데이터베이스 I

(11주차)

학습개요

- 학습 목표
  - 다양한 데이터베이스 프로그래밍 인터페이스의 특징을 이해한다.
  - MFC를 이용한 ODBC 프로그래밍 기법을 익힌다.

- 학습 내용
  - 데이터베이스 프로그래밍 인터페이스
  - ODBC 프로그래밍
  - 실습

개요

- 데이터베이스(Database)
  - 통합되어 저장되고 관리되는 데이터의 집합
- DBMS(Database Management System)
  - 데이터베이스를 관리하는 소프트웨어 시스템
    - Oracle, IBM DB2, MS SQL Server, ...
  - 응용 프로그램과 데이터베이스를 매개하는 역할
    - 데이터 중복 최소화
    - 효과적인 데이터 공유
    - 데이터의 일관성과 무결성 유지
    - 데이터 보안 보장

데이터베이스 프로그래밍 인터페이스

- 전용(Proprietary) 프로그래밍 인터페이스
  - DBMS마다 별도로 제공하는 비표준 프로그래밍 인터페이스
    - 성능이나 효율성은 뛰어나지만 DBMS에 종속적이므로 DBMS를 바꾸면 응용 프로그램 코드를 새로 작성해야 함
- 공용(Universal) 프로그래밍 인터페이스
  - 일관된 인터페이스로 다양한 종류의 DBMS에 접근할 수 있는 표준 프로그래밍 인터페이스
    - 성능이나 효율성은 전용 인터페이스보다 떨어지지만 DBMS를 바꿔도 응용 프로그램 코드를 재활용할 수 있음

윈도우 데이터베이스 프로그래밍 인터페이스

- DAO(Data Access Objects)
  - MS 제트(Jet) 데이터베이스 엔진을 이용하여 데이터베이스에 접근하는 인터페이스
  - MFC의 CDaoDatabase 클래스로 프로그래밍 가능
- ODBC(Open Database Connectivity)
  - 하나의 인터페이스로 다양한 종류의 DBMS에 접근할 수 있게 만든 성공적인 공개 인터페이스
  - MFC의 CDatabase 클래스로 프로그래밍 가능

윈도우 데이터베이스 프로그래밍 인터페이스

- RDO(Remote Data Objects)
  - COM(Component Object Model) 기술을 이용하여 ODBC를 포장한 인터페이스
- OLE DB(Object Linking and Embedding for Database)
  - COM 기술에 기반한 새로운 데이터베이스 프로그래밍 방법으로 성능이 뛰어나며, 윈도우 운영체제의 강력한 지원을 받고 있음
  - OLE DB 공급자를 통해 다양한 종류의 DBMS에 접근할 수 있고, ODBC용 OLE DB 공급자를 사용하여 기존의 ODBC도 지원
  - 윈도우 운영체제에서만 동작한다는 단점이 있음

