مجلة جامعة تشرين للبحوث والدراسات العلمية – سلسلة العلوم الأساسية المجلد (30) العدد (1) 2008
Tishreen University Journal for Research and Scientific Studies - Basic Sciences Series Vol. (30) No. (1) 2008

تسريع استرجاع البيانات باستخدام الجداول المؤقتة

الدكتور محسن الحسين *
الدكتورأحمد الكردي
محمد حجّوز***

(تاريخ الإيداع 20 / 11 / 2007. قُبِل للنشر في 2008/1/16)

□ الملخّص □

إنّ الشغل الشاغل للمبرمجين ومدراء قواعد البيانات الذين يتعاملون مع كم هائل من البيانات هو الحفاظ على بياناتهم من الاختراق والوصول إليها بأسرع ما يمكن من قبل مستخدمي القاعدة الشرعيين، ولطالما قامت الشركات المصممة لقواعد البيانات على تصميم أنظمة إدارة قواعد بيانات قوية من حيث عدم الاختراق والقدرة الكبيرة على التخزين والسرعة العالية في الوصول إلى البيانات. إلاّ أنه تبقى هناك بعض المصاعب التي قد تواجهنا عند التعامل مع حجم ضخم من البيانات.

لهذا نطرح في بحثنا هذا فكرة جديدة يمكن من خلالها تسريع استرجاع البيانات وذلك باستخدام الجداول المؤقتة. نعتمد في الطريقة الجديدة على الجداول المؤقتة في تسريع استرجاع البيانات، والتي تفيد في بناء استعلامات قوية بروابط متعددة وتخلصُنا من استرجاع سجلات فارغة. وبتطبيق البحث على الجدول (students) الذي يحوي 10145 سجلاً والجدول (cities) الذي يحوي بيانات 14 محافظة، استنتجنا أنه تم توفير حوالي 17.91% من زمن المعالجة و 90.89% من موارد الجهاز.

كلمات مفتاحية:
الصفحات في الشجرات الثنائية، جداول مؤقتة، ربط.

* أستاذ مساعد – كلية الهندسة المعلوماتية . جامعة البعث– سورية.
** مدرس – قسم الرياضيات . كلية العلوم . جامعة البعث – سورية.
*** طالب ماجستير . قسم الرياضيات . كلية العلوم . جامعة البعث– سورية.

مجلة جامعة تشرين للبحوث والدراسات العلمية _ سلسلة العلوم الأساسية المجلد (30) العدد (1) 2008
Tishreen University Journal for Research and Scientific Studies - Basic Sciences Series Vol. (30) No. (1) 2008

Speeding up Retrieving Data Using Temporary Tables

Dr. Mohsen ALHouseen*
Dr. Ahmed Alkordi**
Mohammad Hujjoz***

(Received 20 / 11 / 2007. Accepted 16/1/2008)

□ ABSTRACT □

The most important task of programmers and database administrators, who work on huge data, is to protect data from hijacking and allow legal users to access data as quickly as possible. So, MSDB companies work hard to design strong databases with the capability of storing very large data, very fast accessing, and preventing illegal accessing. But there are some difficulties that may face us when dealing with very huge data.

So, in this research, a new method has been devised to speedup retrieving data by using temporary tables which help us to build dynamic queries with multi joins, and avoid retrieving null's records. Results show that the new method reduced the computer resources by 90.89% and processing time by 17.91%.

Keywords:

Pages in B*trees, Temporary Tables, Join.

* **Associate Professor, Faculty of Informatics Engineering, Albaath UniversityHoms, Syria.**
** **Associate Professor, Faculty of Informatics Engineering, Albaath UniversityHoms, Syria.**
*** **Postgraduate Student, Department of Mathematics, Faculty of Sciences, Albaath University, Homs, Syria.**

1. مقدمة (Introduction):

في طور العولمة المتزايدة، والتضخم الهائل في أرصدة الحياة البشرية في جميع المجالات العلمية والاقتصادية والسياسية والاجتماعية، وللنسبة المرتفعة والمتفجرة في الحاجة المُلحّة للبيانات على جميع الأصعدة، وبحسب الإحصائيات الدولية للنمو المستمر للبيانات والذي يقدر بحوالي 30% سنوياً (حسب إحصائيات المجلس الاستشاري العالمي (*Advisory Council*) الصادرة في 2007/2/22م [1]). وبما أن العالم أصبح قرية صغيرةٍ في متناول الجميع من خلال مواقع الويب المنتشرة انتشاراً واسعاً والتي تعتمد اعتماداً كبيراً على البيانات. كان لابد من إيجاد وسائل ناجعة للتعامل مع هذا الكم الهائل من البيانات، لذلك تم إيجاد قواعد البيانات لتُسهل التعامل معها، ولجأت الشركات المصممة لقواعد البيانات إلى زيادة القدرة التخزينية لها والعمل المستمر على أمثلتها أي جعلها مثالية في التخزين والسرعة، ووضعت أدوات عديدة لعمل ذلك، وبقي الشغل الشاغل لمديري قواعد البيانات هو تأمين السرعة القصوى لاسترجاع البيانات التي يحتاجها المستخدمون مع تطور الشبكة العالمية لنقل البيانات وازدياد عدد مستخدميها.

2. البحث وأهدافه (Research And Objectives):

نظراً للتضخم الهائل في البيانات . كما ذكرنا بالفقرة السابقة . بدأ التفكير في البحث عن أفضل الطرق التي تساعد في استرجاع البيانات والتعامل معها، وأول ما يفكر به مدراء قواعد البيانات لتسريع استرجاع للبيانات هو إنشاء فهارس على الجداول وهي طريقة رائعة ولكنها لا تكفي وقد تنعكس سلباً في بعض الأحيان على عملية تسريع استرجاع للبيانات إذا ما تم استخدام الفهارس بشكلٍ غير ملائم.

هذا ونناقش في هذا البحث إحدى الطرق المستخدمة في تسريع استرجاع البيانات في قواعد البيانات ألا وهي الجداول المؤقتة (Temporary Tables) بعد التعريف بالمشكلة وحتمياتها والأعمال المقدمة في هذا المجال.

3. طريقة البحث والمواد المستخدمة (Research Method & Parts):

1.3 نظام إدارة قواعد البيانات MaxDB:

يقع النظام MaxDB تحت أنظمة إدارة قواعد البيانات العلائقية (Relational Database Management) System (RDBMS) ويعتمد لغة الاستعلام البنيوية (ANSI American National Standards Institute) (ANSI SQL-92)، وتم تسمية هذا النظام بهذا الاسم في عام 2003 بعد أن اتحدت الشركتان MySQL و SAP (Systems, Applications, and Products) في شركة واحدة، عنيت هذه الشركة في تطوير التطبيقات التي تعالج البيانات، وقد ارتفع عدد مستخدمي هذا النظام بشكل كبير في السنوات الأخيرة وبحسب إحصائيات عام 2005 فقد بلغ عدد الدول التي يُستخدم فيها هذا النظام 120 دولة [2]، وقد زادت شعبيته بشكل سريع كونه من الأنظمة مفتوحة المصدر ويمكن تحميله بسرعة وسهولة دون أي تكلفة مادية، كما يمتاز بالوثوقية العالية والشمولية و سهولة التعامل، وسوف نعتمد في شرح طريقة تسريع استرجاع البيانات باستخدام الجداول المؤقتة على هذا النظام علماً أن معظم أنظمة إدارة قواعد البيانات تقوم على المبدأ نفسه ولكن بتعليماتٍ مختلفة.

