

Bibliografia

- [1] AutoAdmin project, Database Group, Microsoft Research, <http://research.microsoft.com/dmx/autoadmin/>
- [2] AGRAWAL, S., CHAUDHURI, S., NARASAYYA, V. **Automated Selection of Materialized Views and Indexes for SQL Databases**. In: PROCEEDINGS OF THE INTERNATIONAL CONFERENCE ON VERY LARGE DATABASES (VLDB), p. 496-505, 2000
- [3] AGRAWAL, S.; CHAUDHURI, S., KOLLAR L., MARATHE A., NARASAYYA, V. SYAMALA, M. **Database Tuning Advisor for Microsoft SQL Server 2005**, In: PROCEEDINGS OF THE INTERNATIONAL CONFERENCE ON VERY LARGE DATABASES (VLDB), p. 1110-1121, 2004.
- [4] NICOLAS, B. AND CHAUDHURI, S. , **Automatic Physical Database Tuning: A Relaxation-based Approach**. In: PROCEEDINGS OF THE 2005 ACM SIGMOD INTERNATIONAL CONFERENCE ON MANAGEMENT OF DATA (SIGMOD), p. 237-238, 2005.
- [5] BURLESON, DONALD, **Creating a Self-Tuning Oracle Database**, Rampant, 2004.
- [6] COSTA, R. L. C.; LIFSCHITZ, S.. **Index self-tuning and agent-based databases**. In: PROCEEDINGS OF THE LATIN-AMERICAN CONFERENCE ON INFORMATICS (CLEI), p 76, Abstracts Proceedings; 12pp. CD-ROM Proceedings , 2002.
- [7] CHAUDHURI S., DATAR M., **Index selection for databases: a hardness study and a principled heuristic solution**. In: IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, 16 (11): 1313-1323, 2004.
- [8] CHAUDHURI, S. , NARASAYYA, V., **Microsoft Index Tuning Wizard for SQL Server 7.0**, In: PROCEEDINGS OF THE ACM SIGMOD INTERNATIONAL CONFERENCE ON MANAGEMENT OF DATA, p. 553-554, 1998.
- [9] CHAUDHURI, S. NARASAYYA, V., **AutoAdmin “what-if” index analysis utility**, In: PROCEEDINGS OF THE ACM SIGMOD INTERNATIONAL CONFERENCE ON MANAGEMENT OF DATA, p. 367-377, 1998.
- [10] CUNHA, E., **Estudo de viabilidade de uma Plataforma de Baixo Custo para Datawarehouse**, Master's thesis, Departamento de Informática, Universidade Federal do Paraná, 2004.
- [11] DIAS, K., RAMACHER, M., SHAFT, U., VENKATARAMANI, V., WOOD, G., **Automatic Performance Diagnosis and Tuning in Oracle**, In: PROCEEDINGS OF THE CIDR CONFERENCE, p. 84-94, 2005.

- [12] ELMASRI, R. NAVATHE, S., **Sistemas de Bancos de Dados**, LTC, 2002
- [13] GARCIA-ARELLANO C.M., LIGHTSTONE,S., LOHMAN,G., MARKL, V., STORM,A. **A Self-Managing Relational Database Server: Examples from IBM's DB2 Universal Database for Linux Unix and Windows**, In: IEEE TRANSACTIONS ON SYSTEMS, MAN AND CYBERNETICS SPECIAL ISSUE ON ENGINEERING AUTONOMIC SYSTEMS, 2005.
- [14] GRAEFE, G., **The Value of Merge-Join and Hash-Join in SQL Server**, In: PROCEEDINGS OF THE INTERNATIONAL CONFERENCE ON VERY LARGE DATABASES (VLDB), p. 250-253, 1999.
- [15] HARRISON, GUY, **Oracle SQL High-Performance Tuning**, Prentice Hall, 1997.
- [16] HAAS, PETER J., KANDIL ,M., LERNER,A., MARKL,V. POPIVANOV,I., RAMAN, V., ZILIO,DANIEL C.: **Automated statistics collection in action**. In: PROCEEDINGS OF THE 2005 ACM SIGMOD INTERNATIONAL CONFERENCE ON MANAGEMENT OF DATA (SIGMOD), p. 933-935, 2005
- [17] HORN P., **Autonomic Computing: IBM's Perspective on the State of Information Technology**, 2001. <http://www.research.ibm.com/autonomic>, acessado 27/08/05.
- [18] KÖNIG, A AND NABAR, S., **Scalable Exploration of Physical Database Design**. In: PROCEEDINGS OF INTERNATIONAL CONFERENCE ON DATA ENGINEERING (ICDE), 2006
- [19] KENDALL, E.A. KRISHNA, P.V.M. PATHAK, C.V. AND SURESH, C.B., **A framework for agent systems**. In: Fayad, M.; Schmidt, D.; Johnson, R., editors, IMPLEMENTING APPLICATIONS FRAMEWORKS: OBJECT ORIENTED FRAMEWORKS, p. 113-154. John Wiley & Sons, 1999.
- [20] LOHMAN G., LIGHTSTONE, S., **SMART: Making DB2 (More) Autonomic**. In: PROCEEDINGS OF THE INTERNATIONAL CONFERENCE ON VERY LARGE DATABASES (VLDB), p. 877-879, 2002.
- [21] LIFSCHITZ S., MORELLI, E.T., **Auto-Sintonia em SGBDs: o fim do DBA?** In: SQL Magazine, 8 (1): 8-9, 2003.
- [22] LIFSCHITZ S., SALLES, M.V., **Autonomic Index Management**. In: PROCEEDINGS OF THE INTERNATIONAL CONFERENCE ON AUTONOMIC COMPUTING (ICAC), p. 304-305, 2005.
- [23] LIFSCHITZ, S.; MILANÉS, A., SALLES, M.V., **Estado da Arte em Auto-Sintonia de Sistemas de Bancos de Dados Relacionais**, Technical report, Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro (PUC-Rio), 2004.
- [24] LOHMAN G., VALENTIN, G., ZILIO, D., ZULIANI, M., SKELLEY,A. **DB2 advisor: An optimizer smart enough to recommend its own indexes**. In: PROCEEDINGS OF THE IEEE INTERNATIONAL CONFERENCE ON DATA ENGINEERING (ICDE), p. 101-110, 2000.

- [25] LI, W., ZILIO, DANIEL C., BATRA, VISHAL S., SUBRAMANIAN, M., ZUZARTE, C., NARANG, I: **Load Balancing for Multi-tiered Database Systems through Autonomic Placement of Materialized Views**. In: PROCEEDINGS OF THE IEEE INTERNATIONAL CONFERENCE ON DATA ENGINEERING (ICDE), p. 102, 2006.
- [26] MILANÉS, A., **Uma arquitetura para auto-sintonia global de SGBDs usando agentes**, Master thesis, Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro (PUC-Rio), 2004.
- [27] MILANÉS, A. LIFSCHITZ, S. **Design and Implementation of a Global Self-Tuning Architecture**. In: In: PROCEEDINGS OF SIMPÓSIO BRASILEIRO DE BANCOS DE DADOS (SBBB), p. 70-84, 2005
- [28] MORELLI, E.T; LIFSCHITZ, S., **Estudo dos Malefícios Gerados pela Fragmentação de Índices em Sistemas de Bancos de Dados Relacionais**, Technical report, Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro (PUC-Rio), 2006.
- [29] http://www.osdl.org/lab_activities/kernel_testing/osdl_database_test_suite/osdl_dbt-3/
- [30] Oracle 10g. <http://www.oracle.com>
- [31] PostgreSQL. <http://www.postgresql.org>
- [32] SALLES, M. V., **Criação Autônoma de Índices em Bancos de Dados**, Master thesis, Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro (PUC-Rio), 2004.
- [33] SHASHA, DENNIS, BONNET, PHILIPPE; **Database Tuning**, Morgan Kaufmann Publishers, 2003.
- [34] STORM, A.J., GARCIA-ARELLANO C.M., LIGHTSTONE, S., DIAO, Y., SURENDRA, M. **Adaptive Self-tuning Memory in DB2 UDB**, PROCEEDINGS OF THE INTERNATIONAL CONFERENCE ON VERY LARGE DATABASES (VLDB), 2006.
- [35] SATTler, K., SCHALLEHN, E. GEIST, I. **Autonomous Query-driven Index Tuning**. In: PROCEEDINGS OF THE 2005 ACM SIGMOD INTERNATIONAL CONFERENCE ON MANAGEMENT OF DATA (SIGMOD), p. 793-795, 2006
- [36] SQL Server 2005. <http://www.microsoft.com/sql>
- [37] <http://pt.wiktionary.org/wiki/bonus>
- [38] <http://www.tpc.org>
- [39] <http://www.tpc.org/tpch>
- [40] Transaction Processing Performance Council. **TPC Benchmark H (Decision Support) Standard Specification Revision 2.3.0**
- [41] BOOCH, G.; RUMBAUGH, J.; JACOBSON, I.. **UML: Guia do Usuário**. Campus, 2000.
- [42] WEISS, G. editor. **Multiagent Systems: a Modern Approach to Distributed Artificial Intelligence**. The MIT Press, Cambridge, MA, USA, 1999.

