قواعد البيانات

إن التقدم المتسارع في متطلبات الأنظمة المعلوماتية، والتضخم الكبير في حجم البيانات التي تعالجها تلك الأنظمة، أدى الى ظهور أنواع جديدة من قواعد البيانات سميت لاحقا ب NoSQLوهي اختصار ل Not Only SQL، ويشار اليها أيضاً ب قواعد البيانات غير العلائقية Non-Relational Databases، حيث يتم تخزين البيانات بطرق مختلفة عن الطريقة التقليدية التي يتم تخزين البيانات فيها على شكل جداول وأعمدة، والمستخدمة في أنظمة قواعد البيانات العلائقية RDBMS والتي تشكل لغة الاستعلام SQL نواة أساسية فيها.

يعتبر نموذج NoSQL أيضاً كاستجابة لتقنية web 2.0، حيث ظهرت الحاجة لمعالجة البيانات غير المنظمة Unstructured Data بطريقة مرنة وسهلة، وهو الامر الذي لا توفره أنظمة البيانات العلائقية التي يجب أن تكون فيها بنية الجداول محددة ومعرفة مسبقا بشكل دائم.

إن مصطلح NoSQL لا يشير الى نموذج واحد لتخزين البيانات، حيث أن كل نوع من قواعد البيانات غير العلائقية يتبع نموذج تخزين معين، سوف ندرس في هذا المحور نموذج يسمى بالنموذج المعتمد على المستندات Document-Based Model، حيث يتم تخزين البيانات على شكل مستندات Documents ضمن مجموعات Collections وكل مستند يتألف من ثنائية حقل – قيمة Key – Value Pair، ويعتبر هذا النموذج مشابه نوعا ما لنموذج الجداول والأعمدة المتبعة في قواعد البيانات العلائقية، إلا أن الاختلاف يبرز في امكانية تخزين مستندات ذات بنية مختلفة ضمن نفس المجموعة، وأيضا في امكانية استخدام ما يعرف بالمستندات المتداخلة Nested Documents حيث يمكن لمستند ما أن يحوي على مستند جزئي بداخله أو على لائحة من المستندات الجزئية الداخلية والتي من غير الضروري أن يكون لها نفس البنية.

سوف نقوم في هذا المحور ب اجراء مقارنة عملية ما بين MongoDB وSQL Server، حيث أن MongoDB هي قاعدة بيانات غير العلائقية وتتبع نموذج Document-Based Model المذكور سابقاً، ظهرت MongoDB بداية في عام 2007 وتطورت لاحقاً لتصبح من أشهر الأنظمة المستخدمة قي قواعد البيانات غير العلائقية، أما SQL Server فهو نظام يتيح إدارة قواعد البيانات العلائقية RDBMS وقد طُوِّر من قبل شركة مايكروسوفت.

الفصل الأول – مقدمة

المناقلة Transaction:

في سياق قواعد البيانات وأنظمة تخزين البيانات، فإن المناقلة هي عملية أو أكثر على البيانات يتم التعامل معها كوحدة عمل واحدة، والتي إما تكتمل بالكامل أو لا تكتمل على الإطلاق، وتترك نظام التخزين في حالة متسقة Consistent. المثال الكلاسيكي للمناقلة هو عملية نقل مبلغ مالي بين حسابين مصرفيين، فالمناقلة هنا تحتاج لعمليتين، الأولى هي إنقاص المبلغ من الحساب المرسل، ثم اضافته الى الحساب المستقبل، فإذا حصل خطأ في أي من العمليتين السابقتين تعتبر المناقلة غير ناجحة ولا يتم بتم حفظ أية تعديلات.

خصائص ACID:

ACID هو اختصار يشير إلى مجموعة من 4 خصائص رئيسية تحدد المناقلة: الذرية Atomicity و الاتساق Consistency و العزلة Isolation و المتانة Durability. إذا كانت عملية قاعدة البيانات تحتوي على خصائص ACID هذه، فيمكن تسميتها مناقلة ACID، وتسمى أنظمة تخزين البيانات التي تطبق هذه العمليات أنظمة مناقلات Transactional Systems.

تضمن مناقلات ACID أن كل قراءة أو كتابة أو تعديل للجدول لها الخصائص التالية:

* الذرية Atomicity: يتم التعامل مع كل عبارة Statement في المناقلة (لقراءة البيانات أو كتابتها أو تحديثها أو حذفها) كوحدة واحدة. إما أن يتم تنفيذ العبارة بأكملها، أو لم يتم تنفيذ أي منها. تمنع هذه الخاصية حدوث فقدان البيانات وتلفها إذا حصل خطأ ما خلال تنفيذ العبارة Statement Execution، مثلاً انقطاع التيار الكهربائي خلال تنفيذ تعليمة تقوم بتعديل 1000 سجل.
* الاتساق Consistency: يشير الاتساق إلى الحفاظ على قيود سلامة البيانات، حيث لن تنتهك المناقلة المتسقة قيود التكامل الموضوعة على البيانات، يضمن فرض الاتساق أنه في حالة دخول قاعدة البيانات إلى حالة غير قانونية (في حالة حدوث انتهاك لقيود تكامل البيانات)، فإنه سيتم إحباط العملية وإرجاع التغييرات إلى حالتها القانونية السابقة، مثلاً ادخال قيمة سالبة الي حقل الرصيد في حساب بنكي.
* العزلة Isolation: وتعني أنه عندما يقوم العديد من المستخدمين بالقراءة والكتابة من نفس الجدول دفعة واحدة، فإن المناقلات المتزامنة لا تتداخل مع بعضها البعض أو تؤثر على بعضها البعض. يمكن أن يحدث كل طلب كما لو أن كل الطلبات كانت تحدث طلباً تلو الآخر، على الرغم من أنها تحدث في وقت واحد فعلياً.
* المتانة Durability: تضمن هذه الخاصية حفظ التغييرات التي تم إجراؤها على البيانات من خلال المناقلات المنفذة بنجاح، حتى في حالة فشل النظام.

أهمية مناقلات ACID:

تضمن مناقلات ACID أعلى وثوقيه Reliability وتكامل Integrity ممكنين للبيانات، حيث إنها تضمن عدم وقوع البيانات في حالة عدم تناسق بسبب عملية اكتملت جزئيًا فقط. على سبيل المثال، بدون مناقلات ACID، إذا كنا نكتب بعض البيانات إلى جدول في قاعدة البيانات، لكن الطاقة انقطعت بشكل غير متوقع، فمن المحتمل أنه لم يتم حفظ سوى بعض البيانات ولم يتم حفظ البعض الآخر، في هذه الحالة أصبحت قاعدة البيانات الآن في حالة غير متسقة ومن الصعب للغاية استرجاعها وسيستغرق وقتًا طويلاً.

مبدأ CAP:

في عام 1999 قدم العالم Brewer, Fox ما سماه ب مبدأ CAP (CAP Principle) والذي عرف لاحقاً باسم نظرية CAP (CAP Theorem).

بشكل رئيسي فإن أي نظام قاعدة بيانات موزع يهتم باكتساب ثلاث صفات رئيسية:

1. الاتساق Consistency: وتعني أن أي عملية قراءة سوف تحصل على المعلومات المحدثة في آخر عملية كتابة
2. التوافر Availability: وتعني أن النظام متاح بشكل دائم ويمكن اجراء عمليات قراءة منه في أي وقت.
3. سماحية التقسيم Partition Tolerance: القدرة على إضافة عقد بسهولة الى النظام بحيث يتم توزيع البيانات ضمن هذه العقد لتحقيق عملية قياس أفقية Horizontal Scaling.

يشير مبدأ CAP إلى أنه لا يمكن لنظام قاعدة بيانات موزع، تحقيق هذه الصفات الثلاثة بشكل كامل ولكن يمكن تحقيق اثنتين من هذه الصفات الثلاث بشكل صحيح.

تم اثبات هذه النظرية لاحقاً في عام 2002 عن طريق ورقة بحثية للباحثين Nancy Lynch and Seth Gilbert من معهد MIT.

ابتكر Brewer قائمة بالخصائص تمت تسميتها لاحقاً بـ "تكوينات CAP" "CAP Configuration"، حيث يمكن استخدامها لتحديد نقاط القوة المعمارية لأنظمة قواعد البيانات المختلفة:

1. **CA:** "لا يمكن لقواعد البيانات تحقيق الاتساق والتوافر معاً الا في حالة غياب التقسيم عبر شبكة"

تعتبر أنظمة قواعد البيانات العلائقية قواعد بيانات "CA”، حيث تتمتع بالقدرة على توفير مناقلات ACID قوية.

1. **CP**: "يمكن تحقيق الاتساق والتقسيم عن الطريق التضحية بالتوافر".

فمثلاً عند تحديث بيانات ضمن عقدة ما، يتم تحديث البيانات على جميع العقد التي تمثل تكرار Replica Set للعقدة المحدثة بشكل عاجل، وتكون العقد غير متوافرة حتى انتهاء عملية التحديث.

1. **AP**: "يمكن تحقيق التوافر والتقسيم عن طريق التضحية بالاتساق".

فمثلاً عند فمثلاً عند تحديث بيانات ضمن عقدة ما، لا يتم تحديث البيانات على جميع العقد التي تمثل تكرار Replica Set للعقدة المحدثة بشكل عاجل، وبالتالي تكون العقد متوافرة ولكن من الممكن الحصول على نتائج غير متسقة من احدى العقد الى ان يتم تحديثها.

**بشكل أساسي، تختلف قواعد بيانات NoSQL من الناحية المعمارية عن قواعد البيانات العلائقية لأنها مصممة لجني مزايا أداء القراءة والكتابة عن طريق سماحية التقسيم (القياس أفقيًا Horizontal Scaling) مع ترك الاتساق أو التوافر للتفاوض بناء على متطلبات المشروع.**

**معرفة هذا الاختلاف أمر بالغ الأهمية لفهم المواقف التي قد تكون فيها قاعدة بيانات NoSQL أكثر ملاءمة من RDBMS.**