Администрирование БД

Системные и пользовательские БД SQL Server 2000

Архитектура базы данных

- На физическом уровне каждая БД SQL Server 2000 хранится как минимум в двух файлах:
  - **Файл данных** (data file)
  - **Файл журнала транзакций** (transaction log file)
- Каждая БД имеет один основной файл данных и может иметь дополнительные файлы данных, каждый из которых используется только этой БД.
- Основной файл имеет, как правило, расширение .mdf. Основной файл содержит данные таблиц и индексов, а также служебную информацию, которая используется SQL Server при работе. Он содержит системные таблицы, где хранится описание объектов БД, информацию о расположении дополнительных файлов БД.
- Дополнительные файлы имеют, как правило, расширение .ndf. Дополнительные файлы используются, например, для размещения БД на различных дисках.

Архитектура БД

- SQL Server 2000 хранит информацию о расположении всех файлов БД в двух местах:
  - в системной БД master
  - в основном файле БД
- Сервер в основном использует информацию из БД master. Основной файл используется в случае, когда информация в БД master должна быть изменена – при обновлении сервера, при восстановлении БД master и добавлении БД к серверу при помощи хранимой процедуры sp_attach_db.

Файлы БД

- Каждый файл данных имеет *логическое имя*, используемое при обработке операторов Transact-SQL и *физическое имя*, используемое Windows.
- Логическое имя должно быть уникально для отдельной БД и должно соответствовать правилам идентификации объектов SQL Server.
- Физическое имя должно удовлетворять требованиям именования файлов в операционной системе.

Файлы БД

- Дополнительными свойствами файла являются *идентификатор начальный размер, величина приращения при увеличении и максимальный размер*.
- Эти данные хранятся на *странице заголовка файла*, которая является первой страницей любого файла данных.
- SQL Server уникально идентифицирует страницы по номеру файла и номеру страницы. Для нумерации страниц используются последовательные номера, начиная с 0.

Выделение пространства для таблиц и индексов

- Для обеспечения хранения информации необходимо выделить свободное место в файле данных и присовоить его соответствующему объекту.
- Свободное пространство, выделяемое для таблиц и индексов, измеряется в *экстентах* .
- Экстент – блок размером 64 кбайта и состоит из 8 страниц по 8 Кб каждая.
- Экстенты бывают однородные и смешанные.

Выделение пространства для таблиц и индексов

- При создании новой таблицы или индекса SQL Server находит смешанный экстент, содержащий свободную страницу и связывает ее с созданным объектом.
- Страница содержит данные, относящиеся только к одному объекту. Если объекту требуется дополнительное пространство, то SQL Server распределяет свободное пространство из смешанных экстентов, пока необходимо число страниц не достигнет 8, затем выделяется однородный экстент для данного объекта.
- Если в файлах данных нет свободного пространства и разрешено их автоматическое увеличение, то SQL Server будет увеличивать их размер по алгоритму циклического обслуживания.

Таблица Index Allocation Map

- При размещении страницы объекта в однородном или смешанном экстенте используется таблица IAM, чтобы отследить все страницы, выделенные таблице или индексу.
- Каждая страница IAM может указывать на 512000 страниц. Страницы IAM расположены в файле случайным образом и связаны друг с другом: первая страница IAM указывает на вторую и т.д.
- Другие системные страницы:
  - PFS – страница свободного пространства страниц. Если необходимо добавить данные в страницы объекта используется PFS для определения страниц со свободным пространством.
  - GAM – страница глобальной карты размещений (используется для выделения экстента из свободного пространства). Страница GAM – это битовая карта 64000 экстантов, каждому из которых соответствует 1 бит.
  - SGAM – страница дополнительной глобальной карты размещений (используется для выделения пространства в смешанном экстенте). Каждая страница SGAM – это битовая карта 64000 экстентов, используемая SQL Server для определения смешанных экстентов со свободными страницами.

Хранение страниц индексов и страниц данных

- При отсутствии индекса SQL Server хранит данные на любой незаполненной странице в любом доступном экстенте, принадлежащем таблице.
- Такое неорганизованное хранение данных называется *кучей (heap)*. Страницы в куче хранятся беспорядочно и никак не связаны друг с другом.
- Если на таблице не определены индексы, то для поиска записи SQL Server просматривает всю таблицу (для определения страниц таблицы используется страницы IAM).
- Для ускорения поиска используются *индексы* для быстрого поиска данных. Каждый индекс хранит значение индексированного столбца таблицы в виде B-дерева.
- В SQL Server 2000 используются два вида индексов:
  - кластерный;
  - некластерный.

Кластерный индекс

- Для кластерного индекса сервер физически сортирует страницы в файле данных исходя из значений ключа кластерного индекса.
- Самый нижний уровень кластерного уровня представляет собой уровень данных.
- При размещении новой строки в таблицу SQL Server 2000 просматривает двоичное дерево, определяя место размещение (и перемещая существующие строки данных и индексов).