- [43] WEIKUM, G. AND HASSE, C. MONKEBERG, A. AND ZABBACK, P., **The Confort Automatic Tuning Project'** , In: Informations Systems 19(5), 1994.
- [44] D. ZILIO, C. ZUZARTE, S. LIGHTSTONE, W. MA, G. LOHMAN, R. COCHRANE, H. PIRAHESH, L. COLBY, J. GRYZ, E. ALTON, D. LIANG, AND G. VALENTIN. **Recommending Materialized Views and Indexes with IBM DB2 Design Advisor.** In: PROCEEDINGS OF THE INTERNATIONAL CONFERENCE ON AUTONOMIC COMPUTING (ICAC), p. 180-188, 2004.

Apêndice A

Benchmark TPCB

No segundo capítulo apresentou-se a Heurística de Benefícios acompanhada por uma bateria de testes baseada em um conjunto de tabelas e consultas extraídas do *benchmark* TPCB. Neste apêndice revela-se o esquema de dados que suporta as consultas, bem como apresentam-se detalhes sobre cada uma destas, onde enfatizam-se os ajustes que foram necessários para validar a implementação do Agente de Benefícios utilizando uma carga de trabalho com características OLAP.

A.1 TPC

TPC (Transaction Processing Council) [38] representa uma entidade que oferece diversos conjuntos de testes (*benchmarks*) visando avaliar desempenhos de produtos. Composta por um amplo leque de empresas, tais como Microsoft, Oracle, EDS ou Bull, oferece aos fabricantes a possibilidade de submissão de seus produtos às regras oferecidas pelo TPC para obter argumentos quantitativos que os diferenciem da concorrência.

Alguns *benchmarks* particulares do TPC largamente utilizados pela indústria merecem destaque:

- TPC-C: também conhecido por *order entry benchmark*, possui características OLTP (*Online Transaction Processing*), ou seja, a presença de muitas transações, porém com poucos comandos, que tanto consultam quanto atualizam uma base de dados.
- TPC-H: classificado como OLAP (*Online Analytical Processing*), atende às necessidades de sistemas com ênfase ao apoio à decisão, onde as consultas costumam ser onerosas e as atualizações desprezíveis.

Diferente de TPC-C, ocorrem parametrizações em TPC-H, o que faz com que dificilmente uma mesma consulta seja executada duas vezes utilizando os mesmos filtros. Há, entretanto, algumas variações deste *benchmark*, onde aumenta-se o número de comandos que alteram dados: TPC-H-UPD25 e TPC-H-UPD75 compõem-se por 25% e 75%, respectivamente de comandos de atualização.

Existem ainda outros *benchmarks* na TPC, agora obsoletos: TPC-A e TPC-W, precursores de TPC-C; TPC-D e TPC-R, por sua vez antecessores de TPC-H, além de TPC-B, cujo objetivo consiste em ponderar quantas transações por segundo uma determinada base poderia suportar.

A.2 TPC-H

O *benchmark* TPC-H baseia-se em um modelo de dados relativamente simples, apresentado na Figura A.1. Trata-se de um problema genérico envolvendo clientes, fornecedores e compras de itens, subdivididos em componentes.

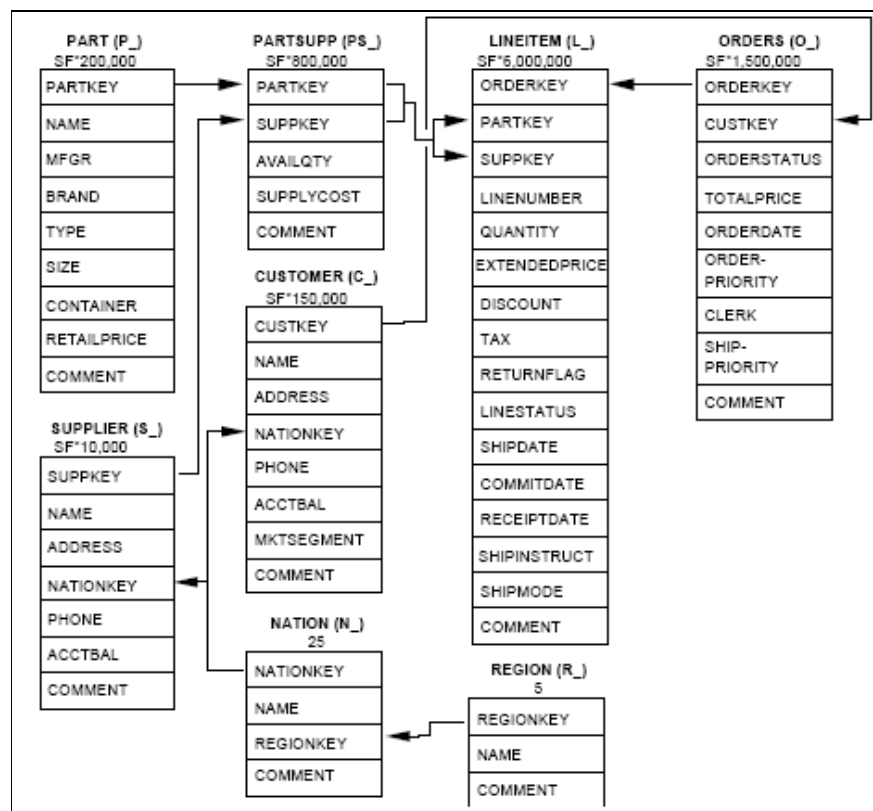


Figura A.1: esquema de Dados base de TPC-H

Algumas observações devem ser ressaltadas, uma vez analisada a figura A.1:

- O número que aparece logo abaixo do nome da tabela representa sua cardinalidade. Esta pode ser fixa (tabelas **region** e **nation**), ou variável, onde multiplica-se uma constante por um *scale factor* determinado no momento da criação da base. Por exemplo, caso SF valha 3, haverá 450.000 tuplas na tabela de clientes (**customer**).
- Estão representadas cinco regiões (continentes), que congregam vinte e cinco nações (tabelas **region** e **nation**, respectivamente). Clientes e Fornecedores (tabelas **supplier** e **customer**) estão associados às nações. Enquanto os primeiros realizam pedidos de compras (tabela **orders**), os segundos fornecem componentes (tabela **part**) de itens de compra. Como um fornecedor pode oferecer vários itens e um item pode ser disponibilizado por vários fornecedores, existe uma tabela para registrar esta relação NxN (**partsupp**). Finalmente, a tabela mais volumosa do modelo (**lineitem**) associa itens de compra a pedidos.
- Os parênteses ao lado dos nomes das tabelas indicam o prefixo utilizado para denominar os campos da tabela em questão. Desta forma, a chave primária da tabela de fornecedores chama-se S_SUPPKEY;
- As flechas indicam as associações entre chaves primárias e estrangeiras. Assim, vemos que a chave primária da tabela **nation** está vinculada ao campo **nationkey** na tabela de fornecedores (tabela **supplier**);

O tamanho total da base de dados depende do fator de escalabilidade (SF – *scale factor*). Para SF=1, a base completa ocupa aproximadamente 1 GB e os volumes de cada tabela são os que aparecem na Figura A.1. Já para SF=30, a base ocupará 30 GB e a tabela de itens de pedidos de compra possuirá cento e oitenta milhões de tuplas (30 x 6 milhões).

A base de dados sofre acessos de um conjunto de consultas possuindo características *ad-hoc*, ou seja, não se conhece nem a ordem de execução, nem os

parâmetros de cada uma das 22 consultas (leitura) e duas funções (uma insere dados e a outra elimina).

O *benchmark* TPC-H compõe-se por vários testes, como revela a Tabela A.1.

Teste	Detalhes
<i>Refresh Function 1</i> (RF1)	Insere tuplas nas duas maiores tabelas: orders e lineitem .
<i>Refresh Function 2</i> (RF2)	Elimina tuplas das tabelas acima.
<i>Power</i>	Compreendido pela <i>Refresh Function 1</i> , grupo completo de análises de consultas e <i>Refresh Function 2</i> .
<i>Throughput</i>	Dispara vários fluxos (<i>streams</i>) de comandos em paralelo. Invoca as vinte e duas consultas seguidas pelas <i>Refresh Functions</i> .
<i>Performance</i>	Reúne os testes <i>Power</i> e <i>Throughput</i>

Tabela A.1: testes constituindo o *benchmark* TPC-H

Antes de executar os testes, alimenta-se o banco de dados com base em arquivos texto previamente gerados. Isto acontece graças a uma ferramenta normalmente apelidada por DBGEN (*Database Generator*), seguindo preceitos de [40]. A ordem na qual são lidas as vinte e duas consultas, bem como seus parâmetros, também são gerados por um programa gerador denominado: QGEN (*Query Generator*)

A quantidade de fluxos de comandos utilizados no teste *Throughput* deve acompanhar o fator de escalabilidade da base de dados. A especificação oficial do *benchmark* [40] apresenta a seguinte correlação:

SF	Fluxos
1	2
10	3
30	4
100	5
300	6

O teste *Throughput* produz duas medidas: a vazão (quantos comandos por segundo) e tempo transcorrido.

As funções de atualização (RF1 e RF2) trabalham sobre as maiores tabelas, **orders** e **lineitem**; a primeira função insere tuplas e a segunda exclui. Vale ressaltar que, como a execução de uma RF sempre vem seguida da outra, a

quantidade de tuplas permanece estável, já que RF1 insere 0,1% de tuplas, enquanto RF2 elimina outras 0,1% de tuplas.

Tanto RF1 quanto RF2 recebem por argumento o fator de escalabilidade, coerente com o tamanho atual das tabelas. Assim, na tabela **lineitem**, quando SF=1, a execução de RF1 causará a inserção de seis mil tuplas: $6.000.000 \times 0,01$ (veja Figura A.1). Caso fosse necessário aumentar o percentual de tuplas afetadas, as funções poderiam ser executadas com argumento SF=30, o que, em bases com fator de escalabilidade um, teríamos o aumento trinta vezes maior (de 0,1% para 3% : 180.000 tuplas).

