第3章 数 据 库

数据库是为了满足多个用户的多种应用的需要，按照一定的数据模型在计算机系统中组织、存储和使用的互相联系的数据集合。例如，在电子商务网站中，需要存储和管理的客户信息、订单信息、产品信息等数据，在业务处理过程中是一组相互关联的数据，可以保存在一个数据库中；而与学校管理相关的学生信息、成绩信息、课程信息等数据，是另一组相互关联的数据，可以保存在另一个数据库中。

在 SQL Server 中，数据库所包含的内容不只是数据，还包括与数据管理和操作相关的各种对象，如数据库关系图、表、视图、同义词、可编程性、Service Broker、存储、安全性等对象，而通常所说的数据则保存在其中的一个数据库对象——数据表中。由此可见，数据库这一概念在 SQL Server 中已成为一个存储数据库对象的容器。作为一种数据库管理的系统软件，数据库对 SQL Server 来说，不仅是管理的对象，也是系统运行的基础。了解和掌握数据库管理，是深入学习 SQL Server 应用的基础。本章介绍 SQL Server 2019 中数据库的应用和管理，在后续章节中将详细介绍数据库中的各种对象。

【本章要点】

- SQL Server 2019 系统数据库。
- 数据库文件与文件组。
- 创建数据库。
- 设置数据库选项。
- 管理数据库。
- 数据库快照。

3.1 SQL Server 2019 的系统数据库

在 SQL Server 2019 中数据库分为两大类：系统数据库和用户数据库。系统数据库用于保存系统运行所需的各种数据，包括用户数据库信息和其他系统性信息。用户数据库是由用户创建的，用于保存某些特定信息的数据库。

系统数据库由 SQL Server 系统预设。在 SQL Server 2019 安装完成后，就默认创建了 5 个系统数据库：master、model、msdb、tempdb 和 resource。这 5 个系统数据库是系统运行的基础，在 SQL Server 系统中起着非常重要的作用。

3.1.1 master

master 数据库是 SQL Server 系统中最重要的系统数据库，记录了 SQL Server 系统运行所需的系统信息。这些系统信息包括：

(1) 所有登录名和用户 ID 及所属角色。

(2) 所有的系统配置信息(如数据排序规则、安全规则等)。

(3) 服务器中其他系统数据和用户数据等信息，如数据库的名称、数据库文件的物理位置等。

(4) SQL Server 的初始化信息。

(5) 各种特殊的系统表，如存储缓存使用规则、可用字符集、可用语言列表、系统错误和警告信息等的数据表。

如果 master 数据库出现错误，就会导致 SQL Server 无法启动或运行错误。因此，手工改动 master 数据库是一种不明智的行为。

3.1.2 model

model 数据库是模板数据库。在 SQL Server 中创建用户数据库时，都会以 model 数据库为模板，创建拥有相同对象和结构的数据库。

如果修改 model 数据库，之后创建的所有数据库都将继承这些修改。因此，如果希望新创建的数据库都具有相同的特性，如希望在所有新创建的数据库中都建有某个相同的数据表，那么可以预先把这个表建在 model 数据库中。

另外，model 数据库也是另一个系统数据库 tempdb 的模板，对 model 数据库的改动也会反映在 tempdb 数据库中。因此，对 model 数据库的更改也需要十分小心，避免造成不必要的麻烦。而且，由于 SQL Server 服务每次重启时，都会重建 tempdb 数据库，因此，要求 model 数据库必须始终存在于 SQL Server 系统中。

3.1.3 msdb

msdb 数据库是存储代理服务信息的数据库。

在 SQL Server 2019 中，代理服务(SQL Server Agent)可以代替用户完成一些事先预定义的作业任务，如在每晚 00:00:00 自动执行数据库的备份操作。SQL Server 代理服务运行所需的作业信息，如作业运行的时间、频率、操作步骤、警报等信息都保存在 msdb 数据库中。

因此，没有特殊原因也要尽量避免手工修改 msdb 数据库。

3.1.4 tempdb

tempdb 是一个临时数据库。每次 SQL Server 服务重新启动时，会重建 tempdb 数据库；在 SQL Server 服务停止或关闭时，tempdb 数据库中的数据会丢失。

tempdb 数据库用于保存 SQL Server 运行过程中产生的需要临时存储的数据。例如，SQL Server 在执行数据查询时，有时需要临时存放查询的结果以备后续使用，这些查询结果可以存放在 tempdb 数据库中。用户在使用过程中也可能需要创建临时表，这些临时表

也会存放在 tempdb 中。虽然 tempdb 提供的是临时存放的功能，但在 SQL Server 服务没有停止或者重新启动前，且存放的临时数据没有明确要求删除时，这些临时数据还是会一直存在，直到明确删除或者 SQL Server 服务发生变动。有关临时表的特性将在第 4 章中介绍。

由此可见，tempdb 也是 SQL Server 中非常重要的一个数据库，没有特殊需要，也不应手工修改。如果一旦发生错误，可通过重启 SQL Server 服务来重建一个空的 tempdb 数据库。

3.1.5 Resource

Resource 是自 SQL Server 2005 版起新增的一个系统数据库。在 SQL Server 2005 版以前，所有可执行的系统对象都存储在 master 数据库中。这些可执行系统对象是指不存储数据的系统对象，包括：系统存储过程、系统视图、系统内置函数、系统触发器等。

但是 Resource 在 SQL Server 中对用户是不可见的，用户不能直接操作该数据库。这也是用户无法在 SQL Server Management Studio 中找到这个系统数据库的原因。之所以将原先存放于 master 数据库的可执行对象迁移到新数据库中，并不让用户直接访问的原因是为了提高系统的安全性。改存到 Resource 数据库之后的好处还在于：一方面 Resource 数据库是只读的，可以避免被误修改；另一方面，可执行对象统一存放在 Resource 数据库，便于管理和升级。

事实上，这些可执行对象除了限制直接访问外，并不禁止用户使用。只是用户需要通过 sys 架构才能调用这些系统可执行对象。通过 sys 架构，用户可以在当前的数据库中（如某个用户数据库）来引用存储在 Resource 数据库中的对象，如通过“select * from sys.objects”来引用存放在 Resource 数据库中的“objects”对象。在 SQL Server 2019 中，Resource 数据库文件保存在安装路径的 Binn 文件夹中，文件名为 mssqlsystemresource.mdf 和 mssqlsystemresource.ldf。

提示：

（1）ReportServer 数据库，是一个用于保存 SQL Server 报表服务信息的数据库。因此，在安装 SQL Server Reporting Services 后，就会产生此数据库；且这个数据库只可用于给定的 Reporting Server 实例，也只能通过 Reporting Server 修改和访问。

（2）ReportServerTempDB 数据库。与 ReportServer 数据库类似，ReportServerTempDB 数据库也是一个由安装 SQL Server Reporting Services 产生，且只能由 Reporting Server 修改和访问的数据库。ReportServerTempDB 数据库主要用于存储 Reporting Services 运行过程中的非持久化数据。

（3）示例数据库。另外，为了便于用户学习使用 SQL Server，系统还提供了两个学习的示例数据库：AdventureWorks 和 AdventureWorksDW。AdventureWorks 是一个基于虚拟的主营金属和复合材料自行车生产的大型跨国公司业务经营的数据库，AdventureWorksDW 是为展示 SQL Server 2019 数据仓库、报表服务、数据分析服务等新特性提供的示例数据库。对应的示例数据库版本有 AdventureWorks 2017/2016/2014/2012 等。AdventureWorks 和 AdventureWorksDW 并未在 SQL Server 2019 的默认安装过程中自动安装，如果用户需要，可以到以下地址下载并安装：https://docs.microsoft.com/zh-cn/sql/samples/adventureworks-install-configure?view=sql-server-ver15。

3.2 数据库文件及文件组

虽然 SQL Server 中的数据与其他数据一样，都是保存在计算机硬盘中的。但是由于 SQL Server 所保存的数据量往往非常大的，如一些超大型数据库可以达到 TB 级以上；所面对的用户数量也可能非常大，如一些大型门户网站、金融服务系统等，并发用户可达数万、数十万，甚至更多。因此，SQL Server 为保证系统有较高的响应性能和管理效率，在数据存储方面采取了一些特殊的方式。

3.2.1 SQL Server 数据存储原理

1. 数据存取过程

在计算机系统中，内存和外存是两种最基本的存储设备。一般内存执行速度快，但容量小，在停电状态下不易保存数据；而外存存储容量大，但速度慢，可以在停电状态下继续保存数据。因此，在一般应用系统中，往往采用两者协同工作的方式来处理和存储数据。如图 3-1 所示是应用系统常见的数据存储和处理方式。

从图 3-1 中可见，一般的应用系统会把存于外存（一般是硬盘）中的数据读取到内存，在内存中数据处理（如修改、删除等操作）完毕后，再把数据保存回硬盘的数据文件中。新增数据也是先到内存，然后再保存到硬盘。这种对数据的处理和存储方式虽然简单，但是存在不少问题，尤其是对数据操作的可靠性比较差。例如，在内存中修改完数据后，如果在保存过程中出现系统错误或者掉电等情况，此项修改操作的数据就可能丢失。另外，由于数据只保存在数据文件中，在出现数据文件被破坏时，很可能导致数据全部丢失。

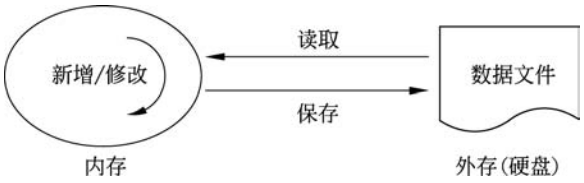

图 3-1 一般应用程序的数据存取过程

SQL Server 为提高数据存储的可靠性，采用了优先写日志的方式。即在 SQL Server 中存储数据的文件除了数据文件外，还增加了事务日志文件。数据文件用于保存数据，日志文件用于保存各种操作事务，如修改、新增数据的事务。SQL Server 存取数据的过程如图 3-2 所示。

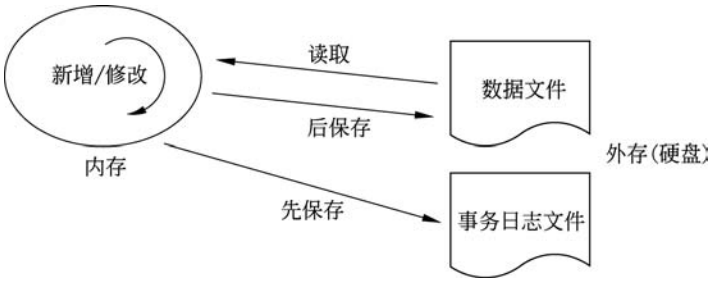

图 3-2 SQL Server 的数据存取过程

首先，SQL Server 在新增数据时，会先在内存中处理好新增的数据，然后在写入硬盘数据文件之前，先把该项操作保存到事务日志文件中，之后才把数据写入数据文件。

如果是修改数据，SQL Server 会先从硬盘数据文件中把待修改的数据读取到内存中，在内存中修改处理完毕后，再把处理操作作为一项事务保存到事务日志文件中；最后才把内存中的修改结果保存到数据文件。

SQL Server 这种优先写事务日志的数据存储方式，对维护数据的可靠性有很多好处。由于数据操作的过程事先存储在事务日志文件中，实际是对数据的每一项操作新增了一个备份，一旦数据出现意外问题，这种存储方式可以提供恢复的基础。其次，由于事务日志文件保存的内容在某些场合数据量会小于数据文件的数据量，从而为 SQL Server 提供了一种只需备份少量事务日志的数据库备份方式。相关内容将在第 11 章中进行介绍。

2. 存储空间分配

在 Windows 系统中，如果有文件需要保存时，操作系统会寻找硬盘中可用的空间分配给文件进行保存。虽然 Windows 操作系统在硬盘空间的分配上具有一定的智能性，但是往往无法解决大文件对空间位置的最优分配问题。因此，会经常出现一些大文件被保存在硬盘多个不同位置的情况，即所谓的文件碎片。文件碎片多了，硬盘驱动器读写数据的效率就会降低，这也就是需要对磁盘碎片进行经常性整理的原因。SQL Server 为防止碎片过多，同时也为了兼顾存取效率，在存储空间分配中使用了较小的数据存储单元，即页和盘区。

页是 SQL Server 数据文件存储的最小单位，页的大小为 8192B，即 8KB。其中，96B 用于保存头部信息，记录此页的相关信息，另外在页尾存储用于记录数据行位置的行偏移和其他一些信息。因此，一页实际可保存的数据量为 8060B。根据页保存数据类型的不同，页可以划分为：数据页、全局分配图页、索引页、索引分配图页、页面自由空间页和文本/图像页。

(1) 数据页。用于保存 SQL Server 中除文本和图像之外的各种数据。

(2) 全局分配图页(GAM)。由于 SQL Server 分配空间时，不是以单个页为单元进行分配，而是采取每次 8 页的量进行分配，因此有可能会存在已分配但尚未使用的页。“全局分配图页”可用于跟踪已分配且可用的页。

(3) 索引页。在索引数据与记录数据分开保存的场合(如非聚集索引)，索引页用于保存索引数据。

(4) 索引分配图页。跟踪已分配且可用的索引页。

(5) 页面自由空间页(Page Free Space)：是一种特殊的用于跟踪数据库中其他所有页面上可用空间的页。

(6) 文本/图像页。由于 SQL Server 中可能会保存大型文档和图像数据，这些数据量较大，如果与其他数据一起存放在相同的数据页中，会降低存取效率。因此，SQL Server 使用专门的文本/图像页保存这类数据。

