

IDŹ DO

PRZYKŁADOWY ROZDZIAŁ



SPIS TREŚCI

KATALOG KSIĄŻEK

KATALOG ONLINE

ZAMÓW DRUKOWANY KATALOG

TWÓJ KOSZYK

DODAJ DO KOSZYKA

CENNIK I INFORMACJE

ZAMÓW INFORMACJE
O NOWOŚCIACH

ZAMÓW CENNIK

CZYTELNIA

FRAGMENTY KSIĄŻEK ONLINE

Oracle. Optymalizacja wydajności

Autorzy: Ed Whalen, Mitchell Schroeter

Tłumaczenie: Bartłomiej Jabłoński

ISBN: 83-7197-797-2

Tytuł oryginału: [Oracle Performance Tunning](#)

Format: B5, stron: 384



Uzyskanie maksymalnej wydajności złożonego systemu, jakim jest Oracle, to zadanie skomplikowane i trudne. Książka „Oracle. Optymalizacja wydajności”, napisana przez czołowych specjalistów w dziedzinie strojenia wydajności i optymalizacji baz danych, pomoże Ci stawić mu czoła.

Wyjątkowość tej książki polega na jej całościowym podejściu do tematu. Uzyskanie maksymalnej wydajności i dostrojenie bazy Oracle wymaga nie tylko optymalizacji zapytań SQL; należy również zastanowić się nad konfiguracją sprzętu, na którym ma być uruchomiony system, wydajnością podstawowych operacji wykonywanych na twardych dyskach, a także rozważyć, jakie cechy najnowszej wersji Oracle'a 9i mogą być przydatne do przyspieszenia pracy aplikacji.

Książka omawia:

- Podstawowe pojęcia
- Parametry konfiguracyjne Oracle'a
- Strojenie serwera
- Zaawansowane opcje wpływające na wydajność
- Strojenie wydajności przetwarzania
- Perspektywy wydajnościowe
- Skrypty UTLBSTAT i UTLESTAT
- Wpływ sprzętu na Oracle'a
- Operacje wejścia-wyjścia i ich znaczenie dla strojenia serwera
- Wykorzystywanie planu wykonania i śledzenie SQL, strojenie indeksów
- Użycie optymalizatora Oracle'a
- Strojenie zapytań SQL
- Oracle 9i Real Application Clusters
- Strojenie procesu tworzenia kopii zapasowych
- Konfiguracja systemu odpornego na awarie
- Wydajność Oracle'a w sieci

Przy opracowywaniu tej książki autorzy wykorzystali swoją bogatą wiedzę, nie tylko na temat Oracle'a, ale także na temat sprzętu komputerowego i systemów operacyjnych. Skorzystaj z ich doświadczenia i skonfiguruj serwer bazodanowy wydajny, stabilny i odporny na awarie.



Spis treści

	O Autorach	11
	Wstęp	13
Część I	Strojenie instancji	15
Rozdział 1.	Podstawy strojenia	17
	Podstawowe pojęcia	17
	Czym jest strojenie?	17
	Czy strojenie jest konieczne?	18
	Kiedy zaprzestać strojenia?	18
	Cele strojenia	19
	Strojenie wydajności przetwarzania	19
	Strojenie czasu odpowiedzi	19
	Strojenie dla dużej liczby użytkowników	20
	Strojenie niezawodności	20
	Strojenie ładowania danych	21
	Metodologia strojenia	21
	Badanie problemu	22
	Wyznaczenie problemu	24
	Wyznaczenie rozwiązania i ustanowienie celów	24
	Testowanie rozwiązania	26
	Analiza rezultatu	27
	Co wpływa na wydajność serwera Oracle?	27
	Wąskie gardła systemu	28
	Strojenie systemu	29
	Ograniczenia systemowe	31
	Wyznaczanie rozmiaru i pojemności systemu	31
	Różnica między rozmiarem a pojemnością	32
	Etapy wyznaczania rozmiaru systemu	32
	Etapy wyznaczania pojemności	33
	Podsumowanie	33
Rozdział 2.	Parametry konfiguracyjne systemu Oracle	35
	Uruchamianie instancji Oracle	36
	Łączenie się z Oracle	36
	Polecenie STARTUP	37
	Uruchamianie instancji	37
	Usuwanie typowych błędów	38
	Zatrzymywanie instancji	39
	Parametry inicjalizacyjne	42
	Parametry strojenia instancji	43
	Parametry wyznaczające wielkości zasobów	43
	Podsumowanie	44

Rozdział 3. Strojenie serwera Oracle	45
Parametry inicjalizujące wymieniane w niniejszym rozdziale	46
Parametry SGA	46
Parametry obszaru PGA i obszaru użytkownika	47
Parametry Undo	48
Inne parametry	48
Skalowanie SGA	49
Strojenie systemu operacyjnego	49
Strojenie prywatnych obszarów SQL i PL/SQL	50
Strojenie obszaru wspólnego	51
Strojenie bufora danych	56
Zarządzanie segmentami wycofania i informacją wycofania	61
Strojenie serwera w trybie SMU	62
Strojenie serwera w trybie RBU	63
Podsumowanie strojenia obszarem wycofania	69
Bufor dziennika powtórzeń	70
Rywalizacja o bufor dziennika	70
Strojenie punktów kontrolnych	70
Strojenie sortowań	71
Strojenie obszaru SORT AREA	72
Strojenie pozostawianego obszaru sortowania	72
Strojenie obszarem tymczasowym	72
Minimalizacja rywalizacji o listę wolnych bloków	73
Podsumowanie	74
Rozdział 4. Wydajność — opcje zaawansowane	75
Parametry wykorzystywane w tym rozdziale	75
Indeksy	77
Koncepcja indeksu	77
Rodzaje indeksów	78
Indeksy B*-tree	78
Jak działają indeksy bitmapowe	81
Co warto indeksować?	82
Sposób indeksowania	83
Równoległe wykonywanie zapytań w systemie Oracle	85
Wykonywanie równoległe	85
Równoległe tworzenie indeksu	91
Równoległe ładowanie	92
Równoległe odtwarzanie	92
Klastry	93
Klastry haszowe	96
Kiedy haszować?	97
Odczyty wieloblokowe	98
Partycjonowanie	99
Koncepcja partycjonowania	99
Partycjonowanie zakresowe	100
Partycje z listą wartości	101
Partycjonowanie z kluczem haszowym	102
Partycje złożone	102
Korzyści z partycjonowania	103
Partycje a indeksy	104
Stabilność planu	104
Serwer wielokanałowy	104
Serwer dedykowany	105
Serwer wielokanałowy	105
Podsumowanie	107

