UNIVERSIDADE LUSÓFONA
de Humanidades e Tecnologias
Humani nihil alienum

DEPARTAMENTO DE COMUNICAÇÃO, ARTES
E TECNOLOGIAS DA INFORMAÇÃO

INFORMÁTICA DE GESTÃO

Trabalho Final de Curso

Ano lectivo 2009/2010 – 2ºSemestre

Refactoring de bases de dados para modelos cloud

Aluno: Luís Manuel Tavares Teixeira
Número: a20073892

Professora Orientadora: Inês Oliveira
Co-Orientador: João Martins
Apoio Tecnológico: Raúl Ribeiro

Lisboa, 15 de Outubro de 2010

Agradecimentos

Tenho muito a agradecer, principalmente aos professores que me ensinaram, aos colegas com os quais debati, à família a que expressei os meus sentimentos e as minhas angústias, aos amigos que se disponibilizaram para me ouvir e expressar as minhas escorregadelas e as minhas conquistas.

Um agradecimento especial à professora orientadora Draª Inês Oliveira pelo seu contributo e cooperação fundamental para a realização deste trabalho e aos colaboradores da |create|it|, principalmente ao meu co-orientador João Martins e ao Raúl Ribeiro, que não se cansaram em me apoiar e explorar os meus conhecimentos no sentido de extrair do meu esforço, o meu melhor neste projecto.

Todo este meio envolvente provou ser de pessoas da minha confiança e é com grande alegria que os lembro neste momento.

Índice

Índice Figuras

Tabela de abreviaturas e acrónimos

ACID – Atomicity, consistency, isolation, durability. Propriedades de Transacções.

CRUD - Create/Read/Update/Delete. Operações efectuadas sobre bases de dados ou outros repositórios de informação

TI – Tecnologias de Informação

NIST - National Institute of Standards and Technology

SaaS - Software As A Service

IaaS - Infrastructure as a Service

PaaS - Platform as a Service

CapEx - Despesas de capital

OpEx - Despesas operacionais

DaaS - Base de dados-as-a-Service

IP – Internet Protocol

SGBD - Sistema Gestor de Base de Dados

CAP - Teorema de PAC[1] da Brewer

HTTP - HyperText Transport Protocol

SQL - Structured Query Language

ISACA - Information Systems Audit and Control Association

[1] Teorema que define os requisitos de uma aplicação distribuída (ver Capítulo 3).

Resumo

O modelo *Cloud Computing* é um modelo que disponibiliza via internet o acesso a um conjunto partilhado de recursos de computação configurável.

Este modelo está pronto para proporcionar muitos benefícios e segurança de informação. Contudo antes de escolher este tipo de serviços tem que se ter em conta vários aspectos, como os riscos e as preocupações de segurança. Como acontece com qualquer tecnologia emergente, o *Cloud Computing* oferece a promessa de elevada recompensa em termos de contenção de custos, recursos, agilidade e velocidade de aprovisionamento.

O projecto tem como principal objectivo apoiar a migração para *cloud* de sistemas actualmente desenvolvidos com abordagens tradicionais. Tal como o seu título indica, ”Refactorização de bases de dados para modelos cloud”, pretende-se alterar o desenho de uma base de dados sem alterar o seu comportamento original. O método utilizado para este efeito foi o da normalização e desnormalização. A normalização é a transformação de um conjunto de relações iniciais em formas ditas Normais e a desnormalização é o processo inverso dos passos seguidos para se atingir uma forma normal. A desnormalização permite melhorar as operações de pesquisas e eliminar JOINs (que o *Cloud Computing* não suporta).

A base tecnológica de armazenamento utilizada para suporte de dados na *cloud* foi a *Windows Azure storage table*. Estes serviços fornecem armazenamento estruturado na *Cloud*. É importante salientar que as tabelas de *Windows Azure* não são tabelas de base de dados relacionais, como as das bases de dados tradicionais. O serviço das tabelas é altamente flexível, e baseado num modelo simples de entidades e propriedades. Em termos simples, as tabelas contêm entidades, e as entidades têm um conjunto de propriedades.

Palavras-Chave: *Cloud Computing*, bases de dados, refactorização, desnormalização, *Windows Azure*.

Abstract

The cloud computing model is a model that provides internet access to a shared set of configurable computing resources. This model is ready to provide many benefits, and information security. But before choosing this type of services, we have to take into account aspects such as risks and security concerns. As any emerging technology, cloud computing offers the promise of high rewards in terms of cost containment, resources, flexibility, speed of supply.

The project's main objective is to support the migration to cloud, systems currently developed with traditional approaches. As its title indicates, Refactorização de bases de dados para modelos cloud" intended to change the design of a database without changing its original behavior. The method used for this purpose was the normalization and denormalization. Normalization is the transformation of an initial set of relationships in ways those standards and denormalization is the reverse process of the steps taken to achieve a normal way.Denormalization can enhance the research operations and eliminate JOINs (which does not support the Computing Cloud).

The base technology that is used for support data storage is Windows Azure storage table. These services provide structured storage in the cloud. It is important to note that the tables of Windows Azure are not the base tables of relational databases, like those of traditional databases. The table service is highly flexible, and based on a simple model of entities and properties. In simple terms, the tables contain entities and the entities have a set of properties.

Keywords: *Cloud Computing*, data bases, refactoring, desnormalization, *Windows Azure*.

Introdução

Escolhi abraçar este projecto "Refactoring de bases de dados para modelos cloud" porque acho que *Cloud Computing* será o futuro próximo a nível de tecnologia de informação. Trata-se de um modelo que disponibiliza via internet o acesso a um conjunto partilhado de recursos de computação configurável (por exemplo, redes, servidores, armazenamento, aplicações e serviços), que podem ser rapidamente aprovisionados e lançados com um mínimo de esforço de gestão ou interacção. Este pode ser composto por vários modelos de serviço. Neste sentido foi necessário aprofundar conhecimentos em áreas que não constam do programa curricular do curso.

