

Informacije o predmetu

Predmet: Administriranje Baze Podataka
Predavač: dr Dušan Stefanović

Студијски програм: СРТ и КОТ

Статус предмета: Изборни

Семестар: VI

Број часова: 2 + 1 + 1

Број ЕСПБ: 5

CILJ PREDMETA

Projektovanje

Implementacija

Programiranje

Administriranje Baze Podataka

SADRŽAJ PREDMETA

1	Projektovanje baze podataka, EER, referencijalni integritet i ograničenja	<i>Modelovanje i preslikavanje podataka u relacioni model SQL upiti</i>
2	Programiranje i automatizacija RDBM baze podataka	<i>Korisničke i ugrađene funkcije Procedure Trigeri</i>
3	Transakcije i Pogledi (eng. Views)	<i>Konkurentni pristup i osobine transakcije (ACID) Osobine pogleda i prednosti korišćenja pogleda</i>
4	Tehnike za optimizaciju baze podataka	<i>Indeksi Optimizacija upita</i>
5	Kreiranje korisnika i dodela privilegija	<i>Upravljanje korisničkim nalogima i definisanje nivoa pristupa DBMS-u</i>
6	Replikacija, Backup & Oporavak	<i>Primena replikacije i metode za backup i oporavak podataka</i>
7	Bezbednost baze podataka	<i>Primeri dobre prakse u zaštiti baze podataka Posledice neadekvatne zaštite baze podataka</i>

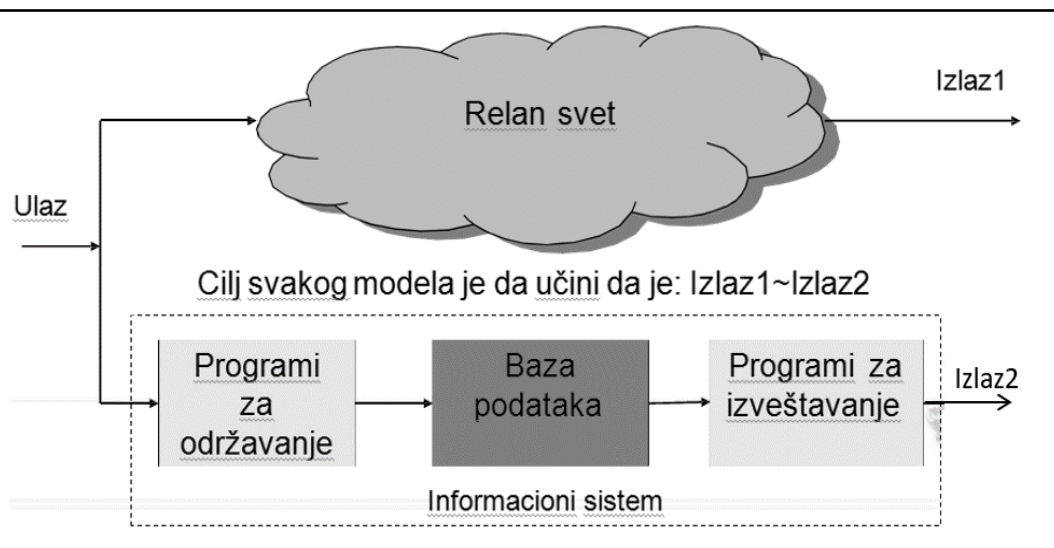
[FINALNI PROJEKAT]

PREDUSLOVI ZA SLUŠANJE PREDMETA

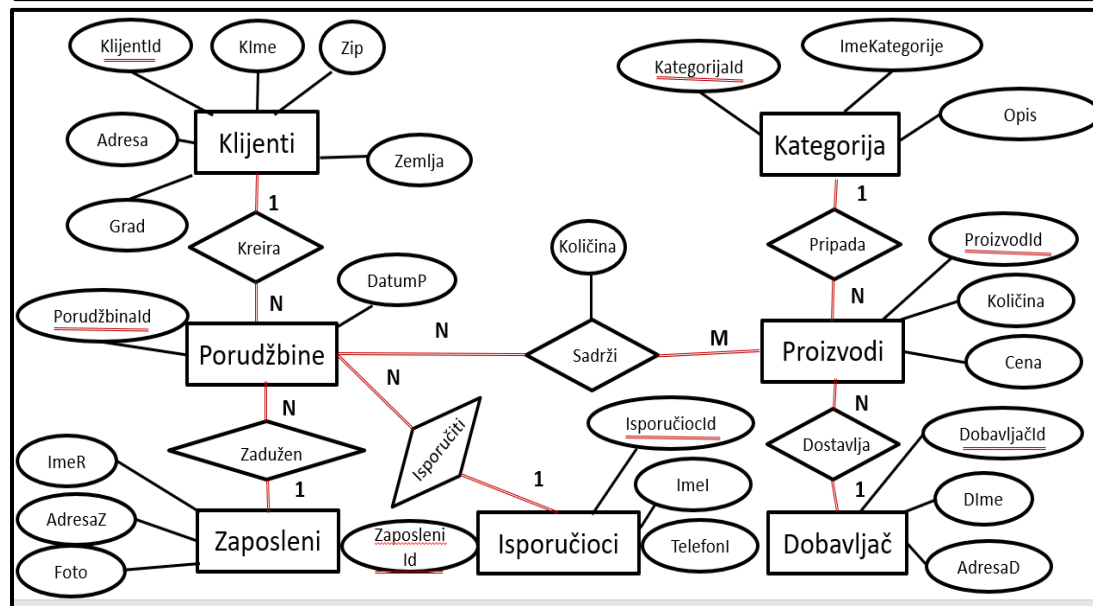
 Potrebno je da student bude upoznat sa konceptom dizajna relacione baze podataka (**NORMALIZACIJA**) i sa kreiranjem **EER DIJAGRAMA**

