ANALIZA I OBRADA PODATAKA

Upute za laboratorijske vježbe

[1. Osnove SQL-a 1. dio](#h.nebkxpgy1rgy)

[Zadatak:](#h.x96hkgluyfuk)

[ORDER BY](#h.4o1ccje4xdru)

[WHERE](#h.5wzsjfek1umk)

[NULL](#h.rv13kww4l3db)

[BETWEEN](#h.rtrvyqveqr6o)

[IN](#h.szfys5jyimcw)

[IZRAČUNATA POLJA](#h.x0iycumay1aj)

[Agregatne funkcije](#h.j2rfo7vcu2t5)

[GROUP BY](#h.9jh732ynbzc3)

[HAVING](#h.xqoklrovknb6)

[2. Osnove SQL-a - 2.dio](#h.tv1zjkjbow7u)

[IN](#h.4jp7njr2lm3b)

[Subqueries](#h.97fg8spl44eh)

[Korištenje sql querija kao izvora podataka](#h.th5xgkychq05)

[Korištenje oznaka tablica ( aliases )](#h.4ocq3q2b8h3c)

[Povezivanje podataka iz više tablica](#h.hfbvteagpzzl)

[Unutarnji JOIN](#h.875m59eiv86g)

[Vanjski Join](#h.tqt4s7hrhx0l)

[LEFT JOIN](#h.3jo17n2j6jq9)

[RIGHT JOIN](#h.s4azo0nz2xxy)

[FULL JOIN](#h.6mehu846d92h)

[UNION](#h.s1m7dr5lldm0)

[CASE](#h.jeuiaoh4dlqg)

[3. Višedimenzijska analiza - OLAP](#h.8rm4oax65std)

# 1. Osnove SQL-a 1. dio

 Reference:       [*http://www.learn-sql-tutorial.com/SimpleSelects.cfm*](http://www.google.com/url?q=http%3A%2F%2Fwww.learn-sql-tutorial.com%2FSimpleSelects.cfm&sa=D&sntz=1&usg=AFQjCNEtXW1diezThs2iSXw39s07DHHjWw)

[*http://www.learn-sql-tutorial.com/AdvancedSelects.cfm*](http://www.google.com/url?q=http%3A%2F%2Fwww.learn-sql-tutorial.com%2FAdvancedSelects.cfm&sa=D&sntz=1&usg=AFQjCNGcYFiy7ecRMWcPZitM_amIDrK0sw)

<http://www.w3schools.com/sql/sql_intro.asp>

Online Northwind baza podataka: <http://www.w3schools.com/sql/sql_tryit.asp>

### Zadatak:

Na SQL SERVER-u , aplikacija SQL SERVER QUERY ANALYZER, baza podataka NorthWind, unijeti upite prema zadacima, testirati ih, te poslati kao odgovor.

**SELECT**

Select naredba se koristi dohvat podataka iz tablica. Najjednostavniji oblik naredbe je:

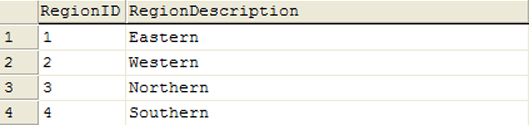
SELECT polje1, polje2,... FROM tablica1

Select naredba na SQL SERVER-u nije osjetljiva na velika i mala slova, kao ni nazivi polja i tablica. Kod drugih sustava za upravljanje bazama podataka to ne mora biti slučaj. Primjerice, MYSQL razlikuje velika i mala slova, pa ime tablice Firma i FIRMA ne označava istu tablicu. Također, SQL interpreter ne zahtijeva da se dijelovi naredbi pišu u novom redu, međutim zbog preglednosti koda se preporuča pisanje svakog dijela SQL naredbe u novom redu.

SELECT \* FROM Employees;  
  
SELECT \*  
FROM Employees;

Primjer: --Prikazati sva polja iz tablice Regions

SELECT \*  
FROM Region;



Zadatak:

1. Odaberati polja ContactName, CompanyName, ContactTitle i Phone iz tablice Customers .
2. Odaberati polja EmployeeID, Title, FirstName, LastName, and Region iz tablice Employees .

