- 놀이공원 데이터베이스 설계 -



**과 목: 데이터베이스**

**학 과: 컴퓨터공학과**

**담당교수: 허훈식 교수님**

**이 름: 박다수, 정지운, 윤혁진, 김지웅**

**제 출 일: 2019.12.02**

목 차

1. 업무 및 요구사항 분석 **……………………………………………** Page 3

1-1 개요

1-2 에버랜드의 구조(데이터 및 기능 요구)

2. 개념적 설계**……………………………………………………………** Page 5

2-1 개념적 설계란

2-2 엔티티 타입

2-3 관계와 관계 타입

2-4 최종 ER 스키마 다이어그램

3. 논리적 설계**…………………………………………………………** Page 18

3-1 논리적 설계란

3-2 정규화 와 정규형

3-3 놀이공원 정규화 및 테이블 목록

4. 물리적 설계**…………………………………………………………** Page 22

4-1 물리적 설계란?

4-2 저장 레코드 형식 설계

4-3 뷰와 인덱스

5. 구현 : 기능에 대한 질의문**……………………………………** Page 26

5-1 VIEW 생성 질의 및 응용

5-2 인덱스 구현 질의문

5-3 트랜잭션 구현 질의문

6. 느낀점 : 프로젝트를 마치며**…………………………………** Page 37

**1. 업무 및 요구사항 분석**

**1-1 개요**

에버랜드를 기반으로, 놀이동산 전반적인 업무를 기록 분석하고, 데이터 관리 시스템 개발을 목표로 한다.

**1-2 에버랜드의 구조(데이터 및 기능 요구)**

**●** 놀이기구 : 각각의 존에는 서로 다른 놀이기구들이 여러가지 배치되어 있다.

놀이기구에는 놀이기구를 식별하기 위한 놀이기구 번호와, 놀이기구 이름, 탑승자의 안전을 위한 키 제한 안내 정보, 어느 존에 놀이기구가 배치되어있는지 알기 위해서, 존을 식별할 수 있는 존 번호가 필요하다.

※기능 요구: 놀이기구는 하나의 존을 무조건 가져야 한다.

**●** 시설안내 : 놀이공원에는 존마다 다양한 편의시설(매표소, 손님 서비스 센터, 모유시설, 인포메이션 등)이 존재한다. 각 편의시설들은, 놀이공원에서 관리하기 쉽도록 편의시설 번호가 주어져 있고, 편의시설 이름, 편의시설 이 어느 존에 위치되어 있는지 알기 위한 존번호 같은 정보를 필요로 한다.

※기능 요구: 편의시설은 하나의 존마다 존재한다.

**●** 존 : 놀이공원은 5개의 존(유러피안 어드벤쳐, 매직랜드, 주토피아, 글로벌 페어, 아메리칸 어드벤처)구역으로 이루어져 있다. 존은 놀이공원에서 알아보기 편하도록, 고유 존 번호가 저장 되어 있고, 존 이름, 존이 얼마나 큰지를 나타내는 존 면적의 정보가 존재한다.

※기능 요구: 여러개의 놀이기구를 담당한다, 여러개의 편의시설이 존재한다, 하나의 존에는 여러개의 특별체험과 엔터테인먼트가 있다, 존은 티켓을 구매한 고객만이 갈 수 있다.

**●** 사원 : 놀이공원에는 7개의 부서가 존재하며, 각 부서에 사원들이 배치된다. 부서는 취직하는사원들의 사원 번호와, 사원 이름, 사원의 개인 핸드폰 번호, 그 사원이 일할 존 번호, 부서 번호를 저장 한다.

※기능 요구: 사원은 하나의 부서에서만 일해야 한다, 부서를 관리하는 매니저가 한명씩 있다.

**●** 부서 : 놀이공원은 7개의 부서가 존재한다( 어트렉션 서비스, 라이프 가드, 상품 판매, 패스트푸드, 안내/렌탈 서비스, 공연기술 보조, 그린 서비스 ) 각 부서는 고유한 부서번호를 가지고, 부서 이름, 부서의 전화번호를 가지며, 해당 부서의 매니저 직원번호와, 그 매니저의 임명 날짜에 대한 정보를 저장한다.

※기능 요구: 하나의 부서에는 매니저가 한명있다, 일할 여러명의 사원들을 데리고 있을 수 있다.

**●** 엔터테인먼트 존: 놀이공원의 각각의 존 구역들은, 엔터테이먼트 즐길거리(퍼레이드, 뮤지컬 등)을 개최하는데 각각의 엔터테이먼트는 고유 번호가 존재하고, 어느 존에서 열리는지 인식하기 위해서, 존 번호와, 엔터테이먼트 시작 시간을 저장한다.

※기능 요구: 엔터테이먼트를 무조건 개최하지 않는 존 구역이 있을수도 있다.

**●** 티켓: 놀이공원에 입장하기 위해선, 티켓이 무조건 필요하다. 티켓은 3가지 종류가 있으며(주간, 야간, 2일권) 각각의 가격은 다르다. 티켓 종류에 해당하는 티켓 고유 번호가 존재한다.

※기능 요구: 고객은 티켓의 종류중 하나를 무조건 구매해야 한다.

**●** 고객: 놀이공원에 입장하는 손님은, 티켓을 발급받으면서, 티켓 종류에 따른 티켓 고유 번호를 받게 되고, 고유한 손님 번호와 손님 이름을 놀이공원측에 주게 된다. 또한 카드사에 따라서 할인된 표 가격을 얻을 수도 있다. 숙박시설을 예약한 손님은, 방 예약 고유번호도 저장한다. 또한 놀이기구를 예약하려면 키 제한보다 키가 커야되는데 이를 위해서 고객의 키 정보도 저장받는다.

※기능 요구: 고객은 모두 카드사 번호를 가지고 있다, 고객은 1개의 티켓을 구매해야 한다. 고객은 숙박종류중 하나를 택할 수 있다, 고객은 구매한 표에 따라 이용시간 만큼 다양한 존을 체험할 수 있다, 고객은 숙박시설의 여러방을 예약할 수 있다.

**●** 숙박시설: 놀이공원 쪽에는 놀이공원 측에 위치한 글램핑 숙박시설과, 외부에 존재하는 케빈 호스텔, 엔터 호스텔, 에버 홈브릿지 숙박건물이 존재하며 고객이 예약하여 숙박을 할 수 있다.

※기능 요구: 숙박시설은 예약한 고객이 고른 방만 사용할 수 있다.

**●** 특별체험: 놀이공원의 존 구역들은, 특별한 체험들(사파리 스페셜 투어, 장미원, 뮤직 가든) 같은 것들을 오픈할 수 있다. 특별 체험은 각각의 고유 번호를 발급받는다.

※기능 요구: 존은 특별체험을 개최하거나 하지 않을 수 있다.

**●** 카드사: 놀이공원에서 티켓을 결제할 때, 고객의 카드사에 따라서 할인을 얻을 수 있다. 카드사 번호를 저장하며, 그에 따라 할인 비율이 저장된다.

※기능 요구: 카드사에 따른 혜택이 존재한다.

**●** 예약: 놀이공원에는 핸드폰 모바일을 통해 놀이기구를 우선 예약해서, 줄을 기다리는 사람들보다 빨리 탈 수 있는 프리패스 기능이 존재한다. 우선예약을 하는 고객은, 고유 번호를 발급받고, 그 고유 번호는 고객 번호 와 놀이기구 번호를 저장한다. (고유 번호: 100번부터 시작)

※기능 요구: 고객은 놀이기구를 핸드폰으로 우선예약할 수 있다, 한 개의 놀이기구를 우선 예약한 고객은 다른 놀이기구를 예약하지 못한다.

