==========

Query optimizer

The optimizer objective is to find alternative ways to execute a query to evaluate the "cost" of each alternative then to choose the one with the lowest cost. To understand the function of the query optimizer, let's use a simple example. Assume that you want to list all products provided by a vendor based in Florida. To acquire that information. you could write the following query:

مُحسِّن الاستعلام

هدف المحسن هو إيجاد طرق بديلة لتنفيذ استعلام لتقييم "تكلفة" كل بديل ثم اختيار البديل الأقل تكلفة. لفهم وظيفة مُحسِّن طلب البحث ، دعنا نستخدم مثالاً بسيطًا. افترض أنك تريد سرد جميع المنتجات التي يقدمها بائع مقره في فلوريدا. للحصول على تلك المعلومات. يمكنك كتابة الاستعلام التالي:

```
SELECT    P_CODE, P_DESCRIPT, P_PRICE, V_NAME, V_STATE
FROM      PRODUCT, VENDOR
WHERE     PRODUCT.P_CODE = VENDOR.V_CODE
          AND VENDOR.V_STATE='FL'
```

Furthermore, let's assume that the database statistics indicate that:

- The PRODUCT table has 7,000 rows.
- The VENDOR table has 300 rows.
- Ten vendors are located in Florida.
- One thousand products come from vendors in Florida.

علاوة على ذلك ، لنفترض أن إحصائيات قاعدة البيانات تشير إلى ما يلي:

- يحتوي جدول المنتج على 7000 صف.
- يحتوي جدول VENDOR على 300 صف.
- يوجد عشرة بائعين في فلوريدا.
- يأتي ألف منتج من البائعين في فلوريدا.

It's important to point out that only the first two items are available to the optimizer. The second two items are to illustrate the choices that the optimizer must make. Armed with the information in the first two items, the optimizer would try to find the most efficient way to access the data. The primary factor in determining the most efficient access plan is the I/O cost. (Remember, the DBMS always tries to minimize I/O operations.) Table bellow shows two sample access plans for the previous query and their respective I/O costs.

من المهم الإشارة إلى أن أول عنصرين فقط متاحان للمحسن. العنصران الثانيان هما لتوضيح الاختيارات التي يجب أن يقوم بها المُحسِّن. مسلحًا بالمعلومات الموجودة في العنصرين الأولين ، سيحاول المُحسِّن العثور على الطريقة الأكثر فعالية للوصول إلى البيانات. العامل الأساسي في تحديد خطة الوصول الأكثر كفاءة هو تكلفة الإدخال / الإخراج. (تذكر أن نظام إدارة قواعد البيانات يحاول دائمًا تقليل عمليات الإدخال / الإخراج.) يعرض الجدول أدناه خطتي وصول نموذجيتين للاستعلام السابق وتكاليف الإدخال / الإخراج الخاصة بهما.

Comparing Access Plans and I/O Costs

PLAN	STEP	OPERATION	I/O OPERATIONS	I/O COST	RESULTING SET ROWS	TOTAL I/O COST
A	A1	Cartesian product (PRODUCT, VENDOR)	7,000 + 300	7,300	2,100,000	7,300
	A2	Select rows in A1 with matching vendor codes	2,100,000	2,100,000	7,000	2,107,300
	A3	Select rows in A2 with V_STATE = 'FL'	7,000	7,000	1,000	2,114,300
B	B1	Select rows in VENDOR with V_STATE = 'FL'	300	300	10	300
	B2	Cartesian Product (PRODUCT, B1)	7,000 + 10	7,010	70,000	7,310
	B3	Select rows in B2 with matching vendor codes	70,000	70,000	1,000	77,310

The objective of a query optimization routine is to minimize the total cost associated with the execution of a request. The costs associated with a request are a function of the:

- Access time (I/O) cost involved in accessing the physical data stored on disk.
- Communication cost associated with the transmission of data among nodes in distributed database system.
- CPU time cost associated with the processing overhead of managing distributed transactions.

