KPMG
cutting through complexity

Elastic Scale for Azure SQL Databases

Andreas Neuhauser
KPMG Advisory GmbH

25.04.2015

Profile

Andreas Neuhauser

Solution Architect
KPMG Advisory GmbH

Certified Scrum Product Owner
Certified Scrum Master
Certified Professional for Requirements Engineering

aneuhauser@kpmg.at

http://www.kpmg.systems

@andreasneuhauser

2015
Global Azure
BOOTCAMP
Saturday, April 25, 2015

Agenda

Introduction
Microsoft Azure
SQL Database

Sharding
Basics
Why?
Tenancy Models

Elastic Scale

Demos

Microsoft Azure

>57%

Fortune 500 using Azure

>300k

Active websites

More than 1,000,000

SQL Databases in Azure

>30 TRILLION storage objects

>300 MILLION AAD users

>1.65 MILLION

Developers registered with Visual Studio Online

>3 MILLION requests/sec

>13 BILLION authentication/wk

Get started

Visit azure.microsoft.com

SQL Database

Database-as-a-Service

Azure SQL Database

SQL Server database technology as a service

Fully Managed

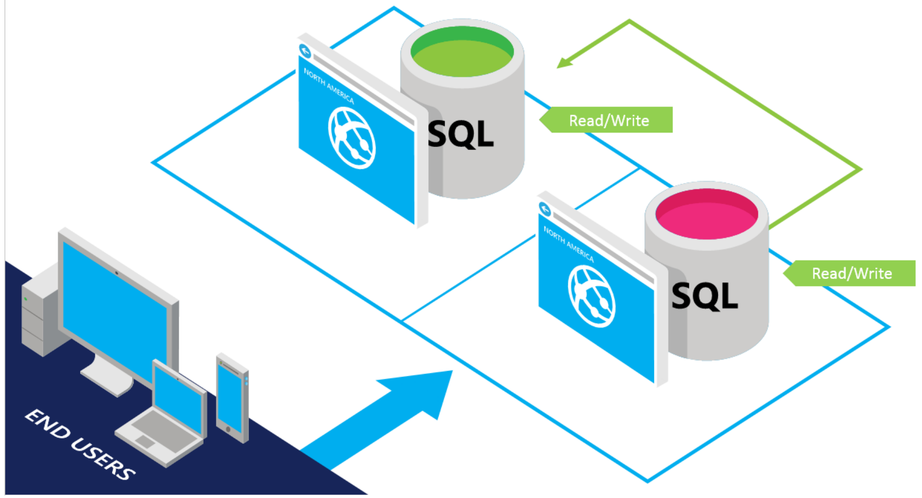
Designed to scale out elastically with demand

