Rad sa tabelama

Rad sa tabelama

Podaci u relacionoj bazi podataka su organizovani u vidu međusobno povezanih tabela.

ERD → Relaciona BP

entitet → tabela (relacija)

atribut → kolona u tabeli

instanca entiteta → red (vrsta) u tabeli

vrednost atributa → sadržaj ćelije u tabeli (podatak)

Svojstva tabela

- Svako polje u tabeli ima jedinstvenu vrednost (prva normalna forma);
- Sadržaj svih polja jedne kolone pripada istom tipu podataka (integritet kolona);
- Svaka vrsta je jedinstvena (ne postoje dve vrste u tabeli sa istim sadržajem u svim poljima);
- Redosled kolona nije bitan;
- Redosled vrsta nije bitan;
- Svaka kolona ima jedinstven naziv.

Svojstva tabela

Svaka tabela koja ima navedena svojstva predstavlja jednu relaciju podataka.

Veze među tabelama se ostvaruju pomoću stranog ključa (koristi se i termin spoljni ključ).

Sistem za upravljanje bazama podataka

- Sistem za upravljanje bazama podataka – **SUBP** (Database-Management System – **DBMS**) je softverski sistem za kreiranje, održavanje i korišćenje baza podataka.
- SUBP funkcioniše uz pomoć **rečnika podataka** i **jezika baze podataka**.

Rečnik podataka

- **Rečnik podataka** je baza podataka koja sadrži **metapodatke.**
- Metapodaci su podaci o podacima, uključujući opis podataka, podatke o vlasništvu nad podacima, pristupne puteve, prava pristupa i prolaznost (trajnost) podataka.
- Rečnik podataka je dakle metabaza, odnosno baza podataka o bazi podataka.
- Svaka baza podataka sadrži svoj rečnik podataka.

Rečnik podataka

Tabele u rečniku podataka sadrže informacije o:

- Korisnicima i njihovim privilegijama;
- Tabelama, kolonama i njihovim tipovima podataka;
- Dodeljenim privilegijama nad objektima baze podataka;
- Memorijskim strukturama baze podatka, itd.

Jezici za rad sa bazom podataka

Jezici za rad sa bazom podataka su:

- **DDL** (Data Definition Language) – jezik za kreiranje, menjanje i brisanje objekata baze podataka (tabele, kolone, ograničenja,...);
- **DML** (Data Manipulation Language) – jezik za pretraživanje, dodavanje, ažuriranje i brisanje podataka;
- **DCL** (Data Control Language) – jezik za kontrolu pristupa podacima;
- **TCL** (Transaction Control Language) – jezik za upravljanje transakcijama.

TRANSAKCIJA: Logička jedinica posla koja se sastoji iz jedne ili više operacija nad bazom podataka.

Strukturirani upitni jezik - SQL

- SQL je jezik koji se koristi za manipulaciju bazom podataka.
- Uz pomoć SQL-a mogu se dodavati, brisati, menjati i pretraživati podaci i tabele.
- SQL pokriva funkcije jezika DDL, DML, DCL i TCL.

Strukturirani upitni jezik - SQL

- Strukturirani jezik za upite (Structured Query Language - SQL) je razvio IBM.
- Kasnije je jezik prihvaćen i od drugih većih firmi koje su razvijale i prodavale sisteme za upravljanje bazama podataka (npr. Oracle).
- Vremenom su razni sistemi za upravljanje bazama podataka uvodili svoje specifične varijante SQL-a.

Funkcije SQL

- **DDL** (Data Definition Language) :
  CREATE, ALTER, DROP, RENAME,...
- **DML** (Data Manipulation Language) :
  SELECT, INSERT, UPDATE, DELETE, TRUNCATE,...
- **DCL** (Data Control Language):
  GRANT, REVOKE
- **TCL** (Transaction Control Language):
  COMMIT, ROLLBACK, SAVEPOINT

Naredba DESCRIBE

Naredba DESCRIBE opisuje strukturu tabele.

Sintaksa je:

DESCRIBE table name;

Ova naredba se izvršava nad rečnikom podataka.

Primeri:

DESCRIBE employees;
DESCRIBE d_clients;
DESCRIBE d_songs;

Naredba SELECT *

Naredba SELECT * vraća (prikazuje) sve podatke iz svih redova u tabeli.

Sintaksa je:

SELECT * FROM table name;

Ova naredba se izvršava nad određenom tabelom.

