습니다. 다음은 한 상점에서 받은 영수증의 예입니다.

과제

영수증의 정보를 사용하여 엔티티 및 속성 목록을 만드십시오.

연습: Moonlight P&O

- Moonlight Coffee의 모든 사원은 "Global Pricing" 또는 "HQ"와 같은 매장 또는 판매점에서 일합니다. 전사원은 국가 본점 중 하나에서 급료를 받습니다. 예를 들어, Jill은 런던에서 상점 관리자로 일하고, Werner는 독일에서 회계 업무를 맡고 있는 재무 관리자입니다.
- 모든 상점은 국가별로 하나의 본점에 속해 있습니다("the countries"). 국가별로 본점이 하나씩 있습니다. HQ 자체를 제외한 모든 국가별 본점과 부서는 HQ에 보고합니다.
- 전사원은 시간제로 근무할 수 있습니다. Lynn은 9월 1일부터 Product Development 업무에 근무 시간의 80%가 배정되었습니다. 그 이전에는 정규직으로 근무했습니다.

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

연습 3-5: Moonlight P&O

목표

이 연습의 목적은 새로운 정보와 새로운 요건을 기준으로 ER 모델을 반복적으로 생성하는 것입니다.

시나리오

당신은 여전히 Moonlight Coffees의 계약직으로 근무하고 있으며, 분명히 아주 훌륭히 업무를 수행하고 있습니다!

연습 3-5: Moonlight P&O

과제

1. 다음의 직원 및 조직 정보를 기준으로 엔티티 관계 모델을 생성하십시오.
2. 이 정보를 기준으로 도표를 확장 또는 수정하십시오.
3. 그리고 다시 다음을 수행합니다.
4. 다음과 같은 새로운 정보를 수용할 수 있도록 필요하고 가능한 경우 모델을 변경합니다.
 - Jan은 프라하에 있는 두 개의 상점에서 교대로 근무합니다.
 - 지난 해 Tess는 브라질에서 상점 관리자직을 사임하고 토론토로 이주하였습니다. 최근에 그녀는 토론토 공항의 판매점으로 합류했습니다.
 - 직접 보고서의 수를 줄이기 위해, 부서와 각 국의 본점들은 본사 대신 다른 부서에 보고할 수 있습니다.
 - 룩셈부르크에 있는 상점은 벨기에에 보고합니다.
 - 권한 충돌을 방지하기 위해 전사원은 동시에 한 부서와 한 상점에서 일할 수 없습니다.
5. 당신이 작성한 모델로 다음 질문들에 답변할 수 있습니까?
 - 현재 Operations 부서에서 근무하는 사람은 누구입니까?
 - 현재 프랑스 몽마르뜨르에 있는 Moonlight La Lune에서 근무하는 사람은 누구입니까?
 - 현재 프랑스의 Marketing 부서에서 근무하는 직원이 있습니까?
 - 사원 수를 기준으로 최대 국가는 어디입니까? 매니저 수를 기준으로 하면 어떻습니까? 시간제 근무자 수를 기준으로 하면 어떻습니까?
 - Lynn이 회사에 근무한지 5년째 되는 날은 언제입니까? 마찬가지로 Moonlight에서 근무하는 Tess의 근무 5년째 되는 날은 언제입니까?
 - 퇴직자 수가 가장 낮은 국가는 어디입니까?

Oracle Internal & OAI Use Only

price list

25 Phillis Road, Atlanta

visit us at www.moonlight.com

연습: 가격 목록

	small	medium	large	
regular coffee	2.25	2.90	3.50	
cappuccino	2.90	3.60	4.20	
café latte	2.60	3.20	3.90	
special coffee	3.10	3.70	4.40	
espresso	2.25	2.90	3.50	
coffee of the day	2.00	2.50	3.00	
<i>decaffeinated</i>	.25	.50	.75	<i>extra</i>
<i>black tea</i>	2.25	2.90	3.50	
<i>infusions</i>	2.60	3.20	3.90	
<i>herbal teas</i>	2.90	3.60	4.20	
<i>tea of the day</i>	2.00	2.50	3.00	
<i>decaffeinated</i>	.25	.50	.75	<i>extra</i>
<i>milk</i>	1.25	1.90	2.50	
<i>soft drinks</i>	2.25	2.90	3.50	
<i>soda water</i>	2.25	2.90	3.50	
<i>mineral water</i>	2.90	3.60	4.20	
apple pie				3.50
strawberry cheesecake			3.50	
whole wheat oats muffin with almonds				3.90
blackberry muffin				4.50
fruitcake				4.50
cake of the day				4.00
additional whipped cream				.75

Sales Tax included
September 16

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

연습 3-6: 가격 목록

목표

이 연습의 목적은 개념적 데이터 모델에 대한 기초로 실제 데이터의 간단한 소스를 사용하는 것입니다.

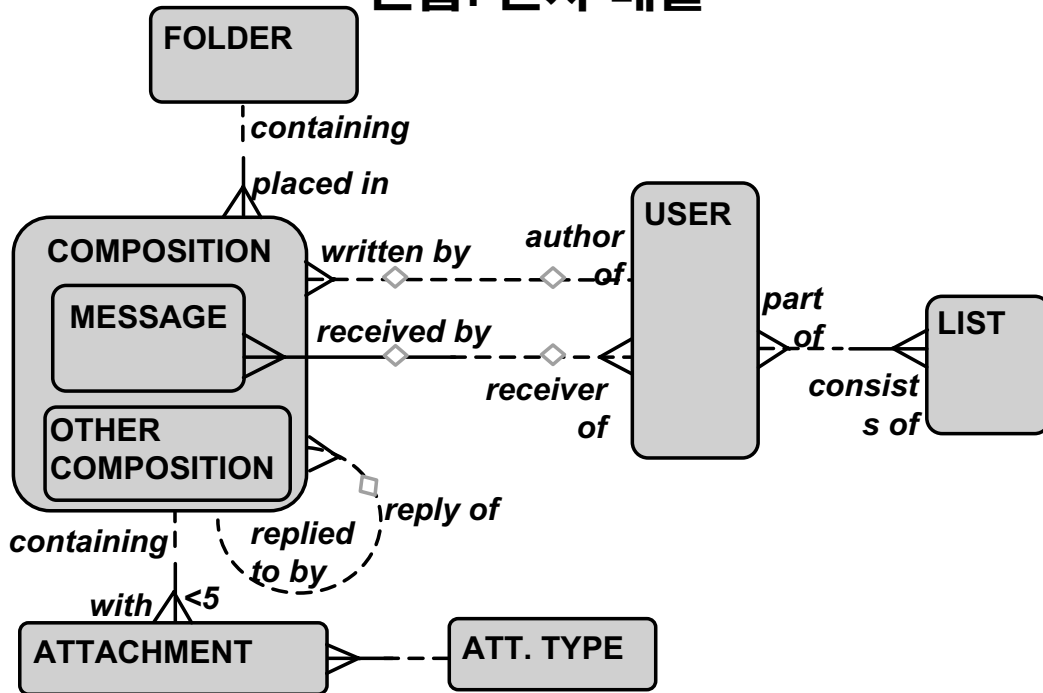
시나리오

당신은 Moonlight Coffees에서 계약직으로 일하고 있습니다.

과제

Moonlight Coffee Store 중 한 매장의 가격표를 기준으로 ER 모델을 만드십시오.

연습: 전자 메일



ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

연습 3-7: 전자 메일

목표

이 연습의 목표는 기존의 개념적 데이터 모델을 확장하는 것입니다.

시나리오

과제

제공된 모델을 시작점으로 사용합니다. 다음 기능을 편리하게 사용할 수 있도록 엔티티, 속성 및 관계를 추가, 삭제 또는 변경합니다.

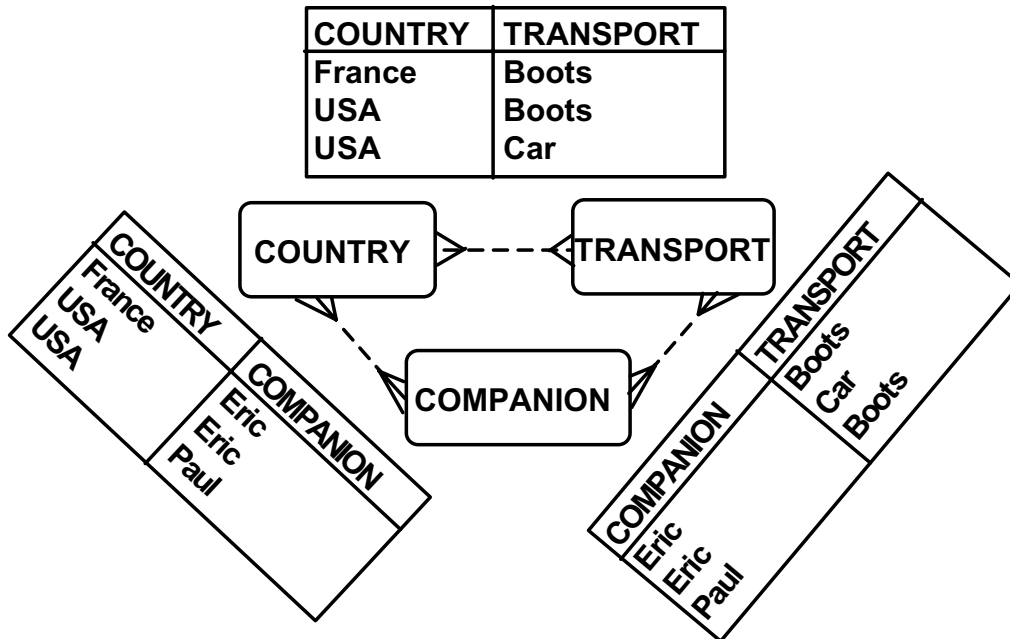
1. 사용자는 다른 사용자에게 애칭(별칭)을 생성할 수 있어야 합니다.
2. 폴더는 다른 폴더를 포함할 수 있습니다.
3. 사용자는 composition을 전달할 수 있어야 합니다. forward는 전달된 메시지와 함께 자동으로 전송되는 새 메시지입니다.
4. 모든 폴더와 목록은 사용자가 소유합니다.

해결 과제:

5. 메일 목록은 사용자와 다른 목록을 모두 포함할 수 있습니다.
6. 메일 목록은 "giovanni_papini@yahoo.com"과 같은 외부 주소를 포함할 수 있습니다.
7. 애칭은 외부 주소에 대한 별칭일 수 있습니다.

연습: 휴일

"Paul과 나는 미국에서 여행을 했습니다. Eric과 나는 프랑스에서 여행을 하고 작년에 미국에서 차를 한 대 빌렸습니다."



ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

연습 3-8: 휴일

목표(54페이지 참조)

이 연습의 목적은 개념적 데이터 모델에 대해 품질 검사를 수행하는 것입니다.

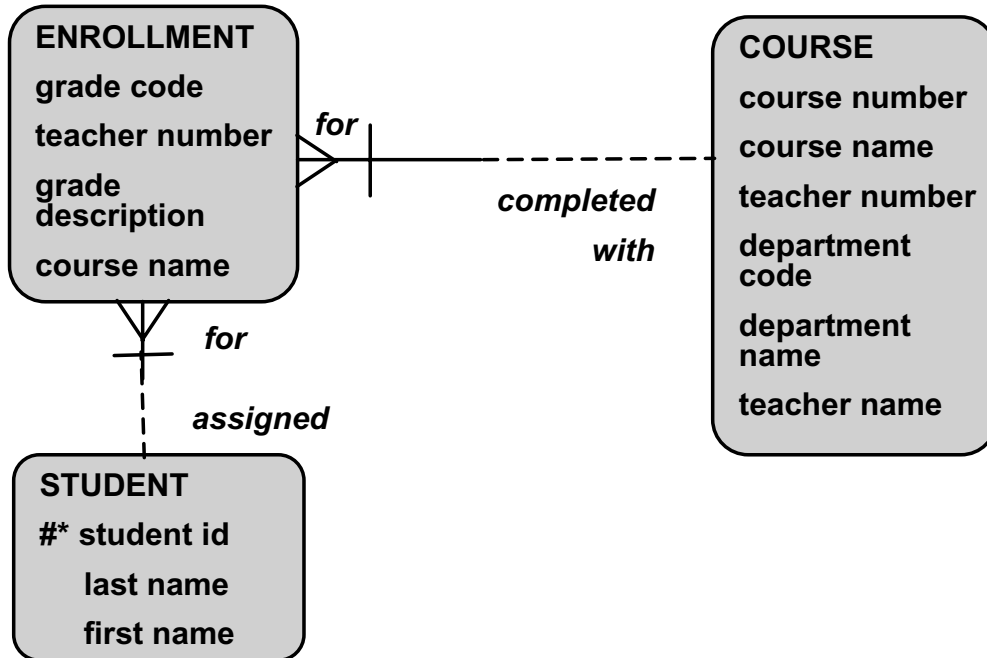
시나리오

"Paul과 나는 미국에서 여행을 했습니다. Eric과 나는 프랑스에서 여행을 하고 작년에 미국에서 차를 한 대 빌렸습니다."

과제

시나리오 텍스트를 기준으로 하는 아래 제공된 모델에 주석을 다십시오.

연습: ER 모델 정규화



ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

연습 3-9: ER 모델 정규화

목표

이 연습의 목적은 정규화되지 않은 ER 모델을 세번째 정규 폼에 배치하는 것입니다.

과제

1. 다음 ER 모델에 대해, 정규화 규칙에 따라 각 엔티티를 평가하고, 잘못 배치된 속성을 식별하고, 잘못 배치된 각 속성이 위반하는 정규화 규칙이 무엇인지 설명하십시오.
2. 세번째 정규화 폼에 ER 도표를 다시 그리십시오. (선택 사항)

Oracle Internal & OAI Use Only

4

제약 조건

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

Oracle Internal & OAI Use Only

개요

- **고유 식별자**
- **호**
- **도메인**
- **기타 다양한 제약 조건**

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

소개

이 단원은 업무에 적용되는 제약 조건에 대해 다룹니다. 제약 조건은 업무 규칙이라고도 합니다. 이러한 제약 조건 중 일부는 쉽게 모델링할 수 있습니다. 일부는 도표로 표시할 수 있지만 그 결과 명확성이 감소하므로 만족스럽지 않을 수도 있습니다. 일부 제약 조건은 전혀 모델링할 수 없습니다. 이러한 제약 조건은 별도의 문서에 나열해야 합니다.

목표

이 단원을 마치면 다음을 수행할 수 있습니다.

- 실제 세계에서 식별 문제 설명
- 모델에 고유 식별자를 추가하고 식별자가 표시되는 방식 파악
- 올바른 고유 식별자와 잘못된 고유 식별자 인식
- 모델에서 호가 필요한 시점 결정
- 호와 하위 유형 간의 유사성 설명
- ER 도표에 표시할 수 없는 다양한 유형의 업무 제약 조건 설명

렘브란트



ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

식별

언어를 통한 식별

렘브란트가 네델란드에서 태어났다는 사실은 누구나 알고 있다고 가정해도 무리가 없을 것입니다. 아마도 대부분의 사람들은 렘브란트가 농가에서 Pajamas의 사생아로 태어났다는 사실은 모를 것입니다. 렘브란트에게는 쌍둥이 누이가 있었으며 결혼을 하지는 않았지만 많은 자녀를 두었습니다. 렘브란트와 그의 자손들은 모두 꼬리 끝에 네 개의 흰색 줄무늬가 있기 때문에 쉽게 그들을 알아볼 수 있습니다.

식별이란 우리가 이야기하는 사물이나 사람을 인지하는 것입니다. 분명히 렘브란트라는 이름은 유명한 화가의 이름만은 아니며, 다른 사람이나 심지어 고양이도 동일한 이름을 가질 수 있습니다.

일상 대화에서 우리는 일반적으로 우리와 우리가 대화를 나누는 사람들이 충분히 동일한 상황을 공유하고 서로의 직업과 관심사항에 대해 충분히 알고 있어서 서로 무슨 이야기를 하고 있는지 이해할 수 있다고 가정할 수 있습니다. 언어는 언제나 많은 모호함을 안고 있어서 구체적이지 않은 의사소통 수단이지만, 사람들은 매우 잘 해석하여 이해합니다.