Ideal for simple and complex applications

Full support for TDS and ODBC

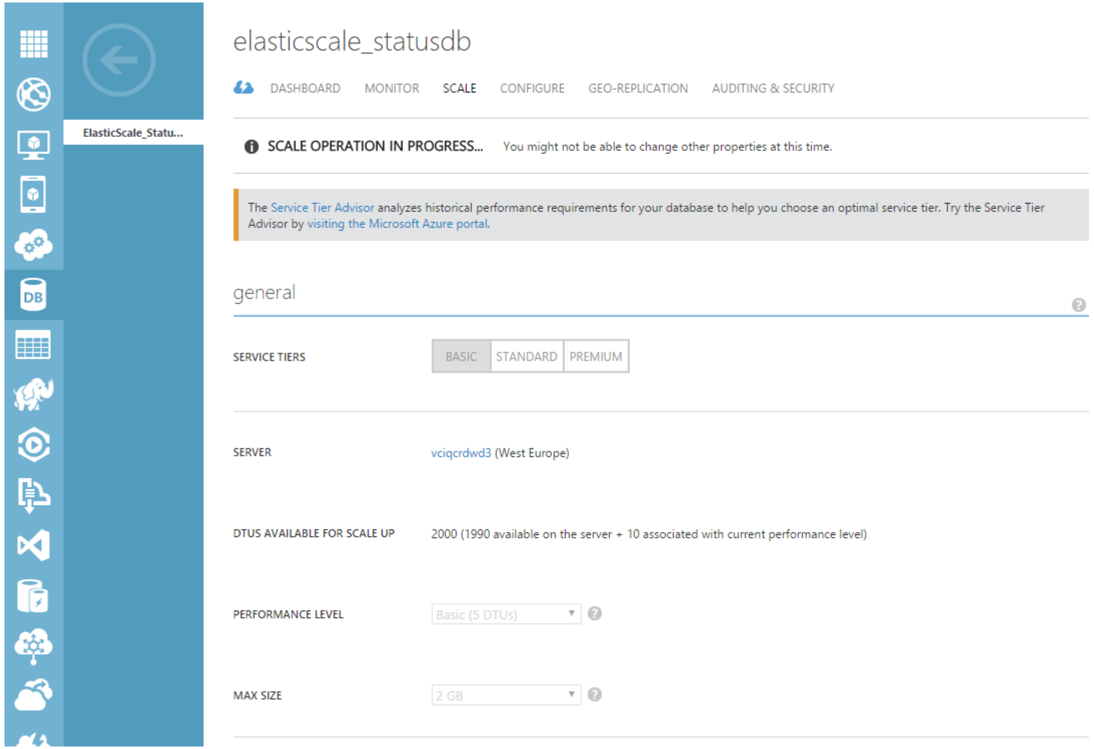
Familiar language and framework support

Cross Datacenter failover and backups to support disaster recovery scenarios



Demo

Service Tiers
Scale Up



elasticscale_statusdb

DASHBOARD MONITOR SCALE CONFIGURE GEO-REPLICATION AUDITING & SECURITY

SCALE OPERATION IN PROGRESS... You might not be able to change other properties at this time.

The Service Tier Advisor analyzes historical performance requirements for your database to help you choose an optimal service tier. Try the Service Tier Advisor by visiting the Microsoft Azure portal.

general

SERVICE TIERS: [✓ BASIC] STANDARD PREMIUM

SERVER: vciqcrdwd3 (West Europe)

DTUS AVAILABLE FOR SCALE UP: 2000 (1990 available on the server + 10 associated with current performance level)

PERFORMANCE LEVEL: Basic (5 DTUs)

MAX SIZE: 2 GB

Sharding

Pattern for the Cloud

Sharding

„Sharding is a horizontal scaling strategy in which resources from each shard (or node) contribute to the overall capacity of the sharded database.“
(Source: Wilder B., Cloud Architecture Patterns)

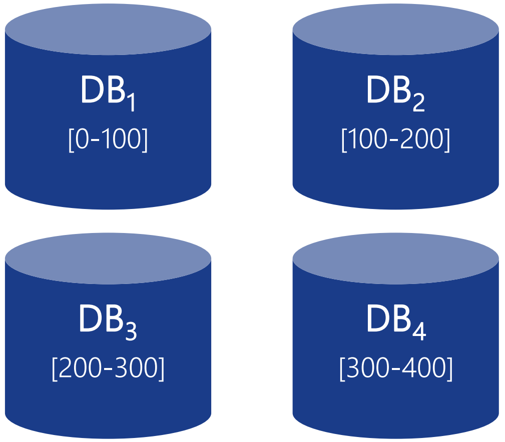
→ „Shared nothing“ Architecture

Shard Key

Determines which shard node stores database row

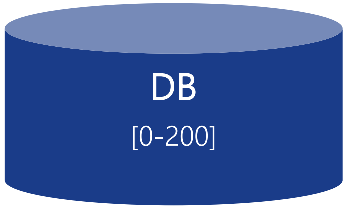
Original database = Collection of all shards

