

Ruminations on Multi-Tenant Databases

Dean Jacobs

Stefan Aulbach

Technische Universität München

Software as a Service

- ◆ Service provider hosts an application that multiple customers access over the Internet
 - sales, marketing, support, HR, payroll, planning, manufacturing, inventory, financials, purchasing
- ◆ Leverage economy of scale to reduce the total cost of ownership of the application
 - Capital expenditures – hardware, software
 - Operational expenditures – bandwidth, personnel
- ◆ Particularly appealing for small- to medium-sized businesses that don't have a complex data center

Multi-Tenancy

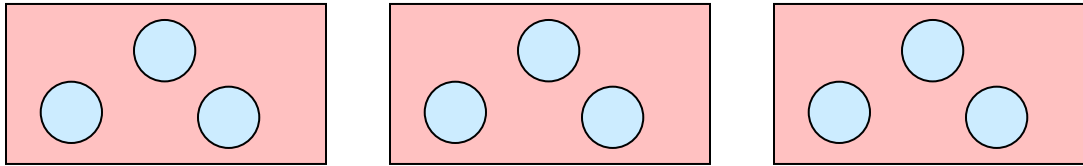
- ◆ Consolidate multiple businesses (tenants) onto the same operational system
- ◆ Pool resources to improve their utilization
 - Avoid provisioning each tenant for their maximum load
 - Breaks down isolation: weakens security, increases resource contention, interferes with optimizations
- ◆ Provide a tenant-aware administrative framework to improve management efficiency
 - Manage **farms** of individual multi-tenant servers
 - Support bulk operations such as rolling upgrade
 - Support tenant migration within and across farms

Multi-Tenant Databases

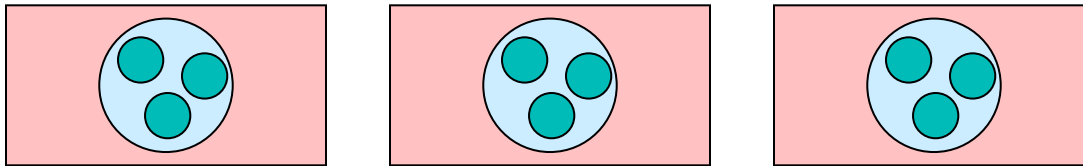
- ◆ Assume the application has a **base schema** that may be extended by each tenant
 - New columns for existing tables and new tables
 - Common for enterprise applications like CRM and ERP
- ◆ Pool database resources
 - Processes, memory, connections, prepared statements
 - Trade-offs against isolation
- ◆ Provide a tenant-aware administrative framework
 - Manage farms of individual multi-tenant databases
 - Support DML and DDL operations across tenants
 - Support tenant migration between databases

Implementation Options

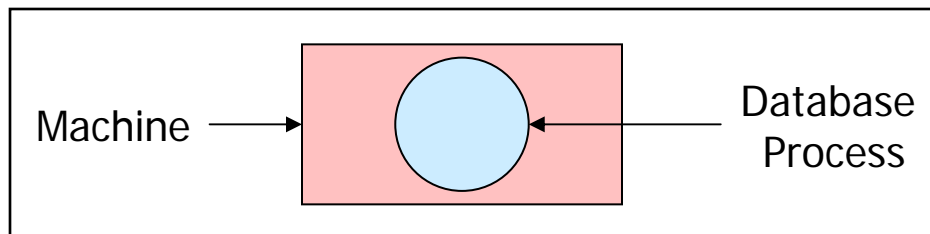
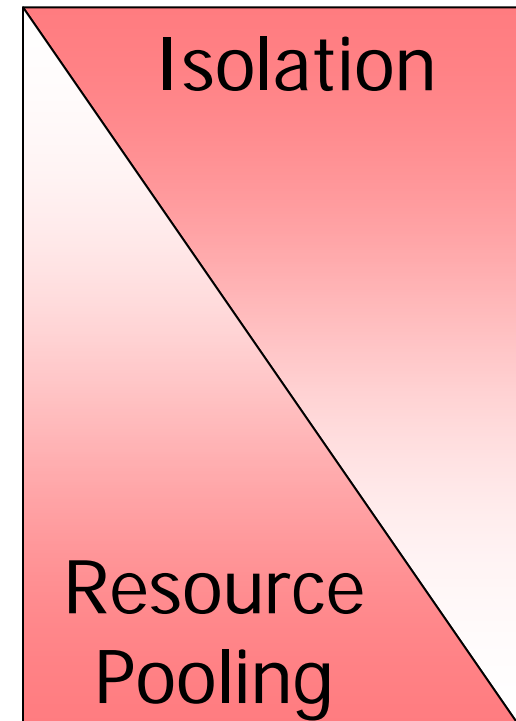
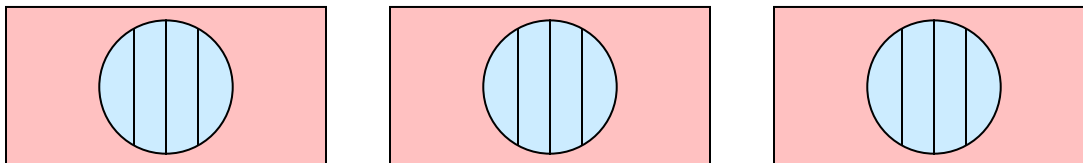
◆ Shared Machine



