10 장

데이터베이스 설계 II

Sejong Oh
Dankook University

목차

1. 함수적 종속성

- **개요**
  - 잘못 설계된 데이터베이스(구체적으로는 테이블들)는 나중에 테이블에 대한 삽입, 변경, 삭제 연산의 수행시 여러 가지 문제를 일으킬 수 있다.
  - 이러한 문제들을 **연산 이상(anomaly)**이라고 한다.
  - 연산 이상을 일으키는 원인은 테이블 내에 존재하는 **데이터의 중복** 때문
  - 테이블 내에 존재하는 데이터의 중복을 해소하는 과정을 **정규화 (normalization)**라고 한다.
  - 데이터의 중복을 해소하는 것을 다른 말로 표현하면 **함수적 종속성 (functional dependency)**을 제거한다고 말한다.

1. 함수적 종속성

- **연산 이상**

<표 9-3> 연산 이상의 종류와 의미

연산이상	설명
삽입 이상	테이블에 새로운 데이터를 삽입할 때 불필요한 데이터도 함께 삽입을 해야 하는 현상
갱신 이상	중복 튜플이 존재하는데, 이중 어느 한쪽만 갱신하여 두 튜플간에 불일치가 발생하는 현상
삭제 이상	어떤 튜플을 삭제 했는데 남아 있어야하는 다른 데이터까지 함께 삭제가 되는 현상

1. 함수적 종속성

- 예제 상황

고객주문

고객번호	과일ID	과일명	주문수량	고객명	주소
1001	F001	사과	10	김철수	대구
1001	F002	배	5	김철수	대구
1001	F003	딸기	20	김철수	대구
1002	F001	사과	8	박용하	용인
1002	F004	오렌지	5	박용하	용인
1003	F002	배	5	김수빈	서울

<그림 10-16> 고객주문 테이블 (기본키: 고객번호, 과일ID)

한명의 고객이 여러 과일을 주문할 수도 있고, 동일 과일이 여러 고객에 의해 주문이 될 수 있으므로 고객번호나 과일ID 단독으로는 기본키가 될 수 없다. 따라서 고객번호, 과일ID 두 개의 컬럼이 함께 기본키의 역할

1. 함수적 종속성

- **(1) 삽입 이상**
  - 테이블에 새로운 데이터를 삽입할 때 불필요한 데이터도 함께 삽입을 해야 하는 현상
  - 아직 주문기록이 없는 '부산'에 사는 새로운 고객 '이수민'의 정보를 테이블에 저장해야 하는 경우

고객주문

고객번호	과일ID	과일명	주문수량	고객명	주소
1001	F001	사과	10	김철수	대구
1001	F002	배	5	김철수	대구
1001	F003	딸기	20	김철수	대구
1002	F001	사과	8	박용하	용인
1002	F004	오렌지	5	박용하	용인
1003	F002	배	5	김수빈	서울
1004	**NULL**	**NULL**	**NULL**	**이수민**	**부산**

기본키 컬럼에는 NULL이 입력될 수 없음

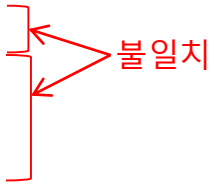
새로운 고객 정보를 입력하기 위해 주문하지도 않은 과일 ID을 함께 저장해야 함

1. 함수적 종속성

- **(2) 갱신이상**
  - 중복 튜플이 존재하는데, 이중 일부의 튜플만 갱신하여 중복 튜플들 간에 불일치가 발생하는 현상
  - 고객주문 테이블에서 '김철수' 고객이 대구에서 광주로 이사를 하게 되어 고객 정보를 수정해야 하는 상황

고객주문

고객번호	과일ID	과일명	주문수량	고객명	주소
1001	F001	사과	10	김철수	광주
1001	F002	배	5	김철수	대구
1001	F003	딸기	20	김철수	대구
1002	F001	사과	8	박용하	용인
1002	F004	오렌지	5	박용하	용인
1003	F002	배	5	김수빈	서울



중복된 정보중 일부만 수정함으로 해서 불일치 발생

1. 함수적 종속성

- **삭제 이상**
  - 어떤 튜플을 삭제 했는데 남아 있어야하는 다른 데이터까지 함께 삭제가 되는 현상
  - 고객주문 테이블에서 '김철수' 고객이 주문을 취소하여 주문정보를 테이블에서 삭제하는 경우

고객주문

고객번호	과일ID	과일명	주문수량	고객명	주소
1001	F001	사과	10	김철수	대구
1001	F002	배	5	김철수	대구
1001	F003	딸기	20	김철수	대구
1002	F001	사과	8	박용하	용인
1002	F004	오렌지	5	박용하	용인
1003	F002	배	5	김수빈	서울

주문정보의 삭제

고객정보도 삭제됨

주문정보는 삭제하되 '김철수' 고객의 정보는 남아 있어야하나, 주문 정보를 삭제하면 고객정보도 함께 삭제가 될 수밖에 없음

1. 함수적 종속성

- **함수적 종속성**
  - 삽입 이상, 갱신 이상, 삭제 이상이 발생하는 이유는 테이블의 설계가 잘못 되었기 때문
  - 테이블이 바람직하지 않은 **함수적 종속성(functional dependency)**을 포함하고 있을 때 문제가 발생
  - 함수적 종속성의 정의

어떤 릴레이션을 구성하는 속성들의 부분집합 X와 Y가 있을 때 임의의 X의 값을 하나 선택했을 때, Y가 유일한 값으로 대응된다면 'X가 Y를 함수적으로 결정한다' 또는 'Y는 X에 함수적으로 종속되어 있다'라고 말한다.

  - 함수적 종속성의 표현

$$X \longrightarrow Y$$

X : **결정 속성**, Y : **종속 속성**

1. 함수적 종속성

- 함수적 종속성의 예

고객주문

고객번호	과일ID	과일명	주문수량	고객명	주소
1001	F001	사과	10	김철수	대구
1001	F002	배	5	김철수	대구
1001	F003	딸기	20	김철수	대구
1002	F001	사과	8	박용하	용인
1002	F004	오렌지	5	박용하	용인
1003	F002	배	5	김수빈	서울

'고객번호가 정해지면 고객명은 자동적으로 정해진다' {고객번호}⟶ {고객명}

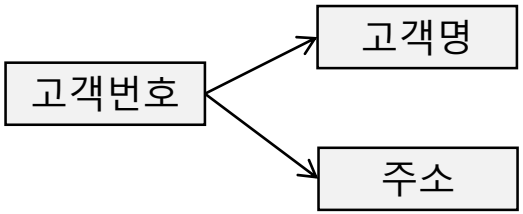
'고객번호가 정해지면 주소도 자동적으로 정해진다' {고객번호}⟶ {주소}

{고객번호}⟶ {고객명, 주소}

'과일ID가 정해지면 과일명도 자동적으로 정해진다' {과일ID} ⟶ {과일명}

1. 함수적 종속성

- **함수 종속 다이어그램**
  - 어떤 릴레이션의 함수적 종속 관계를 다이어그램으로 표시한 것

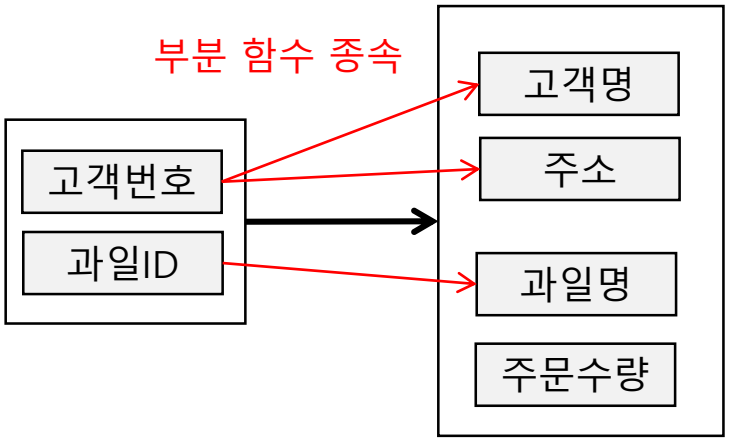


<그림 10-20> 함수 종속 다이어그램의 예

1. 함수적 종속성

- **주식별자와 함수적 종속성**
  - 주식별자(기본 키)는 릴레이션(테이블)내에서 인스턴스(튜플)을 식별하는 역할을 하므로 주식별자가 아닌 모든 속성들의 집합은 주식별자에 함수적으로 종속
  - 이는 모든 릴레이션이 공통적으로 가지는 성질, 정상적인 함수적 종속

- **완전 함수 종속과 부분 함수 종속**
  - **완전 함수 종속**(full functional dependency) : 주식별자 여러개인 릴레이션에서 주식별자가 아닌 속성들이 주식별자 전체 속성에 함수적으로 종속된 경우
  - **부분 함수 종속**(partial functional dependency) : 주식별자가 아닌 속성들이 주식별자의 부분 속성에 함수적으로 종속된 경우



데이터베이스 설계에서 문제가 되는 부분은 부분 함수 종속

<그림 10-21> 고객주문 릴레이션에 대한 함수 종속 다이어그램

1. 함수적 종속성

- 바람직하지 않은 함수적 종속성은 연산 이상을 일으키는 원인
- 올바른 테이블의 설계는 바람직하지 않은 함수적 종속성을 해소해 나가면서 설계하는 것 → 정규화

부분 함수 종속 포함

고객주문

고객번호	과일ID	과일명	주문수량	고객명	주소
1001	F001	사과	10	김철수	대구
1001	F002	배	5	김철수	대구
1001	F003	딸기	20	김철수	대구
1002	F001	사과	8	박용하	용인
1002	F004	오렌지	5	박용하	용인
1003	F002	배	5	김수빈	서울

고객

고객번호	고객명	주소
1001	김철수	대구
1002	박용하	용인
1003	김수빈	서울

주문

고객번호	과일ID	주문수량
1001	F001	10
1001	F002	5
1001	F003	20
1002	F001	8
1002	10F004	5
1003	F002	5

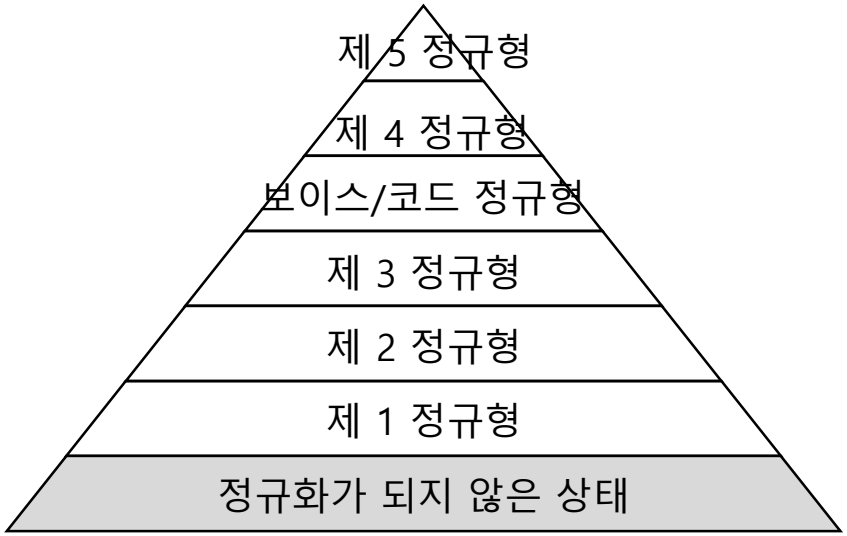
과일

과일ID	과일명
F001	사과
F002	배
F003	딸기
F004	오렌지

<그림 10-22> 함수적 종속성(부분 함수 종속)을 포함하지 않는 설계의 예

2. 정규화

- **개요**
  - 데이터베이스 설계의 과정은 바람직하지 않은 함수적 종속성이 존재하지 않도록 테이블을 정규화해가는 과정
  - 규화는 여러 단계의 수준에서 이루어진다.
  - 정규화가 이루어진 정도를 **정규형(normal Form)**이라고 하는데 제 1 정규형을 만족하면 제 1 정규화가 이루어졌다고 표현
  - 제 3 정규형까지를 이해하고 활용할 수 있으면 충분



<그림 10-23> 정규형의 단계

2. 정규화

- **제 1 정규형**
  - 어떤 릴레이션의 설계가 다음 조건을 만족하면 제 1 정규형에 속한다.