**2. 개념적 설계**

**2-1 개념적 설계란**

**●** 개념적 설계란 모든 물리적인 사항과 독립적으로 한 조직체에서 사용되는 정보의 모델을 구축해가는 과정을 뜻한다.

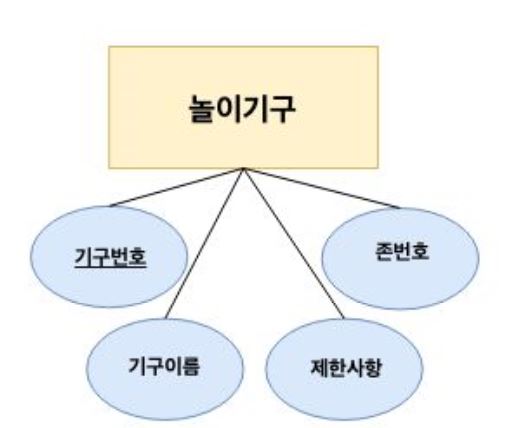
**●** 사용자들의 요구사항 명세로부터 개념적 스키마가 만들어지며, 높은 추상화 수준의 데이터 모델을 기반으로 정형적인 언어의 데이터 구조를 명시한다, 대표적인 모델이 ER모델을 말한다.

**●** 개념적 설계의 단계에서 엔티티 타입, 관계 타입, 애트리뷰트들을 식별하고, 애트리뷰트들 의도 메인을 결정한다. 그 후 후보 키와 기본 키 애트리뷰트들을 결정한다. 마지막으로 완성된 개념적 스키마(ER 스키마)는 ER 다이어그램으로 표현된다.

**2-2 엔티티 타입**

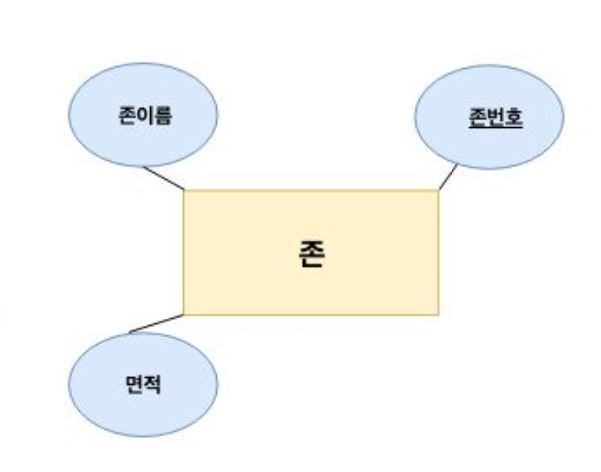
**●** 1-2 에버랜드의 구조(데이터 및 기능 요구)의 정보를 바탕으로 놀이공원의 엔티티를 선정하였다.

**●** 놀이기구(ATTRACTION)



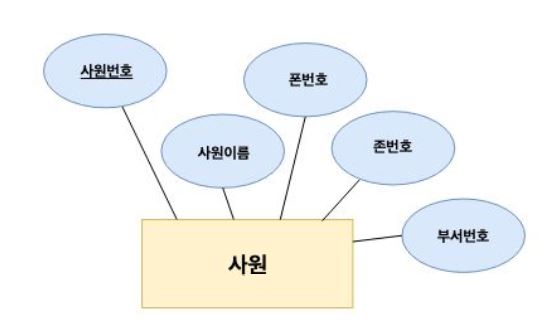
* 기구번호(ATTNO),기구이름(ATTNAME),키제한사항(RESTRICTION),존번호(ZONENO)
* 기본키: 기구번호(ATTNO)
* 외래키:존번호(ZONENO)=> 존(ZONE)의 (ZONENO)참조

**●** 존(ZONE)



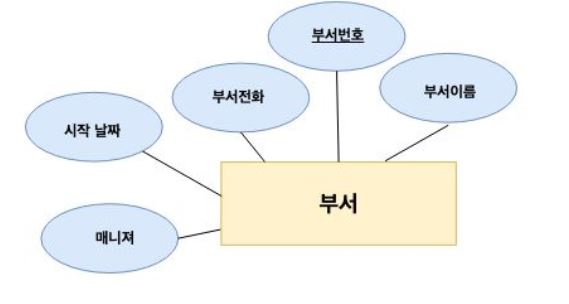
* 존번호(ZONENO), 존이름(ZONENAME), 존면적(AREA)
* 기본키: 존번호(ZONENO)

**●** 사원(EMPLOYEE)



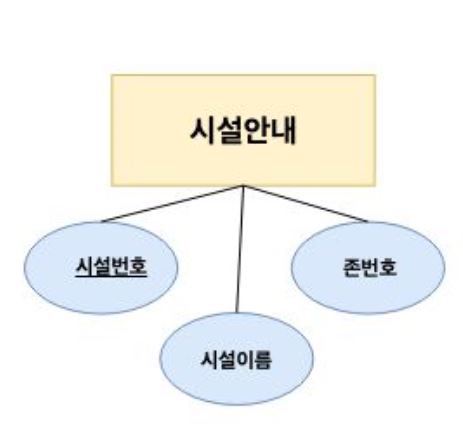
* 사원번호(EMPNO), 사원이름(EMPNAME), 사원 폰번호(PHONE), 존번호(ZONEENO), 부서번호(DEPTNO)
* 기본키: 사원번호(EMPNO)
* 외래키: 존번호(ZONENO) => 존(ZONE)의(ZONENO)참조, 부서번호(DEPTNO) => 부서(DEPARTMENT)의 (DEPTNO) 참조

**●** 부서(DEPARTMENT)



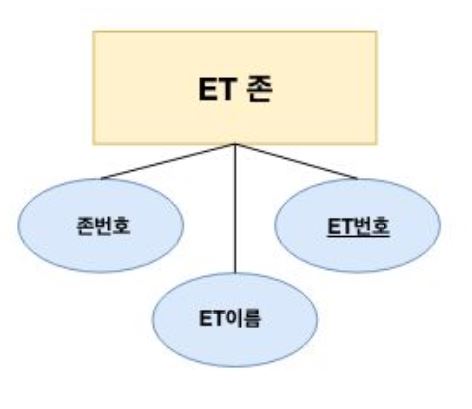
* 매니저 번호(MANAGER), 매니저 시작 날짜(STARTDATE), 부서전화(DEPTPHONE), 부서번호(DEPTNO), 부서이름(DEPTNAME)
* 기본키: 부서번호(DEPTNO)

**●** 시설안내(PUBLICFACIL)



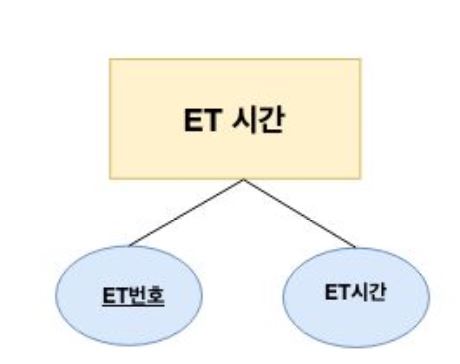
* 시설번호(PUBNO), 시설이름(PUBNAME), 존번호(ZONENO)
* 기본키: 시설번호(PUBNO),
* 외래키: 존번호(ZONENO) => 존(ZONE)의 (ZONENO) 참조

**●** 엔터테이먼트 존(ENTERTAINMENT ZONE)



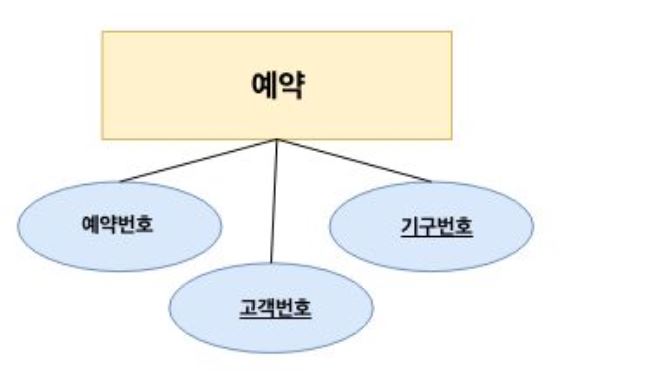
* 존번호(ZONENO), ET이름(ENTERNAME), ET번호(ENTERZONENO)
* 기본키: ET번호(ENTERZONENO)
* 외래키: 존번호(ZONENO) => ZONE의 (ZONENO) 참조

