Learn about the architecture of MS SQL Server

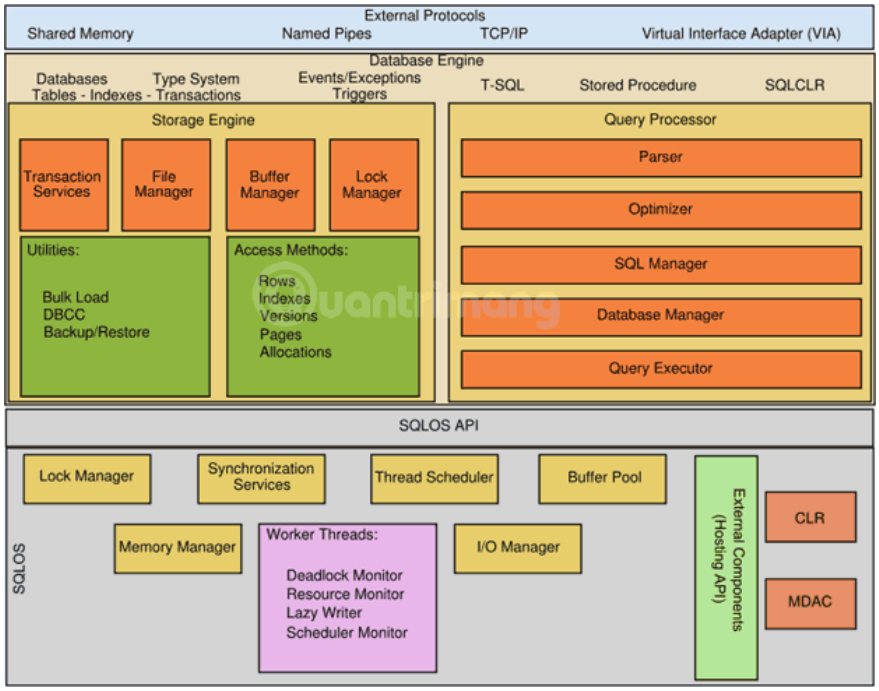
In the previous articles, you already know briefly about SQL Server, how to install SQL Server on the computer. In this section we will learn about the architecture of SQL Server.

In the previous articles, you already know briefly about SQL Server, how to install SQL Server on the computer. In this section we will learn about the architecture of SQL Server.

We will break the SQL Server architecture into the following sections to make it easier to understand:

1. General architecture - General
2. Memory architecture - Memory
3. Data file architecture - Data file
4. Log file architecture - Log file

Now we will go into the details of each type of SQL Server architecture categorized above.



General architecture - General

1. **Client: The** place where the request is made.
2. **Query:** SQL queries are high-level languages.

3. **Logical Units:** Keywords, expressions, operators, .
4. **N / W Packets:** Code related to the network.
5. **Protocols:** In SQL Server we have 4 protocols:
    1. Shared memory: For local connections and troubleshooting purposes.
    2. Named pipes: For connections on LAN.
    3. TCP / IP: For WAN connections.
    4. VIA-Virtual Interface Adapter: Special hardware requirements are set by the vendor and are not supported by the SQL 2012 version.
6. **Server:** Where SQL Services is installed and has a database.
7. **Relational Engine:** This is where real execution will be done. It contains a Query analyzer, Query optimizer and Query runtime.
8. **Query Parser (Command Parser)** and **Compiler (Translator): These** two guys are responsible for checking the syntax of the query and converting the query to the machine's language.
9. **Query Optimizer:** It will prepare output as an Execution Plan by taking the input as a query, statistics and the Algebrizer tree.

1. **Execution Plan:** Like a road map, contains the order of steps taken as part of the execution of the query.
2. **Query Executor:** This is where the query is executed step by step, with the help of the Execution Plan and also where the Storage Engine will be contacted.
3. **Storage Engine:** Responsible for storing and retrieving data in storage systems (drives, SANs, .), manipulating data, locking and managing transactions.
4. **SQL OS:** Located between the host machine (Windows OS) and SQL Server. All operations are performed on the database engine that is "taken care of" by SQL OS. SQL OS provides various operating system services, such as memory management with buffer pool, log buffer, deadlock detection, using block and lock structures.
5. **Checkpoint:** Checkpoint is an internal process, writing all modified pages (called Dirty Page) from the Buffer Cache to the physical drive. In addition, it also logs log from Log Buffer to physical file. Recording Dirty Pages into drives is also known as the Hardening of dirty pages.
6. **Lazy Writer:** Lazy Writer will push Dirty Pages and hard drives for a completely different reason, which is to free up memory in the Buffer Pool. This happens when SQL Server is running out of memory. This process is controlled by an Internal process and has no settings for it.