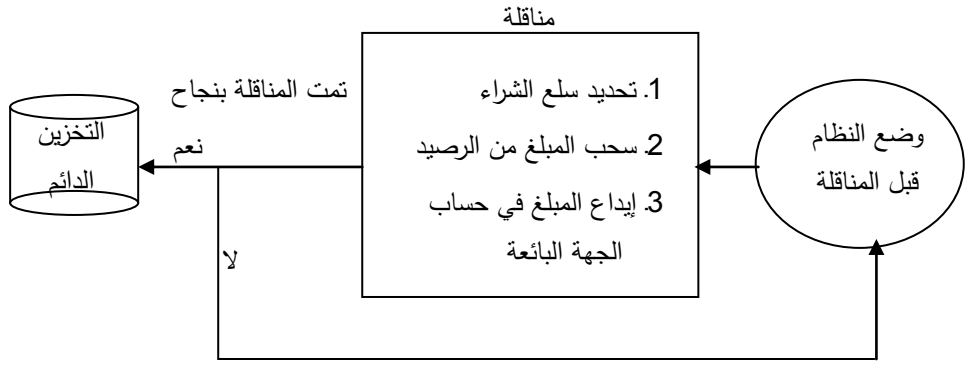
قبل البدء في توضيح عمل البحث، لا بد من توضيح بعض المبادئ الأساسية في عمل قواعد البيانات العلائقية لأنه بفهم هذه المبادئ سوف ندرك أهمية الجداول المؤقتة ولماذا أنها تسرّع عملية استرجاع البيانات.

2.3 المناقلات (Transactions):

المناقلة (المبادلة) هي: وحدة عمل منطقية تحوّل سلسلة متتابعة من العمليات المنفصلة إلى كتلة تنفيذ واحدة. وهي تنقل النظام من وضعية استقرار إلى وضعية استقرار أخرى.

تعتبر المناقلات هامة جداً في الحفاظ على إنجاز مجموعة من العمليات دفعة واحدة أو عدم إنجازها، وهذه الفكرة هامة جداً في عمل قواعد البيانات، فإذا فرضنا أنك تريد إنجاز عملية شراء إلكتروني عبر إحدى مواقع الويب الخاصة بالتجارة الإلكترونية وهذه العملية تتطلب ثلاث عمليات جزئية هي تحديد السلع التي تود شراءها ثم سحب المبلغ المطلوب من حسابك في البنك ثم تحويل المبلغ لحساب الجهة البائعة، افرض أنه لم تتم من العمليات الثلاث السابقة سوى عملية تحديد السلع وعملية سحب المبلغ من حسابك ثم انقطع التيار الكهربائي أو توقف الويب عن العمل، أي أن عملية توديع المبلغ في حساب الجهة البائعة لم يتم، بالتالي توقفت عملية الشراء علماً أنه تم خصم المبلغ المسحوب من حسابك دون فائدة، من هنا تأتي أهمية المناقلات في الحفاظ على إتمام جميع العمليات دفعة واحدة أو عدم إنجاز أي عملية [3].

بذلك نستطيع القول أن المناقلة تعتبر ناجحة إذا نجحت جميع العمليات المنفصلة المتسلسلة التي تتألف منها المناقلة، وتعتبر المناقلة فاشلة إذا فشلت إحدى العمليات المنفصلة المتسلسلة التي تتألف منها المناقلة لأي سبب كان.



الشّكل (1): عمل المناقلة.

يتم حفظ جميع التغييرات التي تمت من بدء المناقلة، فإذا نجحت المناقلة تم تثبيت هذه التغيرات (commit)، أما إذا فشلت نتيجة فشل إحدى العمليات فيتم إعادة النظام إلى وضعه قبل بداية المناقلة والتراجع عن جميع التعديلات التي سببتها العمليات السابقة ضمن المناقلة نفسها. تسمى عملية التراجع تلك بردّ المناقلة (rollback).

- عندما تصبح التعديلات التي قامت بها المناقلة دائمة نقول إن المناقلة قد تمّت. لا يمكن لمناقلة إلا أن تكون ناجحة أو فاشلة أي لا يوجد ما يسمى مناقلة ناجحة جزئياً.

قد نتساءل ما علاقة المناقلات بالجداول المؤقتة. في الحقيقة تُطبّق المناقلات بشكل كامل على الجداول العادية غير المؤقتة، وعند تثبيت المناقلة يتم نقل التغيير المطلوب على البيانات بشكلٍ نهائي من الذاكرة إلى أقراص التخزين، كما أن المناقلة تقوم بتطبيق عدة شروط ووضع خطة لتنفيذ التعليمات لكي تتم دفعة واحدة وهذا كله يستغرق وقتاً لا يستهان به في إنجاز العمليات، أما الجداول المؤقتة . كما سنرى لاحقاً . فإنها تخزن في الذاكرة فقط مما يسرع ويسهل التعامل معها.

3.3 طريقة تخزين البيانات (Data Storage Method) :

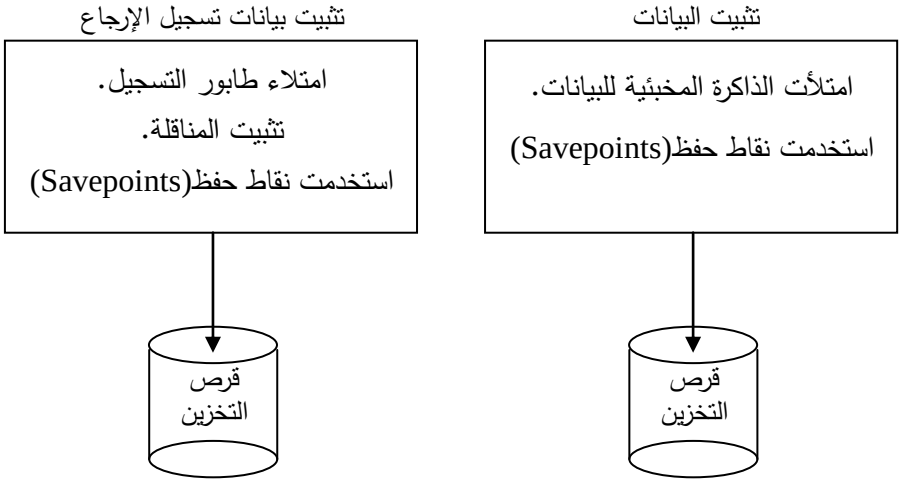
إذا قام مستخدم قاعدة البيانات بتغيير البيانات في القاعدة (أي إجراء مناقلة)، فإن نظام قواعد البيانات يقوم أولاً بتخزين البيانات المُعدّلة في الذاكرة المخبئية للبيانات (Data cache). وتُخزن تغيرات التسجيل في متغير تسجيل الإرجاع في طابور التسجيل (Log Queue) في الذاكرة، ولاحقاً يقوم نظام قواعد البيانات بكتابة التغيرات من الذاكرة المخبئية إلى قرص التخزين الدائم في الحالات التالية:

- إذا امتلأت الذاكرة المخبئية للبيانات.
- إذا استخدمت نقاط حفظ (Savepoints): تستخدم نقاط الحفظ لنقل التغيرات من الذاكرة المخبئية للبيانات إلى أقراص التخزين.

كما يقوم نظام قواعد البيانات بكتابة متغيرات تسجيل الإرجاع من طابور التسجيل إلى ملف التسجيل في القرص في الحالات التالية:

- إذا امتلأ طابور التسجيل.
- إذا تم تثبيت المنافلة باستخدام الأمر (Commit).
- إذا استخدمت نقاط الحفظ (Savepoints).

تفيد متغيرات تسجيل الإرجاع في تسجيل كل التغيرات التي تطرأ على قواعد البيانات واستخدامها عند حدوث أي طارئ في إرجاع وضع قاعدة البيانات إلى ما كانت عليه قبل حدوث المشكلة.



الشّكل (2): تثبيت البيانات وبيانات تسجيل الإرجاع.

كل العمليات السابقة تتم على الجداول العادية، فالمستخدم يتعامل مع البيانات على شكل جداول، وفي الحقيقة هذه الجداول ليست فيزيائية (أي لا تُخزن بياناتها كما تشاهدها في الجدول) فكل بيانات الجداول تخزن في ملف البيانات الخاص بالقاعدة، وتخزن بيانات تسجيل الإرجاع في ملف التسجيل الخاص بالقاعدة، هذا يعني أننا نتعامل مع الجداول بشكل منطقي ريثما يتم نقل بياناتها لقرص التخزين، بينما لا ينطبق هذا الكلام على الجداول المؤقتة كما سنرى لاحقاً.

4.3 طريقة الوصول للبيانات (Data Access Method):

كي ندرك أهمية الجداول المؤقتة لابد من توضيح البنيات المنطقية لتخزين البيانات والطرق المتبعة في البحث

عن البيانات.