O projecto tem como principal objectivo apoiar a migração para *cloud* de sistemas actualmente desenvolvidos com abordagens tradicionais. Tal como o seu título indica, "Refactorização de bases de dados para modelos cloud", pretende-se alterar o desenho de uma base de dados sem alterar o seu comportamento original. O método utilizado para este efeito foi o da normalização e desnormalização. A normalização é a transformação de um conjunto de relações iniciais em formas ditas Normais e a desnormalização é o processo inverso dos passos seguidos para se atingir uma forma normal. A desnormalização permite melhorar as operações de pesquisas e eliminar JOINs (que o *Cloud Computing* não suporta).

A base tecnológica de armazenamento utilizada para suporte de dados na *cloud* foi a *Windows Azure storage table*. Estes serviços fornecem armazenamento estruturado na *Cloud*. É importante salientar que as tabelas de *Windows Azure* não são tabelas de base de dados relacionais, como as das bases de dados tradicionais. O serviço das tabelas é altamente flexível, e baseado num modelo simples de entidades e propriedades. Em termos simples, as tabelas contêm entidades, e as entidades têm um conjunto de propriedades.

No primeiro capítulo consta o enquadramento teórico de refactorização de bases de dados e *Cloud Computing*. No segundo capítulo descreve-se o método utilizado para a refactorização de bases de dados para *Cloud Computing*, a normalização e desnormalização de base de dados. No terceiro capítulo apresenta-se um segundo método que também pode ser utilizado para a

referida refactorização, mas que não foi posto em prática. Este capítulo é ainda constituído pelo teorema de PAC de Brewer, sistemas pioneiras de armazenamento de dados e o Windows Azure Storage Table. O quarto capítulo refere-se aos resultados obtidos e respectiva discussão. Por fim apresentam-se as conclusões do trabalho efectuado, mencionando-se as perspectivas futuras sobre este projecto.

Capítulo 1

1 - Enquadramento teórico

1.1 - Refactorização de Base de dados

De acordo com o site (http://www.agiledata.org), refatorização de base de dados consiste em efectuar alterações no seu esquema que o melhorem sem alterar a sua semântica ou o seu comportamento externo. Este processo permite de forma rápida e eficaz melhorar a sua capacidade de suportar novas necessidades dos clientes, tais como pesquisas e outras operações, servindo de apoio ao desenvolvimento de software evolutivo. Além disso possibilita ainda corrigir problemas existentes em esquemas de base de dados legados.

1.2 - *Cloud Computing*

O National Institute of Standards and Tecnology (NIST) e o Cloud Security Alliance. definem *Cloud Computing* como um modelo que disponibiliza via internet o acesso a um conjunto partilhado de recursos de computação configurável (por exemplo, redes, servidores, armazenamento, aplicações e serviços), que podem ser rapidamente aprovisionados e lançados com um mínimo de esforço de gestão ou interacção. Este pode ser composto pelos seguintes modelos de serviço:

- **Infrastructure as a Service (IaaS)** - Disponibilidade de infra-estruturas a correr na *Cloud* onde se podem ser instalados os Sistemas operativos que o cliente pretende (*Amazon EC2*)
- **Platform as a Service (PaaS) –** é uma plataforma que permite instalar aplicação desenvolvida a medida pelo cliente (Azure).
- **Software as a Service (SaaS)** – é uma plataforma que permite o utilizador aceder e utilizar software instalado na cloud, como por exemplo um navegador *Web* (por exemplo, web-based e-mail ou SalesForce.com).

Estes modelos de serviço podem ser implantados num dos cinco modelos de implantação descritos na tabela seguinte. Os riscos e benefícios globais irão variar de modelo para modelo, sendo importante notar que quando se consideram os diferentes tipos de serviços e modelos de implantação, as empresas devem considerar os riscos que os acompanham.

Modelos de Implantação de *Cloud Computing*		
Modelo de Implantação	Descrição da infra-estrutura de nuvem	Nível de Risco
Nuvem Privada	• Operado exclusivamente para uma organização • Pode ser gerida pela organização ou um terceiro • Podem existir nas instalações do cliente (*on-premise*) ou noutra localização (*off-premise*)	• Serviços Cloud com mínimo risco • Pode não fornecer a escalabilidade e agilidade de serviços públicos de nuvem
Nuvem de Comunidade	• Partilhadas por diversas organizações • Suporte a uma comunidade específica que tem uma missão ou interesse comum. • Pode ser administrada por organizações ou terceiros • podem residir on/off-premise	• Mesmo que nuvens privadas, mas os dados podem ser armazenados com os dados dos concorrentes.

Nuvem Pública	• Colocada à disposição do público em geral ou de um grupo grande da indústria • Detida por um fornecedor de serviços na nuvem	• Mesmo que nuvens comunidade, mas os dados podem ser armazenados em locais desconhecidos e não podem ser facilmente recuperáveis.
Nuvem Híbrida	A composição de duas ou mais nuvens (Privado, público ou comunidade) que permanecem entidades únicas, mas são unidos por tecnologia padronizada ou proprietária que permite que a portabilidade dos dados e de aplicações	•Verifica-se o risco de agregação de fundir diferentes modelos de implantação • Classificação e rotulagem de dados será benéfica para o gestor de segurança para assegurar que os dados são atribuídos à tipo correcto de nuvem.

Tabela 1-Modelos de Implantação do *Cloud Computing*

Com a crescente e contínua procura de TI, muitas empresas estão a examinar *Cloud Computing* como uma opção real para o futuro dos seus negócios (ISACA, 2006). As características que lhe associam são atraentes tais como a maior agilidade, elasticidade, capacidade de armazenamento e redundância para gestão dos activos (*assets*). Movendo-se serviços de TI para a *Cloud*, as empresas podem tirar proveito do uso de serviços num modelo *on-demand*. Além disso, é necessário um menor investimento inicial, o que permite às empresas maior flexibilidade com novos serviços de TI.