 Potrebno je da student bude upoznat sa pisanjem SQL iskaza Data Definition i Data Manipulation:
CREATE, ALTER, INSERT, UPDATE, & DELETE

Projektovanje - Modelovanje

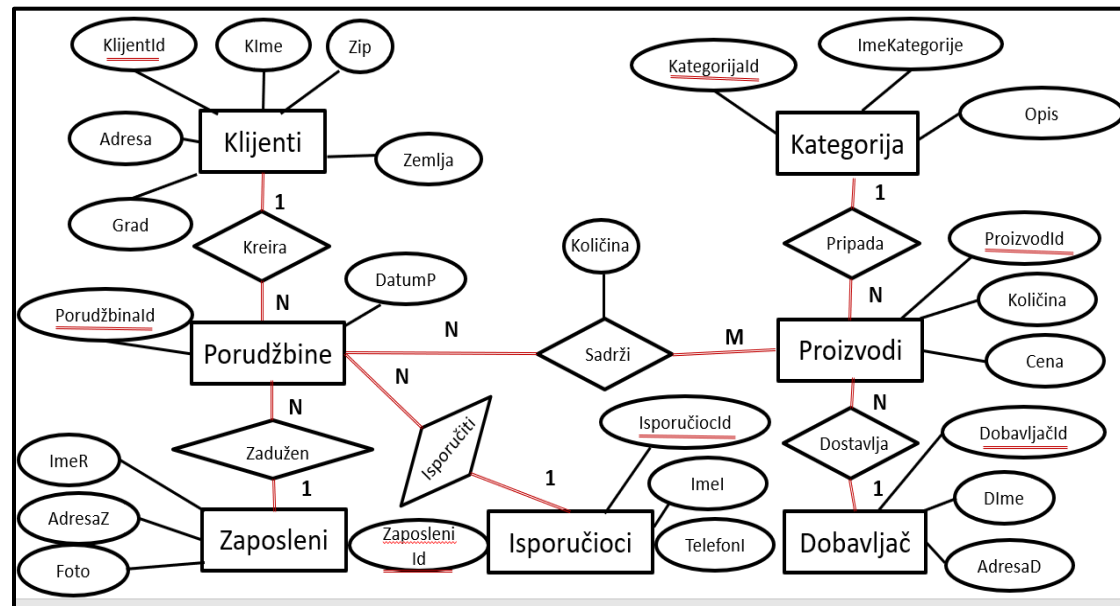


Modelovanje - realni svet preslikava na određeni broj podataka

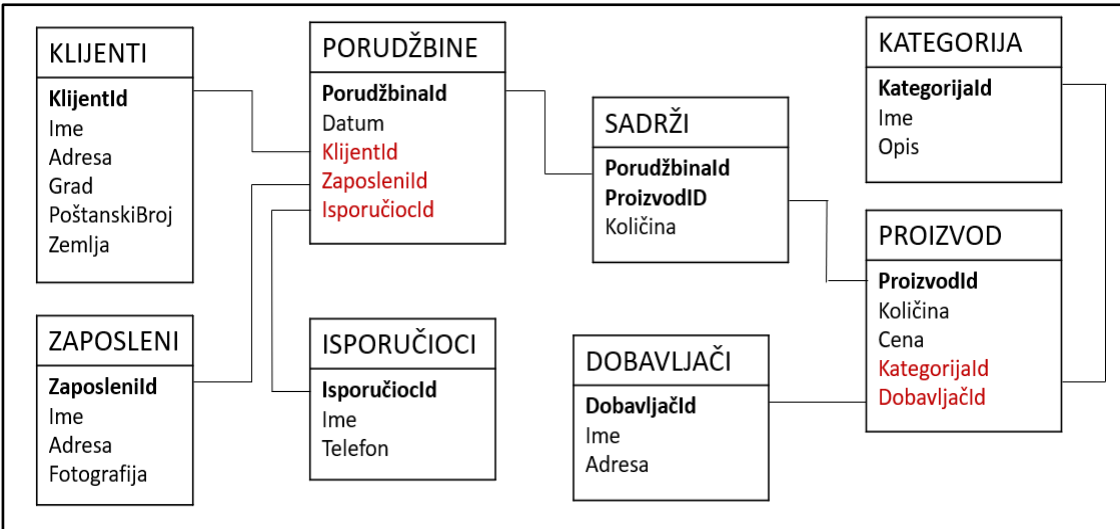


Opisan model naručivanja proizvoda kroz ER dijagram

Projektovanje – Relacioni model



Opisani model naručivanja proizvoda



Relacioni model naručivanja proizvoda

Dokumentovanje – određivanje šeme baze podataka

Dokumentovanje Entiteta

Ime Entiteta	Opis	Alternativna imena	Pojava
RADNIK	Opšti naziv za sve zaposlene u preduzeću	OSOBLE ZAPOSLEN	Sve osobe zaposlene u kompaniji min=2 max=1000
PROJEKAT	Svaki posao za koji je otvoren radni nalog	POSAD	Svi poslovi kompanije min=1 max=1000
SEKTOR	Deo kompanije koji obavlja neku delatnost	ODELJENJE	Sve organizacione jedinice kompanije min=1 max=20
CLAN_PORODICE	Dete radnika kompanije	DETE	Samo oni članovi porodice koji ostvaruju prava socijalnog programa preko radnika kompanije min=1 Max=4000 (4* broj radnika)

Dokumentovanje Atributa

Ime Entiteta/ Veze	Atribut	Opis	Alias	Složen	Viševrednosni	Izveden	Default	Not Null	Unique	Tip Podataka /dužina
RADNIK	MatBR (PK)	Matični br. radnika	MBR	-	-	-		+	+	Integer
	Ime	Ime radnika		Lime Prezime SSlovo	- - -	- - -		++ +	-	Varchar(15) Varchar(15) CHAR(1)
	DatRodj	Datum rođenja radnika		-	-	-		-	-	Date
	Pol	Pol radnika								CHAR(1)
	Plata	Plata radnika	LD							Number
	Adresa	Adresa stanovanja								Varchar(30)

Dokumentovanje Veza

Ime Veze	Entiteti	Ograničenja	Atributi veze
NADZOR (Rekurzivni tip veze)	RADNIK RADNIK	0..n 0..1	
