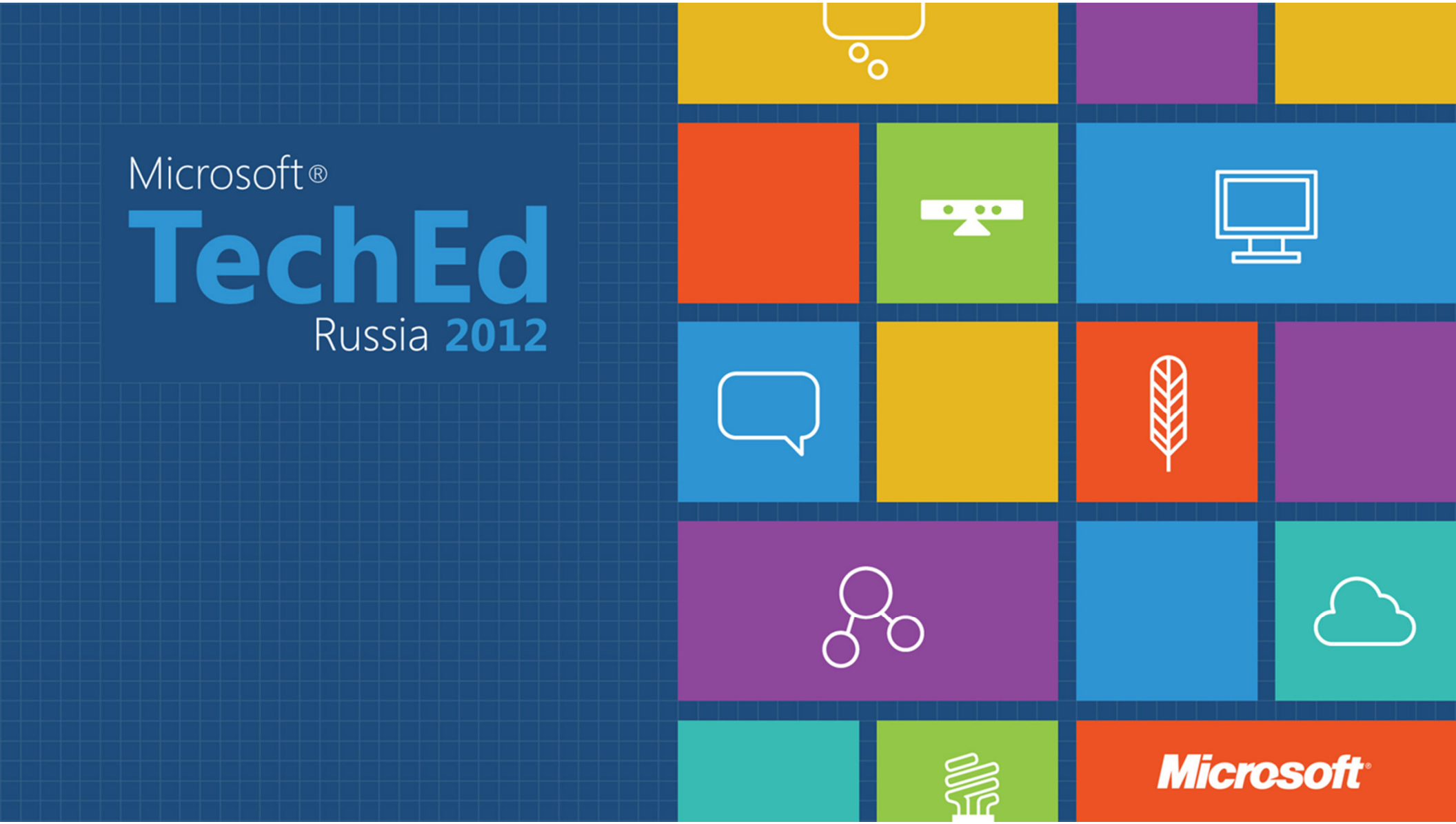

Microsoft®
TechEd
Russia 2012
Microsoft

Microsoft®
TechEd
Russia 2012

Data Partitioning in VLDB

Dmitri Korotkevitch

http://aboutsqlserver.com

About me

- 10+ years of experience working with SQL Server
- Microsoft SQL Server MVP
- Microsoft Certified Master
- Microsoft Certified Professional Developer

- Blog: http://aboutsqlserver.com
- Email: dmitri@aboutsqlserver.com

Agenda

- Why Partitioning?
- Partitioning by Scaling-Out
- Partitioning through Federations in Windows Azure
- Data Partitioning

Why Partitioning?