Rozdział 5. Strojenie wydajności przetwarzania	109
Parametry zaprezentowane w niniejszym rozdziale	109
Korzystanie z mechanizmu Grupy Konsumentek	110
Przegląd Grup Konsumentek	110
Konfiguracja Grup Konsumentek	111
Dodawanie użytkowników do planu	114
Monitorowanie grup konsumentek	115
Strojenie zasobów użytkownika	115
Systemy OLTP	116
Czas odpowiedzi	116
Przenoszenie niektórych funkcji do innego systemu	116
Dystrybucja raportów historycznych	117
Dystrybucja bieżących raportów	118
Obsługa rozproszenia w aplikacjach	118
Podsumowanie	119
Rozdział 6. Perspektywy wydajnościowe systemu Oracle	121
Perspektywy V\$ a perspektywy G\$	122
Przegląd dynamicznych perspektyw wydajnościowych	122
Wykorzystywanie perspektyw	127
Zapytania wykorzystujące perspektywy dynamiczne	127
Skrypty UTLBSTAT/UTLESTAT i pakiet STATSPACK	128
Narzędzia do badania wydajności	128
Podsumowanie	129
Rozdział 7. Skrypty UTLBSTAT i UTLESTAT	131
UTLBSTAT/UTLESTAT	131
Uruchamianie skryptów UTLBSTAT/UTLESTAT	131
Wynik wykonania skryptów UTLBSTAT/UTLESTAT	132
Interpretacja statystyk BSTAT/ESTAT	132
Pakiet STATSPACK	148
Instalowanie pakietu STATSPACK	149
Korzystanie z pakietu STATSPACK	149
Administrowanie pakietem STATSPACK	150
Raport STATSPACK	151
Podsumowanie	166
Część II Strojenie sprzętu komputerowego	167
Rozdział 8. System Oracle i sprzęt komputerowy	169
Parametry opisane w tym rozdziale	169
Instancja Oracle od środka	170
Struktura pamięci	171
Procesy	173
Architektura systemu	175
Procesory i bufory	175
Rodzaje procesorów	176
Procesory 32- i 64-bitowe	179
Architektura pamięci komputera	180
Rodzaje magistral	181
Magistrale wejścia-wyjścia	182
Sieć	182
Klastry	182
Podsumowanie	182

Rozdział 9. Operacje wejścia-wyjścia	183
Dysk twardy.....	183
Przegląd.....	184
Wydajność dysku.....	187
Macierze RAID	190
Macierze sprzętowe i programowe.....	191
Rozcinanie	191
RAID 0	192
RAID 1	192
RAID 10	193
RAID 2	194
RAID 3	195
RAID 4	195
RAID 5	197
Podsumowanie kontroli parzystości.....	198
Przegląd wydajności RAID-ów	199
Optymalizacja wydajności.....	201
Optymalizacja ruchu głowicy.....	201
Bufor kontrolera	202
Sprzętowy XOR	203
Rozmiar paska.....	203
RAID wewnętrzny i zewnętrzny.....	204
Wewnętrzne systemy RAID	204
Zewnętrzne systemy RAID	204
Systemy SAN.....	205
Systemy sieciowego przechowywania danych NAS	207
Podsumowanie	208
Rozdział 10. System Oracle i operacje wejścia-wyjścia	209
Parametry przedstawione w tym rozdziale.....	209
Zależność Oracle od urządzeń wejścia-wyjścia.....	210
Dlaczego opóźnienie odczytu jest ważne.....	211
Opóźnienie zapisu.....	212
Odporność na awarie.....	213
Konfiguracja operacji wejścia-wyjścia w bazie Oracle.....	214
Wydajność, odporność na awarie i koszty	214
Bezpieczeństwo inwestycji.....	214
Strojenie operacji wejścia-wyjścia.....	217
Rywalizacja o dysk	217
Badanie rywalizacji dysków.....	218
Rozwiązywanie problemów rywalizacji o dysk.....	220
Redukcja niepotrzebnych odwołań	224
Migracja rekordów i łańcuchy bloków	225
Dynamiczne rozszerzanie.....	226
Parametry PCTFREE i PCTUSED	227
Przegląd technik zmniejszania liczby operacji wejścia-wyjścia.....	231
Rozmiar bloku.....	231
Bloki różnych rozmiarów.....	233
Fragmentacja.....	234
Podsumowanie	236

