게임 서버 컴퓨터

실행 및 관리 데이터베이스:

일반 유저들을 위한 GUI시스템 구축

|  |  |
| --- | --- |
| 이름: | 장현규 |
| 학번: | 17011549 |
| 학과: | 컴퓨터공학 |

목차

1. 최종 요구사항 분석
   1. 해당 프로그램의 기능
   2. 엔터티, 어트리뷰트, 릴레이션
2. 최종 ER-DIAGRAM
   1. ER-DIAGRAM 설명
3. 최종 DB Schema
   1. 생성된 테이블의 스키마 설명
4. 최종 SQL(Create, Insert) 캡쳐 화면
5. 프로그램 구현단계
6. 프로그램 실행 캡쳐 화면
   1. Schema Check
   2. SELECT Query
   3. Insert Query
   4. Update Query
   5. Delete Qeury
   6. Custom Query
7. 최종 요구사항 분석
   1. 해당 프로그램 기능
      1. 목적: 서버 컴퓨터를 대여하는 회사가 게임서버와 서버컴퓨터 및 사용자들을 관리하는 것
      2. 필수 역할

\*서버 컴퓨터 관리

\*게임 서버 및 플레이어 관리

\*서버 컴퓨터 간 오류 감지 관계 관리

\*서버 관리자 관리

\*사용되고 있는 게임 목록 관리

* 1. 게임 서버 관리에서 나타나는 엔터티, 릴레이션 과 각각의 어트리뷰트
     1. 플레이어: 실세계에서 게임서버를 통해 게임에 접속하여 게임을 즐기는 사람들

어트리뷰트:

게임ID : 플레이어들을 구분하기 위한 키 어트리뷰트 중 하나

핑: 플레이어들의 인터넷 상태를 나타내는 요소로 핑이 높으면 서버의 안정성을 위하여 추방될 수 있음

1.2.2 게임서버: 서버 컴퓨터를 통해 게임을 실행한 후 플레이어들이 참가하여 게임을 진행하는 서버

어트리뷰트:

컴퓨터 ID: 어떤 서버 컴퓨터에서 작동하고 있는지 확인하기 위함

게임 참가 인원: 현재 서버에 있는 인원수를 나타냄

서버번호: 서버들을 구분해주기 위한 키 어트리뷰트

시작시간: 게임 서버가 시작한 시간

게임 고유코드: 현재 서버에서 실행되고 있는 게임을 판별하기 위한 고유 코드번호

1.2.3 게임: 서든 어택이나 메이플과 같은 온라인 게임으로 플레이어들이 즐겨 하는 게임들

어트리뷰트:

게임이름: 게임의 타이틀 명

제작회사: 게임을 제작한 회사

플레이 시간: 게임이 시작되고 종료까지 걸리는 시간

최대 참가인원: 최대 인원수 제한을 위한 요소

게임 고유코드: 게임들을 구별하기 위한 키 어트리뷰트

* + 1. 서버 컴퓨터: 게임 서버가 구동되고 있는 컴퓨터

어트리뷰트:

컴퓨터 ID: 서버 컴퓨터들을 구별하기 위한 키 어트리뷰트

물리적 위치: 서버 컴퓨터가 현재 위치하고 있는 국가를 나타냄

관리자 ID: 해당 컴퓨터를 책임지고 있는 관리자의 ID

정기점검 요일: 해당 컴퓨터의 정기점검이 이루어지는 요일

핑 제한: 플레이어의 최대핑을 제한하여 안정적인 서버운영 관리

* + 1. 서버 관리자: 실세계에서 서버 컴퓨터를 관리하고 플레이어들의 문의를 처리함

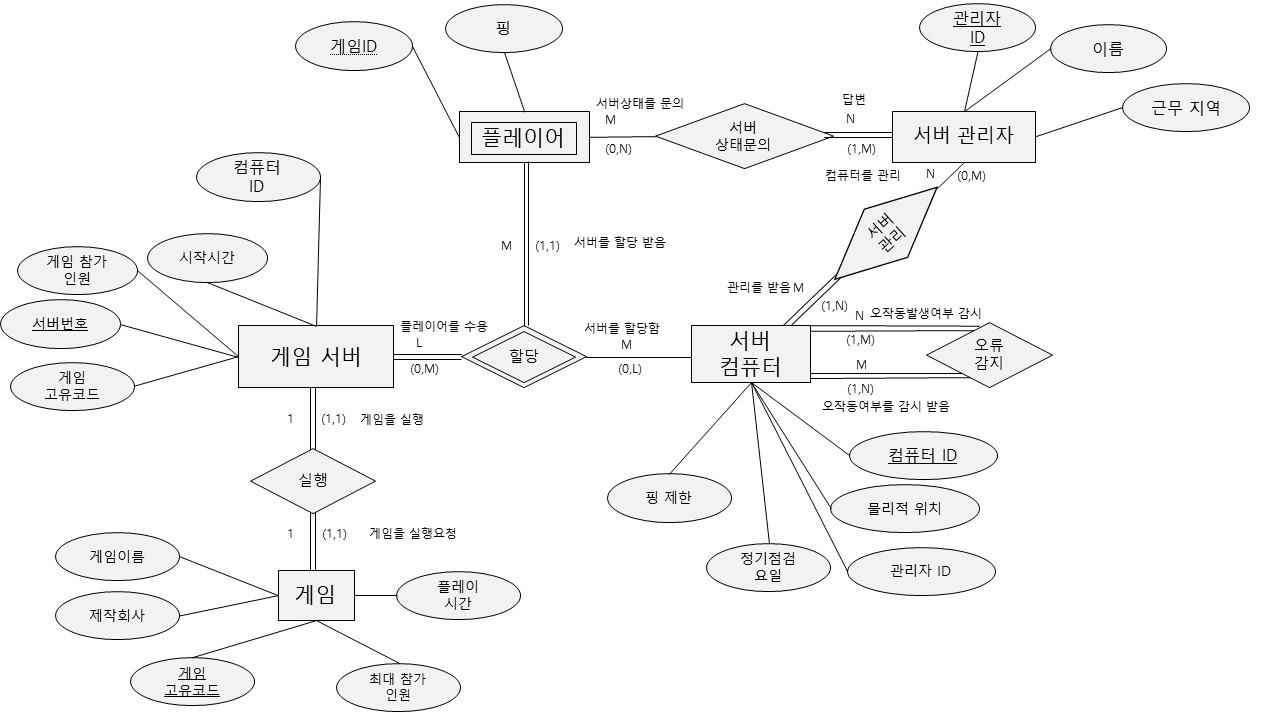
어트리뷰트:

관리자 ID: 관리자들을 구분하기 위한 키 어트리뷰트

근무 지역: 관리자들의 근무지역

이름: 관리자들의 실제이름

* + 1. 실행 Relation: 게임 서버와 게임사이의 관계로 게임이 게임 서버에서 구동되고 있음을 알림
    2. 서버 상태문의 Relation: 플레이어와 서버관리자의 관계로 플레이어들의 서버관련 질의와 서버관리자의 답변으로 이루어짐
    3. 서버 관리: 서버 관리자와 서버컴퓨터의 관계로 서버 관리자의 정기점검과 같은 관리와 서버 컴퓨터의 관리 받음으로 실행됨
    4. 오류감지: 서버 컴퓨터들과의 관계로 서로가 서로의 오류를 감지하여 보고하는 관계를 나타냄
    5. 할당: 플레이어, 게임서버, 서버 컴퓨터들과의 관계를 나타낸 것으로 플레이어는 서버에 할당되고, 게임서버는 서버컴퓨터에 할당되며 서버 컴퓨터는 이 할당된 플레이어와 게임 서버를 수용함.

2. 최종 ER-DIAGRAM

게임 서버 컴퓨터 관리 ER-DIAGRAM

2.1 DIAGRAM을 Relation을 통해 설명

플레이어

서버 관리자

서버 문의

상태문의

(1,M)

(0,N)

M

N

서버상태를 문의

답변

플레이어

특징: M:N관계, 서버 관리자는 전체 참여 플레이어는 부분참여로 관계가 이루어짐.



게임 서버

(0,M)

(1,1)

M

1

서버를 할당 받음

플레이어를 수용

할당

특징: 1:M관계, 게임 서버는 부분참여 플레이어는 전체참여로 관계가 이루어짐. 현재 플레이어는 게임서버를 Owner Entity type으로 가지며 게임 서버는 플레이어에게 키값을 제공



서버 컴퓨터

(0,M)

(1,1)

M

1

서버를 할당 받음

서버를 할당함

할당

특징: 1:M 관계, 서버 컴퓨터는 부분참여 플레이어는 전체참여로 관계가 이루어짐.

게임 서버

서버 컴퓨터

서버

상태문의

(0,M)

(1,1)

M

1

서버가 실행됨

서버를 실행함

할당

특징: 1:M 관계, 게임서버는 서버 컴퓨터가 있어야만 존재할 수 있음.