A.3 Consultas Escolhidas

Nesta seção apresentamos cada consulta TPC-H escolhida no Capítulo 2, adaptada à sintaxe PostgreSQL, em suas versões utilizadas pelo *toolkit* DBT-3 [29]. Destacam-se as alterações necessárias para que o Agente de Benefícios pudesse tratá-las, bem como os ajustes realizados para execuções no SQL Server 2005 e Oracle 10g.

As consultas escolhidas foram: 1, 2, 4, 10, 17 e 19. As figuras A2 a A7 mostram os códigos.

Houve a necessidade de substituir referências a funções por valores discretos, pois a implementação corrente do Agente de Benefícios ainda não trata funções sobre tipos de dados **time** e **date** no corpo do comando **select** (**date** e **interval**). Também buscou-se aumentar a seletividade com intuito ressaltar o papel do Agente, assim reduziram-se intervalos em filtros e trocaram-se operadores relacionais “<=” e “>=” por “=”.

Consulta 1

```

1.  Select
2.      l_returnflag,
3.      l_linestatus,
4.      Sum(l_quantity) as sum_qty,
5.      Sum(l_extendedprice) as sum_base_price,
6.      Sum(l_extendedprice * (1 - l_discount)) as
    sum_disc_price,
7.      Sum(l_extendedprice * (1 - l_discount) * (1 +
    l_tax)) as sum_charge,
8.      Avg(l_quantity) as avg_qty,
9.      Avg(l_extendedprice) as avg_price,
10.     Avg(l_discount) as avg_disc,
11.     count(*) as count_order
12. From
13.     Lineitem
14. Where
15.     l_shipdate <= date'1998-12-01' - interval ':1
    days'
16. Group by
17.     l_returnflag,
18.     l_linestatus
19. Order by
20.     l_returnflag,
21.     l_linestatus;

```

Figura A.2: totais consolidados para transações financeiras ocorridas em um determinado período

Alteração necessária: a linha 15 foi trocada por esta:

```
L_shipdate = '19981201'
```

No SQL Server 2005, a mesma linha mudou para:

```
L_shipdate = 'Dec  1 1998'
```

No Oracle 10g, a mesma linha mudou para:

```
L_shipdate = '1998-12-01'
```

Consulta 2

```

1. Select
2.     S_acctbal, s_name,    n_name,    p_partkey,
3.     P_mfgr,    s_address, s_phone,   s_comment
4. From
5.     Part,
6.     supplier,
7.     partsupp,
8.     nation,
9.     Region
10. Where
11.     P_partkey = ps_partkey
12.     and s_suppkey = ps_suppkey
13.     and p_size = 20
14.     and p_type like '%COPPER'
15.     and s_nationkey = n_nationkey
16.     and n_regionkey = r_regionkey
17.     and r_name = 'AMERICA'
18.     and ps_supplycost = (
19.         select min(ps_supplycost)
20.         From
21.             partsupp,
22.             supplier,
23.             nation,
24.             Region
25.         Where
26.             p_partkey = ps_partkey
27.             And s_suppkey = ps_suppkey
28.             And s_nationkey = n_nationkey
29.             And n_regionkey = r_regionkey
30.             And r_name = 'AMERICA'
31.     )
32. order by
33.     S_acctbal desc,
34.     N_name,
35.     S_name,
36.     P_partkey

```

Figura A.3: seleciona qual fornecedor deve ser escolhido para realizar um pedido de um determinado componente em uma dada região

Para favorecer a seletividade, foi acrescentada a seguinte linha:

```
LIMIT 1
```

Em SQL Server 2005, a primeira linha linha foi substituída por:

```
Select Top 1
```

E no Oracle 10g, a linha 11 mudou para:

```
p_partkey = ps_partkey and rownum = 1
```

Consulta 4

```

1.  Select
2.      o_orderpriority,
3.      count(*) as order_count
4.  From
5.      Orders
6.  Where
7.      o_orderdate >= date '1994-05-01'
8.      And o_orderdate < date '1994-05-01' + interval '3
    month'
9.      And exists (
10.         Select
11.             *
12.         From
13.             Lineitem
14.         Where
15.             l_orderkey = o_orderkey
16.             and l_commitdate < l_receiptdate
17.     )
18. group by
19.     o_orderpriority
20. order by
21.     o_orderpriority;
```

Figura A.4: verifica eficácia do sistema de controle de prioridades de pedidos e checa níveis de satisfação de clientes.

Alteração necessária: trocadas as linhas 7 e 8 por estas:

```
o_orderdate >= '19980801'
and o_orderdate < '19980801'
```

SQL Server 2005:

```
o_orderdate >= 'Aug  1 1998'
and o_orderdate < 'Aug  8 1998'
```

Oracle 10g:

```
o_orderdate >= '1998-08-01'
and o_orderdate < '1998-08-08'
```

Consulta 10

```
1.  Select
2.      c_custkey,
3.      c_name,
4.      sum(l_extendedprice * (1 - l_discount)) as
      revenue,
5.      c_acctbal,
6.      n_name,
7.      c_address,
8.      c_phone,
9.      c_comment
10. From
11.     customer,
12.     orders,
13.     lineitem,
14.     nation
15. Where
16.     c_custkey = o_custkey
17.     and l_orderkey = o_orderkey
18.     and o_orderdate >= date '1994-11-01'
19.     and o_orderdate < date '1994-11-01' + interval '3
      month'
20.     and l_returnflag = 'R'
21.     and c_nationkey = n_nationkey
22. group by
23.     c_custkey,
24.     c_name,
25.     c_acctbal,
26.     c_phone,
27.     n_name,
28.     c_address,
29.     c_comment
30. order by
31.     revenue desc
32. LIMIT 20;
```

Figura A.5: mostra clientes que podem ter tido problemas com entregas.

Alteração necessária: trocadas linhas 19 e 20 por estas:

```
o_orderdate >= '19980801'
and o_orderdate < '19980808'
```

SQL Server 2005:

```
o_orderdate >= 'Aug  1 1998'
o_orderdate < 'Aug  8 1998'
```

Oracle 10g:

```
o_orderdate >= '1998-08-01'
o_orderdate < '1998-08-08'
```

Consulta 17

```

1.  Select
2.      sum(l_extendedprice) / 7.0 as avg_yearly
3.  From
4.      lineitem,
5.      Part
6.  Where
7.      P_partkey = l_partkey
8.      and p_brand = 'Brand#31'
9.      and p_container = 'LG JAR'
10.     and l_quantity < (
11.         Select
12.             0.2 * avg(l_quantity)
13.         From
14.             Lineitem
15.         Where
16.             l_partkey = p_partkey
17.     );

```

Figura A.6: determina a possível perda decorrente caso alguns pedidos tenham reduzidas pequenas quantidades de componentes. Isto permitiria reduzir o nível de *overhead*, permitindo maior concentração nas grandes compras.

Nenhuma alteração foi necessária. Também foi executada sem alterações no SQL Server 2005 e Oracle 10g

Consulta 19

A versão original, exibida a seguir, apresentava um custo de execução proibitivo. Seguindo idéias sugeridas por [10], a consulta foi reescrita, o que levou seu custo original a uma queda de 99,99995%.

```

1.  Select
2.      sum(l_extendedprice* (1 - l_discount)) as
3.      revenue
4.  From
5.      lineitem,
6.      Part
7.  Where
8.      (
9.          p_partkey = l_partkey
10.         and p_brand = 'Brand#21'
11.         and p_container in ('SM CASE', 'SM BOX',
12.         'SM PACK', 'SM PKG')
13.         and l_quantity >= 3 and l_quantity <= 13
14.         and p_size between 1 and 5
15.         and l_shipmode in ('AIR', 'AIR REG')
16.         and l_shipinstruct = 'DELIVER IN PERSON'
17.     )
18.  Or

```

```

17.      (
18.          p_partkey = l_partkey
19.          and p_brand = 'Brand#55'
20.          and p_container in ('MED BAG', 'MED BOX',
    'MED PKG', 'MED PACK')
21.          and l_quantity >= 15 and l_quantity <= 25
22.          and p_size between 1 and 10
23.          and l_shipmode in ('AIR', 'AIR REG')
24.          and l_shipinstruct = 'DELIVER IN PERSON'
25.      )
26.      Or
27.      (
28.          p_partkey = l_partkey
29.          and p_brand = 'Brand#34'
30.          and p_container in ('LG CASE', 'LG BOX',
    'LG PACK', 'LG PKG')
31.          and l_quantity >= 23 and l_quantity <= 33
32.          and p_size between 1 and 15
33.          and l_shipmode in ('AIR', 'AIR REG')
34.          and l_shipinstruct = 'DELIVER IN PERSON'
35.      );

```

Figura A.7: exibe descontos produzidos segundo diferentes estratégias utilizadas.