Некластерный индекс

- Уровень листьев (нижний уровень) некластерного индекса содержит указатель, с помощью которого SQL Server определяет, где найти строку данных, соответствующую значению ключа индекса.
- При добавлении новой строки, новая строка индекса добавляется в B-дерево, а новая строка данных помещается в любую страницу кучи.

Файлы журнала транзакций

- Каждая БД содержит хотя бы один файл журнала транзакций.
- Журнал транзакций хранит записи обо всех изменениях БД и содержит информацию необходимую для отмены изменений или их выполнений повторно.
- Каждый файл транзакций – отдельный файл операционной системы, используемый только одной БД, и имеет по умолчанию расширение .ldf.
- Каждый журнал транзакций имеет логическое имя, используемое в предложениях Transact-SQL, и физическое.
- К дополнительным свойствами файла транзакций относятся:
  - Идентификатор;
  - Начальный размер;
  - Приращение, при увеличении размера;
  - Максимальный размер.

Структура файла транзакций

- Файл журнала транзакций содержит последовательность записей.
- Каждая запись имеет *порядковый номер в журнале* (Log Sequence Number, LSN). Журнал транзакций БД SQL Server рассматривает как единый журнал, безотносительно в скольких физических файлах он располагается.
- SQL Server 2000 логически делит каждый физический файл журнала транзакций на несколько *виртуальных файлов журнала* (VLF). Количество виртуальных файлов определяется динамически в зависимости от размера физического файла.

Принцип работы журнала транзакций

- SQL Server 2000 использует буферный кэш – хранящуюся в оперативной памяти структуру, куда перемещаются страницы данных с диска. Приложения работают с данными, расположенными в кэше.
- Измененная страница в кэше, еще не записанная на диск, называется *затронутой страницей* (dirty page).
- Изменения вносятся вначале в журнал транзакций, а потом на диск. За выполнением данной процедуры следят встроенные механизмы SQL Server 2000.
- Запись затронутой страницы из кэша на диск называется *сбросом страниц* (page flushing).
- Запись журнала содержит всю информацию, необходимую для отмены или повторения любого изменения БД.
- SQL Server 2000 периодически записывает затронутые страницы из кэша на диск. Запись данных производится в результате процесса – *контрольная точка*, либо поток *отложенной записи* просматривает затронутые страницы, записывает их на диск и освобождает пространство для новых страниц.
- Если транзакция отменена или не завершена, журнал транзакций гарантирует, что все изменения внесенные незавершенными транзакциями будут отменены.

Процесс контрольной точки

- Процесс контрольной точки используется для оптимизации использования буферного кэша, уменьшения простоя в случае сбоя в работе кэша.
- Процесс контрольной точки выполняется в следующих случаях:
  - если используется оператор CHECKPOINT
  - если используется оператор ALTER DATABASE
  - при корректном завершении работы экземпляра SQL Server
  - при автоматическом запуске процесса контрольной точки (выполняется периодически в зависимости от числа записей в активной части журнала).
- Процесс контрольной точки записывает наименьший LSN, необходимый для отмены невыполненной транзакции – минимальный LSN (minLSN).
- При запуске SQL Server начинается процесс восстановления каждой БД. Проверяются журналы транзакций на наличие незавершенных транзакций. MinLSN указывает наименьший LSN, с которого начинается просмотр.

Процесс контрольной точки

- Записи с номерами транзакций меньше minLSN не являются активными. Для уменьшения дискового пространства занятого журналом они могут быть удалены из файла журнала. Минимальной единицей, которая удаляется при усечении файла транзакций – виртуальный файл журнала.
- При использовании *простой модели восстановления* процесс контрольной точки просто удаляет неактивную часть из всех виртуальных файлов и потом они используются повторно.
- При использовании *модели восстановления отдельных операций* или *модели восстановления результатов импорта* необходимо предварительно создать копию журнала транзакций, а потом удалить неактивную часть.
- Процесс контрольной точки освобождает пространство в физическом файле журнала транзакций.

Потоки операционной системы

- SQL Server использует *рабочий поток* и *поток отложенной записи* для периодического обращения к буферу памяти, планирования асинхронной записи на диск затронутых страниц и освобождения неактивных страниц.
- Рабочие потоки являются потоками других процессов, таких как процесс асинхронного чтения данных, инициированных пользователем или приложениям. Таки потоки обращаются к буферу, ожидая выполнения своего задания.
- Поток отложенной записи периодически проверяет список свободных буферов и, если его размер меньше заданного, он сканирует буферный кэш и освобождает пространство.

