

项目名称 Project Name		密级 Confidentiality Level
数据库管理系统		仅供收件方查阅
项目编号 Project ID	版本 Version	文档编号 Document Code
BP	1.0	Project ID_SD_003

# Database Management System Software System Design Specification

## 数据库管理系统系统设计说明书

Prepared by 拟制		Date 日期	
Reviewed by 评审人		Date 日期	
Approved by 批准		Date 日期	

# 目录

1 Introduction 简介 .....	6
1.1 Purpose 目的 .....	6
1.2 Scope 范围 .....	6
1.2.1 Name 软件名称 .....	6
1.2.2 Functions 软件功能 .....	6
1.2.3 Applications 软件应用 .....	6
2 Level 0 Design Description 第 0 层设计描述 .....	7
2.1 Software System Context Definition 软件系统上下文定义 .....	7
2.2 Design Considerations 设计思路 .....	8
2.2.1 Design Alternatives 设计方案 .....	8
2.2.2 Design Constraints 设计约束 .....	9
2.2.2.1 Standards compliance 遵循标准 .....	9
2.2.2.2 Hardware Limitations 硬件限制 .....	9
2.2.2.3 Technology Limitations 技术限制 .....	9
3 Level 1 Design Description 第 1 层设计描述 .....	9
3.1 System Architecture 系统结构 .....	9
3.2 Decomposition Description 分解描述 .....	11
3.2.1 数据库管理 .....	11
3.2.2 表管理 .....	11
3.2.3 字段管理 .....	12
3.2.4 数据管理 .....	12
3.3 Dependency Description 依赖性描述 .....	13
4 Level 2 Design Description 第二层设计描述 .....	13
4.1 创建数据库 .....	13
4.2 创建表 .....	14
4.3 添加字段 .....	15
4.4 数据管理 .....	16
4.4.1 插入记录 .....	16
4.4.2 查询记录 .....	18
5 Data Structure 数据结构/Database Design 数据库设计 .....	19
5.1 数据类型 .....	19
5.2 完整性 .....	19
5.2.1 实体完整性 .....	19
5.2.2 参照完整性 .....	19
5.2.3 自定义完整性 .....	19
5.3 数据库文件 .....	20
5.4 数据库描述文件 .....	21
5.4.1 文件名 .....	21
5.4.2 文件结构 .....	21
5.4.3 数据库结构 .....	21
5.5 表描述文件 .....	21
5.5.1 文件名称 .....	21

5.5.2 文件结构 .....	21
5.5.3 表格信息结构 .....	22
5.6 表定义文件.....	22
5.6.1 文件名称 .....	22
5.6.2 文件结构 .....	22
5.6.3 字段结构 .....	22
5.7 记录文件.....	23
5.7.1 文件名称 .....	23
5.7.2 文件结构 .....	23
5.7.3 记录结构 .....	23
5.8 完整性描述文件.....	23
5.8.1 文件名称 .....	23
5.8.2 文件结构 .....	23
5.8.3 字段结构 .....	23
5.9 索引描述文件.....	24
5.9.1 文件名称 .....	24
5.9.2 文件结构 .....	24
5.9.3 字段结构 .....	24
5.9.4 索引数据文件 .....	24
6 UI Design 界面设计 .....	25
6.1 主界面.....	25
6.2 编辑表格.....	25
6.3 查询记录.....	26
6.4 新建表.....	26
6.5 添加字段.....	27
6.6 插入记录.....	27
7 Detailed Design of Module 模块详细设计 .....	28
7.1 DataStructure。h 文件.....	28
7.1.1 Overview简介 .....	28
7.1.2 DatabaseBlock .....	28
7.1.2.1 简介 .....	28
7.1.2.2 定义 .....	28
7.1.3 TableBlock.....	28
7.1.3.1 简介 .....	28
7.1.3.2 定义 .....	28
7.1.4 FieldBlock.....	29
7.1.4.1 简介 .....	29
7.1.4.2 定义 .....	29
7.2 Global.h 文件.....	29
7.2.1 Overview简介 .....	29
7.2.2 宏 .....	29
7.3 Entity 类.....	29
7.4 View/Dialog 类 .....	30
7.4.1 CDBView.....	30

7.4.2 CTableView .....	30
7.4.3 CRecordsView .....	31
7.4.4 CNewTableDlg .....	31
7.4.5 CFieldDlg.....	31
7.4.6 CRecordDlg .....	31
7.5 Logic 类 .....	32
7.5.1 CDBLogic.....	32
7.5.2 CTableLogic .....	32
7.5.3 CrecordLogic .....	32
7.5.4 CFileLogic .....	33
7.6 Dao 类.....	33
7.6.1 CDBDao .....	33
7.6.2 CTableDao.....	33
7.6.3 CRecordDao.....	34
7.7 Helper 类.....	34
7.7.1 CFileHelper.....	34
7.7.2 CCharHelper .....	35
7.7.3 CTimeHelper .....	35
7.8 CMainFrame 类 .....	35
7.9 CRKDBMSDoc 类 .....	36
7.10 CAppException 类.....	36
8 Error Design 出错处理设计 .....	36

**Keywords 关键词:**

DBMS RDBMS MFC SDI Dialog C/S结构 MVC 完整性 数据库 表 字段 记录 二进制文件

**Abstract 摘要:**

本系统是数据库管理系统的第一个版本，包括数据库创建、数据库表的管理、记录的添加与查询等。该文档分别对各功能模块的设计进行了描述，从而使软件开发人员可以更好地分析和设计软件，同时也方便客户更好地提出意见。

**List of abbreviations 缩略语清单:**

Abbreviations 缩略语	Full spelling 英文全名	Chinese explanation 中文解释
MFC	Microsoft Foundation Classes	微软基础类库
MFC SDI	Single Document Interface	单文档界面
DBMS	Database Management System	数据库管理系统
RDBMS	Relational Database Management System	关系型数据库管理系统
MVC	Model View Controller	模型(model)—视图(view)—控制器(controller)

