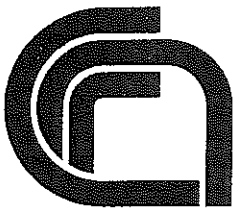


Consiglio Nazionale delle Ricerche

Distribuzione dei dati, ottimizzazione e monitoraggio delle prestazioni in Microsoft SQL Server 2000

Nicola Aloia

Rapporto
CNUCE-B4-2002-006

CNUCE

Pisa

Distribuzione dei dati, ottimizzazione e monitoraggio delle prestazioni in SQL Server 2000

Nicola Aloia

CNUCE - Istituto del C.N.R.

Indice

1 Introduzione

Il crescente sviluppo di applicazioni Web che utilizzano grandi quantità di dati, richiede attenzione ancora maggiore alle prestazioni di server di basi di dati. L'obiettivo è di ridurre il tempo di risposta di ogni operazione e migliorare il throughput dell'intero sistema, riducendo al contempo il traffico di rete, l'I/O su disco e il tempo di CPU. L'ottimizzazione delle prestazioni di sistemi ed applicazioni di basi di dati coinvolge produttori di DBMS, progettisti, programmatori e amministratori di basi di dati.

In questo lavoro non discuteremo le attività inerenti alla progettazione della base di dati e del software applicativo, sebbene fondamentali nel determinare le prestazioni di applicazioni che fanno un uso intensivo dei dati, ma concentreremo l'attenzione nell'analisi delle possibilità e degli strumenti a disposizione dell'amministratore di basi di dati in Microsoft® SQL Server™ 2000, per la distribuzione dei dati, l'elaborazione parallela di query, il monitoraggio e l'ottimizzazione delle prestazioni.

In genere, Microsoft SQL Server gestisce automaticamente le risorse disponibili determinando, caso per caso, la strategia da usare per soffisfare le richieste di elaborazione, tuttavia, l'amministratore di basi di dati (o di sistema) può intervenire manualmente, con la definizione di opportune regole, per migliori ottimizzazioni. Gli interventi manuali possono comprendere:

1. bilanciamento del carico di elaborazione tra più server
2. ottimizzazione delle queries.
3. impostazioni del sistema operativo e di SQL Server.

Discuteremo i punti precedenti e illustreremo gli strumenti forniti per monitorare e ottimizzare le prestazioni

2 Bilanciamento del carico di elaborazione

Per ottenere elevati livelli di prestazioni, richiesti soprattutto da siti Web di grandi dimensioni, vengono in genere utilizzati sistemi a più livelli, in cui il carico di elaborazione dei diversi livelli viene bilanciato su più server.

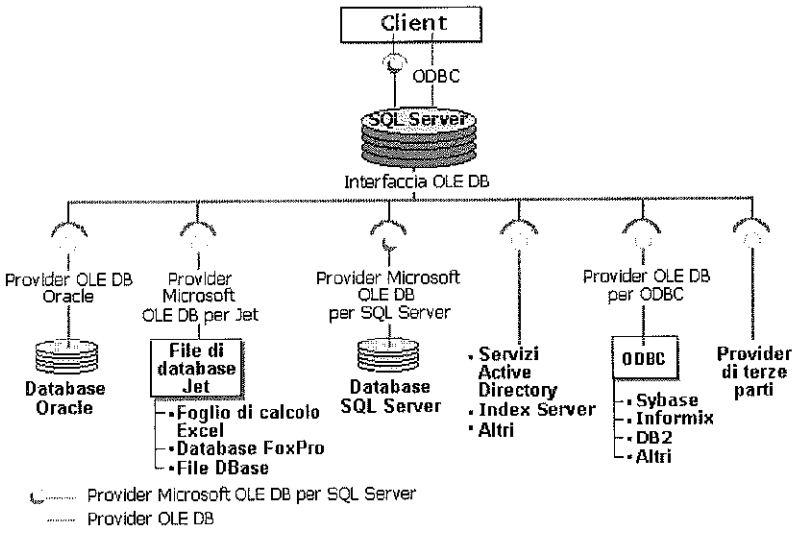
Microsoft SQL Server 2000 non è un sistema di gestione di basi di dati distribuite, ma fornisce un insieme di strumenti e di infrastrutture per realizzare molte delle funzionalità richieste ad un

DDBMS, purtroppo non sempre in maniera trasparente, e spesso con un cospicuo carico di lavoro demandato all'applicazione.

SQL Server 2000 supporta le query distribuite tramite le API OLE DB, la specifica di Microsoft per l'accesso universale ai dati. Le query distribuite consentono agli utenti di SQL Server di accedere a:

- Dati distribuiti in più istanze di SQL Server.
- Dati eterogenei archiviati in sorgenti relazionali e non relazionali.

In figura è mostrata l'architettura per l'accesso a dati distribuiti.



SQL Server esegue due tipi di ottimizzazione specifici per le query distribuite:

- Esecuzione remota di query utilizzata con provider di comandi SQL OLE DB.
- Accesso indicizzato utilizzato con provider di indici OLE DB.

Un provider OLE DB è considerato un provider di comandi SQL se soddisfa i seguenti requisiti minimi:

- Supporta l'oggetto **Command** e tutte le relative interfacce obbligatorie.
- Supporta la sintassi DBPROPVAL SQL SUBMINIMUM, la sintassi SQL-92 Entry Level o livello superiore oppure la sintassi ODBC Core Level o livello superiore. Il provider deve esporre questo livello di sottolinguaggio tramite la proprietà DBPROP_SQLSUPPORT OLE DB.

Un provider OLE DB è considerato un provider di indici se soddisfa i seguenti requisiti minimi:

- Supporta l'interfaccia **IDBSchemaRowset** con set di righe dello schema TABLES, COLUMNS e INDEXES.

- Supporta l'apertura di un set di righe su un indice con **IOpenRowset**.
- L'oggetto Index supporta tutte le interfacce obbligatorie: **IRowset**, **IRowsetIndex**, **IAccessor**, **IColumnsInfo**, **IRowsetInfo** e **IConvertTypes**.
- I set di righe aperti sulla tabella di base indicizzata (tramite **IOpenRowset**) devono supportare l'interfaccia **IrowsetLocate.**

SQL Server delega al provider di comandi SQL la maggior quantità possibile di operazioni di valutazione di una query distribuita. Nella documentazione a nostra disposizione non vengono forniti dettagli sugli algoritmi utilizzati.

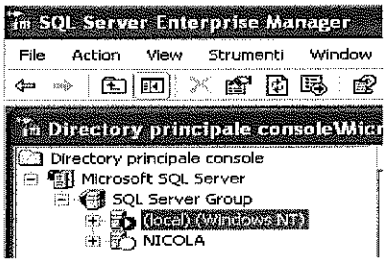
Con Sql Server è possibile distribuire il carico di elaborazione tramite:

- partizionamento dei dati
- replica di basi di dati
- cluster di server

2.1 Partizionamento dei dati

SQL Server consente di condividere il carico di elaborazione su un gruppo di server mediante il partizionamento orizzontale dei dati. Tali server vengono gestiti autonomamente, ma collaborano all'elaborazione delle richieste provenienti dalle applicazioni. Microsoft SQL Server 2000 chiama federazione un gruppo di server cooperativo di questo tipo. Per raggiungere livelli di prestazioni elevati, tuttavia, molto viene demandato all'applicazione per instradare le istruzioni SQL al server membro opportuno, ciò implica un'opportuna progettazione della base di dati e dei programmi che ne faranno uso.

Per federazioni di membri basati su SQL Server si definisce un *Server group* tramite strumenti visuali di Enterprise Manager e Wizard. In figura è mostrato un esempio di Server group con due server membri.



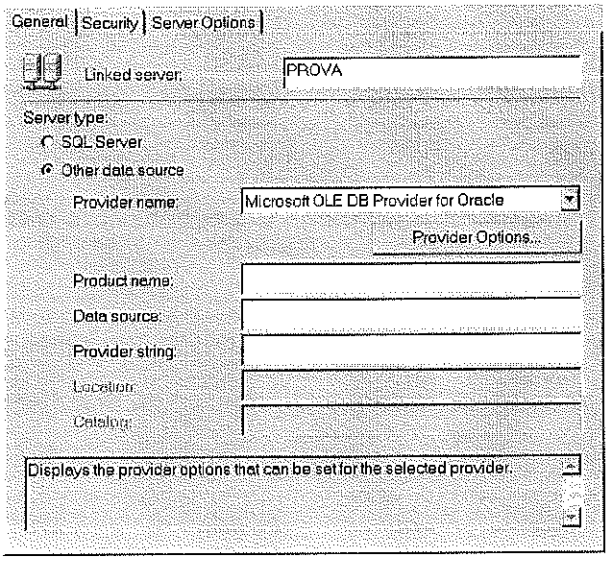
Per creare un partizionamento, è necessario eseguire le operazioni seguenti:

- Aggiungere le definizioni dei server collegati, se non fanno parte di un Server group, tramite il commando *sp_addlinkedserver*, oppure attraverso gli strumenti visuali di *Enterprise Manager*.

Definizione di SQL Server collegati

```
EXEC sp_addlinkedserver
   @server = 'LONDON Mktg',
   @srvproduct = 'Oracle',
   @provider = 'MSDAORA',
   @datasrc = 'MyServer'
```

Esempio di collegamento mediante dbprocedure di sistema..



Esempio di server collegato tramite le OLE DB.

- Impostare l'opzione **lazy schema validation**, utilizzando **sp_serveroption**, per ogni definizione di server collegato. Ciò consente di ottimizzare le prestazioni in quanto garantisce che il Query Processor richiederà metadati per le tabelle collegate solo quando i dati della tabella membro risulteranno effettivamente necessari.

- Definire le relazioni membro.

Esempio di partizionamento orizzontale:

```
-- Sul Server1:
CREATE TABLE Customer_33
  (CustomerID    INTEGER PRIMARY KEY
                 CHECK (CustomerID BETWEEN 1 AND 32999),
  ...
-- Sul Server2:
CREATE TABLE Customer_66
  (CustomerID    INTEGER PRIMARY KEY
                 CHECK (CustomerID BETWEEN 33000 AND 65999),
  ...
-- Sul Server3:
CREATE TABLE Customer_99
  (CustomerID    INTEGER PRIMARY KEY
                 CHECK (CustomerID BETWEEN 66000 AND 99999),
  ...
```

- Creare una vista in ogni server membro da cui si vogliono utilizzare frammenti distribuiti orizzontalmente.