Everything stored in the same place

Same schema, indexes and compression

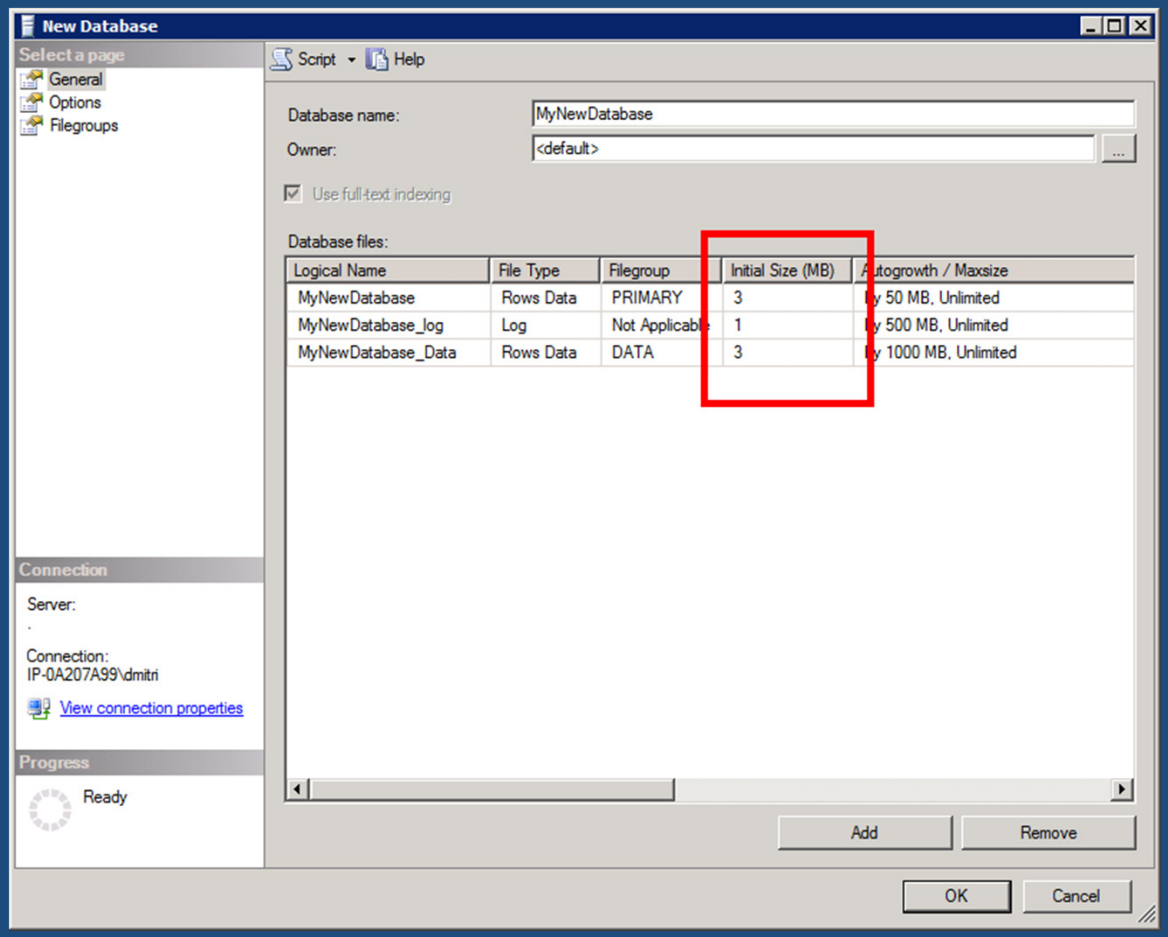
Maintenance challenges

Challenges with Backups

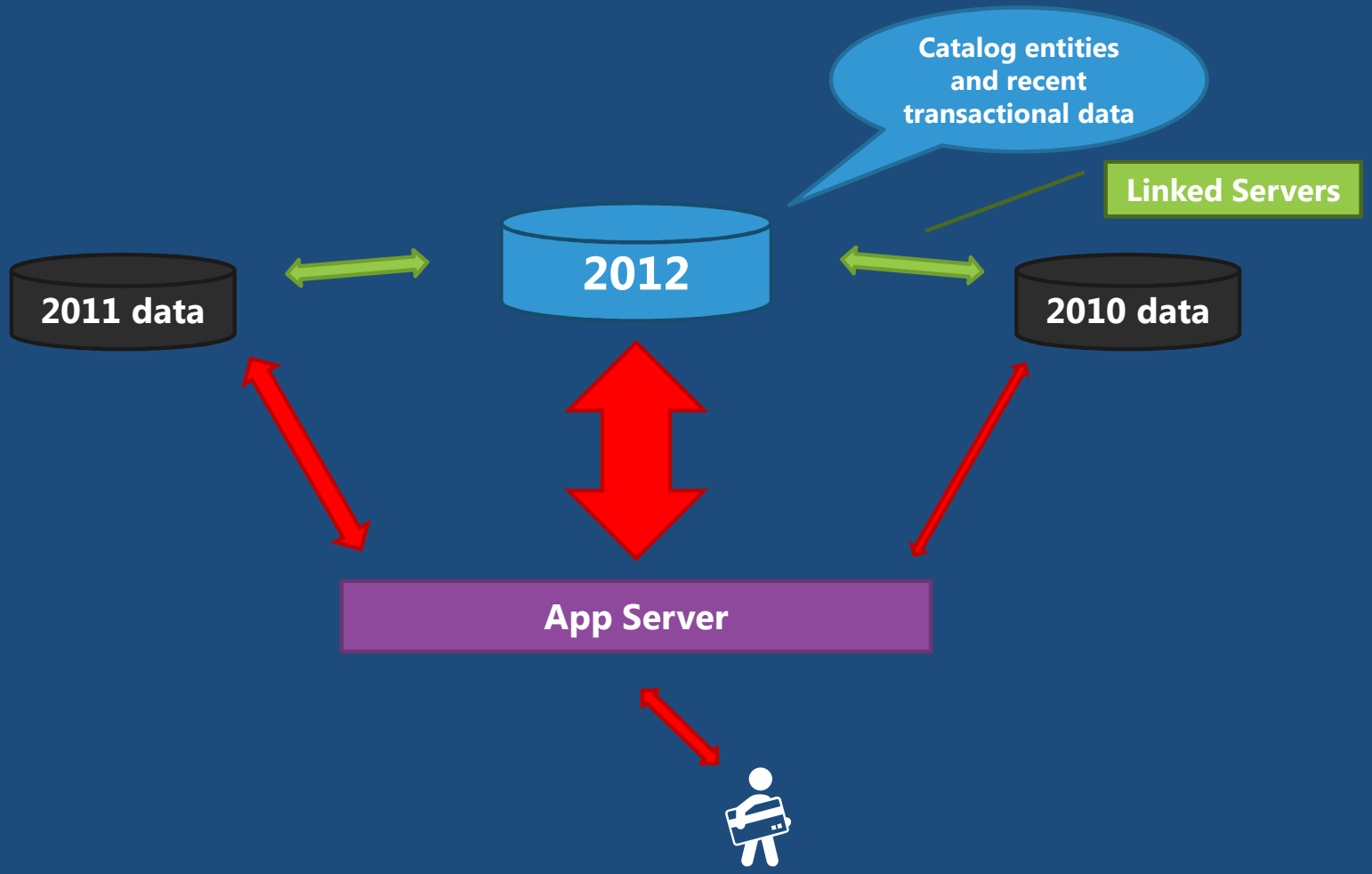
Long Disaster Recovery time

Less accurate statistics

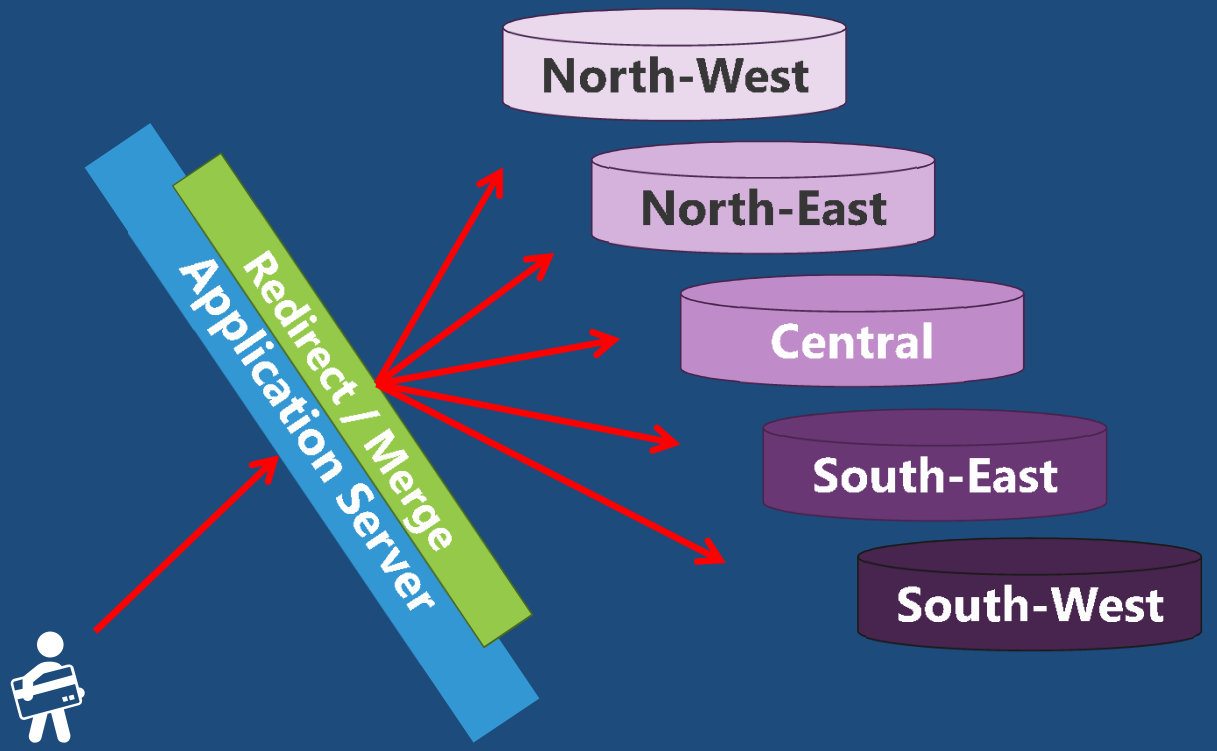