식별

컴퓨터는 해석의 여지가 많지 않은 좀 더 구체적인 방식으로 의사소통이 이루어져야 합니다. "Rembrandt the painter" 또는 "Rembrandt van Rijn, born in 1606" 또는 심지어 이들을 모두 조합한 "Rembrandt van Rijn, the painter, born in 1606"이라고 말해 주어야 시스템은 이 렘브란트를 이름이 동일한 다른 유명인과 구별할 수 있습니다.

식별의 문제

식별의 문제에는 세 가지 측면이 있습니다. 하나는 실제 세계에서 식별입니다. 즉, 매우 유사한 특성을 가지는 실제 세계의 두 가지 사물을 어떻게 구분합니까? 이것이 가장 어려운 측면입니다. 두번째는 데이터베이스 시스템 내에서의 식별입니다. 즉, 테이블에 있는 행들을 어떻게 구분합니까? 이것은 첫번째보다는 덜 복잡합니다. 세번째 문제는 표현에 대한 것입니다. 즉, 테이블에 있는 행이 어떤 실세계 사물을 표현하는지 어떻게 알 수 있습니까?

실제 세계에서 식별

실제 세계의 많은 사물들은 식별이 불가능하지는 않지만 어렵습니다. 가령, 두 대의 택시, 두 명의 고객, 두 가지 버전의 계약 또는 두 번의 쇼스타코비치의 제4현악 사중주 공연을 구별하는 것은 어렵습니다. 일반적으로 실제 세계 사물은 확실한 식별이 불가능합니다. 우리는 상당한 모호함을 안고 살아가야 합니다. 예를 들어, MN4606이라는 번호판을 단 건너편의 자동차가 내가 지난 주에 보았던 그 번호판을 가진 차와 동일한 자동차라는 사실을 어떻게 확신할 수 있습니까? 심지어 나는 그것이 동일한 번호판인지 여부도 확신할 수 없습니다. 보통의 상황에서는 의심할 이유가 없지만, 그것은 확실성과는 다른 것입니다. 때때로 사람들은 혼란을 일으키는 데 대해 나름대로 이유가 있습니다.

다행히 실제 세계의 일부 사물들은 자신의 활동 범위 내에 있기 때문에 식별이 더 쉽습니다. 이러한 범위에서는 규칙을 정의할 수 있습니다. 예를 들어, 회사가 송장을 발송할 때 송장마다 고유 번호를 부여할 수 있습니다. 업무상 사람들이 ElectronicMail 사용자 이름(ID)을 생성할 때 이러한 이름을 고유하게 지정할 수 있습니다.

데이터베이스 내에서의 식별

일반적으로 데이터베이스 시스템은 동일한 테이블 내에서 행이 두 번 저장되지 않도록 또는 더 정확하게 말해서 값의 특정한 조합이 두 번 저장되지 않도록 보장할 수 있습니다. 기술적인 문제는 사용되는 표준 소프트웨어에 의해 해결됩니다.

표현

나머지 문제는 어떤 실제 세계 사물이 테이블의 특정 행에 의해 표현되는가를 항상 알 수 있도록 해야 합니다. 이 문제의 해결책은 상황에 따라 크게 달라집니다. 동일한 회사에 근무하는 두 명의 다른 사원이 동일한 성, 동일한 성과 이니셜 또는 동일한 성과 이니셜, 생년월일을 가질 가능성은 어느 정도일까요?

식별과 표시

EMPLOYEES

Name	Initials	Birthdate
PAPINI	G.	02-FEB-1954
HIDE	T.M.	11-JUN-1961
PAPINI	G.	02-FEB-1945
BAKER	S.J.T.	24-SEP-1958

G. Papini, please?



ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

식별과 표시

분명히 회사가 5명의 사원을 고용할 때와 50,000명의 사원을 고용할 때 대답은 달라질 수 있습니다.

EMPLOYEE에 대한 새로운 식별 속성, 이를 테면 ID를 추가해도 위의 문제는 부분적으로 해결될 뿐이라는 점을 알아야 합니다. 이것은 데이터베이스 안에서는 매우 유용할 것입니다. 그러나 보통은 사원들이 다른 사람들의 ID는 몰랐고 자신의 ID도 잘 알지 못하는 실제 세계에서는 별로 도움이 되지 않을 것입니다. 이러한 종류의 ID 속성은 종종 내부 식별로만 효과가 있고 외부 식별로는 효과가 없습니다.

고유 식별자 예제

JOB	Name
COMPUTER IN NETWORK	IP Address
TELEPHONE	Country code, Area code, Telephone number
EMPLOYEE	Employee number <i>or</i> Name, Initials, Birth Date
MAIL LIST	Name, Owner

ORACLE

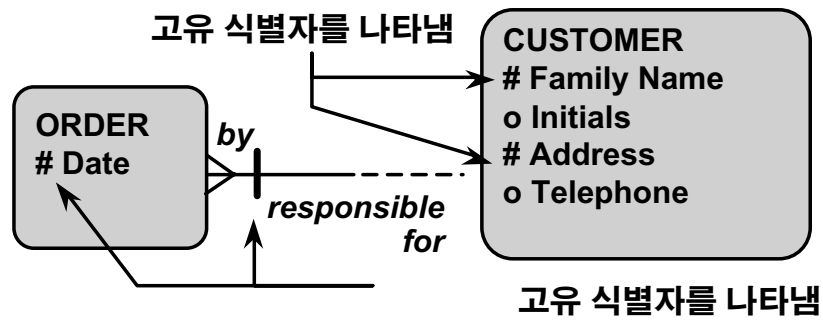
Copyright © Oracle Corporation, 2002. All rights reserved.

고유 식별자

이야기하는 내용을 알려면, 모든 엔티티에 대해 엔티티 인스턴스를 고유하게 식별하는 하나의 값 또는 값들의 조합을 찾아야 합니다. 이러한 값 또는 조합을 엔티티에 대한 고유 식별자라고 합니다.

Oracle Internal & OAI Use Only

고유 식별자



ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

고유 식별자

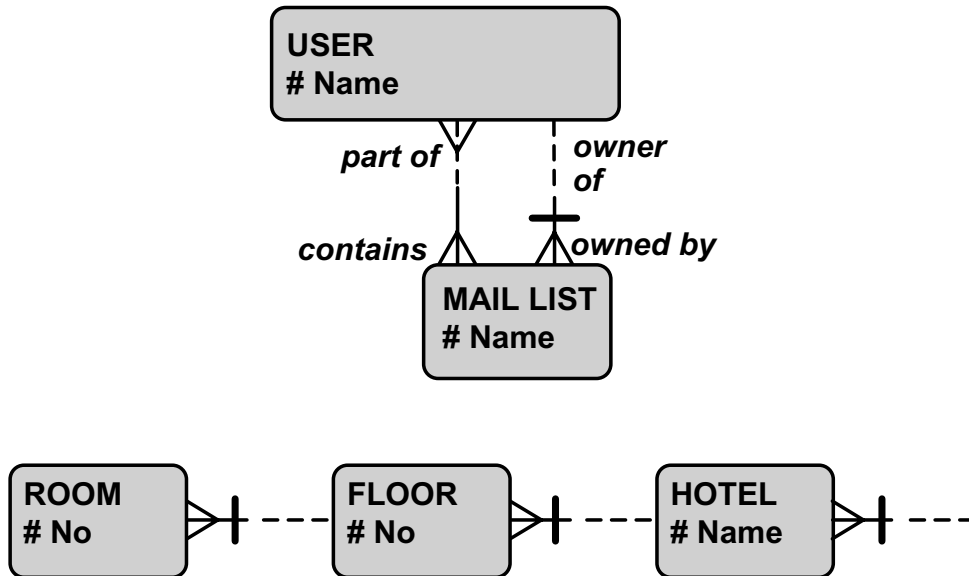
MAIL LIST 예제는 고유 식별자가 반드시 속성의 조합일 필요는 없다는 것을 보여줍니다. MAIL LIST의 소유자는 실제로 관계에 의해 표시됩니다.

UID 표시

ER 도표에서 엔티티의 UID 구성 요소는 다음과 같이 표시됩니다.

- 속성 번호
- 관계에 대해 관계 끝을 가로지르는 작은 세로줄로(세로줄 표시 관계)

고유 식별자



ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

고유 식별자

단일 속성 UID

위 모델은 ElectronicMail의 USER가 Name 속성에 의해서만 식별된다는 것을 보여줍니다. 많은 엔티티는 단일 속성에 의해 식별됩니다. 단일 속성 UID에 대한 일반적인 후보 속성(사용 가능할 경우)은 Id, Code, Name, Description, Reference입니다.

복수 속성 UID

하나의 엔티티는 여러 속성으로 구성된 하나의 UID를 가집니다. 예를 들어, 소프트웨어 패키지는 일반적으로 Oracle Designer, version 9i와 같이 이름과 버전에 의해 식별될 수 있습니다.

고유 식별자

구성된 UID

위의 그림에서 MAIL LIST는 LIST의 Name과 LIST를 소유한 USER에 의해 식별됩니다. 이것은 LIST의 OWNER와 Name 조합이 고유한 쌍이어야 한다는 것을 의미합니다.

이것은 모든 USER가 자신의 LIST 인스턴스에 고유한 이름을 지정해야 하지만, 다른 사용자가 지정하는 이름에 대해 우려할 필요가 없다는 것을 의미합니다. 또한 인스턴스가 서로 다른 USER에 의해 소유되는 한, 시스템은 동일한 이름을 가진 많은 LIST 인스턴스를 가질 수 있다는 것을 의미합니다.

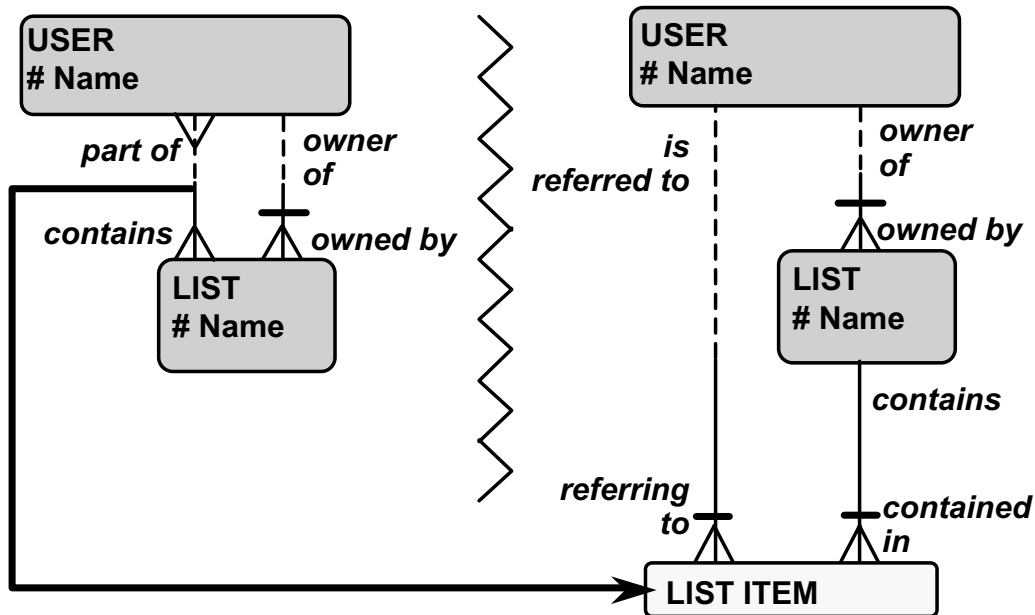
Name이 고유해야 하기 때문에 USER는 또한 이 메일 시스템 내에 구성된 UID를 가진다고 주장할 수도 있습니다. 이를 증명하기 위해 USER에서 PROVIDER로의 관계 외에 대단히 높은 레벨의 엔티티인 MAIL PROVIDER를 추가할 수 있습니다. 그러면 이 관계는 USER UID의 일부가 됩니다.

연쇄 구성된 UID

엔티티가 다른 엔티티에 대해 세로줄 표시 관계를 가지고, 그 다른 엔티티는 다시 세번째 엔티티에 대해 세로줄 표시 관계를 가지는 등의 계속 이어지는 관계를 가지는 경우는 일반적입니다.

Oracle Internal & OAI Use Only

복수 관계 UID



ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

UID: 관계로만 구성됨

고유 식별자는 또한 관계로만 구성될 수 있습니다.

도표의 오른쪽 하단에 보이는 LIST ITEM 엔티티는 LIST와 USER 간의 분석된 m:m 관계에서 비롯된 것입니다.

위 모델은 LIST ITEM이 USER와 LIST의 조합에 의해 식별된다는 것을 보여줍니다. 즉, 위 모델은 ITEM이 상이한 USER를 참조하는 한, LIST는 사용자가 원하는 만큼 많은 ITEM을 포함할 수 있다는 것을 보여줍니다.

이 결과, 다음과 같은 정의가 도출됩니다.

엔티티의 UID(고유 식별자)는 값의 고유성을 선언하는 제약 조건입니다. 즉, UID는 하나 이상의 속성, 하나 이상의 관계 또는 엔티티의 속성과 관계의 조합으로 구성됩니다.

결과적으로 UID의 모든 구성 요소가 선택적인 것은 아닙니다.

간접 식별

식별은 간접적인 구조를 사용하여 발생합니다. 즉 엔티티의 인스턴스가 자신이 참조하는 또 다른 엔티티의 인스턴스에 의해서만 식별될 때 발생합니다.

UID: 관계로만 구성됨

예제

- 많은 사무실에서는 직원들을 배지로 식별하고, 배지는 코드로 식별합니다.
- 전세계적으로는 여권에 부착된 사진으로 사람을 식별합니다.
- 유럽 공동체의 모든 젓소는 귀에 부착하기로 되어 있는 딱지의 번호로 식별됩니다.
- 암스테르담 국제 공항에 자동차를 주차할 때는 입구에 있는 슬롯에 신용 카드를 넣어 주차장에 진입합니다. 주차 이벤트는 차량을 주차한 사람의 신용 카드로 식별됩니다. 이것은 이중 간접 식별입니다.

분명히 이러한 식별 구조는 100% 신뢰할 수 없지만, 특정 상황에서 최대한 의존할 수 있는 구조일 것입니다.

이러한 간접 식별 모델은 다음 그림의 오른쪽 하단에 나타나 있습니다. S의 인스턴스는 자신이 참조하는 T라는 단일 인스턴스에 의해 식별됩니다. 즉, UID는 한 개의 관계로만 구성됩니다.

복수 UID

엔티티는 여러 개의 UID를 가질 수 있습니다. 앞에서 Employee Number로 식별될 수 있고, Name, Initial 및 Birth Date를 조합하여 식별될 수 있는 EMPLOYEE의 예제를 보았습니다.

특정 시점, 일반적으로 분석이 끝날 때쯤 UID 중 하나가 기본 UID로 승격됩니다. 다른 모든 UID는 보조 UID로 불립니다.

일반적으로 가장 간결하고 기억하기 쉬운 UID를 기본 UID로 선택하게 됩니다. 물론 그 이유는 UID가 관련 테이블에 있는 하나 이상의 외래 키 열을 도출하기 때문입니다. 이러한 열은 너무 커도 안됩니다. 엔티티의 기본 UID는 선택적인 요소로 구성되지 않는 것이 좋습니다.

