

D E O

## Uvod



## POGLAVLJE

1

# Uvod u programski jezik PL/SQL

**M**i smo imali priliku da vidimo veoma dobro napisan programski kod koji je sačinjavao veoma loše aplikacije. Uzmite, na primer, veoma dobro napisane viruse ili softverske kompanije koje prave privlačne, ali beskorisne aplikacije! Programiranje nije samo pisanje sintaksno ispravnog programskog koda. To je zanimanje koje znanje kombinuje sa genijalnošću, sposobnostima za komunikaciju, stavom prema drugim ljudima i disciplinom, kako bi se izgradila uspešna karijera i kako bi se napravile svetski priznate aplikacije.

U ovoj knjizi mi se nećemo baviti samo sintaksom i pravilima pisanja programskog koda. Mi ćemo dati odgovor na pitanje "Zbog čega to treba da koristim?". To je pitanje koje svi mi postavljamo kada se susretнемo sa novim osobinama. Naše razmatranje prevazilazi činjenicu da Oracle može nešto da uradi. Pokazaćemo vam kako i zašto nešto radi.

U ovom prvom poglavlju postavićemo osnove za razmatranja u preostalom delu knjige. Razmatraćemo sledeće:

- Programska jezik SQL i njegovu interakciju sa relacionom bazom podataka.
- Kako se u programskom jeziku PL/SQL koristi SQL radi povećanja mogućnosti.
- Koncepte programiranja. Osim toga, uporedićemo proceduralne programske jezike i objektno-orientisano programiranje.
- Istorijat i osobine programskog jezika PL/SQL.
- Prednosti (i nedostatke) programskog jezika.
- Kako da pristupite informacijama u preostalom delu knjige i kako da najbolje iskoristite potpuno prerađeni sadržaj koji se nalazi u knjizi.

## Uvod u programske jezike

Programski jezici Java, C++, PL/SQL i Visual Basic su neki od najpopularnijih savremenih programskih jezika. Ovi programski jezici se međusobno veoma razlikuju i imaju jedinstvene karakteristike. Iako su to potpuno različiti programski jezici, neki od njih imaju zajedničke osobine. Oni se mogu kategorizovati prema zajedničkim osobinama. Programske jezike koje smo naveli možemo svrstati u dve kategorije: proceduralne programske jezike i objektno-orientisane programske jezike.

Proceduralni programski jezici, kakvi su PL/SQL i Visual Basic, su linearni. Programi napisani u njima započinju rad na početku, a završavaju na kraju. Ovo je pojednostavljena definicija proceduralnih programskih jezika, ali ipak naglašava najvažniju razliku između proceduralnih i objektno-orientisanih programskih jezika. Svaki programski red, pre nego što se izvrši, mora sačekati da se izvrši prethodni programski red. Za mnoge programere početnike najbolje je da iskustvo i znanje steknu koristeći proceduralne programske jezike. Pred vama je niz koraka koje program mora da izvrši, a to je upravo način na koji se programski kod izvršava - korak po korak.

Objektno-orientisani programski jezici (OOP), kakvi su Java i C++, su po prirodi apstraktniji. U objektno-orientisanim programskim jezicima se koriste strukture koje se nazivaju *objekti*. Na primer, umesto da napišemo programski kod pomoću kojeg se iz struktura podataka dobijaju informacije o knjizi, mi možemo napraviti *objekat* koji ćemo nazvati BOOK (KNJIGA).

Svaki *objekat* ima atribute: broj stranica, cenu, naslov i tako dalje. Atributima se opisuje objekat. Metodi su više orijentisani na izvršavanje akcija. Metodi operišu sa podacima, izdvajaju ih i menjaju. Ukoliko je potrebno da, na primer, promenite cenu, vi ćete pozvati metod pomoću kojeg ćete to obaviti. Ovakav postupak se razlikuje od postupka koji se koristi u proceduralnom programskom jeziku u kojem morate izvršiti niz koraka kako biste postigli isti efekat.

Među retkim programskim jezicima koji se mogu smatrati proceduralnim i objektno-orijentisanim programskim jezicima je i programski jezik PL/SQL. Objekti su uvedeni u Oracle 8, iako se u prvim verzijama nisu mogle koristiti napredne osobine kakve su nasleđivanje, evolucija tipova podataka i dinamičko otpremanje metoda. Sa pojmom Oracle 9iR1, Oracle je krupnim koracima počeo da grabi ka potpuno objektno-orijentisanim programiraju u programskom jeziku PL/SQL. Od Oraclea 10g većina osobina objektno-orijentisanih programskih jezika je u potpunosti ugrađena.

#### NAPOMENA

Osobine objektno-orijentisanih jezika koje ovde spominjemo su u potpunosti objašnjene u Poglavljima 14 i 15.

### Beleška za programere početnike

Kao i mnogi programeri, ja sam svoje iskustvo stekao koristeći programski jezik Basic. Sintaksa ovog programskog jezika se veoma lako uči, a sve što važi u ovom programskom jeziku se može primeniti na većinu drugih. Verujem da ćete i vi uvideti da se mnogo toga može primeniti na programski jezik PL/SQL.

Osobina programskog jezika PL/SQL koja mi je najdraža nije njegova tesna integracija sa bazom podataka (iako je programski jezik veoma tesno integriran), niti napredni koncepti i osobine programskog jezika (kada budete u celosti pročitali knjigu bićete zapanjeni šta se sve može uraditi) ili bilo koja druga funkcionalnost koja postoji. Osobina koja je meni najdraža jeste strukturalni pristup programiranju. Za svaku rezervisanu reč BEGIN postoji rezervisana reč END. Za svako IF postoji END IF.

Kao predavač koji mnoge studente, koji prethodno nikada nisu programirali, uči programskom jeziku PL/SQL, ja sa sigurnošću znam da možete naučiti ovaj programski jezik. Programski jezik PL/SQL je strukturiran, linearan i ne pravi greške. To je veoma dobro! Naučite strukturu i pravila. Ukoliko ne budete poštovali pravila odmah ćete, prilikom pokretanja programskog koda, dobiti odgovarajuću poruku.

Warning: Procedure created with compilation errors.