1

서버 관리자

서버 컴퓨터

서버 관리

상태문의

(1,1)

(0,M)

M

서버를 관리함

관리 받음

특징 : 1:M 관계, 서버 관리자는 여러대의 컴퓨터를 관리함, 서버 컴퓨터는 한명의 서버 관리자를 할당 받음. 서버 컴퓨터는 서버 관리자가 반드시 있어여함.

#gui에서 total participation을 input query 메뉴에 따로 구현하였음

오류 감지

감지

서버 컴퓨터

컴퓨터

오작동발생여부 감시

(1,N)

(1,M)

N

M

오작동여부를 감시받음

특징: M:N관계, 서버 컴퓨터는 다른 서버 컴퓨터로부터 오류 발생여부를 감시받음

#gui에서 total participation을 구현하기 위해 input query메뉴에 따로 구현하였음

3. 최종 DB SCHEMA

서버관리자:

SERVER\_MANAGER(ID,Mname,PLACE)

ID: 서버 매니저를 구별하는 키 어트리뷰트

Mname: 서버 매니저의 이름

PLACE: 서버매니저의 근무 국가

게임:

GAME(GAMEID, TITLE, MAKER, TIME, MAX\_P)

GAMEID: 게임을 구분하기 위한 고유의 코드

TITLE: 게임의 이름

MAKER: 게임 제작사 이름

TIME: 실행 시간

MAX\_P: 게임에 접속할 수 있는 최대 사용자의 인원수

서버컴퓨터:

SERVER\_COMPUTER(COMID, PING, CHECK\_DATE, LOCATION, MANAGER\_ID)

COMID: 서버컴퓨터를 구별하기 위한 고유한 코드

PING: 받아들일 수 있는 최대핑

CHECK\_DATE: 정기점검 요일

LACATION: 서버가 위치한 국가

MANAGER\_ID: 해당 서버 컴퓨터를 관리하는 매니저(Foreign key)

게임 서버:

GAME\_SERVER(SERVER\_ID, GAME\_ID,PEOPLE,START\_TIME,COMID)

SERVER\_ID: 서버를 구별하기 위한 고유한 코드

GAME\_ID: 현재 실행중인 게임을 참조하기 위한 Foreign key

PEOPLE: 현재 접속중인 인원수

START\_TIME: 서버에서 구동중인 게임의 시작 시간

COMID: 현재 서버가 돌아가고 있는 게임의 서버의 위치

플레이어:

PLYAER(SID\_,PID,PING)

SID\_:현재 플레이어가 있는 서버의 아이디 (Foreign key & primary key)

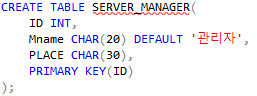
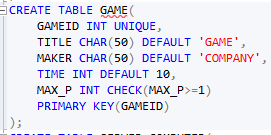
PID: 현재 플레이어의 ID

PING: 현재 플레이어의 PING

ERROR\_DETECT(SUPERVISE, SUPERVISED)

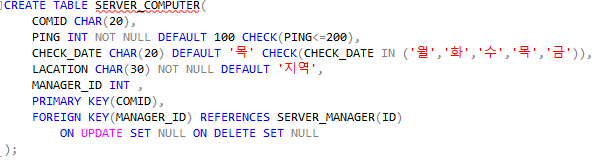
SUPERVISE: 감시를 담당하고 있는 컴퓨터

SUPERVISED: 감시받고 있는 컴퓨터

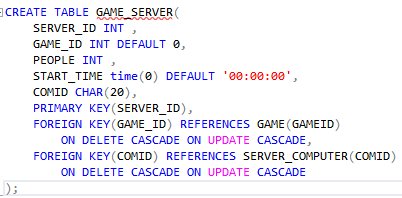
4. 최종 SQL(Create, Insert) 캡쳐 화면

2 게임

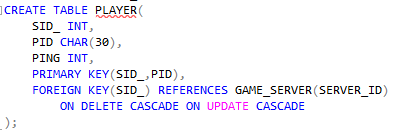
1 서버 매니저



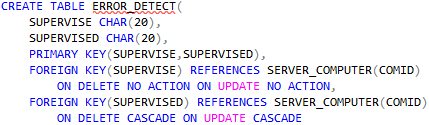
3 서버 컴퓨터



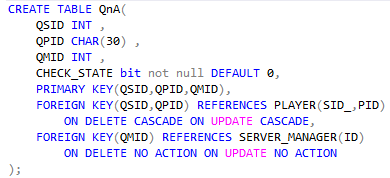
4 게임 서버



5 플레이어



6 오류 감지



1. 서버관련 문의

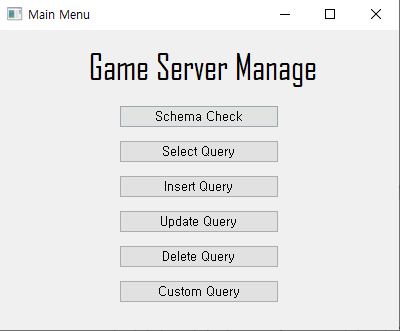
5. 프로그램 구현 단계

사용한 C/C++ GUI 라이브러리: QT visual studio tool

|  |  |
| --- | --- |
| 메뉴 이름 | 기능 내용 및 구현 단계 |
| Schema Check | DB에 존재하는 Table에 대한 Schema 확인 – 구현 |
| SELECT Query | 전체 조회: 모든 테이블을 조회하여 그 결과를 테이블 형태로 보여준다 – 구현  게임이 종료된 서버 조회: 게임시간을 참고하여 현재시간과 시작시간의 차를 구한 뒤 종료시간을 구한다. 종료시간이 지난 게임 서버를 보여준다 – 구현 JOIN 사용  처리되지 않은 QnA 조회: 처리 상태가 0(처리되지 않음) 인 튜플을 보여준다 – 구현  서버 인원 조회: 선택한 서버내에 있는 인원들의 정보를 보여준다. – 구현  특정 서버 점검요일 조회: 선택한 점검요일에 해당되는 게임 서버를 보여준다 – 구현 JOIN사용  특정 인원 수 이상 서버 조회: 정수를 입력 받아 해당 정수 이상의 인원을 가지고 있는 서버를 보여준다. - 구현  특정 값을 포함하는 튜플 검색: 특정 테이블을 선택한 뒤 검색하고 싶은 값을 입력하면 해당하는 어트리뷰트를 포함하고 있는 모든 튜플이 검색된다.  \*모두 구현됨\* |
| INSERT Query | 일반 입력: 테이블을 선택한 뒤 키 값을 입력하면 해당 테이블에 값이 입력됨 – 구현  QnA 자동입력: QnA를 입력할 때 서버번호, 사용자id만 입력하면 할당된 질의 응답이 가장 적은 서버 관리자에게 자동으로 할당됨 – 구현 JOIN사용  서버컴퓨터 자동입력: 서버컴퓨터를 입력할 때 서버 관리자를 제외한 다른 정보들을 입력하면 관리하는 서버가 가장 적은 서버 관리자를 자동으로 할당함 – 구현 JOIN 사용  플레이어 자동입력: 플레이어ID, 게임코드,ping을 입력하면 알아서 서버를 할당함. 만약 할당 가능한 서버가 없다면 서버를 생성할 것인지 물어본후 서버 생성- 구현 JOIN 사용  \*모두 구현됨  서버 컴퓨터 감시 체계 자동 할당: 오류 감지를 받고 있지 않은 컴퓨터들에게 감지 컴퓨터를 할당함. 감지하는 컴퓨터가 가장 적은 컴퓨터부터 우선 할당되어짐 – 구현 JOIN 사용  \*모두 구현됨 |
| UPDATE Query | 값 수정: 선택한 테이블을 GUI의 테이블 구조에 보여준뒤 간단히 해당 GUI의 테이블을 수정한후 실행 버튼을 누르면 값이 수정됨 – 구현  서버관리자 자동 할당: 서버 컴퓨터들중 서버 매니저를 할당 받지 못한 컴퓨터들에게 서버 매니저를 할당함 – 구현 JOIN 사용  서버 요일 일괄 변경: 특정 서버 관리자가 담당하고 있는 모든 서버 컴퓨터의 점검 요일을 수정 – 구현  게임 종료 서버 자동 시간 갱신: 시간이 지나 종료된 서버가 있으면 시간을 현재 시간으로 갱신함 – 구현  \*모두 구현됨 |
| DELETE Query | 키값을 이용한 삭제: 테이블을 선택한 뒤 테이블에 해당하는 기본키값을 입력해주면 삭제가 진행됨 – 버그 수정후 구현 완료  답변 완료 QnA 일괄 삭제: 답변이 완료된(CHECK\_STATE가 1인) 답변을 모두 삭제함 – 구현  서버 컴퓨터 삭제 자동 처리-오류를 감지하는 컴퓨터를 삭제할 때 오류 감지를 해제해 주고 삭제를 진행함 – 구현  서버 관리자 삭제 자동 처리: 완료되지 않은 답변이 있어 서버 관리자를 제거하지 못할 시 사용됨. 답변 여부를 물어본 후 해당되는 답변을 삭제뒤 서버 관리자 삭제 – 구현  \*모두 구현됨 |
| Custom Query | 사용자가 임의로 Query를 작성하여 수행 – 구현 |

