**myERP 프로젝트 작업 기획서**

|  |  |  |
| --- | --- | --- |
| **1.** |  | **프로젝트 개요** |

1. 프로젝트명 : myERP
2. 개발목적
   1. 시스템 표준화를 통한 데이터 일관성 유지와 효과적인 인력 재배치
   2. 업무의 효율화, 필요정보의 공유화 및 사원 복지 및 직무 능력 향상 극대화
   3. 효율적인 퇴직자 관리를 이용하여 임직원 휴가 및 휴직 시 대체 인력 수급
3. 개발기간

- 2022.01.15 ~ 현재까지 (진행 중)

1. 개발범위 및 기능
   1. 개발 범위

- 기초 자료, 사원 관리, 가족 사항, 퇴직자 관리, 보고서

* 1. 프로그램 기능
     1. 부서, 직책, 종교 명칭에 대한 회사 업무 맞춤형 커스터마이징
     2. 사원 정보 표준화와 부서 간 인사 이동 구체화
     3. 사원별 부양 가족에 따른 복지 및 관리 체계화
     4. 지속적인 퇴직자 관리를 통한 회사 유휴인력 충원 및 업무 효율성 향상
     5. 직책, 부서 및 퇴직에 따른 인력 이동 및 사유 보고서

1. 개발환경

- Microsoft Visual Studio 2017, MSSQL Server 2019

|  |  |  |
| --- | --- | --- |
| **2.** |  | **프로젝트 세부 내용** |

1. 기본 준수사항

[로그인 후 사용 순서]

1) 기초 정보 입력(최초 사용 시 등록) : 수정, 삭제 가능

- 명칭 및 코드 표준화를 위함

2) 기초 정보 관리에 따른 사원 정보 추가 : 수정, 삭제 가능

- 사원 번호 : 입사년도+부서코드+성별(0:M, 1:F)+일련번호 (ex. 2018PM00003)

- 사원 번호 생성은 추후 자동 생성 및 부여 가능하도록 수정할 예정

- 부서 변경 시 사원 번호에는 변경 사항 없음

- 퇴직자에 대한 부양가족은 DB 내용 유무와 관계 없이 노출되지 않음

3) 사원 정보에 따른 가족 사항 관리

- 퇴직자에 대한 부양가족은 DB 내용 유무와 관계 없이 노출되지 않음

- 추후 가족 구성원 직계 존비속 중복 등록 여부 확인 및 가족 사랑 이벤트

( 꽃다발, 모바일 상품권 등) 추가 예정

4) 퇴직자 관리

- 퇴직 사유에 따른 재임용 및 보충 인력 관리

- 퇴직 상태로 변경 시 퇴직 사유 및 부가 사항 필수 입력

5) 보고서

- 퇴직 처리 후 필수 사항 미입력 사원은 보고서에 표시되지 않음

- 보고서 자료의 pdf 및 excel 문서 변환 기능 추가 예정

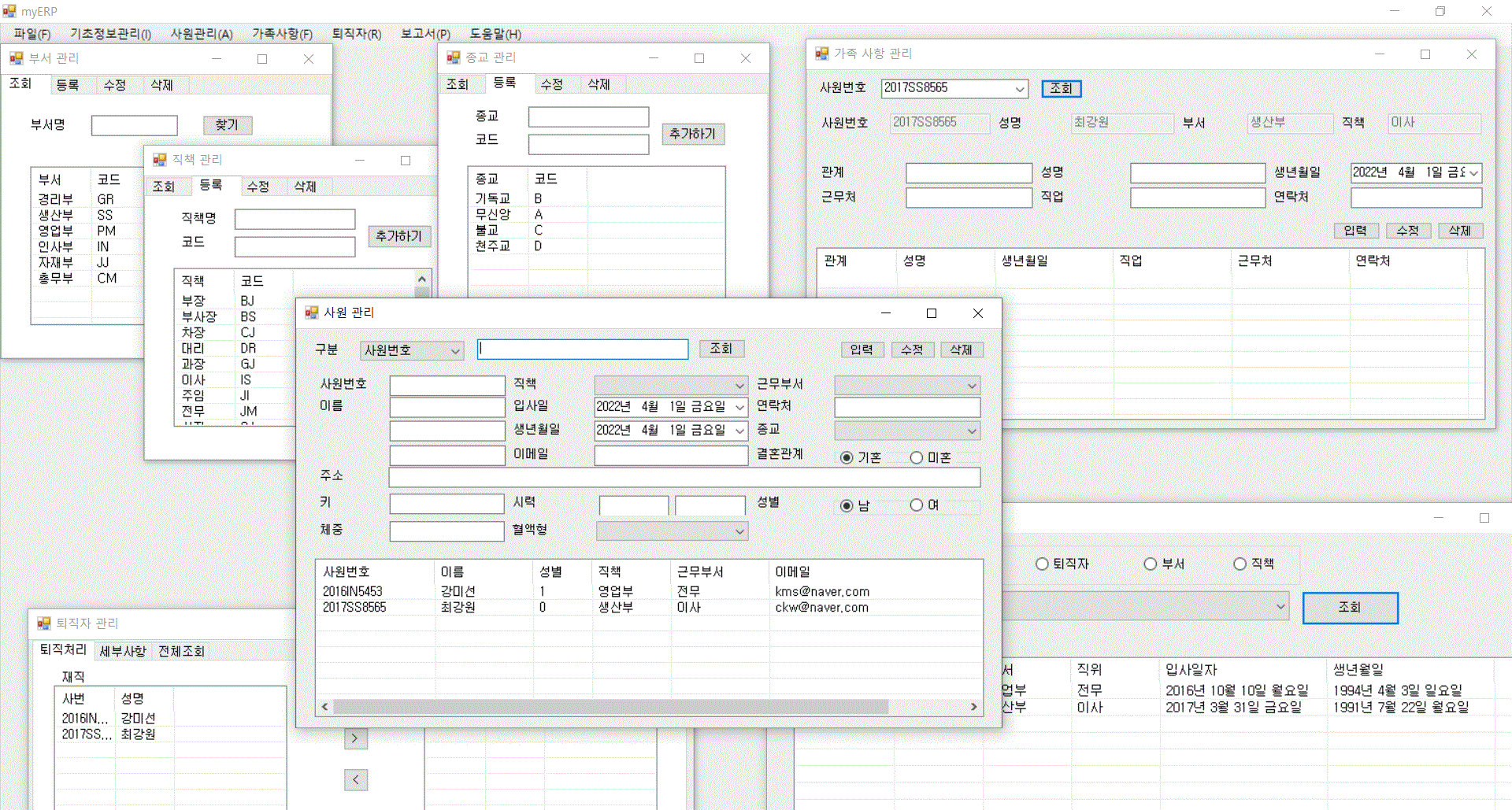
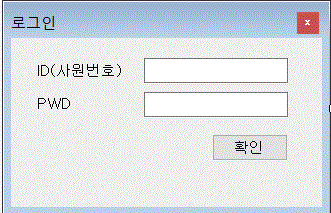
6) 추후 보강 내용

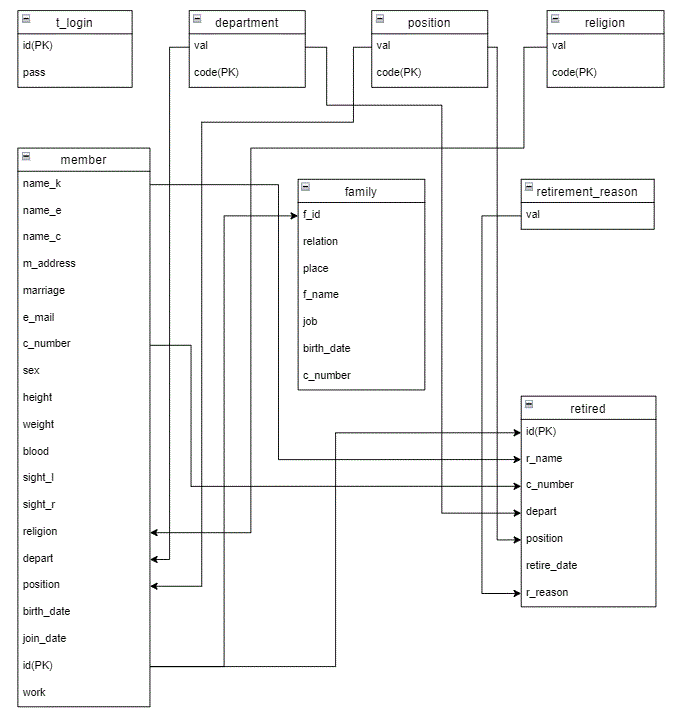
- web 기반의 그룹웨어 형태의 ERP로 모바일 및 UWP 기반 플랫폼 가변화

- 네트워크 및 다중쓰레드를 통한 사내 게시판, 메신저 및 전자 문서 결재 기능

1. 개발환경

|  |  |  |
| --- | --- | --- |
| 구 분 | 시 스 템 환 경 | 비 고 |
| 1 | Windows 10 professional Edition |  |
| 2 | Microstft C# Programming language |  |
| 3 | Microsoft .NET 4.8 FrameWork |  |
| 4 | Microsoft SQL Server 2019 Developer Edition |  |
| 5 | Xmind Mind Map |  |
| 6 | Microsoft VSCode Editor/ERWin Diagram |  |

1. 구축 세부 내용
   1. 프로젝트 구성 및 UI 디자인
   2. 데이터베이스 스키마 구성



* 1. 관리자 기능

|  |  |  |
| --- | --- | --- |
| 구분 | 내용 | 비고 |
| 접근방법 | SSO 인증 ( 초기 ID: mgr, Pass: 1234) |  |
| 관리내용 | 기초 정보, 사원, 가족 사항, 퇴직자, 보고서 관리 |  |
| 사원관리 | 사원번호, 성명, 연락처, 부서, 직책 |  |

1. 취약점 점검항목

|  |  |  |
| --- | --- | --- |
| 구분 | 내용 | 비고 |
| 퇴직자 | 퇴직자 신규 추가 시 세부 사항 미입력 시 보고서에서 조회 불가 |  |
| 사원 정보 | 동일한 사번으로 등록 시 등록 불가 |  |

1. 프로세스 정의서

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 작 성 일 | 2022.02.28 | | 수 정 일 |  | |
| 기 능 명 | 로그인 | | Form 명 | login | |
| 프로세스명 | | 프로세스설명 | | | 관련엔티티 |
| txtClear() | | login\_state 데이터 초기화 | | |  |
| btn\_Login\_Click | | 로그인 버튼 클릭 시 id, pw 확인 후 화면 전환 | | |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 작 성 일 | 2022.03.01 | | 수 정 일 |  | |
| 기 능 명 | myERP | | Form 명 | ERP\_main | |
| 프로세스명 | | 프로세스설명 | | | 관련엔티티 |
| aboutMyERPToolStripMenuItem\_Click | | Department Form 활성화 | | |  |
| 직책ToolStripMenuItem\_Click | | Position Form 활성화 | | |  |
| 종교ToolStripMenuItem\_Click | | Religion Form 활성화 | | |  |
| 사원등록RToolStripMenuItem\_Click | | Member Form 활성화 | | |  |
| 가족사항FToolStripMenuItem\_Click | | Family Form 활성화 | | |  |
| 퇴직자RToolStripMenuItem\_Click | | Retired Form 활성화 | | |  |
| 보고서ToolStripMenuItem\_Click | | Report Form 활성화 | | |  |
| 끝내기tCtrlXToolStripMenuItem\_Click | | myERP Form 종료 | | |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 작 성 일 | 2022.03.02 | | 수 정 일 |  | |
| 기 능 명 | 부서 관리 | | Form 명 | Department | |
| 프로세스명 | | 프로세스설명 | | | 관련엔티티 |
| btn\_d\_search\_Click | | DB에서 textbox와 일치하는 값 불러옴 | | |  |
| btn\_d\_add\_Click | | textbox에 입력된 값을 읽어 DB에 저장 | | |  |
| lvDepart\_u\_SelectedIndexChanged | | listview에서 선택된 값의 index 저장 | | |  |
| btn\_d\_update\_Click | | textbox에 입력된 값으로 DB 수정 | | |  |
| d\_tabMenu\_SelectedIndexChanged | | tab 이동 시 각 tab에 있는 listview에 DB 데이터를 다시 select | | |  |
| lvDepart\_d\_SelectedIndexChanged | | listview에서 선택된 값의 index 저장 | | |  |
| btn\_d\_delete\_Click | | DB에서 사용중인 코드가 있으면 삭제 불가 messagebox, 없으면 삭제 | | |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 작 성 일 | 2022.03.02 | | 수 정 일 |  | |
| 기 능 명 | 직책 관리 | | Form 명 | Position | |
| 프로세스명 | | 프로세스설명 | | | 관련엔티티 |
| btn\_p\_search\_Click | | DB에서 textbox와 일치하는 값 불러옴 | | |  |
| btn\_p\_add\_Click | | textbox에 입력된 값을 읽어 DB에 저장 | | |  |
| lvPosition\_u\_SelectedIndexChanged | | listview에서 선택된 값의 index 저장 | | |  |
| btn\_p\_update\_Click | | textbox에 입력된 값으로 DB 수정 | | |  |
| p\_TabMenu\_SelectedIndexChanged | | tab 이동 시 각 tab에 있는 listview 값을 DB에서 다시 select | | |  |
| lvPosition\_d\_SelectedIndexChanged | | listview에서 선택된 값의 index 저장 | | |  |
| btn\_delete\_Click | | DB에서 사용중인 코드가 있으면 삭제 불가 messagebox, 없으면 삭제 | | |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 작 성 일 | 2022.03.02 | | 수 정 일 |  | |
| 기 능 명 | 종교 관리 | | Form 명 | Religion | |
| 프로세스명 | | 프로세스설명 | | | 관련엔티티 |
| btn\_r\_search\_Click | | DB에서 textbox와 일치하는 값 불러옴 | | |  |
| btn\_r\_add\_Click | | textbox에 입력된 값을 읽어 DB에 저장 | | |  |
| lvReligion\_u\_SelectedIndexChanged | | listview에서 선택된 값의 index 저장 | | |  |
| btn\_r\_update\_Click | | textbox에 입력된 값으로 DB 수정 | | |  |
| r\_TabMenu\_SelectedIndexChanged | | tab 이동 시 각 tab에 있는 listview 값을 DB에서 다시 select | | |  |
| lvReligion\_d\_SelectedIndexChanged | | listview에서 선택된 값의 index 저장 | | |  |
| btn\_r\_delete\_Click | | DB에서 사용중인 코드가 있으면 삭제 불가 messagebox, 없으면 삭제 | | |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 작 성 일 | 2022.03.04 | | 수 정 일 |  | |
| 기 능 명 | 사원 관리 | | Form 명 | Member | |
| 프로세스명 | | 프로세스설명 | | | 관련엔티티 |
| textClear | | textbox 값을 공백으로 초기화 | | |  |
| new\_Show | | DB에서 값을 가져와 listview에 출력 | | |  |
| Member\_Load | | form이 load될 때 DB에서 값을 가져와 combobox 항목으로 추출 | | |  |
| btn\_m\_insert\_Click | | 입력된 정보들을 DB에 저장 | | |  |
| lvMember\_SelectedIndexChanged | | listview에서 선택된 값에 일치하는 정보를 textbox, combobox, radiobutton 등에 출력 | | |  |
| btn\_m\_update\_Click | | textbox의 값을 읽어 DB 데이터 수정 | | |  |
| btn\_m\_delete\_Click | | index에 일치하는 DB 데이터 삭제 | | |  |
| cb\_section\_SelectedIndexChanged | | combobox에서 선택된 값의 index 저장 | | |  |
| btn\_m\_search\_Click | | index(분류)를 기반으로 DB에서 일치하는 분류에 따른 값을 listview에 출력 | | |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 작 성 일 | 2022.03.07 | | 수 정 일 |  | |
| 기 능 명 | 가족 관리 | | Form 명 | Family | |
| 프로세스명 | | 프로세스설명 | | | 관련엔티티 |
| textClear | | textbox 값을 공백으로 초기화 | | |  |
| new\_show | | DB에서 값을 가져와 listview에 출력 | | |  |
| btn\_f\_search\_Click | | combobox에서 선택된 값과 일치하는 DB 데이터를 추출해 listview에 출력 | | |  |
| Family\_Load | | form이 load될 때 DB에서 데이터 추출하여 combobox에 항목 생성 | | |  |
| lvFamily\_SelectedIndexChanged | | listview에서 선택된 값의 index를 찾아 그에 따른 데이터를 textbox, datetimepicker에 출력 | | |  |
| btn\_f\_insert\_Click | | textbox, combobox 등에 입력된 값을 DB에 저장 | | |  |
| btn\_f\_update\_Click | | textbox, combobox 등에 입력된 값으로 DB 수정 | | |  |
| btn\_f\_delete\_Click | | index에 일치하는 값을 DB에서 삭제 | | |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 작 성 일 | 2022.03.10 | | 수 정 일 |  | |
| 기 능 명 | 퇴직자 관리 | | Form 명 | Retired | |
| 프로세스명 | | 프로세스설명 | | | 관련엔티티 |
| textClear | | textbox 값을 공백으로 초기화 | | |  |
| radio\_state | | radiobutton 상태에 따른 SQL 적용하여 DB에서 추출 후 listview에 데이터 출력 | | |  |
| new\_show | | DB에서 데이터 추출 후 listview에 데이터 출력 | | |  |
| Retired\_Load | | form이 load되면 DB에서 데이터 추출 후 listview, combobox에 출력 및 항목 추가 | | |  |
| btn\_r\_search\_Click | | combobox에서 선택한 값에 일치하는 데이터를 DB에서 추출하여 combobox, textbox에 출력 | | |  |
| btn\_AU\_add\_Click | | combobox 및 textbox 등의 값을 읽어 DB에 저장 | | |  |
| btn\_AU\_update\_Click | | datetimepicker, combobx의 값을 DB에서 수정 | | |  |
| rb\_S\_reason\_CheckedChange | | radio\_state 함수 호출하여 이벤트 실행 | | |  |
| rb\_S\_depart\_CheckedChanged | | radio\_state 함수 호출하여 이벤트 실행 | | |  |
| rb\_S\_date\_CheckedChanged | | radio\_state 함수 호출하여 이벤트 실행 | | |  |
| rb\_S\_position\_CheckedChanged | | radio\_state 함수 호출하여 이벤트 실행 | | |  |
| btn\_add\_Click | | 재직 상태였던 사원 정보를 퇴직 상태로 변경하고 DB에 적용 후 세부사항 입력 message 띄움 | | |  |
| btn\_del\_Click | | 퇴직 상태였던 사원 정보를 재직 상태로 변경하고 DB의 퇴직 정보 삭제 및 재직 상태 적용 | | |  |
| lv\_served\_SelectedIndexChanged | | listview에서 선택된 값의 index 저장 | | |  |
| tabControl1\_SelectedIndexChanged | | tab이 변경될 때 퇴직 상태인 사원번호를 표시하는 combobox에 DB값 불러오기 | | |  |
| lv\_retired\_SelectedIndexChanged | | listview에서 선택된 값의 index 저장 | | |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 작 성 일 | 2022.03.14 | | 수 정 일 |  | |
| 기 능 명 | 보고서 | | Form 명 | Report | |
| 프로세스명 | | 프로세스설명 | | | 관련엔티티 |
| rb\_total\_CheckedChanged | | '전체' radiobutton 선택 시 사원 정보를 출력하기 위한 listview의 column을 추가 | | |  |
| rb\_reason\_CheckedChanged | | '퇴직자' radiobutton 선택 시 combobox에 DB에서 추출한 퇴직 사유를 항목으로 표시 및 퇴직 사유에 따른 퇴직자 출력을 위한 listview의 column 추가 | | |  |
| rb\_depart\_CheckedChanged | | '부서' radiobutton 선택 시 combobox에 DB에서 추출한 부서 종류를 항목으로 표시 및 부서별 사원 정보 출력을 위한 listview의 column 추가 | | |  |
| rb\_position\_CheckedChanged | | '직책' radiobutton 선택 시 combobox에 DB에서 추출한 직책 종류를 항목으로 표시 및 직책별 사원 정보 출력을 위한 listview column추가 | | |  |
| btn\_report\_Click | | radiobutton에서 선택된 값에 따른 데이터를 DB에서 추출 후 listview에 출력 | | |  |