**●** 엔터테이먼트 시간(ENTERTAINMENT TIME)



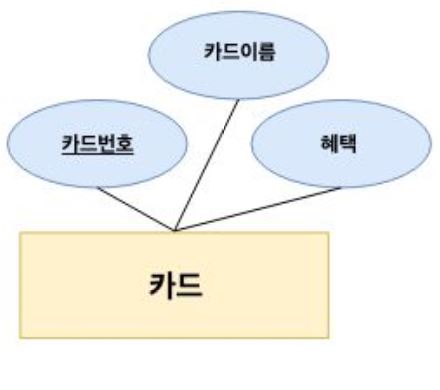
* ET시간 번호(ENTERTIMENO), ET시간(ENTERTIME)
* 기본키: ET시간 번호(ENTERTIMENO)
* 외래키: ET시간 번호(ENTERTIMENO) => 엔터테이먼트존(ENTERTAINMENTZONE)의 (ENTERZONENO) 참조

**●** 예약(RESERVE)



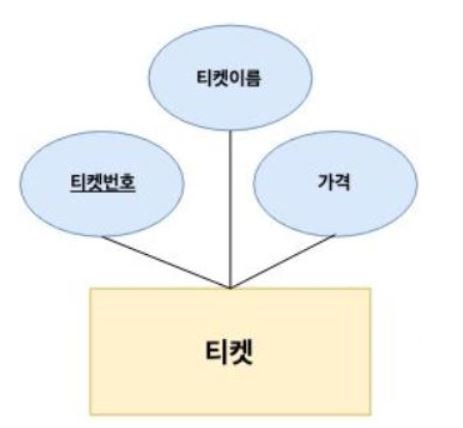
* 예약번호(ATTNO), 고객번호(CUSTNO), 기구번호(RESERVENO)
* 기본키: 고객번호(CUSTNO)
* 외래키: 기구번호(ATTNO) => 놀이기구(ATTRACTION)의 (ATTNO) 참조

**●** 카드(CARD)



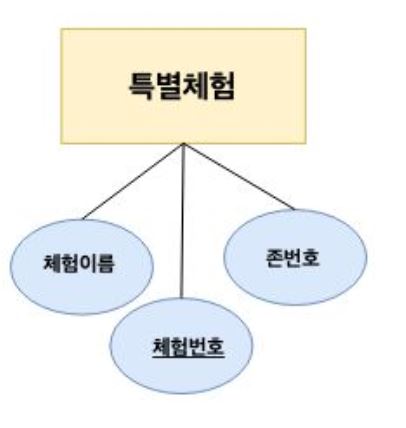
* 카드번호(CARDNO), 카드이름(CARDNAME), 카드혜택(BENEFIT)
* 기본키: 카드번호(CARDNO)

**●** 티켓(TICKET)



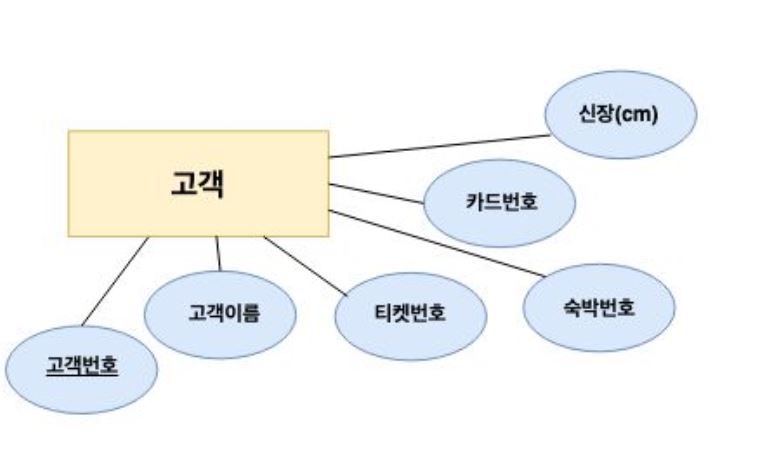
* 티켓번호(TICKETNO), 티켓이름(TICKETNAME), 티켓가격(TICKETPRICE)
* 기본키: 티켓번호(TICKETNO)

**●** 특별체험(SPECIAL)



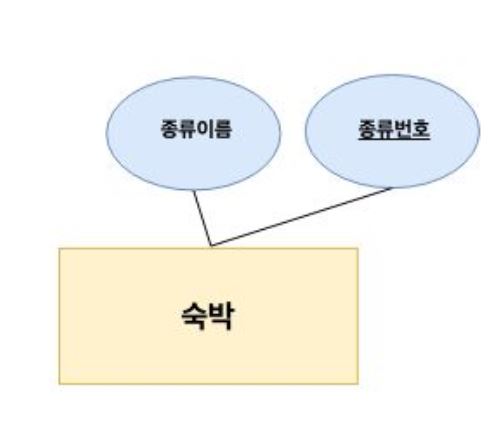
* 체험이름(SPECIALNAME), 체험번호(SPECIALNO), 존번호(ZONENO)
* 기본키: 체험번호(SPECIALNO)
* 외래키: 존번호(ZONENO) => 존(ZONE)의 (ZONENO) 참조

**●** 고객(CUSTOMER)



* 고객번호(CUSTNO), 고객이름(CUSTNAME), 티켓번호(TICKETNO), 숙박번호(ROOMSTNO), 카드번호(CARDNO), 고객신장(CUSRESTICTION)
* 기본키: 고객번호(CUSTNO)
* 외래키: 카드번호(CARDNO) => 카드(CARD)의 (CARDNO)참조, 티켓번호(TICKETNO) => 티켓(TICKET)의 (TICKETNO) 참조, 숙박번호(ROOMSTNO) => 숙박(ROOM)의 (ROOMSTNO) 참조

**●** 숙박(ROOM)



* 숙박종류 이름(ROOMNAME), 숙박종류 번호(ROOMSTNO)
* 기본키: 숙박종류 번호(ROOMSTNO)

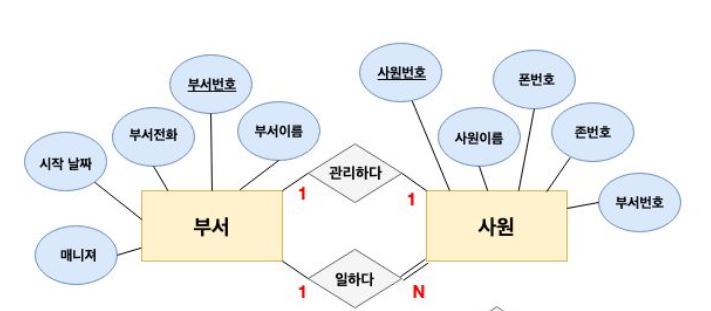
**2-3 관계와 관계 타입**

**●** 관계는 엔티티들 사이에 존재하는 연관이나 연결로서, 두 개 이상의 엔티티 타입들 사이의 사상이다.

**●** 요구사항 기재시, 동사는 ER 다이어그램에서 관계로 표현된다.

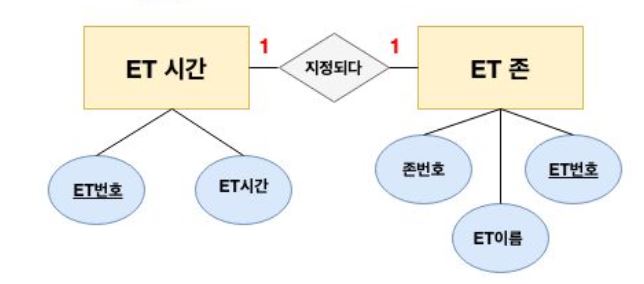
**●** ER 다이어그램에서 다이아몬드 모양으로 관계를 표시한다.