#### SAVET

Očigledno je da struktura ne garantuje pravljenje dobrog programskog koda. Struktura samo olakšava učenje programskog jezika. Potrudite se da pravilno koristite forme programskog koda, da usvojite odgovarajuća pravila imenovanja, dokumentujete akcije i vežbate, vežbate, vežbate... Ne dozvolite sebi da koristite prečice koje će programski kod koji pišete učiniti neefikasnim i teškim za održavanje. Kao što je slučaj sa bilo kojim programskim jezikom, možete napisati očajan programski kod koji se može kompajlirati bez problema.

Možda ste primetili da najbolji programeri ne moraju biti nadareni za tehniku. Najbolji programeri su dobri u komunikaciji sa drugim ljudima i imaju sposobnost da se stave u položaj korisnika i mušterija. U etapi projektovanja programa ove sposobnosti su od neprocenjive važnosti. U toku projektovanja programa vi se srećete sa rukovodiocima projekta, drugim programerima, administratorima baze podataka (DBA), krajnjim korisnicima, QA inženjerima i menadžmentom. Svaka od ovih grupa ljudi postavlja različite ciljeve koje treba ispuniti tokom postupka pravljenja sistema, i svaka od ovih grupa će pred vas postaviti različite zahteve. Vaš stav prema ljudima i sposobnost da sa njima komunicirate određuje uspeh ili neuspeh projekta i, na kraju krajeva, određuje koliko daleko ćete dogurati kao programer.

## PL/šta?

Dakle, šta je to PL/SQL? To je proceduralno (a ponekad i objektno-orientisano) proširenje programskog jezika SQL, koje je Oracle napravio ekskluzivno za bazu podataka Oracle. Ukoliko poznajete jedan drugi programski jezik, programski jezik Ada, uvidećete da između ovih programskih jezika postoje velike sličnosti. Razlog zbog kojeg su ova dva programska jezika toliko slična jeste taj što je programski jezik PL/SQL proizašao iz programskog jezika Ada, od kojeg je pozajmio mnoge koncepte.

Skraćenica PL u imenu PL/SQL označava *proceduralni programski jezik* (engl. procedural language, PL). Programski jezik SQL koristimo za izdvajanje (SELECT), zapisivanje (INSERT), ažuriranje (UPDATE) i uklanjanje (DELETE) podataka. Programski jezik SQL koristimo za pravljenje i održavanje objekata i korisničkih naloga, kao i za upravljanje dozvolama za korišćenje instanci koje smo napravili.

## Structured Query Language (SQL)

SQL (izgovara se sequel ili se izgovara tako što se čitaju slova koja čine skraćenicu) je ulaz, odnosno prozor, u bazu podataka. SQL je programski jezik četvrte generacije (4GL) koji je napravljen tako da se može brzo naučiti i lako koristiti.

Osnovna sintaksa programskog jezika SQL nije tvorevina koja je nastala u Oracleu. Sintaksa ovog programskog jezika je nastala na radu doktora E.F. Koda (E.F. Codd) i IBM-a u ranim 1970-im godinama. Američki nacionalni institut za standardizaciju (American National Standards Institute, ANSI) priznaje programski jezik SQL i objavljuje standarde za ovaj programski jezik.

Oracle se pridržava standarda koje institut za standardizaciju ANSI preporučuje za programski jezik SQL ali, takođe, ugraduje dodatke kroz pomoći program SQL\*Plus. Zahvaljujući pomoćnom programu SQL\*Plus, u bazi podataka Oracle možete koristiti komande i osobine koje nisu deo standarda koji je preporučen za programski jezik SQL. SQL\*Plus je pomoći program koji se može koristiti u različitim oblicima:

- **Sa komandne linije** Komande možete unositi sa UNIX prompta ili DOS prompta

- **Kao grafički korisnički interfejs** SQL\*Plus Client, SQL Worksheet, Enterprise Manager
- **Web stranica** iSQL\*Plus, Enterprise Manager u verziji 10g

Kada je instaliran samo klijent, možemo podesiti uspostavljanje veze preko mreže sa udaljenim bazama podataka. U Oracleu 10g je podešavanje još lakše jer možete upotrebiti Enterprise Manager i iSQL\*Plus, koji se mogu podesiti prilikom instaliranja.

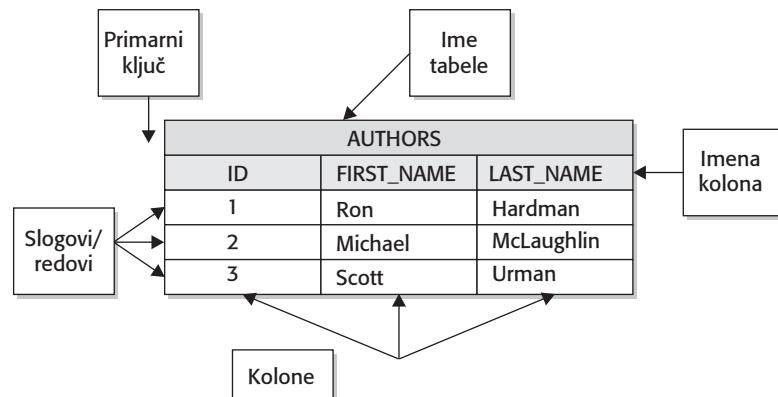
### Pregled relacionih baza podataka

SQL je prozor u bazu podataka, ali šta je baza podataka? *Baza podataka* je bukvalno sve što skladišti podatke. Elektronska baza podataka može biti jednostavna koliko i radna tabela ili dokument u kojem se obrađuje tekst.

Kao što prepostavljate, zapisivanje velikih količina podataka u radnu tabelu ili dokument u kojem se obrađuje tekst može veoma brzo postati neuspešno. U ovakvim jednodimenzionalnim bazama podataka ne postoji efikasan način za izdvajanje redundantnih podataka, za konzistentno unošenje podataka ili za dobijanje podataka.

Oracle je *sistem za upravljanje relacionom bazom podataka*, ili RDBMS (engl. relational database management system). Tabele se sastoje od kolona kojima se definiše tip podataka koji se u njih može zapisati (slova, cifre i tako dalje). U svakoj tabeli postoji najmanje jedna *kolona*. Kada se podaci zapisuju u tabelu, oni se zapisuju u *redove*. Ovo važi za sve relacione baze podataka (pogledajte sliku 1.1).

