

DEADLOCKS

Detection, Troubleshooting, and Prevention

- ▶ Trevor Barkhouse
trevor.barkhouse@microsoft.com



2012 MAY, 10-11
DALLAS, TX

Deadlocks are like ants...





Contact Information



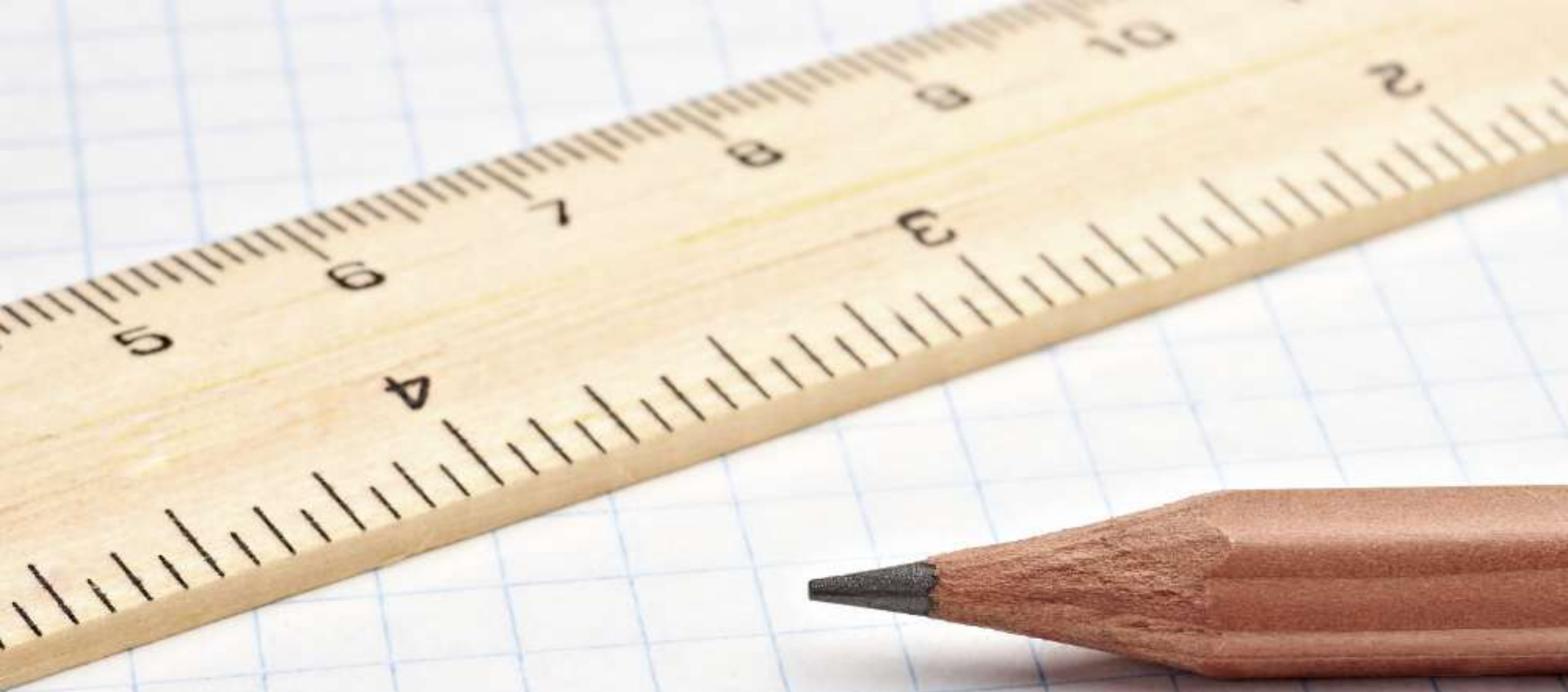
trevor.barkhouse@microsoft.com

Agenda

- ▶ Deadlocking basics
- ▶ Detecting deadlocks
- ▶ Troubleshooting deadlocks
- ▶ Preventing deadlocks



Deadlocking Basics



What is a deadlock?