RADI_NA	RADNIK PROJEKAT	1..n 1..n	Sati (Broj sati angažovanja radnika na projektu)
RUKOVODI	RADNIK ODELJENJE	0..n 0..1	DatPost (Datum Postavljenja Rukovodioca Odeljenja)
RADI_U	RADNIK SEKTOR	1..1 4..n	
JE_NOSILAC	SEKTOR PROJEKAT	0..n 1..1	
ZAVISI_OD (slab tip veze)	RADNIK ČLAN_PORODICE	0..n 1..1	

Implementacija – vrste podataka

Predstavljanje celih brojeva

TINYINT, SMALLINT, MEDIUMINT, INT, BIGINT

Predstavljanje razlomljenih brojeva

Decimal, Float, Double

Predstavljanje string podataka

Char, Varchar

Predstavljanje datumskih podataka

DATE, TIME, DATETIME, TIMESTAMP, YEAR

Predstavljanje binarnih podataka

Binary, Varbinary

Predstavljanje tekstualnih podataka

TINYTEXT, TEXT, MEDIUMTEXT, LONGTEXT

Predstavljanje velikih binarnih objekata

TINYBLOB, BLOB, MEDIUMBLOB, and LONGBLOB

Pravilan izbor tipa podataka
utiče na

- bolje performanse
- ispravnost unetih podataka
- manje zauzeće memorije.

- PODSETNIK -

NAJČEŠĆI MySQL TIPOVI PODATAKA

Tip podataka	Karakteristika
TINYINT	<i>Integer (-128 to 127)</i>
SMALLINT	<i>Integer (-32768 to 32767)</i>
MEDIUMINT	<i>Integer (-8388608 to 8388607)</i>
INT	<i>Integer (-2147483648 to 2147483647)</i>
BIGINT	<i>Integer (-9223372036854775808 to 9223372036854775807)</i>
FLOAT	<i>Decimal (precise to 23 digits)</i>
DOUBLE	<i>Decimal (23 to 53 digits)</i>
DECIMAL	<i>Decimal (to 65 digits – most precise)</i>

Tip podataka	Karakteristika
CHAR	<i>String (0 – 255)</i>
VARCHAR	<i>String (0 – 255)</i>
TINYTEXT	<i>String (0 – 255)</i>
TEXT	<i>String (0 – 65535)</i>
DATE	<i>YYYY-MM-DD</i>
DATETIME	<i>YYYY-MM-DD HH:MM:SS</i>
TIMESTAMP	<i>YYYYMMDDHHMMSS</i>
ENUM	<i>Jedna od brojnih unapred podešenih opcija</i>