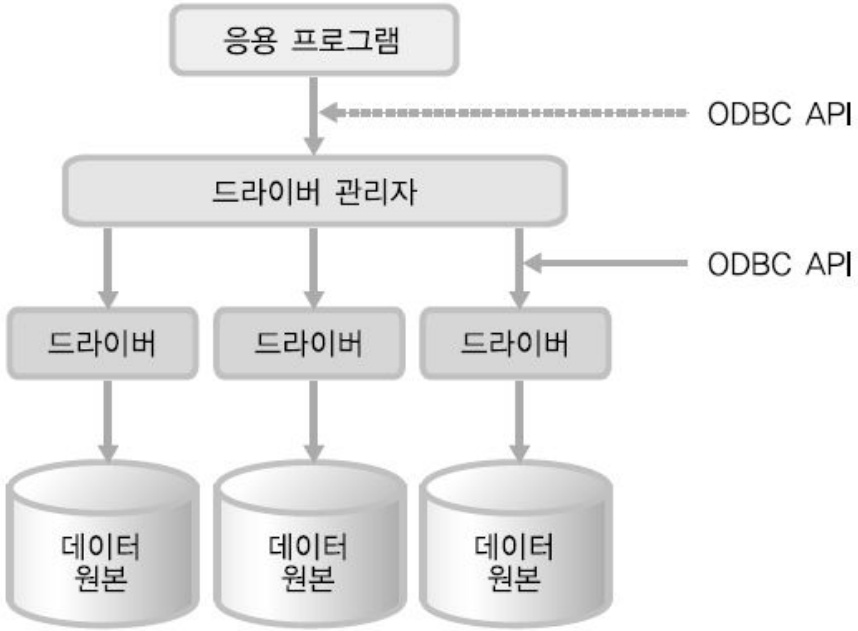
윈도우 데이터베이스 프로그래밍 인터페이스

- ADO(ActiveX Data Objects)
  - OLE DB가 제공하는 기능을 좀 더 쉽게 사용할 수 있게 만든 COM 기술 기반의 프로그래밍 인터페이스
  - 내부적으로 OLE DB를 이용하므로 다양한 DBMS에 접근할 수 있고 뛰어난 성능을 냄
  - 언어 독립적이어서 베이직, C/C++, 자바 등 다양한 언어로 프로그래밍 가능

※ 일반적으로 OLE DB는 시스템 프로그래밍 인터페이스,
ADO는 응용 프로그래밍 인터페이스로 분류함

ODBC 구조

- ODBC 구조



ODBC 구조

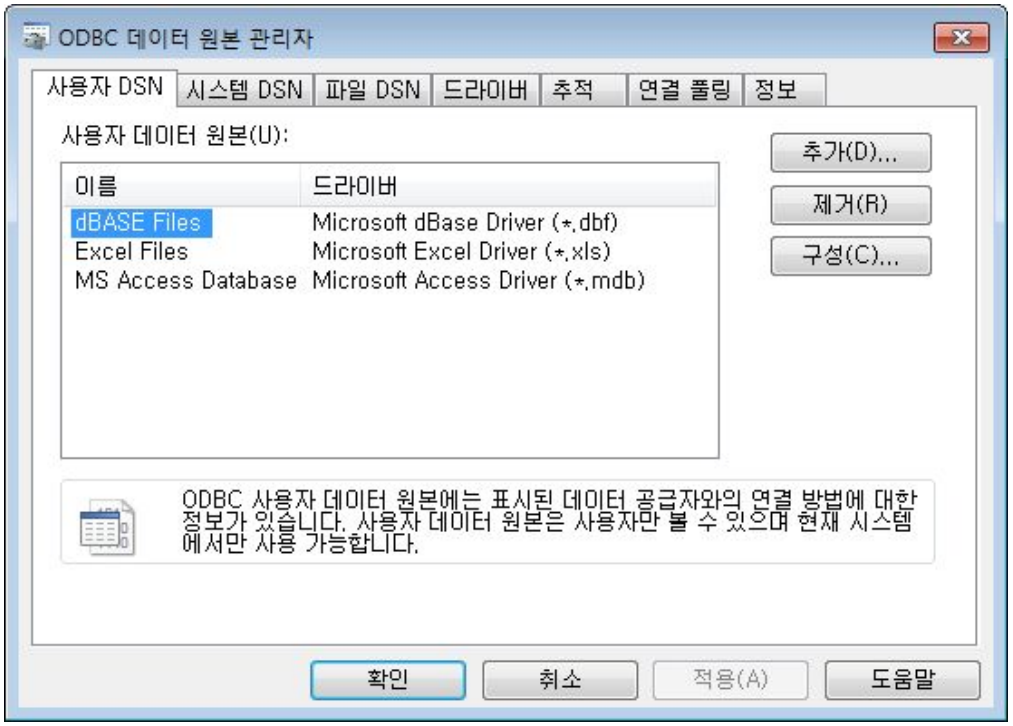
- 응용 프로그램
  - ODBC API를 이용해 작성하고, ODBC API를 통해 드라이버 관리자와 통신
  - 데이터베이스에 연결할 때는 ODBC API를 사용하지만, 실제 데이터를 다룰 때는 관계형 데이터베이스 조작 언어인 SQL을 사용
- 드라이버 관리자
  - 응용 프로그램과 특정 DBMS 드라이버를 매개하는 역할
  - 응용 프로그램이 요구한 데이터베이스에 접근할 수 있도록 ODBC 드라이버를 로드하고, 응용 프로그램과 동일한 API를 이용하여 드라이버의 함수를 호출

