



PASS
SUMMIT 2016

Data Partitioning

For DB Architects and Mere Mortals

Dmitri Korotkevitch
<http://aboutsqlserver.com>





Please silence
cell phones

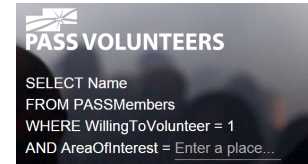
Explore Everything PASS Has to Offer



FREE ONLINE WEBINAR EVENTS



FREE 1-DAY LOCAL TRAINING EVENTS



VOLUNTEERING OPPORTUNITIES



LOCAL USER GROUPS
AROUND THE WORLD



ONLINE SPECIAL INTEREST
USER GROUPS



PASS COMMUNITY NEWSLETTER

PASS BLOG

WHITE PAPERS

SESSION RECORDINGS

FREE ONLINE RESOURCES



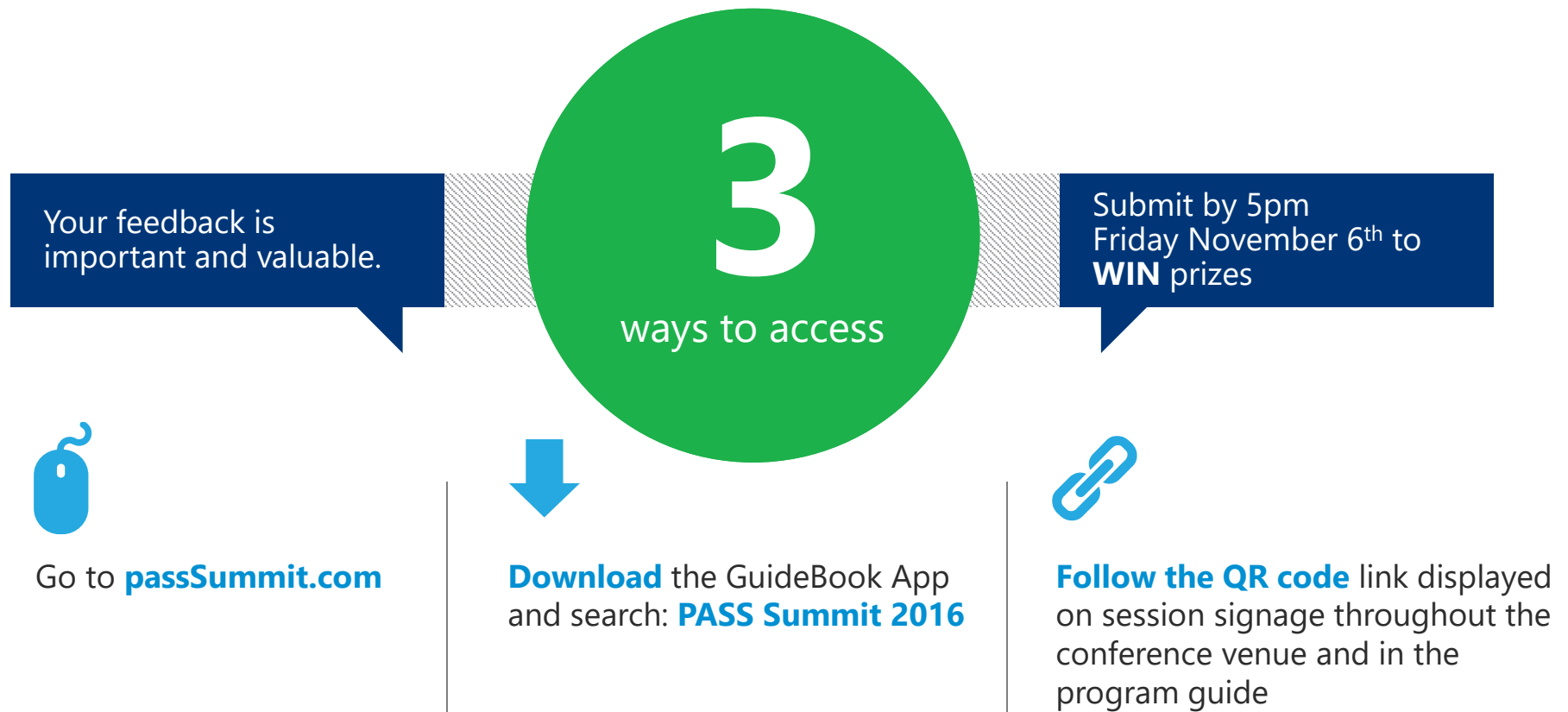
BUSINESS ANALYTICS TRAINING



BA INSIGHTS NEWSLETTER



Session Evaluations





Dmitri Korotkevitch

20+ Years in IT

15+ Years working with SQL Server

Microsoft Data Platform MVP

Microsoft Certified Master

Author:

- Pro SQL Server Internals
- Expert SQL Server In-Memory OLTP

<http://aboutsqlserver.com>

dk@aboutsqlserver.com

[🐦 @aboutsqlserver](https://twitter.com/aboutsqlserver)



Agenda

Architecture-focused discussion about data partitioning

- Partitioning in the single database
- “Distributed data” partitioning
- Going non-relational

Challenge: Inaccurate Statistics

(C) Gavin Bell <https://www.flickr.com/photos/gavinbell/535261899>



Challenge: Mixed Workload Support



Challenge: Disaster Recovery / RTO



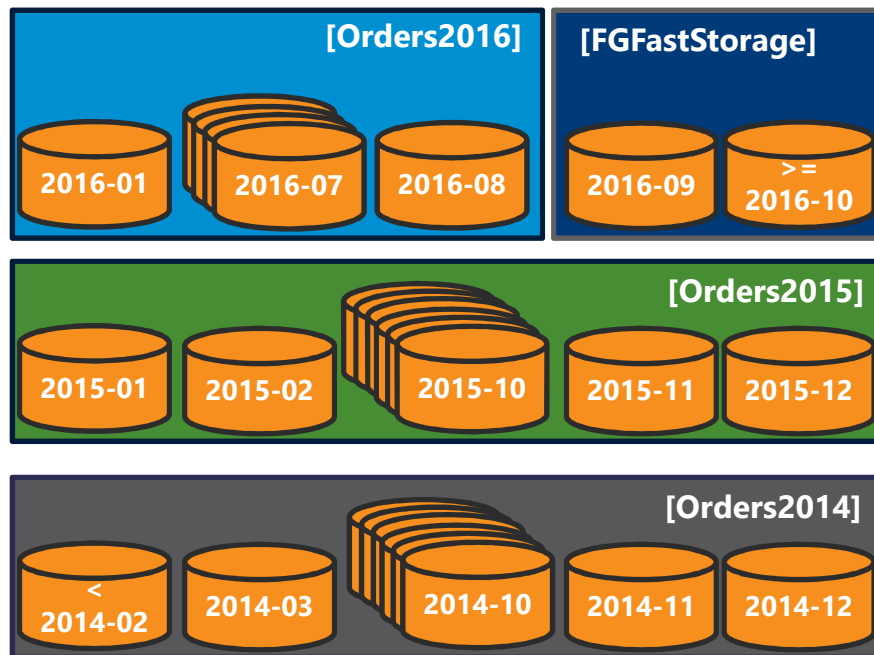
Challenge: Index Maintenance



Challenge: Storage Cost



Partitioned Tables



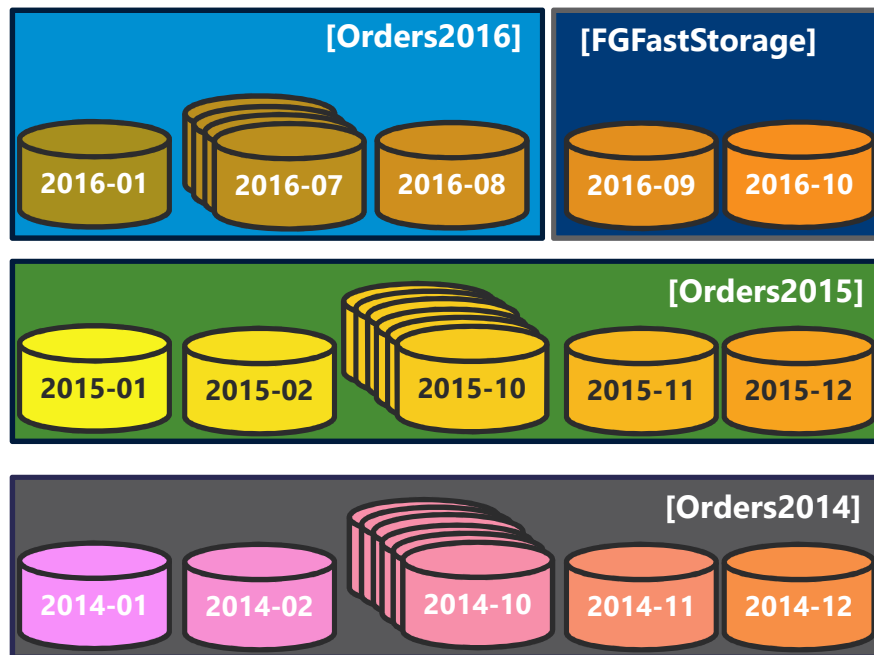
```
create partition function pfOrders(datetime2(0))
as range right
for values('2014-02-01','2014-03-01','2014-04-01',
/*...*/ '2015-01-01','2015-02-01','2015-03-01',
/*...*/ '2016-01-01','2016-02-01','2016-03-01',
/*...*/ '2016-08-01','2016-09-10','2016-10-01');

create partition scheme psOrders
as partition pfOrders
to ([Orders2014],[Orders2014],[Orders2014],
/*...*/ [Orders2015],[Orders2015],[Orders2015],
/*...*/ [Orders2016],[Orders2016],[Orders2016],
/*...*/ [Orders2016],[FGFastStorage],[FGFastStorage]);

create table dbo.OrdersPT
(
    OrderDate datetime2(0) not null,
    OrderId int not null,
    /* ... */
);

create unique clustered index IDX_Orders_OrderDate_OrderId
on dbo.Orders(OrderDate,OrderId)
with ( data_compression = page on partitions(1 to 24),
       data_compression = row on partitions(25 to 31),
       data_compression = none on partitions(32 to 33) )
on pfOrders(Orders,OrderDate);
```