Demonstration

Deliberately causing a deadlock



Deadlock Detection

Error message 1205

- ▶ Msg 1205, Level 13, State 51, Line 1
- ▶ Transaction (Process ID 52) was deadlocked on lock resources with another process and has been chosen as the deadlock victim. Rerun the transaction.

Error message suppression

```
private void associateSchedule
(
    int scheduleID,
    int userID
)
{
    using
    (
        SqlConnection connection = new SqlConnection(Application["ConnectionString"])
    )
    {
        using
        (
            SqlCommand command = new SqlCommand
            (
                "[Scheduler].[AssociateSchedule]",
                connection
            )
        )
        {
            try
            {
                command.Parameters.Add("@ScheduleID", SqlDbType.Int).Value = scheduleID;
                command.Parameters.Add("@UserID", SqlDbType.Int).Value = userID;
                connection.Open();
                command.ExecuteNonQuery();
            }
            catch (SqlException exception)
            {
                redirectToFriendlyErrorPage(exception);
            }
        }
    }

    return;
}
```


**If a deadlock occurs
on your server,
and nobody is around
to see the error message...**



A nighttime photograph of a city skyline, likely Chicago, with several tall skyscrapers illuminated. In the foreground, a road shows light trails from moving vehicles, and a street sign for 'Beckley Ave' is visible. The word 'Demonstration' is overlaid in white text on the left side of the image.

Demonstration

Configuring 1205 errors to be logged



Deadlock Troubleshooting

Read Bart Duncan's blog posts!

1. ["Deadlock Troubleshooting, Part 1"](#)
 - ▶ ["Interpreting Trace Flag 1204 Output"](#)
2. ["Deadlock Troubleshooting, Part 2"](#)
3. ["Deadlock Troubleshooting, Part 3"](#)

Troubleshooting overview

1. Capture diagnostic data
2. Identify:
 - ▶ the ***parties*** to the deadlock and
 - ▶ the ***resources*** under contention
3. Take corrective action

A nighttime photograph of a city skyline, likely Chicago, with several tall skyscrapers illuminated. In the foreground, a road shows light trails from moving vehicles, and a street sign for 'Beckley Ave' is visible. The word 'Demonstration' is overlaid in white text on the left side of the image.

Demonstration

Diagnostics:
SQL Server 2000 (or earlier)

Demonstration

Diagnostics:
Trace flag 1222

Demonstration

Diagnostics:
Deadlock Graph event class

A nighttime photograph of a city skyline, likely Chicago, with several tall skyscrapers illuminated. In the foreground, there are light trails from cars on a road, and a street sign for Beckley Ave is visible. The word "Demonstration" is overlaid in white text on the left side of the image.

Demonstration

Diagnostics:
Deadlock Graphs and XQuery

A warning about XQuery



A nighttime photograph of a city skyline, likely Chicago, with several tall skyscrapers illuminated. In the foreground, a road shows light trails from moving vehicles, and a street sign for Beckley Ave is visible. The word "Demonstration" is overlaid in white text on the left side of the image.

Demonstration

Diagnostics:
Extended Events

Cheat at deadlock analysis!

1. Use the Database Engine Tuning Advisor
2. Ask for help

Demonstration

Setting the deadlock priority

Fault-retry logic

The following example is taken directly from *Inside Microsoft SQL Server 2005: Query Tuning and Optimization* by Kalen Delaney (found on pages 365 and 366).

```
DECLARE @Tries tinyint, @Error int;
SET @Tries = 1;
WHILE @Tries <= 3
BEGIN
    BEGIN TRANSACTION;
    BEGIN TRY
        -- <code goes here>
        IF XACT_STATE() = 1 COMMIT;
        BREAK;
    END TRY
    BEGIN CATCH
        SET @Error = ERROR_NUMBER();
        IF @Error = 1205
        BEGIN
            IF XACT_STATE() = -1 ROLLBACK;
        END
        SET @Tries = @Tries + 1;
        CONTINUE;
    END CATCH;
END;
```



Deadlock Prevention



- Access tables consistently



- Keep transactions short



- Add useful indexes



- Avoid user interaction within transactions



- Reschedule batch processes



- Adjust locking strategy



- Use bound connections

- Access tables consistently

Procedure 1

[Sales].[OrderItems]