# 1 Introduction 简介

## 1.1 Purpose 目的

本文档对数据库管理系统概要设计和详细设计进行说明,用于指导项目组下阶段的编码实现和单元测试工作。本文档供项目组成员、客户项目代表、测试组成员、QA 等阅读。

## 1.2 Scope 范围

### 1.2.1 Name 软件名称

数据库管理系统。

### 1.2.2 Functions 软件功能

参考《软件需求规格说明书》。

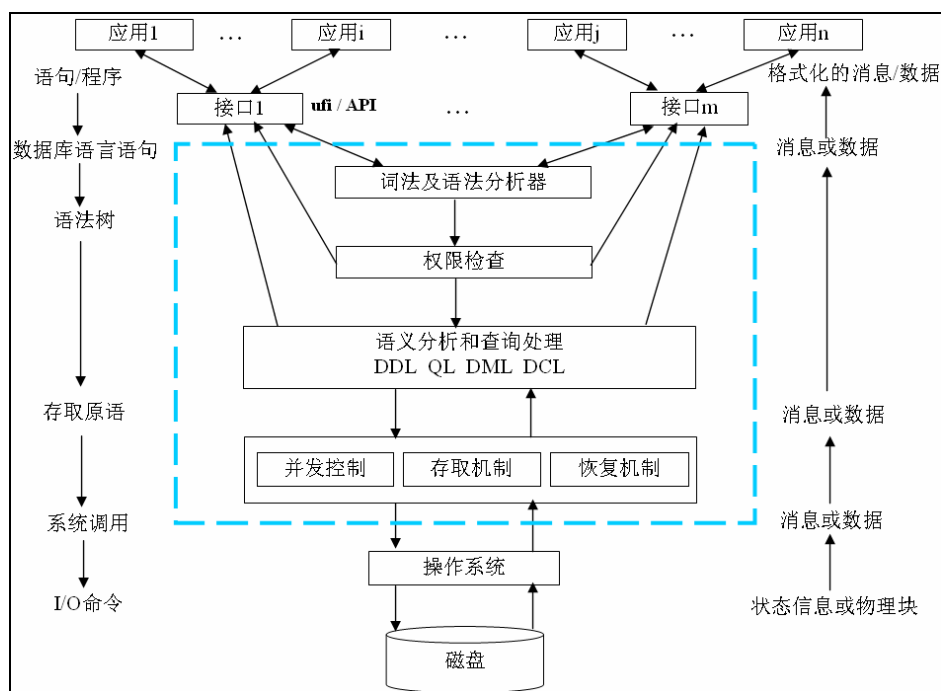
### 1.2.3 Applications 软件应用

本系统为关系数据库管理系统,数据存储使用文件,为小型的数据库系统。适合数据量不大的小型数据存储与查询。

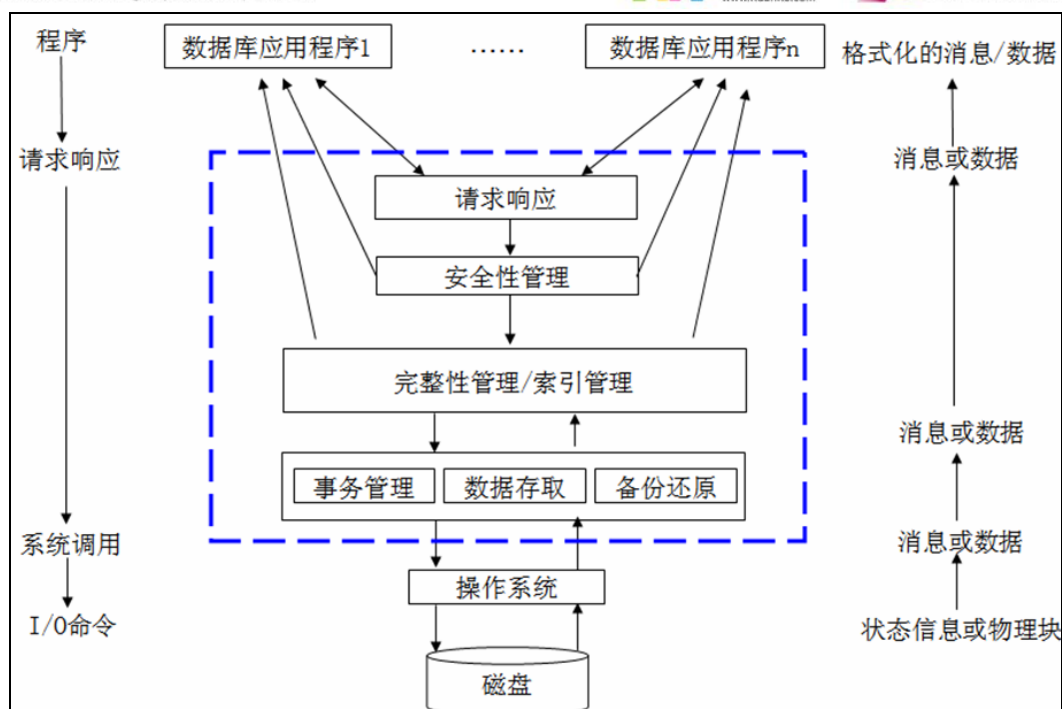
## 2 Level 0 Design Description 第 0 层设计描述

### 2.1 Software System Context Definition 软件系统上下文定义

数据库管理系统(DBMS)的体系结构包括：DBMS 体系结构、用户接口、语法分析、查询处理、目录管理、并发控制、恢复机制、物理存储管理等。本系统主要实现模拟 DBMS 中 DDL、DML、DQL 等功能。



本系统的程序结构如下图所示：

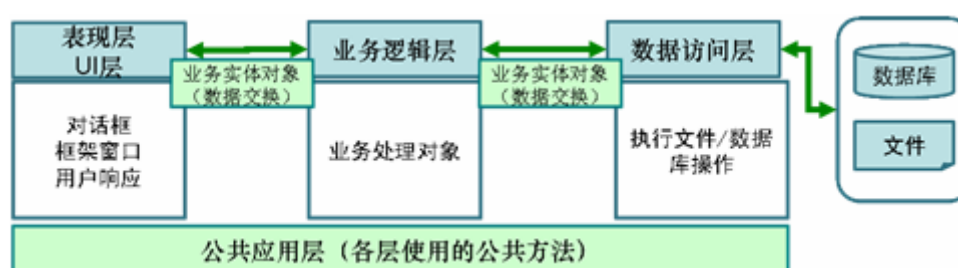


## 2.2 Design Considerations 设计思路

### 2.2.1 Design Alternatives 设计方案

#### 1、程序结构设计

本系统为 MFC SDI 单文档工程。为提高程序的可读性，可维护性，对程序进行分层，程序分为表示层，逻辑层和数据存储层。



程序软件结构可以按逻辑职责划分，分为“表示层”、“业务逻辑层”、“数据访问层”三层。各层之间使用“实体类”(数据对象) 进行数据的传递。另外，程序中各层中都可能用到且与业务无关的公共类作为“工具类”。

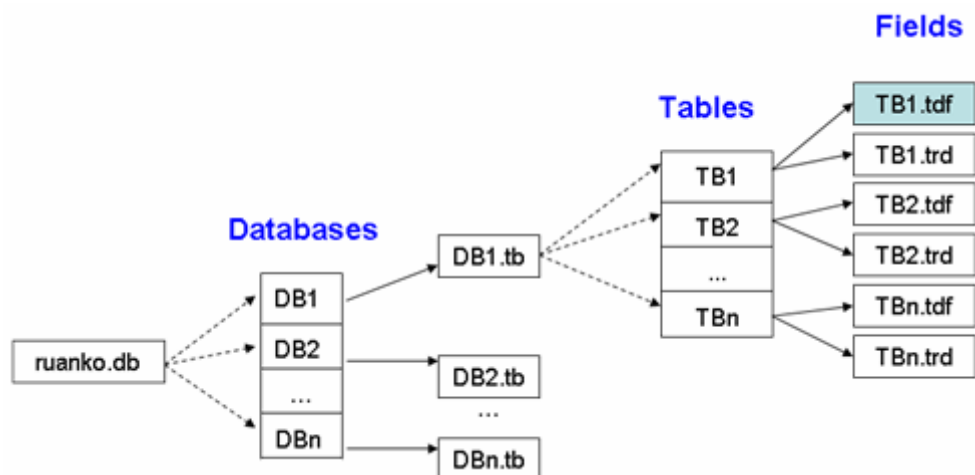
#### 2、数据存储结构

系统数据使用操作系统的二进制文件存储，以文件夹与文件来保存定义信息与数据信息。DBMS 系统定义文件包括：数据库描述文件(ruanko.db)、表描述文件(\*.tb)、表定义文



件(保存字段信息, \*.tdf)、索引描述文件(\*.tid)、完整性描述文件(\*.tic)。DBMS 的数据文件包括: 记录文件(\*.trd)、索引数据文件(\*.ix)、日志文件(\*.log)、事务数据文件(\*.tac)、临时文件(\*.tmp)等。

数据库描述文件、表描述文件、表定义文件、记录文件之间的关系图如下:



## 2.2.2 Design Constraints 设计约束

### 2.2.2.1 Standards compliance 遵循标准

本软件产品应严格遵循如下规范,不能和规范相违背,可以扩充规范中不存在的需求:《软酷卓越实验室 COE 技术要求规范》、《软酷卓越实验室 COE 编程规范要求》。

### 2.2.2.2 Hardware Limitations 硬件限制

CPU 和内存要求,最低配置,CPU 要求在 1GHZ、内存 128MB。

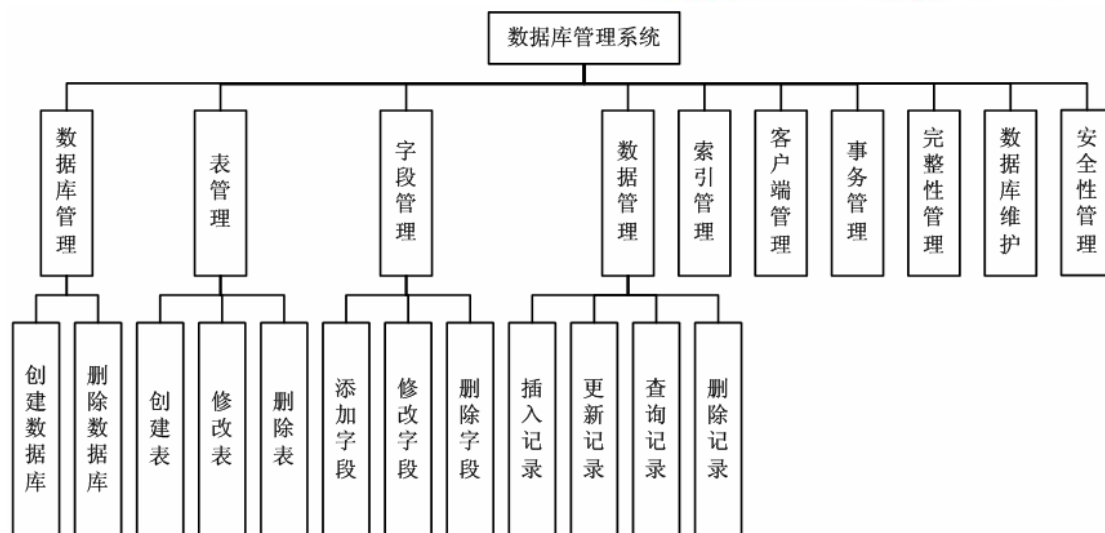
### 2.2.2.3 Technology Limitations 技术限制

编程规范: 软酷卓越实验室 COE 编码规范。

## 3 Level 1 Design Description 第 1 层设计描述

### 3.1 System Architecture 系统结构

系统按照分而治之的思想进行开发,将功能划分成若干个模块进行分别管理和开发,系统详细的模块划分如下图所示:



各模块之间使用按三层结构进行划分，如下图所示：

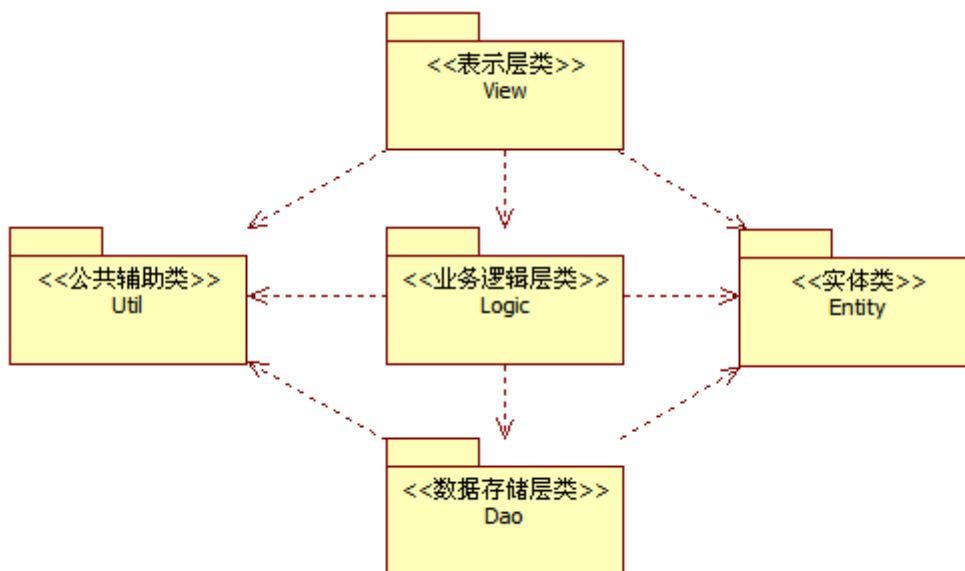
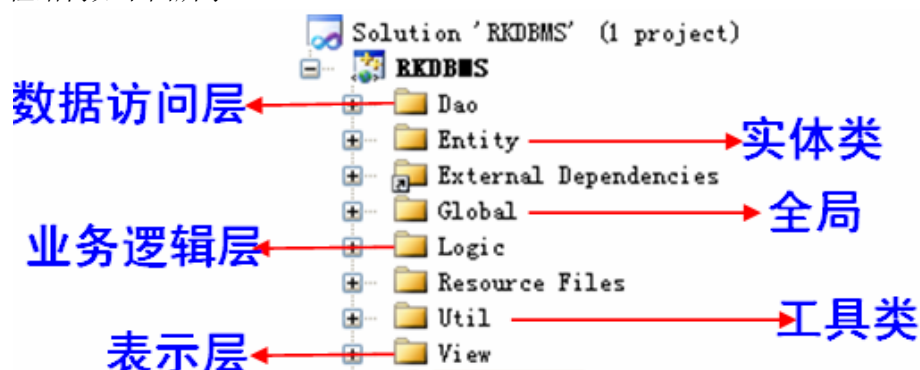


图 3-1 程序结构图

## 1、工程结构

系统为 MFC SDI 工程，工程名为：RKDBMS。

工程结构如下图所示：



## 2、工程代码结构

工程在逻辑上使用文件夹进行划分，共分为：表示层、业务逻辑层、数据存储层、实体类、公共辅助类。分别用文件夹过滤器：View、Logic、Dao、Entity、Util 来进行分隔。

层次	过滤器	类
表示层	View	框架类CMainFrame 视图类CRKDBMSView 对话框类
业务逻辑层	Logic	文档类CRKDBMSDoc 业务逻辑处理类
数据访问层	Dao	对数据文件进行读写操作的类
	Entity	实体类
	Util	工具类
	Global	全局定义的文件 应用程序类CRKDBMSApp

## 3.2 Decomposition Description 分解描述

### 3.2.1 数据库管理

#### 1、Overview 简介

完成数据库的创建与删除功能，实现数据库定义文件的创建、修改与查询。

#### 2、Functions 功能列表

模块名称	功能名称	功能描述
数据库管理	创建数据库	实现数据库的创建功能。对应 SQL 语句： CREATE DATABASE <database name>。
	删除数据库	实现数据库的删除功能。对应SQL语句：DROP DATABASE <database name>。

### 3.2.2 表管理

#### 1、Overview 简介

完成表的添加、修改与删除功能，实现表定义文件的创建、修改与查询。

#### 2、Functions 功能列表

模块名称	功能名称	功能描述
表管理	创建表	实现数据库表的创建功能，对应的 SQL 语句： CREATE TABLE <table name>。
	修改表	实现数据库表的修改功能，对应的 SQL 语句：

ALTER TABLE <table name> <alter table

		action>。
	删除表	实现数据库表的删除功能，对应的 SQL 语句： DROP TABLE <table name>。

### 3.2.3 字段管理

#### 1、Overview 简介

完成表字段的添加、修改与删除功能，实现表字段定义文件的创建、修改与查询。

#### 2、Functions 功能列表

模块名称	功能名称	功能描述
字段管理	添加字段	在已建的表中，添加字段。对应 SQL 语句： ALTER TABLE <table name> ADD COLUMN <column name> <column definition>。
	修改字段	修改表中的字段信息。对应 SQL 语句：ALTER TABLE <table name> MODIFY COLUMN <column name> <alter column action>。
	删除字段	删除表中的字段。对应 SQL 语句：ALTER TABLE <table name> DROP COLUMN <column name> <drop behavior>。

### 3.2.4 数据管理

#### 1、Overview 简介

实现数据的存储、更新、修改与查询的功能。

#### 2、Functions 功能列表

模块名称	功能名称	功能描述
数据管理	插入记录	向数据库表中插入一条记录。对应 SQL 语句： INSERT INTO <table name> <column name list> VALUES <insert value list>。
	更新记录	更新数据库表中的记录。对应的 SQL 语句为： UPDATE <table name> SET <column name> = <update value> [ WHERE <search condition> ]。
	查询记录	查询表中的全部记录。对应 SQL 语句为： SELECT * FROM <table name> [ WHERE <search condition> ]。
	删除记录	删除表中的记录。对应 SQL 语句为：DELETE FROM <table name> [ WHERE <search condition> ]。