A versão reescrita apresenta os mesmos resultados da original, porém seu custo caiu de forma considerável.

```

1.  Select
2.      sum(l_extendedprice* (1 - l_discount)) as revenue
3.  From
4.      lineitem,
5.      Part
6.  Where
7.      p_partkey = l_partkey
8.      and l_shipmode in ('AIR', 'AIR REG')
9.      and l_shipinstruct = 'DELIVER IN PERSON'
10.     And
11.     ((
12.         p_brand = 'Brand#21'
13.         and p_container in ('SM CASE', 'SM BOX', 'SM
PACK', 'SM PKG')
14.         and (l_quantity >= 3 and l_quantity <= 13)
15.         and p_size between 1 and 5
16.     )
17.     Or
18.     (
19.         p_brand = 'Brand#55'
20.         and p_container in ('MED BAG', 'MED BOX',
    'MED PKG', 'MED PACK')
21.         and (l_quantity >= 15 and l_quantity <= 25)
22.         and p_size between 1 and 10
23.     )
24.     Or
25.     (

```

```
26.         p_brand = 'Brand#34'
27.         and p_container in ('LG CASE', 'LG BOX', 'LG
    PACK', 'LG PKG')
28.         and (l_quantity >= 23 and l_quantity <= 33)
29.         and p_size between 1 and 15
30.     ));
```

Figura A.8: nova versão para consulta 19.

As linhas 7, 8 e 9 da nova versão apareciam repetidas vezes na versão original. Este fato triplicava a realização da junção entre duas grandes tabelas, **parts** e **lineitem**, o que explica os altos custos registrados na versão original.

Esta consulta também foi executada com SQL server 2005 e Oracle 10g, em sua versão reescrita.

Durante a fase de testes, levantou-se a hipótese de estar a consulta escrita de forma ineficiente, de propósito, com intuito de provocar testes de *stress*. Infelizmente, exceto [10], não foi possível obter-se certeza quanto à utilização dessa consulta em outros testes envolvendo cargas TPC-H

A.3 Demais Consultas

As outras consultas não foram escolhidas para compor a bateria de testes por uma ou mais razões listadas a seguir:

- Apresentavam características ainda não suportadas pela implementação atual do Agente de Benefícios. Por exemplo: subqueries em cláusula *from*;
- Os níveis de seletividade apresentados não foram suficientes para criação de índices pelo Agente;

Apêndice B

Avaliação de Ferramentas de Apoio ao DBA

B.1 - Introdução

O objetivo deste trabalho consiste em analisar diversas ferramentas de apoio ao trabalho desempenhado pelo DBA, especialmente no que tange à melhoria do desempenho, isto é, mecanismos que permitam reduzir durações de transações, bem como aumentar a quantidade de comandos executados por unidade de tempo (vazão ou *throughput*). Visitaram-se diversos fabricantes e procurou-se examinar em detalhes pelo menos uma ferramenta de cada um.

Os fabricantes visitados com respectivos endereços na Internet aparecem listados na Tabela B1.

Fabricante	Endereço
1. BMC	http://www.bmc.com
2. Embarcadero	http://www.embarcadero.com
3. Idera	http://www.idera.com
4. Veritas	http://www.veritas.com

Tabela B.1: relação de fabricantes visitados.

Durante os trabalhos de análise, utilizou-se uma base única constituída por seis tabelas.

Alunos (31 linhas)

Campo	Tipo	Tamanho	Nulabilidade	Observações
Matricula	numérico	3	Não	Chave primária
Nome_aluno	Texto	30	Não	
Tel_aluno	Texto	10	Sim	
Endereco_aluno	Texto	30	Sim	
Cidade_aluno	Texto	20	Sim	
Uf_aluno	Texto	2	Sim	

Cursos (10 linhas)

Campo	Tipo	Tamanho	Nulabilidade	Observações
Cod_curso	Numérico	3	Não	Chave primária
Nome_curso	Texto	60	Não	
Carga_horaria	Numérico	3	Sim	
Preco	Numérico	7,2	Sim	
Pre-requisito	Numérico	3	Sim	

Instrutores (10 linhas)

Campo	Tipo	Tamanho	Nulabilidade	Observações
Cod_instrutor	numérico	3	não	Chave primária
Nome_instrutor	Texto	30	não	
Tel_instrutor	Texto	10	Sim	
Admissao	Data		Sim	

Turmas (20 linhas)

Campo	Tipo	Tamanho	Nulabilidade	Observações
Cod_turma	Numérico	3	não	Chave primária
Cod_curso	Numérico	3	sim	Chave estrangeira
Cod_instrutor	Numérico	3	sim	Chave estrangeira
Preco_hora_instrutor	Numérico	7,2	sim	
Sala	Numérico	2	sim	

Historico (100 linhas)

Campo	Tipo	Tamanho	Nulabilidade	Observações
Cod_turma	Numérico	3	não	Chave primária e estrangeira
Matricula	Numérico	3	não	Chave primária e estrangeira
Nota	Numérico	7,2	sim	

Antigos_Alunos (5.500.000 linhas)

Campo	Tipo	Tamanho	Nulabilidade	Observações
Matricula	numérico	3	Não	
Nome_aluno	Texto	30	Não	
UF	Texto	2	Sim	

Esta última tabela, apesar do excessivo tamanho, não possui índices de forma proposital, com intuito de receber sugestões das ferramentas analisadas.

B.2 - BMC

O fabricante BMC, sediado em Houston-Texas, possui extenso leque de produtos que atende desde o desenvolvedor ao responsável pela rede, passando pelo DBA. Existem ferramentas voltadas para Bancos de Dados contemplando os seguintes SGBDRs: Oracle, Informix, Sybase, Microsoft SQL Server e IBM DB2 (tanto mainframe, quanto Linux e Windows).

A família de produtos de apoio à gerência de Bancos de Dados de maior destaque chama-se **SmartDBA**. Em termos de apoio à melhoria de performance, ela constitui-se de diversos módulos, uns mais abrangentes que outros. A Tabela B.2 lista os módulos.

Módulo	Descrição
Monitoramento de Performance (Oracle, DB2, Sybase e SQL Server)	Oferece monitoramento, alertas para problemas em potencial e diagnóstico de problemas. Com base nas informações geradas, o DBA logra tomar medidas corretivas com maior probabilidade de acerto. Destaque para SmartDBA Cockpit , DBXray e PATROL for Database Knowledge Module .
Gerenciamento de Espaço (Oracle e DB2)	Estende as funcionalidades do módulo anterior oferecendo um completo gerenciamento de espaço através da ferramenta Space Management . Ela provê diversas alternativas para reorganização do espaço ocupado, além de inúmeros relatórios.
Análise de SQL e Tuning (Oracle e DB2)	Através de um agente não-intrusivo (<i>High Speed Data Collector</i>), ou diretamente consultando metadados, colhe informações na memória do Servidor. Isto permite realizar diversas análises de desempenho, bem como sugerir índices para tabelas muito consultadas. A ferramenta SQL Explorer , representante deste grupo, será vista em detalhes a seguir.
Garantia de Utilização ² (Oracle)	Acrescenta ao módulo anterior a possibilidade de planejamento de capacidade. Ferramenta: PATROL Performance Assurance .

Tabela B.2: relação de módulos da família SmartDBA do fabricante BMC

² Este termo procura traduzir a palavra **assurance**, com idéia de garantir que algo deve ocorrer.

O produto a ser estudado em detalhes será o SQL Explorer, que possui versões para Oracle e DB2. O público alvo seriam DBAs realizando otimizações reativas, ou seja, analisando comandos ineficientes que já ocorreram, ou desenvolvedores trabalhando de forma pró-ativa, melhorando comandos que ainda não ocorreram.

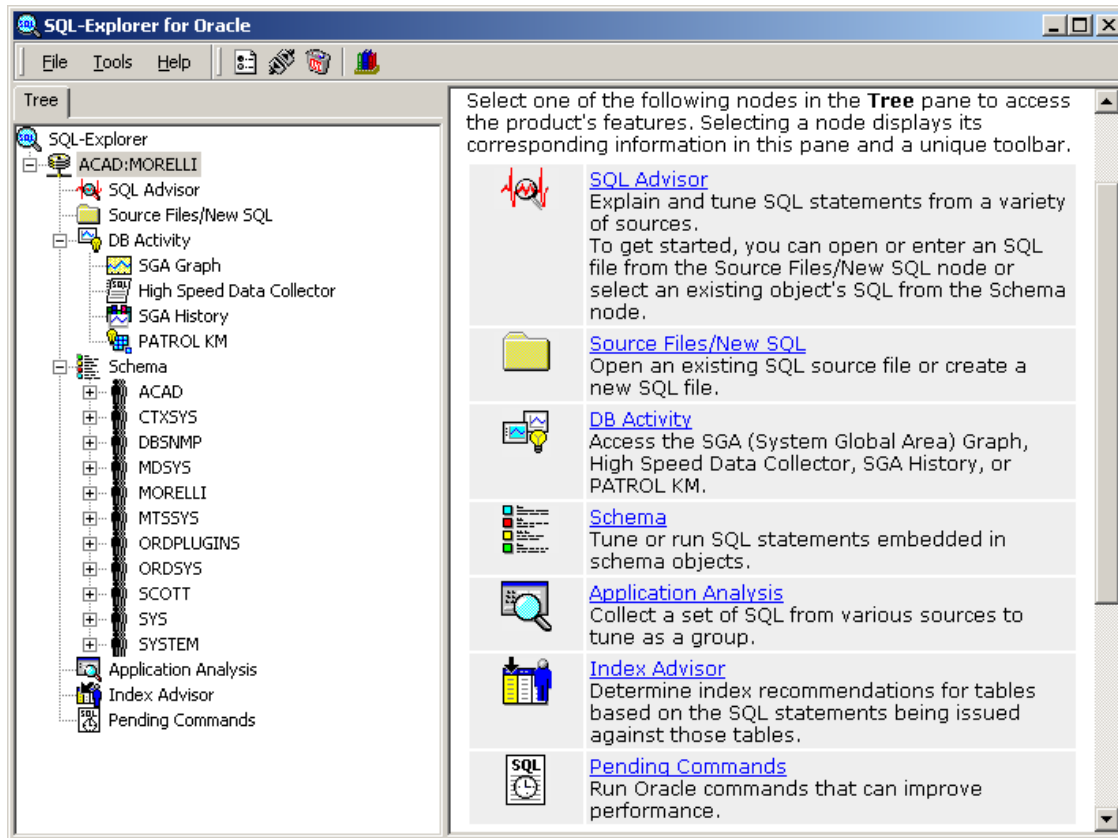


Figura B.1: ambiente da ferramenta BMC SQL Explorer

A Tabela B.3 descreve os elementos presentes na Figura B.1

Componente	Descrição
SQL Advisor	Armazena análises de comandos. Mais detalhes a seguir.
Source Files/New SQL	Permite abrir um arquivo contendo comandos por analisar ou simplesmente digitar um comando cuja execução será explicada (<i>explained</i>).
DB Activity	Concentra os mecanismos de coleta de informações à memória do Servidor. Tanto podem ser gerados gráficos a partir de metadados (SGA Graph ou SGA History), como via agentes externos instalados à parte (High Speed Data Collector ou Patrol KM).