Este novo paradigma oferece assim serviços atraentes para qualquer negócio que deseje aprimorar os recursos de TI. No entanto, convém notar que, juntamente com os benefícios advêm riscos e preocupações de segurança que devem ser tidos em consideração tal com consta na tabela 1

1.3 - Benefícios *Cloud Computing* (ISACA, 2006)

Os benefícios principais do *Cloud Coumputing* são:

Especialização. Verifica-se a especialização de uma série de conhecimentos necessários para a operação de sistemas deste tipo, como sejam a segurança, extensibilidade, manutenção

(correcções e actualizações), salvaguarda de dados (backups), entre outros. Num modelo tradicional, cada organização cliente tem que deter colaboradores com estes conhecimentos. Com *Cloud Computing* estas capacidades ficam concentradas em especialistas do fornecedor, que são partilhadas por todos os clientes.

Economias de Escala. A plataforma ideal é muito cara de montar. Os servidores, o armazenamento de dados, as redes de equipamento, os custos com energia, etc., podem resultar num enorme custo a suportar por um único produto ou projecto. Com o *Cloud computing* este investimento pode ser amortizado num grande número de projetos. Mesmo se um projecto falhar, este pode ser substituido por um ou mais novos projectos que continuam a amortização do investimento inicial.

Os beneficios apresentados acima reflectem-se ainda nos seguintes:

Redução de custos. Todos os recursos, incluindo equipamentos caro interconectante, servidores, pessoal, etc. são partilhados, contribuindo para reduzir os custos, especialmente para projectos e protótipos de pequena dimensão.

Desvio CapEx a OpEx. Possibilita que as empresas transfiram dinheiro de despesas de capital (CapEx) para as despesas operacionais (OpEx). Isto permite ao cliente focar-se no aumento do valor do seu negócio propriamente dito, em vez de preocupar-se com, por exemplo, a construção e manutenção da infraestrutura.

Agilidade. Permite uma maior rapidez na gestão de recursos. Quando um projecto é financiado e iniciado, é feito o aprovisionamento; quando o projeto termina o seu tempo de vida, basta finalizar o contrato com o fornecedor *Cloud*.

Escalabilidade. Com capacidade ilimitada, serviços na Cloud oferecem maior flexibilidade e escalabilidade para a evolução de necessidades de TI. Provisionamento e execução é feito sob *demand*, permitindo a picos de tráfego e reduzindo o tempo para implementar novos serviços.

Manutenção simplificada. Correcções e actualizações rápidas são comuns em toda a infraestrutura, como são os *backups*.

Larga escala: Permite desenvolver grandes protótipos e efectuar testes de carga muito mais facilmente. Por exemplo, pode-se facilmente criar 1.000 servidores em *Cloud* para testar carregar a sua aplicação, e libertá-los quando o teste é terminado. Este factor também se aplica a situações em que um número elevado de servidores é necessário durante um curto espaço de tempo, como sucede por exemplo nas noites de eleições.

Diversas plataformas de apoio: Permite suportar uma grande variedade de plataformas cliente, incluindo *browsers*, dispositivos móveis, entre outros. Possibilita assim chegar a uma base mais ampla de utilizadores.

Aprovação mais rápida de gestão. Como tem custos muito baixos, torna mais fácil a aprovação do processo e a gestão, mais acelerada, suportando a rapidez da inovação. Na verdade, os custos podem ser tão baixos, que colaboradores ou até particulares podem facilmente financiar-se e demonstrar os benefícios da sua solução, evitando a inércia organizacional.

Desenvolvimento rápido. Fornece muitos serviços de base, e o desenvolvimento pode ser feito com as formas e linguagens de programação tradicionais. Estes serviços, bem como *templates* e outras ferramentas, podem acelerar significativamente o ciclo de desenvolvimento.

1.4 - Riscos e problemas de segurança com *Cloud Computing* (ISACA, 2006)

Muitos dos riscos frequentemente associados com a *Cloud Computing* não são novos, e podem ser encontrados nas empresas hoje, nomeadamente:

- Empresas clientes precisam de ter especial cuidado na escolha de um fornecedor. A reputação, a história e a sustentabilidade devem ser factores a considerar.
- O fornecedor de *cloud* muitas vezes assume a responsabilidade de tratamento da informação, que é uma parte crítica do negócio. A não realização de acordos de níveis de serviço (SLA) pode afectar a confidencialidade, não só mas também a disponibilidade, afectando severamente as operações de negócios.

- A natureza dinâmica do *Cloud Computing* pode resultar em confusão quanto a onde a informação realmente reside. Quando é necessária a recuperação de informação, isso pode gerar atrasos.
- Acesso de terceiros às informações sensíveis cria um risco de comprometimento de informações confidenciais. Na *Cloud Computing*, isso pode representar uma ameaça significativa para garantir a protecção da propriedade intelectual (IP) e segredos comerciais.
- As nuvens públicas permitem que os sistemas tenham alta disponibilidade e níveis de serviço muitas vezes impossíveis de criar em redes privadas, sem incorrerem em despesas extraordinárias. A desvantagem desta disponibilidade é o potencial de mistura da informação com outros clientes de *Cloud*, incluindo os concorrentes.
- Devido à natureza dinâmica de *Cloud*, a informação não pode ser localizada imediatamente em caso de um desastre. A continuidade do negócio (*Business Continuity*) e planos de recuperação de desastre devem estar bem documentados e testados. O fornecedor de *Cloud* deve compreender o papel que ela desempenha em termos de suporte, de resposta a incidentes e recuperação. Os objectivos de tempo de recuperação devem ser indicados no contrato.

Capítulo 2

2 - Normalização e Desnormalização de Bases de Dados

A Normalização (Gonçalves, 2008) é a transformação de um conjunto de relações iniciais em formas ditas Normais. Designam-se por Formas Normais as propriedades das relações que garantem a não existência de alguma forma de redundância. Cada forma normal satisfaz um determinado critério e elimina um diferente tipo de redundância. O objectivo é obter relações que obedeçam aos critérios da forma normal de ordem mais elevada e, por conseguinte, que garantam a eliminação do maior número possível de fontes de redundância.