盘区是连续 8 个页的集合。因此，SQL Server 分配存储空间是以 1 盘区/次为单位进行分配的。盘区根据实际保存数据的不同，可以划分为两类：单一盘区(也称统一盘区)和混合盘区(也称混合区)。单一盘区中所存放的数据为一个数据对象所有，如某盘区 8 个页，存放的都是“数据表 1”的数据；混合盘区存放的数据来自多个对象，如有“数据表 1”和“数据表 2”的数据等。当混合区中的表或索引的大小增长到 8 页时，系统会将表或索引存放到专门的单一盘区中，以提高访问的效率。

3.2.2 SQL Server 数据库文件

SQL Server 运行在 Windows 操作系统平台上。对于操作系统而言，应用程序的数据都是以文件的形式进行管理的。SQL Server 数据也是以文件的形式保存在 Windows 系统中。

由上述介绍可知，SQL Server 采用两类文件来保存数据：数据文件和事务日志文件。数据文件存储数据，事务日志文件记录各种对数据库执行的操作。数据文件还可往下分为两类：主数据文件和辅助数据文件。

1. 主数据文件

主数据文件(Primary Data File，扩展名为 MDF)是 SQL Server 数据库中最重要的文件，每个 SQL Server 数据库有且仅有一个主数据文件。在主数据文件中可以保存 SQL Server 数据库中的所有数据，包括用户对象和系统对象(如系统表)。

2. 辅助数据文件

辅助数据文件(Secondary Data File，扩展名为 NDF)，也称为次数据文件，在 SQL Server 中用于保存用户数据，如用户数据表、用户视图等；但是不能保存系统数据。与主数据文件在 SQL Server 数据库中有且只能有一个不同，辅助数据文件在一个数据库中可以有多个，一个数据库最多可以有 32 767 个辅助数据文件。

一般地，对于小型数据库，主数据文件基本可以满足数据存储需要，不需要增加辅助数据文件。但是在一些数据量较大的大型数据库中，应用辅助数据文件可以给数据存储带来很多的好处，主要表现在以下几方面。

(1) 扩展数据存储空间。因为在硬盘中文件不能跨分区存储，而主数据文件只能有一个，因此，有可能在数据量特别大的时候，出现分区空间不足或者单个硬盘空间不足的问题。应用辅助数据文件，可以把多个辅助数据文件分别存储在其他盘区或不同硬盘中，从而实现对存储空间的扩展。

(2) 提高系统性能。用户对数据的使用可以分为两种基本方式：读取和写入。在一个大型多用户系统中，如果能够明确区分一部分数据表是以读为主，另一部分数据表是以写入为主，那么，就可以有针对性地把写入类表和读取类表分别存放在不同的硬盘中。由此可以实现一块硬盘只负责读取，另一块硬盘只负责写入，这样的分工可以减少硬盘读取的交互，有助于提高系统性能。

(3) 提高系统安全性。采用辅助数据文件，把用户数据存放在不同硬盘中，有助于避免或减少因部分硬盘损坏造成所有数据丢失的损失。

(4) 提高系统的可管理性。采用多辅助数据文件，再配合文件组，可以为数据备份，提供基于文件和文件组的备份方式。即用户能够以文件或文件组为单位，备份其中的部分数据，而不是所有数据。

3. 事务日志文件

事务日志文件(Log File，扩展名为 LDF)，是 SQL Server 数据库中用于记录操作事务的文件。在 SQL Server 数据库中，事务日志文件也是不可缺少的数据库文件。但与主数据文件在每个数据库中只能有一个不同，事务日志文件可以有多个，最多可达 32 767 个。为提高系统的可靠性和安全性，可以将事务日志文件与主数据文件分别存放在不同硬盘分区，

如果有多个硬盘，建议存放在不同硬盘中。

事务日志文件由系统管理，如果需要查看，可以通过第三方软件来查看，如 Lumigent Log Explorer for SQL Server。图 3-3 是通过该软件查看事务日志文件的实例。

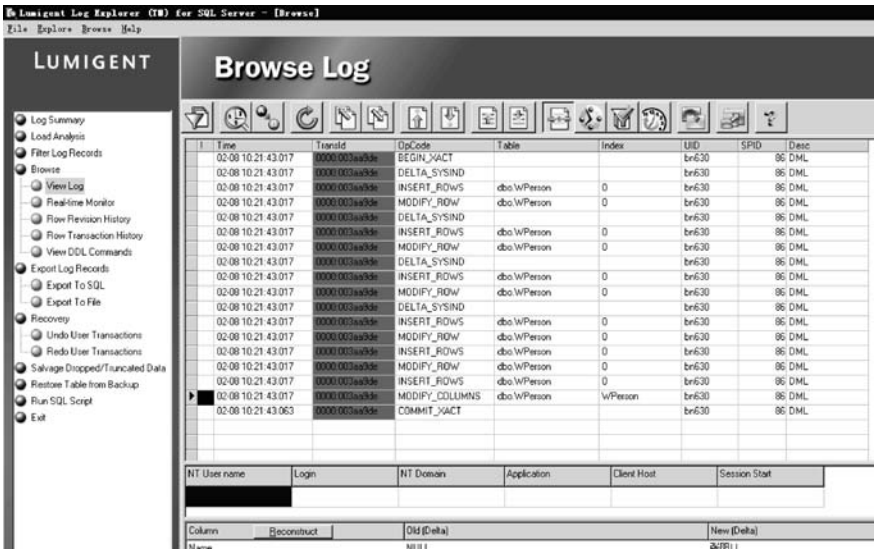

图 3-3　Lumigent Log Explorer for SQL Server 查看事务日志文件

3.2.3　文件组

在数据文件或事务日志文件数量较多的场合，可以通过文件组对数据文件和事务日志文件进行分组管理。文件组是文件的逻辑分组。在 SQL Server 2019 中文件组可以划分为两大类：主文件组(Primary File Group)，次文件组(Secondary File Group)。还有一类特殊的文件组是默认文件组(Default File Group)。

(1) 主文件组是每个数据库默认提供的文件组，该文件组不能被删除。主数据文件只能置于主文件组中。

(2) 次文件组是由用户创建的文件组，在一个数据库中用户可以根据管理需要创建多个次文件组。次文件组也被称为用户定义文件组(User-defined File Group)。

(3) 默认文件组是在新增数据库文件时，如果未明确指定该数据文件所属的文件组，那么该数据文件就会被放置在默认文件组中。系统默认的文件组对应主文件组，但可以修改，如可以将某个用户文件组设置为默认文件组。

应用文件组管理数据库文件有很多好处。一方面，在数据库文件较多的场合，可以有序管理数量众多的数据文件。另一方面，通过文件组可以实现将数据库对象，如数据表等分置于不同的硬盘或分区中。前述已介绍，辅助数据文件可以分别置于多个硬盘或硬盘分区中，但是数据库对象(如数据表等)并不能直接置于指定的数据文件中。要实现将数据表分置于指定的数据文件中，以便将数据表置于不同分区或硬盘，就需要使用文件组；因为数据表可以置于不同文件组中。

3.3 创建数据库

在 SQL Server 2019 中创建数据库的主要途径有两种：SQL Server Management Studio 和 T-SQL 语句，另外还可以通过模板、数据导入等方式来创建。

3.3.1 使用 SSMS 创建数据库

在 SQL Server Management Studio 中创建数据库的步骤如下。

(1) 选择“开始”→“程序”→Microsoft SQL Server Tools 18→SQL Server Management Studio 18。选择本机“(Local)”为目标服务器，采用“Windows 身份验证”建立连接，进入 SQL Server Management Studio。

(2) 在“对象资源管理器”窗口中，展开服务器，选择“数据库”节点，右击“数据库”，在右键菜单中选择“新建数据库”命令。

(3) 在如图 3-4 所示“新建数据库”窗口中，输入数据库名称。在“数据库文件”列表中可见两个基本的数据文件：主数据文件和日志文件，名称由系统根据数据库名称自动命名。其中，主数据文件所在的文件组为 PRIMARY，即主文件组。事务日志文件由于不能与主文件保存在同一文件组，因此，事务日志文件组的状态为“不适用”。这两个文件的默认存储路径由“数据库默认位置”指定(关于“数据库默认位置”的设定请参见图 2-19)。

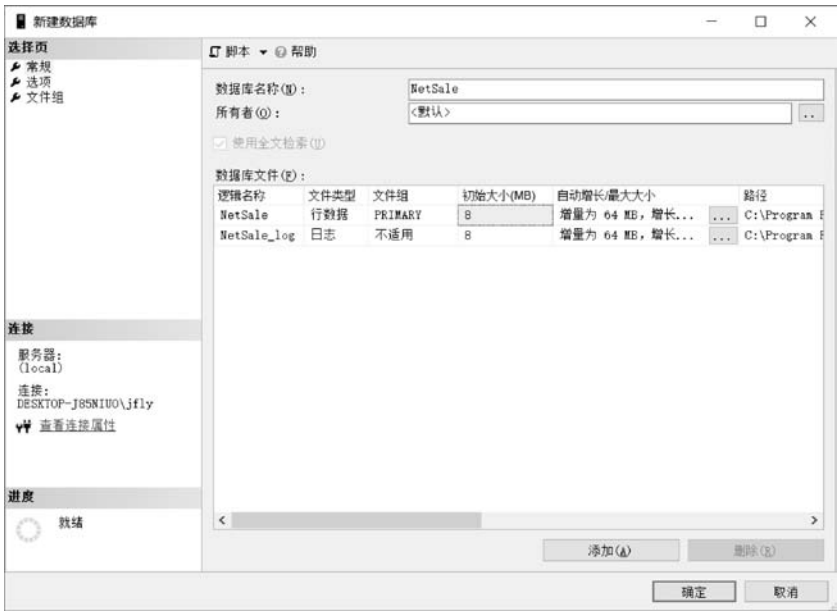

图 3-4 使用 SQL Server Management Studio 创建数据库

(4) 文件大小。主数据文件和事务日志文件存储空间的“初始大小”都为 8MB，可以直接输入需要的数值进行修改。

(5) 单击“确定”按钮，完成数据库创建。在“对象资源管理器”窗口中，可以看到新创建的数据库出现在“数据库”节点中。如果需要，可以单击“刷新”按钮来刷新节点所包含的项目。

3.3.2 使用 T-SQL 语句创建数据库

创建数据库的 T-SQL 语句为 CREATE DATABASE，该语句的基本语法如下。

```
CREATE DATABASE database_name
    [ ON
        [ PRIMARY ] [ <filespec> [ ,…n ]
        [ ,<filegroup> [ ,…n ] ]
    [LOG ON { <filespec> [ ,…n ] } ]
    ]
    [ COLLATE collation_name ]
    [ WITH<external_access_option> ]
]
[; ]
<filespec> : : =
{
(
    NAME = logical_file_name ,
        FILENAME = { 'os_file_name' | 'filestream_path' }
        [ , SIZE = size [ KB | MB | GB | TB ] ]
        [ , MAXSIZE = { max_size [ KB | MB | GB | TB ] | UNLIMITED } ]
        [ , FILEGROWTH = growth_increment [ KB | MB | GB | TB | % ] ]
) [ ,…n ]
}
<filegroup> : : =
{
FILEGROUP filegroup_name [ CONTAINS FILESTREAM ] [ DEFAULT ]
    <filespec> [ ,…n ]
}
```

主要参数和关键词含义如下。

database_name：指定新创建的数据库的名称，可长达 128 个字符。

PRIMARY：指定主数据文件的名称及路径。

LOG ON：指定事务日志文件的名称及路径。

NAME：指定数据库文件的逻辑名称，是数据库文件在 SQL Server 中的标识符，与图 3-4 数据库文件列表中的“逻辑名称”对应。

FILENAME：指定数据库文件在操作系统中的文件名称和路径，该操作系统文件名称和 NAME 的逻辑名称一一对应。

SIZE：指定数据库文件的初始存储空间大小。

MAXSIZE：指定数据库文件的最大可用存储空间大小，设置为 UNLIMITED，表示不限制最大可用存储空间。

FILEGROWTH：指定文件每次增加容量的大小，当指定数据为 0 时，表示文件不增长。

除了上述几个常用的参数之外，SQL Server 2019 的 CREATE DATABASE 语句还可以指定 FILESTREAM、DEFAULT_FULLTEXT_LANGUAGE 等参数。

使用 T-SQL 语句创建数据库的操作步骤如下。

（1）在 SQL Server Management Studio 中，单击工具栏中“新建查询”按钮，在“查询编辑器”窗口中，输入以下代码。

```
CREATE DATABASE NetSale
ON
PRIMARY(NAME = NetSale_Data,filename = 'c:\data\NetSale_Data.mdf',Size = 8MB,maxsize = 20MB,
fileGrowth = 1MB),
(NAME = NetSale_Data_1,
filename = 'C:\data\NetSale_Data1.ndf',size = 1MB,maxsize = 20MB,filegrowth = 2MB),
(NAME = NetSale_Data_2,filename = 'C:\data\NetSale_Data2.ndf',size = 2MB,maxsize = 20MB,
filegrowth = 2MB)
Log on
(NAME = NetSale_Log,filename = 'c:\data\NetSale_Log.ldf',size = 1MB,maxsize = 20MB,
filegrowth = 10%)
Go
```

