پر گردد. مقدار رشد می‌تواند به درصد بیان شود. در این صورت میزان رشد فایل بر حسب اندازه فایل محاسبه می‌گردد. مقدار رشد فایل نمی‌تواند کمتر از ۶۴ کیلوبایت باشد.

- **FOR ATTACH**، اگر بانک اطلاعاتی را با دستور Deattach جدا کرده باشید، این پارامتر برای اتصال مجدد فایل به کار می‌رود.
- **پارامتر COLLATE**، برای تعیین زبان ذخیره سازی داده از طریق Collation_name به کار می‌رود.
- **پارامتر FILEGROUP**، برای اضافه کردن گروه فایل به کار می‌رود.

برای ایجاد بانک اطلاعاتی می‌توانید از دستور CREATE DATABASE به صورت زیر استفاده کنید:

```
CREATE DATABASE database_name
ON PRIMARY ( ویژگی‌های فایل داده)
LOG ON     (ویژگی‌های فایل کارنامه)
```

database_name، نام بانک اطلاعاتی است که می‌خواهید ایجاد کنید.

مثال ۱-۱. پرس‌وجویی که بانک اطلاعاتی به نام PublishDB را ایجاد می‌کند که نام فایل داده آن publish_data.mdf است. این فایل در پوشه ClassDatabase درایو C قرار می‌گیرد. اندازه این فایل ۱۰ مگابایت و حداکثر اندازه آن ۱۵۰ مگابایت می‌باشد. نام فایل کارنامه publish_log.ldf می‌باشد که در پوشه ClassDatabase درایو C به اندازه ۱۲ و حداکثر اندازه ۱۲۰ مگابایت ایجاد می‌شود.

```
CREATE DATABASE PublishDB
ON
(NAME=publish_data,FILENAME='C:\classDatabase\publish_data.mdf',
  SIZE = 10, MAXSIZE = 100MB
 )
LOG ON
(NAME=publish_log, FILENAME ='C:\classDatabase\publish_log.ldf',
  SIZE = 12MB, MAXSIZE = 120
 )
 COLLATE Arabic_CS_AS_KS_WS
 GO
```

همان‌طورکه می‌بینید برای COLLATE مقدار Arabic_CS_AS_KS_WS انتخاب گردید تا بتوان اطلاعات فارسی را در جدول بانک اطلاعات ذخیره نمود.

مثال ۲-۱. پرس‌وجویی که بانک اطلاعاتی Library با ویژگی‌های زیر ایجاد می‌کند:

- دارای یک فایل داده به نام lib_data است. اندازه این فایل 3074KB می‌باشد که می‌تواند تا پر شدن دیسک رشد یابد و رشد فایل در هر مرحله 1024KB است. این فایل در درایو C پوشه ClassDatabase با نام lib_data.mdf قرار می‌گیرد.
- یک فایل کارنامه به نام lib_log دارد که در درایو C پوشه ClassDatabase با نام lib_log.ldf ذخیره می‌شود. اندازه این فایل نیز 3072KB است. اما حداکثر اندازه آن 2048GB می‌باشد. درصد رشد این فایل ۱۰ درصد است.

```
CREATE DATABASE Library
 ON  PRIMARY
 (NAME=N'lib_data',FILENAME=N'c:\classdatabase\lib_data.mdf' ,
```

```
    SIZE = 3072KB , MAXSIZE = UNLIMITED, FILEGROWTH = 1024KB )
LOG ON
 (NAME=N'lib_log',FILENAME=N'c:\classdatabase\lib_data.ldf' ,
   SIZE = 3072KB , MAXSIZE = 2048GB , FILEGROWTH = 10%)
GO
```

۱-۳-۱. گروه‌های فایل

در بانک اطلاعاتی می‌توان گروه‌هایی برای فایل‌ها تعریف نمود تا با قرار دادن فایل‌های داده و ایندکس‌ها روی دیسک‌های مختلف کارائی سیستم را افزایش داد. دو نوع گروه فایل در SQL Server وجود دارد که عبارت‌اند از:

۱. **گروه فایل اصلی**[1]، حاوی فایل‌های داده اصلی[2] و فایل‌هایی است که گروه فایل آن‌ها مشخص نگردید. همه صفحات[3] اختصاص داده شده به جداول سیستم در این گروه قرار می‌گیرند. هر بانک اطلاعاتی حداقل یک گروه فایل اصلی دارد.

۲. **گروه فایل‌های تعریف شده توسط کاربر**[4]، با پارامتر FILEGROUP از طریق دستورهای CREATE DATABASE و ALTER DATABASE تعریف می‌شوند. عملکرد این دستورات را در ادامه می‌بینید.

