

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

Fedora Core 2. Księga eksperta

Autor: Bill Ball, Hoyt Duff

Tłumaczenie: Grzegorz Kowalczyk (wstęp, rozdz. 1 – 11,
14, 15), Przemysław Szeremiota (rozdz. 16 – 27, dod. A – C)

ISBN: 83-7361-553-9

Tytuł oryginału: [Red Hat Linux and Fedora Unleashed](#)

Format: B5, stron: 1080



Kompletny przewodnik po systemie Fedora Core

- Zainstaluj system i poprawnie nim administruj
- Skonfiguruj serwer WWW, FTP, poczty elektronicznej i plików
- Przekompiluj jądro systemu
- Wykorzystaj mechanizmy bezpieczeństwa do ochrony danych i użytkowników

Fedora Core to najnowsza, bezpłatna i otwarta kontynuacja linii systemu Red Hat Linux, najpopularniejszej dystrybucji systemu Linux na świecie. Jej poprzednik – Red Hat Linux – zyskał zasłużoną sławę i popularność nie tylko wśród komputerowych guru, ale także „zwykłych” użytkowników. Docenili jego stabilność, łatwość obsługi i ogromny wybór różnego rodzaju aplikacji, dostępnych nieodpłatnie, podobnie jak sam system. Fedora Core posiada te same zalety. Można zainstalować ten system nie tylko na komputerze PC, ale także na serwerach sieciowych, urządzeniach PDA i komputerach klasy mainframe.

„Fedora Core 2. Księga eksperta” to kompendium wiedzy na temat systemu Fedora Core 2 Linux. Zawiera szczegółowe informacje na temat jego instalacji, używania i administrowania nim. Opisuje sposoby konfigurowania systemu oraz korzystania z niego zarówno na stacjach roboczych, jak i na serwerach sieciowych. Przedstawia zasady używania środowiska tekstowego i graficznego, konfigurowania usług systemowych, zarządzania kontami użytkowników i wiele innych informacji cennych zarówno dla użytkownika, jak i dla administratora.

- Instalacja i konfigurowanie systemu
- System plików, logowanie i przydzielanie praw użytkownikom
- Korzystanie ze środowiska X Window
- Zarządzanie usługami systemowymi
- Instalowanie oprogramowania
- Administrowanie kontami użytkowników i systemem plików
- Tworzenie kopii bezpieczeństwa
- Konfiguracja sieci i drukarek
- Instalacja i konfigurowanie serwera DNS, WWW, FTP i poczty elektronicznej
- Usługi baz danych
- Tworzenie skryptów powłoki oraz oprogramowania w językach C++ oraz Perl
- Konfigurowanie jądra systemu
- Aplikacje biurowe, multimedialne i emulatory innych systemów operacyjnych

Wydawnictwo Helion
ul. Chopina 6
44-100 Gliwice
tel. (32)230-98-63
e-mail: helion@helion.pl



Spis treści

O Autorach.....	25
Wprowadzenie	27
Część I Instalacja i konfiguracja	41
Rozdział 1. Wprowadzenie do systemu Fedora	43
Fedora Core? Co to takiego?	45
Wewnątrz systemu Fedora	46
Co warto wiedzieć przed zainstalowaniem systemu Fedora	49
Cechy systemu plików systemu Fedora.....	51
System Fedora w zastosowaniach biznesowych	52
System Fedora w zastosowaniach domowych	54
Jak korzystać z dokumentacji systemu Fedora oraz Red Hat Linux	55
Współpraca przy tworzeniu projektu Fedora oraz jego dokumentacji	57
Warto zajrzeć.....	58
Rozdział 2. Przygotowania do instalacji systemu Fedora	61
Planowanie instalacji systemu Fedora	62
Zagadnienia biznesowe	62
Zagadnienia systemowe	63
Zagadnienia dotyczące użytkowników systemu	65
Lista kontrolna zagadnień preinstalacyjnych.....	65
Planowanie instalacji.....	67
Wymagania sprzętowe systemu Fedora.....	68
Minimalne wymagania sprzętowe systemu Fedora Core 2	68
Wykorzystywanie starszego sprzętu.....	69
Planowanie wykorzystania zasobów dysku twardego w zależności od klasy instalacji systemu Fedora	70
Kontrola kompatybilności sprzętu.....	71
Przygotowania do rozwiązywania potencjalnych problemów ze sprzętem	73
Przygotowanie i zastosowanie listy zasobów sprzętowych	77
Przygotowania do procesu instalacji.....	78
Przygotowania do instalacji systemu z dysków CD-ROM.....	83
Tworzenie nośnika startowego ze sterownikami	84

Podział dysku na partycje przed i w trakcie instalacji	84
Wybieranie schematu partycjonowania dysku twardego	86
Rozmieszczanie poszczególnych elementów systemu plików Linux na różnych partycjach	86
Instalacja systemu przy użyciu znanej z systemu Red Hat metody kickstart	88
Warto zajrzeć	90

Rozdział 3. Instalacja systemu Fedora Core 93

Nim rozpoczniesz instalację	93
Przegląd posiadanego sprzętu komputerowego	93
Wybór rodzaju instalacji	94
Wybór opcji instalacji oprogramowania	95
Planowanie strategii podziału dysków na partycje	95
Program rozruchowy (ang. boot loader)	96
Wybór sposobu instalacji systemu Fedora	97
Instalacja z dysku CD-ROM	98
Tworzenie startowych dysków instalacyjnych	99
Instalacja z wykorzystaniem partycji dysku twardego	100
Instalacja z wykorzystaniem sieci komputerowej	100
Instalacja krok po kroku	102
Rozpoczęcie instalacji	102
Podział dysku twardego na partycje	108
Wybór, konfiguracja i instalacja programu rozruchowego (ang. boot loader)	111
Konfiguracja połączeń sieciowych	113
Firewall i konfiguracja zabezpieczeń	114
Wybór dodatkowych języków systemu	115
Ustawienia strefy czasowej	115
Ustawienia hasła dla użytkownika root i tworzenie kont użytkowników	116
Wybór i instalacja oprogramowania	118
Zakończenie instalacji	119
Logowanie i zamykanie systemu	120
Warto zajrzeć	121

Rozdział 4. Fedora zainstalowana — co dalej? 123

Rozwiązywanie problemów z konfiguracją systemu po zakończeniu procesu instalacji	123
Kudzu kontra Twój komputer	125
Konfiguracja klawiatury i urządzeń wskazujących	127
Konfiguracja klawiatury w systemie Linux	127
Konfiguracja urządzeń wskazujących w systemie Fedora	131
Konfiguracja karty graficznej	136
Konfiguracja karty dźwiękowej	138
Konfiguracja nowej karty dźwiękowej	139
Wykrywanie i konfiguracja modemów	139
Konfiguracja modemów podłączanych przez port szeregowy	140
Zastosowanie polecenia minicom do pracy z modemem	143
Konfiguracja modemów typu controllerless (winmodemów) do pracy z laptopami	143
Konfiguracja mechanizmu zarządzania energią	144
APM	144
ACPI	147

Ustawianie daty i czasu	152
Zastosowanie polecenia date	153
Zastosowanie polecenia hwclock	153
Zastosowanie polecenia system-config-date	154
Zarządzanie urządzeniami PCMCIA	155
Zastosowanie magistrali PCMCIA	155
Sterowanie usługą PCMCIA	157
Konfiguracja i używanie napędów CD, DVD oraz CD-RW	158
Kontrola przypisania napędu	158
Inicjalizacja napędów IEEE1394 CD	159
Warto zajrzeć	163

Rozdział 5. Pierwsze kroki z systemem Fedora Core..... 165

Praca z systemem plików systemu Linux	166
Przeglądanie systemu plików systemu Linux	167
Korzystanie z podstawowych poleceń zawartych w katalogach /bin oraz /sbin	169
Składowanie jądra startowego i przeglądanie urządzeń w katalogach /boot i /dev	170
Korzystanie i edycja plików zawartych w katalogu /etc	170
Ochrona zawartości katalogów użytkowników — /home	173
Wykorzystywanie zawartości katalogu /proc do obsługi jądra systemu	173
Katalog /usr — oprogramowanie współdzielone	175
Katalog /tmp do przechowywania plików tymczasowych	175
Katalog /var — pliki „różne”	176
Logowanie i praca z systemem Linux	176
Logowanie za pomocą konsoli trybu tekstowego	176
Używanie konsoli wirtualnych	177
Podstawowe techniki używania klawiatury i myszy podczas pracy z konsolą systemu Linux	178
Wylogowanie się	180
Logowanie i wylogowanie ze zdalnego komputera	180
Zmiana informacji o użytkowniku	182
Korzystanie z dokumentacji	183
Używanie stron podręcznika man	184
Wyszukiwanie i przeglądanie dokumentacji pakietów oprogramowania	184
Używanie powłoki	185
Korzystanie ze zmiennych środowiskowych	187
Przemieszczanie się i wyszukiwanie z poziomu powłoki	189
Zarządzanie plikami z poziomu powłoki	190
Kompresja i dekompresja plików z poziomu powłoki	191
Korzystanie z edytorów tekstu	192
Korzystanie z edytora vi i vim	193
Korzystanie z programu emacs	195
Zarządzanie prawami dostępu	195
Przydzielanie praw dostępu	197
Prawa dostępu do katalogu	198
Korzystanie z praw SUID (Set User ID) oraz SGID (Set Group ID)	200
Użytkownik root i jego zadania	201
Tworzenie kont użytkowników	203
Usuwanie kont użytkowników	203
Zamykanie systemu	204
Przeładowanie systemu	205
Warto zajrzeć	206

Rozdział 6. Środowisko graficzne X Window System..... 207

Podstawowe pojęcia związane z systemem X Window	208
Korzystanie z systemu X.Org	210
Składniki pliku konfiguracyjnego X.Org	211
Konfiguracja systemu X Window	216
Uruchamianie systemu X Window	221
Korzystanie z menedżera ekranu.....	221
Uruchamianie systemu X Window z poziomu konsoli za pomocą polecenia startx	224
Korzystanie z menedżera okien X Window	226
Korzystanie z programu switchdesk.....	227
Tab Window Manager.....	228
Motif Window Manager.....	228
sawfish Window Manager.....	229
Metacity Window Manager.....	231
Środowiska graficzne GNOME oraz KDE	232
GNOME: Obiektowy model środowiska sieciowego GNU	233
KDE: Środowisko graficzne KDE.....	234
Warto zajrzeć.....	236

Część II Zarządzanie systemem Fedora 237**Rozdział 7. Zarządzanie usługami systemowymi..... 239**

Jak przebiega proces uruchamiania systemu Fedora Core 2 Linux	240
Inicjalizacja procesu uruchamiania systemu.....	240
Ładowanie jądra systemu Linux.....	241
Usługi systemowe oraz poziomy uruchamiania.....	242
Definicje poszczególnych poziomów uruchamiania systemu	243
Uruchamianie systemu Fedora Core 2 na domyślnym poziomie uruchamiania	244
Uruchamianie systemu Fedora Core na wybranym poziomie uruchamiania z wykorzystaniem programu ładującego GRUB	247
Tajemnice skryptów init oraz końcowa faza inicjalizacji systemu	248
Uruchamianie usług systemowych przez demona xinetd	249
Sterowanie usługami podczas uruchamiania systemu operacyjnego	250
Zastosowanie polecenia chkconfig	250
Narzędzia konfiguracyjne wykorzystujące pełny, graficzny interfejs użytkownika	253
Manualne zatrzymywanie i uruchamianie usług systemowych	254
Zmiana poziomów uruchamiania.....	255
Rozwiązywanie problemów z usługami systemowymi	256
Warto zajrzeć.....	257

Rozdział 8. Zarządzanie oprogramowaniem i zasobami systemu 259

Używanie polecenia RPM do zarządzania oprogramowaniem.....	260
Tekstowy i graficzny klient RPM.....	261
Korzystanie z polecenia rpm z poziomu wiersza poleceń	263
Organizacja pakietów w systemie RPM	266
Wyodrębnianie jednego pliku z pakietu RPM.....	266
Zarządzanie pakietami w środowisku graficznym.....	267
Korzystanie z Red Hat Network i alternatywnych metod zarządzania oprogramowaniem	268
APT	270
YUM	271

Kompilacja kodu źródłowego aplikacji	273
Przygotowywanie pakietów RMP z plików źródłowych src.rpm	273
Praca z plikami źródłowymi RPM	275
Kompilacja na podstawie źródłowych archiwów tar	276
Narzędzia do monitorowania systemu	277
Monitorowanie systemu z poziomu konsoli	279
Korzystanie z polecenia kill do sterowania procesami	280
Korzystanie i sterowanie priorytetami	281
Wyświetlanie informacji o zajętej i dostępnej pamięci za pomocą polecenia free	283
Limitowanie dostępnej przestrzeni dyskowej	284
Graficzne narzędzia do zarządzania procesami i systemem	284
Narzędzia do monitorowania procesów i systemu dla środowiska KDE	287
Warto zajrzeć	288

Rozdział 9. Zarządzanie kontami użytkowników 289

Konta użytkowników	289
Identyfikator użytkownika (UID) oraz identyfikator grupy (GID)	292
Prawa dostępu do plików	292
Zarządzanie grupami użytkowników	294
Narzędzia do zarządzania grupami użytkowników	295
Zarządzanie kontami użytkowników	297
Narzędzia przeznaczone do zarządzania kontami użytkowników	298
Dodawanie nowych użytkowników	299
Monitorowanie poczynań użytkowników systemu	301
Zarządzanie systemem hasel	302
Podstawowe założenia systemu hasel	302
Plik hasel	303
Cieniowanie hasel (ang. shadow passwords)	304
Zarządzanie bezpieczeństwem hasel	308
Wsadowa zmiana hasel	308
Nadawanie zwykłym użytkownikom praw administratora systemu	309
Tymczasowe przełączenie konta użytkownika przy użyciu polecenia su	309
Nadawanie użytkownikom praw do wykonywania wybranych poleceń z poziomu użytkownika root — polecenie sudo	311
Kontrola dostępu za pomocą powłoki okrojonej	314
Proces logowania użytkownika	315
Limitowanie ilości dostępnego miejsca na dyskach	317
Implementacja systemu limitów dyskowych	318
Manualna konfiguracja limitów dyskowych	318
Warto zajrzeć	319

Rozdział 10. Zarządzanie systemem plików 323

Podstawy systemu plików Fedora Core 2	323
Fizyczna struktura systemu plików na dysku	326
Partycje systemu plików	326
Sieciowe i dyskowe systemy plików	328
Przeglądanie systemów plików komputera	329
System plików ext3	330
Struktura systemu plików ext3	330
Opcje dziennika w systemie ext3	331
Weryfikacja spójności plików w systemie ext3 przy użyciu narzędzia fsck	333

Inne systemy plików dostępne w systemie Fedora Core 2.....	334
System plików Reiser.....	334
Systemy plików JFS oraz XFS.....	335
Systemy plików MS-DOS.....	335
Systemy plików CD-ROM.....	336
Tworzenie systemu plików.....	336
Dysk jako urządzenie pamięci masowej.....	337
Tworzenie tablicy partycji.....	338
Tworzenie systemu plików na partycjonowanym dysku.....	341
Tworzenie systemu plików DOS za pomocą polecenia mkdosfs.....	344
Montowanie systemu plików.....	345
Polecenie mount.....	346
Polecenie umount.....	347
Automatyczne montowanie systemu plików przy użyciu pliku konfiguracyjnego /etc/fstab.....	347
Narzędzia z interfejsem graficznym przeznaczone do montowania systemu plików.....	350
Relokacja systemu plików.....	351
Instalacja nowego dysku.....	351
Tworzenie tablicy partycji i formatowanie dysku.....	353
Zamontowanie nowej partycji i przeniesienie danych.....	353
LVM — zarządzanie wolumenami logicznymi.....	354
Praca z systemem plików.....	355
Tworzenie testowego systemu plików.....	355
Korzystanie z programu dumpe2fs.....	357
Montowanie systemu plików w trybie tylko do odczytu.....	360
Konwersja istniejącego systemu plików ext2 na system ext3.....	361
Tworzenie RAM-dysku rozruchowego.....	362
Zawartość pliku obrazu initrd.....	363
Strojenie dysku.....	363
Strojenie dysku twardego — BIOS oraz jądro systemu.....	364
Polecenie hdparm.....	365
Strojenie systemu plików.....	366
Polecenie mke2fs.....	366
Polecenie tune2fs.....	367
Polecenie e2fsck.....	367
Polecenie badblocks.....	367
Opcja noatime polecenia mount.....	368
Zarządzanie plikami urządzeń znakowych, blokowych i specjalnych.....	368
Konwencja nazewnicza dla urządzeń blokowych i znakowych.....	370
Tworzenie mknod za pomocą polecenia mknod.....	370
Warto zajrzeć.....	373