Every shard has the same schema



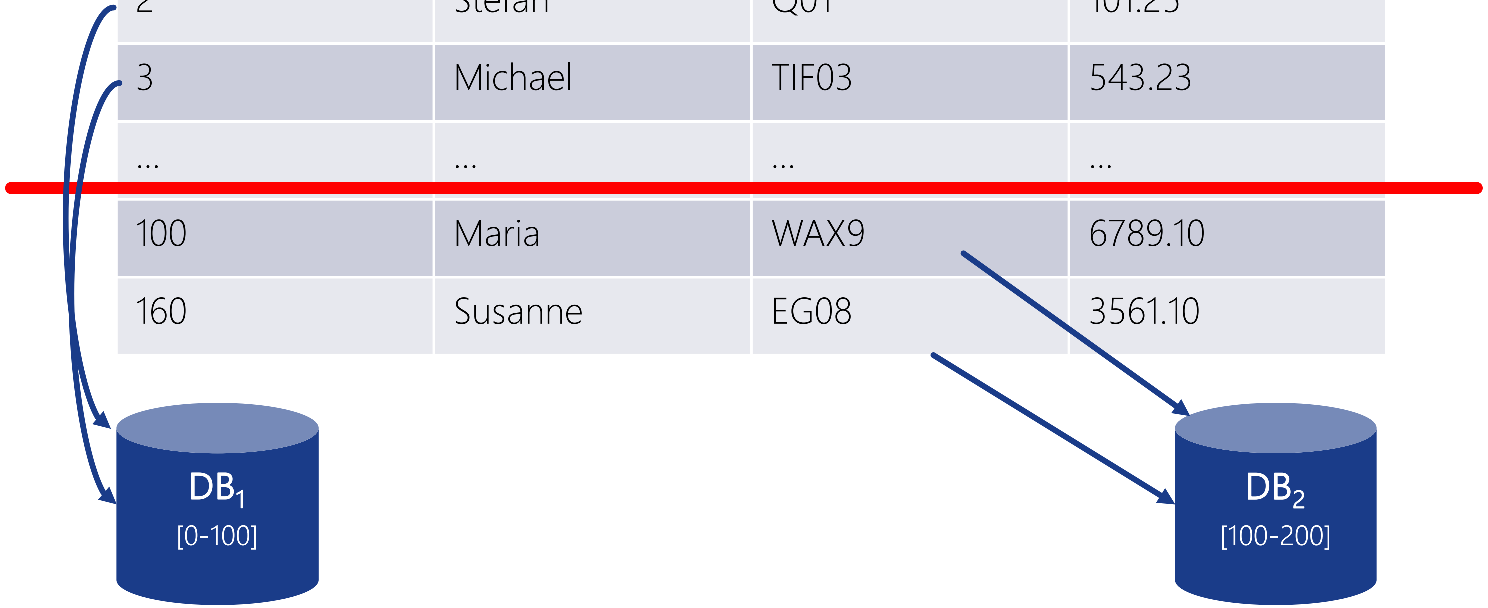
Before

CustomerId	Name	Account	Amount
1	Andreas	VIF	200.36
2	Stefan	Q0T	101.25
3	Michael	TIF03	543.23
…	…	…	…
100	Maria	WAX9	6789.10
160	Susanne	EG08	3561.10