در هنگام ایجاد گروه‌های فایل به نکات زیر توجه کنید:

۱. هر بانک اطلاعاتی که ایجاد می‌کنید، یک گروه فایل پیش‌فرض برای آن ایجاد می‌گردد (گروه فایل پیش‌فرض، همان گروه فایل اصلی است).

۲. می‌توان به جای پشتیبان‌گیری[5] و ترمیم[6] کل بانک اطلاعاتی از گروه فایل پشتیبان گرفت یا گروه فایل را ترمیم نمود.

۳. گروه فایل امکان توزیع اشیا مختلف بانک اطلاعاتی را بر روی درایوهای دیسک فراهم می‌نماید.

۴. یک گروه فایل می‌تواند شامل چند فایل باشد.

۵. یک فایل بانک اطلاعاتی فقط در یک گروه فایل قرار می‌گیرد (یک فایل بانک اطلاعاتی نمی‌تواند در چند گروه فایل قرار گیرد).

۶. فایل کارنامه در هیچ گروه فایلی قرار نمی‌گیرد. زیرا، همان‌طور که بیان گردید، فایل کارنامه در فایل جداگانه‌ای قرار می‌گیرد.

۷. جداول، ایندکس‌ها و داده‌های نظیر ntext، text و Image می‌توانند به یک گروه فایل، متناظر گردند.

۲-۳-۱. تغییر خواص بانک اطلاعاتی موجود

در بخش ۱-۳-۱ روش ایجاد بانک اطلاعاتی را دیدید. گاهی نیاز است خواص فایل‌های بانک اطلاعات موجود از قبیل اندازه، حداکثر اندازه و غیره را تغییر دهید. برای این منظور می‌توانید از دستور ALTER DATABASE به صورت زیر استفاده کنید:

```
ALTER DATABASE database_name
ADD FILE (ویژگی‌های فایل داده)
```

---

[1].Primary File Group [2].Primary Data File [3].Pages [4]. User Defined File Group
[5].Backup [6]. Recovery

ADD FILEGROUP = نام گروه فایل
MODIFY FILE (ویژگی‌های فایل موجود)
MODIFY NAME = نام جدید
ADD LOG FILE(ویژگی‌های فایل کارنامه)
REMOVE FILE نام فایل منطقی
REMOVE FILEGROUP نام گروه فایل

database_name، نام بانک اطلاعاتی است که می‌خواهید ویژگی‌های آن را تغییر دهید.

مثال ۳ – ۱. پرس‌وجویی که اندازه و حداکثر اندازه فایل Publish_data از بانک اطلاعاتی PublishDB را به ترتیب به ۱۵۰ و ۲۰۰ مگابایت تغییر می‌دهد.

```
ALTER DATABASE PublishDB
MODIFY FILE (NAME= Publish_data, SIZE= 150, MAXSIZE= 200)
```

۳–۳–۱. حذف بانک اطلاعاتی موجود

همان طور که دیدید، فایل‌های داده و کارنامه بانک اطلاعات فضای روی دیسک را اشغال می‌کنند. بنابراین، اگر بانک اطلاعاتی را نیاز نداشته باشید باید آن را حذف کنید تا فضای اشغال شده توسط آن به سیستم عامل برگردد. برای حذف بانک اطلاعات دستور DROP DATABASE به صورت زیر استفاده می‌شود:

```
DROP DATABASE database_name [1,...,n];
```

database_name[1,...,n]، نام بانک‌های اطلاعاتی هستند که می‌خواهید حذف شوند. اگر بانک اطلاعاتی را حذف کنید، کلیه فایل‌های داده و کارنامه متعلق به آن حذف خواهند شد.

مثال ۴ – ۱. پرس‌وجویی که بانک اطلاعات به نام myDatabase با ویژگی‌های زیر ایجاد می‌کند:

- این بانک دارای یک فایل به نام my_data1 است که در درایو C پوشه classDatabase با نام my_data1.mdf قرار می‌گیرد. اندازه این فایل 5MB تا حداکثر 100MB به صورت ۵ مگابایت، ۵ مگابایت رشد می‌یابد.

```
CREATE DATABASE myDatabase
ON
( NAME = my_data1,
  FILENAME = 'c:\classDatabase\my_data1.mdf',
  SIZE = 5MB, MAXSIZE = 100MB, FILEGROWTH = 5MB
)
GO
```

مثال ۵ – ۱. پرس‌وجویی که فایلی به نام my_data2 در پوشه classDatabase درایو C با نام my_data2.mdf به بانک اطلاعاتی myDatabase اضافه می‌کند. اندازه این فایل ۵ مگابایت است و تا حداکثر ۱۰۰ مگابایت به صورت ۵ مگابایت، ۵ مگابایت می‌تواند رشد یابد.

```
ALTER DATABASE myDatabase
  ADD FILE( NAME = my_data2,FILENAME='c:\classDatabase
\my_data2.mdf',SIZE=5MB,MAXSIZE=100MB,FILEGROWTH = 5MB )
GO
```