1. 클래스 설계서

|  |  |  |  |
| --- | --- | --- | --- |
| 클래스 다이어그램(기본) | | | |
| 서브시스템명 | Visual Studio 2017 | 작성일 | 2022.02.20 |
|  | | | |

1. 클래스 명세

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 클래스 명세 | | | | | | |
| 서브시스템명 | Visual Studio 2017 | | 작성일 | | | 2022.02.28 |
| 클래스명 | login | | | | | |
| 메소드 설명 | | | | | | |
| 메소드명 | txtClear | 리턴  타입 |  | 파라미터 |  | |
| 설명 | id, password textbox 빈칸으로 초기화 | | | |
| 메소드명 | btn\_Login\_  Click | 리턴  타입 |  | 파라미터 | Object sender,  EventArgs e | |
| 설명 |  | | | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 클래스 명세 | | | | | | |
| 서브시스템명 | Visual Studio 2017 | | 작성일 | | | 2022.03.01 |
| 클래스명 | myERP | | | | | |
| 메소드 설명 | | | | | | |
| 메소드명 | aboutMyERP  ToolStripMenuItem\_Click | 리턴  타입 |  | 파라미터 | object sender,  EventArgs e | |
| 설명 | 기초정보관리 메뉴의 부서를 누르면 Department Form 실행 | | | |
| 메소드명 | 직책ToolStrip  MenuItem\_  Click | 리턴  타입 |  | 파라미터 | Object sender,  EventArgs e | |
| 설명 | 기초정보관리 메뉴의 직책을 누르면 Position Form 실행 | | | |
| 메소드명 | 종교ToolStrip  MenuItem\_  Click | 리턴  타입 |  | 파라미터 | Object sender,  EventArgs e | |
| 설명 | 기초정보관리 메뉴의 종교를 누르면 Religion Form 실행 | | | |
| 메소드명 | 사원등록R  ToolStripMenuItem\_Click | 리턴  타입 |  | 파라미터 | Object sender,  EventArgs e | |
| 설명 | 사원등록 메뉴를 누르면  Member Form 실행 | | | |
| 메소드명 | 가족사항F  ToolStripMenuItem\_Click | 리턴  타입 |  | 파라미터 | Object sender,  EventArgs e | |
| 설명 | 가족사항 메뉴를 누르면  Family Form 실행 | | | |
| 메소드명 | 퇴직자R  ToolStripMenuItem\_Click | 리턴  타입 |  | 파라미터 | Object sender,  EventArgs e | |
| 설명 | 퇴직자 메뉴를 누르면  Retired Form 실행 | | | |
| 메소드명 | 보고서ToolStripMenuItem  \_Click | 리턴  타입 |  | 파라미터 | Object sender,  EventArgs e | |
| 설명 | 보고서 메뉴를 누르면  Report Form 실행 | | | |
| 메소드명 | 끝내기tCtrlX  ToolStripMenuItem\_Click | 리턴  타입 |  | 파라미터 | Object sender,  EventArgs e | |
| 설명 | myERP 종료 | | | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 클래스 명세 | | | | | | |
| 서브시스템명 | Visual Studio 2017 | | 작성일 | | | 2022.03.02 |
| 클래스명 | Department | | | | | |
| 메소드 설명 | | | | | | |
| 메소드명 | btn\_d\_search\_Click | 리턴  타입 |  | 파라미터 | object sender,  EventArgs e | |
| 설명 | DB에서 textbox에 입력한 부서명과 일치하는 데이터를 listview에 출력 | | | |
| 메소드명 | btn\_d\_add  \_Click | 리턴  타입 |  | 파라미터 | Object sender,  EventArgs e | |
| 설명 | textbox에 입력된 부서명, 코드를 읽어 DB에 저장 | | | |
| 메소드명 | lvDepart\_u\_  SelectedIndexChanged | 리턴  타입 |  | 파라미터 | Object sender,  EventArgs e | |
| 설명 | listview에서 선택된 값의 index를 숨겨진 textbox에 저장 | | | |
| 메소드명 | btn\_d\_update\_Click | 리턴  타입 |  | 파라미터 | Object sender,  EventArgs e | |
| 설명 | textbox에 입력된 부서명, 코드로  DB 값을 수정 | | | |
| 메소드명 | d\_tabMenu\_  SelectedIndexChanged | 리턴  타입 |  | 파라미터 | Object sender,  EventArgs e | |
| 설명 | tab 이동 시 추가, 수정된 값을 보여주기 위해 각 tab에 있는 listview에 DB 데이터를 다시 select | | | |
| 메소드명 | lvDepart\_d\_  SelectedIndexChanged | 리턴  타입 |  | 파라미터 | Object sender,  EventArgs e | |
| 설명 | listview에서 선택된 값의 index를  숨겨진 textbox에 저장 | | | |
| 메소드명 | btn\_d\_delete  \_Click | 리턴  타입 |  | 파라미터 | Object sender,  EventArgs e | |
| 설명 | DB에서 사용중인 부서명일 경우에는 삭제 불가능 message를 띄우고, 아닐 경우는 DB에서 해당 부서, 코드 삭제 | | | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 클래스 명세 | | | | | | |
| 서브시스템명 | Visual Studio 2017 | | 작성일 | | | 2022.03.02 |
| 클래스명 | Position | | | | | |
| 메소드 설명 | | | | | | |
| 메소드명 | btn\_p\_search\_Click | 리턴  타입 |  | 파라미터 | object sender,  EventArgs e | |
| 설명 | DB에서 textbox에 입력한 직책명과 일치하는 데이터를 listview에 출력 | | | |
| 메소드명 | btn\_p\_add  \_Click | 리턴  타입 |  | 파라미터 | Object sender,  EventArgs e | |
| 설명 | textbox에 입력된 직책명, 코드를 읽어 DB에 저장 | | | |
| 메소드명 | lvPosition\_u\_  SelectedIndexChanged | 리턴  타입 |  | 파라미터 | Object sender,  EventArgs e | |
| 설명 | listview에서 선택된 값의 index를 숨겨진 textbox에 저장 | | | |
| 메소드명 | btn\_p\_update\_Click | 리턴  타입 |  | 파라미터 | Object sender,  EventArgs e | |
| 설명 | textbox에 입력된 직책명, 코드로  DB 값을 수정 | | | |
| 메소드명 | p\_tabMenu\_  SelectedIndexChanged | 리턴  타입 |  | 파라미터 | Object sender,  EventArgs e | |
| 설명 | tab 이동 시 추가, 수정된 값을 보여주기 위해 각 tab에 있는 listview에 DB 데이터를 다시 select | | | |
| 메소드명 | lvPosition\_d\_  SelectedIndexChanged | 리턴  타입 |  | 파라미터 | Object sender,  EventArgs e | |
| 설명 | listview에서 선택된 값의 index를  숨겨진 textbox에 저장 | | | |
| 메소드명 | btn\_delete  \_Click | 리턴  타입 |  | 파라미터 | Object sender,  EventArgs e | |
| 설명 | DB에서 사용중인 직책명일 경우에는 삭제 불가능 message를 띄우고, 아닐 경우는 DB에서 해당 직책, 코드 삭제 | | | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 클래스 명세 | | | | | | |
| 서브시스템명 | Visual Studio 2017 | | 작성일 | | | 2022.03.02 |
| 클래스명 | Religion | | | | | |
| 메소드 설명 | | | | | | |
| 메소드명 | btn\_r\_search\_Click | 리턴  타입 |  | 파라미터 | object sender,  EventArgs e | |
| 설명 | DB에서 textbox에 입력한 종교명과 일치하는 데이터를 listview에 출력 | | | |
| 메소드명 | btn\_r\_add  \_Click | 리턴  타입 |  | 파라미터 | Object sender,  EventArgs e | |
| 설명 | textbox에 입력된 종교명, 코드를 읽어 DB에 저장 | | | |
| 메소드명 | lvReligion\_u\_  SelectedIndexChanged | 리턴  타입 |  | 파라미터 | Object sender,  EventArgs e | |
| 설명 | listview에서 선택된 값의 index를 숨겨진 textbox에 저장 | | | |
| 메소드명 | btn\_r\_update\_Click | 리턴  타입 |  | 파라미터 | Object sender,  EventArgs e | |
| 설명 | textbox에 입력된 종교명, 코드로  DB 값을 수정 | | | |
| 메소드명 | r\_tabMenu\_  SelectedIndexChanged | 리턴  타입 |  | 파라미터 | Object sender,  EventArgs e | |
| 설명 | tab 이동 시 추가, 수정된 값을 보여주기 위해 각 tab에 있는 listview에 DB 데이터를 다시 select | | | |
| 메소드명 | lvReligion\_d\_  SelectedIndexChanged | 리턴  타입 |  | 파라미터 | Object sender,  EventArgs e | |
| 설명 | listview에서 선택된 값의 index를  숨겨진 textbox에 저장 | | | |
| 메소드명 | btn\_r\_delete  \_Click | 리턴  타입 |  | 파라미터 | Object sender,  EventArgs e | |
| 설명 | DB에서 사용중인 종교명일 경우에는 삭제 불가능 message를 띄우고, 아닐 경우는 DB에서 해당 종교, 코드 삭제 | | | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 클래스 명세 | | | | | | |
| 서브시스템명 | Visual Studio 2017 | | 작성일 | | | 2022.03.04 |
| 클래스명 | Member | | | | | |
| 메소드 설명 | | | | | | |
| 메소드명 | txtClear | 리턴  타입 |  | 파라미터 |  | |
| 설명 | Member Form에 존재하는 textbox를 모두 빈 칸으로 설정(초기화) | | | |
| 메소드명 | new\_show | 리턴  타입 |  | 파라미터 |  | |
| 설명 | 정보 변경 후 DB 값을 새로 가져오기 위한 select 및 listview에 출력 과정 | | | |
| 메소드명 | Member\_Load | 리턴  타입 |  | 파라미터 | Object sender,  EventArgs e | |
| 설명 | Member Form이 Load될 때 DB에서 가져온 값을 combobox 항목으로 추출 | | | |
| 메소드명 | btn\_m\_insert\_Click | 리턴  타입 |  | 파라미터 | Object sender,  EventArgs e | |
| 설명 | textbox, combobox 등에 입력된 값들을 DB에 insert | | | |
| 메소드명 | lvMember\_  SelectedIndexChanged | 리턴  타입 |  | 파라미터 | Object sender,  EventArgs e | |
| 설명 | listview에서 선택된 값에 일치하는 정보를 DB에서 select 하여  textbox, combobox 등에 출력 | | | |
| 메소드명 | bnt\_m\_update\_Click | 리턴  타입 |  | 파라미터 | Object sender,  EventArgs e | |
| 설명 | textbox, combobox 등의 값을 읽어 DB에서 데이터를 수정 | | | |
| 메소드명 | btn\_m\_delete\_Click | 리턴  타입 |  | 파라미터 | Object sender,  EventArgs e | |
| 설명 | listview에서 선택된 값의 인덱스에 일치하는 정보를 member, family 테이블에서 삭제 | | | |
| 메소드명 | cb\_section\_  SelectedIndexChanged | 리턴  타입 |  | 파라미터 | Object sender,  EventArgs e | |
| 설명 | combobox에서 선택된 값의  index를 저장 | | | |
| 메소드명 | btn\_m\_search\_Click | 리턴  타입 |  | 파라미터 | Object sender,  EventArgs e | |
| 설명 | radiobutton에서 선택된 값에 따라 분류 기준을 달리하여 그에 따른 값을 listview에 출력 | | | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 클래스 명세 | | | | | | |
| 서브시스템명 | Visual Studio 2017 | | 작성일 | | | 2022.03.07 |
| 클래스명 | Family | | | | | |
| 메소드 설명 | | | | | | |
| 메소드명 | txtClear | 리턴  타입 |  | 파라미터 |  | |
| 설명 | Family Form에 존재하는 textbox를 모두 빈 칸으로 설정(초기화) | | | |
| 메소드명 | new\_show | 리턴  타입 |  | 파라미터 |  | |
| 설명 | 정보 변경 후 DB 값을 새로 가져오기 위한 select 및 listview에 출력 과정 | | | |
| 메소드명 | btn\_f\_search\_Click | 리턴  타입 |  | 파라미터 | Object sender,  EventArgs e | |
| 설명 | 찾는 값에 일치하는 정보를 DB에서 select 하여 listview에 출력 | | | |
| 메소드명 | Family\_Load | 리턴  타입 |  | 파라미터 | Object sender,  EventArgs e | |
| 설명 | Family Form이 Load될 때 DB에서 데이터를 select하여 combobox의 항목으로 추가 | | | |
| 메소드명 | lvFamily\_  SelectedIndexChanged | 리턴  타입 |  | 파라미터 | Object sender,  EventArgs e | |
| 설명 | listview에서 선택된 값에 해당하는 index를 찾아 그에 대한 정보를 textbox, datetimepicker 등에 출력 | | | |
| 메소드명 | btn\_f\_insert  \_Click | 리턴  타입 |  | 파라미터 | Object sender,  EventArgs e | |
| 설명 | combobox, textbox 등에 입력한 값을 DB에 insert | | | |
| 메소드명 | btn\_f\_update\_Click | 리턴  타입 |  | 파라미터 | Object sender,  EventArgs e | |
| 설명 | combobox, textbox 등에 입력한 갑을 DB에서 update | | | |
| 메소드명 | btn\_f\_delete  \_Click | 리턴  타입 |  | 파라미터 | Object sender,  EventArgs e | |
| 설명 | listview에서 클릭한 인덱스의 값과 일치하는 데이터를 DB에서 찾아 delete | | | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 클래스 명세 | | | | | | |
| 서브시스템명 | Visual Studio 2017 | | 작성일 | | | 2022.03.10 |
| 클래스명 | Retired | | | | | |
| 메소드 설명 | | | | | | |
| 메소드명 | txtClear | 리턴  타입 |  | 파라미터 |  | |
| 설명 | Retired Form에 존재하는 textbox를 모두 빈 칸으로 설정(초기화) | | | |
| 메소드명 | radio\_state | 리턴  타입 |  | 파라미터 |  | |
| 설명 | radiobutton 상태에 따른 SQL문을 적용하여 DB에서 select하여  listview에 출력 | | | |
| 메소드명 | new\_show | 리턴  타입 |  | 파라미터 |  | |
| 설명 | 정보 변경 후 DB 값을 새로 가져오기 위한 select 및 listview에 출력 과정 | | | |
| 메소드명 | Retired\_Load | 리턴  타입 |  | 파라미터 | Object sender,  EventArgs e | |
| 설명 | Retired Form이 Load될 때 DB에서 데이터를 select하여 재직자 및 퇴직자 listview, combobox 등에 데이터 출력 | | | |
| 메소드명 | btn\_r\_search\_Click | 리턴  타입 |  | 파라미터 | Object sender,  EventArgs e | |
| 설명 | combobox에서 선택한 값에 일치하는 데이터를 DB에서 추출하여  combobox, textbox 등에 출력 | | | |
| 메소드명 | btn\_AU\_add  \_Click | 리턴  타입 |  | 파라미터 | Object sender,  EventArgs e | |
| 설명 | textbox, combobox 등에 입력된 값을 읽어 DB에 저장 | | | |
| 메소드명 | btn\_AU\_update\_Click | 리턴  타입 |  | 파라미터 | Object sender,  EventArgs e | |
| 설명 | datetimepicker, combobox에 입력된 값을 읽어  DB에서 일치하는 사번 정보 update | | | |
| 메소드명 | rb\_S\_reason\_  CheckedChanged | 리턴  타입 |  | 파라미터 | Object sender,  EventArgs e | |
| 설명 | radio\_state 함수 호출하여 이벤트 실행 | | | |
| 메소드명 | rb\_S\_depart\_  CheckedChanged | 리턴  타입 |  | 파라미터 | Object sender,  EventArgs e | |
| 설명 | radio\_state 함수 호출하여 이벤트 실행 | | | |
| 메소드명 | rb\_S\_date\_  CheckedChanged | 리턴  타입 |  | 파라미터 | Object sender,  EventArgs e | |
| 설명 | radio\_state 함수 호출하여 이벤트 실행 | | | |
| 메소드명 | rb\_S\_position\_CheckedChanged | 리턴  타입 |  | 파라미터 | Object sender,  EventArgs e | |
| 설명 | radio\_state 함수 호출하여 이벤트 실행 | | | |
| 메소드명 | btn\_add\_Click | 리턴  타입 |  | 파라미터 | Object sender,  EventArgs e | |
| 설명 | 재직 상태였던 사원 정보를 퇴직 상태로 변경하고 DB에 적용 후 세부사항 입력 message를 띄움 | | | |
| 메소드명 | btn\_del\_Click | 리턴  타입 |  | 파라미터 | Object sender,  EventArgs e | |
| 설명 | 퇴직 상태였던 사원 정보를 재직 상태로 변경하고 DB의 퇴직 정보 삭제 및 재직 상태 적용 | | | |
| 메소드명 | lv\_served\_  SelectedIndexChanged | 리턴  타입 |  | 파라미터 | Object sender,  EventArgs e | |
| 설명 | listview에서 선택된 값의 인덱스를 저장 | | | |
| 메소드명 | tabControl\_1\_  SelectedIndexChanged | 리턴  타입 |  | 파라미터 | Object sender,  EventArgs e | |
| 설명 | tab이 변경될 때 퇴직 상태인 사원 정보를 표시하는 combobox에 DB값 불러오기 | | | |
| 메소드명 | lv\_retired\_  SelectedIndexChanged | 리턴  타입 |  | 파라미터 | Object sender,  EventArgs e | |
| 설명 | listview에서 선택된 값의 index를 저장 | | | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 클래스 명세 | | | | | | |
| 서브시스템명 | Visual Studio 2017 | | 작성일 | | | 2022.03.14 |
| 클래스명 | Report | | | | | |
| 메소드 설명 | | | | | | |
| 메소드명 | rb\_total\_  CheckedChanged | 리턴  타입 |  | 파라미터 | Object sender,  EventArgs e | |
| 설명 | '전체' radiobutton 선택 시 사원 정보를 출력하기 위한 listview column을 추가 | | | |
| 메소드명 | rb\_reason\_  CheckedChanged | 리턴  타입 |  | 파라미터 | Object sender,  EventArgs e | |
| 설명 | '퇴직자' radiobutton 선택 시 combobox에 DB에서 추출한 퇴직 사유를 항목으로 표시 및 퇴직 사유에 따른 퇴직자 출력을 위한 listview의 column 추가 | | | |
| 메소드명 | rb\_depart\_  CheckedChanged | 리턴  타입 |  | 파라미터 | Object sender,  EventArgs e | |
| 설명 | '부서' radiobutton 선택 시 combobox에 DB에서 추출한 부서 종류를 항목으로 표시 및 부서별 사원 정보 출력을 위한 listview의 column 추가 | | | |
| 메소드명 | rb\_position\_  CheckedChanged | 리턴  타입 |  | 파라미터 | Object sender,  EventArgs e | |
| 설명 | '직책' radiobutton 선택 시 combobox에 DB에서 추출한 직책 종류를 항목으로 표시 및 직책별 사원 정보 출력을 위한 listview column 추가 | | | |
| 메소드명 | btn\_report  \_Click | 리턴  타입 |  | 파라미터 | Object sender,  EventArgs e | |
| 설명 | radiobutton에서 선택된 값에 따른 데이터를 DB에서 추출 후  listview에 출력 | | | |