Schema	Relação de usuários do Banco de Dados corrente e seus respectivos objetos (tabelas, índices, procedures etc.). Uma interessante funcionalidade deste módulo consiste em clicar o botão direito sobre uma tabela e, caso ela ainda não tenha estatísticas, selecionar Create Analyze Command. O comando gerado pode ser executado imediatamente ou armazenado em Pending Commands.
Application Analysis	Permite analisar um grande conjunto de comandos que possam estar distribuídos por diversas porções de código.
Index Advisor	Analisa carências ou excessos (por ineficiência ou redundância) de índices em tabelas presentes em comandos executados. Mais detalhes a seguir.
Pending Commands	Possui comandos a serem executados em ocasiões de menor utilização do Banco.

Tabela B.3: elementos da interface do produto BMC SQL Explorer

Suponha que fosse necessário avaliar o impacto causado pelo seguinte comando:

```
Select nome_aluno Aluno, nome_Curso, nome_Instrutor
from alunos a, historico h, turmas t , instrutores i,
cursos c
where a.matricula = h.matricula
and t.cod_turma = h.cod_turma
and t.cod_instrutor = i.cod_instrutor
and c.cod_curso = t.cod_curso
and upper(nome_curso) like '%INTRODUÇÃO%'
and upper(nome_instrutor) like 'PEDRO%'
order by 2,1, 3;
```

Figura B.2: consulta típica acessando várias tabelas

Resumidamente, o comando apresentado pela Figura B.2 mostra os nomes dos alunos que participaram de cursos introdutórios cujas turmas tiveram como instrutor pessoas cujos nomes começam por Pedro.

Existem várias formas de analisar um comando. Uma delas, poderia ser clicando o botão direito do mouse sobre **Source Files/New SQL** e escolhendo **New SQL Statement**. Na janela recém aberta, o comando por analisar seria copiado e depois o botão **Explain** seria pressionado.

O resultado seria algo semelhante à Figura B.3

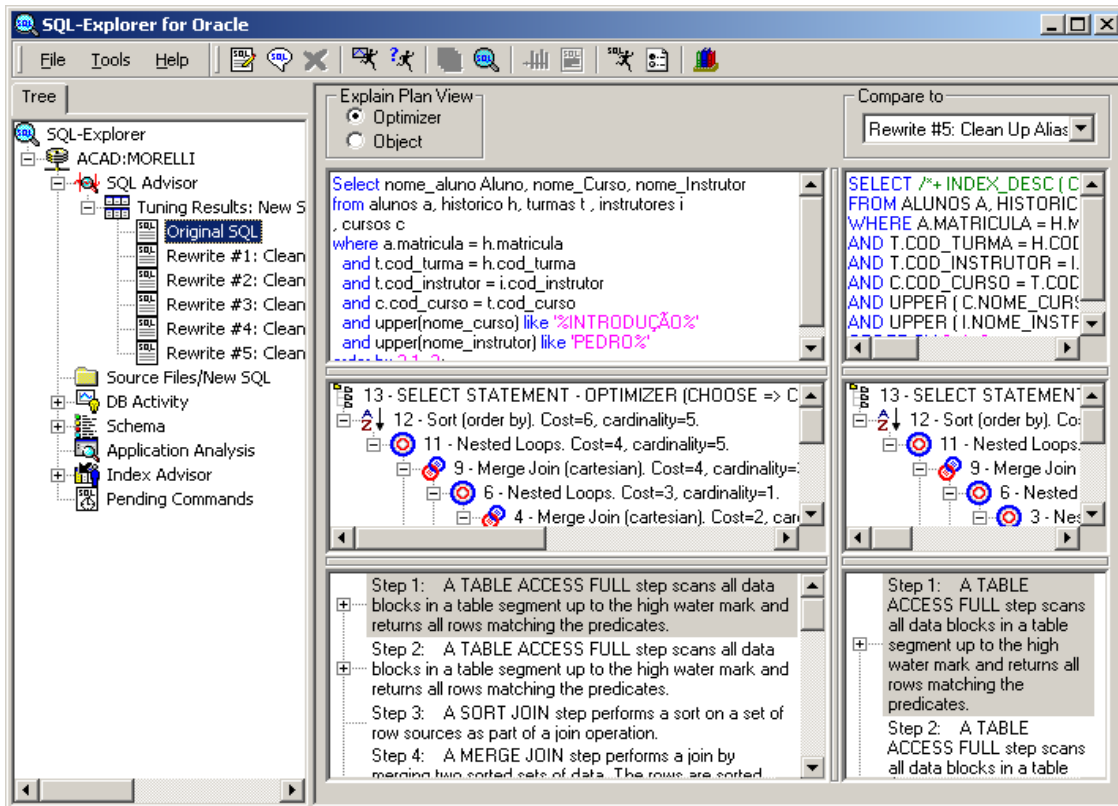


Figura B.3: resultado de análise de comando SQL

Alguns aspectos interessantes:

- ✓ Acontecem várias reescritas do comando original. E todas podem ser comparadas entre si;
- ✓ O plano de execução de cada comando aparece na janela intermediária e as explicações de cada etapa na janela inferior;
- ✓ Diversas estatísticas dos comandos analisados podem ser obtidas expandindo **DB Activity** e selecionando **SGA Graph**;
- ✓ Aliás, objetivando melhorar comandos ineficientes, estes também podem ser extraídos a partir dos dados coletados sob **DB Activity**;
- ✓ Clicando o botão direito sobre algum comando e selecionando **Get Execution Statistics**, pode-se criar simulações sobre a execução do comando, isto é, sem necessariamente executar o comando, obtém-se uma idéia de seu custo total e duração estimada. Recomenda-se a utilização deste recurso para avaliar se mudanças no comando têm efeito esperado.

Vale observar que mudanças sugeridas aos comandos podem ser documentadas através de relatórios HTML.

Uma vez coletados vários comandos e realizadas algumas otimizações, recomenda-se a utilização do recurso **Index Advisor** para identificar ausência ou excesso de índices. Por exemplo, a partir do comando da Figura B.2, o resultado apresentado pela Figura B.4 poderia ser obtido. Perceba-se a sugestão de novos índices para a tabelas HISTORICO (campo MATRICULA) e TURMAS (campos COD_CURSO e COD_INSTRUTOR)

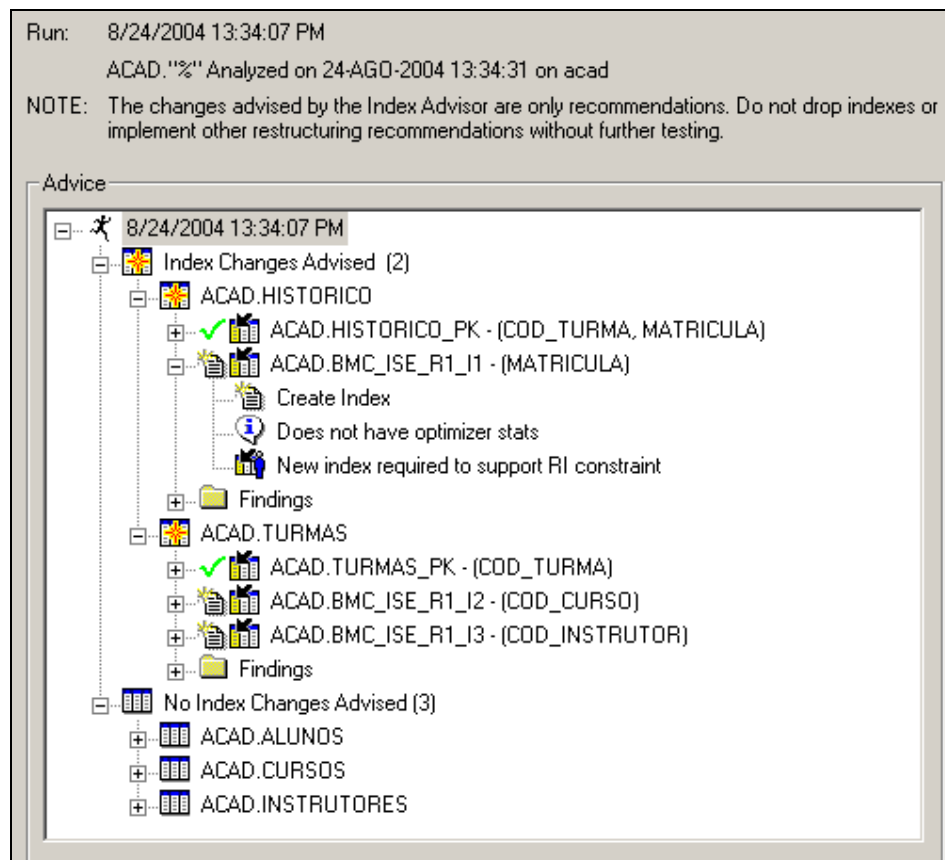


Figura B.4: resultado de um Index Advisor.

