

Multitenancy con SQL Server e Azure SQL Database



Sponsors



About me

- Gianluca Hotz | @glhotz | ghotz@ugiss.org
- Independent Consultant
 - 25+ years on SQL Server (from 4.21 back in 1996)
 - Database modeling & development, sizing & administration, modernization (upgrades & migrations), performance tuning, security
- Community
 - 25 years Microsoft [MVP](#) SQL Server/Data Platform (since 1998)
 - VMware Experts SQL Server
 - Founder and president [UGISS](#) (ex «PASS Chapter»)



Nel 2023,
industria SaaS ha
un valore
approssimativo
~\$195 miliardi
([Gartner](#)).

Negli ultimi 7
anni, industria
SaaS è cresciuta
di **~500%**

App SaaS
costituiscono
70% uso totale
del software
azienda
([BetterCloud](#))

Startup SaaS
stanno [facendo](#)
enorme avanzata
per diventare
prossima
industria da **1**
trilione di \$

Condivisione risorse di elaborazione, rete e archiviazione tra utenti di una soluzione (tenant), garantendo isolamento e sicurezza

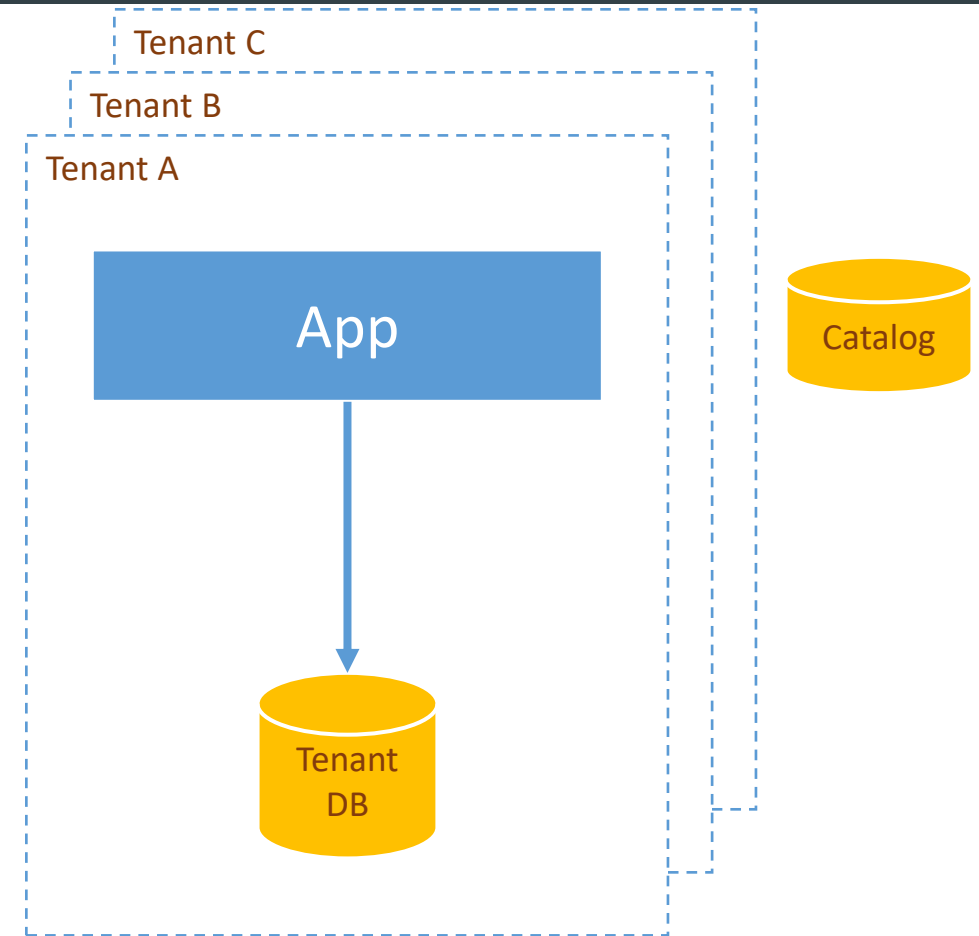
Fondamentale per creatori soluzioni SaaS massimizzare utilizzo risorse, controllare costi e ottenere margini elevati

Progettare e gestire soluzioni multi-tenant richiede ricerca miglior compromesso tra prezzo, prestazioni e gestione