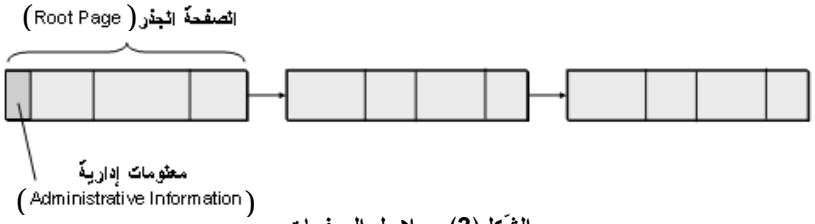
يملك نظام قواعد البيانات البنيات المنطقية التالية:

I- صفحة السلاسل (Page chains)

II- الصفحات في الشجرات الثنائية (Pages in B* trees)

I- صفحة السلاسل (Page chains) :

يستخدمها نظام قواعد البيانات لتخزين مدخلات تسجيل الإرجاع وبيانات الذاكرة المخبئية الحية ليتم الوصول إليها لاحقاً. وهذه الصفحات هي عبارة عن سلسلة من الصفحات المتتالية تجمّع مع بعضها بشكل منطقي. تدعى الصفحة الأولى من هذه السلسلة بالصفحة الجذر (Root Page) وتحوي معلومات الإدارة الداخلية مثل ID الخاص بأخر صفحة.

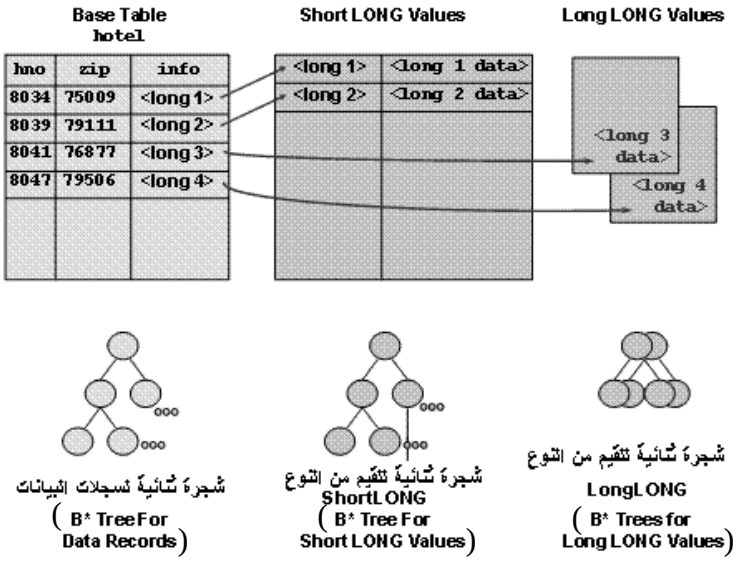


الشّكل(3): سلاسل الصفحات.

II- الصفحات في الشجرات الثنائية (Pages in B* trees) :

ينشأ نظام قاعدة البيانات الشجرات الثنائية عند إنشاء فهارس على الجدول، وتساهم هذه الشجرات في تسريع عملية البحث عن البيانات.

هناك ثلاثة أنواع من هذه الشجرات هي (شجرات ثنائية تنشأ لسجلات البيانات، شجرات ثنائية تنشأ للحقول التي تملك قيماً من نوع (ShortLONG)، و شجرات ثنائية تنشأ للحقول التي تملك قيماً من نوع (longLONG) كما في الشّكل (4).

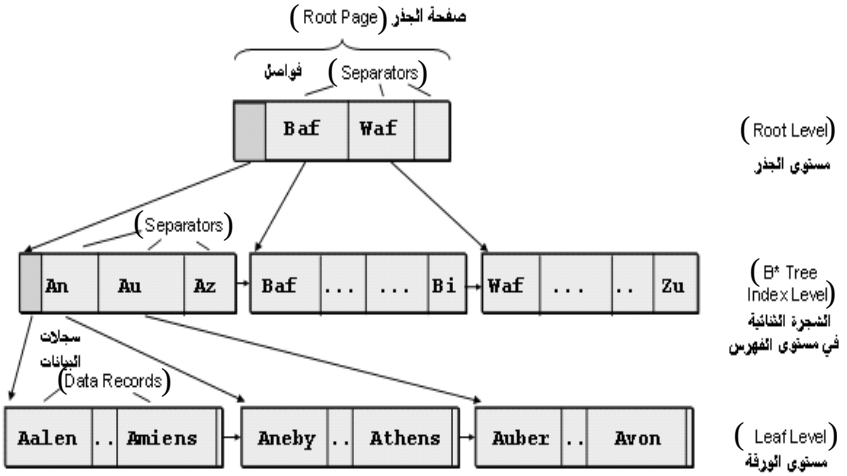


الشّكل(4): أنواع الشجرات الثنائية.

تتألف الشجرة الثنائية من مستوى الجذر (root level) ومستوى الفهرس (Index Level) ومستوى الورقة

(Leaf Level) وكل مستوى يمكن أن يحوي صفحة واحدة أو أكثر.

عندما يبحث نظام قاعدة البيانات عن سجلٍ ما في جدولٍ معين فإنه يبدأ من صفحة الجذر في الشجرة الثنائية وتضيق عملية البحث خطوة بخطوة إلى أن يصل إلى المطلوب كما في الشّكل(5).



الشّكل(5): شجرة ثنائية بثلاث مستويات.

تحوي صفحات الجذر ومستويات الشجرة الثنائية فواصل (Separators) تشير إلى العناوين المنطقية للصفحات (أرقام الصفحات) في المستوى المنخفض التالي. كما يحوي الفاصل الجزء الهام للمفتاح الأولي الذي يستخدم للكشف عن المدخل التالي حتى يكون مختلف عن المدخلات السابقة.

5.3 أنواع الجداول (Table Types):

الجدول بشكل عام يتألف من مجموعة من السجلات وكل سجل يملك العدد نفسه من الحقول تعود للعنصر نفسه وكل حقل يملك نوعاً محدداً من البيانات، ويمكن تقسيم أنواع الجداول إلى ما يلي [4]:

- الجدول الأساسي (Basic Table): تعتبر جداول النظام والجداول التي تحوي بيانات التطبيق جداول أساسية، وتخزّن بيانات هذه الجداول بشكلٍ دائم.
- جداول النتيجة (Result Tables): ينشئ نظام قاعدة البيانات جداول نتيجة دون اسم عند تنفيذ جملة (SELECT) وتحذف هذه الجداول مع انتهاء الجلسة، كما ينشئ النظام جداول نتيجة باسم عند تنفيذ التعليمتين (DECLARE) و(CURSOR).
- جداول الربط (Join Tables): تعتبر هذه الجداول حالة خاصة من جداول النتيجة، حيث ينشئها نظام قاعدة البيانات عند ربط جدولين أو أكثر مع بعضها.
- جدول النظام (System Table): هو جدول أساسي لكنه يحتوي معلومات عن النظام مثل (معلومات حول أغراض القاعدة، حالاتها، إحصائياتها، مراقبة تغيّر البيانات).
- الجداول المؤقتة (Temporary Tables): تعتبر حالة خاصة من الجداول الأساسية إلا أن نظام قواعد البيانات يقوم بحذفها في نهاية الجلسة.
- جدول المشهد (View Table): يُشتق هذا النوع من الجداول من جدول أساسي، وتُخزن تعريفاته بشكل

دائم، ولا يحوي هذا النوع من الجداول البيانات إنما يمكن الحصول على نسخة منها من الجدول الأساسي.

6.3 الجداول المؤقتة (Temporary Tables):

الجداول المؤقتة هي عبارة عن مجموعة من السجلات والحقول، كل حقلٍ يملك نوعاً محدداً من البيانات وتعود بيانات كل سجل لعنصرٍ واحد. تُنشأ هذه الجداول بشكلٍ مؤقت للاستفادة منها أكثر من مرة خلال الجلسة الواحدة أو الاتصال الواحد. تشبه هذه الجداول إلى حدٍ كبير الجداول العادية من حيث الإنشاء والبنية، إلا أنّها لا تخزن بشكلٍ دائمٍ على قرص التخزين كما لا يمكن إجراء عمليات التقسيم (Partitioning) عليها.