U bazi podataka Oracle vlasnik tabele je korisnik ili šema. Šema je kolekcija objekata, kao što su tabele, čiji vlasnik je korisnik baze podataka. Moguće je da u jednoj bazi podataka postoje dve tabele koje imaju isto ime ukoliko su njihovi vlasnici različiti korisnici.



**SLIKA 1.1** Struktura tabele

Drugi proizvođači relacionih baza podataka ne moraju primenjivati ovakvu organizaciju. U SQL Serveru se, na primer, koristi drugačija terminologija. Baza podataka SQL Server više nalikuje šemi u bazi podataka Oracle, a server SQL Server podseća na bazu podataka Oracle. Međutim, krajnji rezultat je isti. Objekti, kao što su recimo tabele, uvek imaju vlasnika.

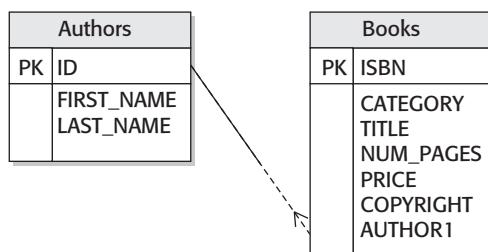
Moguće je sve naše podatke zapisati u jednoj tabeli, kao kada bismo ih zapisivali u radnoj tabeli, ali na taj način ne bismo mogli da iskoristimo prednosti relacionih osobina baze podataka Oracle. Na primer, tabela u koju su zapisani podaci o knjigama izdavačke kuće Oracle Press ne bi bila potpuna bez informacija o autorima. Moguće je da autor napiše više različitih knjiga. U modelu u kojem se koristi jedna datoteka, odnosno jedna tabela, ime autora se prikazuje više puta. Ovakva redundantnost podataka se može izbeći deljenjem podataka u dve tabele u kojima će postojati kolona pomoću koje se odgovarajući podaci mogu povezati. Na slici 1.2 vidite kako se jedna ovakva tabela može podeliti na dve zasebne tabele.

Na slici 1.2 su prikazane dve tabele, tabele AUTHORS i BOOKS. U tabelu AUTHORS se samo jednom zapisuje ime i prezime autora. Svakom redu podataka u tabeli se pridružuje identifikator (ID) pomoću kojeg se obezbeđuje jedinstvenost podataka i njihovo postojanje (null znači prazno, a not null znači da polje nije prazno).

Pošto postoji tabela AUTHORS, to znači da informaciju o autoru ne moramo ponavljati za svaku knjigu koju je napisao. U tablu BOOKS ćemo dodati kolonu AUTHOR1 u u nju ćemo zapisati odgovarajući ID (identifikator autora) iz tabele AUTHORS za svaku knjigu u tabeli BOOKS. Pomoću stranog ključa (FOREIGN KEY) nad kolonom BOOKS.AUTHOR1 možemo, koristeći SQL, uspostaviti vezu između dve tabele. Pogledajmo primer:

#### NAPOMENA

Možda ćete poželeti da upotrebite skript CreateUser.sql koji se na web sajtu nalazi u direktorijumu koji je posvećen ovom poglavljju. Pomoću ovog skripta se pravi korisnik čije je ime plsql i odobravaju mu se odgovarajuće dozvole.



**SLIKA 1.2** ERD za tabele Books i Authors.

```
-- Available online as part of PlsqlBlock.sql
CREATE TABLE authors (
    id      NUMBER PRIMARY KEY,
    first_name VARCHAR2(50),
    last_name  VARCHAR2(50)
);

CREATE TABLE books (
    isbn      CHAR(10) PRIMARY KEY,
    category   VARCHAR2(20),
    title     VARCHAR2(100),
    num_pages NUMBER,
    price      NUMBER,
    copyright  NUMBER(4),
    author1    NUMBER CONSTRAINT books_author1
                REFERENCES authors(id)
);
```

Pošto u tabele unesemo nekoliko slogova, možemo zadati upit SELECT pomoću kojeg ćemo tabele spojiti u skladu za zavisnošću koja postoji.

```
SELECT b.title, a.first_name, a.last_name
FROM authors a, books b
WHERE b.author1 = a.id;
```

Prethodnim upitom se spajaju dve tabele i dobijaju se podaci kao da su zapisani u jednoj tabeli (datoteci). Razlike koje postoje u ovakvom načinu zapisivanja podataka su manja redundantnost podataka (podaci se ne ponavljaju), manje mogućnosti za pravljenje grešaka i veća fleksibilnost. Da bih mogao da dodam informacije o izdavaču sve što treba da uradim jeste da napravim tabelu PUBLISHER koja sadrži ID i da u tabeli BOOKS napravim kolonu nad kojom će postojati strani ključ (FOREIGN KEY) koji će pokazivati na kolonu PUBLISHER.ID.

#### NAPOMENA

Detaljan opis programskog jezika SQL ćete pronaći u dokumentaciji koja se nalazi na web adresi <http://otn.oracle.com>.

### Poređenje programskih jezika PL/SQL i SQL

Programski jezik SQL nam omogućava potpun rad sa podacima. Kada kažem potpun rad to znači da možemo dobiti sve podatke koje zatražimo ... na kraju krajeva ... u većini slučajeva.

Što se tiče efikasnog načina dobijanja podataka, ne postoje nikakve grancija, a nekoliko programskih osobina, koje postoje u većini programskih jezika, se mogu upotrebiti. Pomoću programskog jezika SQL ne možemo:

- Ciklično prolaziti kroz slogove i raditi sa njima pojedinačno.
- Zaštititi programski kod enkripcijom i trajno ga zapisati na serverskom računaru umesto na klijentskom računaru.
- Obradivati izuzetke.
- Raditi sa promenljivama, parametrima, kolekcijama, sloganima, nizovima, objektima, kurosima, izuzecima, BFILE-ovima i tako dalje.

Iako je programski jezik SQL veoma moćan, a SQL\*Plus (Oracleov interfejs za programski jezik SQL) sadrži komande i ugrađene funkcije koje nisu deo standarda ANSI, programski jezik SQL pre obezbeđuje način za rad sa bazom podataka nego što omogućava programiranje. Programski jezik PL/SQL predstavlja proširenje programskog jezika SQL. U programski jezik PL/SQL su ugrađene osobine koje smo pomenuli, kao i mnoge druge osobine.