SQL Server constantly monitors memory usage to assess resource availability and competition, ensuring a certain amount of free space is always available. When it detects any resource conflicts, it will trigger Lazy Writer to move some Dirty Page into the drive and free up memory. It uses the Least Recently Used algorithm (LRU) to determine which page will be pushed to the hard drive. If Lazy Writer is always active, it can create bottlenecks with memory.

Memory architecture - Memory

The following are the salient features of memory architecture:

1. One of the basic design goals of all database software is to minimize disk I / O because the process of reading and writing discs is one of the most resource intensive actions.
2. Windows internal memory can be called with Virtual Address Space, shared by Kernel mode (OS mode) and User (application like SQL Server).
3. SQL Server's user address space is divided into two parts: MemToLeave and Buffer Pool.

4. The size of MemToLeave (MTL) and Buffer Pool (BPool) is decided by SQL Server during the boot process.
5. Buffer Management is an important component if you want to achieve high I / O performance. It includes two mechanisms: Buffer Manager to access and update database and Buffer Pool pages to cut I / O files into the database.
6. The Buffer Pool is divided into several parts, most importantly Buffer Cache and Procedure Cache. Buffer Cache keeps data pages in memory so that frequently accessed data can be extracted from the cache. The replacement process will read the data pages from the drive. Reading data from the cache optimizes performance by minimizing the number of I / O operations, which are slower than accessing data from memory.
7. The procedure cache holds the stored procedures and the excecution plan to optimize the number of times the execution plan is created. You can find information about capacity and operation in the Procedure Cache using the DBCC PROCCACHE command.
8. Other parts of the Buffer Pool include:
    1. System-level data structures: Contains Instance level data about databases and locks.
    2. Log Cache: Dedicated to reading and writing transaction pages.
    3. Connection Context: Each connection with Instance has a small memory area to record the current state of the connection. This information includes stored procedures and user-defined function parameters, mouse cursor location and more.
    4. Stack Space: Windows allocates stack space for each thread that starts with SQL Server.

Data file architecture - Data file

This architecture has the following components:

File Group:

Database files can be grouped together into file groups for distribution and management purposes. A file can only be a member of a file group. Log files cannot be grouped into File Group because the log file size is managed separately from the data volume.

There are two types of File Group in SQL Server as Primary and User-defined. Primary contains the main data files and any files not specifically assigned to File Group. All pages for the system table are allocated in Primary. User-defined are user-defined file groups, which are specified by using the `file group` keyword in the command to create the database or delete the database.

A File Group in each database acts as the default file group. When SQL Server assigns a page to a table or index (not in any File Group when creating), that page will be in the default file group. To convert the default file group from File Group to another File Group, db_owner fixed database role is required.

Primary is the default file group. Users need to have db_owner fixed database role to back up files and individual file groups.

File

The database has 3 types of Primary files (main data file), Secondary (secondary data file) and Log (log file). Primary is the starting point of the database and points to other files in the database.

Each database has a Primary. You can set the extension for the main data file as well, but the recommendation is to .mdf. Secondary data file is a file other than the main data file. A database may have many or only one additional data file. The extension for the extra data file should be set to .ndf.

Log files keep all the information used to recover the database. The database must have at least one log file. We can have multiple log files for a database. The extension should be set to .ldf.

The location of all files in the database is recorded in both the master database and the Primary file of the database. In most cases, the database tool uses the file location from the master database.

The file has two names: Logical and Physical. Logical is used to refer to the file in all T-SQL commands. The Physical name is OS_file_name, it must follow the operating system rules. Data files and log files can be placed on FAT or NTFS file systems, but cannot be placed on compressed file systems. There can be up to 32,767 files in a database.

Extent

Extent is a basic unit in which space is allocated for each table and index. Each Extent is 8 adjacent pages or 64KB. SQL Server has 2 types of Extent which are Uniform and Mixed. Uniform is made up of a single, Mixed object made up of up to 8 objects.

Page

Page (page) is the basic unit in SQL Server's data storage. The size of a page is 8KB. Starting each page is 96byte title, used to store system information such as page type, amount of free space on the page and ID of the page owner. There are 9 types of data pages in SQL Server:

1. Data: Data rows with all data from text, ntext and image.
2. Index: Index items.
3. TexImage: Data of text, ntext and image.
4. GAM: Information about the specified extent.
5. SGAM: Information about extent is allocated at the system level.
6. Page Free Space (PFS): Information about available free space on pages.
7. Index Allocation Map (IAM): Information about extent used by tables or indexes.
8. Bulk Changed Map (BCM): The extent information is changed by mass operation from the last backup command.
9. Differential Changed Map (DCM): Information about the extent has changed since the last database backup command.

Log file architecture - Log file

Transaction logs on SQL Server work properly when it is a sequence of log records. Each log is identified by the Log Sequence Number (LSN), which contains the ID of the transaction that it belongs to.