Partitioned Views



```
create table dbo.Orders2014_01
(
    OrderDate datetime2(0) not null,
    OrderId int not null,
    /*..*/
    constraint CHK_Orders2014_01
    check (OrderDate >= '2014-01-01' and OrderDate < '2014-02-01')
) on [FG2014];
/*..*/
create table dbo.Orders2016_10
(
    OrderDate datetime2(0) not null,
    OrderId int not null,
    /*..*/
    constraint CHK_Orders2016_10
    check (OrderDate >= '2016-10-01' and OrderDate < '2016-11-01')
) on [FGFastStorage];

create view dbo.Orders(OrderDate, OrderId, /*..*/)
with schemabinding
as
    select OrderDate, OrderId, /*..*/
    from dbo.Orders2014_01

    union all

    select OrderDate, OrderId, /*..*/
    from dbo.Orders2016_10
```

Comparing approaches

Partitioned Tables

- Enterprise and Developer editions only
- 1,000 / 15,000 partitions max
- Moving data between filegroups keeping table online requires coding
- Statistics on the table level
- Same schema and indexes
- No partition-level online index rebuild prior SQL Server 2014

Partitioned Views

- All editions of SQL Server
- 255 tables/partitions max
- Data can be moved between filegroups online (Enterprise Edition) (* it depends)
- Separate statistics per table
- Different schema and indexes per table. **Can combine In-Memory OLTP, B-Tree and columnstore data**
- Online index rebuild within the table (Enterprise Edition) (* it depends)

Challenges

	PARTITIONED TABLES	PARTITIONED VIEWS
Inaccurate statistics	Does not solve	Solves
Different schema and indexing	Not supported	Supported
Different compression	Supported	Supported
Piecemeal restore and partial backup and availability	Supported	Supported
Per-partition online index rebuild	SQL Server 2014+	Supported
Development overhead	None	Significant
Tiered storage and online data movement	Requires development	Supported
Sliding window and <i>metadata-level</i> data purge	Supported	Supported

Tiered Storage and Mixed Workload

B-Tree Indexes
Fusion IO
(or In-Memory OLTP)