Rozdział 11. Tworzenie kopii bezpieczeństwa danych, odzyskiwanie danych i odtwarzanie systemu..... 375

Wybór strategii wykonywania kopii bezpieczeństwa danych.....	375
Dlaczego dochodzi do utraty danych?.....	376
Ocena wymaganego zakresu kopii bezpieczeństwa oraz dostępności zasobów systemowych.....	378
Ocena strategii wykonywania kopii bezpieczeństwa.....	381
Dokonaj właściwego wyboru.....	385

Wybór urządzeń i nośnika przeznaczonego do wykonywania kopii bezpieczeństwa danych	386
Wymienne nośniki danych	386
Tworzenie i przechowywanie kopii bezpieczeństwa na dyskach sieciowych.....	389
Tworzenie i przechowywanie kopii bezpieczeństwa na urządzeniach taśmowych.....	389
Zastosowanie oprogramowania dedykowanego do wykonywania kopii bezpieczeństwa danych	390
Polecenie tar	391
Tworzenie kopii bezpieczeństwa danych przy użyciu polecenia cpio	393
GNOME File Roller — graficzne narzędzie do archiwizacji danych.....	395
KDE ark oraz kdat — graficzne narzędzie do archiwizacji danych.....	397
Zastosowanie polecenia dd do tworzenia kopii bezpieczeństwa danych	398
Zastosowanie pakietu Amanda.....	399
Alternatywne pakiety oprogramowania do archiwizacji danych	400
Kopiowanie plików	401
Kopiowanie plików przy użyciu polecenia tar	402
Pakowanie, szyfrowanie i wysyłanie potoków tar.....	403
Kopiowanie plików przy użyciu polecenia cp	404
Kopiowanie plików przy użyciu polecenia cpio.....	405
Kopiowanie plików przy użyciu polecenia mc.....	405
Kopiowanie plików przy użyciu polecenia scp	406
Kopiowanie plików przy użyciu polecenia rsync	408
Odzyskiwanie usuniętych plików	410
Proces odzyskiwania plików w systemie plików ext2	411
Reformatowanie z opcją -S jako rozwiązanie po wystąpieniu krytycznych błędów systemu plików	412
Odzyskiwanie usuniętych plików przy użyciu polecenia mc	412
Odtwarzanie systemu.....	413
Dysk awaryjny systemu Fedora Core 2 Linux.....	414
Tworzenie kopii i odtwarzanie głównego sektora rozruchowego.....	414
Ręczne odtwarzanie tablicy partycji.....	415
Uruchamianie systemu z awaryjnego dysku CD	416
Zastosowanie programu ładującego GRUB	417
Zastosowanie opcji Recovery Facility.....	417
Warto zajrzeć.....	420

Część III Administrowanie usługami systemu 423

Rozdział 12. Drukowanie w systemie Fedora 425

Podstawy drukowania w systemie Fedora	426
Konfiguracja i zarządzanie usługami drukowania	427
Szybkie wprowadzenie do graficznej konfiguracji drukarki	428
Szybkie wprowadzenie do tekstowej konfiguracji drukarki	429
Zarządzanie usługami drukowania	429
Definiowanie i konfiguracja drukarek lokalnych.....	432
Tworzenie kolejek wydruków	432
Edycja ustawień drukarki	435
Definiowanie drukarek sieciowych.....	436
Dostosowanie uprawnień drukowania w sieci lokalnej	437
Drukowanie za pomocą protokołu SMB	439
Konfiguracja i używanie drukarek bezpośrednio podłączonych do sieci	440

Sterowanie drukowaniem z poziomu konsoli	442
Konfigurowanie drukarek z poziomu konsoli	442
Używanie podstawowych poleceń drukowania	443
Zarządzanie zadaniami wydruku	443
Użycie opartego na WWW interfejsu systemu CUPS	445
Tworzenie w systemie CUPS wpisu o drukarce	445
Unikanie problemów z obsługą drukarek	449
Urządzenia wielofunkcyjne	449
Używanie drukarek USB i tradycyjnych	449
Warto zajrzeć	450

Rozdział 13. Łączność sieciowa..... 453

Budowa sieci TCP/IP	453
Adresowanie TCP/IP	454
Stosowanie maskarady IP w systemie Fedora	456
Porty	457
Organizacja sieci	458
Tworzenie podsieci	458
Maski podsieci	459
Adresowanie do jednego, do grupy lub do wszystkich komputerów	460
Urządzenia sprzętowe sieci	460
Karty sieciowe	460
Okablowanie sieciowe	464
Koncentratory	465
Mosty	466
Przełączniki	466
Routery	467
Inicjalizowanie nowego sprzętu sieciowego	467
Używanie narzędzi konfiguracji sieci	470
Konfigurowanie interfejsów sieciowych z wiersza poleceń	470
Pliki konfiguracji sieci	475
Używanie graficznych narzędzi konfiguracyjnych	478
Protokół dynamicznej konfiguracji hosta (DHCP)	480
Jak działa protokół DHCP	481
Wykorzystanie protokołu DHCP podczas instalacji i uruchamiania systemu	482
Instalacja i konfiguracja oprogramowania DHCP	483
Używanie protokołu DHCP do konfigurowania hostów	484
Inne zastosowania protokołu DHCP	487
Używanie sieciowego systemu plików (NFS)	487
Instalacja oraz uruchamianie i zatrzymywanie usług NFS	487
Konfigurowanie serwera NFS	488
Konfigurowanie klienta NFS	489
Korzystanie z pakietu Samba	491
Konfigurowanie pakietu Samba bezpośrednio w pliku /etc/samba/smb.conf	492
Testowanie konfiguracji poleceniem testparm	495
Uruchamianie demona smbd	496
Montowanie udziałów SMB	498
Konfigurowanie połączeń Samba przy użyciu programu SWAT	498

Sieci bezprzewodowe	502
Zakres obsługi sieci bezprzewodowych w systemie Fedora.....	503
Obsługa sieci komórkowych	505
Zalety sieci bezprzewodowych.....	505
Wybór spośród dostępnych protokołów transmisji bezprzewodowej.....	506
Zabezpieczanie sieci bezprzewodowej.....	506
Zabezpieczanie sieci	507
Ustawianie zapory przy użyciu programów lokkit i system config-securitylevel.....	508
Hasła i dostęp fizyczny.....	509
Zabezpieczanie na poziomie protokołu TCP/IP	510
Konfiguracja i użycie programu Tripwire	511
Urządzenia.....	512
Zabezpieczanie usługi DHCP.....	513
Zabezpieczanie usługi NFS.....	513
Zabezpieczanie usługi Samba.....	513
Śledzenie aktualnych wiadomości o zabezpieczaniu systemu Linux	513
Używanie poprawek i uaktualnień w celu zapewniania bezpieczeństwa sieci	514
Warto zajrzeć.....	514
Zagadnienia ogólne	514
Protokół DHCP	514
Sieci bezprzewodowe	515
Bezpieczeństwo.....	516
Książki	516

Rozdział 14. Zarządzanie usługami DNS..... 517

Konfiguracja systemu DNS	518
Plik /etc/hosts	519
Podstawowe pojęcia związane z systemem DNS	520
Przechowywanie informacji o strukturze DNS na serwerze nazw.....	522
W jaki sposób system DNS dostarcza usługę tłumaczenia nazw.....	522
Tłumaczenie nazw w praktyce	523
Odwrotne tłumaczenie adresów (ang. reverse resolution).....	526
Wyniki pracy programu tłumaczącego	528
Własna nazwa domeny i serwery DNS.....	529
Przegląd narzędzi związanych z usługami DNS.....	530
Polecenie dig	531
Polecenie host.....	532
Polecenie nslookup.....	532
Polecenie whois.....	533
Konfiguracja lokalnego buforującego serwera nazw	535
Konfiguracja serwera nazw przy użyciu pakietu BIND.....	536
Plik konfiguracyjny rndc.conf.....	537
Plik konfiguracyjny named.conf.....	538
Rejestracja zdarzeń.....	544
Konfiguracja programu tłumaczącego.....	544
Uruchamianie demona serwera nazw named.....	545
Konfiguracja serwera DNS do obsługi domen rzeczywistych	547
Strefa prosta	547
Strefa odwrotna	548
Rejestracja domeny	549

Rozwiązywanie problemów z serwerem DNS.....	550
Problemy związane z przekazywaniem odpowiedzialności (delegowaniem domen).....	550
Problemy związane z wyszukiwaniem w tył.....	551
Zapewnienie poprawności numerów seryjnych.....	552
Rozwiązywanie problemów związanych z plikami strefy.....	552
Narzędzia przydatne w diagnozowaniu problemów.....	553
Korzystanie z narzędzia konfiguracyjnego BIND Fedora Core 2.....	553
Zapewnienie bezpieczeństwa serwerów DNS.....	555
Zagadnienia bezpieczeństwa w systemie UNIX.....	555
Zagadnienia bezpieczeństwa systemu DNS.....	557
Rozdzielony DNS.....	560
Warto zajrzeć.....	561

Rozdział 15. Połączenie z siecią Internet 563

Konfiguracja połączeń — informacje ogólne.....	564
Zaczynamy od podstaw: Interfejs localhost.....	565
Sprawdzanie dostępności interfejsu lo.....	566
Manualna konfiguracja interfejsu lo.....	566
Konfiguracja połączeń typu dial-up.....	567
Manualna konfiguracja połączeń typu dial-up.....	569
Zastosowanie kreatora połączeń internetowych.....	571
Konfiguracja połączeń DSL.....	575
Zastosowanie protokołu PPPoE.....	576
Manualna konfiguracja połączeń PPPoE.....	576
Rozwiązywanie problemów z połączeniami z siecią Internet.....	578
Konfiguracja serwera dostępowego dial-up dla połączeń PPP.....	579
Warto zajrzeć.....	583

Rozdział 16. Zarządzanie serwerem WWW Apache 585

Serwer WWW Apache.....	585
Instalowanie serwera Apache.....	587
Instalacja serwera z pakietów RPM.....	587
Samodzielna kompilacja kodu źródłowego serwera.....	589
Uruchamianie i zatrzymywanie serwera Apache.....	591
Ręczne uruchamianie serwera Apache.....	591
Korzystanie ze skryptu /etc/rc.d/init.d/httpd.....	593
Sterowanie serwerem Apache — polecenie service.....	595
Sterowanie serwerem Apache — polecenie chkconfig.....	595
Sterowanie serwerem Apache — interfejs redhat-config-services.....	596
Ustawienia konfiguracyjne serwera.....	597
Dyrektywy konfiguracyjne.....	597
Edycja pliku httpd.conf.....	598
Moduły MPM.....	601
Pliki konfiguracyjne .htaccess.....	602
Uwierzytelnianie i kontrola dostępu.....	604
Ograniczanie dostępu dyrektywami allow oraz deny.....	604
Uwierzytelnianie.....	605
Kontrola dostępu raz jeszcze.....	608

Moduły serwera Apache.....	608
mod_access	610
mod_alias	611
mod_asis.....	611
mod_auth.....	612
mod_auth_anon.....	612
mod_auth_dbm.....	612
mod_auth_digest.....	613
mod_autoindex.....	613
mod_cgi.....	613
mod_dir oraz mod_env.....	613
mod_expires.....	614
mod_headers	614
mod_imap.....	614
mod_include.....	614
mod_info oraz mod_log_config.....	615
mod_mime oraz mod_mime_magic.....	615
mod_negotiation.....	615
mod_proxy.....	615
mod_rewrite.....	615
mod_setenvif.....	616
mod_speling.....	616
mod_status.....	616
mod_ssl.....	616
mod_unique_id.....	616
mod_userdir.....	616
mod_usertrack.....	617
mod_vhost_alias.....	617
Serwery wirtualne.....	617
Węzły wirtualne rozróżniane adresami IP.....	618
Węzły wirtualne rozróżniane nazwami	618
Rejestrowanie	619
Dokumenty dynamiczne.....	621
CGI.....	622
Wstawki SSI.....	623
Podstawowe dyrektywy SSI.....	624
Przekazywanie sterowania.....	628
Graficzny interfejs konfiguracji serwera Apache	629
Konfigurowanie węzłów wirtualnych.....	630
Konfigurowanie serwera	631
Konfigurowanie serwera pod kątem wydajności szczytowej.....	631
Inne serwery WWW dostępne dla dystrybucji Fedora	632
thttpd.....	632
Sun ONE Web Server.....	632
Stronghold.....	633
Zope.....	633
Zeus Web Server.....	634
TWiki.....	634
Warto zajrzeć.....	635

Rozdział 17. Administrowanie usługami baz danych 637

Krótkie wprowadzenie do baz danych.....	638
Zasada działania relacyjnych baz danych.....	639
Podstawy języka SQL	641
Tworzenie tabel.....	642
Wypełnianie tabel danymi.....	643
Pobieranie informacji z bazy danych.....	644
Wybór bazy danych: MySQL kontra PostgreSQL.....	646
Szybkość	647
Blokowanie danych.....	647
Przetwarzanie transakcji a ochrona spójności danych — reguły ACID	648
Podzapytania SQL.....	649
Języki proceduralne i wyzwalacze.....	650
Dostępne aplikacje	650
Instalowanie i konfigurowanie bazy danych MySQL.....	651
Inicjalizowanie katalogu danych bazy MySQL.....	652
Przypisywanie hasła do konta użytkownika głównego bazy danych MySQL.....	653
Tworzenie bazy danych.....	653
Przyznawanie i odbieranie uprawnień w bazie danych MySQL.....	654
Instalowanie i konfigurowanie bazy danych PostgreSQL	656
Inicjalizowanie katalogu danych bazy PostgreSQL	656
Tworzenie bazy danych.....	658
Tworzenie kont użytkowników bazy danych PostgreSQL	658
Usuwanie kont użytkowników bazy danych PostgreSQL	659
Przyznawanie i odbieranie uprawnień użytkownikom bazy danych PostgreSQL	659
Programy-klienty baz danych.....	660
Dostęp do bazy danych za pośrednictwem SSH.....	661
Dostęp do serwera bazy danych za pośrednictwem programu klienta wzposażonego w interfejs graficzny	662
Dostęp do serwera bazy danych za pośrednictwem interfejsu WWW.....	662
Program klienta bazy danych MySQL	663
Program klienta bazy danych PostgreSQL	664
Interfejsy graficzne.....	665
Warto zajrzeć.....	666

Rozdział 18. Bezpieczny transfer plików a usługa FTP 667

Programy-klienty FTP	667
Bezpieczny transfer plików — sftp	668
Programy klientów protokołu FTP	669
Wierszowy interfejs klienta FTP	670
Graficzne programy klientów protokołu FTP.....	677
Serwery FTP.....	681
Serwer z uwierzytelnianiem, czy anonimowy?	681
Oprogramowanie serwera FTP dla Fedora Core 2	682
Pozostałe serwery FTP	682
Instalowanie oprogramowania serwera FTP.....	683
Użytkownik usługi FTP.....	684
Konfigurowanie usługi xinetd dla serwera FTP	686
Konfigurowanie usługi xinetd dla serwera wu-ftpd.....	687
Uruchamianie serwera vsftpd.....	688

Konfigurowanie serwera vsftpd.....	688
Kontrola poczynań użytkowników anonimowych.....	689
Pozostałe pliki konfiguracyjne serwera vsftpd.....	690
Konfigurowanie serwera wu-ftpd.....	692
Parametry ogólne serwera wu-ftpd — plik ftpaccess.....	693
Kontrola dostępu.....	693
Informacje o użytkownikach.....	696
Opcje rejestrowania działalności serwera.....	700
Opcje uprawnień.....	702
Inne opcje.....	704
Struktura pliku zamknięcia serwera.....	705
Konfiguracja reguł konwersji plików.....	705
Usuwanie przedrostka nazwy.....	706
Usuwanie przyrostka nazwy.....	706
Dodawanie przedrostka nazwy.....	706
Dodawanie przyrostka nazwy.....	706
Zewnętrzne polecenie konwersji.....	707
Typy.....	707
Opcje konwersji.....	707
Pole opisu.....	708
Przykład zastosowania reguł konwersji.....	708
Blokowanie i dopuszczanie połączeń — plik ftphosts.....	708
Administrowanie serwerem FTP.....	709
Zestawienie aktywnych połączeń.....	710
Licznik połączeń.....	711
Przygotowanie planowego zatrzymania serwera.....	711
Analiza pliku dziennika serwera.....	713
Warto zajrzeć.....	713
Rozdział 19. Obsługa poczty elektronicznej.....	717
Wysyłanie i odbieranie poczty elektronicznej.....	719
Oprogramowanie MTA.....	720
Wybór oprogramowania MTA.....	722
Oprogramowanie MDA.....	723
Oprogramowanie MUA — programy pocztowe.....	723
Wybór programu pocztowego.....	724
Program mail.....	724
Program mutt.....	727
Program Evolution.....	729
Program Balsa.....	731
Program KMail.....	731
Program Mozilla Mail.....	731
Pozostałe programy pocztowe.....	733
Załączniki — wysyłanie plików binarnych.....	733
BinHex.....	735
yenc.....	735
uuencode oraz uuencode.....	735
Podstawy konfigurowania i stosowania programu Sendmail.....	737
Maskarada.....	738
Smart Hosts.....	739