- PODSETNIK -

6 OSNOVNIH ELEMENATA SQL ISKAZA

POČETAK
ISKAZA



KRAJ
ISKAZA

SELECT

*Prikaz kolone ili kolona koje želimo da
prikažemo u rezultatu upita*

SELECT naziv kolone

FROM

*Saopštavanje upitu iz koje tabele ili
tabela da uzme podatke*

FROM naziv tabele

WHERE

(Opcioni) Uslov za filtriranje zapisa

WHERE logički uslov

GROUP BY

*(Opcioni) Kolona ili kolone po kojima se
grupišu podaci u upitu*

GROUP BY naziv kolone

HAVING

*(Opcioni) Uslov za filtriranje rezultata
dobijenog nakon grupisanja*

HAVING logički uslov

ORDER BY

*(Opcioni) Definiše redosled prikaza
rezultata upita*

ORDER BY naziv kolone

- PODSETNIK - NAJČEŠĆI JOIN TIPOVI

INNER JOIN

*Vraća zapise koji postoje u **OBE** tabele, ne uključuje zapise koji se ne podudaraju*

FROM leftTable
INNER JOIN rightTable

LEFT JOIN

*Vraća SVE zapise iz **LEVE** tabele, i samo zapise koji se podudaraju iz **DESNE** tabele*

FROM leftTable
LEFT JOIN rightTable

RIGHT JOIN

*Vraća SVE zapise iz **DESNE** tabele, i samo zapise koji se podudaraju iz **LEVE** tabele*

FROM leftTable
RIGHT JOIN rightTable

FULL OUTER JOIN

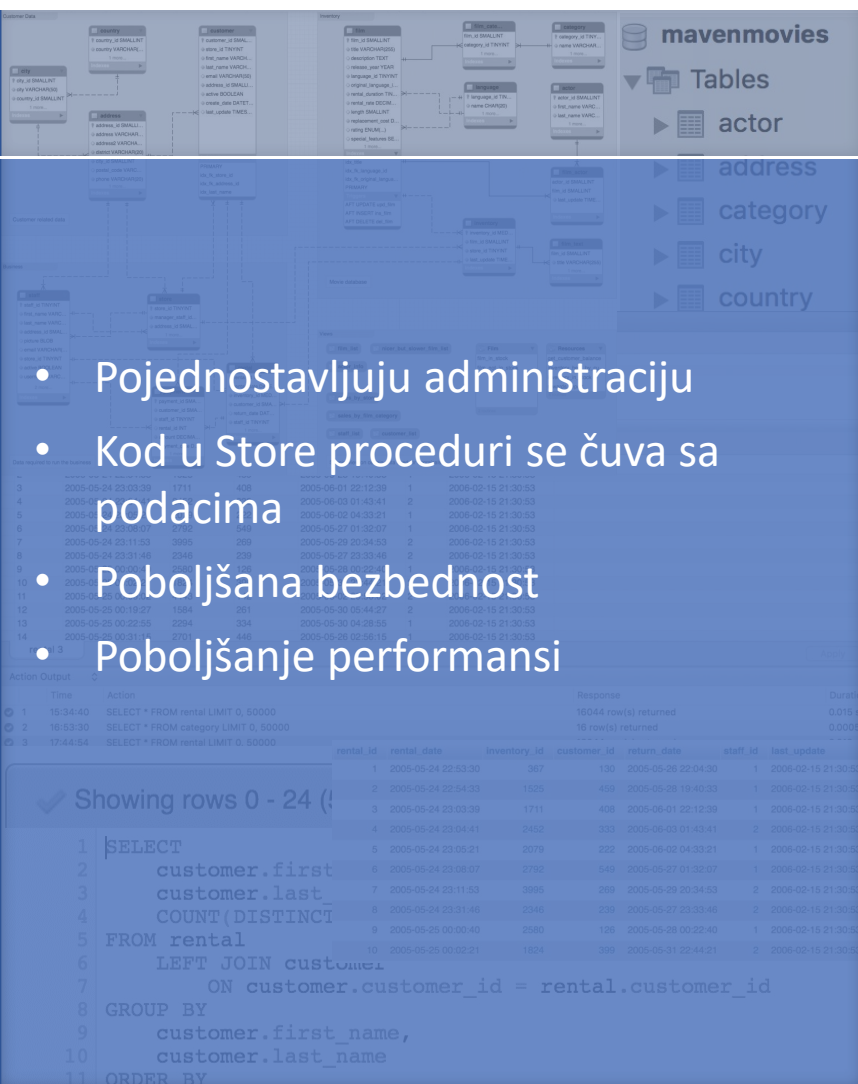
*Vraća SVE zapise iz **OBE** tabele, uključujući zapise koji se ne podudaraju*

FROM leftTable
FULL JOIN rightTable

UNION

*Vraća sve podatke iz jedne tabele sa **pridruženim** podacima iz druge tabele*

SELECT FROM firstTable
UNION
SELECT FROM secondTable



- Pojednostavljaju administraciju
- Kod u Store proceduri se čuva sa podacima
- Poboljšana bezbednost
- Poboljšanje performansi

PROGRAMIRANJE - STORE PROCEDURE

```
USE thriftshop;

-- changing the delimiter
DELIMITER //
-- creating the procedure
CREATE PROCEDURE sp_selectAllInventory()
BEGIN
    SELECT * FROM inventory;
END //
-- changing the delimiter back to the default
DELIMITER ;

-- calling the procedure that we have created
CALL sp_selectAllInventory();

-- if we later want to DROP the procedure, we can use the this...
-- DROP PROCEDURE IF EXISTS sp_selectAllInventory();
```