Availability SLA: 99.995%
Performance SLA: <10ms

B-Tree Indexes
RAID-10 SSD



Availability SLA: 99.9%
Performance SLA: <50ms

Columnstore Indexes
RAID-5 SAS
R/O Filegroup



Availability SLA: 98%



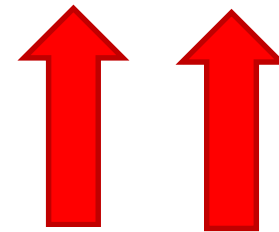
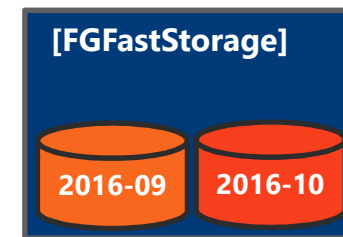
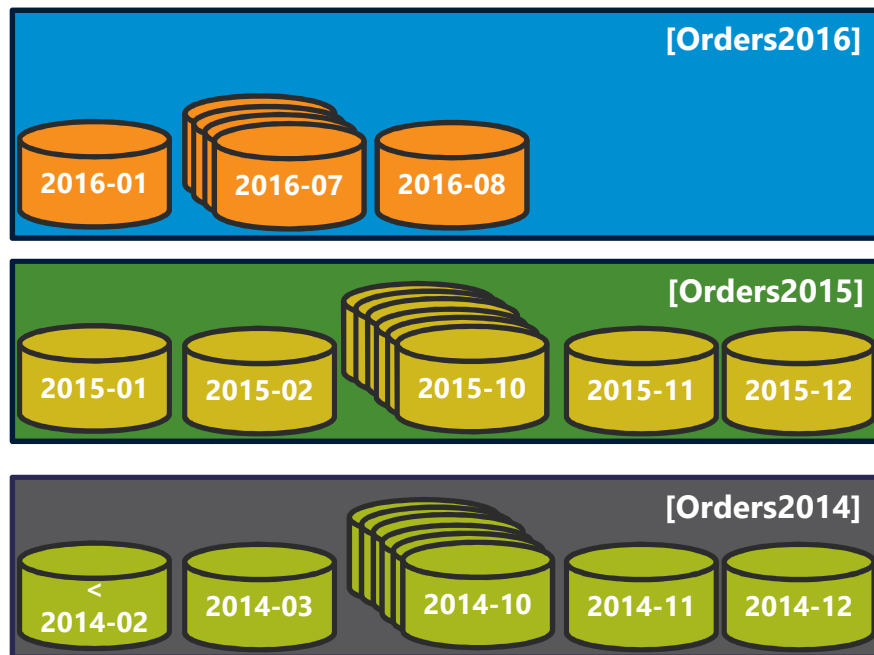
Moving Data

Demo

Moving Data Online

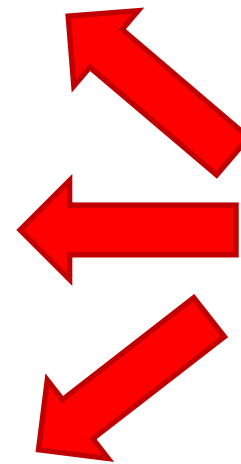
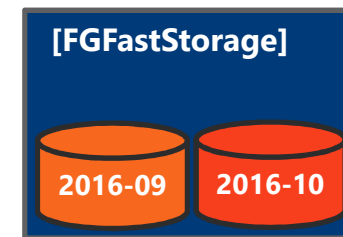
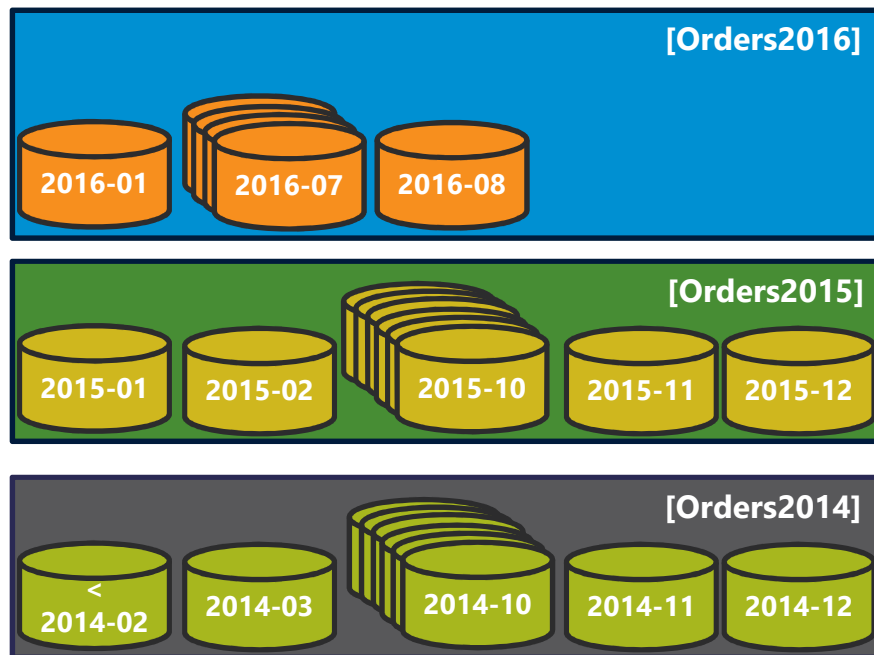
	MOVING PARTITION TO DIFFERENT FILEGROUP	ONLINE INDEX REBUILD WITH LOB COLUMNS IN THE INDEX	ONLINE INDEX REBUILD WITHOUT LOB COLUMNS IN THE INDEX	MOVING DATA TO DIFFERENT DISK ARRAY
SQL Server 2012-2016 Enterprise Edition	Requires development	Supported	Supported	Supported in every edition (adds fragmentation)
SQL Server 2005-2008R2 Enterprise Edition		Not supported	Supported	
Non-Enterprise Editions	N/A	Not supported	Not supported	