الهدف من إجراء تحسين الاستعلام هو تقليل التكلفة الإجمالية المرتبطة بتنفيذ الطلب. التكاليف المرتبطة بالطلب هي دالة لـ:

- تكلفة وقت الوصول (I / O) المتضمنة في الوصول إلى البيانات المادية المخزنة على القرص.
- تكلفة الاتصال المرتبطة بنقل البيانات بين العقد في نظام قاعدة البيانات الموزعة.
- تكلفة وقت وحدة المعالجة المركزية المرتبطة بتكاليف المعالجة لإدارة المعاملات الموزعة.

Most of the algorithms proposed for query optimization are based on two principles:

1. The selection of the optimum execution order.
2. The selection of sites to be accessed to minimize communication costs.

Within those two principles, a query optimization algorithm can be evaluated on the basis of its operation mode - the timing of its optimization.

تستند معظم الخوارزميات المقترحة لتحسين الاستعلام إلى مبدأين:

1. اختيار أمر التنفيذ الأمثل.
2. اختيار المواقع التي سيتم الوصول إليها لتقليل تكاليف الاتصالات.

ضمن هذين المبدأين ، يمكن تقييم خوارزمية تحسين الاستعلام على أساس وضع التشغيل - توقيت تحسينها.

==========

Operation modes can be classified as:

1- **Automatic query optimization** means that the DDBMS finds the most cost-effective access path without user intervention.

2- Manual query optimization requires that optimization be selected and scheduled by the end user or programmer.

يمكن تصنيف أوضاع التشغيل على النحو التالي:

1- يعني تحسين الاستعلام التلقائي أن DDBMS يجد مسار الوصول الأكثر فعالية من حيث التكلفة دون تدخل المستخدم.

2- يتطلب تحسين الاستعلام اليدوي أن يتم اختيار التحسين وجدولته من قبل المستخدم النهائي أو المبرمج.

Query optimization algorithms can also be classified according to when the optimization is done. within this timing classification ,query optimization can be classified as:

1. **Static query optimization** takes place at compilation time. In other words, the best optimization strategy, is selected when the query is compiled by the DBMS. This approach is common when SQL statements are embedded in procedural programming languages such as C# or Visual Basic .NET. When the program is submitted to the DBMS for compilation, it creates the plan necessary to access the database. When the program is executed, the DBMS uses that plan to access the database.
2. **Dynamic query optimization** takes place at execution time, Database access strategy is defined when the program is executed. Therefore, access strategy is dynamically determined by the DBMS at run time, using the most up-to-date information about the database. Although dynamic query optimization is efficient, its cost is measured by run-time processing overhead. The best strategy is determined every time the query is executed; this could happen several times in the same program.

يمكن أيضًا تصنيف خوارزميات تحسين الاستعلام وفقًا لوقت إجراء التحسين. ضمن تصنيف التوقيت هذا ، يمكن تصنيف تحسين الاستعلام على النحو التالي:

1. يتم إجراء تحسين الاستعلام الثابت في وقت التجميع. بمعنى آخر ، يتم تحديد أفضل استراتيجية تحسين عندما يتم تجميع الاستعلام بواسطة نظام إدارة قواعد البيانات (DBMS). هذا الأسلوب شائع عندما يتم تضمين عبارات SQL في لغات البرمجة الإجرائية مثل C # أو Visual Basic .NET. عندما يتم تقديم البرنامج إلى DBMS للتجميع ، فإنه ينشئ الخطة اللازمة للوصول إلى قاعدة البيانات. عند تنفيذ البرنامج ، يستخدم نظام إدارة قواعد البيانات تلك الخطة للوصول إلى قاعدة البيانات.