- «Tenant»
 - Letteralmente: «inquilino», «locatario», «affittuario»
 - Wikipedia^[1]: multi-tenant si riferisce ad una architettura software in cui una sua singola istanza è eseguita da un server ed è fruita da diverse organizzazioni che, ciascuna con le sue peculiarità ambientali che costituiscono concettualmente uno specifico tenant (come in un immobile le cui unità o vani sono affittati a locatari diversi)
 - In pratica, nel contesto SaaS: il cliente del servizio
- «Sharding»
 - Partizione orizzontale dati per riduzione in componenti più piccole e veloci da gestire
 - In pratica, nel contesto SaaS: partizionamento per tenant ovvero cliente
 - Per praticità, di seguito sharding key = identificativo tenant
- «Data-Dependent Routing»
 - DDR è la capacità di utilizzare i dati in una query (da un catalogo) per indirizzare la richiesta a un database appropriato

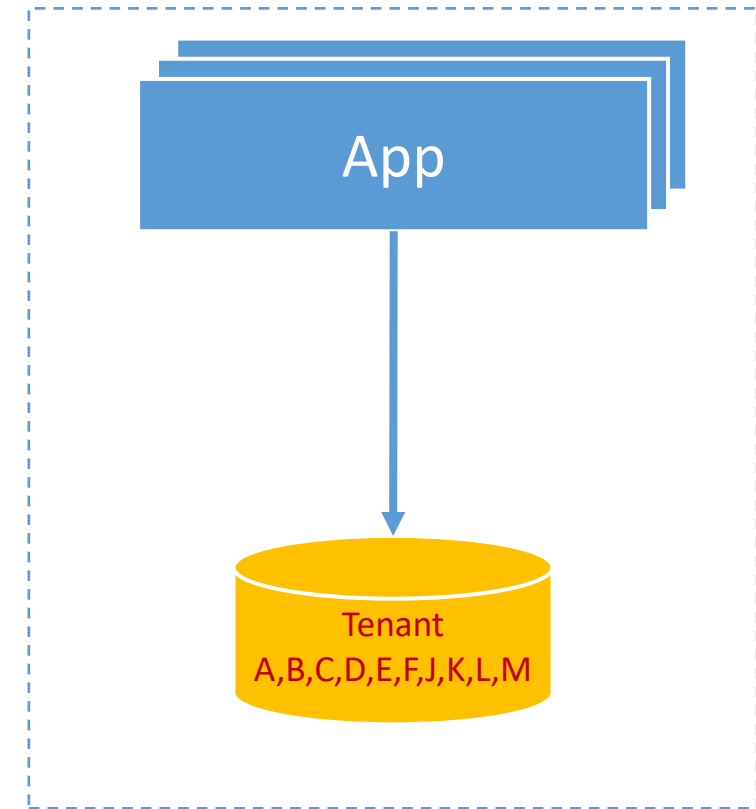
«Single-tenant application & database»

- Un'istanza applicativa per tenant
- Un database per tenant
- Catalogo opzionale (consigliato)
- Massimo isolamento
 - Sicurezza, disponibilità, prestazioni
- Massima flessibilità
 - Livelli di servizio personalizzati
 - Diverse versioni/rilasci scaglionati

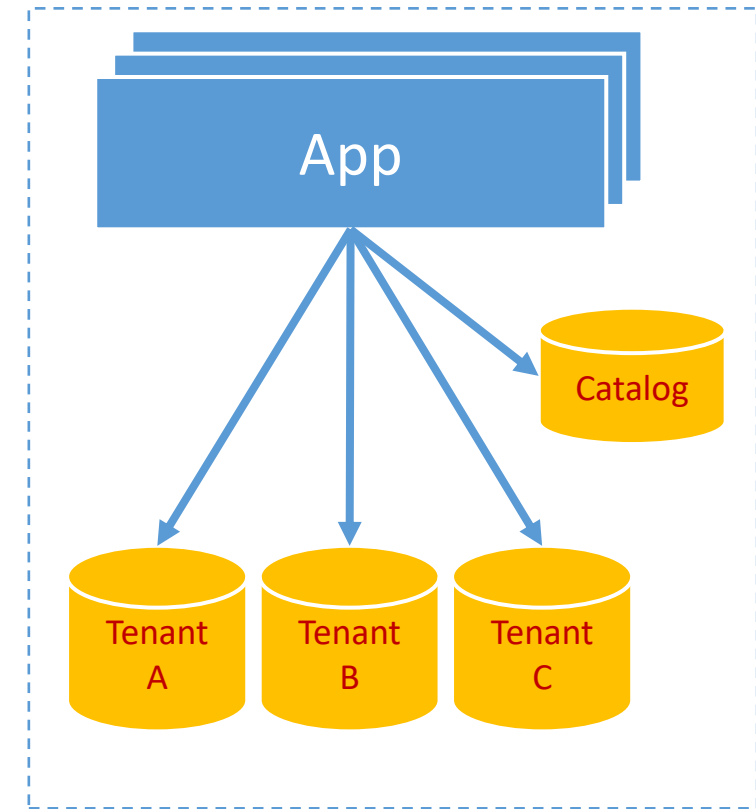


«Single multi-tenant database»