مثال ۶-۱. پرس‌وجویی که گروه فایلی به نام myFileGroup را به بانک اطلاعاتی myDatabase اضافه می‌کند.

```
ALTER DATABASE myDatabase
 ADD FILEGROUP myFileGroup
GO
```

مثال ۷-۱. پرس‌وجویی که فایل my_data3 در پوشه classDatabase درایو C با نام my_data3.mdf را به گروه myFileGroup بانک اطلاعاتی myDatabase اضافه می‌کند. این فایل 4MB است که می‌تواند تا 50MB با گام 5MB رشد کند.

```
ALTER DATABASE myDatabase
ADD FILE(NAME=my_data3,FILENAME=N'c:\classdatabase
    \my_data3.mdf',SIZE=4MB,MAXSIZE=50MB,FILEGROWTH=5MB)
 TO FILEGROUP myFileGroup
 GO
```

مثال ۸-۱. پرس‌وجویی که فایل my_data2 را از بانک اطلاعاتی myDatabase حذف می‌کند.

```
ALTER DATABASE myDatabase
 REMOVE FILE my_data2
GO
```

مثال ۹-۱. پرس‌وجویی که اندازه فایل my_data1 از بانک اطلاعاتی myDatabase را به ۱۰ مگا بایت تغییر می‌دهد.

```
ALTER DATABASE myDatabase
MODIFY FILE   (NAME = my_data1, SIZE = 10MB)
GO
```

مثال ۱۰-۱. پرس‌وجویی که فایل my_data4 در پوشه classDatabase درایو C با نام my_data4.mdf را به گروه myFileGroup بانک اطلاعاتی myDatabase اضافه می‌نماید.

```
ALTER DATABASE myDatabase
ADD FILE(NAME=my_data4,FILENAME=N'c:\classdatabase\
    my_data4. mdf',SIZE=4MB,MAXSIZE=50MB,FILEGROWTH=5MB)
 TO FILEGROUP myFileGroup
 GO
```

مثال ۱۱-۱. پرس‌وجویی که گروه فایل myFileGroup1 را به myDatabase اضافه می‌کند.

```
ALTER DATABASE myDatabase
 ADD FILEGROUP myFileGroup1
GO
```

مثال ۱۲-۱. پرس‌وجویی که گروه فایل myFileGroup را از بانک اطلاعات myDatabase حذف می‌کند.

```
ALTER DATABASE myDatabase
REMOVE FILE my_data3
```

```
GO
ALTER DATABASE myDatabase
REMOVE FILE my_data4
GO
ALTER DATABASE myDatabase
REMOVE FILEGROUP myFileGroup
GO
```

همان گونه که مشاهده می‌شود، برای حذف گروه فایل ابتدا باید فایل‌های موجود در آن گروه فایل را حذف نمود، سپس گروه فایل را حذف کرد.

مثال ۱۳-۱. پرس‌وجویی که فایل my_data3 را به گروه فایل myFileGroup1 از بانک اطلاعات myDatabase اضافه می‌کند.

```
ALTER DATABASE myDatabase
ADD  FILE(NAME=my_data3,FILENAME=N'c:\classdatabase\
   my_data3 .mdf',SIZE=4MB,MAXSIZE=50MB,FILEGROWTH=5MB)
 TO FILEGROUP myFileGroup1
 GO
```

مثال ۱۴-۱. پرس‌وجویی که گروه فایل myFileGroup1 از بانک اطلاعاتی myDatabase را به گروه فایل پیش‌فرض تغییر می‌دهد.

```
ALTER DATABASE myDatabase
  MODIFY FILEGROUP myFileGroup1 DEFAULT
GO
```

مثال ۱۵-۱. پرس‌وجویی که گروه فایل PRIMARY بانک اطلاعاتی myDatabase را به گروه فایل پیش‌فرض تغییر می‌دهد.

```
ALTER DATABASE MyDatabase
  MODIFY FILEGROUP [PRIMARY] DEFAULT
GO
```

دقت کنید که PRIMARY باید بین [ ] یا " " قرار گیرد.