Część III	Strojenie aplikacji i zapytań SQL.....	237
Rozdział 11.	Wykorzystywanie planu wykonania i śledzenie SQL.....	239
	Śledzenie SQL.....	240
	Przygotowania do śledzenia	240
	Kontrolowanie śledzenia	240
	Funkcjonalność śledzenia SQL.....	241
	Funkcjonalność TKPROF.....	242
	Interpretowanie raportu śledzenia	244
	Polecenie EXPLAIN PLAN.....	249
	Przygotowanie do analizy planu wykonania.....	249
	Wywołanie EXPLAIN PLAN	250
	Pobieranie wyniku analizy	250
	Rejestracja aplikacji	251
	Podsumowanie	252
Rozdział 12.	Strojenie indeksów	255
	Parametry omówione w tym rozdziale	256
	Rodzaje indeksów.....	257
	Korzystanie z indeksów B*-tree	258
	Co powinno być indeksowane?	259
	Utrzymywanie indeksów.....	261
	Tabele zorganizowane indeksowo	262
	Indeksy bitmapowe.....	262
	Kiedy korzystać z indeksów bitmapowych	263
	Indeksy oparte na funkcji.....	264
	Wskazówki optymalizatora.....	264
	Monitorowanie i analiza indeksów.....	265
	Monitorowanie indeksów.....	265
	Podsumowanie	265
Rozdział 13.	Optymalizator Oracle.....	267
	Co to jest optymalizator?	268
	Jak pracuje optymalizator?.....	268
	Parametry inicjalizujące.....	269
	Metody optymalizowania.....	270
	Korzystanie z pakietu DBMS_STATS.....	272
	Tworzenie tabel statystyk	272
	Zbieranie statystyk	273
	Kasowanie statystyk.....	274
	Odtwarzanie statystyk	274
	Inne funkcje pakietu DBMS_STATS	275
	Praca ze statystykami	275
	Polecenie ANALYZE.....	276
	Wywoływanie polecenia ANALYZE.....	276
	Statystyki słownika danych.....	278
	Przetwarzanie transakcyjne.....	280
	Realizacja polecenia SQL.....	282
	Tworzenie kursora	282
	Parsowanie polecenia	282
	Przygotowanie zapytania SELECT	284
	Zmienne wiązane	284
	Wykonywanie polecenia	284
	Równoległe wykonywanie poleceń.....	285
	Przesyłanie rekordów	285
	Analiza poleceń SQL	286

Projektowanie poleceń SQL.....	287
Pakiety, procedury i funkcje.....	288
Używanie wskazówek.....	288
Podsumowanie.....	289
Rozdział 14. Strojanie poleceń SQL.....	291
Optymalne polecenia SQL.....	291
Jak zidentyfikować niepoprawnie zoptymalizowane polecenia.....	292
Rodzaje złączeń.....	292
Algorytm nested loops.....	292
Algorytm sort-merge.....	293
Algorytm hash join.....	294
Strojanie poleceń SQL.....	294
Strojanie istniejącej aplikacji.....	295
Projektowanie nowych aplikacji.....	299
Podsumowanie.....	304
Rozdział 15. Wskazówki optymalizatora.....	307
Implementacja wskazówek.....	308
Składnia wskazówek.....	308
Błędy stosowania.....	309
Wskazówki wielokrotne.....	309
Wskazówki.....	310
Cele optymalizatora.....	310
Metody dostępu.....	312
Kolejność łączenia.....	316
Operacje łączenia.....	317
Wskazówki zapytań równoległych.....	319
Transformacje zapytań.....	321
Pozostałe wskazówki.....	323
Podsumowanie.....	326
Część IV Zagadnienia zaawansowane.....	327
Rozdział 16. Oracle9i Real Application Clusters.....	329
Przegląd technologii RAC.....	329
Systemy komputerowe.....	331
Podsystem dysku wspólnego.....	331
Połączenie międzyserwerowe.....	331
Blokady.....	332
Konfiguracja klastra.....	333
Kiedy używać konfiguracji RAC.....	338
Strojanie RAC.....	338
Konfiguracja i określanie mocy komputera.....	339
Instancja i strojenie blokad.....	339
Strojanie aplikacji.....	339
Podsumowanie.....	339
Rozdział 17. Strojanie tworzenia kopii zapasowych i odtwarzania.....	341
Parametry wymieniane w tym rozdziale.....	342
Zapis na dysk — przypomnienie.....	343
Tworzenie kopii bezpieczeństwa.....	343
Proces odtwarzania.....	344
Sposoby tworzenia kopii zapasowej.....	344
Ręczne wykonywanie zimnej kopii zapasowej.....	344
Ręczne wykonywanie gorącej kopii zapasowej.....	345
Wykonywanie gorącej kopii zapasowej za pomocą narzędzia RMAN.....	345

Wykonywanie gorącej kopii zapasowej w architekturze SAN	346
Charakterystyka dostępu do danych podczas ręcznego tworzenia kopii zapasowej	346
Charakterystyka dostępu do danych w narzędziu RMAN	347
Obciążenie systemu w czasie tworzenia kopii bezpieczeństwa	347
Cele wykonywania kopii zapasowej	348
Dopasowywanie metody do systemu	348
Wykonywanie zimnej kopii zapasowej bazy	349
Gorąca kopia zapasowa	350
Strojenie ręcznego sporządzania kopii zapasowych	354
Strojenie sporządzania kopii zapasowej wykonywanej narzędziem RMAN	354
Bufor RMAN	354
Synchroniczne i asynchroniczne operacje wejścia-wyjścia	355
Parametry strojenia narzędzia RMAN	355
Monitorowanie pracy narzędzia RMAN	356
Wydajność tworzenia kopii	356
Procesor	356
Operacje wejścia-wyjścia	357
Sieć	358
Podział kopii	358
Weryfikacja wydajności	360
Co można testować na serwerze?	360
Co można testować w systemie operacyjnym?	360
Podsumowanie	363
Rozdział 18. Tworzenie systemu odpornego na awarie	365
Parametry opisane w niniejszym rozdziale	365
Dlaczego należy planować awarie	366
Jak przetrwać awarię?	366
Odległa kopia lustrzana	367
Oracle9i Data Guard	367
Replikacja	369
Planowanie awarii	370
Etap planowania	370
Dokumentacja	371
Scenariusze	372
Odtwarzanie po awarii	373
Strojenie kopii systemów	373
Podsumowanie	374
Rozdział 19. Wydajność systemu Oracle w sieci	375
Architektura sieci	375
Komponenty sprzętowe	376
Protokoły sieciowe	378
Strojenie komponentów sieciowych	379
Strojenie oprogramowania	379
Strojenie Oracle	379
Projekt sieci	380
Rozważania na temat przepustowości	380
Segmentacja sieci	381
Mostki, routery i koncentratory	381
Podsumowanie	382
Dodatki	383
Skorowidz	385

Rozdział 5.