6. 프로그램 실행 캡쳐 화면

메인메뉴



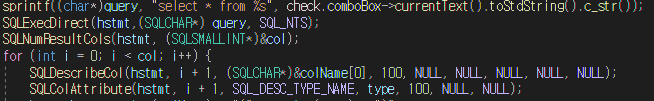
SQL문 따로 없음

Schema Check

  
선택한 테이블의 스키마를 보여줌

테이블 선택

사용된 SQL문



Select Query

전체 조회

모드 선택

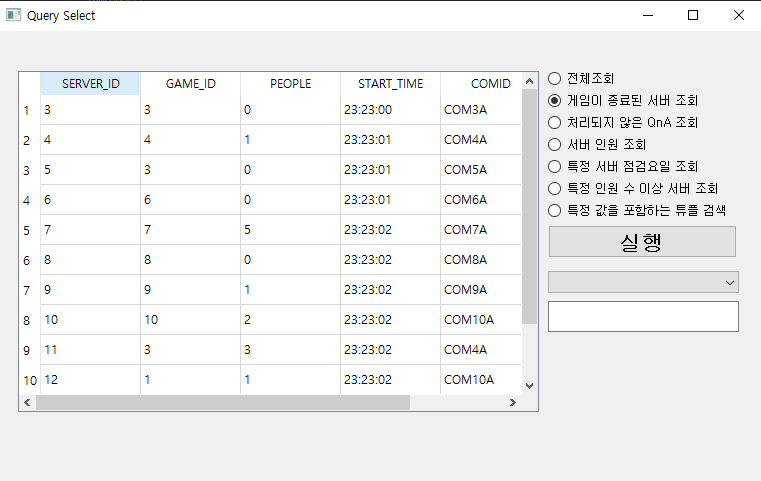


전체조회 선택 후 실행을 누르면 모든 테이블이 보여짐

사용된 SQL



게임이 종료된 서버 조회

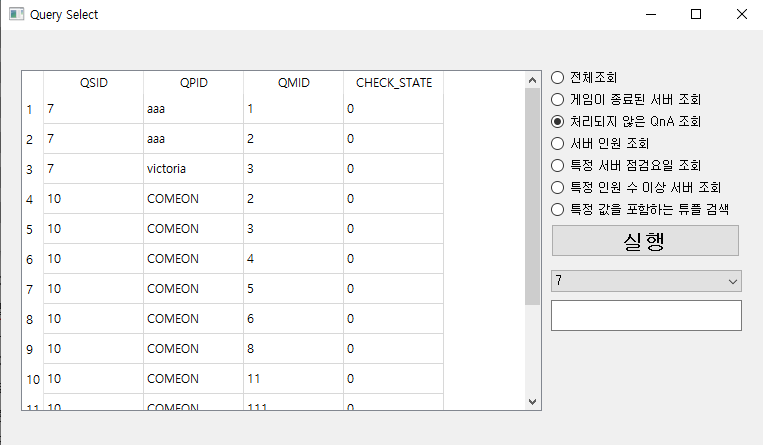


게임 서버들중 게임이 종료된 서버를 보여줌. 현재 시각은 다음날 아침 05:17분으로 모든 서버의 게임이 종료되어 있음을 보여줌

사용된 SQL

SELECT SERVER\_ID, GAME\_ID, PEOPLE, START\_TIME, COMID FROM GAME\_SERVER E, GAME G WHERE E.GAME\_ID = G.GAMEID AND DATEDIFF(MINUTE, E.START\_TIME, CONVERT(TIME, GETDATE())) >= G.TIME OR DATEDIFF(MINUTE, E.START\_TIME, CONVERT(TIME, GETDATE()))<0