> 릴레이션에 속한 모든 속성의 도메인이 단일 값(atomic value)으로만 구성되어 있어야 한다.

  - 테이블에 있는 튜플들의 모든 컬럼은 단일 값만을 저장해야 한다
  - 단일값 : 더 이상 나누면 의미를 상실하는 값

주문

주문번호
주문과일

주문

주문번호	주문과일
1001	사과, 배, 딸기
1002	사과, 오렌지
1003	배

단일값이 아님

<그림 10-24> 제 1 정규화 대상 릴레이션 (테이블)

· 현재 판매되는 과일은 몇 종류인가?
· 고객들이 한번에 주문하는 과일의 종류는 평균 몇 개인가?
· 주문번호별로 주문한 과일의 개수를 보이시오

⇨ 처리 어려움

2. 정규화

- 제 1 정규화의 시행

주문

주문번호 주문과일

주문

주문번호	주문과일
1001	사과
1001	배
1001	딸기
1002	사과
1002	오렌지
1003	배

"주문번호별로 주문한 과일의 개수를 보이시오"

```
select 주문번호, count(*)
from 주문
group by 주문번호 ;
```

2. 정규화

- 제 1 정규형이기는 하나 초보 설계자가 자주 하는 설계의 실수

주문

주문번호
주문과일1 주문과일2 주문과일3

주문

주문번호	주문과일1	주문과일2	주문과일3
1001	사과	배	딸기
1002	사과	오렌지	NULL
1003	배	NULL	NULL

그림 10-26> 바람직하지 않은 제 1 정규형

<문제점>

- 주문시 한번에 과일을 세 개까지(주문과일 컬럼의 수까지)만 주문할 수 있다.
- 한번에 10개의 과일을 주문할 수 있기 위해서는 주문과일 컬럼의 수를 늘려야 한다. 그렇게 되면 주문 과일의 개수가 적은 튜플에는 많은 수의 NULL이 저장된다.
- SQL문이 복잡해 진다. (예: 주문번호 1004의 주문 과일 목록을 보이시오)

2. 정규화

- **제 2 정규형**
  - 어떤 릴레이션의 설계가 다음 조건을 만족하면 제 2 정규형에 속한다.

· 릴레이션이 제 1 정규형에 속한다.

· 릴레이션에서 주식별자와 일반 속성들 사이에는 완전 함수 종속 관계만 존재한다.

(주삭별자의 일부 속성에 종속되는 속성은 존재하지 않는다)

도시

도시명 국가코드
도시면적 도시인구수 국가명 공식언어

도시

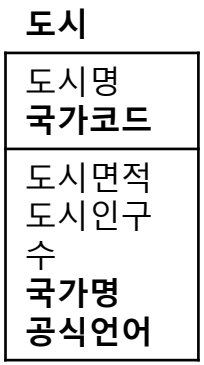
도시명	국가코드	도시면적	도시인구수	국가명	공식언어
Seoul	KOR	3400	1000	대한민국	Korean
Pusan	KOR	800	380	대한민국	Korean
Inchon	KOR	750	255	대한민국	Korean
Chicago	USA	1100	289	미국	English
Dallas	USA	980	118	미국	English
London	GBR	890	728	영국	English

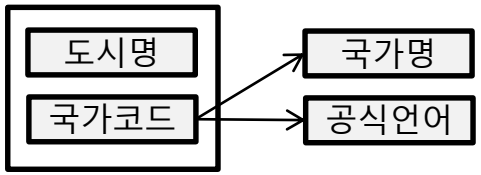
<그림 10-27> 제 2 정규화가 필요한 설계의 예

국가명과 공식언어가 주식별자의 일부인 국가코드에 함수적으로 종속

2. 정규화

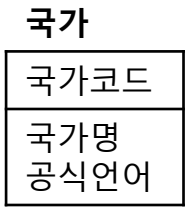
- 제 2 정규화

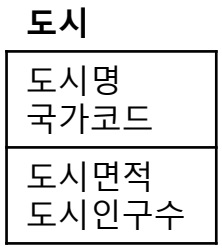


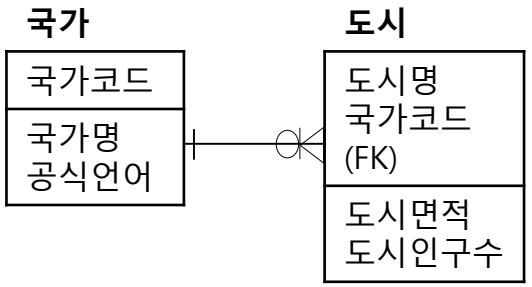


부분 함수 종속 존재

<제 2 정규화 과정>







① 함수적 종속관계를 새로운 릴레이션으로 생성

② 원 릴레이션에서 종속 속성을 제거

③ 새로운 릴레이션(부모)과 원 릴레이션(자식)의 관계 연결

2. 정규화

도시

도시명	국가코드	도시면적	도시인구수	국가명	공식언어
Seoul	KOR	3400	1000	대한민국	Korean
Pusan	KOR	800	380	대한민국	Korean
Inchon	KOR	750	255	대한민국	Korean
Chicago	USA	1100	289	미국	English
Dallas	USA	980	118	미국	English
London	GBR	890	728	영국	English

제 2 정규화

국가

국가코드	국가명	공식언어
KOR	대한민국	Korean
USA	미국	English
GBR	영국	English

도시

도시명	국가코드	도시면적	도시인구수
Seoul	KOR	3400	1000
Pusan	KOR	800	380
Inchon	KOR	750	255
Chicago	USA	1100	289
Dallas	USA	980	118
London	GBR	890	728

2. 정규화

- **제 3 정규형**
  - 어떤 릴레이션의 설계가 다음 조건을 만족하면 제 3 정규형에 속한다.

· 릴레이션이 제 2 정규형에 속한다.

· 릴레이션에서 주식별자가 아닌 속성들간에는 함수적 종속성이 존재하지 않는다.

판매

판매번호 상품번호
판매수량 AS담당자ID AS담당자이름 AS담당자연락처

판매

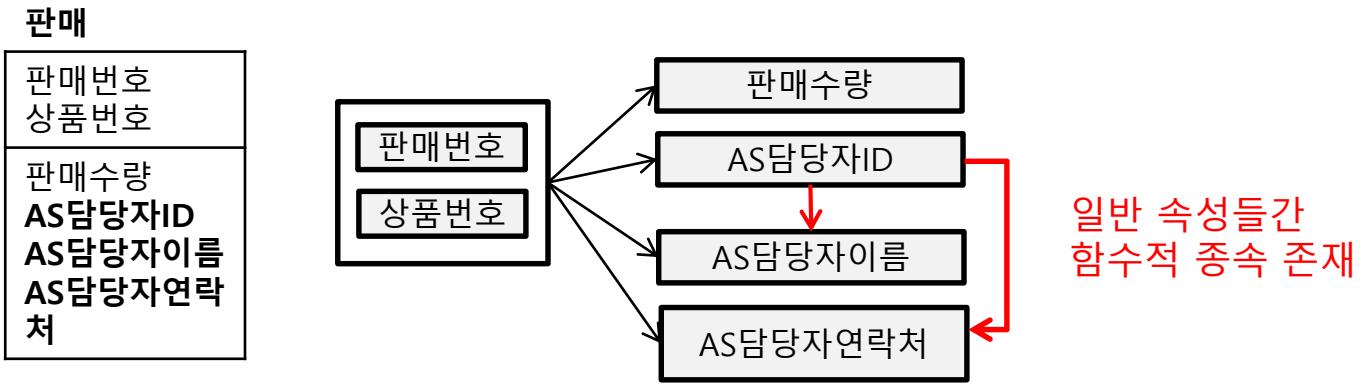
판매번호	상품번호	판매수량	AS담당자ID	AS담당자이름	AS담당자연락처
S001	P100	50	72001	최승일	010-5213-7642
S001	P200	30	72003	김미나	010-1422-7421
S002	P100	100	72004	박한솔	010-5233-6421
S003	P100	60	72001	최승일	010-5213-7642
S003	P200	50	72002	한민구	010-4551-9876
S003	P300	100	72003	김미나	010-1422-7421

<그림 10-30> 제 3 정규화가 필요한 설계

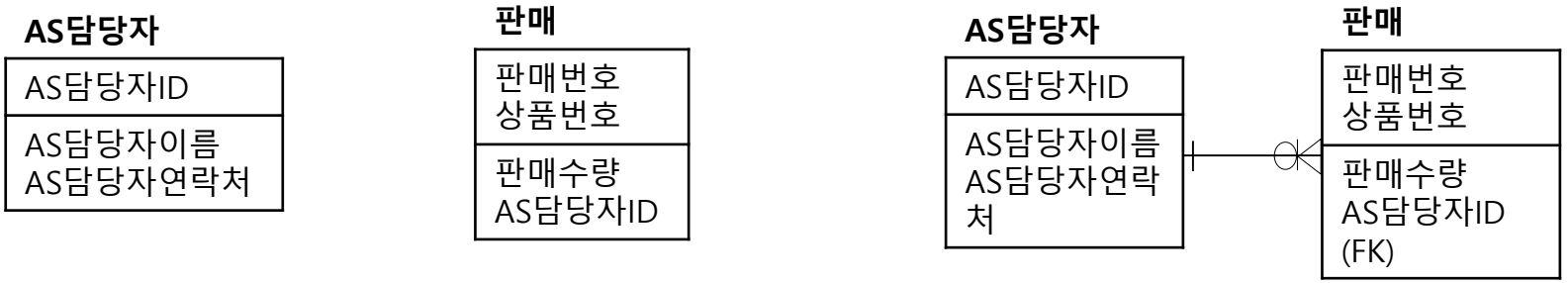
'AS담당자이름'과 'AS담당자연락처'가 'AS담당자ID'에 종속

2. 정규화

- 제 3 정규화



<제 3 정규화 과정>



① 함수적 종속관계를 새로운 릴레이션으로 생성

② 원 릴레이션에서 종속 속성을 제거

③ 새로운 릴레이션(부모)과 원 릴레이션(자식)의 관계 연결

2. 정규화

판매

판매번호	상품번호	판매수량	AS담당자ID	AS담당자이름	AS담당자연락처
S001	P100	50	72001	최승일	010-5213-7642
S001	P200	30	72003	김미나	010-1422-7421
S002	P100	100	72004	박한솔	010-5233-6421
S003	P100	60	72001	최승일	010-5213-7642
S003	P200	50	72002	한민구	010-4551-9876
S003	P300	100	72003	김미나	010-1422-7421

제 3 정규화

AS담당자

AS담당자ID	AS담당자이름	AS담당자연락처
72001	최승일	010-5213-7642
72002	한민구	010-4551-9876
72003	김미나	010-1422-7421
72004	박한솔	010-5233-6421

판매

판매번호	상품번호	판매수량	AS담당자ID
S001	P100	50	72001
S001	P200	30	72003
S002	P100	100	72004
S003	P100	60	72001
S003	P200	50	72002
S003	P300	100	72003

2. 정규화

Note. 이행적 종속성

- 어떤 릴레이션에서 X, Y, Z라는 3 개의 속성이 있을 때 X→Y, Y→Z 이란 종속 관계가 있을 경우, X→Z가 성립될 때 이행적 함수 종속(transitive functional dependecy)이라고 한다.
- 즉, X를 알면 Y를 알고 그를 통해 Z를 알 수 있는 경우를 말한다.
- <그림 10-31>을 보면 {판매번호, 상품번호} → {AS담당자ID} 이고 또한 {AS담당자ID}→{AS담당자이름, AS담당자 연락처}의 관계가 성립한다.
- 따라서 이행적 종속성이 존재한다고 말할 수 있다.
- 결국 제3 정규화는 이행적 종속성을 해소하는 과정이라고 말할 수 있다.

2. 정규화

- **보이스/코드 정규형**
  - 어떤 릴레이션의 설계가 다음 조건을 만족하면 보이스/코드 정규형(BCNF; Boyce/Codd normal form)에 속한다.

> · 릴레이션이 제 3 정규형에 속한다.
> · 릴레이션에서 주식별자의 일부 속성이 일반 속성에 함수적으로 종속하는 경우는 존재하지 않는다.