Interwał czasowy kolejnych prób dostarczenia poczty	739
Kompilowanie pliku sendmail.mc	740
Przekazywanie poczty	740
Aliasy adresów poczty elektronicznej	741
Odrzucanie poczty przychodzącej z podejrzanych węzłów	741
Pobieranie poczty — program Fetchmail	742
Instalowanie programu Fetchmail	743
Konfigurowanie programu Fetchmail	743
Wybór oprogramowania MDA	747
Procmail	747
Spamassasin	748
Squirrelmail	748
Skanery antywirusowe	749
Oprogramowanie specjalne MDA	749
Alternatywy dla Microsoft Exchange Server	750
Microsoft Exchange Server i Outlook	750
CommuniGate Pro	751
Samsung Contact (dawniej HP OpenMail)	751
Bynari	751
SuSE OpenExchange	751
Kroupware	752
OpenGroupware (dawniej SKYRIX 4.1)	752
phpgroupware	752
PHPProjekt	753
IMP-Horde	753
Wnioski	753
Warto zajrzeć	753
Zasoby sieci WWW	753
Książki	755

Rozdział 20. Grupy dyskusyjne

i inne narzędzia komunikacji w zespole 757

Grupy dyskusyjne — wprowadzenie	758
Grupy dyskusyjne	758
Wybór czytelnika grup dyskusyjnych	759
Czytnik slrn	759
Czytnik Pan	761
Czytnik KNode	762
Czytnik Mozilla	762
Komunikacja w zespole — TWiki	763
Internet Relay Chat	766
Komunikator GAIM	768
GnomeMeeting i wideokonferencje	769
Listy dystrybucyjne poczty elektronicznej — Mailman	770
Konfigurowanie lokalnego serwera grup dyskusyjnych	774
Typy serwerów grup dyskusyjnych	774
Skład pakietu INN	776
Instalowanie pakietu INN	776
Konfigurowanie serwera innd	778
Uruchamianie serwera innd	785
Warto zajrzeć	787

Część IV Programowanie i praca biurowa 789**Rozdział 21. Narzędzia programistyczne języków C i C++ 791**

Linux a programowanie w języku C	792
Programowanie w języku C++	793
Programowanie w językach C i C++ w Linuksie — zaczynamy	794
Proces tworzenia programu	795
Elementy języków C i C++	797
Narzędzia służące do zarządzania projektem programistycznym dostępne w dystrybucji Fedora	798
Kompilacja programów za pośrednictwem programu make	798
Konfigurowanie kodu za pomocą narzędzia autoconf	800
Zarządzanie projektami programistycznymi — RCS i CVS	801
Tworzenie bibliotek — polecenie ar	804
Narzędzia diagnostyczne	804
Stosowanie kompilatora GNU C	806
Prosty program w języku C	807
Narzędzia prototypowania graficznego	808
Program KDevelop	809
QT Designer	810
Programowanie w GNOME — narzędzie Glade	811
Dodatkowe źródła informacji	812
Warto zajrzeć	813

Rozdział 22. Programowanie skryptów powłoki 815

Powłoki dostępne w dystrybucji Fedora Core 2	816
Wiersz poleceń powłoki	817
Porównywanie wzorców w powłoce	818
Przekierowywanie wejścia i wyjścia programów	819
Potoki danych	820
Przetwarzanie w tle	820
Podstawy tworzenia i uruchamiania skryptów powłoki	821
Tworzenie i uruchamianie prostego skryptu powłoki bash	822
Uruchamianie nowo utworzonego skryptu powłoki	823
Udostępnianie skryptów w systemie	824
Wskazywanie powłoki do interpretacji skryptów	824
Zmienne w skryptach powłoki	826
Przypisywanie wartości do zmiennych	826
Odwołania do wartości zmiennych	827
Parametry pozycyjne	827
Przykład wykorzystania parametru pozycyjnego	828
Pozyskiwanie wartości z wiersza polecenia za pomocą parametrów pozycyjnych	828
Skryptowa automatyzacja zadań	829
Zmienne wbudowane	831
Znaki specjalne	832
Działanie znaków podwójnego cudzysłowu	832
Działanie znaków pojedynczego cudzysłowu	833
Działanie znaku lewego ukośnika	834
Działanie znaku pojedynczego cudzysłowu otwierającego	834

Wyrażenia porównania	835
Wyrażenia porównania w powłokach pdksh i bash	835
Wyrażenia porównania w powłoce tcsh	840
Instrukcje pętli	843
Instrukcja for	843
Instrukcja while	845
Instrukcja until	847
Instrukcja repeat (tcsh)	847
Instrukcja select (bash i pdksh)	848
Instrukcja shift	848
Instrukcje warunkowe	849
Instrukcja if	849
Instrukcja case	850
Instrukcje break oraz exit	852
Funkcje w skryptach powłoki	852
Warto zajrzeć	853

Rozdział 23. Język Perl **855**

Perl w systemie Linux	856
Wersje języka Perl	857
Prosty program w języku Perl	857
Zmienne i struktury danych w Perlu	859
Typy zmiennych	859
Zmienne specjalne	860
Operatory	861
Operatory porównania	861
Operatory logiczne	862
Operatory arytmetyczne	862
Inne operatory	862
Specjalne stałe znakowe	863
Instrukcje warunkowe: if oraz unless	864
Instrukcja if	864
Instrukcja unless	865
Pętle	866
Instrukcja for	866
Instrukcja foreach	866
Instrukcja while	867
Instrukcja until	867
Instrukcje last oraz next	868
Instrukcje do ... while oraz do ... until	868
Wyrażenia regularne	868
Dostęp do powłoki	869
Przełączniki	870
Moduły Perla i CPAN	873
Kody przykładowe w języku Perl	874
Wysyłanie poczty elektronicznej	874
Porządkowanie dzienników	876
Wysyłanie wiadomości do grup dyskusyjnych	877

Jednowierszowce.....	878
Sortowanie	878
Przetwarzanie na poziomie wiersza polecenia.....	879
Warto zajrzeć.....	880
Książki	880
Grupy dyskusyjne.....	881
WWW	881
Inne	882
Rozdział 24. Zarządzanie jądrem i modułami jądra.....	883
Jądro systemu Linux.....	884
Drzewo kodu źródłowego Linuksa.....	884
Rodzaje jąder.....	887
Zarządzanie modułami.....	888
Kiedy kompilować jądro	890
Wersje jądra.....	891
Pobieranie kodu źródłowego jądra	892
Łatanie jądra.....	894
Kompilacja jądra	896
Wybór interfejsu konfiguracji.....	897
Konfiguracja jądra za pomocą interfejsu make xconfig	899
Tworzenie obrazu RAM-dysku początkowego	901
Gdy coś pójdzie nie tak.....	901
Błędy kompilacji	901
Błędy czasu wykonania, błędy programu rozruchowego i wyjątki jądra.....	903
Sterowanie działającym jądrem — sysctl	904
SELinux.....	905
Przygotowania do instalacji.....	905
Konfiguracja jądra.....	906
Budowa i instalacja	907
Eksperymenty z nowym API.....	909
Warto zajrzeć.....	911
Rozdział 25. Aplikacje biurowe.....	913
Pakiety oprogramowania biurowego w dystrybucji Fedora Core 2	913
OpenOffice.org.....	914
Pakiet GNOME Office	919
Pakiet KOffice.....	922
Komunikacja z palmtopami.....	925
Oprogramowanie dla palmtopów uruchamiane z poziomu wiersza poleceń	926
Oprogramowanie dla palmtopów z interfejsem graficznym	926
Obsługa skanerów w dystrybucji Fedora Core 2	927
Narzędzia projektowania stron WWW	929
Faks	931
Inne aplikacje użytkowe dystrybucji Fedora Core 2.....	933
Aplikacje biurowe dla systemu Microsoft Windows	935
Warto zajrzeć.....	936

Rozdział 26. Aplikacje multimedialne	937
Nagrywanie płyt CD i DVD	938
Nagrywanie płyt CD z poziomu wiersza poleceń	939
Nagrywanie płyt DVD z poziomu wiersza poleceń	941
Nagrywanie płyt CD w środowisku graficznym	943
Muzyka i dźwięki	947
Karty dźwiękowe	947
Rejestracja dźwięku	948
Formaty dźwięku	948
Odtwarzacze muzyczne	950
Strumieniowe transmisje dźwięku	951
Oglądanie telewizji i filmów	952
Wymagany sprzęt	952
Formaty wideo	954
Oglądanie filmów	955
Oglądanie telewizji	956
Cyfrowy magnetowid	957
Odtwarzacze DVD i wideo	957
Cyfrowe aparaty fotograficzne	958
Kamery internetowe	958
Cyfrowe aparaty fotograficzne	959
Korzystanie ze skanerów	960
Obróbka grafiki	961
Program GIMP	962
Obsługa formatów graficznych	963
Wykonywanie zrzutów ekranu	966
Linux a gry	967
Instalowanie sterowników kart nVidia	968
Instalowanie gry Unreal Tournament 2003	969
Instalowanie gry Wolfenstein — Enemy Territory	970
Warto zajrzeć	973
Rozdział 27. Emulatory i narzędzia obsługi innych platform.....	975
Emulatory dostępne w dystrybucji Fedora Core 2	975
Emulator DOSBox	976
Uruchamianie programów dla środowiska Windows — Wine	977
Przygotowanie do uruchomienia emulatora	979
Konfiguracja Wine	979
Uruchamianie aplikacji dla środowiska Win32	986
Granie z WineX	987
Produkty firmy Codeweaver	989
Instalowanie, konfigurowanie i korzystanie z VMware	990
Instalowanie VMware	991
Konfigurowanie VMware	991
Uruchamianie sesji VMware	991
Windows w Linuksie — Win4Lin	993
Konfigurowanie Win4Lin	994
Uruchamianie sesji Win4Lin	994

Emulowanie systemu Mac OS.....	996
Praca zdalna.....	997
Virtual Network Computing.....	997
Aplikacje pomocnicze VNC.....	1000
Cygwin dla Windows.....	1000
Kompilatory międzyplatformowe i narzędzia obsługi innych platform.....	1001
Międzyplatformowa obsługa dyskietek.....	1002
MTools.....	1003
HFSutils.....	1004
Emulacja formatów binarnych aplikacji.....	1005
Warto zajrzeć.....	1006
Dodatki.....	1007
Dodatek A Fedora Core i Linux w Internecie.....	1009
Witryny WWW i wyszukiwarki.....	1010
Wyszukiwanie informacji w sieci WWW.....	1010
Google Twoim przyjacielem.....	1011
Lista pakietów dystrybucji.....	1012
Certyfikaty kwalifikacji.....	1012
Wsparcie techniczne.....	1012
Dokumentacja.....	1013
Podręczniki i instrukcje.....	1013
Fedora Project.....	1014
Red Hat Linux.....	1014
Minidystrybucje Linuksa.....	1015
Inne dystrybucje dla platformy PC.....	1016
Dystrybucja dla komputerów z procesorami PowerPC.....	1016
Linux na laptopach i palmtopach.....	1016
Środowisko X Window System.....	1017
Grupy dyskusyjne.....	1017
Listy dystrybucyjne poczty elektronicznej.....	1019
Listy dystrybucyjne projektu Fedora.....	1020
Listy dystrybucyjne systemu Red Hat.....	1020
IRC.....	1021
Dodatek B Skorowidz zagadnień.....	1023
Dodatek C Najważniejsze polecenia systemu Linux.....	1025
Narzędzia plikowe.....	1025
Narzędzia internetowe.....	1028
Narzędzia obsługi sieci.....	1029
Narzędzia systemowe.....	1029
Narzędzia obróbki plików tekstowych.....	1032
Narzędzia administracyjne.....	1033
Skorowidz.....	1035

Rozdział 17.

Administrowanie usługami baz danych

Bieżący rozdział stanowić będzie wprowadzenie do korzystania z baz danych MySQL i PostgreSQL, dwóch systemów baz danych rozprowadzanych wraz z dystrybucją Fedora Core 2. Czytelnik będzie miał okazję poznać podstawowe funkcje tych systemów, porównać ich możliwości i poznać zalety i wady obu. Informacje te pomogą przy dokonywaniu wyboru systemu, którego zadaniem będzie obsługa baz danych danej organizacji. Omówienie zawierać będzie również opis procedur instalacji, inicjalizacji i konfiguracji tych popularnych serwerów baz danych, jak również sposobu administrowania ich działaniem i utrzymywania bezpieczeństwa i spójności danych zarządzanych przez te serwery.

Administrator bazy danych ma w organizacji kilka zadań, zależnych w dużej mierze od rozmiaru i zakresu działania organizacji, liczebności zespołu działu informatycznego i tak dalej. W zależności od struktury danej organizacji zakres odpowiedzialności administratora bazy danych może obejmować:

- ◆ Instalowanie i konserwacja **serwerów baz danych** — konserwacja oznacza tu zadanie mające na celu podtrzymanie dostępności danych i dbanie o stabilność serwera, a więc na przykład instalowanie publikowanych łat eliminujących błędy i luki w zabezpieczeniach czy aktualizowanie oprogramowania. Administrator bazy danych potrzebuje do realizacji tych zadań uprawnień użytkownika uprzywilejowanego oraz umiejętności zarządzania oprogramowaniem (patrz rozdział 8., „Zarządzanie oprogramowaniem i zasobami systemu”). Nie zaszkodzi też znajomość jądra systemu, wykorzystywanych systemów plików i innych elementów, mających wpływ czy to na wydajność, czy też bezpieczeństwo dostępu do danych.
- ◆ Instalowanie i konserwacja oprogramowania **klientów baz danych** — klient bazy danych to program, który w imieniu użytkownika nawiązuje połączenie z serwerem bazy danych i udostępnia użytkownikowi otrzymane dane (patrz podrozdział „Klienty baz danych”), czy to odwołując się do serwera lokalnego, czy też nawiązując połączenie za pośrednictwem sieci komputerowej. Odpowiedzialność administratora w tym zakresie może obejmować instalowanie i konserwację oprogramowania klienckiego. Rozdział ten prezentuje między innymi sposób instalowania takiego oprogramowania oraz korzystania z oprogramowania klienckiego, z poziomu wiersza poleceń oraz za pośrednictwem interfejsu graficznego.

- ◆ Zarządzanie kontami i użytkownikami — zarządzanie kontami i użytkownikami obejmuje dodawanie i usuwanie użytkowników bazy danych, przypisywanie haseł i zarządzanie hasłami i tak dalej. W rozdziale pokazana zostanie między innymi procedura przyznawania i odbierania użytkownikowi bazy danych uprawnień dostępu do danych i przypisywania hasła (prezentacja obejmować będzie oba systemy baz danych).
- ◆ Zapewnianie bezpieczeństwa bazy danych — w tym zakresie należy zadbać o takie elementy, jak właściwa kontrola dostępu, blokująca dostęp do danych osobom nieupoważnionym, oraz system uprawnień, pozwalający na regulowanie zakresu działalności użytkowników bazy danych. W rozdziale pokazany zostanie sposób zarządzania dostępem do bazy danych za pośrednictwem WWW, klienta protokołu SSH oraz lokalnego klienta wyposażonego w interfejs graficzny. Bezpieczeństwo danych jest również silnie uzależnione od starannego zaplanowania a następnie konsekwentnego wdrażania harmonogramu wykonywania kopii zapasowych danych.
- ◆ Zapewnianie spójności danych — z wszystkich informacji przechowywanych na dysku serwera najważniejsze mogą okazać się te przechowywane w bazie danych. Zapewnienie spójności tych danych obejmuje regulowanie współbieżnego dostępu do danych i zapewnianie, że żadne z modyfikacji wprowadzonych w tym samym czasie przez różnych użytkowników nie zostaną ani utracone, ani nie będą się dublować.