مثال ۱۶-۱. پرس‌وجویی که فایل my_log1 با نام my_log1.ldf درایو C پوشه classDatabase را به بانک اطلاعاتی myDatabase اضافه می‌کند (یک فایل کارنامه به بانک اطلاعاتی اضافه می‌نماید).

```
ALTER DATABASE myDatabase
ADD LOG FILE(NAME=my_log1,FILENAME='c:\classDatabase
    \my_log1.ldf',SIZE=4MB,MAXSIZE=20MB,FILEGROWTH=4MB)
GO
```

مثال ۱۷-۱. پرس‌وجویی که بانک اطلاعاتی myDatabase را به نام myDatabase1 تغییر نام می‌دهد.

```
ALTER DATABASE myDatabase
 MODIFY NAME = myDatabase1
GO
```

مثال ۱۸-۱. پرس‌وجویی که بانک اطلاعاتی myDatabase1 را حذف خواهد کرد. یعنی، تمام فایل‌های داده و کارنامه مربوط به این بانک موجود در پوشه ClassDatabase درایو C را حذف خواهد کرد.

```
DROP DATABASE myDatabase1
```

نکته: در هنگام اجرای دستورات ALTER DATABASE و DROP DATABASE، اگر بانک اطلاعاتی که می‌خواهید خواص آن را تغییر دهید یا آن را حذف نمایید، موجود نباشد، بانک اطلاعاتی SQL Server پیغام خطا صادر می‌کند. ولی، اگر در هنگام اجرای دستور CREATE DATADBASE، چنانچه بانک اطلاعاتی که می‌خواهید ایجاد کنید از قبل موجود باشد، پیغام خطا ظاهر می‌گردد.

۴-۱. ایجاد و تغییر ساختار جدول

همان‌طور که می‌دانید، جدول از مجموعه‌ای از رکوردها تشکیل می‌شود و رکورد نیز از مجموعه‌ای از فیلدها تشکیل می‌گردد و فیلدها نیز حاوی تعدادی خواص هستند. بنابراین، قبل از این که به ایجاد جدول بپردازیم، باید خواص فیلدها را بررسی کنیم. این خواص در زیر آمده‌اند:

- **نام فیلد:** هر فیلد در جدول یک نام **یکتا**[1] دارد که از قانون نام‌گذاری شناسه‌ها در بانک اطلاعات پیروی می‌کند.
- **نوع فیلد:** برای هر فیلد باید تعیین شود، اولاً چه نوع داده‌ای را ذخیره می‌کند، ثانیاً تعداد بایت‌هایی که ذخیره می‌نماید، چقدر است. انواع داده‌ها و مقدار فضایی که هر یک از انواع در بانک اطلاعات نیاز دارند، در جدول ۲- ۱ آمده است.
- **محدودیت‌های فیلد:** این ویژگی تعیین می‌کند هر فیلد چه محدودیت‌هایی (قیدهای) دارد. برخی از این محدودیت‌ها در زیر آمده‌اند:

۱. محدودیت کلید اولیه[2]: فیلدی کلید اولیه است که دارای شرایط زیر باشد (مانند شماره کارمندی، شماره دانشجویی، شماره وام، شماره مشتری، شماره درس، کد گروه و کد شهر):

- مقدار این فیلد تکراری نباشد.
- اطلاعات رکوردها بر اساس این فیلد مرتب باشند.
- مقدر این فیلد نمی‌تواند تهی[3] باشد.

مثال ۱۹-۱. دستوری که فیلد کد شهر را به عنوان کلید اولیه جدول City تعریف می‌کند.

```
PRIMARY KEY (CityID)
```

مثال ۲۰-۱. دستوری که فیلد کد مؤلف را کلید اولیه معرفی می‌کند.

```
PRIMARY KEY (atID)
```

مثال ۲۱-۱. دستوری که فیلدهای شابک کتاب و کد مؤلف را به عنوان کلید اولیه تعریف می‌کند.

```
PRIMARY KEY (ISBN, atID)
```