### ORDER BY

SINTAKSA:

SELECT polje, polje

FROM tablica

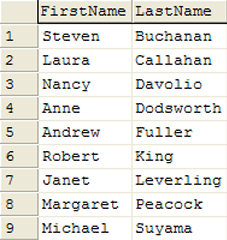
ORDER BY polje;

Primjeri:

SELECT FirstName, LastName

FROM Employees

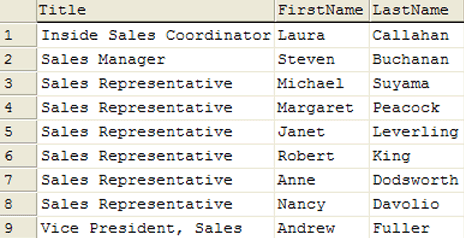
ORDER BY LastName;



SELECT Title, FirstName, LastName

FROM Employees

ORDER BY Title **ASC**, LastName **DESC**;



**Zadaci:**

1. Kreirajte popis sa imenima i prezimenima zaposlenika, te datumima dolaska na posao ( Hire Date ) sortiran tako da su na početku liste zaposlenici koji su kasnije primljeni.
2. Kreirajte popis narudžbi (Orders) sortiran prema cijeni transporta (Freight)  sortirane od najskuplje do najjeftinije. Prikažite slijedeća polja: OrderID, OrderDate, ShippedDate, CustomerID, i Freight.

### WHERE

SINTAKSA:

SELECT stupac, stupac

FROM tablica

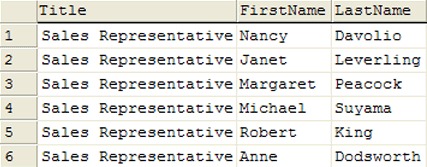
WHERE uvjeti;

Primjer:

SELECT Title, FirstName, LastName

FROM Employees

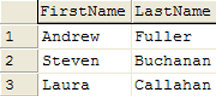
WHERE Title = 'Sales Representative';



SELECT FirstName, LastName

FROM Employees

WHERE Title **<>** 'Sales Representative';



**Zadaci:**

1. Kreirajte listu svih firmi i imena kontakta kupaca u Buenos Aires-u.
2. Kreirajte listu proizvoda koja pokazuje ime, jediničnu cijenu (unit price) za proizvode kojih nema na skladištu.
3. Kreirajte listu svih zaposlenika ( ime, prezime ) čija prezimena počinju slovima N - Z

### NULL

NULL označava polje u koje nije unesen nikakav podatak. ( NULL nije jednak 0, jer je 0 uneseni podatak )

Odabrati sve zaposlenike za koje nije unesena regija ( podatak u tablici je prazan - NULL )

SELECT FirstName, LastName

FROM Employees

WHERE Region **IS NULL**;

Odabrati zaposlenike za koje **je** unesena regija ( podatak u tablici nije prazan -  IS NOT NULL )

SELECT FirstName, LastName

FROM Employees

WHERE Region **IS NOT NULL**;

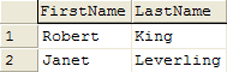
### 

### BETWEEN

SELECT FirstName, LastName

FROM Employees

WHERE LastName BETWEEN 'J' AND 'M';



Ovaj upit ima isti učinak kao i:

SELECT FirstName, LastName

FROM Employees

WHERE **LastName >= 'J' AND LastName <= 'M'**;

### IN

Odabir onih polja čije se vrijednosti nalaze na popisu. Ekvivalentan je naredbi OR

SELECT TitleOfCourtesy, FirstName, LastName

FROM Employees

WHERE City IN ('London','Seattle');

ima isti efekt kao i:

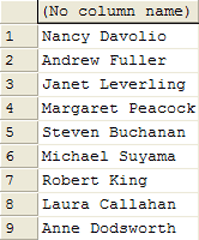
... WHERE City ='London' **OR** City = 'Seattle');

### IZRAČUNATA POLJA

Primjer:

SELECT FirstName + ' ' + LastName

FROM Employees;



Ako je cijena prijevoza ( Freight ) veća od 500, plaća se 10% poreza. Prikazati tako uvećanu cijenu.