1. 데이터모형설계(ERD) : 테이블 간의 모형설계

|  |
| --- |
|  |

1. 화면설계정의
   1. 메인화면

|  |  |  |  |
| --- | --- | --- | --- |
| 메뉴명(경로) | 로그인 후 메인화면 | 화면명 | myERP |
| 화면설명 | 로그인 화면에서 id, password 입력 후 인증 완료 시 화면 | | |
| 관련기능 | 기초정보관리(부서), 기초정보관리(직책), 기초정보관리(종교), 사원관리, 가족사항, 퇴직자, 보고서, 도움말 접근 | | |
|  | | | |

* 1. 서브화면

|  |  |  |  |
| --- | --- | --- | --- |
| 메뉴명(경로) | 실행 후 첫 화면 | 화면명 | login |
| 화면설명 | 프로그램 실행 후 첫 화면으로,  id와 password를 입력 후 버튼을 눌러 인증한다. | | |
| 관련요구사항 | 초기 id: mgr, pass: 1234 | | |
| 관련기능 | ID, PWD 입력 후 확인 버튼을 눌렀을 때 일치하는 경우, myERP 화면으로 전환하고 불일치 시 화면 하단에 불일치 label 출력 | | |
|  | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| 메뉴명(경로) | myERP 화면에서 기초정보관리 Menu | 화면명 | 부서 관리 |
| 화면설명 | 부서명 및 부서 코드를 조회, 등록, 수정, 삭제 할 수 있는 화면 | | |
| 관련요구사항 | 조회 시 부서명으로 조회,  사용중인 부서명은 수정은 가능하지만 삭제는 불가 | | |
| 관련기능 | Tab으로 메뉴 전환이 가능하며 부서 조회 시 부서명으로 조회가 가능하고, 부서명, 부서코드 수정 시 DB의 데이터에 적용 가능하나 DB에서 사용중인 부서명, 코드는 삭제 불가능 | | |
|  | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| 메뉴명(경로) | myERP 화면에서 기초정보관리 Menu | 화면명 | 직책 관리 |
| 화면설명 | 직책명 및 직책 코드를 조회, 등록, 수정, 삭제 할 수 있는 화면 | | |
| 관련요구사항 | 조회 시 직책명으로 조회,  사용중인 직책명은 수정은 가능하지만 삭제는 불가 | | |
| 관련기능 | Tab으로 메뉴 전환이 가능하며 직책 조회 시 직책명으로 조회가 가능하고, 직책명, 직책코드 수정 시 DB의 데이터에 적용 가능하나 DB에서 사용중인 직책명, 코드는 삭제 불가능 | | |
|  | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| 메뉴명(경로) | myERP 화면에서 기초정보관리 Menu | 화면명 | 종교 관리 |
| 화면설명 | 종교명 및 종교 코드를 조회, 등록, 수정, 삭제 할 수 있는 화면 | | |
| 관련요구사항 | 조회 시 종교명으로 조회,  사용중인 종교명은 수정은 가능하지만 삭제는 불가 | | |
| 관련기능 | Tab으로 메뉴 전환이 가능하며 종교 조회 시 종교명으로 조회가 가능하고, 종교명, 종교코드 수정 시 DB의 데이터에 적용 가능하나 DB에서 사용중인 종교명, 코드는 삭제 불가능 | | |
|  | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| 메뉴명(경로) | myERP 화면에서 사원관리 Menu | 화면명 | 사원 관리 |
| 화면설명 | 재직 중인 사원번호, 성명, 부서에 따라 사원 조회가 가능하며,  사원 등록, 수정, 삭제 할 수 있는 화면 | | |
| 관련요구사항 | 사원 등록 시 사원번호가 일치하면 등록 불가하며, 사원 정보가 삭제되면 사원의 가족 정보도 삭제 | | |
| 관련기능 | 재직 중인 사원의 사원번호, 성명, 부서 기준으로 조회 가능하며, 사원에 대한 정보를 등록, 수정, 삭제하는 화면으로, 사원 정보가 삭제되면 해당 사원에 대한 가족 정보도 모두 삭제된다. | | |
|  | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| 메뉴명(경로) | myERP 화면에서  가족 사항 Menu | 화면명 | 가족 사항 관리 |
| 화면설명 | 재직 중인 사원의 가족 사항을 관리하는 화면으로,  사원 번호로 조회하며 등록, 수정, 삭제 가능 | | |
| 관련요구사항 | 재직 중인 사원의 가족 사항만 관리 가능하고, 해당 화면에서는 사원에 대한 정보는 수정 불가 | | |
| 관련 기능 | 재직 중인 사원의 사원 번호가 combobox에 항목으로 나타나며 사번으로 조회 시 listbox에 해당 사원의 가족 사항이 출력된다.  listbox의 첫번째 column을 클릭하면 세부 사항을 확인, 수정, 삭제할 수 있다. | | |
|  | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| 메뉴명(경로) | myERP 화면에서  퇴직자 Menu | 화면명 | 퇴직자 관리 |
| 화면설명 | 재직 중인 사원을 퇴직 처리하거나, 퇴직 중인 사원을 재직 상태로 전환 및 퇴직한 사원에 대한 퇴직사유, 퇴직 날짜 수정 가능 | | |
| 관련요구사항 | 재직 중인 사원을 퇴직 상태로 저장한 경우 필수적으로 퇴직 사원에 대한 세부 사항을 입력하여야 한다. | | |
| 관련 기능 | 재직 중인 사원을 퇴직 처리하거나, 퇴직 사원을 재직 처리할 수 있고 퇴직 중인 사원을 재직 처리할 경우 퇴직 관련 정보는 삭제되며, 퇴직 사원의 사유, 날짜 수정 가능하고 전체 조회 가능 | | |
|  | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| 메뉴명(경로) | myERP 화면에서  보고서 Menu | 화면명 | 보고서 |
| 화면설명 | myERP에서 등록했던 정보들을 바탕으로 사원 조회하는 화면 | | |
| 관련요구사항 | 전체, 부서별, 직책별 조회는 재직 중인 사원에 대한 조회이며,  퇴직자 버튼은 퇴직 사원의 퇴직 사유별 조회 | | |
| 관련 기능 | radio button에서 조회 기준을 선택하고 combobox에서 세부 기준을 선택하면 그에 따른 조회 결과를 보여주며,  추후 해당 결과를 문서로 저장하는 기능을 추가할 예정 | | |
|  | | | |

1. 테이블목록

|  |  |  |
| --- | --- | --- |
| 순번 | 테이블 ID | 테이블 설명 |
| 1 | t\_login | 로그인에 대한 정보를 담고 있는 테이블로, 로그인 시 인증에 사용 |
| 2 | position | 직책에 대한 정보를 담고 있는 테이블로, 직책별 조회 및 combobox 항목 등에 사용 |
| 3 | department | 부서에 대한 정보를 담고 있는 테이블로, 부서별 조회 및 combobox 항목 등에 사용 |
| 4 | religion | 종교에 대한 정보를 담고 있는 테이블로, 종교 combobox 항목 등에 사용 |
| 5 | retirement\_reason | 퇴직 사유를 저장하는 테이블로 퇴직 사유별 정렬, combobox 항목 등에 사용 |
| 6 | member | 사원에 대한 정보를 담고 있는 테이블로, 해당 테이블의 사원 번호가 조회 기준이 됨 |
| 7 | family | member 테이블의 사원에 대한 가족 사항을 저장하며, 사원 정보 삭제 시 가족 정보도 함께 삭제 |
| 8 | retired | member 테이블의 work 값이 0인 경우 퇴사 처리 되며, 세부 사항이 저장되는 테이블 |

