1

Osnove RDBMS-a: Šta sačinjava SQL Server bazu podataka

ŠTA ĆETE NAUČITI U OVOM POGLAVLJU:

- Shvatićete koji to objekti sačinjavaju SQL Server bazu podataka
- Naučićete koje su to vrste podataka dostupne za upotrebu u SQL Serveru 2012
- Otkrićete kako se imenuju objekti

Šta to sačinjava jednu bazu podataka? Podaci, sasvim sigurno. (Od kakve koristi je baza podataka u kojoj se ne smeštaju nikakvi podaci?) Međutim, *sistem za upravljanje relacionim bazama podataka* (*Relational Database Management System – RDBMS*) zapravo je mnogo više od podataka. Sadašnji RDBMS-ovi ne samo što skladište vaše podatke, oni takođe upravljaju tim podacima umesto vas, ograničavajući vrstu podataka koji mogu da se nađu u sistemu i olakšavaju izlazak podataka iz tog sistema. Ukoliko vi samo želite da na nekom sigurnom mestu čuvate podatke, mogli biste da koristite bilo koji sistem za skladištenje podataka. RDBMS-ovi vam dozvoljavaju da odete dalje od skladištenja podataka u carstvo definisanja načina kako bi podaci trebalo da izgledaju ili *poslovnih pravila* podataka.

Nemojte pomešati ono što ja zovem „poslovna pravila podataka“ sa opštijim pojmom poslovnih pravila koja pokreću ceo vaš sistem (na primer, sprečavanje neke osobe da vidi bilo šta pre nego što se prijavi na sistem, ili automatsko prilagođavanje tekućeg obračunskog perioda na prvi dan u mesecu). Te vrste pravila mogu se nametnuti na praktično bilo kom nivou sistema (u današnje vreme, obično u srednjem ili klijentskom sloju višeslojnog sistema). Umesto toga, ja ovde mislim na poslovna pravila koji su usko vezana za podatke. Na primer, ne možete imati porudžbenicu sa negativnom količinom. Pomoću RDBMS-a možete da uključite ova pravila direktno u samu bazu podataka.

Shvatanje baze podataka kao baze koja je odgovorna za podatke unutar nje, kao i najboljih metoda za unošenje i izvlačenje podataka iz baze podataka služi kao polazna tačka za ovu knjigu. Ovo poglavlje je pregled onoga što se nalazi u ostatku knjige. Većina stvari o kojima se govori u ovom poglavlju obuhvaćena je u kasnijim poglavljima, ali je svrha ovog

poglavlja da vam predstavi mapu ili plan koji treba da imate na umu kako budete dalje napredovali kroz knjigu. Imajući to na umu, pružiću vam prvoklasne informacije o:

- objektima baza podataka
- vrstama podataka
- drugim konceptima baza podataka koji obezbeđuju integritet podataka.

OPŠTI PREGLED OBJEKATA BAZE PODATAKA

Primerak RDBMS-a, kao što je SQL Server sadrži mnogo *objekata* (*objects*). Objekatski čistunci mogu okolišiti sa time da li Microsoft-ov izbor onoga šta treba (ili ne treba) nazvati objektom zapravo zadovoljava normalnu definiciju objekta, ali u svrhe SQL Servera, može se reći da bi lista nekih važnijih objekata baza podataka mogla sadržati sledeće:

- samu bazu podataka
- dnevnik transakcija
- tabele
- indekse
- grupe datoteka
- dijagrame
- prikaze
- snimljene procedure
- funkcije koje definiše korisnik
- sekvence
- korisnike
- uloge
- programske sklopove
- izveštaje
- tekstualne kataloge
- vrste podataka koje definiše korisnik

Objekat baze podataka

Baza podataka u stvarnosti predstavlja objekat najvišeg nivoa na koji se možete uputiti unutar datog SQL Servera. (Tehnički rečeno, sam server se može smatrati objektom, ali ne iz realne „programerske“ perspektive, pa neću dublje zalaziti u detalje.) Većina drugih objekata u SQL Serveru, ali ne svi, predstavlja potomke objekta baze podataka.

NAPOMENA *Ukoliko ste već upoznati sa SQL Serverom, možete pomisliti nešto poput: „Šta? Šta se to dogodilo prijavljivanjima ili zadacima SQL agenta?" SQL Server sadrži nekoliko drugih objekata (kao što je već navedeno) koji postoje kao podrška bazi podataka. Uz izuzetak vezanih servera, i možda paketa servisa za integraciju, oni su pre svega u domenu administratora baze podataka, pa im, kao takvim, nećete pridavati više pažnje tokom procesa dizajniranja i programiranja. (Oni se mogu programirati putem nečega što se naziva SQL objekat za upravljanje [SQL Management Objects – SMO], a što je van interesovanja ove knjige.) Iako ima nekih izuzetaka od ovog pravila, uopšteno govoreći mislim da su po svojoj prirodi ovi objekti napredniji, pa zbog toga nisu obuhvaćeni u verziji knjige za početnike.*

Baza podataka u opštem slučaju predstavlja grupu elemenata koji uključuju najmanje jedan skup tabelarnih objekata i, mnogo češće, druge objekte, kao što su snimljene procedure i prikazi koji pripadaju odgovarajućem grupisanju podataka smeštenih u tabelama baze podataka.

Koje vrste tabela smeštate u samo jednu bazu podataka, a koje idu u izdvojenu bazu? O tome ću kasnije detaljnije pričati, ali za sada ću se okrenuti jednostavnom pristupu i reći da će bilo koji podatak, za koji se generalno smatra da pripada samo jednom sistemu, ili se bitno odnosi na njega, biti smešten u jednu bazu podataka. RDBMS, kao što je SQL Server, može imati više baza podataka na samo jednom serveru ili može imati samo jednu. Broj baza podataka koje se nalaze na pojedinačnom SQL Serveru zavisi od faktora, kao što su kapacitet (snaga CPU-a, I/O ograničenja diska, memorija, i tako dalje), autonomija (želite da jedna osoba ima upravljačka prava na serveru na kome je pokrenut sistem, a da neko drugi ima administratorska prava na nekom drugom serveru), kao jednostavno i to koliko baza podataka vaša kompanija ili klijent poseduje. Neki serveri imaju samo jednu proizvodnu bazu podataka; drugi ih imaju više. Takođe, bilo koja verzija SQL Servera, koja se u današnje vreme proizvodi, ima više primeraka SQL Servera – potpunih sa odvojenim pravima za prijavljivanje i upravljanje – na istom fizičkom serveru. (SQL Server 2000 se već pet godina koristio pre nego što je zamenjen, pa pretpostavljam da se u prodavnicama može nabaviti ta ili kasnija verzija.)

NAPOMENA *Siguran sam da se mnogi od vas pitaju, „Mogu li imati različite verzije SQL Servera u istom sistemu – recimo, SQL Server 2008 i SQL Server 2012?" Odgovor je potvrdan. Možete mešati SQL Server 2008 i SQL Server 2012 u istom sistemu. Lično, uopšte nemam poverenja u takvu konfiguraciju, čak i za scenarije migracija, ali ako imate takve potrebe, to se može uraditi.*

Kada prvi put učitate SQL Server, počinjete sa najmanje četiri sistemske baze podataka:

- `master`
- `model`
- `msdb`
- `tempdb`

Potrebno je da se sve one instaliraju kako bi vaš server ispravno funkcionisao. (Zaista, bez nekih od njih, neće se uopšte ni pokrenuti.) Odatle, stvari se različito odvijaju zavisno od toga šta ste birali prilikom instaliranja. Primeri nekih baza podataka koje možete videti uključuju sledeće:

- **ReportServer:** Baza podataka koja opslužuje konfiguraciju servera za izveštavanje, kao i za potrebe skladištenja modela
- **ReportServerTempDB:** Radna baza podataka servera za izveštavanje
- **AdventureWorks:** Uzorak baze podataka
- **AdventureWorksDW:** Uzorak za korišćenje sa servisima za analiziranje

Osim ovih sistemski instaliranih primera, možete, kada pretražujete internet ili koristite druge priručnike, naći uputstvo za par starijih uzoraka:

- `pubs`
- `Northwind`

Pošto ovi primerci nisu korišćeni ni u prethodnom izdanju ove knjige, više se njima neću baviti, ali ih ipak pominjem najviše zbog toga što bude lepe uspomene iz jednostavnijih vremena, i delom zato što negde možete naleteti na njih.