Primeri:

SELECT * FROM employees;

SELECT * FROM d_clients;

Naredba SELECT

Da bi se dobio podskup podataka, koristi se SELECT-rečenica čija je sintaksa:

```
SELECT column name 1, column name 2, ...
FROM table name
WHERE condition;
```

Primer:

```
SELECT id, salary
FROM employees
WHERE salary>1000;
```

Termini

Kroz kurs baza podataka koristiće se sledeći termini:

- Ključna reč
- Klauzula
- Rečenica (naredba).

Ključna reč

Ključna reč ukazuje na poseban SQL-iskaz.

Na primer, ključne reči su:

- SELECT
- DESCRIBE
- FROM
- WHERE

Klauzula

Klauzula je deo SQL-rečenice.

Na primer, klauzule su:

- **SELECT** id, salary
- **SELECT** *
- **FROM** employees
- **ORDER BY** salary

Rečenica

Rečenica (naredba) je jedna klauzula ili kombinacija dve ili više klauzula, kojom se izvršava neka operacija nad bazom podataka. Završava se znakom “;“.

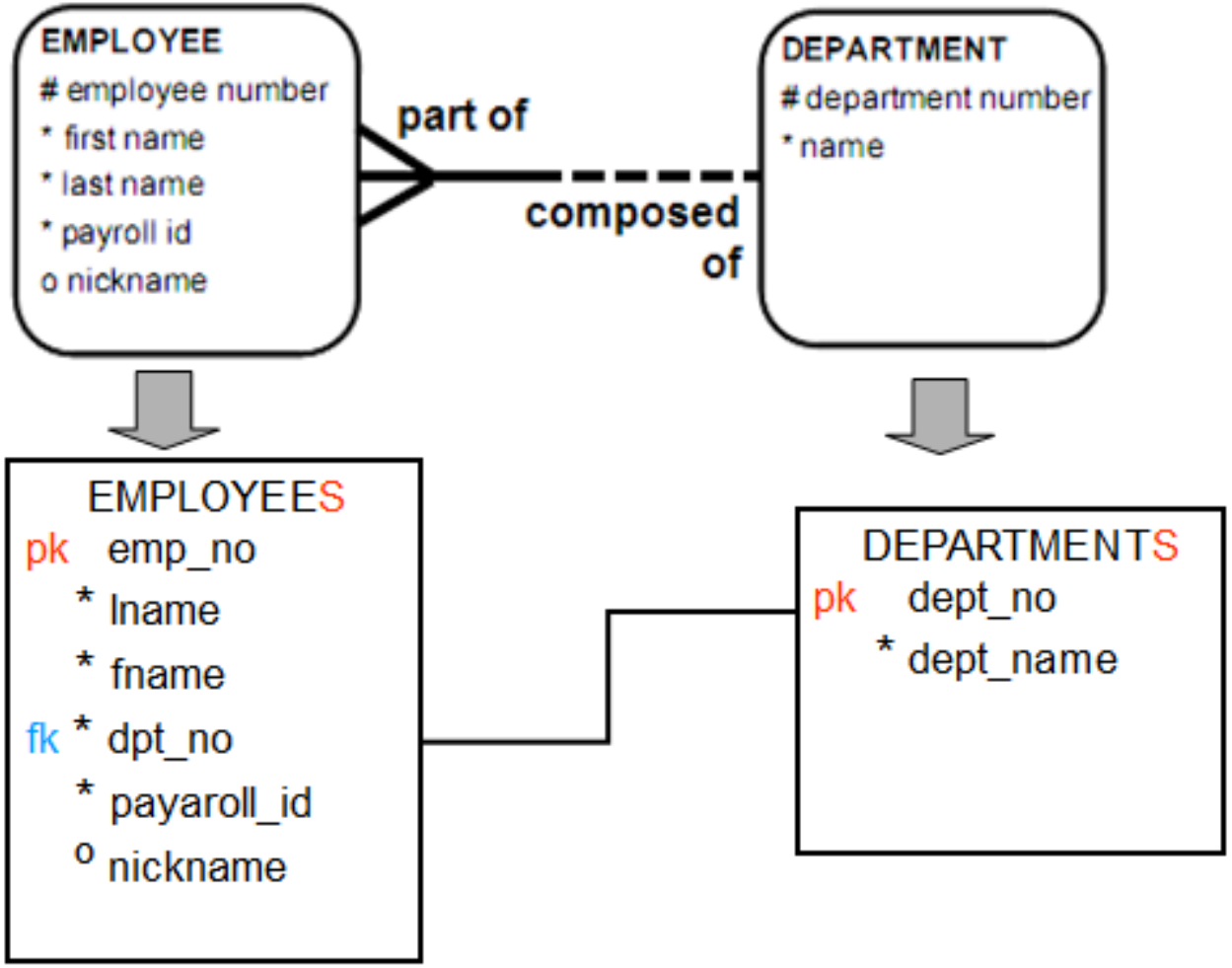
Na primer, rečenice sa jednom, dve ili više klauzula su:

- **DESCRIBE** d_songs**;**
- **SELECT** title **FROM** d_songs**;**
- **SELECT * FROM** employees**;**
- **SELECT** id, salary **FROM** employees **ORDER BY** salary**;**

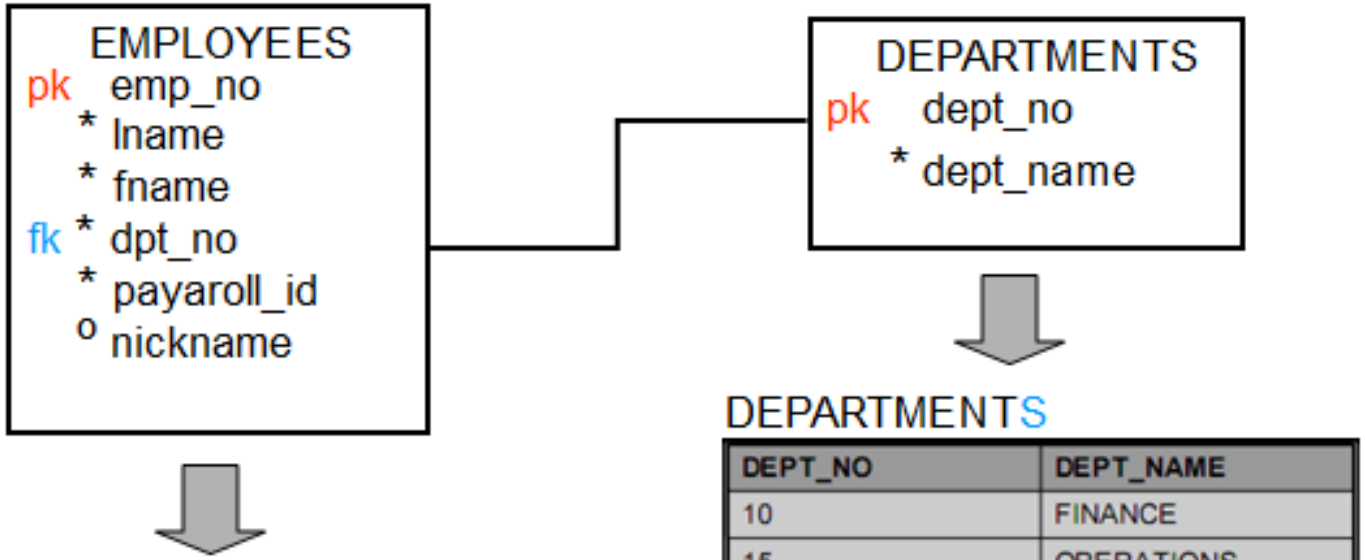
APEX

Oracle Application Express – APEX je alat za brz razvoj aplikacija sa Oracle-ovim bazama podataka.

Proces kreiranja baze podataka za neki poslovni sistem



Proces kreiranja baze podataka za neki poslovni sistem



DEPARTMENTS

DEPT_NO	DEPT_NAME
10	FINANCE
15	OPERATIONS
22	MANUFACTURING

EMPLOYEES

EMP_NO	LNAME	FNAME	DEPT_NO	PAYROLL_ID	NICK NAME
100	SMITH	DANA	10	21215	Dana
310	ADAMS	TYLER	15	59877	Ty
210	CHEN	LAWRENCE	10	1101	Larry
405	GOMEZ	CARLOS	10	52	Chaz
378	LOUNGANI	NEIL	22	90386	Neil

Prikaz strukture i sadržaja tabela

DESCRIBE departments;

Object Type **TABLE** Object **DEPARTMENTS**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default
DEPARTMENTS	DEPARTMENT_ID	Number	-	4	0	1	-	-
	DEPARTMENT_NAME	Varchar2	30	-	-	-	-	-