ODBC 구조

- 드라이버
  - ODBC API 구현을 제공하며 특정 DBMS에 종속적임
  - MS SQL Server처럼 DBMS가 자체 엔진을 제공하는 경우에는 드라이버가 SQL문을 DBMS에 전달. 엑셀 파일처럼 자체 엔진을 제공하지 않는 경우에는 드라이버가 직접 SQL문을 처리
- 데이터 원본(Data Source)
  - 데이터베이스에 접근하기 위해 필요한 정보와 데이터베이스 자체를 총칭
  - 데이터 원본이 있어야 ODBC를 이용해 해당 데이터베이스에 접근 가능

ODBC 구조

- ODBC 데이터 원본 관리자



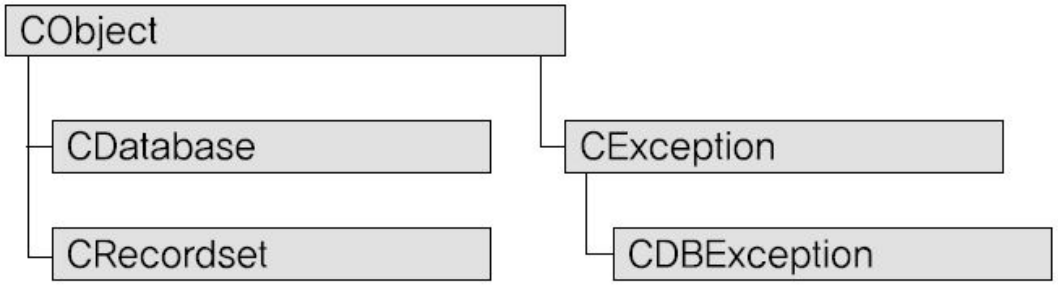
ODBC 구조

- 데이터 원본 종류

분류		특성
기계 데이터 원본 (Machine Data Sources)	사용자 DSN	정보가 레지스트리에 저장되며, 현재 로그인한 사용자만 접근 가능
	시스템 DSN	정보가 레지스트리에 저장되며, 모든 사용자가 접근 가능
파일 데이터 원본 (File Data Sources)	파일 DSN	정보가 디스크 파일에 저장되며, 이 파일을 가진 사용자만 접근 가능

MFC ODBC 클래스

- MFC 클래스 계층도



MFC ODBC 클래스

- 데이터베이스 클래스
  - CDatabase 클래스는 데이터베이스와의 연결을 나타내며, 제공되는 멤버 함수를 이용하여 연결된 데이터베이스를 조작함
  - 데이터베이스에 접근하려면 먼저 CDatabase 객체를 만들어야 함

```
// 데이터베이스 객체 생성
CDatabase db;
db.OpenEx(…);
…
// 종료 처리
db.Close();
```

MFC ODBC 클래스

- 레코드셋 클래스
  - 레코드셋(Recordset)이란?
    - 데이터 조작 연산을 통해 얻은 레코드 집합
  - CRecordSet 클래스는 레코드셋을 나타내며, 데이터를 검색/추가/삭제/갱신 등을 할 때 핵심 역할을 함

- 데이터베이스 사용 절차
  ① CDatabase 객체 생성
  ② CRecordSet 객체 생성
    - CDatabase 객체를 CRecordSet 클래스 생성자에 전달
  ③ CRecordSet 클래스가 제공하는 다양한 멤버 함수를 통해 데이터 조작