### 3.3 Dependency Description 依赖性描述

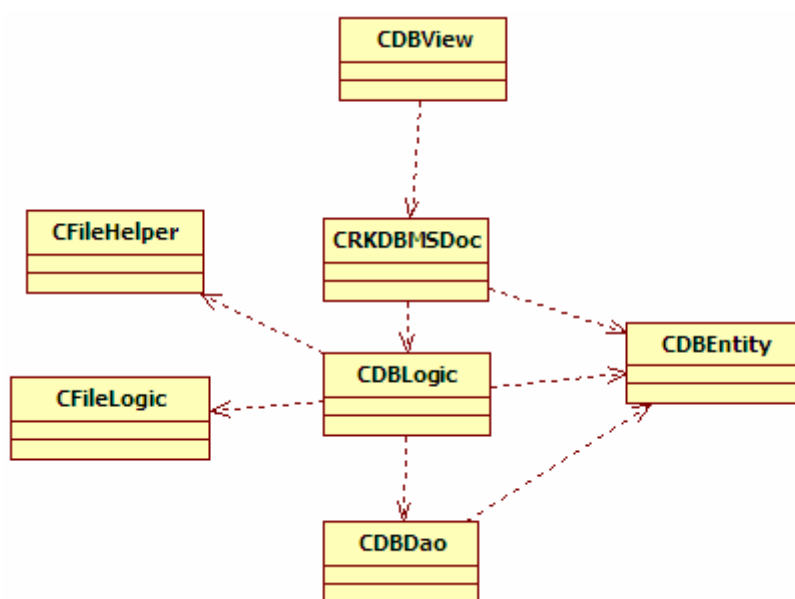
本项目是Windows窗口程序，依赖于操作系统。数据存储依赖于操作系统的文件管理系统。

## 4 Level 2 Design Description 第二层设计描述

### 4.1 创建数据库

系统默认创建一个名为“Ruanko”的数据库。数据库基本信息保存在 CDBEntity 中，使用树视图 CDBView 显示创建的数据库名。通过文档类 CRKDBMSDoc，将数据库信息传递给视图类。数据库信息通过 CRKDBMSDoc 类传递给 CDBLogic 类，CDBLogic 类将数据库信息传递给 CDBDao 类，CDBDao 类将数据库信息保存在“Ruanko.db”文件中。

类与类关系图如下：



#### 1、CDBView

显示数据库结构视图类，在树视图中显示数据库名。

#### 2、CRKDBMSDoc

文档类，设置默认数据库名，存储与读取数据，控制界面的显示与刷新。

#### 3、CDBLogic

数据库业务逻辑类，判断默认的数据库是否存在，如果不存在则创建默认数据库。

#### 4、CDBDao

数据库信息操作类，将数据库描述信息以二进制格式保存到“Ruanko.db”文件中。

#### 5、CFileLogic

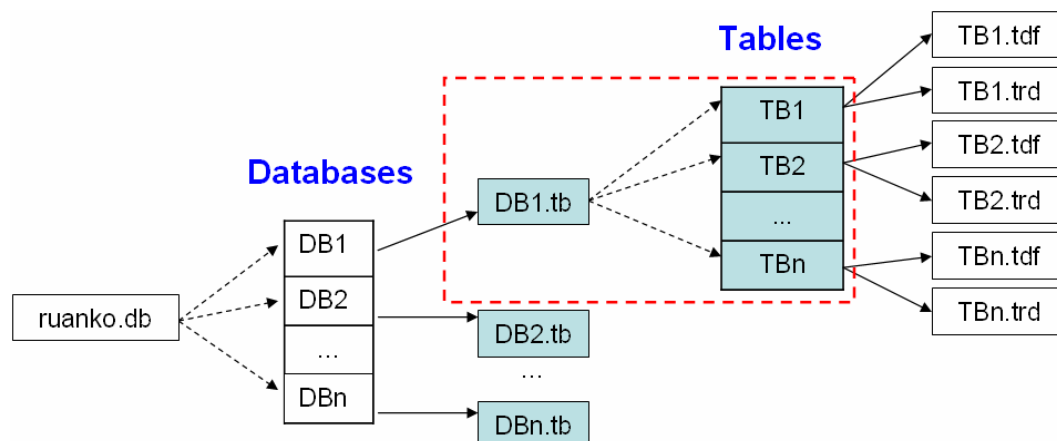
根据数据库名，获取数据库描述文件绝对路径。

## 6、CFileHleper

创建数据库描述文件路径中的文件夹和数据库描述文件。

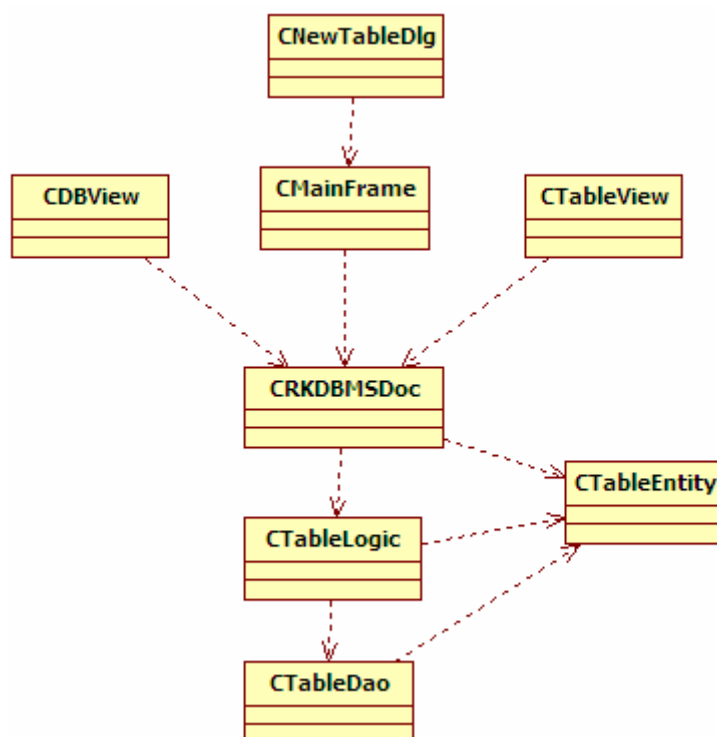
## 4.2 创建表

根据数据库名 Ruanko 创建一个数据库表描述文件 Ruanko.tb，用于保存数据库中的表格信息。通过读取表描述文件(Ruanko.db)，即可找到表结构定义文件(\*.tdf)和数据记录文件(\*.trd)。



数据库信息通过 CDBEntity 保存，表信息通过 CTableEntity 保存。通过 CMainFrame 类响应创建表菜单事件，显示创建表对话框。通过创建表 CNewTableDlg 获取输入的表名。将输入的表名传递给 CRKDBMSDoc。CRKDBMSDoc 将数据库名和表信息传递给 CTableLogic。CTableLogic 将表文件路径和表信息传递给 CTableDao。CTableDao 将表信息以二进制格式保存到文件中。保存成功之后，在 CDBView 中显示表名，在 CTableView 中显示表结构。