Mixed Approach



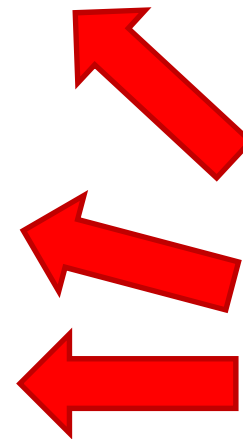
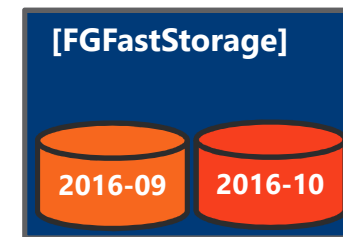
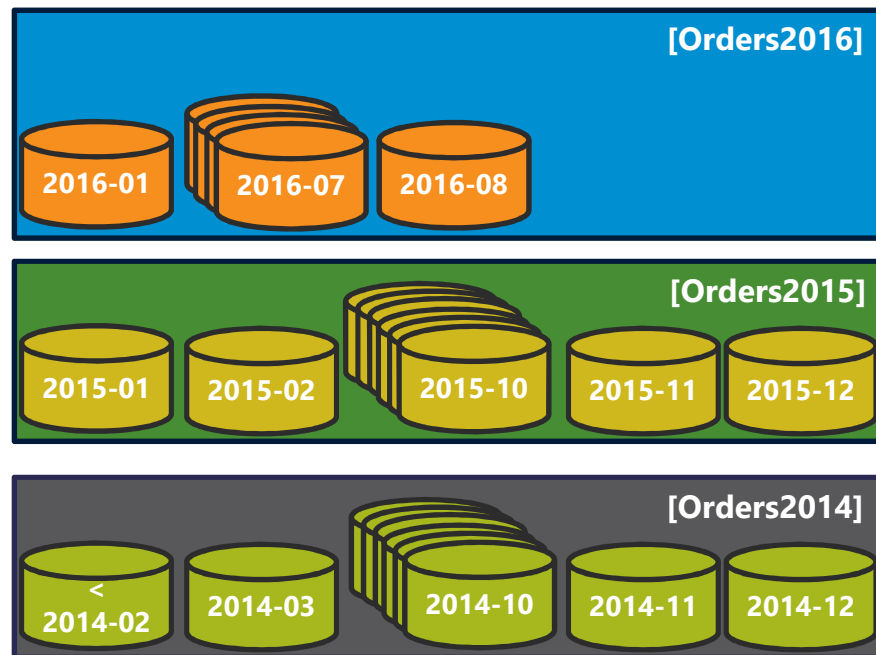
Regular Tables

