BAZAT E TË DHËNAVE (HYRJE)

ACCESS

1.Çka është baza e të dhënave

Bazë e të dhënave është çfarëdo grumbulli informatash të organizuara në grup.Informatat duhet të jenë të organizuara ashtu që lehtë mund të ju qasemi .P.SH. notesi i numrave të telefonit është një bazë e të dhënave jo e kompjuterizuar.Është e organizuar me renditje alfabetike dhe përmban informacion për emrin,adresën dhe numrin e telefonit.Pra ne i qasemi një numri të telefonit në bazë të emrit.Në bazat e mëdha elektronike të cilat mirëmbahen në kompjuter na mundësohet që të manipulojmë më lehtë dhe më shpejtë me të dhënat.

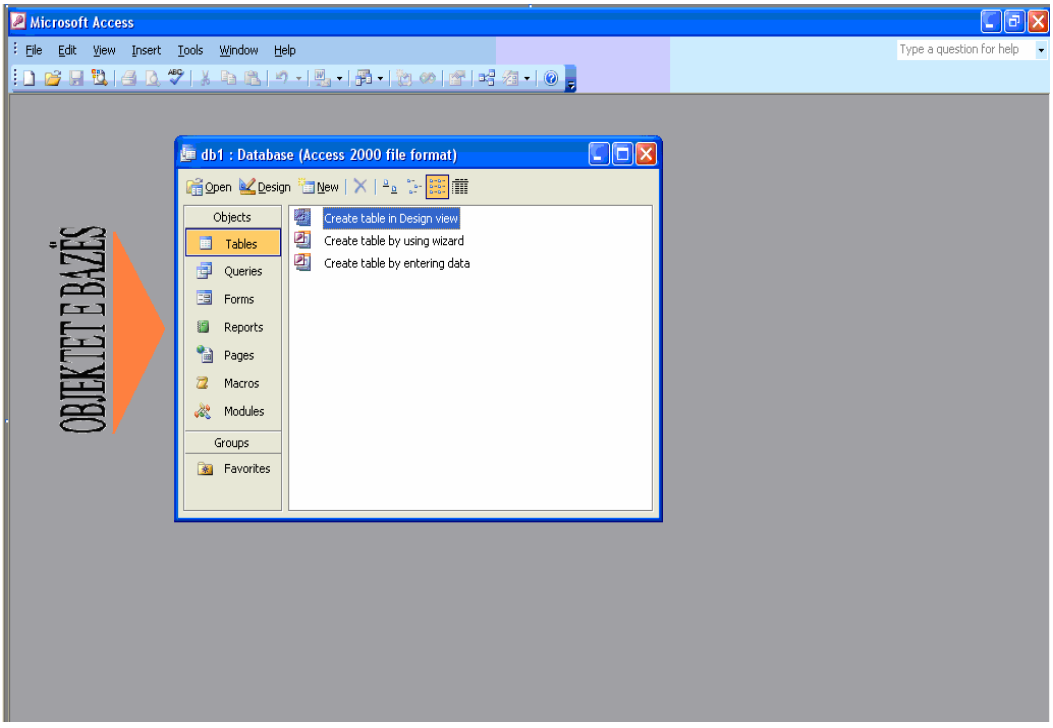
Programet që ngërthejnë në vete baza të të dhënave siç janë Microsoft Access ose Microsoft FoxPro, të shumtën e rasteve janë të tipit serverë për datoteka sepse më së shumti i'u dedikohen bazave që manipulohen vetëm nga një shfrytëzues, përderisa programet si Microsoft SQL Server, Oracle ose Informix janë të tipit klient/server sepse natyra e punës së tyre, më së shumti ndërtohet mbi parimin e punës me shumë shfrytëzuesë.

Së pari ushtrimet do t'i fillojmë me ACCESS dhe pastaj do të kalojmë në MYSQL.
ACCESS është një program i Microsoftit për baza të të dhënave, pra është pjesë e Microsoft Office.

2. Krijimi i një baze të të dhënave

Me krijimin e një baze të të dhënave në ACCESS ne krijojmë një vend për tabelat,format,pyetësorët,raportet si edhe për objektet tjera të bazës.Nëse krijojmë një bazë të re atëherë duhet që t'i krijojmë tabelat,pyetësorët dhe objektet tjera vetë.Në të njëjtën kohë mund të shfrytëzojmë Database Wizard për krijimin e bazës.

Me startimin e ACCESS-it klikojmë në New në toolbar pastaj klikojmë në Blank Database… i vëmë emrin asaj baze të të dhënave dhe e ruajmë, pastaj hapet dritarja:



Kjo dritare përmban **objektet e bazës të të dhënave** (në anën e majtë të dritares) :

a) Tabelat (Tables)
b) Pyetësorët (Queries)
c) Format (Forms)
d) Raportet (Reports)
e) Faqet (Pages)
f) Makrot (Macros)
g) Modulet (Modules)

Ndërsa në anën e djathtë është lista për krijimin e atyre objekteve.