No caso específico das bases de dados, por exemplo, no caso de associações de cardinalidade de M:N a regra é explicitar sempre a Entidade Associativa correspondente logo a partir do nível lógico de modelação. Ou seja: requer introduzir uma tabela adicional de relação que relaciona as chaves das tuas tabelas associadas. Já no que refere a a associações de cardinalidadede 1:N não obriga a que seja explicitada como Entidade Associativa, mesmo que tenha atributos próprios. Ou seja: a chave da entidade da entidade do lado N é transferida para a tabela que realiza a entidade do lado 1.

A Desnormalização (Ligia, 2010), como o termo indica, é o processo inverso dos passos seguidos para se atingir uma forma normal. Normalmente, torna-se necessário violar certas regras de normalização para satisfazer os requisitos do mundo real de consultas específicas. As figuras mostram uma base de dados normalizada e o resultado da sua desnormalização.

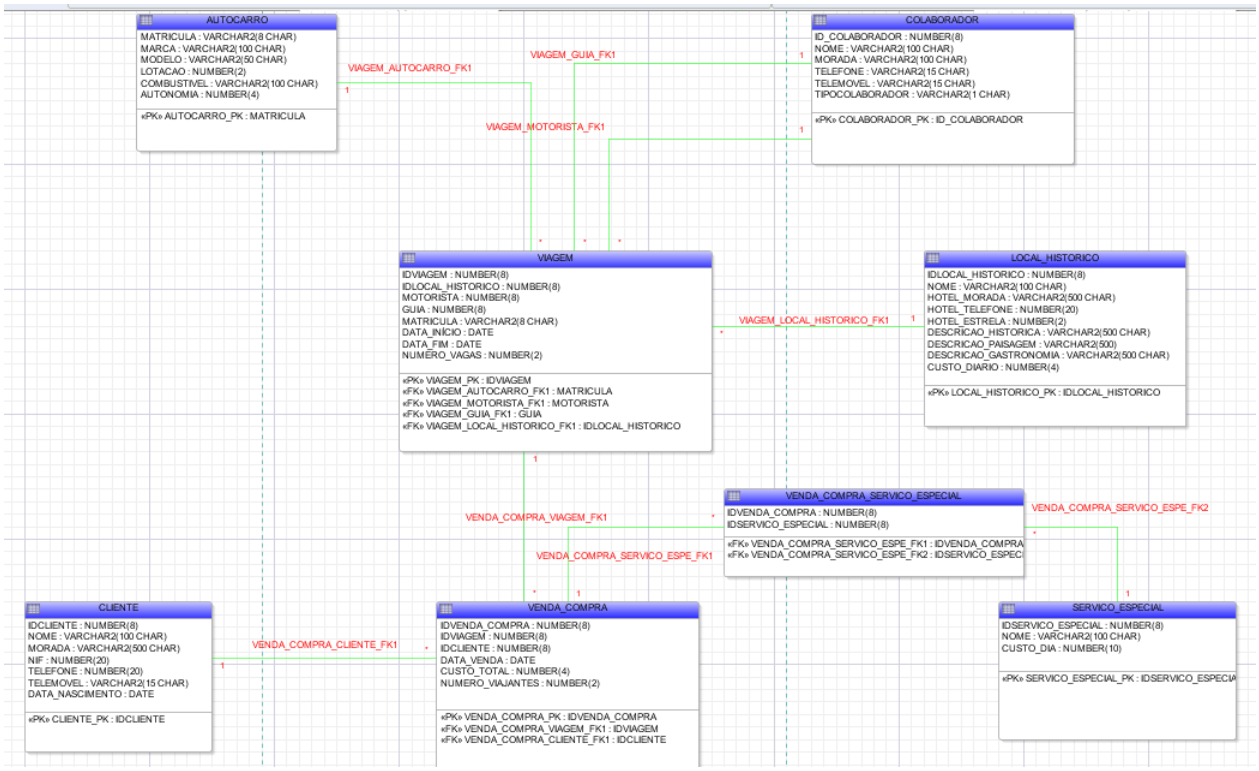

Figura 1 - Base de dados Normalizada

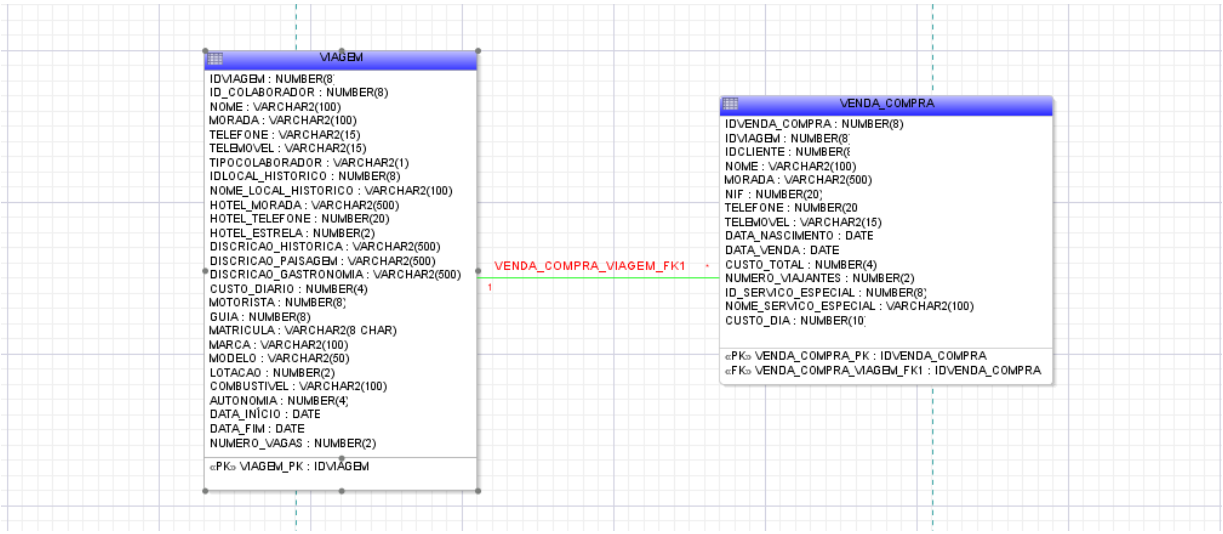

Figure 2 - Base de dados Desnormalizada

Como se pode ver, inicialmente existiam as tabelas de AUTOCARRO, COLABORADOR, VIAGEM, LOCAL_HISTORICO, CLIENTEe, VENDA_COMPRA, VENDA_COMPRA_SERVICO_ESPECIAL e SERVICO_ESPECIAL e depois de normalizada estas foram fundidas, ficando apenas duas tabelas a de VIAGEM e VENDA_COMPRA.

Este processo permite-nos adaptar a forma das tabelas para combinar conjuntos de dados numa mesma tabela, passando a ter duplicação, mas permitindo que as consultas sejam executadas de forma mais eficiente. Este conceito é compativel com um dos requisitos para armazenamento de dados em modelos *Cloud Computing*, em que a escalabilidade é essencial, mas só isto não basta.

Em modelos de dados completamente normalizados, a propensão é ter muitas entidades/tabelas e relacionamentos, para obter dados desejados. No entanto precisa-se de um número imenso de JOINs para obter os dados necessários para cada consulta realizada. Este processo e a sua implementação está correcto, isto é, enquadra-se no conceito de normalização. Na sua implementação física em base de dados, os JOINs tendem a resultar em sobrecarga no tempo de processamento quando se realizam consultas. Um JOIN requer cruzar índices, recuperar os dados das várias tabelas envolvidas, combinar estes dados e envir ao destinatário responsavel pela consulta. O caso acima referido é simples, mas quando estamos a falar de

milhões de linhas a serem varridas e a dezenas ou centenas de linhas a ser retornadas, esse processo torna-se bastante dispendioso.

Neste sentido torna-se necessário o conceito de Desnormalização. A desnormalização de bases de dados nestas situações, em que se pode criar uma entidade desnormalizada pode oferecer um benefício de desempenho, à custa da violação de uma das formas normais. Esta violação provoca redundância de dados, pois estamos a armazenar em mais de uma tabela, ou várias vezes numa mesma tabela, dados duplicados que de outra forma só existiram uma única vez na base de dados.

Capítulo 3

3 - Enquadramento teórico

3.1 - Teorema de PAC da Brewer

O teorema PAC de Brewer (Browne, 2009) diz que existem três principais requisitos a ter em conta quando se concebem e desenvolvem aplicações num ambiente distribuído, que são:

Consistência. Um serviço que é consistente opera totalmente ou não opera.