- Applicazione multi-tenant
 - Scale-out orizzontale indipendente
- Un database per tutti i tenant
- Minor isolamento
- Minor flessibilità

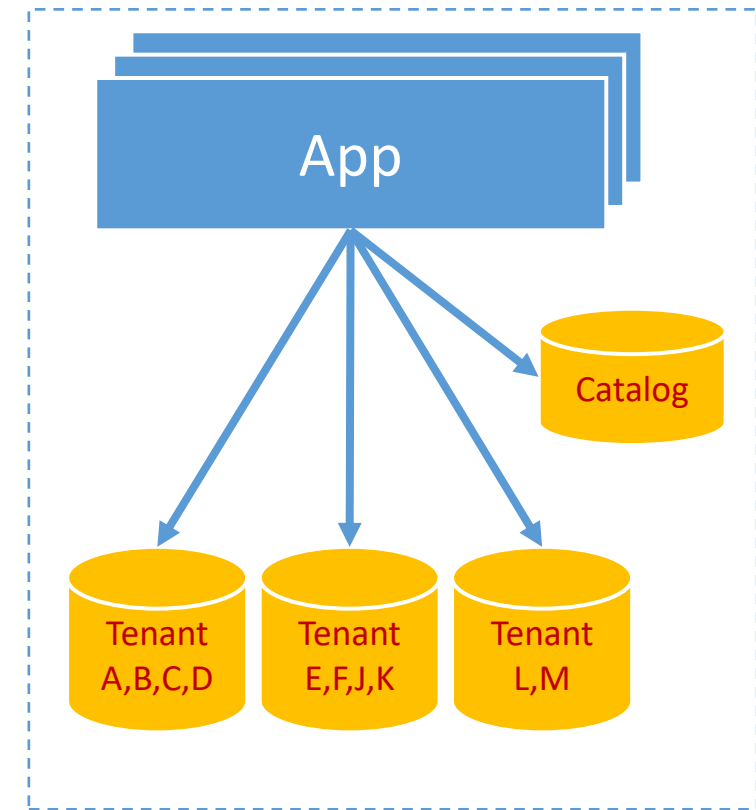


- Applicazione multi-tenant
 - Scale-out orizzontale indipendente
- Un database per ogni tenant
- Catalogo
 - DDR tenant->database
- Ottimo isolamento
- Ottima flessibilità



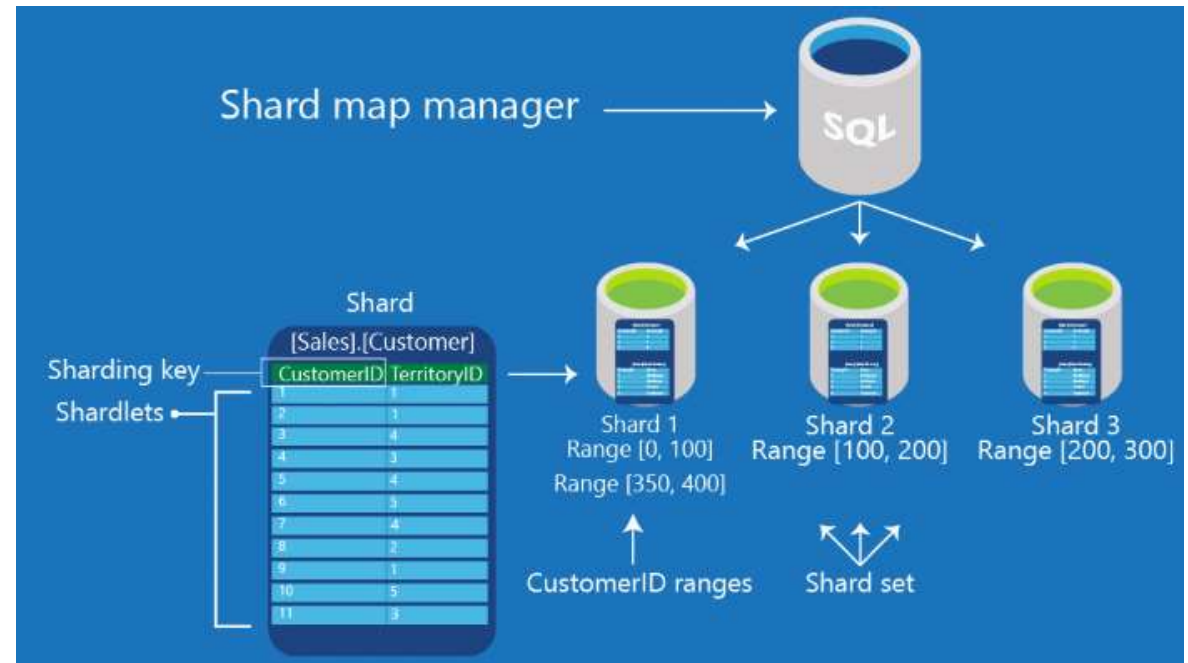
«Sharded multi-tenant databases»