Chegou-se ao resultado da figura B.4 clicando-se com o botão direito sobre **Index Advisor** e selecionando **New Index Advisor Run**. Depois bastou informar o usuário possuindo as tabelas sobre as quais foi executado o comando.


Como visto, um papel proeminente desempenhado pelo Index Advisor consiste em analisar índices com base nos comandos coletados. Além da sugestão

de novos, também pode ser sugerida a eliminação de outros que não tenham sido utilizados no período de coleta.

Algumas observações devem ser levantadas a respeito do Index Advisor:

- ✓ Baseia-se inteiramente nas estatísticas do banco, portanto, estas devem ser mantidas atualizadas permanentemente;
- ✓ Não há forma de automatizar a criação ou eliminação dos índices. Isto deve ser executado manualmente pelo DBA ou desenvolvedor.
- ✓ Não se deve confiar cegamente nas informações mostradas, isto é, antes de criar ou, principalmente, eliminar um índice sugerido, deve-se realizar estudos mais detalhados. Deve-se ressaltar que, muitas vezes, o conjunto de comandos analisados não constitui uma amostragem suficientemente representativa.

Existem duas formas de coleta de dados da memória do Servidor; ou via metadados, produzindo consultas a partir de visões com prefixo V\$, ou através de um produto independente, o High Speed Data Collector, instalado onde estiver a base de dados. Quando este documento foi produzido, o HSDC encontrava-se na versão 2.1.

A coleta de metadados ainda pode ser obtida utilizando uma **SQL Collection**, consistindo de um histórico mais extenso de comandos. Para implementar este mecanismo, deve-se clicar em  (**Set up SQL Collection**), estando ativo o item **SGA History**.

Imagine que tenham sido submetidos os três comandos apresentados pela Figura B.5 sobre a tabela ANTIGOS_ALUNOS que possui milhões de linhas e nenhum índice:

```
select nome_aluno from Antigos_Alunos
where matricula between 50000 and 50005;

select nome_aluno from Antigos_Alunos where matricula = 65000;

select max(matricula) from Antigos_Alunos;
```

Figura B.5: comandos em busca de índices hipotéticos

Uma interessante funcionalidade do SQL Explorer consiste em criar índices hipotéticos que permitem avaliar o quanto um comando pode ser melhorado. Após analisar, digamos, o primeiro comando, clica-se com o botão direito sobre ele e

escolhe-se **What-if?**. Será aberta uma Caixa de Diálogo onde deve ser informado o nome do novo índice e como será constituído. Suponha que tenha-se “criado” um índice sobre o campo MATRÍCULA. Após atualizadas estatísticas sobre o comando (clique direito e **Get Execution Statistics**), chega-se à figura B.6

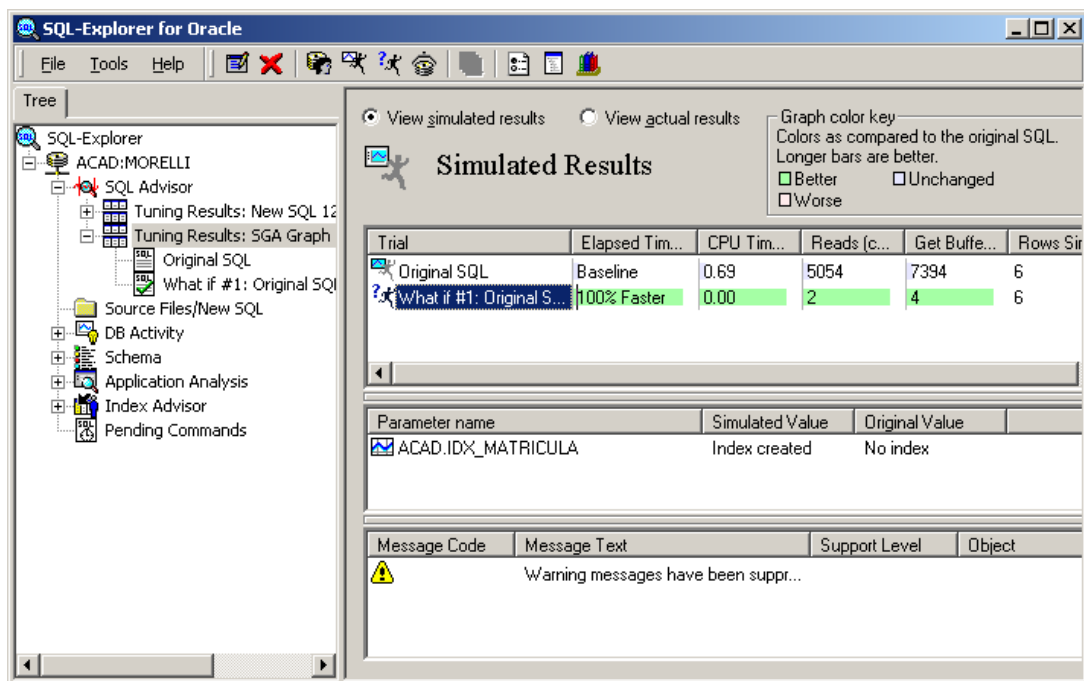


Figura B.6: comparação entre dois comandos: um com índice hipotético e outro sem.

Obtém-se o resultado exibido na Figura B.6 graças à criação de uma **Virtual Instance**.

A Tabela B.4 informa as características atendidas pelo produto

	Explicação de Planos de Execução	☑
	Avaliação de custos de comandos	☑
	Sugestões para mudanças em comandos	☑
	Sugestões de utilização de Estruturas Auxiliares (Índices, Visões Materializadas)	☑
	Sugestões de atualizações de estatísticas	☑
	Lista de comandos mais agressivos, classificados por consumo de recursos (tempo, cpu, leituras físicas, leituras lógicas)	☑
	Apresenta estado atual da memória (dados e comandos)	☑
	Taxa de recompilação de comandos	☑
	Gerência de Bloqueios	
	Ocorrência de Deadlocks (quem foi cancelado)	

Tabela B.4: resumo de funcionalidades do produto BMC SQL Explorer

B.3 - Embarcadero

O fabricante Embarcadero, sediado em San Francisco-California, alardeia de oferecer produtos cobrindo todas as etapas da manipulação de dados. Ainda que as ferramentas somente possam ser instaladas em ambiente Windows, os SGBDRs atendidos são: IBM DB2, Oracle, Sybase, RDB e Microsoft SQL Server.

Começando pelo **ER/Studio**, que permite modelar problemas de bancos de dados, seguindo pelo **DT/Studio**, que integra diversas bases, migrando ou transformando dados entre elas. **Rapid/SQL** encarrega-se do desenvolvimento de código e trabalha em sincronia com **SQL Tuner** (disponível somente para Oracle), responsável por identificar SQLs agressivos e analisar os ganhos de performance após a criação de novos índices ou alteração de existentes. **DBArtisan** oferece um completo ambiente de apoio ao DBA com tarefas de agendamento, controle de utilização de espaço e monitoramento. Finalmente, deve-se destacar o **Extreme Test** para gerar dados de teste e **Performance Center**, oferecendo a possibilidade de identificar e resolver problemas de performance em bases heterogêneas e de forma simultânea, antes que tornem-se mais graves.

Infelizmente, a política de testes de seus produtos resulta bastante antipática. Após preencher um longo questionário, informa que um representante comercial entrará em contato em “no máximo” 24 horas.

B.4 - Idera

O fabricante Idera, também sediado em Houston-Texas, dedica-se exclusivamente ao SGBDR Microsoft SQL Server. O produto carro-chefe denomina-se **SQL Suite**, que, na verdade, constitui um agregado de outros. Disponível em duas versões, **Standard**, onde constam **SQL Diagnostic Manager** (monitoramento e diagnóstico de problemas), **SQL Tool** (apoio à administração como, por exemplo, gerência de espaço, segurança ou alterações em estruturas de objetos) e **SQL Config** (acompanha mudanças no ambiente); e **Advanced**, onde acrescenta-se o **SQL Schedule** (agiliza o gerenciamento de tarefas agendadas).

Ainda são oferecidos outros dois produtos: **DTx**, para transferência de dados; e **SQL Check**, gratuita ferramenta que apresenta um diagnóstico geral do ambiente através de uma interface bastante intuitiva.

O produto a ser estudado em detalhes será o SQL Diagnostic Manager, que, uma vez instalado, introduz um novo serviço, o **SQL diagnostic manager**, cuja principal função consiste em coletar informações dos diversos servidores SQL Server aos quais tenha acesso. Apesar de tratar-se de um ótimo apoio ao DBA, seu trabalho é totalmente passivo, já que não toma nenhuma medida no sentido de corrigir problemas levantados, nem sugere quaisquer medidas tais como criação de índices ou estatísticas.