---

[1] .Unique [2].Primary Key [3].Null

جدول ۲-۱ انواع داده در SQL

نام	تعداد بایت	هدف
int	4	اعداد صحیح از ۲,۱۴۷,۴۸۳,۶۴۸- تا ۲,۱۴۷,۴۸۳,۶۴۷ است.
intBig	8	اعداد صحیح از $2^{63}-$ تا $2^{63}-1$ است.
binary(n)	n	داده‌های دودویی حداکثر ۲۵۵ بایت را اشغال می‌کنند.
bit	1	مقادیر ۰ یا ۱ را می‌پذیرد.
char(n)	n	حداکثر تا ۸۰۰۰ کاراکتر را می‌پذیرد و به ازای هر کاراکتر یک بایت را اشغال می‌کند.
datetime	8	تاریخ و زمان را نگهداری می‌کند.
decimal(p, s)	حداکثر 17	اعداد اعشاری یا صحیح $10^{38}+$ تا $10^{38}-$ را نگهداری می‌کند.
float(n)	8	از مقدار $1.79E^{308}-$ تا $1.79E^{308}$ را نگهداری می‌کند و دقت آن تا ۱۵ رقم است.
real(n)	4	تقریباً از مقدار $3.40E^{38}-$ تا $3.40E^{38}$ را نگهداری می‌کند (با تقریب ۷ رقم).
image		داده‌های تصویری تا 2GB را در صفحات ۸ کیلوبایتی ذخیره می‌کند.
money	8	داده‌های ارزی از ۹۲۲۳۳۷۲۰۳۶۸۵۴۷۷.۵۸۰۸- تا ۹۲۲۳۳۷۲۰۳۶۸۵۴۷۷.۵۸۰۷ است.
varchar(n)	n*2	داده‌های یونیکد از ۱ تا ۸۰۰۰ کاراکتر را نگهداری می‌کند.
varchar(max)	n*2	برای نگهداری کاراکترهای غیر یونیکد تا حداکثر ۲ گیگابایت منهای یک حرف استفاده می‌شود.
ntext	n*2	داده‌های یونیکد با طول متغیر که طول بیشتر از ۸۰۰۰ کاراکتر تا ۲ گیگابایت است.
numeric	حداکثر 17	مترادف Decimal است.
real	4	مترادف float است.
smalldatetime	4	ترکیب تاریخ و زمان با طول کوتاه را نگهداری می‌کند.
smallint	2	اعداد صحیح از ۳۲,۷۶۸- تا ۳۲,۷۶۷ را نگهداری می‌کند.
currency	8	مقادیر پولی با حداکثر ۴ رقم اعشار را نگهداری می‌کند.
sysname(n)	n*2	برای ذخیره کردن اسامی اشیای سیستم به کار می‌رود و به ازای هر کاراکتر دو بایت را اشغال می‌کند.
timestamp		برای نگهداری مهر زمانی در بانک اطلاعاتی به کار می‌رود.
tinyint(n)	n	اعداد صحیح ۰ تا ۲۵۵ بایت که n تعداد بایت‌ها را مشخص می‌کند.
varbinary(n)	n	الگوی بیتی تا ۲۵۵ بایت را نگهداری می‌کند.
uniqueidentifier	16	عدد ۱۶ بایتی هگزا دسیمال که شناسه یکتای عمومی را نشان می‌دهد.
XML	حداکثر 2GB	برای نگهداری اسناد XML به کار می‌رود.
varbinary(max)		اطلاعات بایتی را به صورتی در نظر می‌گیرد که می‌توان تصاویر را در آن ذخیره کرد.

۲. ایجاد محدودیت کلید خارجی[1]: کلید خارجی برای ایجاد ارتباط بین دو جدول به کار می‌رود. یعنی، با استفاده از کلید اولیه یا کلید فرعی یک جدول و کلید خارجی جدول دیگر می‌توان ارتباط بین این دو جدول را برقرار کرد. کلید خارجی در واقع کلید اولیه یا کلید فرعی جدول دیگر در همان بانک است. برای تعریف کلید خارجی به صورت زیر عمل می‌شود:

FOREIGN KEY (فیلد۱, ... ,فیلدn) REFERENCES نام جدول ۲

مثال ۲۲-۱. دستوری که فیلد pubID را بین جدول Publishers و PubBook به عنوان کلید خارجی تعریف می‌کند.

```
FOREIGN KEY (pubID) REFERENCES Publishers
```

وقتی که محدودیت جامعیت ارجاع نقض گردد، معمولاً عملی که موجب نقض گردیده است، رد می‌شود. یعنی، اگر در جدول GroupBook، بخواهید کد نوع کتاب ۰۱ را حذف کنید، در صورتی که این کد در جدول Books استفاده شده باشد، دستور حذف از جدول GroupBook لغو خواهد شد. اما، بخش FOREIGN KEY را می‌توان طوری تعریف کرد که تراکنش‌های حذف و به هنگام رسانی لغو نگردند. برای این منظور می‌توانید از گزینه‌های زیر استفاده کنید:

- **گزینه ON DELETE CASCADE**، موجب می‌شود تا عمل حذف لغو نگردد. ابتدا، تمام جداولی که با این جدول از طریق این فیلد ارتباط دارند، مقداری که در حال حذف است به صورت آبشاری حذف خواهد کرد. سپس، در جدول اصلی مقدار مورد نظر حذف می‌گردد. یعنی، اگر بخواهید در جدول GroupBook، کد نوع گروه کتاب ۰۱ را حذف کنید، ابتدا در جدول Books، رکوردهایی با کد نوع گروه کتاب ۰۱ حذف خواهند شد. سپس، در جدول GroupBook، رکوردی با کد نوع گروه کتاب ۰۱ حذف می‌گردد.
- **گزینه ON UPDATE CASCADE**، اگر در هنگام به روز رسانی در فیلدی که محدودیت جامعیت در آن تعریف شده باشد، این محدودیت نقض شود، به هنگام روز رسانی رد نمی‌گردد. به عنوان مثال، اگر بخواهید در جدول GroupBook، کد نوع گروه کتاب ۰۱ را به ۰۳ تغییر دهید. ابتدا، در جدول Books، کد نوع‌های گروه کتاب ۰۱ را به ۰۳ تغییر می‌دهد. سپس، در جدول GroupBook این تغییر را اعمال می‌کند.

