**УДК 004.652**

*Черницька І.О., асистент*

*Полтавський національний технічний університет*

*імені Юрія Кондратюка*

**Аналіз паралельних систем баз даних**

*В даній роботі було розглянуто та проведено аналіз форм паралелізму, архітектур паралельних баз даних, можливостей транзакційних систем з урахуванням постановки задач для паралельних систем управління базами даних, які повинні виконувати складні запити над великою кількістю даних.*

Ключові слова: *СУБД, система управління базою даних, паралелізм, архітектура паралельних баз даних.*

Постійно прогресуюче інформаційне навантаження, що пов'язане з інтелектуальним аналізом даних, побудовою гістограм, розвитком геоінформаційних систем, соціальних мереж, електронних бібліотек та ін. призводить до необхідності вдосконалення методів обробки надвеликих об’ємів інформації.

Основним ефективним засобом для зберігання та обробки інформації вважається використання систем управління базами даних. Надвеликі дані формують надвеликі бази даних і потребують паралельної обробки транзакцій на багатопроцесорних і багатоядерних обчислювальних системах.

Відомо, що така СУБД, по-перше, повинна являти собою систему високої готовності до забезпечення оперативної обробки запиту користувача, виконуючи при цьому паралельну обробку великої кількості транзакцій. По-друге, мати можливість до динамічного нарощування з метою адаптації до збільшення розмірів бази даних та зростаючих вимог продуктивності. По-третє, забезпечити високу доступність даних в багатопроцесорних системах. Отже, основними вимогами до такої системи управління баз даних є:

* Висока масштабованість.
* Висока продуктивність.
* Висока доступність даних.

Характерним рішенням для задоволення цих вимог є впровадження паралельних СУБД. В основі базової концепції паралельної системи баз даних є паралелізм, тому з метою забезпечення ефективного функціонування різних модифікацій СУБД використовується широкий спектр його форм.

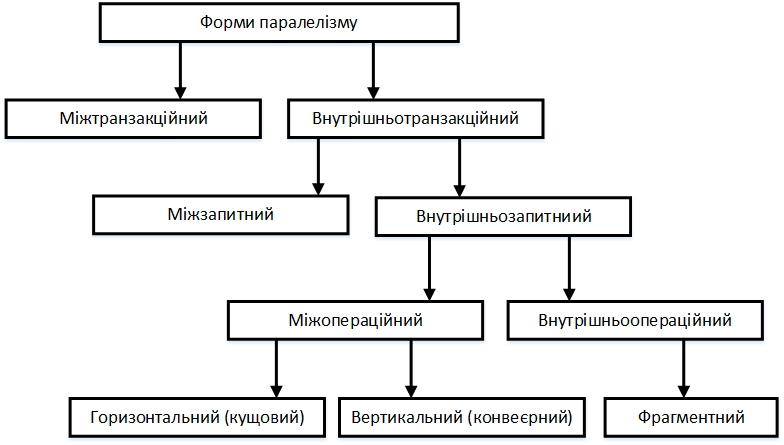


Рис.1. Види паралельної обробки транзакцій.

**Міжтранзакційний паралелізм** дозволяє підвищити ефективність виконання безлічі незалежних транзакцій над однією і тією ж базою даних. Є присутнім в однопроцесорних системах і заснований на перекритті затримок введення-виведення.

**Внутрішньотранзакційний паралелізм** дає можливість паралельно виконувати окремі транзакції, може бути реалізований у формах міжзапитного та внутрішньозапитного паралелізму.

**Міжзапитний паралелізм** передбачає паралельне виконання окремих SQL операторів, які належать одній і тій же транзакції. Кількість SQL операторів та попередні обмеження між окремими операторами здатні впливати на степінь міжзапитного паралелізму. Цей вид паралелізму не використовується в СУБД.

**Внутрішньозапитний паралелізм** дозводяє паралельне виконання окремого SQL оператора (запиту). Є характерним для реляційних систем баз даних завдяки можливості ефективного розпаралелюванню реляційних операцій над наборами кортежів. Даний вид паралелізму може реалізовуватися у вигляді міжопераційного та внутрішньоопераціонного паралелізму.

**Міжопераційний паралелізм** є пралельним виконанням реляційних операцій в межах одного і тому ж плану запиту. Може реалізовуватися як горизонтальній паралелізм так і вертикальний.

**Горизонтальний (кущовий) паралелізм** передбачає паралельне виконання по представленому плану запита незалежних піддерев дерева. Однак, існує проблема, що полягає в складності забезпечення можливості генерування вихідних даних в правильний час і в правильному темпі двома підпланами одного плану. Степінь кущового паралелізму буде обмежуватися кількістю піддерев у плані запита і зазвичай не перевищує 10.

**Вертикальний (конвеєрний) паралелізм** організовується на основі механізму конвеєризації при паралельному виконанні різних операцій плану запита, що будується на передачі даних від постачальника до споживача по сформованому потоку даних у вигляді конвеєру між суміжними операціями в дереві запиту. Однак, степінь вертикального паралелізму обмежена кількістю операцій, що надходять до конвеєру. Для реляційних систем баз даних довжина конвеєра рідко перевищує 10 операцій.

**Внутрішньоопераційний паралелізм** зазвичай реалізується у формі фрагментного паралелізму, що полягає у попередньому поділенні відношень на фрагменти, які будуть оброблятися на різних вузлах багатопроцесорної системи.

**Фрагментний паралелізм** передбачає фрагментацію відношень, тобто розбиття на непересічні частини, що є аргументом реляційної операції. При цьому одиночна реляційна операція виконується у вигляді декількох паралельних процесів, кожен з яких обробляє окремий фрагмент відношення. Отримувані результуючі фрагменти зливаються в загальне результуюче відношення.