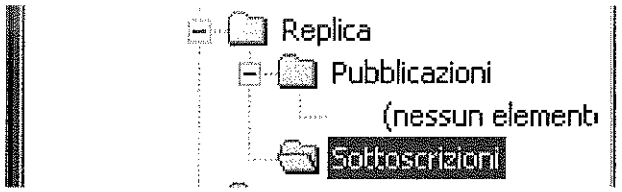
```
Esempio:

CREATE VIEW Customers AS
   SELECT * FROM CompanyDatabase.TableOwner.Customers_33
UNION ALL
   SELECT * FROM Server2.CompanyDatabase.TableOwner.Customers_66
UNION ALL
   SELECT * FROM Server3.CompanyDatabase.TableOwner.Customers_99
```

Naturalmente per poter essere utilizzate in operazioni di aggiornamento le viste devono soddisfare i requisiti di modificabilità.

2.2 Replica di database

Un'altra possibilità di ottimizzazione delle prestazioni, da considerare solo per basi di dati poco movimentate, si ottiene tramite la funzione di replica di SQL Server 2000, che permette di mantenere copie sincronizzate di dati e oggetti di database. Anche questa funzione si attiva da *Enterprise Manager*.



2.2.1 Architettura della replica

La funzione di replica di Microsoft SQL Server 2000 utilizza la metafora di un'industria editoriale per rappresentare i componenti e i processi in una topologia di replica. Il modello è composto da:

server di pubblicazione, server di distribuzione, server di sottoscrizione, pubblicazioni, articoli e sottoscrizioni.

Sono disponibili inoltre diversi processi di replica responsabili per la copia e il trasferimento dei dati tra il server di pubblicazione e il server di sottoscrizione. Tali processi sono denominati agente snapshot, agente di distribuzione, agente di lettura dei log, agente di lettura code e agente di merge

- *Il server di pubblicazione* rende i dati disponibili per la replica in altri server e può gestire una o più pubblicazioni, ognuna delle quali rappresenta un set di dati correlato logicamente. Oltre ad essere il server in cui vengono selezionati i dati da replicare, il server di pubblicazione consente di individuare quali dati sono stati modificati durante la replica transazionale e di aggiornare le informazioni relative alle pubblicazioni disponibili in tale sito.

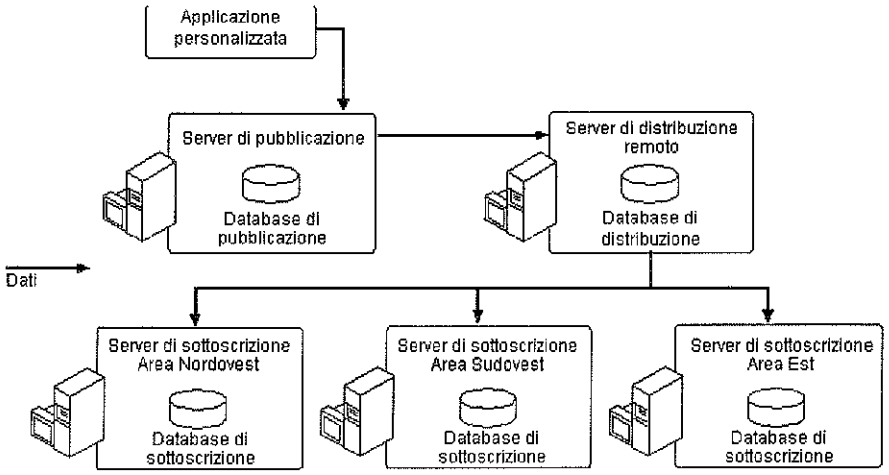
- *Il server di distribuzione* mantiene il database di distribuzione e i metadati oltre ad informazioni storiche su dati e transazioni. Il ruolo del server di distribuzione varia in base al tipo di replica implementato. .

- *I server di sottoscrizione* ricevono i dati replicati. In base al tipo di replica e alle opzioni utilizzate, il server di sottoscrizione può inoltre distribuire al server di pubblicazione le modifiche apportate ai dati o pubblicare nuovamente i dati negli altri server di sottoscrizione.

- *Un articolo* può essere costituito da un'intera tabella, da una partizione orizzontale o verticale, da una definizione di stored procedure o di vista, oppure da una funzione definita dall'utente

- *Una pubblicazione* è una raccolta di uno o più articoli di un database. Il raggruppamento di più articoli semplifica la selezione di un set di dati correlato logicamente e degli oggetti del database che si desidera replicare in una singola operazione.

- *Una sottoscrizione* è la richiesta di una copia di dati o di oggetti di database da replicare. La sincronizzazione o la distribuzione dei dati di una sottoscrizione può essere richiesta sia dal server di pubblicazione (sottoscrizioni push) che dal server di sottoscrizione (sottoscrizioni pull). Una pubblicazione può supportare una combinazione di sottoscrizioni push e pull



Microsoft SQL Server 2000 supporta la replica da e verso sorgenti eterogenee di dati, tra le quali Microsoft Exchange, Microsoft Access, Oracle e DB2.

SQL Server 2000 utilizza tre tipi di replica: replica snapshot, replica transazionale, replica di tipo merge.

- La *replica snapshot* copia i dati o gli oggetti di database esattamente nello stato in cui si trovano in un determinato momento. La replica snapshot viene utilizzata nel caso in cui i dati di origine siano relativamente statici, l'allineamento non sia un elemento critico dell'applicazione, e la quantità di dati da replicare sia ridotta.
- La *replica transazionale* viene utilizzata nel caso in cui sia necessario replicare i dati al momento della modifica. L'integrità transazionale viene mantenuta tra i server di sottoscrizione eseguendo tutte le modifiche nel server di pubblicazione e quindi replicandole nei server di sottoscrizione.
- La *replica di tipo merge* consente a più siti di utilizzare in modo autonomo un set di server di sottoscrizione e quindi di eseguire un merge nel server di pubblicazione. Questo tipo di replica supporta la definizione di regole di risoluzione dei conflitti (caso in cui gli stessi dati siano stati modificati da più server di sottoscrizione).

2.3 Cluster di server

I componenti COM+ di Microsoft Windows 2000 sono progettati per l'utilizzo in cluster di server di applicazioni. Ogni server include gli stessi componenti COM+ e Windows 2000 bilancia il carico di elaborazione del cluster inviando le richieste al server con il minor carico di elaborazione. SQL Server 2000 non supporta questo tipo di clustering, ma offre semplicemente la possibilità di definire un *clustering di failover* che consente di mantenere la disponibilità del sistema su livelli alti, infatti

in caso di malfunzionamento di un nodo il controllo è ceduto ad un altro nodo del cluster (per i sistemi operativi Windows NT 4.0, Enterprise Edition, Windows 2000 Advanced Server oppure Windows 2000 Datacenter Server e Microsoft Cluster Service).

Con SQL Server 2000 Enterprise Edition è stato introdotto il supporto per i protocolli SAN (System Area Network) utilizzando l'architettura VIA (Virtual Interface Architecture), le cui specifiche, che corrispondono a una definizione generica di rete SAN, sono state definite da Compaq, Intel, Microsoft e altre società. Rispetto alle reti LAN o WAN, le reti SAN supportano volumi più elevati di operazioni di trasferimento di messaggi, in quanto riducono i carichi della CPU e la latenza dei messaggi, sono inoltre più affidabili rispetto alle reti LAN o WAN e vengono implementate in gruppi di cluster di server fisicamente vicini, ad esempio nella stessa stanza.

SQL Server 2000 supporta l'implementazione Giganet delle reti SAN basate su VIA, sui sistemi operativi NT Server, Windows 2000 Data Center, Advanced Server e Server.

3 Ottimizzazione di query

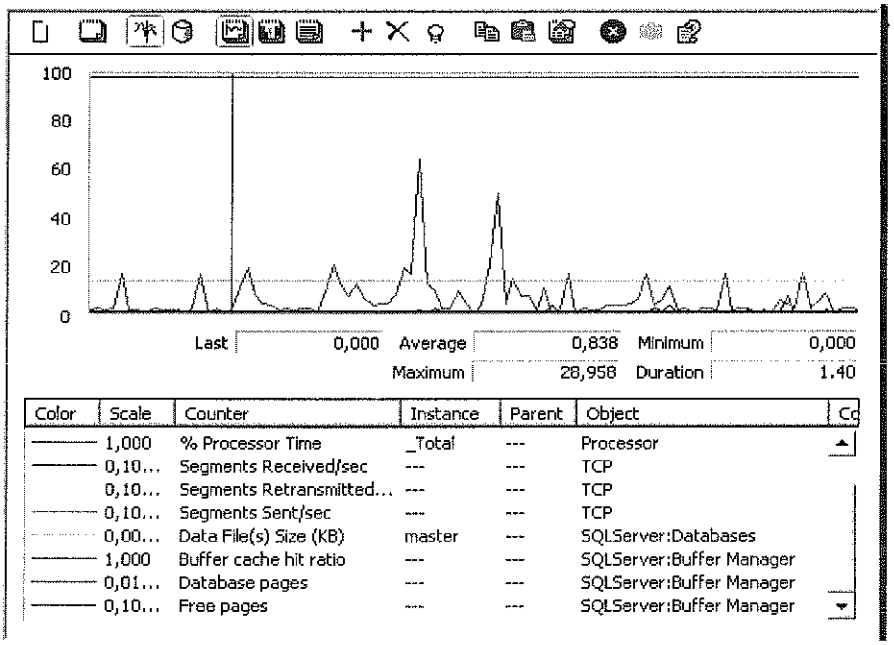
In alcuni casi si può risolvere un problema di prestazioni operando unicamente a livello di ottimizzazione del server, intervenendo ad esempio sulle dimensioni della memoria, sul tipo di file system, sul numero e il tipo di processori e così via, ma nella maggior parte dei casi i problemi di prestazioni sono risolvibili solo analizzando l'applicazione. In questo capitolo illustreremo le possibilità offerte da SQL Server, per l'analisi e l'ottimizzazione delle query.

Il fatto che l'esecuzione di alcune operazioni richieda tempi più lunghi del previsto può essere imputabile ai seguenti fattori:

- Comunicazioni di rete lente.
- Memoria insufficiente.
- Mancanza di statistiche utili o statistiche obsolete.
- Inadeguata definizione di indici.
- Striping dei dati inefficace.

3.1 Comunicazioni di rete lente.

Occorre verificare se il calo delle prestazioni può essere dovuto totalmente o in parte ad altri componenti. Queste verifiche si possono effettuare tramite gli strumenti del sistema operativo, per esempio con *Performance Monitor* di Windows NT è possibile analizzare gli oggetti e i contatori delle prestazioni specifici di SQL Server nonché i dati relativi ad altri oggetti, quali processori, memoria, cache, thread e processi. A ogni oggetto è associato un set di contatori che misurano l'utilizzo del dispositivo, le lunghezze delle code, i ritardi e altri indicatori di throughput e di congestione interna. In figura un esempio di monitoraggio delle prestazioni.