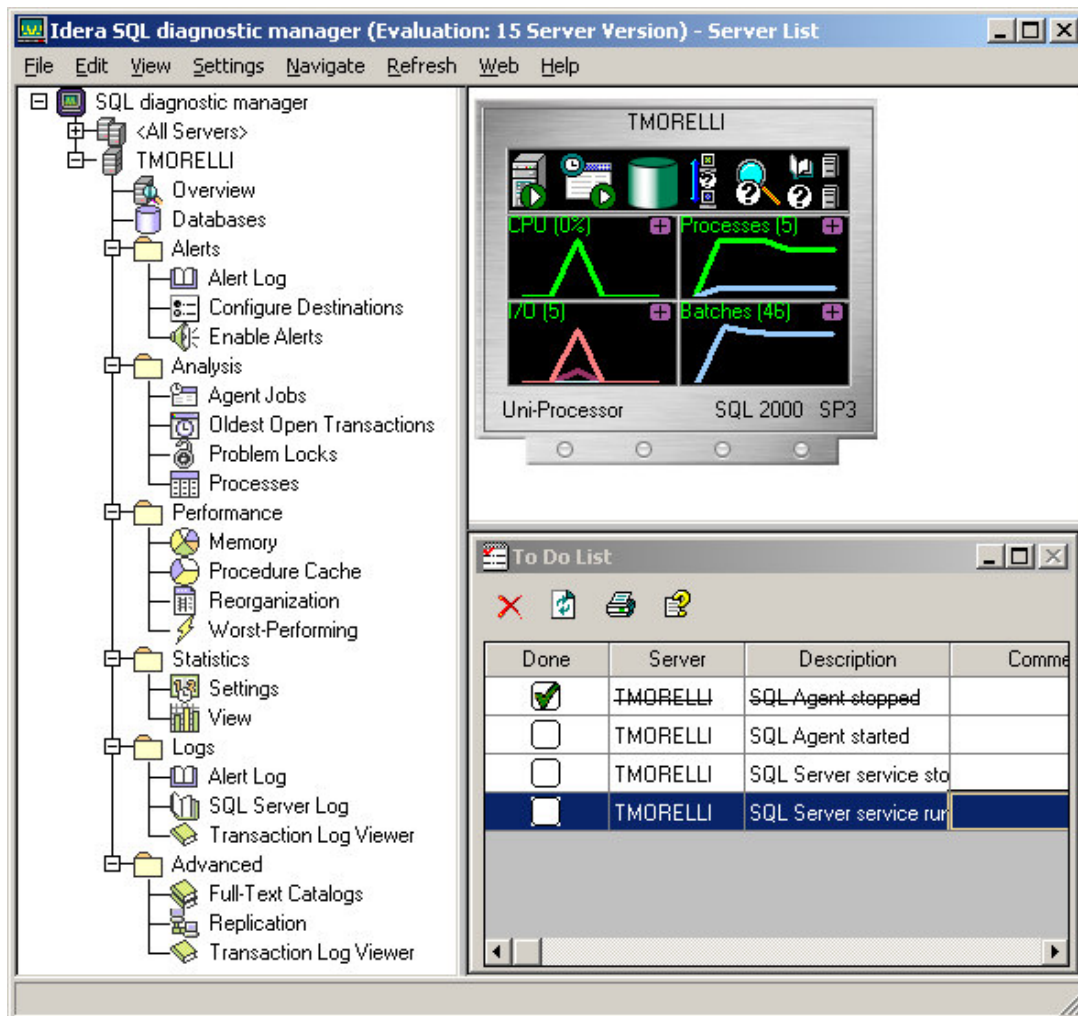


Figura B.7: ambiente da ferramenta Idera SQL Diagnostic Manager

O painel à esquerda, organizado em forma de árvore, possui uma entrada (**All Servers**) onde mostra-se um resumo de todos os servidores gerenciados e, ainda tantas entradas quantos forem estes servidores³. A Tabela B.5 descreve os recursos dedicados a cada Servidor

Componente	Descrição
Overview	Resumo de todos os Bancos de Dados do servidor corrente, além de diversos medidores de performance, tais como utilização de memória ou acesso a discos.
Databases	Exibe mais detalhes sobre o Banco de Dados escolhido. Visualizam-se todas as tabelas, o histórico de cópias de segurança e a utilização de espaço em disco. Vale notar que não é possível, por exemplo, aumentar um arquivo, tarefa de outra ferramenta, a SQL Tool .
Alerts	Em Alert Log , informa mensagens mais relevantes extraídas do prontuário (SQL Server Logs); permite redirecionar o destino das mensagens de alerta (Configure Destinations), além de configurá-las, inibindo, ativando ou alterando os valores utilizados como parâmetro para dispará-las (Enable Alerts).
Analysis	Agent Jobs mapeia estado das tarefas agendadas (quais falharam, quais estão ativas, etc.); Oldest Open Transactions revela as transações que não foram concluídas. Muitas vezes, longas transações causam problemas de bloqueios, assunto tratado em Problem Locks . Curioso ressaltar que a ferramenta não toma nenhuma iniciativa quando ocorrem deadlocks. Finalmente, Processes exibe conexões correntes.
Performance	Memory e Procedure Cache exibem gráficos revelando conteúdos da memória (dados e comandos, respectivamente). Reorganization trata da fragmentação de dados em Worst Performing destaca os comandos mais agressivos.

³ A possibilidade de visualizar informações condensadas a partir de todos os servidores gerenciados constitui-se na primeira grande diferença em relação ao Enterprise Manager, ferramenta nativa do SQL Server para apoio ao DBA.

Statistics	Analisa carências ou excessos (por ineficiência ou redundância) de índices em tabelas presentes em comandos executados. Mais detalhes a seguir.
Logs	Visualiza prontuários do servidor, bem como permite inspecionar o Transaction Log dos bancos.
Advanced	Trata índices sobre grandes textos (full-text), replicação

Tabela B.5: serviços oferecidos pela ferramenta SQL Diagnostic Manager

Imagine que tenha sido submetido o primeiro comando apresentado na Figura B.5 sobre a tabela ANTIGOS_ALUNOS que possui milhões de linhas e nenhum índice.

A Figura B.8 mostra o item **Worst Performing**, onde o comando recém executado pode aparecer no grupo **Worst-Performing SQL Batches**

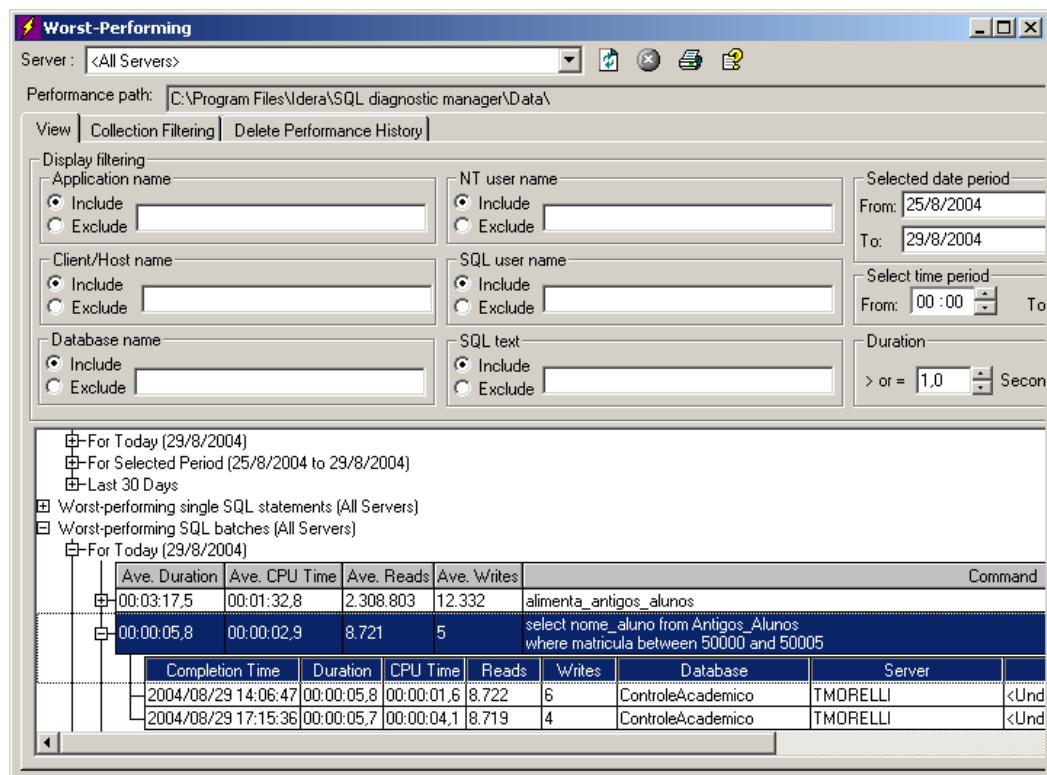


Figura B.8: identificando uma consulta agressiva

Vale lembrar, contudo, que nem o plano de execução, nem sugestões quanto a índices aparecem. A tabela B.6 informa as características atendidas pelo produto:

	Explicação de Planos de Execução	
	Avaliação de custos de comandos	
	Sugestões para mudanças em comandos	
	Sugestões de utilização de Estruturas Auxiliares (Índices, Visões Materializadas)	
	Sugestões de atualizações de estatísticas	
	Lista de comandos mais agressivos, classificados por consumo de recursos (tempo, cpu, leituras físicas, leituras lógicas)	☑
	Apresenta estado atual da memória (dados e comandos)	☑
	Taxa de recompilação de comandos	
	Gerência de Bloqueios	☑
	Ocorrência de Deadlocks (quem foi cancelado)	

Tabela B.6: resumo de funcionalidades do produto Idera SQL Diagnostic Manager

B.5 - Veritas

A californiana Veritas, que recentemente incorporou a **Precise**, divide seus produtos em quatro grandes grupos: **Data Protection** (cópias de segurança), **High Availability** (gerência de grupos de Servidores e replicação), **Storage, Server & Service-Level Automation** (controle de espaço e apoio à administração do Sistema Operacional) e **Application Performance** (otimização de aplicativos). As ferramentas de apoio ao DBA encontram-se neste último grupo, com destaque para **InDepth** (detecta problemas de performance e os corrige), **Inform** (notifica problemas em potencial) e **Insight** (mede tempos de resposta de operações).