1. 테이블명세서

|  |  |  |
| --- | --- | --- |
| 테이블 ID | 테이블 설명 | 길이 |
| t\_login | 로그인에 대한 정보를 저장 |  |
| DB 테이블 생성 명령문(DML) | | |
| create table t\_login(id varchar(30), pass varchar(30), primary key(id)); | | |

|  |  |  |
| --- | --- | --- |
| 테이블 ID | 테이블 설명 | 길이 |
| position | 직책명, 직책 코드를 저장 |  |
| DB 테이블 생성 명령문(DML) | | |
| create table position(val varchar(15), code varchar(10), primary key(code)); | | |

|  |  |  |
| --- | --- | --- |
| 테이블 ID | 테이블 설명 | 길이 |
| department | 부서명, 부서 코드를 저장 |  |
| DB 테이블 생성 명령문(DML) | | |
| create table department  (val varchar(15), code varchar(10), primary key(code)); | | |

|  |  |  |
| --- | --- | --- |
| 테이블 ID | 테이블 설명 | 길이 |
| religion | 종교명, 종교 코드를 저장 |  |
| DB 테이블 생성 명령문(DML) | | |
| create table religion(val varchar(15), code varchar(10), primary key(code)); | | |

|  |  |  |
| --- | --- | --- |
| 테이블 ID | 테이블 설명 | 길이 |
| retirement\_reason | 퇴직사유를 저장 |  |
| DB 테이블 생성 명령문(DML) | | |
| create table retirement\_reason(val varchar(15)); | | |

|  |  |  |
| --- | --- | --- |
| 테이블 ID | 테이블 설명 | 길이 |
| retired | 퇴직 사원에 대한 정보를 저장 |  |
| DB 테이블 생성 명령문(DML) | | |
| create table retired(id varchar(30), r\_name varchar(10), c\_number varchar(30), depart varchar(10), position varchar(10), retire\_date varchar(30), r\_reason varchar(15), primary key(id)); | | |

|  |  |  |
| --- | --- | --- |
| 테이블 ID | 테이블 설명 | 길이 |
| member | 사원에 대한 정보를 저장 |  |
| DB 테이블 생성 명령문(DML) | | |
| create table member(name\_k varchar(10), name\_e varchar(20),  name\_c varchar(10), m\_address varchar(50), marriage int, e\_mail varchar(30), c\_number varchar(30), sex int, height int, weight\_ int, blood varchar(10), sight\_l varchar(10), sight\_r varchar(10), religioin varchar(15), depart varchar(10), position varchar(10), birth\_date varchar(30), join\_date varchar(30), id varchar(30), work int primary key(id)); | | |

|  |  |  |
| --- | --- | --- |
| 테이블 ID | 테이블 설명 | 길이 |
| family | 사원 별 가족에 대한 정보를 저장 |  |
| DB 테이블 생성 명령문(DML) | | |
| create table family(f\_id varchar(30), relation varchar(10), place varchar(15), f\_name varchar(10), job varchar(10), birth\_date varchar(30), c\_number varchar(30)); | | |

1. 프로젝트 개발 일정

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 월 | | | | | | | | | | | |
| 1 | 2 | | | | | | 3 | | | | 4 |
| 일 | | | | | | | | | | | |
| 15 | 18 | 19 | 25 | 26 | 27 | 28 | 14 | 15 | 28 | 29 | 2 |
| 프로젝트 아이디어 구상 및 계획 |  |  |  |  |  |  |  |  |  |  |  |  |
| 테이블 및 폼 기능 세부 목표 설정 |  |  |  |  |  |  |  |  |  |  |  |  |
| MSSQL DB 테이블 생성 |  |  |  |  |  |  |  |  |  |  |  |  |
| 모든 폼생성 및 DB 연동 작업 |  |  |  |  |  |  |  |  |  |  |  |  |
| 예외 처리 및 전체적인 수정 |  |  |  |  |  |  |  |  |  |  |  |  |
| 프로젝트 작업 기획서 작성 |  |  |  |  |  |  |  |  |  |  |  |  |

|  |  |  |
| --- | --- | --- |
| **3.** |  | **프로젝트 소스 코드** |

***[login Form]***

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Data.SqlClient;

namespace myERP

{

public partial class login : Form

{

private string Constr = "server=localhost;uid=myERP;pwd=1234;database=myERP";

public login()

{

InitializeComponent();

m\_id.Text = "mgr";

m\_pass.Text = "1234";

}

private void txtClear() // login\_state 데이터 초기화

{

this.m\_id.Text = "";

this.m\_pass.Text = "";

}

private void btn\_Login\_Click(object sender, EventArgs e)

{

var Conn = new SqlConnection(Constr);

Conn.Open();

var Comm = new SqlCommand("Select pass from t\_login " + "where id = '" + this.m\_id.Text + "'", Conn);

var myRead = Comm.ExecuteReader();

if(myRead.Read())

{

string strpwd = myRead[0].ToString();

if(strpwd==this.m\_pass.Text)

{

myRead.Close();

Conn.Close();

ERP\_main main = new ERP\_main();

//main.UserId = this.m\_id.Text;

main.Show();

this.Hide();

}

else

{

this.login\_state.Text = "로그인 실패";

txtClear();

}

}

else

{

this.login\_state.Text = "로그인 실패";

txtClear();

}

myRead.Close();

Conn.Close();

}

}

}

***[ERP\_main Form]***

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace myERP

{

public partial class ERP\_main : Form

{

public ERP\_main()

{

InitializeComponent();

}

private void aboutMyERPToolStripMenuItem\_Click(object sender, EventArgs e)

{

Department depart = new Department();

depart.Show();

}

private void 직책ToolStripMenuItem\_Click(object sender, EventArgs e)

{

Position position = new Position();

position.Show();

}

private void 종교ToolStripMenuItem\_Click(object sender, EventArgs e)

{

Religion religion = new Religion();

religion.Show();

}

private void 사원등록RToolStripMenuItem\_Click(object sender, EventArgs e)

{

Member mem = new Member();

mem.Show();

}

private void 가족사항FToolStripMenuItem\_Click(object sender, EventArgs e)

{

Family family = new Family();

family.Show();

}

private void 퇴직자RToolStripMenuItem\_Click(object sender, EventArgs e)

{

Retired retired = new Retired();

retired.Show();

}

private void 끝내기tCtrlXToolStripMenuItem\_Click(object sender, EventArgs e)

{

Application.Exit();

}

private void 보고서ToolStripMenuItem\_Click(object sender, EventArgs e)

{

Report r = new Report();

r.Show();

}

}

}

***[Department Form]***

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Data.SqlClient;

namespace myERP

{

public partial class Department : Form

{

private string Constr = "server=localhost;uid=myERP;pwd=1234;database=myERP";

public Department()

{

InitializeComponent();

lvDepart\_s.View = View.Details;

lvDepart\_s.Items.Clear();

string Searchval, Searchcode;

try

{

SqlConnection dbcon = new SqlConnection();

dbcon.ConnectionString = Constr;

String Sql = "Select \* from department order by val asc";

SqlCommand dbcmd = new SqlCommand(Sql, dbcon);

dbcon.Open();

SqlDataReader reader = dbcmd.ExecuteReader();

if (reader.HasRows)

{

while (reader.Read())

{

Searchval = reader["val"].ToString();

Searchcode = reader["code"].ToString();

ListViewItem item = new ListViewItem();

item.Text = Searchval;

item.SubItems.Add(Searchcode);

this.lvDepart\_s.Items.Add(item);

}

}

reader.Close();

dbcon.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "오류!");

}

}

private void btn\_d\_search\_Click(object sender, EventArgs e)

{

lvDepart\_s.View = View.Details;

lvDepart\_s.Items.Clear();

string SearchValue = s\_depart.ToString();

string Searchval, Searchcode;

try

{

SqlConnection dbcon = new SqlConnection();

dbcon.ConnectionString = Constr;

String Sql = "Select \* from department where val='" + s\_depart.Text + "' order by val asc";

SqlCommand dbcmd = new SqlCommand(Sql, dbcon);

dbcon.Open();

SqlDataReader reader = dbcmd.ExecuteReader();

if (reader.HasRows)

{

while (reader.Read())

{

Searchval = reader["val"].ToString();

Searchcode = reader["code"].ToString();

ListViewItem item = new ListViewItem();

item.Text = Searchval;

item.SubItems.Add(Searchcode);

this.lvDepart\_s.Items.Add(item);

}

}

reader.Close();

dbcon.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "오류!");

}

// textbox 초기화

s\_depart.Text = "";

}

private void btn\_d\_add\_Click(object sender, EventArgs e)

{

var Conn = new SqlConnection(Constr);

Conn.Open();

var strSQL = "insert into department " + "values('" + a\_depart.Text + "', '" + a\_d\_code.Text + "')";

var myCom = new SqlCommand(strSQL, Conn);

myCom.ExecuteNonQuery();

Conn.Close();

MessageBox.Show("데이터가 저장되었습니다.", "알림");

//데이터 새로고침

lvDepart\_a.View = View.Details;

lvDepart\_a.Items.Clear();

string Aval, Acode;

try

{

SqlConnection dbcon = new SqlConnection();

dbcon.ConnectionString = Constr;

String Sql = "Select \* from department order by val asc";

SqlCommand dbcmd = new SqlCommand(Sql, dbcon);

dbcon.Open();

SqlDataReader reader = dbcmd.ExecuteReader();

if (reader.HasRows)

{

while (reader.Read())

{

Aval = reader["val"].ToString();

Acode = reader["code"].ToString();

ListViewItem item = new ListViewItem();

item.Text = Aval;

item.SubItems.Add(Acode);

this.lvDepart\_a.Items.Add(item);

}

}

reader.Close();

dbcon.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "오류!");

}

// textbox 초기화

a\_depart.Text = "";

a\_d\_code.Text = "";

}

private void lvDepart\_u\_SelectedIndexChanged(object sender, EventArgs e)

{

if (lvDepart\_u.SelectedItems.Count != 0)

{

int SelectRow = lvDepart\_u.SelectedItems[0].Index;

u\_depart.Text = lvDepart\_u.Items[SelectRow].SubItems[0].Text;

ud\_hidden2.Text = lvDepart\_u.Items[SelectRow].SubItems[0].Text;

u\_d\_code.Text = lvDepart\_u.Items[SelectRow].SubItems[1].Text;

ud\_hidden.Text = lvDepart\_u.Items[SelectRow].SubItems[1].Text;

}

}

private void btn\_d\_update\_Click(object sender, EventArgs e)

{

var Conn1 = new SqlConnection(Constr);

Conn1.Open();

var Conn2 = new SqlConnection(Constr);

Conn2.Open();

var Conn3 = new SqlConnection(Constr);

Conn3.Open();

var strSQL1 = "update department set val='" + u\_depart.Text + "', code= '" + u\_d\_code.Text + "' where code='" + ud\_hidden.Text + "'";

var strSQL2 = "update member set depart='" + u\_depart.Text + "' where depart='" + ud\_hidden2.Text + "'";

var strSQL3 = "update retired set depart='" + u\_depart.Text + "' where depart='" + ud\_hidden2.Text + "'";

var myCom1 = new SqlCommand(strSQL1, Conn1);

myCom1.ExecuteNonQuery();

var myCom2 = new SqlCommand(strSQL2, Conn2);

myCom2.ExecuteNonQuery();

var myCom3 = new SqlCommand(strSQL3, Conn3);

myCom3.ExecuteNonQuery();

Conn1.Close();

Conn2.Close();

Conn3.Close();

MessageBox.Show("데이터가 저장되었습니다.", "알림");

//데이터 새로고침

lvDepart\_u.View = View.Details;

lvDepart\_u.Items.Clear();

string Aval, Acode;

try

{

SqlConnection dbcon = new SqlConnection();

dbcon.ConnectionString = Constr;

String Sql = "Select \* from department order by val asc";

SqlCommand dbcmd = new SqlCommand(Sql, dbcon);

dbcon.Open();

SqlDataReader reader = dbcmd.ExecuteReader();

if (reader.HasRows)

{

while (reader.Read())

{

Aval = reader["val"].ToString();

Acode = reader["code"].ToString();

ListViewItem item = new ListViewItem();

item.Text = Aval;

item.SubItems.Add(Acode);

this.lvDepart\_u.Items.Add(item);

}

}

reader.Close();

dbcon.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "오류!");

}

// textbox 초기화

u\_depart.Text = "";

u\_d\_code.Text = "";

}

private void d\_tabMenu\_SelectedIndexChanged(object sender, EventArgs e)

{

lvDepart\_s.View = View.Details;

lvDepart\_a.View = View.Details;

lvDepart\_u.View = View.Details;

lvDepart\_d.View = View.Details;

lvDepart\_s.Items.Clear();

lvDepart\_a.Items.Clear();

lvDepart\_u.Items.Clear();

lvDepart\_d.Items.Clear();

string Sval, Scode, Aval, Acode, Uval, Ucode, Dval, Dcode;

try

{

SqlConnection dbcon = new SqlConnection();

dbcon.ConnectionString = Constr;

String Sql = "Select \* from department order by val asc";

SqlCommand dbcmd = new SqlCommand(Sql, dbcon);

dbcon.Open();

SqlDataReader reader = dbcmd.ExecuteReader();

if (reader.HasRows)

{

while (reader.Read())

{

Sval = reader["val"].ToString();

Scode = reader["code"].ToString();

Aval = reader["val"].ToString();

Acode = reader["code"].ToString();

Uval = reader["val"].ToString();

Ucode = reader["code"].ToString();

Dval = reader["val"].ToString();

Dcode = reader["code"].ToString();

ListViewItem item = new ListViewItem();

ListViewItem item1 = new ListViewItem();

ListViewItem item2 = new ListViewItem();

ListViewItem item3 = new ListViewItem();

item.Text = Sval;

item.SubItems.Add(Scode);

item1.Text = Aval;

item1.SubItems.Add(Acode);

item2.Text = Uval;

item2.SubItems.Add(Ucode);

item3.Text = Dval;

item3.SubItems.Add(Dcode);

this.lvDepart\_s.Items.Add(item);

this.lvDepart\_a.Items.Add(item1);

this.lvDepart\_u.Items.Add(item2);

this.lvDepart\_d.Items.Add(item3);

}

}

reader.Close();

dbcon.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "오류!");

}

}

private void lvDepart\_d\_SelectedIndexChanged(object sender, EventArgs e)

{

if (lvDepart\_d.SelectedItems.Count != 0)

{

int SelectRow = lvDepart\_d.SelectedItems[0].Index;

dd\_hidden2.Text = lvDepart\_u.Items[SelectRow].SubItems[0].Text;

dd\_hidden.Text = lvDepart\_u.Items[SelectRow].SubItems[1].Text;

}

}

private void btn\_d\_delete\_Click(object sender, EventArgs e)

{

int state = 0; // 삭제 가능 상태

var Conn1 = new SqlConnection(Constr);

Conn1.Open();

var strSQL1 = "select depart from member where depart='" + dd\_hidden2.Text + "'";

var myCom1 = new SqlCommand(strSQL1, Conn1);

myCom1.ExecuteNonQuery();

SqlDataReader reader1 = myCom1.ExecuteReader();

while (reader1.Read())

{

if (reader1["depart"].ToString() == dd\_hidden2.Text)

{

state = 1; // 사용중인 코드로 인한 삭제 불가 상태

}

}

reader1.Close();

Conn1.Close();

var Conn2 = new SqlConnection(Constr);

Conn2.Open();

var strSQL2 = "select depart from retired where depart='" + dd\_hidden2.Text + "'";

var myCom2 = new SqlCommand(strSQL2, Conn2);

myCom2.ExecuteNonQuery();

SqlDataReader reader2 = myCom2.ExecuteReader();

while (reader2.Read())

{

if (reader2["depart"].ToString() == dd\_hidden2.Text)

{

state = 1; // 사용중인 코드로 인한 삭제 불가 상태

}

}

reader2.Close();

Conn2.Close();

if(state==0)

{

var Conn = new SqlConnection(Constr);

Conn.Open();

var strSQL = "delete from department where code='" + dd\_hidden.Text + "'";

var myCom = new SqlCommand(strSQL, Conn);

myCom.ExecuteNonQuery();

Conn.Close();

MessageBox.Show("데이터가 삭제되었습니다.", "알림");

}

else if(state==1)

{

MessageBox.Show("사용 중인 코드입니다.", "알림");

}

//데이터 새로고침

lvDepart\_d.View = View.Details;

lvDepart\_d.Items.Clear();

string Aval, Acode;

try

{

SqlConnection dbcon = new SqlConnection();

dbcon.ConnectionString = Constr;

String Sql = "Select \* from department order by val asc";

SqlCommand dbcmd = new SqlCommand(Sql, dbcon);

dbcon.Open();

SqlDataReader reader = dbcmd.ExecuteReader();

if (reader.HasRows)

{

while (reader.Read())

{

Aval = reader["val"].ToString();

Acode = reader["code"].ToString();

ListViewItem item = new ListViewItem();

item.Text = Aval;

item.SubItems.Add(Acode);

this.lvDepart\_d.Items.Add(item);

}

}

reader.Close();

dbcon.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "오류!");

}

}

}

}

***[Position Form]***

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Data.SqlClient;

namespace myERP

{

public partial class Position : Form

{

private string Constr = "server=localhost;uid=myERP;pwd=1234;database=myERP";

public Position()