SELECT OrderID, Freight, Freight \* 1.1

FROM Orders

WHERE Freight >= 500;

Zadatak:

1. Kreirati listu zaposlenika formatiranu na slijedeći način:

Andrew Fuller radi na radnom mjestu Vice president, Sales i dolazi iz London-a.

## Agregatne funkcije

Agregatne funkcije izračunavaju rezultat iz većeg broja redova.

COUNT - broj redova koji su vraćeni

SUM - suma vrijednosti u svim redovima

MAX - največa vrijednost u svim redovima

MIN - najmanja vrijednost u svim redovima

AVG - prosječna vrijednost u svim redovima

Primjer:

SELECT COUNT (ID) FROM EMPLOYEES;

SELECT MAX (UnitPrice) AS NAJSKUPLJI, MIN (UnitPrice) AS NAJJEFTINIJI FROM Products

### GROUP BY

Pomoću GROUP BY možemo agregatne funkcije primijeniti na sve iste vrijednosti nekog polja. Primjerice, možemo vidjeti koliko zaposlenika ima u kojem gradu

Primjeri:

SELECT FIRSTNAME, CITY FROM EMPLOYEES;

SELECT COUNT(FIRSTNAME), CITY FROM EMPLOYEES GROUP BY CITY;

Zadatak:

1. Kreirajte listu svih proizvoda sa prosječnom, minimalnom i maksimalnom cijenom prema grupi proizvoda (CategoryID)

### HAVING

Pomoću klauzule HAVING filtriramo podatke koje smo dobili grupiranjem. U HAVING klauzuli možemo koristiti samo ona polja po kojima smo grupirali podatke.

Primjer:

SELECT COUNT(FIRSTNAME), City

FROM Employees

GROUP BY City

HAVING City= 'London';

Zadatak:

1. Kreirajte listu svih proizvoda sa prosječnom, minimalnom i maksimalnom cijenom prema grupi proizvoda (CategoryID) i prikažite samo one podatke gdje je grupa proizvoda veća od 3.

**Redoslijed korištenja klauzula kod naredbe SELECT**

1. SELECT
2. FROM
3. WHERE
4. GROUP BY
5. HAVING
6. ORDER BY

**Neke dodatne funkcije koje se mogu koristiti u SQL upitima:**

Apsolutna vrijednost:  ABS  -->  ABS (-5) = 5

Najbliži veći cijeli broj: CEILING  -->  CEILING ( 2.7 ) = 3, CEILING (2.1) = 3

Najbliži manji cijeli broj: FLOOR --> FLOOR ( 2.3 ) = 2, FLOOR ( 2.7 ) = 2

Potenciranje: POWER  --> POWER ( 3,2 ) = 9

Zaokruživanje na broj decimala: ROUND  --> ROUND ( 2.53653,2 ) = 2.54

Kvadratni korijen: SQRT  --> SQRT ( 16 ) = 4

Ispis teksta malim slovima:  LOWER --> LOWER ("SPLit") = "split"

Ispis teksta velikim slovima:  UPPER --> UPPER ("SPLit") = "SPLIT"

Brisanje praznih znakova sa desne strane: RTRIM --> RTRIM ("Split   ") = "Split" Brisanje praznih znakova sa lijeve strane: LTRIM --> LTRIM ("   Split") = "Split" Dio teksta: SUBSTRING --> SUBSTRING  ("Split", 2,2 ) = "pl"

Zbrajanje datuma:  DATEADD  --> DATEadd (year,2, '06/09/09' ) = 6.9.2011

Razlika datuma: DATEDIFF  --> DATEDIFF (year, '06/09/09', '06/09/10' ) = 1

Trenutno vrijeme: GETDATE --> GETDATE()

# 2. Osnove SQL-a - 2.dio

[*http://www.learn-sql-tutorial.com/SimpleSelects.cfm*](http://www.google.com/url?q=http%3A%2F%2Fwww.learn-sql-tutorial.com%2FSimpleSelects.cfm&sa=D&sntz=1&usg=AFQjCNEtXW1diezThs2iSXw39s07DHHjWw)