O produto a ser estudado em detalhes será o gerenciador de desempenho **VERITAS InDepth**, que possui versões para Oracle, DB2 e SQL Server. Seu grande mérito consiste em oferecer condições para que rapidamente descubram-se razões para más performances. Como são armazenados dados históricos, permitem-se rápidas comparações onde evidenciam-se possíveis distorções.

O produto em questão está organizado segundo uma arquitetura cliente-servidor. No lado cliente estão o instalador de agentes (coletor de informações de performance, entre outros), uma interface gráfica e o gerenciador de agentes. A ordem de instalação dos componentes faz diferença no resultado final, assim, deve-se iniciar instalando a porção cliente, depois a servidora e finalmente invoca-se o instalador de agentes. Além desta ordem, algumas sutilezas devem ser destacadas:

Durante a instalação do lado servidor informa-se uma porta TCP/IP através da qual será estabelecida a comunicação entre os componentes. Esta porta será pedida durante a instalação dos agentes.

A chave de instalação (temporária ou permanente) será pedida durante a instalação dos agentes.

Antes de realizar a instalação, devem ser criados dois Bancos de Dados no Servidor por monitorar: um para armazenar informações temporárias do Coletor (dois mega-bytes bastam) e outro para que sirva como repositório de informações históricas (uns 250 MB).

Criam-se dois novos serviços: **Precise MS SQL Batch Controller** e **Precise MS SQL Listener**.

Uma vez concluídas as instalações das porções servidora e cliente, além dos agentes, pode-se verificá-las ativando o **Agent Manager**, cujo aspecto aparece na Figura B.9

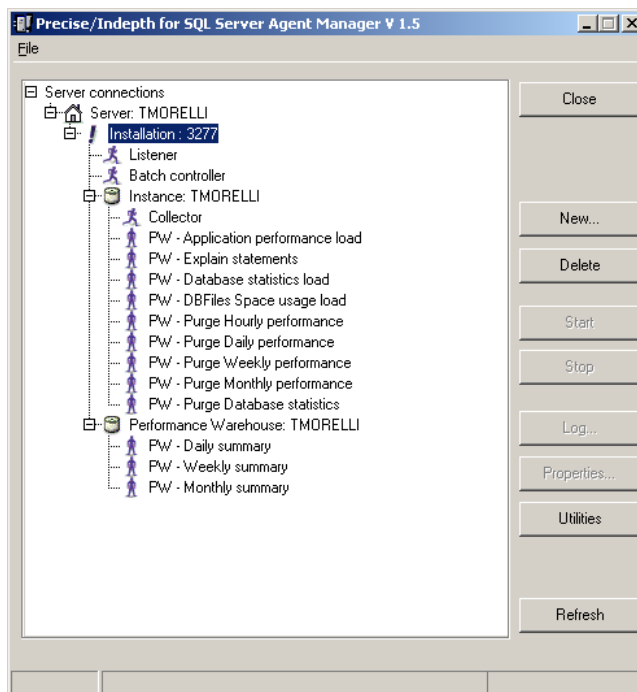


Figura B.9: Agent Manager

E para acompanhar efetivamente o servidor em questão, invoca-se **Precise InDepth for SQL Server**, cuja tela principal aparece na Figura B.10

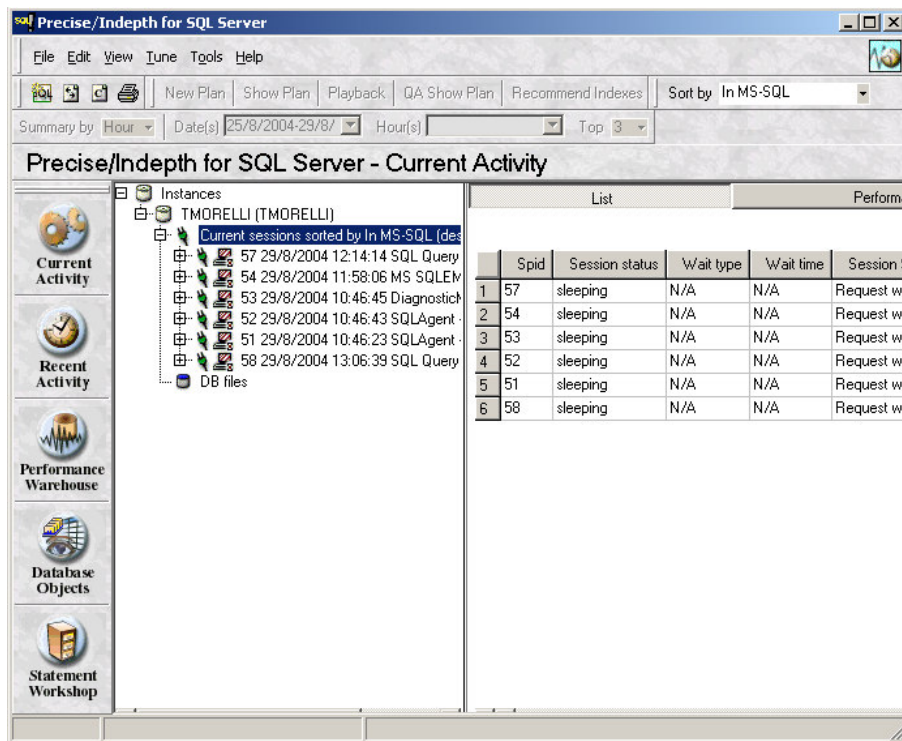


Figura B.10: ambiente da ferramenta InDepth

A Tabela B.7 descreve as principais funcionalidades do ambiente.

Componente (Workspaces)	Descrição
Current Activity	Descreve principais características das conexões e suas atividades no minuto ou período (normalmente uma hora) correntes.
Recent Activity	Mostra o histórico de operações por período. Graças a este recurso, pode-se analisar o desempenho geral passado.
Performance Warehouse	Revela informações consolidadas. Elas são necessárias já que os registros da atividade recente devem ser apagados periodicamente.
Database Objects	Permite navegar pelos objetos dos Bancos de Dados e analisar interessantes características tais como estruturas, níveis de fragmentação ou quais comandos acessam determinada tabela.
Statement Workshop	Comandos SQL organizados hierarquicamente em grupos (cabinets e folders). Provêm das atividades corrente ou recente, ou ainda a partir da digitação manual (comando File, New, SQL Statement).

Tabela B.7: principais elementos da ferramenta InDepth

Suponha uma determinada conexão tivesse executado o segundo comando apresentado pela Figura B.5. Para visualizá-lo deve-se localizar a sessão em **Current Activity** e clicando o botão direito ativa-se **Associate with Recent batches (in bufer)**. Expandindo a árvore recém montada até o final, clicando em **Text** surgiria o comando.

Estando o comando ativo, ativam-se vários botões da barra de ferramentas (**New Plan, Playback, QA Show Plan**). Clicando no primeiro estende-se mais um pouco a árvore mostrando o plano de execução do comando. Como a tabela **ANTIGOS_ALUNOS** não possui índices, ocorreu uma varredura completa (**Table Scan**). Ativando o indicativo desta operação, habilita-se mais um botão: **Recommended Indexes**. Ele não apenas sugere o índice a ser criado, como também oferece o comando SQL necessário para criá-lo. Algo como:

```
CREATE CLUSTERED INDEX [Antigos_Alunos1]
ON [dbo].[Antigos_Alunos] ( [matricula] ASC)
```

A tabela B.8 a seguir informa as características atendidas pelo produto.

	Explicação de Planos de Execução	<input checked="" type="checkbox"/>
	Avaliação de custos de comandos	<input checked="" type="checkbox"/>
	Sugestões para mudanças em comandos	
	Sugestões de utilização de Estruturas Auxiliares (Índices, Visões Materializadas)	<input checked="" type="checkbox"/>
	Sugestões de atualizações de estatísticas	<input checked="" type="checkbox"/>
	Lista de comandos mais agressivos, classificados por consumo de recursos (tempo, cpu, leituras físicas, leituras lógicas)	<input checked="" type="checkbox"/>
	Apresenta estado atual da memória (dados e comandos)	<input checked="" type="checkbox"/>
	Taxa de recompilação de comandos	
	Gerência de Bloqueios	<input checked="" type="checkbox"/>
	Ocorrência de Deadlocks (quem foi cancelado)	

Tabela B.8: resumo de funcionalidades do produto Veritas InDepth

Durante os testes do produto, observou-se uma característica muito estranha. Assim que uma nova conexão era aberta, e atualizando-se a lista em **Current Activity** ou **Recent Activity**, esta conexão surgia com a hora de conexão defasada em uma hora. Em consequência disto, não era possível associá-la aos comandos mais recentes. Infelizmente, a Veritas não oferece nenhum tipo de suporte para versões *trial* do InDepth.