[Sales].[OrderHeaders]

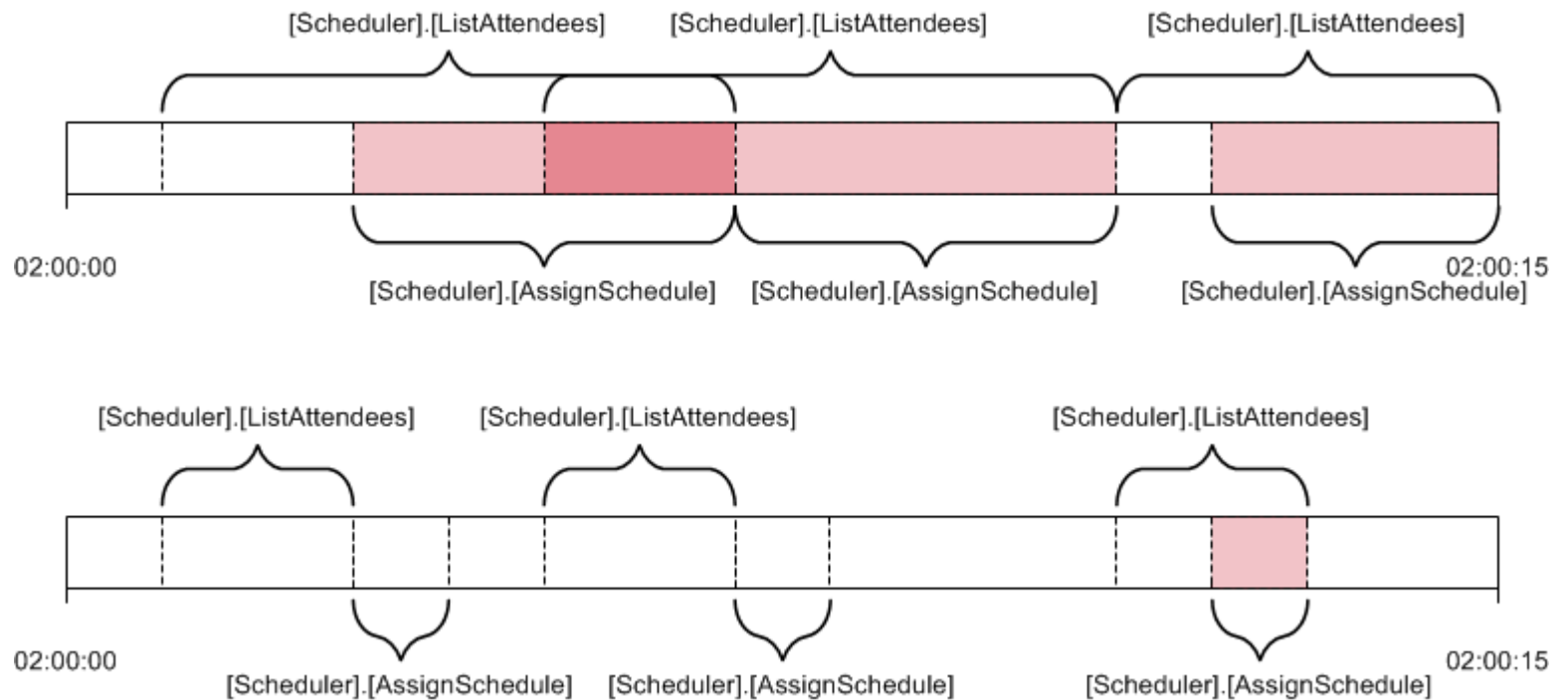
Procedure 2

[Sales].[OrderHeaders]

[Sales].[OrderItems]