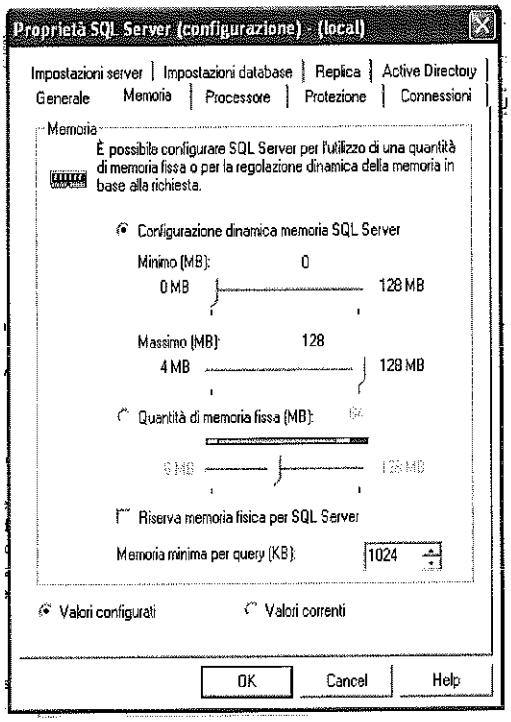


Il monitor di sistema fornisce dati statistici in tempo reale sull'attività e le prestazioni di SQL Server. È possibile aggiungere avvisi di sistema al verificarsi di determinate situazioni, creare file di log, esportare i dati di grafici e log in altre applicazioni per manipolarli, creare report dai file di log, etc..

3.2 Memoria insufficiente

Per default, in SQL Server, i requisiti di memoria variano dinamicamente in base alle risorse di sistema disponibili. Se SQL Server necessita di una maggior quantità di memoria, richiede al sistema operativo eventuale memoria disponibile. Se SQL Server non utilizza completamente la memoria allocata, la rilascia al sistema operativo. È possibile disattivare l'opzione per l'utilizzo dinamico della memoria utilizzando le opzioni di configurazione **min server memory**, **max server memory** e **set working set size**. Si utilizza **min server memory** per garantire a SQL Server una quantità minima di memoria, **max server memory** per limitare la quantità di memoria utilizzabile e

set working set size per assegnare una determinata quantità di memoria ad una operazione. La figura mostra l'operazione di configurazione tramite la console di *Enterprise Manager*. L'impostazione di questi parametri può influenzare le prestazioni di altre componenti dell'ambiente in cui gira SQL Server, per cui è necessario operare con molta attenzione.



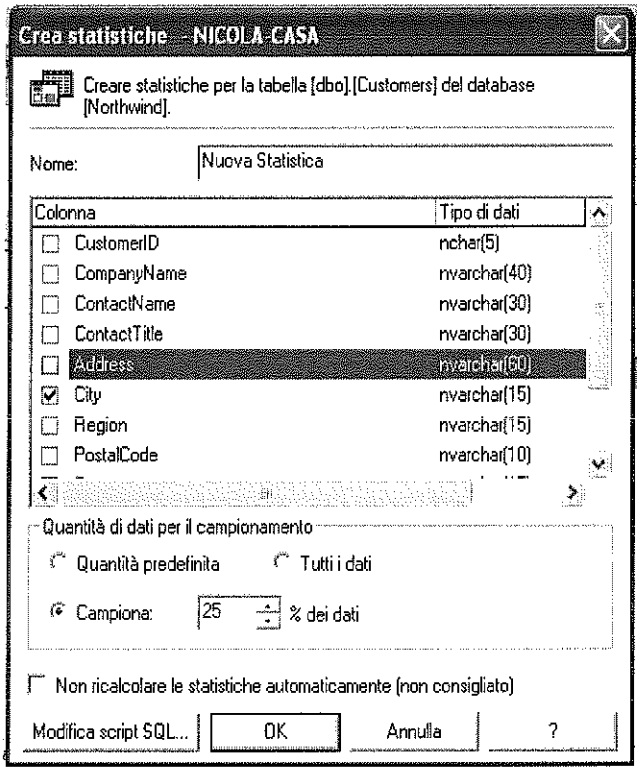
È possibile monitorare l'utilizzo della memoria, e di altre risorse, utilizzate da un determinato processo tramite la console di *Enterprise Manager*. In Figura un esempio, in cui sono mostrate, il numero di pagine della cache assegnate al processo, il numero totale di letture e scritture effettuate su disco, il tempo di cpu, in millisecondi, ed altre informazioni.

Informazioni processo 13 elementi

Stato	Transazioni...	Comando	Tempo di at...	CPU	I/O fisico	Utilizzo di memoria
background ...	0	LAZY WRITER	591	10	0	0
background ...	0	TASK MANAGER	0	0	0	2
background ...	0	TASK MANAGER	0	0	0	2
background ...	0	TASK MANAGER	0	0	0	2
sleeping ...	0	LOG WRITER	20	10	0	0
background ...	0	SIGNAL HANDLER	0	20	0	4
background ...	0	LOCK MONITOR	4857	0	0	12
background ...	0	TASK MANAGER	0	0	2	2
runnable ...	2	SELECT INTO	0	631	209	174
background ...	0	TASK MANAGER	4991648	0	0	2
sleeping ...	0	CHECKPOINT SLEEP	4991648	0	0	0
background ...	0	TASK MANAGER	0	0	2	2
background ...	0	TASK MANAGER	0	0	0	2

3.3 Mancanza di statistiche utili o statistiche obsolete.

Le statistiche relative alla distribuzione dei valori in una colonna vengono create automaticamente sulle colonne indicizzate. È possibile creare statistiche sulle colonne non indicizzate, manualmente tramite SQL *Query Analyzer* (figura) o l'istruzione *CREATE STATISTICS*. Le statistiche possono essere utilizzate dal Query Processor per determinare la strategia ottimale di valutazione di una query. Quando l'opzione AUTO_CREATE_STATISTICS è impostata su ON, SQL Server crea automaticamente le statistiche per le colonne senza indici che vengono utilizzate in un predicato.



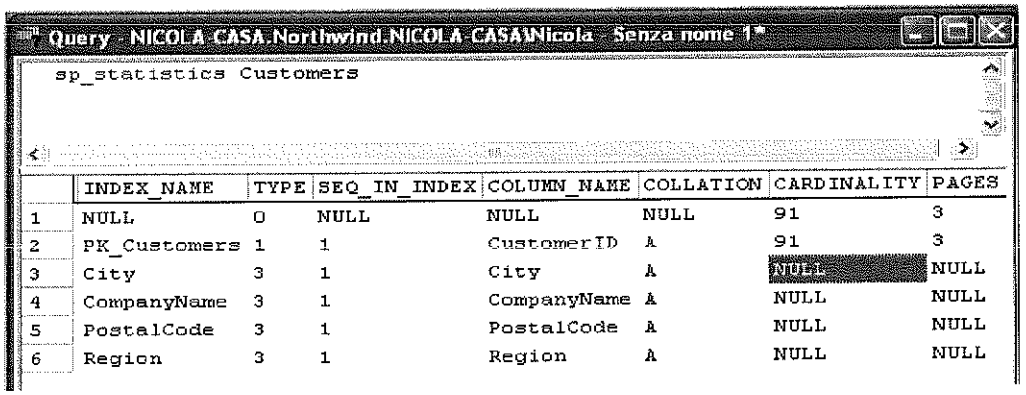
Periodicamente SQL Server aggiorna automaticamente le informazioni statistiche quando vengono modificati i dati. La frequenza di aggiornamento è determinata dal volume di dati nella colonna o nell'indice e dalla quantità dei dati modificati. È possibile ridurre il costo dell'aggiornamento automatico delle statistiche eseguendo il campionamento dei dati anziché un'analisi completa, utilizzando le clausole SAMPLE e FULLSCAN dell'istruzione UPDATE STATISTICS (o con lo strumento visuale della precedente figura). La clausola FULLSCAN specifica la scansione di tutti i dati della tabella, mentre la clausola SAMPLE specifica la percentuale o il numero di righe da campionare.

È inoltre possibile evitare che SQL Server gestisca le statistiche di una determinata colonna o indice, nel qual caso sarà necessario aggiornare manualmente le informazioni statistiche. Di seguito sono elencati i comandi per la creazione e l'aggiornamento manuale di statistiche.

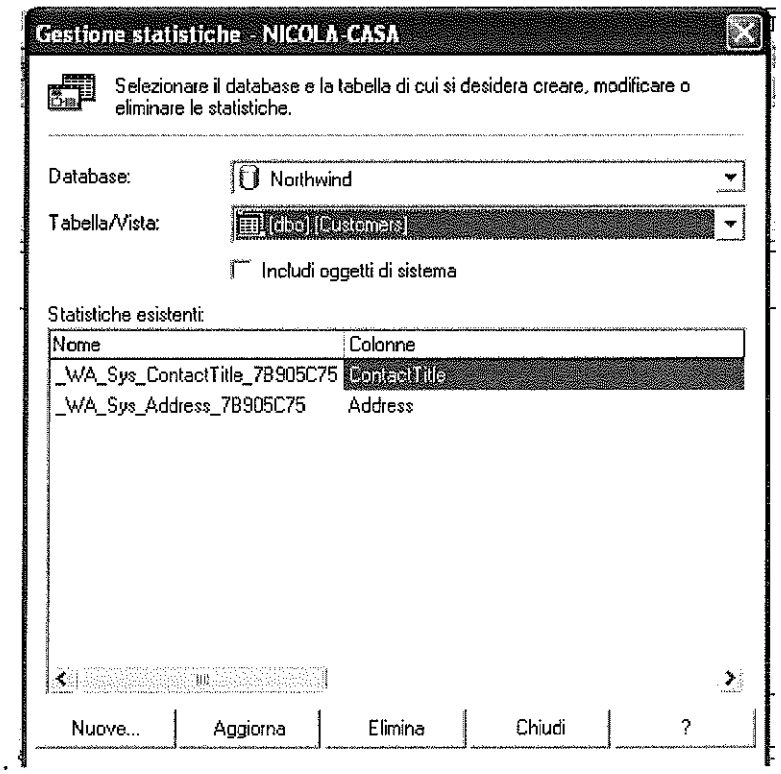
```
CREATE STATISTICS statistics_name
ON { table | view } ( column [ ,...n ] )
  [ WITH
    [ [ FULLSCAN
      | SAMPLE number { PERCENT | ROWS } ] [ , ] ] [ NORECOMPUTE ] ]
```

```
UPDATE STATISTICS table | view
  [ index | ( statistics_name [ ,...n ] ) ]
  [  WITH [
      [ FULLSCAN ]
      | SAMPLE number { PERCENT | ROWS } ]
      | RESAMPLE ] [ [ , ] [ ALL | COLUMNS | INDEX ] [ [ , ] NORECOMPUTE ] ]
```

È possibile visualizzare le statistiche disponibili per una tabella tramite il comando *sp_statistics* oppure tramite gestione statistiche in QueryAnalizer (figura).