Disponibilidade. Significa apenas que o serviço está disponível (para opera e totalmente ou não operar).

Tolerância a partição. Gilbert & Lynch definen tolerância a partição como "Nenhum conjunto de falhas inferior à falha total da rede pode fazer com que o sistema responda correctamente".

3.2 – Sistemas Pioneiros de Armazenamento de Dados

Para responder à crescente necessidade de armazenamento de dados, começou-se muito cedo a procurar a melhor forma de armazenar os dados.

Neste processo tentaram-se criar infra-estruturas de armazenamento, tendo surgido vários sistemas como por exemplo o Linda Tuples, o Facebook Cassandra e a Amazon SimpleDB. Estes sistemas foram pioneiros nesta área, servindo de inspiração ao Windows Azure Storage Tables da Microsoft.

3.2.1 - Linda Tuples

Linda é um modelo de coordenação e comunicação entre vários processos paralelos, funcionando com os objectos da memória compartilhada, virtual ou associativa (http://en.wikipedia.org)

Tuples (http://en.wikipedia.org), é uma implementação do paradigma de memória associativa para a computação paralela/distribuída. Esta fornece um repositório de *tuples* que podem ser acedidos simultaneamente, um processo não precisa ter nenhuma noção de outros processos excepto para os tipos de *tuples* consumidos ou produzidos.

3.2.2 - Facebook Cassandra (Lakshman, 2008)

A Cassandra é conhecida por ser a base do armazenamento de dados utilizado no Facebook. Fiabilidade em larga escala é um desafio muito grande. Interrupções no serviço podem resultar em impacto negativo significativo. Daí a Cassandra pretende executar em cima de uma infra-estrutura de centenas de nós (possivelmente espalhados por *datacenters* diferentes). Nestas escalas, componentes pequenos e grandes falham continuamente, de maneira que a Cassandra gere o estado da persistência tendo em consideração essas falhas, e mantém a fiabilidade e escalabilidade dos sistemas de software que dependem destes serviços. Cassandra tem alcançado diversos objectivos - escalabilidade, alto desempenho, alta disponibilidade e aplicabilidade.

3.2.3 - Amazon SimpleDB

Amazon SimpleDB (Amazon, 2009) é um serviço *web* para execução de consultas sobre dados estruturados em tempo real. Este serviço funciona em estreita articulação com a Amazon Simple Storage Service (Amazon S3) e Amazon Elastic Compute Cloud (Amazon EC2),

oferecendo colectivamente, a capacidade de armazenar, processar e consultar conjuntos de dados na *Cloud*. Estes serviços foram projectados para tornar a computação em escala *web* mais fácil e mais rentável para os desenvolvedores.

Tradicionalmente este tipo de funcionalidade foi realizado com bases de dados relacional em *cluster*, o que requer um investimento inicial considerável, traz mais complexidade do que é normalmente necessário e muitas vezes exige um adminsitrador de bases de dados para manter e administrar. Em contraste, a Amazon SimpleDB é fácil de usar e oferece o núcleo de funcionalidade de uma base de dados: pesquisa em tempo real e em dados estruturados, sem a complexidade operacional. A Amazon SimpleDB não requer nenhum esquema, indexa automaticamente os seus dados e fornece uma API simples para o armazenamento e acesso. Isso elimina a carga administrativa de modelagem de dados, o índice de manutenção de ajuste de desempenho. Os programadores têm acesso a esta funcionalidade no ambiente privado de computação da Amazon, escalável instantaneamente e com pagamento somente pelo que se usa.