SELECT * FROM departments;

DEPARTMENT_ID	DEPARTMENT_NAME
10	Administration
20	Marketing
50	Shipping
60	IT
80	Sales

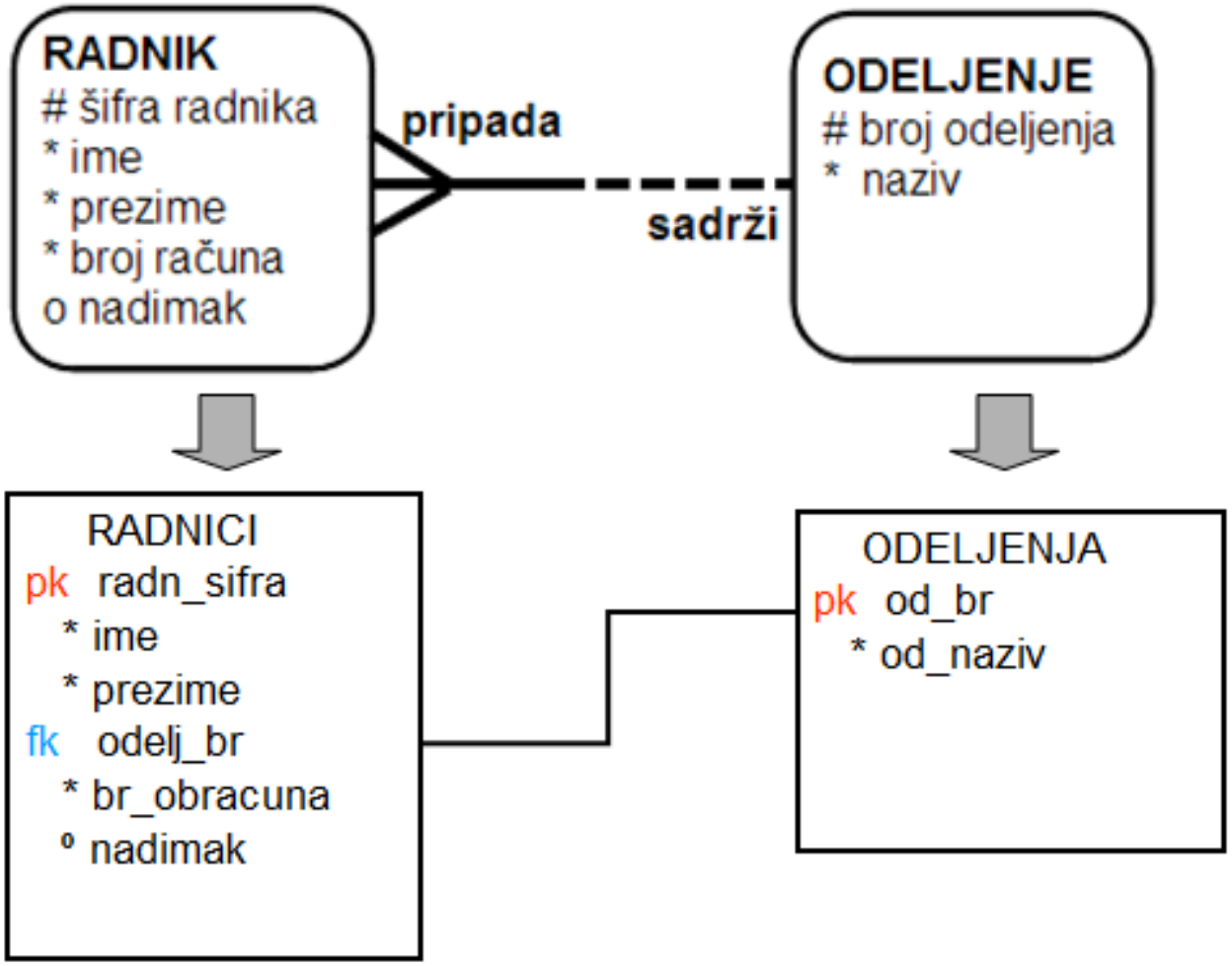
Kreiranje tabela

Za kreiranje tabela koristi se SQL rečenica čija je sintaksa:

```
CREATE TABLE table
(column1 datatype (…),
 column2 datatype (…),
 … (… ));
```

