

Aplicação com o TPC-DS e comparação com TPC-H usado na Smartix

Igor Alves, João Vitor Pioner

Pontifícia Universidade Católica do Rio Grande do Sul (PUCRS)

***Resumo.**Esse artigo é uma contribuição de pesquisa do SmartIX com base no trabalho de reprodutibilidade. Esse artigo descreve a fundamentação e trabalhos relacionados e relata uma descrição do trabalho anterior.*

1. Introdução:

No trabalho anterior, fizemos uma reprodução do artigo “SmartIX: A database indexing based on reinforcement learning”. SmartIX é um agente de aprendizado de máquina para automaticamente escolher índices em bancos de dados relacionais e ensina a máquina a otimizar o banco de dados para não desperdiçar espaço e memória.

O objetivo deste projeto é adequar o smartix para aceitar outro benchmark, no nosso caso será o TPC-DS, e comparar o desempenho com o benchmark TPC-H, usado na versão original do smartix. Este trabalho falará sobre o TPC-DS e as diferenças com o TPC-H.

2. Fundamentação:

2.1. Smartix

O “Smartix” é um agente de aprendizado de máquina de reforço que escolhe índices em banco de dados relacionais e sua motivação é abstrair da tarefa do administrador de databases que envolve uma análise de todas as colunas candidatas quais podem improvisar na configuração de banco de dados. A smartix usa o reinforcement learning para explorar o espaço dos índices possíveis dentro da database provida pelo benchmark usado.

Esta IA funciona com 5 componentes:

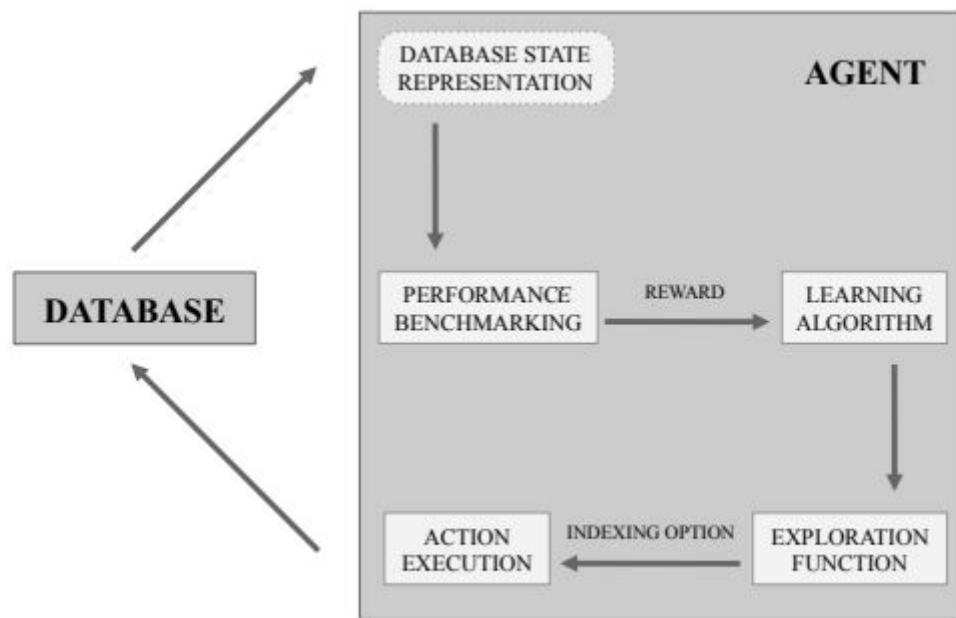


Figura 1: Representação da arquitetura do agente Smartix

- Database State Representation: converte a database atualizada em um estado para o agente interpretar.
- Performance Benchmarking: computa as métricas do estado convertido no componente anterior, após computar ele terá como output o valor da recompensa.
- Learning Algorithm: este componente receberá a recompensa e a vai prever os valores dos estados usando uma função de aproximação e atualiza os parâmetros se baseando nos algoritmos de reinforcement learning.
- Exploration Function: responsável por escolher um movimento para ser executado na database, seja usando a informação dada pelo Learning Algorithm ou escolhendo de forma semi-aleatória.
- Action Execution: Aplica a alteração que foi pedida pelo componente anterior.