上述代码创建了一个名称为 NetSale 的数据库。数据库的主数据文件逻辑名为 NetSale _Data，在操作系统中对应的文件名和路径为 c:\data\ NetSale_Data.mdf，初始空间大小为 8MB，分配的最大可用空间为 20MB，增长率为每次增加 1MB。另外，该数据库还有两个辅助数据文件和一个事务日志文件。

（2）单击工具栏中的“执行”按钮，执行完成后，可以在“对象资源管理器”窗口的“数据库”节点中找到新建的数据库 NetSale。

（3）在 Windows 系统中打开 C:\Data 文件夹，可以看到新增加四个数据库文件，如图 3-5 所示。

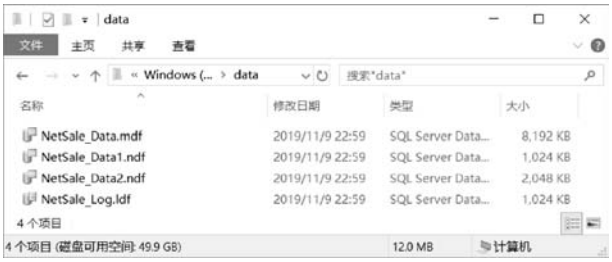

图 3-5　新增的数据库文件

3.4　设置数据库选项

3.4.1　数据库选项及设置

配置 SQL Server 服务器的选项，可以定制服务器行为和特性。同样，可以通过设置数据库选项值来定制数据库特性。在 SQL Server 2019 中提供了 50 多个数据库选项，包括：排序规则、恢复模式、兼容级别、页验证、默认游标、ANSI NULL 默认值、ANSI_NULLS 已启用、数据库状态等。

以下是部分常用数据库选项的含义。

(1) 排序规则。用于设置数据库中字符串数据的排序和比较规则,默认的排序规则是安装时选定的语言和区域的设置。

(2) 恢复模式。此项设置会影响数据库的备份与恢复方式,共有三个选项可供选择:完整、大容量日志和简单。

① 完整:允许对数据库记录完整的事务日志,可以执行数据备份和日志备份。

② 大容量日志:是完整恢复模式的附加模式。该模式以最小方式记录大容量操作,如在执行大数据量的导入操作时,采用此种恢复模式可以减少日志空间的使用。

③ 简单:数据库不能做事务日志备份,因此也无法采用事务日志备份来恢复数据。

(3) 兼容级别。用于设置数据库与早期 SQL Server 版本之间的兼容性。

(4) 包含类型。此项分为"无"和"部分"两个选项。"无"表示此数据库完全包含数据库;"部分"表示此数据库为部分包含数据库。在完全包含数据库中,任何对象和函数不能跨越应用程序模型和数据库引擎之间的边界,而部分包含数据库则允许对象跨越这种边界,并且部分包含数据库中的用户可以不通过服务器的登录名,直接连接到数据库。相关应用将在后续章节中详细介绍。

(5) 页验证。此项的主要作用是检验页面数据的正确性。SQL Server 向磁盘页写入数据时,有可能出现停电或其他硬件故障,造成数据存盘错误,即磁盘 I/O 错误。页验证的 CheckSum 选项值可以通知 SQL Server 为数据页生成一个检查和值,保存在页的页头中。当该页被读取时,会重新计算此页并生成一个检查和值。这个值会与页头原先存储的值进行比较,如果相同,则该数据页有效,否则就是已被损坏的。

(6) 默认游标。有两个选项值:Local 和 Global。设置为 Local,则表示在该数据库中所建的游标是局部游标,只能供创建它的过程调用。设置为 Global,则表示所建游标是全局游标,创建游标的数据库连接上的其他过程都可以调用这个游标。

(7) ANSI NULL 默认值。在 SQL Server 创建数据表时,可以指定列是否允许为空,如果指定某列允许为空,则该列可保存空值。如果未指定某列允许为空,而此选项设置为 True,则该列可以接收空值,此项设置为 False 时,该列不允许为空。

(8) ANSI_NULLS 已启用。如果此项设置为 True,则所有与 NULL 值的比较求得的值均为 UNKNOWN。如果此项设置为 False,当两个值都为 NULL 时,结果为 True。

(9) 递归触发器已启用。本项设置允许使用递归触发器,即在一个触发器中可以调用其他触发器。

(10) 数据库为只读。此项设置为 True,表示该数据库不允许写入,将数据库设为只读可以提高数据读取的性能,因为不需要对读取的数据执行锁的操作。对一些只提供查询而不需要写入的数据库可以设为只读。

(11) 数据库状态。表示当前数据库所处的状态,常用值有:ONLINE、OFFLINE、RESTORING、RECOVERING、RECOVERY PENDING、SUSPECT、EMERGENCY。

① ONLINE 或 NORMAL 表示数据库状态正常,处于在线或联机状态。

② OFFLINE 表示数据库处于离线状态,不能供用户使用。

③ RESTORING 表示数据库正处于还原状态,此时数据库不能使用。

④ RECOVERING 表示数据库正在恢复过程中,若恢复成功,数据库会自动转为在线状态;若恢复失败,数据库会处于可疑状态(SUSPECT)。

⑤ RECOVERY PENDING 表示数据库恢复未完成，可能是由于缺少系统资源造成的故障，需要其他操作以继续恢复后续进程，数据库不可用。

⑥ SUSPECT 表示数据库处于可疑状态，可能是主文件受损，数据库不能使用。

⑦ EMERGENCY 表示数据库处于紧急状态，只能供 sysadmin 固定服务器角色的成员访问。

数据库状态的转换如图 3-6 所示。

在启动 SQL Server 服务、创建和附加数据库时，数据库会处于 RECOVERING 状态。如果恢复成功，数据库就会自动转换为 ONLINE 状态，此时数据库可用。如果恢复失败，并且失败的原因是由资源问题引起，则数据库会设置成为 RECOVERING PENDING 状态，可由管理员解决资源问题后，再转换成为 ONLINE 状态；如果失败是由其他原因引起，则数据库会转换成为 SUSPECT 状态，此状态可由人工设置成为 EMERGENCY 状态。如果是还原数据库操作，则在还原阶段，数据库会处于 RESTORING 状态，还原结束，数据库会转换成为 RECOVERING 状态。如果恢复成功，即还原没有错误，则数据库会转换成为 ONLINE 状态。处于 ONLINE 状态的数据库也可转设为 OFFLINE，处于 OFFLINE 状态的数据库也可以转换为 ONLINE。

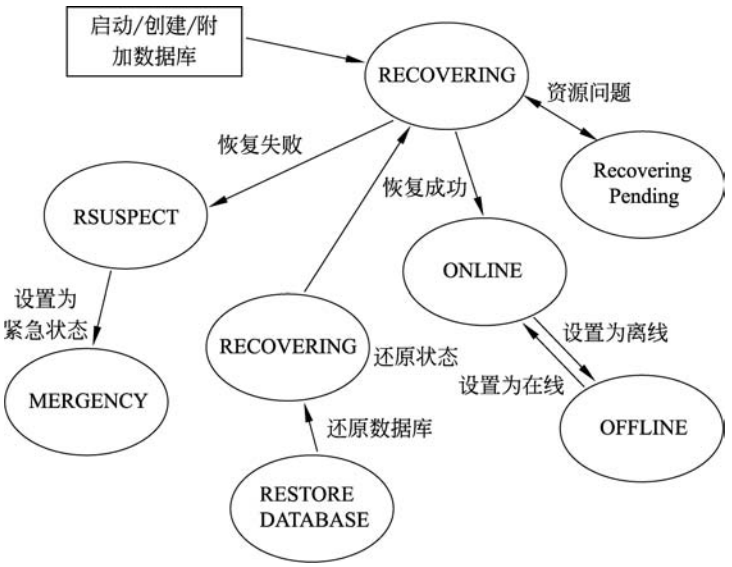

图 3-6　数据库状态转换

(12) FileStream 非事务访问：此项可以为从文件系统到 FileTables 中非事务性访问存储的 FILESTREAM 数据指定选项值。值可取以下选项之一：OFF、READ_ONLY 或 FULL。如果在服务器上未启用 FILESTREAM，则该值将设置为 OFF 并且被禁用。此项设置与 FileTable 相关。

(13) FILESTREAM 目录名称：此项为与所选数据库相关联的 FILESTREAM 数据指定目录名称。

(14) 限制访问。此项用于指定哪些用户可以访问此数据库，选值有 MULTI_USER、SINGLE_USER、RESTRICTED_USER。MULTI_USER 表示数据库状态正常，允许多个用户同时访问此数据库。SINGLE_USER 表示数据库处于维护状态，一次只允许一个用户

访问此数据库。RESTRICTED_USER 表示只有 db_owner、dbcreator 或 sysadmin 角色的成员才能使用此数据库。

(15) 已启用加密。True 表示对数据库启用加密设置,数据库中所有文件组将进行加密,如果有文件组设置为只读,则加密操作失败。

(16) 自动创建统计信息。此项用于控制系统是否能自动为优化查询创建统计信息,设置为 True,则为优化查询创建统计信息,统计信息可以为查询性能的优化提供依据。

(17) 自动更新统计信息。设置为 True,则 SQL Server 会自动更新统计信息。如果设置为 False,则需要手工更新统计信息。在系统资源较充分时,应将此项设置为 True。

(18) 自动关闭。用户连接到数据库时,数据库处于打开状态。此项设置为 True,则表示所有用户都断开连接后,数据库关闭。对于系统资源较紧张的平台,此项可以设置为 True,否则应设为 False,因为频繁关闭和打开数据库也会降低系统性能。

(19) 自动收缩。SQL Server 会定期扫描数据库,检查可用空间的情况。如果此项设置为 True,则如果可用空间占数据库已分配空间的 25%以上时,超过部分会被自动收回,即对数据库执行收缩操作。

(20) 自动异步更新统计信息。如果设置为 True,则新查询会启动统计信息自动更新,但不会等待最新的统计信息,即该查询不会使用最新的统计信息。更新后的统计信息可供后续查询使用。如果设置为 False,则新查询将会启动统计信息自动更新,并会等待更新结束,然后使用最新的统计信息来完成本次查询。要使此项起作用,需要将“自动更新统计信息”项设置为 True。

配置数据库选项可以在 SQL Server Management Studio 中完成,也可以通过 T-SQL 完成。在 SQL Server Management Studio 中设置数据库选项的操作步骤如下。

(1) 在“对象资源管理器”中右击要进行设置的数据库,在右键菜单中选择“属性”命令。

(2) 在如图 3-7 所示的“数据库属性”窗口中,选择左侧的“选项”,可以在右侧列出的选项列表中进行设置。

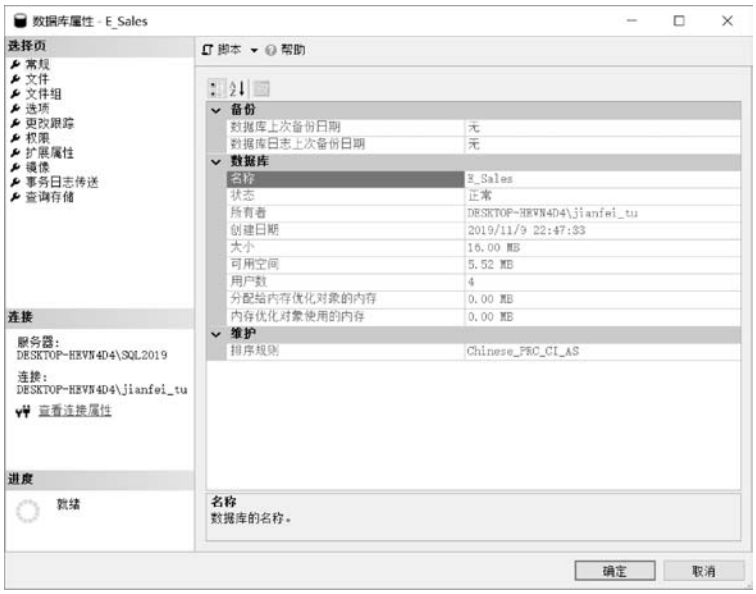

图 3-7 数据库选项设置

配置数据库选项的 T-SQL 语句与数据库修改的语句相同，都是 ALTER DATABASE。以下代码修改数据库 NetSale 的数据恢复模式为 SIMPLE，并将自动收缩设置为 ON。

```
USE master
GO
ALTER DATABASE NetSale
SET RECOVERY SIMPLE,
AUTO_SHRINK ON
GO
```

3.4.2 查看数据库信息

在 SQL Server 2019 中用户数据库的信息保存在系统数据库中，可以通过对系统数据库执行查询或通过系统存储过程、系统视图和函数来查看用户数据库的信息。

1. 应用系统视图、函数查看数据库信息

(1) 通过系统视图 sys.databases 可以查看数据库的基本信息，包括各选项的设置值。

(2) 通过 sys.database_files 视图可以查看数据库的文件信息。

(3) 通过 sys.filegroups 视图可以查询数据库文件组信息。

(4) 通过 DATABASEPROPERTYEX 函数可以查看数据库选项的值。

例如，以下代码展示了上述各项的使用方法。

```
select * from sys.databases
GO
select * from sys.database_files
GO
select * from sys.filegroups
GO
select DATABASEPROPERTYEX('E_Sales','Status')
GO
```