MFC ODBC 클래스

- 레코드셋 클래스

```
// 데이터베이스 객체 생성
CDatabase db;
db.OpenEx(…);

// 레코드셋 객체 생성
CRecordset rs(&db);
rs.Open(…);

// 데이터 검색, 추가, 삭제, 갱신, …
…
// 종료 처리
rs.Close();
db.Close();
```

MFC ODBC 클래스

- 예외 처리 클래스
  - CDBException 클래스는 데이터베이스를 조작할 때 발생하는 오류를 나타냄

```
try {
    // 데이터베이스 조작
    …
}
catch(CDBException *e)
{
    e->ReportError();
    e->Delete();
}
```

주요 함수

- 데이터베이스 객체 생성

```
BOOL CDatabase::OpenEx(
① LPCTSTR lpszConnectString,
② DWORD dwOptions = 0
);
```

  - lpszConnectString
    - 사용할 데이터 원본 이름과 더불어 ID, 암호 같은 부가 정보를 문자열로 넘겨줌. ID와 암호가 없으면 생략 가능
      (예) db.OpenEx(_T("DSN=student;UID=chulsoo;PWD=abc123"), 0);
    - dwOptions : 비트 마스크로 옵션 지정

주요 함수

- 레코드셋 객체 생성

```
CRecordset::CRecordset(CDatabase* pDatabase = NULL);
```

  - pDatabase
    - 데이터베이스 객체. NULL을 사용하면 CRecordset 클래스 내부적으로 데이터베이스 객체를 생성

```
BOOL CRecordset::Open(
① UINT nOpenType = AFX_DB_USE_DEFAULT_TYPE,
② LPCTSTR lpszSQL = NULL,
③ DWORD dwOptions = none
);
```

주요 함수

- 레코드셋 객체 생성
  - nOpenType
    - 열기 형식 지정
      - CRecordset::dynaset, CRecordset::snapshot, CRecordset::dynamic, CRecordset::forwardOnly
  - lpszSQL
    - SQL SELECT문을 사용하면 조건에 맞는 레코드셋을 얻을 수 있음
  - dwOptions
    - 읽기 전용(CRecordset::readOnly), 추가 전용(CRecordset::appendOnly) 등 다양한 옵션 지정

주요 함수

- 레코드 출력
  - 현재 레코드 설정 함수

함수	기능
MoveFirst()	첫 번째 레코드를 현재 레코드로 설정한다.
MoveLast()	마지막 레코드를 현재 레코드로 설정한다.
MoveNext()	다음 위치의 레코드를 현재 레코드로 설정한다.
MovePrev()	이전 위치의 레코드를 현재 레코드로 설정한다.

주요 함수

- 레코드 출력
  - 현재 레코드 위치 판단 함수

함수	기능
IsBOF()	현재 레코드가 첫 번째 레코드 바로 전 위치로 설정되었다면 TRUE를 리턴한다. 레코드가 전혀 없는 경우에도 TRUE를 리턴한다.
IsEOF()	현재 레코드가 마지막 레코드 바로 다음 위치로 설정되었다면 TRUE를 리턴한다. 레코드가 전혀 없는 경우에도 TRUE를 리턴한다.

주요 함수

- 레코드 출력

```
void CRecordset::GetFieldValue(
    short nIndex,
    CString& strValue
);
```

  - nIndex
    - 필드를 가리키는 인덱스(0부터 시작)
  - strValue
    - 해당 필드의 데이터가 저장됨

주요 함수

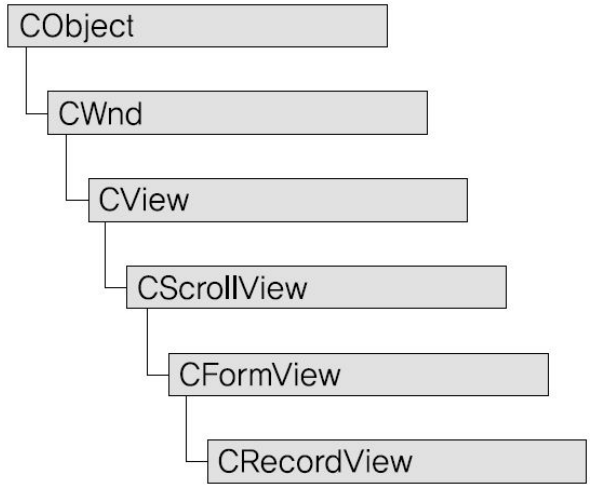
- 종료

```
rs.Open(...);
…
rs.Close();
rs.Open(...); // 새로운 레코드셋을 얻는다.
…
rs.Close();
```