Паралельна фрагментація підрозділяється на вертикальну і горизонтальну. Вертикальна фрагментація виконує розбиття відношень на фрагменти по стовпцях (атрибутам). Горизонтальна фрагментація розбиває відношення на фрагменти по рядках (кортежам). Паралельні СУБД, що підтримують фрагментний паралелізм, використовують головним чином горизонтальну фрагментацію. Теоретично фрагментний паралелізм може забезпечити достатньо високу степінь розпаралелювання реляційних операцій.

В даний час широкого використання набули кластерні СУБД з ПЗ проміжного рівня. В залежності від того, які функції виконує проміжне ПЗ кластерні системи баз даних є орієнтовані на додатки класу OLTP або на додатки класу OLAP.

Згідно сценарію OLTP (Online Transaction Processing, оперативна обробка транзакцій) СУБД обробляє велику кількість коротких транзакцій. Таким чином проміжне ПЗ забезпечує міжтранзакційний паралелізм. Це дозволяє підвищити доступність системи при великій кількості клієнтів за рахунок їх розподілення для оброки на декількох екземплярах СУБД. Але, на жаль, ця транзакційна система, не надає можливості обробляти надвеликі дані.

У випадку сценарію OLAP (Online Analytical Proicessing, оперативна аналітична обробка) передбачається, що СУБД буде виконувати складні запити над великою кількістю даних. У такому разі проміжне ПЗ, використовуючи внутрішньозапитний паралелізм, здійснює прийом запитів користувача, проводить їх перетворення і розподіляє на вузли кластера, що поєднують в собі економічність і високу обчислювальну потужність. А зрештою виконує злиття отриманих часткових результатів та передає їх користувачеві. Недоліком даних систем можна вважати використання повної реплікації всіх таблиць бази даних на вузлах кластерної системи.

Для підвищення функціональності програмного забезпечення СУБД однією з головних задач є вдосконалення апаратної частини.

Керуючись поширеною системою класифікації паралельних систем баз даних, запропонованою Майклом Стоунбрейкером, можна розділити всі паралельні системи на три базових класи:

**SE (Shared-Everything)** - архітектура з розділеною пам'яттю і дисками. Представляє собою системи баз даних, в яких всі диски доступні всім процесорам з однаковим часом доступу і все процесори поділяють спільну оперативну пам'ять. Слід зазначити, що кожен процесор в системі SE має власну кеш - пам'ять.

**SD (Shared-Disks)** - архітектура з розділеними дисками. Будь-який процесор має доступ до будь-якого диску, проте кожен процесор має свою приватну оперативну пам'ять. Передача даних в таких системах здійснюється процесорами за допомогою високошвидкісної мережі.

**SN (Shared-Nothing)** - архітектура без спільного використання ресурсів. Характеризується наявністю у кожного процесора власної оперативної пам'яті і власного диска. Процесорні вузли з'єднані високошвидкісною мережею, здійснюють обмін повідомленнями між процесорами.

В зв’язку з тим, що системи, як правило, мають складну структуру апаратних компонентів і поєднує в собі властивості різних класів архітектур, класифікація Стоунбрейкера стала застарілою. З появою багатопроцесорних систем, що поєднують в собі риси як SE так і SN класифікацію Стоунбрейкера було розширено шляхом введення двох додаткових класів архітектур паралельних машин баз даних.

**CE (Clustered-Everything)** - архітектура з SE кластерами, об'єднаними за принципом SN.

**CD (Clustered-Disk)** - архітектура з SD кластерами, об'єднаними за принципом SN.

Також можна розглядати цілу ієрархію віртуальних машин, де кожна наступна машина є платформою для реалізації попередньої. Таким прикладом є **CDN**, що представляє собою гібридну архітектуру, яка має набір однотипних SD кластерів, об'єднаних за принципом SN. Особливістю даної системної архітектури є те, що на верхніх рівнях системної ієрархії SD кластери розглядаються як SN системи. Тобто кожному процесорному вузлові логічно призначається окремий диск. Якщо брати до уваги вимоги до паралельної системі баз даних високої готовності, то найкращим вибором буде CDN архітектура.

Отже, вдосконалення програмного забезпечення та постійний розвиток технічної частини сприяє прогресуючим змінам в області інформаційних технологій. А використання паралелізму суттєво підвищує функціональні можливості СУБД. Таким чином паралельні СУБД є найбільш дієвим підходом для задоволення потреб багатьох важливих прикладних областей, яким необхідна виключно висока можливість обробки надвеликих об’ємів інформації.

*Використані джерела*

1. Соколинский Л.Б. Параллельные системы баз данных. Издательство Московского университета, 2013. 184 с.
2. Пан К., Соколинский Л., Цымблер М. Интеграция параллелизма в СУБД с открытым кодом // Открытые системы. СУБД. 2013. № 9. С. 56-58.
3. Костенецкий П.С., Соколинский Л.Б. Моделирование иерархических многопроцессорных систем баз данных // Программирование, 2013. Т. 39, №. 1. С 3-22.
4. Базы данных и СУБД. Программирование на PL/SQL Электронное учебное-методическое пособие. Составители: доцент, к.ф.-м.н. А.М. Гудов, к.т.н. С.Ю. Завозкин, к.ф.-м.н. Т.С. Рейн Кемерово, 2009
5. Новиков Б.А., Домбровская Г.Р. Настройка приложений баз данных. – СПб.: БХВ-Петербург, 2006. – 240 с.: ил.