执行结果如图 3-8 所示。

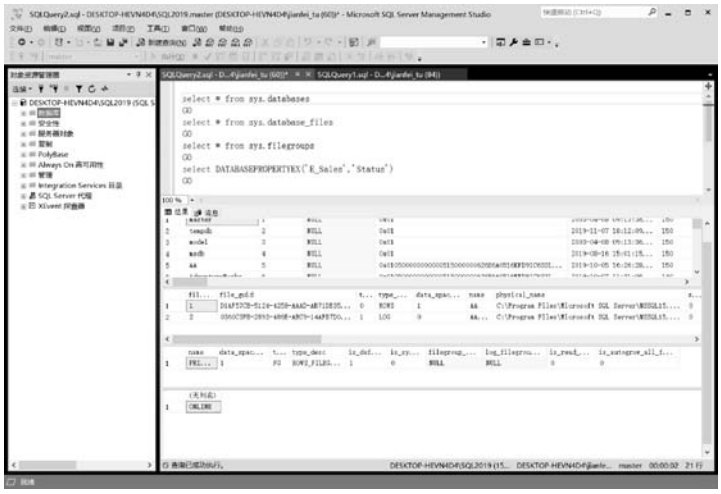

图 3-8 应用系统视图查看数据库信息

2. 应用系统存储过程查看数据库信息

(1) 通过系统存储过程 sp_spaceused 可以查看数据库空间的使用情况。

(2) 通过系统存储过程 sp_helpdb 可以查看数据库的信息。

执行的操作代码如下。

```
USE NetSale
GO
sp_spaceused
GO
sp_helpdb
GO
sp_helpdb E_Sales
GO
```

执行结果如图 3-9 所示。

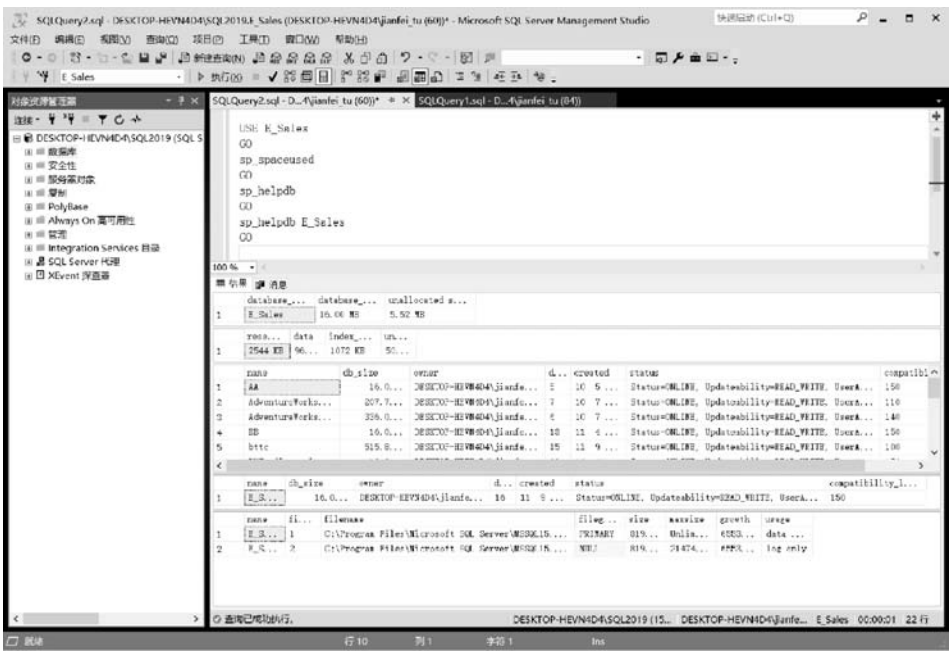

图 3-9　应用系统存储过程查看数据库信息

3.5　管理数据库

数据库在运行过程中,因各种原因,可能需要修改数据库参数、选项等值。例如,调整数据库的空间大小,增加数据库文件,管理数据库文件组,调整文件所属的文件组等。修改数据库的操作既可以在 SQL Server Management Studio 中完成,也可以通过 T-SQL 语句来实现。本节采用上述两种方法,说明数据库修改的过程和方法。

3.5.1 扩大数据库空间

数据库在实际使用过程中，会由于数据量增加导致原先分配的数据库空间不够用，就需要给数据库增加存储空间。有两种常用的方法可以扩大数据库存储空间：手工改动数据文件的大小；设置数据库文件的自动增长方式。

1. 手工扩大数据库文件大小

在 SQL Server Management Studio 中手工扩大数据库空间的操作过程如下。

(1) 在“对象资源管理器”窗口中，展开服务器、数据库节点。右击要调整空间大小的数据库，在右键菜单中选择“属性”命令。

(2) 在“数据库属性”窗口中，单击左侧的“文件”选项，如图 3-10 所示。

(3) 在“数据库文件”列表下，分别调整数据文件、事务日志文件的大小。如将主数据文件大小调整到 20MB，事务日志文件调整到 15MB。

(4) 单击“确定”按钮，保存对数据库文件大小的修改。

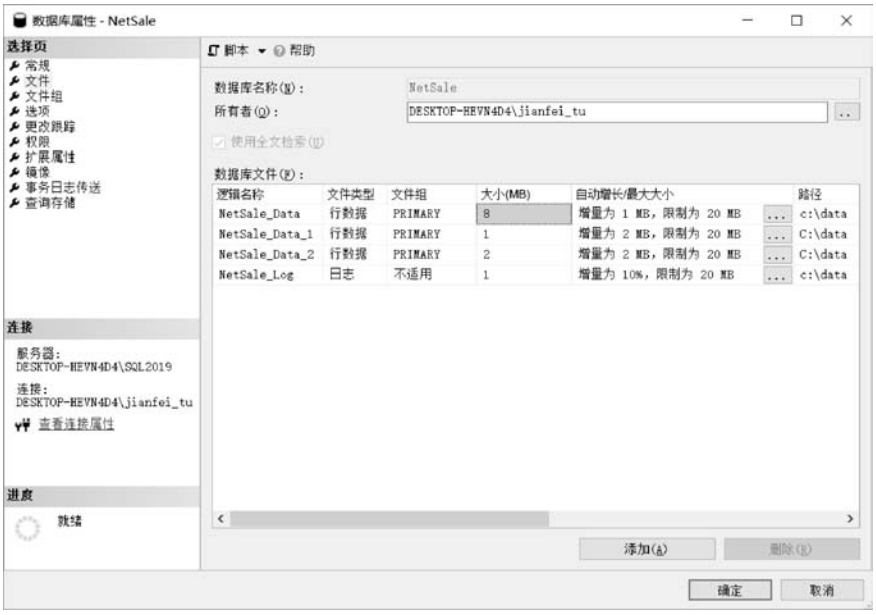

图 3-10 在数据库属性中修改数据库文件大小

上述操作过程，也可以通过执行以下 T-SQL 代码实现。该 T-SQL 代码使用 ALTER DATABASE 语句，通过 Modify File 修改数据库 NetSale 的主数据文件，使主数据文件从原先的 8MB，扩大为 15MB。

```
USE master
GO
ALTER DATABASE NetSale
Modify File (NAME = NetSale_Data,size = 15MB)
```

2. 设置数据库自动增长方式

手工调整数据库大小只能不定期地由管理员来操作，需要管理员经常性地监控数据库

的使用情况；在数据数量比较大的场合，往往会产生很大的工作量。通过对数据库增长方式进行设定，可以让 SQL Server 系统自动根据使用情况进行调整。当数据库的使用量超过设置的数据库文件大小时，可以按照设置的增长方式自动增长。

在 SQL Server Management Studio 中设置数据库自动增长方式的操作步骤如下。

(1) 在如图 3-10 所示的“数据库属性”窗口中，在“数据库文件”列表中选择要调整自动增长方式的数据文件，单击“自动增长/最大大小”栏后的按钮。

(2) 在如图 3-11 所示的“更改自动增长设置”对话框中，选中“启用自动增长”复选框，设置文件增长方式和最大文件大小。

图 3-11　设定数据文件自动增长的方式

① 文件增长。分为两种方式：“按百分比增长”，即每次在现有文件大小基础上增加指定百分比的量增长；“按 MB”，即每次在现有大小的基础上增加设定的增长量。

② 最大文件大小。用于设定该数据文件最大可以扩展的空间，“限制为(MB)”，即文件最大不能超过设定大小；“无限制”，最终可用大小由数据文件所在的硬盘分区可用空间的大小决定。当选择“限制为(MB)”时，当数据量超过设置的大小时，超过部分不能保存。

(3) 事务日志文件的大小也可以做相应的设置。

3.5.2　收缩数据库空间

如果预先分配的数据库空间过大，超过数据库的实际使用需要；或者在使用过程中清除了大量数据，造成所占空间浪费。这时可以通过收缩数据库空间来收回多余的存储空间。

在 SQL Server 2019 中，可以通过 4 种方式来收缩数据库。

(1) 在如图 3-10 所示窗口中直接修改数据库文件的大小，但是这种方法较适用于刚分配尚未使用的情况。

(2) 通过设置数据库选项 AUTO_SHRINK 为 True，使 SQL Server 自动监控数据库空间的使用情况，把可用自由空间限制在 25%以内。

(3) 通过收缩整个数据库的空间，来实现多余空间的回收。

(4) 收缩指定的数据文件来实现收缩。

1. 收缩整个数据库

收缩整个数据库的操作，可以使用 SQL Server Management Studio 和 T-SQL 语句来实现。在 SQL Server Management Studio 中收缩整个数据库的操作步骤如下。

(1) 在“对象资源管理器”窗口中，展开服务器、数据库节点。右击待收缩的数据库，在右键菜单中选择“任务”→“收缩”→“数据库”命令。

(2) 在如图 3-12 所示的“收缩数据库”窗口，选中“在释放未使用的空间前重新组织文件，选中此项可能会影响性能”复选框，设置其下的“收缩后文件中的最大可用空间”，如为 50%，即表示将数据库的可用空间从原先的 72%收缩到 50%。

(3) 单击“确定”按钮，执行收缩。

收缩数据库的 T-SQL 语句为 DBCC SHRINKDATABASE，语法如下：

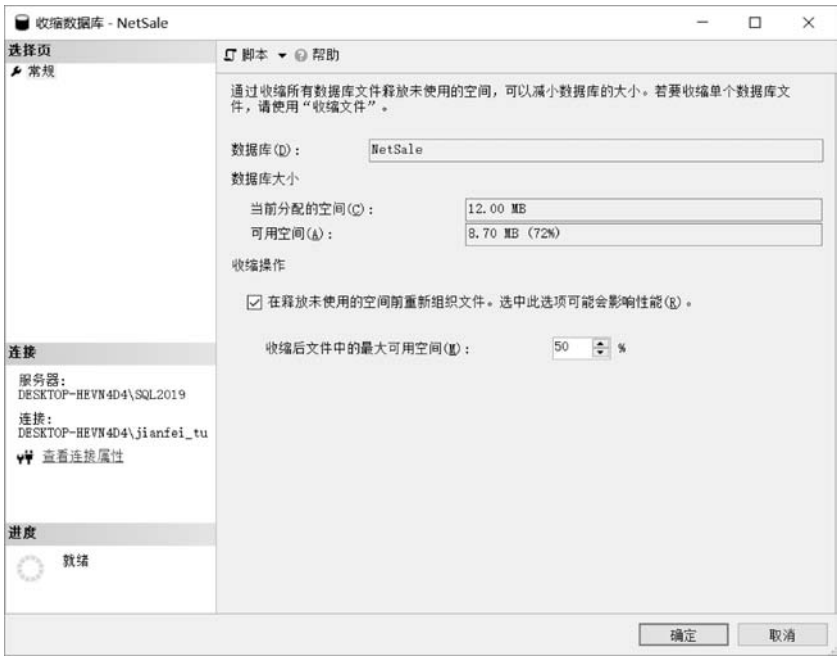

图 3-12　收缩数据库

```
DBCC SHRINKDATABASE
( database_name | database_id | 0
    [ , target_percent ]
    [ , { NOTRUNCATE | TRUNCATEONLY } ]
)
[ WITH NO_INFOMSGS ]
```

各参数的主要含义如下。

database_name | database_id | 0：指定收缩的数据库的名称或 ID。如果指定为 0，表示为当前数据库，即在“查询编辑器”窗口中输入代码时，在工具栏“可用数据库”项选中的数据库。

target_percent：收缩的目标大小，如 10%。

NOTRUNCATE | TRUNCATEONLY：二选一参数。NOTRUNCATE 参数表示收缩时将数据文件中的数据移动到前面的数据页，但并不将多余的空间释放出来，数据文件的大小并未改变。因此，数据库好像并未收缩，此项只针对数据文件，对日志文件不产生作用。TRUNCATEONLY 参数表示收缩时将文件末尾的未使用空间释放出来，但文件内数据并不移动。由于 TRUNCATEONLY 只释放文件尾部的未使用空间，因此 target_percent 参数将不起作用。

WITH NO_INFOMSGS：表示取消严重级别从 0 到 10 的所有消息。

例如，要将上例中 NetSale 数据库的可用空间收缩至整个数据空间的 20%，可以执行如下代码。

```
DBCC SHRINKDATABASE(NetSale,20 % )
```

