**بسمه تعالی**



برنامه نویسی بانک اطلاعاتی SQL Server

**نویسنده : محمد حسین فخرآوری**

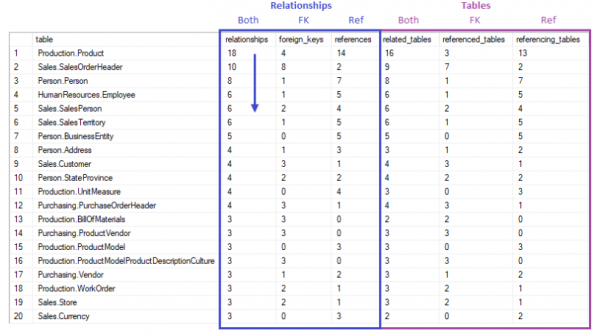
لیست جداول با بیشترین ارتباطات (Relationship)

توسط دستور زیر می‌توان لیست جداول با بیشترین ارتباط شامل FKها و ارجاعات از جداول دیگر را بدست آورد.



|  |
| --- |
| SELECT tab AS [table], COUNT(DISTINCT rel\_name) AS relationships,  COUNT(DISTINCT fk\_name) AS foreign\_keys,  COUNT(DISTINCT ref\_name) AS [references],  COUNT(DISTINCT rel\_object\_id) AS related\_tables,  COUNT(DISTINCT referenced\_object\_id) AS referenced\_tables,  COUNT(DISTINCT parent\_object\_id) AS referencing\_tables  FROM  (  SELECT SCHEMA\_NAME(tab.schema\_id)+'.'+tab.name AS tab, fk.name AS rel\_name,  fk.referenced\_object\_id AS rel\_object\_id,  fk.name AS fk\_name, fk.referenced\_object\_id, NULL AS ref\_name, NULL AS parent\_object\_id  FROM sys.tables AS tab  LEFT JOIN sys.foreign\_keys AS fk ON tab.object\_id=fk.parent\_object\_id  UNION ALL  SELECT SCHEMA\_NAME(tab.schema\_id)+'.'+tab.name AS tab, fk.name AS rel\_name,  fk.parent\_object\_id AS rel\_object\_id, NULL AS fk\_name, NULL AS referenced\_object\_id,  fk.name AS ref\_name, fk.parent\_object\_id  FROM sys.tables AS tab  LEFT JOIN sys.foreign\_keys AS fk ON tab.object\_id=fk.referenced\_object\_id  ) q  GROUP BY tab  ORDER BY COUNT(DISTINCT rel\_name) DESC; |

* **Table:** نام جدول به همراه نام schema
* **Relationships:** تعداد ارتباطات FK ها و FK reference ها
* **Foreign\_Keys:** تعداد FK های یک جدول
* **References:** تعداد FK reference ها از جداول دیگر
* **Related\_Tables:** تعداد جداول متمایز شرکت کننده در relation فارق از نوع ارتباط
* **referenced\_tables:** تعداد جداول متمایزی که جدول مورد نظر توسط FK ها به آنها رجوع کرده است (Primary Table) توجه داشته باشید که یک جدول ممکن است چندین بار توسط جدولی مورد ارجاع قرار گیرد بنابراین تعداد FK ها و تعداد جداول مورد ارجاع میتواند یکسان نباشد
* **Referencing\_Tables:** تعداد جداول متمایزی که به جدول مورد نظر ارجاع داده اند

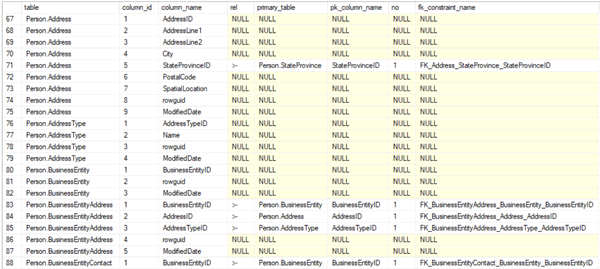


لیست ستون‌های یک جدول به همراه Foreign Key‌های آنها

دستور زیر اسامی تمامی ستون‌های جداول یک دیتابیس به همراه اطلاعات Foreign Key تعریف شده روی آن ستون (در صورت وجود) را لیست خواهد کرد.