- Applicazione multi-tenant
 - Scale-out orizzontale indipendente
- Più database per più tenant
- Catalogo
 - DDR tenant->database->shard-key
- Pessimo isolamento
- Pessima flessibilità



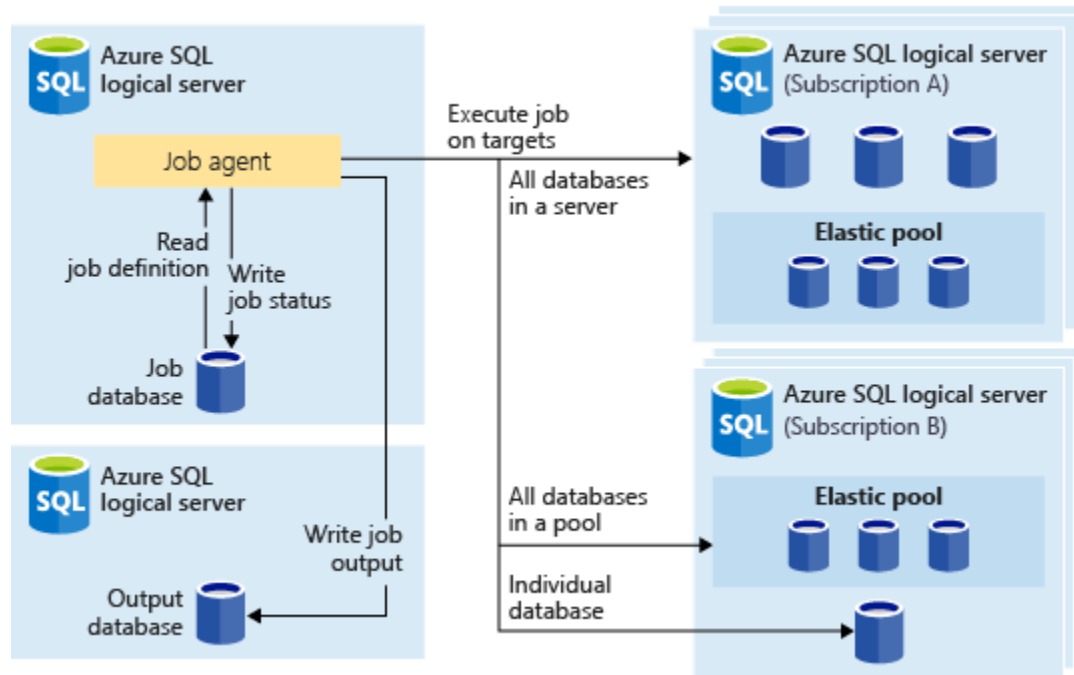
- «Tenant-per-schema»
- Colonna con «sharding key»
 - «Entity Framework Global Filters»
 - «Elastic Database Tools»
 - Supporto per «Entity Framework» e «Dapper»
- Sicuramente altre soluzioni, importante padroneggiare i concetti/pattern
- Ragionare in termini «web scale»...
 - non 100 o 1.000 ma 10.000, 100.000, 1 milione...
 - investire subito in «devops», farlo dopo costa molto...

- Da usare o prendere come modello
 - **Elastic Database client library**
 - Elastic Database split-merge tool
 - Elastic Database jobs
 - Elastic Database query (preview)
 - Elastic Transactions
- Repository Github
 - Client Library & esempi
 - Rilasci ripresi a Ottobre 2023
 - <https://github.com/Azure/elastic-db-tools>
- Migrazione da [SQL Azure Federation](#)...
- **Anche per database-per-tenant**