	INDEX_NAME	TYPE	SEQ_IN_INDEX	COLUMN_NAME	COLLATION	CARDINALITY	PAGES
1	NULL	0	NULL	NULL	NULL	91	3
2	PK_Customers	1	1	CustomerID	A	91	3
3	City	3	1	City	A	NULL	NULL
4	CompanyName	3	1	CompanyName	A	NULL	NULL
5	PostalCode	3	1	PostalCode	A	NULL	NULL
6	Region	3	1	Region	A	NULL	NULL



È inoltre possibile monitorare la classe di evento **Missing Column Statistics** per determinare la mancanza di statistiche per una colonna utilizzata da una query, tramite *SQL Profiler*, oppure attivando un avviso tramite *SQL Agent*.

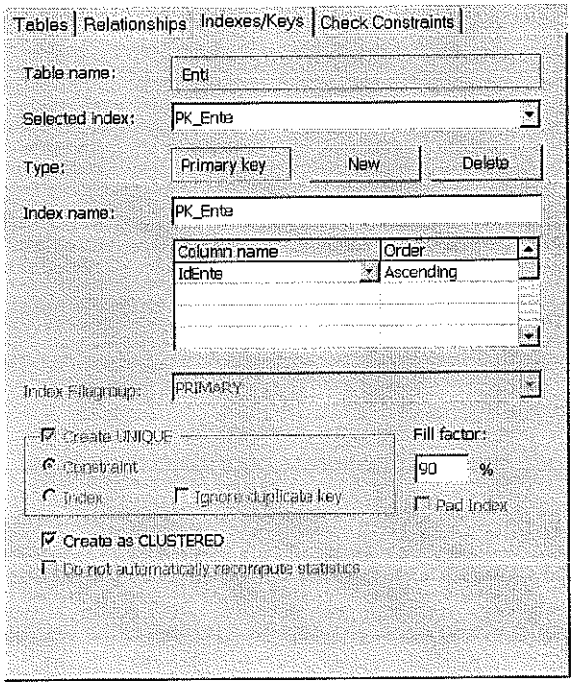
3.4 Inadeguata definizione degli indici.

La definizione di indici è un fattore chiave della progettazione fisica della base di dati, che determina in modo significativo le prestazione di applicazioni con uso intensivo dei dati. In questo paragrafo non considereremo gli aspetti legati alla progettazione, ma discuteremo le possibilità offerte da Microsoft SQL Server per la definizione, il monitoraggio e l'ottimizzazione degli indici.

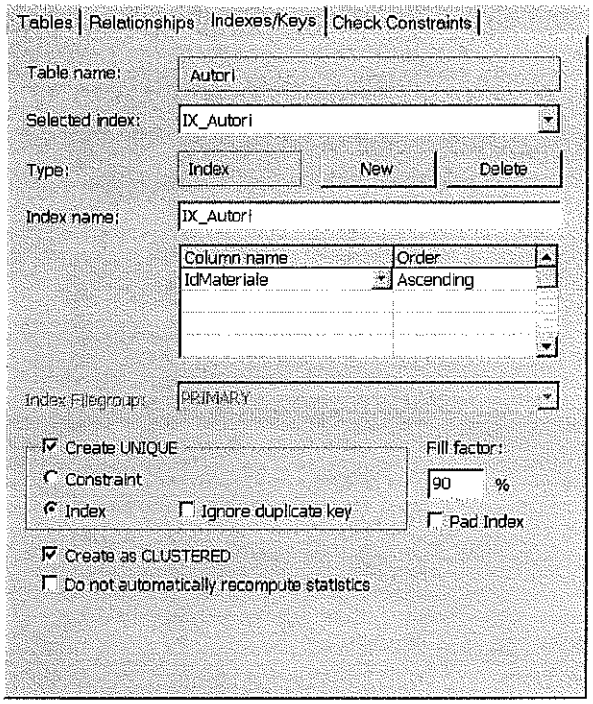
Come per la maggior parte dei DBMS commerciali, in Sql Server è possibile personalizzare il tipo di indice più adatto a seconda della situazione. Di seguito vengono indicate le varie modalità supportate:

- Cluster e non cluster
- Univoci e non univoci
- A colonna singola e a più colonne
- In ordine crescente o decrescente per le colonne dell'indice

Microsoft® SQL Server™ 2000 supporta indici definiti su qualsiasi tipo di colonna di una tabella (o vista, ved. oltre), incluse le colonne calcolate (con alcune limitazioni). L'estensione relazionale ad oggetti di SQL Server supporta la definizione di indici invertiti sugli attributi testuali di una tabella. Gli indici si definiscono coi comandi SQL oppure con gli strumenti visuali di Enterprise Manager. Nella figura è mostrata la definizione di una chiave primaria.



Nella figura sottostante è mostrata la definizione di un indice. È possibile specificare l'univocità, come vincolo di integrità o come indice, in questo caso è possibile definire il comportamento in caso di tentativi di inserimento di valori duplicati nella colonna; infatti impostando *ignore duplicate key*, il sistema non inserisce la riga che ha violato il vincolo, ma la transazione non viene abortita (come invece avverrebbe per default). È possibile definire il fattore di riempimento delle foglie dell'indice ed opzionalmente dei nodi intermedi (*pad index*, non è possibile esprime un fattore di riempimento diverso da quello delle foglie però).

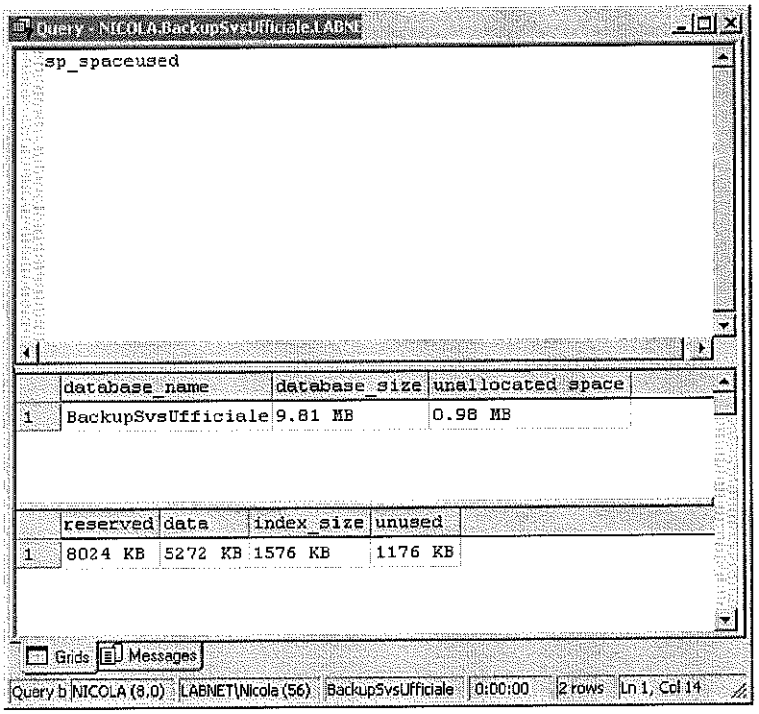


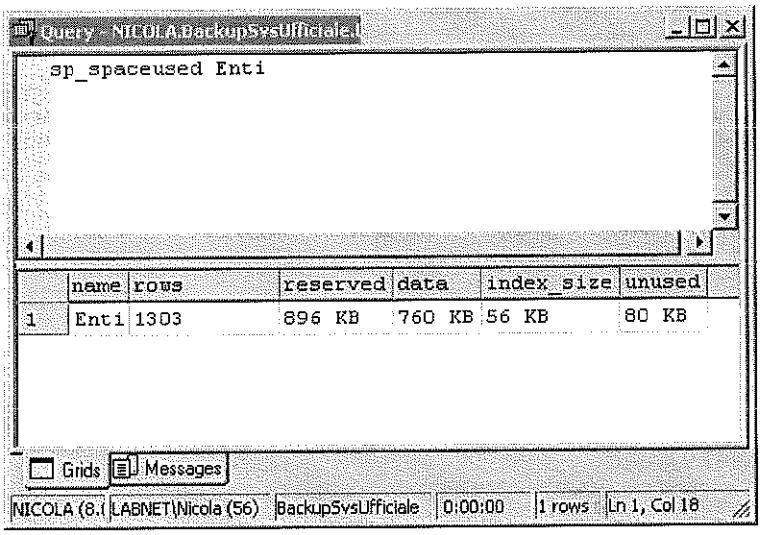
Altre proprietà definibile per gli indici sono la clusterizzazione e l'esclusione del calcolo automatico delle statistiche al variare degli oggetti nell'indice, in questo caso occorre provvedere manualmente tramite l'invocazione di opportune procedure disponibile al DBA. È inoltre possibile specificare il nome del filegroup (ved. 3.5.2) in cui memorizzare l'indice.

Con Microsoft SQL Server:

- È possibile creare al massimo 249 indici non cluster per tabella (inclusi gli indici creati dai vincoli PRIMARY KEY o UNIQUE).
- Le dimensioni massime di tutte le colonne di lunghezza non variabile che costituiscono l'indice sono di 900 byte.
- Lo stesso indice può comprendere al massimo 16 colonne.

Lo spazio per tabelle e indici viene allocato in incrementi di un extent (otto pagine da 8 KB). Quando un extent risulta pieno, ne viene allocato un altro. Per gli indici di tabelle molto piccole o vuote vengono allocate pagine singole fino ad otto, dopodiché vengono allocati extent. Per visualizzare un report sulla quantità di spazio allocato e utilizzato da un indice, utilizzare la procedura **sp_spaceused**. Nelle due figure seguenti è mostrata l'allocazione di un database e di una tabella rispettivamente.





Le righe di una tabella vengono archiviate in una sequenza specifica solo se nella tabella viene creato un indice cluster. Se la tabella dispone solo di indici non cluster, le righe di dati verranno archiviate come heap non ordinato.