NAPOMENA *Za ovo izdanje, primeri će se praviti kod kuće ili potiču iz novijih AdventureWorks uzoraka. AdventureWorks baza podataka je sigurno robusni primer i sjajan je u davanju primera za skoro sve preokrete koje možete iskoristiti u SQL Serveru 2012. Međutim, tu postoji i jedan problem, a to je složenost. AdventureWorks baza podataka ponekad može biti izuzetno složena da bi predstavljala bazu podataka za obučavanje. Sadrži karakteristike koje se verovatno koriste samo u izuzetnim slučajevima, a koristi ih kao dominantne karakteristike. Prema tome, dozvolite mi da istaknem da se AdventureWorks ne bi trebalo nužno koristiti kao šablon za ono što će se raditi u drugim sličnim aplikacijama.*

Master baza podataka

Svaki SQL Server, bez obzira na verziju ili prilagođene izmene, ima `glavnu` bazu podataka (`master` database). Ova baza podataka sadrži poseban skup tabela (sistemskih tabela) koji prati rad sistema kao celine. Na primer, kada pravite novu bazu podataka na serveru, neka odrednica se smešta u `sysdatabases` tabelu u `master` bazi podataka. Sve proširene i sistemski snimljene procedure, bez obzira na to u kojoj su bazi podataka namenjene za upotrebu, smeštaju se u ovu bazu podataka. Očigledno je da je, pošto se skoro sve što opisuje vaš server smešta ovde, ova baza podataka ključna za vaš sistem i ona se ne može obrisati.

Sistemske tabele, uključujući one koje se nalaze u `master` bazi podataka, u prošlosti su povremeno bile korišćene u maloj količini kako bi pružile informacije o konfiguraciji sistema, kao što su one o tome da li su određeni objekti postojali pre nego što ste obavili neke operacije na njima. Microsoft je godinama upozoravao programere da ne koriste sistemske tabele direktno, ali je, pošto je postojalo malo drugih opcija, većina programera ignorisala ovaj savet. Na sreću, Microsoft je počeo da nudi druge opcije u obliku sistemskih i informacionih šematskih prikaza; sada možete iskoristiti te prikaze kako biste došli do metapodataka siste-

ma, a prema svojim potrebama, i to uz pun blagoslov Microsofta. Na primer, ukoliko želite da napravite objekat koji već postoji u nekoj određenoj bazi podataka, pojavljuje se greška. Ukoliko želite da prinudno izvršite ovo pitanje, mogli biste da obavite testiranje kako biste videli da li tabela već sadrži odrednicu u `sys.objects` tabeli za tu bazu podataka. Ukoliko je sadrži, taj objekat ćete obrisati pre nego što ga ponovo budete pravili.

UPOZORENJE *Ako ste prilično slobodni, možda ćete sebi reći, „Super, jedva čekam da krenem da se bavim tim sistemskim tabelama!" Ne usuđujte se! Korišćenje sistemskih tabela u bilo kom obliku predstavlja pravu opasnost. Microsoft uopšte ne garantuje za kompatibilnost* `master` *baza podataka u različitim verzijama. Zaista, oni praktično garantuju da će se one promeniti. Na sreću, postoji nekoliko alternativa (kao što su sistemske funkcije, sistemski snimljene procedure i* `information_schema` *prikazi) za preuzimanje velikog dela metapodataka koji su smešteni u sistemskim tabelama.*

Model baza podataka

Baza podataka `model` ima prikladan naziv u tom smislu da ona predstavlja model na kome se može zasnovati neka kopija. Baza podataka `model` oblikuje šablon za svaku novu bazu podataka koju pravite. To znači da možete, ukoliko to i želite, da izmenite bazu podataka `model` ukoliko želite da promenite standarde koji određuju izgled novonapravljenih baza. Na primer, možete da dodate skup kontrolnih tabela koje uključujete u svaku bazu podataka koju pravite. Takođe, možete uključiti i nekoliko korisničkih grupa koje bi bile kopirane u svaku novu bazu koja je napravljena u sistemu. Obratite pažnju na to da je, pošto ova baza podataka služi kao šablon za bilo koju drugu bazu podataka, to neophodna baza podataka i mora da ostane na sistemu; ne možete je obrisati.

Postoji nekoliko stvari koje treba imati na umu pri menjanju `model` baze podataka:

- **Bilo koja baza podataka koju pravite mora biti najmanje onoliko velika kao `model` baza.** To znači da ukoliko promenite da `model` baza bude veličine 100 MB, ne možete da napravite bazu podataka koja je manja od toga.
- **Slične cake postoje i pri dodavanju objekata ili menjanju podešavanja, što može dovesti do neželjenih posledica.** Kao takve, u 90 odsto instalacija, najtoplije preporučujem da ih ostavite po strani.

Baza podataka msdb

`msdb` je mesto na kojem proces SQL agenta smešta sve sistemske poslove. Ukoliko isplanirate da se proces pravljenja rezervnih kopija u bazi podataka obavlja noću, postoji odrednica u `msdb`-u. Isplanirajte da se snimljena procedura izvrši jedanput, i da i ona ima odrednicu u `msdb`-u. Drugi glavni podsistemi u SQL Serveru na sličan način iskorišćavaju `msdb`. SSIS paketi i definicije upravljanja zasnovanog na pravilima primeri su drugih procesa koji takođe imaju koristi od `msdb`-a.

Baza podataka tempdb

`Tempdb` je jedno od ključnih radnih okruženja za vaš server. Kad god izdate složeni ili veliki upit da SQL Server treba da napravi privremene tabele za njegovo rešavanje, on to čini

u `tempdb`-u. Kad god sami pravite privremenu tabelu, ona se pravi u `tempdb`-u, iako vi mislite da je pravite u aktuelnoj bazi podataka. (Pravi se jedna alternativna u lokalnoj bazi podataka na koju možete da se uputite, ali se prava tabela zapravo pravi u `tempdb`-u.) Kad god postoji potreba za privremenim skladištenjem podataka, oni se verovatno smeštaju u `tempdb`.

`tempdb` se dosta razlikuje od bilo koje druge baze podataka. Ne samo što su objekti koji se u njoj nalaze privremeni, već je i sama baza privremena. Ima tu karakteristiku da je jedina baza podataka u vašem sistemu koja se iznova pravi svaki put kada pokrenete svoj SQL Server.

NAPOMENA *Tehnički rečeno, vi zapravo sami možete napraviti objekte u* `tempdb`*-u. Toplo preporučujem da to ne radite. Privremene objekte možete napraviti u bilo kojoj bazi podataka kojoj možete pristupiti u svom sistemu – oni će biti smešteni u* `tempdb`*-u. Pravljenjem objekata direktno u* `tempdb`*-u ne postižete ništa, već samo pravite zbrku za upućivanje na stvari širom raznih baza podataka. Ovo je još jedna od onih „Ne zalazite tamo!" vrsta stvari.*

ReportServer

Ova baza podataka će postojati samo ukoliko instalirate ReportServer. (To nužno ne mora da bude isti server kao što je mašina baze podataka, ali imajte na umu da ako je to neki drugi server, on zahteva posebnu licencu.) ReportServer baza podataka smešta sve trajne metapodatke za vaš primerak servera za izveštavanje. Obratite pažnju na to da je ovo isključivo operaciona baza podataka za dati primerak servera za izveštavanje i da ne bi trebalo da se modifikuje (i da joj se retko pristupa) niotkuda osim kroz server za izveštavanje.

ReportServerTempDB

Ova baza obavlja istu osnovnu funkciju kao ReportServer baza podataka, osim što ona skladišti povremene podatke (kao što su radni podaci za izveštaj koji se izvršava). Ovo je takođe isključivo operaciona baza podataka, pa ne bi trebalo da joj pristupate ili je menjate na bilo koji način osim kroz server za izveštavanje.