Tipi i objektit	Përshkrimi
Tabela	Ky objekt definon strukturën e një baze në Access.Tabelat përmbajnë numër të madh të dhënash në rreshta dhe në kolona.Këto të dhëna mund të futen, ndryshohen, ruhen dhe të kthehen.
Pyetësori	Është mënyrë për kërkimin e informacioneve në tabelë.Kur ekzekutojmë një pyetësor, të dhënat paraqiten në të quajturin Recordset.Pastaj këto të dhëna mund të ndryshohen ose të shtypen (printohen).
Forma	Lejon futjen e të dhënave, shikimin si dhe ndryshimin e tyre.Forma mund të shfrytëzohet si alternativë për paraqitjen e të dhënave në rreshta dhe në kolona.
Raporti	Është dizajn për të dhënat që do të shtypen(printohen).Raportet përfshijnë raportet nga baza e të dhënave.Në raporte poashtu mund të kryhen llogaritje matematikore.
Faqja	Është një web faqe që mund t'i qasemi dhe të punojmë me bazën e të dhënave në Access përmes internetit dhe intranetit.
Makro	Është një varg i urdhërave të cilat ekzekutohen si një.
Moduli	Ky objekt përmban aplikacione të Visual Basic me të cilat mund të ndryshojmë funksionet në bazë të të dhënave.

Ushtrime

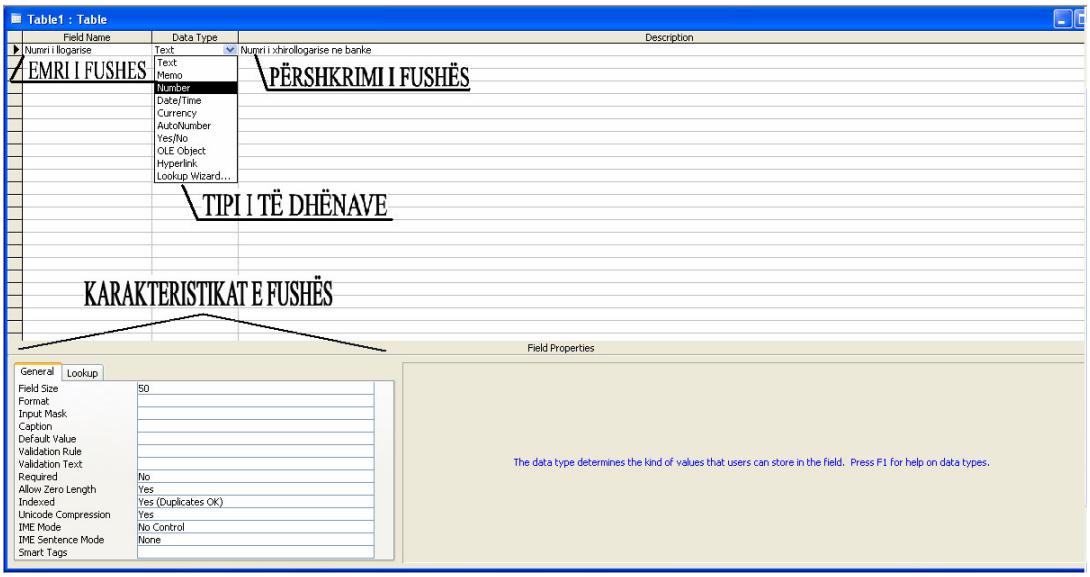
Krijoni një bazë të të dhënave me emrin BANKA.mdb.

Tabelat

Në Access janë pesë hapa për dizajnimin e tabelave.Hapi i parë përfshin kuptimin e sistemit duke përfshirë të dhënat që do të futen, raportet që do të nxirren, emrat si dhe çka do të gjenerohet nga të dhënat si dalje.Hapi i dytë përfshin përcaktimin e numrit të tabelave që duhen dhe informacioni që do të ruhet në ato tabela.Pra vendimi se sa tabela do t'i ketë baza e të dhënave është shumë i rëndësishëm.Hapi i tretë është për dizajnimin e tabelave duke vendosur se cilat fusha do t'i përmabjë, çfarë tipi i të dhënave do të futen në secilën fushë dhe madhësia e fushave.Hapi i katërtë përfshin emrin e tabelës dhe në fund në hapin e pestë testimi i strukturës së tabelës.Ky hap është shumë i rëndësishëm për faktin se na ndihmon të përcaktojmë se cilat fusha nuk janë të përfshira si dhe ndonjë gabim në madhësinë e tyre.Pastaj tabela mund të modifikohet.

Krijimi i tabelës në Design View

Tabela është temel i bazës të të dhënave.Të gjithë pyetësorët, raportet dhe format i shfrytëzojnë fushat dhe të dhënat nga tabela si bazë për dalje.Duhet që së paku të krijojmë një tabelë para se të krijojmë objektet tjera në bazë të të dhënave.Kur të krijojmë një tabelë të re duke shtypur New hapet një dialog ku mund të zgjedhim mënyrën e krijimit të tabelës: Datasheet View, Design View, Table Wizard, Import Table për importimin e të dhënave nga ndonjë burim dhe Link Table për lidhjen e të dhënave nga ndonjë burim.
Krijimi i tabelës në Design View na jep kontroll më shumë mbi dizajnimin e tabelës. Pra zgjedhim Design View dhe shtypim "OK" dhe hapet kjo dritare.