**●** 부서와 사원 간의 관계



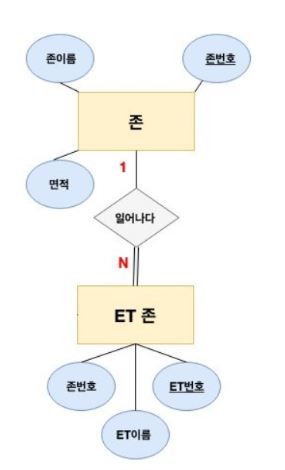
* 부서와 사원은 일하다와 관리하다의 관계를 맺고 있다.
* 부서는 일할 여러명의 사원을 관리할 수 있다.
* 사원은 하나의 부서에서만 일해야 한다.
* 각 부서마다 직원을 관리하는 최소 한명의 매니저들이 있다.
* 사원과 부서는 1:1 관계이며 직원은 부서에 전체참여된다.

**●** 엔터테이먼트 시간과, 엔터테이먼트 존의 관계



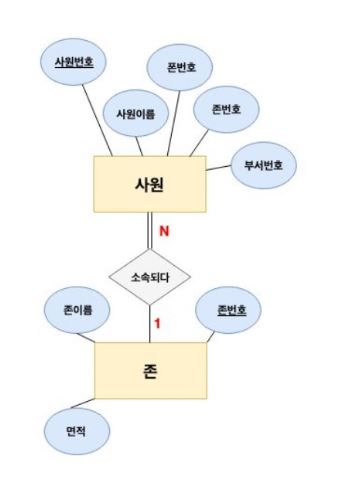
* 엔터테이먼트와 엔터테이먼트 존은 지정되다의 관계를 가지고 있다.
* 각각의 엔터테이먼트 존은 체험할 수 있는 엔터테이먼트 공연, 체험들을 계획할 수도 있고 하지 않을 수 있다.
* 엔터테이먼트 존에서, 엔터테이먼트를 계획하면, 엔터테이먼트 시간이 지정된다. (1:1)의 관계

**●** 존과, 엔터테이먼트의 관계



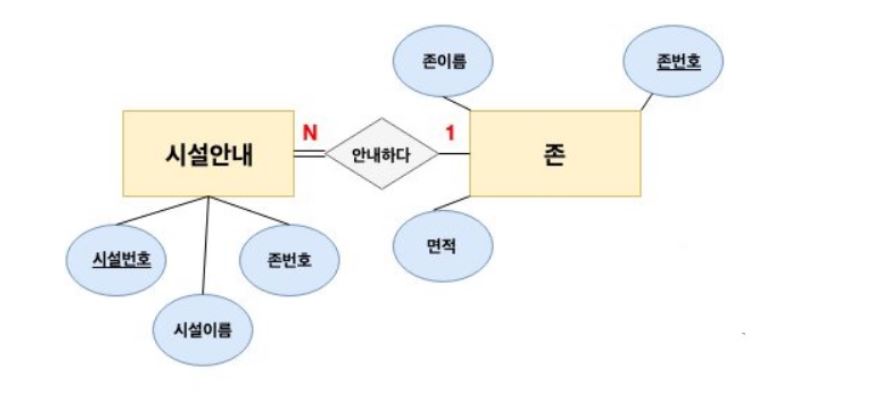
* 존과 엔터테이먼트 존은 일어나다의 관계를 가지고 있다.
* 엔터테이먼트 존은 하나의 존에 전체참여 하며. 엔터테이먼트 존을 가지는 존은 엔터테이먼트 행사가 일어나게 된다.
* 각각의 엔터테이먼트 존은, 여러 존에 들어갈 수 없고 하나의 존에 참여한다 (1:N관계, 전체참여: 부분참여)

**●** 사원과, 존의 관계



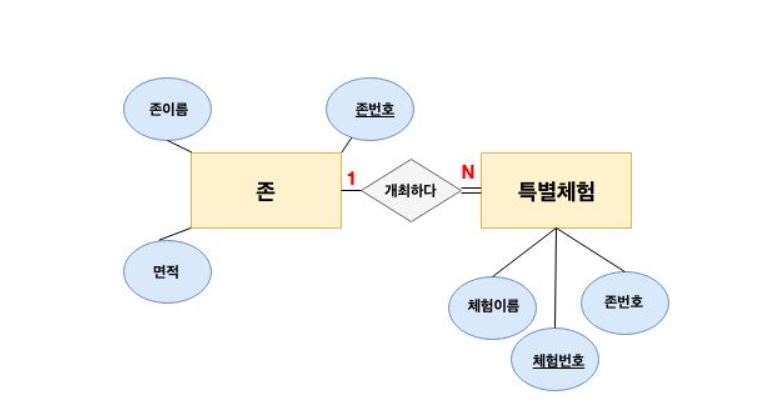
* 사원과 존은 소속되다의 관계를 가지게 된다
* 각각의 사원은 하나의 존에 전체 참여하며, 사원이 없는 존이 없어야 한다.
* 같은 부서라도 다른 존에 배치될 수 있다.
* 사원과 존(N:1, 전체참여)

**●** 시설안내와, 존의 관계



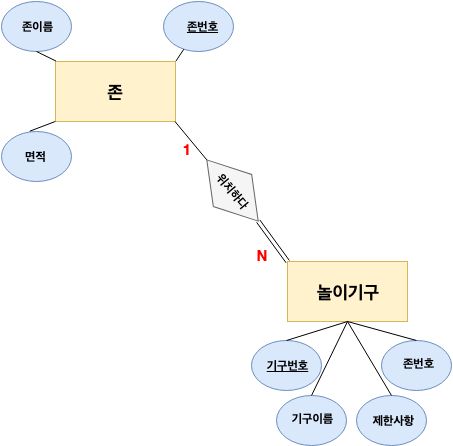
* 시설안내와 존은, 안내하다의 관계를 가지고 있다.
* 시설안내는 각각의 존에 전체참여 하며, 시설안내가 없는 존이 없어야 한다.
* 시설안내와 존(N:1, 전체참여)

**●** 존과, 특별체험의 관계



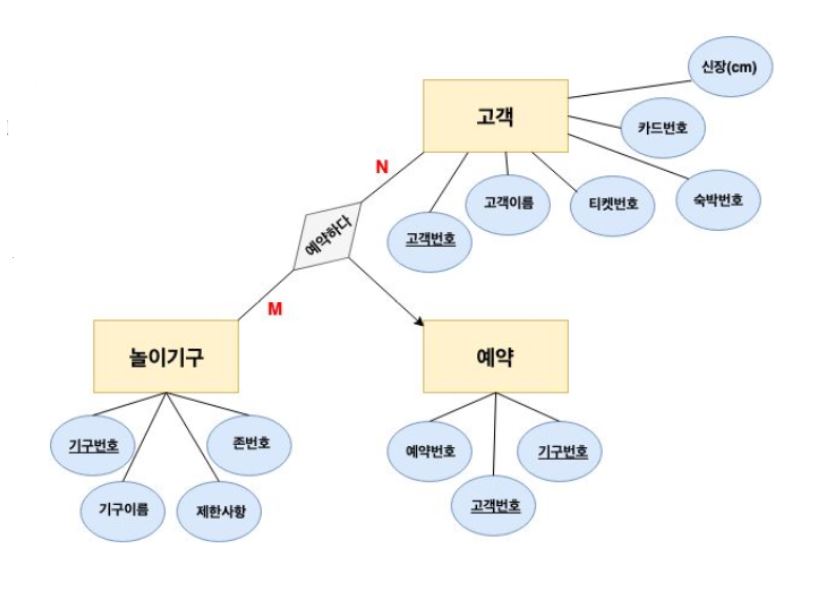
* 존과 특별체험은 개최하다의 관계를 가진다.
* 각각의 존들은, 페스티벌과 같은 특별체험을 개최하거나 하지 않을 수 있으며, 특별체험 은 하나의 존에 전체참여 한다.
* 존과 특별체험(1:N, 전체참여)