استفاده از گزینه‌های ON DELETE CASCADE و ON UPDATE CASCADE به صورت زیر است:

```
CREATE TABLE Books
 (
   ...
   FOREIGN KEY (groupID) REFERENCES GroupBook
   ON DELETE CASCADE,
   ON UPDATE CASCADE,
   ...
 )
```

[1].Foreign Key

۳. **محدودیت DEFAULT**، این محدودیت مقداری را تعیین می‌کند که اگر کاربر برای فیلد مقدار وارد نکند، آن مقدار در این فیلد قرار می‌گیرد. یعنی، مقدار پیش‌فرض فیلد را تعیین می‌نماید.

۴. **محدودیت UNIQUE**، تضمین می‌کند صفات (فیلدهای) تعیین شده تشکیل کلید کاندید را دهند. یعنی، رکوردهای (چند تایی‌های) جدول (رابطه) **یکتا** هستند (هیچ دو رکوردی در جدول تکراری نیستند). فیلدهای با محدودیت UNIQUE می‌توانند تهی (NULL) باشند، مگر این که با محدودیت NOT NULL معرفی گردند. این محدودیت به صورت زیر به کار می‌رود:

```
UNIQUE (فیلد ۱, فیلد۲, ..., فیلد n)
```

۵. **محدودیت NOT NULL**، تضمین می‌کند مقدار فیلد در هیچ رکوردی (چند تایی) تهی نباشد. به عنوان مثال، نام کتاب نمی‌تواند تهی باشد. پس، به صورت زیر تعریف می‌گردد:

```
Title  varchar (30) NOT NULL
```

۶. **محدودیت CHECK**، یکی از رایج‌ترین محدودیت‌ها CHECK است. این محدودیت تضمین می‌کند مقادیر صفت (فیلد)، شرط تعیین شده را داشته باشد.

مثال ۲۳-۱. دستوری که تضمین می‌کند که فیلد مدرک اساتید(Ranking)، فقط یکی از مقادیر "دکتری"، "فوق‌لیسانس" یا "لیسانس" را بپذیرد.

```
CHECK (Ranking IN ('لیسانس' , 'فوق‌لیسانس' , 'دکتری'))
```

مثال ۲۴-۱. دستوری که تضمین می‌کند که مقدار نمره دانشجویان بین ۰ تا ۲۰ باشد.

```
CHECK  (Score >= 0.00  AND  Score <= 20.00)
```

۷. **محدودیت IDENTITY**، موجب می‌شود تا این فیلد مانند یک فیلد Autonumber در اکسس عمل کند. در جدول‌هایی که فیلد کلید وجود ندارد، معمولاً فیلدی از این نوع انتخاب می‌کنند تا به عنوان فیلد کلید عمل کند. این فیلد با اضافه شدن هر رکورد، به طور خودکار یک واحد به مقدار آن اضافه خواهد شد.

۱-۴-۱. ایجاد جدول

برای ایجاد جداول بانک اطلاعاتی می‌توانید از دستور CREATE TABLE به صورت زیر استفاده کنید:

```
CREATE TABLE table_name
 (Column1 type1 Constraint1,
  Column2 type2 Constraint2,
    .
    .
    .
  Columnn typen Constraintn)
```

در این ساختار table_name، نام جدولی است که می‌خواهید ایجاد کنید. $Column_1$ تا $Column_n$، نام فیلدهای (ستون‌های) جدول را تعیین می‌کنند. $type_1$ تا $type_n$، نوع ستون‌های $Column_1$ تا $Column_n$ را مشخص می‌کنند و $Constraint_1$ تا $Constraint_n$ به ترتیب محدودیت‌هایی را تعیین می‌کنند که باید بر روی

ستون‌های $Column_1$ تا $Column_n$ اعمال شوند. اگر ستونی محدودیت نداشته باشد، می‌توانید بخش Constraint آن ستون را حذف کنید.

مثال ۲۵ – ۱. پرس‌وجویی که جدول GroupBook را ایجاد می‌کند (مشخصات فیلدهای جدول GroupBook در جدول ۱–۱ آمده است).

```
USE PublishDB
GO
CREATE TABLE GroupBook
(
  groupID     varchar(6) PRIMARY KEY,
  groupName   varchar(30) NOT NULL
)
GO
```

همان‌طور که در این دستورات می‌بینید، فیلد groupID کلید اولیه تعریف شده است و groupName به صورت NOT NULL تعریف گردید تا مقدار تهی را نپذیرد.