• MySQL trigger je sačuvan program koji se pokreće automatski uvek kada se desi događaj (event) koji je povezan sa tabelom

• Događaj (event) je povezan sa 3 DML iskaza

- INSERT
- UPDATE
- DELETE

• Trigger je način da povezane tabele ostanu sinhronizovane tokom vremena

```

1 SELECT
2   COUNT(DISTINCT customer_id) AS customer_count,
3   COUNT(DISTINCT rental_id) AS rental_count,
4   COUNT(DISTINCT inventory_id) AS inventory_count,
5   COUNT(DISTINCT staff_id) AS staff_count,
6   LEFT JOIN customer
7     ON customer.customer_id = rental.customer_id
8 GROUP BY
9   customer.first_name,
10  customer.last_name
11 ORDER BY
  
```

AUTOMATIZACIJA - TRIGERI

```

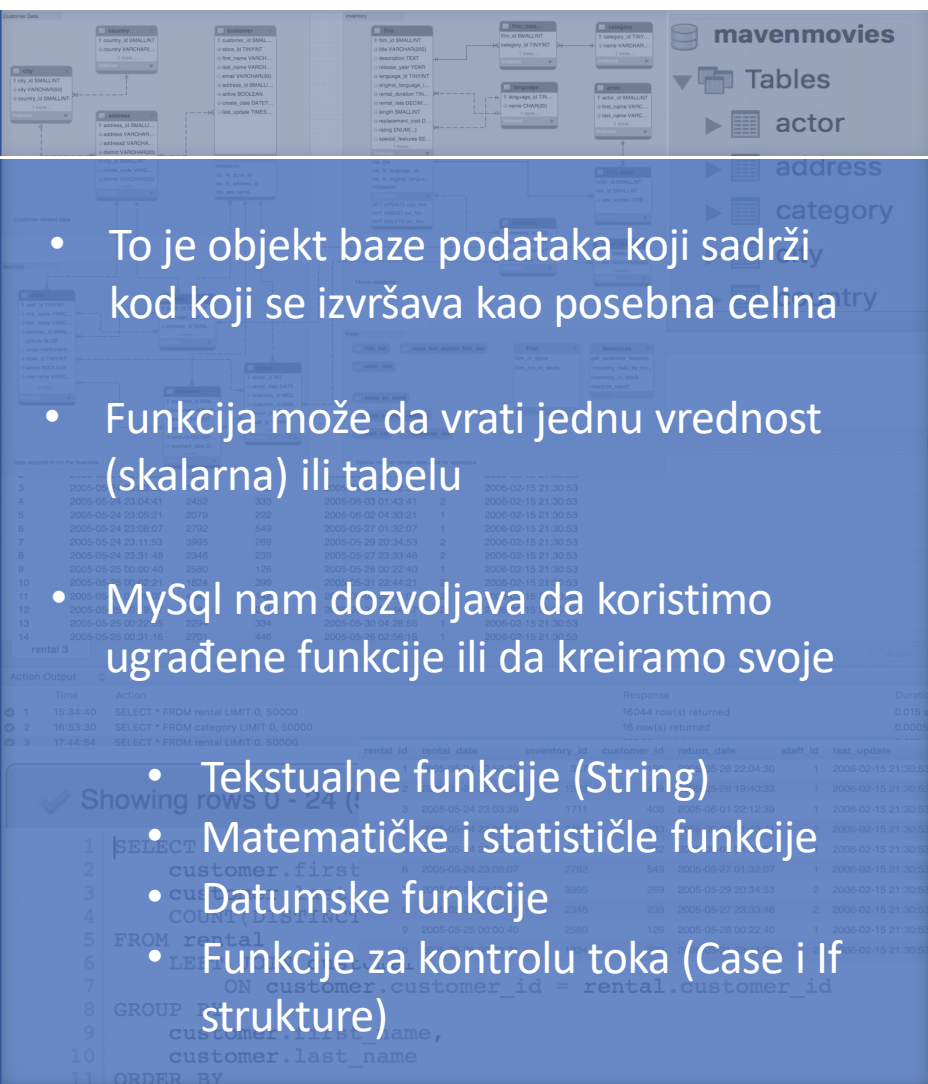
CREATE TRIGGER purchaseUpdateInventory
AFTER INSERT ON customer_purchases
FOR EACH ROW
UPDATE inventory
-- subtracting an item for each purchase
SET number_in_stock = number_in_stock - 1
WHERE inventory_id = NEW.inventory_id;

INSERT INTO customer_purchases VALUES
(13,NULL,3,NULL), -- inventory_id = 3, velour jumpsuit
(14,NULL,4,NULL) -- inventory_id = 4, house slippers
;

SELECT * FROM inventory;
  
```

Rezultat (SELECT * FROM inventory)

inventory_id	item_name	number_in_stock
1	fur coat	0
2	mocassins	4
3	velour jumpsuit	12
4	house slippers	6
5	brown leather jacket	3
6	broken keyboard	6
8	kneeboard	2
9	pro wings sneakers	0
10	wolf skin hat	1
11	fur fox skin	1
12	plaid button up shirt	8
13	flannel zebra jammies	6



The screenshot shows a database interface with a schema diagram on the left and a table of rental data on the right. The schema diagram includes tables like 'rental', 'customer', 'inventory', 'book', 'category', and 'actor'. The table of rental data has columns for rental_id, rental_date, inventory_id, customer_id, return_date, staff_id, and last_update. The table contains 24 rows of data.