**●** 존과, 놀이기구의 관계

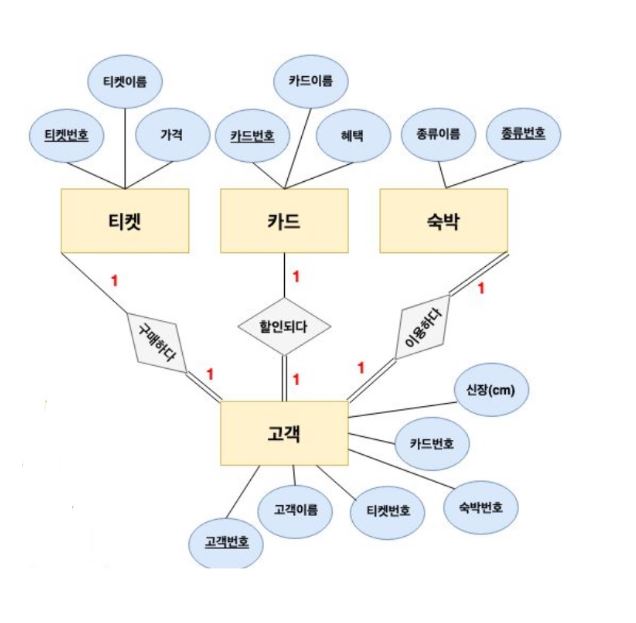


* 존과 놀이기구는 위치하다의 관계를 가진다.
* 놀이기구는 존에 전체참여하며, 존에는 놀이기구가 무조건 한 개 씩은 있어야 한다.
* 존과 놀이기구(1:N, 전체참여)

**●** 놀이기구, 고객, 예약의 관계

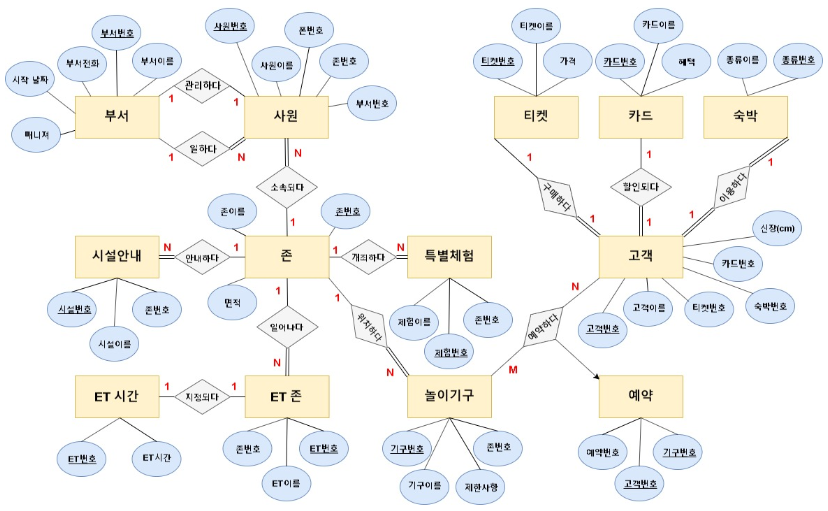


* 고객은 놀이기구를 예약한다는 의미로 엮어져 있는 관계이다.
* 고객은 여러 놀이기구를 예약해서 탈 수도 있고, 줄을 서서 기다려 탈 수도 있다.
* 예약은 복합 애트리뷰트로, 고객이 놀이기구를 예약하면, 예약에 대한 정보가 저장 된다.
* 티켓, 카드, 숙박 과 고객간의 관계



* 고객과 티켓은 구매하다의 관계를 가진다
* 고객은 하나의 티켓을 무조건 구매해야, 놀이공원에 입장할 수 있다
* 티켓과 고객은(1:1, 전체관계)
* 카드와 고객은 할인되다의 관계를 가진다
* 고객이 티켓을 구매할 시, 고객의 카드사 종류(삼성,현대..)에 따라서, 티켓에 대해 할인을 받을 수 있다.
* 카드와 고객은(1:1, 전체관계)
* 숙박과 고객은 이용하다의 관계를 가진다
* 고객은 놀이공원 내의 글램핑, 외부 숙박 시설을 예약할 수 있다. 숙박은 고객이 예약을 한 방에 더 이상 예약을 받지 못한다
* 숙박과 손님은(1:1, 전체관계)

**2-4 놀이공원의 최종 ER 스키마 다이어그램**



**3. 논리적 설계**

**3-1 논리적 설계란?**

**●** 논리적 설계란 데이터베이스 관리를 위해서, 선택한 DBMS의 데이터 모델을 사용하여논리적 스키마(외부 스키마 포함)을 생성하는 것이다.

**●** 개념적 스키마에 알고리즘을 적용하여 논리적 스키마를 생성한다. 단 관계 데이터모델을 사용하는 경우에는, ER 모델로 표현된 개념적 스키마를 관계 데이터베이스 스키마로 사용함.

**●** 데이터베이스 설계자가 요구사항 수집과 분석 후에 바로 논리적 설계 단계로 가는 경우가 있는데, 이런 경우 좋은 데이터베이스 스키마 관게가 생성되지 않는다.

**●** 관계 데이터베이스 스키마를 더 좋은 관계 데이터베이스 스키마로 변환하기 위해서 정규화 과정을 적용한다.

**3-2 정규화 와 정규형**

**●** 정규화는 원래의 릴레이션을 손실없이 분해함으로써 중복과 갱신 이상을 최소화하여 일관성과 정확성을 유지하는 과정이다. 주어진 릴레이션 스키마를 함수적 종속성과 기본 키를 기반으로 분석함

**●** 제 1 정규형 : 한 릴레이션 R이 제 1 정규형을 만족할 필요 충분 조건은 릴레이션 R의 모든 애트리뷰트가 원자값을 갖는다. 릴레이션의 모든 애트리뷰트에 다치 / 복합 애트리뷰트가 나타나지 않으면 제 1 정규형을 만족한다.

**●** 제 2 정규형 : 한 릴레이션 R이 제 정규형을 만족할 필요 충분 조건은 릴레이션 R 이 제 1 정규형을 만족하면서, 키가 아닌 모든 에트리뷰트들 R의 기본 키에 완전히 함수적으로 종속하는 것이다.

- 기본 키가 두 개 이상의 애트리뷰트로 구성되었을 경우에만 제 1 정규형이 제 2 정규형도 만족한다. 기본 키가 한 개의 애트리뷰트로 이루어진 릴레이션이 제 1 정규형을 만족하면 제 2 정규형도 만족한다.

**●** 제 3 정규형 : 한 릴레이션 R이 제 3 정규형을 만족할 필요 충분 조건은 릴레이션 R이 제 2 정규형을 만족하면서, 키가 아닌 모든 애트리뷰트들은 릴레이션 R의 기본 키에 이행적으로 종속하지 않는 것이다.

**●** BCNF : 한 릴레이션 R이 BCNF를 만족할 필요 충분 조건은 릴레이션 R이 제 3정규형을 만족하고, 모든 결정자가 후보 키어야 한다.

**3-3 놀이공원 정규화 및 테이블 목록**

**●** 데이터의 중복성과 무결성을 유지하기 위해 모든 테이블에서 정규화를 진행하였고, 제1 정규화 과정에서 기존 Entertainment릴레이션이 있었으나, ET시간에서 원자값이 아닌 복합된 값을 가지는 애트리뷰트가 있어,이를 EntertainmentZone과 EntertainmentTime 으로 분해했다. 나머지 제2정규화 과정과 제3정규화 과정에서는 큰 변화가 없어 정규화를 끝냈다.