2.2. TPC-H

O TPC é uma corporação não prófita que produz benchmarks para controlar a performance do banco de dados. O TPC-H tem redes ad-hoc orientadas a empresas que guardam grandes quantidades de banco de dados, sendo o H representado uma decisão de suporte de versões de benchmark. TPC-H é um bom proxy para consultar tarefas porque possui um business-oriented ad-hoc que conseguem suportar a um grande volume de dados. O TPC-H é um benchmark de suporte à decisão e consiste de um pacote de produtos orientada a consulta ad-hoc e modificação de dados concorrente. As consultas e os dados no banco de dados tem ampla relevância em indústrias mantendo implementação em grau de facilidade suficiente. O sistema de manipulação de banco de dados comercialmente disponível (DBMS) é usado para implementar o banco de dados. o ER deste modelo tem 8 tabelas e foi montado seguindo a terceira forma normal.

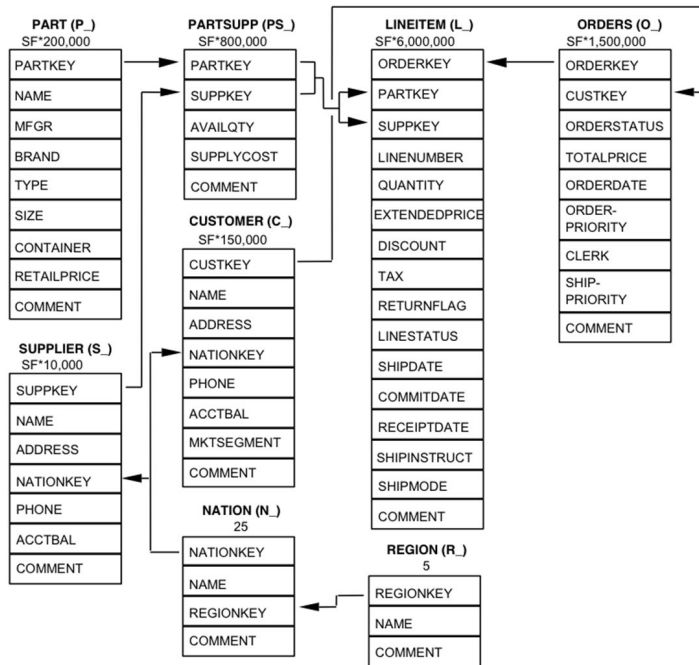


Figura 2: Relações entre as colunas das oito tables que são definidas de forma separada e individual.

A métrica de performance do TPC-H é expressa em $QphH@Size(Queries-per-Hour)$, no qual é obtida computando as métricas do $Power@Size$ e do $Throughput@Size$. O $Power@Size$ avalia quão rápido o DBMS computa as respostas das consultas sendo composta por: *first Refresh Function(RF1)* que insere 0,1% dos dados baseados no número de dados da tabela de ordens e lineitem nestas mesmas tabelas. A segunda função é a *single query stream* composto de 22 consultas geradas pela QGen. A última função é a *second Refresh Function(RF2)*, que remove a mesma porcentagem de fileiras do RF1. Essa métrica é computada usando a seguinte fórmula:

$$Power@Size = \frac{3600}{\sqrt[24]{\pi^{22} QI(i,0) \times \pi^2 RI(j,0)}} \times SF$$

sendo 3600 o número de segundos por hora, QI é o tempo de execução de cada consulta, RI é o tempo de execução das j funções RF1 e RF2 em um fluxo de consultas s e SF é o fator de escala da database. Já o $Throughput@Size$ estipula a habilidade do sistema de processar muitas consultas no menor tempo possível, explorando a vantagem do paralelismo do I/O e da CPU. A performance é calculada através da seguinte fórmula:

$$Throughput@Size = \frac{S \times 22}{T_s} \times 3600 \times SF$$

no qual T_s é o tempo total necessário para rodar o teste para S fluxos. $QphH@Size$ é calculada pela fórmula:

$$QphH@Size = \sqrt{Power@Size \times Throughput@Size}$$

2.3 TPC-DS:

Nesse trabalho, usaremos a arquitetura TPC-DS, que foi desenvolvida pela TPC depois do TPC-H. O TPC-DS usa as vantagens do TPC-H e do TPC-R para transformá-los num benchmark DSS moderno. Ele deve ser aplicado para alguma indústria que deverá transformar dados externos e operacionais em inteligência de indústrias e modela tarefas de suporte de decisão de fornecedores de produtos de vendas típicas. Seu objetivo é ajudar na leitura em relacionar intuitivamente aos componentes do benchmark, sem acompanhar o segmento da indústria para minimizar a relevância do benchmark.