في الحقيقة يتجاهل الكثير من مدراء قواعد البيانات والمبرمجون استخدام الجداول المؤقتة حتى أنّ بعضاً منهم يهمشونها بشكلٍ كامل ويستعيضون عنها باستخدام الجداول العادية، وهذا ممكن ولكنه يسبب بعض الإرباك والبطء وتكرار تنفيذ بعض الاستعلامات المعقدة. من هنا بدأت حتميات المشكلة ومن هنا ظهر استخدام الجداول المؤقتة.

دعونا نتعرف في الفقرات القادمة على طريقة إنشاء الجداول المؤقتة وكيف أنّها تسرّع عملية استرجاع البيانات.

I- فوائد الجداول المؤقتة (Temporary Tables Advantages):

في كثيرٍ من الأحيان يقوم المبرمجون أو مدراء قواعد البيانات بالبحث في جدولين أو أكثر عن نتائجٍ (معلومات) يحتاجونها، ويكون بين هذه الجداول علاقات . كما هو معروف في قواعد البيانات العلائقية . وغالباً ما تكون هذه العلاقة من نوع واحد لمتعدد (مثل الطالب ينتمي لمحافظة واحدةٍ أما المحافظة فينتمي إليها أكثر من طالب أو لا ينتمي إليها أحد في حال كان الطالب يحمل شهادة غير سورية)، وهذا قد يسبب مشكلة في إعادة سجلات فارغة لا نريدها، أو علينا بناء استعلامات معقدة وطويلة للوصول إلى الهدف وكل هذا يحتاج لوقتٍ وجهدٍ كبيرين.

بالتالي يمكن تلخيص فوائد الجداول المؤقتة بالنقاط الآتية:

- إنّ من أهم ميّزات الجداول المؤقتة هي توفير الوقت والجهد المصروفين في معالجة عدة استعلامات تستخدم لاسترجاع بيانات ضخمة ومتنوعة، حيث تستخدم لوضع نتائج الاستعلام فيها ومن ثم استخدامها في استعلاماتٍ جديدة بسيطة ومفهومة للحصول على النتائج المطلوبة دون استرجاع سجلات فارغة (Null Records).
- لا تأخذ الجداول المؤقتة حجماً كبيراً في الذاكرة فهي لا تّخزن على أقراص التخزين بل تبقى في الذاكرة طالما الجلسة (Session) مفتوحة أو الاتصال (Connection) موجود ونستخدمها في كل مرة نحتاجها خلال الجلسة، يمكن تشبيه الجداول المؤقتة بالصفوف (Array) في البرمجة. كما لا يمكن للمستخدم خلال جلسته أن ينشئ أكثر من جدول مؤقت بنفس الاسم أي يجب أن يكون الجدول المؤقت ذا اسم فريد.
- تُقدم الجداول المؤقتة فائدة كبيرة في حماية أمن البيانات حيث لا يمكن لأي مستخدم لقاعدة البيانات أن يصل إلى بيانات الجداول المؤقتة في جلسة مستخدم آخر، ويمكن لكل مستخدم أن ينشئ جداول مؤقتة حتى لو كانت تملك الاسم نفسه مع جداول مستخدم آخر فكل مستخدم له جلسته الخاصة.
- تحذف الجداول المؤقتة بشكلٍ تلقائي من الذاكرة مع انتهاء الجلسة أو مع قطع الاتصال بقاعدة البيانات، لذلك سميّت بالجداول المؤقتة، وهذا العمل مفيد جداً في حماية سرية البيانات فقد تخترق قاعدة البيانات والوصول لبياناتها، أما بيانات الجداول المؤقتة فلا يمكن الوصول إليها لأنها تحذف مع انتهاء الجلسة، كما تفيد هذه الميّزة في القضاء على مشكلة تضخم البيانات فقد ينسى المستخدم حذف الجداول العادية التي استعاضها

بالجداول المؤقتة التي أنشأها خلال جلساته أو قد ينقطع التيار الكهربائي أو يحدث خلل في النظام مما يسبب تراكماً في هذه الجداول.

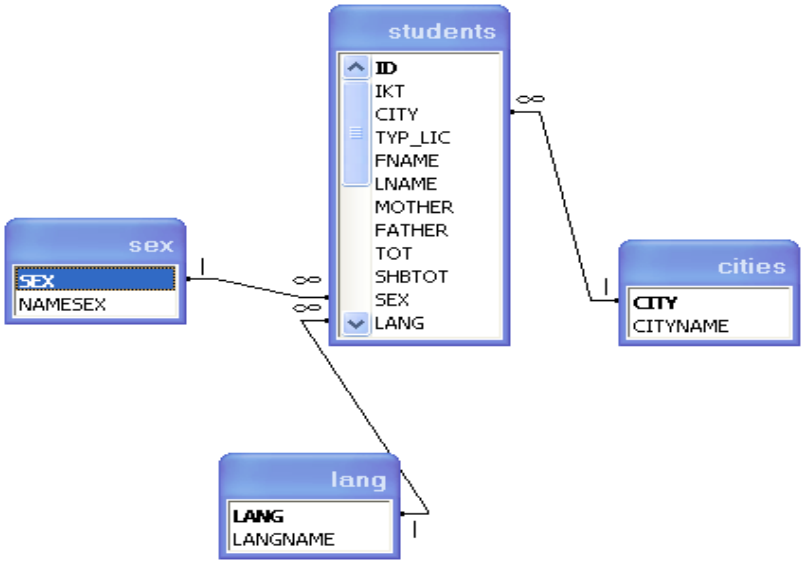
- يمكن إنشاء الجداول المؤقتة من جداول عادية مباشرةً أي لا داعي لإنشاء الجداول المؤقتة أولاً ثم إدخال نتيجة استعلام ما في هذا الجدول كما سنرى في فقرة تطبيق البحث.
- تُقدم الجداول المؤقتة فائدة كبيرة في ربط نتائج استعلام مع جدولٍ أخر ولتوضيح ذلك افرض أنك قمت ببناء استعلام معقد وحصلت بتنفيذه على النتائج المطلوبة، وتريد ربط هذه النتائج مع جدولٍ آخر. فما العمل ياترى بهذه الحالة؟ نقوم في البداية بإنشاء جدول مؤقت من الاستعلام المعقد مباشرةً ثم نربط هذا الجدول المؤقت مع الجدول الأخر.

سنتعرف فيما يلي وبشكلٍ تطبيقي على أهمية وفوائد الجداول المؤقتة.

7.3 تطبيقات البحث (Research Applications):

سنعتمد في تطبيق البحث على نظام إدارة قواعد البيانات العلائقية (SAP MaxDB) كما ذكرنا سابقاً، واستخدام البرنامج (SQL Studio) الخاص بهذا النظام والذي يسمح بكتابة الاستعلامات وتوضيح الوقت والكلفة المطلوبة لتنفيذها، كما سنعتمد على قاعدة بيانات (maxdb) والمستخدم (dbadmin) الذي يمكنه الاتصال بهذه القاعدة.

تتألف هذه القاعدة من أربعة جداول هي جدول المحافظات (cities)، جدول اللغات (lang)، وجدول الجنس (sex) ، وجدول الطلاب (students) وتوجد بين هذه الجداول علاقات موضّحة بالشّكل التالي:



الشّكل(6): العلاقات الموجودة بين جداول القاعدة (maxdb).

- يحوي الجدول (cities) بيانات المحافظات السورية التي ينتمي إليها الطلاب.
- يحوي الجدول (sex) جنس كل طالب (ذكراً أم أنثى).
- يحوي الجدول (lang) اللغات الأجنبية التي يدرسها الطالب.
- يحوي الجدول (students) بيانات الطلاب البالغ عددهم 110145 طالباً وطالبة.

سنقوم بتطبيق البحث على الجدولين (cities) و (students) اللذين يرتبطان مع بعضهما بعلاقة واحد لمتعدد (أي كل طالب ينتمي لمحافظة واحدة أما المحافظة فتحوي الكثير من الطلاب) ونذكر هنا أنه قد تكون هناك شهادات غير سورية، لكن في قاعدتنا هذه لا يوجد لدينا طلاب يحملون شهادات من خارج سورية. لكننا سنقوم في إحدى فقرات تطبيق البحث بإضافة هذا النوع من الشهادات إلى الجدول cities لضرورة توضيح كيف أن الجداول المؤقتة تقضي على عملية استرجاع سجلات فارغة لا حاجة لنا بها.