**●** 정규화 대상



**●** 정규화 결과



**●** 모든 테이블 정규화 결과

**●** 놀이기구 테이블



**●** 부서 테이블



**●** 티켓 테이블



**●** 숙박 테이블



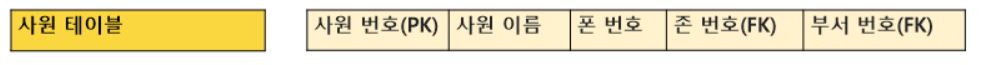
**●** 존 테이블



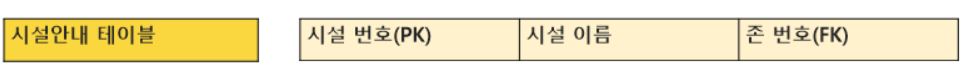
**●** 카드 테이블



**●** 사원 테이블



**●** 시설안내 테이블



**●** ET 시간 테이블



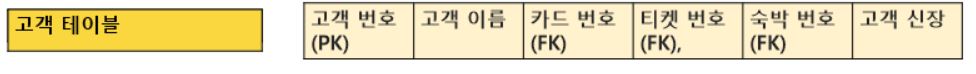
**●** ET 존 테이블



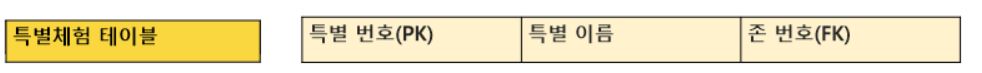
**●** 예약 테이블



**●** 고객 테이블



**●** 특별체험 테이블



**4. 물리적 설계**

**4-1 물리적 설계란?**

**●** 실제로 데이터베이스를 생성하기 위한 저장 구조와 접근 경로를 설계하는 과정이다.

**●** 사용자가 요구하는 수준의 접근 속도를 보장하고 저장 용량을 최적화 할 수 있다.

**4-2 저장 레코드 형식 설계**

**●** ATTRACTION (놀이기구)

create table attraction

(

attno number not null, attname char(25), restriction number(20), zoneno number,

primary key(attno),

foreign key(zoneno) references zone(zoneno)

);

**●** ZONE(존)

create table zone

(

zoneno number not null,

zonename char(30),

area number,

primary key(zoneno)

);

**●** EMPLOYEE (사원)

create table employee

(

empno number not null, empname char(10), phone char(13), zoneno number, deptno number,

primary key(empno), foreign key(zoneno), references zone(zoneno), foreign key(deptno),

references department(deptno)

);

**●** PUBLICFACIL(시설안내)

create table publicfacil

(

pubno number not null, pubname char(35), zoneno number, primary key(pubno),

foreign key(zoneno) references zone(zoneno)

);

**●** ENTERTAINMENTZONE (엔터테이먼트 존)

create table EntertainmentZone

(

enterZoneno number not null, entername char(30), zoneno number, primary key(enterZoneno),

foreign key(zoneno) references zone(zoneno)

);

**●** ENTERTAINMENTTIME(엔터테이먼트 시간)

create table EntertainmentTime

(

enterTimeno number not null, entertime char(10),

primary key(enterTimeno,entertime)

);

**●** RESERVE (예약)

create table reserve

(

attno number not null, custno number not null, reserveno number, primary key(attno,custno),

foreign key(attno) references attraction(attno),

foreign key(custno) references customer(custno)

);

**●** CARD (카드)

create table card

(

cardno number not null, cardname char(10), benefit char(7),

primary key(cardno)

);

**●** TICKET (티켓)

create table ticket

(

ticketno number not null, ticketname char(10), ticketprice number,

primary key(ticketno)

);

**●** SPECIAL (특별체험)

create table special

(

specialno number not null, specialname char(35), zoneno number, primary key(specialno),

foreign key(zoneno) references zone(zoneno)

);

**●** CUSTOMER (고객)

create table customer

(

custno number not null, custname char(10), cardno number, ticketno number, roomstno number,

primary key(custno), foreign key(cardno) references card(cardno), foreign key(ticketno) references ticket(ticketno), foreign key(roomstno) references room(roomstno)

);

**●** ROOM (숙박)

create table room

(

roomstno number not null, roomname char(20),

primary key(roomstno)

);

**4-3 뷰와 인덱스**

**●** 뷰 : 허용된 데이터를 제한적으로 보여주기 위해서 하나 이상의 테이블에서 유도된 테이블이다. 뷰의 경우 검색은 기본 테이블과 거의 동일하지만 삽입, 삭제, 갱신에 제약을 받고, 뷰를 통해서만 접근을 하면 다른 나타나지 않은 데이터들을 보호 할 수 있기 때문에 보안적인 측면에서 매우 우수하다.

- 놀이공원의 데이터 편의성을 위하여 6개의 VIEW 생성

( 목차 5. 구현 : 기능에 대한 질의문 에서 구현)

**●** 인덱스 : DBMS에서 검색 속도를 높이기 위해 사용하는 기술중 하나, 인덱스는 색인으로써 해당 테이블의 애트리뷰트를 색인화 하여 검색시 해당 테이블의 레코드를 전부 스캔하는 것이 아닌 색인화 되어있는 인덱스 파일만 검색해서 빠르게 검색하게 해준다.

- 만약 엄청 큰 놀이공원이라고 가정한다면,(EX:유니버셜 스튜디오 ) 해당 존(ZONE)과 놀이기구(ATTRACTION)를 인덱스로 묶어서, 검색 시 좀 더 빠른 결과가 가능하며 해당하는 놀이기구를 찾을 수 있게 한다. 중복 시 질의문에서, 제일 앞에 중복이 없는 놀이기구를 검색하고 그다음 존번호를 검색하는 것이 좀더 빠르게 찾을 수 있다. ( 목차 5. 구현 : 기능에 대한 질의문 에서 구현)

**5. 구현 : 기능에 대한 질의문**

**5-1 VIEW 생성 질의 및 응용**

**●** #VIEW 1.

CREATE VIEW EMP\_INFO

AS SELECT EMP.EMPNO, EMP.DEPTNO, EMP.EMPNAME, EMP.PHONE, DEP.DEPTNAME, DEP.DEPTPHONE

FROM EMPLOYEE EMP, DEPARTMENT DEP

WHERE EMP.DEPTNO = DEP.DEPTNO

ORDER BY EMP.DEPTNO;

설계 목적 : 1. 직원의 직원번호와 부서번호, 직원이름, 각 직원의 핸드폰 번호, 해당 직원이 속하는 부서 이름과 부서 번호를 한 눈에 알아보기 쉽도록 하기 위해서 뷰를 설계하였다.

2. 해당 부서의 직원의 정보를 검색하기 위해서는, EMPLOYEE 테이블과, DEPARTMENT 테이블을 조인 하여 질의를 해야 하는 귀찮음과 번거로움이 있지만, 위의 뷰를 만들어 그 불편함을 줄이고, 편하게 데이터를 조회 할 수 있게 하여, 반복과 편의성을 높이게 한다

SELECT \*FROM EMP\_INFO 질의문을 통해서, 각 부서에 어느 사원이 일하는지 한 눈에 알아볼 수 있다.



**●**  응용: 상품 판매 부서의 직원들의 정보를 알고 싶을때

SELECT \*

FROM EMP\_INFO

WHERE DEPTNAME = '상품 판매';



**●** #VIEW 2.

CREATE VIEW DEP\_MANAGERINFO

AS SELECT EMP.EMPNAME, EMP.PHONE, DEP.DEPTNAME,DEP.DEPTNO, DEP.STARTDATE

FROM EMPLOYEE EMP, DEPARTMENT DEP

WHERE EMP.EMPNO = DEP.MANAGER

ORDER BY DEP.DEPTNO;

설계 목적: 1. 각 부서별 매니저가 누구인지 한눈에 알아볼 수 있도록 하기 위해서

설계한 뷰이다, 뷰의 애트리뷰트에는 매니저 사원의 이름과 핸드폰 번호, 해당 매니저의 부서 이름과 부서번호, 매니저를 시작한 날짜로 이루어져 있다.

2. 각 부서의 매니저를 확인하기 위해서는, EMPLOYEE(사원)테이블과, DEPARTMENT(부서) 테이블을 조인하여서 DPARTMENT 테이블의 MANAGER 번호와, EMPLOYEE 테이블의 EMPNO가 같은 것을 뽑아내는 질의문을 만들어야 하는데, 매번 매니저가 누군지 확인할 때마다 위 같은 질의문을 사용하기에는 너무 번거롭다 따라서 위의 상세 과정들을 축약하기 위해, DEP\_MANAGERINFO 뷰를 만들었다.