Mixed Approach



**Partitioned
Tables**

Mixed Approach

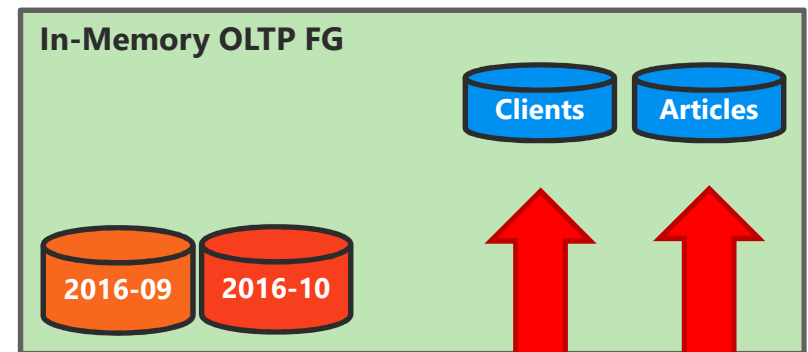
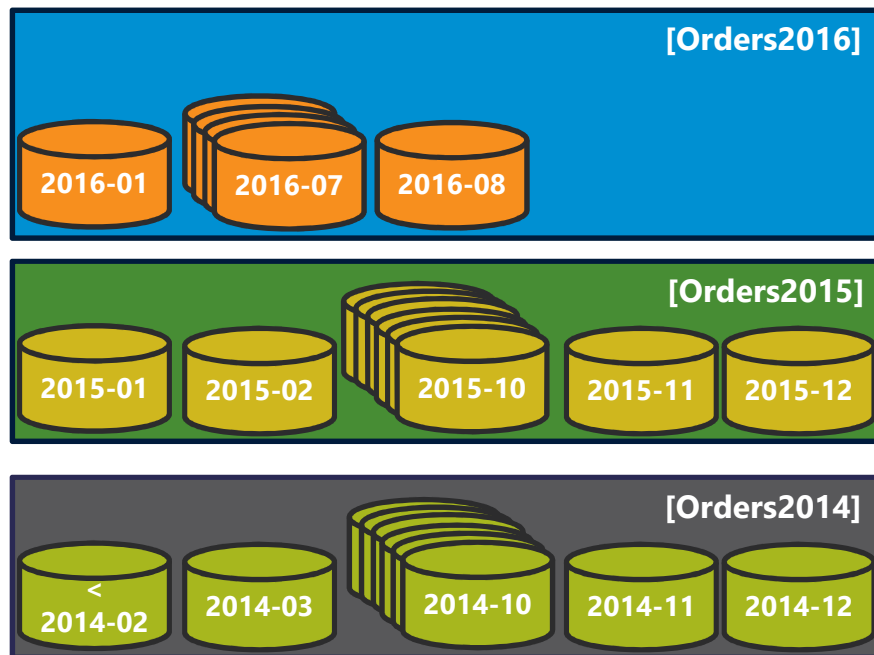


**Partitioned
View**

Mixed Approach

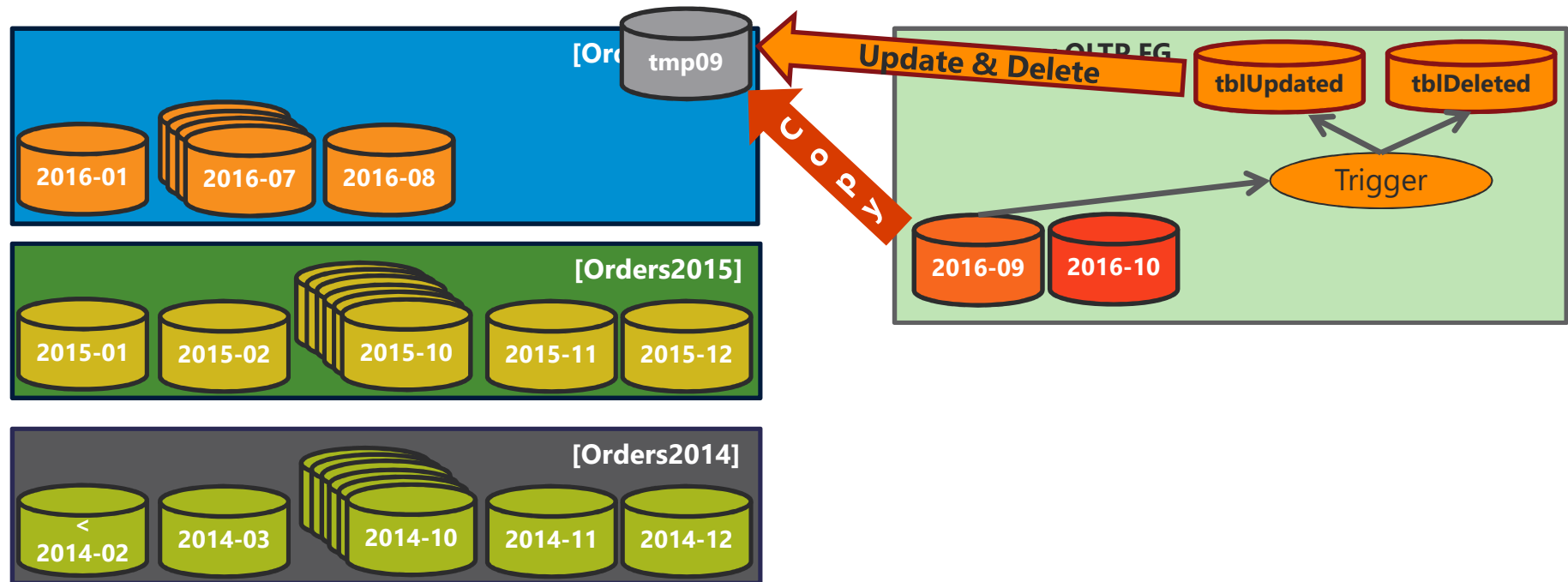


What about In-Memory OLTP?