类与类关系图如下：



#### 1、CMainFrame

框架窗口类，显示视图，响应创建表菜单事件。

#### 2、CNewTableDlg

新建表对话框类，通过控件映射获取输入的表名。

#### 3、CDBView

显示数据库结构视图类，在树视图的数据库结点下显示表名。

#### 4、CTableView

显示表结构视图类。使用列表控件显示表结构。

#### 5、CRKDBMSDoc

文档类，存储数据库和表信息，并控制界面显示和更新。

#### 6、CTableLogic

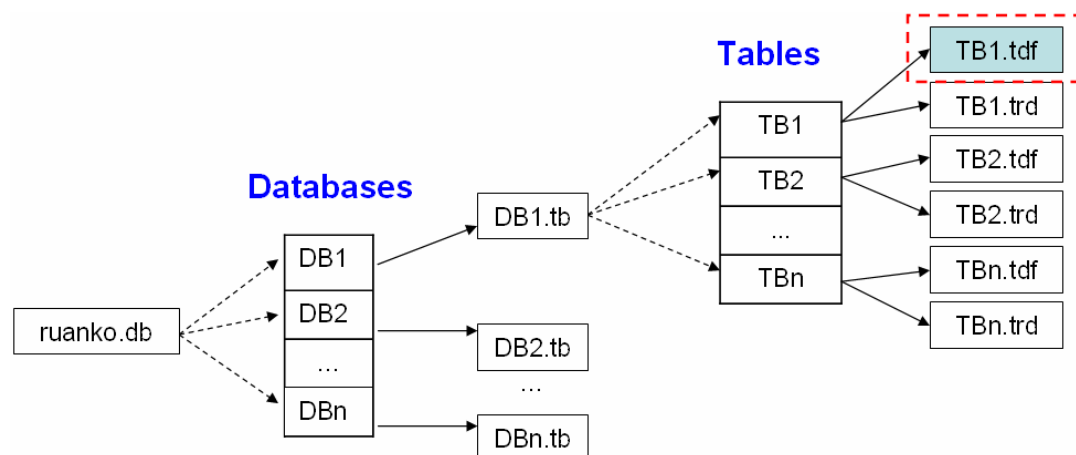
表业务逻辑类。

#### 7、CTableDao

表文件操作类，将表信息以二进制格式保存到文件中。

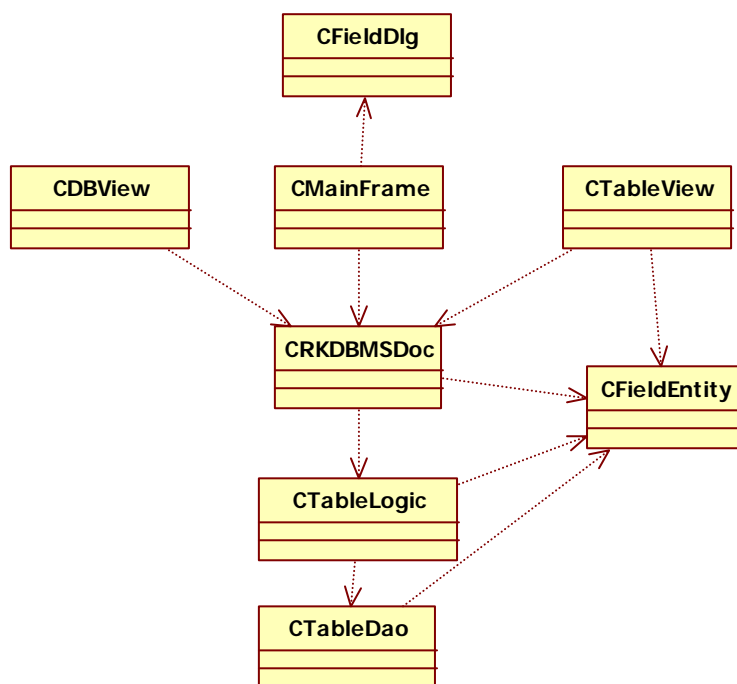
## 4.3 添加字段

根据表名创建表定义文件(\*.tdf)，保存用户定义的表结构。将文件路径保存到表描述文件中，通过读取表描述文件即可找到表定义文件(\*.tdf)。将字段信息保存到表定义文件中。



CMainFrame 类响应添加字段事件，在 CDBView 中用户选择表，在 CFieldDlg 中接收输入的字段信息，字段信息保存在 CFieldEntity 中。CRKDBMSDoc 将数据库名、表和字段传递给 CTableLogic。CTableLogic 将字段信息保存到表定义文件中。CDBView 将字段名显示在表结点的列子结点下，CTableView 显示字段信息。

类与类之间关系如下：



#### 1、CFieldDlg

添加字段对话框类，接收输入的字段信息。

#### 2、CMainFrame

框架窗口类，响应添加字段事件。

#### 3、CRKDBMSDoc

文档类，保存数据库信息、表信息和字段信息，并控制界面显示和更新。

#### 4、CDBView

数据库视图类，显示数据库中的表名以及表中的字段名。

#### 5、CTableView

表结构视图类，显示选择的表中的字段信息。

#### 6、CTableLogic

表业务逻辑类，调用 CTableDao 类方法，将字段信息保存到表定义文件(\*.tdf)中，并修改表描述文件(\*.tb)。

#### 7、CTableDao

表数据库处理类，将字段信息保存到表定义文件中。

#### 8、CFieldEntity

字段信息实体类。

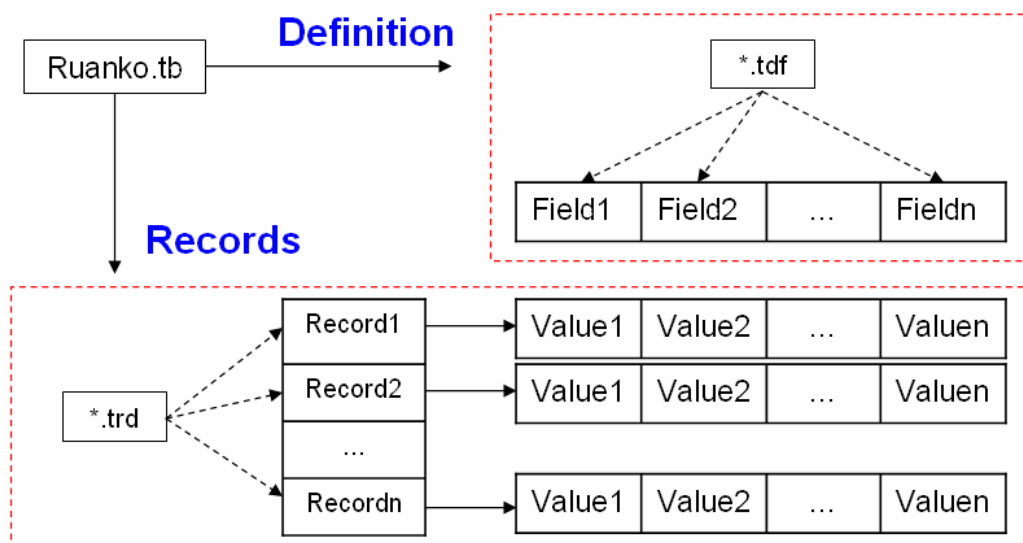
## 4.4 数据管理

数据管理模块分为插入记录和查询记录。

### 4.4.1 插入记录

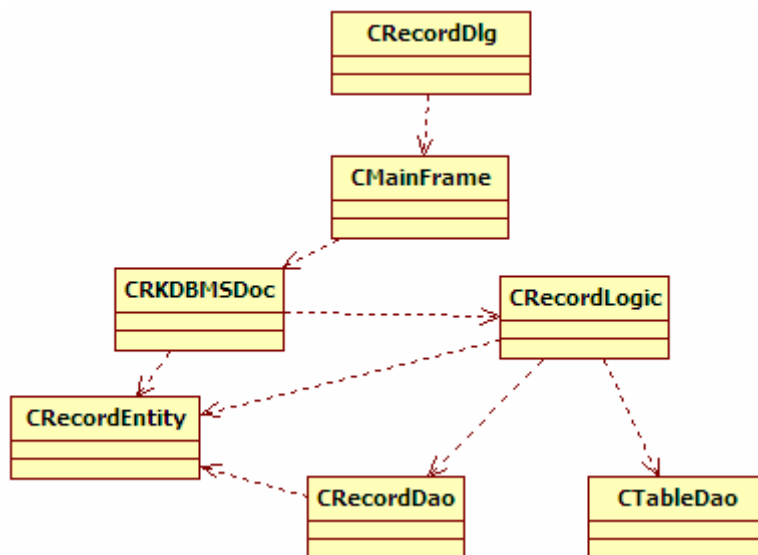


表描述文件”Ruanko.tb”中保存了表的多种文件路径：\*.tdf 文件用于保存表定义信息，\*.trd 文件用于保存记录数据。插入记录时，按表定义文件(\*.tdf)中记录的字段顺序，将记录数据保存到这个文件中。



在 CMainFrame 类响应插入记录消息响应函数，在 CRecordDlg 对话框中显示表中的字段，编辑字段的值，组装成记录信息保存在 CRecordEntity 中。并传递给 CRKDBMSDoc，CRKDBMSDoc 将数据库名、表和字段信息传递给 CRecordLogic。CRecordLogic 将记录信息传递给 CRecordDao 并调用 CTableDao 类的方法更新表描述信息。CRecordDao 类将记录信息保存到记录数据文件中。