When to partition data?



Vertical Partitioning



Data Sharding (Share-Nothing)



Things to think about

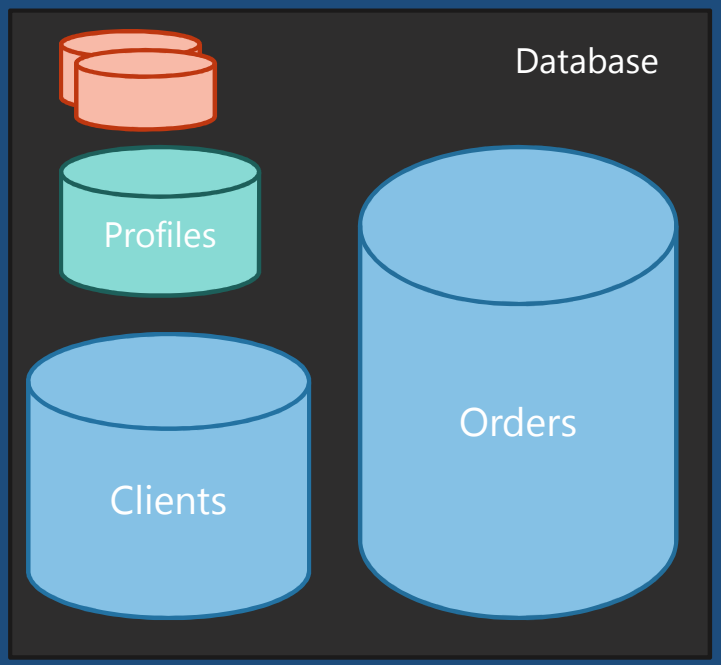
Cons

- **Development cost – YMMV**
- Maintenance cost
- Hardware and License Cost (OS, SQL Server)
- Potential performance problems with Linked servers
- Legacy support