2. 收缩指定的数据文件

收缩整个数据库，会收缩可收缩的数据文件。SQL Server 2019 提供的对指定数据文件进行收缩的功能，为数据库空间压缩提供了更为灵活的操作方式。

在 SQL Server Management Studio 中对指定数据文件进行收缩的操作步骤如下。

(1) 在“对象资源管理器”窗口，右击要进行收缩的数据库，在右键菜单中选择“任务”→“收缩”→“文件”命令。

(2) 在如图 3-13 所示的“收缩文件”窗口中，可以对指定的数据文件和日志文件分别进行收缩。例如，要收缩主数据文件，可以在“文件类型”中选择“数据”，“文件名”选择主数据文件名。收缩操作可以分为以下三种。

① 释放未使用的空间。即将指定文件中未使用的空间释放出来，数据在文件中不移动。

② 在释放未使用的空间前重新组织页。此项用于将指定文件收缩到指定大小，并将数据重新组织，使用时需要指定收缩后的文件大小。

③ 通过将数据迁移到同一文件组中的其他文件来清空文件。此项用于将指定文件中的数据移动到同文件组中的其他文件，然后清空此文件。此项不适合于主数据文件，因为主数据文件在每个数据库必须有一个，且主数据文件中存放的系统数据对象也无法存放到其他数据文件中。

(3) 如果需要收缩事务日志文件的大小，可以在“文件类型”中选择“日志”。

收缩单个数据库文件的 T-SQL 代码为 DBCC SHRINKFILE，语法如下。

```
DBCC SHRINKFILE
(
    { file_name | file_id }
    { [ , EMPTYFILE ]
    | [[ , target_size ] [ , { NOTRUNCATE | TRUNCATEONLY } ] ]
    }
)
[ WITH NO_INFOMSGS ]
```

各参数的主要含义如下。

file_name | file_id：用于指定要收缩的文件在 SQL Server 中的逻辑名或文件 id。

EMPTYFILE：可选参数，表示是否清空指定收缩的文件，如要清空，则会把该文件中的数据移动到同文件组的其他文件中。

target_size：指定文件收缩的目标大小。

NOTRUNCATE | TRUNCATEONLY，WITH NO_INFOMSGS：与收缩整个数据库的 DBCC SHRINKDATABASE 中的含义相同。

例如，以下代码对数据库 NetSale 中的主数据文件执行收缩，收缩到 60MB。

```
USE NetSale
GO
DBCC SHRINKFILE (NetSale,60)
GO
```

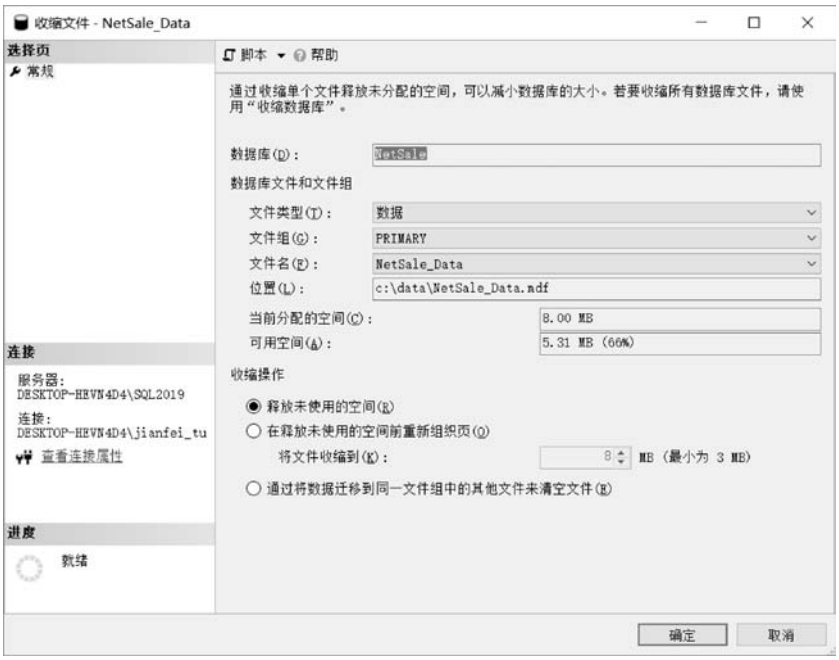

图 3-13　收缩文件

注意：数据库最大的收缩量不能小于现有数据库的实际使用量，即如果一个数据库中实际使用量为 10MB，最后收缩完成后的数据库不能小于 10MB。

3.5.3　管理数据库文件

由前述介绍可知，数据库中可以有多个数据文件和事务日志文件。通过增加数据文件和日志文件，可以扩大数据库存储空间，提高数据库运行的性能。但如果数据文件过多，也会造成数据库管理困难，如系统迁移，尤其是硬盘更换时可能会遗漏文件。因此，多余的文件也需要及时清除。

1. 增加数据库文件

在 SQL Server 2019 中，增加数据库文件也可以通过 SQL Server Management Studio 和 T-SQL 语句来实现。

在 SQL Server Management Studio 中增加数据文件的操作步骤如下。

(1) 在“对象资源管理器”窗口，展开服务器、数据库节点；右击要增加文件的数据库，在右键菜单中选择“属性”命令。

(2) 在“数据库属性”窗口中，单击左侧的“文件”选项，如图 3-14 所示。如果要增加数据文件或事务日志文件，可以单击“数据库文件”列表下侧的“添加”按钮，在列表中会新增一个空行。

(3) 在新增的空行中输入文件的逻辑名称。如果是新增数据文件，“文件类型”可选为“行数据”，如果是事务日志文件，可选“日志”；继续设定“文件组”“初始大小”“自动增长方式”“路径”“文件名”等项参数。

(4) 设置完毕后，单击“确定”按钮，新添加的文件会新增到此数据库中。

新增数据库文件的 T-SQL 语句为 ALTER DATABASE 和 ADD FILE，例如：

```
ALTER DATABASE NetSale
ADD FILE
(
    NAME = NetSale_LOG2,
    FILENAME = 'C:\data\NetSale_LOG2.log',
    SIZE = 8MB,
    MAXSIZE = 100MB,
    FILEGROWTH = 64MB)
```

上述代码向数据库 NetSale 中新增一个事务日志文件，该文件的逻辑名称为 NetSale_LOG2，物理名称为 C:\data\NetSale_LOG2.log，初始大小为 8MB，最大可用空间为 100MB，增长率为每次增加 64MB。

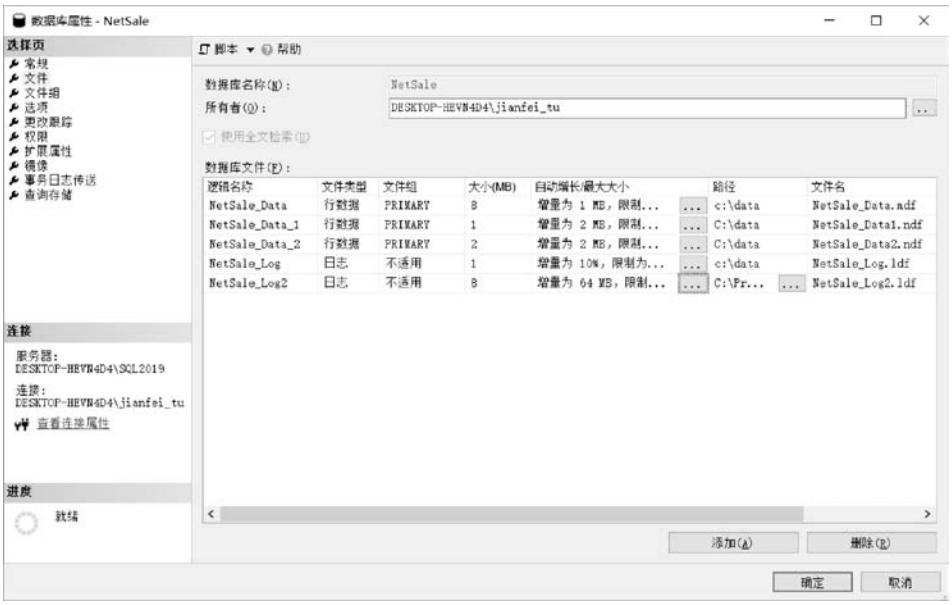

图 3-14　增加数据库文件

2. 删除数据库文件

在 SQL Server Management Studio 中删除数据库文件的操作步骤如下。

(1) 在“对象资源管理器”窗口中，展开服务器、数据库节点；右击要删除文件的数据库，在右键菜单中选择“属性”命令。

(2) 在“数据库属性”窗口中，选择“文件”项，如图 3-14 所示。“数据库文件”列表中选中要删除的数据文件或事务日志文件，然后单击“删除”按钮，在列表中选中的文件项会被删除。

(3) 单击“确定”按钮，完成删除操作。

删除数据库文件的 T-SQL 语句为 ALTER DATABASE 和 REMOVE FILE，例如：

```
ALTER DATABASE NetSale
REMOVE FILE NetSale_LOG2
```

上述代码表示从数据库 NetSale 中删除文件 NetSale _LOG2。

注意：不能删除主数据文件，也不能将所有事务日志文件都删除。

3.5.4 管理文件组

文件组是 SQL Server 对数据库文件进行逻辑管理的工具。使用文件组可以对数据库文件进行分组管理，可以简化管理的复杂度，也可以提升数据库的性能。另外，通过文件组还可以将数据表保存在不同硬盘或分区中。

系统在创建新数据库时，会同时创建主文件组 PRIMARY。因此，默认情况下，所有数据文件都会存放在主文件组中。如果用户创建了用户文件组，可以将部分数据文件存放到用户文件组中。但是主数据文件只能存放在主文件组中。

在 SQL Server 2019 中，增加/删除文件组可以使用 SQL Server Management Studio 和 T-SQL 语句来实现。

1. 增加文件组

在 SQL Server Management Studio 中增加文件组的操作如下。

(1) 在“对象资源管理器”窗口中，展开服务器、数据库节点；右击要增加文件组的数据库，在右键菜单中选择“属性”命令。

(2) 在“数据库属性”窗口中，选择左侧的“文件组”选项，在如图 3-15 所示窗口中，单击“添加”按钮。

(3) 在“行”列表下会新增一个空行，可以输入新文件组的名称，如 NewUserFileGroup。如果选中“默认组”项，则该文件组将被设为默认文件组。今后新增的文件不做特别说明，都会存于该默认组中。如果选中“只读”项，该文件组将无法添加文件，但是可以在文件添加完毕后，将“只读”项选中，则该文件组中的文件只能读取，不能修改。

如果选中“自动增长所有文件”，即表示当文件组中的文件在填满现有可用存储空间后，会自动增长文件空间。这种自动增长文件空间的设置，跟 SQL Server 往文件组内的文件中填充数据的策略和文件存储空间的增长方式设置有关。当文件组下有多于 1 个的文件时，SQL Server 数据库引擎按文件中的可用空间比例将数据写入文件组中的每个文件，这种策略被称为按比例填充。例如，如果文件 f1 有 100MB 可用空间，文件 f2 有 200 MB 可用空间，则从文件 f1 中使用一个盘区，从文件 f2 中使用两个盘区，直至两文件基本同时填满。当文件组中的所有文件填满后，SQL Server 会采用循环方式一次自动扩展一个文件以容纳更多数据，即先扩展文件组中第一个设置自动增长的文件，扩展空间填满后，再扩展第二个设置自动增长的文件，直至填满，然后再次循环。如果文件设置不允许自动增长，那么此文件在存储空间用完后，即不再填充数据。

(4) 选择“文件”项，单击“添加”按钮，在数据库文件列表中新增一个文件，在该新文件的“文件组”项中选择新建的文件组 NewUserFileGroup，即把该文件存于此新文件组中。

2. 删除文件组

如果需要删除文件组，可以在如图 3-15 所示窗口的“行”列表中选中要删除的文件组，然后单击“删除”按钮。但是，主文件组不能删除。删除文件组会同时删除此文件组下的文件，因此，如果只需要删除文件组，而不删除文件，可以在删除前将文件移动到其他文件组。

移动文件所在的文件组，可以在如图 3-14 所示窗口中，将指定的数据库文件“文件组”项的值改为目标文件组。

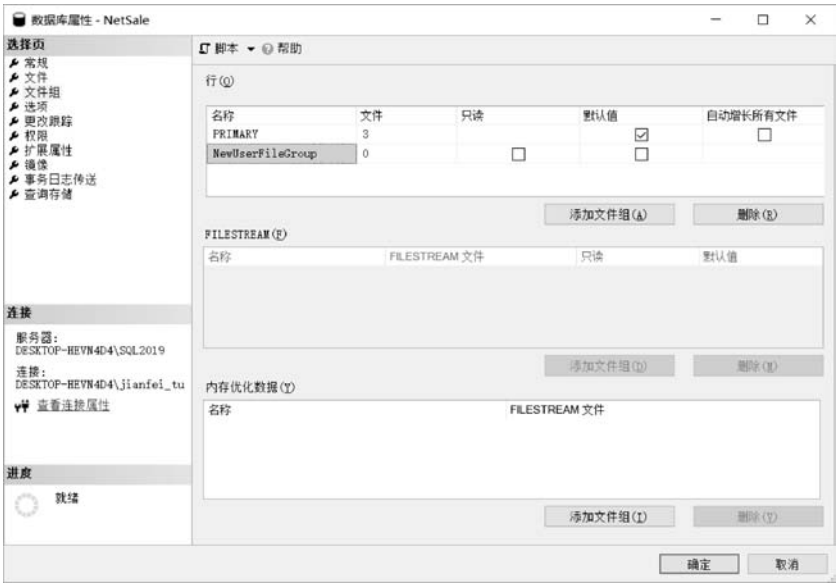

图 3-15　新增文件组

增加文件组的 T-SQL 语句为 ALTER DATABASE 和 ADD FILEGROUP，以下代码实现了在数据库 NetSale 中添加一个文件组 NewUserFileGroup，并新增一个数据文件 NetSale_data4 到该文件组中。

```
ALTER DATABASE NetSale
ADD FILEGROUP NewUserFileGroup1
GO
ALTER DATABASE NetSale
ADD FILE
(
    NAME = NetSale_data4,
    FILENAME = 'C:\data\NetSale_data4.ndf',
    SIZE = 5MB,
    MAXSIZE = 100MB,
    FILEGROWTH = 5MB
)
TO FILEGROUP NewUserFileGroup1
GO
```