2. يحدث تحسين الاستعلام الديناميكي في وقت التنفيذ ، ويتم تحديد استراتيجية الوصول إلى قاعدة البيانات عند تنفيذ البرنامج. لذلك ، يتم تحديد استراتيجية الوصول ديناميكيًا بواسطة DBMS في وقت التشغيل ، باستخدام أحدث المعلومات حول قاعدة البيانات. على الرغم من أن تحسين الاستعلام الديناميكي فعال ، إلا أنه يتم قياس تكلفته من خلال النفقات العامة لمعالجة وقت التشغيل. يتم تحديد أفضل استراتيجية في كل مرة يتم فيها تنفيذ الاستعلام ؛ يمكن أن يحدث هذا عدة مرات في نفس البرنامج.

==========

Query optimization techniques can be classified according to the type of information that is used to optimize the query:

1. **A statistically based query optimization algorithm** uses statistical information about the database. The statistics provide information about database characteristics such as size, number of records, average access time, number of requests serviced, and number of users with access rights. These statistics are then used by the DBMS to determine the best access strategy.

   The statistical information is managed by the DDBMS and is generated in one of two different modes: dynamic or manual. In the **dynamic statistical generation mode**, the DDBMS automatically evaluates and updates the statistics after each access. In the **manual statistical generation mode**, the statistics must be updated periodically through a user-selected utility such as IBM's RUNSTAT command used by DB2 DBMSs.
2. A **rule-based query optimization algorithm** is based on a set of user-defined rules to determine the best query access strategy. The rules are entered by the end user or database administrator, and they typically are very general in nature.

يمكن تصنيف تقنيات تحسين الاستعلام وفقًا لنوع المعلومات المستخدمة لتحسين الاستعلام:

1. تستخدم خوارزمية تحسين الاستعلام الإحصائي المعلومات الإحصائية حول قاعدة البيانات. توفر الإحصائيات معلومات حول خصائص قاعدة البيانات مثل الحجم وعدد السجلات ومتوسط وقت الوصول وعدد الطلبات التي تمت خدمتها وعدد المستخدمين الذين لديهم حقوق وصول. ثم يتم استخدام هذه الإحصائيات بواسطة نظام إدارة قواعد البيانات (DBMS) لتحديد أفضل استراتيجية وصول.

تتم إدارة المعلومات الإحصائية بواسطة DDBMS ويتم إنشاؤها في أحد وضعين مختلفين: ديناميكي أو يدوي. في وضع التوليد الإحصائي الديناميكي ، يقوم DDBMS تلقائيًا بتقييم وتحديث الإحصائيات بعد كل وصول. في وضع الإنشاء الإحصائي اليدوي ، يجب تحديث الإحصائيات بشكل دوري من خلال أداة مساعدة يختارها المستخدم مثل أمر RUNSTAT الخاص بشركة IBM والمستخدم بواسطة DB2 DBMSs.

2. تعتمد خوارزمية تحسين الاستعلام المستندة إلى القواعد على مجموعة من القواعد المحددة من قبل المستخدم لتحديد أفضل استراتيجية للوصول إلى الاستعلام. يتم إدخال القواعد من قبل المستخدم النهائي أو مسؤول قاعدة البيانات ، وعادة ما تكون عامة جدًا في طبيعتها.

USING HINTS TO AFFECT OPTIMIZER CHOICES

Although the optimizer generally performs very well under most circumstances, in some instances the optimizer might not choose the best execution plan. There are some occasions when the end user would like to change the optimizer mode for the current SQL statement. In order to do that, you need to use hints. Optimizer hints are special instructions for the optimizer that are embedded inside the SQL command text. Table bellow summarizes a few of the most common optimizer hints used in standard SQL.

استخدام التلميحات للتأثير على اختيارات المُحسِّن

على الرغم من أن المحسن يؤدي بشكل عام أداءً جيدًا في معظم الظروف ، إلا أنه في بعض الحالات قد لا يختار المحسن أفضل خطة تنفيذ. هناك بعض المناسبات التي يرغب فيها المستخدم النهائي في تغيير وضع المحسن لعبارة SQL الحالية. للقيام بذلك ، تحتاج إلى استخدام التلميحات. تلميحات المحسن هي إرشادات خاصة للمحسن مضمنة داخل نص أمر SQL. يلخص الجدول أدناه بعض تلميحات المحسن الأكثر شيوعًا المستخدمة في SQL القياسي.