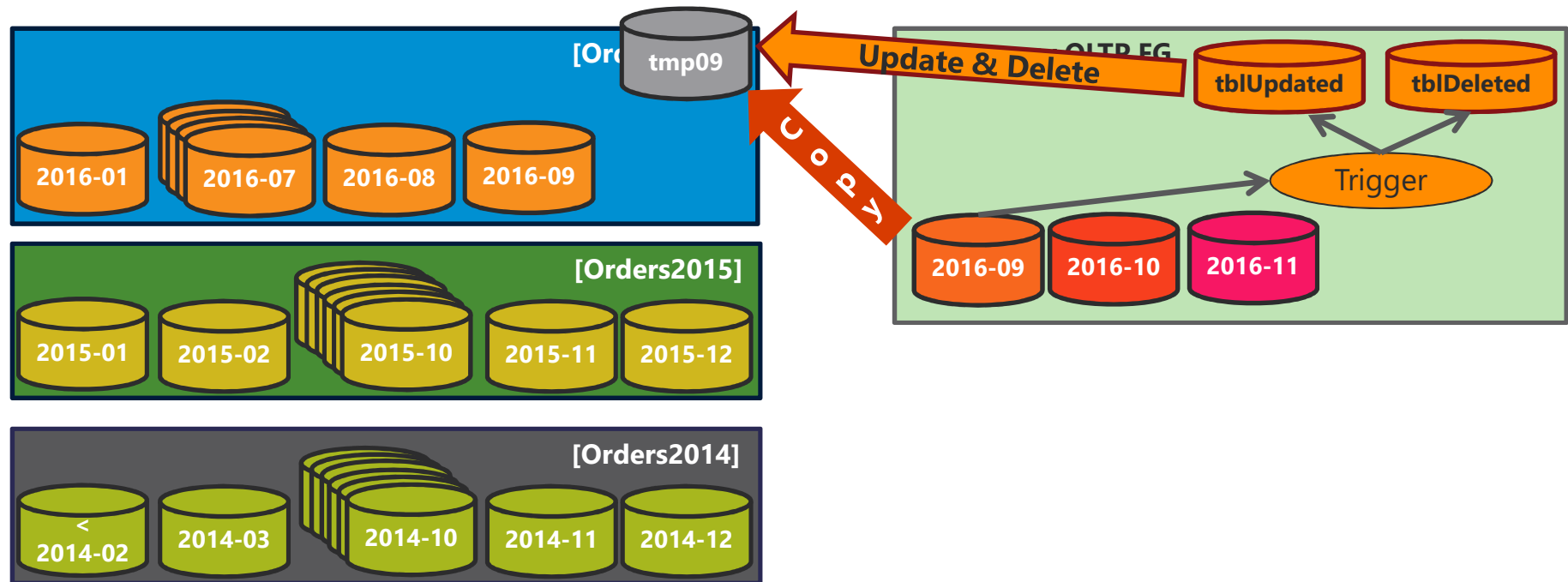


**Native
Compilation**

What about In-Memory OLTP?



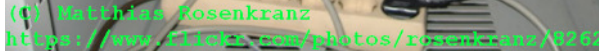
What about In-Memory OLTP?





Moving In-Memory OLTP Data

Demo





Linked Servers

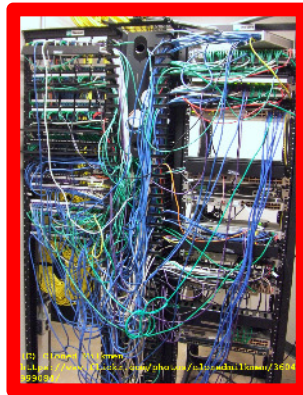
Demo

Ugly Workaround

```
select @ArchivedStores = count(*) from @tvpStores where Archived = 1;
if @ArchivedStores > 0
begin
    set @Stores =
        (select StoreId as [@Id] from @tvpStores where Archived = 1 for xml path);
    insert into #RemoteOrders
        exec RemoteServer.DB.dbo.GetOrders @Stores, @StartDate, @StopDate;
end

select ...
from
    ( select .. from dbo.Orders
      union all
      select .. from #RemoteOrders )
```

Stretch Database



Pros:

- Transparency
- Transactional consistency

Cons:

- Limited control
- price, Price, PRICE!!!!