Strojenie wydajności przetwarzania

Jedną z metod strojenia systemu jest strojenie przetwarzania (ang. *workload*). Można to przeprowadzić przypisując poszczególnym zadaniom pewne atrybuty, aby w ten sposób zmniejszyć lub zwiększyć ich wydajność. Do tego celu mogą służyć różne narzędzia:

- ◆ mechanizm zarządzania grupami konsumentkami i zasobami;
- ◆ mechanizm profili użytkowników;
- ◆ mechanizm baz rozproszonych.

Powyższa lista nie jest zamknięta — zawsze można odnaleźć własne rozwiązanie.

Wybór metody strojenia systemu zależy od wyznaczonego celu. Poniżej wymieniono kilka takich przykładowych celów:

- ◆ optymalizacja wydajności dla wszystkich aplikacji i użytkowników;
- ◆ optymalizacja wydajności dla jednego użytkownika lub grupy użytkowników;
- ◆ optymalizacja wydajności dla określonego zadania lub typów zadań;
- ◆ cele związane ze specyfiką aplikacji.

W niniejszym rozdziale zostaną opisane metody optymalizacji systemu pod kątem wydajności przetwarzania.

Parametry zaprezentowane w niniejszym rozdziale

Rozdział 2. *Parametry konfiguracyjne systemu Oracle* zawiera wprowadzenie dotyczące sposobu korzystania z parametrów inicjalizujących. W niniejszym rozdziale zaprezentowano tematy związane z poniższymi parametrami:

- ♦ `RESOURCE_LIMIT` — określa, czy limity zdefiniowane w profilach użytkownika będą stosowane;
- ♦ `RESOURCE_MANAGER_PLAN` — specyfikacja głównego planu dla instancji. Menedżer zasobów wczytuje ten plan, jak również wszystkie plany podrzędne, dyrektywy i grupy konsumenckie.

Korzystanie z mechanizmu Grupy Konsumenckiej

Serwer Oracle umożliwia alokację zasobów przyznanych grupie, do której należy dany użytkownik. Pozwala to na definiowanie grup i podgrup użytkowników oraz na rozdział zasobów pomiędzy te grupy. W systemie Oracle9i funkcjonalność menedżera zasobów została zoptymalizowana pod kątem wzrostu wydajności i pozwala teraz na bardziej precyzyjną kontrolę grup konsumenckich.

Mechanizm grup konsumenckich udostępnia płynny, konfigurowalny system, dzięki któremu można skalować system z dużą dokładnością. Korzystanie z grup konsumenckich pozwala na:

- ♦ przydzielanie minimalnej ilości czasu procesora dla pewnych procesów lub grup procesów;
- ♦ ograniczanie stopnia zrównoleglenia wykonywania zapytań w zależności od właściwości użytkownika lub grupy użytkowników;
- ♦ kontrolowanie długo trwających procesów, tak aby nie przekraczały ustalonych limitów.

W kolejnych podrozdziałach opisano mechanizm grup konsumenckich, sposobu ich konfiguracji. Podano także odpowiedni przykład ich wykorzystania.

Przegląd Grup Konsumenckich

Podstawową zaletą grup konsumenckich jest zdolność do klasyfikowania różnego rodzaju użytkowników. Tym grupom można następnie wyznaczać limity poszczególnych zasobów, których będą mogli używać. Nie jest to równoznaczne z ograniczaniem zasobów dla wykonywania poszczególnych zadań lub typów zadań. Ale też i ograniczanie zasobów dla poszczególnych zadań nie jest wykluczone — można stosować oba te rodzaje limitów.

Z uwzględnieniem tych ograniczeń pewne zadania, jak np. raporty, procesy wsadowe lub też inne, wymagające wielu zasobów, mogą być wykonywane równolegle z innymi, bieżącymi zadaniami. Ich wpływ na wydajność jest jednak ograniczony. Takie rozwiązania stosuje się, aby zaspokoić potrzeby zarówno użytkowników OLTP, jak i tych, którzy wykonują dużo raportów.

Sposób realizowania tego rozwiązania jest wyznaczony przez potrzeby użytkowników i konfigurację bazy danych. W dalszej części niniejszego rozdziału znajdują się przykłady wykorzystania grup konsumenckich.

Konfiguracja Grup Konsumenckich

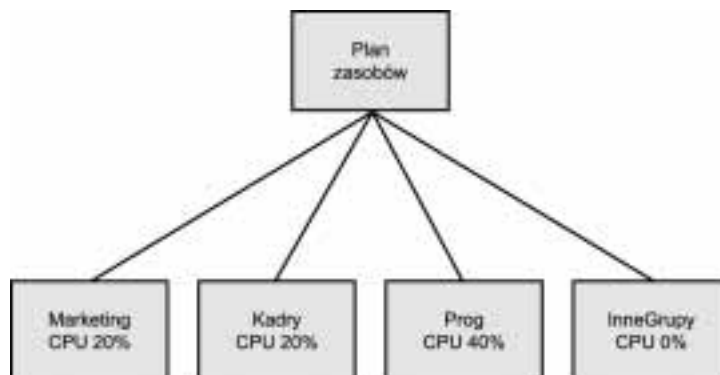
Do zarządzania grupami konsumenckimi służą dwa pakiety: `DBMS_RESOURCE_MANAGER` i `DBMS_RESOURCE_MANAGER_PRIVS`. Parametr `RESOURCE_MANAGER_PLAN` włącza mechanizm kontroli zasobów. Te trzy elementy sterują konfiguracją grup konsumenckich.

Pierwszym etapem konfiguracji jest utworzenie planu zasobów (ang. *resource manager plan*) i wskazanie, że właśnie ten plan ma być wykonywany. Plan zawiera definicje wielu grup i przypisane im limity zasobów, z których te grupy mogą korzystać.