Pros

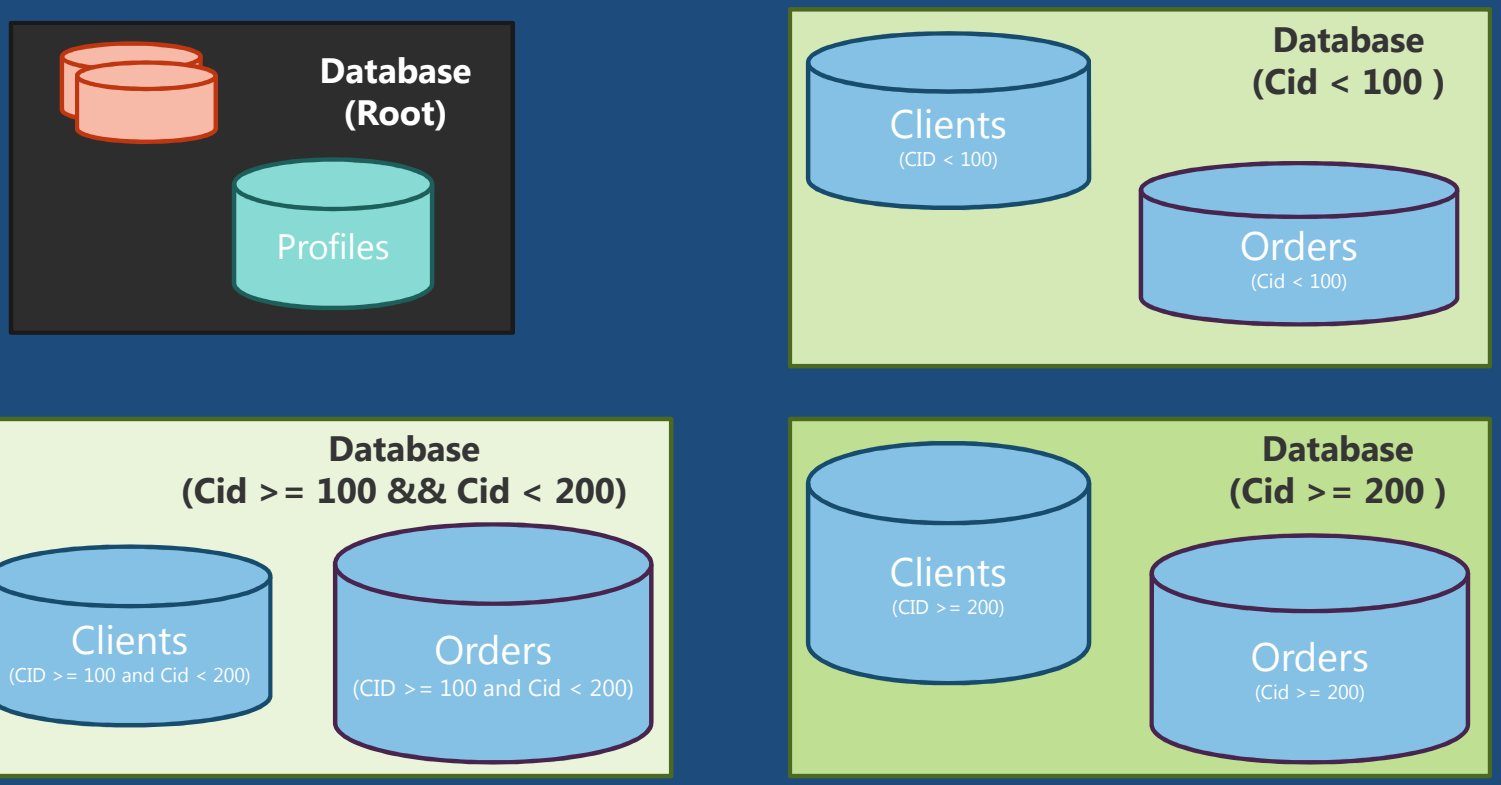
- High***er*** availability in some cases
  - Share Nothing – when shard is down only subset of the customers is affected
  - Vertical Partitioning – when "historical" data is unavailable, customers can still access current data
- Can help with resource bottlenecks
  - Especially with the Clouds

Windows Azure SQL Database Federations

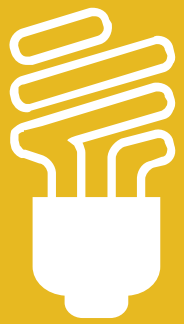



```
CREATE FEDERATION ClientFederation (CID int RANGE)
go

ALTER FEDERATION ClientFederation
SPLIT AT (CID=100)
go

ALTER FEDERATION ClientFederation
SPLIT AT (CID=200)
go
```