Krótkie wprowadzenie do baz danych

Usługi baz danych w systemie Linux, wykorzystujące oprogramowanie omawiane w rozdziale, to usługi oparte na **architekturze klient-serwer**. Klienci bazy danych to programy pośredniczące we wprowadzaniu danych oraz inicjowaniu zapytań do serwera danych, zwracających wyznaczone kryteriami zapytania zbiory danych. Do uruchomionego serwera można się odwołać zarówno za pośrednictwem klienta działającego w oknie terminala, jak i dysponującego interfejsem graficznym. Bazy danych można przy tym podzielić na bazy plikowe i relacyjne. **Plikowa baza danych** może być przechowywana w zwykłym pliku, w którym poszczególne pola i rekordy są oddzielane znakami spacji czy nowego wiersza. Przykładem plikowej bazy danych jest systemowy plik haseł `/etc/passwd`. Innym przykładem mógłby być plik prostej książki adresowej, zawierającej następujące wpisy:

```
Doe~John-505 Some Street~Anytown-NY-12345-555-555-1212
```

Do przeszukiwania i wyodrębniania informacji z utworzonej tak prymitywnej bazy danych można wykorzystywać popularne narzędzia, takie jak `grep`, `awk` czy programy w języku Perl. Programy te mogą się zresztą świetnie sprawdzać, dopóki baza danych nie będzie zbyt obszerna; w ogólności jednak plikowe bazy danych są obciążone następującymi ograniczeniami:

- ◆ Nie dają się dobrze skalować. Plikowe bazy danych nie mogą realizować swobodnego dostępu do danych. Przeszukiwanie zawsze odbywa się sekwencyjnie. Oznacza to, że wyszukiwanie pewnych informacji jest realizowane przez odczytywanie i analizowanie kolejnych wierszy pliku. W miarę wzrostu rozmiaru bazy danych metoda ta przestaje jednak być efektywna.

- ♦ Plikowe bazy danych nie nadają się do wykorzystania w środowisku wielodostępnym. Taka baza danych, w zależności od konfiguracji, może albo zezwalać na modyfikację danych tylko jednemu użytkownikowi naraz, albo — kiedy modyfikacje będą przeprowadzane równocześnie przez kilku użytkowników — dopuszczać potencjalną możliwość utraty i nadpisania danych w wyniku dostępu współbieżnego.

Powyższe ograniczenia czynią plikowe bazy danych niewystarczającymi w poważniejszych zadaniach; plikowa baza danych nie jest odpowiednia nawet dla niewielkiej firmy, a co dopiero potężnej korporacji. Co innego relacyjne bazy danych, określane często skrótem RSZBD (relacyjne systemy zarządzania bazami danych). Te pozwalają na określenie powiązań pomiędzy poszczególnymi elementami danych; dane te przechowywane są w tabelach a w ramach tabel w polach, podobnie trochę jak to ma miejsce w przypadku arkusza kalkulacyjnego. Dane te można efektywnie przeszukiwać i porządkować. Z tego względu przedmiotem zainteresowania niniejszego rozdziału będą właśnie relacyjne bazy danych.

Przykładami relacyjnych baz danych są Oracle, DB2, Microsoft SQL Server czy rozprawdane nieodpłatnie PostgreSQL i MySQL. Omówienie obejmować będzie dwie ostatnie — Czytelnik będzie miał okazję poznać podstawy administrowania bazami danych, pozna też podstawy języka SQL, będącego standardową metodą manipulowania danymi przechowywanymi w bazach danych.

Zasada działania relacyjnych baz danych

Prawidłowe utworzenie i efektywne zarządzanie relacyjną bazą danych wymaga od administratora bazy znajomości podstawowych zasad regulujących sposób przechowywania informacji w takich bazach danych; przyda się też wiedza na temat tego, jak użytkownicy odwołują się do serwera bazy danych i za jego pośrednictwem manipulują danymi. Punkt ten nie zawiera co prawda szczegółowego wykładu teoretycznego na temat zasad funkcjonowania baz danych, ale zawarte tu informacje powinny wystarczyć do przyswojenia sobie choćby podstawowych koncepcji.

Relacyjny system zarządzania bazą danych przechowuje dane w tabelach, które można sobie wyobrazić jako arkusze kalkulacyjne. Każda z kolumn tabeli to tzw. pole tabeli; kolumna może np. zawierać nazwisko czy adres. Każdy wiersz tabeli to pojedynczy rekord tabeli (zwany też krotką). Tabela posiada nazwę, za pośrednictwem której można się do niej odwoływać, pozyskując z niej dane bądź umieszczając w niej nowe informacje. Przykładową prostą tabelę relacyjnego systemu zarządzania bazą danych ilustruje rysunek 17.1.

Rysunek 17.1.

Powyższa ilustracja pokazuje sposób przechowywania danych — prezentowana tabela zawiera cztery wiersze (rekordy), z których każdy zawiera po siedem pól (kolumn)

Imię	Nazwisko	Adres	Miasto	Stan	Kod	Telefon
John	Doe	501 Somestreet	Anytown	NY	55011	555-555-1212
Jane	Doe	501 Somestreet	Anytown	NY	55011	555-555-1212
John	Palmer	205 Anystreet	Sometown	NY	55055	123-456-7890
Robert	Johnson	100 Easystreet	Easytown	CT	12345	111-222-3333

W przykładzie prezentowanym na rysunku 17.1 baza danych zawiera tylko jedną tabelę. Większość systemów relacyjnych baz danych utrzymuje jednak dużo bardziej złożone struktury. Na rysunku 17.2 widać przykład bazy danych o nazwie `baza_danych`, zawierającej dwie tabele.

baza_danych

Telefony

Nazwisko	Imię	Telefon
Doe	John	555-555-1212
Doe	Jane	555-555-1212
Palmer	John	123-456-7890
Johnson	Richard	111-222-3333

Ssaki_serengeti

Nazwa potoczna	Rząd	Rodzina	Rodzaj	Gatunek	Podgatunek
Lew afrykański	drapieżne	kotowate	Panthera	leo	
Słoń afrykański	trąbowce	śloniowate	Loxodonta	africana	
Surykatka	drapieżne	łasowate	Suricata	suricata	
Hiena	drapieżne	hienowate	Crocuta	croucuta	
Żyrafa Thomsona	parzystokopytne	żyrafy	Giraffa	camelopardalis	reticulata
Zebra stepowa	parzystokopytne	koniowate	Equus	burchelli	bohmi
Gepard	drapieżne	kotowate	Acinonyx	jubatus	
Gazela Thomsona	parzystokopytne	krętorogie	Gazella	thomsoni	
Hipopotam	parzystokopytne	hipopotamowate	Hippopotamus	amphibius	

Rysunek 17.2. *Pojedyncza baza danych może zawierać dwie tabele; tutaj telefony oraz ssaki_serengeti*

W przykładowej bazie danych `baza_danych` tabela `telefony` zawiera cztery rekordy (wiersze), z których każdy podzielony jest na trzy pola (kolumny) danych. Tabela `ssaki_serengeti` zawiera z kolei dziewięć rekordów, podzielonych na sześć pól.

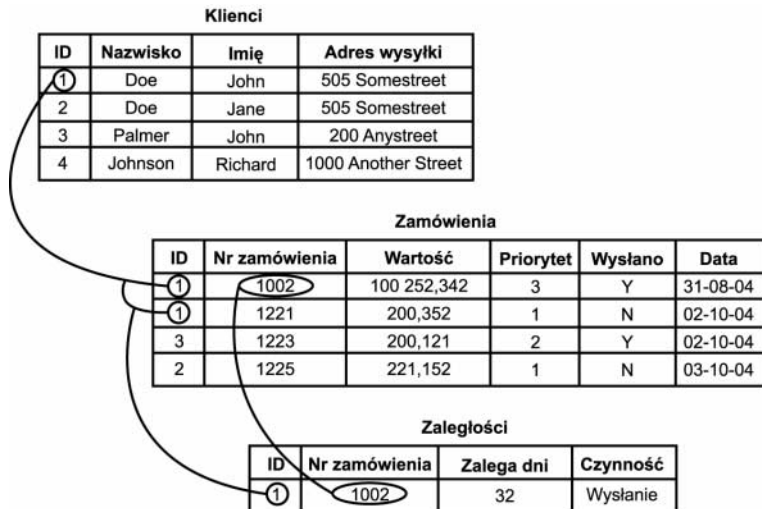
Jeżeli Czytelnik podejrzewa, że obie te tabele (a konkretnie zawarte w nich dane) nie mają ze sobą zupełnie nic wspólnego, to ma rację. W relacyjnym systemie baz danych użytkownik może tworzyć w jednej bazie danych wiele tabel, również wtedy, kiedy tabele te nie są nijak ze sobą powiązane.

W ramach przykładu poprawnego wykorzystania możliwości relacyjnej bazy danych wyobraźmy sobie niewielką firmę sprzedającą jakiś wymyślony produkt i dysponującą skomputeryzowaną bazą danych klientów. W bazie tej obok nazwisk, adresów i numerów telefonów klientów powinny znajdować się informacje o składanych przez nich zamówieniach i wystawionych fakturach. Model taki można zaimplementować za pomocą trzech tabel relacyjnej bazy danych. Przykład struktury takiej bazy danych prezentowany na rysunku 17.3.

W przykładzie z rysunku 17.3 każdy rekord konsumenta zawiera pole z indywidualnym identyfikatorem klienta. Pole to przechowuje niepowtarzalny numer, na podstawie którego można danego klienta skojarzyć z innymi informacjami, co pozwoli na określenie zamówień i stanu płatności danego klienta. Niepowtarzalność identyfikatora klienta jest bardzo ważną cechą — dwaj klienci mogą się tak samo nazywać, mieszkać w jednym domu i korzystać z tego samego telefonu, ale muszą być oznaczeni różnymi numerami

Rysunek 17.3.

Za pomocą trzech powiązanych wzajemnie tabel można stworzyć system bazy danych śledzących zamówienia i płatności wszystkich klientów



identyfikującymi. W tabelach Zamówienia i Zaległości zastosowanie identyfikatora klienta pozwala na uniknięcie dublowania informacji (nazwiska, adresu itd.). W prezentowanym układzie wyszukiwanie zamówień czy zaległości w płatnościach dowolnego z klientów bazuje na pojedynczym kluczu wyszukiwania (identyfikatorze klienta), zamiast na wielu takich kluczach (nazwisko, adres itd.). Rezultaty są więc zwracane dużo szybciej, a samo wyszukiwanie łatwiejsze w implementacji.

Mając pewne wyobrażenie o sposobie działania relacyjnych baz danych można przystąpić do nauki sposobu wprowadzania i pozyskiwania danych z baz danych. Wymaga to bowiem znajomości języka SQL.

Podstawy języka SQL

SQL to język zapytań do serwerów baz danych, „rozumiany” przez chyba wszystkie dostępne na rynku serwery relacyjnych baz danych. Zapytania języka SQL służą do konstruowania kryteriów pobierania danych i definiowania danych wprowadzanych do bazy. Przy czym tak, jak we wszystkich językach programowania, zapytania języka SQL muszą odpowiadać pewnej składni, definiującej strukturę i znaczenie poszczególnych elementów zapytania.

W większości przypadków język SQL służy do konstruowania zapytań osadzonych w kodzie aplikacji klienckich, tak, aby użytkownicy nie musieli sami konstruować zapytań. Z tego względu większość użytkowników baz danych nie jest świadoma pośrednictwa języka SQL i jego roli — zresztą słusznie; użytkownicy końcowi nie muszą przecież nic wiedzieć o strukturze zapytań SQL.

Jednak administrator bazy danych powinien znać przynajmniej podstawy języka SQL, nawet, jeśli w zakresie jego obowiązków nie leżą zadania typowo programistyczne. Na szczęście język SQL bardzo przypomina język naturalny (oczywiście angielski — *przyp. tłum.*), co znakomicie ułatwia jego opanowanie, przynajmniej w podstawowym stopniu.



Kolejne akapity w żadnym razie nie aspirują do miana podręcznika języka SQL. Mają po prostu wprowadzić Czytelnika w absolutne podstawy tego języka. Czytelnicy zainteresowani bardziej szczegółowymi informacjami o programowaniu w języku SQL powinni zajrzeć do jednej z wielu książek szerzej traktujących to zagadnienie, z publikacją *MySQL* (New Riders Publishing,) czy *Sams Teach Yourself MySQL in 21 Days* albo *PostgreSQL* (Sams Publishing)¹.

Tworzenie tabel

Relacyjny system zarządzania bazą danych przechowuje dane w tabelach, przypominających nieco te, znane z arkuszy kalkulacyjnych. Zanim jednak będzie można umieścić dane w tabeli, należy taką tabelę utworzyć, definiując jej kolumny. Służy do tego zapytanie CREATE.

Dla przykładu, tabela `ssaki_serengeti` z rysunku 17.2, posiada sześć kolumn (pól): `nazwa_potoczna`, `rząd`, `rodzina`, `rodzaj`, `gatunek` oraz `podgatunek`.

Język SQL pozwala na przypisywanie do każdej z kolumn tabeli konkretnego typu danych, określającego charakter informacji przechowywanych w tej kolumnie. Wśród dostępnych typów są INT (liczba całkowita), FLOAT (liczba zmiennoprzecinkowa), CHAR (ciąg znaków o ustalonej długości) oraz VARCHAR (ciąg znaków o zmiennej długości).

Dostępnych jest też kilka typów specjalnych, jak DATE (data) czy ENUM (zbiór wyliczeniowy). Pole typu ENUM przyjmuje wartości wyłącznie ze zdefiniowanego wcześniej zbioru. Można taki zbiór wykorzystać np. do określania płci zwierzęcia wpisywanego do bazy danych — kod płci może zostać oparty na zbiorze wyliczeniowym, zawierającym tylko dwa elementy: F (od łac. *femininum*, gram. rodzaj żeński) oraz M (od łac. *masculinum*, gram. rodzaj męski).

Już pobieżnie analizując tabelę `ssaki_serengeti` można stwierdzić, że kolumny tej tabeli zawierają dane w postaci wyłącznie ciągów znakowych. Ciągi te cechują się różną liczbą znaków. Na tej podstawie można zdecydować o przypisaniu do tych kolumn typu VARCHAR. Tabela `ssaki_serengeti` ma jeszcze jedną ciekawą właściwość. Otóż wszystkie kolumny każdego wiersza, poza kolumną ostatnią, muszą zawierać jakieś wartości. Ostatnia kolumna może pozostać pusta, ponieważ nie wszystkie zwierzęta wymienione w tabeli mają przypisaną klasyfikację podgatunku. Oba te spostrzeżenia przydadzą się przy tworzeniu tabeli.

Jak już wspomniano, do tworzenia tabeli służy zapytanie CREATE języka SQL. Zapytanie to ma następującą składnię:

```
CREATE TABLE nazwa_tabeli (nazwa_kolumny, typ_kolumny(parametry_typu), opcje, ...);
```

Przed przystąpieniem do konstruowania zapytania CREATE warto wspomnieć o dwóch rzeczach:

¹ Albo np. *Relacyjne bazy danych* Marka Whitehorna i Billa Marklyna (Helion, 2003), czy *SQL. Księga eksperta* Hansa Ladanyi'a (Helion, 2000) — *przyj. tłum.*

- ♦ Zapytania języka SQL mogą zawierać zarówno wielkie, jak i małe litery. Prawidłowe jest więc zarówno zapytanie pisane w całości literami wielkimi (CREATE TABLE ...), jak i literami małymi (create table ...) bądź o mieszanej wielkości (Create Table ...). Warto jednak wyrobić sobie nawyk pisania słów kluczowych języka SQL literami wielkimi; przyda się to podczas pisania odwołujących się do bazy danych aplikacji w języku C czy Perl — znakomicie ułatwia to późniejszą analizę kodu źródłowego (słowa kluczowe w popularnych językach programowania pisane są niemal zawsze małą literą).
- ♦ Nie trzeba się przejmować liczbą odstępów pomiędzy kolejnymi elementami zapytania. Będzie się można o tym przekonać przy okazji tworzenia tabeli `ssaki_serengeti`.

Poniższe zapytanie SQL tworzy tabelę `ssaki_serengeti` według schematu z rysunku 17.2:

```
CREATE TABLE ssaki_serengeti
(
  nazwa_potoczna VARCHAR(25) NOT NULL,
  rzad VARCHAR(25) NOT NULL,
  rodzina VARCHAR(25) NOT NULL,
  rodzaj VARCHAR(25) NOT NULL,
  gatunek VARCHAR(25) NOT NULL,
  podgatunek VARCHAR(25) NULL,
);
```

Jak widać, zapytanie kończy się znakiem średnika. Znak ten stanowi sygnał końca każdego zapytania języka SQL (niekiedy można go jednak pominąć — przykłady takich sytuacji zostaną zaprezentowane w dalszej części rozdziału).