Każdy użytkownik jest członkiem jakiejś grupy. Jedna grupa konsumencka może także być przypisana do innej. Grupy są następnie zarządzane za pomocą planów zasobów. Odpowiedni przykład pokazano na rysunku 5.1, gdzie zaprezentowano schemat rozdziału czasu procesora pomiędzy cztery różne grupy — każdej z nich przydzielono pewien procent czasu.

Rysunek 5.1.

Zasoby rozdzielone pomiędzy cztery grupy



W ramach każdej grupy można wyspecyfikować limit dotyczący:

- ♦ stopnia zrównoleglenia;
- ♦ liczby aktywnych sesji;
- ♦ czasu procesora;
- ♦ maksymalnego czasu wykonania;
- ♦ wielkości dostępnego obszaru wycofania.

Konfiguracja menedżera zasobów polega na utworzeniu planu, jego zmodyfikowaniu i przydziale użytkowników do grup.

Tworzenie planu

Pierwszym etapem tworzenia planu jest utworzenie obszaru tymczasowego (ang. *pending area*). Obszar ten jest wykorzystywany do definiowania planu i jego weryfikacji, zanim zostanie on wykorzystany w systemie. Tworzy się go za pomocą polecenia:

```
EXEC DBMS_RESOURCE_MANAGER.CREATE_PENDING_AREA;
```

Plan tworzy się za pomocą procedury `CREATE_PLAN`, zdefiniowanej w pakiecie `DBMS_RESOURCE_MANAGER`:

```
EXEC DBMS_RESOURCE_MANAGER.CREATE_PLAN(PLAN => 'sample_plan', -  
    COMMENT => 'Sample Plan');
```

Wydanie tego polecenia powoduje utworzenie planu o nazwie `sample_plan`. Przed przypisaniem go do systemu trzeba zdefiniować grupy i limity.

Tworzenie grupy konsumentkiej

Grupę konsumentką tworzy się za pomocą procedury `CREATE_CONSUMER_GROUP`:

```
EXEC DBMS_RESOURCE_MANAGER.CREATE_CONSUMER_GROUP(CONSUMER_GROUP => 'sales', -  
    COMMENT => 'sample sales group');
```

Procedura ta tworzy grupę, ale podobnie jak w poprzednim przypadku, przed jej wdrożeniem należy zdefiniować jej limity.

Ustawienie limitów w grupie

Po utworzeniu grupy można zdefiniować zasady funkcjonowania planu. Dyrektywy planu mogą zawierać ograniczenia na czas procesora, stopień zrównoleglenia, maksymalny czas wykonywania, itd. Przykład definiowania takich reguł pokazano poniżej:

```
EXEC DBMS_RESOURCE_MANAGER.CREATE_PLAN_DIRECTIVE(PLAN => 'sample_plan', -  
    GROUP_OR_SUBPLAN => 'sales', COMMENT => 'sample sales plan', -  
    CPU_P1 => 40, PARALLEL_DEGREE_LIMIT_P1 => 4, -  
    MAX_EST_EXEC_TIME => 3600, UNDO_POOL => 100);
```

Oczywiście definiowanie limitów tylko dla jednej grupy nie ma większego sensu, dlatego też konieczne jest utworzenie innych grup. Podobnie jak grupy można też definiować plany podrzędne. Definicja planu podrzędnego jest podobna do definicji grupy. W poleceniu, zamiast nazwy grupy, podaje się nazwę pod-planu.

Do zakończenia definicji planu, podobnego do tego zaprezentowanego na rysunku 5.1, dodaje się kolejne ograniczenia:

```
EXEC DBMS_RESOURCE_MANAGER.CREATE_PLAN_DIRECTIVE(PLAN => 'sample_plan', -  
    GROUP_OR_SUBPLAN => 'marketing', COMMENT => 'sample marketing plan', -  
    CPU_P1 => 20, PARALLEL_DEGREE_LIMIT_P1 => 4, -  
    MAX_EST_EXEC_TIME => 3600, UNDO_POOL => 100);
```

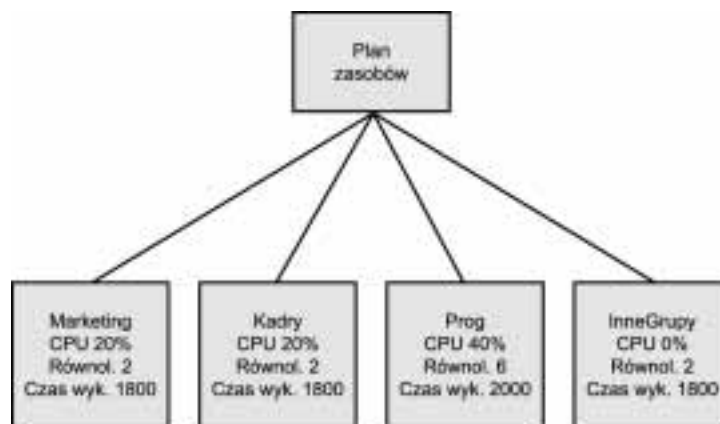
```
EXEC DBMS_RESOURCE_MANAGER.CREATE_PLAN_DIRECTIVE(PLAN => 'sample_plan', -  
    GROUP_OR_SUBPLAN => 'hr', COMMENT => 'sample hr plan', -  
    CPU_P1 => 20, PARALLEL_DEGREE_LIMIT_P1 => 4, -  
    MAX_EST_EXEC_TIME => 3600, UNDO_POOL => 100);
```

```
EXEC DBMS_RESOURCE_MANAGER.CREATE_PLAN_DIRECTIVE(PLAN => 'sample_plan', -
  GROUP_OR_SUBPLAN => 'OTHER_GROUPS', COMMENT => 'required group', -
  CPU_P1 => 0, PARALLEL_DEGREE_LIMIT_P1 => 4, -
  MAX_EST_EXEC_TIME => 3600, UNDO_POOL => 100);
```

Przykładowy sposób rozdzielania zasobów między cztery grupy przedstawiono na rysunku 5.2.