Seus objetivos são alcançados se as especificações do benchmark do TPC tem testes de benchmark implementados com tecnologias, sistemas, produtos e preços geralmente disponíveis para usuários, relevantes ao segmento de mercado modelados ou representados pelo benchmark e implementado por um número significativo de usuários modelados ou representados. Além disso, o banco de dados do TPC-DS deverá usar comercialmente dados de processamentos de software disponíveis para ser implementado além de tarefas executadas pela interface SQL.

Os componentes do TPC-DS podem ser usados para acessar um sistema de topologia e metodologias de implementação com uso de comércios neutros tecnicamente rigorosos e diretamente comparáveis. O TPC-DS modela indústrias que devem manipular, vender e distribuir produtos e consiste múltiplas dimensões e mesas de fato. Cada dimensão tem uma única chave como coluna e pode ser classificada de três formas. A dimensão estática não muda com o tempo, a histórica muda indicando o período de tempo e a não-histórica não mantém as mudanças feitas na dimensão de dados.

2.3.1. Esquema das relações

Os antecessores do tpcds(tpc-d, tpc-h e tpc-r) foram montados na [terceira forma normal](#)(3NF). Conforme os anos se passam, a indústria foi se aproximando da ideia de um [esquema de estrela](#) por ser denormalizado, por consequência, foi decidido que o TPC-DS teria uma [estrutura de floco de neve](#) que seria um híbrido do esquema tradicional usado nos antecessores e que a indústria almeja.

O ER do TPC-DS possui duas fact tables sendo elas: Store_Sales e Store_Returns que estão associadas a eles, as associated dimension tables: Customer e Store.

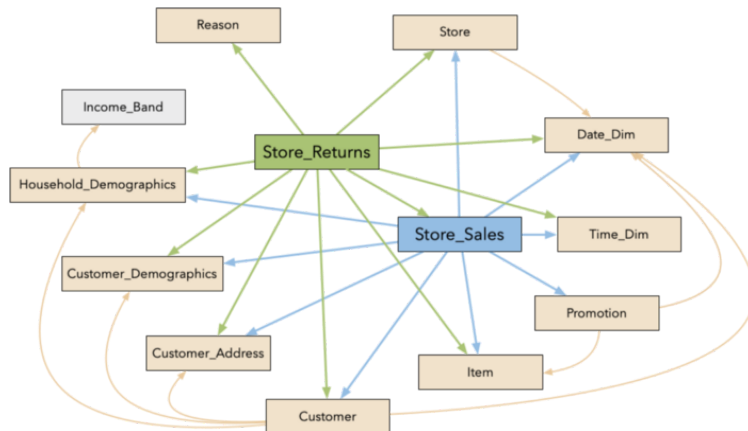


Figura 3: ER do Tpc-ds

As tables verde e azul são as tables que se aproximam do esquema de estrela, as laranjas são as dimension tables normalizadas que foram introduzidas pela estrutura snowflake. Dimension tables normalizadas também podem se relacionar com tables que não tem relação com as fact tables, neste caso o **Income_Band**.

Em comparação com o TPC-H, a estrutura do TPC-DS é significativamente mais complicado que seu antecessor:

	TPC-H	TPC-DS
Schema type	3rd Normal Form	Multiple Snowflake
Number of tables	8	24
Number of columns (min)	3	3
Number of columns (max)	16	34
Number of columns (avg)	~ 7.6	18
Number of foreign keys	9	104

2.3.2.Dataset

Como o TPC-H, TPC-DS se utiliza do conceito de fator de escala(SF) para controlar o tamanho da database. SF são valores discretos e correspondem com o tamanho aproximado da database em Gb. SF é um valor importante para calcular a eficiência e é o diferencial na hora de comparar databases a ponto de ser impossível comparar a eficiência de outras databases com SF diferente.