3.4.1 Viste Indicizzate

Microsoft SQL Server 2000 supporta la definizione di indici per le viste. La definizione di un indice su una vista comporta la memorizzazione fisica dei dati corrispondenti e l'aggiornamento automatico (rispetto alle modifiche apportate ai dati delle tabelle di base su cui essa è definita).

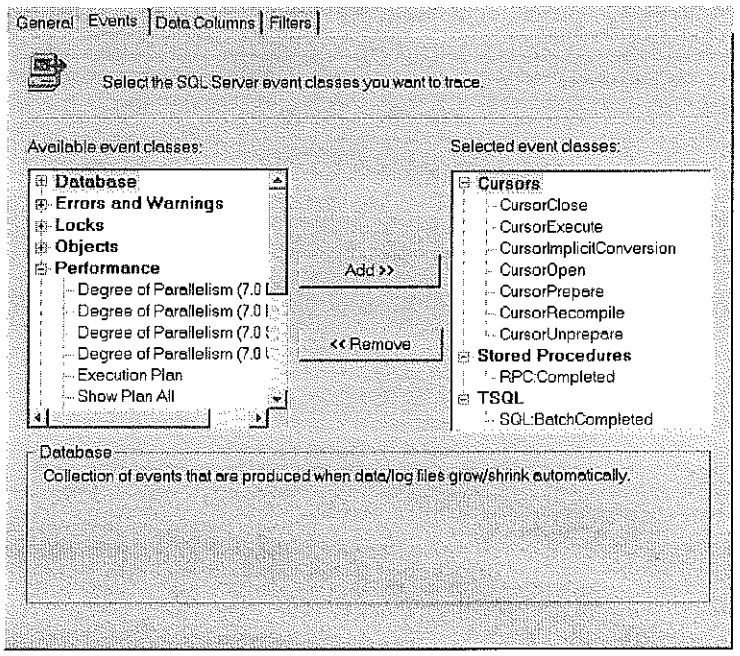
Un vantaggio della creazione di un indice su una vista è rappresentato dal fatto che il Query Optimizer può utilizzarlo anche nelle query che non specificano direttamente il nome della vista nella clausola FROM; lo svantaggio principale deriva dal sovraccarico di lavoro nell'aggiornamento dei dati, per cui è necessario porre attenzione durante la progettazione della base di dati.

3.4.2 Ottimizzazione degli indici.

In Microsoft SQL Server sono disponibili due strumenti di ausilio del DBA, per l'analisi e l'ottimizzazione degli indici: *Index Tuning Wizard* e *Query Analyzer*.

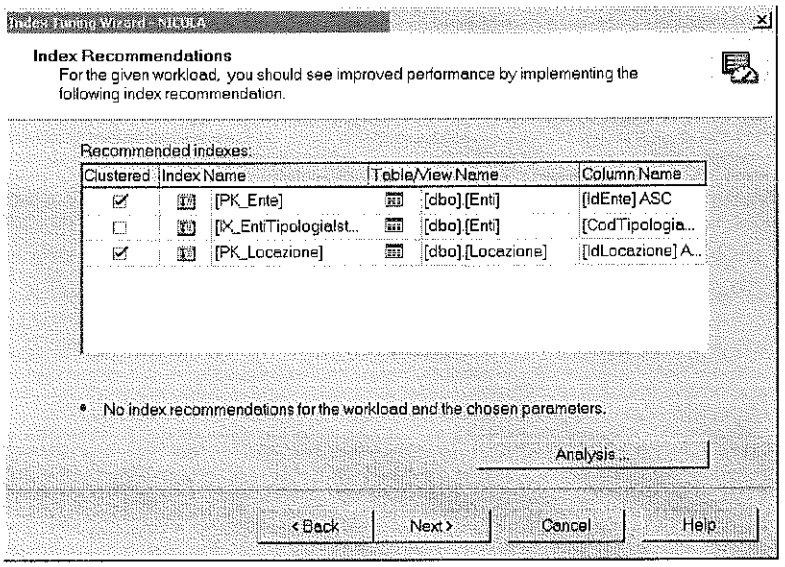
3.4.2.1 Index Tuning Wizard

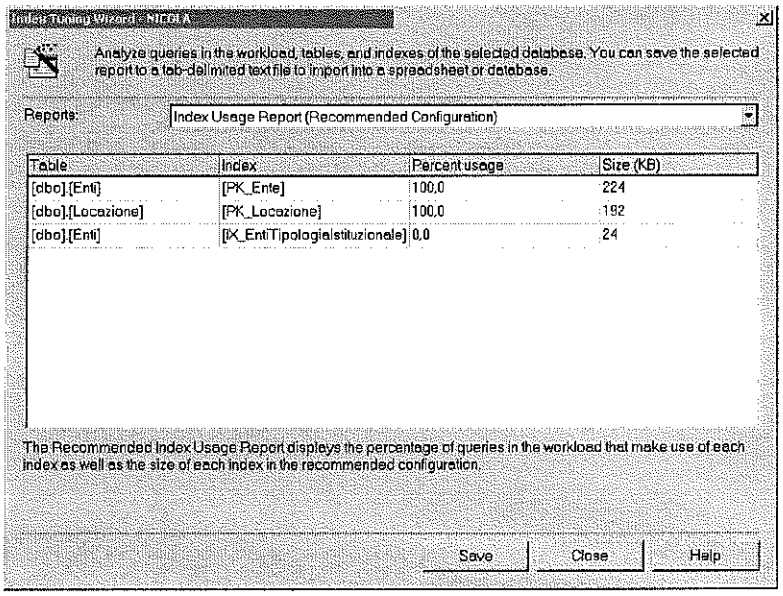
Con questo strumento è possibile attivare un'ottimizzazione guidata degli indici. Il wizard è in grado di analizzare i dati del database e formulare dei consigli (che il DBA può optare di far eseguire direttamente) per l'ottimizzazione in base ad un carico di lavoro. Un carico di lavoro è costituito da uno script SQL o da una traccia di *SQL Profiler*. Con lo strumento visuale *SQL Profiler* è infatti possibile (tra l'altro) registrare informazioni relativi a vari eventi, nelle due figure seguenti è mostrata la fase di definizione degli eventi ed un frammento della traccia generata.



EventClass	TextData	NTUserName	LoginName	CPU	Reads	Writes	Duration	Client
Scan:Started		Sabrina	Infea...					1740
SQL:BatchCompleted	DELETE FROM RichiesteFinanziamento ...	Sabrina	Infea...	0	91	3	0	1740
SQL:BatchCompleted	IF @@TRANCOUNT > 0 COMMIT TRAN	Sabrina	Infea...	0	2	1	0	1740
Scan:Started		Sabrina	Infea...					1740
SQL:BatchCompleted	SELECT max(IdEnte) FROM Enti	Sabrina	Infea...	16	2	0	13	1740
Scan:Started		Sabrina	Infea...					1740
SQL:BatchCompleted	SELECT max(IdDocumento) FROM docume...	Sabrina	Infea...	0	2	0	0	1740
Scan:Started		Sabrina	Infea...					1740
SQL:BatchCompleted	SELECT max(IdServizio) FROM Servizi	Sabrina	Infea...	0	2	0	0	1740
Scan:Started		Sabrina	Infea...					1740
SQL:BatchCompleted	SELECT max(IdEsperienza) FROM Esper...	Sabrina	Infea...	0	0	0	0	1740
Scan:Started		Sabrina	Infea...					1740
SQL:BatchCompleted	SELECT max(IdManifestazione) FROM M...	Sabrina	Infea...	0	2	0	0	1740
Scan:Started		Sabrina	Infea...					1740
SQL:BatchCompleted	SELECT max(IdVisita) FROM VisitaSog...	Sabrina	Infea...	0	2	0	0	1740
Scan:Started			sa					0
Scan:Started			sa					0
Scan:Started			sa					0

Una volta attivato *Index Tuning Wizard* e scelto il carico di lavoro, il DBA è invitato a specificare il livello di approfondimento (da 1 a 3) nell'analisi dei dati. Dopo l'esecuzione il wizard presenta i suoi consigli ed il risultato dell'analisi dei dati coinvolti dal carico di lavoro. Nelle figure che seguono è mostrata la segnalazione dei consigli e la finestra dei report generati dall'analisi dei dati. Il DBA può decidere di accettare i consigli del wizard ed attivare (o schedulare) la procedura generata per apportare le modifiche alla base di dati.





3.4.2.2 Query analyzer

SQL Query Analyzer è uno strumento grafico che consente di:

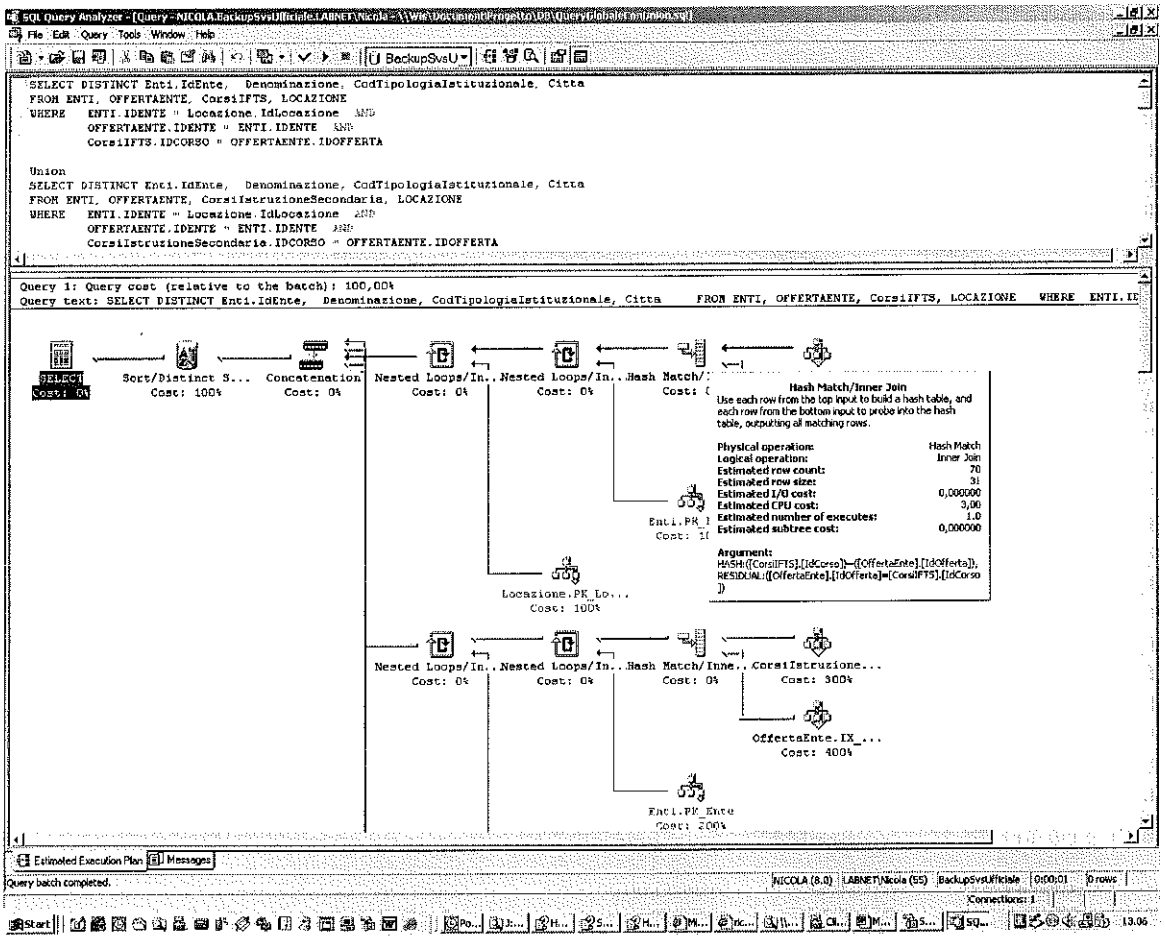
- Creare query e script SQL ed eseguirli.
- Utilizzare script predefiniti e generare script di oggetti del database.
- Eseguire e fare il debug di stored procedure.
- Inserire, aggiornare o eliminare le righe di una tabella, in modalità grafica
- Eseguire il debug dei problemi relativi alle prestazioni delle query (Visualizza piano di esecuzione, Visualizza traccia del server, Visualizza statistiche del client, Ottimizzazione guidata indici).