Windows Azure SQL Database Federations



Microsoft®
TechEd
Russia 2012

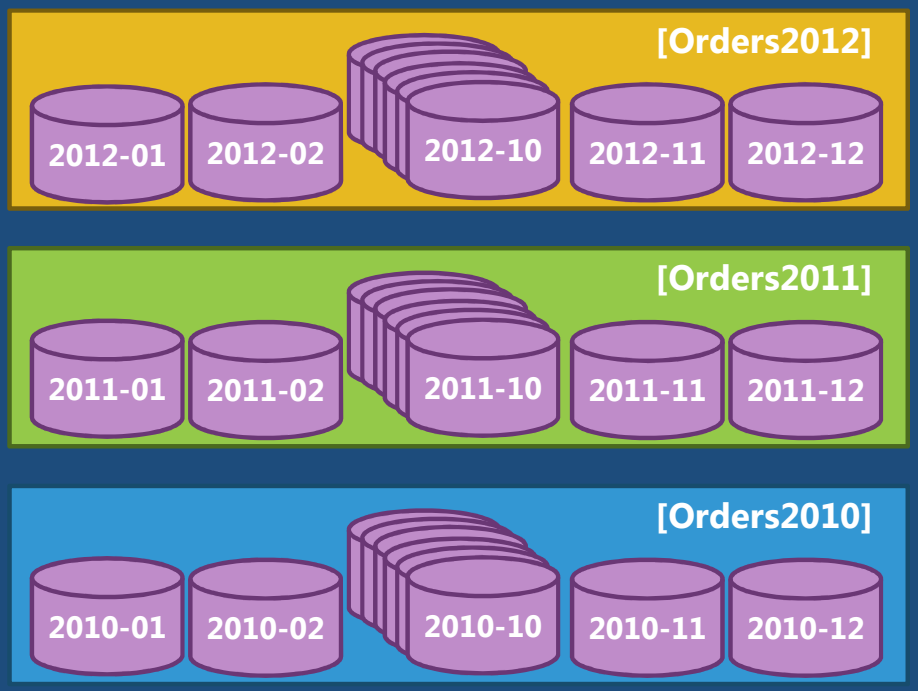


Windows Azure SQL Database Federations

Windows Azure SQL Database Federations

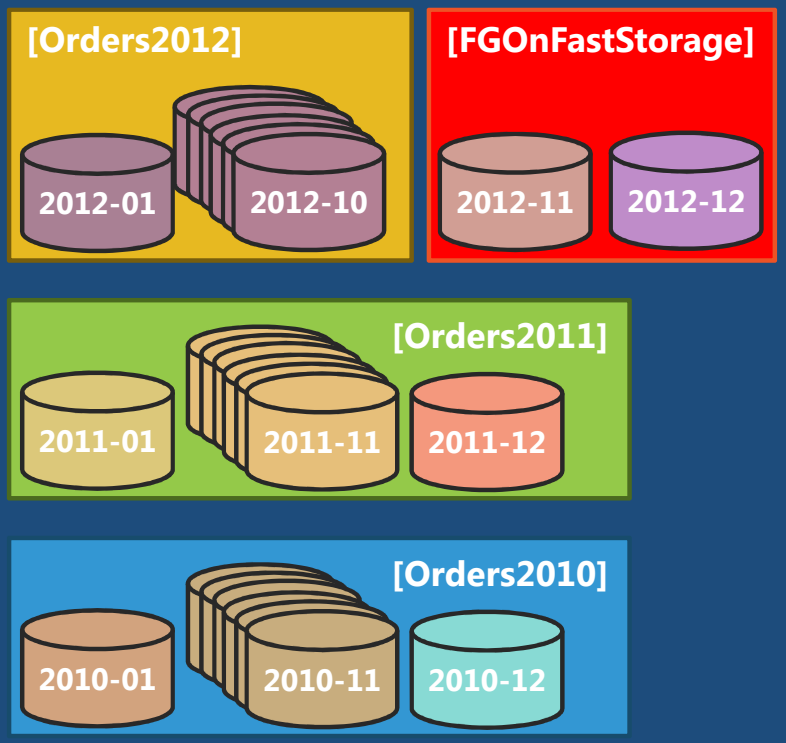
- Share-Nothing approach
  - No cross-database / cross-federations joins
  - No fan-out queries
    - Herve Roggero's Enzo Shard Library on CodePlex
- Federation members can *technically* have different schemas
- Help to scale-out beyond 150GB DB size limit
- Carefully choose Federation Key
  - Federate multiple related tables
  - Split in the way that spreads data equally across the members

Partitioned Tables




```
CREATE PARTITION FUNCTION pfOrders(DATETIME)
AS RANGE RIGHT
FOR VALUES('2010-01-01','2010-02-01',
/*...*/'2011-01-01','2011-02-01',
/*...*/'2012-11-01','2012-12-12');

CREATE PARTITION SCHEME psOrders
AS PARTITION pfOrders
TO ([Orders2010], [Orders2010],
/*...*/ [Orders2011], [Orders2011],
/*...*/ [Orders2012], [Orders2012]);

CREATE TABLE dbo.Orders
(
    OrderID INT not null,
    OrderDate DATETIME not null,
    /*..*/
    CONSTRAINT PK_Orders
    PRIMARY KEY CLUSTERED
    (OrderDate, OrderId)
    ON psOrders(OrderDate)
);
```

Partitioned Views




```
CREATE TABLE dbo.Orders2010_01
(
    OrderID INT not null,
    OrderDate DATETIME not null,
    /*..*/
    CONSTRAINT PK_Orders2010_01
    PRIMARY KEY CLUSTERED (OrderDate, OrderId),
    CONSTRAINT CHK_Orders2010_01 CHECK
    (OrderDate >= '2010-01-01' and OrderDate < '2010-02-01')
) ON [Orders2010];
/*..*/
CREATE TABLE dbo.Orders2012_12
(
    OrderID INT not null,
    OrderDate DATETIME not null,
    /*..*/
    CONSTRAINT PK_Orders2010_01
    PRIMARY KEY CLUSTERED (OrderDate, OrderId),
    CONSTRAINT CHK_Orders2010_01 CHECK
    (OrderDate >= '2012-12-01' and OrderDate < '2013-10-01')
) ON [FGOnFastStorage]
go

CREATE VIEW dbo.Orders(OrderId, OrderDate /*...*/)
WITH SCHEMABINDING
AS
    SELECT OrderId, OrderDate /*..*/ from dbo.Orders2010_01
    UNION ALL
    SELECT OrderId, OrderDate /*..*/ from dbo.Orders2012_12;
```