Wypełnianie tabel danymi

Po utworzeniu tabel można rozpocząć ich wypełnianie informacjami. Można to robić ręcznie, konstruując kolejne zapytania INSERT, zgodnie z następującą składnią:

```
INSERT INTO nazwa_tabeli VALUES ('wartość1', 'wartość2', 'wartość3', ...);
```

Wykonanie zapytania INSERT powoduje wstawienie do kolejnych kolumn tabeli `nazwa_tabeli` wartości `wartość1`, `wartość2`, `wartość3` i tak dalej. Wartości te tworzą pojedynczy wiersz (rekord) tabeli. Podawane wartości trafiają do kolejnych kolumn wiersza, chyba że w zapytaniu określone zostaną wprost kolumny docelowe (np. wtedy, kiedy wiersz określony jest tylko dla niektórych kolumn). Kolumny docelowe dla wartości określa się następująco:

```
INSERT INTO nazwa_tabeli (kolumna1, kolumna2) VALUES ('wartość1', 'wartość2');
```

Pojedynczym zapytaniem INSERT można też wypełnić więcej niż jeden wiersz; składnia takiego zapytania wygląda następująco:

```
INSERT INTO nazwa_tabeli VALUES ('wartość1', 'wartość2'), ('wartość3', 'wartość4');
```

W powyższym zapytaniu wartości `wartość1` oraz `wartość2` tworzą wiersz pierwszy, wartości `wartość3` i `wartość4` należą do drugiego wstawianego wiersza.

Poniższy przykład ilustruje sposób wprowadzenia do tabeli wiersza opisującego lwa afrykańskiego:

```
INSERT INTO ssaki_serengeti VALUES('Lew afrykański', 'Drapieżne', 'Kotowate',  
  'Panthera', 'Leo', NULL);
```

W bazie danych MySQL, jeśli wiersz nie ma definiować któregoś z ostatnich pól, należy w miejscu wartości dla tego pola podać wartość pustą (NULL). W przypadku bazy danych PostgreSQL ostatnie kolumny mogą po prostu zostać pominięte. Rzecz jasna, kolumny środkowe muszą być zawsze określone wprost — jeśli nie mają wartości, należy zamiast niej podać wartość NULL.

Zwykle zapytanie INSERT osadzone jest w aplikacji stanowiącej interfejs bazy danych, aby użytkownicy wprowadzający dane do bazy nie musieli samodzielnie konstruować zapytań.

Pobieranie informacji z bazy danych

Po to tworzy się bazy danych, aby można było efektywnie i wygodnie korzystać z przechowywanych w nich informacji, pobierając je, porządkując i tworząc na ich podstawie rozmaite zestawienia. Podstawowa operacja pobierania danych implementowana jest w języku SQL zapytaniem SELECT, o następującej składni:

```
SELECT kolumna(-ny) FROM nazwa_tabeli WHERE kryterium_wyszukiwania;
```

Pierwsze dwa elementy zapytania — klauzula SELECT oraz klauzula FROM — są elementami wymaganymi. Klauzula WHERE jest elementem opcjonalnym. W przypadku jego braku zapytanie wybiera wszystkie wiersze tabeli *nazwa_tabeli*.

Kolumny to wyliczenie nazw kolumn, które mają zostać uwzględnione w zbiorze danych wyjściowych. W przypadku, kiedy do zbioru wynikowego ma trafić więcej niż jedna kolumna, nazwy kolumn należy w klauzuli SELECT rozdzielić przecinkami. Zamiast wypisywać nazwy wszystkich kolumn po kolei, można zastosować symbol wieloznaczny — gwiazdkę (*). Dla przykładu, poniższe zapytanie spowoduje wyświetlenie wszystkich kolumn tych wierszy tabeli, które spełniają kryteria wyszukiwania:

```
SELECT * FROM ssaki_serengeti;
```

Jeśli natomiast wydruk ma zawierać np. tylko nazwy potoczne gatunków, zapytanie powinno przyjąć postać:

```
SELECT nazwa_potoczna FROM ssaki_serengeti;
```

Zbiór wynikowy może też zawierać np. tylko rodzaj i nazwę gatunku:

```
SELECT rodzaj, gatunek FROM ssaki_serengeti;
```

Tyle, że wynik powyższego zapytania raczej nie zadowoli biologów. Otóż nazwa zwierzęcia składa się zawsze z nazwy rodzaju i gatunku. Dopiero one tworzą pełną nazwę. Na szczęście zarówno MySQL, jak i PostgreSQL oferują funkcję łączenia ciągów. W obu bazach danych funkcja ta jest implementowana nieco inaczej.

W bazie danych MySQL do połączenia wartości kolumn rodzaj i gatunek można wykorzystać funkcję CONCAT, jak poniżej:

```
SELECT CONCAT(rodzaj, " ", gatunek) AS Pełna_Nazwa FROM ssaki_serengeti;
```

Powyższe zapytanie spowoduje wyświetlenie połączonych wartości kolumn w kolumnie o nazwie Pełna_Nazwa. W wywołaniu funkcji CONCAT drugim argumentem jest ciąg zawierający pojedynczy znak spacji. Gdyby go tam zabrakło, wartości kolumn zostałyby ze sobą „sklejone”, co nie wyglądałoby najlepiej na wydruku.

W bazie danych PostgreSQL funkcja łączenia ciągów wywoływana jest dwoma symbolami potoku (||). Odpowiednikiem prezentowanego powyżej zapytania w wersji dla bazy PostgreSQL byłoby więc następujące zapytanie:

```
SELECT (rodzaj||' '||gatunek) AS Pełna_Nazwa FROM ssaki_serengeti;
```

Tutaj ujmowanie łączonych podciągów w nawiasy jest nieobowiązkowe, choć zwiększa czytelność zapytania. Jak poprzednio, również tym razem ujęty w znaki (tutaj pojedynczego) cudzysłowu znak odstęp pomiędzy łączonymi ciągami wprowadza spację rozdzielającą podciągi optycznie. Brak tego znaku spowodowałby rzecz jasna sklejenie obu podciągów do postaci pojedynczego wyrazu.

Zazwyczaj przy pobieraniu danych interesuje nas jedynie podzbiór wszystkich danych przechowywanych w tabeli czy bazie danych. Zwykle taki podzbiór konstruuje się z wierszy o pewnych charakterystycznych cechach. Wyszukiwanie według tych cech można wymusić, umieszczając w zapytaniu SELECT klauzulę WHERE. Dla przykładu, aby wyszukać w tabeli ssaki_serengeti wszystkie ssaki drapieżne, należałoby skonstruować zapytanie następującej treści:

```
SELECT * FROM ssaki_serengeti WHERE rzad="Drapieżne";
```

Jeśli tabela ssaki_serengeti zawiera dane takie, jak na rysunku 17.2, wynikiem zapytania powinny być wiersze opisujące lwa afrykańskiego, surykatkę, hienę i geparda. To bardzo proste zapytanie — język SQL pozwala na definiowanie dużo bardziej złożonych operacji wyszukiwania. Zapytania takie mogą zawierać w klauzuli WHERE warunki logiczne, określane operatorami AND (logiczny iloczyn) oraz OR (logiczna suma). Jeśli na przykład z uzyskanego poprzednio zbioru wynikowego chcielibyśmy wyodrębnić wyłącznie te drapieżne, które równocześnie nie należą do rodziny hienowatych, zapytanie powinno przybrać następującą postać:

```
SELECT * FROM ssaki_serengeti WHERE rzad="Drapieżne" AND rodzina!="Hienowate";
```

W programowaniu operator != oznacza zazwyczaj „różny od”. Jeśli za tabelę źródłową przyjąć tabelę z rysunku 17.2, to zbiór wynikowy zapytania zawierać będzie wiersze lwa afrykańskiego, surykatki i geparda — hiena, jako hienowata, nie trafi do zbioru wynikowego.

W przypadku, gdyby zbiór wynikowy miał obejmować wszystkie drapieżne i parzystokopytne z wyjątkiem tych, które należą do rodzin hienowatych oraz hipopotamowatych, należałoby zastosować następującą kombinację operatorów AND i OR:

```
SELECT * FROM ssaki_serengeti WHERE rzad="Drapieżne" OR rzad="Parzystokopytne"  
AND rodzina!="Hienowate" AND rodzina!="Hipopotamowate";
```

Powyższe zapytanie wybierze do zbioru wynikowego lwa afrykańskiego, surykatkę, geparda, żyrafę Thomsona, zebrową stepową, geparda i gazelę Thomsona. Pominięte zostaną natomiast hipopotam oraz hiena i słoń afrykański.



Jednym z najczęstszych błędów programistów aplikacji obsługujących bazy danych jest niewłaściwe zastosowanie operatorów logicznych. Na przykład, jeśli ktoś wyrazi życzenie: „Znajdź wszystkie zwierzęta rodzaju Panthera oraz Acinonyx”, to z pozoru żądanie to przełożone wprost na język SQL powinno spowodować wybranie z tabeli wierszy opisujących lwa afrykańskiego i geparda. Jakież będzie zaskoczenie programisty, gdy zapytanie to nie zwróci żadnego wiersza! A stanie się tak, ponieważ po przełożeniu żądania wprost na język SQL kryterium wyszukiwania okazuje się przynależność gatunku do dwóch rodzajów, co w systematyce gatunków jest, rzecz jasna, nie do pomyślenia. Niezależnie więc od liczby wierszy, takie zapytanie nie zwróci żadnego z nich. W miejsce oraz (AND) należy więc zastosować operator lub (OR).

Oczywiście powyższa prezentacja nie wyczerpuje wszystkich możliwości języka SQL. Ale, jak już wspomniano, podrozdział ten nie ma stanowić podręcznika języka SQL; jego zadaniem było wyłącznie zaznajomienie Czytelnika z absolutnymi podstawami tego języka, niezbędnymi każdemu szanującemu się administratorowi baz danych.

Wybór bazy danych: MySQL kontra PostgreSQL

Pierwszym krokiem do korzystania z baz danych w Linuksie jest wybór serwera bazy danych, odpowiadającego konkretnym potrzebom. Dla systemu Linux powstało wiele implementacji serwerów baz danych; są wśród nich zarówno programy dostępne nieodpłatnie, jak i te kosztujące tysiące dolarów. Te ostatnie, jak baza danych Oracle, nie będą jednak omawiane w niniejszej książce. Ten podrozdział zostanie w całości poświęcony dwóm darmowym systemom zarządzania bazami danych: MySQL oraz PostgreSQL.

W obu przypadkach mamy do czynienia z produktami wysokiej jakości; oba umieszczone są na dołączonych do książki płytach CD-ROM. Oba też, pomimo tego, że są rozprowadzane nieodpłatnie, dają sobie radę nawet z bardzo złożonymi projektami. Wykorzystywane są powszechnie w organizacjach komercyjnych, agencjach rządowych (z MySQL korzysta na przykład NASA), instytutach badawczych i placówkach edukacyjnych.

Obie bazy danych są bardzo zaawansowane i prawdopodobnie każda z nich będzie spełniać potrzeby Czytelnika. Różnią się jednak od siebie na tyle, że wybór jednej z nich może wpłynąć na efektywność dostępu do danych bądź prostotę programowania aplikacji bazodanowych. Niektóre z podstawowych cech obu baz danych, jak i różnice w implementacji poszczególnych funkcji, zostaną omówione w kolejnych punktach; informacje te powinny pomóc w wyborze jednej z nich dla konkretnego projektu.

Szybkość

Do niedawna kryterium to wskazywało jednoznacznie na bazę danych MySQL. Baza ta cieszy się opinią bardzo szybkiej. Do niedawna też baza PostgreSQL wypadła w tym porównaniu błodo.

Nowsze wersje bazy danych PostgreSQL zostały jednak znacznie ulepszone (przede wszystkim w zakresie wydajności operacji dostępu do dysku i porządkowania danych). W pewnych sytuacjach, na przykład w okresach największego natężenia żądań dostępu do danych, baza danych oparta na PostgreSQL okazuje się wręcz szybsza od MySQL, co będzie można sprawdzić w następnym podrozdziale. Mimo wszystko sława MySQL wciąż jest jak najbardziej zasłużona.

Blokowanie danych

W bazie danych stosuje się zapobiegający naruszeniu integralności danych mechanizm, polegający na blokowaniu danych na czas dostępu do nich. Póki dane są zablokowane przez jeden proces, inny proces nie może się do nich odwołać. Wszystkie procesy próbujące odwoływać się do zablokowanych danych muszą poczekać na zwolnienie blokady. Po jej zwolnieniu blokadę zakłada następny proces, a reszta dalej musi czekać na swoją kolej.

Oczywiście operacje na bazie danych zajmują zwykle bardzo niewiele czasu, więc w środowiskach o niewielkiej liczbie współbieżnych procesów odwołujących się do bazy danych blokady są zakładane na stosunkowo krótkie okresy, nie wprowadzające żadnych zauważalnych opóźnień dostępu. Jednak już w środowisku, w którym ze wspólnej bazy danych korzysta wielu użytkowników, blokowanie może doprowadzić do zmniejszenia wydajności ich pracy.

W bazie danych MySQL blokowanie implementowane jest w sposób zasadniczo odmienny od przyjętego w bazie danych PostgreSQL.

W starszych wersjach MySQL blokady obejmowały całe tabele, co w przypadku dużego natężenia odwołań prowadziło do powstawania zatorów. Kiedy bowiem jeden z użytkowników modyfikował pojedynczy wiersz tabeli, nikt inny nie mógł równocześnie wprowadzać innych wierszy. Jeśli tabela zawierała, powiedzmy 500 000 wierszy, to na czas realizacji operacji na dowolnym z tych wierszy pozostałe pozostawały niedostępne. W środowiskach z niewielką liczbą użytkowników nie stanowiło to większego problemu, ponieważ większość operacji modyfikacji wiersza przebiega w bardzo krótkim czasie. Jednak przy większej liczbie użytkowników blokowanie całych tabel mogło doprowadzić do zauważalnych opóźnień.

W przypadku bazy danych PostgreSQL blokowanie obejmowało zawsze wyłącznie pojedyncze wiersze. Blokowany jest zawsze ten wiersz, który jest aktualnie modyfikowany. Reszta tabeli może w tym czasie być dostępna dla pozostałych użytkowników. Blokowanie z dokładnością do pojedynczych wierszy znacząco redukuje opóźnienia wynikające ze współbieżnego dostępu do tabeli w środowiskach wielodostępnych. Stąd wniosek, że w środowiskach cechujących się znacznym natężeniem żądań dostępu do danych PostgreSQL może sprawdzać się lepiej niż MySQL.



Implementacja mechanizmu blokowania w bazie danych MySQL omówiona jest szczegółowo na stronie http://www.mysql.com/doc/en/Internal_locking.html. Dystrybucja Fedora zawiera pakiet MySQL w wersji 3.23.58-9, ale już wersja 4.0 (dostępna w czasie przygotowywania wydania) zawiera mechanizm blokowania z dokładnością do pojedynczych wierszy. Dostępna jest również do pobrania wersja beta 4.1 oraz wersja alfa 5.0 pakietu MySQL.

Mechanizm blokowania stosowany w bazie danych PostgreSQL omówiony jest w dokumencie dostępnym pod adresem <http://www.postgresql.org/docs/7.3/interactive/mvcc.html>.

Przetwarzanie transakcji a ochrona spójności danych — reguły ACID

Bazy danych MySQL i PostgreSQL różnią się również jakością ochrony spójności danych. Skrót ACID rozszyfrowywany jest do postaci czterech reguł, które taką spójność zapewniają:

- ◆ **Niepodzielność** (ang. *atomicity*) — zwana też atomowością, odnosi się do głównej cechy atomów, jaką miała być właśnie ich niepodzielność. W kontekście baz danych oznacza to możliwość traktowania szeregu operacji jako niepodzielnych, zwanych też **transakcjami**. Transakcja może zostać wykonana albo w całości, albo wcale. Jeżeli którakolwiek z operacji składowych transakcji nie zostanie wykonana, należy unieważnić całą transakcję.
- ◆ Jeśli, na przykład, w ramach transakcji dojdzie do usunięcia pierwotnego wiersza, ale nie uda się — czy to w wyniku awarii zasilania, czy też załamania serwera — zrealizować operacji wprowadzenia nowego wiersza, niepodzielność transakcji powinna zagwarantować przywrócenie wartości pierwotnej wierszowi, którego aktualizacja została zakłócona.
- ◆ **Spójność** (ang. *consistency*) — oznacza, że żadna z transakcji nie może spowodować przejścia bazy danych ze stanu spójności do stanu niespójności. Stan niespójności objawia się naruszeniem zdefiniowanych dla danych relacji i może być spowodowany błędem w aplikacji odwołującej się do danych, przekłamaniami transmisji sieciowych i tym podobnymi. Gwarancja spójności to gwarancja, że każda z transakcji, która miałaby naruszyć spójność bazy danych, zostanie wycofana. Zapobiega to na przykład usunięciu wiersza czy pola, do którego odnoszą się inne wiersze bazy danych (usunięcie takie spowodowałoby „osierocenie” owych wierszy).
- ◆ **Izolacja** (ang. *isolation*) — izolacja to gwarancja pełnej niezależności wszystkich transakcji. W przypadku dwóch transakcji manipulujących tymi samymi danymi nie może dojść do sytuacji, w której modyfikacje wprowadzane przez jedną transakcję zakłócają działanie drugiej. Sposób implementacji mechanizmu izolacji jest zwykle konfigurowany przez programistę bazy danych. Jednym z podstawowych środków zapewniania izolacji jest blokowanie danych.
- ◆ **Trwałość** (ang. *durability*) — po zatwierdzeniu transakcji wprowadzone przez nią modyfikacje danych nie mogą zostać utracone z powodu awarii zasilania czy błędu systemu, sieci i tak dalej. Trwałość jest zwykle gwarantowana mechanizmem

dzienników transakcji. W przypadku awarii komputera serwer bazy danych, po załączeniu zasilania, może przeanalizować zawartość plików dziennika transakcji i utrwalić te wprowadzone przez transakcje zmiany, których nie udało się zapisać na dysk przed awarią.