◆ Shared Process



◆ Shared Table



Shared Machine

PostgreSQL	MaxDB	COTS 1	COTS 2	COTS 3
55 MB	80 MB	171 MB	74 MB	273 MB

Memory requirements for a database with
one empty CRM schema instance

- ◆ Cannot scale beyond tens of tenants per server
- ◆ Appropriate for applications with a smaller number of larger tenants, e.g., for banking

Shared Process

PostgresSQL	MaxDB	COTS 1	COTS 2	COTS 3
79 MB *	80 MB	616 MB *	2061 MB	359 MB

Memory requirements for a database with
10,000 empty CRM schema instances

* extrapolated

- ◆ Should scale up to thousands of tenants
- ◆ If each tenant gets their own table space then migration entails simply moving files
- ◆ Connection pooling is possible, but then tenant identity must be managed by the application

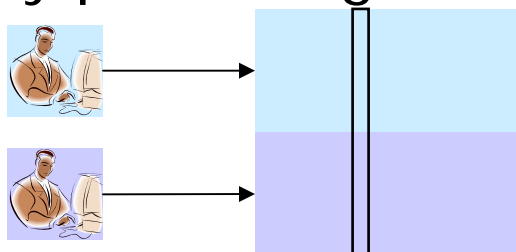
Shared Table

- ◆ Data from many tenants in the same tables
 - Add a tenant id column
 - Tenant queries must fix the value for this column
- ◆ Extend the base schema using generic columns
 - May be varchar or a mix of types
 - The database must compactly represent sparse tables

TenId	Account	Name	...	Val0	Val1	...	Val100
1041	0021	Acme		1/3/95	----		----
1041	0029	Ball		3/7/72	----		----
1053	0016	Gump		red	35		----
1053	0049	Wonk		blue	18		----

Shared Table

- ◆ The Good News - Everything is pooled
 - Processes, memory, connections, prepared statements
 - Easy DML and DDL operations across tenants
 - Add, remove, and extend tenants with DML (not DDL)
- ◆ The Bad News - Isolation is very weak
 - Irrelevant data infects query processing
 - Optimization Statistics
 - Table Scans
 - Data locality
 - No indexes or integrity constraints on generic columns
 - Migration requires querying the operational system



Possible Improvements

- ◆ Shared Process – increase resource pooling
 - Keep one copy of the meta-data about the base schema
 - Allow prepared statements over table names
 - Allow queries over table names
 - Dynamically set the principal for a database connection
- ◆ Shared Table – increase isolation
 - Take the tenant ID into account in query optimization, indexing, and data placement
 - Could be done from inside the database or from outside in a SQL transformation layer

Our Current Research

- ◆ Adapt an open source database for multi-tenancy
- ◆ Measure its scalability and performance using a benchmark for multi-tenant databases