|  |
| --- |
| SELECT  SCHEMA\_NAME(tab.schema\_id)+'.'+tab.name AS [table],  col.column\_id, col.name AS column\_name,  CASE WHEN fk.object\_id IS NOT NULL THEN '>-' ELSE NULL END AS rel,  SCHEMA\_NAME(pk\_tab.schema\_id)+'.'+pk\_tab.name AS primary\_table,  pk\_col.name AS pk\_column\_name, fk\_cols.constraint\_column\_id AS no,  fk.name AS fk\_constraint\_name  FROM sys.tables tab  JOIN sys.columns col ON col.object\_id=tab.object\_id  LEFT OUTER JOIN sys.foreign\_key\_columns fk\_cols ON fk\_cols.parent\_object\_id=tab.object\_id AND fk\_cols.parent\_column\_id=col.column\_id  LEFT OUTER JOIN sys.foreign\_keys fk ON fk.object\_id=fk\_cols.constraint\_object\_id  LEFT OUTER JOIN sys.tables pk\_tab ON pk\_tab.object\_id=fk\_cols.referenced\_object\_id  LEFT OUTER JOIN sys.columns pk\_col ON pk\_col.column\_id=fk\_cols.referenced\_column\_id AND pk\_col.object\_id=fk\_cols.referenced\_object\_id  WHERE SCHEMA\_NAME(tab.schema\_id)+'.'+tab.name=N'WareHouse.Outputs'; |

* **Table:** نام جدول به همراه نام schema
* **Column\_ID:** شماره (id) ستون
* **Column\_Name:** نام ستون
* **Rel:** نماد ارتباط , مشخص کننده Foreign Key و جهت ارتباط
* **Primary\_Table:** جدول مرجع (اصلی)
* **Pk\_Column\_Name:** نام ستون ارجاع شده (عامل ارتباط) در جدول اصلی
* **No:** شماره (id) ستون در foreign key ایجاد شده
* **Fk\_Constraint\_Name:** نام constraint مربوط به این foreign key

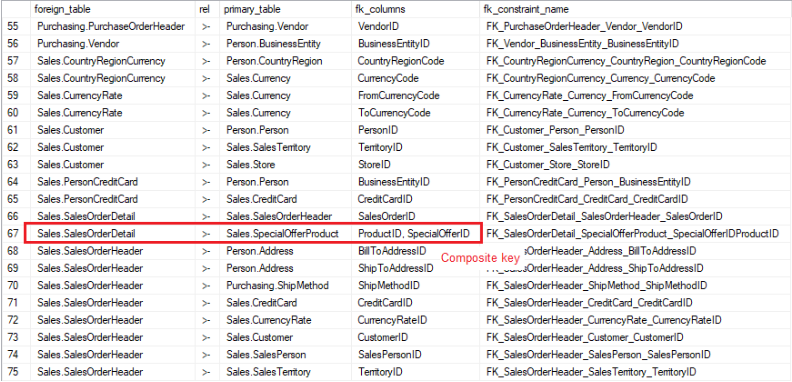


**لیست Foreign Key Constraintهای یک دیتابیس**

توسط دستور زیر می‌توان لیست Foreign Key Constraintهای تعریف شده در یک دیتابیس را بدست آورد.

|  |
| --- |
| SELECT SCHEMA\_NAME(fk\_tab.schema\_id)+'.'+fk\_tab.name AS foreign\_table,  '>-' AS rel, SCHEMA\_NAME(pk\_tab.schema\_id)+'.'+pk\_tab.name AS primary\_table,  SUBSTRING(column\_names, 1, LEN(column\_names)-1) AS [fk\_columns], fk.name AS fk\_constraint\_name  FROM sys.foreign\_keys fk  JOIN sys.tables fk\_tab ON fk\_tab.object\_id=fk.parent\_object\_id  JOIN sys.tables pk\_tab ON pk\_tab.object\_id=fk.referenced\_object\_id  CROSS APPLY  (  SELECT col.[name]+', '  FROM sys.foreign\_key\_columns fk\_c  JOIN sys.columns col ON fk\_c.parent\_object\_id=col.object\_id AND fk\_c.parent\_column\_id=col.column\_id  WHERE fk\_c.parent\_object\_id=fk\_tab.object\_id AND fk\_c.constraint\_object\_id=fk.object\_id  ORDER BY col.column\_id  FOR XML PATH('')  ) D(column\_names)  WHERE SCHEMA\_NAME(pk\_tab.schema\_id)+'.'+pk\_tab.name='Security.Users'  ORDER BY foreign\_table |

* Foreign\_Table: نام جدول دارای FK به همراه نام Schema
* Rel: نماد نشان دهنده ارتباط
* Primary\_Table: نام جدول اصلی (Referenced Table) همراه با نام Schema
* FK\_Columns: لیست اسامی ستون های FK
* FK\_Constraint\_Name: نام Foreign Key Constraint