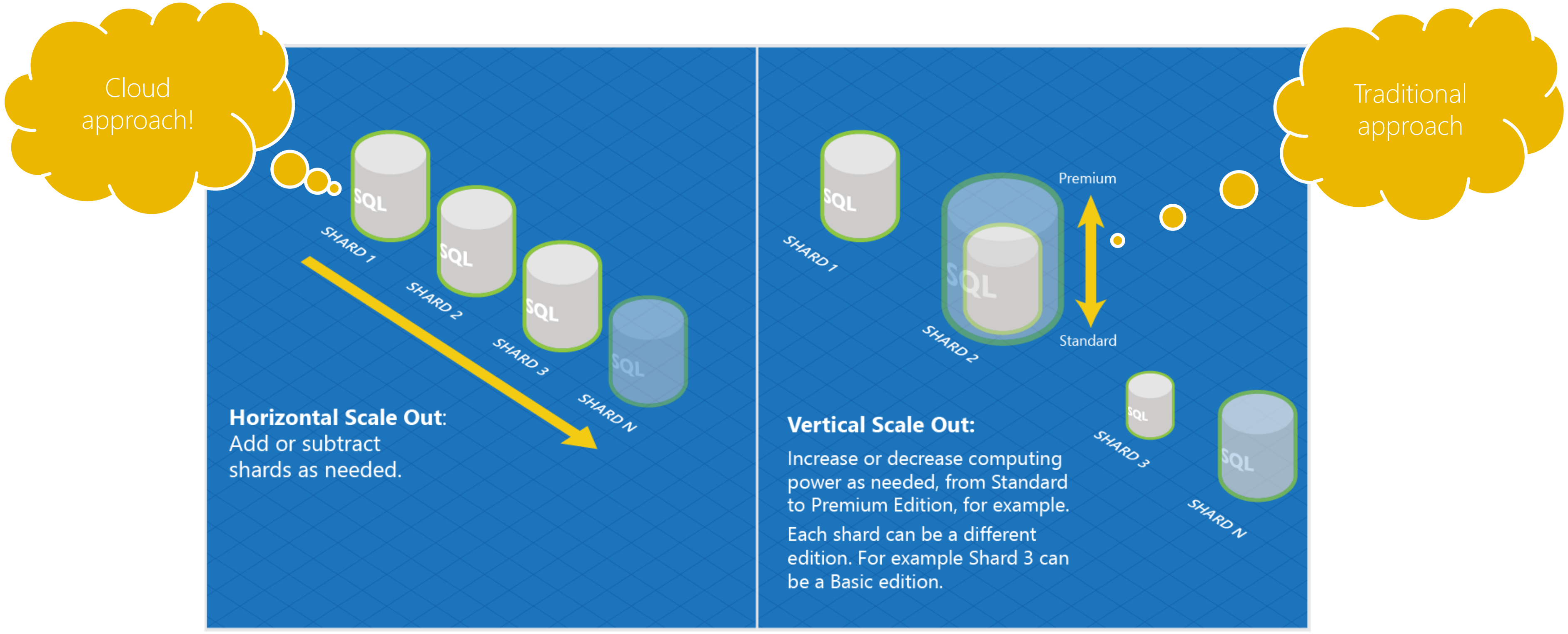


After

CustomerId	Name	Account	Amount
1	Andreas	VIF	200.36
2	Stefan	Q0T	101.25
3	Michael	TIF03	543.23
…	…	…	…
100	Maria	WAX9	6789.10
160	Susanne	EG08	3561.10



Why Sharding?



Horizontal Scale Out: Add or subtract shards as needed.

Vertical Scale Out:

Increase or decrease computing power as needed, from Standard to Premium Edition, for example.

Each shard can be a different edition. For example Shard 3 can be a Basic edition.

Source: http://azure.microsoft.com/en-us/documentation/articles/sql-database-elastic-scale-introduction/

When do sharding?

Amount of data

The total amount of data is too large to fit within the constraints of a single database