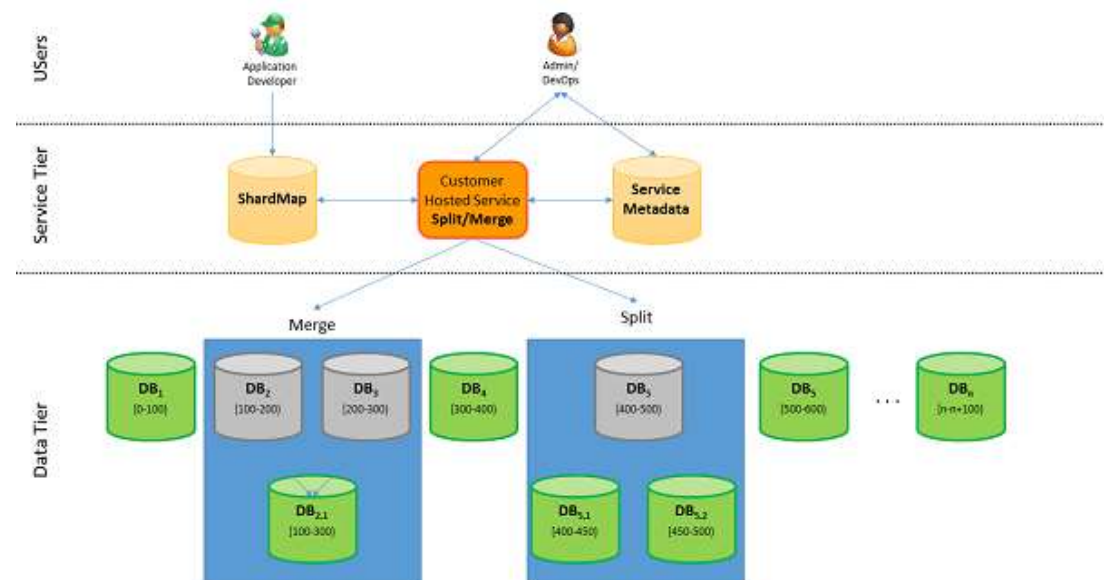


<https://learn.microsoft.com/en-us/azure/azure-sql/database/elastic-scale-introduction>

Elastic Database split-merge tool



Elastic Database jobs (preview)

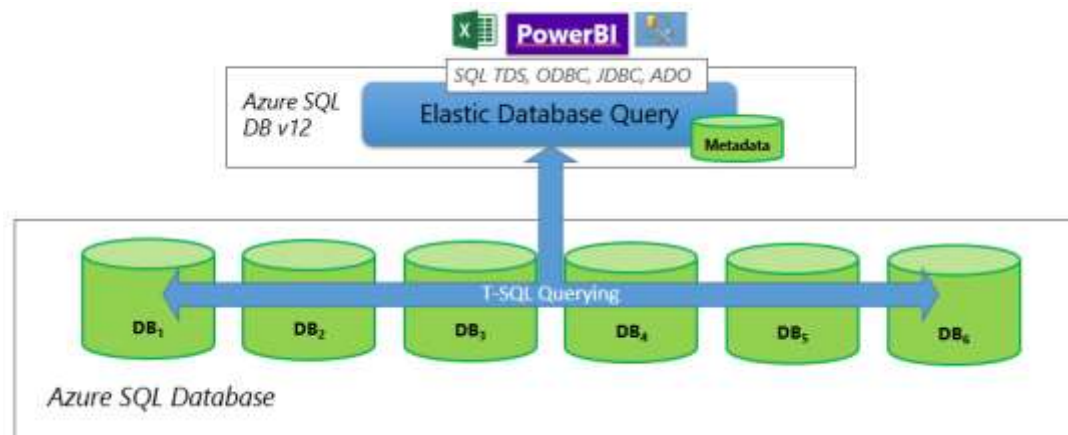


<https://learn.microsoft.com/en-us/azure/azure-sql/database/elastic-jobs-overview>

<https://learn.microsoft.com/en-us/azure/azure-sql/database/elastic-scale-overview-split-and-merge>