#### NAPOMENA

Nemojte se brinuti ukoliko niste čuli za sve programske osobine koje smo pomenuli! Ovu knjigu služi upravo zato da biste ih naučili. Sve te osobine su detaljno objašnjene u poglavljima koja slede.

Sve osobine programskog jezika SQL postoje u programskom jeziku plsq. Zapravo, počevši od Oraclea 9iR1, PL/SQL parser je identičan SQL parseru, što obezbeđuje da se komande jednakostretiraju bez obzira na mesto izvršavanja. Pre Oraclea 9iR1 su postojali slučajevi u kojima su se SQL upiti potpuno drugačije tretirali. To više nije slučaj.

Pogledaćemo primer upita nad tabelama BOOKS i AUTHORS koji smo ranije izvršili, ali ovaj put ćemo ga upotrebiti u PL/SQL primeru.

```
-- Available online as part of PlsqlBlock.sql
SET SERVEROUTPUT ON
DECLARE
    v_title books.title%TYPE;
    v_first_name authors.first_name%TYPE;
    v_last_name authors.last_name%TYPE;

    CURSOR book_cur IS
        SELECT b.title, a.first_name, a.last_name
        FROM authors a, books b
        WHERE a.id = b.author1;
    BEGIN
        DBMS_OUTPUT.ENABLE(1000000);
        OPEN book_cur;
    LOOP
        FETCH book_cur INTO v_title, v_first_name, v_last_name;
        EXIT WHEN book_cur%NOTFOUND;
```

```
IF v_last_name = 'Hardman'  
THEN  
    DBMS_OUTPUT.PUT_LINE('Ron Hardman co-authored ''||v_title);  
ELSE  
    DBMS_OUTPUT.PUT_LINE('Ron Hardman did not write ''||v_title);  
END IF;  
END LOOP;  
  
CLOSE book_cur;  
  
EXCEPTION  
WHEN OTHERS  
THEN  
    DBMS_OUTPUT.PUT_LINE(SQLERRM);  
END;  
/
```

U prethodnom primeru smo upotrebili upit SELECT koji smo ranije koristili, ali ovoga puta se ciklično prolazi kroz sve rezultate upita, i utvrđuje se da li je "Hardman" prezime autora i ukoliko jeste, onda se izlaz adekvatno formatira. Mogućnosti programskog jezika SQL četvrte generacije (SQL 4GL) se kombinuje sa mogućnostima proceduralnog programskog jezika treće generacije (3GL).

#### NAPOMENA

Obratite pažnju na strukturu poslednjeg programskog bloka. Za svaku rezervisanu reč begin postoji ezervisana reč end.

### Poređenje programskih jezika PL/SQL i Java

U bazu podataka Oracle 8i je uvedena podrška za programski jezik Java, i Java Stored Procedures (uskladištene procedure napisane u programskom jeziku Java). Zbog čega ne bismo onda koristili samo programski jezik Java?

Programski jezi PL/SQL je, i oduvek je bio, veoma dobro integriran u bazu podataka Oracle. Oracle nastavlja da unapređuje performanse programskega jezika PL/SQL dodavanjem osobina kao što je ugrađena sposobnost kompajliranja programskega koda napisanog u programskom jeziku PL/SQL. To znači da se prilikom kompajliranja programskega koda on prevodi u programske jezik C (programske jezik koji je korišćen za pravljenje Oraclea). To znači da prilikom izvršavanja programskega koda ne postoji prevodenje između sintakse koja se koristi u programskom jeziku PL/SQL i sintakse koja se koristi u programskom jeziku C. Na ovaj način su značajno poboljšane performanse - performanse su bolje i do 30 procenata u poređenju sa režimom u kojem se programski kod interpretira (što je osnovni režim rada).

Druga prednost programskega jezika PL/SQL nad programskim jezikom Java jeste njegova kompaktnost. Upit napisan u programskom jeziku SQL možete pretvoriti u programske blok jezika PL/SQL (programske blokovi su objašnjeni u Poglavlju 3) tako što ćete ispred upita dodati rezervisanu reč BEGIN, a iza upita rezervisanu reč END.

Isto pravilo se ne može primeniti u programskom jeziku Java. Sledeci blok programskog koda je najjednostavniji programski blok koji se može napraviti u programskom jeziku PL/SQL:

```
BEGIN  
    NULL;  
END;  
/
```

Isprobajte prethodni programski kod - zasigurno će se izvršiti. Ovaj programski blok ne radi apsolutno ništa, ali se ipak izvršava.

Evo još nekoliko specifičnih osobina programskog jezika PL/SQL:

- Programska jezik plsq koristi isti parser kao i programska jezik SQL. Zbog toga sasvim sigurno postoji konzistentnost između dva interfejsa.
- Programska kod napisan u programskom jeziku PL/SQL se može izvršiti u programskom kodu koji je napisan u programskom jeziku SQL.
- Osobine objektno-orientisanih programske jezika se kontinuirano ugrađuju u programska jezik PL/SQL, te se na taj način uklanjaju razlozi za korišćenje programske jezike Java.

To ne znači da *uvek* treba da koristite programska jezik PL/SQL i da *nikada* ne koristite programska jezik Java. U programskom jeziku Java postoji veliki broj osobina koje još uvek ne postoje u programskom jeziku PL/SQL. Programska jezik Java ipak nije zamena za programska jezik PL/SQL. Programska jezik PL/SQL je samo alternativa.

#### **N A P O M E N A**

Od kada je programska jezik Java počeo da se koristi u radu sa bazom podataka, ja sam veoma često slušao glasine da će istisnuti programska jezik PL/SQL. To jednostavno nije istina.