Throughput

The transaction throughput of the overall workload exceeds the capabilities of a single database

Isolation

Tenants may require physical isolation from each other, so separate databases are needed for each tenant

Geography

Different sections of a database may need to reside in different geographies for compliance, performance or geopolitical reasons

Not All Tables are Sharded

Sharded Tables

Any given row is stored on exactly one shard node

Responsible for the bulk of the data size and database traffic

Reference Tables

Replicated into each shard to maintain autonomy

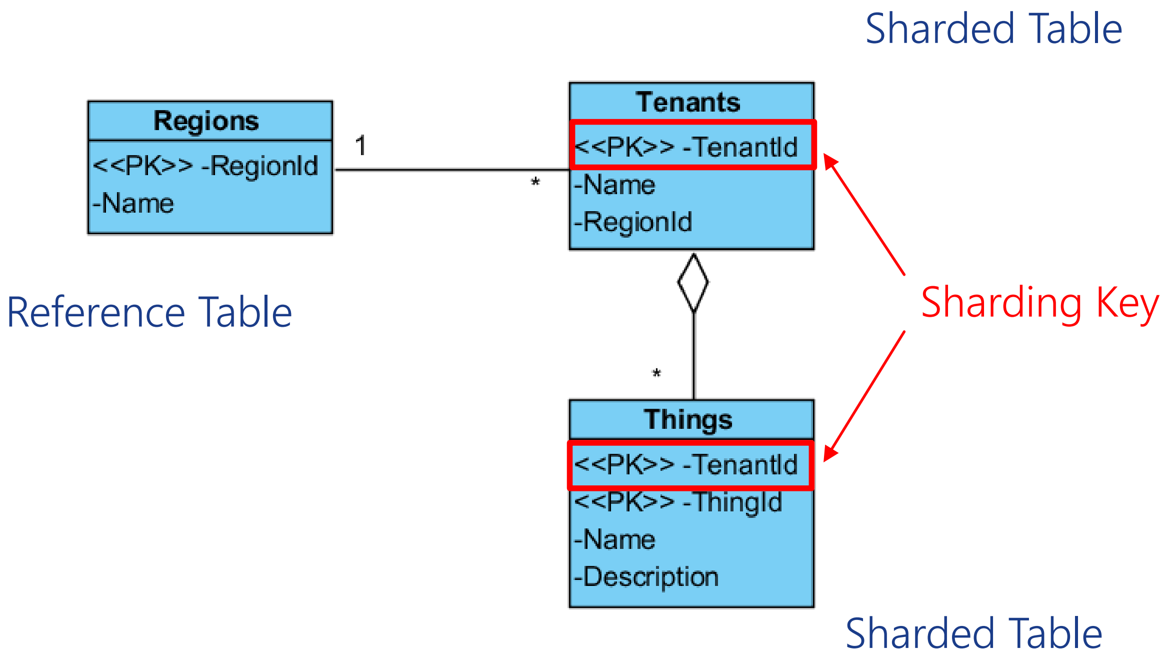
Typically read-mostly and much smaller than business data