AdventureWorks

SQL Server je obuhvatao uzorke baza podataka mnogo pre nego što se ova baza pojavila; premda su stari uzorci imali svoje nedostatke. Na primer, sadržali su nekoliko loših postupaka za dizajn. Osim toga, bili su pojednostavljeni i fokusirani na demonstriranje određenih koncepata baze podataka, a ne na SQL Server kao proizvod ili čak bazu podataka kao celinu. Isključiću argument da li AdventureWorks ima iste probleme. Recimo samo da je AdventureWorks bio, pored ostalih stvari, pokušaj rešavanja tog problema.

Od početnih faza razvoja SQL Servera 2005, Microsoft je znao da je želeo daleko robusniji uzorak baze podataka koji bi služio kao uzorak za što je moguće veći deo proizvoda. AdventureWorks je rezultat tog napora. Koliko god slušali od mene kako se žalim na njenu složenu prirodu za početnog korisnika, ona je remek delo u tome što odmah pokazuje *sve*. U redu, ne baš zaista *sve*, ali to je skoro kompletan uzorak, sa realnijim kompletom podata-

ka, složenim strukturama i odeljcima koji pokazuju uzorke za veliku većinu karakteristika proizvoda. U tom smislu, zaista je sjajan.

AdventureWorks će biti nešto poput vaše domaće baze podataka – dosta ćete je koristiti dok budete prorađivali primere iz ove knjige.

AdventureWorksDW

Ovo je uzorak servisa za analiziranje (Analysis Services). DW označava skladište podataka (Data Warehouse), što je tip baze podataka preko koga se izrađuje većina projekata servisa za analiziranje. Možda je najveća stvar u vezi sa ovim uzorkom to što je Microsoft imao predosećaj, pa spojio uzorak baze podataka za transakciju i uzorak za analizu, pružajući celokupan skup uzoraka koji pokazuju kako ova dva uzorka zajedno funkcionišu.

Baze podataka za podršku pri odlučivanju su detaljnije obrađene u poglavljima 17 i 18, i ovu bazu podataka ćete da koristite, pa imajte to na umu kada pokrenete servise za analiziranje i počnete da ih koristite. Pogledajte sada koje su to razlike između ove dve baze podataka. Namenjene su da se koriste u istoj fiktivnoj kompaniji, ali imaju različitu svrhu: učite iz toga.

Baza podataka pubs

Ahhh, `pubs`! Skoro kao da mi je stari prijatelj. `pubs` je jedan od prvih primeraka baza podataka i bio je isporučen uz SQL Server kao deo instalacije pre verzije SQL Server 2005. Danas se može samo zasebno preuzeti sa Microsoftove veb lokacije. I dalje možete naći dosta članaka za obuku i knjiga koje se odnose na `pubs`, ali Microsoft nije obećao koliko dugo će nastaviti da ga čini dostupnim za preuzimanje. `pubs` apsolutno nema ništa sa funkcionisanjem SQL Servera. On je jednostavno tu kao dosledno mesto za vašu obuku i eksperimentisanje. On vam nije potreban kako biste radili vežbanja iz ove knjige, ali ćete možda hteti da ga preuzmete i instalirate kako biste se obučili i proradili druga vežbanja koja možete naći na internetu.

Baza podataka Northwind

Ukoliko je vaše dosadašnje programersko iskustvo podrazumevalo korišćenje Accessa i Visual Basica, trebalo bi da ste već donekle upoznati sa bazom podataka `Northwind`. `Northwind` je dodat SQL Serveru počevši od verzije 7.0, ali je uklonjen iz osnovne instalacije od verzije SQL Server 2005. Baš kao i `pubs`, za sada se može preuzeti nezavisno od osnovne SQL Server instalacije. (Na sreću, on je deo istog uzorka i instalacije za preuzimanje kao što je i `pubs`.) Poput baze `pubs`, ni baza `Northwind` vam nije potrebna kako biste radili vežbanja iz ove knjige, ali je korisno da je imate kada želite da radite razna vežbanja i obuke koje ćete naći na internetu.

KORIŠĆENJE BAZA PODATAKA NORTHWIND I PUBS SA SQL SERVEROM 2012

Ovi uzorci baza podataka zastarevaju i podrška za njih se smanjuje. I dalje ih možete koristiti, ali morate proći kroz proces konverzije o kojoj ovde neću detaljisati. Ukoliko zaista želite da nađete i koristite ove baze podataka, možete to učiniti, ali će vam za to biti potrebno još malo dodatnog napora.

Dnevnik transakcija

Verovali ili ne, sama datoteka baze podataka nije ono mesto gde se većina stvari dešava. Iako se podaci odatle čitaju, sve promene koje obavljate ne idu prvo u samu bazu podataka. Umesto toga, serijski se zapisuju u *dnevniku transakcija* (*transaction log*). Kako vreme odmiče, u nekom trenutku bazi podataka se izdaje *kontrolna tačka* (*checkpoint*); baš u tom vremenskom trenutku sve promene u dnevniku se prenose na stvarnu datoteku baze podataka.

Baza podataka ima slobodan pristup, dok je dnevnik po svojoj prirodi serijski. Dok nasumična priroda datoteke baze podataka dozvoljava brz pristup, serijska priroda dnevnika dozvoljava stvarima da se prate po pravom redosledu. Dnevnik akumulira promene koje se smatraju obavljenim, a zatim po nekoliko njih u istom trenutku zapisuje u fizičku datoteku(e) baze podataka.

Mnogo detaljnije o tome kako se stvari evidentiraju čitaćete u poglavlju 14, ali za sada, zapamtite da je dnevnik prvo mesto na disku u koje se podaci smeštaju, a zatim se kasnije prenose na pravu bazu podataka. Da biste imali funkcionalnu bazu podataka, potrebni su vam i datoteka baze podataka i dnevnik transakcija.

Najjednostavniji objekat baze podataka: tabela

Baze podataka su sačinjene od mnogo stvari, ali samu osnovu za stvaranje jedne baze podataka najviše predstavlja tabela (tables). Tabela se može posmatrati kao nešto što je isto što i glavna knjiga nekog računovođe ili Excel proračunska tabela i sastoji se od podataka *domena* (*domain* data) (kolone) i podataka *entiteta* (*entity* data) (redovi). Pravi podaci za bazu podataka smeštaju se u tabele.

Svaka definicija tabele takođe sadrži *metapodatke* (*metadata*) (deskriptivni opis o podacima) koji opisuju prirodu podataka koje će ona sadržati. Svaka kolona ima svoj skup pravila o tome šta se može smestiti u određenu kolonu. Kršenje pravila bilo koje kolone može dovesti do toga da sistem odbije umetnuti red, ispravku postojećeg reda, ili brisanje reda.

	LocationID	Name	CostRate	Availability	ModifiedDate
1	1	Tool Crib	0.00	0.00	1998-06-01 00:00:00.000
2	2	Sheet Metal Racks	0.00	0.00	1998-06-01 00:00:00.000
3	3	Paint Shop	0.00	0.00	1998-06-01 00:00:00.000
4	4	Paint Storage	0.00	0.00	1998-06-01 00:00:00.000
5	5	Metal Storage	0.00	0.00	1998-06-01 00:00:00.000
6	6	Miscellaneous Storage	0.00	0.00	1998-06-01 00:00:00.000
7	7	Finished Goods Storage	0.00	0.00	1998-06-01 00:00:00.000
8	10	Frame Forming	22.50	96.00	1998-06-01 00:00:00.000
9	20	Frame Welding	25.00	108.00	1998-06-01 00:00:00.000
10	30	Debur and Polish	14.50	120.00	1998-06-01 00:00:00.000
11	40	Paint	15.75	120.00	1998-06-01 00:00:00.000
12	45	Specialized Paint	18.00	80.00	1998-06-01 00:00:00.000
13	50	Subassembly	12.25	120.00	1998-06-01 00:00:00.000
14	60	Final Assembly	12.25	120.00	1998-06-01 00:00:00.000

SLIKA 1-1

Pogledajte tabelu Production.Location u bazi podataka AdventureWorks. (Prikaz dat na slici 1-1 uzet je iz SQL Server studija za upravljanje. Ovo je osnovna alatka i u sledećem poglavlju ćete videti kako da je iskoristite na najbolji način.)

