**ЗАДАНИЕ**

Формируем документ, в котором описываем  
- прогноз по возможному росту базы

* описываем возможные угрозы и методы защиты от них
* репликации
* кластеризации

**РЕШЕНИЕ**

Ниже прогноз роста БД на год

Основные предположения:

У нас небольшой интернет-магазин, который производит доставку в пределах Москвы (или самовывоз).

Для прогноза роста данных необходимо понимать сколько места сейчас занимают таблицы в базе и какие планы по развитию магазина имеет руководство.

Планы руководства:

К концу года клиентская база вырастет до 5 000 активных клиентов. Активный клиент делает как минимум 1 заказ в 2 месяца. Соответственно к концу года в базе каждый месяц будет совершаться 2 500 заказов. (Всего клиентов будет 50 000 – активных и неактивных)

В заказах на поставку, в списке дистрибьютеров и производителях сильных изменений не будет. 3 поставки в неделю, итого 144 поставки в году.

На сайте очень мало оставляют рецензии. Руководство хочет изменить эту ситуацию, чтобы с каждого 10-го заказа была рецензия.

Линейка предоставляемой продукции должна быть увеличена в 10 раз к концу года.

Что нужно понимать еще для расчета:

Сколько в среднем атрибутов у каждого продукта. Ответ: 7

Сколько в среднем товаров в заказе клиента. Ответ: 2 продукта

Сколько в среднем товаров в заказе поставку. Ответ: 3 продукта

Для того чтобы посмотреть, сколько места занимает база – выполняем следующий скрипт:

declare @table nvarchar(128)

declare @sql nvarchar(max)

set @sql = ''

DECLARE tableCursor CURSOR FOR

SELECT name from sys.tables

open tableCursor

fetch next from tableCursor into @table

CREATE TABLE #TempTable( Tablename nvarchar(max), Bytes int, RowCnt int)

WHILE @@FETCH\_STATUS = 0

begin

set @sql = 'insert into #TempTable (Tablename, Bytes, RowCnt) '

set @sql = @sql + 'select '''+@table+''' "Table", sum(t.rowsize) "Bytes", count(\*) "RowCnt" from (select (0'

select @sql = @sql + ' + isnull(datalength([' + name + ']), 1) '

from sys.columns where object\_id = object\_id(@table)

set @sql = @sql + ') as rowsize from ' + @table + ' ) t '

exec (@sql)

FETCH NEXT FROM tableCursor INTO @table

end

PRINT @sql

CLOSE tableCursor

DEALLOCATE tableCursor

select \* from #TempTable

select sum(bytes) "Sum" from #TempTable

DROP TABLE #TempTable

Таблицы в БД, которые будут показывать наибольший рост:

1. Таблица заказов – orders
2. Тaблица – supply
3. Таблица заказов items\_in\_order
4. Таблица заказанных продуктов на поставку products\_in\_supply\_order
5. attribute\_value – таблица со значениями атрибутов продуктов
6. Таблица клиентов сlient
7. review - таблица с отзывами по продуктам

Ниже таблица, которая показывает, сколько занимает каждая таблица и сколько строк она содержит, а также соотношение байт на строку

|  |  |  |  |
| --- | --- | --- | --- |
| **Table Name** | **Bytes** | **RowCnt** | **Row/Bytes** |
| client | 38 989 | 247 | 157,85 |
| orders | 5 637 | 169 | 33,35 |
| distributor\_contact | 5 280 | 40 | 132 |
| items\_in\_order | 4 444 | 202 | 22 |
| attribute\_value | 2 475 | 116 | 21,33 |
| attributes | 1 439 | 30 | 47,96 |
| review | 1 354 | 5 | 270,8 |
| product | 1 352 | 17 | 79,52 |
| attributes\_check | 1 211 | 61 | 19,85 |
| distributor | 1 030 | 10 | 103 |
| price\_change | 1 014 | 39 | 26 |
| products\_in\_supply\_order | 1 008 | 56 | 18 |
| responsible\_courier | 869 | 15 | 57,93 |
| supply | 843 | 32 | 26,34 |
| product\_promotion | 828 | 16 | 51,75 |
| promo\_planned | 628 | 12 | 52,33 |
| product\_hierarcy | 616 | 15 | 41,06 |
| type\_attributes\_list | 546 | 42 | 13 |
| updated\_new\_distributors | 534 | 5 | 106,8 |
| manufacturer | 516 | 6 | 86 |
| prod\_type\_minimal | 378 | 8 | 47,25 |
| delivery\_status | 103 | 5 | 20,6 |
| attribute\_type\_technical | 67 | 5 | 13,4 |
| payment\_status | 63 | 3 | 21 |
| supply\_status\_id | 51 | 3 | 17 |
| attribute\_type\_general | 31 | 3 | 10,33 |
| **Итог** | **71 306** | **1 162** | **61,36** |

Теперь у нас все готово для расчета. С учетом предположений и планов руководства выходит следующая цифра:

|  |  |  |  |
| --- | --- | --- | --- |
| **Tablename** | **Bytes** | **RowCnt** | **Комментарий** |
| product | 13 520 | 170 |  |
| client | 7 931 499 | 50 247 | Желаемое руководством кол-во клиентов + текущее кол-во клиентов |
| supply | 3 846 | 146 | 3 поставки в неделю + текущее кол-во поставок |
| orders | 505 962 | 15 169 | в конце года должно быть 2500 заказов в месяц (в прогнозе учтен постепенный рост числа заказов по месяцам) |
| items\_in\_order | 667 436 | 30 338 | В среднем 2 продукта в заказе |
| products\_in\_supply\_order | 7 884 | 438 |  |
| attribute\_value | 25 390 | 1 190 | В среднем 7 значений атрибутов на один продукт |
| review | 429 489 | 1 586 | в прогнозе учтен постепенный рост числа заказов по месяцам |
| Остальные таблицы | 16724,4 |  | Допустим, что мелкие таблицы с редкими инсертами тоже увеличатся на 10% |
| **Всего** | **9 601 751** |  |  |

Итого в конце года БД будет занимать **9,6 мб**

Каждый день в ночное время будет делаться горящий бекап БД. Допустим будут храниться все бекапы за каждую прошедшую неделю и бекапы по каждому дню за последний месяц. И того в год набегает 9,6 \* 48 + 9,6 \* 30 = 748 мб

Также будет настроена репликация Master – Slave, это так же потребует памяти.

* **описываем возможные угрозы и методы защиты от них**

**Перечисление основных угроз:**

1. Потеря данных в следствие неисправности диска

Решение: создание бекапов базы данных, на отдельном хранилище с частотой 1 день либо чаще, если позволяет текущая нагрузка на БД

1. Медленная работа базы

Решение: Хранение индексов на отдельном быстром носителе

Хисторизациия данных прошлых лет на отдельном носителе

1. Внесены некорректные данные БД, данные успели отзеркалиться в реплику

Использование бекапов для восстановления

1. Потеря данных вследствие природного катаклизма/пожара или иного ЧП

Использование облачных ресурсов для хранения бекапов, благо размер БД позволяет это делать.