Log records data modifications or activities performed or retrieves images before and after data is edited. The previous image is a copy of the data before the operation is performed, the following image is a copy of the data after the operation has been performed.

The steps to recover an operation depend on the type of log.

1. Logical operation is logged.
2. To go to the previous logical operation, the operation will be performed again.
3. To return to the logic operation behind, the reverse logic operation will be performed.
4. Previous and next images are logged.
5. To go to the previous operation, the following image will be applied.
6. To return to the following operation, the previous image will be applied.

Various operations have been recorded in the transaction log. The following operations will be available there:

1. Start and end each transaction.
2. All data modifications (insert, update, delete), including changes to system storage procedures or data definition language commands (DDL) to the table, including the system table.
3. All extent and allocation, cancellation of page allocation.
4. Create or delete tables and indexes.

Rollback operations are also logged. Each transaction will hold a space in the log to make sure there is enough log space needed for rollback to execute the command or error message. This space will be released when the transaction is completed.

The part of the log file from the first log (required to successfully restore the entire database) to the last log is called the log activity section or the active log. This is the required log section for full database recovery. No part of the active log is garbled. LSN of the first log record is called the minimum recovery LSN (Min LSN).

SQL Server Database Engine divides each Physical log file into several virtual log files. The virtual log file is not fixed in size and does not have a fixed number of virtual log files for each log file Physical.

Database Engine selects the size of the virtual log file automatically when it creates or extends the log file. Database Engine tries to maintain a small number of virtual files. The size of the virtual log file cannot be configured or set up by an administrator. Only when the log Physical file is defined in small size and value growth_increment, does the virtual log file affect the system performance.

The size value is the initial size for the log file and growth_increment is the amount of space added to the file each time the file requires new space. When the log file reaches a large size because there are many small increments, they will have many virtual log files. This can slow down the database startup and log backup and recovery operations.

The advice is that you should assign the log file the size value close to the required final size and the relatively large growth_increment value. SQL Server uses write-ahead log (WAL), ensuring that there is no modification to the data written to the drive before the related log is written to the drive. This helps maintain the ACID for transaction properties.

Algebrizer in SQL

I want to talk about Algebrizer a bit: The Algebrizer is a process in the process of query execution. It started to work after Parser. When Query Parser finds a correct query, it will go to the Algebrizer and the Algebrizer's work begins:. The Algebrizer is responsible for verifying the objects and column names (that you provided in the query or being referenced by the query). For example, if the column name is mistakenly written in the query, the

Algebrizer is responsible for confirming it and creating an error. Algebrizer also identifies all types of data being processed in a given query. Algebrizer verifies whether GROUP BY and whether the combined columns are placed in the right place. For example, if you write the following query and only press Ctrl + F5 to parse, there will be no errors. But if you press F5 to run the query, the Algebrizer will work and return an error.

```
USE AdventureWorks
GO
SELECT MakeFlag,SUM(ListPrice)
FROM Production.Product
 GROUP BY ProductNumber
```

Checkpoint in SQL Server

There are 4 types of checkpoint in SQL Server 2012:

1. **Automatic:** This type is the most common checkpoint, running in the form of a background process to make sure the SQL Server Database can be restored within the time specified by Recovery Interval in the Server Configuration Option.
2. **Indirect:** This checkpoint is only available on SQL Server 2012. It is also a background running process but only for certain users who specify the recovery time for a specific database in configuration options. When the Target_Recovery_Time for a specific database is selected, it will overwrite the Recovery Interval assigned to the server, avoiding the Automatic checkpoint on the database.
3. **Manual:** This Checkpoint runs like any other SQL command, when you create the checkpoint command, it will run to complete. This checkpoint only runs on the current database. You can specify Checkpoint_Duration in the option to specify how long you want the checkpoint to be completed.
4. **Internal:** As a user, you cannot control this type of checkpoint in specific actions such as:
    1. Shut off a checkpoint action on all databases unless the shutdown fails, not normal (use Shutdown with nowith command).
    2. When the recovery model was changed from FullBulk-logged to Simple.
    3. While backing up the database.
    4. If the database is in Simple recovery mode, the checkpoint process will automatically perform or when the log is full 70%, or based on the Server's Recovery Interval option.
    5. The ALTER DATABASE command to add or delete log / data files also initiates a checkpoint.
    6. Checkpoint also takes place when the database recovery model is Bulk-logged and the minimum write operation is performed.

This is probably the "sweetest" part of SQL Server, but the idea of ??understanding its architecture will help to understand how everything works, if there is an error, it is somewhat, . This makes it easier to work with the database.

In the next section, we will learn about Management Studio and gradually go into the basic commands of SQL Server.

Previous article: Instructions for installing MS SQL Server

Next lesson: Manage MS SQL Server with Management Studio

articles on tips and guides. Thank you for reading and for following us regularly.