Comparing Approaches

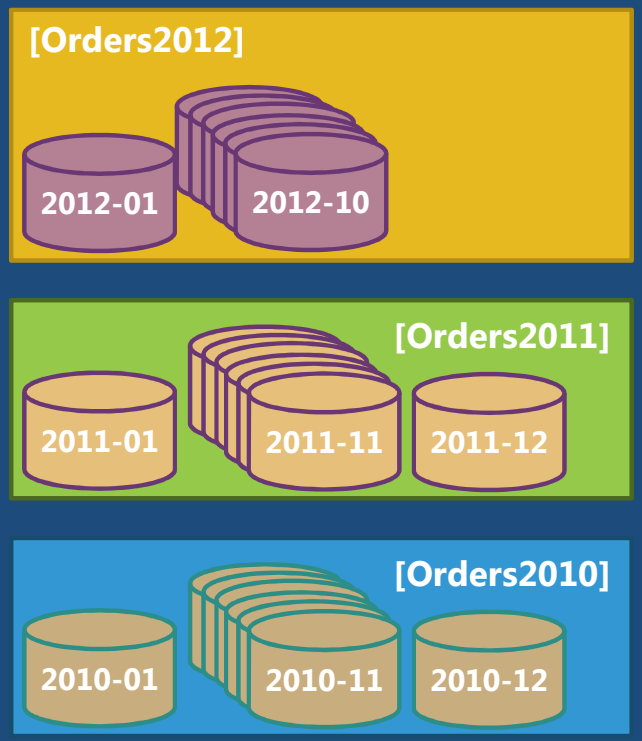
Partitioned Tables

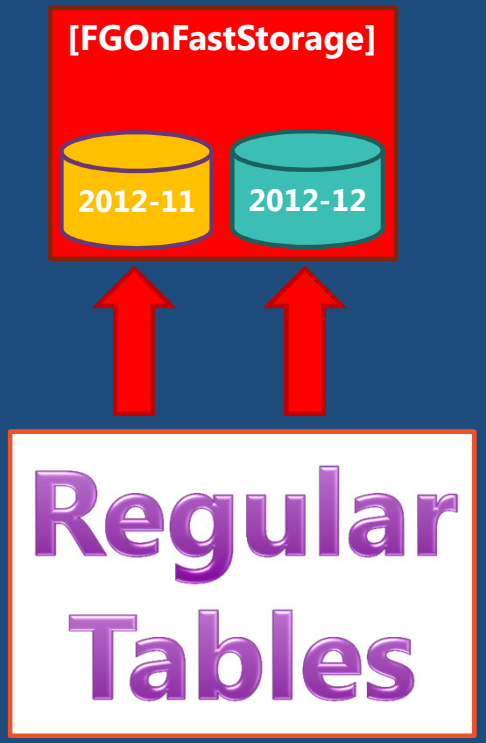
- Enterprise (and Developer) Edition only
- 1,000 / 15,000 partitions max
- No easy way to move partition to another FG online
- Statistics on table level
- One schema and indexes
- No online partition level index rebuild
- Easy maintenance but schema changes are time consuming
- Replication friendly

Partitioned Views

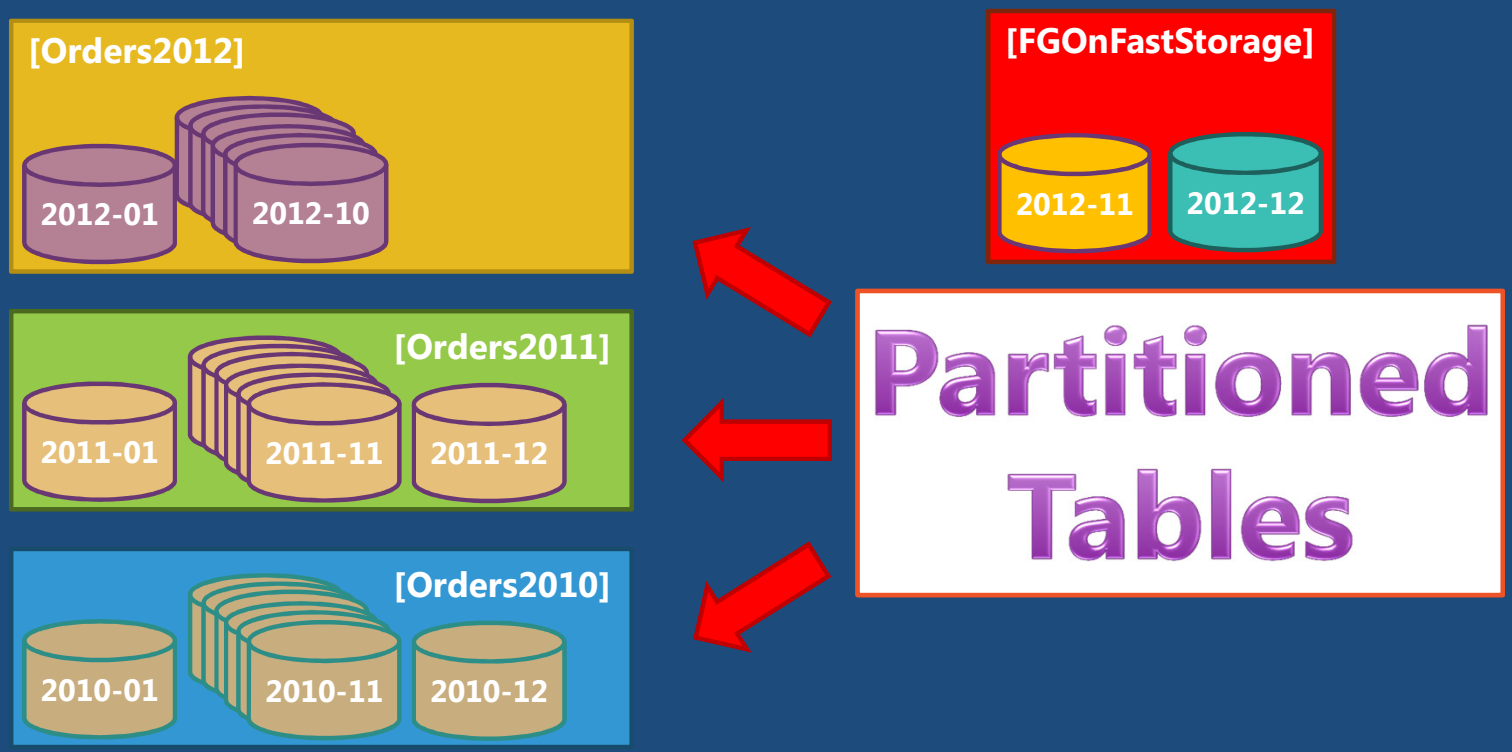
- All SQL Server editions
- 255 tables max
- "Partition" can be moved to another FG online (Enterprise Edition)
- Statistics on "partition" level
- Schema and indexes can vary per "partition"
- Online index rebuild on "partition" level (Enterprise Edition)
- Maintenance overhead
  - Especially with Replication