성적

학번	강사ID	클래스ID	성적
S001	T01	C100	A
S001	T02	C200	A
S001	T03	C300	B
S002	T01	C100	B
S003	T01	C100	A
S003	T03	C300	C

<그림 10-33> 보이스/코드 정규화가 필요한 설계

2. 정규화

성적

학번	강사ID	클래스ID	성적
S001	**T01**	C100	A
S001	T02	C200	A
S001	T03	C300	B
S002	T01	C100	B
S003	T01	C100	A
S003	T03	C300	C
S001	**T01**	**C400**	**B**

기본키 중복으로 저장 안됨

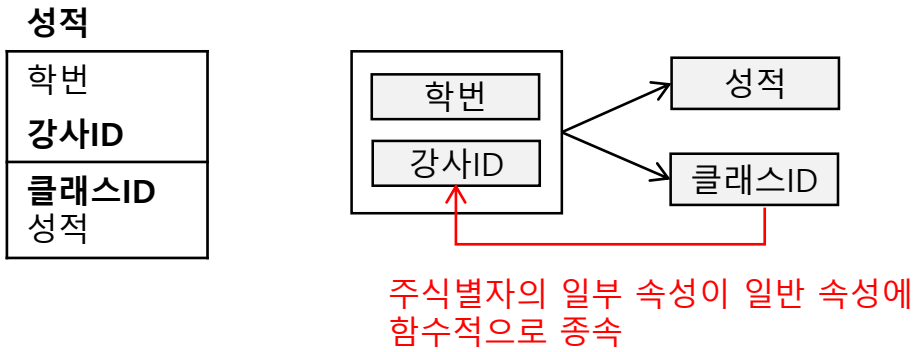
강사 T01은 C100과 C400 두개의 클래스를 담당하고 있는데, 학번이 S001인 학생이 C100과 C400 두개의 클래스를 모두 수강하여 각각 A와 B를 받았다고 하자.

<그림 10-34> 보이스/코드 정규화가 안된 테이블에서의 튜플 삽입 문제

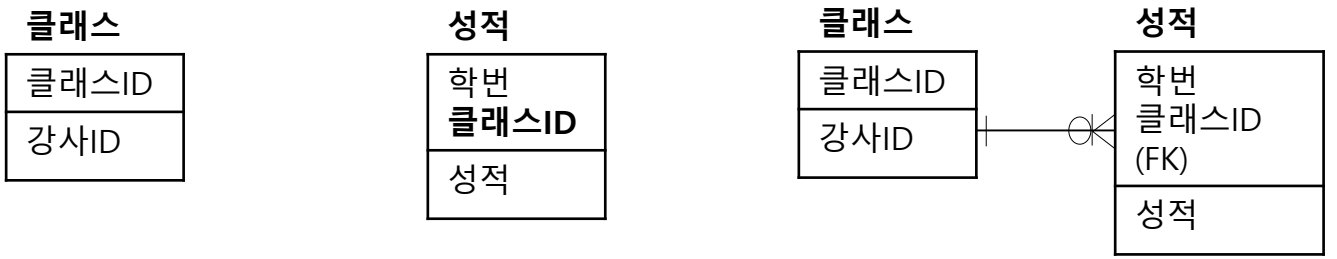
C100에 대한 정보는 저장을 했으므로 C400에 대한 정보를 저장해야 하는데 기본키 컬럼의 중복에 의하여 저장을 할 수가 없는 상황이 발생

2. 정규화

- 보이스/코드 정규화



<보이스/코드 정규화 과정>



① 함수적 종속관계를 새로운 릴레이션으로 생성

② 원 릴레이션에서 종속 속성을 결정 속성으로 대체

③ 새로운 릴레이션(부모)과 원 릴레이션(자식)의 관계 연결

2. 정규화

성적

학번	강사ID	클래스ID	성적
S001	T01	C100	A
S001	T02	C200	A
S001	T03	C300	B
S002	T01	C100	B
S003	T01	C100	A
S003	T03	C300	C

보이스/코드 정규화

클래스

클래스ID	강사ID
C100	T01
C200	T02
C300	T03
C400	**T01**

성적

학번	클래스ID	성적
S001	C100	A
S001	C200	A
S001	C300	B
S002	C100	B
S003	C100	A
S003	C300	C
S001	**C400**	**B**

2. 정규화

- **제 4 정규형**
  - 어떤 릴레이션의 설계가 다음 조건을 만족하면 제 4 정규형에 속한다.

· 릴레이션이 보이스/코드 정규형에 속한다.

· 릴레이션에서 속성들 간에 다가 종속(multivalued dependency)관계가 두개 이상 존재하지 않는다.

업무배정

사번	프로젝트ID	역할ID
S001	T01	NULL
S001	T02	NULL
S001	NULL	R1
S001	NULL	R2
S002	T01	R2
S002	NULL	R3

한명의 사원에 대해 여러 개의 프로젝트와 여러개의 역할이 존재할 수 있는데 이를 다가 종속이라고 한다.

<그림 10-37> 제 4 정규화를 필요로 하는 설계의 예

2. 정규화

업무배정

사번	프로젝트ID	역할ID
S001	T01	NULL
S001	T02	NULL
S001	NULL	R1
S001	NULL	R2
S002	T01	R2
S002	NULL	R3

제 4 정규화

프로젝트_배정

사번	프로젝트ID
S001	T01
S001	T02
S002	T01

역할_배정

사번	역할ID
S001	R1
S001	R2
S002	R2
S002	R3

3. 정규화 사례

- **정규화 정리**
  - 정규화 과정은 릴레이션(테이블)에 포함된 중복 데이터를 해소해 나가는 과정이다.
    - 즉, 데이터베이스에는 중복된 데이터가 저장되어서는 안된다는 대 원칙을 구현하기 위한 절차가 정규화
    - 예를 들면 사번이 'S001'인 사원의 이름이 '홍길동'이라는 정보는 데이터베이스 내에 단 한번만 기록되어야 한다
    - 동일 정보가 여러 번 저장되는 경우 앞에서 살펴본 삽입 이상, 갱신 이상, 삭제 이상이 발생할 수 있다.
  - 정규화는 여러 단계로 진행이 될 수 있다. 정규화를 실시하면 릴레이션이 분해되므로 릴레이션의 개수가 늘어난다.
  - 정규화에 의해 릴레이션 A가 A'과 B로 분해가 되면 A'와 B는 외래 식별자에 의해 연결된다. A'와 B를 조인하면 A의 정보를 복원할 수 있다.
  - 정규형에 대해 잘 이해하고 있으면 처음부터 정규화가 필요 없는 데이터베이스의 설계가 가능

3. 정규화 사례

- **정규화 대상 전표**

판매 전표

판매일자 : *2022.09.02*
판매부서 : *DH01* 부서명 : *영업 1부*
판매사원번호 : *12437* 사원명 : *김철수*
판매일련번호: *3*

고객번호 : *YS02* 고객명 : *영진전자*
고객주소 : *서울시 영등포구 여의도동 137-27*

제품번호	제품명	단가	수량	금 액
DW01	TV 12”	500	10	5,000
DW09	선풍기	200	20	4,000
DW20	라디오	100	30	3,000

3. 정규화 사례

- **기본 테이블의 설계**
  - 판매 전표에 있는 모든 데이터 항목을 컬럼으로 만든다.
  - 주식별자는 판매일자, 부서번호, 사원번호, 일련번호

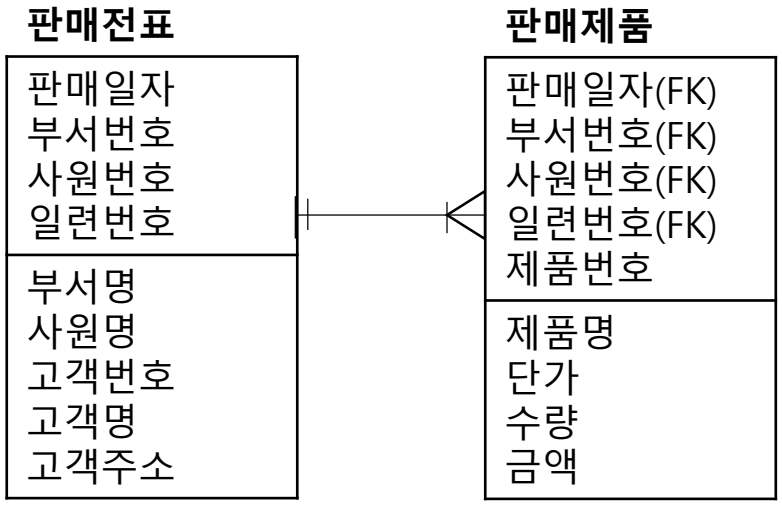
판매전표

판매일자 부서번호 사원번호 일련번호
부서명 사원명 고객번호 고객명 고객주소 제품번호1 제품명1 단가1 수량1 금액1 제품번호2 제품명2 단가2 수량2 금액2 제품번호3 제품명3 단가3 수량3 금액3

반복속성 존재

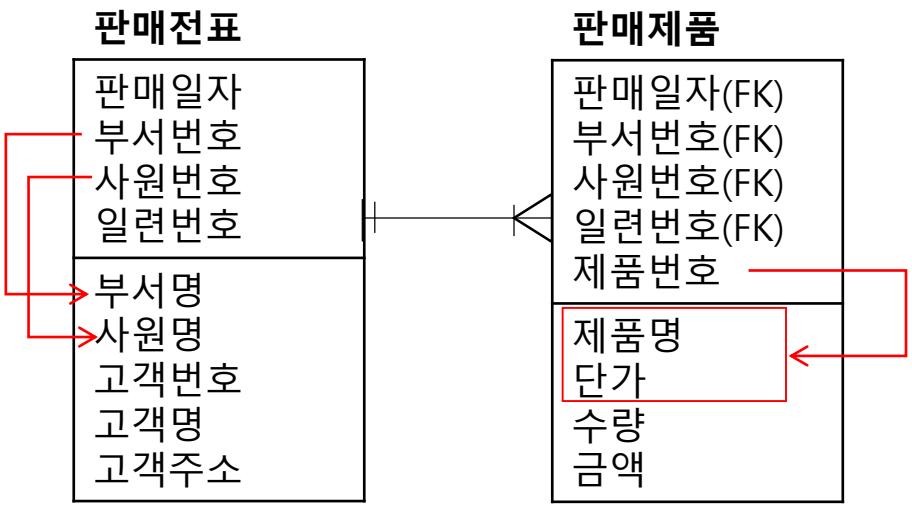
3. 정규화 사례

- **제 1 정규화**
  - 튜플의 각 컬럼에 원자값이 아닌 값들이 저장되는 부분은 없는지, 혹은 릴레이션 내에서 반복되는 속성들은 없는지 살펴본다.
  - 반복 속성이 존재하므로 이를 해소한다