将用户文件组设置为“默认文件组”的 T-SQL 代码如下。

```
ALTER DATABASE NetSale
MODIFY FILEGROUP NewUserFileGroup DEFAULT
GO
```

修改已存在文件组名称的 T-SQL 代码如下。

```
ALTER DATABASE NetSale
MODIFY FILEGROUP NewUserFileGroup NAME NewUserGROUP
GO
```


删除用户文件组的 T-SQL 代码如下。

```
ALTER DATABASE NetSale
REMOVE FILEGROUP NewUserGROUP
GO
```

3.5.5 删除数据库

如果数据库使用完毕后不再需要，可以进行删除。删除数据库的操作可以通过 SQL Server Management Studio 和 T-SQL 来完成。

在 SQL Server Management Studio 中删除数据库的操作步骤如下。

(1) 在“对象资源管理器”窗口中，展开服务器、数据库节点；右击待删除的数据库，在右键菜单中选择“删除”命令。

(2) 在如图 3-16 所示的“删除对象”窗口中，可以选中“删除数据库备份和还原历史记录信息”复选框，表示同时从系统数据库 msdb 中删除该数据库的备份和还原的历史记录。选中“关闭现有连接”复选框，表示断开删除前已建立的用户与该数据库的连接。

(3) 单击“确定”按钮，该数据库相关的数据文件和事务日志文件，以及文件组都会从系统中被删除。

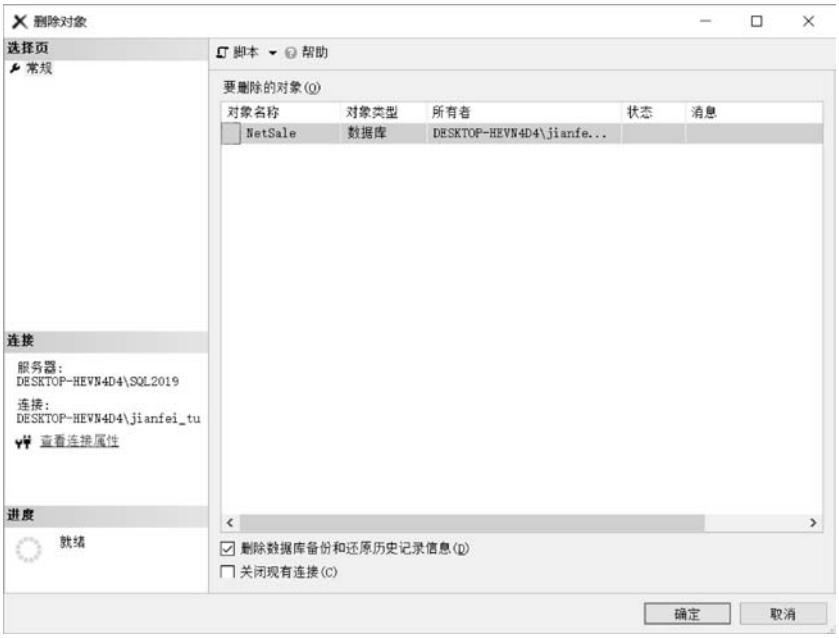

图 3-16 删除数据库

删除数据库的 T-SQL 语句为 DROP DATABASE，语法如下。

```
DROP DATABASE { database_name | database_snapshot_name } [ ,…n ] [; ]
```

各参数含义如下。

database_name | database_snapshot_name，待删除的数据库或者数据库快照的名称。

DROP DATABASE 可以删除单个数据库，也可一次删除多个数据库，如删除单个数据库的语句如下。

```
DROP DATABASE NetSale
```

一次删除多个数据库的语句如下。

```
DROP DATABASE NetSale, NetSale1, NetSale2
```

注意：删除数据库是个不可逆的过程，请在删除数据库前务必确认无误，以免误删。

3.5.6 分离数据库

处于在线状态的数据库，其数据库文件由 SQL Server 占用，无法通过 Windows 操作系统对该数据库的文件进行操作。在有些场合，例如，需要将数据库从一台服务器复制到另一台服务器，或者需要从一个硬盘或分区迁移到另一个硬盘或分区时，可以将数据库从 SQL Server 服务中分离出来，然后对数据库文件执行复制或移动。

在 SQL Server Management Studio 中分离数据库的操作如下。

(1) 在“对象资源管理器”窗口中，展开服务器、数据库节点；右击要分离的数据库，在右键菜单中选择“任务”→“分离”命令。

(2) 在如图 3-17 所示的“分离数据库”窗口中，如果看到“消息”栏有“活动连接”的消息，即说明已有用户连接到该数据库；需要选中“删除连接”，此选项表示在分离时需要先把当前的“连接”删除。

选中“更新统计信息”表示当 SQL Server 在分离数据库前更新状态信息，如全文索引，这样不会丢失之前生成的全文索引信息。

(3) 单击“确定”按钮，分离数据库。

注意：分离数据库，实际上只是从 SQL Server 的 master、msdb 等系统数据库中删除与被分离数据库有关的信息，实际上并没删除硬盘中的数据库文件，这与数据库删除是完全不同的。因此，分离后的数据库文件还存在，如果需要可以复制到其他硬盘、分区或服务器上。

分离数据库的 T-SQL 语句，需要调用系统存储过程 sp_detach_db，语法如下。

```
sp_detach_db [ @dbname = ] 'database_name'
    [ , [ @skipchecks = ] 'skipchecks' ]
    [ , [ @keepfulltextindexfile = ] 'KeepFulltextIndexFile' ]
```

各参数含义如下。

@dbname：是指待分离的数据库名称。

@skipchecks：默认值为 NULL，表示分离前会更新统计信息，为 True 表示跳过更新统计信息，直接分离。

@keepfulltextindexfile：设置为 True(默认值)，表示保留被分离数据库中生成的所有全文索引文件。

下例代码，分离了数据库 NetSale，且分离前会更新统计信息和保留全文索引。

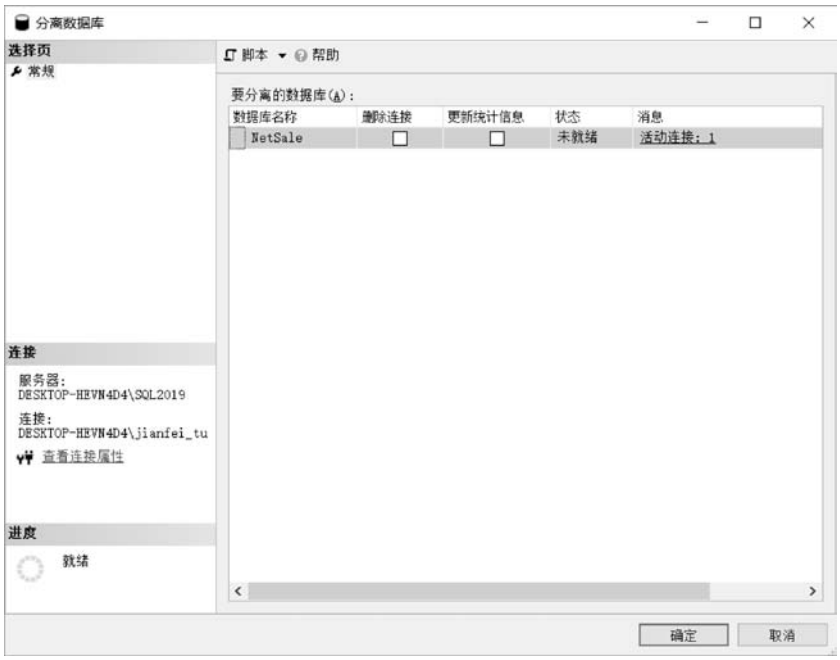

图 3-17　分离数据库

```
use master
Go
sp_detach_db 'Netsale'
GO
```

3.5.7　附加数据库

分离后的数据库如果需要重新使用，可以被再次附加到 SQL Server 中。附加操作，实际就是将数据库的信息保存到 SQL Server 的 master、msdb 等系统数据库中，以便 SQL Server 可以再次使用和管理这个数据库。

在 SQL Server 2019 中，附加操作同样可以采用两种方式来完成：通过 SQL Server Management Studio 和 T-SQL 语句。

在 SQL Server Management Studio 中附加数据库的操作步骤如下。

(1) 在“对象资源管理器”窗口中，右击“数据库”节点，在右键菜单中选择“附加”命令。

(2) 在如图 3-18 所示的“附加数据库”窗口中，可以看到“要附加的数据库”和“数据库详细信息”列表都为空，需要添加数据库文件。单击“添加”按钮。

(3) 在如图 3-19 所示的“定位数据库文件”窗口中，找到并添加所要附加数据库的主数据文件(MDF 文件)，然后单击“确定”按钮，返回到前一窗口。

(4) 此时，在“附加数据库”对话框中会根据添加的主数据文件，更新“要附加的数据库”列表和“数据库详细信息”列表，如图 3-20 所示。

(5) 确认信息无误，单击“确定”按钮，系统执行数据库附加操作。附加完成后，该数据库会出现在“对象资源管理器”的数据库节点下，如果未看到该数据库，可单击工具栏中的

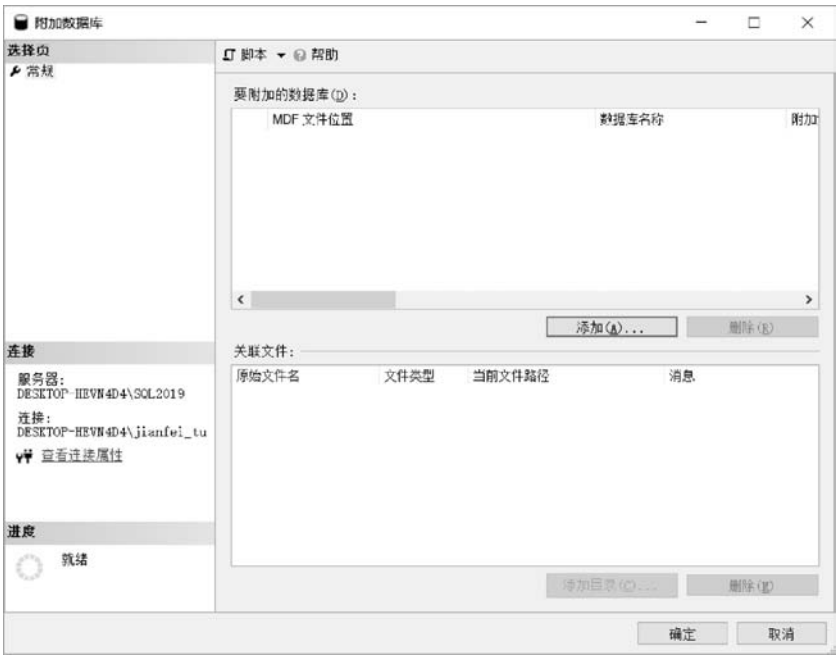

图 3-18　附加数据库

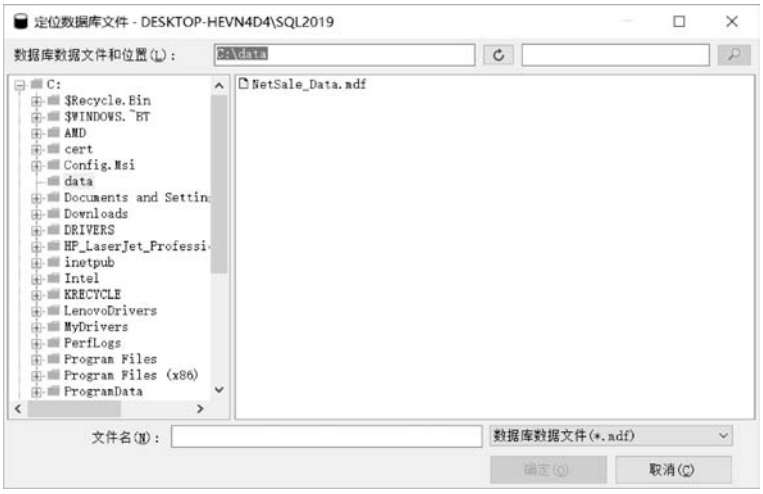

图 3-19　选择要附加的数据库的主数据文件

“刷新”按钮,刷新数据库节点。

数据库附加操作还可以通过系统存储过程 sp_attach_db 来实现,T-SQL 的语法如下。

```
sp_attach_db [ @dbname = ] 'dbname'
    , [ @filename1 = ] 'filename_n'[ , … 16 ]
```