I- إنشاء الجداول المؤقتة (Create Temporary Tables):

يمكن إنشاء الجداول المؤقتة في نظام إدارة قواعد البيانات MaxDB بطريقتين:

- الطريقة الأولى : إنشاء جدول مؤقت بشكلٍ منفصل غير مرتبط باستعلام ثم إدخال البيانات إليه كما يلي:

إنشاء جدول مؤقت بالاسم (TempTable)

```
CREATE TABLE TEMP.TempTable
(
  CarID integer,
  CarName varchar(50)
)
```

إدخال البيانات إلى الجدول المؤقت (TempTable):

```
INSERT INTO TEMP.TempTable VALUES(1,'NISSAN')
```

- الطريقة الثانية: إنشاء جدول مؤقت من جدولٍ آخر أو من عدة جداول باستخدام الاستعلام كما يلي:

```
CREATE TABLE TEMP.TempTable AS
SELECT * FROM "DBADMIN"."cities"
```

هذا ويتم التعامل مع الجداول المؤقتة كما هو الحال مع الجداول العادية ولا مجال لذكر كل الحالات هنا لأنها خارج بحثنا.

II- استخدام الجداول المؤقتة في تسريع استرجاع البيانات:

(Using Temporary to Speedup Data Retrieval)

سنقوم في هذه الفقرة بتوضيح كيف يمكن للجداول المؤقتة أن تسرّع عملية استرجاع البيانات من خلال ربط الجداول المؤقتة مع جداول عادية ومن خلال التخلص من استرجاع السجلات الفارغة.

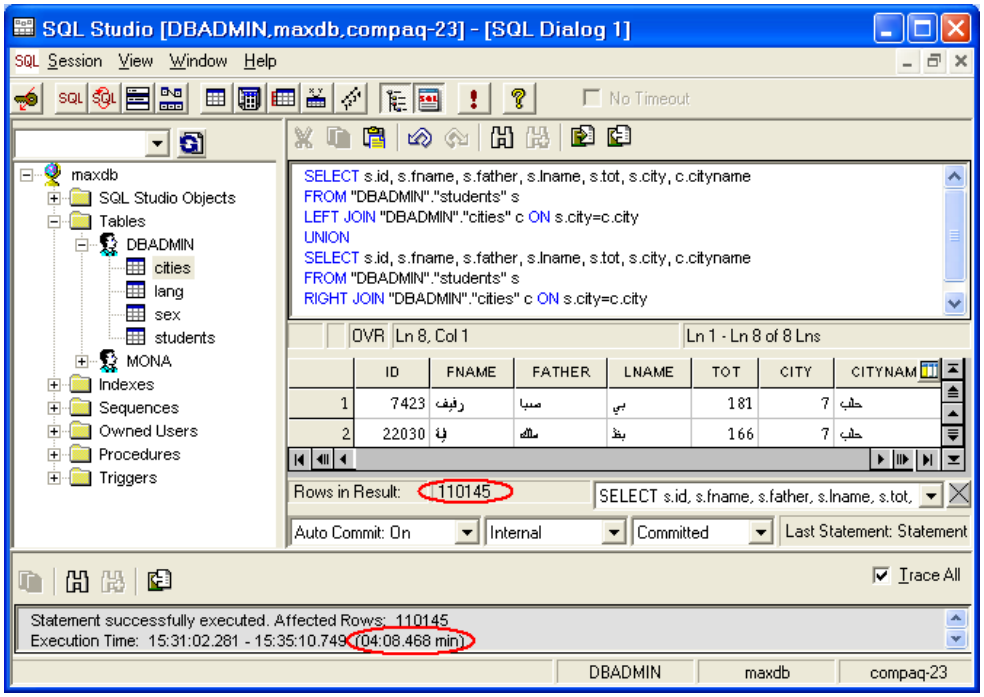
- تسريع استرجاع البيانات باستخدام ربط Join الجداول المؤقتة مع جداول عادية:

كما نعلم أنّ ربط الجداول يتم عن طريق المفتاح الأساسي (Primary Key) والمفتاح الثانوي (Foreign Key) ويملك الربط في أنظمة إدارة قواعد البيانات العلائقية عدة أنواع من أشهرها الربط الداخلي (INNER JOIN) والربط الخارجي (OUTER JOIN) وسوف نعتمد في مثالنا هذا على الربط الخارجي.

يتألف الربط الخارجي من ثلاثة أنواع هي الربط الخارجي الكامل (FULL) والربط الخارجي اليساري (LEFT) والربط الخارجي اليميني (RIGHT). ولتوضيح نتيجة البحث سوف نأخذ الربط الخارجي اليساري والخارجي اليميني على الجدولين (students) و (cities). يعيد الربط الخارجي اليساري جميع السجلات من الجدول الأول (students) وفقط السجلات من الجدول الثاني (cities) التي تتطابق فيها قيمة حقل الربط (city) بين الجدولين ويعيد الربط الخارجي اليميني جميع السجلات من الجدول الثاني (cities) وجميع السجلات من الجدول الأول (students) التي تتطابق فيها قيمة حقل الربط (city) بين الجدولين.

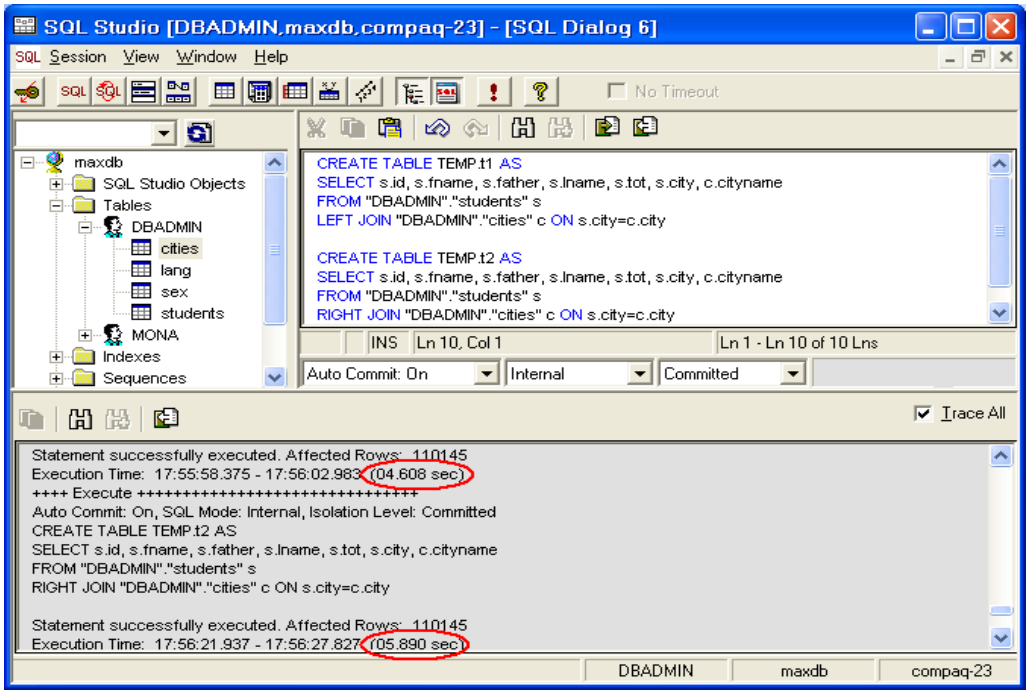
ننوه هنا إلى أن الجدول (cities) يحوي بيانات المحافظات السورية الأربع عشرة فقط دون الشهادات غير السورية. نريد الآن استرجاع بيانات الطلاب مع مجموع درجاتهم ومحافظاتهم من خلال الربط الخارجي اليساري والربط الخارجي اليميني للجدولين السابقين ثم إجراء اتحاد (UNION) بين الاستعلامين كما في الشّكل(7).