Baza danych PostgreSQL spełnia wszystkie cztery kryteria ACID; MySQL szybko zaś nadrabia zaległości. Dla przykładu, najnowsza wersja (dostępna w ramach projektu Fedora Project oraz na stronie domowej PostgreSQL, dostępnej pod adresem <http://www.mysql.com/>) obsługuje już transakcje, co oznacza mniejsze ryzyko utraty danych w wyniku awarii. Wielu programistów uważa jednak, że to właśnie PostgreSQL jest pod tym względem lepiej przystosowany do środowisk z wieloma użytkownikami i wielką liczbą regularnie aktualizowanych danych.



Firma rozprawdzająca MySQL Server, czyli MySQL AB zaleca regularne wykonywanie kopii zapasowych bazy danych MySQL. W nowszych wersjach bazy zaimplementowane zostały jednak mechanizmy zabezpieczające — wersja 4.0 zawiera szereg udoskonaleń implementacji reguł ACID, w tym automatyczne przywracanie danych po awarii. Więcej informacji na ten temat zawiera dokumentacja publikowana w witrynie <http://www.mysql.com/>.

Podzapytania SQL

Zastosowanie podzapytań pozwala na połączenie w jedną kilku operacji, przy czym operacje te mogą odwoływać się do zbiorów wynikowych pozostałych operacji. Podzapytania umożliwiają konstruowanie naprawdę złożonych zapytań SQL. Ich zastosowanie eliminuje też ryzyko modyfikacji danych, które mogłyby zajść pomiędzy poszczególnymi operacjami, gdyby były one wykonywane osobno. Podzapytania obsługiwane są tylko przez bazę danych PostgreSQL; w bazie MySQL możliwość ta jest niedostępna, choć istnieje kilka sposobów symulacji podzapytań.

Import danych tabeli z plików tekstowych

W obu omawianych bazach danych zapytania tworzące tabelę bazy danych można wprowadzać z poziomu wiersza polecenia programów-klientów bazy danych. Operacja taka jest jednak dość ryzykowna: w przypadku najmniejszej nawet pomyłki w tekście zapytania trzeba całość pisać od nowa. Na szczęście zarówno MySQL, jak i PostgreSQL, potrafią też odczytywać zapytania SQL ze zwykłych plików tekstowych. Można więc co dłuższe i częściej wykorzystywane zapytania zapisać w zwykłym pliku tekstowym, nadając mu ewentualnie rozszerzenie *.sql*, informujące o wartości pliku.

Zapytania SQL zapisane w pliku tekstowym (tu *ssaki.sql*) można wprowadzać do bazy danych za pośrednictwem następujących poleceń (pierwsze z nich dotyczy bazy danych MySQL, drugie — bazy danych PostgreSQL):

```
$ mysql nazwa_bazy_danych < ssaki.sql
$ psql nazwa_bazy_danych < ssaki.sql
```

Oczywiście, aby takie polecenia miały szansę zadziałać, muszą być wykonywane w imieniu użytkownika dysponującego odpowiednimi uprawnieniami w bazie danych *nazwa_bazy_danych*. Patrz podrozdział „Przyznawanie i odbieranie uprawnień w bazie danych MySQL” oraz „Tworzenie kont użytkowników bazy danych MySQL”

Języki proceduralne i wyzwalacze

Język proceduralny to w tym kontekście zewnętrzny język programowania, który można wykorzystać do programowania funkcji i procedur bazy danych. Można w ten sposób implementować funkcje niedostępne z poziomu języka SQL. **Wyzwalacz** z kolei to procedura lub funkcja zewnętrzna, która jest uruchamiana w reakcji na określone zdarzenie bazy danych. Wyzwalacz może na przykład zgłaszać wyjątek, kiedy zapytanie INSERT definiuje dla którejś z kolumn nieoczekiwaną bądź niepoprawną wartość.

Dla przykładu, w bazie danych wykorzystywanej do śledzenia populacji zwierząt na pewnym obszarze, można wykorzystać wyzwalacz do podnoszenia wyjątku w sytuacji, kiedy użytkownik wprowadzi współrzędne obserwacji, które nie mają (z punktu widzenia obszaru obserwacji) żadnego sensu. W bazie danych PostgreSQL można taki wyzwalacz napisać w języku proceduralnym o nazwie PL/pgSQL. MySQL dysponuje co prawda ograniczoną liczbą procedur wbudowanych i wyzwalaczy, jednak nie udostępniła programiście własnego języka proceduralnego. Oznacza to, że w bazach danych MySQL nie można stosować wyzwalaczy; podobne efekty można jednak osiągnąć przez pomyślowe programowanie aplikacji klienckich.

Dostępne aplikacje

W tym momencie Czytelnik może stwierdzić, że dalsze porównywanie, z uwagi na wykryte dotychczas braki MySQL, nie ma większego sensu. Nie jest to jednak prawdą, ponieważ MySQL ma zaletę, której brak bazie PostgreSQL — dużą liczbę gotowych aplikacji. Dla przykładu, aby utworzyć na stronie WWW forum dyskusyjne, można skorzystać z wykorzystującej bazę danych MySQL programu Phorum. Pozwala to na zaoszczędzenie mnóstwa pracy, która byłaby potrzebna w przypadku samodzielnego implementowania forum.



Choć faktycznie większość z opisywanych wcześniej funkcji jest w bazie MySQL niedostępna, to na rynku dostępna jest ulepszona wersja bazy danych, o nazwie MySQL-Max, przystosowana do obsługi środowisk produkcyjnych opartych na systemie Linux z najnowszą wersją jądra. Więcej informacji o tym produkcie publikowanych jest w witrynie <http://www.mysql.com/>.

Bazy danych i Fedora — pierwsze kroki

Aby udostępnić użytkownikom systemu Fedora usługi baz danych, należy wykonać szereg czynności, z których pierwszą jest poprawne uruchomienie serwera bazy danych (zwykle realizowane w ramach rozruchu systemu). Użytkownik *root* musi więc upewnić się, czy serwer bazy danych jest uwzględniony na liście usług do uruchomienia na poszczególnych poziomach uruchomieniowych (patrz rozdział 7., „Zarządzanie usługami systemowymi”). Następnie należy utworzyć konto użytkownika głównego bazy danych i zainicjalizować serwer (w przypadku baz danych MySQL polega to na wykonaniu polecenia `mysql_install_db`; szczegóły opisane są w punkcie „Inicjalizowanie katalogu danych bazy MySQL”). Użytkownik główny bazy danych musi otrzymać hasło, po czym należy utworzyć jedną lub kilka baz danych. Dopiero wtedy użytkownik główny bazy danych może przystąpić do wprowadzania użytkowników i przypisywania im uprawnień w ramach utworzonych baz danych.

Instalowanie i konfigurowanie bazy danych MySQL

Dystrybucja Fedora zawiera stabilną, darmową wersję bazy danych MySQL. Oprogramowanie bazy MySQL można też pobrać z witryny <http://www.mysql.com/>. Oprogramowanie to jest tam udostępniane w postaci kodu źródłowego oraz wersji skompilowanej, jak również w postaci pakietu RPM. Instalację oprogramowania MySQL można przeprowadzić w ramach pierwotnej instalacji systemu, albo później skorzystać z programu `system-config-packages`. Sposób instalowania pakietów RPM pokazany był w rozdziale 8.

W tym podrozdziale opisana zostanie jedynie procedura instalacji oprogramowania MySQL z kodu źródłowego i postaci skompilowanej; opis dotyczyć będzie wersji 4.0.



Wskazówka Dużo prostszą procedurą jest instalacja oprogramowania MySQL z pakietu RPM. W takim przypadku wszystkie niezbędne skrypty uruchomieniowe są instalowane automatycznie; automatycznie są też tworzone niezbędne do działania bazy konta użytkowników oraz grupy użytkowników. Instalacja z kodu źródłowego czy postaci skompilowanej jest znacznie bardziej pracochłonna, pozwala jednak na dostosowanie instalacji do własnych potrzeb.

Wersja skompilowana rozprowadzana jest w postaci 9,8-megabajtowego archiwum. Archiwum to zawiera zestaw wskazówek co do procedury instalacji, zapisanych w pliku *INSTALL-BINARY*. Są w nim opisane kolejne etapy instalacji podstawowej, składającej się z szeregu poleceń tworzących grupę i użytkownika `mysql`, umieszczających pliki binarne w odpowiednim podkatalogu katalogu `/usr/local` oraz ustawiające dla nowoutworzonego użytkownika i grupy uprawnienia dostępu do owego katalogu.

Na początek należy jednak rozpakować archiwum:

```
# tar zxvf mysql-standard-4.0.20-pc-linux-i686.tar.gz
```

Kod źródłowy rozprowadzany jest pod postacią 15,2-megabajtowego archiwum (w formacie SRPM bądź formacie programu `tar`). Po jego pobraniu i rozpakowaniu należy archiwum rozpakować (odpowiednio: poleceniem `rpm` bądź poleceniem `tar`). Archiwum zawiera plik wskazówek odnośnie kompilacji i instalacji oprogramowania, zapisany pod nazwą *INSTALL-BINARY*. Plik zawiera instrukcje dotyczące instalacji „szybkiej”, podobne do tych odnoszących się do wersji skompilowanej.

Po zakończeniu wszelkich wymaganych czynności poinstalacyjnych można przejść do katalogu `/usr/local/mysql` i uruchomić serwer. Do uruchamiania serwera służy skrypt `mysql_safe` (znajdujący się w katalogu `/usr/local/mysql/bin`):

```
# bin/mysqld_safe --user=mysql &
[1] 24798
Starting mysqld daemon with databases from /var/lib/mysql
```

W przypadku instalowania oprogramowania bazy danych MySQL z pakietu RPM, niezbędne wpisy użytkownika i grupy serwera bazy danych MySQL zostaną automatycznie wprowadzone do odpowiednich plików systemowych. W przypadku instalacji samodzielnej należy owe wpisy utworzyć; zarówno użytkownik serwera MySQL, jak i jego

grupa, powinny nosić nazwę `mysql`. Konto użytkownika i grupy `mysql` powinny być skonfigurowane tak, aby zablokować możliwość logowania się w imieniu użytkownika serwera bazy danych — konto `mysql` służy wyłącznie do uruchamiania programu serwera. Tworzeniu kont użytkowników poświęcony był rozdział 9., zatytułowany „Zarządzanie kontami użytkowników”.

Inicjalizowanie katalogu danych bazy MySQL

Po zainstalowaniu oprogramowania MySQL należy zainicjalizować **tabelę uprawnień** (ang. *grant table*), czyli tabelę definiującą uprawnienia dostępu do wszystkich baz danych i tabel (a nawet poszczególnych kolumn). Służy do tego program `mysql_install_db`, umieszczony w podkatalogu *scripts* katalogu instalacyjnego bazy danych MySQL. W przypadku instalacji z kodu źródłowego, oprogramowanie bazy danych instalowane jest domyślnie w katalogu `/usr/local/mysql`. Program `mysql_install_db` inicjalizuje tabelę uprawnień i tworzy na potrzeby bazy danych konto użytkownika głównego.



Właścicielem katalogu danych MySQL musi być użytkownik, w imieniu którego uruchamiany jest serwer bazy danych MySQL (prawo własności katalogu można zmienić za pomocą polecenia `chown`). Użytkownik ten powinien być również jedynym użytkownikiem posiadającym jakiegokolwiek uprawnienia w katalogu (innymi słowy, należy owe uprawnienia zdefiniować za pomocą programu `chmod` jako 700). Każda inna konfiguracja to luka w zabezpieczeniach i zagrożenie dla spójności bazy danych.

Skrypt `mysql_install_db` powinien być uruchamiany przez użytkownika `root`:

```
# mysql_install_db
Installing all prepared tables
040716 13:35:30 ./bin/mysqld: Shutdown Complete
```

To start `mysqld` at boot time you have to copy `support-files/mysql.server` to the right place for your system

PLEASE REMEMBER TO SET A PASSWORD FOR THE MySQL root USER !
To do so, start the server, then issue the following commands:
`./bin/mysqladmin -u root password 'new-password'`
`./bin/mysqladmin -u root -h tiger.ck.polsl.gliwice.pl password 'new-password'`
See the manual for more instructions.

NOTE: If you are upgrading from a MySQL <= 3.22.10 you should run the `./bin/mysql_fix_privilege_tables`. Otherwise you will not be able to use the new GRANT command!

You can start the MySQL daemon with:
`cd . ; ./bin/mysqld_safe &`

You can test the MySQL daemon with the benchmarks in the 'sql-bench' directory:
`cd sql-bench ; perl run-all-tests`

Please report any problems with the `./bin/mysqlbug` script!

The latest information about MySQL is available on the web at
<http://www.mysql.com>
Support MySQL by buying support/licenses at <https://order.mysql.com>

Program inicjalizujący przygotowuje bazę danych MySQL do udostępnienia w systemie, wyświetlając przy tym przydatne informacje. Kolejną niezbędną czynnością jest określenie hasła dla głównego użytkownika bazy danych MySQL (patrz punkt następny).



Domyślnie konto użytkownika głównego bazy danych nie posiada hasła. Jest to jedna z pierwszych luk, jakie należy po instalacji oprogramowania serwera bazy danych bezzwłocznie załatać — użytkownik główny bazy danych ma dostęp do wszelkich jej ustawień.

Przypisywanie hasła do konta użytkownika głównego bazy danych MySQL

Aby do konta użytkownika głównego bazy danych przypisać hasło, należy w imieniu tego użytkownika nawiązać połączenie z serwerem bazy danych. Służy do tego polecenie `mysql -u root`. Polecenie to nawiązuje połączenie z serwerem za pośrednictwem klienta MySQL (patrz podrozdział „Program klienta bazy danych MySQL”).

```
# mysql -u root
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 1 to server version: 4.0.20-standard
```

```
Type 'help;' or '\h' for help. Type '\c' to clear the buffer.
```

```
mysql>
```

Kiedy konsola wyświetli znak zachęty klienta MySQL, należy wprowadzić następujące polecenie:

```
mysql> SET PASSWORD FOR root = PASSWORD("tajnehasło");
Query OK, 0 rows affected (0.00 sec)
```

Ciąg *tajnehasło* należy rzecz jasna zastąpić własnym hasłem. Polecenie `SET PASSWORD` można też wykorzystywać do przypisywania (i modyfikacji) haseł do kont innych użytkowników bazy danych.

Po wprowadzeniu hasła można opuścić program klienta MySQL; wystarczy w tym celu wpisać polecenie `exit`.

Tworzenie bazy danych

W MySQL bazy danych tworzy się za pośrednictwem zapytania `CREATE DATABASE`. Wcześniej jednak należy nawiązać połączenie z serwerem MySQL, wpisując `mysql -u root -p` i naciskając klawisz *Enter* — polecenie to powoduje nawiązanie połączenia z serwerem w imieniu użytkownika głównego bazy danych; próba nawiązania połączenia zostanie poprzedzona monitem o podanie hasła. Jeśli hasło będzie właściwe, na ekranie ukaże się znak zachęty klienta MySQL. Dopiero wtedy można wprowadzić zapytanie `CREATE DATABASE`. Dla przykładu, prezentowane poniżej zapytanie tworzy bazę danych o nazwie zwierzeta:

```
# mysql -u root -p
```

```

Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 2 to server version: 4.0.20-standard

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> CREATE DATABASE zwierzeta;
Query OK, 1 row affected (0.00 sec)
mysql>

```

Innym sposobem utworzenia bazy danych jest skorzystanie z pośrednictwa programu `mysqladmin`, który udostępni użytkownikowi głównemu bazy danych opcję `create`, przyjmującą argument w postaci nazwy bazy danych. Dla przykładu, aby utworzyć bazę danych o nazwie `gady`, należałoby skonstruować następujące polecenie:

```
# mysqladmin -u root -p create gady
```

Przyznawanie i odbieranie uprawnień w bazie danych MySQL

Administrator bazy danych będzie zapewne chciał przydzielić sobie pewne **uprawnienia**, których nie będą mieli pozostali użytkownicy bazy danych. Uprawnienia te (ang. *privileges*), mogą być przyznawane i odbierane na czterech różnych poziomach:

- ♦ na poziomie globalnym — uprawnienie globalne obejmuje wszystkie bazy danych serwera;
- ♦ na poziomie bazy danych — obejmuje wszystkie tabele danej bazy danych;
- ♦ na poziomie tabeli — obejmuje wszystkie kolumny danej tabeli w określonej bazie danych;
- ♦ na poziomie kolumny — pozwala na dostęp do określonej kolumny w ramach pewnej tabeli i bazy danych.