### **Istorijat i osobine programskog jezika PL/SQL**

Ono što vi uočavate kao bogat skup osobina u najnovijim verzijama programskog jezika PL/SQL je zapravo 13 godina (toliko je godina prošlo u vreme kada pišemo ovu knjigu) kontinuiranog razvoja i unapređivanja programskog jezika koji su se odvijali u Oracleu. PL/SQL je programska jezik koji je nastao iz potreba, kako u samom Oracleu tako i van njega. Iako su mnoge osobine nastale kako bi se udovoljilo zahtevima koje su postavili programeri baza podataka, veliki broj osobina je nastao zbog potreba Oraclea za funkcionalnošću u svom razvojnog programu. Za mene kao programera je veoma ohrabrujuće saznanje da Oracle koristi iste tehnologije na koje se i ja oslanjam u svojoj karijeri programera.

Teško mogu zamisliti da u bazi podataka Oracle ne postoji programska jezik PL/SQL, iako nije prošlo mnogo vremena od kada se prvi put pojavio.

#### **Verzija 1.x**

Programski jezik PL/SQL verzija 1.0 je predstavljen 1991. zajedno sa verzijom 6.0 servera podataka. Kao što je slučaj sa svakim novim programskim jezikom i ovom programskom jeziku su

nedostajale osobine koje očekujete da postoje u dorađenim verzijama. Međutim, programeri koji koji koriste Oracle su ga veoma dobro prihvatili, pošto im je pružio osobine kao što je uslovno grananje pomoću komande IF-THEN, koja tada nije postojala u programskom jeziku SQL.

### **Verzija 2.x**

Od verzije 2.3 programskog jezika PL/SQL (koja je predstavljena zajedno sa verzijom 7.0 baze podataka), Oracle u programski jezik ugrađuje mogućnost korišćenja uskladištenih procedura i funkcija, i osim toga, u programski jezik ugrađuje brojne pakete. Programski jezik PL/SQL je ključ uspeha Oracleovih alata za programiranje, a Oracle Applications se značajno oslanja na dobru integraciju programskog jezika PL/SQL i servera podataka.

### **Verzija 8.0**

U bazu podataka Oracle 8.0 je ugrađena podrška za rad sa objektima. Iako podrška za rad sa objektima u ovoj verziji baze podataka nije bila naročito bogata, ipak nam je nagovestila u kom pravcu Oracle razvija programski jezik. Objektno-orientisana poboljšanja se i dalje kontinuirano ugrađuju, pa tako postoje i u najnovijoj verziji 10gR1. Bitna izmena koja se pojavila u verziji 8.0 jeste mogućnost pravljenja verzija u programskom jeziku PL/SQL i serveru podataka. Programski jezik PL/SQL je počeo da prati isti redosled verzija kao server podataka u koji je ingerisan.

### **Verzija 8.1**

Sa pojavom baze podataka Oracle 8i, u prvom planu nije više promovisanje osobina programskog jezika PL/SQL već integracija programskog jezika Java u "bazu podataka za Internet". To, ipak, ne znači da u programskom jeziku PL/SQL ne postoje novine. Jedno od unapređenja koje mi je omiljeno jeste ugrađeni dinamički SQL (engl. Native Dynamic SQL (NDS)) koji nam je doneo komandu EXECUTE IMMEDIATE! Ja prosto obožavam ovu komandu - zapravo, komandu EXECUTE IMMEDIATE čete pronaći u gotovo svakom primeru skripta za pravljenje šeme koji postoji u ovoj knjizi.

### **Verzija 9.0**

U bazi podataka Oracle 9iR1 smo videli značajna unapređenja programskog jezika PL/SQL. U narednom spisku su rezimirana samo najbitnija unapređenja:

- Programske jezice SQL i PL/SQL sada koriste isti parser što doprinosi konzistentnosti. Pre ovog unapređenja uspešno izvršavanje iskaza u prozoru SQL\*Plus nije značilo uspešno izvršavanje u programskom jeziku PL/SQL.
- Semantika znakova, koja nam omogućava da definišemo veličinu, odnosno tačnost, promenljive ili kolone u znakovima, odnosno bajtovima, je ugrađena u verziju 9iR1. Svi Unicode znakovi nisu jednako napravljeni. Ovi znakovi se mogu razlikovati u broju bajtova koje zauzimaju. Preciznost se u bazi podataka Oracle zadaje u bajtovima, a ne u znakovima! Deklaracija promenljive koja je zadata sa VARCHAR2 (2) znači da promenljiva može da prihvati dva bajta, a ne dva znaka. Neki znakovi koji postoje u azijskim jezicima zauzimaju tri bajta, što znači da dodeljivanje jednog znaka kineskog pisma ne mora biti uspešno kada se taj znak dodeljuje promenljivoj koja može da prihvati dva bajta. E, to može da nervira!

- Podrška za rad sa objektima sada obuhvata nasleđivanje i evoluiranje tipova podataka. To su inače bili veoma uočljivi nedostaci u podršci za rad sa objektima u programskom jeziku PL/SQL.
- Ugrađen kompjajler sada omogućava da se programski kod napisan u programskom jeziku PL/SQL sada kompjajlira u programski kod napisan u programskom jeziku C (Oracle je napisan u programskom jeziku C), čime se smanjuje vreme njegovog izvršavanja, pošto se prilikom izvršavanja ne mora vršiti nikakva interpretacija.

### Verzija 9.2

Mnoge osobine baze podataka Oracle 9iR2 su zapravo poboljšanja unapređenja koja su se pojavila u verziji 9iR1. Osobine objekata su poboljšane, pri čemu su ugrađene funkcije i podrška za konstruktore koje prave korisnici. U Oracle Text je ugrađen paket CTXXPATH koji u programskom jeziku PL/SQL olakšava rad sa XML dokumentima koji su zapisani u XMLTYPE obliku.