Dritarja në Design View është e ndarë në dy pjesë.Pjesa e lartë përmban vendin ku e shfrytëzojmë për futjen e emrit të fushës, tipit të të dhënavë në atë fushë si dhe përshkrimin e saj.Pjesa e poshtme e dritares përmban karakteristikën e fushës së zgjedhur.

Emri i fushës

Në pjesën e sipërme të Design View te Field Name mund të fusim emrin e fushës. Emri i fushës identifikon të dhënat në atë fushë si: ID, Emri, Mbiemri etj.Emrat e fushave mund të jenë të gjatë 64 karakter dhe të përmbajnë shkronja, numra dhe hapësira (por jo të fillojnë me hapësirë).Emri i fushës nuk mund të ketë pikë(.),pikëçuditëse(!), kuotë (`), kllapat e mesme ([ ]).Emrat e fushave duhet të jenë unik.

Caktimi i tipit të të dhënave

Secila fushë ka një tip të të dhënave.Tipi i të dhënave i tregon Access-it se çfarë vlera doni të ruani në atë fushë dhe sa hapësirë të rezervoj për atë fushë.Në fushën Data Type zgjedhim tipin e të dhënave.

Tabela e mëposhtme përshkruan tipet e të dhënave në dispozicion nga Access-i.

Tipi i të dhënave	Përshkrimi
Text	Text është një tip i të dhënave që përmban çdo kombinim të shkronjave, numrave, shenjave të pikësimit dhe hapësirës.Ku si mundësi e parazgjedhur është 50 karakter ndërsa maksimumi është 255 karakter.
Memo	Memo është e ngjashme me Text vetëm se Memo mund të përmbajë deri 65.535 karakter.Memo mund të shfrytëzohet për shënime dhe për përshkrime të gjata në bazë të të dhënave.
Number	Fusha Number mund të përmbajë vetëm karakter numerik, presjen (për mijëshe), pikën (për pikë decimale) dhe minus (vijë, për shenjë të numrave negativë).Number shfrytëzohet vetëm kur kemi të bëjmë me kalkulime me vlerat e fushave.P.SH. edhe pse zip kodi dhe numri i telefonit përmbajnë numra ato nuk do t'i përdorim për kalkulime.Për atë ato nuk duhen të jenë të tipit Number.
Date/Time	Date/Time përmbanë datën dhe/ose kohën.Access-i automatikisht vlerëson futjen e datës duke u siguruar se është valide.P.SH. Nëse dojmë të fusim datën 31/02/1999 Access-i nuk do ta lejoj sepse muaji Shkurt nuk i ka 31 ditë.Fusha Date/Time është e përdorshme për kalkulimin e datës dhe kohës.

Currency	Currency është e ngjashme me fushën Number dhe mund të përdoret për kalkulime.Numrat në këtë fushë paraqiten automatikisht me shenjën e dollarit.
AutoNumber	Tipi i të dhënave AutoNumber automatikisht cakton numrin unik(prej 1) secilës të dhënë.Nuk mundemi të fusim vlerë në një fushë ku më parë Access-i ka caktuar vlerën.Këtë tip të të dhënave e përdorim kur dojmë që të kemi identifikim unik për secilën të dhënë.
Yes/No	Fusha Yes/No shënon dy kondita PO dh JO.Ky tip i të dhënave përdoret kur ekzistojnë vetëm dy mundësi (P.SH. e saktë dhe jo e saktë) për vlerën e fushës.
OLE Object	Fusha OLE Object lidh atë fushë me një aplikacion të Windows-it.Tipi i të dhënave OLE Object përdoret për grafikë ose audio datoteka.
Hyperlink	Fusha Hyperlink shfrytëzohet për lidhjen me ndonjë faqe në internet, document të Word-it në intranet ose edhe formë në bazë të të dhënave.Hyperlink mund të përmbajë përshkrim, adresë dhe nën adresë.Secila pjesë është e ndarë me shenjën # dhe mund të përmbajë 2.048 karakter.
Lookup Wizard	Me zgjedhjen e tipit të të dhënave Lookup Wizard na udhëheq në krijimin e fushës Lookup.Fusha Looup mund të përmbajë listë të vlerave nga ndonjë tabelë ose listë të vlerave që i japim ne.

Përshkrimi i fushave

Në fushën Description shkruajmë përshkrimin për secilën fushë (Field Name).

Vënia e çelësit primar (Primary Key)

Access-i punon në mënyrë më efikase nëse vejmë çelësin primar në tabelë.Çelësi primar është një fushë ose një grup i cili në mënyrë unike identifikon çdo të dhënë.Për atë vlera në fushën ku është çelësi primar duhet të jetë unike në atë tabelë.Janë shumë përparësi me vënien e çelësit primar.Së pari çelësi automatikisht indeksohet gjë që e bën gjetjen e informacionit më shpejtë.Së dyti kur e hapim tabelën të dhënat sortohen sipas çelësit primar.Dhe në fund çelësi primar ndalon futjen e vlerave të dyfishta në fushën me çelës

primar.Më së lehti është që si çelës primar të vehet fusha me tipin e të dhënave AutoNumber.Çelësi primar nuk mund të vihet në fushat ku tipi i të dhënave është Memo, OLE ose Hyperlink.Për të krijuar çelës primar në më shumë se një fushë mbajmë pullën "CTRL" të shtypur dhe zgjedhim fushat që dëshirojmë pastaj klikojmë në pullën Primary Key.