- To je objekt baze podataka koji sadrži kod koji se izvršava kao posebna celina
- Funkcija može da vrati jednu vrednost (skalarna) ili tabelu
- MySql nam dozvoljava da koristimo ugrađene funkcije ili da kreiramo svoje
 - Tekstualne funkcije (String)
 - Matematičke i statističke funkcije
 - Datumske funkcije
 - Funkcije za kontrolu toka (Case i If strukture)

PROGRAMIRANJE – CASE struktura

```
SELECT OrderID, Status,
CASE Status
  WHEN 1 THEN "Odobren"
  WHEN 2 THEN "U obradi"
  ELSE "Nije odobren"
END
FROM OrderDetails;
```

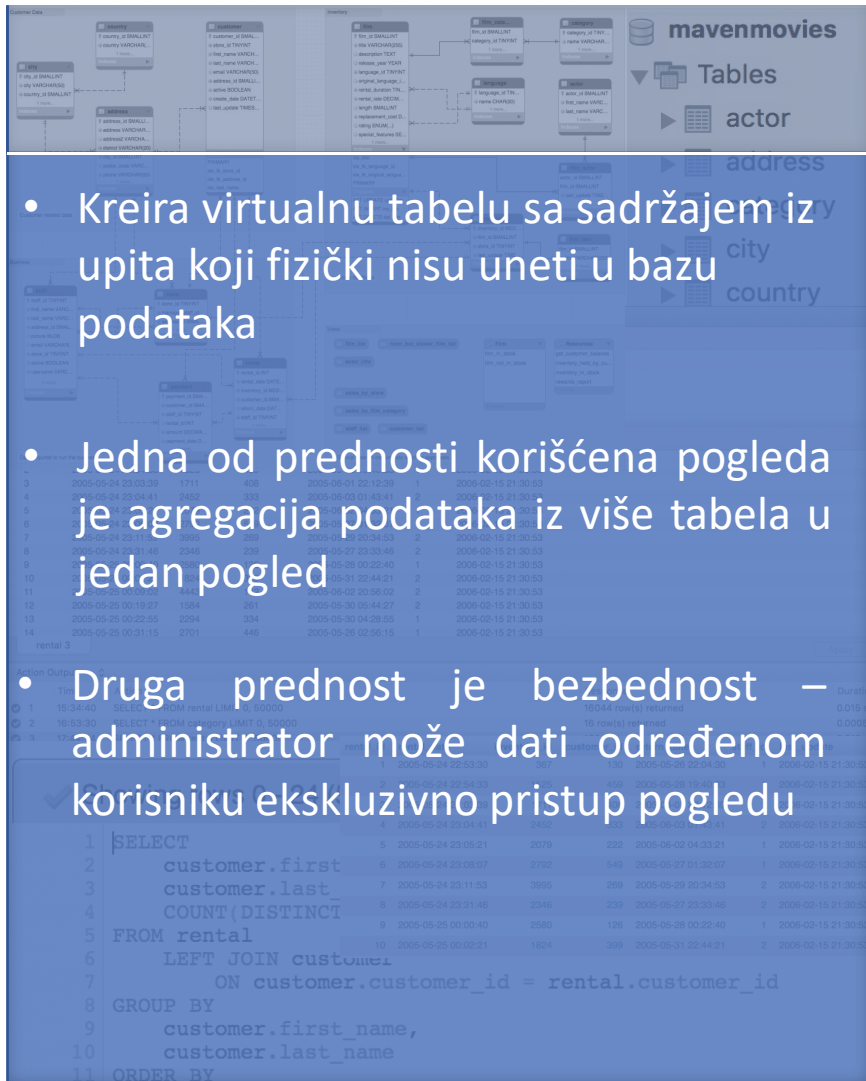
PROGRAMIRANJE – IF struktura

```
SELECT book_name,isbn_no,
IF(
  (SELECT COUNT(*) FROM book_mast WHERE pub_lang='English') >
  (SELECT COUNT(*) FROM book_mast WHERE pub_lang<>'English'),
  (CONCAT("Pages: ",no_page)),
  (CONCAT("Price: ",book_price))) AS "Page / Price"
FROM book_mast;
```

Uslov

Ako je uslov ispunjen

Ako uslov nije ispunjen



- Kreira virtualnu tabelu sa sadržajem iz upita koji fizički nisu uneti u bazu podataka
- Jedna od prednosti korišćena pogleda je agregacija podataka iz više tabela u jedan pogled
- Druga prednost je bezbednost – administrator može dati određenom korisniku ekskluzivno pristup pogledu

```

1 SELECT
2   customer.first
3   customer.last
4   COUNT(DISTINCT
5 FROM rental
6   LEFT JOIN customer
7     ON customer.customer_id = rental.customer_id
8 GROUP BY
9   customer.first_name,
10  customer.last_name
11 ORDER BY