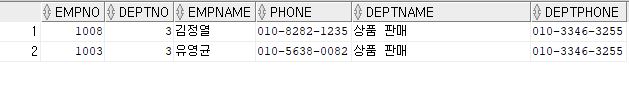


**●** 응용: 3번 부서의 매니저를 알고 싶을 때

SELECT EMPNAME

FROM DEP\_MANAGERINFO

WHERE DEPTNO =3;

**●** #VIEW 3.

CREATE VIEW ZONE\_EMPINFO

AS SELECT EMP.EMPNO, EMP.EMPNAME, EMP.PHONE, ZONE.ZONENO, ZONE.ZONENAME

FROM EMPLOYEE EMP, ZONE ZONE

WHERE EMP.ZONENO = ZONE.ZONENO

ORDER BY ZONE.ZONENO;

설계 목적: 1. 사원마다 부서는 같더라도, 배정되서 일하는 존은 다르다, 그래서 놀이공원의 각 존마다 어떤 사원이 할당되 있는지 한눈에 알아볼 수 있도록 하기위해서 설계한 뷰이다. 뷰의 애트리뷰트에는 사원번호, 사원이름, 사원의 핸드폰 번호, 존 번호, 존의 이름으로 이루어져 있다.

2. 각 존에 할당 배치된 사원들을 확인하기 위해서는 EMPLOYEE(사원)테이블과, ZONE(존) 테이블을 조인하여서 MPLOYEE 테이블의 ZONENO 번호와, ZONE 테이블의 ZONENO가 같은 것을 뽑아내는 질의문을 만들어서 확인해야 한다, 사원들의 부서는 같더라도 역할이 배정되는 존은 다르기 때문이다. 존에 어느 직원이 일하고 있는지 확인할 때마다 위와 같은 상세 과정을 거쳐야 하므로, 이를 축약하기 위해 ZONE\_EMPINFO 뷰를 만들었다.

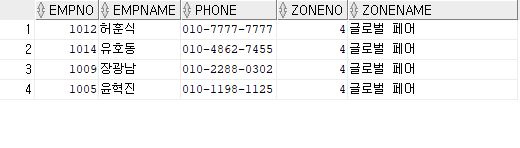


**●** 응용: 글로벌 페어 존에 일하는 사원 현황을 확인하고 싶을 때

SELECT \*

FROM ZONE\_EMPINFO

WHERE ZONENAME = '글로벌 페어';



**●** #VIEW 4.

CREATE VIEW RESERVE\_STATE

AS SELECT CUS.CUSTNO, CUS.CUSTNAME, CUS.ROOMSTNO, RES.ROOMNAME

FROM CUSTOMER CUS, ROOM RES

WHERE CUS.ROOMSTNO = RES.ROOMSTNO AND RES.ROOMSTNO<5

ORDER BY RES.ROOMSTNO;

설계 목적:1. 놀이공원 손님들중 숙박시설을 예약한 손님들의 현황을 한눈에 파악할 수 있도록 하기 위해서 설계한 뷰이다. 뷰의 애트리뷰트에는 고객번호, 고객이름, 숙박 시설 예약번호, 숙박 시설 이름으로 이루어져 있다.

2. 놀이공원 손님들의 숙박시설 예약현황을 확인하기 위해서는 CUSTOMER(고객)테이블과, ROOM(예약시설) 테이블을 조인하여서, CUSTOMER 테이블의 ROOMSTNO 번호와, ROOM 테이블의 ROOMSTNO가 같은 것을 뽑아내는 질의문을 만들어 확인 해야 한다. 이 과정을 생략하기 위해 RESERVE\_STATE 뷰를 만들었다.



**●** 응용: 놀이공원 손님들 중, 숙박시설을 예약한 고객들을 확인하라.

SELECT \*FROM RESERVE\_STATE;



**●** #VIEW 5.

CREATE VIEW ZONE\_ATTRACTIONINFO

AS SELECT ZONE.ZONENO, ZONE.ZONENAME, ATT.ATTNO, ATT.ATTNAME, ATT.RESTRICTION

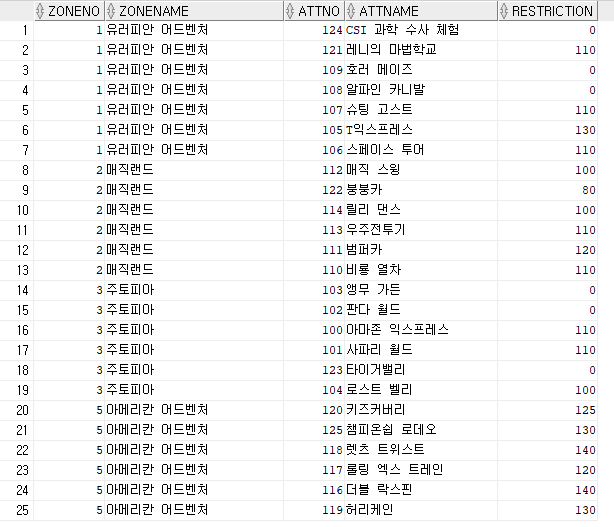
FROM ZONE ZONE, ATTRACTION ATT

WHERE ZONE.ZONENO = ATT.ZONENO

ORDER BY ATT.ZONENO;

설계 목적:1, 놀이공원의 각 존에는, 다양한 놀이기구들이 설치되어 있다. 이 놀이기구들이 어느 존에 배치되어 있는지 한눈에 보기 쉽도록 하기 위해서 설계한 뷰이다. 뷰의 애트리뷰트에는 존번호, 존이름, 놀이기구번호, 놀이기구이름, 놀이기구 키 제한 으로 이루어져 있다

2. 놀이기구들이 어느 존에 배치되어 있는지 확인하기 위해서는 ZONE(존)테이블과, ATTRACTION(놀이기구)테이블을 조인하여서, ZONE 테이블의 ZONENO 번호와, ATTRACTION 테이블의 ZONENO가 같은 것을 뽑아내는 질의문을 만들어 확인 해야 한다. 이 과정을 생략하기 위해서 ZONE\_ATTRACTIONINFO 뷰를 만들었다.



**●** 응용: 2번존, 매직랜드의 놀이기구 현황을 파악하라.

SELECT \*

FROM ZONE\_ATTRACTIONINFO

WHERE ZONENO = 2;



**●** #VIEW 6.

CREATE VIEW ENTERTAINMENT\_TIMEINFO

AS SELECT TIME.ENTERTIMENO, ENTZONE.ENTERNAME, TIME.ENTERTIME, ENTZONE.ZONENO

FROM ENTERTAINMENTTIME TIME, ENTERTAINMENTZONE ENTZONE

WHERE TIME.ENTERTIMENO = ENTZONE.ENTERZONENO

ORDER BY ENTZONE.ENTERZONENO;

설계 목적:1. 놀이공원의 존에는 각각의 즐길거리(엔터테이먼트)를 시간대별로 기획한다.

각각의 존이 기획한 엔터테이먼트 이름과, 시간대를 한눈에 확인 할 수 있도록 설계한 뷰이다.

2. 각각의 존에 해당하는 엔터테이먼트 이름과, 시간대별을 매칭하여 나열하기 위해서는ENTERTAINMENTTIME(엔터테이먼트 시간) 테이블과, ENTERTAINMENTZONE(엔터테이먼트 존) 테이블을 조인하여서 ENTERTAINMENTTIME 의 ENTERTIMENO 번호와, ENTERTAINMENTZONE의 ENTERZONENO가 같은 것을 뽑아내는 질의문을 써서 확인 해야 하는데, 이 과정을 생략하기 위해서 ENTERTAINMNET\_TIMEINFO 뷰를 만들었다.