도표에 표시된 UID

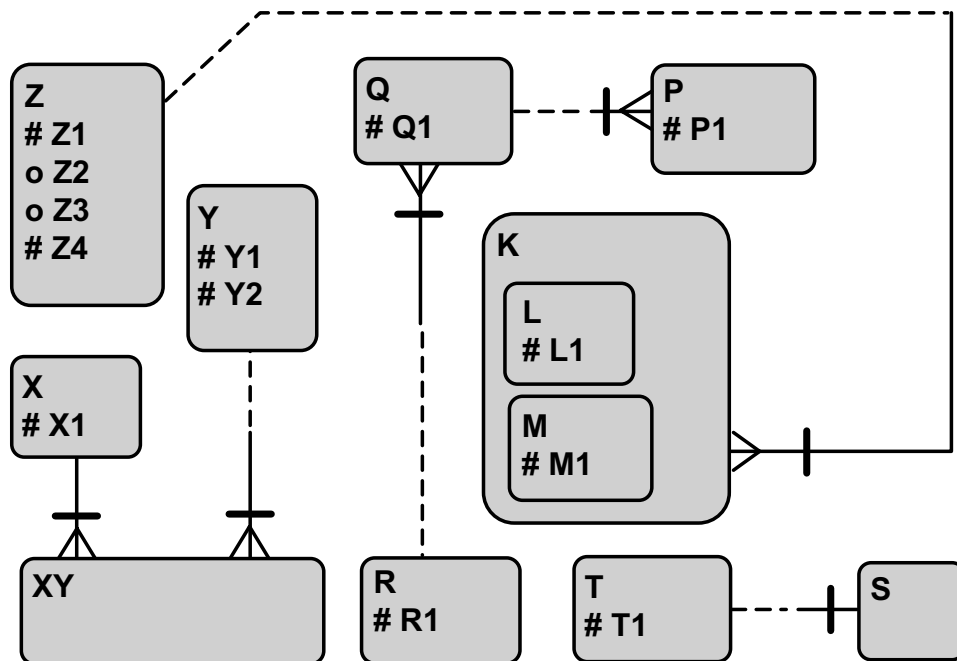
ER 도표에는 기본 UID만 표시되어 있습니다.

UID로 인해 도출되는 사항

고유 식별자는 기본 키 및 고유 키 제약 조건을 도출합니다.

Oracle Internal & OAI Use Only

올바르게 정의된 고유 식별자

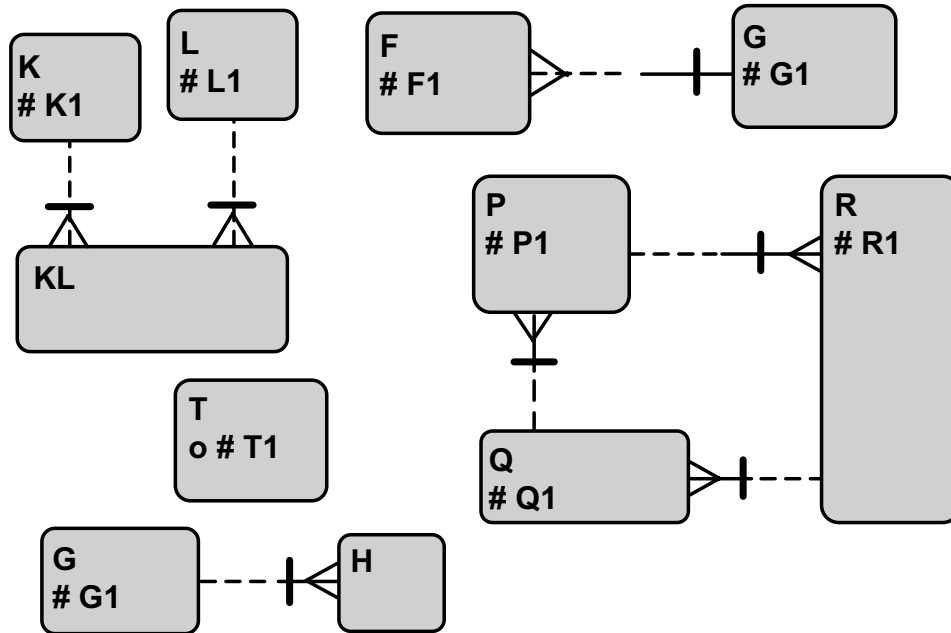


ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

Oracle Internal & OAI Use Only

잘못된 고유 식별자

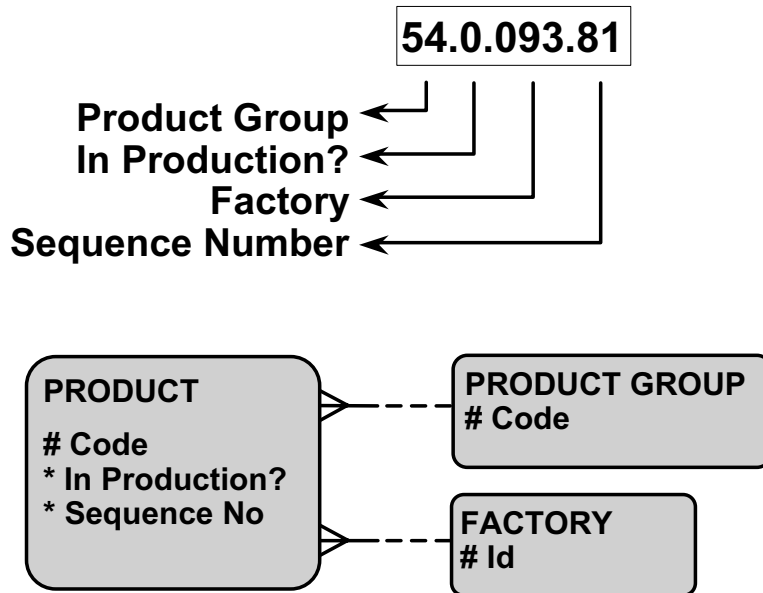


ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

Oracle Internal & OAI Use Only

정보를 포함하는 코드



ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

정보를 포함하는 식별자

실제 세계의 사물을 코딩할 때는 특별한 주의가 필요합니다. 한동안 사용되어 온 코드는 종종 정보를 포함하고 있습니다. 54.0.093.81과 같은 제품 코드를 사용하는 회사를 예로 들 수 있습니다. 이 경우 54는 제품 그룹을 나타내고, 0은 이 제품이 여전히 생산되고 있음을 나타내며, 093은 제품이 제조되는 공장을 식별하고, 81은 일련 번호입니다. 이러한 코드는 최대의 정보를 최소의 비트로 압축해야 했던 시대의 산물입니다.

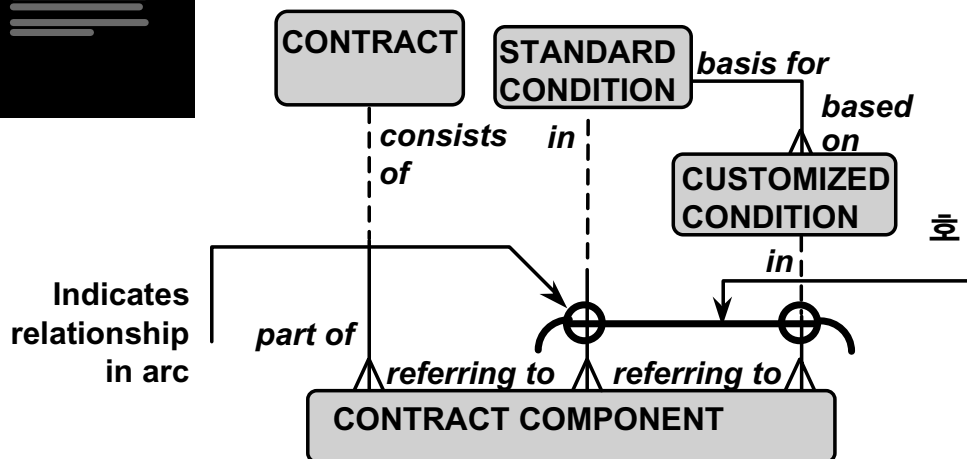
이 예제는 다음과 같이 개념적으로 모델링될 것입니다.

Code 속성은 호환성 때문에 동일한 코드를 포함할 것이지만, 이제는 아무런 의미가 없습니다. 이전의 의미는 속성과 관계로 이전되었기 때문입니다. 제품 54.0.093.81은 이제 공장 123에서 생산될 수도 있고 더 이상 제품 그룹 54에 속하지 않을 수도 있습니다.

호



"계약은 계약 구성 요소로 구성됩니다. 구성 요소는 표준 조건 또는 사용자 정의된 조건입니다."



ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

호

ElectronicMail이 웹에서 자사의 다양한 메일 화면에 위치하는 광고 영역을 임대한다고 가정합시다. 이러한 임대는 계약에 의해 관리되며, 계약은 하나 이상의 표준 조건과 사용자 정의된 조건으로 구성됩니다. 임대는 CONTRACT, CONTRACT COMPONENT, STANDARD CONDITION 및 CUSTOMIZED CONDITION이라는 네 가지 엔티티로 모델링될 수 있습니다. 위 모델을 참조하십시오. 다음 계약 조건은 어떻게 모델링할 수 있습니까? CONTRACT COMPONENT의 모든 인스턴스는 STANDARD CONDITION 또는 CUSTOMIZED CONDITION 중 하나를 참조하되 동시에 두 가지를 모두 참조하지는 않습니다.

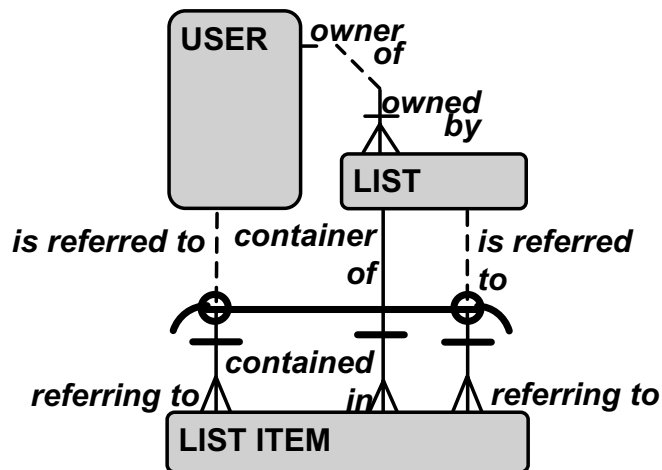
호는 둘 이상의 엔티티 관계에 대한 제약 조건으로, 해당 엔티티의 임의의 인스턴스가 특정 시점에서 가질 수 있는 유효 관계가 호에 있는 관계 중에서 한 개뿐임을 나타냅니다. 호는 하나의 배타적 관계를 모델링하거나 관계 전반에 걸쳐 모델링할 수 있습니다. 그러므로 호를 배타적 호라고도 합니다.

엔티티 속성에 대해서는 유사한 제약 조건 구성이 없습니다.

호 표시

호는 엔티티 주변에 호 모양의 선으로 그려집니다. 관계가 호에 속하는 경우에만 호와 관계선이 교차하는 곳에 작은 원이 그려집니다.

배타적 호



ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

필수 호

MAIL LIST에 다른 MAIL LIST뿐 아니라 USER도 포함될 수 있다고 가정합니다. 이는 특정 LIST ITEM이 USER나 LIST를 참조할 수 있다는 것을 의미합니다. 더 엄밀히 말하면, 이는 USER나 LIST에 대한 참조이지만 동시에 두 가지 모두에 대한 참조는 아니어야 합니다.

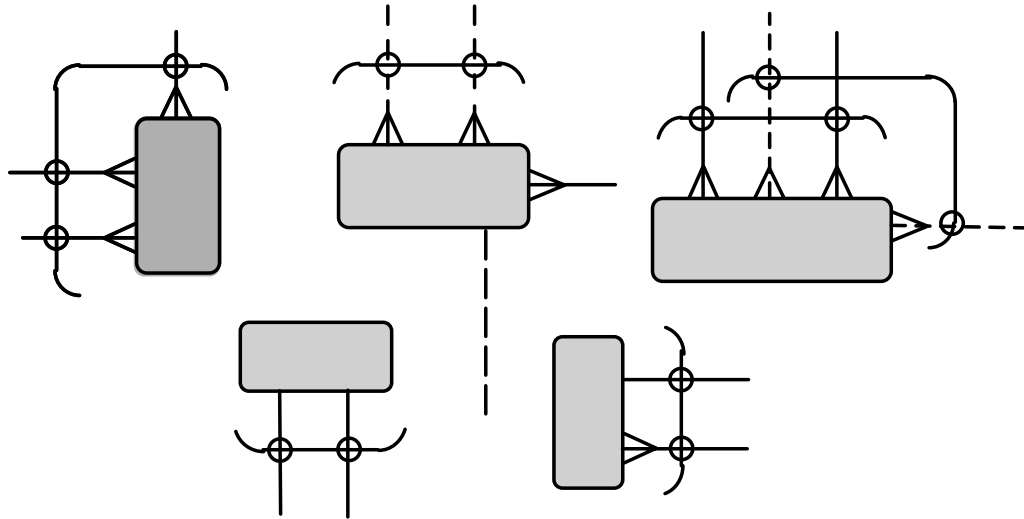
주

- LIST ITEM에서 LIST(회색으로 표시된 부분)로의 관계 *contained in/container of*는 호와 교차하는 부분에 작은 원이 없으므로 호의 일부가 아닙니다.
- UID의 일부인 관계는 또한 호의 일부일 수도 있습니다.
- LIST가 자신 이외의 LIST만 포함할 수 있다는 제약 조건은 모델에 표시할 수 없습니다.

호에서 필수 관계와 선택 관계의 비교

위의 예제에서와 같이 두 개의 필수 관계를 가로질러 호가 그려지는 경우, 이는 CONTRACT COMPONENT의 모든 인스턴스가 하나의 유효 관계를 가져야 함을 의미합니다. 두 개의 선택 관계를 가로질러 호가 그려지는 경우, 이는 하나의 인스턴스가 하나의 유효 관계를 가질 수 있음을 의미합니다.

가능한 호 구성



ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

호로 인해 도출되는 사항

오라클 데이터베이스에서 호는 일반적으로 검사 제약 조건으로 구현됩니다. 검사 제약 조건은 ISO 표준 관계형 데이터베이스 객체가 아니라는 점을 유념하십시오. 즉, 다른 데이터베이스 시스템에서는 호가 다르게 구현되어야 합니다.

호에 대한 몇 가지 규칙

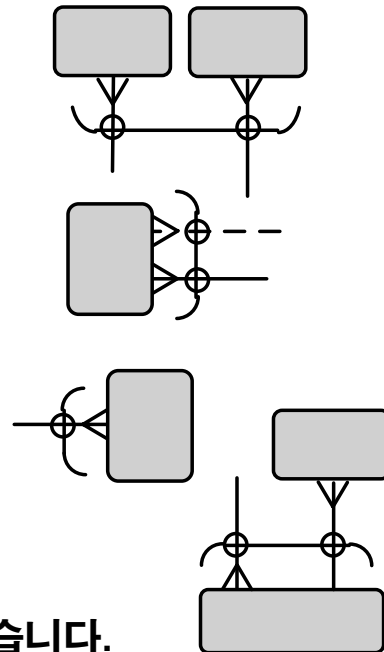
- 한 개의 호는 항상 한 개의 엔티티에 속합니다.
- 호는 셋 이상의 관계를 포함할 수 있습니다.
- 엔티티의 모든 관계가 하나의 호에 포함될 필요는 없습니다.
- 한 개의 엔티티는 여러 개의 호를 가질 수 있습니다.
- 호는 항상 동일한 선택 가능성을 가진 관계로 구성되어야 합니다.
즉, 호에 있는 모든 관계는 필수이거나 모두 선택 사항이어야 합니다.
- 호에 있는 관계는 드물기는 하지만 그 정도가 다를 수 있습니다.

호 관련 참고 사항

- 하나의 관계를 둘 이상의 호에 포함시키지 마십시오. 명확성이 감소합니다.
- 호 대신 하위 유형 모델링을 고려하십시오. (다음 단락 참조)

몇 가지 잘못된 호 구성

- 호는 한 개의 엔티티에 "속합니다".
- 호에 있는 관계는 동일한 선택 가능성을 가져야 합니다.
- 호는 적어도 두 개의 관계를 포함해야 합니다.



호는 정확할 수 있지만 구현하기가 상당히 어렵습니다.

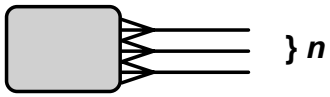
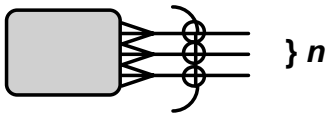
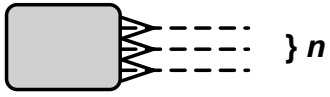
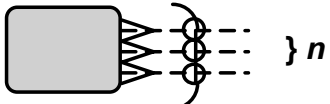
ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

잘못된 호

호를 사용하여 가능한 관계 제약 조건을 모두 포착할 수는 없습니다. 예를 들어, 세 개의 관계 중 두 개가 유효해야 한다면, 이것은 표시할 수 없습니다. 아래 표는 호가 표시할 수 있는 내용을 보여줍니다.