类与类之间关系如下图所示：



#### 1、CRecordDlg

添加记录对话框类，接收输入的记录信息。

#### 2、CMainFrame

框架窗口类，响应添加记录事件。

#### 3、CRKDBMSDoc

文档类，保存数据库信息、表信息和记录信息，并控制界面显示和更新。

#### 4、CRecordLogic

记录信息业务逻辑类，调用 CRecordDao 类方法将记录信息保存到表记录文件(\*.trd)中，并调用 CTableDao 类方法更新表描述信息。

#### 5、CRecordDao

记录信息数据操作类，将记录信息保存到表记录文件中。

#### 6、CTableDao

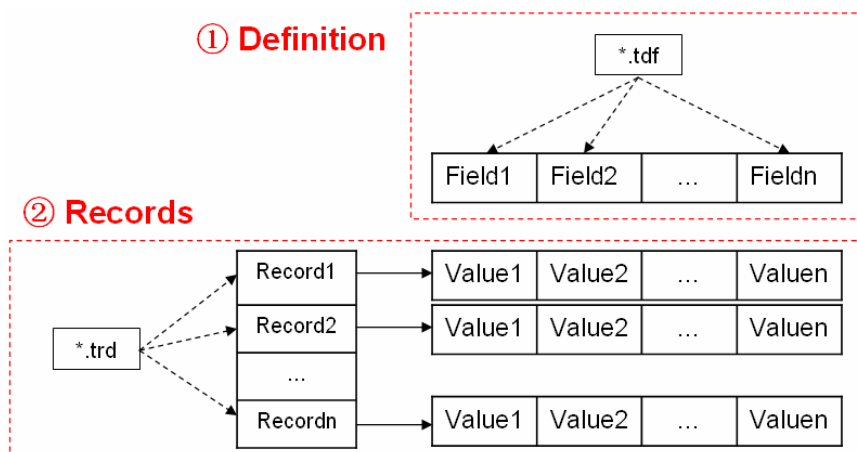
表信息操作类，更新表描述信息。

#### 7、CRecordEntity

记录信息实体类。

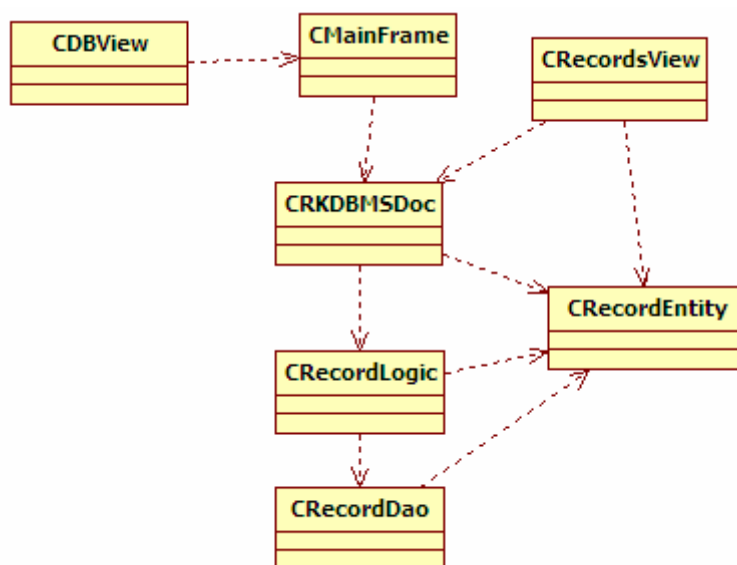
### 4.4.2 查询记录

查询记录时，先读取表定义文件(\*.tdf)查询表结构，然后根据表结构读取记录文件(\*.trd)中的数据，返回给逻辑层。



CMainFrame 类响应查询时间，在 CDBView 中获取选择的表。CRKDBMSDoc 将当前表信息传递给 CRecordLogic，CRecordLogic 将表信息传递给 CRecordDao 类。CRecordDao 读取 “\*.trd” 文件，获取记录信息，并通过 CRecordEntity 数组返回给逻辑类。最后在 CRecordsView 中显示记录信息。

类与类关系图如下所示：



#### 1、CMainFrame

框架窗口类，响应查询记录操作。

## 2、CRKDBMSDoc

文档类，保存数据库信息、表信息和记录信息，并控制界面显示和更新。

## 3、CDBView 类

数据库视图类，显示数据库中的表名和字段名。

## 4、CRecordsView

记录视图类，显示表中的记录信息。

## 5、CRecordLogic

记录信息业务逻辑类，调用 CRecordDao 方法，获取记录信息。

## 6、CRecordDao

记录信息数据操作类，读取记录文件(.trd)，获取文件中的记录信息。

## 7、CRecordEntity

记录信息实体类。

# 5 Data Structure 数据结构/Database Design 数据库设计

## 5.1 数据类型

DBMS类所类型	说明	大小	程序类型
INTEGER	整数	4byte	int
BOOL	布尔类型	1byte	bool
DOUBLE	浮点数	2byte	double
VARCHAR(n)	变长字符串，最长255个长，以\0结尾标识字符串结束	(n+1)byte	char[n+1]
DATETIME	日期时间类型	16byte	SYSTEMTIME

## 5.2 完整性

### 5.2.1 实体完整性

主键：PRIMARY KEY

### 5.2.2 参照完整性

外键：FOREIGN KEY

### 5.2.3 自定义完整性

#### 1、检查约束：CHECK

- 2、UNIQUE 约束: UNIQUE
- 3、NOT NULL 约束: NOT NULL
- 4、默认约束: DEFAULT
- 5、自增: IDENTITY

## 5.3 数据库文件

约定: DBMS\_ROOT为DBMS程序安装的根目录。本数据库所有的数据, 以二进制文件存储数据。

### 1、数据库描述文件

数据库程序安装时, 在[DBMS\_ROOT]文件夹下, 创建一个ruanko.db文件, 保存当前数据库的信息。保存数据库类型(系统数据库, 用户数据库), 数据库名称, 数据库存放路径(文件夹的路径)。

### 2、数据库(1 文件夹+2 文件)

#### (1)数据库文件夹

在创建一个数据库时, 在[DBMS\_ROOT]/data文件夹下创建一个以数据库名命名的文件夹。如: 数据库名为Ruanko, 则创建[DBMS\_ROOT]/data/Ruanko文件夹。

#### (2)表描述文件

在创建的数据库文件夹下, 创建一个与数据库同名的文件, 后缀为tb。如: 在数据库Ruanko时, 创建[DBMS\_ROOT]/data/Ruanko/Ruanko.tb文件。

#### (3)日志文件

在创建的数据库文件夹下, 创建一个与数据库同名的文件, 后缀为log。如: 在数据库Ruanko时, 创建[数据库数据文件夹]/Ruanko.log文件。

默认路径: [DBMS\_ROOT]/data/Ruanko。

### 3、表(4 个文件)

当创建一个表格时, 分别创建文件用来进行表信息保存, 表记录存储, 表索引存储。

#### (1)表定义文件

在数据库文件夹下, 创建一个与表名同名的文件, 后缀为tdf。如: 在数据库Ruanko中创建Account表格, 则创建: [DBMS\_ROOT]/data/Ruanko/Account.tdf文件。

#### (2)记录文件

在数据库文件夹下, 创建一个与表名同名文件, 后缀为trd。如: 在数据库Ruanko中创建Account表格, 则创建: [数据库数据文件夹]/Account.trd文件。

默认路径: [DBMS\_ROOT]/data/Ruanko。

#### (3)完整性描述文件

在数据库文件夹下, 创建一个与表名同名的文件, 后缀为 tic。如: 在数据库 Ruanko 中创建 Account 表格, 则创建: [DBMS\_ROOT]/data/Ruanko/Account.tic 文件。

#### (4)索引描述文件

在数据库文件夹下，创建一个与表名同名文件，后缀为tid。如：在数据库Ruanko中创建Account表格，则创建：[DBMS\_ROOT]/data/Ruanko/Account。tid文件。