[http://www.learn-sql-tutorial.com/SubqueriesJoinsUnions.cfm](http://www.google.com/url?q=http%3A%2F%2Fwww.learn-sql-tutorial.com%2FSubqueriesJoinsUnions.cfm%23&sa=D&sntz=1&usg=AFQjCNEBXdaXUjT33LytBNfMqHFSBT93CQ)

Zadatak za vježbu:

1. Kreirati listu svih proizvoda sa prosječnom, minimalnom i maksimalnom cijenom prema grupi proizvoda (CategoryID) i prikažite samo one podatke gdje je grupa proizvoda veća od 3
2. Kreirajte listu svih zaposlenika (ime, prezime) čija prezimena počinju slovima N - Z
3. Kreirajte popis koji sadrži ime i državu svih kupaca čija šifra sadrži slova HO, počinje sa slovima LA ili završava sa slovom B
4. Kreirajte popis svih kupaca kod kojih prva 2 znaka šifre nisu jednaka kao prva 2 znaka u imenu
5. Iz tablice Customers dobiti popis svih država u kojima ima kupaca (na popisu jedna država može biti prikazana samo jednom)
6. Kreirati popis svih kupaca iz Njemačke, Engleske i SAD-a koji nisu iz Portlanda i Londona.
7. Kreirati popis klijenata prema slijedećoj šabloni:

Kupac Island Trading je iz države UK, Garden House Crowther Way , PO31 7PJ Cowes. Kontakt osoba je Helen Bennett.

### IN

Operator IN služi za vraćanje vrijednosti koje se nalaze na popisu, odvojene zarezima.

Primjer: Prikazati popis svih klijenata iz Berlina, Graza i Barcelone. Prikazati naziv, šifru, grad i državu.

SELECT CompanyName, CustomerID, Country

FROM customers

WHERE City IN ('Berlin','Graz', 'Barcelona' );

## Subqueries

Služe za dohvat podataka koristeći širi skup kriterija od onog kojeg možemo dobiti pomoću where.

Sintaksa:

SELECT stupac, stupac

FROM tablica

WHERE kriterij1 = ( ili IN ) ( select stupac from ... )

Pritom moraju biti zadovoljeni slijedeći uvjeti:

* Subquery mora vraćati jedan stupac, i to istog tipa podataka kao i parametar s kojim s uspoređuje.
* Ako subquery vraća samo 1 redak, može se koristiti =, a ako ne, mora se koristiti IN ili NOT IN

Primjer:

SELECT CompanyName

FROM Customers

WHERE CustomerID IN (

SELECT CustomerID

   FROM Orders

   WHERE

OrderDate **BETWEEN '1-Jan-1997'**

**AND '31-Dec-1997'**

);

ZADACI:

1. Kreirati izvještaj koji pokazuje  imena svih  proizvoda u kategoriji "Seafood".
2. Kreirati listu svih dobavljača proizvoda kategorije 'seafood'.
3. Kreirati popis svih prodavača koji su prodali nešto u Irskoj.
4. Kreirati popis svih firmi koje prodaju proizvode u kategoriji čiji ID je 8

### Korištenje sql querija kao izvora podataka

Umjesto tablice, kao izvor podataka možemo definirati upit.

SELECT NAZIV FROM ( SELECT CompanyName AS NAZIV FROM CUSTOMERS WHERE COUNTRY ='venezuela' ) upit

### 

### Korištenje oznaka tablica ( aliases )

Umjesto punog imena tablice, može se koristiti skraćena oznaka, što se često koristi radi pisanja preglednijeg koda, kao obaveznog naziva izvora podataka dobivenog SQL-om, itd.

PRIMJER:

SELECT e.EmployeeID, e.FirstName, e.LastName,

FROM Employees e

## Povezivanje podataka iz više tablica

### Unutarnji JOIN

Osnovni koncept relacijskih baza je čitanje i obrada podataka iz više tablica.

SELECT TABLICA1.POLJE1, TABLICA2.POLJE2.....

FROM TABLICA1

JOIN TABLICA2 ON ( TABLICA1.ID.. = TABLICA2.ID... )

WHERE ....