Concentriamo la nostra attenzione sull'ultimo punto perché è l'oggetto del presente lavoro. Per quanto riguarda le azioni:

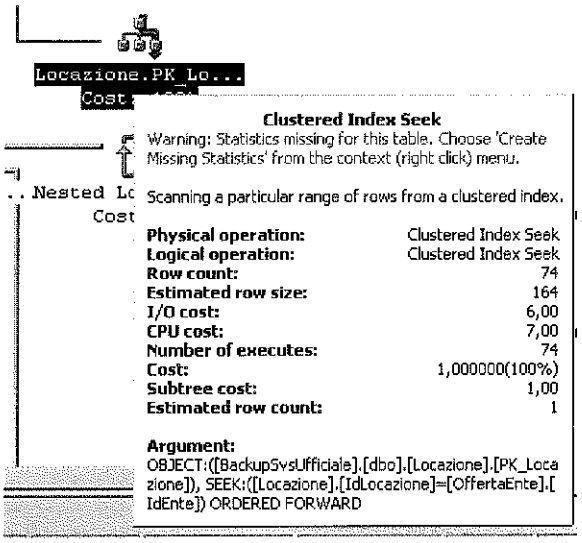
- Visualizza traccia del server,
- Visualizza statistiche del client,
- Ottimizzazione guidata indici

vengono attivati gli strumenti già presentati in precedenza. Una funzione molto comoda è la visualizzazione del piano di esecuzione sia stimato (cioè prima di eseguire la query) che effettivo

(cioè dopo l'esecuzione della query), in modalità testuale o grafica. Nella modalità grafica viene visualizzato un grafo, i cui nodi descrivono le varie operazioni del piano di esecuzione. Ad ogni nodo sono associate (al posizionamento del mouse) informazioni dettagliate sulle operazioni fisiche e sui relativi costi. In figura è visualizzato un frammento del piano di esecuzione di una query.



Ai nodi del grafo possono essere associati degli avvisi (individuabili dall'etichetta di un altro colore), quali ad esempio la mancanza di statistiche (figura sottostante) ed è possibile invocare il modulo gestore di statistiche (ved. 3.3) per la relazione coinvolta.



Di seguito è presentato un frammento di access plan in formato testuale.

```
|--Nested Loops(Inner Join, OUTER REFERENCES:([Enti].[IdEnte]) WITH PREFETCH)
    |--Sort(ORDER BY:([Enti].[IdEnte] ASC))
    |  |--Hash Match(Right Semi Join, HASH:([Union1059])=([Enti].[IdEnte]), RESIDUAL:([Enti].[IdEnte]=[Union1059]))
    |     |--Concatenation
    |     |  |--Sort(DISTINCT ORDER BY:([OffertaEnte].[IdEnte] ASC))
    |     |  |  |--Hash Match(Inner Join, HASH:([Offerta].[IdOfferta])=([OffertaEnte].[IdOfferta]),
RESIDUAL:([OffertaEnte].[IdOfferta]=[Offerta].[IdOfferta]))
Scan(OBJECT:([BackupSvsUfficiale].[dbo].[Manifestazione].[PK_Manifestazione]))
    |     |  |        |  |--Clustered Index Seek(OBJECT:([BackupSvsUfficiale].[dbo].[Offerta].[PK_OffertaEA]),
    |     |--Nested Loops(Inner Join, OUTER REFERENCES:([Union1027]) WITH PREFETCH)
    |        |--Stream Aggregate(GROUP BY:([Union1027]))
    |        |  |--Merge Join(Concatenation)
```

3.4.3 Query Optimizer

In questo paragrafo presentiamo brevemente alcune caratteristiche del query optimizer di Microsoft SQL Server.

A ogni piano di esecuzione possibile corrisponde un costo in termini di quantità di risorse del computer utilizzate. Il Query Optimizer di SQL Server non sceglie esclusivamente il piano di esecuzione con il costo minore in termini di risorse, ma quello che restituisce più rapidamente i risultati con un costo ragionevole. Ad esempio, l'esecuzione parallela di una query, in genere, utilizza una quantità di risorse maggiore, rispetto all'esecuzione seriale, ma può consentire di completarla più rapidamente. Nella risoluzione di una query, l'ottimizzatore potrà inoltre decidere

l'utilizzo di una vista indicizzata con una logica analoga a quella utilizzata per l'utilizzo di un indice per una tabella. Il piano di esecuzione per le query distribuite viene creato in modo che tutti gli accessi alle tabelle remote vengano ritardati fino al momento in cui i dati sono necessari.

Nel caso di una esecuzione batch contenente più istruzioni SQL, tutti i piani ottimizzati per le varie istruzioni vengono incorporati in un singolo piano di esecuzione.

SQL Server supporta la cache delle procedure per memorizzare i piani di esecuzione.

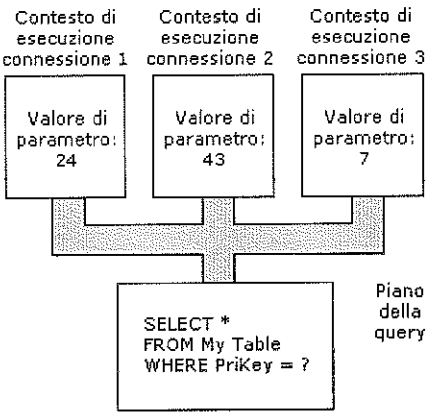
I piani di esecuzione di SQL Server 2000 sono costituiti da due componenti principali:

- Piano della query

  È una struttura dati rientrante di sola lettura che può essere utilizzata da un numero qualsiasi di utenti. In memoria non sono mai disponibili più di una o due copie del piano della query: una copia per tutte le esecuzioni seriali e una per tutte le esecuzioni parallele. La copia parallela copre tutte le esecuzioni parallele, indipendentemente dal loro grado di parallelismo.

- Contesto di esecuzione

  È una struttura dati associata all'esecuzione di una query, contenente i dati specifici, ad esempio i valori dei parametri. Le strutture di dati del contesto di esecuzione vengono riutilizzate.



Quando viene sottoposta un'istruzione SQL, prima di generare un piano di esecuzione viene verificato che non ne esista uno nella cache delle procedure.

Un piano di esecuzione rimane nella cache delle procedure finchè non c'è necessità di spazio. Ad ogni piano della query e contesto di esecuzione è associato il costo per la compilazione della struttura, il numero di riferimenti ad esso e un campo *età*, che viene incrementato del fattore di costo ogni volta che una connessione fa riferimento all'oggetto.

Periodicamente il processo *Lazywriter* di SQL Server esegue la scansione degli oggetti presenti nella cache delle procedure, decrementando di un'unità l'*età* di ogni oggetto. La deallocazione di un oggetto dalla cache avviene se sono soddisfatte tre condizioni:

- Il gestore della memoria richiede memoria che non è al momento disponibile.
- Il campo età dell'oggetto è 0.
- Nessuna connessione fa riferimento all'oggetto.

Microsoft SQL Server rileva automaticamente il grado di parallelismo migliore per ogni istanza di esecuzione di una query, in base a valutazioni sul numero dei processori (almeno 2), il numero di utenti concorrenti, la memoria disponibile, il numero di righe elaborate. Se l'esecuzione parallela della query è possibile, SQL Server determina il numero ottimale di thread e suddivide l'esecuzione della query tra i thread.

Per monitorare il grado di parallelismo per le singole istruzioni è possibile utilizzare SQL Profiler, tramite la classe di evento *Degree Of Parallelism*.

3.5 Striping dei dati inefficace.

SQL Server permette di gestire il posizionamento dei dati nelle unità disco (striping) per migliorare le prestazioni del sistema e implementare nel contempo la tolleranza agli errori.

Per la gestione del posizionamento dei dati nelle unità disco è possibile utilizzare i metodi seguenti:

- Un sistema RAID (Redundant Array of Independent Disks) di tipo hardware e di livello superiore a 0, protegge il sistema dalla perdita di dati in caso di guasto di un supporto e può determinare un miglioramento delle prestazioni.
- Lo striping del disco e lo striping con parità, basati su Windows NT e Windows 2000, implementano le funzionalità RAID a livello software utilizzando qualsiasi componente hardware compatibile con il sistema operativo. Lo striping del disco con parità protegge il sistema dalla perdita di dati nel caso di guasto del supporto. Con lo striping del disco i dati

vengono scritti in zone separate (stripe) all'interno dello stesso volume o di un array di dischi. Lo striping del disco con parità è simile allo striping del disco ma aggiunge a ogni partizione uno stripe di informazioni relative alla parità. Si ottiene in questo modo un livello di tolleranza agli errori equivalente a quello del mirroring del disco, pur richiedendo una quantità di spazio molto più ridotta. Lo striping del disco con parità basato su Windows NT e i volumi RAID-5 di Windows 2000 implementano il livello RAID 5 (vedi appendice).