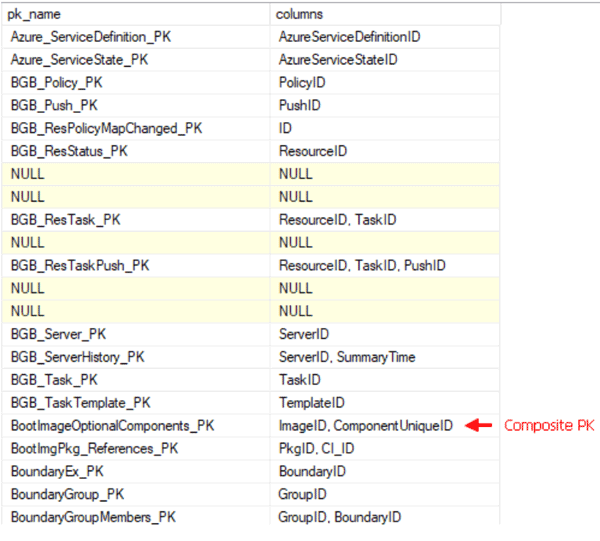


لیست جداول به همراه Primary Key آنها

توسط دستور زیر اسامی می‌توان لیست تمامی جداول به همراه نام Primary Key Constraint آنها را بدست آورد. توسط خروجی این دستور می‌توان جداول فاقد Primary Key را شناسایی کرد.

|  |
| --- |
| SELECT SCHEMA\_NAME(tab.schema\_id) AS [schema\_name], tab.[name] AS table\_name,  pk.[name] AS pk\_name, SUBSTRING(column\_names, 1, LEN(column\_names)-1) AS [columns]  FROM sys.tables tab  LEFT JOIN sys.indexes pk ON tab.object\_id=pk.object\_id AND pk.is\_primary\_key=1  CROSS APPLY(  SELECT col.[name]+', '  FROM sys.index\_columns ic  JOIN sys.columns col ON ic.object\_id=col.object\_id AND ic.column\_id=col.column\_id  WHERE ic.object\_id=tab.object\_id AND ic.index\_id=pk.index\_id  ORDER BY col.column\_id  FOR XML PATH('')  ) D(column\_names)  ORDER BY LEN(column\_names)-1, schema\_name, table\_name; |

* Schema\_Name: نام Schema
* Table\_Name: نام جدول
* Pk\_Name: نامِ Constraint مربوط به PK
* Columns: لیست اسامی ستون‌های تشکیل دهنده pk



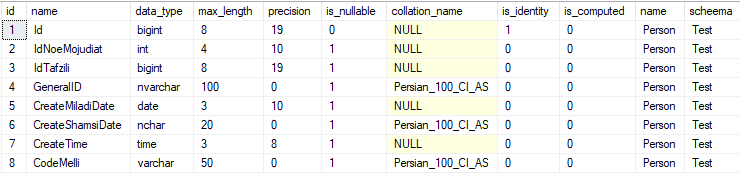
* همانطور که مشاهده می‌شود جداولی که دو ستون PK\_Name و Columns آنها فاقد مقدار باشد. جداول فاقد PK می‌باشند.

لیست تمامی ستون‌های یک جدول خاص

توسط دستور زیر می‌توان لیست تمامی ستون‌های یک جدول خاص از یک دیتابیس را بدست آورد.

|  |
| --- |
| SELECT col.column\_id AS id, col.name, t.name AS data\_type, col.max\_length,  col.precision, col.is\_nullable, col.collation\_name,  col.is\_identity, col.is\_computed, tab.name, SCHEMA\_NAME(tab.schema\_id) AS scheema  FROM sys.tables AS tab  JOIN sys.columns AS col ON tab.object\_id=col.object\_id  LEFT JOIN sys.types AS t ON col.user\_type\_id=t.user\_type\_id  WHERE SCHEMA\_NAME(tab.schema\_id)+'.'+tab.name='WareHouse.Inputs'  ORDER BY tab.name, column\_id; |

* Id: موقعیت ستون در جدول – با شروع شماره از ۱
* Name: نام ستون
* Data\_Type: نوع داده ای ستون
* Max\_Length: حداکثر طول نوع داده‌ای
* Precision: دقت نوع داده‌ای
* is\_nullable: آیا ستون قابلیت پذیرش مقدار null را دارد یا نه
* collation\_name: نوع زبانه
* is\_identity: ستون شمارشی سیستمی
* is\_computed: محاسباتی



فرض کنید یک دیتابیس آزمایشی دارید که می‌خواهید تمام رکوردهای آن‌را حذف کنید. اگر در این دیتابیس انواع و اقسام کلیدهای خارجی و تریگر و امثال آن وجود داشته باشند، صرفا با یک دستور delete ساده کار به پایان نمی‌رسد و موفق به حذف رکوردها نخواهید شد (چون این قید و بندها به همین جهت طراحی شده‌اند تا یکپارچگی دیتابیس حفظ شود).