실습

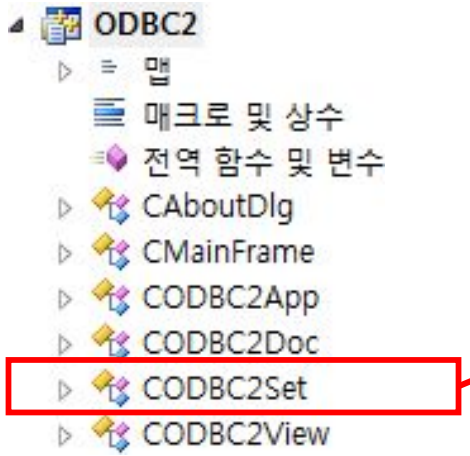
MFC ODBC 프로그래밍 - GUI

- MFC 클래스 계층도



MFC ODBC 프로그래밍 - GUI

- 클래스 구성



CRecordSet 클래스를 상속받아 만들어진 레코드셋 클래스

MFC ODBC 프로그래밍 - GUI

- 레코드셋 클래스 (1/3)

```
class CODBC2Set : public CRecordset
{
public:
    CODBC2Set(CDatabase* pDatabase = NULL);
    DECLARE_DYNAMIC(CODBC2Set)

    long m_ST_NUM;
    CStringW m_ST_NAME;
    BOOL m_ST_MAN;
    CStringW m_ST_PHONE;
public:
    virtual CString GetDefaultConnect();
    virtual CString GetDefaultSQL();
    virtual void DoFieldExchange(CFieldExchange* pFX);
...
};
```

MFC ODBC 프로그래밍 - GUI

- 레코드셋 클래스 (2/3)

```
...
❶ CODBC2Set::CODBC2Set(CDatabase* pdb)
    : CRecordset(pdb)
{
    m_ST_NUM = 0;
    m_ST_NAME = L"";
    m_ST_MAN = FALSE;
    m_ST_PHONE = L"";
    m_nFields = 4;
    m_nDefaultType = dynaset;
}

❷ CString CODBC2Set::GetDefaultConnect()
{
    return _T("DSN=student;...필수적인 내용이 아니므로 삭제 가능...");
}
```