처리되지 않은 QnA 조회

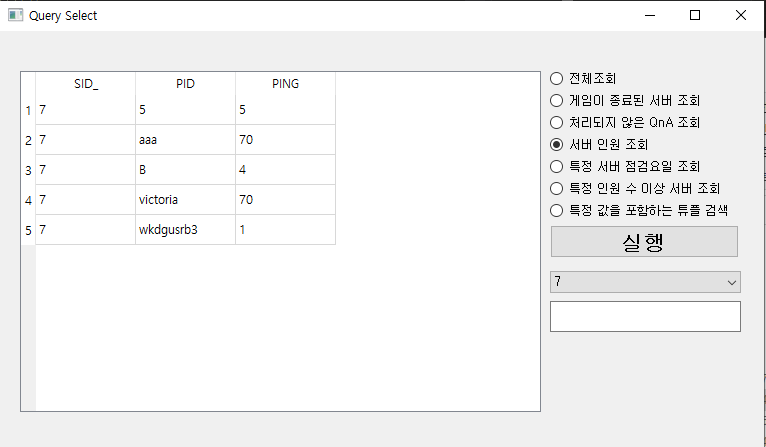


처리가 되지 않은 (CHECK\_STATE가 0인 상태) 튜플을 보여줌

사용된 SQL



서버 인원 조회

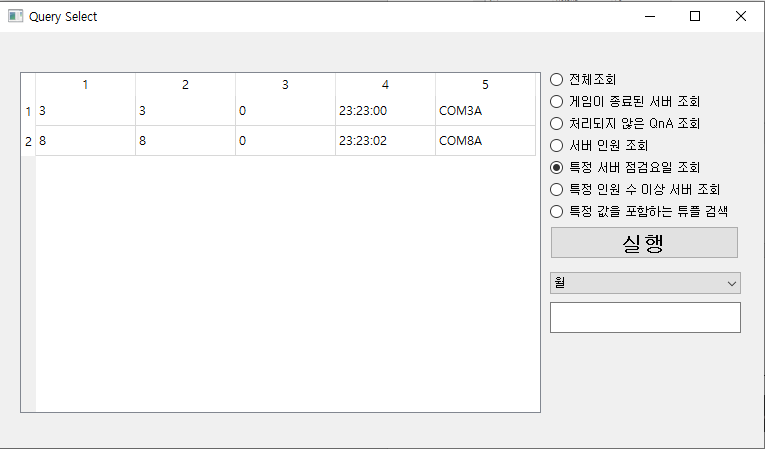


서버 선택

선택한 서버에 해당되는 모든 인원들의 정보를 보여줌



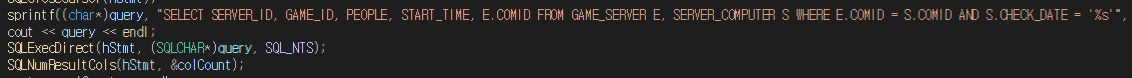
특정 서버 점검요일 조회



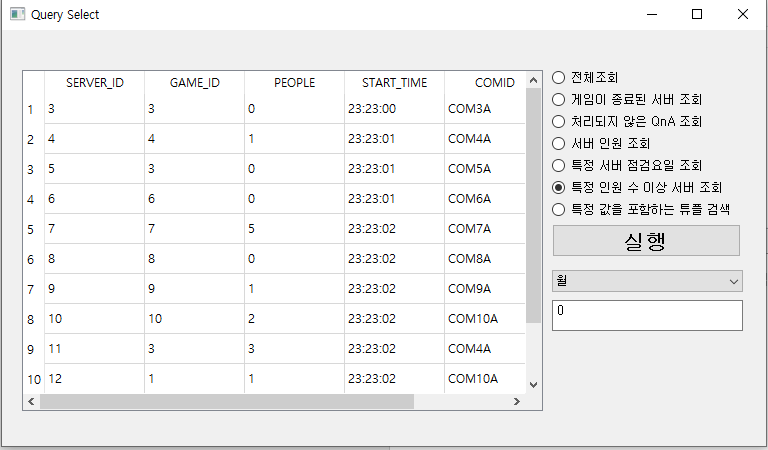
요일 선택

선택한 요일에 해당되는 점검일을 지닌 게임 서버 목록을 보여줌

사용된 SQL



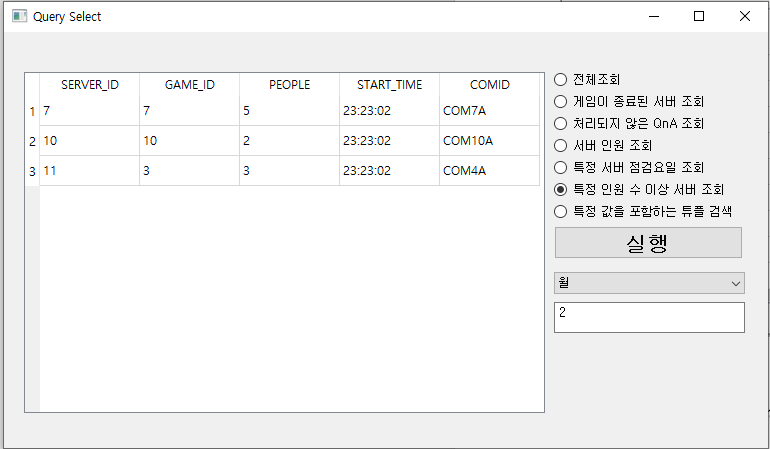
특정 인원 수 이상 서버 조회



값 입력

값을 입력 받아 해당되는 값 이상의 인원수를 지닌 튜플들 조회

#시나리오 인원수가 2 이상인 인원들을 보고 싶을 때



2. 버튼 클릭

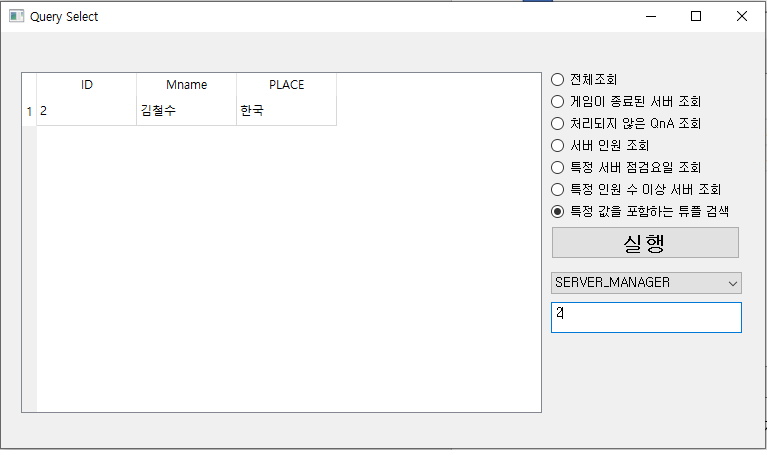
1.값 입력

위와 비교했을 때 인원수가 2 이상인 서버가 남아 있는 것을 확인할 수 있음

사용된 SQL



특정 값을 포함하는 튜플 검색

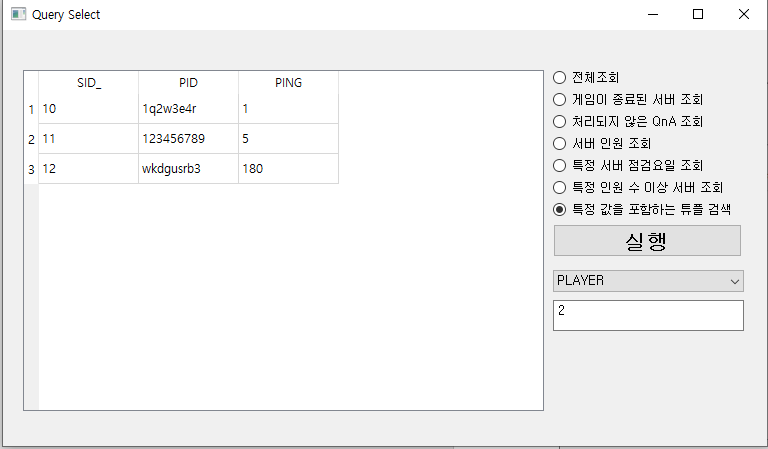


테이블 선택

값을 하나만 입력

특정 테이블에서 특정 값을 포함하고 있는 튜플들 조회

#시나리오: 플레이어들중 2를 포함하는 값을 지닌 튜플을 보고 싶을 때



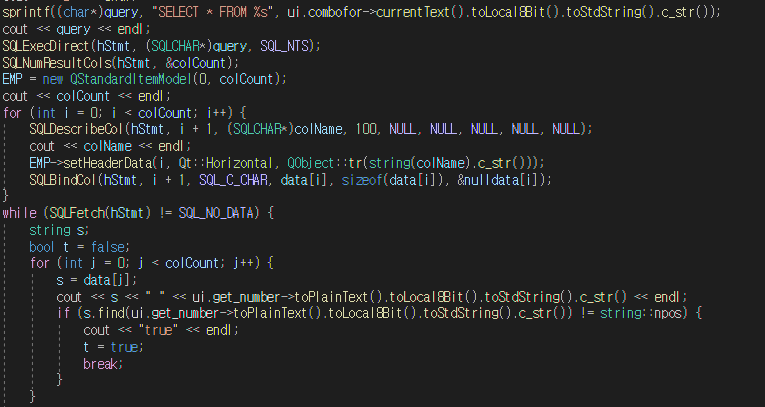
서버 ID에 2가 있어서 나옴

ID에 2가 있어서 나옴

2. 버튼 클릭

1. 2입력

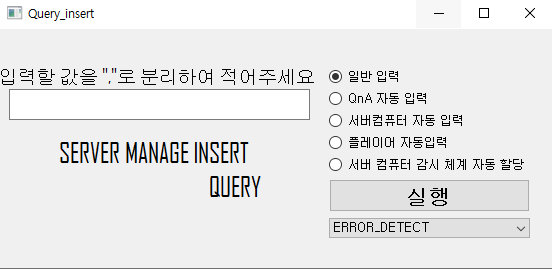
사용된 SQL



특정 테이블을 불러온 뒤 GUI에 구현할 때 값을 지니고 있는지 확인하여 보여주는 것이다.

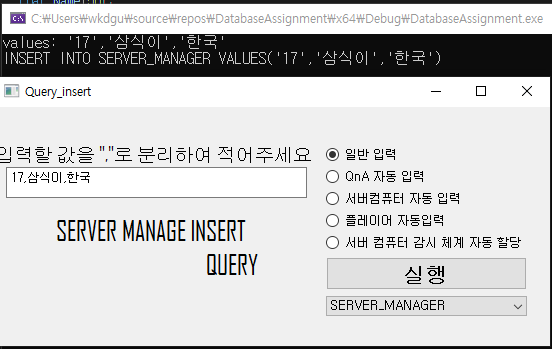
Insert Query

일반 입력



선택한 테이블에 값들을 입력하면 값이 추가됨

#시나리오: 서버 매니저를 추가하고 싶을 때



3. 실행

2. 값입력

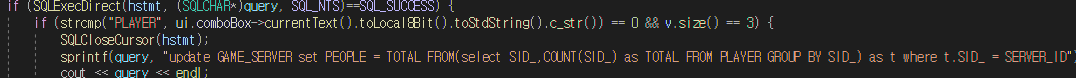
1.테이블 선택

추가된 모습을 확인할 수 있다.

사용된 SQL



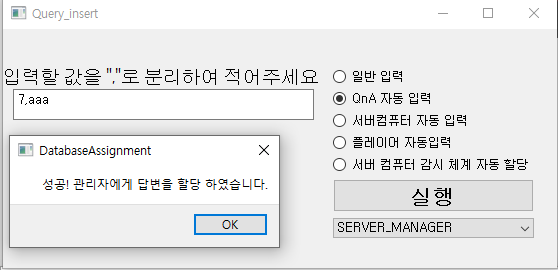
입력된 정보가 플레이어인 경우 PING확인 및 서버의 인원수를 수정해 줘야 하기 때문에 다음과 같은 쿼리문도 사용되었다.



QnA 자동 입력

플레이어의 아이디와 서버를 입력하면 자동으로 값을 서버 관리자 에게 할당됨

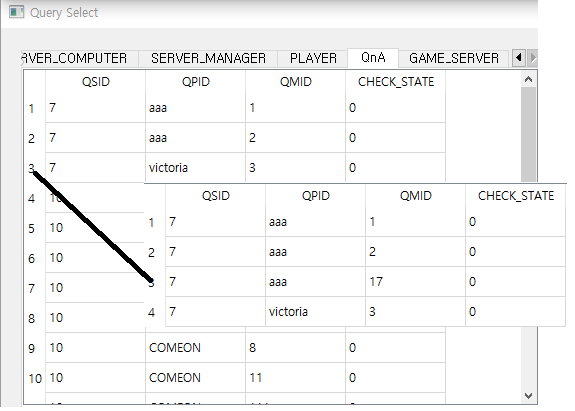
#시나리오 7번서버에 aaa플레이어가 문의를 할 시



2. 버튼 클릭

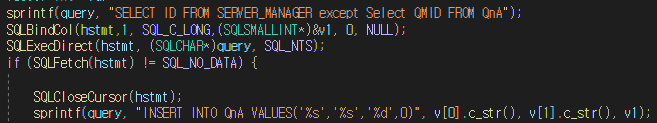
1.값 입력

결과



서버 매니저를 따로 입력하지 않았지만 위에서 새로 추가된 17번 매니저가 질의를 할당받음

사용된 SQL





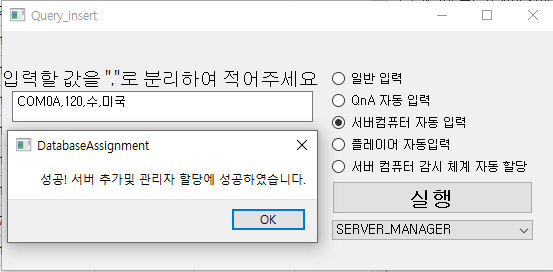
가장 적게 할당 받은 서버 매니저 구하기



서버 자동입력

서버 매니저를 제외한 다른 정보들을 입력하면 자동으로 서버 매니저가 할당된다.