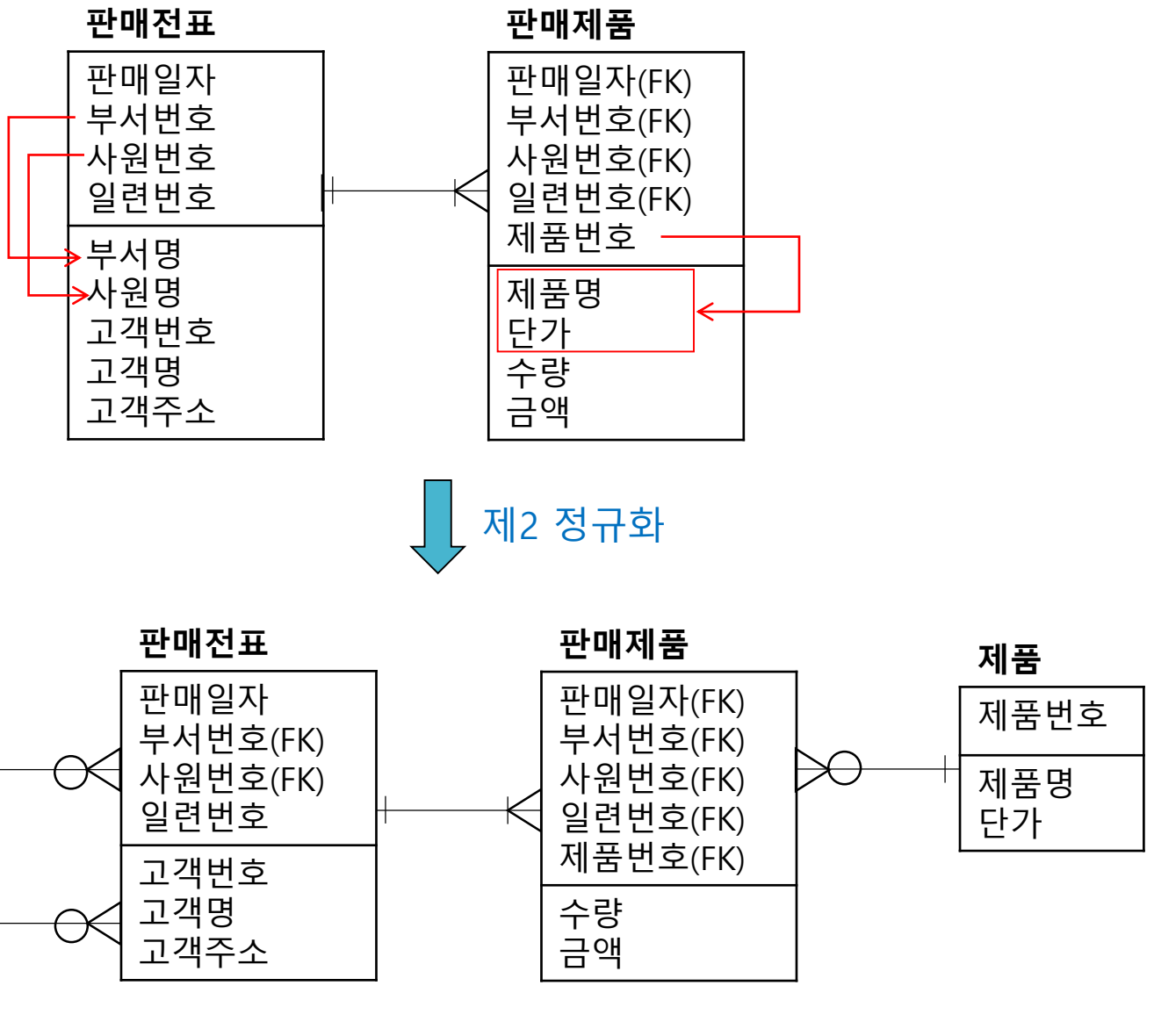


3. 정규화 사례

- **제 2 정규화**
  - 일반 속성들중 주식별자의 일부에 함수적으로 종속된 것들은 없는지 찾아본다.
  - '판매전표'에서는 '부서명'이 '부서번호'에, '사원명'이 '사원번호'에 함수적으로 종속
  - '판매제품'에서는 '제품명'과 '단가'가 '제품번호'에 함수적으로 종속

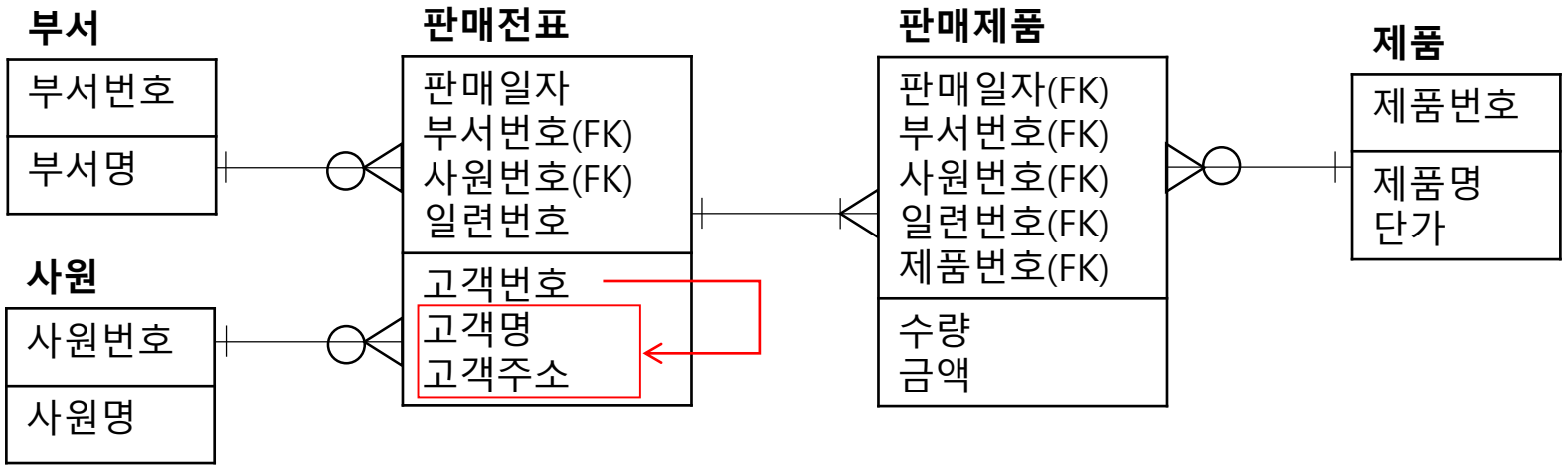


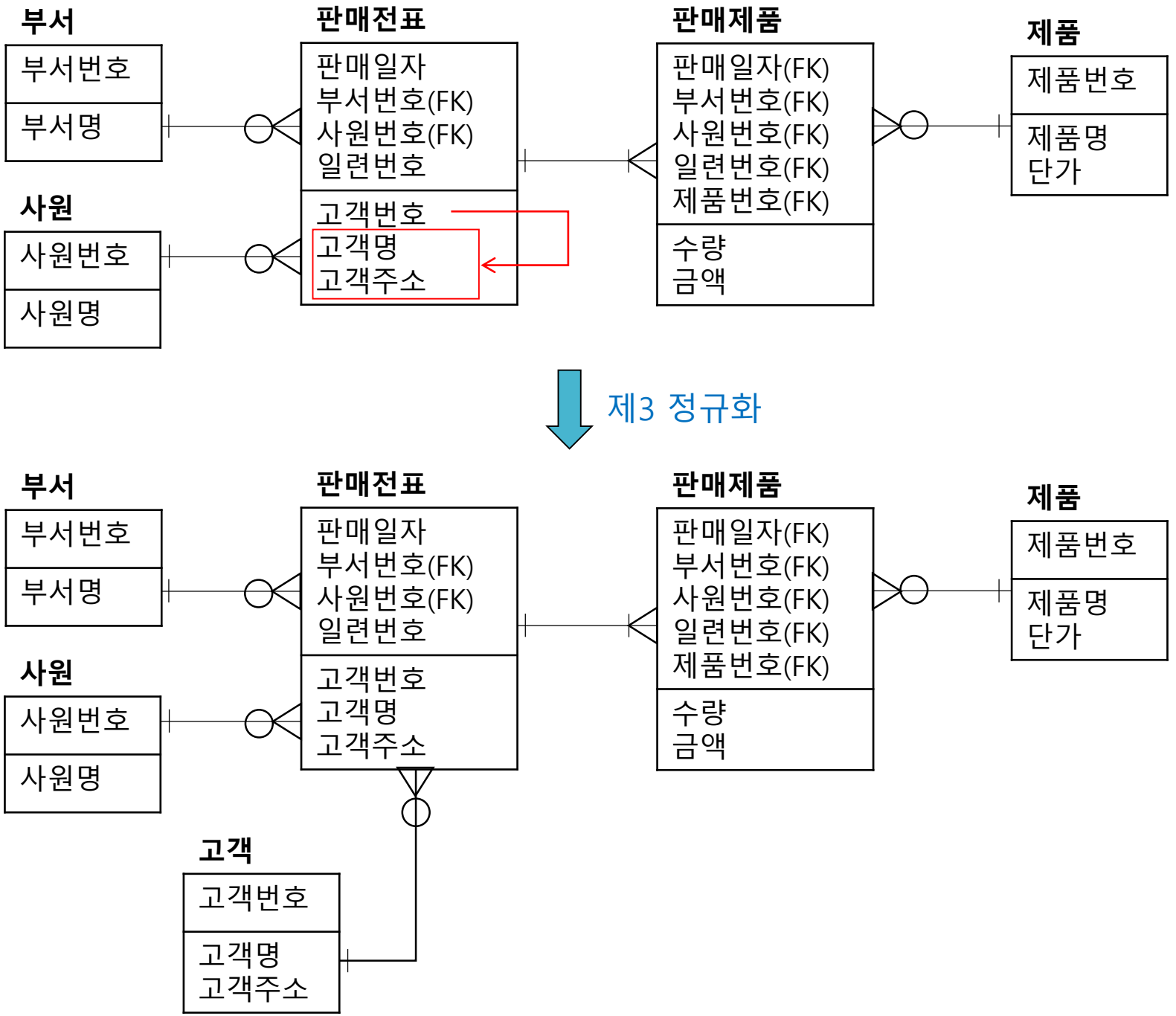
3. 정규화 사례


판매전표
판매일자
부서번호
사원번호
일련번호
부서명
사원명
고객번호
고객명
고객주소
판매제품
판매일자(FK)
부서번호(FK)
사원번호(FK)
일련번호(FK)
제품번호
제품명
단가
수량
금액
제2 정규화
부서
부서번호
부서명
사원
사원번호
사원명
판매전표
판매일자
부서번호(FK)
사원번호(FK)
일련번호
고객번호
고객명
고객주소
판매제품
판매일자(FK)
부서번호(FK)
사원번호(FK)
일련번호(FK)
제품번호(FK)
수량
금액
제품
제품번호
제품명
단가

3. 정규화 사례

- **제 3 정규화**
  - 일반 속성들 간에 함수적 종속관계가 있는지를 찾아 이를 해소
  - '판매전표' 엔티티에서 '고객명'과 '고객주소'가 '고객번호'에 함수적으로 종속




부서
부서번호
부서명
사원
사원번호
사원명
판매전표
판매일자
부서번호(FK)
사원번호(FK)
일련번호
고객번호
고객명
고객주소
판매제품
판매일자(FK)
부서번호(FK)
사원번호(FK)
일련번호(FK)
제품번호(FK)
수량
금액
제품
제품번호
제품명
단가
제3 정규화
부서
부서번호
부서명
사원
사원번호
사원명
판매전표
판매일자
부서번호(FK)
사원번호(FK)
일련번호
고객번호
고객명
고객주소
판매제품
판매일자(FK)
부서번호(FK)
사원번호(FK)
일련번호(FK)
제품번호(FK)
수량
금액
제품
제품번호
제품명
단가
고객
고객번호
고객명
고객주소

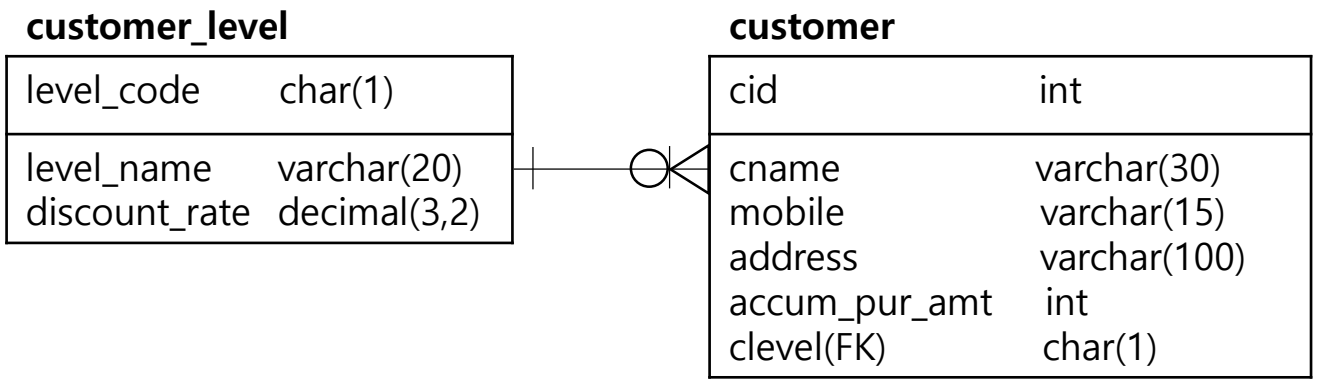
실습. Data Modeler를 이용한 순공학, 역공학 작업

- **개요**
  - 물리적 ERD의 설계가 끝나면 다음 단계는 실제로 데이터베이스를 구축하는 것
  - 대부분의 데이터베이스 설계 도구들은 물리적 ERD로부터 쉽게 데이터베이스를 구축할수 있는 기능을 제공 => **순공학(forward engineering)**
  - 이미 구축된 데이터베이스를 분석하여 데이터베이스의 내용을 물리적 ERD로 보여주는 기능도 제공 => **역공학(reverse engineering)**