Rysunek 5.2.

*Szczegółowy
plan rozdzielania
zasobów pomiędzy
cztery grupy*



Jak wcześniej wspomniano, plan uwzględniany przez system Oracle jest określony w parametrze inicjalizacyjnym, ale parametry te są odczytywane tylko przy starcie instancji. Aby plan można było włączyć bezpośrednio po jego utworzeniu (bez zatrzymywania i ponownego uruchamiania instancji), należy posłużyć się poleceniem ALTER SYSTEM. Wydanie poniższego polecenia spowoduje natychmiastowe wskazanie planu sample_plan:

```
ALTER SYSTEM SET RESOURCE_MANAGER_PLAN = sample_plan;1
```

Jeżeli zachodzi potrzeba wyłączenia menedżera zasobów, wykonuje się podobne polecenie, zamiast nazwy planu podając wartość NULL:

```
ALTER SYSTEM SET RESOURCE_MANAGER_PLAN = '';
```

Polecenie to wyłącza menedżera i wprowadza bazę w zwykły tryb pracy.

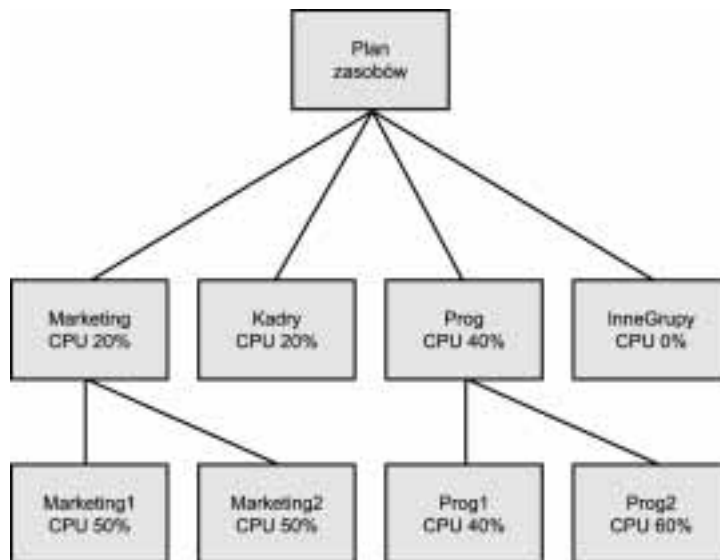
Plany podrzędne mogą być tworzone i podpinane do planów za pomocą tych samych poleceń. Plan podrzędny tworzy się tak samo jak plan zwykły, ale przy specyfikacji ograniczeń definiuje się je dla planu, a nie dla grupy. Jak łatwo sobie to wyobrazić, plan może być bardzo skomplikowany. Przykładowy schemat takiego planu zaprezentowano na rysunku 5.3.

Zarządzanie zasobami za pomocą grup konsumenckich jest użyteczną metodą, ale jak każde zaawansowane narzędzie wymaga ostrożności i zwracania szczególnej uwagi na możliwe niepożądane skutki.

¹ Brakuje tutaj wcześniejszego zatwierdzenia zmian dokonanych w obszarze pending area: EXEC DBMS_RESOURCE_MANAGER.SUBMIT_PENDING_AREA; — *przyt. tłum.*

Rysunek 5.3.

Plan zasobów
i podplanami



Dodawanie użytkowników do planu

Aby umieszczać użytkowników w grupach, najpierw nadaje się im uprawnienia a następnie dodaje się ich do określonej grupy. Dodanie użytkownika do grupy powoduje, że podczas każdej swojej nowej sesji korzysta on z limitów tej grupy. Poniżej znajduje się przykładowe polecenie, powodujące przydzielenie użytkownika scott do grupy sales_group:

```
EXEC DBMS_RESOURCE_MANAGER_PRIVS.GRANT_SWITCH_CONSUMER_GROUP -
('scott', 'sales', TRUE);
EXEC DBMS_RESOURCE_MANAGER.SET_INITIAL_CONSUMER_GROUP('scott', 'sales');
```

Można również, za pomocą procedury SWITCH_CONSUMER_GROUP_FOR_SESS, wprowadzić do grupy sesję aktywną. Należy tylko najpierw określić identyfikator (SID) i numer seryjny przełączanej sesji.

```
EXEC DBMS_RESOURCE_MANAGER.SWITCH_CONSUMER_GROUP_FOR_SESS -
('17', '12345', 'sales');
```

Identyfikator i numer seryjny sesji można uzyskać przeglądając perspektywę V\$SESSION.

Po zdefiniowaniu planu, grup, limitów zasobów i po przydzieleniu użytkowników do grup można przystąpić do dalszej pracy.

Uwaga na niepożądane skutki uboczne

Stosowanie zaawansowanych metod strojenia jest skomplikowanym zadaniem i może skutkować nieprzewidzianymi konsekwencjami. Należy bardzo uważać wykorzystując takie możliwości. Zaleca się przetestowanie działania zmodyfikowanej konfiguracji na oddzielnym, nieprodukcyjnym serwerze.

Monitorowanie grup konsumenckich

Nie ma potrzeby monitorowania grup konsumenckich, aczkolwiek warto zapoznać się z pewnymi informacjami na temat sposobu dystrybuowania zasobów w systemie. Najłatwiej można to sprawdzić przeglądając perspektywę `V$RSRC_CONSUMER_GROUP`. Poniżej zaprezentowano przykładowe zapytanie, które wyszukuje odpowiednie informacje:

```
SELECT name, active_sessions, consumed_cpu_time, queue_length
FROM v$rsrc_consumer_group;
```

NAME	ACTIVE_SESSIONS	CONSUMED_CPU_TIME	QUEUE_LENGTH
SYS_GROUP	1	1943	0
OTHER_GROUPS	0	1083	0
LOW_GROUP	0	0	0

Jeżeli zbyt dużo czasu procesora przydzielono grupie `other_group`, należy do grup przydzielić większą liczbę użytkowników.