3.3 – Armazenamento de Dados em Windows Azure (*Storage Table*)

Os serviços de Windows Azure Table (Lerman, 2010) fornecem armazenamento estruturado na *Cloud*. Windows Azure Table não são tabelas de base de dados relacionais como nas base de dados tradicionais. O serviço da tabela é altamente flexível, tratando-se de um simples modelo de entidades e propriedades. Em termos simples: tabelas contêm entidades e entidades têm propriedades.

O serviço de tabela é concebido para grande extensibilidade e alta disponibilidade e suporta milhares de milhões de entidades e terabytes de dados. Está desenhado para suportar alto volume, mas menor estruturação de objectos. Não existe qualquer limitação no número de tabelas e entidades a criar neste serviço. Também não há limite relativamente à dimensão das tabelas da sua conta. A maior parte das situações do armazenamento de dados em base de dados relacionais usa várias tabelas, cada uma contendo um conjunto predefinido de uma ou mais colunas que são normalmente designados como chave primárias. As tabelas usam estas chaves para definir as relações entre si.

O Windows Azure permite o armazenamento de dados estruturado de duas formas: SQL Azure e o Windows Azure Storage Tables. O primeiro é uma base de dados relacional e alinha estreitamente com o SQL Server. Tem tabelas com esquema definido, chaves, relacionamentos e outros condicionalismos, e pode-se conectar a ele usando uma sequência de conexão, tal como faz com o SQL Server e outros bases de dados.

3.3.1 - Armazenamento de dados para recuperação e persistência eficientes

O Windows Azure Storage Tables fornece de origem a possibilidade de armazenar enormes quantidades de dados, permitindo um acesso eficaz e persistente. Os serviços disponíveis simplificam o armazenamento, limitações, pontos de vista, índices, relações e *stored procedures*. A Windows Azure Table usa chaves que permitem consultas eficientes e pode usar uma PartitionKey para compensação de carga, sempre que for necessário distribuir os dados de uma tabela por múltiplos servidores. Uma tabela não tem um modelo (*schema*) específico, é antes um repositório de linhas/registos com uma quantidade variável de colunas. Podemos ter uma tabela que armazena um tipo específico, mas também podemos ter linhas/registos com diferentes estruturas numa única tabela, como mostra a Figura 4. Este modelo alinha particularmente bem com o modelo de *Tuples* já descrito: cada linha/registo numa tabela é um *tuple*.

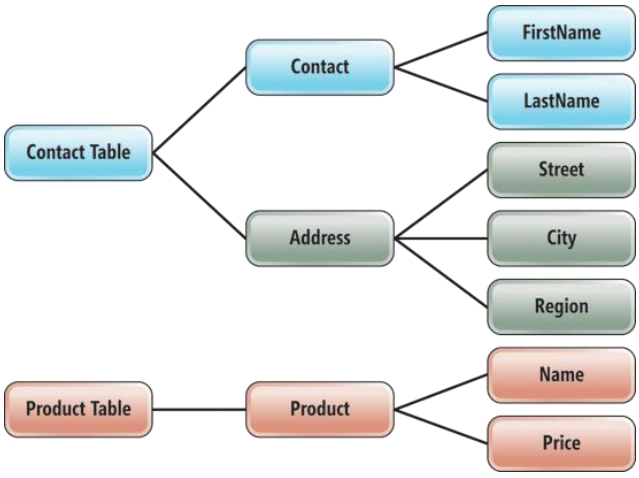

Figura 3 – Uma única tabela em Windows Azure pode conter linhas/registos que representam de forma semelhante ou entidades diferentes.

3.3.2 - O domínio das classes em Windows Azure

No desenvolvimento típico de uma base de dados começa com a sua criação, em seguida define as tabelas, e, para cada tabela, define as colunas, cada uma com um determinado tipo de dados, bem como as relações com as outras tabelas (Lerman, 2010).

Com o serviço de tabelas em Windows Azure, não precisamos de desenhar a base de dados, apenas as classes. Define-se classes e tabela que pertencem uma ou mais classes, então pode-se guardar objectos instanciados armazenados em filas.

Cada classe deve ter três propriedades que são fundamentais na determinação de como os serviços do Windows Azure Table funcionan: *PartitionKey*, *RowKey* e *TimeStamp*. PartitionKey and RowKey são ambos strings.

3.3.3 - PartitionKeys e RowKeys

Nas bases de dados tradicionais utiliza-se um sistema de chaves primárias, chave estrangeira e restrições entre os dois (Lerman, 2010). Em Windows Azure Storage Table o conceito é diferente: a sequência de propriedades PartitionKey e RowKey trabalham juntas como um índice para sua tabela. Por isso, quando as definimos, temos que ter em conta como os dados vão ser consultados. Juntas, prevêem também exclusividade tal como uma chave primária para uma linha da tabela. Cada entidade numa tabela deve ter uma única combinação PartitionKey / RowKey.

É necessário considerar as perguntas quando se define um PartitionKey, porque é também utilizada para separar fisicamente as tabelas, que fornece o equilíbrio e escalabilidade.

3.3.4 - Tabelas de Particionamento

As Windows Azure Storage Tables permitem que as tabelas escalem a milhares de nós de armazenamento, distribuindo as entidades na tabela (Jai Haridas, 2009). Ao distribuir as entidades, pode ser desejável assegurar que um conjunto de entidades fique sempre junto num mesmo nó de armazenamento (ex: toda a informação relativa a um mesmo cliente ficar guardada

no mesmo servidor ou conjunto reduzido de servidores). Isto pode ser controlado por meio da escolha de um valor adequado para a PartitionKey.