  - Data Modeler도 이러한 기능 제공

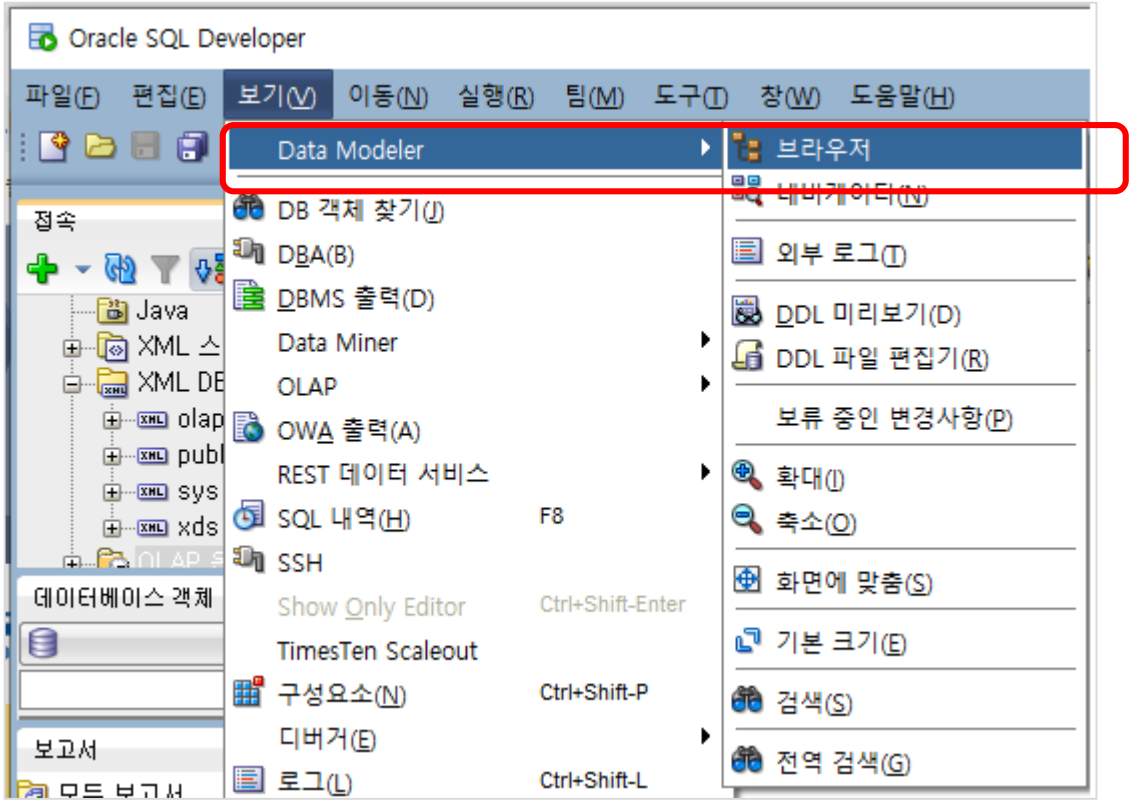
실습. Data Modeler를 이용한 순공학, 역공학 작업

- **순공학 실습**
  - 지난 단원에서 실습했던 고객/고객등급 ERD를 emppdb 에 반영하는 작업을 실습



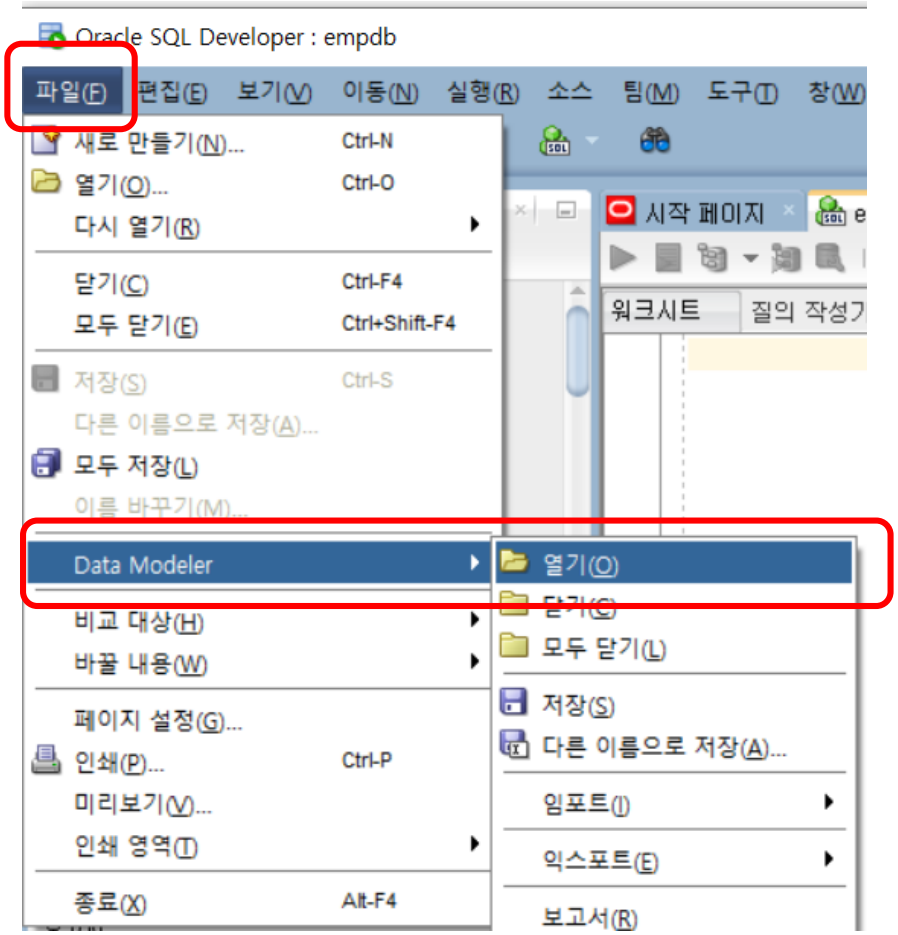
실습. Data Modeler를 이용한 순공학, 역공학 작업

- ① Data Modeler 브라우저를 연다.



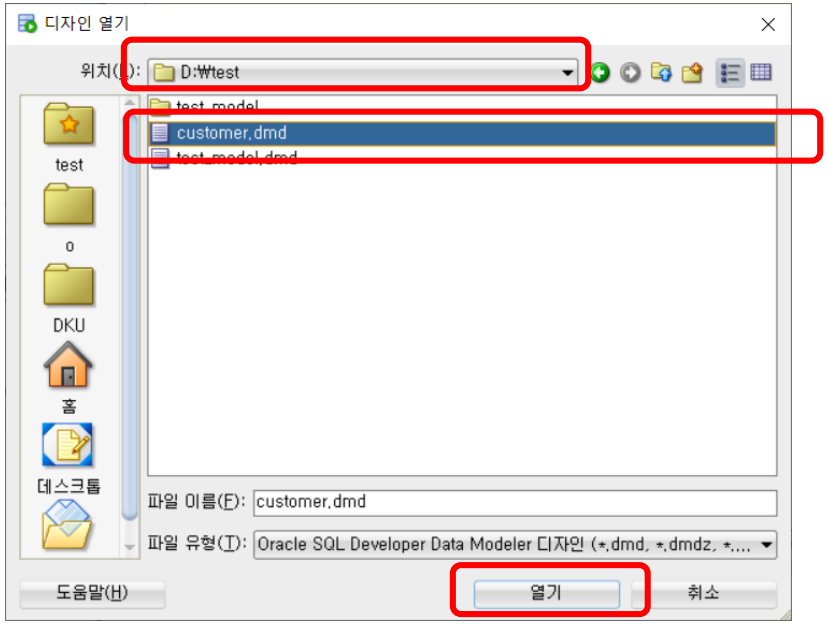
실습. Data Modeler를 이용한 순공학, 역공학 작업

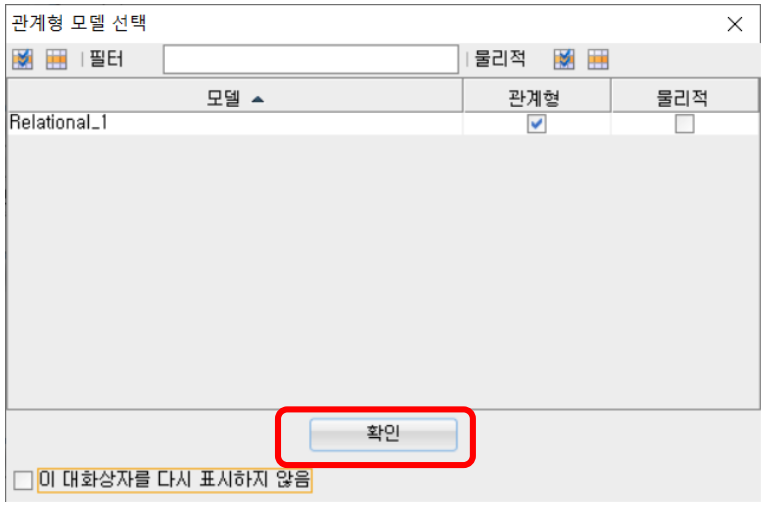
- ②메인 메뉴에서 [파일]→[Data Modeler]→[열기]의 순서로 선택한다.