Karakteristikat(tiparet) e fushës (FIELD PROPERTIES)

Secila fushë ka një varg tiparesh që kontrollojnë mënyrën se si ruhet në tabelë.
Vendosja e tipareve të fushës ndihmon në ndërtimin e një baze të qëndrueshme sepse ato do të përdoren për forma dhe raporte dhe për atë në hapat e mëtutjeshëm të dizajnimit të bazës do të kemi më pak punë.Normalisht vënia e tipareve të fushës bëhet kur të krijojmë tabelën me Design View.Tiparet e fushës janë në panelin Field Properties në Design View.Disa prej atyre tipareve janë të listuara në tabelën e mëposhtme.

Tipari i fushës	Përshkrimi
Field Size	Limiton numrin e karakterëve në fushën Text ndërsa te Number cakton intervalin në të cilin mund të jenë numrat
Format	Kontrollon të dhënat se si do të paraqiten në Datasheet View
Decimal Places	Numri i decimaleve pas pikës dhjetore
Input Mask	Përcakton modelin se si do të futen të dhënat, si p.sh. shenja – te numrat e telefonit
Caption	Saktëson emrin tjetër të fushës i cili do të paraqitet në tabelë, formë dhe raport
Default Value	Është vlerë e parazgjedhur nëse shfrytëzuesi nuk jep ndonjë vlerë për atë fushë
Validation Rule	Kufizon që të dhënat e futura t'i plotësojnë kushtet e caktuara. P.SH. caktojmë që në fushën **çmimi** vlera mos ta kaloj vlerën 25
Validation Text	Është teksti që do të paraqitet nëse thyhet rregulla Validation Rule
Required	Përcakton se fusha nuk duhet të lihet boshe (e pa plotësuar) kur të fusim të dhëna
Allow Zero Length	Përcakton nëse mund ti fusim të dhënat si thonjëzat (" ")
Indexed	Shpejton kthimin e të dhënave në fushë.Të gjithë çelësat primar automatikisht indeksohen

Formatet për tipin e të dhënave Field Size

Kur të caktojmë madhësinë e fushës (Field Size) mund të fusim të dhëna aq sa na lejojnë parametrat e vënë.Pasi të shtypim numrin maksimal të numrave nuk na lejohet më të fusim të dhëna në atë fushë.Madhësinë e fushës mund ta vëmë për Text, Number si dhe AutoNumber.Për Text thjesht shkruajmë numrin e dëshiruar të karakterëve që do të lejon

të futen në një fushë (numri maksimal 255 karakter).Për Number kemi disa opcione tjera të listuara në tabelën e mëposhtme.

Madhësia e fushës	Intervali	Numrat pas pikës dhjetore
Byte	Prej 0 deri 255	Asnjë, të dhënat rrumbullaksohen
Integer	Prej -32768 deri 32767	Asnjë, të dhënat rrumbullaksohen
Long Integer	-2,147,483,648 deri 2,147,483,647	Asnjë, të dhënat rrumbullaksohen
Single	-3.4×10^{38} deri 3.4×10^{38}	Deri në 7
Double	-1.797×10^{308}deri1.797×10^{308}	Deri në 15
ReplicationID	Identifikues unik global	Jo në dispozicion

Formatet për tipet e të dhënave Text dhe Memo

Access-i përdorë katër simbole për Format në Text dhe Memo:

@ tekst karakter (karakter ose hapësirë)

& nuk kërkohet tekst karakter

< të gjithë karakterët në atë fushë i konverton në shkronja të vogla

> të gjithë karakterët në atë fushë i konverton në shkronja të mëdha

Që emrat të paraqiten me shkronja të mëdha pra në Format shkruajmë >, ndërsa për tu paraqit emrat me shkronja të vogla përdorim <.
Për numrat e telefonit shruajmë @@@/@@@-@@@
Pra nëse shkruajmë 044123456 numri në fushë do të paraqitet 044/123-456.
Mund të krijomë edhe forma të tipit dypjesëshë.Këto dy pjesë ndahen me ";"(pikëpresje).Ku në pjesën e parë e shkruajmë formën se si do të jetë numri nëse fusim ndonjë të dhënë ndërsa në pjesën e dytë nëse nuk fusim ndonjë të dhënë.
P.SH. @@@/@@@-@@@;"I panjohur"[RED]
Pra numri i shkruar 044123456 do të paraqitet 044/123-456 ndërsa nëse nuk shkruajmë asgjë atëherë në atë fushë do të shkruhet **I panjohur** me ngjyrë të kuqe.