Tabela prikazana na slici 1-1 sastoji se od pet kolona podataka. Broj kolona ostaje konstantan bez obzira na to koliko se podataka (čak i u slučaju da nema ni jednog) nalazi u tabeli. Tabela trenutno ima 14 zapisa. Broj zapisa će se povećavati i smanjivati kako budete dodavali ili brisali podatke, ali je priroda podataka u svakom zapisu (ili redu) opisana i ograničena *vrstom podataka* (*data type*) za tu kolonu.

Indeksi

Indeks (*index*) je objekat koji postoji samo unutar okvira određene tabele ili prikaza. Indeks funkcioniše kao što funkcioniše indeks koji se nalazi na kraju neke enciklopedije. Postoji neka vrsta referentne vrednosti (ili „ključa“) koja je poređana na određen način i

kada je dobijete, daje vam se još jedan ključ pomoću koga možete da nađete informaciju koju ste i tražili.

Indeksi vam pružaju načine za ubrzanje procesa traženja željenih informacija, a svrstavaju se u dve kategorije:

- **Grupisani** (**Clustered**): Možete imati samo jedan ovakav indeks po tabeli. Ukoliko je indeks *grupisan*, to znači da je tabela na kojoj se zasniva grupisani indeks fizički sortirana prema tom indeksu. Ukoliko indeksirate enciklopediju, grupisani indeks bi bili brojevi stranica (informacije u enciklopediji smeštene su redom po brojevima stranica).
- **Negrupisani** (**non-clustered**): Možete imati više ovakvih indeksa za svaku tabelu. Ovo više spada u domen onoga na šta pomislite kada čujete reč „indeks“. Ova vrsta indeksa ukazuje na neku drugu vrednost koja će vam omogućiti nalaženje podataka. Kada je reč o enciklopediji, to bi bio indeks ključnih reči koji se nalazi na kraju knjige.

Obratite pažnju na to da prikazi koji imaju indekse – ili *indeksni prikazi* (*indexed views*) – moraju da imaju najmanje jedan grupisani indeks pre nego što će moći da imaju i jedan negrupisani indeks.

Okidači

Okidač (*trigger*) je objekat koji postoji samo unutar okvira tabele. Okidači su delovi logičkog koda koji se automatski izvršavaju kada se određene stvari (kao što su umetanja, ispravke ili brisanja) dogode vašoj tabeli.

Okidači se mogu koristiti u raznorazne svrhe, ali se pre svega koriste ili za kopiranje podataka dok se oni unose ili za proveru ažuriranja kako bi se utvrdilo da ono zadovoljava određene kriterijume.

Ograničenja

Ograničenje (*constraint*) je još jedan objekat koji postoji samo unutar granica tabele. Ograničenja su baš ono što i sama reč kaže; ona suzbijaju podatke u vašoj tabeli kako bi ispunili određene uslove. Ograničenja se na neki način takmiče sa okidačima u smislu mogućih rešenja za probleme integriteta podataka. Oni, međutim, ne predstavljaju iste stvari: Svaki ima svoje jasne prednosti.

Grupe datoteka

Podrazumeva se da su sve vaše tabele i sve ostalo što je u vezi sa vašom bazom podataka (osim dnevnika) smeštene u jednoj datoteci. Podrazumeva se da je ta datoteka član onoga što se zove *grupa primarnih datoteka* (*primary filegroup*). Međutim, niste ograničeni na ovo grupisanje.

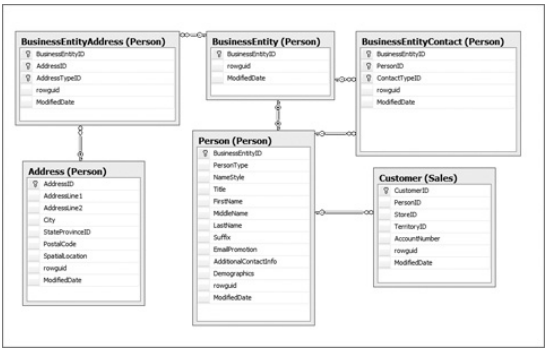
SQL Server vam dozvoljava da definišete nešto malo više od 32 000 *sekundarnih datoteka* (*secondary files*). (Ukoliko vam je potrebno više od toga, možda nije SQL Server taj koji ima problem.) Ove sekundarne datoteke se mogu dodati grupi primarnih datoteka ili praviti kao deo jedne ili više *korisnički definisanih grupa datoteka* (*user-defined filegroups*). Iako postoji samo jedna grupa primarnih datoteka (i ona se zapravo zove „Primarna“), možete imati do 255 grupa datoteka koje definiše korisnik. Korisnički definisana grupa datoteka pravi se kao opcija komande `CREATE DATABASE` ili `ALTER DATABASE`.

Dijagrami

O izradi dijagrama baze podataka detaljno ću diskutovati kada budem govorio o normalizovanju i dizajnu baze podataka. Za sada, dovoljno je reći to da dijagram baze podataka predstavlja vizuelni prikaz dizajna baze podataka, uključujući razne tabele, nazive kolona u svakoj tabeli, kao i odnose između tabela. Na svom dosadašnjem programerskom putovanju možda ste čuli za *dijagram odnosa entiteta* (*entity-relationship diagram, ERD*). U ERD-u baza podataka je podeljena na dva dela: entitete (kao što su „snabdevač" i „proizvod") i odnose (kao što su „nabavke" i „kupovine").

NAPOMENA *Alatke za dizajn koje su sadržane u ovoj bazi podataka su, nažalost, oskudne. Zaista, metodologija izrade dijagrama koju alatke koriste nije u skladu ni sa jednim prihvaćenim standardom u ER programiranju. Ipak, ove alatke za izradu dijagrama zaista pružaju sve „neophodne" stvari, pa su za početak dovoljne.*

Slika 1-2 predstavlja dijagram koji pokazuje neke od raznih tabela u bazi podataka AdventureWorks. Dijagram takođe opisuje mnoga druga svojstva o bazi podataka (iako se može činiti pomalo teškom za razumevanje pošto je to za vas novina). Primetićete majušne ikone ključeva i znaka za beskonačnost. One predstavljaju prirodu odnosa između dve tabele. O odnosima ću naširoko pričati u poglavljima 6 i 8, a dijagramima ću se detaljnije baviti nešto kasnije.



SLIKA 1-2

Prikazi

Prikaz (*view*) predstavlja nešto poput virtuelne tabele. Prikaz se najvećim delom koristi kao i tabela, samo što ne sadrži sopstvene podatke. Umesto toga, prikaz podrazumeva samo unapred isplanirano mapiranje i predstavljanje podataka smeštenih u tabelama. Plan se u bazu podataka smešta u obliku upita. Ovaj upit traži da se podaci nekih kolona, ali ne nužno svih, učitaju iz jedne ili više tabela. Učitani podaci možda (zavisno od definicije prikaza) neće morati da zadovolje posebne kriterijume kako bi se prikazali kao oni koji su dati u tom prikazu.

Sve do SQL Servera 2000 osnovna svrha prikaza bila je da kontrolišu ono što korisnik prikaza vidi. To je za posledicu imalo dve stvari: bezbednost i lakoću upotrebe. Pomoću prikaza, možete kontrolisati ono što korisnici vide, pa ako postoji odeljak tabele kome može pristupiti samo nekoliko korisnika (na primer, detalji o zaradi) možete napraviti prikaz koji će obuhvatati samo one kolone kojima mogu pristupiti svi. Pored toga, prikaz se može izraditi tako da korisnik ne mora da pretražuje po informacijama koje mu nisu potrebne.