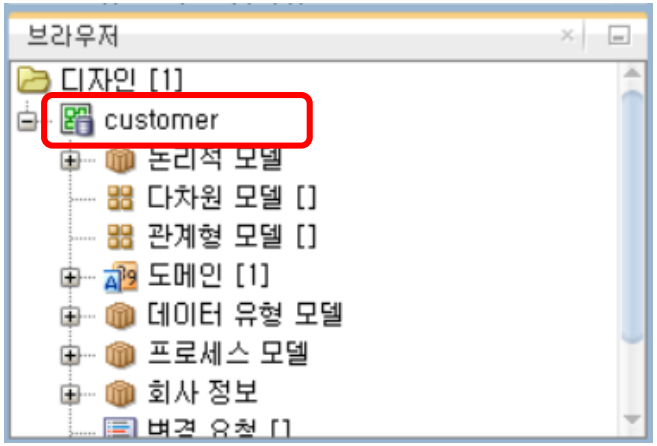


실습. Data Modeler를 이용한 순공학, 역공학 작업

- ③ 지난 실습에서 저장했던 customer.dmd를 읽어온다

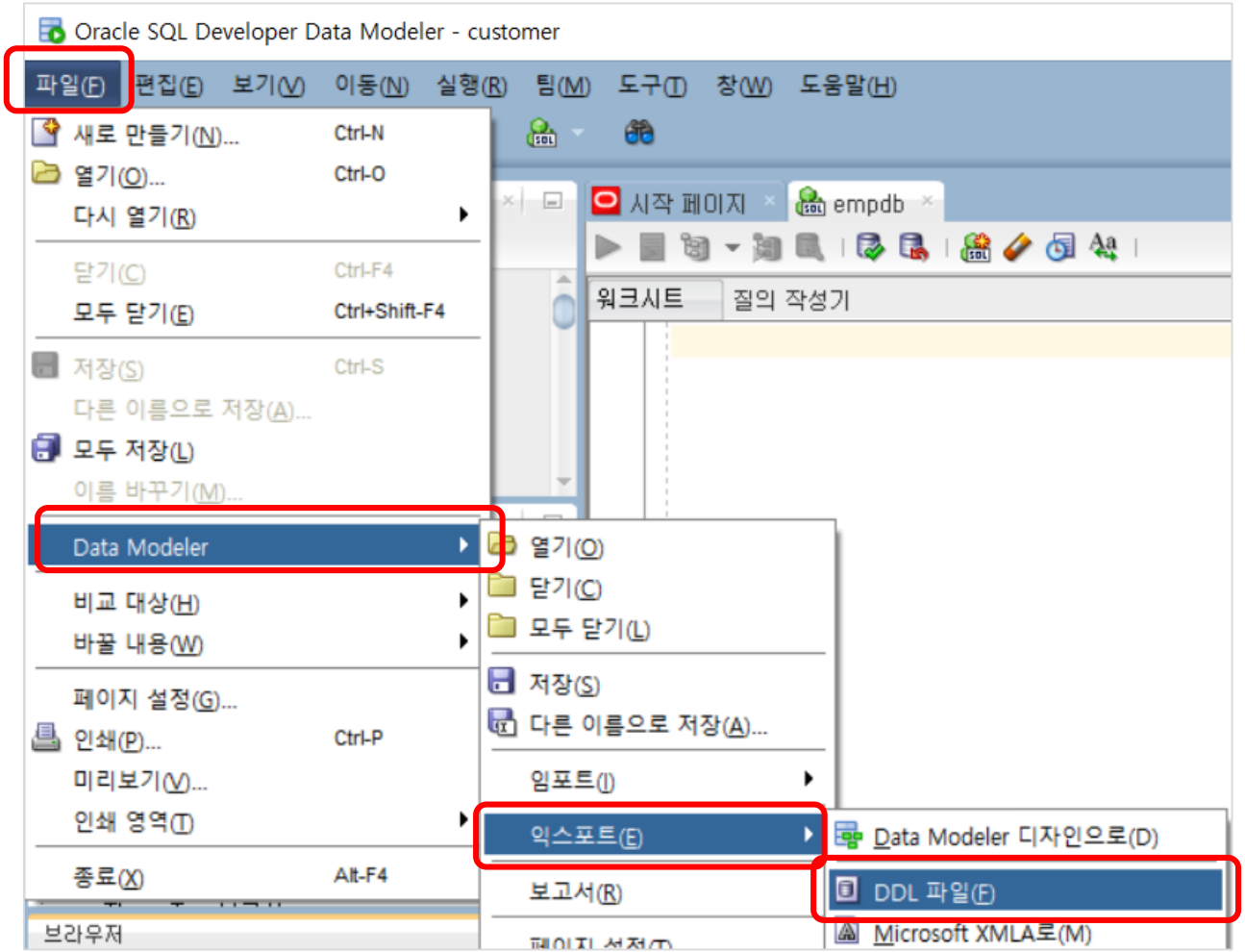






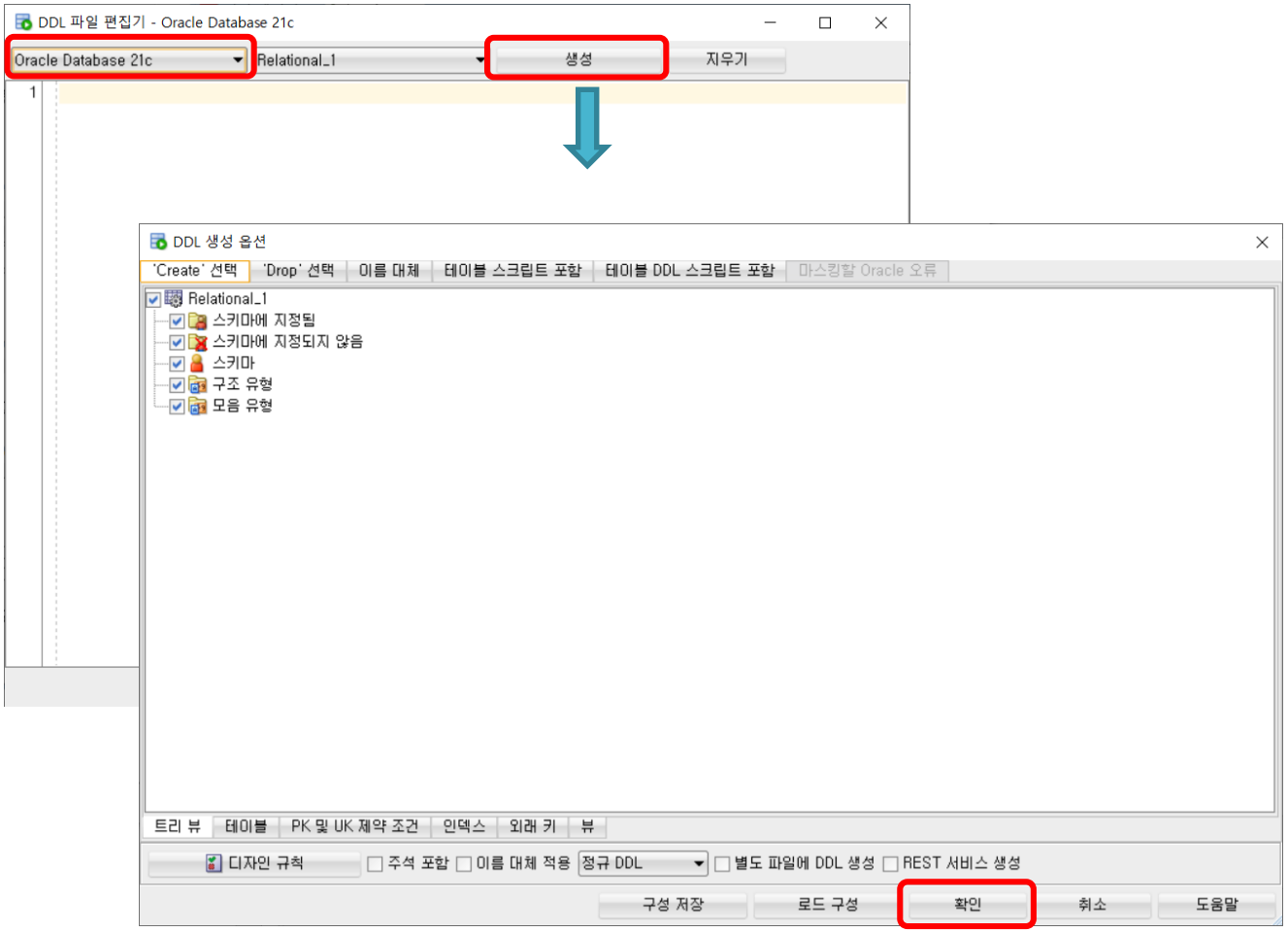
실습. Data Modeler를 이용한 순공학, 역공학 작업

- ④ 메인 메뉴에서 [파일]→[Data Modeler]→[익스포트]→[DDL 파일]의 순서로 선택한다.



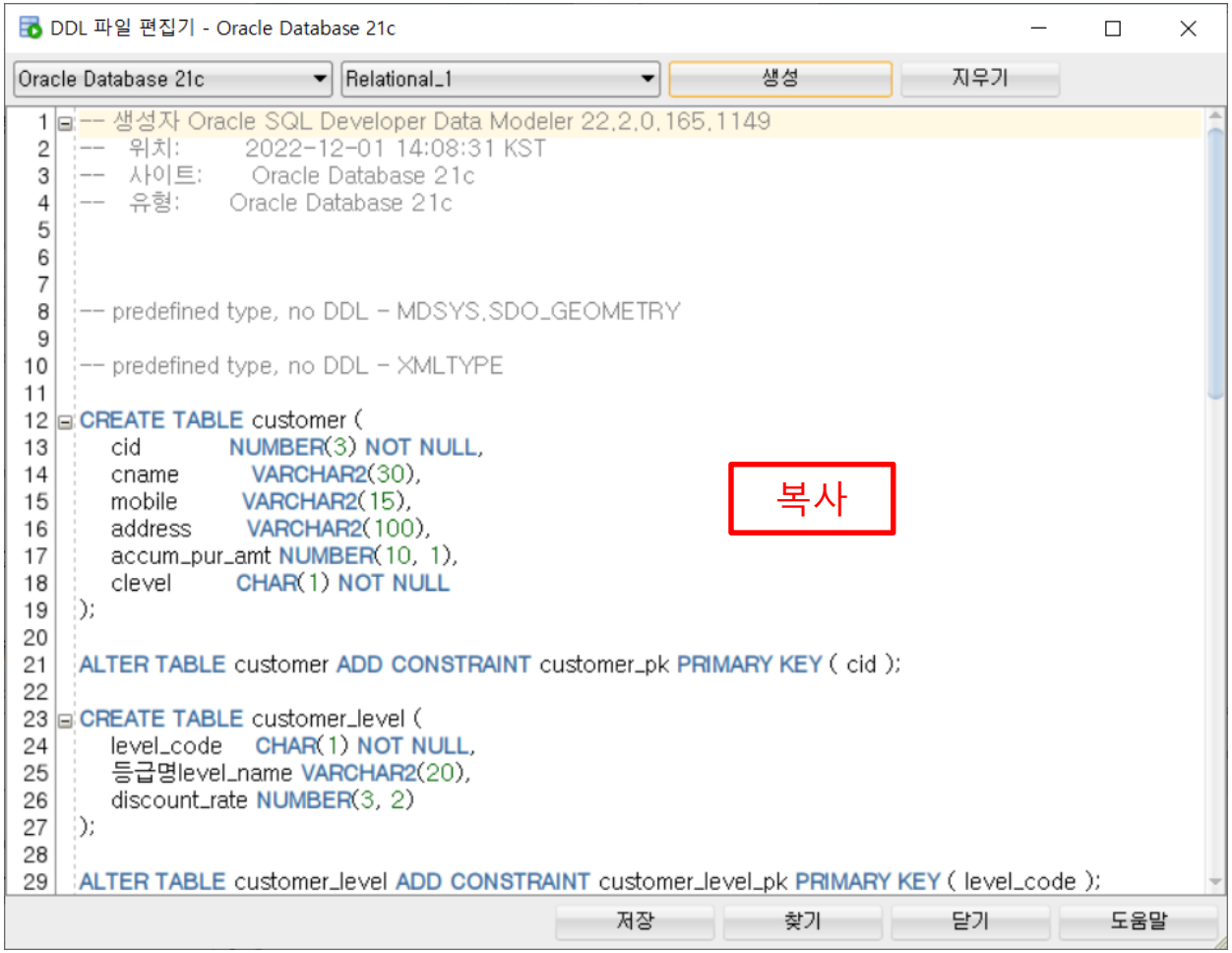
실습. Data Modeler를 이용한 순공학, 역공학 작업

- ⑤ DDL 파일 편집기 창에서 Oracle database 21c 선택 후 [생성] 버튼을 클릭하면 DDL 생성 옵션 화면이 표시되는데 옵션 수정 없이 [확인] 버튼을 클릭한다.