Para implementar a escala deve-se diferenciar a escala de domínio e escala da tupla. A escala de domínio se refere a escala das dimension tables que normalmente não dá para escalonar de maneira linear. Logo, o TPC-DS mede uma escala de maneira sub-linear e evita proporções não realísticas que existiam em benchmarks como o TPC-H. A segunda escala se refere ao número de tuplas existente nos fact tables. No TPC-DS mede esta escala de forma linear com o SF.

2.3.3. Workload

O TPC-DS separa a simulação do workload em 2: o Query workload e o Data maintenance workload. Os dois são essenciais para o funcionamento de um DSS: o Data maintenance sincroniza os dados da database com os dados registrados dentro do TPC-DS enquanto o Query faz as operações na database.

2.3.4. Execução e Métrica

Os benchmarks da TPC possuem métricas simples para comparações de performance. Para calcular, os tempos de execução devem ser medidos. No caso do TPC-DS, existem 4 intervalos para marcar os tempos:

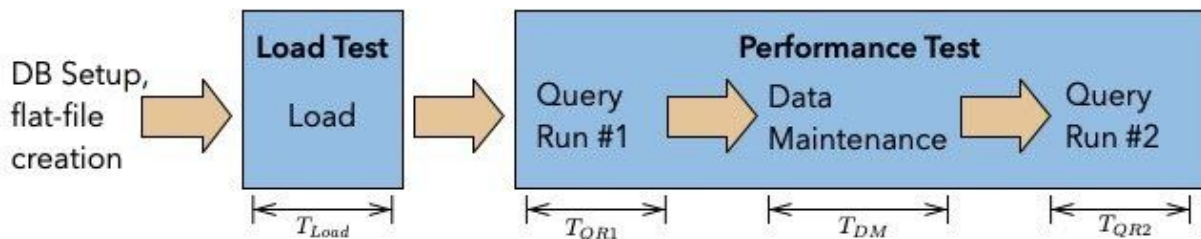


Figura 4: Representação das fases de testes para medir os tempos.

- O tempo de carregamento dos dados(T_{Load})
- Primeira execução da query(T_{QR1})
- Manutenção de dados(T_{DM})
- Segunda execução da query(T_{QR2})

Durante o teste de carregamento, todos os requisitos para medir a performance será organizado: a criação das tables, inserir os dados nas tables, organizando as estruturas auxiliares e outros. Depois tem o teste de performance no qual 99 queries são rodadas duas vezes, e as operações de manutenção é rodado entre as duas execuções das queries.

Após este processo, a fórmula usada para calcular a eficiência é utilizada. A métrica deste benchmark é chamada de Queries per hour for Decision Support, que pode ser calculada pela seguinte fórmula:

$$QphDS@SF = SF \cdot 3600 \cdot \frac{198 \cdot S}{(T_{QR1} + T_{DM} + T_{QR2} + 0.01 \cdot S \cdot T_{Load})}$$

Sendo SF o fator de escala, S o número de query streams per query run e as variáveis T são os tempos extraídos dos testes. O TPC-DS usa as query streams para simular múltiplos usuários operando o mesmo banco em paralelo. Assim, cada query stream executa todos os 99 queries sequencialmente, em ordem diferente. O número mínimo de query streams depende da escala selecionada. Geralmente, uma escala maior significa um número maior de query streams que devem ser processados.

3.Descrição do trabalho:

No plano original do projeto seria seguir o que mais ou menos foi feito no artigo original, que seria: criar uma database, sincronizar a database com o TPC-DS, carregar a database com os dados de teste do TPC-DS, sincronizar com o smartix e no final executar a smartix e treinar por 4,9 dias como foi feito no artigo original. Após os dias de treino, os dados coletados seriam comparados com os dados obtido no artigo. O Smartix possui cinco componentes. O estado do banco de dados converte as informações indexadas para o agente. A performance de benchmark calcula seus resultados da atual que será usada como sinal de recompensa. O algoritmo de aprendizagem tem como predição uma ação de estado usando uma função de aproximação e atualiza as informações baseadas em algoritmos. A função de exploração escolhe uma ação que será executada no banco de dados fazendo informação de aprendizagem explorada ou explorando ações sub-otimizadas escolhidas. O componente de ação de execução envia a opção de índice dependendo da ação escolhida para o banco de dados.