Osim ovih najosnovnijih upotreba prikaza, imate i mogućnost pravljenja onoga što se naziva *indeksni prikaz* (*indexed view*). On je isti kao i svaki drugi prikaz, osim što je jedan

ili više indeksa napravljeno prema tom prikazu. To kao rezultat daje par uticaja na performanse (neke pozitivne i jedan negativan):

- Prikazi koji se odnose na više tabela po pravilu imaju *mnogo* bolje performanse za čitanje sa indeksnim prikazom jer je spoj između tabela već postavljen.
- Grupisanja obavljena u prikazu unapred su izračunata i smeštena kao deo indeksa; opet, to znači da je grupisanje obavljeno jedanput (kada je red umetnut ili ispravljen), a zatim se može čitati direktno iz indeksnih informacija.
- Umetanja i brisanja dodatno troše memoriju jer se indeks mora odmah ažurirati u prikazu; ažuriranja takođe više troše resurse ako je ažuriranje uticalo na glavnu kolonu ili grupni ključ indeksa.

O ovim pitanjima performansi detaljnije ćete učiti u poglavlju 10.

Snimljene procedure

Snimljene procedure (*stored procedures*) (ili *sproc-ovi*) su glavna potpora programerske funkcionalnosti u SQL Serveru. Snimljene procedure generalno predstavljaju uređene serije Transact-SQL naredbi (jezika koji se koristi kako bi se uputio upit Microsoft SQL Serveru) prikupljenih u jednu logičnu jedinicu. One uzimaju u obzir promenljive i parametre, kao i izbor i strukture pravljenja petlji. Sprocovi nude nekoliko prednosti u odnosu na puko slanje pojedinačnih naredbi serveru u tom smislu da:

- su upućeni na korišćenje kratkih imena, a ne dužeg tekstualnog niza, pa je stoga potrebno manje mrežnog saobraćaja kako bi se kôd pokrenuo unutar samog sproca,
- su prethodno optimizirani i prevedeni, štedeći nešto malo vremena svaki put kada se sproc pokrene,
- obuhvataju neki proces, obično iz bezbednosnih razloga ili kako bi prikrili složenost baze podataka,
- se mogu pozivati iz drugih sprocova, čineći ih na taj način podesnim za ponovno korišćenje u donekle ograničenom smislu.

Iako su sproc-ovi suština programerske funkcionalnosti u SQL Serveru, budite pažljivi kada ih koristite. Oni često predstavljaju rešenje, ali isto tako često nisu jedino rešenje. Dobro razmislite da li su oni pravi izbor pre nego što izaberete neki sproc kao opciju kojoj ćete se prikloniti.

Funkcije koje definiše korisnik

Korisnički definisane funkcije (*user-defined functions, UDF-ovi*) imaju izuzetno veliki broj sličnosti sa sprocovima, osim što one:

- **mogu vratiti vrednost za većinu SQL Server vrsta podataka.** Isključene vrste povratnih vrednosti obuhvataju tipove `text`, `ntext`, `image`, `cursor` i `timestamp`.
- **ne mogu imati nikakve negativne efekte.** U osnovi, one ne mogu uraditi ništa što seže van opsega funkcije, kao što je menjanje tabela, slanje elektronske pošte ili menjanje parametara sistema ili baze podataka.

UDF-ovi su slični funkcijama koje biste koristili u standardnom programskom jeziku, kao što je VB.NET ili C++. Možete proslediti više od jedne promenljive i kao rezultat dobiti

neku vrednost. UDF-ovi SQL Servera razlikuju se od funkcija nađenih u mnogim proceduralnim jezicima, po tome što su sve promenljive (osim promenljivih tabela koje se koriste kao parametri) prenete u funkciju po vrednosti. Ukoliko ste upoznati sa prenošenjem promenljivih po referenci (`By Ref`) ili prenošenjem pokazivača, izvinite, ali ovde ne postoji ekvivalent za to. Postoje, ipak, dobre vesti o tome da možete kao rezultat dati poseban tip podataka koji se naziva `table`. Uticaj toga ću ispitati u poglavlju 13.

Sekvence

Sekvence (*sequences*) su nova vrsta objekata predstavljenih u SQL Serveru 2012. Zadatak sekvence je da pruži izvor sekvencijalnih brojeva kojima može pristupiti bilo koji broj procesa, garantujući da ni jedan ni drugi neće dati istu sledeću vrednost u isto vreme. Pošto su to objekti koji samostalno postoje – nisu vezani ni za jednu tabelu – sekvence imaju raznovrsne upotrebe o kojima ćete detaljnije čitati u poglavlju 7.

Korisnici i uloge

Ovi objekti idu u paru. *Korisnici* (*users*) su praktično ekvivalenti prijavljivanjima. Ukratko, ovaj objekat predstavlja identifikator neke osobe kako bi se ona prijavila na SQL Server. Bilo ko ko se prijavljuje na SQL Server mora da se preslika (direktno ili indirektno, zavisno od modela bezbednosti koji se koristi) u korisnika. Korisnici, zauzvrat, pripadaju jednoj ulozi ili većem broju *uloga* (*roles*). Prava za obavljanje određenih akcija u SQL Serveru zatim se mogu dodeliti direktno korisniku ili ulozi kojoj jedan korisnik ili više njih pripada.

Pravila

Pravila i ograničenja pružaju informacije o ograničenju onoga što se može uneti u neku tabelu. Ukoliko neki ažurirani ili umetnuti zapis prekrši pravilo, to umetanje ili ažuriranje će biti odbijeno. Osim toga, pravilo se može koristiti kako bi se definisalo ograničenje na *tipu podataka koji definiše korisnik*. Za razliku od ograničenja, pravila nisu vezana za neku određenu tabelu. Umesto toga ona predstavljaju nezavisne objekte koji mogu biti vezani za više tabela ili čak za određene tipove podataka (koji se, sa svoje strane, koriste u tabelama).

Microsoft nije odobravao pravila u nekoliko izdanja ovog proizvoda, Trebalo bi da se uzmu u obzir samo kod kompatibilnosti sa prethodnim verzijama i trebalo bi da ih izbegavate u novom programiranju.

NAPOMENA *Imajući u vidu to da je Microsoft predstavio neke od novih funkcionalnosti za upravljanje molbama u SQL Serveru 2012, mislim da će karakteristike (poput pravila) koje nisu odobravane u nekoliko ranijih verzija, možda konačno biti uklonjene u sledećoj verziji SQL Servera. Imam potrebu da ponovo naglasim da pravila, kao takva, ne bi trebalo koristiti u novom programiranju. Zaista, odavno je vreme da se prestane sa njihovom upotrebom. Podrazumevane vrednosti*

Podrazumevane vrednosti

Postoje dve vrste podrazumevanih vrednosti. Postoji podrazumevana vrednost koja je sama po sebi objekat, i podrazumevana vrednost koja zapravo i nije objekat, već metapodatak koji opisuje određenu kolonu u nekoj tabeli (na praktično isti način na koji postoje pravila

koja su objekti, i ograničenja koja nisu objekti, već metapodaci). One imaju istu namenu. Ukoliko, kada unosite neki zapis, ne ponudite vrednost kolone, a ta kolona ima definisanu podrazumevanu vrednost, vrednost će automatski biti unesena jer je definisana kao podrazumevana. Obe vrste podrazumevanih vrednosti ispitaćete u poglavlju 6.

Tipovi podataka koje definiše korisnik

Korisnički definisani tipovi podataka su ili oznake tipa sistemski definisanih tipova podataka ili složeni tipovi podataka definisani nekom metodom u .NET programskom sklopu. Mogućnosti su ovde skoro beskonačne. Iako su SQL Server 2000 i raniji imali ideju o korisnički definisanim tipovima podataka, oni su zapravo bili samo ograničeni na različite filtrirajuće ili postojeće tipove podataka. Sa pojavom verzija nakon SQL Servera 2005, u mogućnosti ste da vežete .NET sklopove za vaše tipove podataka, što znači da možete imati tipove podataka koji skladište (sa razlogom) skoro sve što možete smestiti u neki .NET objekat. Zaista, tipovi podataka prostora (`Geographic` i `Geometric`) koji su dodati SQL Serveru 2008 implementirani su korišćenjem tipa koji definiše korisnik na osnovu .NET programskog sklopa. .NET programski sklopovi definitivno predstavljaju temu za napredne korisnike i izvan su oblasti interesovanja ove knjige.