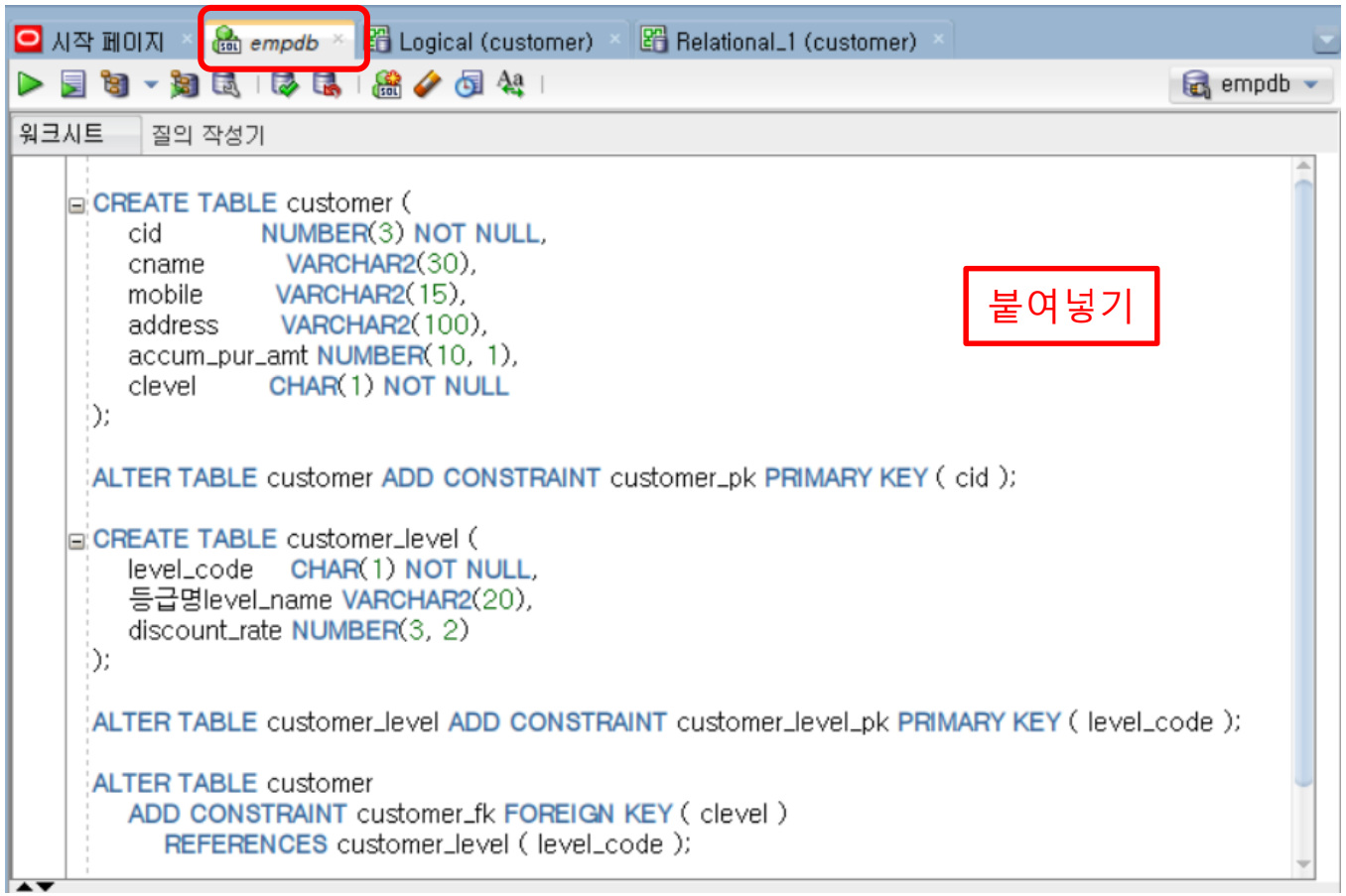


실습. Data Modeler를 이용한 순공학, 역공학 작업

- ⑥ DDL 편집기 화면에 테이블을 생성하기 위한 SQL 코드가 표시되는데 이를 복사하여 SQL 실행창에 붙여 넣기를 한다. (복사할 때 주석문은 생략해도 된다)

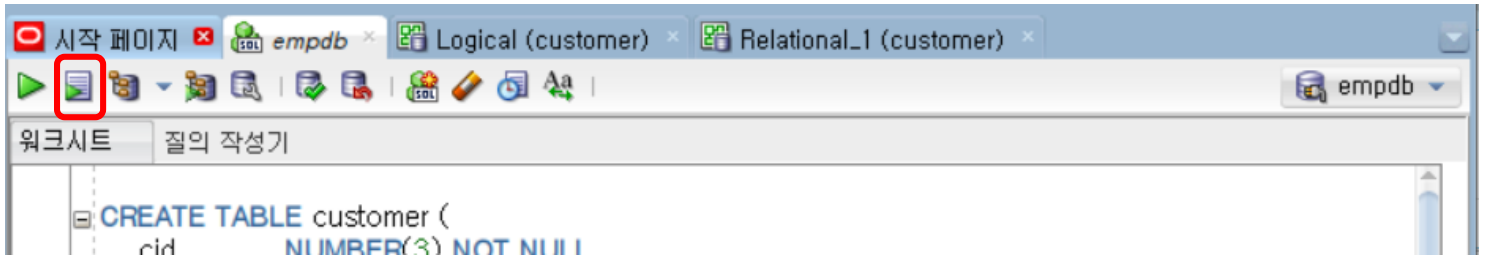


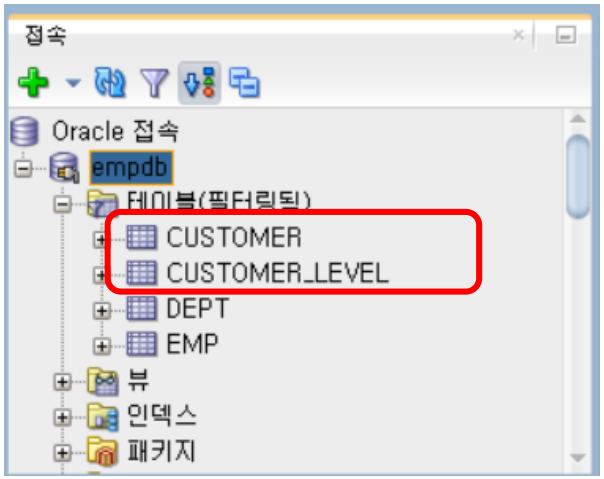
실습. Data Modeler를 이용한 순공학, 역공학 작업



실습. Data Modeler를 이용한 순공학, 역공학 작업

- ⑦ 이제 붙여 넣기를 한 SQL문을 실행하면 테이블들이 생성된다. 스크립트 실행 아이콘을 클릭하면 SQL편집창의 여러 SQL문을 한번에 실행할수 있다. 실행 결과는 접속창에서 확인한다,

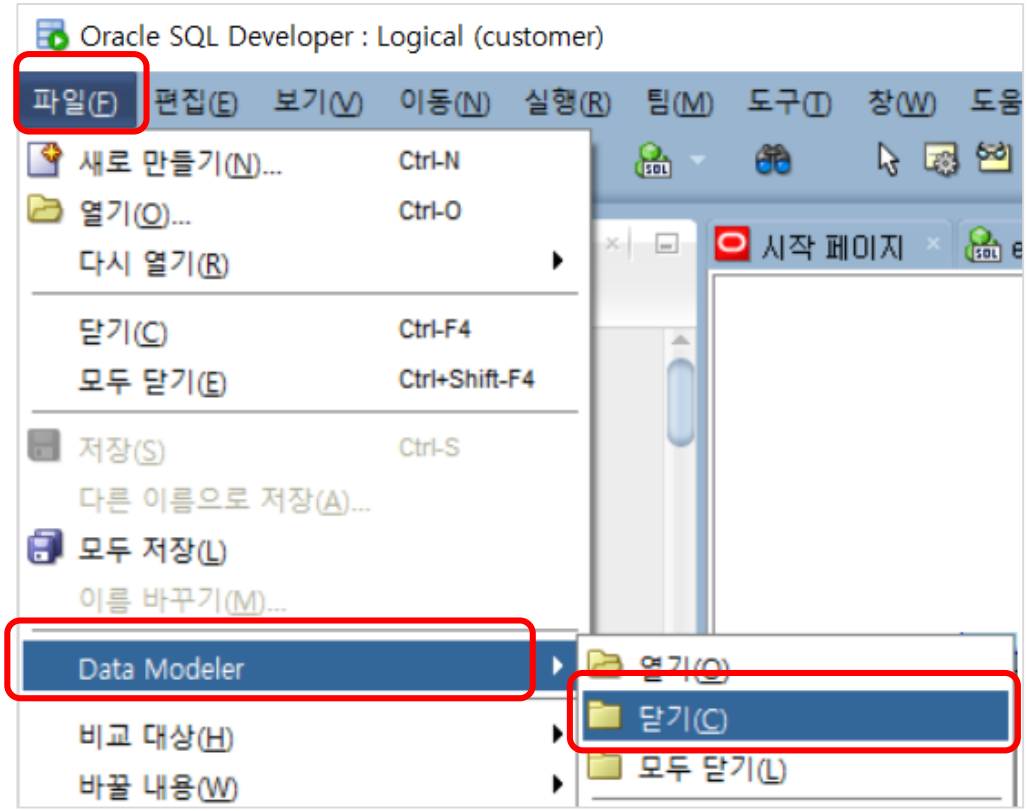




실습. Data Modeler를 이용한 순공학, 역공학 작업

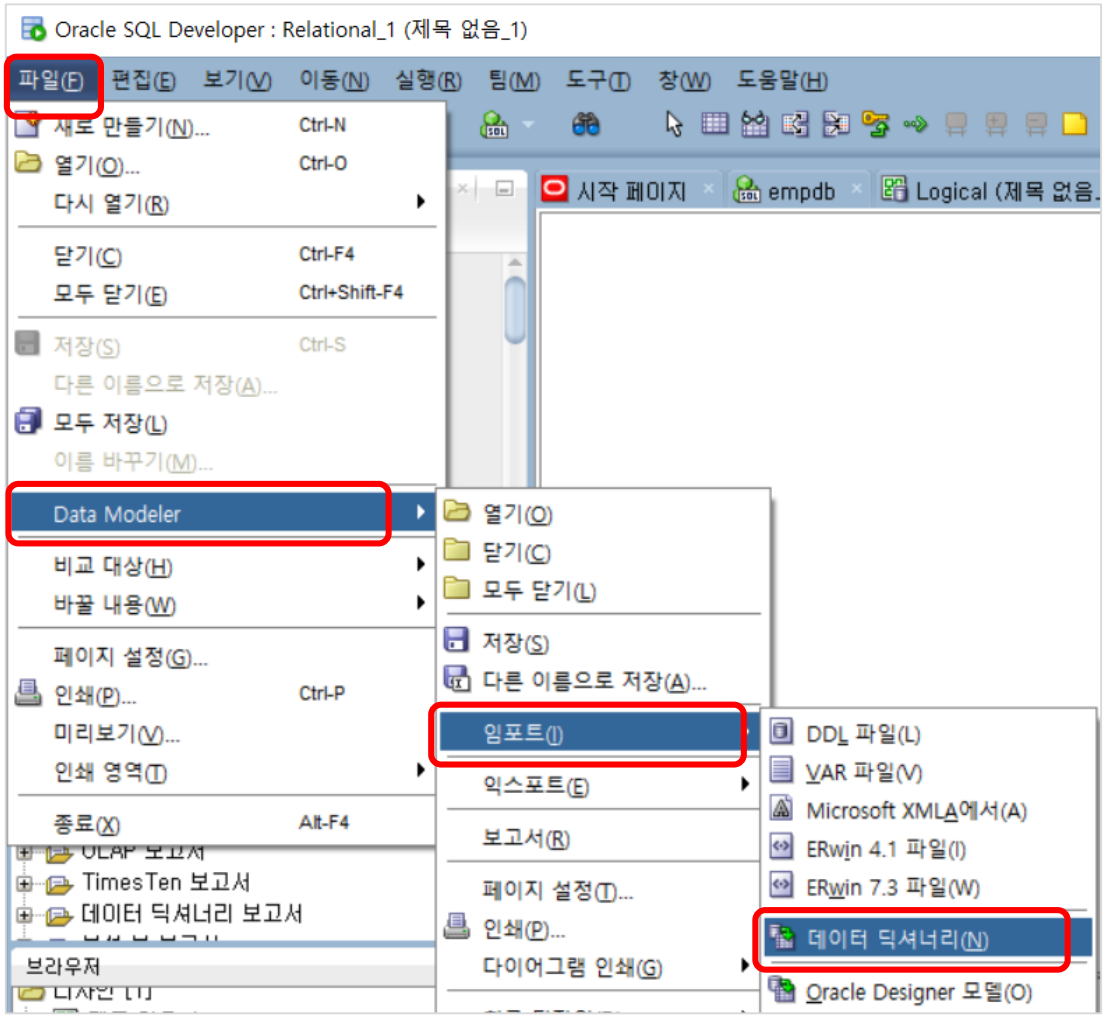
- **역공학 실습**
  - emppdb의 내용을 가져와서 물리적 ERD로 작성

  - ①새로운 ERD를 작성하기 위해서 기존에 열려있는 ERD는 닫도록 한다. 메인 메뉴에서 [파일]→[Data Modeler]→[닫기]의 순서로 선택한다.