Модели восстановления

- В SQL Server 2000 существует три модели восстановления:
  - простая (Simple model)
  - отдельных операций (Full model);
  - результатов импорта (Bulk-Logged model).
- Каждая БД имеет одну из моделей. Выбор модели влияет на размер журнала транзакций, а также возможности резервного копирования и восстановления данных.

Модель восстановления отдельных операций

- Данная модель позволяет восстановить БД до того состояния, в котором она была на момент сбоя или на любой иной указанный момент времени.
- При использовании данной модели в журнал заносятся все операции (в том числе широкомасштабные – такие как, операция создания индекса, копирования данных). Такие операции требуют значительного объема доступного пространства в журнале транзакций.
- При выборе такой модели необходимо регулярно и часто выполнять резервное копирование журнала транзакций.

Модель восстановления результатов импорта

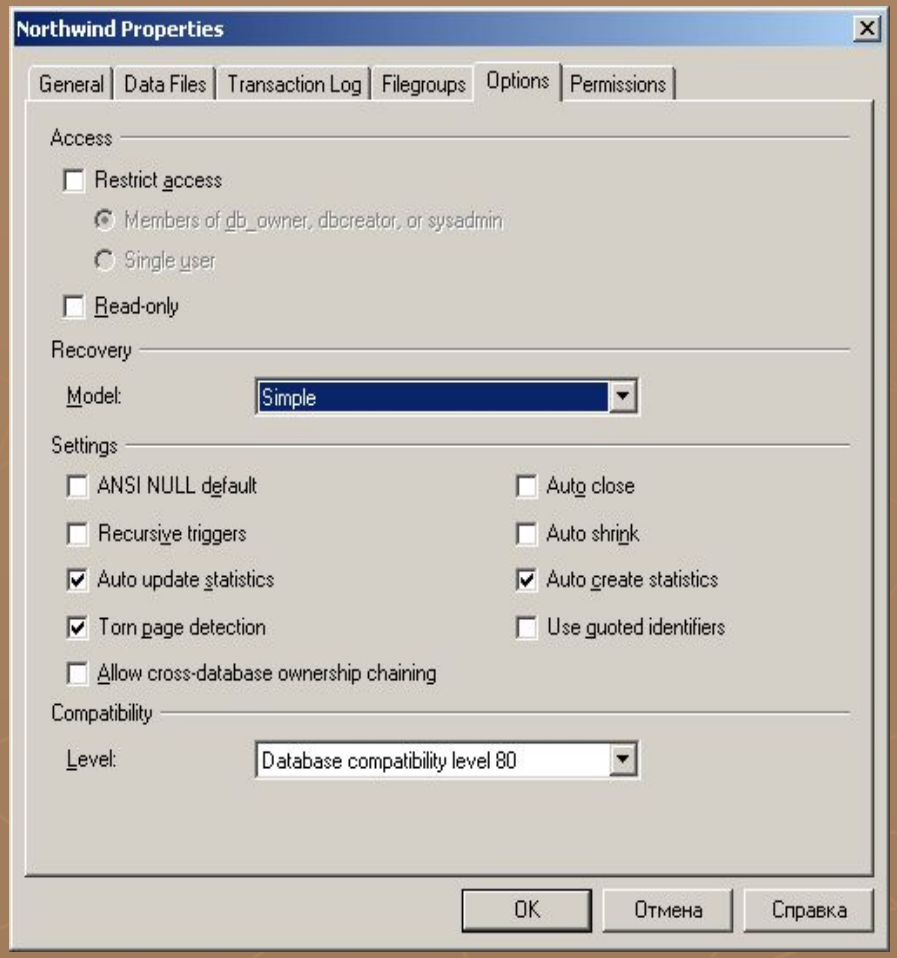
- При использовании данной модели регистрируются все операции, кроме широкомасштабных. Хранимой информации о широкомасштабных операциях недостаточно для восстановления после сбоев. Модель не поддерживает восстановление на выбранный момент времени.
- Данная модель позволяет сократить место для журнала транзакций.
- При выборе данной модели также необходимо периодически выполнять резервное копирование журнала транзакций.

Простая модель

- При использовании простой модели восстановления в журнал транзакций записываются все операции, в том числе и широкомасштабные.
- Но для восстановления данных на используется резервное копирование журнала транзакций, каждый процесс контрольной точки усекает файл журнала. Это позволяет избежать переполнения журнала транзакций, но при длинных транзакциях возможно переполнение журнала.
- При использовании простой модели БД может быть восстановлена только до момента, когда была сделана последняя резервная копия.
- Системная БД tempbd всегда использует простую модель. Пользовательские БД используют простую модель по умолчанию, однако данный режим можно изменить.