Para começar, neste projeto foi utilizado o Ubuntu Linux 18.04. Então para montar a database foi usado o software Postgres 10, com isso feito criamos as tables para TPC-DS inserir seus dados de teste. Para carregar os dados na database, antes tinha que sincronizar o Postgres com o TPC-DS. A partir deste momento não foi possível avançar disso, começou a aparecer erros(que estão relatados no nosso relatório do projeto) e com o tempo passando ficou difícil de executar direito. Nós não temos uma resposta concreta do porque o erro aconteceu, mas suspeitamos algum arquivo corrompido que mesmo deletado talvez estivesse de alguma forma interferindo na sincronização ou, o mais plausível das duas, que o ubuntu utilizado não tinha algum recurso que era necessário para a execução ou o ubuntu em si foi mal instalado.


4. Resultados:

Como não foi possível extrair os dados, os dados mostrados são dados de dois testes diferentes para mostrar as diferenças nos benchmarks citados neste artigo. O teste do TPC-DS é [um teste do Alibaba Cloud Computing](#) com uma database com um SF de 10.000 GB, 4 streams e 396 queries. Já o teste do TPC-H foi realizada no [Dell Poweredge R6415](#) também com uma escala de 10.000GB com 12 streams.

TPC Benchmark™ DS Metrics

Total System Cost (RMB)	TPC-DS Throughput (QphDS@10000GB)	Price/Performance (RMB/QphDS@10000GB)	Availability Date
¥1,126,006.68	18,998,559	¥0.06	As of Publication

(1) Resultado Final do teste do alibaba.

	Dell PowerEdge R6415 using Exasol 6.2		TPC-H Rev. 2.18.0 TPC-Pricing Rev. 2.4.0	
			Report Date July 8, 2019	
Total System Cost	Composite Query per Hour Metric		Price / Performance	
\$809,230	8,667,578 QphH@10000GB		\$ 0.10 \$/QphH@10000GB	
Database Size	Database Manager	Operating System	Other Software	Availability Date
10,000GB	Exasol 6.2	CentOS 7.6	None	July 8, 2019

(2)Resultado Final do teste da Dell.

Convertendo os custos e o valor da performance o TPC-DS aceita um volume de dados maior que o TPC-H e também é um sistema mais barato de se montar, além da diferença significativa dos resultados dos próprios benchmark.

5.Conclusão:

TPC-DS é o benchmark que sucedeu o TPC-H . O TPC-DS continua sendo usado pela indústria aceitando volumes ainda maiores que os antecessores e mesmo assim é consegue ter um custo beneficio maior que os outros. Sobre o projeto, infelizmente não teve muito para mostrar, mas acreditamos que criamos um começo para resolver o problema proposto aqui para uma futura tentativa.

6.Referências:

Nambiar, Ragunath; The Making of TPC-DS; TPC; Disponível em
:<http://www.tpc.org/tpcds/presentations/the_making_of_tpcds.pdf>

Autor desconhecido; TPC Benchmark™ DS -Standard Specification; TPC; Disponível em:
<http://tpc.org/tpc_documents_current_versions/pdf/tpc-ds_v2.13.0.pdf>

Autor desconhecido; TPCBENCHMARK H; TPC; Disponível em:
<http://www.tpc.org/tpc_documents_current_versions/pdf/tpc-h_v2.18.0.pdf>

Menzler, Julian; A Summary of TPC-DS; Disponível em:
<<https://medium.com/hyrise/a-summary-of-tpc-ds-9fb5e7339a35>>

Autor desconhecido; Dell PowerEdge R6415 using Exasol 6.2; Dell; Disponível em:
<http://www.tpc.org/results/fdr/tpch/dell~tpch~10000~dell_poweredge_r6415~fdr~2019-07-09~v01.pdf>

Autor desconhecido; Alibaba Cloud Computing LTD; Disponível em:
<http://www.tpc.org/results/fdr/tpcds/alibaba~tpcds~alibaba_cloud_analyticdb~fdr~2020-06-17~v01.pdf>