Partition Key Document Name	Row Key Version	Property 3 Modification Time	.....	Property N Description	
Examples Doc	V1.0	8/2/2007	.....	Committed version	Partition 1
Examples Doc	V2.0.1	9/28/2007		Alice's working version	
FAQ Doc	V1.0	5/2/2007		Committed version	
FAQ Doc	V1.0.1	7/6/2007		Alice's working version	Partition 2
FAQ Doc	V1.0.2	8/1/2007		Sally's working version	

Figura 4 - Exemplo de partição

A figura acima mostra uma tabela que contém várias versões de documentos. Cada entidade nesta tabela corresponde a uma versão específica de um documento específico. Neste exemplo, a chave de partição da tabela é o nome do documento e a chave da linha é a *string* da versão. O nome do documento, juntamente com a versão identifica uma entidade específica na tabela. Neste exemplo, todas as versões do mesmo documento forma uma única partição e o sistema garante que serão guardados com afinidade geográfica (todos os dados no mesmo nó/servidor ou conjunto de servidores num mesmo *data center*).

3.3.5 - Arquitectura do Table Service do Windows Azure

A figura 6 apresenta a arquitectura do table service do Windows Azure. Esta é constituída pelos seguintes elementos:

- **Coluna** – representa um valor único de cada tabela, nomes das propriedades são case sensitive.
- **Partition Key** – é a propriedade de cada tabela.
- **Row Key** - a segunda chave da propriedade da tabela. Este é o ID único da entidade dentro da partição que ele pertence.
- **O parti**tion key combina com row key para identificar a entidade e uma tabela.

- **Timestamp** – cada entidade tem uma versão mantida pelo sistema.
- **Partition** – um conjunto de entidades de uma tabela com a partição de mesmo valor de chave.
- **Sort Order** – existe um único índice previsto actualmente, onde todas as entidades em uma tabela são ordenadas por partition key e row key. Isto significa que estas consultas especificante com estas chaves serão mais eficientes, e todos os resultados são devolvidos ordenados por PartitionKey e depois por RowKey.

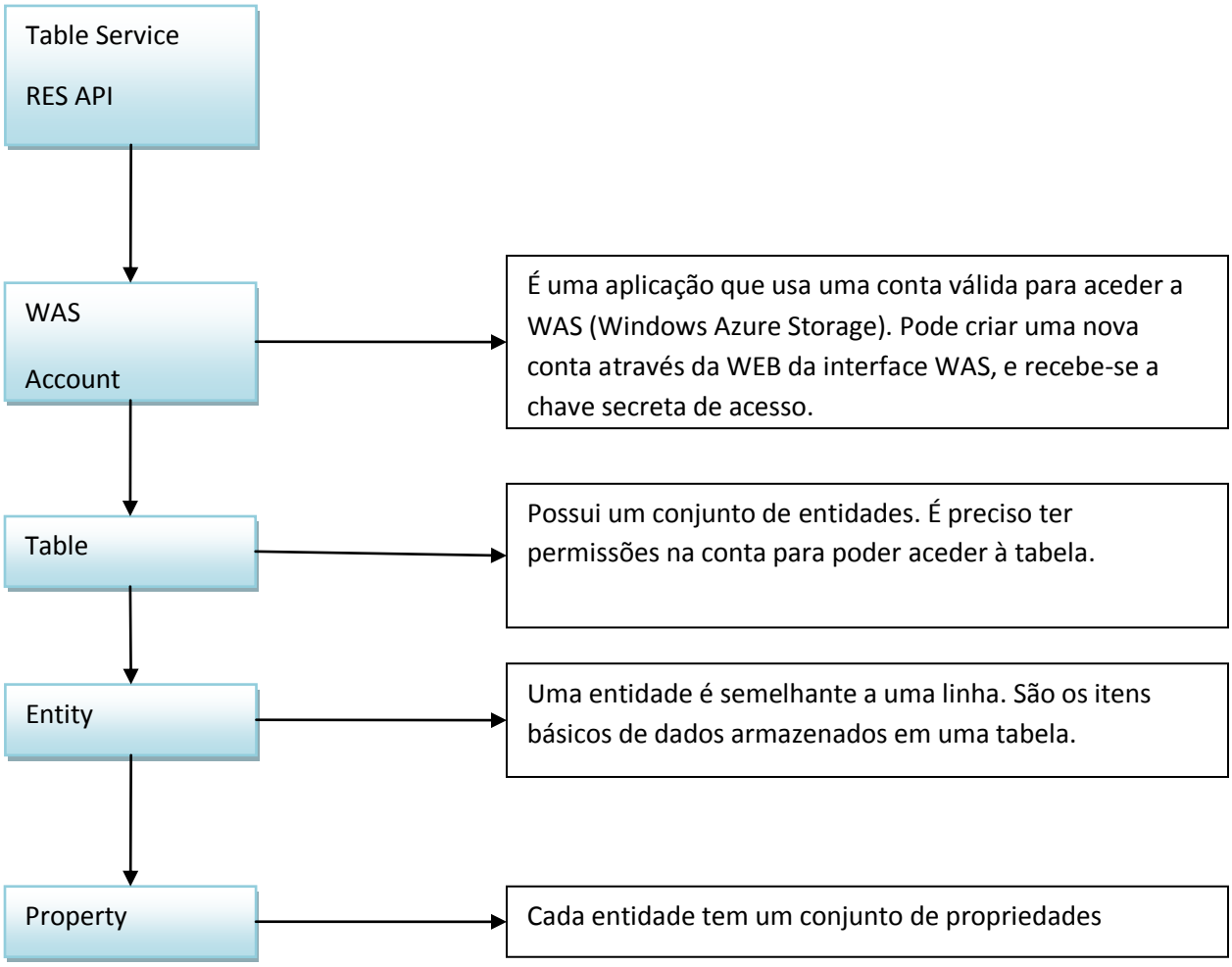

Figura 5 - Arquitectura do Table Service do Windows Azure

Capítulo 4

4 – Caso Prático e Discussão de Resultados

Foi apenas concretizada a técnica de refactorização com base na normalização e desnormalização, ficando para trabalhos futuros a implementação do segundo método, que utiliza a Windows Azures Storage Table. Para demonstração foi desnormalizado um modelo de uma base de dados normalizado de agência de viagens. Esta base de dados foi produzida num trabalho realizado para a Disciplina de Base de dados.