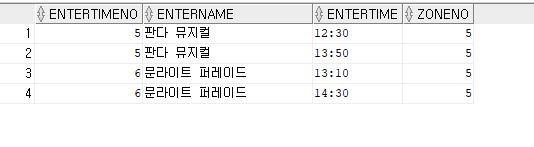


**●** 응용: 5번 존에서 열리는 엔터테이먼트 이름과 시간대를 확인해 보아라.

SELECT \*

FROM ENTERTAINMENT\_TIMEINFO

WHERE ZONENO = 5;



**5-2 인덱스 구현 질의문**

**CREATE** table emp3 as select \* from ATTRACTION;

**CREATE** index idx\_emp3\_composite on emp3(ATTNO,ZONENO);

**SELECT \*** from emp3 where ATTNO > 0 and ZONENO > '0';



* 1. **트랜잭션 구현 질의문**

**●** 1.트랜잭션(1) => 매니저 변경

- 해당부서(Dep\_no)의 매니저(Emp\_no)가 퇴사하면서 같은 부서사원(Man\_no)에게 매니저를 넘기고 기존매니저는 삭제

- Dep\_no = 퇴사하는 매니저의 부서번호, Emp\_no = 퇴사하는 매니저의 사원번호

Date = 매니저 시작 날짜, Man\_no = 새로운 매니저의 사원번호

**SELECT** MANAGER

**FROM** DEPARTMENT

**WHERE** DEPTNO = Dep\_no;

ex) (Dep\_no =1) 1번부서의 매니저의 사원번호

- 해당부서(Dep\_no)의 매니저번호와 해당사원(Emp\_no)의 번호가 같을 경우

**UPDATE** DEPARTMENT

**SET** STARTDATE = Date, MANAGER = Man\_no

**WHERE** DEPTNO = Dep\_no;

**DELETE**

**FROM** EMPLOYEE

**WHERE** EMPNO = Emp\_no;

**●** 2. 트랙잭션(2) => 존과 부서 근무자수 체크 후 사원 추가,갱신

- 사원에 새로추가(Emp\_no, Emp\_name, Phone\_no, Zon\_no, Dep\_no) 되면서 현재 존의 직원이 몇명인지 확인하고 인원을 충당해 준다.

**SELECT** ZONENO, COUNT(ZONENO) AS COUTNZONE

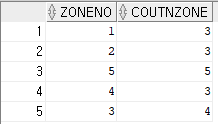
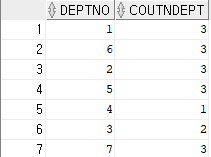
**FROM** EMPLOYEE

**GROUP BY** ZONENO;

**SELECT** DEPTNO, COUNT(DEPTNO) AS COUTNDEPT

**FROM** EMPLOYEE

**GROUP** **BY** DEPTNO;

- 해당 존에 근무하는 직원 수 확인

**INSERT**

**INTO** EMPLOYEE(EMPNO, EMPNAME, PHONE, ZONENO, DEPTNO)

**VALUES** (Emp\_no, Emp\_name, Phone\_no, Zon\_no, Dep\_no);

- 원하는 존번호(Zon\_no), 부서번호(Dep\_no)를 가진 직원정보를 삽입해준다.

**●** 3. 트랜잭션(3) => 예약 저장하기

- 예약하려는고객(Cus\_no)의 키로 탈 수 있는 놀이기구를 보여주고 예약이 가능하게 함

**SELECT** ATTNO, CUSTNO

**FROM** ATTRACTION AS A, CUSTOMER AS C

**WHERE** A.RESTRICTION <= C.CUSRESTICTION AND C.CUSTNO=Cus\_no;

ex) 15번 고객의 경우 탈 수 있는 놀이기구 번호,이름 표시



-놀이기구번호, 고객번호, 100(예약번호)로 새로 RESERVED에 표시

**INSERT**

**INTO** RESERVE(attno, custno, reserveno)

**VALUES** (Att\_no, Cus\_no, 100);

**●** 4. 트랜잭션(4) => 엔터테이먼트 공연시간 변경

- 해당존의 모든 엔터테이먼트의 엔터테이먼트번호, 이름, 공연시간을 출력 후 시간을 변경

**SELECT** ENTERZONENO, ENTERNAME, ENTERTIME

**FROM** ENTERTAINMENTTIME T, ENTERTAINMENTZONE Z

**WHERE** Z.ZONENO = Zon\_no AND Z.ENTERZONENO = T.ENTERTIMENO;

Ex) 3번 존의 해당하는 경우 출력





- 바꿔야 하는 엔터테이먼트번호(Etn\_no), 시간(Time) 확인 후 바꾸어야 할 시간(Ctime)으로 변경

**UPDATE** ENTERTAINMENTTIME

**SET** ENTERTIME = Ctime

**WHERE** ENTERTIMENO = Ent\_no AND ENTERTIME= Time

**●** 5. 트랜잭션 => 카드결제 다른 카드로 변경

- 고객의 카드결제를 다른 카드로 바꿈

**SELECT** CUSTNO, C.CARDNO, CARDNAME

**FROM** CUSTOMER C, CARD D

**WHERE** C.CUSTNO = Cus\_no AND C.CARDNO = D.CARDNO

Ex) 3번 고객의 고객변호, 결제카드번호, 결제카드이름 검색





- 고객(Cus\_no)이 새로 결제한 카드(Car\_no)로 카드번호 바꾸어줌

**UPDATE** CUSTOMER

**SET** CARDNO = Car\_no

**WHERE** CUSTNO=Cus\_no;

**6. 느낀점 : 프로젝트를 마치며**

**●** 박다수: 데이터베이스라는 과목을 2학년에 들어서 처음 접하고 배우게 되었는데, 처음에는 이걸 어떻게 해야하나 생각했지만, 허훈식 교수님의 강의를 들으면서 시험기간 공부와, 자습을 하다보니 점점 재미있어지고, 자신감이 생기게 되었다. 이번 프로젝트도 뜻이 맞는 친구들과 조를 짜서 자료 분석에서부터, 개념적 설계, 논리적 설계, 물리적 설계를 거치고 마지막에 실습으로 오라클에서 데이터를 넣으면서 우리가 했던 것들을 뽑아내는 것까지 눈으로 확인하니 정말 뿌듯한 느낌이 들면서도, 수업시간에 했던 내용들을 더 잘 이해하게 되었다..

**●** 정지운: 처음부터 호기심이 갔던 과목이라서, 이번 프로젝트에 더 적극적으로 참여했던 것 같다. 1차 발표기간 까지 친구들과 함께 준비하면서 협동심과 단결력을 잘 느꼈던 것 같고, 부족한 개념들도 많았지만, 친구들과 함께 정보를 토론하고 나누면서 잘 해결해 나갔던 것 같다. 그리고 데이터베이스를 관리하는 모든 IT관련자분들을 존경하는 시간을 가지게 된 프로젝트의 시간이였다.

**●** 윤혁진: 원래부터 이론보단 실습을 좋아하는 타입이라 이번 프로젝트도 많은 기대를 가지며 준비하는 태도로 임했다. 앞서 배운 모든 개념들을 한 프로젝트에 써 부어야 하는 점이 배운 것을 활용한다는 느낌을 받아서 보람감 같은 것들이 많이 느껴졌다. 팀원들과 힘들었던 기억도 있지만. 함께 머리를 맞대서, 데이터를 어떻게 다룰지 고민했던 시간들이 프로젝트를 끝내고 지나보니 너무 좋은 시간이었다.

**●** 김지웅: 처음 만난 낯선 친구들과 프로젝트를 짜야된다는게 조금 부담되고 걱정됬지만, 알고보니 좋은 친구들이라 마음을 터놓고 같이 프로젝트를 준비할 수 있었다. 데이터베이스가 이러한 과정으로 설계된다는 걸 프로젝트를 통해 깨달았고, 정말 힘든 것 같다고 느껴졌다. 앞으로 데이터베이스를 다루게 될지는 모르겠지만, 다음에 다시 배울 기회가 오면, 한층 더 쉽게 다가갈 수 있을 것 같다.