```

POGLEDI (VIEWS)

```

-- creating a view
CREATE VIEW country_averages AS
SELECT
  country,
  AVG(years_experience) AS yrs_experience,
  AVG(CASE WHEN is_manager = 'Yes' THEN 1 ELSE 0 END) AS pct_mgrs,
  AVG(CASE WHEN education_level = 'Masters' THEN 1 ELSE 0 END) AS pct_masters
FROM salary_survey
GROUP BY 1;

-- selecting from the view
SELECT * FROM country_averages;

```

Rezultat

country	yrs_experience	pct_mgrs	pct_masters
United States	23.0379	0.2613	0.1731
United Kingdom	10.1005	0.2487	0.1005
Germany	10.6379	0.3276	0.3448
Netherlands	9.7955	0.2727	0.1364
Canada	11.3556	0.2593	0.0815
Poland	7.0000	0.3333	0.7083
Ukraine	7.1250	0.5000	0.7500
India	6.4795	0.3973	0.3699
Portugal	8.6667	0.3333	0.2222
South Africa	13.0645	0.2581	0.0000
Greece	10.5000	0.6667	0.4167
Israel	9.2727	0.2727	0.1818
Spain	7.4118	0.4118	0.4118
Norway	9.6667	0.0000	0.0000
France	9.7333	0.2000	0.4667
Jersey	11.6000	0.0000	0.2000
Argentina	6.4000	0.2000	0.0000
Belgium	9.7273	0.0909	0.0909
Switzerland	11.0909	0.2727	0.3182
Ireland	11.3478	0.4348	0.2174
Italy	10.8571	0.4286	0.1905

TRANSAKCIJE

- Rešavaju problem konkurentnog pristupa bazi podataka
- Transakcije se koriste da grupišu više povezanih iskaza i to tako da se svi iskazi izvršavaju ili se ni jedan ne izvršava (nema sredine).
- Svaka transakcija zadovoljava

- Atomičnost
- Konzistentnost
- Izolaciju
- Izdrživost

explain select username
from user_details
where username='Dusan'

Analiza izvršenja upita

id	select_type	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	SIMPLE	user_details	NULL	ALL	NULL	NULL	NULL	NULL	1005418	10.00	Using where

id	select_type	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	SIMPLE	user_details	NULL	ref	ind_username	ind_username	258	const	1	10.00	Using index

OPTIMIZACIJA - INDEKSI

explain select username
from user_details
where username='Dusan'

Analiza izvršenja upita

id	select_type	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	SIMPLE	user_details	NULL	ALL	NULL	NULL	NULL	NULL	1005418	10.00	Using where

id	select_type	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	SIMPLE	user_details	NULL	ref	ind_username	ind_username	258	const	1	10.00	Using index

Indeksi igraju značajnu ulogu u brzini rada same baze jer pomažu DBMS da brže pronađe ono što tražimo

Indeksi nam omogućavaju da brže pristupimo određenom redu u velikoj tabeli

Indeksiranje je trgovina između:

• Bržeg čitanja podataka

• Sporijeg upisa podataka

Showing rows 0 - 24 (16044 rows total)

rental_id	rental_date	inventory_id	customer_id	return_date	staff_id	last_update
1	2005-05-24 23:03:39	1711	408	2005-06-01 22:12:39	1	2006-02-15 21:30:53
2	2005-05-24 23:04:41	2452	333	2005-06-03 01:43:41	2	2006-02-15 21:30:53
3	2005-05-24 23:05:21	2079	222	2005-06-02 04:33:21	1	2006-02-15 21:30:53
4	2005-05-24 23:08:07	2795	540	2005-05-27 01:32:07	1	2006-02-15 21:30:53
5	2005-05-24 23:11:53	3896	380	2005-05-29 20:34:53	2	2006-02-15 21:30:53
6	2005-05-24 23:31:46	2346	239	2005-05-27 23:33:46	2	2006-02-15 21:30:53
7	2005-05-25 00:00:40	2590	126	2005-05-28 00:22:40	1	2006-02-15 21:30:53
8	2005-05-25 00:02:21	1824	399	2005-05-31 22:44:21	2	2006-02-15 21:30:53
9	2005-05-25 00:09:02	4443	142	2005-06-02 20:55:02	2	2006-02-15 21:30:53
10	2005-05-25 00:19:27	1584	281	2005-05-30 05:44:27	2	2006-02-15 21:30:53
11	2005-05-25 00:22:58	2294	334	2005-05-30 04:28:58	1	2006-02-15 21:30:53
12	2005-05-25 00:31:15	2701	448	2005-05-28 02:58:15	1	2006-02-15 21:30:53

```

1 SELECT
2   customer.first_name,
3   customer.last_name,
4   COUNT(DISTINCT rental_id) AS rental_count
5 FROM rental
6 LEFT JOIN customer
7   ON customer.customer_id = rental.customer_id
8 GROUP BY
9   customer.first_name,
10  customer.last_name
11 ORDER BY

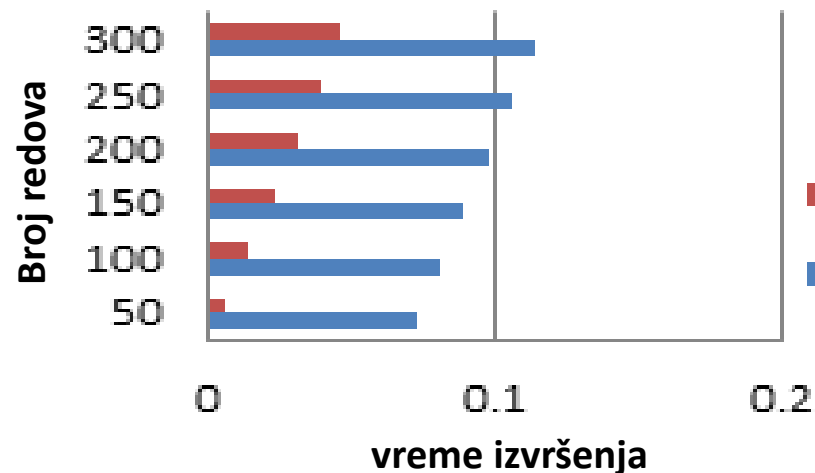
```