Para este efeito começou-se por fundir todas as tabelas que tinham relações de um para muitos (1:N), como é o caso das tabelas AUTOCARRO e VIAGEM (um autocarro pode fazer uma ou mais viagens). Neste sentido a tabela AUTOCARRO foi fundida com a tabela VIAGEM. Alteraram-se os nomes dos atributos com nomes iguais nas duas tabelas originais.

Também as tabelas das relações de muitos para muitos (M:N) foram fundidas numa única tabela, deixando de lado a possibilidade de ser três como na base de dados normalizada. As oito tabelas existentes na base de dados normalizada, passaram a ser duas após a sua desnormalização, ficando ainda com possibilidade de se tornar numa só tabela o processo de desnormalização continuasse. Optou-se por apresentar estas duas tabelas para que no trabalho futuro ser possível fazer duplicação de informação em tabelas "auxiliares", só usadas para suportar *queries* específicas. Por exemplo, se há uma tabela A e uma tabela B, e a query faz um JOIN às duas para obter um campo de A e outro campo de B, em Azure vale a pena criar uma tabela C que tem:

a) Colunas usadas no JOIN;

b) O campo de A;

c) O campo de B.

Isto poderia ser complementado com a explicação de que deve ser responsabilidade da aplicação Web, por exemplo, quando insere/altera/apaga um registo em A ou B, reflectir essa alteração na tabela "auxiliar" C.

Capítulo 5

5.1 – Conclusões

Foram apresentados dois métodos utilizados para fazer refactorização de base de dados com o objectivo de apoiar o desenvolvimento de aplicações para *cloud*: a normalização e desnormalização e a Windows Azures Storage Table. Foi apenas concretizada a técnica de refactorização com base na normalização e desnormalização, ficando para trabalhos futuros a implementação do segundo método, que utiliza a Windows Azures Storage Table e a aplicação Web.

O desenvolvimento deste projecto permitiu-me consolidar conhecimentos acerca da refactorização de base de dados, *cloud computing*, normalização/desnormalização e Windows Azure Storage Tables. A maior parte destes conceitos não consta no programa curricular.

Após a realização deste trabalho posso afirmar que *Cloud Computing* está pronta para proporcionar muitos benefícios e segurança da informação. Contudo antes de escolher este tipo de serviços tem que se ter em conta vários aspectos. Convém notar que juntamente com os benefícios vêm os riscos e as preocupações de segurança que se deve ter em conta. Como acontece com qualquer tecnologia emergente, *Cloud Computing* oferece a possibilidade de elevada recompensa em termos de contenção de custos, recursos, agilidade e velocidade de aprovisionamento. No entanto, como qualquer tecnologia ou abordagem nova, também pode trazer riscos elevados.

5.2 - Trabalho futuro

Futuramente gostaria migrar os dados para *Cloud* e desenvolver uma aplicação Web para demonstrar o trabalho de fundo que foi desenvolvido, através de uma base de dados simples.

Mais tarde em vez de utilizar uma base de dados simples, gostaria de abraçar um projecto com maior dimensão, onde as bases de dados são enormes para puder lidar com vários constrangimentos tanto na desnormalização da base de dados como também da sua migração para *cloud* e construir uma aplicação Web mais robusta.

Bibliografia

- Abreu, L. (2008). ASP.NET 3.5 Curso Completo. In L. Abreu, *ASP.NET 3.5 Curso Completo* (p. 9). Lisboa: Editora de informática, Lda.
- Amazon. (15 de 04 de 2009). *http://awsdocs.s3.amazonaws.com*. Obtido em 10 de 09 de 2010, de Amazon SimpleDB: http://awsdocs.s3.amazonaws.com/SDB/latest/sdb-dg.pdf
- Browne, J. (11 de 01 de 2009). *http://www.julianbrowne.com*. Obtido em 01 de 09 de 2010, de Brewer's CAP Theorem: http://www.julianbrowne.com/article/viewer/brewers-cap-theorem
- Gonçalves, P. M. (2008).
- *http://en.wikipedia.org*. (s.d.). Obtido em 10 de 09 de 2010, de Linda (coordination language): http://en.wikipedia.org/wiki/Linda_(coordination_language)
- *http://en.wikipedia.org*. (s.d.). Obtido em 10 de 09 de 2010, de Tuple space: http://en.wikipedia.org/wiki/Tuple_space
- *http://www.agiledata.org*. (s.d.). Obtido em 20 de 09 de 2010, de The Process of Database Refactoring: Strategies for Improving Database Quality: http://www.agiledata.org/essays/databaseRefactoring.html
- ISACA. (2006). *http://www.isaca.org.* Obtido em 01 de 09 de 2010, de ISACA: http://www.isaca.org/Knowledge-Center/Research/Documents/Cloud-Computing-28Oct09-Research.pdf
- Jai Haridas, N. N. (2009). *WINDOWS AZURE TABLE .*
- Lakshman, A. (25 de 08 de 2008). *http://www.facebook.com*. Obtido em 15 de 09 de 2010, de Cassandra – A structured storage system on a P2P Network: http://www.facebook.com/note.php?note_id=24413138919
- Lerman, J. (07 de 2010). *http://msdn.microsoft.com*. Obtido em 05 de 09 de 2010, de Data Points Windows Azure Table Storage – Not Your Father's Database: http://msdn.microsoft.com/en-gb/magazine/ff796231.aspx

- Ligia. (2010). *professoraligia*. Obtido em 27 de 07 de 2010, de professoraligia: www.professoraligia.com.br/materiais/Normalização%20de%20dados.doc