W rozdziale nie ma miejsca na omówienie wszystkich uprawnień dostępnych w bazie danych MySQL. Ich znaczenie można znaleźć w dokumentacji bazy danych.

Aby dodać konto użytkownika, należy z serwerem bazy danych nawiązać połączenie w imieniu użytkownika głównego, korzystając z polecenia `mysql -u root -p`. Przed nawiązaniem połączenia wyświetlony zostanie monit o podanie hasła. Dopiero po jego wprowadzeniu klient MySQL wyświetli znak zachęty.

Do przyznawania uprawnień służy zapytanie `GRANT`. Jego składnia jest następująca:

```
GRANT uprawnienie ON obiekt TO uzytkownik IDENTIFIED BY 'haslo';
```

Uprawnienia, jakie można przyznawać użytkownikom, identyfikowane są słowami kluczowymi. I tak, słowo kluczowe `ALL` oznacza komplet uprawnień.

Obiekt określa zasób, którego dotyczy uprawnienie.

Użytkownik to nazwa konta użytkownika, który otrzyma uprawnienia.

Hasło to oczywiście hasło, jakie ma zostać przypisane do konta użytkownika. W przypadku, kiedy polecenie dotyczy użytkownika już posiadającego konto opatrzone hasłem, klauzulę IDENTIFIED BY i hasło można pominąć.

Dla przykładu, aby użytkownikowi fefe przyznać komplet uprawnień do bazy danych zwierzeta, należy skonstruować następujące polecenie:

```
GRANT ALL ON zwierzeta.* TO fefe IDENTIFIED BY 'haslofefe';
```

Warto dodać jeszcze wpis:

```
INSERT INTO USER (HOST,USER,PASSWORD) VALUES ('localhost','fefe',Password('haslofefe'));
```

i przeładować:

```
FLUSH PRIVILEGES;
```

Użytkownik fefe będzie mógł po wykonaniu powyższego polecenia nawiązać połączenie z bazą danych zwierzeta, na komputerze lokalnym (localhost), w której będzie dysponował kompletem uprawnień, z możliwością tworzenia i usuwania tabel włącznie. Dla przykładu, użytkownik fefe może zalogować się w bazie danych korzystając z następującego polecenia:

```
$ mysql -h localhost -u fefe -p zwierzeta
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 21 to server version: 4.0.20-standard

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql>
```



Dodatkowe opcje wywołania programu klienta MySQL opisane są w podrozdziale „Program klienta bazy danych MySQL”.

Kiedy użytkownikowi fefe nie będą już potrzebne tak rozległe uprawnienia, można je mu odebrać. Służy do tego polecenie REVOKE. Użytkownikowi fefe można odebrać cały pakiet uprawnień naraz:

```
REVOKE ALL ON zwierzeta FROM fefe;
```

Bardziej zaawansowane aspekty zarządzania bazami danych, uprawnieniami i mechanizmami zabezpieczającymi to niestety temat na osobną książkę. Nieco dokumentów na ten temat wymienionych jest w kończącym rozdział o bazach danych podrozdziale „Warto zajrzeć”. Nie zaszkodzi też sięgnąć po jedną z wielu książek poświęconych bazie MySQL, np. *PHP and MySQL Development*², autorstwa Luke'a Wellinga i Laury Thompson (wydawnictwo Sams Publishing).

² Wydanie polskie: *PHP i MySQL. Tworzenie stron WWW*, Helion, 2002.

Instalowanie i konfigurowanie bazy danych PostgreSQL

Najnowsze wersje oprogramowania serwera baz danych PostgreSQL w postaci kodu źródłowego i pakietów skompilowanych publikowane są na stronach witryny <http://www.postgresql.org/>. Oprogramowanie bazy danych PostgreSQL dystrybuowane jest także w postaci pakietów RPM. Pakietów tych jest kilka; minimalna instalacja powinna obejmować pakiety *postgresql*, *postgresql-server* oraz *postgres-libs*. W razie wątpliwości co do konieczności instalowania pozostałych pakietów warto zapoznać się z treścią pliku */usr/share/doc/postgresql-7.4.2/README.rpm-dist* (plik ten jest tam umieszczany w ramach instalacji pakietu *postgresql*).

W przypadku instalowania oprogramowania serwera PostgreSQL niezbędne do działania serwera konto użytkownika *postgres* (czyli użytkownika, w imieniu którego uruchamiany jest serwer bazy danych PostgreSQL) tworzone jest automatycznie. Można to sprawdzić następującym poleceniem:

```
$ less /etc/passwd | grep postgres
postgres:x:26:26:PostgreSQL Server:/var/lib/pgsql:/bin/bash
```

W przypadku samodzielnej instalacji, czy to z pakietu kodu źródłowego, czy pakietu oprogramowania skompilowanego, czy nawet pakietów RPM innych niż te pochodzące z dystrybucji Fedora, należy konto użytkownika *postgres* utworzyć samodzielnie. Konto owego użytkownika nie powinno dopuszczać logowania; konto to może być dostępne jedynie użytkownikowi uprzywilejowanemu, po wydaniu przez niego polecenia *su* (sposób dodawania do systemu kont nowych użytkowników opisany został w rozdziale 9.). Po utworzeniu konta użytkownika *postgres* można przystąpić do instalowania pobranych pakietów oprogramowania; w przypadku pakietów RPM najlepiej wykorzystać do tego polecenie *rpm -i*.

Inicjalizowanie katalogu danych bazy PostgreSQL

Po pomyślnym zainstalowaniu pakietów RPM należy przystąpić do inicjalizacji katalogu danych. W tym celu należy ów katalog najpierw utworzyć, a do tego potrzebne są uprawnienia użytkownika uprzywilejowanego. W dalszych przykładach niejawnie zakładamy, że katalogiem danych jest */usr/local/pgsql/data/*.

Za pośrednictwem polecenia *mkdir* należy utworzyć katalog */usr/local/pgsql/data* i zmienić prawo własności do katalogu (poleceniami *chgrp* i *chown*) tak, aby jego wyłącznym właścicielem był użytkownik *postgres*. Następnie za pomocą polecenia *su* należy przyjąć uprawnienia użytkownika *postgres* i wykonać w jego imieniu następujące polecenia:

```
# mkdir /usr/local/pgsql
# chown -R postgres /usr/local/pgsql
# chgrp -R postgres /usr/local/pgsql
# su - postgres
-bash-2.05b$ initdb -D /usr/local/pgsql/data
The files belonging to this database system will be owned by user "postgres".
```


This user must also own the server process.

The database cluster will be initialized with locale pl_PL.UTF-8.

```
fixing permissions on existing directory /usr/local/pgsql/data... ok
creating directory /usr/local/pgsql/data/base... ok
creating directory /usr/local/pgsql/data/global... ok
creating directory /usr/local/pgsql/data/pg_xlog... ok
creating directory /usr/local/pgsql/data/pg_clog... ok
selecting default max_connections... 100
selecting default shared_buffers... 1000
creating configuration files... ok
creating template1 database in /usr/local/pgsql/data/base/1... ok
initializing pg_shadow... ok
enabling unlimited row size for system tables... ok
initializing pg_depend... ok
creating system views... ok
loading pg_description... ok
creating conversions... ok
setting privileges on built-in objects... ok
creating information schema... ok
vacuuming database template1... ok
copying template1 to template0... ok
```

Success. You can now start the database server using:

```
/usr/bin/postmaster -D /usr/local/pgsql/data
or
/usr/bin/pg_ctl -D /usr/local/pgsql/data -l logfile start
```

Powyższe komunikaty informują o pomyślnym zainicjalizowaniu katalogu danych oraz stosownym ustawieniu uprawnień do tego katalogu.



Program `initdb` ustawia uprawnienia dostępu do katalogu danych na 700. Nie należy tego trybu dostępu zmieniać — każda inna wartość naraża na szwank bezpieczeństwo danych.

Proces nasłuchu serwera bazy danych PostgreSQL, `postmaster`, uruchamia się następującym poleceniem (naturalnie, wciąż w imieniu użytkownika `postgres`):

```
-bash-2.05b$ postmaster -D /usr/local/pgsql/data &
[1] 26196
LOG:  database system was shut down at 2004-07-16 15:22:49 CEST
LOG:  checkpoint record is at 0/9B1058
LOG:  redo record is at 0/9B1058; undo record is at 0/0; shutdown TRUE
LOG:  next transaction ID: 536; next OID: 17142
LOG:  database system is ready
```

W przypadku, kiedy na katalog danych wybrany zostanie katalog inny niż `/usr/local/pgsql/data`, należy odpowiednio dostosować wywołanie programu `postmaster`.



W dystrybucji Fedora Core 2 katalog danych serwera bazy danych PostgreSQL to `/var/lib/pgsql/data/`. Nie jest to jednak najlepsze miejsce do przechowywania danych, ponieważ w większości systemów katalog `/var` montowany jest na partycjach nie dysponujących wystarczającą ilością miejsca na większą ilość danych. W przypadku podjęcia decyzji o zmianie katalogu domyślnego należy stosownie zmodyfikować skrypt rozruchowy serwera PostgreSQL, zapisany w pliku `/etc/rc.d/init.d/postgres`.

Tworzenie bazy danych

Tworzenie baz danych dla serwera PostgreSQL nie jest czynnością wcale skomplikowaną. Musi jednak być przeprowadzona przez użytkownika dysponującego uprawnieniami do tworzenia baz danych dla serwera PostgreSQL, początkowo zaś uprawnieniami takimi dysponuje wyłącznie użytkownik `postgres`. Utworzenie bazy danych wymaga w takiej sytuacji wykonania następujących poleceń:

```
# su - postgres
-bash-2.05b$ createdb baza_danych
CREATE DATABASE
```

Gdzie `baza_danych` to nazwa bazy danych do utworzenia.

Program `createdb` jest w rzeczywistości wyłącznie swego rodzaju otoczką programu-klienta `psql`, ułatwiająca tworzenie baz danych przez wyeliminowanie konieczności uruchomienia klienta PostgreSQL. Równie dobrze można jednak utworzyć bazę danych bezpośrednio w programie klienta `psql`, wykonując w nim polecenie `CREATE DATABASE`:

```
CREATE DATABASE baza_danych;
```

Zanim będzie można sensownie wykorzystać program klienta serwera PostgreSQL trzeba utworzyć przynajmniej jedną bazę danych. Bazę tę należy tworzyć, będąc zalogowanym w imieniu użytkownika `postgres`. To zaś można osiągnąć wyłącznie z konta użytkownika uprzywilejowanego, z którego należy wykonać polecenie `su` aby przejść uprawnienia użytkownika `postgres`. Aby połączyć się z nowoutworzoną bazą danych, należy uruchomić program `psql`, podając w wierszu wywołania nazwę bazy danych:

```
$ psql baza_danych
```

Jeśli w wierszu wywołania programu-klienta serwera PostgreSQL nie zostanie określona nazwa bazy danych, program spróbuje nawiązać połączenie z bazą danych o nazwie identycznej z nazwą użytkownika, który uruchomił program `psql`.

Tworzenie kont użytkowników bazy danych PostgreSQL

W celu utworzenia konta użytkownika bazy danych należy przejść uprawnienia użytkownika `postgres`. Następnie można skorzystać z polecenia `createuser`, tworzącego konto nowego użytkownika; ów użytkownik może od razu otrzymać uprawnienia dostępu do baz danych i możliwość tworzenia nowych użytkowników bazy danych. Wystarczy twierdząco odpowiedzieć na oba zadane przy okazji tworzenia nowego konta pytania:

```
-bash-2.05b$ createuser bebe
Shall the new user be allowed to create databases? (y/n) y
Shall the new user be allowed to create more new users? (y/n) y
CREATE USER
```

W powyższym przykładzie użytkownik `bebe` ma zaraz po utworzeniu uprawnienia do tworzenia nowych baz danych oraz kont kolejnych użytkowników (oczywiście z uprawnieniami tymi nie należy przesadzać).

Konto nowego użytkownika można też zainicjalizować z poziomu programu klienta `psql`, uruchamiając go z argumentem określającym bazę danych i wpisując w wierszu poleceń programu zapytanie `CREATE USER`, jak poniżej:

```
CREATE USER bebe;
```



PostgreSQL pozwala na pominięcie w zapytaniu tworzącym użytkownika klauzuli `WITH PASSWORD`. Nowoutworzone konto nie jest wtedy zabezpieczane żadnym hasłem. To poważne naruszenie zasad bezpieczeństwa, więc każdorazowo należy pamiętać o uzupełnieniu zapytania tworzącego konto nowego użytkownika klauzulą `WITH PASSWORD` i ciągiem hasła.



Po zakończeniu pracy program klienta `psql` można zamknąć, wprowadzając polecenie `\q`. Spowoduje to powrót do wiersza poleceń powłoki.

Usuwanie kont użytkowników bazy danych PostgreSQL

Do usuwania kont użytkownika bazy danych PostgreSQL służy polecenie `dropuser`, uzupełnione nazwą konta użytkownika:

```
-bash-2.05b$ dropuser bebe  
DROP USER
```

Można też — korzystając z pośrednictwa programu `psql` — zalogować się do serwera bazy danych i wykonać zapytanie `DROP USER`. Oto przykład:

```
-bash-2.05b$ psql demodb  
Welcome to psql 7.4.2, the PostgreSQL interactive terminal.
```

```
Type: \copyright for distribution terms  
      \h for help with SQL commands  
      \? for help on internal slash commands  
      \g or terminate with semicolon to execute query  
      \q to quit
```

```
baza=# DROP USER bebe;  
DROP USER  
baza=# \q  
-bash-2.05b$
```

Przyznawanie i odbieranie uprawnień użytkownikom bazy danych PostgreSQL

Podobnie, jak w bazie danych MySQL, przyznawanie i odbieranie uprawnień w bazie danych PostgreSQL przeprowadza się za pośrednictwem zapytań `GRANT` oraz `REVOKE`. Składnia zapytania jest tu nieco inna niż w przypadku MySQL, jako że PostgreSQL nie uwzględnia klauzuli `IDENTIFIED BY` — użytkownicy baz danych PostgreSQL otrzymują hasła w ramach procedury tworzenia konta użytkownika (`CREATE USER`). Oto składnia zapytania `GRANT`:

```
GRANT uprawnienie ON obiekt TO uzytkownik;
```

Dla przykładu, poniższe polecenie przyznaje użytkownikowi `fefe` wszelkie uprawnienia w ramach bazy danych `baza`:

```
GRANT ALL ON DATABASEbaza TO fefe;
```

Do odbierania uprawnień służy zapytanie `REVOKE`. Oto przykład jego zastosowania:

```
REVOKE ALL ON DATABASE baza FROM fefe;
```

Powyższe polecenie odbiera użytkownikowi `fefe` wszystkie uprawnienia, jakimi dysponował on w ramach bazy danych `baza`.

Zaawansowane zagadnienia administracji bazą danych to kwestie bardzo złożone. Nie sposób ująć ich wszystkich w jednym rozdziale, którego zadaniem nie było zresztą tylko wprowadzenie Czytelników w zagadnienia administracji bazami danych. Więcej informacji na temat administrowania bazami danych PostgreSQL znaleźć można w dokumentacji bazy danych oraz w licznych książkach, poświęconych bazie PostgreSQL w całości (jak choćby *PostgreSQL*³ wydawnictwa Sams Publishing).

Programy-klienty baz danych

Zarówno w bazie danych MySQL, jak i PostgreSQL, system zarządzania bazą danych implementowany jest w architekturze klient-serwer. Najprościej rzecz ujmując, serwer bazy danych obsługuje żądania dostępu do bazy danych; żądania te są przesyłane przez program klienta, który również odbiera odpowiedzi serwera i prezentuje otrzymane zbiory wyników użytkownikowi.

Użytkownik nigdy nie komunikuje się bezpośrednio z serwerem bazy danych, nawet, jeśli pracuje na tym samym węźle, na którym uruchomiony jest proces serwera. We wszystkich odwołaniach do bazy danych pośredniczy program klienta; program ten może być uruchomiony równie dobrze na komputerze, na którym działa serwer, jak i łączyć się z tym komputerem (i procesem serwera) z drugiego końca świata.