Formati: >&&&"-Q"@@@
Shkruhet: abc400
Paraqitet: ABC-Q400

Formatet për tipet e të dhënave Number dhe Currency

Mund të zgjedhim nga gjashtë formatet të paradefinuara të listuara në tabelën e mëposhtme:

Tipi	Numri i futur	Paraqitja e numrit	Formati
General	987654.321	987654.3	# # # # # # #. #
Currency	987654.321	$987,654.32	$# # #,# #0.00
Fixed	987654.321	987654.32	# # # # # #.# #
Standard	987654.321	987,654.32	# # #,# # #.# #
Percent	.987	98.7%	# # #.# #%
Scientific	987654.321	9.88E+05	# # #E+00
Euro	987654.321	€987,654.32	€# # #,# # #. # #

Shenja	Përshkrimi
. dhe ,	Pëdorimi i shenjës për ndarjen e shifrave dhjetore si dhe shenjës ndarëse për vlerat mijëshe.
0	Zëvendësuese për shifrat ose për 0
#	Zëvendësuese për shifrat ose për asnjë shenjë
%	Prezanton vlerën në përqindje

Por mund të krijojmë edhe formatet sipas dëshirës, për pjesën numerike formati është në katër pjesë ku pjesa e parë(1) është për numër pozitiv,pjesa e dytë(2) për numër negative, pjesa e tretë(3) për vlerën zero dhe pjesa e katërtë për vlerën boshe.
P.SH. €# #,# #0.00[GREEN];(€#,# #0.00)[RED];"ZERO";"BOSHE"
Ky format do t'i paraqes numrat pozitiv me ngjyrë të gjelbër, numrat negative me ngjyrë të kuqe, ku ka vlerë 0 do të shkruhet "ZERO" dhe ku nuk shkruhet asgjë do të jetë "BOSHE".

Shembuj

Në një tabelë kemi dy kolona njëra për emrin e mallit e tipit Text dhe tjetra për sasinë e atij malli e tipit Number.Çka duhet bërë që të dhënat në kolonën për sasinë e mallit të paraqiten në formën :

Malli	*Pesha*
Domate	30 kg
Dardha	45 kg
Mollë	50 kg

Duhet që në Design View për fushën **Pesha** te Format të shkruajmë **#“ kg”.**

Formatet për tipin e të dhënave Date/Time

Format	Përshkrimi
General Date	Nëse vlera është vetëm data atëherë ora nuk paraqitet.Nëse vlera është vetëm ora atëherë data nuk paraqitet
Long Date	Dita dhe muaji shkruhen sikur Tuesday, July 6, 2005
Medium Date	Është e tipit 06-Jul-2005
Short Date	7/6/2005
Long Time	Ora paraqitet me orë, minuta dhe sekonda të ndara me dy pika dhe e pasuar me AM ose PM sikur 6:30:15 PM
Medium Time	Është e njëjtë si Long Time vetëm se sekondat nuk paraqiten
Short Time	Ora paraqitet në formatin 24-orësh pa sekonda sikur 18:30

:	Ndarës i kohës si p.sh. 12:34:23
/	Ndarës i datës si p.sh. 12/06/2004
c	Njësoj si General Date
d, dd	Dita e muajit-një ose dy shifra numerike (1-31)
ddd	Tri shkronjat e para të ditëve (Sun-Sat)
dddd	Emri i plotë i ditëve
ddddd	Njësoj si Short Date
dddddd	Njësoj si Long Date
w	Dita e javës (1-7)
ww	Vikendi i vitit (1-53)
m, mm	Muaji i vitit (1-12)
mmm	Tri shkronjate para të muajit (Jan-Dec)
mmmm	Emri i plotë i muajit (January-December)
q	Data e shfaqur si qerek i vitit (1-4)
y	Dita e vitit (1-366)

yy	Dy shifart e fundit të vitit
yyyy	Viti i plotë (0100-9999)
h, hh	Ora- një ose dy shifra numerike (0-23)
n, nn	Minutat-një ose dy shifra numerike (0-59)
s, ss	Sekondat-një ose dy shifra numerike (0-59)
ttttt	Njësoj si Long Time
AM/PM ose A/P	Ora (1-12) me shkronja të mëdha
am/pm ose a/p	Ora (1-12) me shkronja të vogla
AMPM	Ora e definuar me Windows Regional Settings

Shembuj

Të paraqitet data në formën :

1. dita/muaji/viti
2. tri shkronjat e para të ditëve
3. si qerek
4. emri i plotë i muajit

Fomatet për Yes/No

Në tipin e të dhënave Yes/No mund të definojmë formatet edhe sipas dëshirës. Formati në tipin e të dhënave ndahet në tri pjesë.Ku pjesa e parë nuk ndonjë efekt por duhet që gjithnjë të jetë shenja ; (pikëpresje), pjesa e dytë përdoret për vlerën On ose True, pjesa e tretë përdoret për vlerën Off ose False.P.SH. nëse dojmë që të shkruajmë se një student është **i pranishëm** dhe **jo i pranishëm**, kjo shkruhet kështu **;"i pranishëm";"Jo i pranishëm"**.Nëse dëshirojmë që t'i paraqesim vlerat me ngjyra atëherë shkruajmë: **;"I pranishëm"[RED];"Jo i pranishëm"[GREEN]**.

Formatet për tipin e të dhënave Hyperlink

Formati për këtë tip të të dhënave ndahet në tri pjesë:

- Teksti që shfaqet në fushë
- Adresa, shtegu deri te datoteka (UNC) ose web faqe (URL) në internet
- Nënadresa, lokacion specific në datotekë ose faqe.

Pjesët ndarëse ndahen mes veti me simbolin #. P.SH.