Выбор модели восстановления

- С помощью MS SQL Server Enterprise Manager для выбранной БД можно установить режим восстановления



Системные таблицы SQL Server

- SQL Server использует системные таблицы для управления работой СУБД и связанными с ней таблицами.
- Системные таблицы делятся на две группы:
  - системный каталог;
  - каталог базы данных.
- Изменение или удаление системных таблиц повлечь за собой неполадки в работе SQL Server.

Системный каталог

- Системный каталог включает в себя системные таблицы, используемые СУБД для управления системой.
- Системный каталог существует только в БД master.
- Системный каталог содержит метаданные, относящиеся ко всему экземпляру SQL Server и параметры системы.

Таблицы системного каталога

Системная таблица	Описание
Sysaltfiles	Содержит строку информации о каждом файле БД
Sysconfigures	Содержит строку информации о системных параметрах, выбранных до запуска SQL Server и динамически настроенных после запуска
Sysdatabases	Содержит строку информации для каждой БД
Sysdevices	Содержит строку о каждом созданном устройстве резервного копирования, включая логические и физические имена файлов
Syslockinfo	Содержит строку информации о каждом запросе на блокировку объекта БД
Syslogins	Содержит строку информации о каждой учетной записи
Sysmessages	Содержит строку информации о каждой возвращаемой системной ошибке
Sysperfinfo	Содержит строку информации для каждого счетчика производительности

Каталог базы данных

- Каталог базы данных состоит из системных таблиц, используемых для управления отдельной БД. В каждой БД имеется набор таких таблиц.

Системная таблица	**Описание**
Syscomments	Содержит строку информации для каждого представления (view), правила (rules), значения по умолчанию и др. объектов. Изменять или удалять значения в полях данной таблицы нельзя.
Sysindexes	Содержит строку информации о каждом индексе и таблице БД
Sysobjects	Содержит строку информации для каждого объекта БД
Sysusers	Содержит информацию о каждом пользователе или группе пользователей БД

Системные хранимые процедуры

- Системные хранимые процедуры – процедуры составленные из операторов T-SQL, поставляемые вместе с SQL Server.
- Системные процедуры работают с системными таблицами, позволяют получить системную информацию и выполнять задачи администрирования БД.

Использование системных хранимых процедур

Системная хранимая процедура	Описание
SP_configure [‘name’,’value’]	Получение и изменение параметров настройки экземпляра SQL Server
SP_dboption [‘db’, ‘parameter’, ‘value’]	Получение и изменение значений параметров БД
SP_help [‘object’]	Получение информации об отдельном объекте БД
SP_depends [‘object’]	Получение информации о зависимости объектов БД
SP_helpdb [‘db’]	Получение информации о заданной БД (всех БД)
SP_helpfile [‘file name’]	Получение физического имя и атрибутов указанного файла
SP_lock [‘id1’, , ‘id2’]	Вывод информации о текущих блокировках
SP_monitor	Вывод информации о занятости SQL Server с начала его работы
SP_spaceused [‘object’, ‘updateusage’]	Вывод информации о числе записей, дисковом пространстве и т.д.
SP_statistics [‘table’, ‘owner’, ‘id’, ‘index’]	Вывод данных обо всех индексах и информацию о таблицах
SP_who [‘user’]	Вывод информации обо всех пользователях и процессах

Системные функции

- Системные функции – набор встроенных функций, позволяющих обращаться к системным таблицам при помощи операторов T-SQL и получать специфические данные о значениях параметров, объектах и настройках.

Системная функция	**Описание**
DATABASEPROPERTYEX (‘db’,’property’)	Возвращает значения свойства БД
DB_ID(‘db’)	Возвращает идентификатор БД
DB_NAME(id_bd)	Возвращает имя БД
FILE_ID(‘file name’)	Возвращает идентификатор файла
FILE_NAME(id_file)	Возвращает логическое имя файла
FILEPROPERTY(‘file’,’property’)	Возвращает значение свойства файла
GET DATE()	Возвращает текущую системную дату
HOST_NAME()	Возвращает имя хоста
STATS_DATE(id_table,id_ind)	Возвращает дату обновления информации для индекса
USER_ID(‘user_name’)	Возвращает идентификатор пользователя
USER_NAME(id_user)	Возвращает имя пользователя БД

Представление информационной схемы

- Представления информационной схемы – сведения системного каталога и каталога БД в соответствии со стандартами ANSI SQL-92.

Представление информационной схемы	**Описание**
Information_schema.columns	Возвращает список полей, доступных текущему пользователю
Information_schema.schemata	Возвращает список БД, доступных текущему пользователю
Information_schema.tables	Возвращает список таблиц текущей БД, доступных текущему пользователю
Information_schema.table_privileges	Возвращает список привилегий, предоставленных текущему пользоватлею
Information_schema.view_table_usage	Возвращает список таблиц текущей БД, используемых в проекциях