Mixed Approach

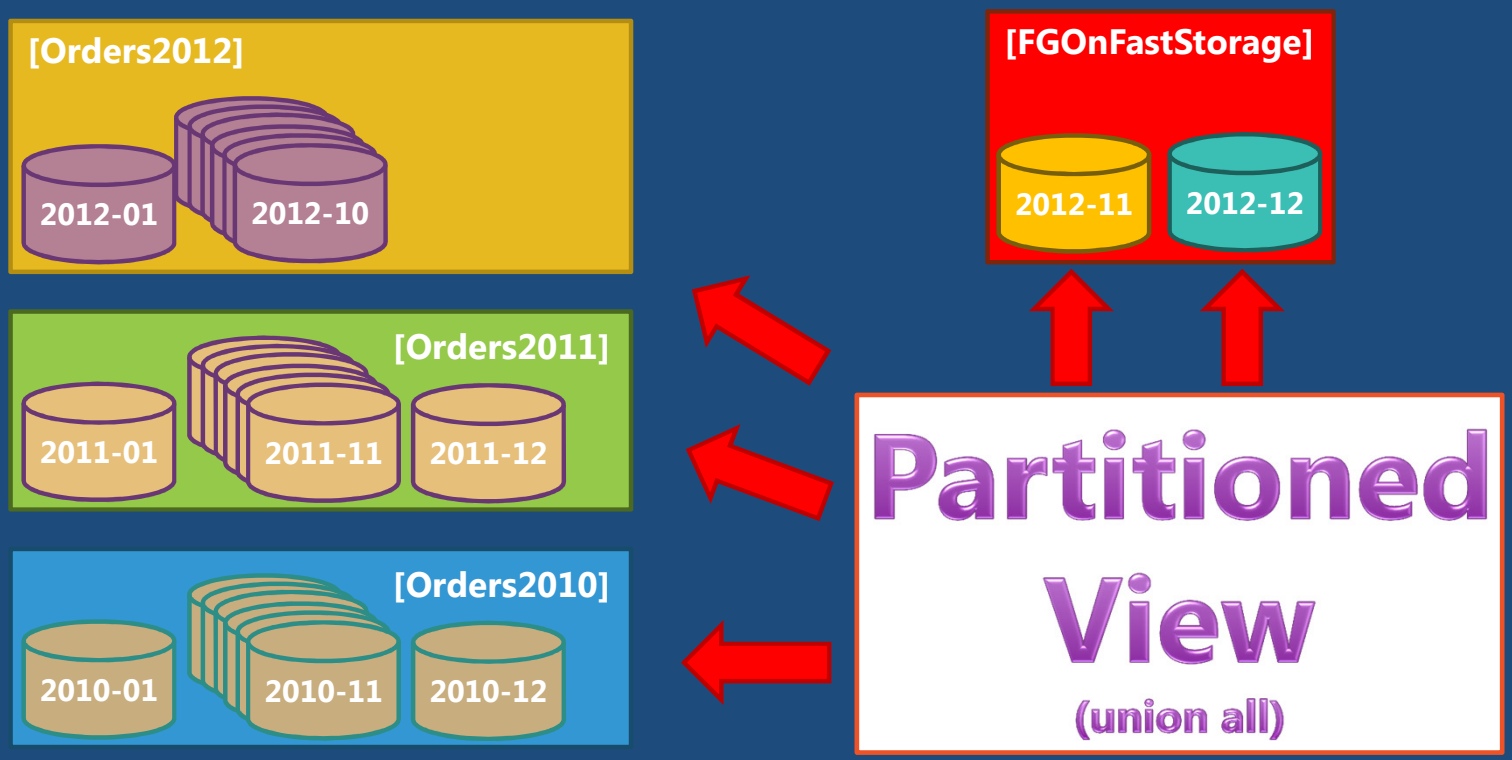




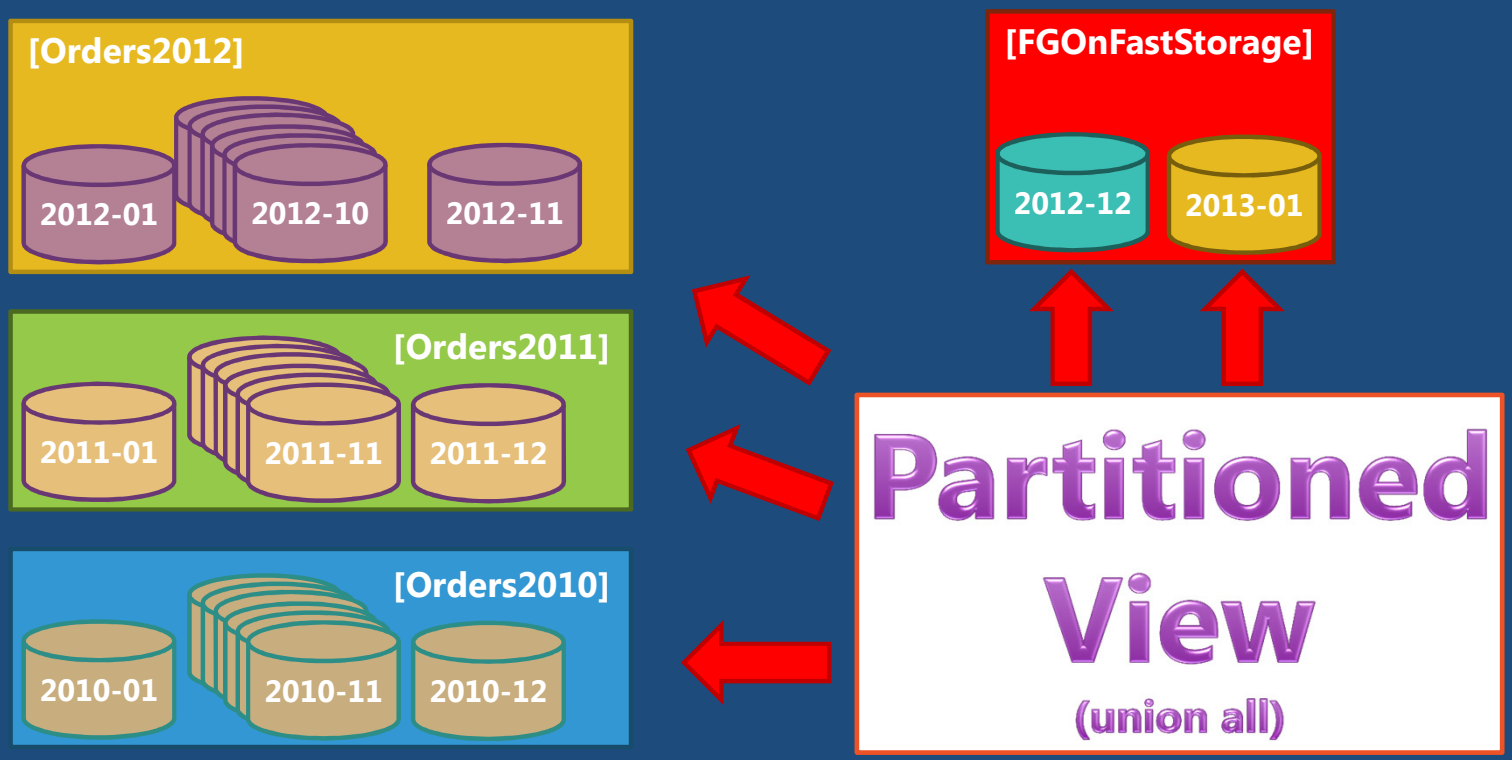
Mixed Approach



Mixed Approach



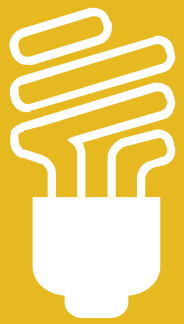
Mixed Approach



Partitioning and Data Layout

- Consider backup and DR strategies and storage system constraints and limitations
- Place related objects together
  - Example: *Orders* and *OrderItems*
- Separate read-only and read-write data
  - Make Filegroups read-only when needed
- HA strategies and dynamic files/FG creation
  - Failover cluster – single copy of DB
  - AlwaysOn, Mirroring, Log Shipping – works as long as folder/disk layout is the same

Microsoft®
TechEd
Russia 2012



AlwaysOn and Partitioning

Potential issues with partitioned tables

- Storage size increase
  - Partition column needs to be added to CI and NCI
    - Example: Datetime (8 bytes) * 10M records per day = ~28GB per NCI per year
- Suboptimal Execution Plans in some cases
- More Info:
  - http://www.simple-talk.com/content/article.aspx?article=1587

We discussed

- Why to partition data
- Different architectural approaches with data partitioning

Q&A

- Thank you very much for attending
- Dmitri Korotkevitch
  - Email: dmitri@aboutsqlserver.com
  - Blog: http://aboutsqlserver.com

© 2012 Microsoft Corporation. All rights reserved. Microsoft, Windows, Windows Vista and other product names are or may be registered trademarks and/or trademarks in the U.S. and/or other countries.
The information herein is for informational purposes only and represents the current view of Microsoft Corporation as of the date of this presentation. Because Microsoft must respond to changing market conditions, it should not be interpreted to be a commitment on the part of Microsoft, and Microsoft cannot guarantee the accuracy of any information provided after the date of this presentation. MICROSOFT MAKES NO WARRANTIES, EXPRESS, IMPLIED OR STATUTORY, AS TO THE INFORMATION IN THIS PRESENTATION.