MFC ODBC 프로그래밍 - GUI

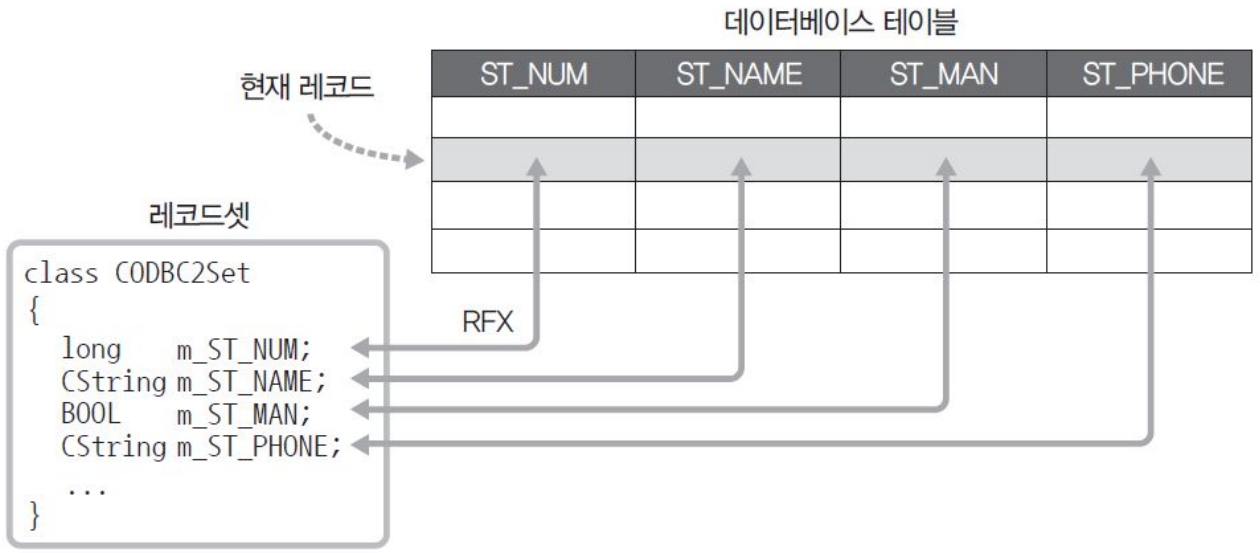
- 레코드셋 클래스 (3/3)

```
❸ CString CODBC2Set::GetDefaultSQL()
{
    return _T("[테이블1]");
}

❹ void CODBC2Set::DoFieldExchange(CFieldExchange* pFX)
{
    pFX->SetFieldType(CFieldExchange::outputColumn);
    RFX_Long(pFX, _T("[ST_NUM]"), m_ST_NUM);
    RFX_Text(pFX, _T("[ST_NAME]"), m_ST_NAME);
    RFX_Bool(pFX, _T("[ST_MAN]"), m_ST_MAN);
    RFX_Text(pFX, _T("[ST_PHONE]"), m_ST_PHONE);
}
...
```

MFC ODBC 프로그래밍 - GUI

- 레코드셋 클래스
  - RFX(Record Field eXchange)



MFC ODBC 프로그래밍 - GUI

- 도큐먼트 클래스

```
class CODBC2Doc : public CDocument
{
protected:
    CODBC2Doc();
    DECLARE_DYNCREATE(CODBC2Doc)


public:
    CODBC2Set m_ODBC2Set;
...
```

MFC ODBC 프로그래밍 - GUI

- 뷰 클래스 (1/2)

```
...
class CODBC2Set;

class CODBC2View : public CRecordView
{
protected:
    CODBC2View();
    DECLARE_DYNCREATE(CODBC2View)

public:
    enum{ IDD = IDD_ODBC2_FORM };
    CODBC2Set* m_pSet;
...
```