NAPOMENA *Budite pažljivi sa ovim! Tip podataka sa kojim vi radite veoma je važan za vaše podatke i njihovo skladištenje. Iako je sposobnost da definišete sopstvene stvari veoma dobra osobina, imajte na umu to da će ona gotovo sigurno doći sa visokom cenom performansi. Razmislite o tome pažljivo, budite sigurni da vam to treba, a zatim, kao i sa svim ostalim, ISPROBAJTE, ISPROBAJTE, ISPROBAJTE!!!*

Tekstualni katalozi

Tekstualni katalozi predstavljaju mapiranja podataka koja ubrzavaju pretragu određenih blokova tekstova unutar kolona kod kojih je uključena mogućnost pretraživanja teksta. U verzijama pre SQL Servera 2008, tekstualni katalozi su bili smešteni izvan same baze podataka (stvarajući na taj način neke značajne probleme za pravljenje rezervnih kopija i oporavak). Od SQL Servera 2008 tekstualni katalozi integrisani su u glavnu mašinu baze podataka i u mehanizme za skladištenje. S obzirom na to da su po svojoj prirodi složeni, tekstualni indeksi su van oblasti interesovanja ovog teksta.

TIPOVI PODATAKA SQL SERVERA

Sada, kada ste se upoznali sa osnovnim objektima SQL Server baze podataka, pogledajte opcije koje SQL Server sadrži kada je u pitanju jedna od ključnih stavki svakog okruženja koje se bavi podacima – tipove podataka. Obratite pažnju na to da, pošto je ova knjiga namenjena programerima, nijedan programer ne bi opstao ni 60 sekundi ako ne razume tipove podataka, pa pretpostavljam da već znate kako oni funkcionišu, te je samo potrebno da saznate o pojedinostima tipova podataka SQL Servera.

U tabeli 1-1 prikazani su primarni tipovi podataka SQL Servera 2012:

TABELA 1-1: Tipovi podataka

NAZIV TIPA PODATAKA	KLASA	VELIČINA U BAJTIMA	PRIRODA PODATAKA
`BBit`	Ceo broj	1	Veličina može da vas zavara. Prvi podatak tipa `bit` u tabeli veličine je jednog bajta; sledećih sedam koristi taj isti bajt. Dozvoljavanje praznih znakova prouzrokuje korišćenje dodatnog bajta.
`Bigint`	Ceo broj	8	Bavi se činjenicom da sve češće koristimo sve veće i veće brojeve. Ovaj tip vam dozvoljava da koristite cele brojeve od -2^{63} do 2^{63}-1. To je više manje oko 92 kvintiliona.
`Int`	Ceo broj	4	Celi brojevi od -2 147 483 648 do 2 147 483 647.
`SmallInt`	Ceo broj	2	Celi brojevi od -32 768 do 32 767.
`TinyInt`	Ceo broj	1	Celi brojevi od 0 do 255.
`Decimal ili Numeric`	Decimalna/ Numerička	Varira	Fiksna preciznost i razmera od -10^{38}-1 do 10^{38}-1. Ova dva naziva predstavljaju sinonime.
`Money`	Novčana	8	Monetarne jedinice od -2^{63} do 2^{63} uz dodatak preciznosti do četiri decimalna mesta. Obratite pažnju na to da ovo može biti bilo koja monetarna jedinica, ne samo dolar.
`SmallMoney`	Novčana	4	Monetarne jedinice od -214.748,3648 do +214.748,3647.
`Float` (takođe, sinonim za ANSI `Real`)	Približni brojevi	Varira	Prihvata argument (od 1 do 53, na primer, `Float (20)`) koji određuje veličinu i preciznost. Obratite pažnju na to da je argument u bitovima, ne bajtovima. Kreće se od -1,79E + 308 do 1,79E + 308.
`DateTime`	Datum/Vreme	8	Podaci o datumu i vremenu od 1. januara 1753. godine do 31. decembra 9999. godine uz preciznost od tri stotinke sekunde.
`DateTime2`	Datum/Vreme	Varira (od 6 do 8)	Ažurirano otelovljenje cenjenijeg tipa podataka `DateTime`. Podržava veće raspone datuma i veću preciznost delova vremena (do 100 nanosekundi). Poput tipa `DateTime`, ne primećuje vremenske zone, ali je usklađen sa .NET `DateTime` tipom podataka.
`SmallDateTime`	Datum/Vreme	4	Podaci o datumu i vremenu od 1. januara 1900. godine do 6. juna 2079. godine, uz preciznost od jednog minuta.

NAZIV TIPA PODATAKA	KLASA	VELIČINA U BAJTIMA	PRIRODA PODATAKA
`DateTimeOffset`	Datum/Vreme	Varira (od 8 do 10)	Sličan tipu podataka `DateTime`, ali isto tako očekuje postavljanje odstupanja od -14.00 do +14.00 časova od vremena po UTC standardu. Vreme se interno snima po UTC-u, i sva poređenja, klase ili indeksiranje zasnivaće se na toj jedinstvenoj vremenskoj zoni.
`Date`	Datum/Vreme	3	Smešta samo podatke o datumu od 1. januara 0001. godine do 31. decembra 9999. godine, kako je definisano u gregorijanskom kalendaru. Podrazumeva format datuma po ANSI standardu (YYYY-MM-DD), ali će se implicitno pretvoriti iz nekoliko drugih formata.
`Time`	Datum/Vreme	Varira	Smešta samo podatke o vremenu sa preciznošću koju bira korisnik, a koja je toliko detaljna da ide do 100 nanosekundi (što je i podrazumevana vrednost).
`Cursor`	Posebna numerička	1	Od pokazivača do kursora. Dok pokazivač zauzima samo jedan bajt, imajte na umu da rezultirajući skup koji čini stvarni kursor takođe zauzima memoriju. Koliko će tačno varirati zavisi od rezultirajućeg skupa.
`Timestamp/ rowversion`	`Posebna numerička (binarna)`	8	Posebna vrednost koja je jedinstvena unutar date baze podataka. Sama baza podataka automatski postavlja vrednost svaki put kada se zapis unese ili ažurira iako naredba `UPDATE` nije korišćena za kolonu vremenske zone (zapravo vam i nije dozvoljeno da direktno ažurirate polje vremenske zone).
`UniqueIdentifier`	Posebna numerička (binarna)	16	Garantuje se da će poseban globalno jedinstveni identifikator (Globally Unique Identifier, GUID) biti jedinstven i kroz prostor i kroz vreme.
`Char`	Znak	Varira	Podaci sa znacima fiksne dužine. Vrednosti kraće od dužine skupa produžene su razmacima do dužine skupa. Podaci su non-Unicode standarda. Maksimalna navedena dužina je 8.000 znakova.