→ All of the data needed for queries must be in the shard!

Elastic Scale Client Library

Demo

ShardMapManager (SMM)
Shards
Mappings



Sharding enables Tenancy Models (1/2)

Single Tenancy - Single tenant per database

Each tenant’s data is stored in a different database
Better isolation of tenants as compared to multi-tenant model

Source: flickr.com

Source: flickr.com

Multi Tenancy - Multiple tenants per database

Multiple tenants share the same database
Less isolation of tenants as compared to single tenant model
Typically more cost-effective than the single tenant model

Sharding enables Tenancy Models (2/2)

Hybrid model

Some tenants share databases, others get their own database

E.g., premium or paying customers get their own databases, while free tier customers share databases

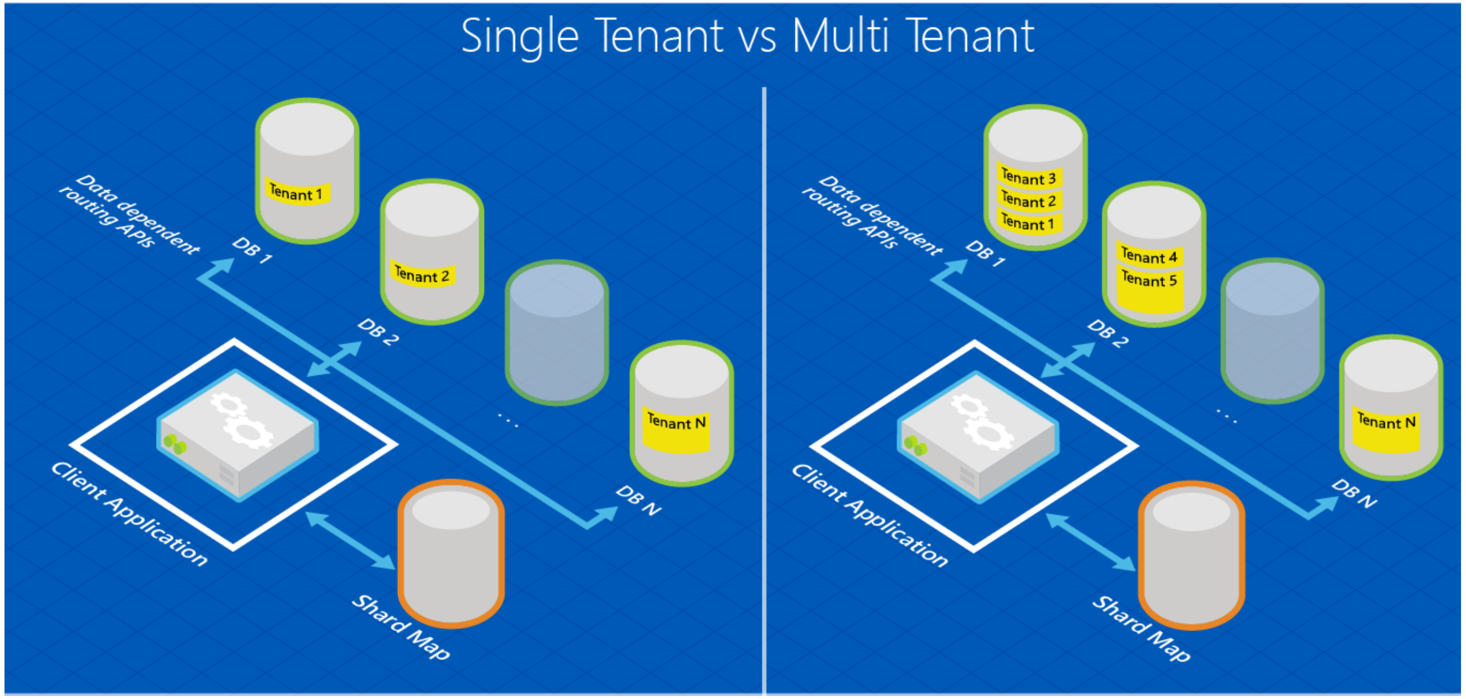
Temporal model

Sharding based on date/time

Most recent shard is constantly loaded with newly arriving data

New shards added when current most recent shard nears capacity

Single vs. Multi Tenant Sharding



Source: http://azure.microsoft.com/en-us/documentation/articles/sql-database-elastic-scale-introduction/

Elastic Scale

Sharding Out-of-the-Box

Elastic Scale

Integrated Sharding support in Azure SQL Database

Provides client libraries and service offerings for sharding

Pushes complexity down the stack towards database

Makes scaling the data tier as easy as the frontend

Appears as a single database to the application → One ConnectionString

Public Preview

Latest version on NuGet: 0.8.0 (March 2015)

Entity Framework Support

Key Capabilities

Shard map management (SMM)

Define groups of shards for your application
Manage mapping of routing keys to shards

Data dependent routing (DDR)

Route incoming requests to the correct shard
Ensure correct routing as tenants move
Cache routing information for efficiency

Multi-shard query (MSQ)

Interactive processing across several shards
Same statement executed on all shards with UNION all semantics

Split/Merge (SM)

Grow or shrink capacity by adding or removing scale units
Dynamically adjust scale factor of scale unit
Trigger adjustment dynamically through policies

Shard Elasticity (SE)

Dynamically adjust scale factor of scale unit
Trigger adjustment dynamically through policies

Why Elastic Scale?

Past

Not popular because sharding logic was custom-built in application code
Increase in cost and complexity