Użytkownicy obu omawianych systemów zarządzania bazami danych mają do dyspozycji programy-klienty, udostępniające interfejs wierszowy. To bardzo prymitywny i z tego względu rzadko wykorzystywany przez użytkowników końcowych sposób komunikowania się z bazą danych. Administrator bazy danych nie ma jednak zazwyczaj wyboru, ponieważ tylko to prymitywne narzędzie jest na tyle elastyczne, że pozwala na testowanie nowych zapytań bez konieczności osadzania ich w kodzie aplikacji użytkowych i konstruowania ich interfejsu. W kolejnych częściach niniejszego rozdziału Czytelnik będzie miał jednak okazję poznać klienta bazy danych MySQL działającego w trybie graficznym, jak również szereg interfejsów WWW, dostępnych i dla baz MySQL, i baz danych PostgreSQL.

³ Albo jednej z poświęconych tej bazie danych książek, wydanych po polski: *PostgreSQL. Praktyczny przewodnik* (Helion, 2002) czy *Bazy danych i PostgreSQL. Od podstaw* (Helion, 2002).

W kolejnych punktach przetestowane zostaną dwie najpopularniejsze metody komunikacji ze zdalnymi bazami danych, metoda odwoływania się do serwera lokalnego oraz koncepcja korzystania z bazy danych za pośrednictwem interfejsu WWW.

Uwaga

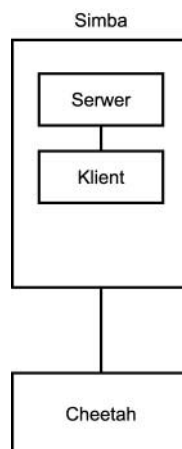
Konfigurując bazę danych warto starannie przemyśleć kwestie uprawnień dostępu do bazy danych. Należy się przy tym zastanowić, czy użytkownicy powinni mieć możliwość tworzenia i usuwania baz danych, czy może powinni zostać ograniczeni do korzystania wyłącznie z tych baz, które zastaną. Nie wszyscy użytkownicy powinni mieć możliwość modyfikowania rekordów poszczególnych baz danych — część z nich powinna zadowolić się dostępem do danych w trybie do odczytu. A co z tzw. resztą świata? Przecież użytkownicy z zewnątrz, łączący się za pośrednictwem sieci Internet, nie powinni być może mieć żadnego dostępu do baz danych? Wszystkie te wątpliwości spadają na głowę administratora bazy danych.

Dostęp do bazy danych za pośrednictwem SSH

W tym punkcie przedstawione zostaną dwa często spotykane tryby dostępu do zdalnych serwerów baz danych. W pierwszym z nich użytkownik loguje się do systemu serwera bazy danych, korzystając z pośrednictwa SSH (aby zapewnić bezpieczeństwo transmisji pomiędzy serwerem bazy danych a węzłem lokalnym). Następnie, już „lokalnie”, czyli na węźle, na którym działa serwer bazy danych uruchamia program klienta. Taki schemat ilustrowany jest rysunkiem 17.4.

Rysunek 17.4.

Użytkownik loguje się do systemu serwera bazy danych, uruchomionego na węźle Simba, samemu zasiadając przed monitorem komputera Cheetah; program klienta bazy danych uruchamiany jest na węźle Simba

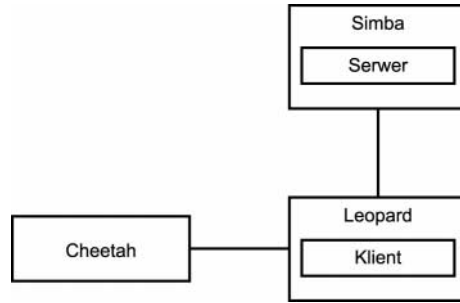


Metoda druga, ilustrowana rysunkiem 17.5, zakłada, że użytkownik loguje się (za pośrednictwem bezpiecznej powłoki SSH) na węźle zdalnym i na nim uruchamia program klienta bazy danych. Serwer jednak w tym scenariuszu działa na jeszcze innym węźle. W takiej konfiguracji w przetwarzanie zaangażowane są trzy węzły: węzeł, przed którym zasiada użytkownik, węzeł programu klienta bazy danych oraz węzeł serwera bazy danych.

Najważniejszym elementem schematu z rysunku 17.5 jest pośredniczący w komunikacji pomiędzy użytkownikiem a serwerem węzeł Leopard. W tym układzie program klienta bazy danych nie jest już uruchamiany na węźle serwera; nie jest też jednak uruchamiany lokalnie, w systemie użytkownika.

Rysunek 17.5.

Użytkownik loguje się z węzła Cheetah na węzeł Leopard i uruchamia na nim program klienta bazy danych; dopiero ten nawiązuje połączenie z węzłem serwera bazy danych (Simba)

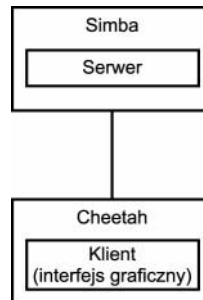


Dostęp do serwera bazy danych za pośrednictwem programu klienta wyposażonego w interfejs graficzny

Użytkownik może zalogować się do serwera bazy danych za pośrednictwem klienta działającego w trybie graficznym; klient ten może działać w systemie Windows, Macintosh czy na stacji roboczej UNIX. Program klienta nawiązuje połączenie z serwerem bazy danych. W takim przypadku program klienta działa lokalnie na węźle, przed którym zasiada użytkownik (patrz rysunek 17.6).

Rysunek 17.6.

Użytkownik uruchamia program klienta wyposażony w interfejs graficzny na węźle Cheetah; program ten nawiązuje połączenie z serwerem bazy danych, uruchomionym na węźle Simba



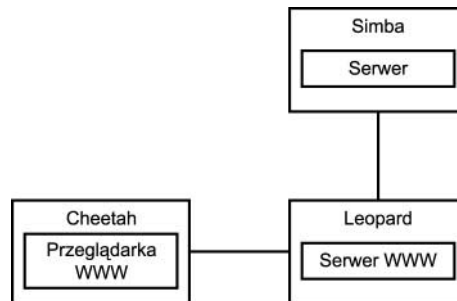
Dostęp do serwera bazy danych za pośrednictwem interfejsu WWW

W tym punkcie przyjrzymy się dwóm prostym przykładom odwoływania się do serwera bazy danych za pośrednictwem odpowiednio spreparowanej strony WWW. W pierwszym przykładzie użytkownik korzysta z zawartości bazy danych, posługując się zbiorem formularzy publikowanych w sieci WWW. Na pierwszy rzut oka zdaje się, że w układzie tym program klienta uruchomiony jest na węźle użytkownika. Nie jest to jednak zgodne ze stanem faktycznym, gdyż program klienta obsługiwany jest przez serwer WWW. Przeglądarka, którą użytkownik uruchomił lokalnie, prezentuje jedynie wynik działania programu klienta i pozwala na przekazywanie do niego danych. Właściwe oprogramowanie klienta składa się tu z serwera WWW (obsługującego żądania nadsyłane przez przeglądarkę) oraz skryptu CGI, serwletu w języku Java czy osadzanego w dokumencie HTML skryptu języka PHP czy JSP.

W kontekście baz danych pojęcia interfejsu i klienta stosuje się często zamiennie. Jednak, co ujawnia rysunek 17.7, klient i interfejs to w omawianym przypadku dwa różne elementy. Interfejsem jest tutaj formularz bądź formularze, prezentowane użytkownikowi przez przeglądarkę WWW. Klient nosi zaś w tym układzie nazwę **warstwy pośredniczącej**.

Rysunek 17.7.

Użytkownik korzysta z bazy danych za pośrednictwem sieci WWW; interfejsem jest tu przeglądarka WWW użytkownika a klientem oprogramowanie działające na węźle Leopard; na węźle Simba działa serwer bazy danych



Podział funkcjonalny może jednak w przypadku dostępu za pośrednictwem interfejsu WWW wyglądać nieco inaczej. Klient może być bowiem aplikacją dwuczęściową, przy czym jedna z tych części działa na komputerze użytkownika, druga zaś jest obsługiwana przez serwer WWW. Dla przykładu, programista aplikacji baz danych może uzupełnić formularz prezentowany w przeglądarce o kod sprawdzający poprawność wprowadzonych do formularza danych. W takiej sytuacji zapytanie jest częściowo przetwarzane w systemie użytkownika, a częściowo tylko w serwerze WWW (często do węzła użytkownika oddelegowywane są operacje kontroli błędów; pozwala to na zmniejszenie obciążenia serwera i natężenia ruchu sieciowego — zapytanie jest sprawdzane pod kątem potencjalnych błędów jeszcze przed przesłaniem go do serwera bazy danych).

Program klienta bazy danych MySQL

Plik wykonywalny programu wierszowego klienta bazy danych MySQL nosi nazwę `mysql`. Program wywołuje się następująco:

```
mysql [opcje] [nazwa bazy danych]
```

Niektóre z opcji rozpoznawanych przez program `mysql` wymienione zostały w tabeli 17.1. Nazwa bazy danych to (podobnie, jak `opcje`) argument opcjonalny — w przypadku jego braku program zostanie uruchomiony bez nawiązywania połączenia z bazą danych.

Tabela 17.1. Wybrane opcje wywołania programu `mysql`

Opcja	Działanie
<code>-h nazwa_węzła</code>	Połączenie z serwerem uruchomionym na węźle zdalnym <code>nazwa_węzła</code>
<code>-u nazwa_użytkownika</code>	Połączenie w imieniu użytkownika <code>nazwa_użytkownika</code>
<code>-p</code>	Wymuszenie zapytania o hasło. Opcja wymagana wtedy, kiedy konto użytkownika, w imieniu którego nawiązywane jest połączenie, zabezpieczone jest hasłem
<code>-P numer_portu</code>	Numer portu, z którym program powinien nawiązać połączenie
<code>-?</code>	Wyświetlenie komunikatu pomocy

Program rozpoznaje znacznie więcej opcji, ale te zaprezentowane w tabeli 17.1 są wykorzystywane najczęściej. Pełny wykaz dostępnych opcji wraz z ich znaczeniem można obejrzeć na stronie podręcznika systemowego `man` pod hasłem `mysql`.



Hasło użytkownika, w imieniu którego nawiązywane jest połączenie, można określić w wierszu wywołania programu `mysql`, umieszczając je za przełącznikiem `-p`. Nie trzeba wtedy czekać na prośbę o podanie hasła. Jednak taki sposób wywoływania programu klienta `mysql` jest bardzo niebezpieczny. Hasło będzie bowiem włączone do ciągu wywołania polecenia, co oznacza, że można je będzie odczytać z listy procesów systemu. To poważne zagrożenie dla bezpieczeństwa danych; warto więc przyjąć zasadę, aby nigdy nie przekazywać hasła użytkownika bazy danych w wierszu wywołania programu `mysql`.

Program można uruchomić bez wskazania docelowej bazy danych. W takim przypadku warto po uruchomieniu programu skorzystać z polecenia `help` — program wyświetli obsługiwane polecenia. Wśród nich jest polecenie nawiązania połączenia z bazą danych:

```
mysql> help

MySQL commands:
Note that all text commands must be first on line and end with ';'
help (\h)   Display this help.
? (\?)     Synonym for `help'.
clear (\c)  Clear command.
connect (\r) Reconnect to the server. Optional arguments are db and host.
edit (\e)   Edit command with $EDITOR.
ego (\G)    Send command to mysql server, display result vertically.
exit (\q)   Exit mysql. Same as quit.
go (\g)     Send command to mysql server.
nopager (\n) Disable pager, print to stdout.
notee (\t)  Don't write into outfile.
pager (\P)  Set PAGER [to_pager]. Print the query results via PAGER.
print (\p)  Print current command.
quit (\q)   Quit mysql.
rehash (\#) Rebuild completion hash.
source (\.) Execute a SQL script file. Takes a file name as an argument.
status (\s) Get status information from the server.
tee (\T)    Set outfile [to_outfile]. Append everything into given outfile.
use (\u)    Use another database. Takes database name as argument.
```

Ostatnie z prezentowanych poleceń, `use`, pozwala na określenie bazy danych, z którą program ma się połączyć (np. `zwierzeta`). Trzeba pamiętać, że połączenie uda się tylko wtedy, kiedy bieżący użytkownik programu `mysql` ma uprawnienia dostępu do wskazanej bazy danych:

```
mysql> use zwierzeta
Database changed
mysql>
```

Program klienta bazy danych PostgreSQL

Program klienta bazy danych PostgreSQL, oferujący wierszowy interfejs poleceń, nosi nazwę `psql`. Podobnie, jak to ma miejsce w przypadku programu `mysql`, program `psql` może zostać uruchomiony z argumentem wskazującym docelową bazę danych. I również `psql` rozpoznaje szereg opcji, z których najczęściej wykorzystywane zostały umieszczone w tabeli 17.2.

Tabela 17.2. Wybrane opcje wywołania programu `psql`

Opcja	Działanie
<code>-h nazwa_węzła</code>	Połączenie z serwerem uruchomionym na węźle zdalnym <i>nazwa_węzła</i>
<code>-p numer_portu</code>	Numer portu, z którym program powinien nawiązać połączenie
<code>-U nazwa_użytkownika</code>	Połączenie w imieniu użytkownika <i>nazwa_użytkownika</i>
<code>-W</code>	Wymuszenie zapytania o hasło. W wersjach bazy danych PostgreSQL nowszych niż wersja 7 zapytanie o hasło jest generowane automatycznie, na żądanie serwera
<code>-?</code>	Wyświetlenie komunikatu pomocy

Pełny wykaz dostępnych opcji wraz z ich znaczeniem można obejrzeć na stronie podrecznika systemowego `man` pod hasłem `psql`.

Interfejsy graficzne

Czytelnicy zainteresowani dostępem do serwera baz danych za pośrednictwem wygodniejszego, bo działającego w trybie graficznym, programu klienta, mogą odetchnąć z ulgą — dostęp taki jest możliwy.

Otóż dla bazy danych MySQL dostępny jest oficjalny klient działający w trybie graficznym: MySQLGUI. MySQLGUI dostępny jest w postaci kodu źródłowego i w wersjach skompilowanych w witrynie MySQL, publikowanej pod adresem <http://www.mysql.com/>.

Dla baz danych MySQL i PostgreSQL dostępne są też interfejsy implementowane za pośrednictwem stron WWW. Wśród takich interfejsów można wymienić phpMyAdmin oraz phpPgAdmin. Oba te produkty korzystają z osadzanego w formularzach HTML języka skryptowego PHP, wymagają więc zainstalowania dla serwera WWW modułu obsługi języka PHP. Nie obejdzie się też, rzecz jasna, bez zainstalowania samego serwera WWW.

Przydatne polecenia systemu Linux

Przy tworzeniu i manipulowaniu bazami danych w dystrybucji Fedora pomocne mogą się okazać następujące polecenia:

- ♦ `createdb` — polecenie tworzące bazę danych PostgreSQL;
- ♦ `createuser` — polecenie tworzące konto użytkownika bazy danych PostgreSQL;
- ♦ `dropdb` — polecenie usuwające bazę danych PostgreSQL;
- ♦ `dropuser` — polecenie usuwające konto użytkownika bazy danych PostgreSQL;
- ♦ `mysql` — interaktywny program klienta bazy danych MySQL;
- ♦ `mysqladmin` — polecenie administrujące zasobami bazy danych MySQL;
- ♦ `mysqldump` — program wykonujący zrzut zawartości bazy danych bądź tabeli MySQL;
- ♦ `pg_ctl` — narzędzie służące do uruchamiania, zatrzymywania i przeładowywania serwera PostgreSQL;
- ♦ `psql` — interaktywny program klienta bazy danych PostgreSQL.

Warto zajrzeć

Oto lista witryn, które warto odwiedzić w poszukiwaniu informacji o bazach danych opisywanych w niniejszym rozdziale:

- ◆ <http://www.mysql.com/> — oficjalna witryna WWW oprogramowania serwera baz danych MySQL. Można tu znaleźć najnowsze wersje oprogramowania, jak również aktualne wersje dokumentacji bazy danych MySQL. Tutaj można też zamówić usługę wsparcia technicznego, przydatnego użytkownikom wykorzystującym bazę danych MySQL w środowisku korporacyjnym (wielu użytkowników korporacyjnych nie spojrzy nawet na oprogramowanie, dla którego nie można uzyskać pomocy technicznej).
- ◆ <http://www.postgresql.org/> — oficjalna witryna projektu serwera baz danych PostgreSQL. Tutaj publikowane są powiadomienia o najnowszych wydarzeniach związanych z projektem, tu należy szukać najnowszych wersji oprogramowania i dokumentacji.
- ◆ <http://www.postgresql.org/docs/7.3/interactive/tutorial.html> — wprowadzenie do korzystania z serwera PostgreSQL; znakomita, dostępna on-line dokumentacja dla wszystkich chcących rozpocząć pracę z bazami danych PostgreSQL.
- ◆ <http://www.pgsql.com/> — witryna firmy, oferującej płatne wsparcie techniczne dla użytkowników baz danych PostgreSQL.