{

InitializeComponent();

lvPosition\_s.View = View.Details;

lvPosition\_s.Items.Clear();

string Sval, Scode;

try

{

SqlConnection dbcon = new SqlConnection();

dbcon.ConnectionString = "server=localhost;uid=myERP;pwd=1234;database=myERP";

String Sql = "Select \* from position";

SqlCommand dbcmd = new SqlCommand(Sql, dbcon);

dbcon.Open();

SqlDataReader reader = dbcmd.ExecuteReader();

if (reader.HasRows)

{

while (reader.Read())

{

Sval = reader["val"].ToString();

Scode = reader["code"].ToString();

ListViewItem item = new ListViewItem();

item.Text = Sval;

item.SubItems.Add(Scode);

this.lvPosition\_s.Items.Add(item);

}

}

reader.Close();

dbcon.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "오류!");

}

}

private void btn\_p\_search\_Click(object sender, EventArgs e)

{

lvPosition\_s.View = View.Details;

lvPosition\_s.Items.Clear();

string Searchval, Searchcode;

try

{

SqlConnection dbcon = new SqlConnection();

dbcon.ConnectionString = "server=localhost;uid=myERP;pwd=1234;database=myERP";

String Sql = "Select \* from position where val='" + s\_position.Text + "'";

SqlCommand dbcmd = new SqlCommand(Sql, dbcon);

dbcon.Open();

SqlDataReader reader = dbcmd.ExecuteReader();

if (reader.HasRows)

{

while (reader.Read())

{

Searchval = reader["val"].ToString();

Searchcode = reader["code"].ToString();

ListViewItem item = new ListViewItem();

item.Text = Searchval;

item.SubItems.Add(Searchcode);

this.lvPosition\_s.Items.Add(item);

}

}

reader.Close();

dbcon.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "오류!");

}

// textbox 초기화

s\_position.Text = "";

}

private void p\_TabMenu\_SelectedIndexChanged(object sender, EventArgs e)

{

lvPosition\_s.View = View.Details;

lvPosition\_a.View = View.Details;

lvPosition\_u.View = View.Details;

lvPosition\_d.View = View.Details;

lvPosition\_s.Items.Clear();

lvPosition\_a.Items.Clear();

lvPosition\_u.Items.Clear();

lvPosition\_d.Items.Clear();

string Sval, Scode, Aval, Acode, Uval, Ucode, Dval, Dcode;

try

{

SqlConnection dbcon = new SqlConnection();

dbcon.ConnectionString = "server=localhost;uid=myERP;pwd=1234;database=myERP";

String Sql = "Select \* from position";

SqlCommand dbcmd = new SqlCommand(Sql, dbcon);

dbcon.Open();

SqlDataReader reader = dbcmd.ExecuteReader();

if (reader.HasRows)

{

while (reader.Read())

{

Sval = reader["val"].ToString();

Scode = reader["code"].ToString();

Aval = reader["val"].ToString();

Acode = reader["code"].ToString();

Uval = reader["val"].ToString();

Ucode = reader["code"].ToString();

Dval = reader["val"].ToString();

Dcode = reader["code"].ToString();

ListViewItem item = new ListViewItem();

ListViewItem item1 = new ListViewItem();

ListViewItem item2 = new ListViewItem();

ListViewItem item3 = new ListViewItem();

item.Text = Sval;

item.SubItems.Add(Scode);

item1.Text = Aval;

item1.SubItems.Add(Acode);

item2.Text = Uval;

item2.SubItems.Add(Ucode);

item3.Text = Dval;

item3.SubItems.Add(Dcode);

this.lvPosition\_s.Items.Add(item);

this.lvPosition\_a.Items.Add(item1);

this.lvPosition\_u.Items.Add(item2);

this.lvPosition\_d.Items.Add(item3);

}

}

reader.Close();

dbcon.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "오류!");

}

}

private void btn\_p\_add\_Click(object sender, EventArgs e)

{

var Conn = new SqlConnection(Constr);

Conn.Open();

var strSQL = "insert into position " + "values('" + a\_position.Text + "', '" + a\_p\_code.Text + "')";

var myCom = new SqlCommand(strSQL, Conn);

myCom.ExecuteNonQuery();

Conn.Close();

MessageBox.Show("데이터가 저장되었습니다.", "알림");

//데이터 새로고침

lvPosition\_a.View = View.Details;

lvPosition\_a.Items.Clear();

string Searchval, Searchcode;

try

{

SqlConnection dbcon = new SqlConnection();

dbcon.ConnectionString = "server=localhost;uid=myERP;pwd=1234;database=myERP";

String Sql = "Select \* from position";

SqlCommand dbcmd = new SqlCommand(Sql, dbcon);

dbcon.Open();

SqlDataReader reader = dbcmd.ExecuteReader();

if (reader.HasRows)

{

while (reader.Read())

{

Searchval = reader["val"].ToString();

Searchcode = reader["code"].ToString();

ListViewItem item = new ListViewItem();

item.Text = Searchval;

item.SubItems.Add(Searchcode);

this.lvPosition\_a.Items.Add(item);

}

}

reader.Close();

dbcon.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "오류!");

}

// textbox 초기화

a\_position.Text = "";

a\_p\_code.Text = "";

}

private void lvPosition\_u\_SelectedIndexChanged(object sender, EventArgs e)

{

if (lvPosition\_u.SelectedItems.Count != 0)

{

int SelectRow = lvPosition\_u.SelectedItems[0].Index;

u\_position.Text = lvPosition\_u.Items[SelectRow].SubItems[0].Text;

up\_hidden2.Text = lvPosition\_u.Items[SelectRow].SubItems[0].Text;

u\_p\_code.Text = lvPosition\_u.Items[SelectRow].SubItems[1].Text;

up\_hidden.Text = lvPosition\_u.Items[SelectRow].SubItems[1].Text;

}

}

private void btn\_p\_update\_Click(object sender, EventArgs e)

{

var Conn1 = new SqlConnection(Constr);

Conn1.Open();

var Conn2 = new SqlConnection(Constr);

Conn2.Open();

var Conn3 = new SqlConnection(Constr);

Conn3.Open();

var strSQL1 = "update position set val='" + u\_position.Text + "', code= '" + u\_p\_code.Text + "' where code='" + up\_hidden.Text + "'";

var strSQL2 = "update member set position='" + u\_position.Text + "' where position='" + up\_hidden2.Text + "'";

var strSQL3 = "update retired set position='" + u\_position.Text + "' where position='" + up\_hidden2.Text + "'";

var myCom1 = new SqlCommand(strSQL1, Conn1);

myCom1.ExecuteNonQuery();

var myCom2 = new SqlCommand(strSQL2, Conn2);

myCom2.ExecuteNonQuery();

var myCom3 = new SqlCommand(strSQL3, Conn3);

myCom3.ExecuteNonQuery();

Conn1.Close();

Conn2.Close();

Conn3.Close();

//데이터 새로고침

lvPosition\_u.View = View.Details;

lvPosition\_u.Items.Clear();

string Searchval, Searchcode;

try

{

SqlConnection dbcon = new SqlConnection();

dbcon.ConnectionString = "server=localhost;uid=myERP;pwd=1234;database=myERP";

String Sql = "Select \* from position";

SqlCommand dbcmd = new SqlCommand(Sql, dbcon);

dbcon.Open();

SqlDataReader reader = dbcmd.ExecuteReader();

if (reader.HasRows)

{

while (reader.Read())

{

Searchval = reader["val"].ToString();

Searchcode = reader["code"].ToString();

ListViewItem item = new ListViewItem();

item.Text = Searchval;

item.SubItems.Add(Searchcode);

this.lvPosition\_u.Items.Add(item);

}

}

reader.Close();

dbcon.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "오류!");

}

// textbox 초기화

u\_position.Text = "";

u\_p\_code.Text = "";

}

private void lvPosition\_d\_SelectedIndexChanged(object sender, EventArgs e)

{

if (lvPosition\_d.SelectedItems.Count != 0)

{

int SelectRow = lvPosition\_d.SelectedItems[0].Index;

dp\_hidden2.Text = lvPosition\_d.Items[SelectRow].SubItems[0].Text;

dp\_hidden.Text = lvPosition\_d.Items[SelectRow].SubItems[1].Text;

}

}

private void btn\_delete\_Click(object sender, EventArgs e)

{

int state = 0; // 삭제 가능 상태

var Conn1 = new SqlConnection(Constr);

Conn1.Open();

var strSQL1 = "select position from member where position='" + dp\_hidden2.Text + "'";

var myCom1 = new SqlCommand(strSQL1, Conn1);

myCom1.ExecuteNonQuery();

SqlDataReader reader1 = myCom1.ExecuteReader();

while (reader1.Read())

{

if (reader1["position"].ToString() == dp\_hidden2.Text)

{

state = 1; // 사용중인 코드로 인한 삭제 불가 상태

}

}

reader1.Close();

Conn1.Close();

var Conn2 = new SqlConnection(Constr);

Conn2.Open();

var strSQL2 = "select position from retired where position='" + dp\_hidden2.Text + "'";

var myCom2 = new SqlCommand(strSQL2, Conn2);

myCom2.ExecuteNonQuery();

SqlDataReader reader2 = myCom2.ExecuteReader();

while (reader2.Read())

{

if (reader2["position"].ToString() == dp\_hidden2.Text)

{

state = 1; // 사용중인 코드로 인한 삭제 불가 상태

}

}

reader2.Close();

Conn2.Close();

if (state == 0)

{

var Conn = new SqlConnection(Constr);

Conn.Open();

var strSQL = "delete from position where code='" + dp\_hidden.Text + "'";

var myCom = new SqlCommand(strSQL, Conn);

myCom.ExecuteNonQuery();

Conn.Close();

MessageBox.Show("데이터가 삭제되었습니다.", "알림");

}

else if(state==1)

{

MessageBox.Show("사용 중인 코드입니다.", "알림");

}

//데이터 새로고침

lvPosition\_d.View = View.Details;

lvPosition\_d.Items.Clear();

string Searchval, Searchcode;

try

{

SqlConnection dbcon = new SqlConnection();

dbcon.ConnectionString = "server=localhost;uid=myERP;pwd=1234;database=myERP";

String Sql = "Select \* from position";

SqlCommand dbcmd = new SqlCommand(Sql, dbcon);

dbcon.Open();

SqlDataReader reader = dbcmd.ExecuteReader();

if (reader.HasRows)

{

while (reader.Read())

{

Searchval = reader["val"].ToString();

Searchcode = reader["code"].ToString();

ListViewItem item = new ListViewItem();

item.Text = Searchval;

item.SubItems.Add(Searchcode);

this.lvPosition\_d.Items.Add(item);

}

}

reader.Close();

dbcon.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "오류!");

}

}

}

}

***[Religion Form]***

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Data.SqlClient;

namespace myERP

{

public partial class Religion : Form

{

private string Constr = "server=localhost;uid=myERP;pwd=1234;database=myERP";

public Religion()

{

InitializeComponent();

lvReligion\_s.View = View.Details;

lvReligion\_s.Items.Clear();

string Sval, Scode;

try

{

SqlConnection dbcon = new SqlConnection();

dbcon.ConnectionString = "server=localhost;uid=myERP;pwd=1234;database=myERP";

String Sql = "Select \* from religion order by val asc";

SqlCommand dbcmd = new SqlCommand(Sql, dbcon);

dbcon.Open();

SqlDataReader reader = dbcmd.ExecuteReader();

if (reader.HasRows)

{

while (reader.Read())

{

Sval = reader["val"].ToString();

Scode = reader["code"].ToString();

ListViewItem item = new ListViewItem();

item.Text = Sval;

item.SubItems.Add(Scode);

this.lvReligion\_s.Items.Add(item);

}

}

reader.Close();

dbcon.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "오류!");

}

}

private void btn\_r\_search\_Click(object sender, EventArgs e)

{

lvReligion\_s.View = View.Details;

lvReligion\_s.Items.Clear();

string SearchValue = s\_religion.ToString();

string Searchval, Searchcode;

try

{

SqlConnection dbcon = new SqlConnection();

dbcon.ConnectionString = "server=localhost;uid=myERP;pwd=1234;database=myERP";

String Sql = "Select \* from religion where val='" + s\_religion.Text + "' order by val";

SqlCommand dbcmd = new SqlCommand(Sql, dbcon);

dbcon.Open();

SqlDataReader reader = dbcmd.ExecuteReader();

if (reader.HasRows)

{

while (reader.Read())

{

Searchval = reader["val"].ToString();

Searchcode = reader["code"].ToString();

ListViewItem item = new ListViewItem();

item.Text = Searchval;

item.SubItems.Add(Searchcode);

this.lvReligion\_s.Items.Add(item);

}

}

reader.Close();

dbcon.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "오류!");

}

// textbox 초기화

s\_religion.Text = "";

}

private void r\_tabMenu\_SelectedIndexChanged(object sender, EventArgs e)

{

lvReligion\_s.View = View.Details;

lvReligion\_a.View = View.Details;

lvReligion\_u.View = View.Details;

lvReligion\_d.View = View.Details;

lvReligion\_s.Items.Clear();

lvReligion\_a.Items.Clear();

lvReligion\_u.Items.Clear();

lvReligion\_d.Items.Clear();

string Sval, Scode, Aval, Acode, Uval, Ucode, Dval, Dcode;

try

{

SqlConnection dbcon = new SqlConnection();

dbcon.ConnectionString = "server=localhost;uid=myERP;pwd=1234;database=myERP";

String Sql = "Select \* from religion order by val asc";

SqlCommand dbcmd = new SqlCommand(Sql, dbcon);

dbcon.Open();

SqlDataReader reader = dbcmd.ExecuteReader();

if (reader.HasRows)

{

while (reader.Read())

{

Sval = reader["val"].ToString();

Scode = reader["code"].ToString();

Aval = reader["val"].ToString();

Acode = reader["code"].ToString();

Uval = reader["val"].ToString();

Ucode = reader["code"].ToString();

Dval = reader["val"].ToString();

Dcode = reader["code"].ToString();

ListViewItem item = new ListViewItem();

ListViewItem item1 = new ListViewItem();

ListViewItem item2 = new ListViewItem();

ListViewItem item3 = new ListViewItem();

item.Text = Sval;

item.SubItems.Add(Scode);

item1.Text = Aval;

item1.SubItems.Add(Acode);

item2.Text = Uval;

item2.SubItems.Add(Ucode);

item3.Text = Dval;

item3.SubItems.Add(Dcode);

this.lvReligion\_s.Items.Add(item);

this.lvReligion\_a.Items.Add(item1);

this.lvReligion\_u.Items.Add(item2);

this.lvReligion\_d.Items.Add(item3);

}

}

reader.Close();

dbcon.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "오류!");

}

}

private void btn\_r\_add\_Click(object sender, EventArgs e)

{

var Conn = new SqlConnection(Constr);

Conn.Open();

var strSQL = "insert into religion " + "values('" + a\_religion.Text + "', '" + a\_r\_code.Text + "')";

var myCom = new SqlCommand(strSQL, Conn);

myCom.ExecuteNonQuery();

Conn.Close();

MessageBox.Show("데이터가 저장되었습니다.", "알림");

//데이터 새로고침

lvReligion\_a.View = View.Details;

lvReligion\_a.Items.Clear();

string Searchval, Searchcode;

try

{

SqlConnection dbcon = new SqlConnection();

dbcon.ConnectionString = "server=localhost;uid=myERP;pwd=1234;database=myERP";

String Sql = "Select \* from religion order by val asc";

SqlCommand dbcmd = new SqlCommand(Sql, dbcon);

dbcon.Open();

SqlDataReader reader = dbcmd.ExecuteReader();

if (reader.HasRows)

{

while (reader.Read())

{

Searchval = reader["val"].ToString();

Searchcode = reader["code"].ToString();

ListViewItem item = new ListViewItem();

item.Text = Searchval;

item.SubItems.Add(Searchcode);

this.lvReligion\_a.Items.Add(item);

}

}

reader.Close();

dbcon.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "오류!");

}

// textbox 초기화

a\_religion.Text = "";

a\_r\_code.Text = "";

}

private void lvReligion\_u\_SelectedIndexChanged(object sender, EventArgs e)

{

if (lvReligion\_u.SelectedItems.Count != 0)