### Verzija 10.0

U programske jezik PL/SQL verzije 10.0 su ugrađene brojne nove osobine:

- Najvažnija novina u programskom jeziku PL/SQL verzije 10gR1 jeste mogućnost korišćenja regularnih izraza. Regularni izrazi su dugo vremena bili zaštitni znak operativnog sistema Unix i skripta Perl, ali se sada mogu koristiti i u bazi podataka Oracle i programskom jeziku PL/SQL. Kratka definicija regularnog izraza je: Regularni izraz pronalazi, izdvaja i manipuliše uzorkom teksta.
- Druga sjajna osobina koja je ugrađena u verziju 10gR1 jeste mogućnost dobijanja upozorenja prilikom kompjajliranja programskog koda. Ne mislim na dobijanje poruka o greškama - to u programskom jeziku PL/SQL odavno postoji. Sada, koristeći parametar `plsql_warnings` ili paket DBMS\_WARNING, možemo da dobijemo upozorenja. Ova upozorenja nam nagovještavaju moguće probleme koji će uticati na performanse i na manje probleme koji prilikom kompjajliranja ne prouzrokuju prijavljivanje grešaka.
- Ugrađeni su novi tipovi podataka - tipovi podataka BINRY\_FLOAT i BINARY\_DOUBLE. To su tipovi podataka za rad sa brojevima u pokretnom zarezu koji predstavljaju alternativu za tip podataka NUMBER.
- Paket DBMS\_LOB omogućava rad sa velikim LOB-ovima - veličine između 8 i 128 terabajta (u zavisnosti od veličine bloka). Za više informacija pogledajte Poglavlje 16.
- Omogućeno je prilagođavanje stringovnih literalova. Ukoliko vam je dozlogrdilo da unutar stringovnog literala zapisujete dva apostrofa, sada možete koristiti g '!...!' (string se zapisuje između znakova uvezika). Na ovaj način u stringu ćete koristiti samo jedan umesto dva neophodna apostrofa. Evo jednostavnog primera neimenovanog bloka:

```
SET SERVEROUTPUT ON
BEGIN
    DBMS_OUTPUT.PUT_LINE('Ron''s');
END;
/
```

Prethodni programski kod prouzrokuje sledeću grešku

```
ORA-01756: quoted string not properly terminated
```

Da bi programski kod mogao pravilno da se izvrši, mi umesto jednog apostrofa moramo upotrebiti dva, pa bi programski kod izgledao ovako: 'Ron"s'. U verziji 10gR1 postoji alternativa za zapisivanje stringovnog literalna:

```
BEGIN
    DBMS_OUTPUT.PUT_LINE(q'!Ron's!');
END;
/
```

Prethodni programski kod se kompajlira bez ikakvih problema i prikazuje Ron's što nam je i bila namera.

## Osnove programskog jezika

U ovom odeljku ćemo se pozabaviti osnovnim karakteristikama programskog jezika PL/SQL. Neke od tih karakteristika su mogućnost izvršavanje programskog koda koji ne mora biti zapisan, zapisivanje programskog koda kako bi kasnije mogao da se upotrebi i razlike između raznih tipova uskladištenih objekata. Razmatraćemo ih na višem nivou kako bismo mogli da vas upoznamo sa konceptima koji postoje u programskom jeziku. Sve ovo ćemo detaljnije objasniti u Poglavlјima 3, 4, 8 i 9.

### Neimenovani blokovi

Neimenovani blokovi (engl. anonymous block) se trajno ne zapisuju i nemaju ime. Neimenovani blokovi se izvršavaju tokom postojanja sesije i ne mogu se pozivati iz druge sesije. Da biste isti programski kod ponovo izvršili, neimenovani blok morate zapisati u OS datoteku i pokrenuti ga, ponovo uneti programski kod ili ga uvrstiti u program koji će programski blok izvršavati po potrebi.

Videte da smo mi u primerima veoma često koristili neimenovane blokove programskog koda. Neimenovani blokovi su savršeni za pravljenje skript ili za obavljanje poslova koje se često ne obavljaju. Sljedeći primer je primer neimenovanog bloka programskog koda:

```
SET SERVEROUTPUT ON
DECLARE
    v_Date TIMESTAMP;
BEGIN
    SELECT systimestamp - 1/24
    INTO v_Date
    FROM dual;
    DBMS_OUTPUT.PUT_LINE('One hour ago: '||v_Date);
END;
/
```

Programski blok počinje rezervisanom rečju DECLARE ili rezervisanom rečju BEGIN i nigde se ne zapisuje nakon izvršavanja.

**NAPOMENA**

Programski blok napisan u jeziku PL/SQL je kompeltan odeljak programskog koda u jeziku PL/SQL. Program napisan u programskom jeziku PL/SQL se sastoji od jednog ili više programskih blokova koji logički dele posao koji treba obaviti. Programske blokove mogu biti ugnježdeni unutar drugih programske blokova. U Poglavlju 3 možete pročitati kompletno objašnjenje strukture programskog bloka.

## Procedure

Procedurama se zadaje ime i one se zapisuju. Procedure nakon izvršavanja mogu, ali i ne moraju, kao rezultat dati *vrednost*. Jedino što se nakon izvršavanja procedure mora dati kao rezultat jeste obaveštenje o njenom uspešnom ili neuspešnom izvršavanju.

Uskladištenim procedurama, odnosno imenovanim procedurama, se prilikom njihovog pravljenja zadaje jedinstveno ime. Korisnik koji ih je napravio je ujedno i njihov vlasnik, izuzev ukoliko se u skriptu za pravljenje procedure ne zada neki drugi vlasnik.

Procedure možete izvršavati sa SQL\*Plus prompta, iz SQL skripta ili iz programskog bloka PL/SQL programskog koda.

## Funkcije

Funkcije se razlikuju od procedura jer *moraju* kao rezultat proizvesti neku vrednost. Struktura funkcija je veoma slična strukturi procedura. Najveća razlika je to što u funkciji mora da postoji komanda RETURN. Funkcijama se zadaje ime i mogu se izvršavati sa SQL\*Plus prompta, iz SQL skripta ili iz programskog bloka PL/SQL programskog koda. Međutim, prilikom izvršavanja funkcije mora postojati način obrade vrednosti koju funkcija daje kao rezultat svog izvršavanja.

## Paketi

*Paketi* su logički grupisane procedure i funkcije. Paketi se sastoje iz dva dela: specifikacije paketa i tela paketa.

*Specifikacija paketa*, odnosno *spec*, je javni deo paketa i pokazuje njegovu strukturu. Kada se u SQL\*Plusu prikazuje opis paketa, prikazuje se zapravo njegova specifikacija. Specifikacija se uvek pravi, odnosno kompajlira, pre tela paketa. Zapravo, moguće je napraviti specifikaciju paketa, a da se nikada ne napravi telo paketa.