Web faqja e Microsoft Net#http://www.msn.com

Input Mask

Për tu siguruar se të dhënat do të futen në mënyrë të duhur krijojmë Input Mask.
Input Mask na lejon të definojmë me anë të kriterit se si do të futen të dhënat në fushë.

0	Numër (0-9; hyrja e detyrueshme;[+] dhe [-] nuk lejohen)
9	Numër ose hapësirë (hyrja jo e detyrueshme; [+] dhe [-] nuk lejohen)
#	Numër ose hapësirë (hyrja e jo e detyrueshme, [+] dhe [-] lejohen)
L	Shkronjë (A-Z, hyrja e detyrueshme)
?	Shkronjë (A-Z, hyrja opcionale)
A	Shkronjë ose numër (hyrja e detyrueshme)
a	Shkronjë ose numër (hyrja opcionale)
&	Çfarëdo karakteri ose hapësirë (hyrja e detyrueshme)
C	Çfarëdo karakteri ose hapësirë (hyrja e opcionale)
<	Konverton të gjithë karakterët pas saj në shkronja të vogla
>	Konverton të gjithë karakterët pas saj në shkronja të mëdha
!	Detyron input mask të plotësohet nga e majta në të djathtë.
****	Paraqet karakterin pas saj p.sh. \A paraqet A
. , : ; - /	Pika dhjetore,për mijëshe,ndarësit e datës dhe kohës

Megjithatë më lehtë për të vënë Input Mask është me Input Mask Wizard.
Ku aty na jepen mundësi të shumta dhe të gatshme.

Shembuj

Input Mask	Shkrimi	Paraqitja
>LL000	af345	AF345
000 000 00	50540130	505 401 30
90/90/0000	01121997	01/12/1997

Decimal Places

Tipari i fushës Decimal Places është valid për të dhënat numerike dhe Currency. Numri i decimaleve mund të jetë prej 0 deri në 15 varësisht nga madhësia e fushës numerike ose Currency. Nëse fusha është Byte, Integer ose Long Integer do të kemi 0 vende decimale. Nëse fusha është Single atëherë kemi prej 0 deri në 7 vende decimale,nëse fusha është Double atëherë kemi prej 0 deri në 15 vende decimale.Nëse definojmë fushën si Currency atëherë Access-i e vën numrin e decimaleve në 2.

Validation Rule dhe Validation Text

Tipet Date/Time, Number dhe Yes/N/ të fushave që në instalimin standard të programit kanë të integruara funksionet e shqyrtimit të vlefshmërisë së të dhënave.Kështu për shembull`në qoftë se do të shkruani vlerën 31.02.05 në fushën e datës apo një shifër në një fushë numerike, do të pasojë fjë njoftim për shkrim të ga`uaz dë të dhënave.Opsionet Validation Rule dhe Validation Text të fushave ofrojnë mundësi të tjera për të kufizuar hedhien e gabuar të të dhënave në bazën e të dhënave.

- **R2egull vlefshmërie** mund të specifikoni për çdo fushë të një tabele.Me këtë përcaktim kufizohet ndieshëm hedhja e gabuar e të dhënave.Në opsionet Validation Rule mund të formuloni njoftimin e gabimit.
- Për të informuar përdoruesin e bazës së të dhënave në rast të shkeljes të rregullit të vërtetësisë mund të përpiloni **njoftime** përkatëse për çdo fushë.Njoftimet e vërtetësisë realizohen në opsionet Validation Text të fushave përkatëse.

Operator krahasimi, vlerë krahasimi

- Si operator krahasues mund të përdorni shenjat krahasuese **(= < >)** si dhe operatorin krahasues **LIKE**
- Vlera e krahasimit për kufizimin e vlerave të të dhënave në rastin e përdorimit të operatorit **LIKE** mund të përmbajë edhe shenjë zëvendësuese**.**
- **N**ë rast se për një fushë ju duhet të përcaktoni më shumë se një rregull, mund të përdorni operatorin lidhës **AND/OR** dhe **NOT/BETWEEN.**
- Si vlera krahasimi mund të përdoret edhe rezultati i një funksioni, si për shembull rezultati i funksionit të dates aktuale **DATE().**

Shembuj

Number	>=5
Date/Time	<#12.01.2005#
Text	>”A”
Yes/No	<>Yes

LIKE "#*"	Shkrimi mund të fillojë me një shifër dhe më pas mund të vijojë me një numër çfarëdo shenjash të ndryshme
LIKE "#####"	Lejohet të shkruhen vetëm numra pesëshifrorë
LIKE "[XY]"	E dhëna që shkruhet duhet të përmbajë një X ose një Y
LIKE "[A-Z]"	Lejohet të shkruhen vetëm shkronjat e mëdha të alfabetit
"Veturë" or "Shtëpi"	>=Date()
Between 3 AND 100	LIKE "[1-9]" AND LIKE "[!6]"
NOT Between 3 AND 100	>#1.1.1850# AND <Date()

Ruajtja e tabelës

Pas dizajnimit të tabelës duhet që atë ta ruajmë duke i dhënë emër.Emri mund të përmbajë deri 64 karakter duke përfshirë edhe hapësirën.Këta karakter mund të jenë shkronja, numra dhe hapësira.Emri nuk duhet të përmbajë pikë (.), pikë çuditëse (!), kuota (`), kllapa të mesmë ([ ]) .Me ruajtjen e tabelës nuk krijohet një datotekë e re por i shtojmë një objekt datotekës së bazës të të dhënave.