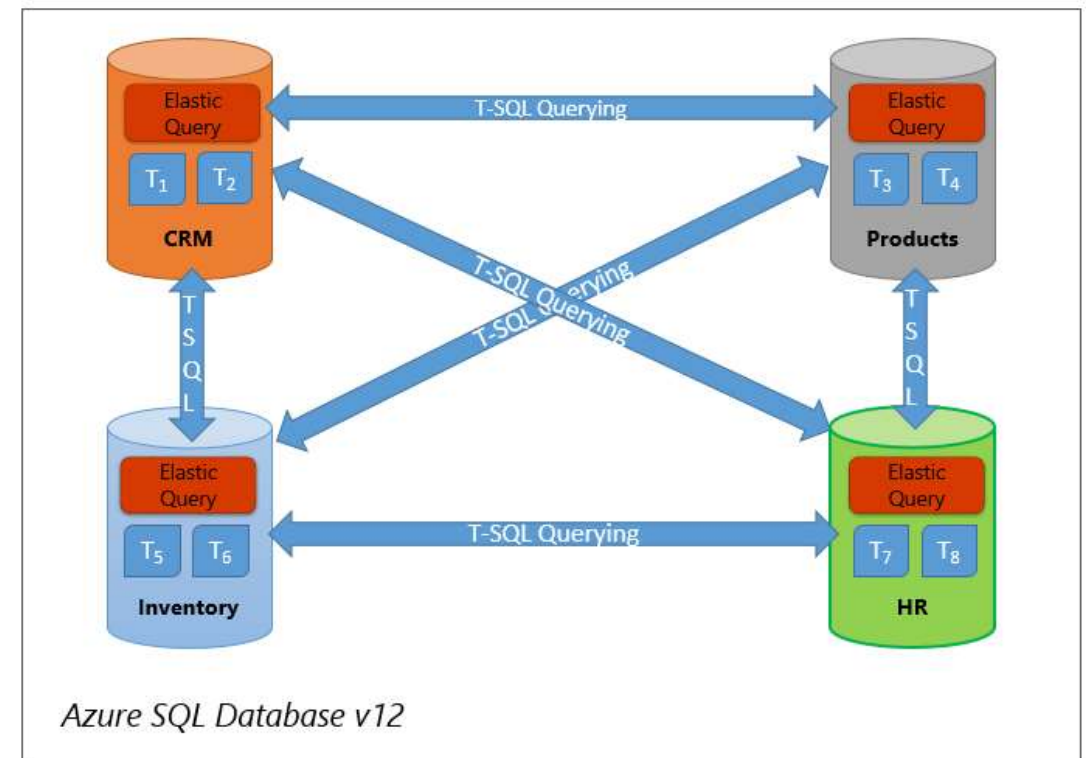
Elastic Database query (preview)

Cross-database (polybase/external tables)



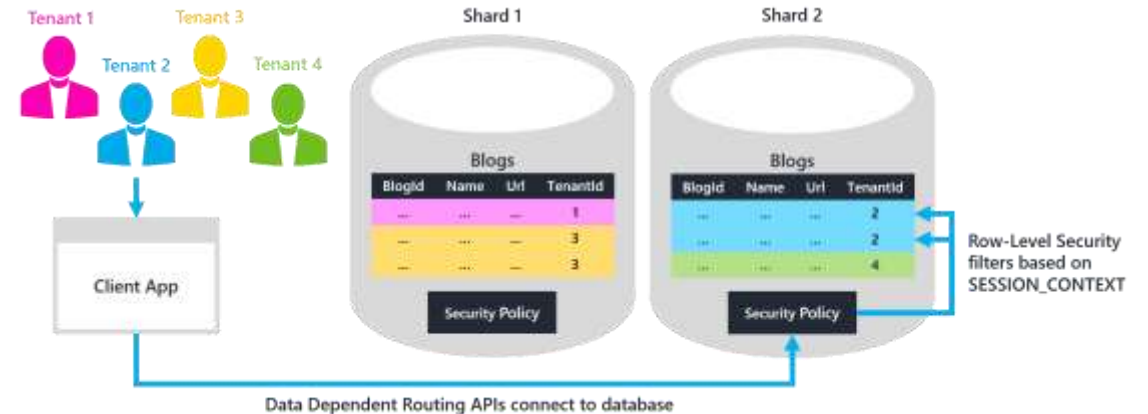
<https://learn.microsoft.com/en-us/azure/azure-sql/database/elastic-query-overview>

Cross-shard UNION ALL



Row Level Security (RLS)

- Modifica schema tabelle
 - Aggiunta colonna «*sharding key*»
 - **DEFAULT** con **SESSION_CONTEXT**
- RLS Policy
 - Funzione «table-valued»
 - applicata a più tabelle
 - predicato su **SESSION_CONTEXT**
 - predicato di filtro/blocco
- Alla connessione
 - (tramite *shard map manager*)
 - «*sharding key*» -> **SESSION_CONTEXT**
 - idealmente, trasparente per il resto...