## Objektni tipovi podataka

Oracleovi objektni tipovi podataka vam omogućavaju da napišete objektno-orientisani programski kod koristeći programski jezik PL/SQL. Objektni tipovi podataka su po svojoj strukturi veoma slični paketima i takođe imaju specifikaciju i telo. Objektni tipovi obezbeđuju nivo apstrakcije za strukture podataka na osnovu kojih su izgrađeni.

Objektni tipovi podataka mogu sadržati atribute i metode. *Atributi* su zapravo karakteristike objekta. Knjiga, na primer, može imati atribute kao što su naslov, broj stranica i tako dalje.

*Metodi* koriste strukture podataka koje postoje u objektu. Sva interakcija između aplikacije i podataka objekta treba da se obavlja pomoću metoda.

Neke od prednosti korišćenja objektnih tipova podataka su

- **Apstrakcija** Programer koji pravi aplikaciju više ne mora da koristi relacione strukture podataka i može da koristi strukture iz svog okruženja.
- **Konzistentnost** Ukoliko se sva interakcija aplikacije obavlja pomoću objekata umesto da se direktno koriste strukture podataka, onda je mogućnost narušavanja objekata daleko manja.
- **Jednostavnost** Umesto da se model iz okruženja pretvara u model koji se predstavlja programskim kodom, model ostaje u svom prirodnom obliku. Ukoliko je potrebno da nešto saznam o objektu kojim je predstavljena knjiga, ja ću koristiti objekat *knjiga*.

Osobine koje se ugrađuju u bazu podataka Oracle još od verzije 9iR1 su nasleđivanje, dinamičko otpremanje metoda i evoluiranje tipova podataka. Ove osobine čine objektno-orientisano programiranje u programskom jeziku PL/SQL mnogo robusnijim.

## Obrada iskaza u programskom jeziku PL/SQL

Kada se izvršava programski blok napisan u programskom jeziku PL/SQL, onda se programski kod prosleđuje PL/SQL mehanizmu. Mehanizam se može nalaziti na samom serveru podataka ili u nekom od alata (recimo, u alatu Oracle Reports) koji su deo PL/SQL mehanizma. Programski kod se zatim raščlanjuje, a SQL programski kod se na izvršavanje prosleđuje SQL parseru ili SQL Statement Executoru. Proceduralni iskazi se prosleđuju Procedural Statement Executoru na izvršavanje.

### Režim interpretiranja programskog koda

Režim *interpretiranja* programskog koda (engl. interpreted mode) je osnovni režim za bazu podataka Oracle. To znači da se uskladištene procedure, funkcije i paketi kompajliraju i zapisuju kao PL/SQL programski kod i da ih baza podataka Oracle (koja je napisana u programskom jeziku C) interpretira prilikom izvršavanja. Kompajliranje PL/SQL programskog koda se brže izvršava u režimu interpretacije, ali izvršavanje programskog koda može biti sporije nego da je kompajliran osnovnim načinom kompajliranja.

### Osnovni način kompajliranja

Osnovni način kompajliranja (engl. native compilation), koji je prvi put uveden u bazu podataka Oracle verzije 9iR1, a unapređen je u verziji 10gR1, prevodi, prilikom kompajliranja, programski kod napisan u jeziku PL/SQL u programski kod napisan u jeziku C. Zbog toga izvršavanje programskog koda može i do 30 procenata biti brže pošto prilikom izvršavanja nije neophodno obaviti interpretiranje.

## Kako najbolje upotrebiti knjigu

Ova knjiga je potpuno prerađena i obuhvata informacije namenjene početnicima, programerima koji poznaju Oracle i iskunim programerima. Primeri programskog koda služe da demonstriraju osobine programskog jezika PL/SQL. Programski kod primera možete slobodno preuzeti sa mreže. Na web sajtu se nalaze direktorijumi za svako poglavlje. U tim direktorijumima se nalazi programski kod koji se koristi u knjizi. Programski kod svakog poglavlja je napisan tako da se može koristiti nezavisno od programskog koda koji se nalazi u ostalim poglavljima - ne postoji zavisnost između programskog koda više poglavlja. Obuhvaćeni su skriptovi za pravljenje šema kako bi se olakšalo testiranje. Skriptove bi trebalo da izmenite kako bi odgovarali okruženju u kojem radite i bazi podataka sa kojom radite.

Za neke teme je potrebno odvojiti više prostora nego što smo mi mogli da odvojimo u ovoj knjizi. Umesto da skratimo objašnjenja kako bi sve što smo želeli da objasnimo stalo u ovu knjigu, mi smo napravili dodatni materijal koji možete preuzeti i koji predstavlja detaljnije objašnjenje tema koje se obrađuju u poglavlju. Nadamo se da će vam ovakva dodatna objašnjenja biti od koristi.

## Ciljna grupa

Ova knjiga je napisana za početnike i iskusne programere koji koriste programski jezik PL/SQL, kao i za administratore baza podataka (DBA), koji žele da iskoriste sve prednosti koje nudi programski jezik PL/SQL. Poglavlja u kojima se obrađuju napredne teme (11-17) zahtevaju potpuno razumevanje sadržaja Poglavlja 1-10. Neće biti na odmet da prelistate prvih nekoliko poglavlja čak i ako ste iskusan PL/SQL programer. U tim poglavljima se nalazi objašnjenje novih osobina i primeri programskog koda koji vam mogu dati ideje koje ćete upotrebiti u aplikacijama koje pravite.

Bez obzira na to koliko ste iskusni, mi smo sigurni da ćete u svakom poglavlju pronaći ponešto što ranije niste otkrili.

## Namena knjige

PL/SQL je razvijen, robustan programski jezik koji je sve bolji sa svakom novom verzijom. Kako složenost raste, to držanje koraka sa novim osobinama postaje sve teže. Mi želimo da vam pomognemo da

- Naučite programski jezik PL/SQL, ukoliko ga ranije niste koristili.
- Usvojite i izgradite dobar i efikasan oblik programskog koda.
- Razumete osobine koje se u drugim knjigama malo i uopšte ne objašnjavaju.
- Otkrijete kolike su mogućnosti ovog programskog jezika!