Strojenie zasobów użytkownika

Strojenie indywidualnych zasobów użytkownika jest kolejną możliwością wpływania na wydajność systemu. Robi się to poprzez tworzenie profili i przypisywanie ich poszczególnym użytkownikom. Każdy użytkownik powinien posiadać zdefiniowany profil. Jeżeli nie zrobiono tego jawnie, to i tak użytkownikom domyślnie jest przypisywany profil `DEFAULT`.

Poprzez profile tworzy się klasy użytkowników. Uprawnienia i limity zasobów wyznacza się dla profili, a te z kolei przypisuje się użytkownikom. Można definiować następujące atrybuty profilu:

Atrybut	Komentarz
<code>SESSIONS_PER_USER</code>	Liczba sesji, którą każdy użytkownik może nawiązać jednocześnie. Gdy limit ten zostanie osiągnięty, próba nawiązania nowego połączenia zakończy się wywołaniem błędu.
<code>CPU_PER_SESSION</code>	Limit czasu wyrażony w 1/100 s. Po upływie tego czasu sesja będzie przerywana.
<code>CPU_PER_CALL</code>	Limit czasu wyrażony w 1/100s. Tyle czasu może trwać każde odwołanie (ang. <i>call</i>) w sesji.
<code>CONNECT_TIME</code>	Całkowity czas połączenia wyrażony w minutach, po upływie którego sesja użytkownika będzie przerywana.
<code>IDLE_TIME</code>	Maksymalny czas wyrażony w minutach, w ciągu którego sesja może pozostawać nieaktywna.
<code>LOGICAL_READS_PER_CALL</code>	Limit logicznych odczytów danych dostępnych w ramach jednego odwołania.
<code>LOGICAL_READS_PER_SESSION</code>	Limit logicznych odczytów danych dostępnych w ramach jednej sesji.
<code>COMPOSITE_LIMIT</code>	Limit kombinacji ważonej kilku zasobów ² .

² Kombinacja ta jest ustalana za pomocą polecenia `ALTER RESOURCE COST` — *przyj. tłum.*

W profilu można skonfigurować więcej parametrów. Tutaj opisane są tylko te związane z wydajnością i ograniczaniem dostępu do zasobów. Poniżej zaprezentowano przykładowe polecenie tworzenia profilu.

```
CREATE PROFILE sales_user LIMIT
  SESSIONS_PER_USER      2
  CPU_PER_SESSION        UNLIMITED
  CPU_PER_CALL            3000
  CONNECT_TIME           45
  IDLE_TIME               5
  LOGICAL_READS_PER_SESSION DEFAULT
  LOGICAL_READS_PER_CALL 1000;
```

Po utworzeniu profilu można przypisać do niego poszczególnych użytkowników.

Wykorzystanie mechanizmu profili do ograniczania ilości dostępnych zasobów dla poszczególnych sesji jest kolejną metodą strojenia wydajności przetwarzania systemu. Może być to użyteczne w wielu sytuacjach.

Systemy OLTP

Typowy system OLTP składa się z wielu podłączonych użytkowników, wykonujących liczne, krótkie operacje i żądających uzyskiwania szybkiej odpowiedzi. Każdy długo trwający proces wsadowy lub duże zapytanie ogranicza wydajność systemu. Dzięki wykorzystaniu mechanizmu profili i ograniczeniu długich operacji wydajność całego systemu może się poprawić. Długie zapytania mogą być wykonywane za pomocą konta użytkownika, który jako jedyny może je wykonywać po godzinach pracy lub pracując na innym systemie.

Czas odpowiedzi

Profile są wykorzystywane do organizowania środowisk, gdzie są dozwolone tylko pewne typy operacji. Profile składają się z limitów, a zatem skutecznie ograniczają wykonywanie długich operacji, które te limity przekraczają. Można przyspieszyć działanie krótkich operacji eliminując z systemu te, które trwają bardzo długo. To może ale nie musi zwiększyć wydajność systemu, ale na pewno w większości przypadków czas uzyskiwania odpowiedzi będzie krótszy.

Przenoszenie niektórych funkcji do innego systemu

Innym wspnianym rozwiązaniem umożliwiającym zwiększenie wydajności przetwarzania jest przeniesienie funkcji do innego systemu. Systemy, w których jednocześnie wykonuje się transakcje OLTP i bardzo długie zapytania, mogą być mało wydajne z uwagi na efekt wykonywania długich zapytań. Jeżeli jest możliwość pozostawienia wszystkich

funkcji OLTP i przeniesienia długich zapytań na inny serwer, może to jednocześnie zwiększyć wydajność obu systemów.

Istnieje wiele sposobów przeprowadzenia takiej operacji w zależności od indywidualnych potrzeb i możliwości systemu. W dalszej części niniejszego rozdziału przedstawiono kilka rozwiązań.

Dystrybucja raportów historycznych

Raporty historyczne, które swoim zakresem obejmują miesiące, a nierzadko lata, często nie muszą uwzględniać danych wygenerowanych przed sekundą. Zazwyczaj takie raporty mogą być uruchamiane na starych danych. W takich sytuacjach istnieje kilka sposobów na utworzenie serwera raportów. Serwer raportów może się opierać na kopiach bezpieczeństwa lub *Oracle Data Guard* (dawniej znanym jako *Standby Database*).