Standardni (unutarnji) Join služi za dohvat podataka iz više tablica kada u svim tablicama kojima postoje podaci u povezanim poljima (veza 1..N).

PRIMJER:

Prikazati popis svih zaposlenika i narudžbi koje su zaprimili:

**SELECT**

**Employees.EmployeeID,**

**Employees.FirstName,**

**Employees.LastName,**

**Orders.OrderID,**

**Orders.OrderDate**

**FROM Employees**

**JOIN Orders ON (Employees.EmployeeID = Orders.EmployeeID)**

**ORDER BY Orders.OrderDate;**

**ZADACI**:

1. Prikazati popis svih proizvoda, sa nazivom kategorije, sortirano prema nazivu proizvoda

2. Korištenjem alias-a tablica kreirajte popis svih prodavača koji su nešto prodali u Irskoj

**Višestruki JOIN**

Na isti način moguće je povezati i više tablica, definirajući više JOIN-a.

Sintaksa:

**SELECT**

**table1.column,**

**table2.column,**

**table3.column**

**FROM table1**

**JOIN table2 ON (table1.column=table2.column)**

**JOIN table3 ON (table2.column=table3.column)**

**WHERE conditions**

Oprez: Prilikom dodavanja novih "join-a" mogu se koristiti samo one tablice koje su već povezane. Nije moguće spominjati tablicu3 u prvom Join-u!

**PRIMJER:**

**SELECT**

**o.OrderID,**

**c.CompanyName,**

**e.FirstName,**

**e.LastName**

**FROM Orders o**

**JOIN Employees e ON (e.EmployeeID = o.EmployeeID)**

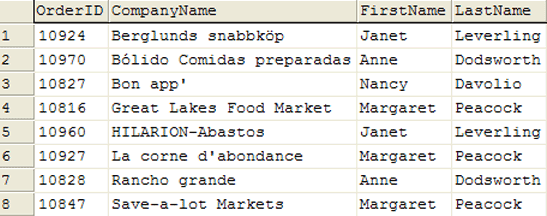
**JOIN Customers c ON (c.CustomerID = o.CustomerID)**

**WHERE**

**o.ShippedDate > o.RequiredDate**

**AND o.OrderDate > '1-Jan-1998'**

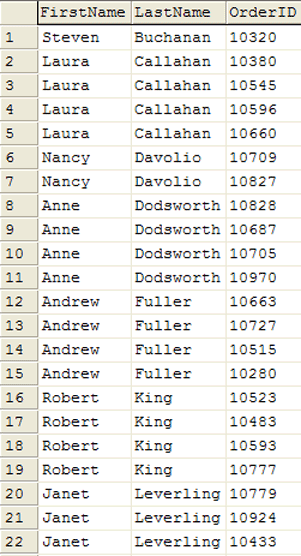
**ORDER BY c.CompanyName;**



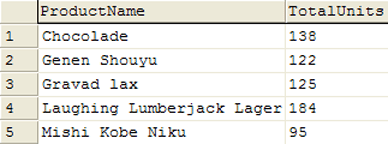
**ZADACI:**

1. Kreirajte listu koja pokazuje šifru narudžbe (order\_ID), ime i prezime zaposlenika za one zaposlenike koji su zakasnili sa isporukom

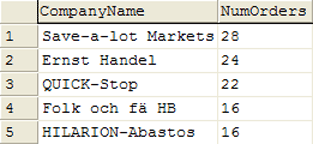
(datum isporuke > traženog datuma)



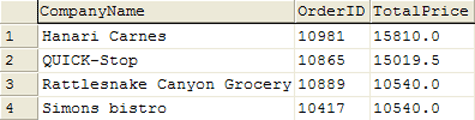
2. Kreirajte popis koji pokazuje koliko je kojeg proizvoda ukupno naručeno. Popis neka prikazuje samo one proizvode kojih je naručeno manje od 200. Popis neka bude sortiran silazno po broju naručenih proizvoda.