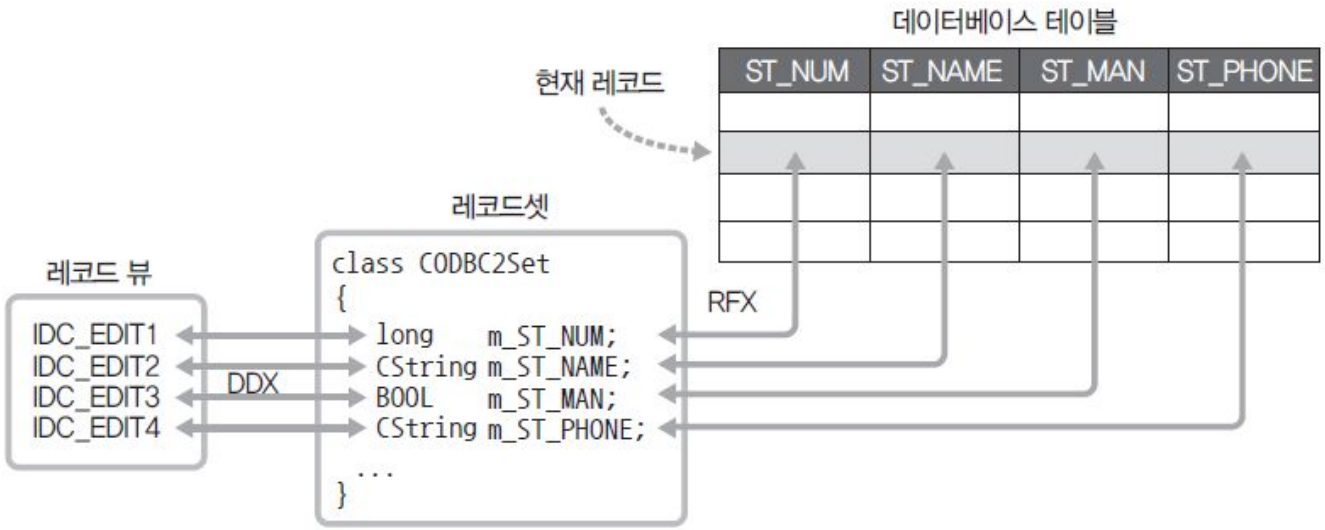
MFC ODBC 프로그래밍 - GUI

- 뷰 클래스 (2/2)

```
...
❶ void CODBC2View::OnInitialUpdate()
{
    m_pSet = &GetDocument()->m_ODBC2Set;
    CRecordView::OnInitialUpdate();
}
...
❷ CRecordset* CODBC2View::OnGetRecordset()
{
    return m_pSet;
}
...
```

데이터베이스 조작

- 레코드 표시
  - DDX와 RFX



데이터베이스 조작

- 레코드 추가

```
m_pSet->AddNew();
m_pSet->m_ST_NUM = 2016;
m_pSet->m_ST_NAME = _T(" 이순신");
m_pSet->m_ST_MAN = 0;
m_pSet->m_ST_PHONE = _T("010-9999-9999");
m_pSet->Update();
```

- 레코드 변경

```
m_pSet->Edit();
m_pSet->m_ST_PHONE = _T("010-1111-1111"); // 전화번호만 변경
m_pSet->Update();
```

데이터베이스 조작

- 레코드 삭제
  - CRecordset::Delete() 함수 호출 ⇨ 삭제가 성공하면 반드시 다른 레코드로 스크롤
- 레코드셋 갱신
  - 레코드 추가, 변경, 삭제 후 변경된 레코드 내용을 현재 레코드셋에 반영하려면 CRecordset::Requery() 함수를 호출하여 레코드셋을 새로 만듦

데이터베이스 조작

- 레코드 필터링(검색)

```
❶ SELECT * FROM 테이블1 WHERE ST_NAME =  '홍길동 ' ;
❷ m_pSet->m_strFilter = _T( " ST_NAME =  '홍길동')";
m_pSet->Requery();
```

- 레코드 정렬

```
❶ SELECT * FROM 테이블1 ORDER BY NAME;
❷ m_pSet->m_strSort = _T("NAME");
m_pSet->Requery();
```

실습

학습정리

- 데이터베이스 인터페이스에는 전용 인터페이스와 공용 인터페이스가 있습니다.
- 윈도우 데이터베이스 프로그래밍 인터페이스로는 DAO, ODBC, RDO, OLE DB, ADO가 있습니다.
- ODBC는 프로그래밍 언어와 DBMS 그리고 운영체제에 독립적으로 만든 데이터베이스 프로그래밍 인터페이스로 주요 구성요소로는 응용 프로그램, 드라이버 관리자. 드라이버, 데이터 원본을 가집니다.
- ODBC 데이터 원본은 ODBC에서 데이터베이스에 접근하기 위해 필요한 정보와 데이터베이스 자체를 총칭하는 용어로 사용자 DSN, 시스템 DSN, 파일 DSN이 있습니다.
- ODBC 핵심 클래스로는 데이터베이스와의 연결과 조작을 담당하는 CDatabase, 데이터의 조회/추가/갱신/삭제 등의 작업을 수행하는 CRecordSet, 데이터베이스 조작 오류를 나타내는 CDBException이 있습니다.
- RFX는 레코드셋과 데이터베이스 사이의 데이터 교환을 자동화하는 기법입니다.