몇 가지 잘못된 호 구성

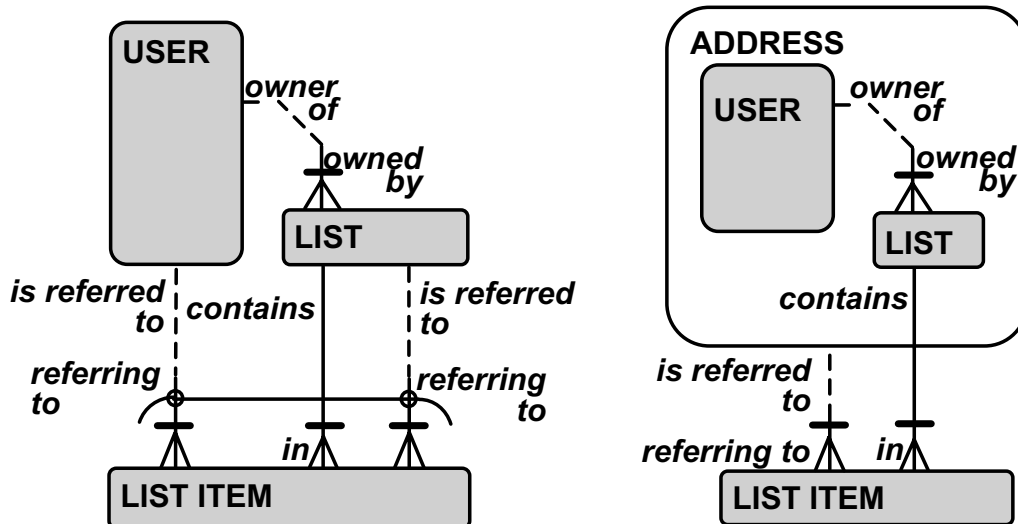
엔티티 인스턴스당 호 내 유효 관계 수	최소값	최대값
	n	n
	1	1
	0	n
	0	1

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

Oracle Internal & OAI Use Only

호 또는 하위 유형



ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

호 또는 하위 유형

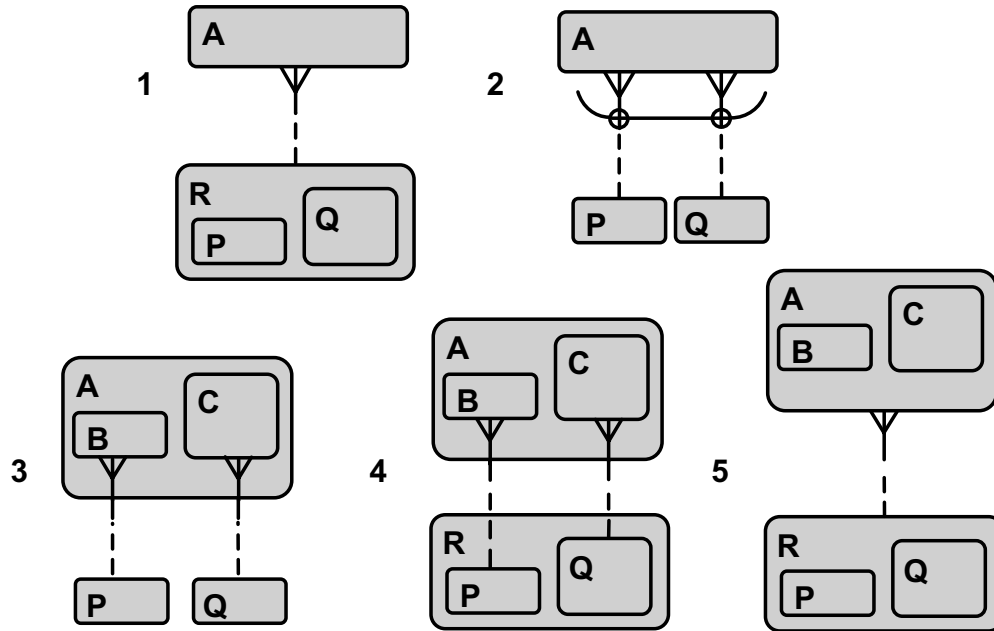
호 내에 있는 관계들은 종종 그 성질이 매우 유사하며, 빈번하게 정확히 동일한 이름을 가집니다. 이러한 경우, 위 그림에서 표시된 것과 같이 호를 하위 유형 구조로 대체할 수 있습니다. 왼쪽에는 LIST ITEM의 두 *referring to* 관계를 모두 포함하는 호가 있습니다. 오른쪽 모델에는 이제 USER와 LIST의 새로운 하위 유형 엔티티인 ADDRESS 엔티티에 연결되어 있는 단 하나의 관계만 남아 있습니다.

두 모델은 동등합니다.

왼쪽 모델은 분명히 존재하는 USER와 LIST 간의 차이점을 강조하고, 오른쪽 모델은 공통점을 강조합니다. 이 공통점은 주로 기능 문제입니다. USERS와 LISTS 모두 LIST의 일부일 수 있으며 두 가지 모두 메시지 작성 화면의 To, Cc, Bcc 필드에서 주소로 사용될 수 있습니다.

일반적으로 모든 호는 상위 유형/하위 유형 구조로 바꿀 수 있으며, 모든 상위 유형/하위 유형 구조는 호로 바꿀 수 있습니다.

호 및 하위 유형



ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

호 및 하위 유형에 대한 세부 정보

호와 하위 유형은 유사한 개념입니다. 위에 출력된 다섯 가지 모델은 모두 동일한 컨텍스트를 보여줍니다.

모델 1과 2는 앞에서 보았던 것과 동일한 모델입니다.

A의 모든 인스턴스가 P 또는 Q와 연관될 경우, P 관련 A와 Q관련 A가 있다고 말할 수 있습니다. 이러한 A의 두 가지 하위 유형은 모델 3과 같습니다.

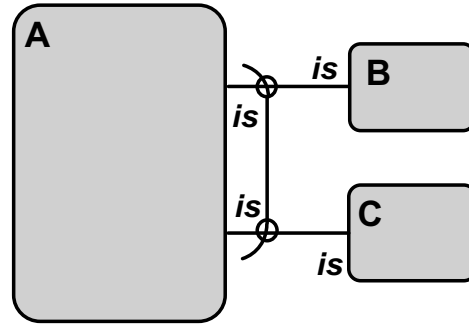
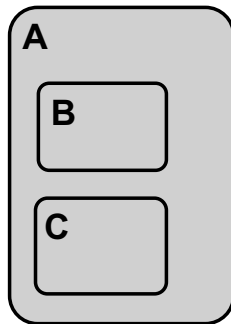
모델 4는 이것을 한 단계 넘어서서 엔티티 A의 하위 유형과 P와 Q의 하위 유형 R을 보여줍니다.

모델 3과 4는 완전히 정확하지만, 두 모델 모두 특정 항목을 두 번 모델링할 가능성이 있습니다.

모델 5만 동일한 정보를 나타내지 않습니다. 모델 3과 4에서 모델링된 것과는 다르게, 모델 5에서는 B의 인스턴스가 Q의 인스턴스와 관련이 있을 수 있습니다.

하위 유형은 호 내의 관계를 숨김

- 모든 A는 B이거나 C입니다.
- 모든 B는 A입니다.
- 모든 C는 A입니다.
- 모든 A는 B이거나 C여야 합니다.
- 모든 B는 A여야 합니다.
- 모든 C는 A여야 합니다.



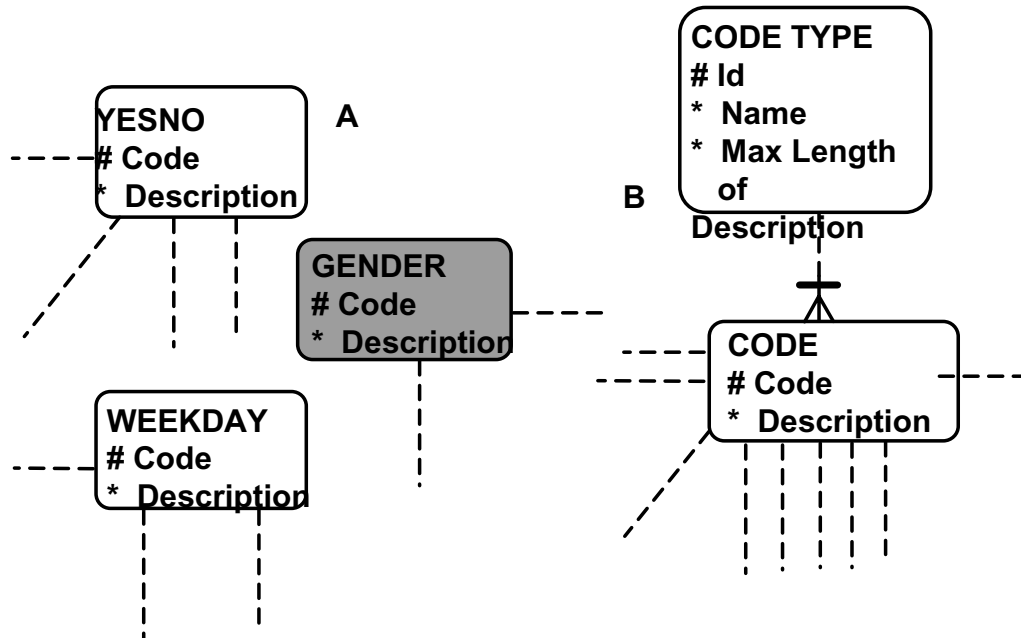
ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

숨겨진 관계

모든 하위 유형은 하위 유형과 상위 유형 간의 관계를 숨깁니다. 더욱이 다음 그림에서 알 수 있듯이 관계는 호 안에 있습니다. 두 관계 모두 필수 1:1 is/is 관계입니다.

값 집합



ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

도메인

아주 일반적인 유형의 속성 제약 조건은 속성이 가질 수 있는 가능한 값을 보여주는 일련의 값입니다. 이러한 집합을 도메인이라 합니다.

매우 일반적인 도메인의 예는 다음과 같습니다.

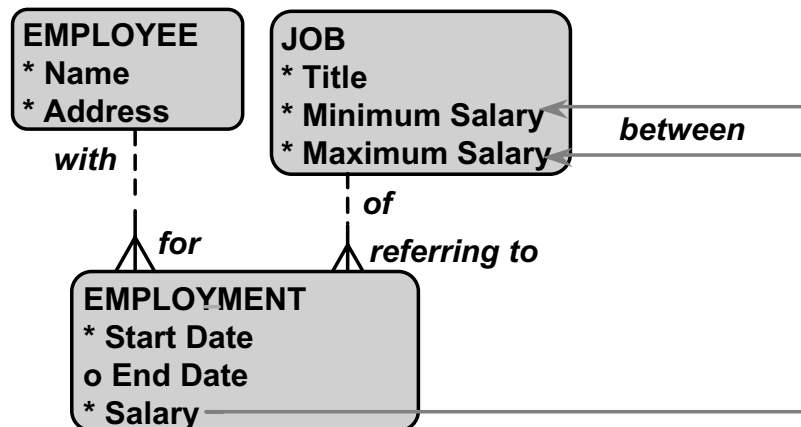
YesNo: Yes, No - **Gender:** Male, Female, Unknown - **Weekday:** Sun, Mon, Tue, Wed, Thu, Fri, Sat

개념적 데이터 모델에서는 이러한 도메인을 보통 Code와 Description이라는 두 가지 속성만 가지는 엔티티로 인식할 수 있습니다. 이러한 도메인 엔티티는 자주 참조되지만 자체적인 "여러" 관계를 가지지 않습니다. (위의 모델 A 참조) 일반적으로 사용자는 시스템이 구축되기 전에 모든 값을 알게 됩니다. 값의 숫자는 대체로 낮습니다. 종종 코드 테이블이 비어 있지 않은 시스템을 제공하게 됩니다.

(때때로 여러) 코드 엔티티에 대한 대체 모델은 좀더 일반적인 2-엔티티 접근 방식입니다. 즉, CODE와 CODE TYPE을 가지는 모델 B입니다. 모델 A는 이해하기 쉬운 엔티티일 뿐만 아니라 엔티티당 관계의 수가 적다는 장점이 있는 반면, 모델 B는 엔티티 수가 명백히 더 적으므로 테이블 수도 더 적습니다.

모든 양의 정수에서 특정 값에 이르는 수많은 값을 가지는 도메인은 일반적으로 모델링되지 않습니다. 이러한 제약 조건은 별도의 문서에 나열하고 설명해야 합니다.

기타 제약 조건: 범위 검사



ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

일부 특수 제약 조건

엔티티 관계 모델이 너무 복잡하지 않은 많은 제약 조건을 표현할 수 있다 해도, 모델링할 수 없는 제약 조건 유형은 많이 있습니다. 이러한 제약 조건은 별도 문서에 나열되고 종종 프로그램적으로 처리되어야 합니다.

범주: 예제

- **조건부 도메인:** 속성에 대한 도메인은 동일한 엔티티의 하나 이상의 속성 값에 따라 달라집니다.
- **상태 값 변환:** 속성이 변경될 수 있는 값의 집합은 해당 속성의 현재 값에 따라 달라집니다.
- **범위 검사:** 숫자 속성은 관련된 인스턴스의 속성값 사이에 있어야 합니다.
- **초기 검사:** 유효 관계는 생성 시에만 존재해야 합니다.
- **조건부 관계:** (관련 엔티티의) 속성이 특수한 값을 가질 경우 관계는 존재해야 하거나 존재하지 않을 수 있습니다.
- **상태 값에 의한 검사:** 검사는 속성에 특정 상태를 나타내는 값이 제공될 때 이루어져야 합니다.
- 또한 위의 조건들의 조합도 있습니다.

제약 조건: 사원 급여는 해당 사원의 직무 급여 범위 내에 있어야 합니다.

기타 제약 조건: 상태 값 변환

EMPLOYEE
 * Name
 * Address
 * Current Marital Status

Possible Marital Status Transitions	to fro	Sin	Mar	Wid	Div	DP
Single	m		✓			✓
Married				✓	✓	
Widowed			✓			✓
Divorced			✓			✓
Domestic Partnership		✓	✓			

ORACLE

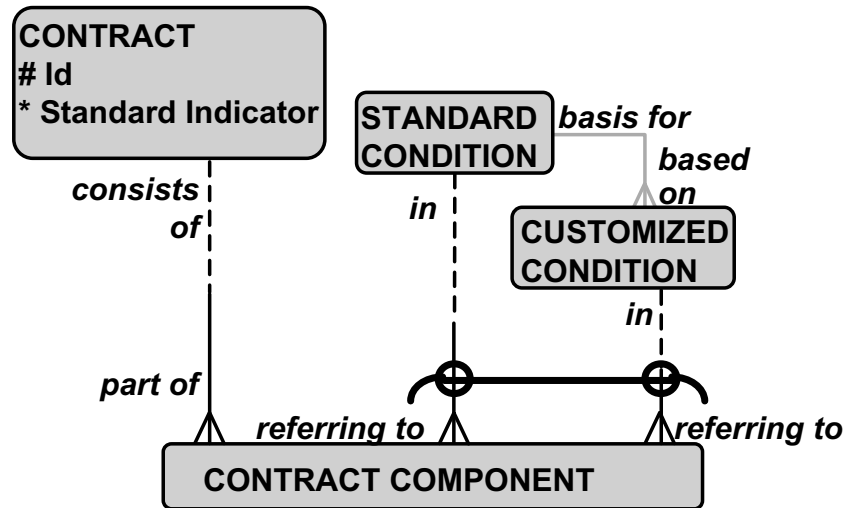
Copyright © Oracle Corporation, 2002. All rights reserved.

기타 제약 조건: 상태 값 변환

제약 조건: 사원의 결혼 여부는 임의의 값에서 기타 다른 값으로 변경할 수 없습니다.

Oracle Internal & OAI Use Only

조건부 관계



ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

조건부 관계

계약 조건: CONTRACT의 Standard Indicator가 Yes로 설정될 경우, CONTRACT COMPONENT는 CUSTOMIZED CONDITION을 참조하지 않을 수 있습니다.