Outside of Relational Model



Raw Data:

- Sensor Readings
- Geo Locations

Row-by-row processing
=> NoSQL

"Reportable" Data:

- Events / Actions
- Trips

Joins; aggregations; analysis;
reporting
=> Relational DB



Sounds Too Good To Be True

Partitioning affects execution plans

- More at: <http://aboutsqlserver.com/publications>
- “Pro SQL Server Internals”

No uniqueness across all the data

Several other restrictions / incompatibilities



We Covered

Partitioning in the single database

“Distributed data” partitioning

Going non-relational



Any Questions?

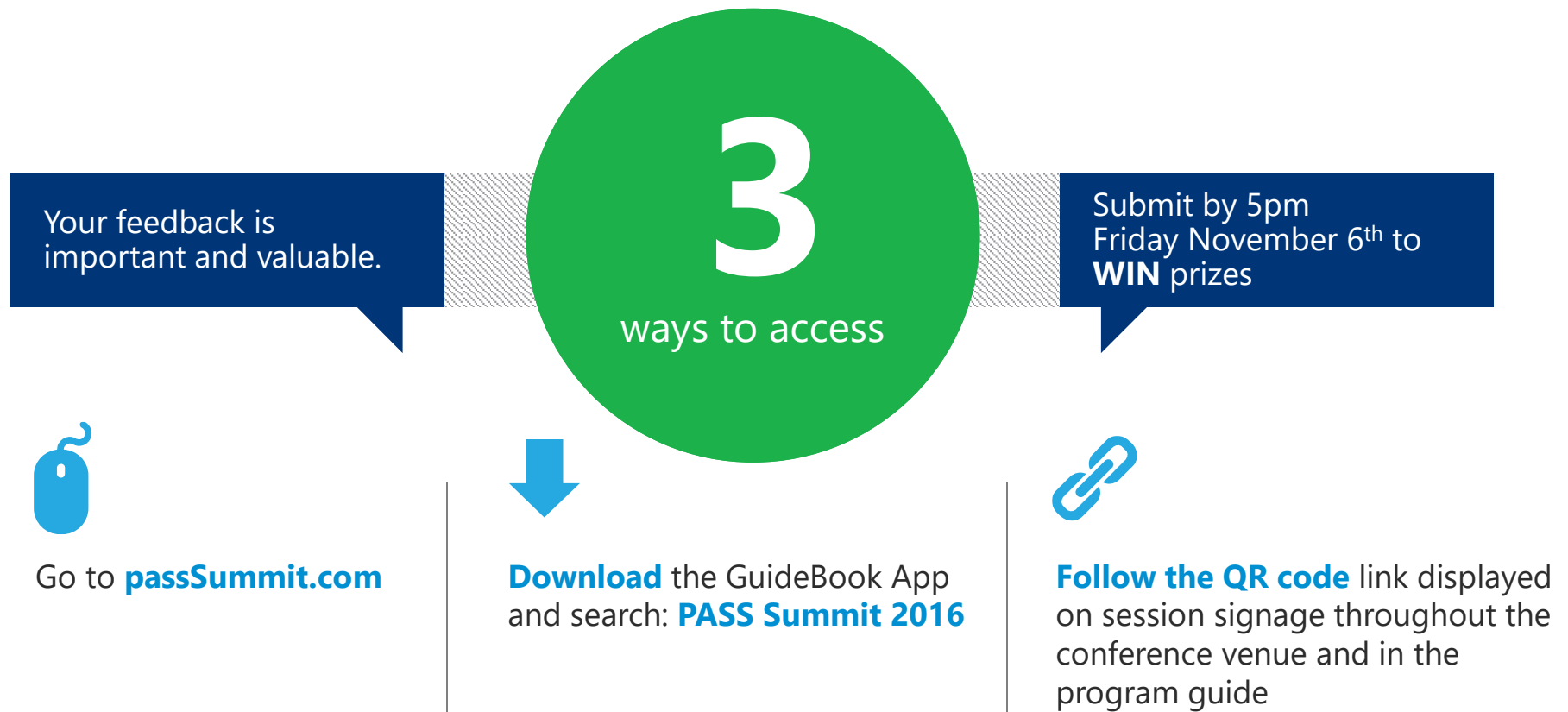
Thank you very much!

Email: dk@aboutsqlserver.com

Blog: <http://aboutsqlserver.com>



Session Evaluations





Thank You

Learn more from
Dmitri Korotkevitch

dk@aboutsqlserver.com

<http://aboutsqlserver.com>