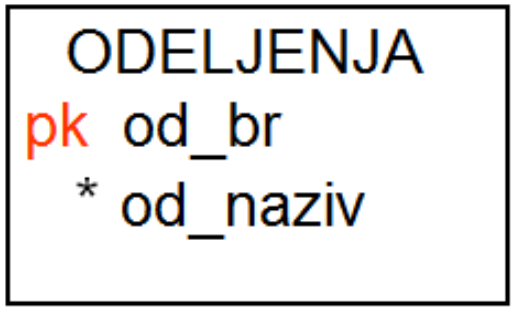
Kreiranje tabela

Primer:



Kreiranje tabela

```
CREATE TABLE odeljenja
(od_br NUMBER(4) PRIMARY KEY,
 od_naziv VARCHAR2(30) NOT NULL);
```




```
DESCRIBE odeljenja;
```

Object Type TABLE Object ODELJENJA

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable
ODELJENJA	OD_BR	Number	-	4	0	1	-
	OD_NAZIV	Varchar2	30	-	-	-	-

```
SELECT * FROM odeljenja;
```

no data found

Unos podataka u tabelu

Komanda INSERT dodaje red podataka u tabelu.

Eksplicitni INSERT

```
INSERT INTO table (column 1, column 2, …)
VALUES (value 1, value 2, …);
```

Primer:

```
INSERT INTO odeljenja (od_br, od_naziv)
VALUES (20, 'Marketing');
```

Unos podataka u tabelu

Implicitni INSERT

```
INSERT INTO table
VALUES (value 1, value 2, …);
```

Primer:

```
INSERT INTO odeljenja
VALUES (80, 'Prodaja');
```

```
SELECT * FROM odeljenja;
```

OD_BR	OD_NAZIV
20	Marketing
80	Prodaja

Kopiranje tabela

Tabela se može prekopirati pomoću sledeće naredbe:

```
CREATE TABLE ime_nove_tabele
AS (SELECT * FROM ime_tabele_koja_se_kopira);
```

Primer:

```
CREATE TABLE org_jedinice
AS (SELECT * FROM odeljenja);
```

SELECT * FROM org_jedinice;

OD_BR	OD_NAZIV
20	Marketing
80	Prodaja

SELECT * FROM odeljenja;

OD_BR	OD_NAZIV
20	Marketing
80	Prodaja

Kopiranje tabela

DESCRIBE org_jedinice;

Object Type TABLE Object ORG_JEDINICE

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable
SLUZBE	OD_BR	Number	-	4	0	-	
	OD_NAZIV	Varchar2	30	-	-	-	-

Kopiranjem tabele nisu prekopirane i sve karakteristike kolona (PK, NN, ...). To se rešava naredbom ALTER.

Modifikovanje tabela

ALTER TABLE se koristi za:

- modifikovanje postojeće kolone

```
ALTER TABLE org_jedinice
MODIFY (od_br NUMBER(5) PRIMARY KEY);
ALTER TABLE org_jedinice
MODIFY (od_naziv VARCHAR2(35) NOT NULL);
```

- dodavanje nove kolone

```
ALTER TABLE org_jedinice
ADD (sediste VARCHAR2(10));
```

- ukidanje kolone

```
ALTER TABLE org_jedinice
DROP COLUMN od_naziv;
```

Pražnjenje i brisanje tabele

- Naredba **TRUNCATE TABLE** koristi se za pražnjenje tabele, tj. briše njen sadržaj ali je ne ukida. Tabela ostaje u strukturi baze podataka i može se ponovo napuniti. Sintaksa je:

TRUNCATE TABLE tabela;

- Naredba **DROP TABLE** u potpunosti briše tabelu iz baze podataka, kako njen sadržaj, tako i njenu strukturu. Sintaksa je:

DROP TABLE tabela;

Ograničenja

Ograničenja (constraints) su objekti baze podataka koji su smešteni u tabeli rečnika podataka USER_CONSTRAINTS i mogu se prikazati naredbom:

```
SELECT *
FROM user_constraints;
```

Kod Oracle baza podataka postoji pet tipova ograničenja.

Svaki tip ograničenja obezbeđuje primenu različite vrste pravila.