参数的含义如下。

@dbname:是指要附加到该服务器的数据库的名称。该名称不能与服务器中现有的数据库名称相重。

图 3-20　确认附加的数据库信息

@filename1：是指用于附加的数据库文件的物理名称，包括路径。文件名可以多达 16 个，但文件名中必须有主数据库文件。

应用上述语句可以构造如下数据库附加的实例，该段代码将数据库 NetSale 附加到系统中，期间使用了 4 个文件。

```
use master
Go
sp_attach_db @dbname = N'NetSale',
    @filename1 = N'C:\data\NetSale_Data.mdf',
    @filename2 = N'C:\data\NetSale_Data1.ndf',
    @filename3 = N'C:\data\NetSale_Data2.ndf',
    @filename4 = N'C:\data\NetSale_Log.ldf';
```

sp_attach_db 系统存储过程在后续版本中将会被删除，SQL Server 推荐使用 CREATE DATABASE database_name FOR ATTACH 来附加数据库。

使用 CREATE DATABASE database_name FOR ATTACH 附加数据库的语法包含在创建数据库的 T-SQL 语法中，下面用实例说明附加的操作步骤。

```
use master
GO
CREATE DATABASE NetSale ON
    (FILENAME = 'C:\data\NetSale_Data.mdf'),
    (FILENAME = 'C:\data\NetSale_Data1.ndf'),
    (FILENAME = 'C:\data\NetSale_Data2.ndf'),
    (FILENAME = 'C:\data\NetSale_Log.ldf')
FOR ATTACH;
GO
```

上述代码将 NetSale 数据库附加到系统中，其中涉及的数据库文件包括 1 个主数据文

件、3 个辅助数据文件和 1 个事务日志文件。

3.5.8 部分包含数据库

从上述介绍的分离、附加操作可知，可以通过分离附加将数据库从一个服务器实例迁移到另一个服务器实例。即在源服务器实例上分离数据库，然后将数据库文件复制到目标服务器，在目标服务器实例上附加数据库。第 11 章介绍的备份还原操作也可以解决上述问题。但是，由于数据库所包含的往往只是用户的数据，而很多与数据库相关的登录名、元数据等内容一般保存在系统数据库中，这些数据无法通过用户数据库的分离附加或备份还原直接迁移过去，还需要其他额外工作来完成。这在一定程度上增加了数据库迁移的复杂性。

部分包含数据库(Contained Database)是 SQL Server 用于解决这一问题的特性，其主要作用是提高数据库在多个数据库引擎服务实例中迁移时的便捷性。通过将用户数据库启用部分包含特性，可以将数据库用户等数据保存在用户数据库中，从而可以最大限度迁移数据库的完整信息。同时，部分包含数据库还可以与 AlwaysOn 结合，提高系统管理和维护的可靠性与高可用性。

1. 启用部分包含数据库

部分包含数据库是服务器级的选项，要启用此特性，需要在服务器上启用此选项。启用部分包含数据库，可以通过 SQL Server Management Studio 和 T-SQL 来实现。

使用 SQL Server Management Studio 启用部分包含数据库的步骤如下。

(1) 在“对象资源管理器”窗口中，右击服务器，在右键菜单中选择“属性”命令，在如图 3-21 所示的“服务器属性”窗口中，选择左侧“高级”选项，在“高级”选项卡中设置“启用包含的数据库”为 True。

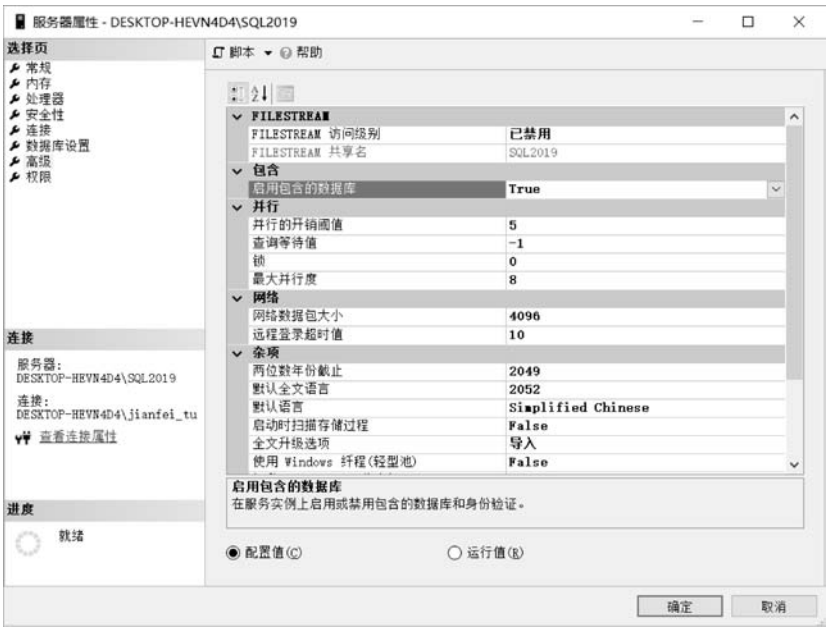

图 3-21 启用包含的数据库

(2) 单击“确定”按钮,保存选项设置值。

使用 T-SQL 启用部分包含数据库的代码如下。

```
sp_configure 'show advanced options'1,
GO
sp_configure 'contained database authentication', 1;
GO
RECONFIGURE;
GO
```

2. 数据库设置

数据库要开启部分包含的特性,需要在数据库选项中做对应设置。设置数据库的此项选项也可以通过 SQL Server Management Studio 完成。操作步骤如下。

(1) 在“对象资源管理器”窗口中,展开服务器,选择“数据库”节点,右击 NetSale 数据库,在右键菜单中选择“属性”命令。在如图 3-22 所示的“数据库属性”窗口中,选择左侧的“选项”,在右侧的“选项”页中设置“包含类型”为“部分”。

(2) 单击“确定”按钮后,保存选项设置值。

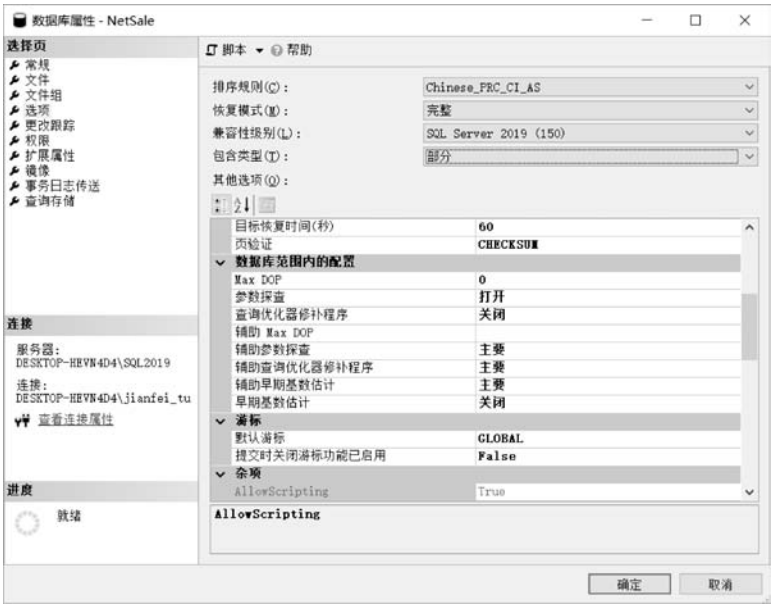

图 3-22 数据库选项设置包含类型

3. 创建数据库级的用户

启用部分包含数据库与未启用之间的差别,可以从创建数据库用户的操作上进行比较。展开 NetSale 数据库的“安全性”节点,右击“用户”,在右键菜单中选择“新建用户”命令。在如图 3-23 所示的“数据库用户-新建”窗口中,输入用户名、密码等。由于此数据库已启用部分包含特性,因此不需要指定登录名。

在未启用部分包含数据库的数据库中创建数据库用户的窗口如图 3-24 所示。通过对比可以发现,此处指定登录名是必须的。

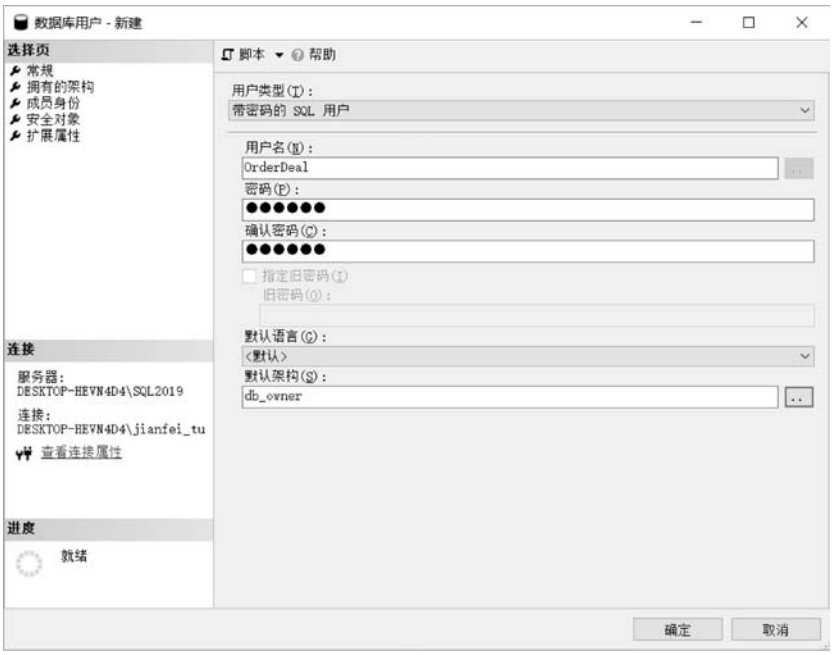

图 3-23　在启用部分包含的数据库中新建数据库用户

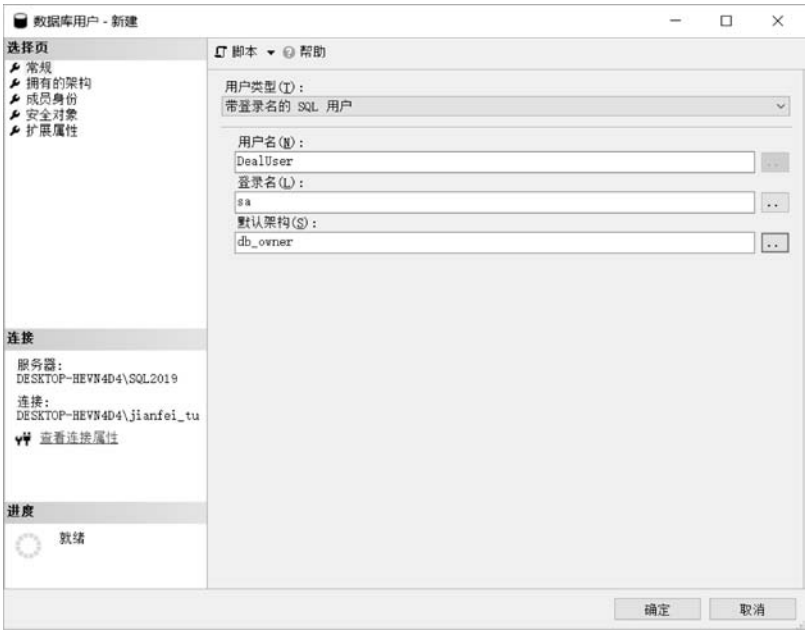

图 3-24　在未启用部分包含的数据库中新建数据库用户

登录名是服务器对象，不能包含在数据库中。因此，通过数据库分离附加或者备份还原迁移到另一服务器时，需要重新建登录名，并映射登录名与数据库用户之间的关系。通过启用部分包含数据库这一新特性，数据库用户会被同时迁移，且可以使用此数据库用户直接连接此数据库。

3.6 数据库快照

所谓“数据库快照”，顾名思义，是指在某一时刻对数据库所做的数据备份。数据库快照如同生活中的“快照”一样，可以完整地反映数据库在执行快照时刻的情况。由于是快照，因此备份出来的数据是一份静态数据，为保持与原时刻一致是不允许修改的。

数据库快照的执行过程可用图 3-25 来表示。

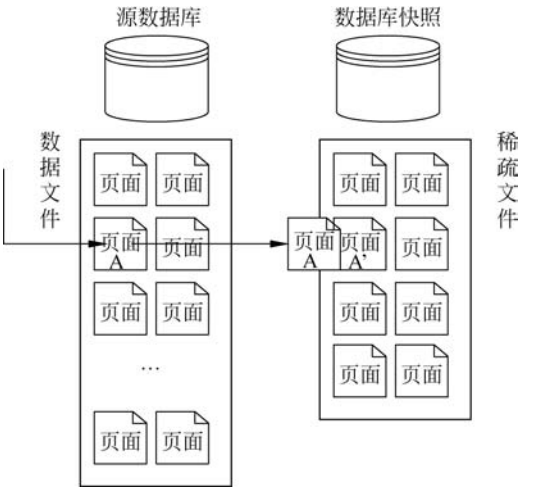

图 3-25 数据库快照过程

在 SQL Server 中，数据库快照在创建初始是一个空的数据库，并未包含源数据库中的数据，只包含一个指向源数据库页面的指针。如果此时要从空的快照数据库中读取数据，SQL Server 会从源数据库中读取需要的数据。只有当源数据库发生数据变化，如需要将新数据写到数据页时，SQL Server 才会执行复制。如图 3-25 所示，当“页面 A”的数据需要被更改时，系统会先把原“页面 A”复制一份到快照数据库文件的对应位置。源数据库中未发生变化的数据页不会被复制到快照数据库中。这种模式在数据库快照中被称为“copy-on-write”，即先将原数据页复制到快照文件中，然后再将被更改的数据写入到源数据库文件的数据页中。