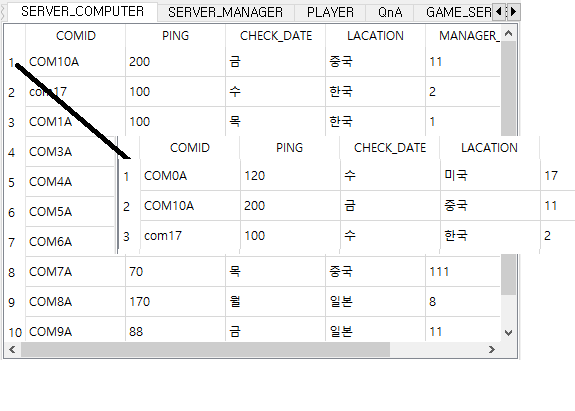
#시나리오 서버를 새로 추가할 때



1. 값을 입력

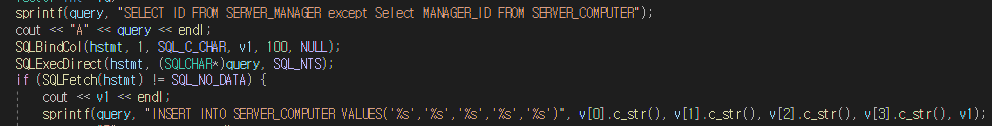
2. 버튼 클릭

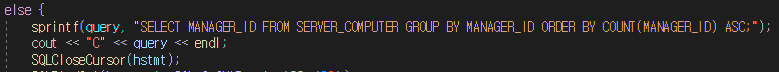
결과

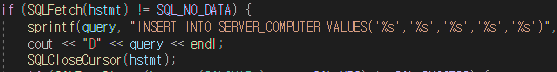


값이 새롭게 추가된 모습을 볼 수 있다.

사용된 SQL



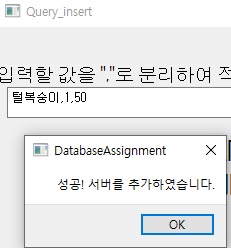
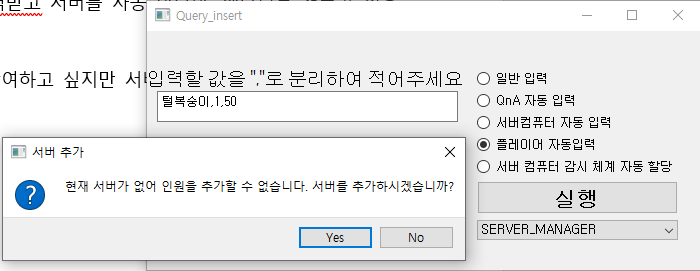
할당된 서버가 가장 적은 관리자에게 할당함



플레이어 자동 입력

플레이어ID, 게임 코드, 핑을 입력받고 서버를 자동 할당함. 해당되는 서버가 없을시 생성후 할당함.

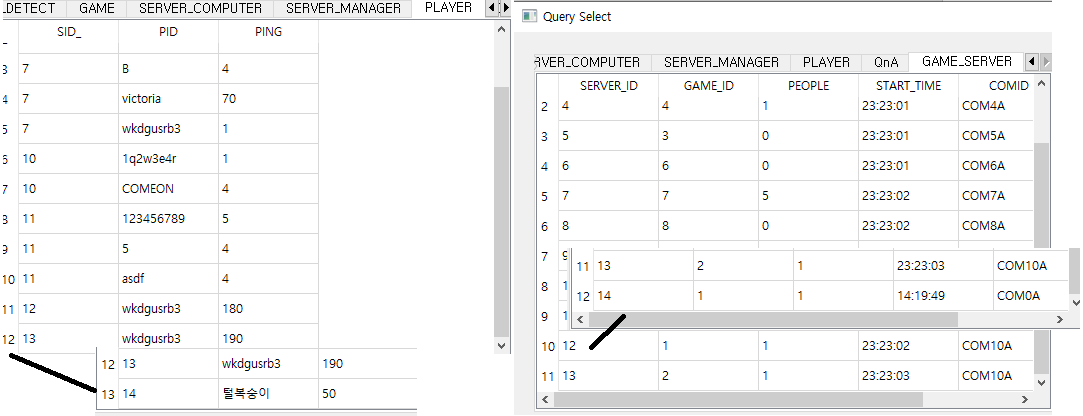
#시나리오 플레이어가 서버에 참여하고 싶지만 서버인원이 가득자 서버에 들어갈 수 없을 때



3. 생성을 원할 시 yes 선택

1. 값을 입력

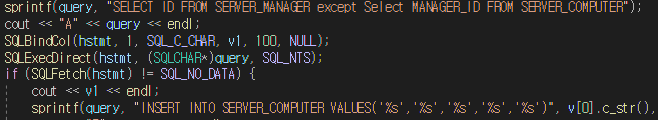
2. 버튼 클릭

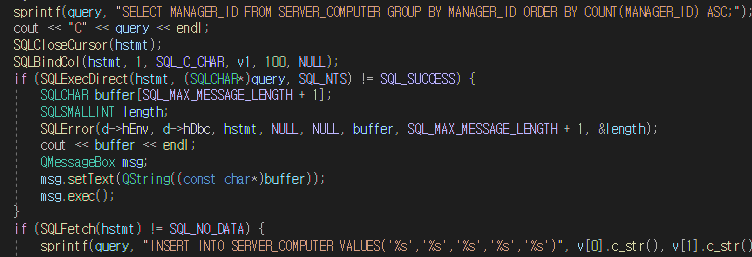


서버가 자동으로 생성된 후 플레이어가 할당된 모습이 보인다.

서버가 존재할 시 그곳으로 자동 할당됨

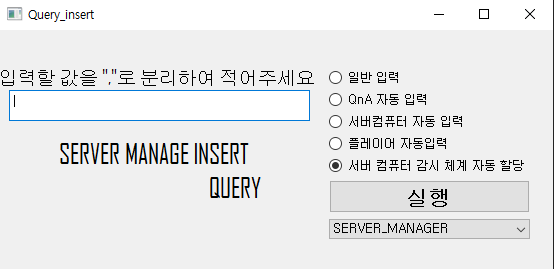
사용된 SQL





서버 감시체계 자동할당:

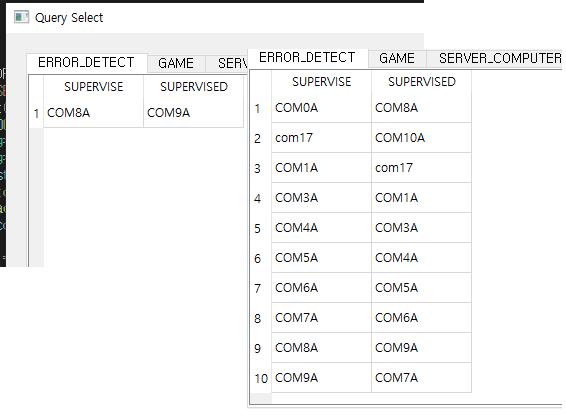
서버 감시체계 관계를 자동으로 할당함



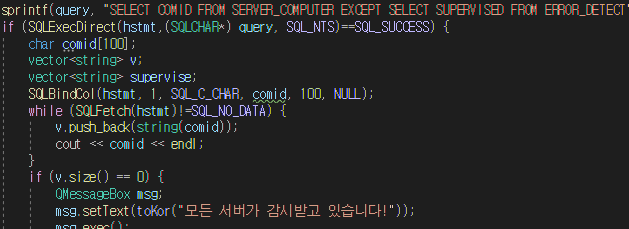
버튼 클릭

결과

서버가 할당된 수가 작은 컴퓨터가 우선 할당됨



사용된 SQL



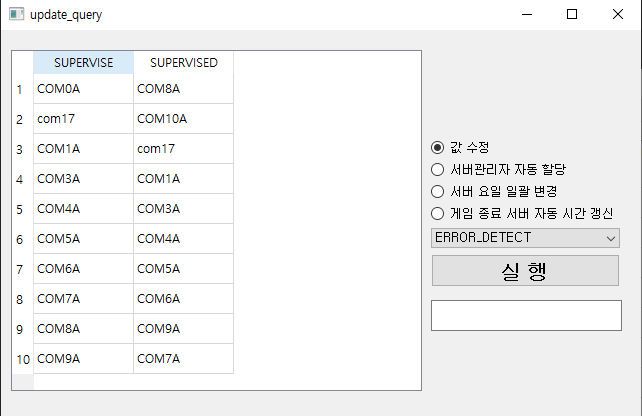






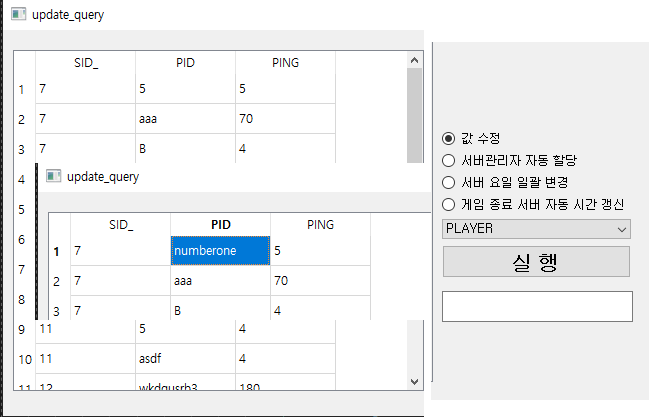


Update Query



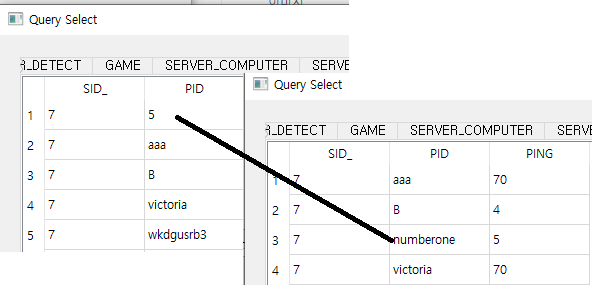
값 수정

GUI에 나타난 테이블의 값을 더블 클릭하여 수정 후 버튼을 누르면 저장됨



2. 버튼 클릭

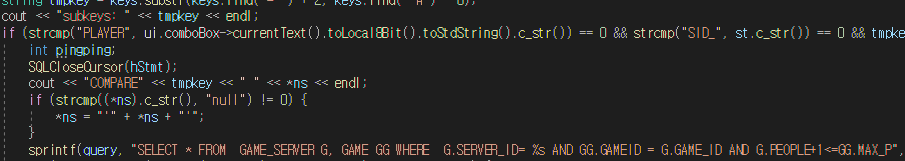
1. 값선택 후 수정

결과

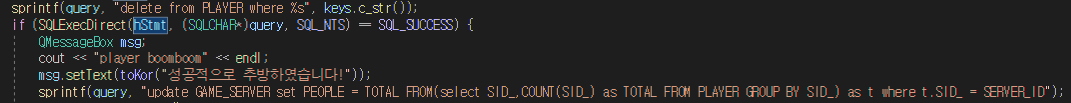
사용된 SQL



키값이 수정될시 수정이 안되는 방법을 해결하기 위해 따로 키값을 저장함







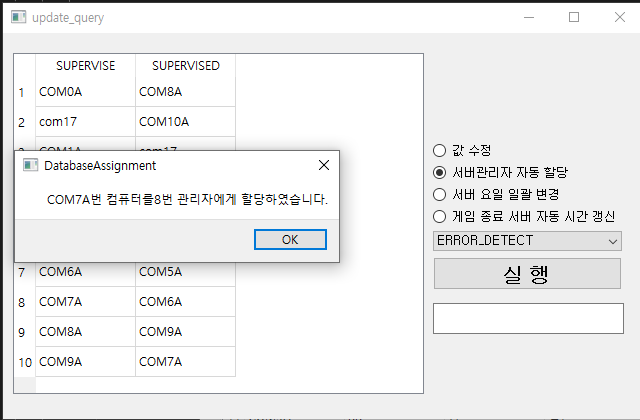
플레이어의 서버를 변경할 시 서버의 최대 핑과 인원수를 비교한 후 수정함



일반적인 수정

서버관리자 자동할당

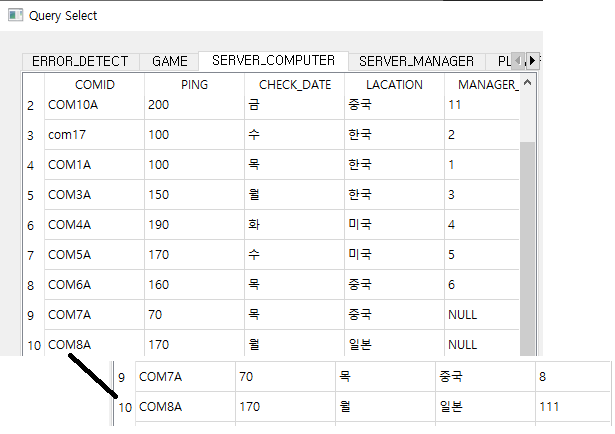
서버 컴퓨터 중 관리자가 할당되지 않은 컴퓨터에게 관리자를 할당함



2.상태확인

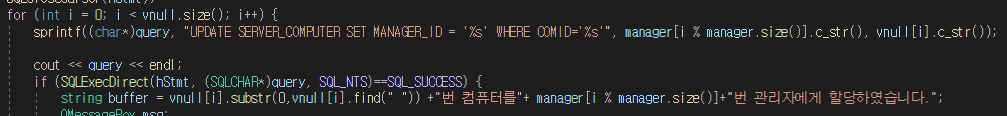
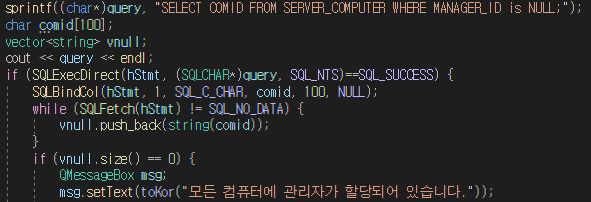
1.버튼클릭

결과



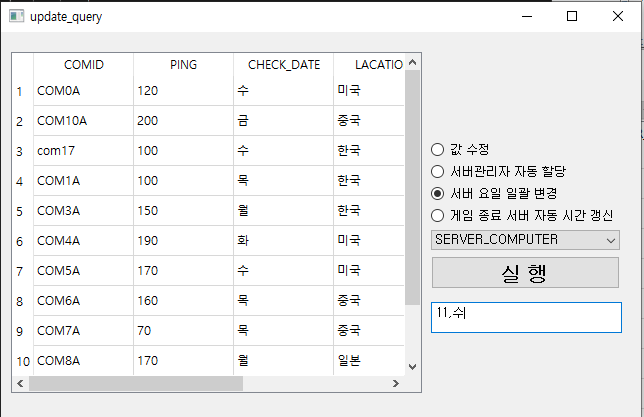
서버 관리자가 할당됨.