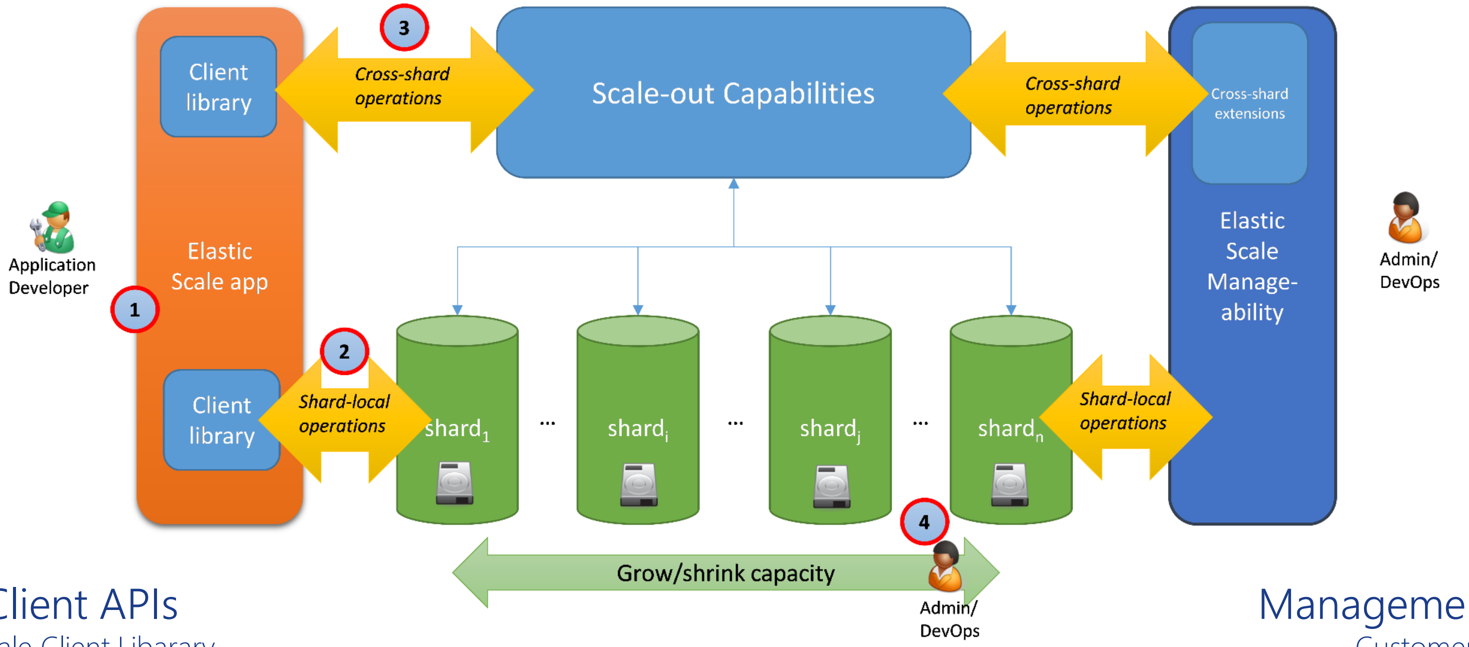
Today: prevent self-sharding

A developer should focus on the business logic rather than building infrastructure for sharding
Focus on application not scalability!

Query one specific shard, Query multiple shards
Application Developer

Capacity, Cost Management, DB Maintenance, DDL
Admin/DevOps

Sharding with Elastic Scale



.NET Client APIs

Elastic Scale Client Libarary

Latest version on NuGet: 0.8.0 (March 2015)

Management Services

Customer-hosted Service

Latest version on NuGet: 0.8.0 (March 2015)

Source: http://azure.microsoft.com/en-us/documentation/articles/sql-database-elastic-scale-introduction/

Elastic Scale Client Library

Demo

Data-Dependent Routing

The same way you've always done it!

```
<connectionStrings>
  <add name="ConnectionString"connectionString=
     "Data Source= [server].database.windows.net;
     Integrated Security=False;
     Initial Catalog=ProductsDb;
     User Id=[login]@[server];
     Password=[password];
     Trusted_Connection=False;
     Encrypt=true;"
    providerName="System.Data.SqlClient"/>
</connectionStrings>
```

Elastic Scale Client Library

Demo

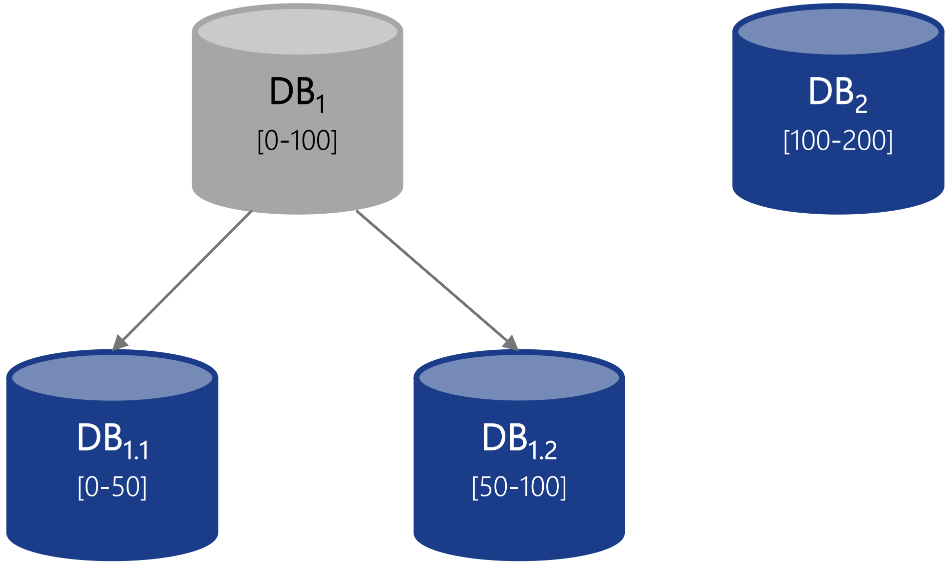
Multi-Shard Querying
(UNION)

