```
--Create a database to use for this example
IF (SELECT DB_ID('HoggysBlog')) IS NULL
CREATE DATABASE HoggysBlog
GO

USE HoggysBlog
GO

--Create a table of consecutive numbers
IF (SELECT OBJECT_ID('Number')) IS NULL
       BEGIN
              CREATE TABLE Number
              (
              n INT
              )

              INSERT INTO Number (n)
              SELECT ROW_NUMBER() OVER (ORDER BY CURRENT_TIMESTAMP) rn
              FROM sys.trace_event_bindings r1, sys.trace_event_bindings
       END

--Create the table used for this example
IF (SELECT OBJECT_ID('PartitionPuzzle')) IS NULL
       BEGIN
              CREATE TABLE PartitionPuzzle
              (
              Puzzle_ID INT IDENTITY (1,1),
              Puzzle_Year INT NOT NULL,

              CONSTRAINT [PK_Puzzle] PRIMARY KEY NONCLUSTERED
              (
                     Puzzle_Id ASC
              )
              )

              CREATE CLUSTERED INDEX CIX_Puzzle ON [PartitionPuzzle]
              (
              [Puzzle_Year] ASC
              )
       END


--Insert some rows into the table
INSERT PartitionPuzzle (Puzzle_Year)
SELECT 1666
FROM Number
WHERE n <= 100

INSERT PartitionPuzzle (Puzzle_Year)
SELECT 1812
FROM Number
WHERE n <= 100


--In first query session run this
BEGIN TRANSACTION
DELETE FROM PartitionPuzzle
WHERE Puzzle_Year = 1666
```

```
--In second query session run this
BEGIN TRANSACTION
DELETE FROM PartitionPuzzle
WHERE Puzzle_Year = 1812

--Both operations succeed because each takes key locks
sp_lock


--In first query session rollback the transaction
ROLLBACK TRANSACTION

--In Second query session rollback the transaction
ROLLBACK TRANSACTION


--Add some more rows to the table

INSERT PartitionPuzzle (Puzzle_Year)
SELECT 1666
FROM Number
WHERE n <= 14900

INSERT PartitionPuzzle (Puzzle_Year)
SELECT 1812
FROM Number
WHERE n <= 14900


--Again in first query session run this
BEGIN TRANSACTION
DELETE FROM PartitionPuzzle
WHERE Puzzle_Year = 1666


--Again in second query session run this
BEGIN TRANSACTION
DELETE FROM PartitionPuzzle
WHERE Puzzle_Year = 1812

--First session is blocking the second session...
sp_who2

--...because locks have escalated from key to table
sp_lock

--Lock escalation is the process of converting many fine-grain locks into fewer coarse-
grain locks, reducing system overhead while increasing the probability of concurrency
contention.
--Repeat previous deletes

--In first query session rollback the transaction
ROLLBACK TRANSACTION

--In Second query session rollback the transaction
ROLLBACK TRANSACTION
```

```
--Now partition the table so that partition level locks will be  taken instead of table
locks

--Create the partition function and partition scheme

CREATE PARTITION FUNCTION pf_PartitionPuzzle(INT) AS RANGE RIGHT FOR VALUES (1066,1812)
GO
CREATE PARTITION SCHEME ps_PartitionPuzzle AS PARTITION pf_PartitionPuzzle ALL TO
([Primary])
GO

--Alter lock escalation from the default of "table" to allow partition level locking
ALTER TABLE PartitionPuzzle SET (LOCK_ESCALATION = AUTO)
GO

--Drop the clustered index and rebuild it on the partition scheme

DROP INDEX CIX_Puzzle ON PartitionPuzzle
GO

CREATE CLUSTERED INDEX CIX_Puzzle ON PartitionPuzzle
(
       Puzzle_Year
)
ON ps_PartitionPuzzle(Puzzle_Year)
GO

--Table is now partitioned on Puzzle_Month column




--Again in first query session run this
BEGIN TRANSACTION
DELETE FROM PartitionPuzzle
WHERE Puzzle_Year = 1666


--Again in second query session run this
BEGIN TRANSACTION
DELETE FROM PartitionPuzzle
WHERE Puzzle_Year = 1812

--Still blocking! - The puzzle is why is a table lock being taken out on the clustered
index when partition level locking should be used?
sp_lock

--In first query session rollback the transaction
ROLLBACK TRANSACTION

--In Second query session rollback the transaction
ROLLBACK TRANSACTION


--Diagnostics
select * from sys.partitions where object_id=object_id('PartitionPuzzle')
--Shows index 1 (Clustered index) is partitioned, but index 2 (primary key) is not.
```

```
--Need to partition align primary key by adding the cluster key to it
ALTER TABLE PartitionPuzzle DROP CONSTRAINT PK_Puzzle
GO

ALTER TABLE PartitionPuzzle ADD  CONSTRAINT PK_Puzzle PRIMARY KEY NONCLUSTERED
(
       Puzzle_ID,
       Puzzle_Year -- *** Adding this column to the primary key aligns it to partitioning
***
)  ON ps_PartitionPuzzle(Puzzle_Year)
GO




--Now repeat previous test

--Again in first query session run this
BEGIN TRANSACTION
DELETE FROM PartitionPuzzle
WHERE Puzzle_Year = 1666


--Again in second query session run this
BEGIN TRANSACTION
DELETE FROM PartitionPuzzle
WHERE Puzzle_Year = 1812

sp_lock
sp_who2

--In first query session rollback the transaction
ROLLBACK TRANSACTION

--In Second query session rollback the transaction
ROLLBACK TRANSACTION
```