3. Kreirajte izvještaj koji pokazuje ukupni broj narudžbi po kupcu od 31.12.1996. Popis neka prikazuje samo one kupce koji imaju više od 15 narudžbi



4. Napraviti popis svih narudžbi na kojima se nalaze oni proizvodi kojih je prodano u vrijednosti od preko 10.000



## Vanjski Join

Vanjski Join služi za dohvat podataka iz tablica u gdje ne mora biti pripadajućih podataka u povezanim tablicama  ( veza 0..N )

### LEFT JOIN

Vraća sve podatke iz prve tablice, i ako u drugoj (povezanoj) tablici nema odgovarajućih podataka.

SELECT

tablica1.polje,

tablica2.polje

FROM tablica1

LEFT JOIN tablica ON (tablica1.ID..= tablica2.ID...)

WHERE ....

**PRIMJER:**

SELECT

c.companyname,

count (o.orderID)

FROM customers c

LEFT JOIN orders o on (c.customerID=o.customerID)

GROUP BY c.companyname

ORDER BY count (o.orderid)

**ZADATAK**:

1. Izraditi isti upit sa unutarnjim JOIN-om i uočiti razliku.

### RIGHT JOIN

Vraća sve podatke iz povezane tablice, čak i ako u prvoj tablici nema odgovarajućih redova.

Zadatak:

Prethodni primjer sa left join-om ispisati koristeći right join.

### FULL JOIN

Vraća sve podatke iz obiju povezanih tablica, neovisno o tome ima li u drugoj tablici podataka.

**ZADATAK**:

1. Koristeći neki oblik JOIN-a vidjeti koliko kupaca iz baze nema nijednu narudžbu.

## UNION

Pomoću UNION se dohvaćaju podaci iz više tablica u jedan "niz". Pravila za korištenje UNION su prilično stroga:

* podaci koji su u nizu moraju biti istog tipa.
* svaki upit mora vratiti isti broj polja,
* sva moraju biti u istom redoslijedu

Ako se u rezultatima pojavi više istih redova, prikazuje se samo jedan. Ako ih želimo prikazati sve, koristimo **UNION ALL**

**PRIMJER:**

SELECT CompanyName, Phone FROM Shippers

UNION

SELECT CompanyName, Phone FROM Customers

UNION

SELECT CompanyName, Phone FROM Suppliers

**ZADATAK**

1. Napraviti jedinstveni telefonski imenik svih zaposlenika, kupaca i dobavljača (prikazati ime i telefonski broj )

## 

## 

## CASE

Case vraća različite vrijednosti kao izvor podataka ovisno o postavljenim kriterijima.

SINTAKSA:

SELECT polje1, polje2, CASE

   WHEN logički\_izraz1 THEN vrijednost1

   WHEN logički\_izraz2 THEN vrijednost2

   ELSE vrijednost3

  END

AS polje3

FROM tablica

Slijedeća naredba kreira novo polje "Kašnjenje", i u njega unosi vrijednosti OK ili ZAKASNIO, ovisno o odnosu vremena isporuke i traženog datuma.

Na ovaj način smo kreirali novi kriterij za usporedbu podataka, "Dimenziju", i definirali smo unutar nje jednu hijerarhiju.

SELECT

c.CustomerID, c.CompanyName, e.EmployeeID, e.FirstName, e.LastName, OrderID,

(CASE

  WHEN ShippedDate < RequiredDate

   THEN 'OK'

   ELSE 'ZAKASNIO'

   END) AS kasnjenje

FROM Orders o

JOIN Employees e ON (e.EmployeeID = o.EmployeeID)

JOIN Customers c ON (c.CustomerID = o.CustomerID)

ORDER BY kasnjenje;

**ZADATAK:**

1. Kreirati popis kupaca, i broj narudžbi koje su napravili. Za kupce koji nemaju nijednu narudžbu napisati "NIJEDNA", a za kupce koji imaju manje od 5, napisati "MALO", za one koji imaju više od 10, "PUNO", a za ostale "SREDNJE".