nastavlja se

TABELA 1-1 *(nastavak)*

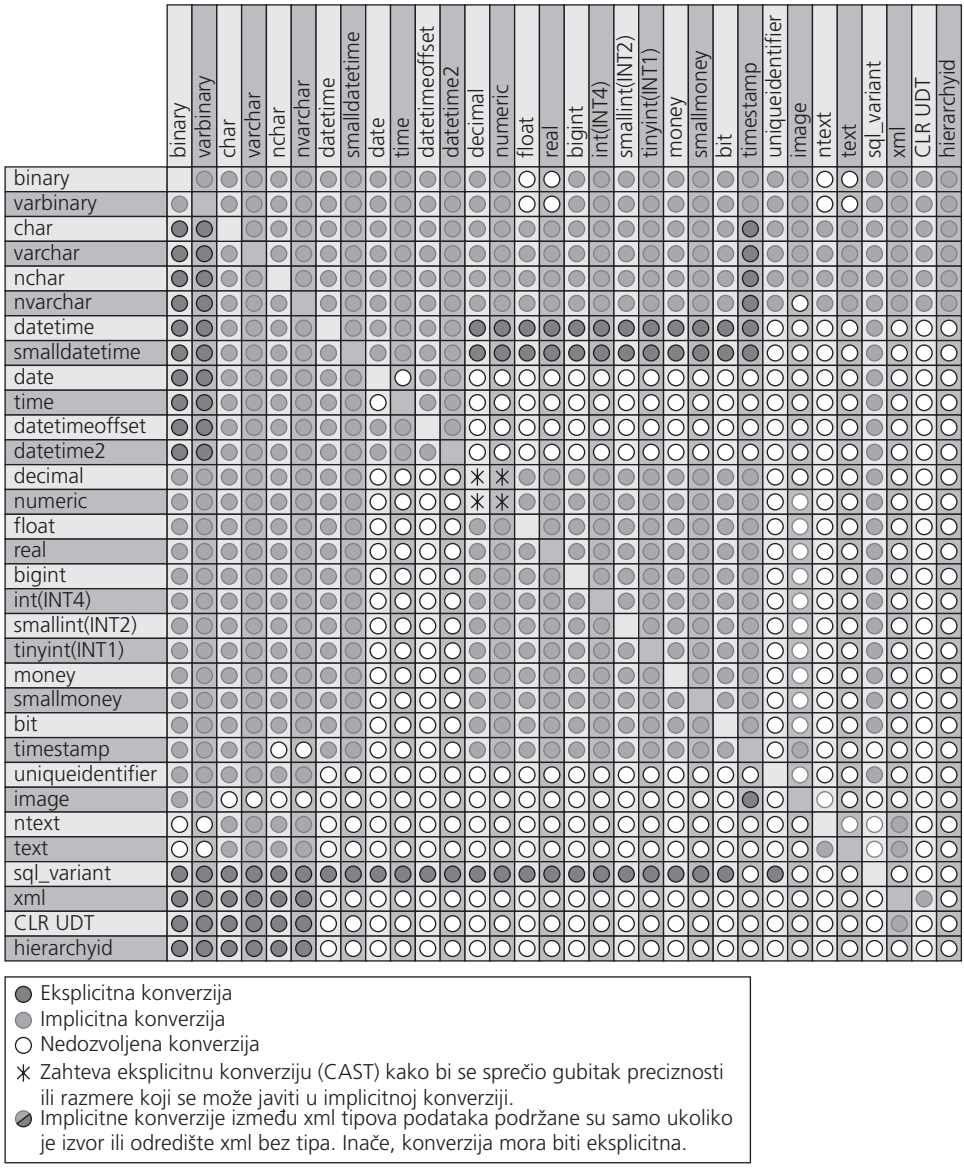
NAZIV TIPA PODATAKA	KLASA	VELIČINA U BAJTIMA	PRIRODA PODATAKA
`VarChar`	Znak	Varira	Podaci znakova promenljive dužine. Vrednosti nisu produžene razmacima. Podaci su non-Unicode standarda. Maksimalna navedena dužina je 8.000 znakova, ali možete koristiti ključnu reč `max` da biste je označili kao suštinski veoma veliko znakovno polje (do 2^31 bajta podataka).
`Text`	Znak	Varira	Zastarela podrška od SQL Servera 2005. Umesto njega koristite `varchar (max)`.
`NChar`	Unicode	Varira	Podaci Unicode znakova fiksne dužine. Vrednosti kraće od dužine skupa produžavaju se razmacima. Maksimalna navedena dužina je 4.000 znakova.
`NVarChar`	Unicode	Varira	Podaci Unicode znakova promenljive dužine. Vrednosti se ne produžavaju. Maksimalna navedena dužina je 4.000 znakova, ali možete koristiti ključnu reč `max` kako biste je označili kao suštinski veoma veliko znakovno polje (do 2^31 bajta podataka).
`Ntext`	Unicode	Varira	Podaci Unicode znakova promenljive dužine. Poput tipa podataka `Text`, ovo je samo zastarela podrška. U ovom slučaju koristite `nvarchar (max)`.
`Binary`	Binarna	Varira	Binarni podaci fiksne dužine sa maksimalnom dužinom od 8.000 bajtova.
`VarBinary`	Binarna	Varira	Binarni podaci promenljive dužine sa maksimalnom navedenom dužinom od 8.000 bajtova, ali možete koristiti ključnu reč `max` kako biste je suštinski označili kao BLOB polje (do 2^31 bajtova podataka).
`Image`	Binarna	Varira	Samo zastarela podrška od SQL Servera 2005. Umesto njega koristite `varbinary (max)` tip.
`Table`	Druga	Posebna	Koristi se prvenstveno u radu sa rezultirajućim skupovima, obično prosleđujući jedan skup iz korisnički definisane funkcije ili kao parametar za snimljene procedure. Ne može se koristiti kao tip podataka unutar definicije tabele (tabele ne možete ugnezditi).

NAZIV TIPA PODATAKA	KLASA	VELIČINA U BAJTIMA	PRIRODA PODATAKA
`HierarchyID`	Druga	Posebna	Poseban tip podataka koji održava informacije o hijerarhijskom pozicioniranju. Pruža posebnu funkcionalnost vezanu za potrebe hijerarhije. Dozvoljena su poređenja dubine, odnosa roditelja/potomka, kao i indeksiranja. Tačna veličina varira sa brojem i prosečnom dubinom čvorova u hijerarhiji.
`Sql_variant`	Druga	Posebna	Ovo je donekle vezano za tip `Variant` u jezicima VB i C++. Suštinski, to je sadržilac koji vam dozvoljava da u njemu smeštate većinu drugih tipova podataka SQL Servera. To znači da ga možete koristiti kada je potrebno da jedna kolona ili funkcija ima mogućnost rada sa više tipova podataka. Za razliku od VB-a, korišćenje ovog tipa podataka primorava vas da ga *eksplicitno* konvertujete kako biste ga pretvorili u određeniji tip podataka.
`XML`	Znak	Varira	Definiše znakovno polje kao da je namenjeno XML podacima. Pruža proveru ispravnosti podataka naspram XML šeme, i omogućava upotrebu posebnih funkcija orijentisanih na XML.
`CLR`	Druga	Varira	Varira zavisno od specifične prirode CLR objekta podržavajući prilagođeni tip podataka zasnovan na CLR-u. Prostorni tipovi podataka poput `GEOMETRY` i `GEOGRAPHY` koji su sastavni deo SQL Servera 2012 implementirani su u CLR tipove.

Većina ovih tipova podataka ima svoje ekvivalente u drugim programskim jezicima. Na primer, `int` u SQL Serveru jednak je tipu `Long` u Visual Basic-u, i za većinu sistema i prevodilačkih kombinacija u C++-u ekvivalentan je `int` tipu sa predznakom.

NAPOMENA *U SQL Serveru ne postoji pojam numeričkih tipova podataka bez predznaka.*

Uopšteno govoreći, tipovi podataka SQL Servera funkcionišu onako kako biste to i očekivali imajući u vidu iskustvo u radu sa većinom drugih savremenih programskih jezika. Dodavanje brojeva daje sumu, dok ih dodavanje nizova nadovezuje. Kada pomešate upotrebu ili dodeljivanje promenljivih ili polja različitih tipova podataka, *implicitno* (ili automatski) se konvertuje izvestan broj tipova. Većina drugih tipova može se eksplicitno konvertovati. (Tačno naglašavate u koji tip želite da konvertujete.) Nekoliko njih se uopšte ne može međusobno konvertovati. Na slici 1-3 dat je grafikon koji pokazuje raznovrsne konverzije koje su moguće.



SLIKA 1-3

Zašto biste morali da konvertujete neki tip podataka? Pa, dozvolite mi da to ilustrujem jednostavnim primerom. Ukoliko ste želeli da ispišete frazu `Today's date is ##/##/####`, gde je ##/##/#### aktuelni datum, mogli ste je napisati na sledeći način:


```
SELECT 'Today''s date is ' + GETDATE()
```

Deo koda Chap01.SQL

NAPOMENA *O Transact-SQL naredbama poput ove, mnogo detaljnije ću govoriti kasnije, ali bi očekivani rezultat prethodnog primera trebalo da bude očigledan za vas.*

Problem je u tome što bi ova naredba dala sledeći rezultat:

```
Msg 241, Level 16, State 1, Line 1
Conversion failed when converting date and/or time from character string.
```

To nije ono što ste želeli, zar ne? Sada, to isto pokušajte pomoću funkcije `CONVERT ()`:

```
SELECT 'Today''s date is ' + CONVERT(varchar(12), GETDATE(),101)
```

Korišćenjem funkcije `CONVERT` na ovaj način, za rezultat dobijate nešto ovako:

```
-----------------------------------
Today's date is 01/01/2012
(1 row(s) affected)
```

Tipovi podataka datuma i vremena, kao što je rezultat funkcije `GETDATE ()` implicitno se ne konvertuju u tekstualni tip podataka, kao što je `Today's date is`, ali ćete ipak redovno nailaziti na ovakve konverzije. Na sreću, funkcije `CAST` i `CONVERT ()` omogućavaju vam da obavljate konverziju između mnogih SQL Server tipova podataka. O funkcijama `CAST` i `CONVERT ()` više ću govoriti u narednom poglavlju.