- Il mirroring (dischi sullo stesso controller) e il duplexing (dischi su controller differenti) basati su Windows NT e Windows 2000 sono entrambi meccanismi a tolleranza d'errore che proteggono il sistema dalla perdita di dati in caso di guasto di un supporto. Questi meccanismi migliorano inoltre le prestazioni di lettura.

3.5.1 Filegroup

I file di database possono essere raggruppati in *filegroup* ai fini dell'allocazione e dell'amministrazione. L'amministratore di sistema può creare filegroup per ogni unità disco e quindi assegnare tabelle e indici specifici oppure dati **text**, **ntext** o **image** di una tabella a particolari filegroup.

I file di log non vengono mai inclusi in un filegroup. Lo spazio di log viene gestito separatamente rispetto allo spazio dei dati.

Esistono tre tipi di filegroup:

- Primario

  Il filegroup primario include tutte le pagine delle tabelle di sistema e qualsiasi altro file non assegnato in modo specifico a un altro filegroup.

- Definito dall'utente

  I filegroup definiti dall'utente vengono specificati utilizzando la parola chiave FILEGROUP in un'istruzione CREATE DATABASE o ALTER DATABASE. È possibile impostare i filegroup utente in sola lettura.

- Default

  In ogni database è disponibile un solo filegroup predefinito. Se si crea una tabella o un indice senza specificare il filegroup al quale deve appartenere, SQL Server li allocherà nelle

pagine del filegroup predefinito. In assenza si specifiche da parte del DBA il filegroup predefinito è il filegroup primario.

4 Elaborazione parallela di query

Microsoft SQL Server rileva automaticamente il grado di parallelismo migliore per ogni istanza di esecuzione di una query, in base a valutazioni sul numero dei processori, il numero di utenti concorrenti, la memoria disponibile, il numero di righe elaborate. Se l'esecuzione parallela della query è possibile, SQL Server determina il numero ottimale di thread e suddivide l'esecuzione della query tra i thread.

Per monitorare il grado di parallelismo per le singole istruzioni è possibile utilizzare SQL Profiler, tramite la classe di evento *Degree Of Parallelism.*

EventClass	TextData	ApplicationName	NTUserName	LoginName	CPU	Reads	Wri
Degree of Parallelism (7.0 Insert)		SQLAgent - ...	Nicola	LABNE...			
Degree of Parallelism (7.0 Insert)			Furfari	LABNE...			
Degree of Parallelism (7.0 Insert)		SQLAgent - ...	Nicola	LABNE...			
Degree of Parallelism (7.0 Insert)		SQLAgent - ...	Nicola	LABNE...			
Degree of Parallelism (7.0 Insert)		SQLAgent - ...	Nicola	LABNE...			
Degree of Parallelism (7.0 Insert)		SQLAgent - ...	Nicola	LABNE...			
Degree of Parallelism (7.0 Insert)			Furfari	LABNE...			
Degree of Parallelism (7.0 Insert)		SQLAgent - ...	Nicola	LABNE...			
Degree of Parallelism (7.0 Insert)		SQLAgent - ...	Nicola	LABNE...			
Degree of Parallelism (7.0 Insert)		SQLAgent - ...	Nicola	LABNE...			
Degree of Parallelism (7.0 Insert)		SQLAgent - ...	Nicola	LABNE...			
Degree of Parallelism (7.0 Insert)			Furfari	LABNE...			
Degree of Parallelism (7.0 Insert)			Furfari	LABNE...			

4.1 Transazioni distribuite

La gestione della transazione è coordinata tra i vari gestori delle risorse tramite un componente server denominato gestore delle transazioni. I database di Microsoft SQL Server possono partecipare alle transazioni distribuite, coordinate da un qualsiasi gestore conforme a X/Open XA, quali ad esempio MS DTC (Microsoft Distributed Transaction Coordinator).

A livello dell'applicazione le transazioni distribuite vengono gestite in modo simile alle transazioni locali. Al termine della transazione l'applicazione ne richiede il commit o il rollback. Il commit distribuito è realizzato dal gestore con un protocollo di commit in due fasi. Le applicazioni SQL Server possono gestire le transazioni distribuite utilizzando sia Transact-SQL che le API di database.

4.2 Servizio MS DTC

Il servizio MS DTC (Microsoft Distributed Transaction Coordinator) è un gestore delle transazioni che consente alle applicazioni client di includere sorgenti di dati diverse in una singola transazione. MS DTC coordina il commit della transazione distribuita in tutti i server coinvolti, con un protocollo a due fasi. Un'installazione di SQL Server può partecipare a una transazione distribuita tramite:

- Chiamata di stored procedure in server remoti che eseguono SQL Server.
- Esecuzione di aggiornamenti distribuiti in più sorgenti OLE DB.

Nell'esempio seguente il cognome dell'autore viene aggiornato nei database locale e remoto. Come si può notare, la topologia della distribuzione dei dati deve essere nota al programmatore, però l'atomicità della transazione distribuita è gestita da SQL Server.

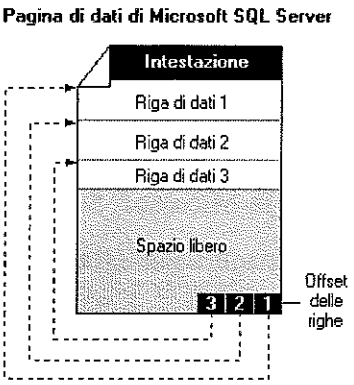
```
USE pubs
GO
BEGIN DISTRIBUTED TRANSACTION
UPDATE authors
   SET au_lname = 'McDonald' WHERE au_id = '409-56-7008'
EXECUTE remote.pubs.dbo.changeauth_lname '409-56-7008','McDonald'
COMMIT TRAN
GO
```

In una configurazione di cluster di server, se il nodo sul quale è attivo MS DTC si blocca, il gestore delle transazioni viene automaticamente riavviato in un altro nodo. La nuova istanza di MS DTC determina in base al file di log, presente sul disco condiviso del cluster, il risultato di transazioni in corso o completate di recente ed esegue il ripristino delle transazioni in dubbio.

5 Ottimizzazione delle prestazioni del server

Microsoft SQL Server ottimizza automaticamente molte delle opzioni di configurazione del server. In caso di necessità è tuttavia possibile ottimizzare le prestazioni del server modificando i componenti seguenti:

- Memoria di SQL Server
- Sottosistema di I/O

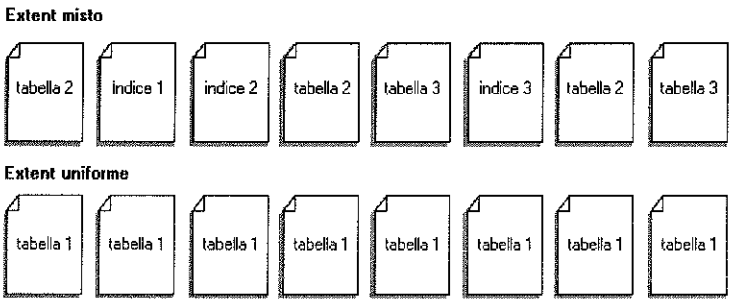


In SQL Server un record non può estendersi su più pagine, la lunghezza massima è di 8060 byte, escludendo le colonne **text**, **ntext** e **image**.

Sono disponibili due tipi di extent:

- Gli extent uniformi sono di proprietà di un unico oggetto, pertanto le otto pagine dell'extent possono essere utilizzate solo da tale oggetto.
- Gli extent misti possono essere condivisi al massimo da otto oggetti.

A una nuova tabella o un nuovo indice vengono in genere allocate pagine di extent misti. Se la tabella o l'indice aumenta di dimensioni fino a includere otto pagine, vengono allocate pagine di extent uniformi.



6.3 File fisici e filegroup del database

I database di SQL Server 2000 includono tre tipi di file:

- File di dati primari

  Il file di dati primario è il punto di partenza e fa riferimento agli altri file del database. In ogni database è disponibile un unico file di dati primario.

- File di dati secondari

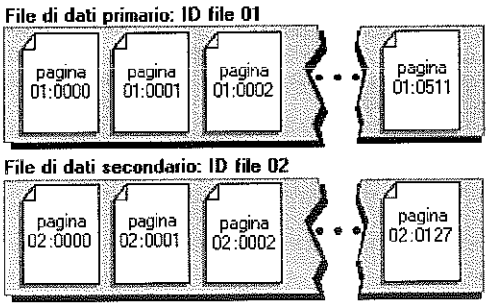
  I file di dati secondari includono tutti gli altri file di dati ad eccezione del file di dati primario.

- File di log

  I file di log includono tutte le informazioni sulla registrazione delle attività utilizzabili per il recupero del database.

I percorsi di tutti i file di un determinato database vengono registrati sia nel file primario che nel database **master**.

Le pagine dei file di dati sono numerate progressivamente. A ogni file viene assegnato un ID numerico. Nell'esempio seguente vengono indicati i numeri di pagina di un database che include un file di dati primario di 4 MB e un file di dati secondario di 1 MB.



La prima pagina di ogni file è la pagina dell'intestazione, che include informazioni sugli attributi del file, le mappe di allocazioni, le proprietà del database, etc.

Le dimensioni dei file possono aumentare automaticamente rispetto ai valori originari in base a un algoritmo round-robin.

6.4 Gestione delle allocazioni di extent e dello spazio libero

In SQL Server vengono utilizzati due tipi di mappa delle allocazioni degli extent:

- Mappa di allocazione globale (GAM, Global Allocation Map)

  Nelle pagine GAM vengono registrati gli extent allocati (fino a 64.000 cioè circa 4 GB). La pagina GAM include un bit per ogni extent dell'intervallo che la riguarda (1 disponibile, 0, allocato).

- Mappa di allocazione globale condivisa (SGAM, Shared Global Allocation Map)

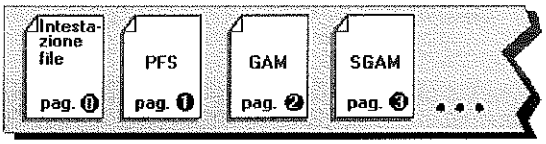
  Nelle pagine SGAM vengono registrate informazioni sugli extent misti con almeno una pagina inutilizzata.

Nelle pagine GAM e SGAM, per ogni extent sono impostati gli schemi di bit seguenti in base all'utilizzo corrente dell'extent.