بعد التنفيذ نلاحظ أنّ عدد السجلات المسترجعة هو 110145 وهو يمثل جميع بيانات طلاب الجدول (students) أي بعد أن أخذنا السجلات المتوافقة بين الجدولين في كلا الاتجاهين اليساري واليميني وإجراء الاتحاد بينهما فإننا نحصل على بيانات الجدول نفسها (students) لأن الاتحاد لا يجمع السجلات التي تملك نفس الـ IDs في الحقل الأساسي (Primary Key) كما أنه لا يوجد محافظات لا ينتمي إليها طلاب. كما نلاحظ أنّ الوقت المستغرق لتنفيذ العملية هو 04:08:468 دقيقة.



الشّكل(7): البيانات المسترجعة من اتحاد استعلامي الربط اليساري والربط اليميني

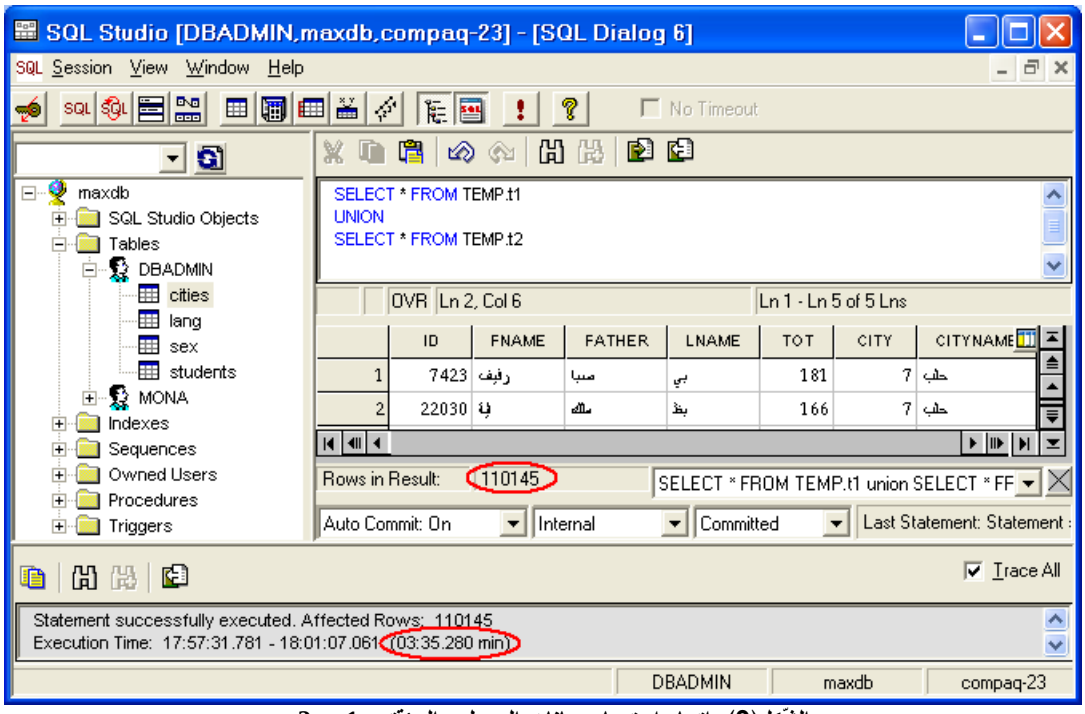
لنحاول الآن القيام بنفس العمل السابق باستخدام الجداول المؤقتة. لذلك سنقوم بإنشاء جدولين مؤقتين من الجدول (students) عن طريق الربط اليساري والربط اليميني مع الجدول (cities) بالاسمين t1 و t2 كما بالشّكل(8).



الشّكل(8):إنشاء جدولين مؤقتين من الجدولين(students) و(cities) باستخدام الربط اليساري واليميني.

نلاحظ من الشّكل(8) أنه تم إنشاء جدولين مؤقتين بسجلات مضاعفة عددها 220290 سجل في كل جدول 110145 سجل وبوقتٍ مستغرق (10:498 ثانية) وذلك بجمع الوقت المستغرق لإنشاء كلا الجدولين (04:608 + 05:890). علماً أن هذا الوقت لا يدخل ضمن الوقت المستهلك لتوضيح الفرق بين الطريقتين لأن الجدول المؤقت ينشأ مرة واحدة ويستخدم في مواضع كثيرة.

لنستعرض الآن سجلات الجدولين المؤقتين t1 و t2 باستخدام (UNION) كما بالشّكل(9).

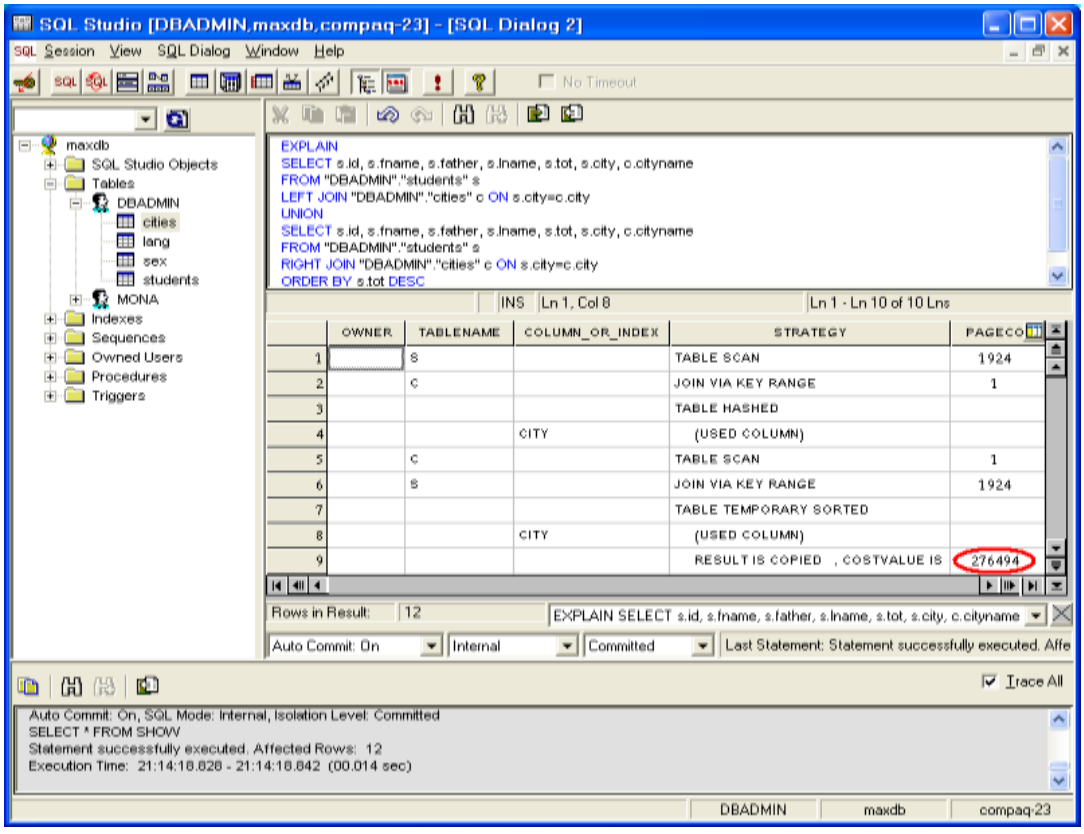


الشّكل(9): اتحاد استرجاع بيانات الجدولين المؤقتين t1 و t2.