{

int SelectRow = lvReligion\_u.SelectedItems[0].Index;

u\_religion.Text = lvReligion\_u.Items[SelectRow].SubItems[0].Text;

ur\_hidden2.Text = lvReligion\_u.Items[SelectRow].SubItems[0].Text;

u\_r\_code.Text = lvReligion\_u.Items[SelectRow].SubItems[1].Text;

ur\_hidden.Text = lvReligion\_u.Items[SelectRow].SubItems[1].Text;

}

}

private void btn\_r\_update\_Click(object sender, EventArgs e)

{

var Conn1 = new SqlConnection(Constr);

Conn1.Open();

var Conn2 = new SqlConnection(Constr);

Conn2.Open();

var strSQL1 = "update religion set val='" + u\_religion.Text + "', code= '" + u\_r\_code.Text + "' where code='" + ur\_hidden.Text + "'";

var strSQL2 = "update member set religion='" + u\_religion.Text + "' where religion='" + ur\_hidden2.Text + "'";

var myCom1 = new SqlCommand(strSQL1, Conn1);

myCom1.ExecuteNonQuery();

var myCom2 = new SqlCommand(strSQL2, Conn2);

myCom2.ExecuteNonQuery();

Conn1.Close();

Conn2.Close();

MessageBox.Show("데이터가 저장되었습니다.", "알림");

//데이터 새로고침

lvReligion\_u.View = View.Details;

lvReligion\_u.Items.Clear();

string Searchval, Searchcode;

try

{

SqlConnection dbcon = new SqlConnection();

dbcon.ConnectionString = "server=localhost;uid=myERP;pwd=1234;database=myERP";

String Sql = "Select \* from religion order by val";

SqlCommand dbcmd = new SqlCommand(Sql, dbcon);

dbcon.Open();

SqlDataReader reader = dbcmd.ExecuteReader();

if (reader.HasRows)

{

while (reader.Read())

{

Searchval = reader["val"].ToString();

Searchcode = reader["code"].ToString();

ListViewItem item = new ListViewItem();

item.Text = Searchval;

item.SubItems.Add(Searchcode);

this.lvReligion\_u.Items.Add(item);

}

}

reader.Close();

dbcon.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "오류!");

}

// textbox 초기화

u\_religion.Text = "";

u\_r\_code.Text = "";

}

private void lvReligion\_d\_SelectedIndexChanged(object sender, EventArgs e)

{

if (lvReligion\_d.SelectedItems.Count != 0)

{

int SelectRow = lvReligion\_d.SelectedItems[0].Index;

dr\_hidden2.Text = lvReligion\_d.Items[SelectRow].SubItems[0].Text;

dr\_hidden.Text = lvReligion\_d.Items[SelectRow].SubItems[1].Text;

}

}

private void btn\_r\_delete\_Click(object sender, EventArgs e)

{

int state = 0; // 삭제 가능 상태

var Conn1 = new SqlConnection(Constr);

Conn1.Open();

var strSQL1 = "select religion from member where religion='" + dr\_hidden2.Text + "'";

var myCom1 = new SqlCommand(strSQL1, Conn1);

myCom1.ExecuteNonQuery();

SqlDataReader reader1 = myCom1.ExecuteReader();

while (reader1.Read())

{

if (reader1["religion"].ToString() == dr\_hidden2.Text)

{

state = 1; // 사용중인 코드로 인한 삭제 불가 상태

}

}

reader1.Close();

Conn1.Close();

if(state==0)

{

var Conn = new SqlConnection(Constr);

Conn.Open();

var strSQL = "delete from religion where code='" + dr\_hidden.Text + "'";

var myCom = new SqlCommand(strSQL, Conn);

myCom.ExecuteNonQuery();

Conn.Close();

MessageBox.Show("데이터가 삭제되었습니다.", "알림");

}

else if(state==1)

{

MessageBox.Show("사용 중인 코드입니다.", "알림");

}

//데이터 새로고침

lvReligion\_d.View = View.Details;

lvReligion\_d.Items.Clear();

string Searchval, Searchcode;

try

{

SqlConnection dbcon = new SqlConnection();

dbcon.ConnectionString = "server=localhost;uid=myERP;pwd=1234;database=myERP";

String Sql = "Select \* from religion order by val";

SqlCommand dbcmd = new SqlCommand(Sql, dbcon);

dbcon.Open();

SqlDataReader reader = dbcmd.ExecuteReader();

if (reader.HasRows)

{

while (reader.Read())

{

Searchval = reader["val"].ToString();

Searchcode = reader["code"].ToString();

ListViewItem item = new ListViewItem();

item.Text = Searchval;

item.SubItems.Add(Searchcode);

this.lvReligion\_d.Items.Add(item);

}

}

reader.Close();

dbcon.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "오류!");

}

}

}

}

***[Member Form]***

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Data.SqlClient;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace myERP

{

public partial class Member : Form

{

private string Constr = "server=localhost;uid=myERP;pwd=1234;database=myERP";

public Member()

{

InitializeComponent();

cb\_section.SelectedIndex = 0;

lvMember.View = View.Details;

lvMember.Items.Clear();

string id, name, sex, position, depart, email;

try

{

SqlConnection dbcon = new SqlConnection();

dbcon.ConnectionString = Constr;

String Sql = "Select \* from member where work=1";

SqlCommand dbcmd = new SqlCommand(Sql, dbcon);

dbcon.Open();

SqlDataReader reader = dbcmd.ExecuteReader();

if (reader.HasRows)

{

while (reader.Read())

{

id = reader["id"].ToString();

name = reader["name\_k"].ToString();

sex = reader["sex"].ToString();

position = reader["depart"].ToString();

depart = reader["position"].ToString();

email = reader["e\_mail"].ToString();

ListViewItem item = new ListViewItem();

item.Text = id;

item.SubItems.Add(name);

item.SubItems.Add(sex);

item.SubItems.Add(position);

item.SubItems.Add(depart);

item.SubItems.Add(email);

this.lvMember.Items.Add(item);

}

}

reader.Close();

dbcon.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "오류!");

}

}

private void textClear()

{

tb\_id.Text = "";

tb\_name\_k.Text = "";

tb\_name\_e.Text = "";

tb\_name\_c.Text = "";

tb\_address.Text = "";

tb\_height.Text = "";

tb\_weight.Text = "";

tb\_email.Text = "";

tb\_sight\_l.Text = "";

tb\_sight\_r.Text = "";

tb\_call.Text = "";

rb\_male.Checked = true;

rb\_married.Checked = true;

}

private void new\_show() // 변경된 정보 적용 후 새로 고침

{

lvMember.View = View.Details;

lvMember.Items.Clear();

string id, name, gender, position, depart, email;

try

{

SqlConnection dbcon = new SqlConnection();

dbcon.ConnectionString = Constr;

String Sql = "Select \* from member where work=1 order by id asc";

SqlCommand dbcmd = new SqlCommand(Sql, dbcon);

dbcon.Open();

SqlDataReader reader = dbcmd.ExecuteReader();

if (reader.HasRows)

{

while (reader.Read())

{

id = reader["id"].ToString();

name = reader["name\_k"].ToString();

gender = reader["sex"].ToString();

position = reader["position"].ToString();

depart = reader["depart"].ToString();

email = reader["e\_mail"].ToString();

ListViewItem item = new ListViewItem();

item.Text = id;

item.SubItems.Add(name);

item.SubItems.Add(gender);

item.SubItems.Add(position);

item.SubItems.Add(depart);

item.SubItems.Add(email);

this.lvMember.Items.Add(item);

}

}

reader.Close();

dbcon.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "오류!");

}

}

private void Member\_Load(object sender, EventArgs e)

{

try

{

SqlConnection dbcon = new SqlConnection();

dbcon.ConnectionString = Constr;

dbcon.Open();

String depart\_Sql = "Select val from department order by val asc";

SqlCommand d\_dbcmd = new SqlCommand(depart\_Sql, dbcon);

SqlDataReader d\_reader = d\_dbcmd.ExecuteReader();

if (d\_reader.HasRows)

{

while (d\_reader.Read())

{

this.cb\_depart.Items.Add(d\_reader["val"].ToString());

}

}

d\_reader.Close();

dbcon.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "오류!");

}

try

{

SqlConnection dbcon = new SqlConnection();

dbcon.ConnectionString = Constr;

dbcon.Open();

String position\_Sql = "Select val from position order by val asc";

SqlCommand p\_dbcmd = new SqlCommand(position\_Sql, dbcon);

SqlDataReader p\_reader = p\_dbcmd.ExecuteReader();

if (p\_reader.HasRows)

{

while (p\_reader.Read())

{

this.cb\_position.Items.Add(p\_reader["val"].ToString());

}

}

p\_reader.Close();

dbcon.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "오류!");

}

try

{

SqlConnection dbcon = new SqlConnection();

dbcon.ConnectionString = Constr;

dbcon.Open();

String religion\_Sql = "Select val from religion order by val asc";

SqlCommand r\_dbcmd = new SqlCommand(religion\_Sql, dbcon);

SqlDataReader r\_reader = r\_dbcmd.ExecuteReader();

if (r\_reader.HasRows)

{

while (r\_reader.Read())

{

this.cb\_religion.Items.Add(r\_reader["val"].ToString());

}

}

r\_reader.Close();

dbcon.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "오류!");

}

}

private void btn\_m\_insert\_Click(object sender, EventArgs e)

{

try

{

var Conn = new SqlConnection(Constr);

Conn.Open();

var strSQL = "insert into member values ('"

+ tb\_name\_k.Text + "', '"

+ tb\_name\_e.Text + "', '"

+ tb\_name\_c.Text + "', '"

+ tb\_address.Text + "', "

+ (this.rb\_married.Checked == true ? 1 : 0) + ",'"

+ tb\_email.Text + "', '"

+ tb\_call.Text + "', "

+ (this.rb\_male.Checked == true ? 0 : 1) + ","

+ int.Parse(tb\_height.Text) + ", "

+ int.Parse(tb\_weight.Text) + ", '"

+ cb\_blood.Text + "', '"

+ tb\_sight\_l.Text + "', '"

+ tb\_sight\_r.Text + "', '"

+ cb\_religion.Text + "', '"

+ cb\_depart.Text + "', '"

+ cb\_position.Text + "', '"

+ dtp\_birth\_date.Text + "', '"

+ dtp\_join\_date.Text + "', '"

+ tb\_id.Text + "', "

+ 1 + ")";

var myCom = new SqlCommand(strSQL, Conn);

myCom.ExecuteNonQuery();

Conn.Close();

MessageBox.Show("데이터가 저장되었습니다.", "알림");

new\_show();

}

catch (Exception ex)

{

MessageBox.Show("등록 불가", "오류!");

}

textClear();

}

private void lvMember\_SelectedIndexChanged(object sender, EventArgs e)

{

if (lvMember.SelectedItems.Count != 0)

{

int SelectRow = lvMember.SelectedItems[0].Index;

s\_id.Text = lvMember.Items[SelectRow].SubItems[0].Text;

var strSQL = "select \* from member where id='" + s\_id.Text + "' and work=1";

try

{

SqlConnection dbcon = new SqlConnection();

dbcon.ConnectionString = Constr;

SqlCommand dbcmd = new SqlCommand(strSQL, dbcon);

dbcon.Open();

SqlDataReader reader = dbcmd.ExecuteReader();

if (reader.HasRows)

{

while (reader.Read())

{

tb\_id.Text = reader["id"].ToString();

tb\_name\_k.Text = reader["name\_k"].ToString();

tb\_name\_e.Text= reader["name\_e"].ToString();

tb\_name\_c.Text= reader["name\_c"].ToString();

tb\_address.Text= reader["m\_address"].ToString();

if(reader["marriage"].ToString()=="0") { rb\_unmarried.Checked = true; }

else { rb\_married.Checked = true; }

tb\_email.Text = reader["e\_mail"].ToString();

tb\_call.Text = reader["c\_number"].ToString();

if(reader["sex"].ToString()=="0") { rb\_male.Checked = true; }

else { rb\_female.Checked = true; }

tb\_height.Text = reader["height"].ToString();

tb\_weight.Text = reader["weight\_"].ToString();

cb\_blood.Text = reader["blood"].ToString();

tb\_sight\_l.Text = reader["sight\_l"].ToString();

tb\_sight\_r.Text = reader["sight\_r"].ToString();

int idx\_r = cb\_religion.FindString(reader["religion"].ToString());

cb\_religion.SelectedIndex = idx\_r;

int idx\_d = cb\_depart.FindString(reader["depart"].ToString());

cb\_depart.SelectedIndex = idx\_d;

int idx\_p = cb\_position.FindString(reader["position"].ToString());

cb\_position.SelectedIndex = idx\_p;

dtp\_birth\_date.Text = reader["birth\_date"].ToString();

dtp\_join\_date.Text = reader["join\_date"].ToString();

tb\_id.Text = reader["id"].ToString();

}

}

reader.Close();

dbcon.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "오류!");

}

}

}

private void btn\_m\_update\_Click(object sender, EventArgs e)

{

var Conn = new SqlConnection(Constr);

Conn.Open();

var strSQL = "update member set name\_k='"

+ tb\_name\_k.Text + "', name\_e='"

+ tb\_name\_e.Text + "', name\_c='"

+ tb\_name\_c.Text + "', m\_address='"

+ tb\_address.Text + "', marriage="

+ (this.rb\_married.Checked == true ? 1 : 0) + ", e\_mail='"

+ tb\_email.Text + "', c\_number='"

+ tb\_call.Text + "', sex="

+ (this.rb\_male.Checked == true ? 0 : 1) + ", height="

+ int.Parse(tb\_height.Text) + ", weight\_="

+ int.Parse(tb\_weight.Text) + ", blood='"

+ cb\_blood.Text + "', sight\_l='"

+ tb\_sight\_l.Text + "', sight\_r='"

+ tb\_sight\_r.Text + "', religion='"

+ cb\_religion.Text + "', depart='"

+ cb\_depart.Text + "', position='"

+ cb\_position.Text + "', birth\_date='"

+ dtp\_birth\_date.Text + "', join\_date='"

+ dtp\_join\_date.Text + "', work=1 where id='" + s\_id.Text + "'";

var myCom = new SqlCommand(strSQL, Conn);

myCom.ExecuteNonQuery();

Conn.Close();

MessageBox.Show("데이터가 저장되었습니다.", "알림");

new\_show();

textClear();

}

private void btn\_m\_delete\_Click(object sender, EventArgs e)

{

var Conn1 = new SqlConnection(Constr);

Conn1.Open();

var Conn2 = new SqlConnection(Constr);

Conn2.Open();

var strSQL1 = "delete from member where id='" + s\_id.Text + "'";

var strSQL2 = "delete from family where f\_id='" + s\_id.Text + "'";

var myCom1 = new SqlCommand(strSQL1, Conn1);

myCom1.ExecuteNonQuery();

var myCom2 = new SqlCommand(strSQL2, Conn2);

myCom2.ExecuteNonQuery();

Conn1.Close();

Conn2.Close();

MessageBox.Show("데이터가 삭제되었습니다.", "알림");

new\_show();

textClear();

}

private void cb\_section\_SelectedIndexChanged(object sender, EventArgs e)

{

hidden\_s.Text = cb\_section.SelectedIndex.ToString();

}

private void btn\_m\_search\_Click(object sender, EventArgs e)

{

lvMember.View = View.Details;

lvMember.Items.Clear();

string id, name, gender, position, depart, email;

try

{

SqlConnection dbcon = new SqlConnection();

dbcon.ConnectionString = Constr;

String Sql;

string sql1 = "Select \* from member where id= '" + tb\_search.Text + "' and work=1 order by id asc";

string sql2 = "Select \* from member where name\_k= '" + tb\_search.Text + "' and work=1 order by id asc";

string sql3 = "Select \* from member where depart= '" + tb\_search.Text + "' and work=1 order by id asc";

string sql4 = "Select \* from member where depart= '" + tb\_search.Text + "' and work=1 order by id asc";

if (int.Parse(hidden\_s.Text) == 0) { Sql = sql1; }

else if (int.Parse(hidden\_s.Text) == 1) { Sql = sql2; }

else { Sql = sql3; }

if (String.IsNullOrEmpty(tb\_search.Text)) { Sql = sql4; }

SqlCommand dbcmd = new SqlCommand(Sql, dbcon);

dbcon.Open();