## Šta je sve obuhvaćeno u knjizi

U svakoj knjizi postoje ograničenja. U našoj knjizi ograničenja su:

- Obradili smo samo jedan deo tema koje su korisne za administratore baze podataka. Ukoliko želite da naučite više o administriranju baze podataka onda treba da posetite adresu <http://otn.oracle.com> ili da pročitate neku od izvrsnih knjiga na tu temu koje je izdao Oracle Press.
- Opis podešavanja performansi je prilično ograničen kako bismo vam pomogli da efikasno koristite PL/SQL. Nismo obuhvatili podešavanje performansi baze podataka.
- Mi vam možemo pružiti samo informacije, savete i primere. Na vama je da ih efikasno iskoristite i napišete dobar programski kod.

## Pretpostavke

Osnovna verzija baze podataka koju smo koristili prilikom pisanja knjige je Oracle 8.1.7.4, terminalska verzija baze podataka Oracle 8i. U knjizi smo obuhvatili verzije 8.1.7, 9iR1, 9iR2 i 10gR1. Da biste u potpunosti iskoristili knjigu, i nove osobine baze podataka Oracle, mi vam savetujemo da sa sajta OTN-a (<http://otn.oracle.com>) preuzmete i potom instalirate Oracle 10gR1. Server podataka možete besplatno preuzeti ukoliko se registrujete (registrovanje je takođe besplatno).

### SAVET

Verzija 10gR1 se instalira sa jednog diska, pa je stoga i preuzimanje mnogo brže nego za bilo koju verziju 9i.

Kao apsolutni minimum mi preporučujemo da imate pristup jednoj Oracleovoj instanci i da imate neophodne dozvole za pravljenje korisničkog naloga i potrebnih objekata. Veoma je važno da znate koju verziju programskog jezika PL/SQL imate kako biste znali koje osobine možete da koristite.

Od baze podataka Oracle 8, oznake verzija programskog jezika PL/SQL se poklapaju sa verzijom baze podataka. U tekstu u kojem se objašnjavaju verzije pre baze podataka Oracle verzije 8 ćete uočiti oznake PL/SQL 1.1, 2x i slično. Da biste saznali koju verziju posedujete postavite upit nad pogledom V\$VERSION.

```
SELECT banner
      FROM v$version;
```

Prethodni upit je dao sledeći rezultat u okruženju u kojem radim:

```
BANNER
-----
Oracle Database 10g Enterprise Edition Release 10.1.0.2.0 - Prod
PL/SQL Release 10.1.0.2.0 - Production
CORE    10.1.0.2.0      Production
TNS for 32-bit Windows: Version 10.1.0.2.0 - Production
NLSRTL Version 10.1.0.2.0 - Production
```

Dakle, ja koristim bazu podataka čija je verzija 10.1.0.2.0 i programski jezik PL/SQL verziju 10.1.0.2.

## Pravila

U knjizi se koriste različiti oblici slova (font) kako bi se istakao i razlikovao tekst. Primeri programskog koda, i spoljašnje reference na objekte baze podataka u tekstu, su prikazane tekstom koji je štampan u fontu COURIER. Reference na promenljive su takođe stampane u fontu COURIER. Elementi u primerima programskog koda koji su od posebne važnosti su štampani **masnim** slovima. Posebnu pažnju treba da obratite na delove knjige koji su obleženi sa Napomena i Savet.

## Primeri

Skriptovi za pravljenje korisničkih nalogu postoje u svakom poglavlju i služe da bi se dale dozvole koje su neophodne u primerima svakog poglavlja pojedinačno. Nemojte koristiti skript za pravljenje šeme iz jednog poglavlja za primere iz drugog poglavlja.

U većini poglavlja se koriste primeri objekata koji se odnose na prodavnici knjiga. Osnovne tabele za većinu primera su tabele AUTHORS i BOOKS, kao što ste videli ranije u ovom poglavlju, na slici 1.1. Postoje male razlike u strukturi šeme između poglavlja kako bismo mogli da vam pokažemo različite osobine.

Tabela BOOKS ima sledeću strukturu:

DESC books		
Name	Null?	Type
ISBN	NOT NULL	CHAR(10)
CATEGORY		VARCHAR2(20)
TITLE		VARCHAR2(100)
NUM_PAGES		NUMBER
PRICE		NUMBER
COPYRIGHT		NUMBER(4)
AUTHOR1		NUMBER
AUTHOR2		NUMBER
AUTHOR3		NUMBER

Tabela AUTHORS ima sledeću strukturu:

DESC authors		
Name	Null?	Type
ID	NOT NULL	NUMBER
FIRST_NAME		VARCHAR2(50)
LAST_NAME		VARCHAR2(50)

Skript za pravljenje šeme u Poglavlju 16, skript `CreateLOBUser.sql`, pravi dva prostora za tabele koji se koriste za zapisivanje parametra iskaza `create table`. Imena prostora za tabele i imena datoteka sa podacima se mogu po potrebi izmeniti u okruženju u kojem radite.

Najzad, obratite pažnju na različite metode koji se koriste za pravljenje šema i primera. Mi smo se potrudili da u knjizi upotrebimo različite tehnike kako bismo vam pokazali da postoji više načina na koje se može obaviti isti posao. Dok proučavate primere, razmišljajte kako, pri pravljenju sopstvenih aplikacija, možete upotrebiti neke od tih tehnika, pravila imenovanja i strategija. Genijalnost, sposobnost komuniciranja, stav, disciplina i znanje će pospešiti bolje korišćenje Oracleovog programskog jezika PL/SQL i pomoći će vam da bolje obavite bilo koji posao koji se pred vama nađe.

## Zaključak

U ovom poglavlju smo vam predstavili koncepte programiranja, opisali smo kako se programski jezik PL/SQL uklapa u objektno-orientisane i proceduralne kategorije programiranja i ukratko smo vas upoznali sa nekim osobinama programskog jezika PL/SQL koje su objašnjene u ovoj knjizi. Ukratko smo opisali osnove relacionih baza podataka i programskog jezika SQL. Pored toga, uporedili smo programski jezik PL/SQL sa programskim jezicima SQL i Java. Najzad, objasnili smo kako je koncipirana ova knjiga i kako da je najbolje iskoristite.

Nadamo se da će vam knjiga biti od koristi i da ćete otkriti stvari za koje niste ni sanjali da su moguće u programskom jeziku PL/SQL.