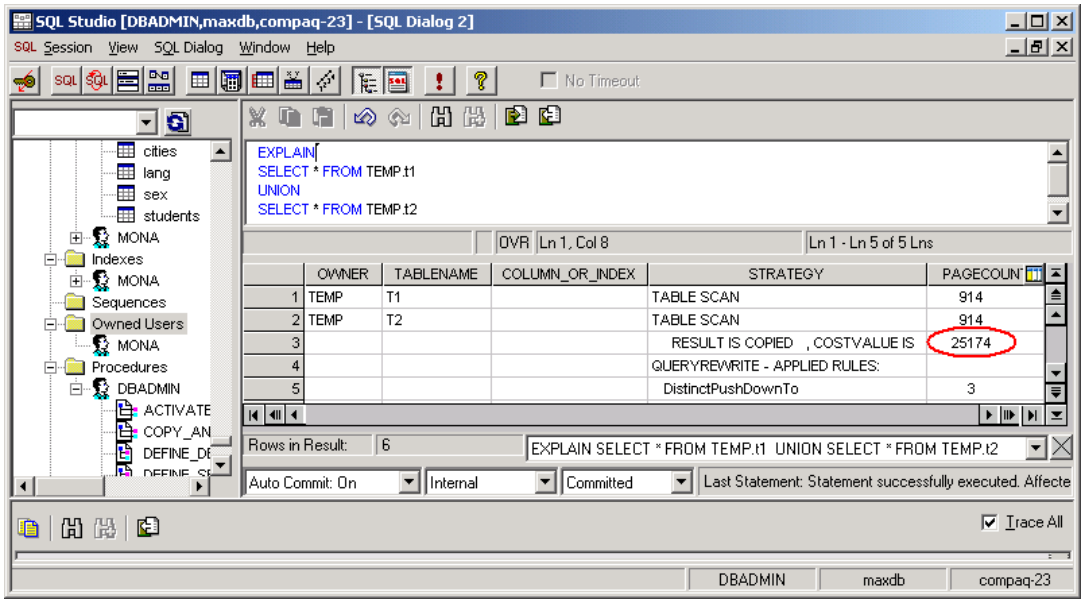
نلاحظ من الشكلين (7) و (9) أن كلا الاستعلامين يعيدان نفس عدد السجلات وهو 110145 سجل ولكن هناك فارقاً زمنياً بين الطريقتين يقدر (33:188 ثانية) وذلك بطرح الوقت المستغرق لكلتا الطريقتين (03:35:280 – 04:08:468). ولتوضيح فوارق الكلفة بين الطريقتين سنستخدم الجملة (EXPLAIN) التي تفيد في معرفة آلية تنفيذ جمل SQL وكلفة الموارد التي تحتاجها كل جملة والتي تحوي الحقول التالية:

مالك الجداول (OWNER) وحقل اسم الجداول (TABLENAME) وحقل الأعمدة المفهرسة (COLUMN_OR_INDEX) وحقل إستراتيجية عمل الجملة (STRATEGY) الذي يحدد نوع الخوارزمية المستخدمة لتحسين الوصول إلى البيانات والذي يحوي على مدى مفتاح الربط المستخدم وقيمة التكلفة، وحقل عدد الصفحات (PAGECOUNT) الذي يحدد عدد الصفحات التي يحتاجها الجدول لتنفيذ هذه الجملة، (وتستخدم الصفحات (Pages) لتنفيذ عمليات القراءة والكتابة فكلما كان عدد الصفحات أقل كلما كان التنفيذ أسرع، كما ذكرنا ذلك سابقاً في طريقة الوصول للبيانات).

نكتب الآن العبارتين الخاصتين بكل طريقة كما يلي:



الشّكل: (10) كلفة تنفيذ استعلام الطريقة الأولى.



الشّكل: (11) كلفة تنفيذ استعلام الطريقة الثانية.

نستنتج من الشكلين (10) و(11) أنّ الطريقة الثانية قد وفرت عدداً من الصفحات يقدر بحوالي 251320 صفحة ينتج عن طرح كلفة الطريقة الأولى من الثانية (276494 – 25174) وبالتالي هناك عدد أقل من القراءات والكتابات تمت على البيانات من وإلى وسائط التخزين والذاكرة.

ونذّكر هنا أنّ جميع التجارب تمت على كمبيوتر بسرعة 3000 ميغا هرتز وذاكرة 1.5 جيجا بايت، ويرتفع الفارق في الزمن والكلفة مع ازدياد عدد الجداول والروابط الموجودة بينها وعدد السجلات في هذه الجداول. ونوضح في الجدول التالي الفوارق بين الطريقتين وحساب النسبة المئوية لهذا الفارق بتقسيم العدد الصغير على العدد الكبير ثم طرحه من 1 ثم ضربه بـ 100 كما يلي:

الجدول (1): فرق الزمن والكلفة والنسبة المئوية بين الطريقتين.

	الطريق الأولى	الطريق الثانية	الفرق
الوقت المستغرق مقدراً بالدقيقة	04:08:468	03:35:280	33:188
عدد الصفحات المُتطلب	276494	25174	251320
النسبة المئوية لفرق الوقت المستغرق لكلتا الطريقتين			17.91
النسبة المئوية لفرق كلفة الموارد لكلتا الطريقتين			90.89

- استخدام الجداول المؤقتة يسّرع استرجاع البيانات ويقضي على ظهور سجلات فارغة:

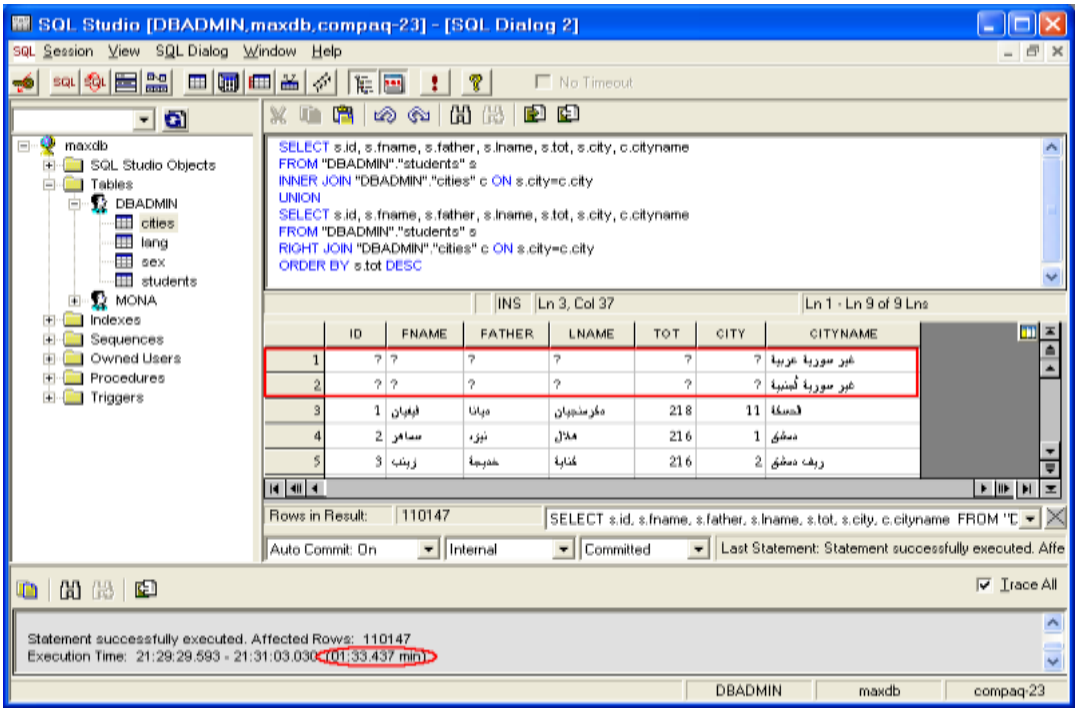
قد يحدث أحياناً . عند ربط الجداول مع بعضها . استرجاع سجلات فارغة، وذلك بسبب عدم التطابق الكامل بين سجلات الجدول الأول مع الثاني، ولتوضيح ذلك نفرض أن جدولاً (cities) يحوي البيانات التالية:

الجدول (2): جدول المحافظات.