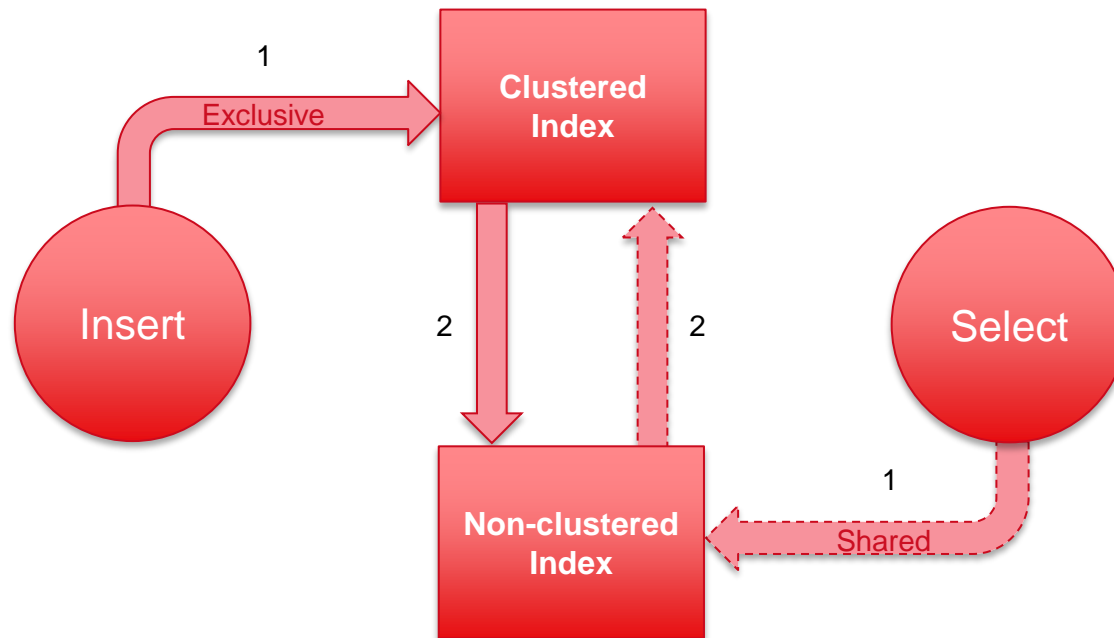


- Keep transactions short



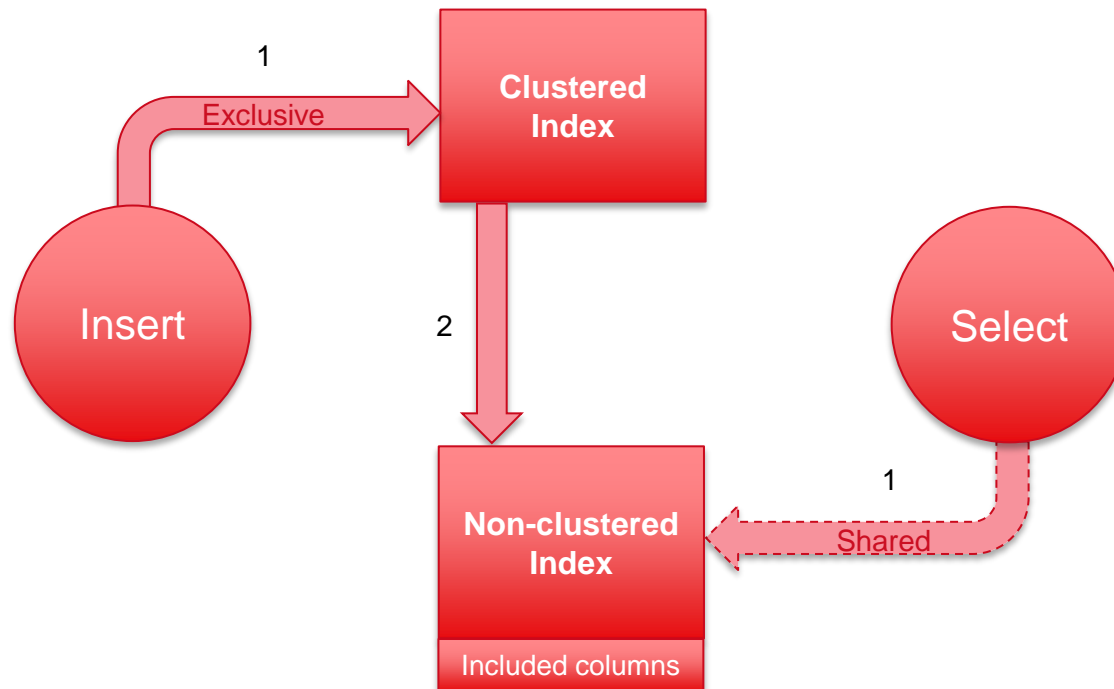
Before...