Prestazioni

- Indici «sharding key» come prima colonna -> sbilanciamento istogramma per selettività
- Lock escalation ~5000 lock
- Contesa strutture condivise a livello database

Sicurezza

- Rilascio errato query senza predicato su «sharding key»
- Divulgazione a seguito di bug (es. dati cache, indici corrotti)
- Clienti richiedono isolamento («compliance»)

Continuità

- Rilascio errato impatta più tenant
- Restore di un singolo tenant (impatto su altri o procedura complessa)

Gestione

- «Sharded Data» e dati «Reference» da replicare (es. CAP, Paesi, valute ecc.)
- Rilasci scaglionati
- Difficile/impossibile gestire versioni diverse e/o variazioni dello schema

Flessibilità

- Difficile gestire risorse per tenant (e fornire livelli di servizio personalizzati)
- Difficile gestire personalizzazioni

Pattern dominante: Database-per-tenant

- Ottimo isolamento: prestazioni e sicurezza
- Semplicità implementazione, gestione e manutenzione
- Flessibilità
- La maggior parte dei SQL Database in Azure usano già questo pattern

Azure PaaS

Virtual Machine in Azure

- Ambienti Windows/Linux
- Marketplace con immagini per i più diffusi RDBMS
- [Azure Dedicated Host](#)

Diversi servizi per «Container»

- [Azure Kubernetes Service \(AKS\)](#)
- [Azure Red Hat OpenShift](#)
- [Azure Container Instances](#)

Altri scenari

- [Azure VMware Solution](#)

Azure IaaS

SQL Server
engine

- **Azure SQL Database**
- **Traditional/Hyperscale**
- **Managed Instance**
- Azure Synapse Analytics
- Azure SQL Edge

OSS
Engines

- Azure Database for PostgreSQL
- Azure Database for MySQL
- Azure Database for MariaDB

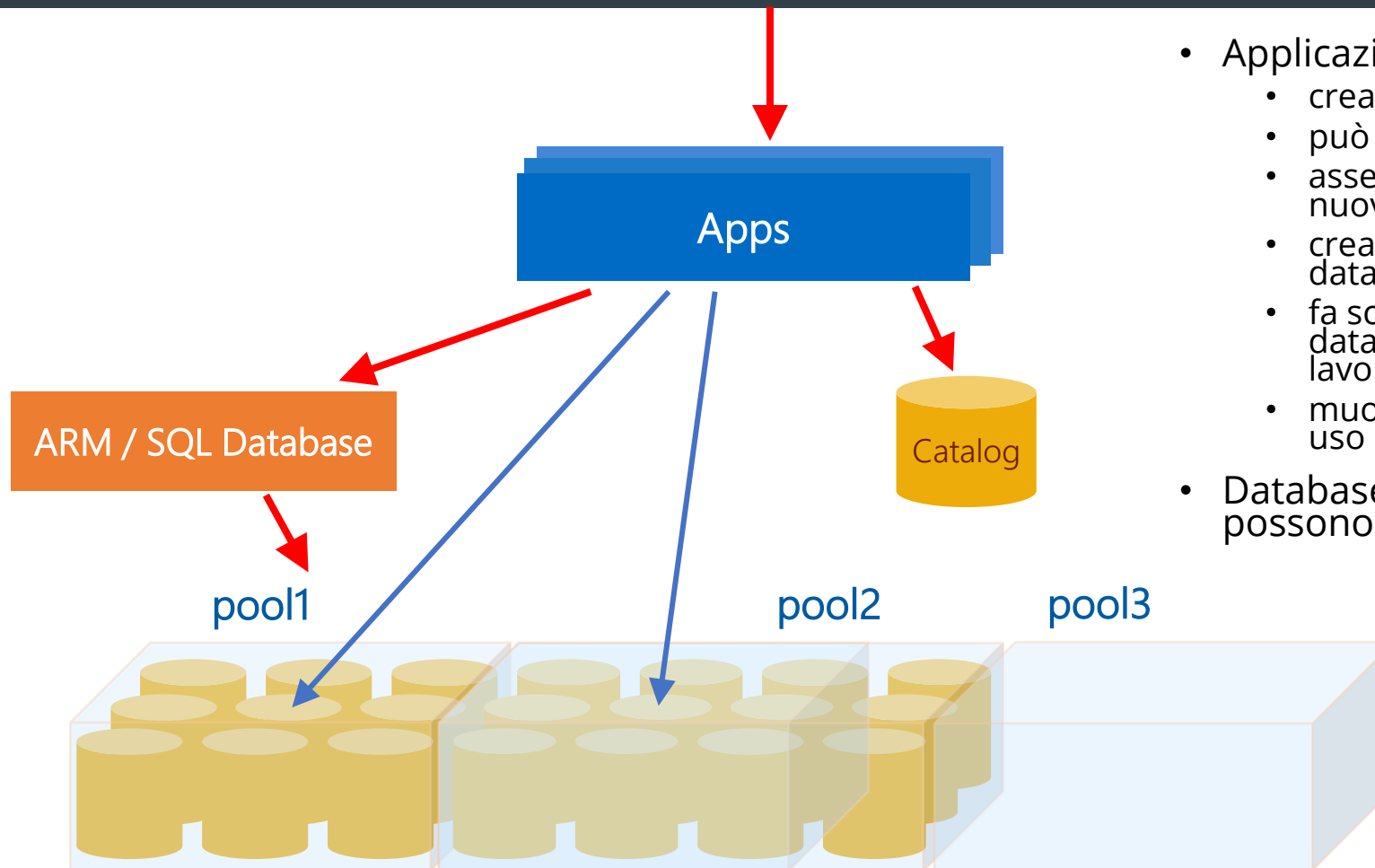
- Numero massimo di database
 - Virtual Machine: **32768** per istanza... **qualche migliaio** più realistico
 - [Managed Instance](#): **100** per istanza
 - [Managed Instance Pool](#) (preview): **500** per pool
 - **50** per istanze **2 vCores**, **100** per istanze **4+ vcores**
- Problemi comuni
 - Contesa buffer cache/procedure cache/altro a livello di istanza
 - Controllo integrità, manutenzione indici
- Problemi aggiuntivi Virtual Machine
 - Installazione e manutenzione software
 - Implementazione e manutenzione backup, alta disponibilità e disaster recovery
 - Es. scaglionamento migliaia di backup? Numero di Availability Group?