OPTIMIZACIJA UPITA

SELECT s.
FROM SH.sales s
WHERE s.prod_id = 14 OR s.prod_id = 17;*

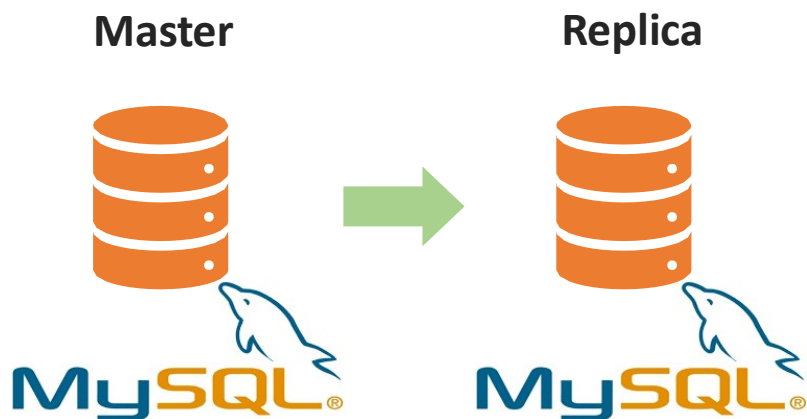
SELECT s.
FROM SH.sales s
WHERE s.prod_id IN (14, 17);*

Rezultat – 73% je ušteda u vremenu izvršenja upita



REPLIKACIJA BAZE PODATAKA

Replikacija nam omogućava da čuvamo isti podatak na dva ili više servera kreiranjem replike (kopije istog podatka)



BENEFITI & PRIMENA:

- Replikacija može da služi i kao forma backup-a
- Kreiranje **read only** replike za kreiranje izveštaja odvaja analizu podataka od produkcije
- U određenim okolnostima, raspodela opterećenja između servera može da poboljša performanse aplikacije

BACKUP & RECOVERY

Backup koristimo da povratimo podatke u slučaju da se oštete ili obrišu

Fizički vs Logički Backup

- **Fizički** backup čuva sirove (raw) podatke u fajlu, dok **Logički** backup čuva i SQL iskaze koji su potrebni da bi se šema baze podataka ponovo kreirala i popunila podacima

Online vs Offline Backup

- **Online** backup se koristi dok server radi. Prednost je što se rad servera ne prekida.
- **Offline** backup se koristi kada je server isključen.

Lokal vs Udaljeni Backup

- **Lokalni** backup se radi na istom hostu na kome se nalazi MySQL server
- **Udaljeni (Remote)** backup se pokreće sa udaljenog hosta.

NAJČEŠĆE BEZBEDONOSNE PRETNJE

1	Slaba (Weak) Autentifikacija	<i>Napadači mogu iskoristiti ograničene ili slabe metode autentifikacije</i>
2	Denial of Service (DoS)	<i>DoS napadači će pokušati da preplave ili sruše sistem, čineći ga beskorisnim za legitimne korisnike</i>
3	Eskalacija Privilegija	<i>Napadači će koristiti ovu strategiju za dobijanje pristupa višim nivoima u sistemu i informacija koje ne želimo da vide</i>
4	SQL Injection	<i>Postavljanje zlonamernog koda u SQL iskazu, putem unosa na web stranici.</i>
5	Buffer Overflow	<i>Napadači mogu da iskoristiti rupe u aplikaciji i da nateraju aplikaciju da se ne ponaša na regularan način</i>
6	Ransomware	<i>Vrsta zlonamernog softvera (Malware) dizajniranog da blokira pristup računarskom sistemu ili podacima dok se ne uplati iznos novca.</i>