Krijimi i tabelës me Table Wizard

Një ndër mënyrat e krijimit e tabelave është edhe Table Wizard (magjistari i tabelës). Me anë të Table Wizard i tërë krijimi i tabelës bëhet me anë të modeleve dhe shablloneve të tipit biznes ose personal.

Ushtrime

Në bazën e të dhënave **BANKA** krijoni tabelën **KLIENTËT** me këto fusha:
Numri i llogarisë
Emri
Mbiemri
Qyteti
Adresa
Tel

Si dhe definoni tipin e të dhënave për këto fusha.Shkruani përshkrimin për secilën fushë dhe si çelës primar të jetë fusha Numri i llogarisë.

Në bazën e të dhënave **BANKA** krijoni tabelën **KREDITË** me këto fusha:
ID
Numri I llogarisë
Tipi i kredisë
Shuma
Përqindja
Kohëzgjatja e kredisë
Aprovimi
Fillimi
ID Nënpunësit
Si dhe definoni tipin e të dhënave për këto fusha.Shkruani përshkrimin për secilën fushë dhe si çelës primar të jetë fusha ID.

Në bazën e të dhënave **BANKA** krijoni tabelën **NËNPUNËSIT** më këto fusha:
ID Nënpunësit
Emri
Mbiemri
Filiala
Si dhe definoni tipin e të dhënave për këto fusha.Shkruani përshkrimin për secilën fushë dhe si çelës primar të jetë fusha ID Nënpunësit.

Datasheet View

Tabela hapet në Datasheet View.Datasheet View është e përshtatshme për futjen, përmisim ose fshirje të të dhënave në tabelë.
Të dhënat janë të radhitura në rreshta dhe shtylla.Kalimi nga një e dhënë te tjetra bëhet me

në fund të tabelës.

Shtimi i të dhënave bëhet në Datasheet View.

Ushtrime

I shtoni këto të dhëna në tabelën KLIENTËT:

Numri i llogarisë	*Emri*	*Mbiemri*	*Qyteti*	*Adresa*	*Tel*
140201	Bujar	Shulemaja	Prishtinë	Bregu i Diellit	044/123-456
140202	Amir	Simnica	Fushë Kosovë	rr. Agim Ramadani	044/321-654
140203	Valbona	Krasniqi	Dardanë	rr. Adem Jashari	044/213-546
140204	Gani	Thaçi	Besianë	rr. Zahir Pajaziti	044/312-645
140205	Lavdim	Kastrati	Prishtinë	Tophane	044/132-465

I shtoni këto të dhëna në tabelën KREDITË:

ID	*Numri i llogarisë*	*Tipi i kredisë*	*Shuma*	*Përqindja*	*Kohëzgjatja e kredisë*	*Aprovimi*	*Fillimi*	*ID Nën*
1	140203	Veturë	4500	6	12	po	10/04/ 2006	3
2	140205	Mobile	1300	7.5	6	po	23/08/ 2006	1
3	140201	Veturë	7000	5.5	36	po	15/02/ 2005	1
4	140203	Shtëpi	45000	4.7	120	jo	20/05/ 2005	3
5	140202	Kompjuter	630	7	12	po	12/07/ 2006	4
6	140204	Kuzhinë	1000	4	12	po	04/09/ 2005	2
7	140205	Lavatriçe	420	3	6	po	05/06/ 2006	1
8	140201	Mobile	1730	5.8	12	po	02/10/ 2006	1
9	140205	Veturë	5250	7.3	18	po	12/04/ 2005	1
10	140202	Tavolinë	420	8.3	12	jo	12/08/ 2005	4

I shtoni këto të dhëna në tabelën NËNPUNËSIT:

ID Nënpunësit	*Emri*	*Mbiemri*	*Filiala*
1	Astrit	Kabashi	Prishtinë
2	Valon	Ramadani	Besianë
3	Faton	Gashi	Dardanë
4	Albana	Gashi	Fushë Kosovë

Ndryshimi i gjërësisë dhe gjatësisë së kolonave gjegjësisht rreshtave

Gjërësia standarde e kolonës është 15.6667.Këtë gjërësi mund ta ndryshoni duke shkuar te menyja kryesore te **Format** dhe te **Column Width.**
Lartësia standarde e rreshtave është 12.75.Këtë lartësi mund ta ndryshoni duke shkuar te menyja kryesore te **Format** dhe te **Row Height**.

Ndryshimi i llojit të shkronjave, qelisë.Lëvizja e kolonës, fshehja e kolonës dhe ngrirja e kolonës

Lloji i shkronjave ndërrohet duke shkuar në Format dhe Font, aty mund të zgjedhim llojin e shkronjave, stilin dhe madhësinë.
Lëvizja e kolonës mund ta bëjmë në Datasheet View, Design View.Selektojmë kolonën dhe e tërheqim deri te vendi ku dëshirojmë.
Fshehja dhe ngrirja e kolonës bëhet në menynë kryesore te Format dhe te Hide Columns (Unhide Columns) përkatësisht Freeze Columns (Unfreeze All Columns).