    Napomena: Najprije kreirati popis kupaca sa brojem narudžbi koje imaju

2. Kreirati izvještaj koji pokazuje koliko kupaca ima nijedna, malo, srednje i puno narudžbi.

3. Kreirati listu koja pokazuje koliko je ukupno prosječno kašnjenje svih isporuka u danima.

4. Kreirati listu koja pokazuje koliko dana su u prosjeku kasnile isporuke koje nisu isporučene na vrijeme.

5. Kreirati popis zaposlenika na način da svi zaposlenici mlađi od 50 godina dobiju oznaku "MLAD" a oni sa 50 i stariji "STAR"

6. Kreirati popis koji pokazuje koliko prosječno godina imaju "stari" zaposlenici, a koliko "mladi"

# 3. Višedimenzijska analiza - OLAP

**Analisys Manager** je aplikacija za kreiranje i pregled višedimenzionalnih struktura podataka (OLAP kocki). Analysis Manager je dio paketa aplikacija SQL Server-a za analizu podataka, Analysis Services.

Kao ulaz u aplikaciju služe podaci iz transakcijskih baza podataka, ili spremišta podataka.Ovako pripremljeni podaci (OLAP Kocke) se kasnije mogu analizirati pomoću drugih alata, ili se nad njima može vršiti dubinska analiza (Data Mining), također koristeći Analysis Services platformu.

**ZADATAK:**

Izraditi OLAP kocku iz podataka u Demo bazi podataka na SQL Serveru, te analizirati podatke pomoću Viewer-a, i pomoću Excel-a.

Koraci za izradu OLAP kocke su slijedeći:

- Postaviti izvor podataka

- Definirati OLAP kocku

- Uvesti podatke iz baze u OLAP kocku

- Analizirati podatke

Platformu Analysis Services aktiviramo u meniju SQL Server-a, pozivajući aplikaciju "Analysis Manager"

Za ove vježbe koristiti će se primjer naveden u Help sustavu aplikacije "Analisys Manager" za SQL Server 2000.

Spajanje na računalo sa Analysis services:

***Start*** - ***Run*** - **mstsc.exe**

***Computer***: [unijeti IP adresu koju ćete dobiti na vježbama]

Podaci za spajanje:

korisničko ime: ***student1*** - ***student12***

lozinka: ***oss***

1. Aktivirati "Analysis Manager"

***Start - Microsoft SQL Server - Analysis Services - Analysis Manager***

Otvara se Microsoft Management Console. Na lijevoj strani ekrana je popis Analysis servera, a na desnoj linkovi za Help sustav.

***Odabrati "Analysis Manager Concepts and Tutorial"***

Zadatak je izraditi OLAP kocke za FoodMart, napuniti ih podacima i pregledati dobivene podatke

Odabrati **"Quick Start"**

1. Korak: Kreiranje izvora podataka.

Da bi se podaci mogli koristiti između više različitih aplikacija, koristi se protokol za razmjenu podataka ODBC ( Open Database Connectivity ). ODBC pruža "interface" prema različitim formatima podataka ( Access, Excel, SQL Server, Oracle, DB2... ) tako da aplikacija koja čita te podatke ih uvijek vidi na isti način. ODBC se postavlja tako da se definira "Izvor podataka" - Data Source Name - DSN.

Prema uputama "How to set up your system data source name (DSN)" postaviti izvor podataka. Umjesto imena **Tutorial** unijeti **DSN1** - **DSN12**. U polje "**Description**" unijeti svoje ime i prezime.

Sada sve aplikacije mogu koristiti podatke iz Access baze "Foodmart 2000", tako da se spoje na navedeni DSN.

2. Korak: Postavljanje strukture podataka u Analysis Server

Aktivirati granu "Analysis Servers". Desni-klik na ime servera, odabrati "New Database". U **"Database Nam**e" unijeti **BAZA1-BAZA12**. U polje "**Description**" unijeti svoje ime i prezime.

3. Korak - Pogledati svojstva nove baze podataka

* Izvori podataka
* Kocke
* Dijeljene Dimenzije
* Modeli rudarenja podataka
* Uloge u bazi podataka

4. Korak: bazu podataka spojiti na izvor podataka

1. Desni klik na Data Sources - New Data Source
2. Odabrati ODBC protokol za spajanje - **Microsoft OLE DB Provider for ODBC Drivers**
3. Na kartici Connection odabrati odgovarajući DSN

5. Korak: Otvoriti Access bazu FoodMart 2000 - i analizirati podatke. Odgovoriti na pitanja:

* Kakve podatke sadrži ova baza?
* Gdje su podaci o prodaji?
* U kojoj tablici su transakcijski podaci?
* Kakve podatke sadrži tablica "sales\_fact\_1998"
* Na temelju znanja sa predavanja, kakva je ovo vrsta baze podataka?