SqlDataReader reader = dbcmd.ExecuteReader();

if (reader.HasRows)

{

while (reader.Read())

{

id = reader["id"].ToString();

name = reader["name\_k"].ToString();

gender = reader["sex"].ToString();

position = reader["position"].ToString();

depart = reader["depart"].ToString();

email = reader["e\_mail"].ToString();

ListViewItem item = new ListViewItem();

item.Text = id;

item.SubItems.Add(name);

item.SubItems.Add(gender);

item.SubItems.Add(position);

item.SubItems.Add(depart);

item.SubItems.Add(email);

this.lvMember.Items.Add(item);

}

}

reader.Close();

dbcon.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "오류!");

}

textClear();

}

}

}

***[Family Form]***

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Data.SqlClient;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace myERP

{

public partial class Family : Form

{

private string Constr = "server=localhost;uid=myERP;pwd=1234;database=myERP";

public Family()

{

InitializeComponent();

}

private void textClear()

{

tb\_f\_id2.Text = "";

tb\_f\_name.Text = "";

tb\_f\_depart.Text = "";

tb\_f\_position.Text = "";

tb\_f\_relation.Text = "";

tb\_f\_name2.Text = "";

tb\_f\_place.Text = "";

tb\_f\_job.Text = "";

tb\_f\_number.Text = "";

}

private void new\_show()

{

lvFamily.View = View.Details;

lvFamily.Items.Clear();

string relation, f\_name, birth\_date, job, place, c\_number;

try

{

SqlConnection dbcon = new SqlConnection();

dbcon.ConnectionString = "server=localhost;uid=myERP;pwd=1234;database=myERP";

String Sql = "Select \* from family where f\_id='" + cb\_f\_id.Text + "'";

SqlCommand dbcmd = new SqlCommand(Sql, dbcon);

dbcon.Open();

SqlDataReader reader = dbcmd.ExecuteReader();

if (reader.HasRows)

{

while (reader.Read())

{

relation = reader["relation"].ToString();

f\_name = reader["f\_name"].ToString();

birth\_date = reader["birth\_date"].ToString();

job = reader["job"].ToString();

place = reader["place"].ToString();

c\_number = reader["c\_number"].ToString();

ListViewItem item = new ListViewItem();

item.Text = relation;

item.SubItems.Add(f\_name);

item.SubItems.Add(birth\_date);

item.SubItems.Add(job);

item.SubItems.Add(place);

item.SubItems.Add(c\_number);

this.lvFamily.Items.Add(item);

}

}

reader.Close();

dbcon.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "오류!");

}

}

private void btn\_f\_search\_Click(object sender, EventArgs e)

{

lvFamily.View = View.Details;

lvFamily.Items.Clear();

string relation, f\_name, birth\_date, job, place, c\_number;

try

{

SqlConnection dbcon = new SqlConnection();

dbcon.ConnectionString = "server=localhost;uid=myERP;pwd=1234;database=myERP";

String Sql = "Select \* from family as f right outer join member as m on f.f\_id=m.id where m.id='" + cb\_f\_id.Text + "'";

SqlCommand dbcmd = new SqlCommand(Sql, dbcon);

dbcon.Open();

SqlDataReader reader = dbcmd.ExecuteReader();

if (reader.HasRows)

{

while (reader.Read())

{

tb\_f\_id2.Text = reader["id"].ToString();

tb\_f\_name.Text = reader["name\_k"].ToString();

tb\_f\_depart.Text = reader["depart"].ToString();

tb\_f\_position.Text = reader["position"].ToString();

relation = reader["relation"].ToString();

f\_name = reader["f\_name"].ToString();

birth\_date = reader["birth\_date"].ToString();

job = reader["job"].ToString();

place = reader["place"].ToString();

c\_number = reader["c\_number"].ToString();

ListViewItem item = new ListViewItem();

item.Text = relation;

item.SubItems.Add(f\_name);

item.SubItems.Add(birth\_date);

item.SubItems.Add(job);

item.SubItems.Add(place);

item.SubItems.Add(c\_number);

this.lvFamily.Items.Add(item);

}

}

reader.Close();

dbcon.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "오류!");

}

}

private void Family\_Load(object sender, EventArgs e)

{

try

{

SqlConnection dbcon = new SqlConnection();

dbcon.ConnectionString = Constr;

dbcon.Open();

String Sql = "Select id from member where work=1 order by id asc";

SqlCommand d\_dbcmd = new SqlCommand(Sql, dbcon);

SqlDataReader d\_reader = d\_dbcmd.ExecuteReader();

if (d\_reader.HasRows)

{

while (d\_reader.Read())

{

this.cb\_f\_id.Items.Add(d\_reader["id"].ToString());

}

}

d\_reader.Close();

dbcon.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "오류!");

}

}

private void lvFamily\_SelectedIndexChanged(object sender, EventArgs e)

{

if (lvFamily.SelectedItems.Count != 0)

{

int SelectRow = lvFamily.SelectedItems[0].Index;

tb\_f\_relation.Text = lvFamily.Items[SelectRow].SubItems[0].Text;

tb\_f\_name2.Text = lvFamily.Items[SelectRow].SubItems[1].Text;

dtp\_f\_birth.Text = lvFamily.Items[SelectRow].SubItems[2].Text;

tb\_f\_job.Text = lvFamily.Items[SelectRow].SubItems[3].Text;

tb\_f\_place.Text = lvFamily.Items[SelectRow].SubItems[4].Text;

tb\_f\_number.Text = lvFamily.Items[SelectRow].SubItems[5].Text;

call\_hidden.Text = lvFamily.Items[SelectRow].SubItems[5].Text;

}

}

private void btn\_f\_insert\_Click(object sender, EventArgs e)

{

var Conn = new SqlConnection(Constr);

Conn.Open();

var strSQL = "insert into family values('"

+ cb\_f\_id.Text + "', '"

+ tb\_f\_relation.Text + "', '"

+ tb\_f\_place.Text + "', '"

+ tb\_f\_name2.Text + "', '"

+ tb\_f\_job.Text + "', '"

+ dtp\_f\_birth.Text + "', '"

+ tb\_f\_number.Text + "'"

+ ")";

var myCom = new SqlCommand(strSQL, Conn);

myCom.ExecuteNonQuery();

Conn.Close();

MessageBox.Show("데이터가 저장되었습니다.", "알림");

new\_show();

textClear();

}

private void btn\_f\_update\_Click(object sender, EventArgs e)

{

var Conn = new SqlConnection(Constr);

Conn.Open();

var strSQL = "update family set relation='"

+ tb\_f\_relation.Text + "', place='"

+ tb\_f\_place.Text + "', f\_name='"

+ tb\_f\_name2.Text + "', job='"

+ tb\_f\_job.Text + "', birth\_date='"

+ dtp\_f\_birth.Text + "', c\_number='"

+ tb\_f\_number.Text + "' where f\_id='" + cb\_f\_id.Text + "'and c\_number='" + call\_hidden.Text + "'";

var myCom = new SqlCommand(strSQL, Conn);

myCom.ExecuteNonQuery();

Conn.Close();

MessageBox.Show("데이터가 변경되었습니다.", "알림");

new\_show();

textClear();

}

private void btn\_f\_delete\_Click(object sender, EventArgs e)

{

var Conn = new SqlConnection(Constr);

Conn.Open();

var strSQL = "delete from family where f\_id='" + cb\_f\_id.Text + "'and c\_number='" + call\_hidden.Text + "'";

var myCom = new SqlCommand(strSQL, Conn);

myCom.ExecuteNonQuery();

Conn.Close();

MessageBox.Show("데이터가 삭제되었습니다.", "알림");

new\_show();

textClear();

}

}

}

***[Retired Form]***

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Data.SqlClient;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace myERP

{

public partial class Retired : Form

{

private string Constr = "server=localhost;uid=myERP;pwd=1234;database=myERP";

public Retired()

{

InitializeComponent();

rb\_S\_reason.Checked = true;

}

private void textClear()

{

tb\_AU\_name.Text = "";

tb\_AU\_call.Text = "";

}

private void radio\_state() // 라디오 버튼 상태에 따른 sql문 적용 후 정렬

{

String Sql= "Select \* from retired order by";

if (rb\_S\_reason.Checked==true) { Sql += " r\_reason asc"; }

else if(rb\_S\_depart.Checked==true) { Sql += " depart asc"; }

else if(rb\_S\_date.Checked==true) { Sql += " retire\_date asc"; }

else if(rb\_S\_position.Checked==true) { Sql += " position asc"; }

lvRetired.View = View.Details;

lvRetired.Items.Clear();

string id, name, call, depart, position, date, reason;

try

{

SqlConnection dbcon = new SqlConnection();

dbcon.ConnectionString = Constr;

dbcon.Open();

SqlCommand dbcmd = new SqlCommand(Sql, dbcon);

SqlDataReader reader = dbcmd.ExecuteReader();

if (reader.HasRows)

{

while (reader.Read())

{

id = reader["id"].ToString();

name = reader["r\_name"].ToString();

call = reader["c\_number"].ToString();

depart = reader["depart"].ToString();

position = reader["position"].ToString();

date = reader["retire\_date"].ToString();

reason = reader["r\_reason"].ToString();

ListViewItem item = new ListViewItem();

item.Text = id;

item.SubItems.Add(name);

item.SubItems.Add(call);

item.SubItems.Add(depart);

item.SubItems.Add(position);

item.SubItems.Add(date);

item.SubItems.Add(reason);

this.lvRetired.Items.Add(item);

}

}

reader.Close();

dbcon.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "오류!");

}

}

private void new\_show() // lv\_served, lv\_retired listview 새로고침

{

lv\_served.View = View.Details;

lv\_retired.View = View.Details;

lv\_served.Items.Clear();

lv\_retired.Items.Clear();

try

{

SqlConnection dbcon = new SqlConnection();

dbcon.ConnectionString = Constr;

dbcon.Open();

String Sql = "Select \* from member where work=1 order by id asc";

String id, name;

SqlCommand dbcmd = new SqlCommand(Sql, dbcon);

SqlDataReader reader = dbcmd.ExecuteReader();

if (reader.HasRows)

{

while (reader.Read())

{

id = reader["id"].ToString();

name = reader["name\_k"].ToString();

ListViewItem item = new ListViewItem();

item.Text = id;

item.SubItems.Add(name);

this.lv\_served.Items.Add(item);

}

}

reader.Close();

dbcon.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "오류!");

}

try

{

SqlConnection dbcon = new SqlConnection();

dbcon.ConnectionString = Constr;

dbcon.Open();

String Sql = "Select \* from member where work=0 order by id asc";

String id, name;

SqlCommand dbcmd = new SqlCommand(Sql, dbcon);

SqlDataReader reader = dbcmd.ExecuteReader();

if (reader.HasRows)

{

while (reader.Read())

{

id = reader["id"].ToString();

name = reader["name\_k"].ToString();

ListViewItem item = new ListViewItem();

item.Text = id;

item.SubItems.Add(name);

this.lv\_retired.Items.Add(item);

}

}

reader.Close();

dbcon.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "오류!");

}

}

private void Retired\_Load(object sender, EventArgs e)

{

try

{

SqlConnection dbcon = new SqlConnection();

dbcon.ConnectionString = Constr;

dbcon.Open();

String Sql = "Select \* from member where work=1 order by id asc";

String id, name;

SqlCommand dbcmd = new SqlCommand(Sql, dbcon);

SqlDataReader reader = dbcmd.ExecuteReader();

if (reader.HasRows)

{

while (reader.Read())

{

id = reader["id"].ToString();

name = reader["name\_k"].ToString();

ListViewItem item = new ListViewItem();

item.Text = id;

item.SubItems.Add(name);

this.lv\_served.Items.Add(item);

}

}

reader.Close();

dbcon.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "오류!");

}

try

{

SqlConnection dbcon = new SqlConnection();

dbcon.ConnectionString = Constr;

dbcon.Open();

String Sql = "Select \* from member where work=0 order by id asc";

String id, name;

SqlCommand dbcmd = new SqlCommand(Sql, dbcon);

SqlDataReader reader = dbcmd.ExecuteReader();

if (reader.HasRows)

{

while (reader.Read())

{

id = reader["id"].ToString();

name = reader["name\_k"].ToString();

ListViewItem item = new ListViewItem();

item.Text = id;

item.SubItems.Add(name);

this.lv\_retired.Items.Add(item);

}

}

reader.Close();

dbcon.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "오류!");

}

try

{

SqlConnection dbcon = new SqlConnection();

dbcon.ConnectionString = Constr;

dbcon.Open();

String depart\_Sql = "Select val from department order by val asc";

SqlCommand d\_dbcmd = new SqlCommand(depart\_Sql, dbcon);

SqlDataReader d\_reader = d\_dbcmd.ExecuteReader();

if (d\_reader.HasRows)

{

while (d\_reader.Read())

{

this.cb\_AU\_depart.Items.Add(d\_reader["val"].ToString());

}

}

d\_reader.Close();

dbcon.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "오류!");

}

try

{

SqlConnection dbcon = new SqlConnection();

dbcon.ConnectionString = Constr;

dbcon.Open();

String position\_Sql = "Select val from position order by val asc";

SqlCommand p\_dbcmd = new SqlCommand(position\_Sql, dbcon);

SqlDataReader p\_reader = p\_dbcmd.ExecuteReader();

if (p\_reader.HasRows)

{

while (p\_reader.Read())

{

this.cb\_AU\_position.Items.Add(p\_reader["val"].ToString());

}

}

p\_reader.Close();

dbcon.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "오류!");

}

try

{

SqlConnection dbcon = new SqlConnection();

dbcon.ConnectionString = Constr;

dbcon.Open();

String religion\_Sql = "Select val from retirement\_reason order by val asc";

SqlCommand r\_dbcmd = new SqlCommand(religion\_Sql, dbcon);

SqlDataReader r\_reader = r\_dbcmd.ExecuteReader();

if (r\_reader.HasRows)

{

while (r\_reader.Read())

{

this.cb\_AU\_reason.Items.Add(r\_reader["val"].ToString());

}

}

r\_reader.Close();

dbcon.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "오류!");

}

}

private void btn\_r\_search\_Click(object sender, EventArgs e)

{

try

{

SqlConnection dbcon = new SqlConnection();

dbcon.ConnectionString = "server=localhost;uid=myERP;pwd=1234;database=myERP";

String Sql = "Select \* from member where id='" + cb\_r\_id.Text + "'";

SqlCommand dbcmd = new SqlCommand(Sql, dbcon);

dbcon.Open();

SqlDataReader reader = dbcmd.ExecuteReader();

if (reader.HasRows)

{

while (reader.Read())

{

cb\_r\_id.Text = reader["id"].ToString();

tb\_AU\_name.Text = reader["name\_k"].ToString();

tb\_AU\_call.Text = reader["c\_number"].ToString();

int idx\_d = cb\_AU\_depart.FindString(reader["depart"].ToString());

cb\_AU\_depart.SelectedIndex = idx\_d;

int idx\_p = cb\_AU\_position.FindString(reader["position"].ToString());

cb\_AU\_position.SelectedIndex = idx\_p;

}

}

reader.Close();

dbcon.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "오류!");

}

try

{

SqlConnection dbcon = new SqlConnection();

dbcon.ConnectionString = "server=localhost;uid=myERP;pwd=1234;database=myERP";

String Sql = "Select \* from retired where id='" + cb\_r\_id.Text + "'";

SqlCommand dbcmd = new SqlCommand(Sql, dbcon);

dbcon.Open();

SqlDataReader reader = dbcmd.ExecuteReader();

if (reader.HasRows)

{

while (reader.Read())

{

dtp\_AU\_date.Text = reader["retire\_date"].ToString();

int idx\_r = cb\_AU\_reason.FindString(reader["r\_reason"].ToString());

cb\_AU\_reason.SelectedIndex = idx\_r;

}

}

reader.Close();

dbcon.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "오류!");

}

}

private void btn\_AU\_add\_Click(object sender, EventArgs e)

{

var Conn = new SqlConnection(Constr);

Conn.Open();

var strSQL = "insert into retired values('"

+ cb\_r\_id.Text + "', '"

+ tb\_AU\_name.Text + "', '"

+ tb\_AU\_call.Text + "', '"

+ cb\_AU\_depart.Text + "', '"