파생된 속성

CONTRACT의 Standard Indicator 속성이 파생 가능하다고 주장할 수도 있습니다. 계약이 CUSTOMIZED CONDITIONS를 포함할 경우, 그 계약은 결과적으로 표준 CONTRACT가 아닙니다. 이것은 사실일 수 있으나, 반드시 그런 것은 아닙니다. 계약이 여러 단계에 걸쳐 각기 다른 권한을 가진 여러 사람에 의해 생성된다고 합니다. 그럴 경우 CONTRACT 생성은 며칠이 걸릴 수 있는 프로세스입니다. 그럴 경우 Standard Indicator는 해당 프로세스의 속성입니다. CONTRACT가 최종 승인될 경우에만 Indicator가 실제 STANDARD 및 CUSTOMIZED CONDITIONS와 일치하는지에 대한 검사가 이루어져야 합니다. 이러한 상황에서 CONTRACT 엔티티는 일반적으로 Yes로 설정될 때 검사를 트리거하는 Completed Indicator 속성을 가집니다.

조건부 관계

규칙은 속성을 도출할 수 있음

모델에서 제약 조건을 파악할 수 없을 경우, 모델 내에서 할 수 있는 최상의 방법은 제약 조건 검사 프로그램이 잘 수행되도록 좋은 모델을 만드는 것입니다. 다음 규칙을 고려하십시오.

Standard Indicator가 No로 설정될 경우, CUSTOMIZED CONDITION은 없으며 CONTRACT를 CUSTOMER에게 보낼 준비가 아직 되지 않았습니다.

이 규칙은 프로시저를 다루며 그런 식으로 모델링될 수 없지만, Ready To Send 상태와 같은 것을 나타내기 위해 CONTRACT 엔티티에서 표시자를 호출합니다.

개요 모델

분석가는 종종 모델링될 수 없어 별도로 문서화해야 하는 제약 조건에 직면하게 됩니다. 이것은 모델의 약점이 아닙니다. 도표의 중요한 목표는 전체적인 그림을 제공하는 것이지만 모든 세부 내용을 보여주는 것이 아닙니다. 모델을 통해 사용자가 주요 영역을 명확히 볼 수 있어야 합니다.

Oracle Internal & OAI Use Only

경계

unrelated entity

EXTERNAL
Id
* Description
* Value

and possible implementation

EXTERNALS

Id	Description	Value
1	Value added tax %	15
2	Maximum available Space per Mail User in Mbyte	500
3	Maximum level of Nested Mail Folders	3
4	Maximum level of Nested Mail Lists	16

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

경계

제약 조건 또는 특수 규칙을 두 번 이상 검사하기 위해서는 모델에 있는 엔티티 중 하나와 직접 관련되지 않는 정보를 사용해야 합니다.

전형적인 예는 모회사나 국가 법률과 같은 외부 소스에 의해 설정된 규칙과 경계입니다.

타당성 있게 가능한 경우, 이러한 규칙은 개념 데이터 모델의 일부이어야 하며, 프로그램에서 하드 코딩되지 않아야 합니다. 그 이유는 명확합니다. 자신의 권한 밖에 있는 규칙이 변경될 경우, 프로그램을 변경할 필요가 없습니다. 테이블 값의 갱신만 필요합니다. 완전한 모델을 개발하는 데 소요되는 시간은 프로그래밍 시간 절감으로 온전히 정당화됩니다.

요약

- **식별**
 - 실제 세계에서는 실제 문제가 될 수 있습니다.
 - 모델은 이것을 극복할 수 없습니다.
- **엔티티는 적어도 한 개의 고유 식별자를 가져야 합니다.**
- **고유 식별자는 속성 또는 관계 또는 이 두 가지 모두로 구성됩니다.**
- **호**
- **ER 모델에는 많은 유형의 제약 조건이 표현되지 않습니다.**

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

요약

실제 세계의 엔티티는 데이터베이스에 표현되기 전에 개별적으로 식별되어야 합니다. 그렇지 않으면 무슨 이야기인지 알지 못할 것입니다. 사람이나 그림과 같이 어떤 엔티티는 정말로 식별이 어렵습니다. 어떤 엔티티는 좀더 식별이 쉬우며 엔티티가 도메인의 일부일 경우에 특히 그러합니다. 예를 들어, 고객에게 보내는 각 송장의 고유 번호와 같이 규칙을 만들 수 있는 경우입니다. 몇몇 고유 식별자는 이미 실제 세계에 존재하며, 종종 엔티티의 속성과 관계의 조합으로 표현됩니다.

도표에서 호는 하나의 엔티티의 관계들에 대한 특정 유형의 제약 조건을 나타냅니다.

많은 업무 제약 조건은 한 도표에 표현될 수 없으므로 별도로 나열해야 합니다. 이렇게 해야 모델은 명확하고 그래픽 요소로 가득차지 않게 됩니다.

연습

- 식별 방법
- 식별
- Moonlight UID
- 테이블
- 모델링 제약 조건

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

Oracle Internal & OAI Use Only

연습: 식별 방법

- 도시
- 고객 담당자
- 기차
- 도로
- 재무 트랜잭션
- 아카데미상(오스카)
- 그림
- TV 프로그램

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

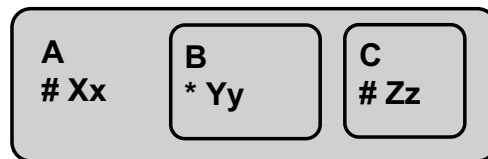
연습 4-1: 식별 방법

과제

적합하다고 간주되는 속성과 관계를 구성하여 다음 엔티티를 식별할 방법을 설명하십시오.

Oracle Internal & OAI Use Only

연습: 식별 1



ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

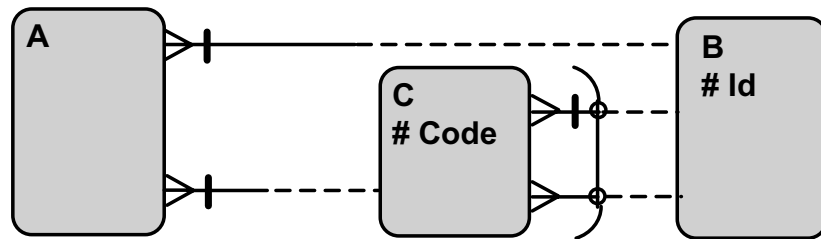
연습 4-2: 식별

과제

다음 도표의 엔티티는 식별 가능합니까?

Oracle Internal & OAI Use Only

연습: 식별 2

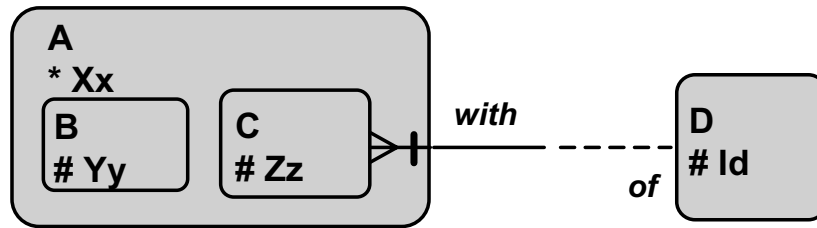


ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

Oracle Internal & OAI Use Only

연습: 식별 3

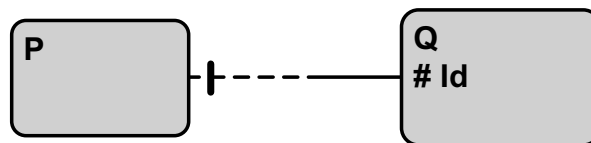


ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

Oracle Internal & OAI Use Only

연습: 식별 4

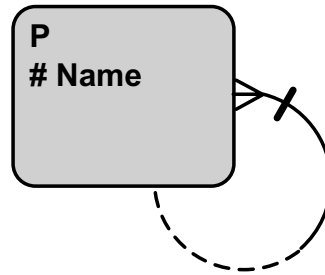


ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

Oracle Internal & OAI Use Only

연습: 식별 5



ORACLE

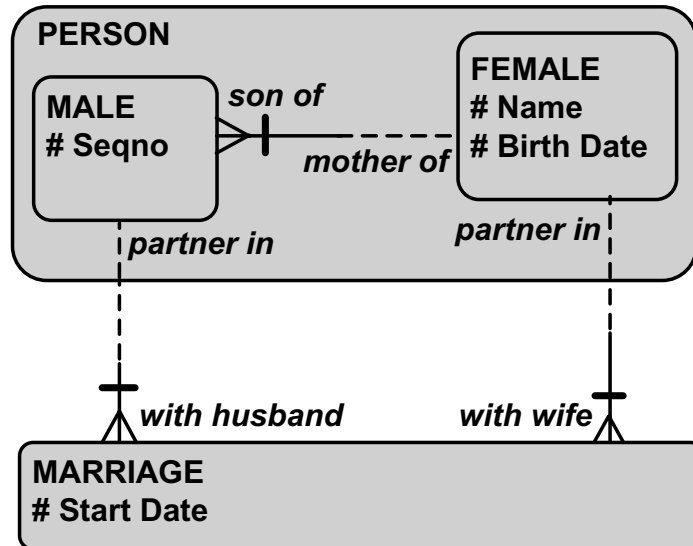
Copyright © Oracle Corporation, 2002. All rights reserved.

연습 4-2: 식별

주: 다음 모델은 사용자에게 익숙하지 않을 수 있는 상황을 설명합니다.

Oracle Internal & OAI Use Only

연습: 식별 6



ORACLE

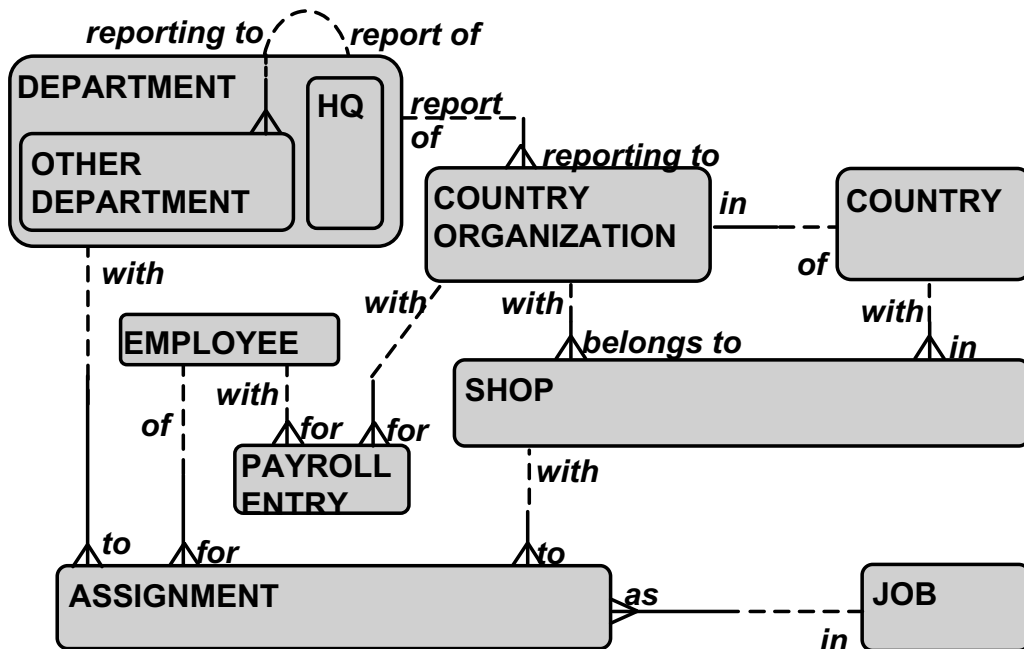
Copyright © Oracle Corporation, 2002. All rights reserved.

연습 4-2: 식별

위 모델을 전제로 다음 질문에 답하십시오.

1. A라는 사람은 두 번 결혼할 수 있습니까?
2. A라는 사람은 같은 날 두 번 결혼할 수 있습니까?
3. A라는 사람은 B라는 사람과 두 번 결혼할 수 있습니까?
4. A라는 사람은 B라는 사람과 같은 날 두 번 결혼할 수 있습니까?
5. A라는 사람은 B라는 사람 및 C라는 사람과 동시에 결혼할 수 있습니까?
6. A라는 사람은 A라는 사람과 결혼할 수 있습니까?

연습: Moonlight UID



ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

연습 4-3: Moonlight UID

목표

이 연습의 목적은 제공된 엔티티에 대한 UID를 정의하는 것입니다.

시나리오

Moonlight Coffees, 조직 모델

과제

지금까지 Moonlight Coffees에 대해 아는 지식을 활용하고 무엇보다 상상력을 동원하십시오.

1. 위 모델을 전제로 다양한 엔티티에 대한 UID를 확인합니다. 적합하다고 간주되는 속성을 모두 추가합니다. 국가 본점은 해당 국가에서 고유한 "사업자 등록 번호"를 가집니다.
2. 누락된 호가 있습니까?

연습: 테이블 1

- 관계형 데이터베이스 시스템에서 데이터는 테이블에 저장됩니다. 데이터베이스 사용자의 테이블은 고유한 이름을 가져야 하며, 적어도 한 개의 열을 가져야 합니다. 열은 테이블 내에 고유한 이름을 가집니다. 열은 데이터 유형을 가져야 하며 널이 아닐 수도 있습니다.
- 테이블은 한 개의 기본 키와 원하는 개수의 고유 키를 가질 수 있습니다. 키는 하나 이상의 테이블 열을 포함합니다. 열은 둘 이상의 키의 일부일 수 있습니다.

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

연습 4-4: 테이블

목표

이 연습의 목적은 제공 컨텍스트와 ER 모델을 대응시키는 것입니다.

과제

ISO 관계형 테이블의 텍스트를 읽어 보십시오.

따옴표로 묶인 텍스트와 이 주제에 대해 본인이 알고 있는 내용을 기준으로 ER 모델의 품질을 확인하십시오. 또한 텍스트에 언급되어 있지만 모델링되지 않은 제약 조건을 나열하십시오.

연습: 테이블 1

하나의 테이블은 여러 개의 외래 키를 가질 수 있습니다. 하나의 외래 키는 항상 하나의 테이블을 또 다른 테이블과 연결합니다. 하나의 외래 키는 다른 테이블의 키 열을 참조하는 한 테이블의 하나 이상의 열로 구성됩니다.

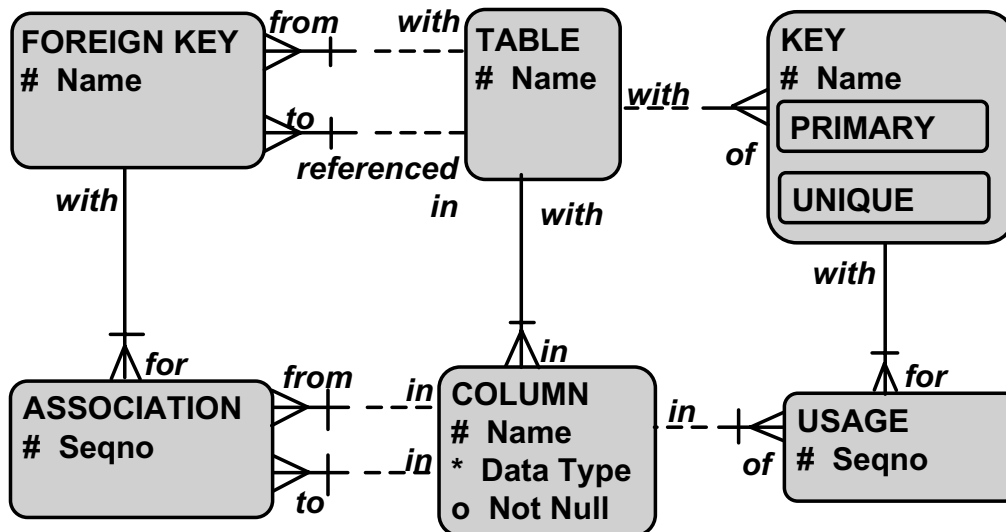
"키와 외래 키 내의 열 순서는 중요합니다."