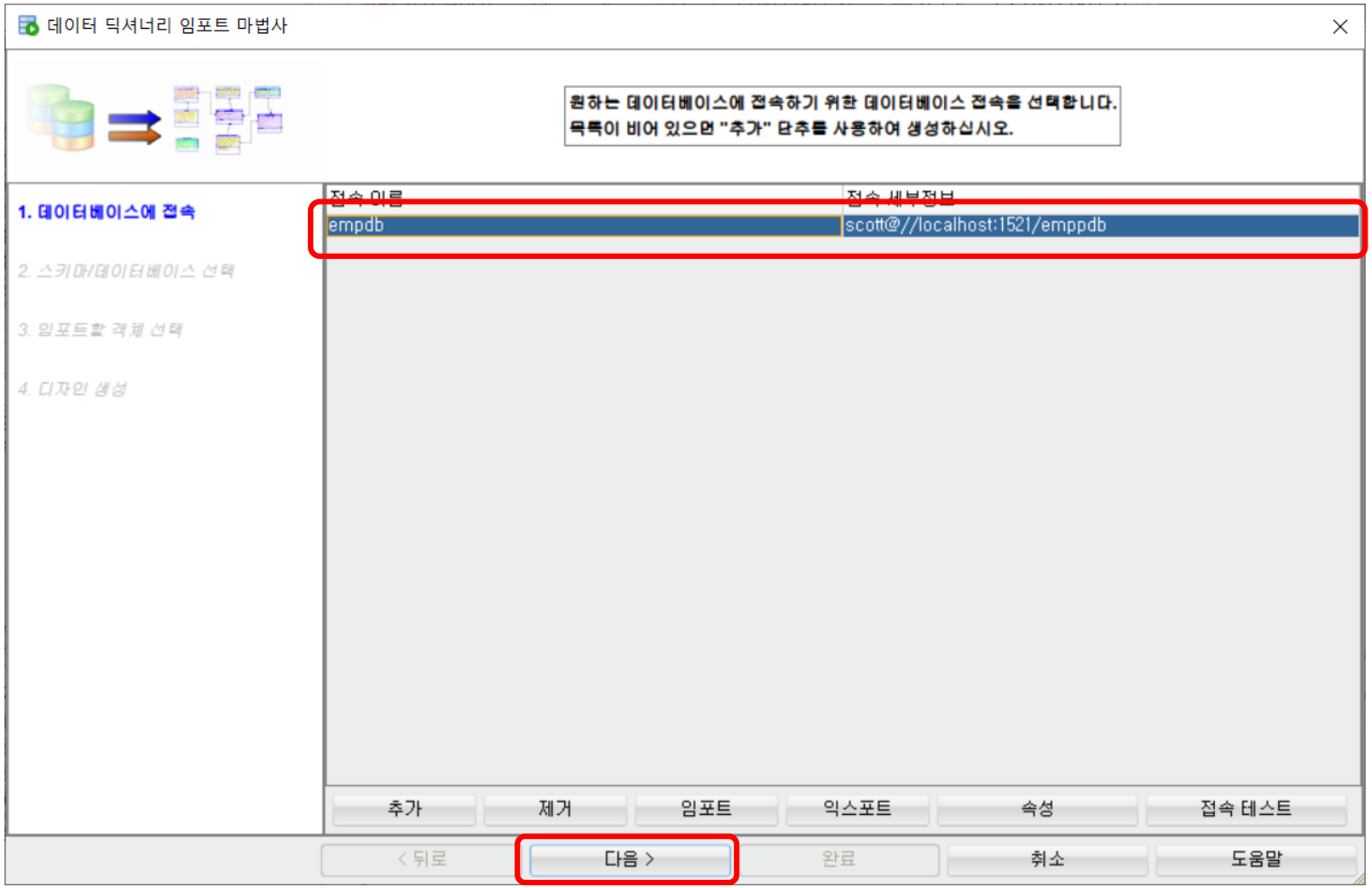
실습. Data Modeler를 이용한 순공학, 역공학 작업

- ② 메인메뉴에서 [파일]→[Data Modeler]→[임포트]→[데이터 딕셔너리]의 순으로 선택한다.



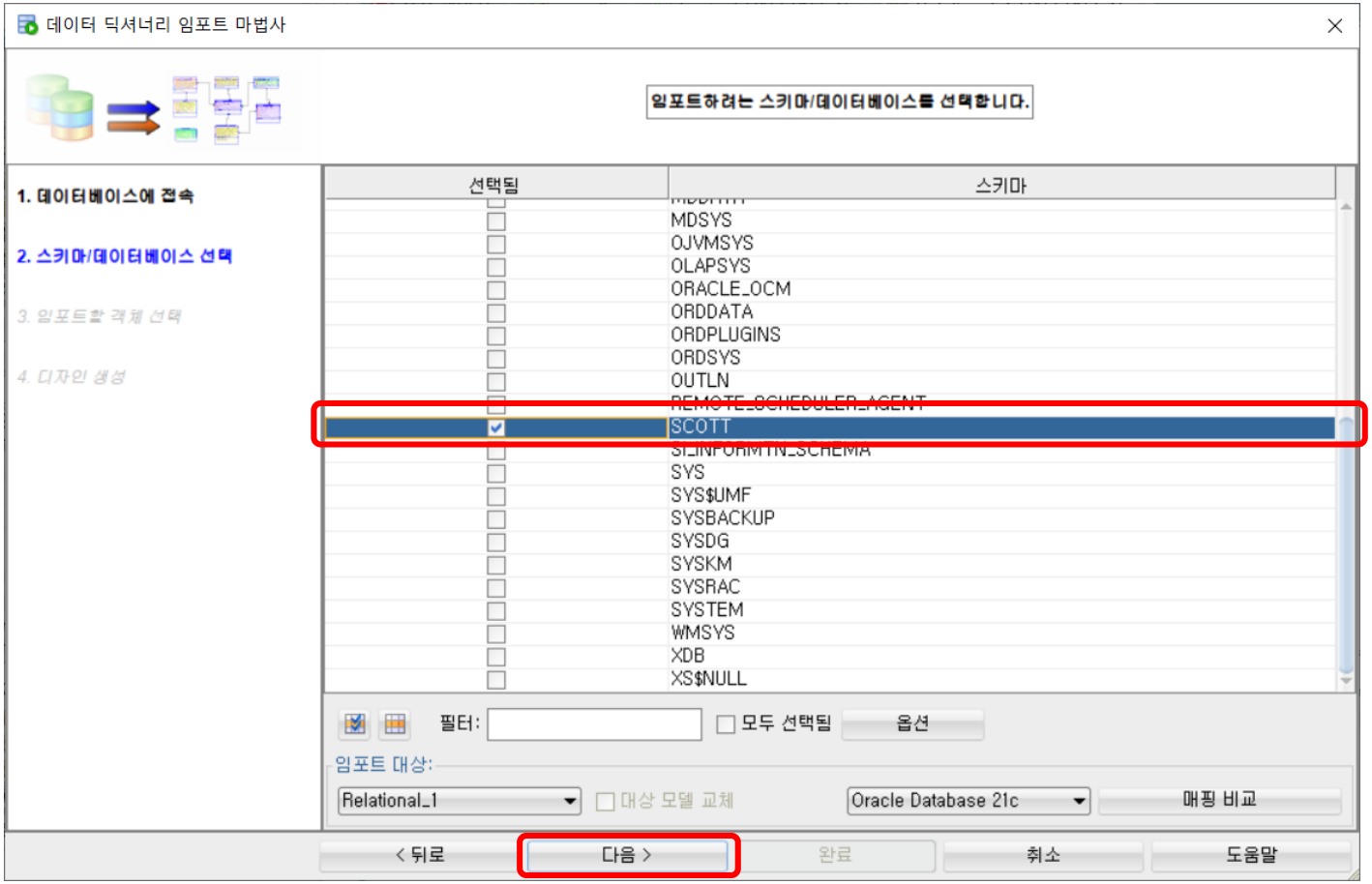
실습. Data Modeler를 이용한 순공학, 역공학 작업

- ③ 데이터 딕셔너리 임포트 마법사가 실행되면 데이터베이스에 접속하기 위한 접속이름을 선택하고 [다음]을 클릭한다.



실습. Data Modeler를 이용한 순공학, 역공학 작업

- ④ 스키마/데이터베이스 선택 단계에서 SCOTT를 선택하고 [다음]을 클릭한다. SCOTT가 생성한 데이터베이스를 선택한다는 의미이다.

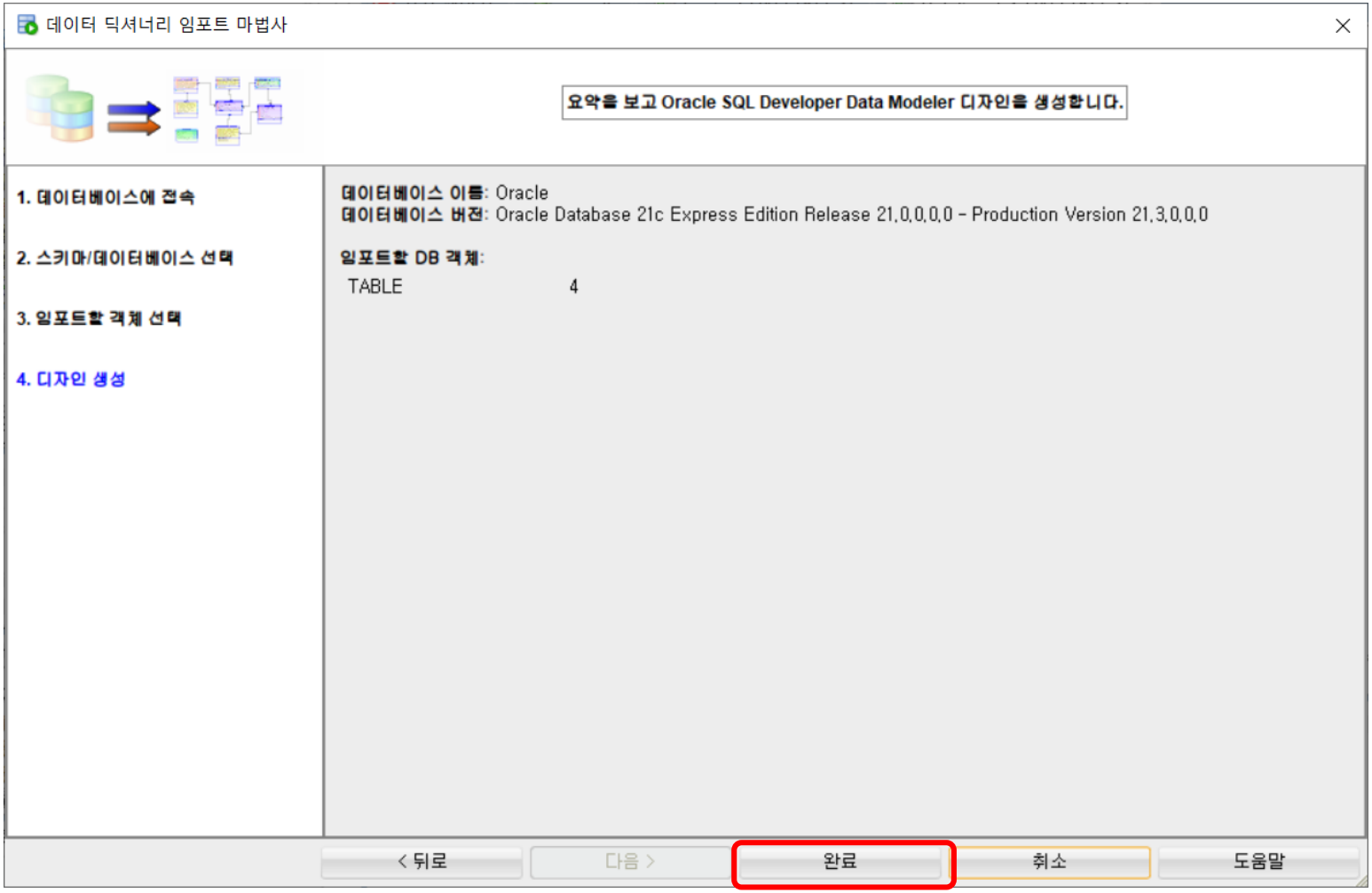


실습. Data Modeler를 이용한 순공학, 역공학 작업

- ⑤ 임포트할 객체 선택 단계에서 필요한 객체를 선택한 후 [다음]을 클릭한다. (여기서는 모든 객체를 선택하였다)

실습. Data Modeler를 이용한 순공학, 역공학 작업

- ⑥ 디자인 생성 단계에서 [완료]를 클릭하여 임포트 작업을 마무리한다.



실습. Data Modeler를 이용한 순공학, 역공학 작업

- ⑦ 임포트 결과와 작업 로그 화면이 같이 표시되는데 작업 로그를 닫으면 다음과 같이 물리적 ERD를 확인할 수 있다.