+ cb\_AU\_position.Text + "', '"

+ dtp\_AU\_date.Text + "', '"

+ cb\_AU\_reason.Text + "'"

+ ")";

var myCom = new SqlCommand(strSQL, Conn);

myCom.ExecuteNonQuery();

Conn.Close();

MessageBox.Show("데이터가 저장되었습니다.", "알림");

textClear();

}

private void btn\_AU\_update\_Click(object sender, EventArgs e)

{

var Conn = new SqlConnection(Constr);

Conn.Open();

var strSQL = "update retired set retire\_date='"

+ dtp\_AU\_date.Text + "', r\_reason='"

+ cb\_AU\_reason.Text + "' where id='" + cb\_r\_id.Text + "'";

var myCom = new SqlCommand(strSQL, Conn);

myCom.ExecuteNonQuery();

Conn.Close();

MessageBox.Show("데이터가 수정되었습니다.", "알림");

textClear();

}

private void rb\_S\_reason\_CheckedChanged(object sender, EventArgs e)

{

radio\_state();

}

private void rb\_S\_depart\_CheckedChanged(object sender, EventArgs e)

{

radio\_state();

}

private void rb\_S\_date\_CheckedChanged(object sender, EventArgs e)

{

radio\_state();

}

private void rb\_S\_position\_CheckedChanged(object sender, EventArgs e)

{

radio\_state();

}

private void btn\_add\_Click(object sender, EventArgs e)

{

var Conn = new SqlConnection(Constr);

Conn.Open();

var strSQL = "update member set work=0 where id='" + retire\_hidden.Text + "'";

var myCom = new SqlCommand(strSQL, Conn);

myCom.ExecuteNonQuery();

Conn.Close();

new\_show();

MessageBox.Show("세부 사항을 입력하세요!", "알림");

}

private void btn\_del\_Click(object sender, EventArgs e)

{

var Conn = new SqlConnection(Constr);

var Conn2 = new SqlConnection(Constr);

Conn.Open();

Conn2.Open();

var strSQL = "update member set work=1 where id='" + retire\_hidden.Text + "'";

var strSQL2 = "delete from retired where id='" + retire\_hidden.Text + "'";

var myCom = new SqlCommand(strSQL, Conn);

var myCom2 = new SqlCommand(strSQL2, Conn2);

myCom.ExecuteNonQuery();

myCom2.ExecuteNonQuery();

Conn.Close();

Conn2.Close();

new\_show();

}

private void lv\_served\_SelectedIndexChanged(object sender, EventArgs e)

{

if (lv\_served.SelectedItems.Count != 0)

{

int SelectRow = lv\_served.SelectedItems[0].Index;

retire\_hidden.Text = lv\_served.Items[SelectRow].SubItems[0].Text;

}

}

private void tabControl1\_SelectedIndexChanged(object sender, EventArgs e)

{

cb\_r\_id.Items.Clear();

radio\_state();

try

{

SqlConnection dbcon = new SqlConnection();

dbcon.ConnectionString = Constr;

dbcon.Open();

String Sql = "Select \* from member where work=0 order by id asc";

SqlCommand dbcmd = new SqlCommand(Sql, dbcon);

SqlDataReader reader = dbcmd.ExecuteReader();

if (reader.HasRows)

{

while (reader.Read())

{

this.cb\_r\_id.Items.Add(reader["id"].ToString());

}

}

reader.Close();

dbcon.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "오류!");

}

}

private void lv\_retired\_SelectedIndexChanged(object sender, EventArgs e)

{

if (lv\_retired.SelectedItems.Count != 0)

{

int SelectRow = lv\_retired.SelectedItems[0].Index;

retire\_hidden.Text = lv\_retired.Items[SelectRow].SubItems[0].Text;

}

}

}

}

***[Report Form]***

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Data.SqlClient;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace myERP

{

public partial class Report : Form

{

private string Constr = "server=localhost;uid=myERP;pwd=1234;database=myERP";

public Report()

{

InitializeComponent();

this.rb\_total.Checked = true;

}

private void rb\_total\_CheckedChanged(object sender, EventArgs e)

{

// listview

lv\_Report.Clear();

ColumnHeader col\_id = new ColumnHeader(); //객체 선언

ColumnHeader col\_name = new ColumnHeader();

ColumnHeader col\_depart = new ColumnHeader();

ColumnHeader col\_position = new ColumnHeader();

ColumnHeader col\_jDate = new ColumnHeader();

ColumnHeader col\_bDate = new ColumnHeader();

col\_id.Text = "사번"; //열 제목 표시

col\_name.Text = "성명";

col\_depart.Text = "부서";

col\_position.Text = "직위";

col\_jDate.Text = "입사일자";

col\_bDate.Text = "생년월일";

lv\_Report.Columns.Add(col\_id); //열 추가 실행

lv\_Report.Columns.Add(col\_name);

lv\_Report.Columns.Add(col\_depart);

lv\_Report.Columns.Add(col\_position);

lv\_Report.Columns.Add(col\_jDate);

lv\_Report.Columns.Add(col\_bDate);

lv\_Report.Columns[0].Width = 100;

lv\_Report.Columns[1].Width = 90;

lv\_Report.Columns[2].Width = 90;

lv\_Report.Columns[3].Width = 90;

lv\_Report.Columns[4].Width = 170;

lv\_Report.Columns[5].Width = 170;

}

private void rb\_reason\_CheckedChanged(object sender, EventArgs e)

{

// listview

lv\_Report.Clear();

ColumnHeader col\_id = new ColumnHeader(); //객체 선언

ColumnHeader col\_name = new ColumnHeader();

ColumnHeader col\_depart = new ColumnHeader();

ColumnHeader col\_position = new ColumnHeader();

ColumnHeader col\_rDate = new ColumnHeader();

col\_id.Text = "사번"; //열 제목 표시

col\_name.Text = "성명";

col\_depart.Text = "부서";

col\_position.Text = "직위";

col\_rDate.Text = "퇴사일자";

lv\_Report.Columns.Add(col\_id); //열 추가 실행

lv\_Report.Columns.Add(col\_name);

lv\_Report.Columns.Add(col\_depart);

lv\_Report.Columns.Add(col\_position);

lv\_Report.Columns.Add(col\_rDate);

lv\_Report.Columns[0].Width = 100;

lv\_Report.Columns[1].Width = 90;

lv\_Report.Columns[2].Width = 90;

lv\_Report.Columns[3].Width = 90;

lv\_Report.Columns[4].Width = 170;

// combobox

cb\_report.Items.Clear();

try

{

SqlConnection dbcon = new SqlConnection();

dbcon.ConnectionString = Constr;

dbcon.Open();

String Sql = "Select \* from retirement\_reason";

SqlCommand dbcmd = new SqlCommand(Sql, dbcon);

SqlDataReader reader = dbcmd.ExecuteReader();

if (reader.HasRows)

{

while (reader.Read())

{

cb\_report.Items.Add(reader["val"].ToString());

}

}

reader.Close();

dbcon.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "오류!");

}

}

private void rb\_depart\_CheckedChanged(object sender, EventArgs e)

{

// listview

lv\_Report.Clear();

ColumnHeader col\_id = new ColumnHeader(); //객체 선언

ColumnHeader col\_name = new ColumnHeader();

ColumnHeader col\_position = new ColumnHeader();

ColumnHeader col\_jDate = new ColumnHeader();

ColumnHeader col\_bDate = new ColumnHeader();

col\_id.Text = "사번"; //열 제목 표시

col\_name.Text = "성명";

col\_position.Text = "직위";

col\_jDate.Text = "입사일자";

col\_bDate.Text = "생년월일";

lv\_Report.Columns.Add(col\_id); //열 추가 실행

lv\_Report.Columns.Add(col\_name);

lv\_Report.Columns.Add(col\_position);

lv\_Report.Columns.Add(col\_jDate);

lv\_Report.Columns.Add(col\_bDate);

lv\_Report.Columns[0].Width = 100;

lv\_Report.Columns[1].Width = 90;

lv\_Report.Columns[2].Width = 90;

lv\_Report.Columns[3].Width = 170;

lv\_Report.Columns[4].Width = 170;

// combobox

cb\_report.Items.Clear();

try

{

SqlConnection dbcon = new SqlConnection();

dbcon.ConnectionString = Constr;

dbcon.Open();

String Sql = "Select \* from department";

SqlCommand dbcmd = new SqlCommand(Sql, dbcon);

SqlDataReader reader = dbcmd.ExecuteReader();

if (reader.HasRows)

{

while (reader.Read())

{

cb\_report.Items.Add(reader["val"].ToString());

}

}

reader.Close();

dbcon.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "오류!");

}

}

private void rb\_position\_CheckedChanged(object sender, EventArgs e)

{

// listview

lv\_Report.Clear();

ColumnHeader col\_id = new ColumnHeader(); //객체 선언

ColumnHeader col\_name = new ColumnHeader();

ColumnHeader col\_depart = new ColumnHeader();

ColumnHeader col\_jDate = new ColumnHeader();

ColumnHeader col\_bDate = new ColumnHeader();

col\_id.Text = "사번"; //열 제목 표시

col\_name.Text = "성명";

col\_depart.Text = "부서";

col\_jDate.Text = "입사일자";

col\_bDate.Text = "생년월일";

lv\_Report.Columns.Add(col\_id); //열 추가 실행

lv\_Report.Columns.Add(col\_name);

lv\_Report.Columns.Add(col\_depart);

lv\_Report.Columns.Add(col\_jDate);

lv\_Report.Columns.Add(col\_bDate);

lv\_Report.Columns[0].Width = 100;

lv\_Report.Columns[1].Width = 90;

lv\_Report.Columns[2].Width = 90;

lv\_Report.Columns[3].Width = 170;

lv\_Report.Columns[4].Width = 170;

// combobox

cb\_report.Items.Clear();

try

{

SqlConnection dbcon = new SqlConnection();

dbcon.ConnectionString = Constr;

dbcon.Open();

String Sql = "Select \* from position";

SqlCommand dbcmd = new SqlCommand(Sql, dbcon);

SqlDataReader reader = dbcmd.ExecuteReader();

if (reader.HasRows)

{

while (reader.Read())

{

cb\_report.Items.Add(reader["val"].ToString());

}

}

reader.Close();

dbcon.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "오류!");

}

}

private void btn\_report\_Click(object sender, EventArgs e)

{

lv\_Report.View = View.Details;

lv\_Report.Items.Clear();

if (rb\_total.Checked==true)

{

try

{

SqlConnection dbcon = new SqlConnection();

dbcon.ConnectionString = Constr;

dbcon.Open();

String Sql = "Select \* from member where work = 1 order by id asc";

String id, name, depart, position, jDate, bDate;

SqlCommand dbcmd = new SqlCommand(Sql, dbcon);

SqlDataReader reader = dbcmd.ExecuteReader();

if (reader.HasRows)

{

while (reader.Read())

{

id = reader["id"].ToString();

name = reader["name\_k"].ToString();

depart = reader["depart"].ToString();

position = reader["position"].ToString();

jDate = reader["join\_date"].ToString();

bDate = reader["birth\_date"].ToString();

ListViewItem item = new ListViewItem();

item.Text = id;

item.SubItems.Add(name);

item.SubItems.Add(depart);

item.SubItems.Add(position);

item.SubItems.Add(jDate);

item.SubItems.Add(bDate);

this.lv\_Report.Items.Add(item);

}

}

reader.Close();

dbcon.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "오류!");

}

}

else if(rb\_reason.Checked==true)

{

try

{

SqlConnection dbcon = new SqlConnection();

dbcon.ConnectionString = Constr;

dbcon.Open();

String Sql = "Select member.id, member.name\_k, member.depart, member.position, retired.retire\_date" +

" from member inner join retired on member.id=retired.id where work = 0 and retired.r\_reason='"

+ cb\_report.Text + "' order by id asc";

String id, name, depart, position, rDate;

SqlCommand dbcmd = new SqlCommand(Sql, dbcon);

SqlDataReader reader = dbcmd.ExecuteReader();

if (reader.HasRows)

{

while (reader.Read())

{

id = reader["id"].ToString();

name = reader["name\_k"].ToString();

depart = reader["depart"].ToString();

position = reader["position"].ToString();

rDate = reader["retire\_date"].ToString();

ListViewItem item = new ListViewItem();

item.Text = id;

item.SubItems.Add(name);

item.SubItems.Add(depart);

item.SubItems.Add(position);

item.SubItems.Add(rDate);

this.lv\_Report.Items.Add(item);

}

}

reader.Close();

dbcon.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "오류!");

}

}

else if(rb\_depart.Checked==true)

{

try

{

SqlConnection dbcon = new SqlConnection();

dbcon.ConnectionString = Constr;

dbcon.Open();

String Sql = "Select \* from member where work = 1 and depart='" + cb\_report.Text + "' order by id asc";

String id, name, position, jDate, bDate;

SqlCommand dbcmd = new SqlCommand(Sql, dbcon);

SqlDataReader reader = dbcmd.ExecuteReader();

if (reader.HasRows)

{

while (reader.Read())

{

id = reader["id"].ToString();

name = reader["name\_k"].ToString();

position = reader["position"].ToString();

jDate = reader["join\_date"].ToString();

bDate = reader["birth\_date"].ToString();

ListViewItem item = new ListViewItem();

item.Text = id;

item.SubItems.Add(name);

item.SubItems.Add(position);

item.SubItems.Add(jDate);

item.SubItems.Add(bDate);

this.lv\_Report.Items.Add(item);

}

}

reader.Close();

dbcon.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "오류!");

}

}

else if(rb\_position.Checked==true)

{

try

{

SqlConnection dbcon = new SqlConnection();

dbcon.ConnectionString = Constr;

dbcon.Open();

String Sql = "Select \* from member where work = 1 and position='" + cb\_report.Text + "' order by id asc";

String id, name, depart, jDate, bDate;

SqlCommand dbcmd = new SqlCommand(Sql, dbcon);

SqlDataReader reader = dbcmd.ExecuteReader();

if (reader.HasRows)

{

while (reader.Read())

{

id = reader["id"].ToString();

name = reader["name\_k"].ToString();

depart = reader["depart"].ToString();

jDate = reader["join\_date"].ToString();

bDate = reader["birth\_date"].ToString();

ListViewItem item = new ListViewItem();

item.Text = id;

item.SubItems.Add(name);

item.SubItems.Add(depart);

item.SubItems.Add(jDate);

item.SubItems.Add(bDate);

this.lv\_Report.Items.Add(item);

}

}

reader.Close();

dbcon.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "오류!");

}

}

}

}

}

|  |  |  |
| --- | --- | --- |
| **4.** |  | **프로젝트 개선 방안** |

- web 기반의 그룹웨어 형태의 ERP로 모바일 및 UWP 기반 플랫폼 가변화

- 네트워크 및 다중쓰레드를 통한 사내 게시판, 메신저 및 전자 문서 결재 기능

- 보고서 메뉴에서 조회한 자료의 pdf 및 excel 문서 변환 기능

- 가족 구성원 직계 존비속 중복 등록 여부 확인 및 가족 복지 혜택 이벤트

- 사원 등록 시 일련 번호를 추가하고 생성 방식을 자동 생성 방식으로 개선

|  |  |  |
| --- | --- | --- |
| **5.** |  | **참고문헌 및 URL** |

- <https://ahhang0k.tistory.com/108> (DB 연동하여 listview로 데이터 추가)

- <https://blackhunydev.tistory.com/232> (listview에서 선택한 데이터 가져오기)

- <https://scrawled-note.tistory.com/entry/WPF-C-TabControl%EC%9D%98Tab-%EC%84%A0%ED%83%9D-%EC%9D%B4%EB%B2%A4%ED%8A%B8> (TabControl 이동 이벤트)

- <https://jw0652.tistory.com/12> (combobox 항목 추가)

- <https://afsdzvcx123.tistory.com/entry/C-%EC%9C%88%ED%8F%BC-RadioButton%EB%9D%BC%EB%94%94%EC%98%A4%EB%B2%84%ED%8A%BC-%EC%BB%A8%ED%8A%B8%EB%A1%A4-CheckedEvent-%EB%B0%9C%EC%83%9D%EC%8B%9C%ED%82%A4%EA%B8%B0> (radiobutton 이벤트 발생)