مثال ۲۶ – ۱. پرس‌وجویی که جدول Books را ایجاد می‌کند (ساختار این جدول را در جدول ۱ – ۱ ببینید).

```
USE PublishDB
GO
CREATE TABLE Books
(
  ISBN     varchar(10) PRIMARY KEY,
  Title  varchar(40) NOT NULL,
  Page int NOT NULL,
  Price decimal(15,0) NOT NULL,
  editNo int,
  printNo int,
  groupID varchar(6) references GroupBook
)
GO
```

این دستور فیلد ISBN (شابک) را کلید اولیه تعریف می‌کند و فیلد groupID را کلید خارجی تعریف کرده تا بین جداول GroupBook و Books ارتباط برقرار کند.

مثال ۲۷ – ۱. پرس‌وجویی که جدول Publishers که ساختار آن در جدول ۱ – ۱ قرار دارد را ایجاد می‌کند.

```
CREATE TABLE Publishers
(
  pubID     varchar(10) PRIMARY KEY,
  pName  varchar(50) NOT NULL,
  Tel varchar(15),
  URL varchar(100),
  cityName varchar(50),
  bFname varchar(20) NOT NULL,
  bLname varchar(20) NOT NULL,
  countBookPrint int,
 )
GO
```

مثال ۲۸ – ۱. پرس‌وجویی که ساختار جدول Authors را ایجاد می‌کند.

```
CREATE TABLE Authors
(
  atID    varchar(10) PRIMARY KEY,
  aFName varchar(20) NOT NULL,
  aLname varchar(20) NOT NULL,
  Age int,
  Ranking varchar(30),
  Email varchar(50),
  Mobile varchar(15),
  sumPayment decimal(18, 0)
)
GO
```

مثال ۲۹ – ۱. پرس‌وجویی که جدول AutBook را ایجاد می‌کند.

```
CREATE TABLE AutBook
(
  ISBN    varchar(10) ,
  atID    varchar(10) ,
  Payment decimal(18, 0),
  PRIMARY KEY(ISBN, atID),
  CONSTRAINT FK_ISBN FOREIGN KEY(ISBN)REFERENCES Books(ISBN),
  CONSTRAINT FK_atID FOREIGN KEY (atID) REFERENCES
             Authors(atID)
)
GO
```

این دستور، دو محدودیت به نام‌های FK_ISBN و FK_atID ایجاد کرده است. محدودیت FK_ISBN برای برقراری ارتباط بین جداول Books و AutBook به کار می‌رود که فیلد ارتباط ISBN می‌باشد و محدودیت FK_atID ارتباط بین جداول Authors و AutBook را برقرار می‌کند که فیلد ارتباط atID است.

مثال ۳۰ – ۱. پرس‌وجویی که ساختار جدول PubBook را ایجاد می‌کند.

```
CREATE TABLE PubBook
(
  ISBN    varchar(10) ,
  pubID    varchar(10) ,
  Payment decimal(18, 0),
  CONSTRAINT PK_ISBN_PUBID PRIMARY KEY(ISBN, pubID),
  CONSTRAINT FK_ISBN1 FOREIGN KEY (ISBN) REFERENCES
             Books(ISBN),
  CONSTRAINT FK_pubID FOREIGN KEY (pubID) REFERENCES
                Publishers(pubID)
)
GO
```

این دستور سه محدودیت زیر را ایجاد می‌کند:

- محدودیت **PK_ISBN_PUBID**، فیلدهای ISBN و pubID را کلید اولیه این جدول تعریف می‌کند.
- محدودیت **FK_ISBN1**، از طریق فیلد atID کلید خارجی را برای ارتباط بین جداول Books و AutBook تعریف می‌کند.
- محدودیت **FK_PubID**، از طریق فیلد pubID کلید خارجی را برای ارتباط بین جداول AutBook و Publishers تعریف می‌کند.

۲-۴-۱. تغییر ساختار جدول با دستور SQL

گاهی نیاز است در جدولی که داده وجود دارد، فیلد جدیدی اضافه نموده، فیلدی را حذف نمایید یا مشخصات فیلد را تغییر دهید. برای این منظور دو روش وجود دارد: ۱. **حذف جدول موجود و ایجاد مجدد همان جدول**. در این روش کلیه اطلاعات موجود در جدول حذف خواهند شد که معقول نمی‌باشد. ۲. **استفاده از دستور ALTER TABLE**. با این دستور می‌توانید ساختار جدول را تغییر دهید، به طوری که اطلاعات قبلی در جدول باقی بماند. دستور ALTER TABLE به صورت زیر به کار می‌رود:

```
ALTER TABLE  table_name
 {
    | { ALTER COLUMN column_name <column_definition>}
    | { ADD <column_definitiON>}
    | { DROP[CONSTRAINT]Constraint_name|COLUMN column_name }
 } [ ; ]
```

پارامترهای این دستور در زیر آمده‌اند:

- **پارامتر table_name**، نام جدولی است که می‌خواهید ساختار آن را تغییر دهید.
- **پارامتر ALTER COLUMN**، خواص فیلدهای موجود را تغییر می‌دهد.
- **پارامتر ADD**، فیلدهای جدیدی به جدول اضافه می‌کند.
- **پارامتر DROP**، فیلدها یا محدودیت‌های موجود را از جدول حذف می‌کند.