CityName	City	CityName	City
درعا	9	دمشق	1
دير الزور	10	ريف دمشق	2
الحسكة	11	حمص	3
إدلب	12	حماة	4
القنيطرة	13	طرطوس	5

6	اللاذقية	14	الرقة
7	حلب	15	غير سورية عربية
8	السويداء	16	غير سورية أجنبية

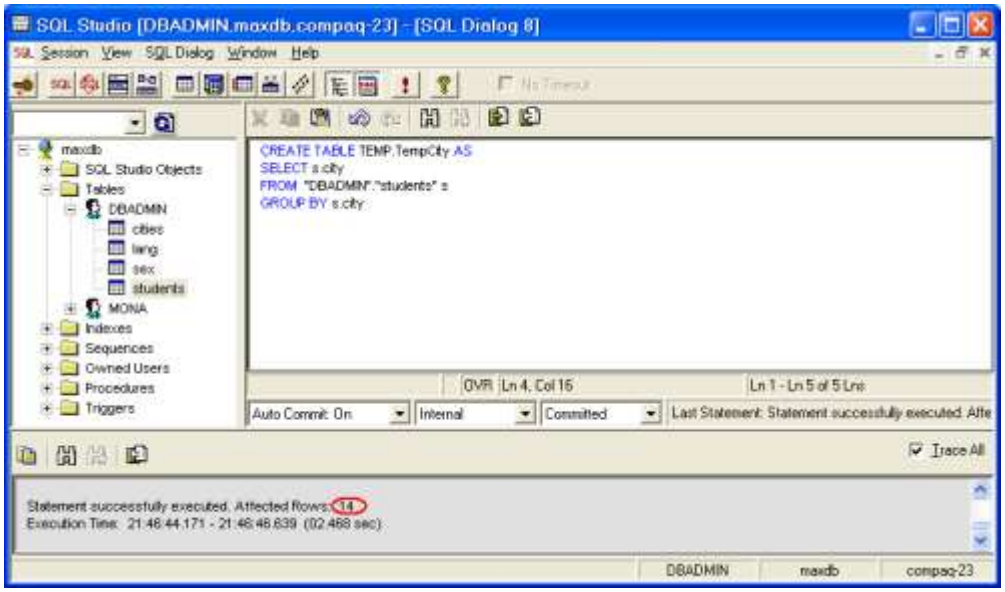
أي تم إضافة سجلين هما (غير سورية عربية، غير سورية أجنبية). الآن لو قمنا بكتابة الاستعلام التالي الذي يربط جدول الطلاب مع جدول المحافظات ربطاً يسارياً ويمينياً، والذي يقوم بإجراء الاتحاد بين السجلات الموجودة في جدول (students) والتي تتطابق مع سجلات الجدول (cities) وبين السجلات الموجودة في الجدول (cities) والتي تتطابق مع سجلات الجدول (students) كما الشّكل (12).



الشّكل(12): السجلات الفارغة المسترجعة بالطريقة الأولى دون الجداول المؤقتة.

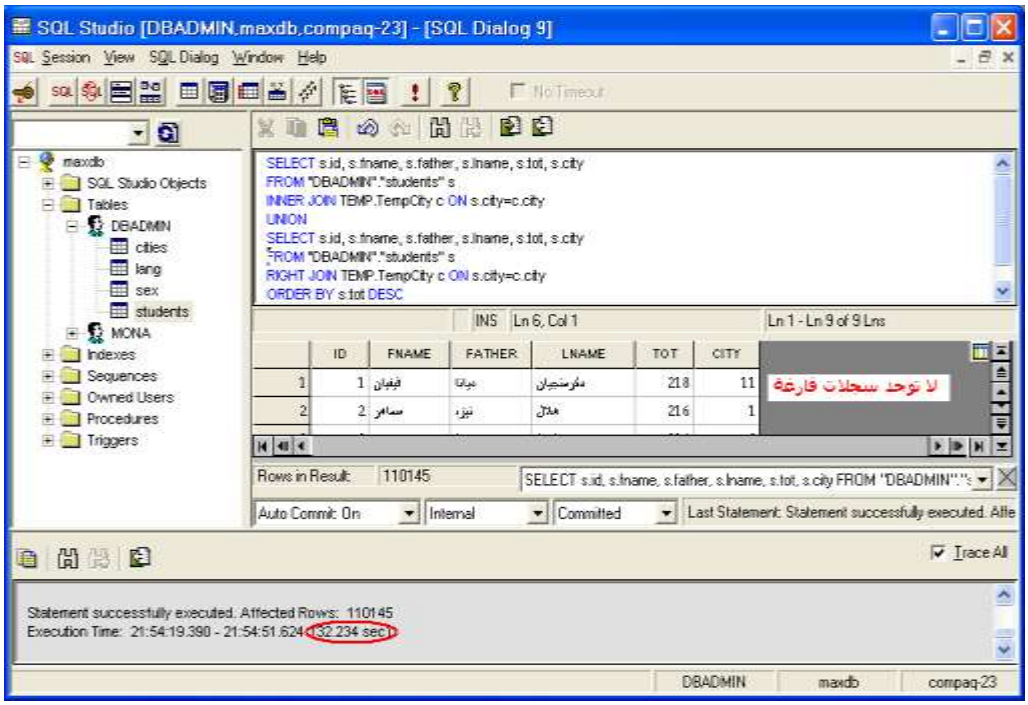
نلاحظ من الشّكل(12) أن السجلين الأول والثاني هما سجلان فارغان لأنه لا يوجد طلاب يحملون شهادات غير سورية وهما ناتجان عن الربط اليميني بين الجدول (cities) و الجدول (students) بالتالي أصبح عدد السجلات 110147 سجلاً، كما أنّ الوقت المستغرق لتنفيذ هذا الاستعلام هو 1:33:437 دقيقة.

لنقم الآن بالعمل نفسه لكن باستخدام الجداول المؤقتة. ويتم ذلك بإنشاء جدول مؤقت من الجدول (students) ويحوي الجدول المؤقت بيانات المحافظات التي ينتمي إليها الطلاب مع إسقاط المحافظات التي لا ينتمي إليها أحد كما في الشكل(13).



الشّكل(13): إنشاء جدول مؤقت من الجدول students للتخلص من ظهور السجلات الفارغة.

نلاحظ من الشّكل(13) أنّه تم إنشاء الجدول المؤقت (TempCity) من الجدول (students) والذي يحوي 14 سجلاً لأنه لا يوجد سوى 14 محافظة ينتمي إليها الطلاب في جدول (students). الآن لو قمنا بربط الجدول المؤقت مع جدول الطلاب كما في الشكل(14).



الشّكل(14): استخدام الجداول المؤقتة يُجنبنا استرجاع سجلات فارغة ويُسرع التنفيذ.

نلاحظ من الشّكل(14) أنّه تم التخلص من السجلات الفارغة حيث عاد عدد السجلات إلى 110145 سجلاً، واحتاج الاستعلام إلى وقتٍ يقدّر 32:234 ثانية، أي أن الفرق بين الطريقتين هو 1:01:203 ناتج عن طرح (1:33:437 – 32:234).

كما أن كل الجداول المؤقتة تحذف بانتهاء الجلسة، فلو قطعت الاتصال وقمت بالاتصال مرةً أخرى وأجريت أي استعلام عن أي جدول مؤقت قمت بإنشائه سابقاً فلن تحصل على نتيجة، لأنه حذف مع انتهاء الجلسة.

8.3 الاستنتاجات والمقترحات (Conclusion & Recommendations):

نستنتج مما سبق أنّ استخدام الجداول المؤقتة يوفر الوقت والجهد الكبيرين المبذولين في بناء الاستعلام ويُسرّع من استرجاع البيانات، وعلى المبرمجين ومديري قواعد البيانات عدم تفويت أي فرصة يستطيعون فيها استخدام الجداول المؤقتة فهي تعطي ديناميكية في الربط وسرعة في الأداء وأماناً في الحفاظ على البيانات.

في النهاية إذ تعرفنا في بحثنا هذا على طريقتين مطورتين استطعنا من خلالهما تسريع استرجاع البيانات، فإنه يبقى المجال مفتوحاً لإيجاد طرائق أخرى للاستفادة من الجداول المؤقتة، وتبقى المسألة مفتوحة في إيجاد طرائق تساعد على استرجاع البيانات بشكلٍ أسرع، فكما ذكرنا سابقاً أن حجم البيانات يزداد بشكل هائل سنوياً ويجب البحث دوماً عن طرائق ناجعة تؤدي إلى تسريع الوصول للبيانات.

المراجع:

1- SUBRGER, P. *Ensuring Successful MaxDB™ Database*. First ed, published by Sap, Berlin, 2005, 213.

2- GRAZIANO, B. *Temporary Tables*. First ed, published by Sqlteam, Washington, 2001, 147.

3- STEPHENS, J.; RUSSELL, C. *Database Design and Optimization* From Novice to Professional. First ed, published by Apress, Inc New York, 2004, 332.

4- TSURANOFF, D. *Temporary Tables And Their Effects*. First ed, published by Sqlserver, New York, 2007, 144.

5- HOFFMEISTER, J. *MaxDB back under the SAP roof!*. Company SAP AG, Posted on Oct. 05, 2007, < http://www.sdn.sap.com/irj/sdn/maxdb>.