并且这种“copy”，只会在源数据库第一次发生更改时执行，后续源数据库同一页再次发生更改时，就不会再次“copy”到数据库快照中。因此，这种模式也被称为“copy-on-first-write”，即“首次写时复制”，这样数据库快照确保能够反映创建快照时刻的数据库状况。

由于某一时刻源数据库更改的可能只是其中的几页数据，复制到快照文件中的也只是对应的几页，而源数据库中其他未修改的页不会被复制，因此，快照数据库的数据文件会比源数据库的数据文件小很多。所以，快照数据库的数据文件也被称为“稀疏文件”，即文件并没有被完全填满，中间会有很多空页。

“稀疏文件”会因为从源数据库中复制过来数据量的增加而逐步扩大。SQL Server 2019 会以每次 64KB(即 8 个页)的大小为“稀疏文件”分配空间。这种“稀疏文件”的存储机制可以有效地提高空间的使用效益，避免相同的数据存储两份。但是在有些场合，快照数据

文件会增长得很快，如对源数据库创建索引或重新生成索引等操作。因为索引会修改源数据库中的很多数据页，所以会造成众多数据页被复制到快照数据文件中。以下例子验证了这一现象。

如图 3-26 所示，源数据库中数据文件分配空间大小为 152MB，占用 152MB，对应的数据库快照的初始数据文件分配空间为 147MB，实际使用只有 128KB；在对源数据库中某一数据表创建索引后，数据库快照中数据文件的实际使用量扩大到 36.9MB，即有大量数据页已经复制到了数据库快照文件。

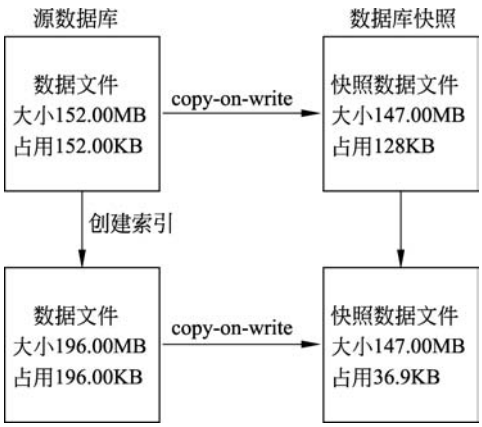

图 3-26　索引影响快照文件空间变化

使用数据库快照的原因有很多。首先，作为数据备份的一种方式，快照备份数据要比其他方式快，且占用系统资源少。其次，数据库快照可为用户保留一份可供读取的历史数据，如历年的客户订单信息、某一时刻的银行业务数据等。由于快照数据是只读的，因此，可以代替源数据库实现某些只需要查询不需要写入的数据服务。

创建数据库快照需要使用 T-SQL 语句，其语法如下。

```
CREATE DATABASE database_snapshot_name
    ON
        (
        NAME = logical_file_name,
        FILENAME = 'os_file_name'
        ) [ ,…n ]
    AS SNAPSHOT OF source_database_name
[; ]
```

各参数含义如下。

database_snapshot_name：快照数据库名称，要求不能与服务器中现有数据库名称相重。

logical_file_name：源数据库的数据文件的逻辑名称。

os_file_name：快照数据库的数据文件在操作系统中的物理名称及路径。

source_database_name：源数据库的名称。

构造实例如下。

```
use master
GO
CREATE DATABASE NetSale_snapshot
ON (NAME = NetSale_data,FILENAME = 'c:\NetSale_Data.mdf')
    AS SNAPSHOT OF NetSale
```

该实例对数据库 NetSale 创建了名称为 NetSale_snapshot 的快照，快照的主数据文件为 c:\ NetSale_Data. mdf。

创建数据快照需要注意以下事项。

(1) 如果源数据库中有多个数据文件，就需要在文件列表中列出这些文件在快照中对应文件的逻辑名称、物理名称与路径，需要一一对应。

(2) 数据库快照只有数据文件，没有事务日志文件。原因是快照是只读的，不需要记录写入操作，因此不需要事务日志文件。

(3) 快照文件必须建在 NTFS 盘区，否则会因无法创建“稀疏文件”而失败。

(4) 数据库快照与源数据库相关。数据库快照必须与源数据库在同一服务器实例上。一个源数据库可以建多个数据库快照，可以以此反映源数据库在不同时间点的状态。但是，如果源数据库因某种原因不能使用，则它的所有数据库快照也将不可用。

(5) 创建快照时，要求当前执行快照的 SQL Server 登录用户拥有对快照文件所在文件夹的写入权限，否则会出现无法创建文件的错误。具体可以把当前用户添加到文件夹的“安全”项用户组中去，并授予对文件夹的写入权限。

(6) 快照不需要时，需要手工删除。

(7) 不能对 master、tempdb、model 数据库等创建数据库快照。

3.7 数据库脚本

在 SQL Server 中，数据库对象都可以通过 T-SQL 代码生成。这一特性也为服务器间数据库或数据库对象的迁移、备份等提供了一种新的方式，即可以先将数据库中的对象和数据生成 T-SQL 的代码脚本，然后将代码脚本复制到另一服务器上去执行，以此重建原数据库的对象及数据。

SQL Server 提供的生成数据库脚本的功能，为这项操作提供了方便的途径。

3.7.1 生成数据库脚本

生成数据库脚本可以在 SSMS 进行操作，操作的过程如下。

(1) 在“对象资源管理器”窗口中，展开服务器和数据库节点，右击需要生成脚本的数据库，如本例中为 NetSale 数据库。在右键菜单中选择“任务”→“生成脚本”命令，然后在如图 3-27 所示的“生成脚本—简介”窗口中，单击“下一步”按钮。

(2) 在如图 3-28 所示的“选择对象”窗口中，可以指定需要生成的数据库对象。选择“为整个数据库及所有数据库对象编写脚本”，则表示会生成当前数据库下的表、视图、存储过程等所有对象；选择“选择具体的数据库对象”，则表示用户可以自己选择需要生成的数据库对象，而不是所有对象。本例选择“为整个数据库及所有数据库对象编写脚本”，然后单击“下一步”按钮。

图 3-27　生成脚本—简介

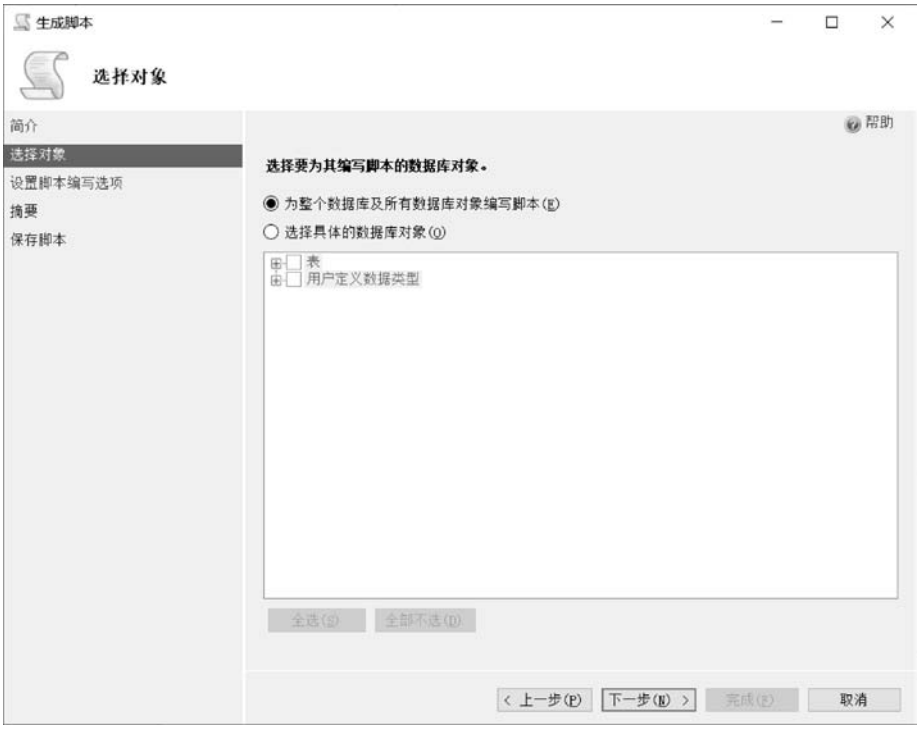

图 3-28　生成脚本—选择对象

（3）在如图 3-29 所示“设置脚本编写选项”窗口中设置生成脚本的选项，可以把生成的脚本代码保存到指定的文件；也可以存储在剪贴板中，在需要的地方按 Ctrl＋V 组合键来粘贴使用；也可以粘贴到新开的“查询编辑器”窗口中。

图 3-29　生成脚本—设置脚本编写选项

（4）为了定制生成脚本的特性，还可以单击如图 3-29 所示窗口中的“高级”按钮，在弹出的如图 3-30 所示的“高级脚本编写选项”对话框中，设置更多的高级选项。这些选项包括：表/视图选项和常规选项。例如，考虑到服务器版本的兼容问题，可以设定“服务器版脚本”，本例选择 SQL Server 2008；选择“要编写脚本的数据的类型”项为“架构和数据”，可以将数据库的数据一并生成。

图 3-30　生成脚本—高级选项

（5）完成选项设置后，单击“下一步”按钮，在如图 3-31 所示的“摘要”窗口中，确认上述操作完成的各项参数设置。如果符合要求继续单击“下一步”按钮；如果需要调整可以单击“上一步”按钮进行重新设置。

（6）最终，脚本生成完毕，如图 3-32 所示。如果生成过程中出错，就会在此窗口中给出错误信息。

图 3-31　生成脚本—摘要

图 3-32　保存脚本

3.7.2 执行数据库脚本

上述操作生成的数据库脚本，可以在目标服务器上执行，从而生成新的数据库及数据库对象。如本例是在 SQL Server 2008 中执行上述脚本，操作方式如下。

首先，通过 SSMS 连接 SQL Server 2008 服务器，然后单击工具栏中的“打开文件”按钮，打开此前生成的脚本文件，如图 3-33 所示的窗口，单击工具栏中的“执行”按钮，执行已打开的代码，完成数据库的新建和数据库对象、数据的迁移。

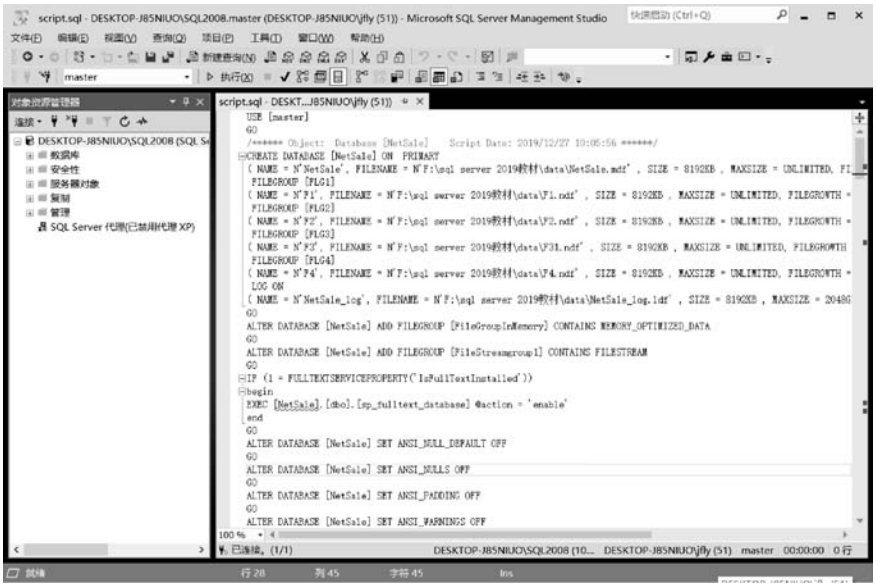

图 3-33 执行数据库脚本

小 结

本章介绍了 SQL Server 2019 中存储空间的分配方式、数据库文件与文件组、创建数据的 SQL Server Management Studio 和 T-SQL 方法、数据库的选项配置、数据库空间扩大与收缩、数据库的分离与附加、数据库脚本的生成与使用、数据库删除及快照等各项内容。

数据库是 SQL Server 的一个核心内容，今后的学习将围绕数据库展开。

习题与思考

1. SQL Server 2019 中系统数据库有哪些？这些系统数据库的作用分别是什么？

2. SQL Server 2019 中数据库文件有哪些？主数据文件和事务日志文件保存的内容有何区别？辅助数据文件的主要用途是什么？如何在数据库中新增辅助数据文件和事务日志文件？

3. 文件组的类别有哪些？文件组在 SQL Server 中的主要作用是什么？

4. 如何通过 SQL Server Management Studio 和 Create Database 语句创建数据库？

5. 在 SQL Server 2019 中数据库的状态有哪些？代表的具体含义是什么？

6. 控制数据库空间大小的途径有哪些？如何扩大和收缩数据库的大小？
7. 数据库分离操作对数据库有何影响？与删除数据库有何区别？如何附加数据库？
8. 稀疏文件的特点是什么？在 SQL Server 2019 中应用稀疏文件有何要求？
9. 什么是部分包含数据库？这一特性的作用有哪些？
10. 数据库快照的特点、用途是什么？如何创建数据库快照？