Source: flickr.com

Management Services

Demo

Splitting



Split-Merge Service

Customer-hosted Service

1 Worker and 1 Web Role

Security

SSL, Certificate-based client authentication, More

Batch

Shardlets are offline for data-dependent routing during movement

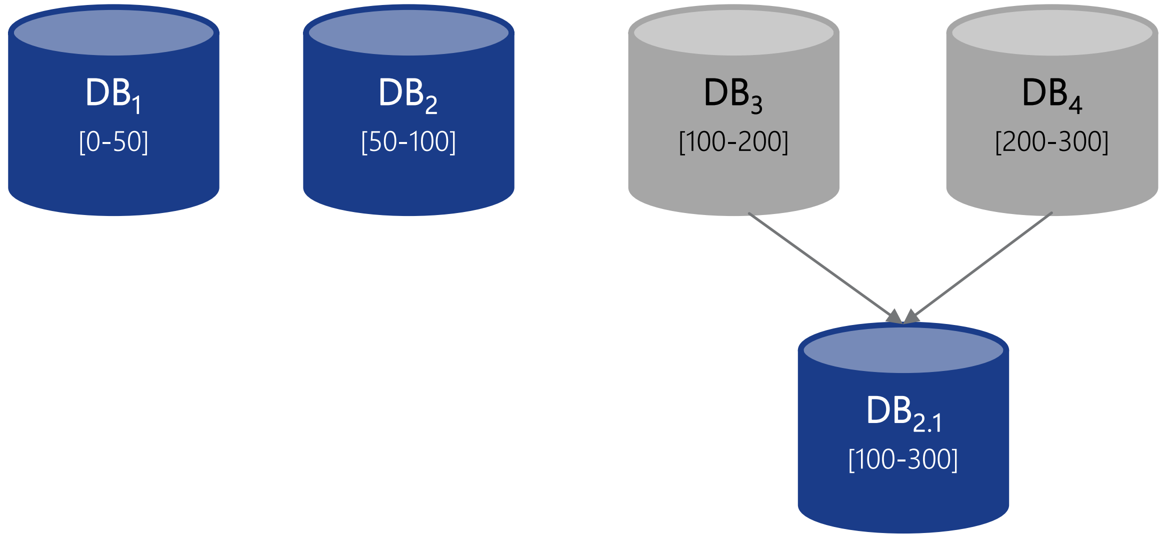
Note

Only needed when existing data needs to be moved!

Management Services

Demo

Merging



Management Services

Demo

Shardlets going „crazy“
→ Dedicated database

Limitations & Best Practices

Service

Shard must exist before Split-Merge operation
Host service in the region where databases reside
Delete Split-Merge service when not performing split/merge/move frequently
Don’t use for production

Sharding Key

Leading column in PK ensuring best performance

More Performance during Split/Merge?

Choose more performant service tiers; Increase only for defined limited period of time

Wrap Up

Elastic Scale

is a Dev-Ops story

enables secure Multi Tenancy and Flexible Data Management

No big changes but **BIG** implications

One Connection String as always

1 Global Application but Data stored nearby customer

No additional costs

Tools

Currently Best option for Split-Merge: PowerShell approach

Shard Elasticity = SQL Database + Azure Automation Service

Links and Resources

Elastic Scale Presentation and Sample

https://speakerdeck.com/aneuhauser
https://github.com/aneuhauser/Samples

Shard Elasticity with Elastic Scale

https://gallery.technet.microsoft.com/scriptcenter/Elastic-Scale-Shard-c9530cbe?clcid=0x409

Azure PowerShell

https://github.com/Azure/azure-powershell

Split/Merge Service Deployment

http://azure.microsoft.com/en-us/documentation/articles/sql-database-elastic-scale-configure-deploy-split-and-merge/

Entity Framework Integration

http://azure.microsoft.com/en-us/documentation/articles/sql-database-elastic-scale-use-entity-framework-applications-visual-studio/

KPMG

cutting through complexity

Q&A

Andreas Neuhauser
KPMG Advisory GmbH

aneuhauser@kpmg.at

http://www.kpmg.systems

@andreasneuhauser