Utilizzo corrente dell'extent	Impostazione del bit nella pagina GAM	Impostazione del bit nella pagina SGAM
Libero, non utilizzato	1	0
Extent uniforme oppure extent misto senza spazio libero	0	0
Extent misto con pagine libere	0	1

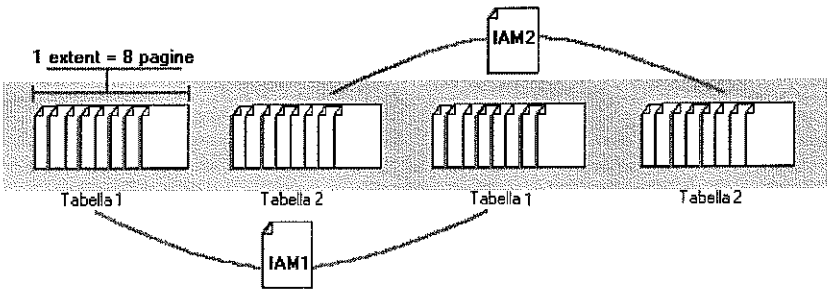
Nelle pagine PFS (Page Free Space) vengono registrate informazioni che indicano se una pagina specifica è stata allocata in un heap o in una colonna **ntext**, **text** o **image** e informazioni sulla quantità di spazio libero in ogni pagina. Ogni pagina PFS include informazioni relative a circa 8.000 pagine e una mappa di bit per ogni pagina. La mappa di bit indica se la pagina è vuota, oppure la percentuale di completamento: 1-50%, 51-80%, 81-95% o 96-100%.

La prima pagina dopo la pagina dell'intestazione di un file di dati (con il numero di pagina 1) è una pagina PFS. Segue una pagina GAM (con il numero di pagina 2) e quindi una pagina SGAM (pagina 3).

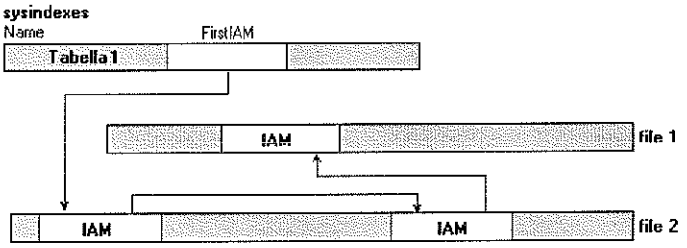


6.4.1 Gestione dello spazio utilizzato dagli oggetti

Nelle pagine IAM (Index Allocation Map) vengono mappati gli extent di un file di database utilizzati da un heap o da un indice. Nelle pagine IAM vengono inoltre mappati gli extent allocati alla sequenza di pagine **ntext**, **text** e **image**. A ognuno di questi oggetti è associata una sequenza di una o più pagine IAM nelle quali sono registrate informazioni su tutti gli extent allocati all'oggetto.



Le pagine IAM vengono allocate per ogni oggetto in base alla necessità e vengono posizionate in modo casuale nel file **sysindexes**.



Il numero di pagine IAM e PFS di un database è ridotto, poiché ognuna di esse include informazioni relative a un numero elevato di pagine di dati. Ciò significa che le pagine IAM e PFS in genere risiedono nel pool di buffer di SQL Server.

Ulteriori strutture dati vengono mantenute per le operazioni di backup differenziale e di bulk copy.

6.4.2 Organizzazione delle pagine di dati

Nelle tabelle di SQL Server 2000 vengono utilizzati due metodi di organizzazione delle pagine di dati:

- Le tabelle con un indice cluster.

  I record vengono archiviati con ordinamento basato sulla chiave dell'indice. L'indice viene implementato come struttura b-tree. Le pagine in ogni livello dell'indice, comprese le pagine di dati nel livello foglia, sono collegate in un elenco con collegamento doppio.

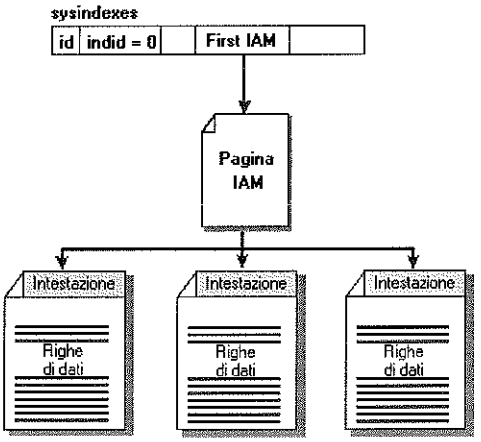
- Tabelle senza un indice cluster.

  I record vengono memorizzati in un heap senza un ordine specifico. Ciò vale anche per la sequenza delle pagine di dati, che non sono tra loro collegate.

Le viste indicizzate hanno la stessa struttura di archiviazione delle tabelle cluster. Tutti i dati **text**, **ntext** e **image** di una tabella vengono archiviati in un unico insieme di pagine.

6.4.3 Struttura degli heap

Agli heap corrisponde una riga in **sysindexes** con **indid** = 0. La colonna **sysindexes.FirstIAM** punta alla prima pagina IAM della sequenza che gestisce lo spazio allocato all'heap. L'unico collegamento logico tra le pagine di dati è quello registrato nelle pagine IAM.



6.5 Pool di memoria di SQL Server

Ogni istanza di Microsoft SQL Server 2000 dispone di uno spazio degli indirizzi con due componenti principali:

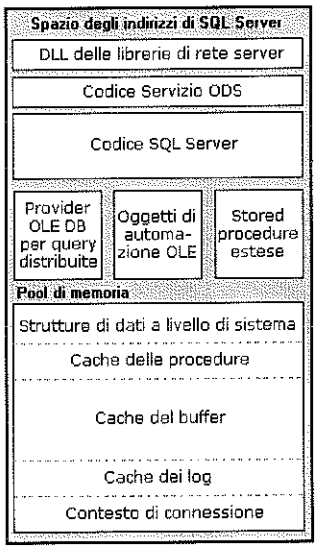
- Codice eseguibile

  Oltre ai file eseguibili e alle DLL utilizzate dal motore di SQL Server e dalle librerie di rete, contiene eventuali query distribuite e stored proceure. Le query distribuite possono caricare una DLL del provider OLE DB nel server che esegue l'istanza di SQL Server.

- Pool di memoria

  Nel pool di memoria vengono allocate quasi tutte le strutture di dati di un'istanza di SQL Server. I tipi principali di oggetto allocati nel pool di memoria sono i seguenti:

- Strutture dati di sistema, ad esempio i descrittori del database e la tabella dei lock
- Cache dei dati
- Cache delle procedure
- Cache dei log
- Contesto di connessione



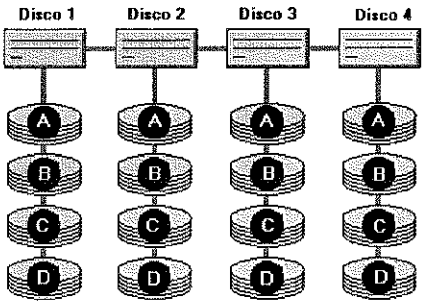
Lo spazio dello stack per ogni thread avviato da SQL Server viene gestito dal sistema operativo[i].

6.6 Livelli RAID e SQL Server

In genere con Microsoft SQL Server vengono implementati i livelli RAID (Redundant Array of Independent Disks) 0, 1 e 5. I livelli RAID superiori a 10 (1 + 0), in genere di sistemi proprietari, consentono di ottenere livelli di prestazioni e tolleranza agli errori ancora migliori.

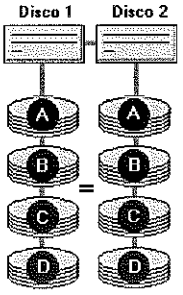
Livello 0

Questo livello è denominato anche striping del disco, in quanto utilizza un file system su disco denominato set di striping. I dati vengono suddivisi in blocchi e distribuiti in un ordine fisso in tutti i dischi dell'array. Il livello RAID 0 determina un miglioramento delle prestazioni di lettura/scrittura tramite la distribuzione delle operazioni in più dischi e l'esecuzione indipendente e simultanea delle operazioni stesse, ma non fornisce tolleranza agli errori.



Livello 1

Questo livello è detto anche mirroring del disco, in quanto utilizza un file system su disco denominato set di mirroring. Il mirroring del disco crea una copia identica e ridondante di un disco selezionato. Tutti i dati creati nel disco primario vengono copiati nel disco di mirroring. Il livello RAID 1 fornisce tolleranza d'errore e migliora in generale le prestazioni a livello di lettura (ma può ridurre le prestazioni a livello di scrittura).



Livello 3

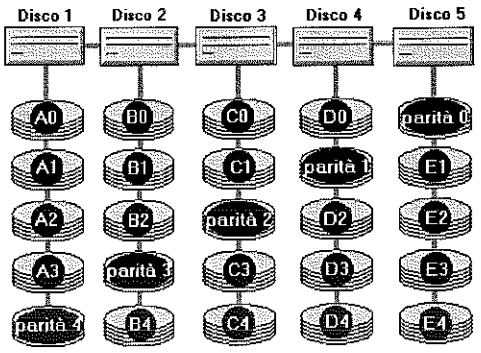
Questo livello utilizza lo stesso metodo di striping di RAID 2, ma il metodo di correzione degli errori richiede un solo disco per i dati di parità. L'utilizzo dello spazio varia a seconda del numero dei dischi dedicati. Il livello RAID 3 consente di ottenere un miglioramento limitato delle prestazioni di lettura/scrittura.

Livello 4

Questo livello utilizza i dati di striping in blocchi o segmenti più grandi di quelli utilizzati da RAID 2 o RAID 3. Come in RAID 3 il metodo di correzione degli errori richiede un solo disco per i dati di parità. I dati utente vengono separati dai dati di correzione degli errori. Il livello RAID 4 non è efficiente quanto altri livelli RAID e in genere non viene utilizzato

Livello 5

Denominato anche striping con parità, questo livello è la strategia più diffusa per le progettazioni recenti. La ridondanza dei dati viene fornita dalle informazioni di parità. I dati e le informazioni di parità sono disposte nell'array in modo da trovarsi sempre su dischi distinti. Lo striping con parità offre prestazioni migliori del mirroring del disco (RAID 1). Tuttavia, quando un membro del set di striping non è disponibile (ad esempio in caso di guasto di un disco) le prestazioni di lettura si riducono.



Livello 10 (1+0)

Questo livello è definito anche mirroring con striping. La configurazione utilizza un array di dischi con striping, dei quali viene eseguito il mirroring in un altro set identico di dischi con striping. Il livello RAID 10 unisce i vantaggi a livello di prestazioni offerti dallo striping ai vantaggi della ridondanza dei dati propria del mirroring, con le migliori prestazioni di lettura/scrittura rispetto a tutti gli altri livelli RAID, ma richiedendo l'utilizzo di un numero di dischi doppio.

[i] La fonte delle informazioni contenute in questo lavoro è in gran parte costituita dalla documentazione in linea di Microsoft SQL Server 2000.