## 5.4 数据库描述文件

### 5.4.1 文件名

ruanko。db

### 5.4.2 文件结构

数据库基本信息 (DatabaseBlock)	.....	.....
----------------------------	-------	-------

### 5.4.3 数据库结构

结构体成员	数据类型	说明
name	CHAR[128]	数据库名称
type	BOOL	数据库类型：系统数据库，用户数据库
filename	CHAR[256]	数据库数据文件夹全路径，保存记录文件与日志文件。
crtime	DATETIME	创建时间

## 5.5 表描述文件

### 5.5.1 文件名称

[数据库名]。tb

### 5.5.2 文件结构

表格信息 (TableBlock)	.....	.....
----------------------	-------	-------

### 5.5.3 表格信息结构

结构体成员	数据类型	说明
name	CHAR[128]	表格名称
record_num	INTERGER	记录数
field_num	INTERGER	该表字段数
tdf	CHAR[256]	表格定义文件路径
tic	CHAR[256]	表格完整性文件路径
trd	CHAR[256]	表格记录文件路径
tid	CHAR[256]	表格索引文件路径
crttime	DATETIME	创建时间
mtime	INTERGER	最后修改时间

## 5.6 表定义文件

### 5.6.1 文件名称

[表名]。tdf

### 5.6.2 文件结构

字段块FieldBlock
字段块FieldBlock
字段块FieldBlock
.....

### 5.6.3 字段结构

结构体成员	数据类型	说明
order	INTERGER	字段顺序
name	CHAR[128]	字段名称
type	INTERGER	字段类型
param	INTERGER	字段类型参数(VARCHAR\CHAR)
mtime	DATETIME	最后修改时间
integrityies	INTERGER	完整性约束信息

## 5.7 记录文件

### 5.7.1 文件名称

[表名]。trd

### 5.7.2 文件结构

记录1	记录2	.....	记录N
-----	-----	-------	-----

### 5.7.3 记录结构

- 1、在 DBMS 中，一条记录，即一组数据类型的存储。由用户自定义。
- 2、基于系统数据存储的特点，所有的块和字段大小，在存储时都调整成 4 的倍数，以提高数据的读取效率。

## 5.8 完整性描述文件

### 5.8.1 文件名称

[表名]。tic

### 5.8.2 文件结构

约束1	约束2	.....	约束N
-----	-----	-------	-----

### 5.8.3 字段结构

结构体成员	数据类型	说明
name	CHAR[128]	约束名称
field	CHAR[128]	字段名称
type	INTERGER	约束类型
param	CHAR[256]	参数

## 5.9 索引描述文件

### 5.9.1 文件名称

[表名]。tid

### 5.9.2 文件结构

索引块1 (IndexBlock)	索引块2 (IndexBlock)	索引块3 (IndexBlock)	.....
----------------------	----------------------	----------------------	-------

### 5.9.3 字段结构

结构体成员	数据类型	说明
name	CHAR[128]	索引名称
unique	BOOLE	是否唯一索引，true为唯一索引，false为非唯一索引
asc	BOOLE	排序方式：true为升序asc，false为降序
field_num	INTEGER	字段数，最多可以保存2个
fields	CHAR[128][2]	字段值，最多可以保存2个
record_file	CHAR[256]	索引对应记录文件的路径
index_file	CHAR[256]	索引数据文件的路径

### 5.9.4 索引数据文件

文件名：[索引名]。ix

文件夹：表格文件夹

索引名：[字段名]Index








## 6.3 查询记录



## 6.4 新建表



新建表

请输入表名：

OK Cancel

## 6.5 添加字段

The screenshot shows a 'Field Properties' dialog box with a blue title bar containing the text '字段属性' and a red close button. The dialog has two main sections: '常规' (General) and '约束条件' (Constraints). In the '常规' section, there are three input fields: '名称:' (Name), '类型:' (Type) which includes a dropdown arrow, and '默认值:' (Default Value). The '约束条件' section contains two checkboxes: '主键' (Primary Key) and '不为空' (Not Null), both of which are currently unchecked. At the bottom of the dialog are two buttons labeled 'OK' and 'Cancel'.

## 6.6 插入记录

[illegible]

## 7 Detailed Design of Module 模块详细设计

### 7.1 DataStructure。h文件

#### 7.1.1 Overview简介

在该文件中，统一定义程序中的数据类型。

#### 7.1.2 DatabaseBlock

##### 7.1.2.1 简介

数据库信息块。

##### 7.1.2.2 定义

```
struct DatabaseBlock{  
    BOOLE type;           // 数据库类型： true系统数据库， false用户自定义数据库  
    VARCHAR name;         // 数据库名称  
    VARCHAR filepath;     // 数据库文件路径  
    DATETIME crtime;      // 创建时间  
};
```

#### 7.1.3 TableBlock

##### 7.1.3.1 简介

表信息块。

##### 7.1.3.2 定义

```
struct TableBlock  
{  
    VARCHAR name;         // 名称  
    INTEGER record_num;   // 记录数  
    INTEGER field_num;    // 字段数  
    VARCHAR tdf;          // 表定义文件路径  
    VARCHAR trd;          // 记录文件路径  
    DATETIME crtime;      // 表格创建时间  
    DATETIME mtime;       // 最后修改时间  
};
```

## 7.1.4 FieldBlock

### 7.1.4.1 简介

字段信息块。

### 7.1.4.2 定义

```
struct FieldBlock
{
    VARCHAR name;      // 名称
    INTEGER type;      // 类型
    INTEGER param;     // 参数
    DATETIME mtime;    // 最后修改时间
    INTEGER integrities; // 完整性约束信息
};
```

## 7.2 Global.h文件

### 7.2.1 Overview简介

全局宏定义文件。

### 7.2.2 宏

宏名	值	描述
UPDATE_OPEN_DATABASE	0x01	打开数据库消息
UPDATE_CREATE_TABLE	0x02	创建表消息
UPDATE_EDIT_TABLE	0x03	修改表消息
UPDATE_ADD_FIELD	0x04	添加字段消息
UPDATE_OPEN_TABLE	0x05	查询记录消息
UPDATE_INSERT_RECORD	0x06	插入记录消息
MENU_DATABASE	1	数据库菜单编号
MENU_TABLE	2	表菜单编号
MENU_FIELD	3	字段菜单编号
MENU_RECORD	4	记录菜单编号
MENU_OTHER	-1	其他菜单编号

## 7.3 Entity类

实体类用于层之间传递数据，根据系统数据信息，每个实体对象表示一类数据。

在设计实体类时，类主要包括三部分内容：

- 1、数据成员：每一个实体类的私有数据成员，和数据库表的字段对应。
  - 2、GetXxx()和 SetXxx()方法：访问权限为 public，用户访问类中的私有数据成员，并分别建立与私有数据成员一一对应的 GetXxx()和 SetXxx()方法。
    - (1) GetXxx()方法：用户获取私有数据成员的值，获取的值通过 GetXxx()方法的返回值返回，GetXxx()的类型和相应的数据成员的类型一致。
    - (2) SetXxx()方法：用户给私有数据成员赋值。
  - 3、无参构造函数：用于初始化私有数据成员，给数据成员赋初值。因为实体类在各层之间进行传递，如果成员变量不初始化，那么未赋值的数据成员在传递时，将出现 null 值。
- 本系统中涉及到的实体类如下：

类名	说明
CDBEntity	数据库实体类
CTableEntity	表信息实体类
CFiledEntity	字段信息实体类
CRecordEntity	记录信息实体类

## 7.4 View/Dialog 类

### 7.4.1 CDBView

#### 1、简介

显示数据库结构视图类，继承于 CTreeView。

#### 2、核心方法

方法原型	功能
virtual void OnInitialUpdate()	视图初始化函数，设置列表控件的样式。
virtual void OnUpdate(CView* pSender, LPARAM lHint, CObject* pHint)	视图更新函数。根据接收到的信息，将信息显示在树控件中。
HTREEITEM AddTableNode(CTableEntity* pTable)	在树控件中添加表结点。
HTREEITEM AddFieldNode(CFieldEntity* pField, HTREEITEM hTableItem)	在树控件中添加字段结点。
HTREEITEM GetTableItem(CString strTableName)	获取表结点数据。