اما اگر واقعا این قیود در این لحظه مهم نبودند و نیاز بود تا تمام رکوردها را حذف کنیم، سریعترین راه حل موجود چیست؟

|  |
| --- |
| --Disable Constraints & Triggers  exec sp\_MSforeachtable 'ALTER TABLE ? NOCHECK CONSTRAINT ALL'  exec sp\_MSforeachtable 'ALTER TABLE ? DISABLE TRIGGER ALL'  --Perform delete operation on all table for cleanup  exec sp\_MSforeachtable 'DELETE ?'  --Enable Constraints & Triggers again  exec sp\_MSforeachtable 'ALTER TABLE ? CHECK CONSTRAINT ALL'  exec sp\_MSforeachtable 'ALTER TABLE ? ENABLE TRIGGER ALL'  --Reset Identity on tables with identity column  exec sp\_MSforeachtable 'IF OBJECTPROPERTY(OBJECT\_ID(''?''), ''TableHasIdentity'') = 1 BEGIN DBCC CHECKIDENT (''?'',RESEED,0) END' |

اسکریپت فوق تمامی رکوردهای دیتابیس جاری را حذف کرده و همچنین فیلدهای identity را نیز مجددا به حالت اول باز می‌گرداند.

نمایش محتویات (Trigger - Stored procedure - FOREIGN KEY constraint - Table function - …)

|  |
| --- |
| SELECT [Scehma]=schema\_name(o.schema\_id), o.Name, o.type ,m.definition  FROM sys.sql\_modules m  JOIN sys.objects o  ON o.object\_id = m.object\_id  WHERE m.definition like '%%' |

نمایش تعداد رکورد های جدول و بررسی حجم جداول

|  |
| --- |
| USE [AdventureWorksDW2008R2];  GO  CREATE TABLE #TableSpaceUsed  (  [name] [NVARCHAR](120) NULL,  [rows] [NVARCHAR](120) NULL,  [reserved] [NVARCHAR](120) NULL,  [data] [NVARCHAR](120) NULL,  [index\_size] [NVARCHAR](120) NULL,  [unused] [NVARCHAR](120) NULL  )  INSERT INTO #TableSpaceUsed  EXEC sys.sp\_MSforeachtable 'sp\_spaceused ''?''';  SELECT \* FROM #TableSpaceUsed ORDER BY CAST([rows] AS INT) DESC;  DROP TABLE #TableSpaceUsed; |

تعداد ردیف ها و فضای استفاده شده توسط سطرهای داده (و کل فضای استفاده شده) برای همه جداول در پایگاه داده شما را لیست می کند:

|  |
| --- |
| SELECT  t.NAME AS TableName,  i.name AS indexName,  SUM(p.rows) AS RowCounts,  SUM(a.total\_pages) AS TotalPages,  SUM(a.used\_pages) AS UsedPages,  SUM(a.data\_pages) AS DataPages,  (SUM(a.total\_pages) \* 8) / 1024 AS TotalSpaceMB,  (SUM(a.used\_pages) \* 8) / 1024 AS UsedSpaceMB,  (SUM(a.data\_pages) \* 8) / 1024 AS DataSpaceMB  FROM sys.tables t  JOIN sys.indexes i ON t.OBJECT\_ID = i.object\_id  JOIN sys.partitions p ON i.object\_id = p.OBJECT\_ID AND i.index\_id = p.index\_id  JOIN sys.allocation\_units a ON p.partition\_id = a.container\_id  WHERE  t.NAME NOT LIKE 'dt%' AND  i.OBJECT\_ID > 255 AND  i.index\_id <= 1  GROUP BY t.NAME, i.object\_id, i.index\_id, i.name  ORDER BY OBJECT\_NAME(i.object\_id) |

بدست آوردن فضای آزاد درایوها

|  |
| --- |
| DECLARE @T TABLE(drive NVARCHAR(50),MBfree NVARCHAR(40))  INSERT @T  EXEC master..xp\_fixeddrives  SELECT \* FROM @T |