HINT	USAGE
ALL_ROWS	Instructs the optimizer to minimize the overall execution time, that is, to minimize the time it takes to return all rows in the query result set. This hint is generally used for batch mode processes. For example: SELECT /*+ ALL_ROWS */ * FROM PRODUCT WHERE P_QOH < 10;
FIRST_ROWS	Instructs the optimizer to minimize the time it takes to process the first set of rows, that is, to minimize the time it takes to return only the first set of rows in the query result set. This hint is generally used for interactive mode processes. For example: SELECT /*+ FIRST_ROWS */ * FROM PRODUCT WHERE P_QOH < 10;
INDEX(name)	Forces the optimizer to use the P_QOH_NDX index to process this query. For example: SELECT /*+ INDEX(P_QOH_NDX) */ * FROM PRODUCT WHERE P_QOH < 10;

SQL PERFORMANCE TUNING

SQL performance tuning is evaluated from the client perspective. Therefore, the goal is to illustrate some common practices used to write efficient SQL code. A few words of caution are appropriate:

1. Most current-generation relational DBMSs perform automatic query optimization at the server end.
2. Most SQL performance optimization techniques are DBMS-specific, and therefore, are rarely portable. even across different versions of the same DBMS. Part of the reason for this behavior is the constant advancement in database technologies.

ضبط أداء SQL

يتم تقييم ضبط أداء SQL من منظور العميل. لذلك ، فإن الهدف هو توضيح بعض الممارسات الشائعة المستخدمة لكتابة تعليمات برمجية SQL فعالة. بضع كلمات تحذير مناسبة:

1. تقوم معظم نظم إدارة قواعد البيانات العلائقية من الجيل الحالي بتحسين الاستعلام التلقائي في نهاية الخادم.
2. معظم تقنيات تحسين أداء SQL خاصة بنظام إدارة قواعد البيانات ، وبالتالي نادرًا ما تكون محمولة. حتى عبر إصدارات مختلفة من نفس نظام إدارة قواعد البيانات. جزء من سبب هذا السلوك هو التقدم المستمر في تقنيات قواعد البيانات.

Does this mean that you should not worry about how a SQL query is written because the DBMS will always optimize it? No, because there is considerable room for improvement. (The DBMS uses general optimization techniques. rather than focusing on specific techniques dictated by the special circumstances of the query execution.) A poorly written SQL query can, and usually will, bring the database system to its knees from a performance point of view. The majority of current database performance problems are related to poorly written SQL code. Therefore, although a DBMS provides general optimizing services, a carefully written query almost always outperforms a poorly written one.

DBMS PERFORMANCE TUNING

هل هذا يعني أنه لا داعي للقلق بشأن كيفية كتابة استعلام SQL لأن نظام إدارة قواعد البيانات (DBMS) سيعمل دائمًا على تحسينه؟ لا ، لأن هناك مجال كبير للتحسين. (يستخدم نظام إدارة قواعد البيانات (DBMS) تقنيات التحسين العامة. بدلاً من التركيز على تقنيات محددة تمليها الظروف الخاصة لتنفيذ الاستعلام.) يمكن لاستعلام SQL المكتوب بشكل سيئ ، وعادةً ما يؤدي إلى ركب نظام قاعدة البيانات على ركبتيه من

==========

وجهة نظر الأداء. ترتبط غالبية مشكلات أداء قاعدة البيانات الحالية بضعف كتابة تعليمات برمجية SQL. لذلك ، على الرغم من أن نظام إدارة قواعد البيانات (DBMS) يوفر خدمات تحسين عامة ، إلا أن الاستعلام المكتوب بعناية يتفوق دائمًا على الاستعلام المكتوب بشكل سيئ.