6. Korak : Napraviti OLAP kocku

Pročitati uvod u Tutorial-u.

OLAP kocka koju ćemo napraviti služiti će za analizu podataka iz prodaje.

Koraci za modeliranje OLAP kocke:

- odabrati tablice za izvor podataka

* + glavnu tablicu (Fact Table)
* Odabrati veličine koje mjerimo (mjere)
  + u glavnoj tablici odabrati polja u kojima su tražene veličine
* Definirati dimenzije
  + Odabrati strukturu dimenzije
    - Ako su svi podaci o jednoj dimenziji u jednoj tablici - Snowflake
    - Ako su podaci hijerarhijski u više tablica - starflake
  + Odabrati hijerarhiju u dimenziji
    - npr. za dimenziju vrijeme:
      * Godina, kvartal, mjesec
      * ak.godina, semestar, tjedan
    - npr. za dimenziju Kupac
      * Država, županija, grad
* Definirati veličinu i brzinu izvršavanja kocke

OLAP struktura jr prilično zahtjevna po pitanju veličine, a veličina kocke ovisi o količini redundantnosti, tj. broju unaprijed izračunatih vrijednosti. Prije izgradnje kocke moguće je unaprijed definirai omjer brzine i veličine kocke.

* Pregledavanje podataka prema traženim kriterijima.

Analysis Manager ima ugrađen OLAP Klijent, za pregled informacija iz OLAP kocke, ali se može koristiti i neki drugi preglednik (npr. Excel.)

Za izradu kocke slijediti upute u Analysis Manager:

Analysis Manager - Concept and Tutorial, Quick Start

* Build a Cube
  + Imenima dimenzija dati hrvatske nazive
  + NAPOMENA: Na kraju Wizard-a, isključiti opciju: ***Share this dimension with other cubes***
* Edit a Cube
* Design Storage and Process the Cube
* Browse Cube Data

Analisys servisi -  izrada kocke

Tutorial u Analisys Manageru

* 1. set up DSN
* 2. set up database
* 3 set up cube - select fact table
* Set parent-child dimension -kreirati novu dimenziju i odabrati schemu Parent-child
* Calculated members - Kocka-Edit-Add Calculated Member ( ako radimo formulu od vrijednosti, za parent odaberemo Measures )
* Member properties - Kocka - edit - Dimensions, odabrati neku, Member Properties, dodati
* Virtual dimensions - Analisys Manager-Shared Dimensions, desni klik, New Dimension. Dimenziju dodati na Kocku: Na kocku - Edit, Dimensions, desni klik, Existing Dimensions, odabrati kreiranu dimenziju i staviti ju na listu.

 Set Dimension Custom Rollups, Write Back, and More

 How to set advanced properties using Dimension Editor

1. In the Analysis Manager tree pane, under the Tutorial database, expand the Shared Dimensions folder, right-click the newly created Account dimension, and then click Edit.
2. Dimension Editor is displayed. The left pane shows a tree view and a properties pane, and the right pane shows dimension schema or data. If the properties pane is not visible, expand it by clicking the Properties button in the lower left corner.
3. The properties pane is refreshed each time the focus is set in the tree pane. In the tree pane, click Account Id. The properties pane now displays the properties attached to the Account dimension members. Click the Advanced tab. Scroll down to the last property, Unary Operators. It is set to True. You can change this property at any time by clicking the expand (...) button next to the property field.
4. In the tree pane, click Account. In the Advanced tab, note that the Members with data property is set to Nonleaf data visible. You set this property earlier in the wizard.
5. Scroll up in the Advanced tab to find the Member Names Unique property, which is set to True.
6. Scroll up further to find the Write-enabled property, which is also set to True.
7. This properties pane contains many more settings available for you to use in accordance with your business needs. The wizard guides you through the most commonly used ones.

###### 