Ukratko, tipovi podataka u SQL Serveru obavljaju praktično istu funkciju koju imaju i u drugim programskim okruženjima. Oni pomažu u sprečavanju programskih grešaka tako što obezbeđuju da je dati podatak iste one prirode koje bi i trebalo da bude (zapamtite da 1/1/1980 ima jedno značenje kao datum, a drugo kao broj) i da je vrsta obavljene operacije ona koju i očekujete.

NULL podaci

Šta bi se desilo kada biste imali red u kome nema podataka za odgovarajuću kolonu – to jest, šta ako jednostavno ne znate koja je vrednost? Na primer, pretpostavimo da imate zapis koji pokušava da smesti informacije o radu kompanije za datu godinu. Sada zamislite da jedno od polja predstavlja procentualni rast tokom prethodne godine, ali nemate nikakve zapisane podatke za tu godinu ispred prvog zapisa u vašoj bazi podataka. Možda biste došli u iskušenje da jednostavno unesete nulu kao vrednost u koloni `PercentGrowth`. Međutim, da li će to dati prave informacije? Ljudi koji nisu upućeni možda bi pomislili da to znači da nije bilo nikakvog procentualnog rasta, a zapravo, vi prosto ne znate vrednost za tu godinu.

Za vrednosti koje su neodređene kaže se da su `prazne` (`NULL`). Čini mi se da, kada svaki put nekom razredu predajem o programiranju, najmanje jedan student od mene traži da definišem `NULL` vrednost. Pa, to je teško, jer po definiciji, `NULL` vrednost znači da ne znate koja je zapravo vrednost. Mogla bi biti 1. Mogla bi biti 347. Mogla bi biti -294. Ukratko, to znači *ne znam, nedefinisana je*, ili možda *neprimenljiva*.

SQL SERVER NAZIVI OBJEKATA

Do sada ste čuli raznorazne stvari o objektima u SQL Serveru. Vreme je da se detaljno pozabavimo imenovanjem objekata u SQL Serveru.

Šta dobija naziv?

U osnovi, u SQL Serveru sve ima svoj naziv. Evo delimične liste:

- Snimljene procedure
- Tabele
- Kolone
- Prikazi
- Pravila
- Ograničenja
- Podrazumevane vrednosti
- Indeksi
- Grupe datoteka
- Okidači
- Baze podataka
- Serveri
- Funkcije koje definiše korisnik
- Sekvence
- Prijavljivanja
- Uloge
- Tekstualni katalozi
- Datoteke
- Tipovi koje definiše korisnik

I lista se nastavlja. Većina stvari kojih se mogu setiti, osim redova (koji zapravo i nisu objekti) imaju svoje ime. Trik je u tome da svako ime, koje date, bude korisno i praktično.

Pravila imenovanja

Kao što sam ranije u poglavlju spomenuo, pravila za imenovanje u SQL Serveru su prilično fleksibilna, dozvoljavajući da se u nazivima nađe nešto kao što su ugrađeni razmaci, pa čak i ključne reči. Kao i kod većine slobodnog ponašanja, međutim, lako je napraviti pogrešan izbor i na taj način napraviti problem.

Evo glavnih pravila:

- Ime vašeg objekta mora početi nekim slovom, po definiciji specifikacije za Unicode 3.2. Ovaj standard uključuje slova na koja je većina zapadnjaka naviknuta: od A do Z, odnosno, od a do z. Da li će se "A" razlikovati od "a" zavisi od načina na koji je vaš server konfigurisan, ali i jedno i drugo slovo je dobro za početak imena objekta. Nakon tog prvog slova, imate punu slobodu da radite šta hoćete; može se upotrebiti skoro svaki znak.
- Ime može da sadrži do 128 znakova za normalne objekte i 116 za privremene objekte.

- Sva imena koja su ista kao ključne reči SQL Servera ili sadrže ugrađene razmake moraju se staviti pod navodnike (“”) ili u uglaste zagrade ([]). Koje reči se smatraju ključnim varira zavisno od nivoa kompatibilnosti na koji ste podesili svoju bazu podataka.

NAPOMENA *Obratite pažnju na to da su dvostruki znaci navoda prihvatljivi kao znaci za razdvajanje u nazivima kolona samo ukoliko postoji podešavanje* `SET QUOTED_IDENTIFIER ON`. *Korišćenjem uglastih zagrada (*`[` *i* `]`*) izbegava se mogućnost da će vaši korisnici imati pogrešno podešavanje.*

Ova pravila se uopšteno govoreći nazivaju pravilima za identifikatore i koriste se za sve objekte u SQL Serveru kojima dajete naziv, ali mogu pomalo varirati ukoliko imate lokalizovanu verziju SQL Servera (onu koja je adaptirana za određene jezike, dijalekte ili geografska područja). Za određene vrste objekata mogu postojati dodatna pravila.

NAPOMENA *Iskoristiću ovo kao prvi povoljan trenutak da oštro iskritikujem temu o imenovanju objekata. SQL Server ima mogućnost da ugradi razmake u nazive, i, u nekim slučajevima, da koristi ključne reči kao imena. Oduprite se iskušenju da koristite bilo koju od ove dve stvari! Kolone sa ugrađenim razmacima u svojim nazivima imaju lepo zaglavlje kada upotrebite naredbu* `SELECT`*, ali postoje drugi načini da se postigne isti rezultat. Korišćenje ugrađenih razmaka i ključnih reči za nazive kolona jeste preklinjanje da dođe do grešaka, zbrke i drugih katastrofa. Kasnije ću govoriti o tome zašto je Microsoft izabrao da dozvoli ovo pravilo, ali za sada, samo zapamtite da ugrađeni razmaci i ključne reči u nazivima treba da vas asociraju na zle sile, mučenje ili smrt. (Ovo neće biti poslednji put da o ovome slušate od mene.)*

REZIME

Kao i većina stvari u životu, sitnice su bitne kada razmišljate o RDBMS-u. Sigurno da svako ko dovoljno zna da čak i pomisli da izabere ovu knjigu, ima predstavu o *konceptu* smeštanja podataka u kolone i redove, čak i ako ne zna da bi ove grupe kolona i redova trebalo nazivati tabelama. Ali tek nekoliko tabela retko kad čini pravu bazu podataka. Stvari koje današnje RDBMS-ove čine velikim jesu one dodatne stvari – objekti koji vam omogućavaju da funkcionalnost i pravila poslovanja koja imaju veze sa podacima, smestite pravo u bazu sa podacima.

Podaci baze podataka imaju svoj *tip* (*type*), kao što ih ima i većina drugih programskih okruženja. Većina stvari koje obavljate u SQL Serveru imaće barem neka razmatranja o tipu. Pregledajte koji su tipovi dostupni, pa razmislite o tome kako se ovi tipovi poklapaju sa tipovima podataka u bilo kom drugom programskom okruženju sa kojim ste upoznati.

VEŽBANJA

1. Koja je svrha baze podataka `master`?
2. Koje su dve razlike između tipova podataka `datetime` i `datetime2`?

➤ ŠTA STE NAUČILI U OVOM POGLAVLJU

TEMA	KONCEPT
Vrste objekata	Nova instalacija SQL Servera pravi baze podataka i dozvoljava pravljenje mnogo vrsta objekata unutar tih baza podataka kao i izvan njih.
Tipovi podataka	SQL Server nudi razne tipove podataka koji se mogu koristiti kako bi se informacije smeštale efikasno i ispravno.
Identifikatori	Svojim objektima možete dati naziv koji počinje nekim slovom, koristeći do 128 znakova (116 za privremene objekte).