사용된 SQL



가장 적은 컴퓨터를 할당 받은 관리자에게 자동으로 할당함

서버 요일 일괄 변경

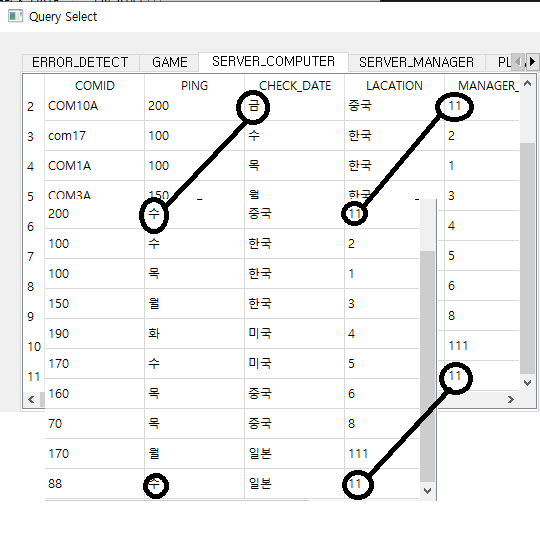


2.버튼 클릭

1.값 입력

기능: 관리자 번호와 변경할 요일을 입력한 후 실행을 누르면 해당되는 서버 컴퓨터의 관리

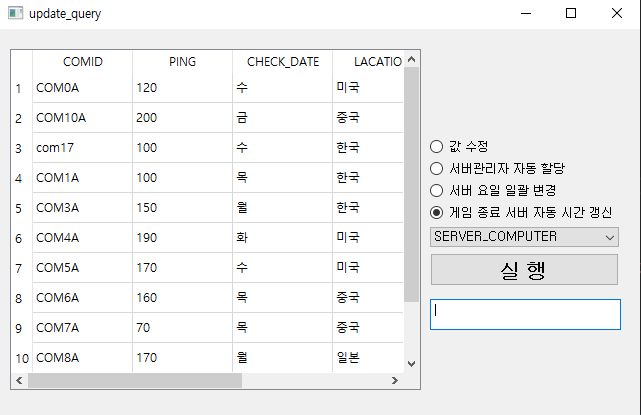
결과: 해당되는 서버 컴퓨터의 요일이 수정됨



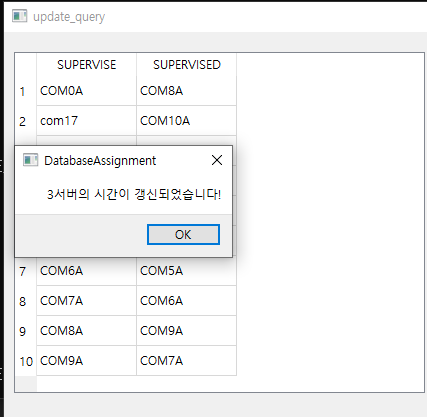
사용된 SQL

게임 종료 서버 자동 시간 갱신

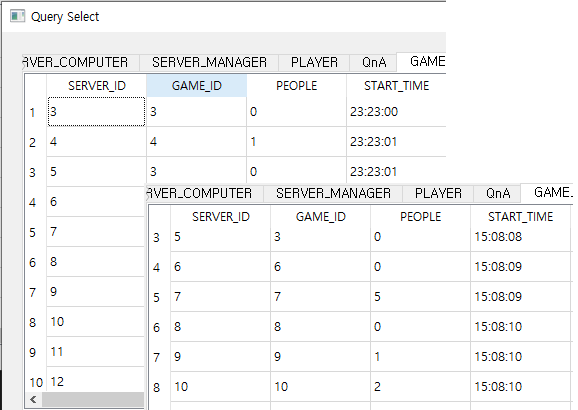


버튼 클릭

기능: 게임 플레이 시간이 끝난 서버의 시간을 현재 시간으로 갱신함

<-갱신되는 서버의 정보가 나옴

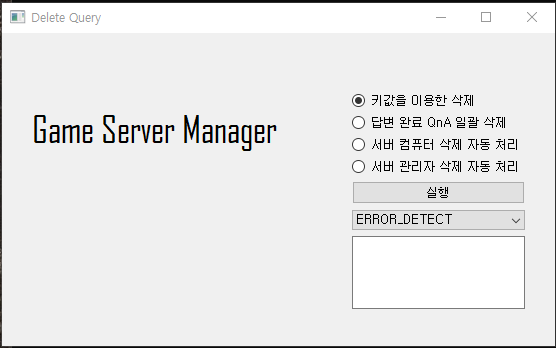
결과: 해당되는 서버의 시간이 모두 변경됨



사용된 SQL

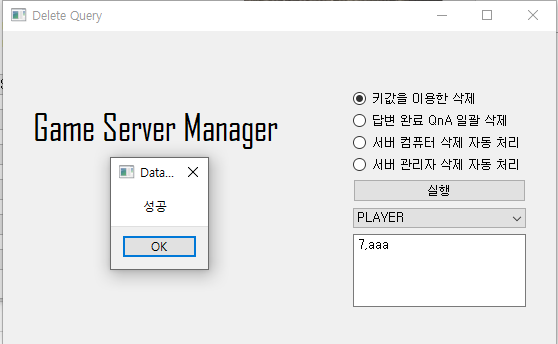
DELETE QUERY



키값을 이용한 삭제

기능: 테이블을 선택한 후 키값을 입력 받아 해당하는 튜플을 삭제함

#시나리오 7번서버 id aaa를 삭제하고 싶을 때

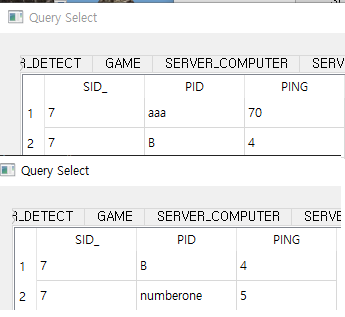


3.실행

1.테이블 선택

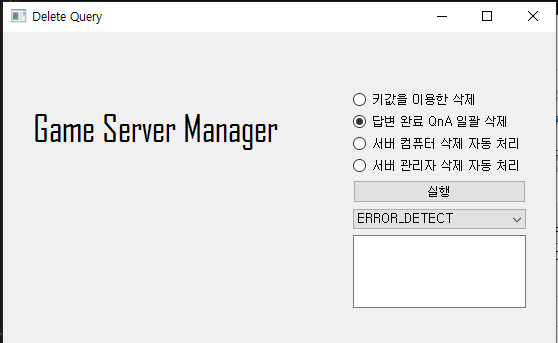
2.기본키 입력

결과: 해당되는 플레이어가 삭제됨



답변 완료 QnA 일괄 삭제

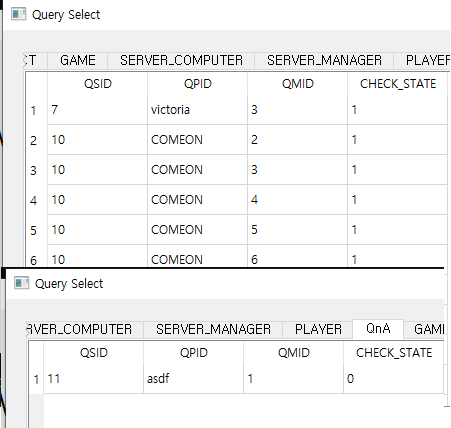
기능: 답변이 완료된(CHECK\_STATE가 1인) 답변들 삭제



2. 버튼 클릭

1.모드 선택

결과: 해당되는 답변들이 삭제됨

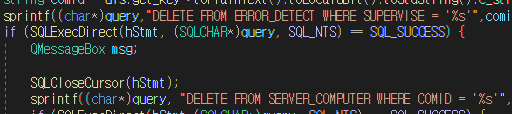


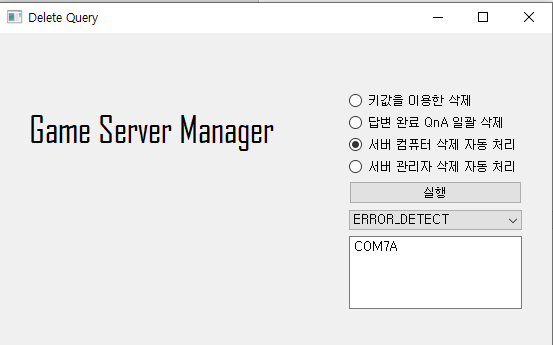
서버 컴퓨터 삭제 자동처리

기능: 서버 컴퓨터가 오류 감지를 담당하고 있으면 삭제가 불가

삭제가 가능할 수 있도록 오류 감지 관계를 모두 해체함

사용된 SQL



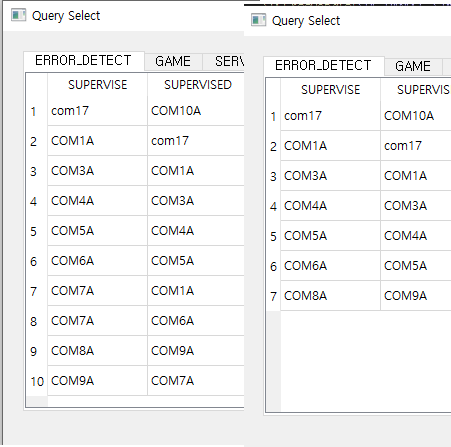
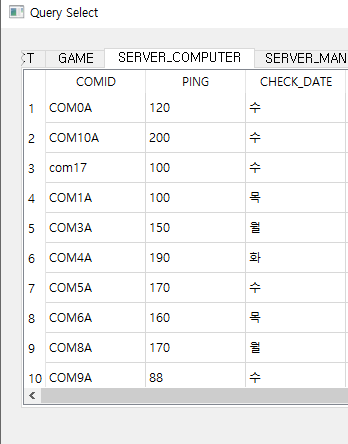