مثال ۳۱-۱. دستوراتی که فیلد جدیدی به نام newColumn در جدول Authors از بانک اطلاعات فعلی (همان PublishDB) اضافه می‌کنند.

```
ALTER  TABLE Authors
   ADD newColumn varchar(10) NULL
 GO
```

مثال ۳۲-۱. پرس‌وجویی که فیلدهای newColumn1 و newColumn2 را به جدول Authors اضافه می‌کنند.

```
ALTER  TABLE Authors
  ADD newColumn1 int , newColumn2 varchar(30)
 GO
```

مثال ۳۳-۱. پرس‌وجویی که اندازه فیلد newColumn2 در جدول Authors را از ۳۰ به ۲۰ کاراکتر تغییر می‌دهد.

```
ALTER  TABLE Authors
  ALTER COLUMN newColumn2 varchar(20)
GO
```

مثال ۳۴-۱. پرس‌وجویی که فیلدهای newColumn، newColumn1 و newColumn2 را از جدول Authors حذف می‌کند.

```
ALTER  TABLE Authors
 DROP COLUMN newColumn, newColumn1, newColumn2
GO
```

مثال ۳۵ - ۱. پرس‌وجویی که جدول Test با دو فیلد x و y از نوع int را به بانک اطلاعاتی فعلی (PublishDB) اضافه می‌کند (فیلد x، مقدار تهی را نمی‌پذیرد).

```
CREATE TABLE Test  ( x int NOT NULL, y int)
GO
```

مثال ۳۶ - ۱. پرس‌وجویی که محدودیت PK_X را به جدول Test اضافه می‌کند که فیلد x را به عنوان کلید اولیه آن جدول در نظر می‌گیرد.

```
ALTER TABLE Test
  ADD CONSTRAINT PK_X PRIMARY KEY(x)
GO
```

نکته: محدودیت‌های جدول را نمی‌توانید تغییر دهید. برای تغییر محدودیت‌های جدول، ابتدا باید آن را حذف کرده، سپس آن را مجدداً ایجاد کنید. برای حذف محدودیت در دستور ALTER TABLE باید از پارامتر DROP CONSTRAINT استفاده نمود.

مثال ۳۷ - ۱. پرس‌وجویی که محدودیت FK_X را از جدول Test حذف می‌کند.

```
ALTER TABLE Test
  DROP CONSTRAINT PK_X
GO
```

۳-۴-۱. حذف جدول با دستور SQL

جدول‌های اضافی را باید از بانک اطلاعاتی حذف نمود. زیرا، از آنجایی که جداول داده‌ها را ذخیره می‌کنند، فضای زیادی از بانک اطلاعاتی را اشغال می‌نمایند. بنابراین، برای آزادسازی فضای بلااستفاده باید جداول اضافی را حذف کرد تا فضای بیشتری در اختیار بانک اطلاعاتی قرار گیرد. برای حذف جدول‌های اضافی بانک اطلاعاتی می‌توانید از دستور DROP TABLE به صورت زیر استفاده کنید:

```
DROP TABLE  table_name [1,...,n ]
```

در این دستور پارامترهای table_name[1,...,n]، لیست جداولی هستند که می‌خواهید از بانک اطلاعاتی حذف کنید. اگر تعداد جداولی که باید حذف شوند، بیش از یکی باشند، بین نام آن‌ها علامت، (کاما) قرار می‌گیرد.

مثال ۳۸ - ۱. دستوراتی که جدول Test را حذف خواهند کرد.

```
DROP TABLE Test
GO
```

نکته: اگر جدولی موجود نباشد و بخواهید آن را حذف کنید یا ساختار آن را تغییر دهید، از طرف SQL، پیغام خطا صادر می‌شود. ولی، اگر جدولی موجود باشد و بخواهید با دستور CREATE TABLE آن را مجدداً ایجاد کنید، SQL پیغام خطا مناسب را نمایش می‌دهد. با حذف جدول، ساختار جدول، داده‌های آن، ایندکس‌ها، تریگرها و محدودیت‌های تعریف شده برای آن نیز حذف خواهند شد.

۵ - ۱. مسائل حل شده

از آنجایی که نام این کتاب آزمایشگاه پایگاه داده است. لذا در این بخش یک بانک اطلاعاتی واقعی‌تر بیان گردیده، تمام مسائل حل شده در کل کتاب از این بانک اطلاعاتی استفاده می‌کند. این بانک اطلاعاتی