Ograničenja

Tipovi ograničenja su:

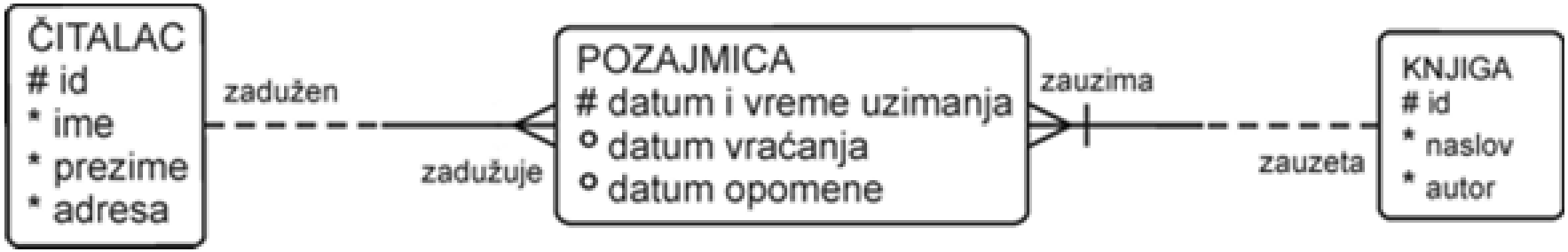
- NOT NULL ograničenje (ne dozvoljava da nijedno polje u koloni sa tim ograničenjem ostane nepopunjeno),
- UNIQUE ograničenje (jedinstvenost),
- PRIMARY KEY ograničenje (primarni ključ),
- FOREIGN KEY ograničenje (spoljni ključ),
- CHECK ograničenje (proverava da li uneta vrednost zadovoljava neki uslov).

Ograničenja

Osnovna pravila za ograničenja su:

- Ograničenje koje se odnosi na više od jedne kolone (složeni ključ) mora biti definisano na nivou tabele;
- Ograničenje NOT NULL se može specificirati samo na nivou kolone, ne na nivou tabele;
- Ograničenja UNIQUE, PRIMARY KEY, FOREIGN KEY i CHECK se mogu definisati bilo na nivou kolone bilo na nivu tabele;
- Ako se u rečenici CREATE TABLE koristi ključna reč CONSTRAINT, ograničenju se mora dati naziv.

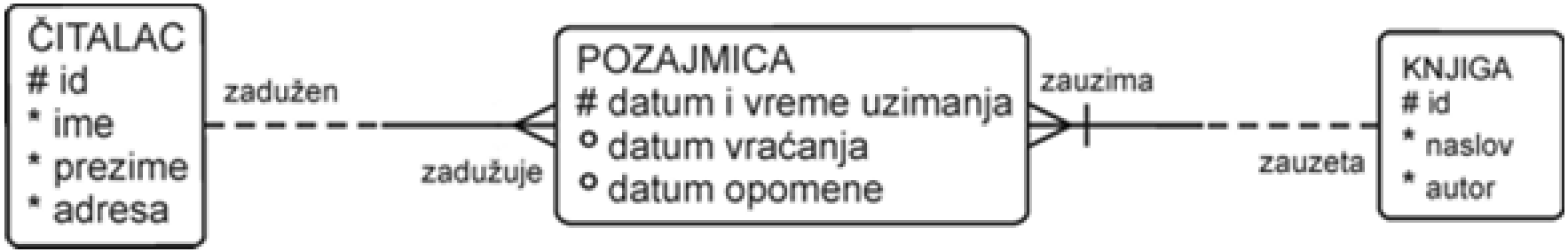
Primer:




```
CREATE TABLE citaoci
(id NUMBER(5) CONSTRAINT cit_id_pk PRIMARY KEY,
 ime VARCHAR2(15) CONSTRAINT cit_ime_nn NOT NULL,
 prezime VARCHAR2(15) NOT NULL,
 adresa VARCHAR2(50) NOT NULL);
```

```
CREATE TABLE knjige
(id NUMBER(6) CONSTRAINT knj_id_pk PRIMARY KEY,
 naslov VARCHAR2(50) NOT NULL,
 autor VARCHAR2(30) NOT NULL);
```

Primer:




```
CREATE TABLE pozajmice
(dat_u DATE,
 dat_v DATE,
 dat_o DATE,
 cit_id NUMBER(5),
 knj_id NUMBER(6),
 CONSTRAINT poz_cit_id_fk FOREIGN KEY(cit_id)
REFERENCES citaoci(id),
 CONSTRAINT poz_knj_id_fk FOREIGN KEY(knj_id)
REFERENCES knjige(id),
 CONSTRAINT pozajmice_pk PRIMARY KEY(knj_id,dat_u));
```

Primer:

Podaci za unos u tabele:

“Čitalac Marko Simić iz Čačka, Kuželjeva 8, id čitaoca = 2566, uzeo je dana 20.11.2007. godine na čitanje knjigu „Dorotej“ Dobrila Nenadića, id knjige = 1222, a vratio je 13.12.2007. i istog dana pozajmio knjigu „Čarobni breg“ Tomasa Mana, id knjige = 2233, koju je vratio 27.12.2007. Knjigu „Dorotej“ je, istog dana kad ju je vratio Marko Simić, uzela na čitanje Slavka Jović iz Čačka, Karađorđeva 15, id čitaoca = 1893, i još je nije vratila. Stevo Slović iz Ljubića, Avalska 3, id čitaoca = 1002, je pozajmio knjigu „Čarobni breg“, id knjige = 2233, dana 8.1.2008. godine i još je nije vratio.“

Primer:

Unos podataka u tabelu:

```
INSERT INTO citaoci (id, ime, prezime, adresa)
VALUES (2566, 'Marko', 'Simić', 'Kuželjeva 8');

INSERT INTO knjige (id, naslov, autor)
VALUES (1222, 'Dorotej', 'Dobrilo Nenadić');

INSERT INTO pozajmice (dat_u, dat_v, dat_o, cit_id, knj_id)
VALUES ('20-NOV-2007', '13-DEC-2007', '', 2566, 1222);
```

Primer:

Unos podataka u tabelu:

```
INSERT INTO pozajmice (dat_u, dat_v, dat_o, cit_id, knj_id)
VALUES ('13-DEC-2007', '27-DEC-2007', '', 2566, 2233);
```

```
ORA-02291:
integrity constraint (RS_TF2_SQL01_T01.POZ_KNJ_ID_FK)
violated - parent key not found
```

Narušen je referencijalni integritet. Nije pronađen roditeljski ključ (odgovarajuća vrednost u koloni druge tabele referenciranoj spoljnim ključem).

Pokušan je unos vrednosti knj_id=2233 u tabelu POZAJMICE, a da pre toga u tabelu KNJIGE nisu uneti podaci o knjizi čiji je id=2233.

Primer:

Nakon dodavanja knjige „Čarobni breg“ čiji je id=2233 u tabelu KNJIGE, moguće je dodati podatke o pozajmljivanju te knjige u tabelu POZAJMICE.

```
INSERT INTO knjige (id, naslov, autor)
VALUES (2233, 'Čarobni breg', 'Tomas Man');

INSERT INTO pozajmice (dat_u, dat_v, dat_o, cit_id, knj_id)
VALUES ('13-DEC-2007', '27-DEC-2007', '', 2566, 2233);
```

Primer:

Nastavak unošenja podataka:

```
INSERT INTO citaoci
VALUES (1893, 'Slavka', 'Jović', 'Karađorđeva 15');
```

Za čitaoca Slavku Jović nije uneta kompletna adresa u tabeli ČITAOCI, pa je potrebno korigovati pomenutu adresu.

```
UPDATE citaoci
SET adresa= 'Karađorđeva 15, Čačak'
WHERE id=1893;
```

Primer:

Nastavak unošenja podataka:

```
INSERT INTO pozajmice
VALUES ('13-DEC-2007', '', '', 1893, 1222);

INSERT INTO citaoci
VALUES (1002, 'Stevo', 'Slović', 'Avalska 3, Ljubić');

INSERT INTO pozajmice
VALUES ('08-JAN-2008', '', '', 1002, 2233);
```

Primer:

Ukoliko korisnik pokuša da obriše red podataka iz tabele ČITAOCI, dobiće obaveštenje o narušavanju referencijalnog integriteta.

```
DELETE FROM citaoci
WHERE id=1893;
```

```
ORA-02292: integrity constraint
(RS_TF2_SQL01_T01.POZ_CIT_ID_FK)
violated - child record found
```

Referencijalni integritet je narušen jer vrsta u jednoj tabeli (tabela POZAJMICE) ostaje bez vrste koju referencira u “roditeljskoj“ tabeli (tabela ČITAOCI) - “Dete ostaje bez roditelja”.

Primer:

Da bi se iz tabele ČITAOCI obrisala Slavka Jović čiji je id=1893, potrebno je prvo obrisati red u tabeli POZAJMICE koji se odnosi na njeno pozajmljivanje knjige.

```
DELETE FROM pozajmice
WHERE cit_id=1893;

DELETE FROM citaoci
WHERE id=1893;
```