- **«Single database»**
 - Risorse assegnate al singolo database
 - «Serverless»: Min/Max vCore in base a SLO con auto pausa (costo zero CPU & RAM)
- **«Elastic Pool»**
 - Pool di risorse condivise da più database, «Auto-Scale» entro range definiti
 - Database aggiunti/rimossi a caldo
- «Hyperscale»
 - Engine specializzato, disponibile «Single», «Serverless» e «Elastic Pool (preview)»
- Caratteristiche comuni
 - «Service Tier/Performance Level» definiscono limiti risorse, HA e DR (inclusi!)
 - «Scale up» e «Scale down» senza interruzione del servizio

«Elastic Pool» migliore opzione per SaaS

- Scalabilità verticale
 - Modello a vCore: da 2 a 128 vCores
- Scalabilità orizzontale
 - General Purpose/Standard: **500** database
 - Business Critical/Premium: **100** database
 - Hyperscale (preview): **25** database
- Attenzione anche a tutti gli altri limiti!
 - Massimo [5000](#) database per «*logical server*», consigliato massimo [1000-2000](#)
 - Default [20](#) «*logical server*» per sottoscrizione in una regione (massimo **250**)
 - Storage per pool, database, tempdb
 - vCore per «*logical server*»
 - ecc. ecc.

«SaaS Elastic Pools Provisioning»



- Applicazione
 - crea il «Pool»
 - può creare database in anticipo
 - assegna database a tenant nuovi
 - crea in anticipo «Pool» e database
 - fa scalare «Pool» per gestire più database/maggior carico di lavoro
 - muove database per bilanciare uso
- Database con troppo carico, possono essere isolati

Azure Database Fleet Manager



<https://techcommunity.microsoft.com/t5/azure-sql-blog/introducing-azure-database-fleet-manager/ba-p/3979576>



<https://techcommunity.microsoft.com/t5/azure-sql-blog/introducing-azure-database-fleet-manager/ba-p/3979576>

- Costo minimo «database-per-tenant» con «Elastic Pool»
 - Troppo alto per alcuni scenari...
 - VM Standard/Enterprise Edition + VM Express Edition alternativa ma...
 - ... ricordate cosa c'è nel costo di un PaaS, es. gestione, backup, HA/DR
- Investire in «devops»
 - Database template per «deployment»
 - Schedulazione avanzata (es. per gruppi logici)
 - Gestione schema e manutenzione indici su larga scala
 - Politiche di archiviazione dati e automazione
- Framework, pattern, tool
 - Devono esservi di aiuto, non dovete lavorare per loro...

- Link contestuali nelle slide
- «Tenancy models for a multitenant solution»
 - <https://learn.microsoft.com/en-us/azure/architecture/guide/multitenant/considerations/tenancy-models>
- «Multitenancy and Azure SQL Database»
 - <https://learn.microsoft.com/en-us/azure/architecture/guide/multitenant/service/sql-database>
- «Multi-tenant SaaS database tenancy patterns»
 - <https://learn.microsoft.com/en-us/azure/azure-sql/database/saas-tenancy-app-design-patterns>
- «Scaling out with Azure SQL Database»
 - <https://learn.microsoft.com/en-us/azure/azure-sql/database/elastic-scale-introduction>
- SQL Server PaaS (Data Saturday #37 Parma 2023)
 - <https://vimeo.com/ugiss/sqlserverpaas>