3. 실행

1.모드 선택

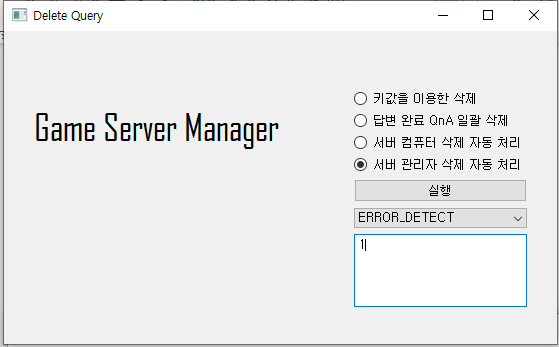
2.컴퓨터 ID 입력

결과: 오류 감지 관계와 해당 서버 컴퓨터가 삭제됨



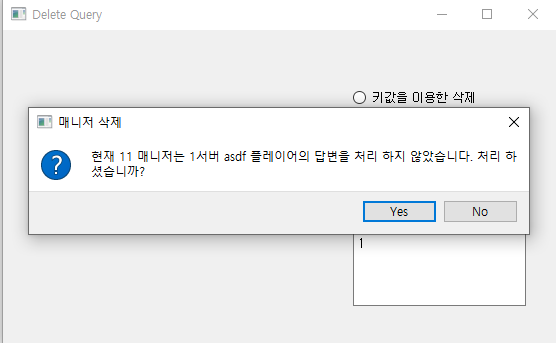
서버 관리자 자동 삭제

기능: 답변이 해결되지 않아 서버매니저를 삭제하지 못하는 경우 답변을 처리하고 서버 매니저를 삭제함



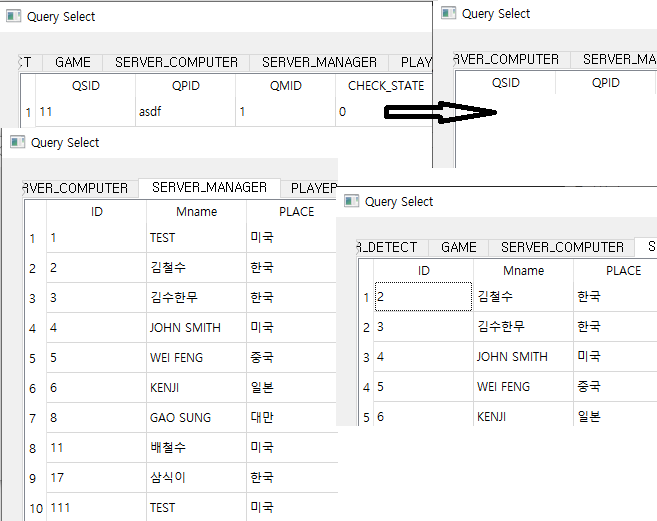
2. 실행 버튼 클릭

1.서버 관리자 기본키 입력



3. 실행

결과: 해당 서버 매니저가 담당하고 있던 답변과 서버 매니저가 삭제되었다.



사용된 SQL



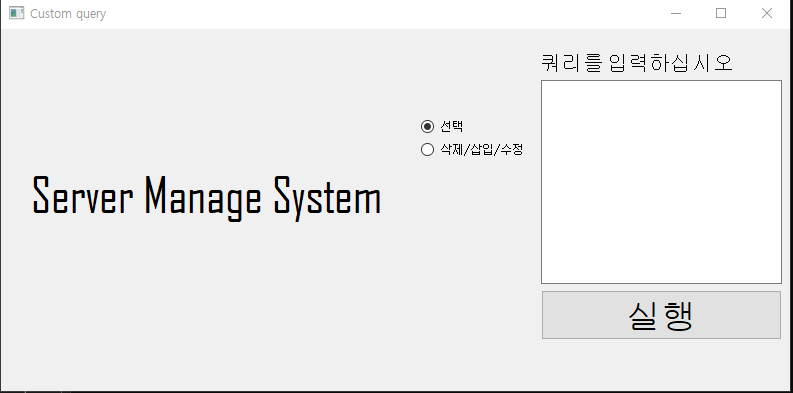
삭제할 답변을 따로 vector에 모아둔다.





Custom Query

기능: 사용자가 직접 쿼리문을 입력한다.



모드 선택

query 입력

버튼 클릭

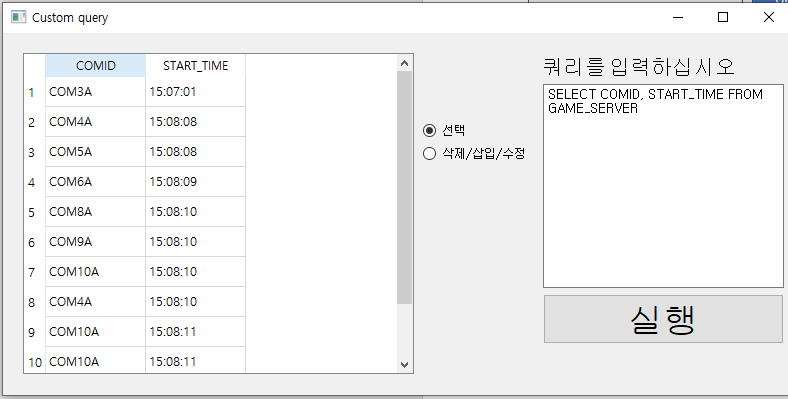
사용된 SQL



외부로부터 쿼리를 입력 받아 이를 실행함

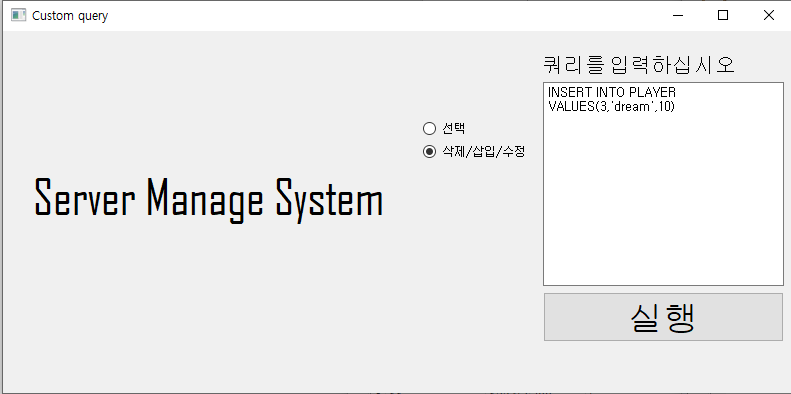
선택 모드 선택 시

실행시 옆에 테이블이 생기면서 결과값 출력

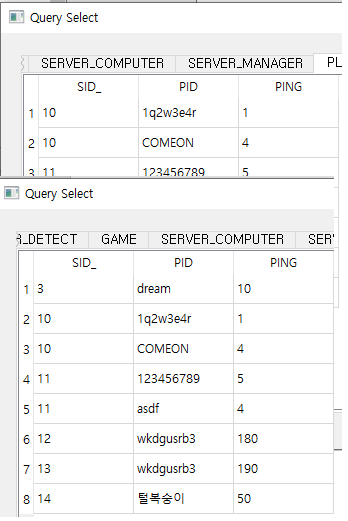


삭제/삽입/수정 모드 선택시

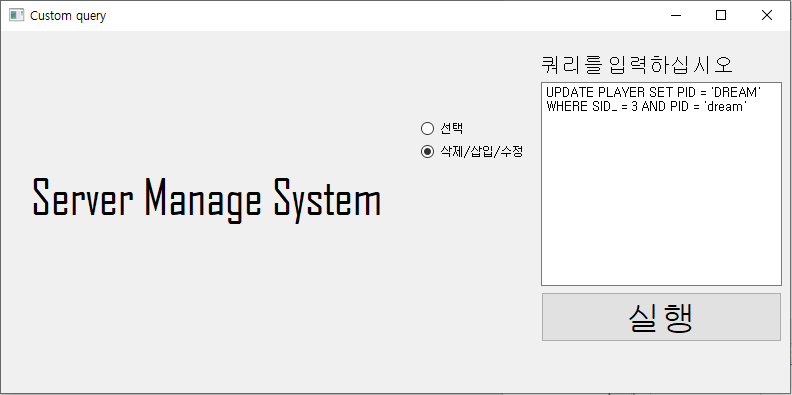
삽입



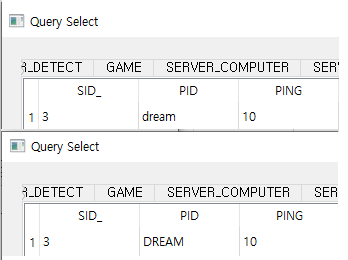
삽입 결과



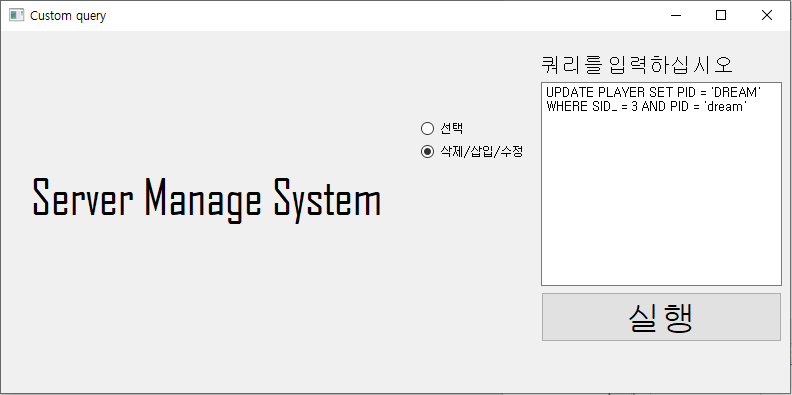
수정

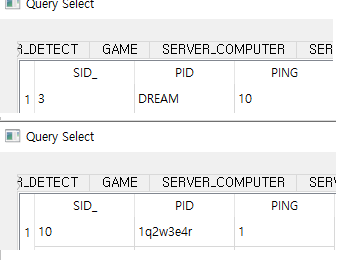


수정 결과



삭제



삭제 결과