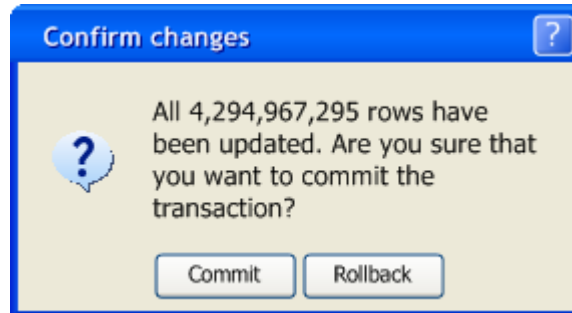
- Add useful indexes



After...

- Add useful indexes






- Avoid user interaction within transactions






- Reschedule batch processes

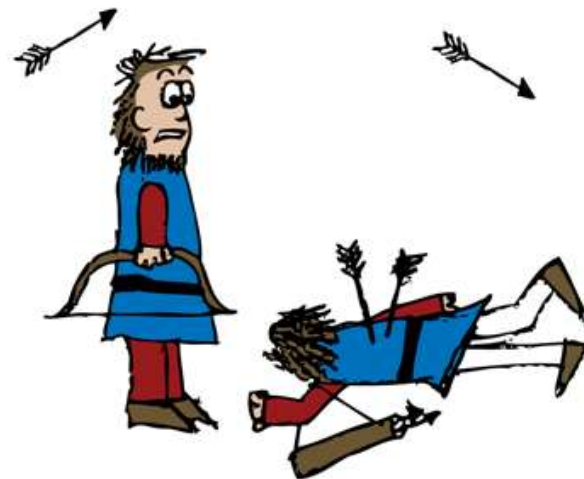
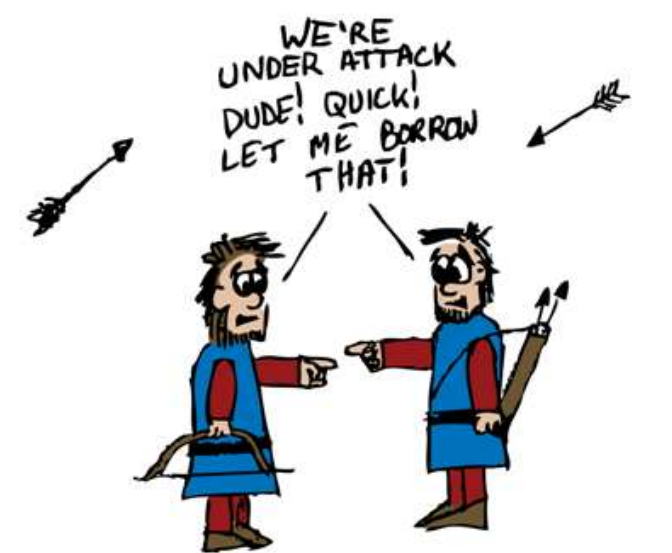
- 
- Consider restrictive locking up-front
 - Evaluate row-versioning isolation levels
 - **Carefully** explore reducing isolation levels or using locking hints

- Adjust locking strategy



If using Multiple Active Result Sets (MARS),
consider using “bound connections” to allow
resource sharing

- Use bound connections



Deadlock Victim

Courtesy of Michael Swart (michaeljswart.com)

Prescriptive Guidance – Proactive

1. Modify error message 1205 to be logged
2. Add fault-retry logic to applications
3. Establish protocols for table access
4. Use explicit transactions, and never include user-interaction within them
5. Stagger batch/scheduled processes

Prescriptive Guidance – Reactive

1. Capture deadlock diagnostic data:
 - ▶ Trace Flag 1204 (SQL Server 2000)
 - ▶ Deadlock Graph event class (SQL Server 2005+)
2. Identify deadlocked processes and resources
3. Set deadlock priorities
4. Run the Database Engine Tuning Advisor
5. Consult with other DBAs

Contact Information



trevor.barkhouse@microsoft.com