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

Oracle Internal & OAI Use Only

연습: 테이블 2

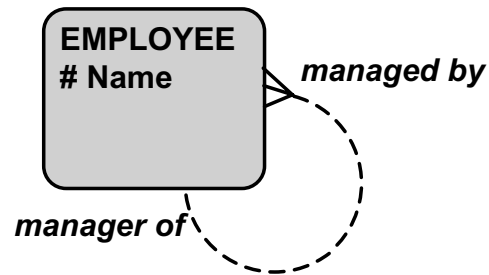


ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

Oracle Internal & OAI Use Only

연습: 제약 조건 1



ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

연습 4-5: 모델링 제약 조건

목표

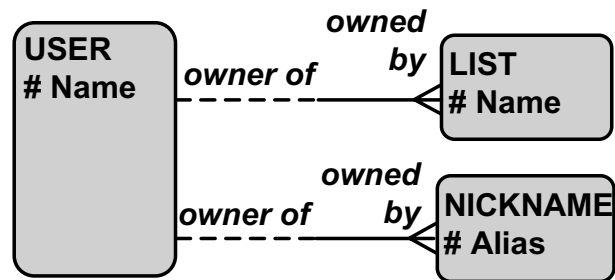
이 연습의 목적은 어떤 제약 조건이 어떻게 모델링될 수 있으며, 어떤 제약 조건은 모델링될 수 없는지에 대해 학습하는 것입니다.

과제

도표를 변경하여 제공된 제약 조건을 모델링하십시오.

1. CEO(최고 경영권자)를 제외한 모든 EMPLOYEE에게는 매니저가 있어야 합니다.

연습: 제약 조건 2



ORACLE

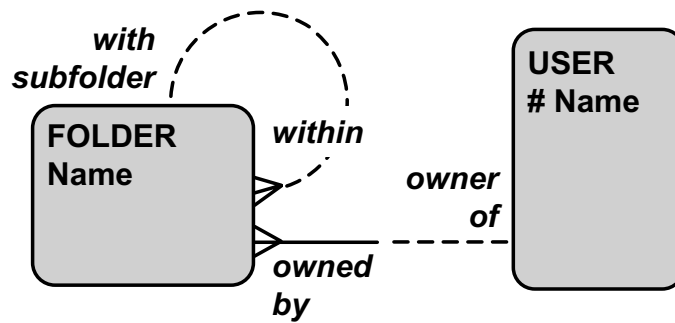
Copyright © Oracle Corporation, 2002. All rights reserved.

연습 4-5: 모델링 제약 조건

2. 사용자는 NICKNAME 및 LIST 이름 둘 다에 대해 동일한 이름을 사용하지 않아도 됩니다.

Oracle Internal & OAI Use Only

연습: 제약 조건 3



ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

연습 4-5: 모델링 제약 조건

3. 최상위 레벨의 FOLDER는 사용자별로 고유한 이름을 가져야 합니다. 하위 폴더는 해당 폴더가 위치하는 폴더 내에서 고유한 이름을 가져야 합니다.

5

모델링 변경

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

Oracle Internal & OAI Use Only

개요

- 날짜 및 시간
- 시간 경과에 따른 모델링 변경
- 가격 변동
- Journaling(저널링)

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

소개

모든 속성 갱신 또는 관계 전이는 정보의 유실을 의미합니다. 대개 유실된 정보는 더 이상 사용되지 않지만, 일부 시스템에서는 이전 속성값의 일부 또는 전부를 계속 추적해야 합니다. 이것은 모델에서 명시적인 시간 차원으로 이어질 수 있는 꽤 복잡한 문제입니다.

많은 엔티티는 사실 이벤트를 표현하는 것이기 때문에 시간은 종종 업무적인 상황에 존재합니다. 이 단원에서는 엔티티 모델에 시간을 통합할 때 제기되는 가능성과 어려움에 대해 설명합니다.

목표

이 단원을 마치면 다음을 수행할 수 있습니다.

- DATE 엔티티 또는 Date 속성 사용에 대한 심사 숙고한 결정
- 수명 주기 속성을 필요로 하는 모든 엔티티에 수명 주기 속성 모델링
- 시간 차원 사용으로 제기되는 모든 제약 조건 나열
- 저널링 처리

변경 및 시간

- 모든 갱신은 정보의 유실을 의미합니다.
- 모델에서의 시간은 모델을 더욱 복잡하게 만듭니다.
- 종종 복합 조인 조건이 있습니다.
- 사용자가 미리 작업을 수행할 수 있습니다.
- 언제 날짜/시간을 엔티티로 모델링하겠습니까?
- 어떤 제약 조건이 제기됩니까?
- 저널링을 어떻게 처리하겠습니까?

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

모델링 시간

여러 모델에서 시간은 한 가지 역할을 합니다. 본질적으로 이벤트인 엔티티(예: PURCHASE, ASSIGNMENT)는 대개 모델의 일부입니다. 사용자가 이러한 엔티티에 대해 기록하는 속성 중 하나는 이벤트의 날짜 또는 날짜와 시간입니다. 대개 날짜와 시간은 고유 식별자의 일부입니다.

두번째 시간 관련 문제는 종종 시스템의 유용성을 극적으로 증대시키는 데 도움을 줍니다. 시스템의 데이터에 시작, 만기, 종료 날짜와 같은 날짜를 추가하여 사용자가 미리 작업을 수행할 수 있도록 할 수 있습니다. 특정한 값 예를 들어, 가스 또는 디젤 가격이 1월 1일부로 변경될 예정이라고 합시다. 이 경우, 새해 전날이 되기 훨씬 전에 시스템에 새로운 값을 알려줄 수 있다면 매우 유용합니다. 모델에 시간 차원을 추가하면 시스템이 현재에만 국한되지 않게 됩니다.

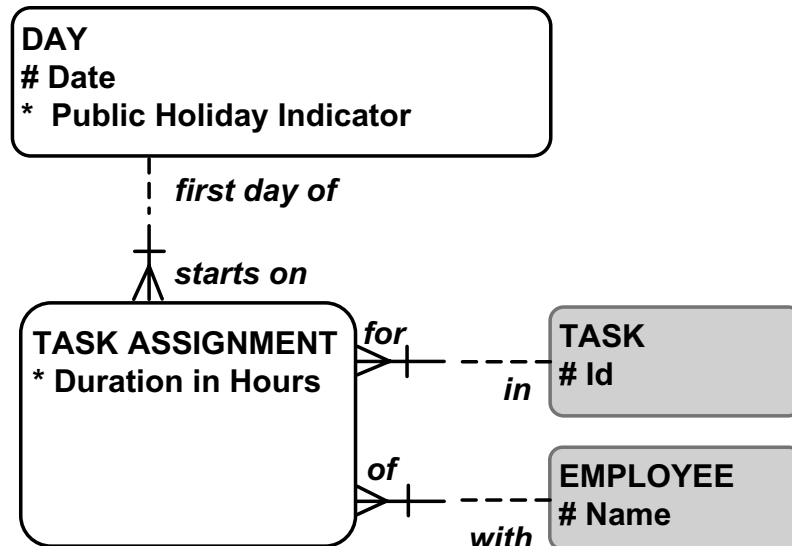
항상 그렇듯이, 이처럼 무언가를 추가하는 데에는 대가가 따릅니다. 개념적 데이터 모델에 시간 차원을 추가하면 모델이 상당히 더 복잡해집니다. 특히 검사해야 할 제약 조건과 업무 규칙의 수가 증가하게 됩니다.

개념적 데이터 모델의 세번째 시간 관련 문제는 로깅 또는 저널링의 개념과 연관이 있습니다. 값 갱신을 허용하되 이전 값 중 일부를 계속 추적하고자 한다고 합시다. 즉, 속성값, 관계, 전체 엔티티의 내역에 대한 기록을 보유해야 할 때 어떻게 하겠습니까?

다음과 같은 7문제가 제기됩니다.

- 언제 날짜/시간을 엔티티로 모델링하고, 언제 속성으로 모델링하겠습니까?
- 시간 관련 데이터를 처리하는 시스템에서 제기되는 제약 조건은 어떻게 처리하겠습니까?
- 저널링을 어떻게 처리하겠습니까?

DAY 엔티티



ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

DAY 엔티티

시스템은 날짜와 씨름하는 과거 정보를 다루는 것뿐만이 아닙니다. 때때로 시스템은 시스템 날짜로부터 파생될 수 있는 요일에 대해 더 많이 알아야 합니다. 예를 들어, 계획 시스템에서는 특정 요일이 공휴일인지 파악해야 합니다. 많은 데이터 웨어하우스 시스템에서는 보통 달력과 다른 달력을 사용합니다. 예를 들면, 일년이 4주 기간 또는 30일로 구성된 월 또는 1분기가 5월 중순에 시작하는 분기로 구분되는 달력을 사용합니다.

어떤 창고에서는 날짜가 판매에 미치는 영향에 대한 통계 분석을 수행하기 위해 특정일에 대한 날짜 정보가 필요합니다. 이러한 경우 일(day)은 자체 속성 또는 관계를 가지며 DAY 엔티티로 모델링되어야 합니다.

위 모델은 작업이 사원에게 할당된 계획 시스템의 일부를 나타낸 것입니다. 작업은 두세 시간에서 많게는 며칠이 걸릴 수도 있습니다.

이 모델을 기준으로 TASK_ASSIGNMENTS 테이블은 DAYS 테이블에 대한 외래 키 열인 Date 열을 포함하게 됩니다.

시간 경과에 따른 모델링 일 수

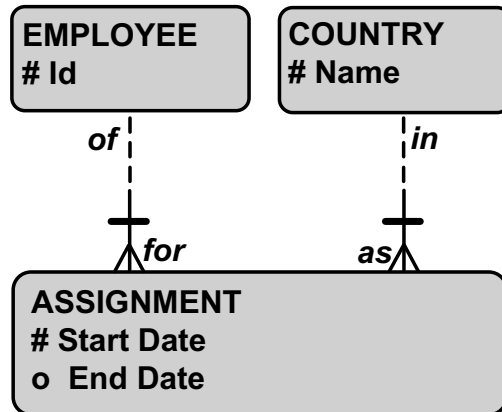
날짜 및 시간

앞에서 언급했듯이, Oracle DATE 열은 항상 날짜와 시간을 포함합니다. 두 개의 DATE 열은 분명히 동일한 날짜를 포함할 수도 있지만 시간 구성 요소가 다르기 때문에 동일하지 않습니다. 따라서 여기에는 특별한 주의가 필요합니다.

예를 들면, 모델링 작업 시 **DateTime** 속성에 이름을 지정하여 하루 중 언제 문제가 되는지를 항상 명시적으로 분명히 밝혀야 합니다. 시와 분이 각자의 역할을 하는 순간, "시간대"와 "일광 절약 시간"의 개념이 중요해질 수 있습니다.

Oracle Internal & OAI Use Only

모델링 변경



ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

시간 경과에 따른 모델링 변경

다음 예제에서 알 수 있듯이, 모델에서 날짜와 시간은 시스템의 복잡성을 상당히 증가시킬 수 있습니다.

이 예제의 상황은 대사관 정보 시스템(Embassy Information System)의 것이지만, 거의 모든 업무 영역에서 이러한 예제의 상황을 찾아볼 수 있을 것입니다.

대사관 직원들은 국가를 위한 업무를 담당하고 있지만, 물론 그러한 업무는 시간이 경과하면서 변경될 수 있습니다. 그러므로 모델에는 Start Date 필수 속성과 End Date 선택 속성을 가진 ASSIGNMENT 엔티티가 필요합니다. Start Date는 ASSIGNMENT에 대한 UID의 일부로 모델링됩니다. 이는 직원이 다른 날에 업무를 시작하는 한, 이 모델에서는 직원이 동일한 국가에서 두 가지 업무를 담당할 수 있음을 의미합니다. 이 모델에서는 또한 다른 국가에 대한 업무라면 직원이 동일한 날에 시작하는 두 개의 업무를 담당할 수 있습니다.

Jacqueline이 다음 달 1일에 칠레에서 모로코로 전근할 것이라는 사실을 오늘 알았다고 가정합니다. 이 사실은 생성 시간이 아직 미래인 Start Date를 가진 새 ASSIGNMENT 인스턴스를 생성함으로써 즉시 시스템에 공급될 수 있습니다. 미래의 사용자는 이러한 종류의 기능에 대해 감사할 것입니다.

시간 경과에 따른 모델링 변경

End Date 중복

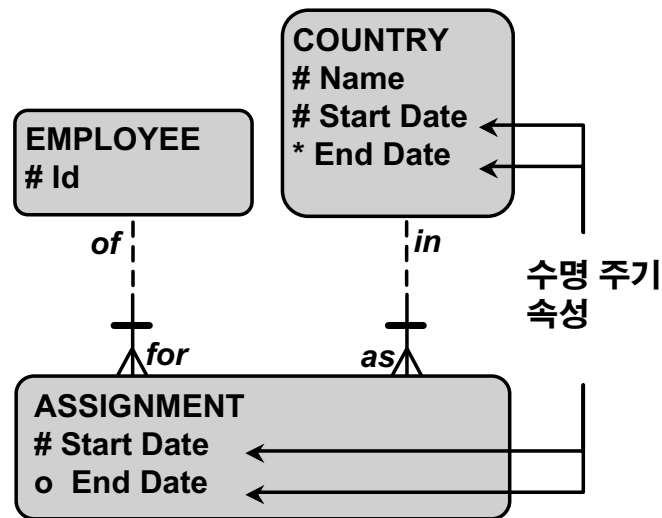
Jacqueline의 업무는 바로 이어지기 때문에 ASSIGNMENT의 End Date 속성이 중복된다고 주장할 수도 있습니다. 즉, Jacqueline의 칠레 업무 End Date와 모로코 업무 Start Date가 일치한다는 것입니다. 이것은 사실일 수 있지만, 대사관 직원이 휴가를 떠나 이삼 년 후에 돌아올 수도 있다는 점을 고려하지 않은 것입니다. 즉, End Date 속성을 모델링하지 않을 경우, 한 사람의 지정된 기간이 지속적이지 않을 가능성을 무시하게 됩니다.

이 모델은 직원이 예를 들어, 온두라스에서 서로 겹치는 두 가지 업무를 담당할 수 있도록 허용합니다! 고유 식별자는 중첩되는 기간에 대한 데이터를 보호하지 않습니다. UID에 End Date를 추가해도 도움이 되지 않습니다.

이 문제를 해결하려면 또 다른 수많은 제약 조건이 필요합니다.

Oracle Internal & OAI Use Only

국가에도 수명 주기가 있습니다.



ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

시간 경과에 따른 모델링 변경

국가에도 수명 주기가 있습니다.

대사관 정보 시스템이 최소한 80년대 말까지 거슬러 올라가는 데이터를 포함한다고 합시다. 그 당시에는 소련과 자이레가 국가였습니다. 소련과 자이레를 참조하는 ASSIGNMENT가 있다고 가정합시다. 자이레의 경우, COUNTRY의 Name 갱신을 고려할 수 있을 것입니다. 콩고 민주 공화국은 본질적으로 자이레의 새 이름일 뿐입니다. 소련의 경우에는 그렇지 않습니다. 옛 국가에 대한 새 이름이 없습니다. 이 옛 국가는 여러 나라로 분리되면서 더 이상 존재하지 않게 되었습니다. 국가의 개념이 매우 안정된 것처럼 보이지만, 정보 시스템 시대에 국가는 근본적으로 변화를 겪을 수 있습니다.

이것은 다음 모델을 도출합니다.

시간 경과에 따른 모델링 변경

시간 관련 제약 조건

시간 차원에서 비롯되는 제약 조건은 많습니다! 다음은 그 중 일부입니다.

- ASSIGNMENT는 ASSIGNMENT의 Start Date에서 유효한 COUNTRY만 참조할 수 있습니다.
- 분명한 것: End Date는 Start Date 이후여야 합니다.
- 업무 규칙: ASSIGNMENT 기간은 중복되어서는 안 됩니다. EMPLOYEE에 대한 ASSIGNMENT의 Start Date는 동일한 EMPLOYEE에 대한 다른 ASSIGNMENT의 Start Date와 End Date 사이에 있지 않을 수 있습니다.
- End Date에 대한 경우도 앞의 제약 조건과 마찬가지로입니다.
- ASSIGNMENT가 아직 시작되지 않은 한 즉, ASSIGNMENT의 Start Date가 아직 미래일 경우, 사용자는 아마도 ASSIGNMENT가 다른 COUNTRY로 전이되는 것을 허용하지 않을 것입니다.
- 이것은 **조건부 비전이성**에 대한 예입니다.