Raportowanie z użyciem kopii zapasowych

W przypadku raportów, które mogą bazować na danych pochodzących sprzed kilku dni, można użyć kopii zapasowych, które mogą być odtwarzane na serwerze raportów. W ciągu dnia taki serwer może pracować na tych danych wiedząc, że w ciągu nocy zostaną one uaktualnione. Jest to dobra metoda organizowania pracy serwera raportów pod warunkiem, że rozmiar bazy umożliwia wykonywanie kopii bezpieczeństwa oraz jej zapisywanie na serwerze raportowym w ciągu nocy. Takie rozwiązanie ma kilka zalet:

- ♦ właściwy serwer OLTP nie jest przeciążany. W większości przypadków takie kopie zapasowe i tak są wykonywane;
- ♦ serwer raportów jest dedykowany specjalnie do sporządzania raportów i nie wpływa negatywnie na wydajność serwera OLTP;
- ♦ rozwiązanie to jest bardzo proste i eleganckie. Charakteryzuje się wysoką bezawaryjnością.

Metoda wykorzystania kopii zapasowych nie jest odpowiednia dla wszystkich sytuacji. Rozmiar bazy musi umożliwiać wykonywanie kopii zapasowych w ciągu nocy. Procedura ta nie może wpływać na wydajność serwera OLTP. Dodatkowo, funkcje raportujące mogą tylko odczytywać dane, ponieważ każdej nocy dane i tak są odtwarzane od nowa.

Raportowanie z użyciem Data Guard

Zasada działania raportowania z zastosowaniem systemów *Data Guard* lub *Automated Standby-Database* jest podobna do tej omówionej poprzednio. Różnica polega na tym, że dane są bardziej aktualne. Wymaga to od serwera raportów ciągłego odczytywania danych. Pochodzą one z archiwalnych plików dziennika powtórzeń i serwer odczytuje je w miarę ich generowania. Taki serwer raportowy będzie zawsze nieco opóźniony w stosunku do systemu OLTP, ale dla większości zastosowań takie opóźnienia są dopuszczalne. Podobnie jak w przypadku raportowania z zastosowaniem kopii zapasowych można wyszczególnić kilka zalet omawianego rozwiązania:

- ♦ obciążenie głównego serwera OLTP wzrasta tylko nieznacznie. Generowanie archiwalnych plików dziennika zachodzi niezależnie od dalszych losów tych plików³;
- ♦ serwer raportów jest dedykowany specjalnie do generowania raportów i nie obniża wydajności serwera OLTP;
- ♦ dodatkową korzyścią jest ochrona systemu OLTP przed awariami — system *Data Guard* powstał właśnie w tym celu;
- ♦ wdrożenie tego rozwiązania jest nieco bardziej skomplikowane w porównaniu do omówionego poprzednio, ale jest to tylko niewielki wzrost trudności.

Należy jednak mieć na uwadze, że takie rozwiązanie nie będzie odpowiednie w każdej sytuacji, ale tym razem nie istnieje konieczność sporządzania kopii bezpieczeństwa każdej nocy. Dodatkowo funkcje raportujące muszą znajdować się w trybie tylko do odczytu, ponieważ system i tak jest ciągle zasilany nowymi danymi.

Powyższe rozwiązania są dobre w systemach, gdzie konieczne jest wykonywanie raportów historycznych. Jeśli natomiast konieczne jest sporządzanie raportów na podstawie danych bieżących, trzeba wdrożyć inne rozwiązania.

Dystrybucja bieżących raportów

Raporty bieżące są raportami, których rezultat musi być otrzymywany w jak najkrótszym terminie. Takie raporty wykonują się dłużej niż normalne transakcje OLTP, a do ich sporządzenia są konieczne najświeższe dane. W przypadku potrzeby generowania takich raportów jedynym rozwiązaniem wydaje się być replikacja. Rozwiązanie to daje zwłokę serwera raportów rzędu zaledwie paru sekund w stosunku do serwera głównego OLTP.

Proces replikacji pozwala na ciągły przesył danych z serwera głównego do raportowego, utrzymując obydwa w ścisłej synchronizacji. W specyficznych sytuacjach, jeśli dane serwera raportowego miałyby być modyfikowane, można zastosować technikę replikacji wielobazowej (ang. *multi-master replication*). Taka replikacja pozwala wielu systemom działać jako baza podstawowa, a dane są wtedy przesyłane do pozostałych „kopii”.

Obsługa rozproszenia w aplikacjach

Po przeniesieniu raportów na inny serwer należy jeszcze zmodyfikować aplikację, tak aby możliwe było odszukiwanie wygenerowanych raportów. W tym celu można dokonać zmian bezpośrednio w kodzie aplikacji. Innym sposobem jest utworzenie specjalnej tabeli, w której pojawiłyby się wpisy dotyczące miejsca lokalizacji danego raportu. Korzystając z takiej tabeli łatwo jest rozproszyć wykonywanie raportów pomiędzy kilka serwerów lub wykonywać te raporty na serwerze głównym OLTP, jeśli tamte akurat nie pracują. Zastosowanie tej tabeli jest rozwiązaniem elastycznym.

³ Tylko jeżeli baza pracuje w trybie ARCHIVELOG — *przypr. tłum.*

Podsumowanie

W niniejszym rozdziale zaprezentowano najczęściej spotykany rodzaj strojenia: strojenie przetwarzania. Nie jest rzadkością wykorzystywanie wielu z prezentowanych tutaj rozwiązań, w zależności od rodzaju funkcji spełnianych przez serwer. Rozwiązaniami opisanymi w tym rozdziale są:

- ♦ wykorzystanie grup konsumenckich i menedżera zasobów;
- ♦ wykorzystanie profili użytkowników;
- ♦ wykorzystanie rozproszenia systemu.

Prawdopodobnie każdy administrator wykonuje takie zadanie w inny sposób, ale zasada działania jest taka sama: należy dążyć do ograniczenia wykonywania pewnych zadań, tak aby pozostałe otrzymały więcej zasobów i mogły się wykonywać bardziej efektywnie. W następnych rozdziałach przedstawiono bardziej tradycyjne metody badania statystyk wydajnościowych.