### 7.4.2 CTableView

#### 1、简介

表结构视图类，显示表的结构，继承于 CListView。

## 2、核心方法

方法原型	功能
virtual void OnInitialUpdate()	视图初始化函数，设置列表控件的样式。
virtual void OnUpdate(CView* pSender, LPARAM lHint, CObject* pHint)	视图更新函数。根据接收到的表信息，将表显示在列表中。

## 7.4.3 CRecordsView

### 1、简介

显示记录视图，继承于 CListView。

### 2、核心方法

方法原型	功能
virtual void OnInitialUpdate()	视图初始化函数，设置列表控件的样式。
virtual void OnUpdate(CView* pSender, LPARAM lHint, CObject* pHint)	视图更新函数。
void AddRecord(CRecordEntity* pRecord)	将记录信息显示在视图的列表控件中。

## 7.4.4 CNewTableDlg

新建表对话框类。获取输入的表名。

## 7.4.5 CFieldDlg

### 1、简介

添加字段对话框类。

### 2、核心方法

方法原型	功能
virtual BOOL OnInitDialog()	对话框初始化函数，设置列表控件的样式和表头。
afx_msg void OnBnClickedOk()	OK按钮BN_CLICKED消息响应函数，将输入的字段信息，保存到字段实体对象中

## 7.4.6 CRecordDlg

### 1、简介

添加记录对话框类。

## 2、核心方法

方法原型	功能
virtual BOOL OnInitDialog()	对话框初始化函数，将表中的字段名和类型显示在列表中。
afx_msg void OnClickListData(NMHDR *pNMHDR, LRESULT *pResult)	列表控件单击事件响应函数，编辑字段的值。
afx_msg void OnKillfocusEditValue()	编辑控件失去焦点消息响应函数，将编辑控件的值赋给列表控件。

## 7.5 Logic 类

### 7.5.1 CDBLogic

#### 1、简介

数据库业务逻辑类。

#### 2、核心方法

方法原型	功能
bool CreateDatabase(CDBEntity &db)	创建数据库
bool GetDatabase(CDBEntity &db)	查询数据库

### 7.5.2 CTableLogic

#### 1、简介

表格业务逻辑类。

#### 2、核心方法

方法原型	功能
bool CreateTable(const CString strDBName, CTableEntity &te)	创建表
bool AddField(const CString strDBName, CTableEntity &te, CFieldEntity &fe)	添加字段
int GetTables(const CString strDBName, TABLEARR &arrTables)	获取表信息

### 7.5.3 CrecordLogic

#### 1、简介



记录数据业务逻辑类。

## 2、核心方法

方法原型	功能
bool Insert(const CString strDBName, CTableEntity &te, CRecordEntity &re)	插入记录
bool SelectAll(CTableEntity &te, RECORDARR &data)	查询所有记录

## 7.5.4 CFileLogic

### 1、简介

文件处理逻辑类。

### 2、核心方法

方法原型	功能
CString GetDBFile(void)	获取数据库描述文件路径
CString GetDBFolder(const CString strDBName)	获取数据库文件夹路径
CString GetTableFile(const CString strDBName)	获取数据库表描述文件路径
CString GetTbDefineFile(const CString strDBName, const CString strTableName)	获取表定义文件路径
CString GetTbRecordFile(const CString strDBName, const CString strTableName)	获取记录文件路径
CString GetAbsolutePath(const CString strRelativePath)	将相对路径改成绝对路径。

## 7.6 Dao 类

### 7.6.1 CDBDao

#### 1、简介

数据库数据操作类。

#### 2、核心方法

方法原型	功能
bool Create(const CString strFilepath, CDBEntity db, bool bAppend = true)	将数据库信息保存到文件中
bool GetDatabase(const CString strFilepath, CDBEntity &db)	从文件中获取数据库信息

### 7.6.2 CTableDao

## 1、简介

表数据操作类。

## 2、核心方法

方法原型	功能
bool Create(const CString strFilePath, CTableEntity &te)	创建表，保存表信息。
bool AddField(const CString strFilePath, CFieldEntity &fe)	添加字段，将字段信息保存到文件中。
int GetTables(const CString strFilepath, TABLEARR &arr)	获取文件中的表信息。
bool GetFields(const CString strFilepath, CTableEntity &te)	获取文件中的字段信息。
bool AlterTable(const CString strFilePath, CTableEntity &te)	修改表结构。

## 7.6.3 CRecordDao

### 1、简介

记录数据操作类。

### 2、核心方法

方法原型	功能
bool Insert(CTableEntity &te, CRecordEntity &re)	插入记录
int SelectAll(CTableEntity &te, RECORDARR &data)	查询所有记录
bool Write(CFile &file, CTableEntity &te, CRecordEntity &re)	将记录数据保存到记录文件中
bool Read(CFile &file, CTableEntity &te, CRecordEntity &re)	从记录文件中，读取所有记录

## 7.7 Helper 类

### 7.7.1 CFileHelper

#### 1、简介

文件处理工具类。

#### 2、核心方法

方法原型	功能
static bool CreateFile(const CString strFileName)	创建文件，且当文件路径中的文件夹不存在时会创建文件夹
static bool IsValidFile(const CString strPath)	判断文件路径的有效性

## 7.7.2 CCharHelper

### 1、简介

字符处理工具类。

### 2、核心方法

方法原型	功能
static void ToChars(char* pDim, CString strSrc, const int nSize)	将CString类型转换为char*类型
static void ToChars(char* pDim, SYSTEMTIME tSrc, const int nSize)	将SYSTEMTIME类型数据转换为char*类型
static void ToChars(char* pDim, bool bVal, const int nSize)	将bool类型数据转换为char*类型
static void ToChars(char* pDim, int nVal, const int nSize)	将int类型数据转换为char*类型
static void ToChars(char* pDim, double dbVal, const int nSize)	将double类型数据转换为char*类型
static CString ToString(char* pSrc, const int nSize)	将char*类型数据转换为CString类型
static void Copy(char* pDim, char* pSrc, const int nSize)	字符串拷贝函数

## 7.7.3 CTimeHelper

### 1、简介

时间处理工具类。

### 2、核心方法

方法原型	功能
static SYSTEMTIME ToSystemTime(CString strTime)	将 CString 类型转换为 SYSTEMTIME类型
static SYSTEMTIME ToSystemTime(CTime t)	将 CTime 类型转换为 SYSTEMTIME类型
static CTime ToCTime(SYSTEMTIME st)	将 SYSTEMTIME 类型转换为 CTime类型

## 7.8 CMainFrame 类

## 1、简介

框架窗口类，提供应用程序的窗口。

## 2、核心方法

方法原型	功能
virtual BOOL OnCreateClient ( LPCREATESTRUCT lpcs, CCreateContext* pContext )	实现拆分窗口功能
void SwitchView(int nViewType)	切换视图

# 7.9 CRKDBMSDoc 类

## 1、简介

文档类，实现程序中与界面相关的逻辑操作，以及对数据进行维护和管理。

## 2、核心方法

方法原型	功能
BOOL OnNewDocument()	新建文档，创建默认数据库。
CDBEntity GetDBEntity()	获取数据库
CTableEntity* CreateTable(CString strName)	创建表
CFieldEntity* AddField(CFieldEntity &field)	添加字段
void LoadTables(void)	加载表
CRecordEntity* InsertRecord(CRecordEntity &record)	插入记录
void LoadRecord(void)	查询所有记录

# 7.10 CAppException 类

## 1、简介

自定义异常类，用于统一处理程序中的异常。

## 2、核心方法

方法原型	功能
CAppException(CString strError)	构造函数，通过异常信息，构造一个CAppException对象。

# 8 Error Design 出错处理设计

为程序定义统一的异常类 CAppException，采用 try, catch, throw 进行异常处理。当底层出现异常，将异常类抛到上一层，并包含异常信息。异常结构如下图所示：