Start Date 속성의 갱신에 대해 다음과 같은 몇 가지 가능한 제약 조건이 있습니다.

- **ASSIGNMENT의 Start Date를 나중 날짜로 갱신할 수 있습니다.** 단, 이 날짜는 참조하는 COUNTRY의 End Date(있을 경우)보다 이전이어야 합니다.
- 현재 Start Date가 여전히 미래일 경우, ASSIGNMENT의 Start Date를 나중 날짜로 갱신할 수 있습니다.
- ASSIGNMENT의 Start Date를 더 이른 날짜로 갱신할 수 있습니다. 단, 이 날짜는 참조하는 COUNTRY의 Start Date보다 이후여야 합니다.
- 새 Start Date가 여전히 미래일 경우, ASSIGNMENT의 Start Date를 더 이른 날짜로 갱신할 수 있습니다.
- COUNTRY의 Start Date는 ASSIGNMENT가 끊기지 않을 경우 나중 날짜로 갱신될 수 있습니다.

유사한 제약 조건이 End Date 속성에 적용됩니다.

참조 논리

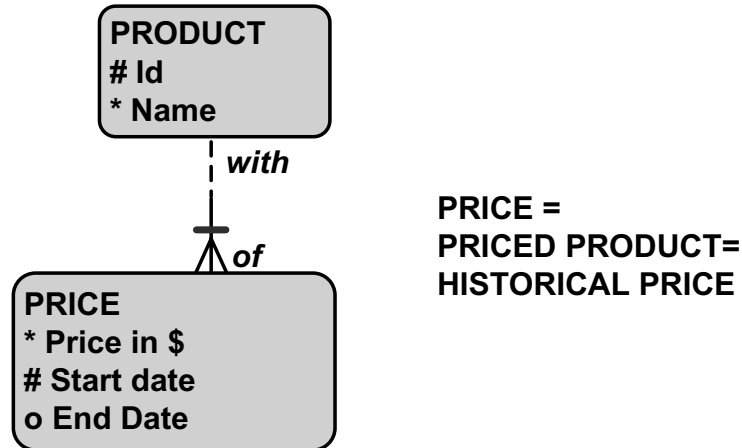
두 가지를 제외하고 이러한 제약 조건은 참조 논리에서만 비롯된 것입니다. 추가적인 업무 제약 조건이 더 많이 있을 수도 있습니다.

시간의 영향을 받는 엔티티가 시간에 영향을 받는 다른 여러 엔티티와 관련될 경우, 무수한 제약 조건을 상상해 보십시오! 다행히 이러한 제약 조건은 모두 유사한 패턴을 가집니다. 즉, 이들은 시간과 관련된 참조 논리에서 비롯됩니다.

구현

Oracle 환경에서 이러한 제약 조건 중 하나는 검사 제약 조건으로 구현될 수 있습니다. (End Date는 Start Date보다 이후여야 합니다.) 나머지는 모두 데이터베이스 트리거로 구현됩니다.

제품 및 가격



ORACLE

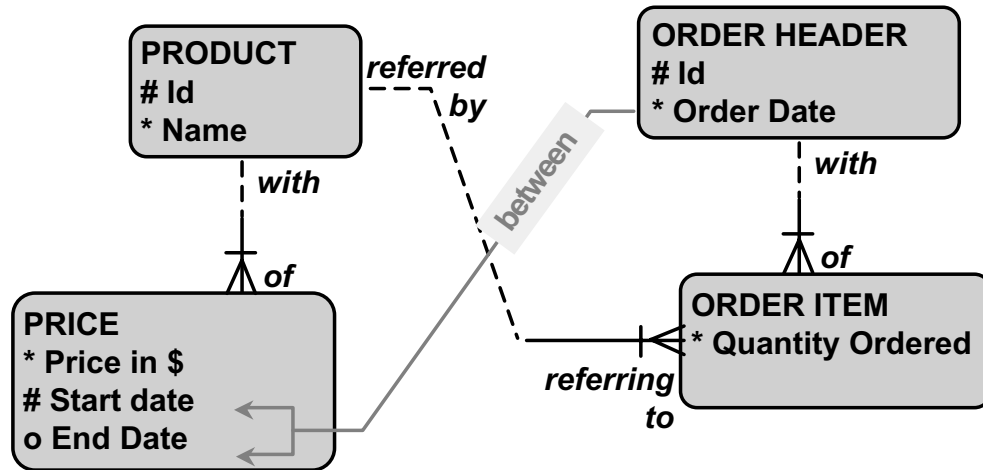
Copyright © Oracle Corporation, 2002. All rights reserved.

시간 예제: 가격

제품에는 가격이 있습니다. 가격은 변합니다. 이전 가격에 관심이 있을 수도 있습니다. 이로 인해 **PRODUCT** 엔티티와 **PRICE** 엔티티를 가진 모델이 도출됩니다. **PRICE** 엔티티는 가격과 해당되는 기간을 포함합니다. 실제 상황에서는 **PRICED PRODUCT**, **HISTORICAL PRICE** (그리고 덜 적합한 **price list** 또는 **price history**)라고도 하는 **PRICE** 개념을 발견하게 됩니다. 즉, 이 모든 이름은 다소간 이 개념을 설명해줍니다.

End Date 속성이 필요하다고 주장할 수도 있습니다. 한 가지 제품 가격의 다양한 기간이 연속적일 경우에는 **End Date**가 필요하지 않습니다. 반면에 과일 및 야채 시장처럼 제품이 항상 구매 가능한 것이 아닐 경우에는 기간에 명시적인 **End Date**가 있어야 합니다.

지불할 가격은 얼마입니까?



ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

시간 예제: 가격

Order Header 및 Order Item 소개

여기에서는 ORDER HEADER 엔티티와 ORDER ITEM 엔티티에 대해 소개합니다. ORDER HEADER는 Order Date, 주문을 낸 CUSTOMER 또는 주문을 처리한 EMPLOYEE와의 관계와 같이 모든 항목에 적용되는 정보를 포함합니다. (명확성을 위해 이러한 관계는 여기에 그려져 있지 않습니다.) ORDER ITEM은 Quantity Ordered를 포함하고 주문된 PRODUCT를 참조합니다. 지불되어야 하는 가격은 PRICE의 Start Date와 End Date 사이에 있는 Order Date와 대응시켜 찾을 수 있습니다. 이러한 "between relationship"은 모델링할 수 없습니다.

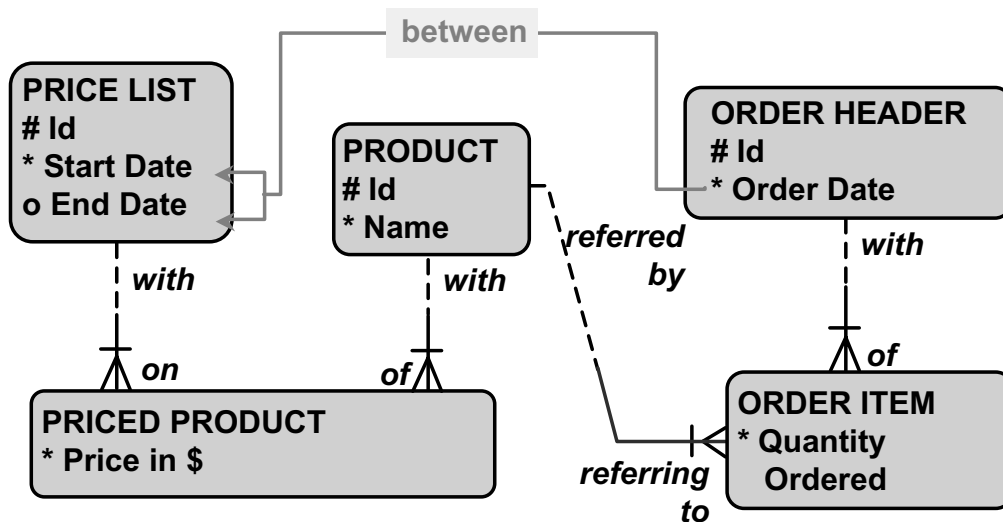
이 모델은 상당히 간단한 제품 가격 책정 모델로, 자주 사용됩니다.

주문

이 모델에서 주문의 개념은 ORDER HEADER와 ORDER ITEM으로 구성됩니다.

주문에 대한 주문 총액을 찾으려면 네 테이블에 대한 조인이 필요합니다.

가격 목록 검색



ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

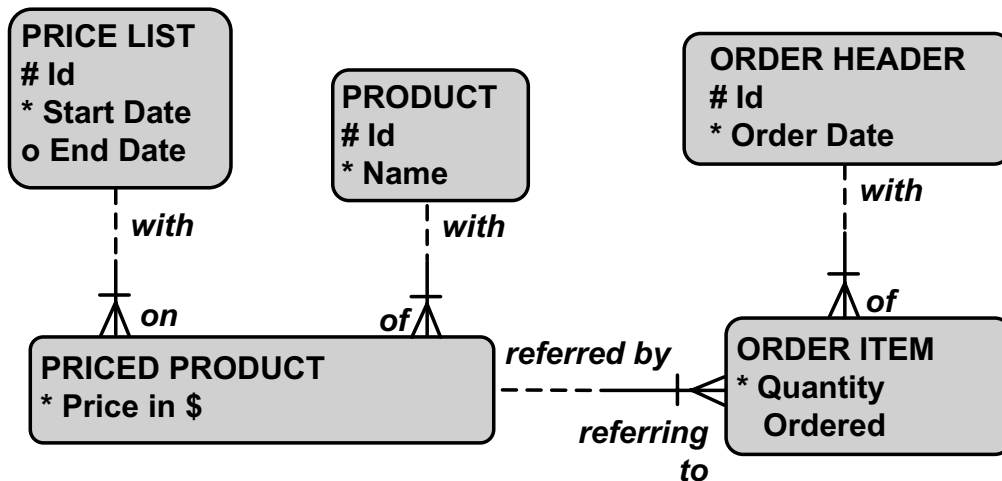
시간 예제: 가격

가격 목록

위 모델의 한 가지 변형은 하나의 그룹으로서 여러 가격이 일반적으로 동시에 변경될 때 종종 사용됩니다. 가격이 유효한 기간은 여러 가격에 대해 동일하며, 그럴 경우 위와 같은 모델이 도출됩니다.

PRICE LIST 엔티티는 다양한 제품에 대한 가격 세트를 표시하며, PRICED PRODUCT는 가격 목록 품목을 표시합니다. 주문된 품목에 대해 지불된 가격을 알려면, ORDER HEADER의 Order Date를 취하고, 해당 날짜에 적용 가능한 PRICE LIST를 취합니다. 그런 다음 ORDER ITEM에서 참조되는 PRODUCT로 이동하고, 거기에서 방금 찾은 PRICE LIST의 PRICED PRODUCT로 이동합니다. 주문에 대한 주문 총액을 찾으려면 다섯 테이블에 대한 조인이 필요합니다.

-priced Product에 대한 주문



ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

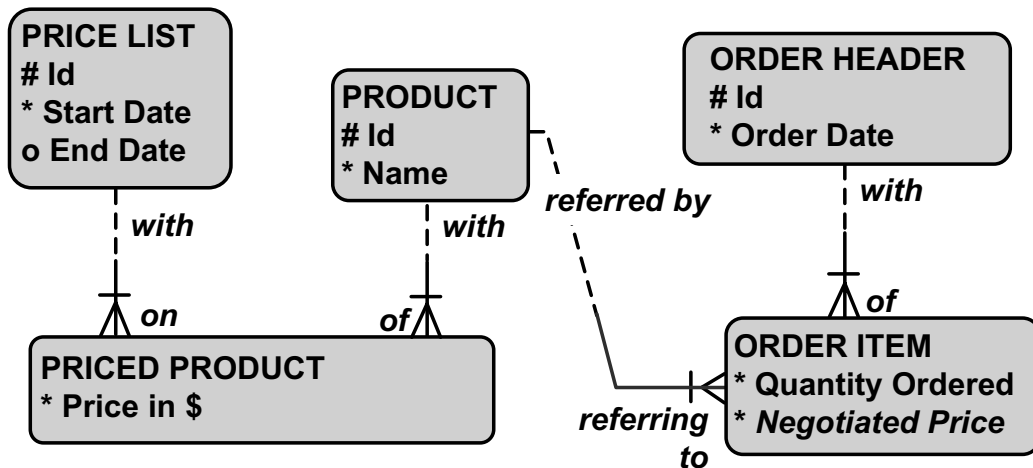
시간 예제: 가격

PRODUCT 또는 PRICED PRODUCT 구매

가격 책정 모델의 또 다른 변형이 여기에 제시됩니다.

여기에서 ORDER ITEM은 PRICED PRODUCT를 직접 참조합니다. ORDER ITEM 생성 시, Order Date는 정확한 PRICE LIST 기간과 일치해야 한다는 제약 조건이 적용됩니다. 주문에 대한 주문 총액을 찾으려면 이제 세 개의 테이블만 필요합니다.

협상 가격



ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

시간 예제: 가격

협상 가격

가격이 협상되어야 할 경우 모델은 더 단순해집니다. Negotiated Price는 이제 ORDER ITEM 엔티티의 속성이며, ORDER ITEM은 PRODUCT를 참조합니다. 모든 참조 제약 조건은 모델링할 수 있습니다.

이 모델은 과생 가능한 정보를 보유하는 것처럼 보일 수도 있지만, 이것은 사실이 아닙니다. 심지어 거의 모든 Negotiated Price가 현재 제품 가격과 같을 경우에도 예외가 발생할 수 있는 약간의 가능성 때문에 Negotiated Price를 ORDER ITEM 레벨에서 모델링해야 합니다. 주문 총액을 찾으려면 두 개의 테이블만 필요합니다. 현재 협상할 것이 전혀 없는 경우라도 많은 분석가들은 이 변형된 형태의 모델을 안전한 것으로 선택한다고 생각할 수 있습니다.

시간 예제: 가격

어느 변형을 언제 사용할 것인가?

일반적으로 협상 가격을 가지는 모델은 ORDER HEADER당 ORDER ITEM의 수가 낮을 경우, 종종 단 하나일 경우, 그리고 예를 들어 중고차 사업 상황에서처럼 그 값이 높을 때 발생하게 됩니다.

PRODUCT를 참조하는 ORDER ITEM은 가격 변동이 빈번하지 않은 상황에서 가장 자주 나타납니다. ORDER HEADER당 품목 수는 대개 한 개 이상이며, 전체 값은 제한됩니다. 전형적인 예제는 패션 산업과 식료품점입니다.

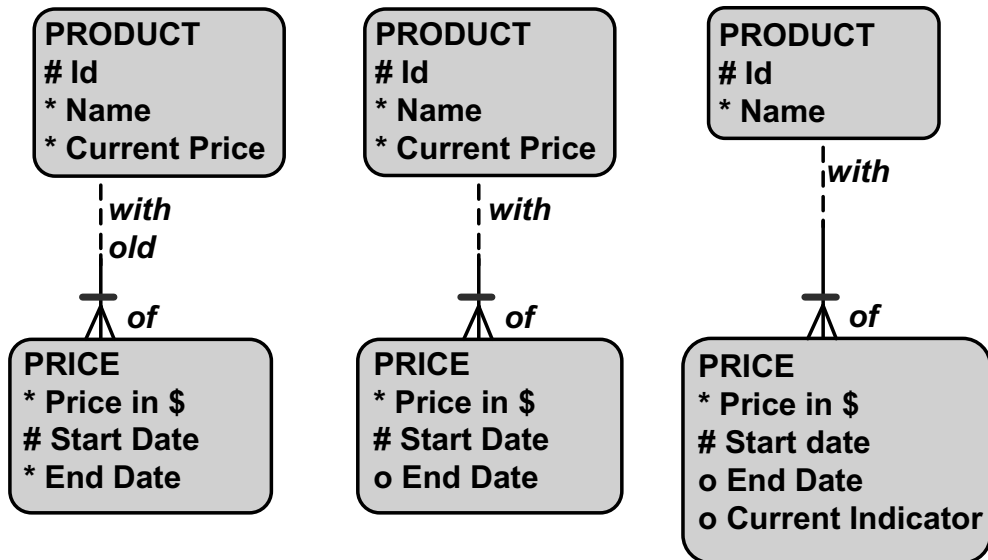
PRICED PRODUCT를 참조하는 ORDER ITEM을 가지는 모델은 신선한 과일 및 야채 시장에 서처럼 가격 변동이 빈번한 사업에 자주 사용됩니다. 이러한 곳의 가격은 심지어 하루 중에도 변경될 수 있습니다.