ضبط أداء **DBMS**

DBMS performance tuning at the server end focuses on setting the parameters used for :

• Data cache. The data cache must be set large enough to permit as many data requests as possible to be serviced from the cache. Each DBMS has settings that control the size of the data cache; some DBMSs might require a restart. This cache is shared among all database users. The majority of primary memory resources will be allocated to the data cache.

• SQL cache. The SQL cache stores the most recently executed SQL statements (after the SQL statements, have been parsed by the optimizer). Generally, if you have an application with multiple users accessing a database, the same query will likely be submitted by many different users. In those cases, the DBMS will parse the query only once and execute it many times, using the same access plan. In that way, the second and subsequent SQL requests for the same query are served from the SQL cache. skipping the parsing phase.

• Sort cache. The sort cache is used as a temporary storage area for ORDER BY or GROUP BY operations. as well as for index-creation functions.

• Optimizer mode. Most DBMSs operate in one of two optimization modes: cost-based or rule-based. Others automatically determine the optimization mode based on whether database statistics are available.

For example. the DBA is responsible for generating the database statistics that are used by the cost-based optimizer. If the statistics are not available, the DBMS uses a rule-based optimizer.

يركز ضبط أداء DBMS في نهاية الخادم على تعيين المعلمات المستخدمة من أجل:

• ذاكرة التخزين المؤقت للبيانات. يجب تعيين ذاكرة التخزين المؤقت للبيانات كبيرة بما يكفي للسماح بأكبر عدد ممكن من طلبات البيانات ليتم خدمتها من ذاكرة التخزين المؤقت. يحتوي كل DBMS على إعدادات تتحكم في حجم ذاكرة التخزين المؤقت للبيانات ؛ قد تتطلب بعض أنظمة إدارة قواعد البيانات (DBMS) إعادة التشغيل. يتم مشاركة ذاكرة التخزين المؤقت هذه بين جميع مستخدمي قاعدة البيانات. ال
سيتم تخصيص غالبية موارد الذاكرة الأولية لذاكرة التخزين المؤقت للبيانات.

• ذاكرة التخزين المؤقت SQL. تخزن ذاكرة التخزين المؤقت لـ SQL أحدث عبارات SQL التي تم تنفيذها (بعد عبارات SQL ، تم تحليلها بواسطة المُحسِّن). بشكل عام ، إذا كان لديك تطبيق به عدة مستخدمين يصلون إلى قاعدة بيانات ، فمن المحتمل أن يتم إرسال نفس الاستعلام من قبل الكثيرين
مستخدمين مختلفين. في هذه الحالات ، سيقوم DBMS بتحليل الاستعلام مرة واحدة فقط وتنفيذه عدة مرات ، باستخدام نفس خطة الوصول. بهذه الطريقة ، يتم تقديم طلبات SQL الثانية واللاحقة لنفس الاستعلام من ذاكرة التخزين المؤقت لـ SQL. تخطي مرحلة الاعراب.

• فرز ذاكرة التخزين المؤقت. يتم استخدام ذاكرة التخزين المؤقت للفرز كمنطقة تخزين مؤقتة لعمليات ORDER BY أو GROUP BY. وكذلك لوظائف إنشاء الفهرس.

• وضع المحسن. تعمل معظم أنظمة إدارة قواعد البيانات (DBMS) في أحد وضعي التحسين: القائم على التكلفة أو المستند إلى القواعد. يقوم الآخرون تلقائيًا بتحديد وضع التحسين بناءً على ما إذا كانت إحصائيات قاعدة البيانات متاحة أم لا.
علي سبيل المثال. مسؤول قواعد البيانات (DBA) مسؤول عن إنشاء إحصائيات قاعدة البيانات التي يستخدمها المُحسِّن المستند إلى التكلفة. في حالة عدم توفر الإحصائيات ، يستخدم نظام إدارة قواعد البيانات مُحسِّنًا قائمًا على القواعد.