PRODUCT에 대한 Current Price 속성을 가지는 모델은 계산대에서 즉시 가격을 알 수 있는 것이 무엇보다 중요한 슈퍼마켓 환경에서 일반적인 모델입니다.

앞에서 언급했듯이, 특정 상황에 맞는 최상의 모델은 기능상의 필요에 따라 달라집니다. 이에 대한 자세한 내용은 비정규화 데이터 및 설계 고려사항에 대한 장을 참조하십시오.

Oracle Internal & OAI Use Only

현재 가격



ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

현재 가격

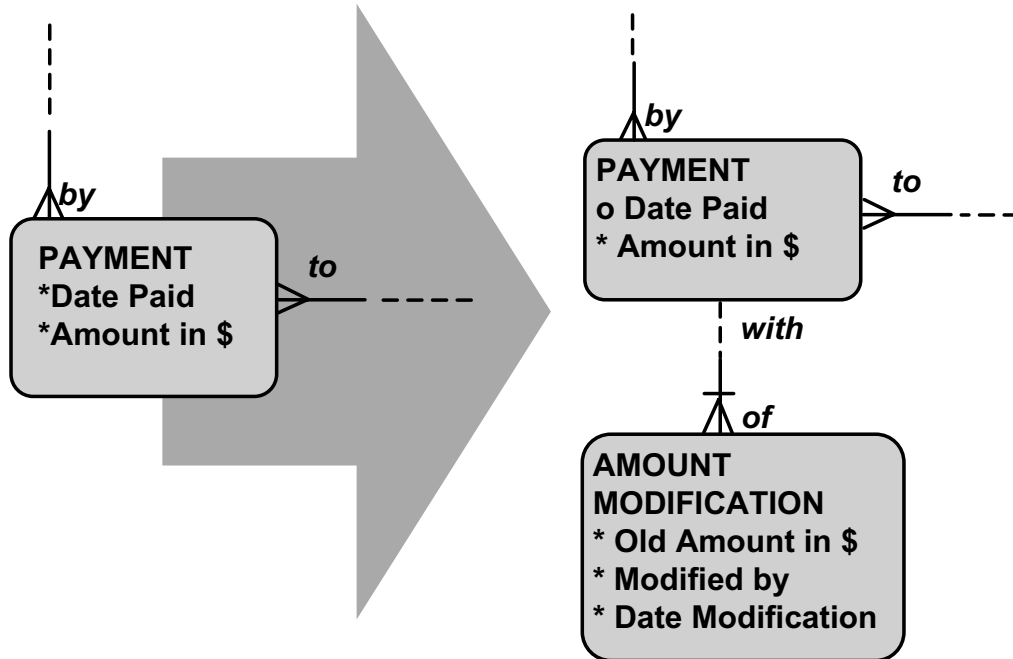
이러한 모델은 앞에서 보았던 PRODUCT-PRICE 모델에 대한 변형입니다.

왼쪽 모델에서 PRODUCT와 PRICE의 1:m 관계는 실제 과거 가격만 보여줍니다. End Date 속성은 필수이기 때문에 과거 가격만 보존된다고 추측할 수 있습니다. 추가적인 제약 조건은 이 값이 항상 과거에 존재해야 한다는 것입니다. PRODUCT의 Current Price는 속성으로 표시됩니다. 이 모델은 중복이 없습니다.

대부분의 상황에서는 PRICE 엔티티를 기준으로 한 개의 테이블에 현재 제품 가격과 과거 가격을 함께 보존하는 것이 좋은 설계 결정입니다. 가운데 모델은 이러한 상황에 대한 ER 표현입니다. 이제 End Date는 선택 사항입니다.

오른쪽 모델은 미묘한 중복을 포함하는 다른 모델입니다. 이러한 유형의 중복에 대한 자세한 내용은 비정규화 데이터에 대한 단원을 참조하십시오.

저널링(Journaling)



ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

저널링

시스템에서 사용자가 특정 정보를 수정하거나 제거할 수 있도록 허용하는 경우, 이전 값은 레코드에 보존해야 하는지에 대한 문제의 질문이 제기됩니다. 이것을 로깅 또는 저널링이라고 합니다. 정보가 재무 성격을 띠는 경우 종종 이러한 상황이 발생합니다.

모델에 대한 결과

저널링은 일반적으로 수정된 값과 수정자와 수정 시기에 대한 정보로 구성됩니다. 물론 이 추가 정보는 원할 경우 확장할 수 있습니다. 개념적 데이터 모델에 대한 결과와는 별도로, 시스템에는 특수한 저널링 기능이 필요합니다. Amount In의 갱신을 허용하는 모든 업무 기능으로 인해 요청된 갱신이 이루어지고, 적절한 값을 사용하여 AMOUNT MODIFICATION 엔티티 인스턴스가 생성됩니다. 물론 시스템에는 로깅된 데이터를 사용하여 무엇인가를 수행하기 위한 특수 기능도 필요합니다.

저널링

Journal 엔티티 없음

엔티티 속성의 일부 또는 전부가 저널링되어야 할 경우에는 원본과 동일한 열을 가지는 전체 shadow 테이블과 변경자, 변경 시기 및 변경 내용에 대한 정보를 저장할 일부 추가 테이블을 유지 관리함으로써 종종 구현됩니다. 이 테이블은 별도의 엔티티에서 비롯되지 않습니다. 즉, 이것은 한 개의 동일한 엔티티에 대한 두번째 특별한 구현입니다.

저널링 레지스터 전용

로깅은 사용자의 갱신 작업을 금지하지 않습니다. 갱신을 완전히 금지하는 것은 기능적인 문제이므로 개념적 데이터 모델에 표시되지 않습니다. 갱신을 완전히 금지하면 철자 오류 또는 기타 실수를 변경할 수 있는 가능성도 차단된다는 점에 유의하십시오.

이 단계에서는 갱신과 관련하여 시스템 동작에 대한 결정을 내려야 합니다. 때때로 이것은 개념적 데이터 모델의 수정으로 이어집니다.

예를 들어, 특정 업무 상황에서 특정 그룹의 사용자에게 PAYMENT 인스턴스의 생성은 허용되지만 변경은 허용되지 않는다고 가정합니다. 변경은 오직 재무 관리자에 의해서만 이루어질 수 있습니다. 사용자가 방금 PAYMENT 인스턴스를 생성했는데 실수를 했음을 알게 되었다고 가정합니다. 이러한 경우 해당 업무에는 잘못된 인스턴스를 정지시킬 수 있는 처리 방법이 필요합니다. 한 가지 처리 방법은 재무 관리자 중 한 명에게 변경을 요청하는 것입니다. 이보다 훨씬 나은 처리 방법은 지불이 무효화될 수 있도록 기능을 추가하는 것입니다. 이것은 모델에서 Neutralized Indicator 속성으로 표시되며 사용자는 이것을 Yes로 설정할 수 있습니다.

Oracle Internal & OAI Use Only

요약

- **이전 값 보존 필요성을 고려합니다.**
- **모델에서의 시간은 복잡합니다:**
 - 암시적 버전
 - 참조
- **저널링**

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

요약

시스템 내 모든 갱신은 정보의 유실을 의미합니다. 이를 피하기 위해 과거 상황에 대한 기록을 보존하는 모델을 생성할 수 있습니다. 때때로 관계는 엔티티의 시간 종속적 상태를 가리킵니다. 즉, 갱신된 엔티티는 사실 엔티티의 새 인스턴스이지 갱신된 기존 인스턴스가 아닙니다. 이 경우, 시간 종속적 참조 제약 조건은 관계만으로 모델링될 수 없습니다.

모델에서의 시간은 복잡한 문제입니다. 많은 모델에는 몇 개의 시간 관련 엔티티가 있습니다.

연습

- 근무 교대
- 딸기 와퍼
- 번들
- 제품 구조

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

Oracle Internal & OAI Use Only

연습: 근무 교대

Museumplein, Amsterdam, March 21					
Shift	1	2	3	4	5
Mon	6:30 11:30	11:30 16:00	16:00 20:30	20:30 23:00	–
Tue	7:00 11:30	11:30 16:00	16:00 20:30	20:30 23:00	–
Wed	7:00 11:30	11:30 16:00	16:00 20:30	20:30 23:00	–
Thu	7:00 11:30	11:30 16:00	16:00 20:30	20:30 23:00	–
Fri	7:00 11:30	11:30 16:00	16:00 20:30	20:30 24:00	–
Sat/Sun	8:00 11:30	11:30 15:00	15:00 18:00	18:00 21:00	21:00 24:00

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

연습 5-1: 근무 교대

목표

이 연습의 목적은 다양한 시간의 측면을 모델링하는 것입니다.

시나리오

몇몇 상점은 하루 24시간 일주일 내내 영업하고, 그 외의 상점은 밤에 문을 닫습니다. 종업원들은 교대로 근무합니다. 근무 교대는 현지 법률을 따릅니다. 다음은 암스테르담에 있는 한 상점에서 정의한 근무 교대를 나타낸 것입니다.

과제

이 근무 교대표에서 찾아 다양한 날짜/시간 요소를 나열하고 개념적 데이터 모델을 만드십시오.

연습: 딸기 와퍼

- 가격은 한 국가 내에서 동일한 수준이며, Global Pricing Department에서 결정합니다. 일반적으로 전 세계적으로 판매되는 정규 제품의 가격은 1년에 한 번 재확정됩니다.
- 현지 특산품의 가격과 출시는 개별 상점에 의해 결정됩니다. 예를 들어, 노르웨이의 유명한 Vafler med Jordbær(신선한 딸기로 만든 맛있는 와퍼)는 여름에만 판매되며, 그 가격은 신선한 딸기의 현지 시가에 따라 달라집니다.

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

연습 5-2: 딸기 와퍼

시나리오

이미 앞 단원에서 가격 목록을 모델링했습니다. 이제 몇 가지 새로운 정보를 사용할 수 있습니다.

과제

다음의 추가 정보가 제공된다고 가정할 경우 모델을 재방문하고 필요에 맞게 변경하십시오.

prijslijst

de Keyzer, Keyzerlei 15, Antwerpen
bezoekt ons op 't Web: www.moonlight.com

inclusief BTW
16 September

	klein	middel	groot	
gewone koffie	60	90	120	
cappuccino	90	110	140	
koffie verkeerd	75	100	130	
speciale koffies	99	125	150	
espresso	60	95	110	
koffie van de dag	45	75	100	
<i>caffeine vrij</i>	<i>5</i>	<i>10</i>	<i>15</i>	<i>toeslag</i>
zwarte thees	60	100	120	
vruchten thees	75	110	130	
kruiden thees	80	120	140	
dag thee	50	85	100	
<i>caffeine vrij</i>	<i>5</i>	<i>10</i>	<i>15</i>	<i>toeslag</i>
frisdranken	60	100	130	
diverse sodas	60	100	130	
mineraal water	75	120	140	
appel taart				180
brusselse wafel				150
portie chocolade bonbons				150
koekje van eigen deeg				120
portie slagroom				30

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

Oracle Internal & OAI Use Only

연습: 번들(1)

SweetTreat(tm)는 큰 음료수 한 병과 케이크로 구성됩니다.
BigBox(tm)는 큰 컵 커피 한 잔과 두 개의 케이크로 구성됩니다.

SuperSweetTreat(tm)는 SweetTreat(tm) 하나와 위핑 크림(케이크 위)으로 구성됩니다.

FamilyFeast(tm)는 두 개의 BigBoxe(tm)와 두 개의 SweetTreats™와 작은 선물로 구성됩니다.

DecafPunch(tm)는 보통 크기의 디카페인 커피 한 잔 또는 보통 크기의 디카페인 차와 블랙베리 머핀 한 개로 구성됩니다.

ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

연습 5-3: 번들

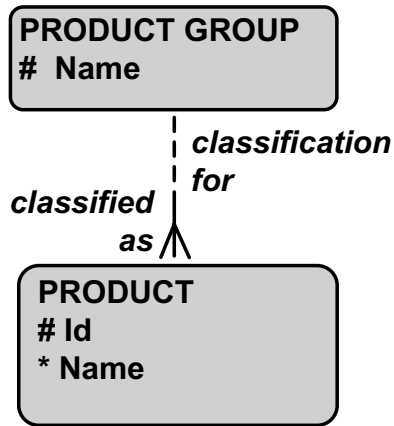
목표

이 단원의 목적은 이전 엔티티의 개념을 확장하는 것입니다.

시나리오

Moonlight에서는 시험 삼아 번들 제품을 일부 상점에서 특별 가격으로 판매합니다. 다음은 몇 가지 예제입니다.

연습: 번들(2)



ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

연습 5-3: 번들

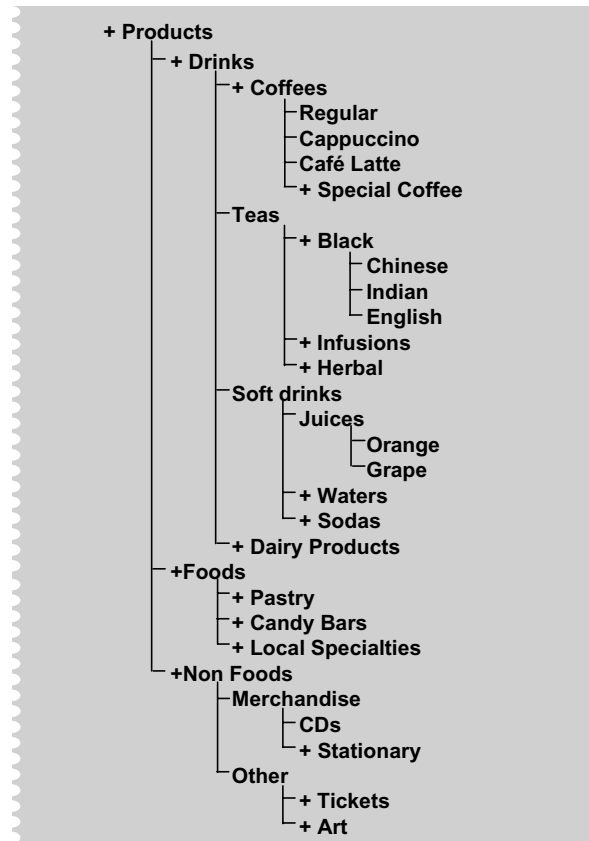
번들 제품은 아주 잘 팔리므로 온갖 종류의 새로운 번들이 나올 것으로 기대됩니다.

다양한 계산을 수행하기 위해서는 시스템에서 이러한 모든 상품이 어떻게 구성되어 있는지를 파악하고 있어야 합니다.

과제

1. 원하는 계산을 수행할 수 있는 방식으로 모델의 상품 부분을 수정하십시오.
2. 모델에 허용되는 방식으로 모델을 변경하십시오.

연습: 제품 구조



ORACLE

Copyright © Oracle Corporation, 2002. All rights reserved.

연습 5-4: 제품 구조

목표

이 연습의 목적은 계층 구조를 모델링하는 것입니다.

시나리오

Moonlight에서는 판매 정보를 사업 최적화 도구로 사용할 수 있도록 만들어야 합니다. 각기 다른 요약 레벨에서 보고할 수 있도록 하기 위해 계층적 제품 구조가 개발 중에 있습니다. 이 계층 구조는 단일 레벨 제품 그룹 분류를 대체해야 합니다. 위 그림은 제품 구조에 대한 현재의 아이디어를 표시한 것입니다. 이 구조는 완성된 상태가 아니지만, 구조가 취해야 할 모양에 대한 아이디어를 제공해 줄 것입니다. + 기호는 구조가 해당 지점에서 확장된다는 것을 의미합니다.

과제

1. 제품 분류 구조에 대한 모델을 생성하십시오.
2. (선택 사항) 번들 제품을 어떤 방식으로 취급할 것입니까?