Filter by selection

Mënyra më e thjeshtë dhe e shpejtë e filtrimit të të dhënave është me anë të Filter By Selection ku mund të zgjedhim fushën e cila ka të dhënë me anë të cilës dojmë t'i filtrojmë të dhënat.Me shfrytëzimin e Filter By Selection zgjedhen të dhënat të cilat plotësojnë kriterin e dhënë me filter.

Filter excluding selection

Filter Excluding Selection është e njëjtë sikur Filter By Selection vetëm se këtu paraqiten të dhënat që nuk plotësojnë kriterin e dhënë nga filteri.

Filter for

Me klikimin me të djathtë fushën në kolonën e cila përmbanë të dhënën me të cilën dojmë të filtrojmë, te Filter For shkruajmë kriterin dhe shtypim Enter.

Filter by form

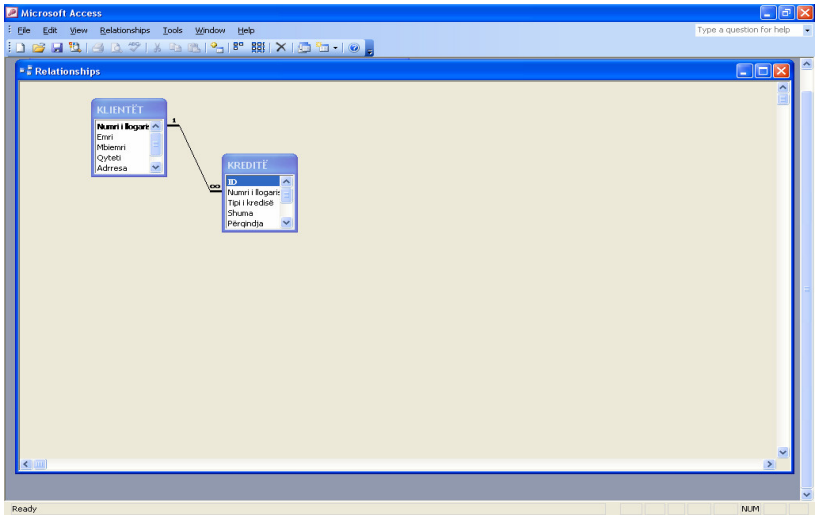
Me zgjedhjen e Filter By Form në dritare paraqitet një listë.Lista përmbanë të dhënat ekzistuese në atë fushë, ashtu që lehtë mund të zgjedhim se me cilën të dhënë dojmë të filtrojmë të dhënat.Ky opcion është i përdorshëm shkaku që nuk kemi nevojë të mbajmë mend të dhënat nëpër fusha.

LIDHJET E TABELAVE (RELATIONSHIP)

Tabelat mund të lidhen mes vete në mënyrë që të kemi qasje më të shpejtë të informacionit, evitimit të futjes të të dhënave të dyfishta, krijimit të pyetësorëve, formave dhe raporteve.Pastaj lidhja na mundëson krijimin e tabelave më të vogla që janë më efikase kur duhet të nxjerrim të dhëna nga to.Kur të bëhet lidhja e dy tabelave duhet që fusha në tabelën e parë të jetë çelës primar të cilin e tërheqim deri te fusha e tabelës së dytë.Pra në tabelën e parë fusha duhet të jetë çelës primar ashtu që të evitohet mundësia e futjes së të dhënave të dyfishta.Fushat që lidhen në të dy tabelat duhet të jenë të të njëjtit tip ose të ngjashëm, nëse tipi i të dhënave është Number ato duhet të kenë të njëjtën madhësi të fushës.
P.SH. Mund të krijojmë një tabelë që përmbanë emrat e klientëve, adresat dhe numrat e telefonave.Poashtu mund të vejmë çelës primar për secilin klientë.Pastaj mund të krijojmë një tabelë tjetër ku do të jenë porositë e bëra nga klientët.Kjo tabelë poashtu mund të ketë një fushë si unike për çdo klientë por jo emrin e klientit, adresën ose numrin e telefonit.Me lidhjen e këtyre dy tabelave nuk është e nevojshme që pas çdo porosie të futet emri, adresa, telefoni i klientit.
Access-i përfshin dy lidhje bazike: një me një dhe një me shumë.Lidhja një me një është kur një e dhënë nga tabela e parë përkon me një të dhënë në tabelën e dytë.Lidhja një me shumë është kur një e dhënë në tabelën e parë përkon me shumë të dhëna në tabelën e dytë.Access-i përcakton vetë tipin e lidhjes.

KRIJIMI I LIDHJES

Krijimi i lidhjeve bëhet në dritaren e lidhjeve(figura e mëposhtme).Hapim bazën e të dhënave, klikojmë në pullën Relationships  pastaj në pullën Show Table  .Zgjedhim tabelën e parë dhe shtypim Add poashtu veprojmë edhe me tabelën e dytë.Pastaj i lidhim fushat nga tabela e parë në të dytën duke tërhequr fushën nga tabela e parë e cila përkon me fushën në tabelën e dytë.Në fund zgjedhim Create dhe e mbyllim dritaren me Close.

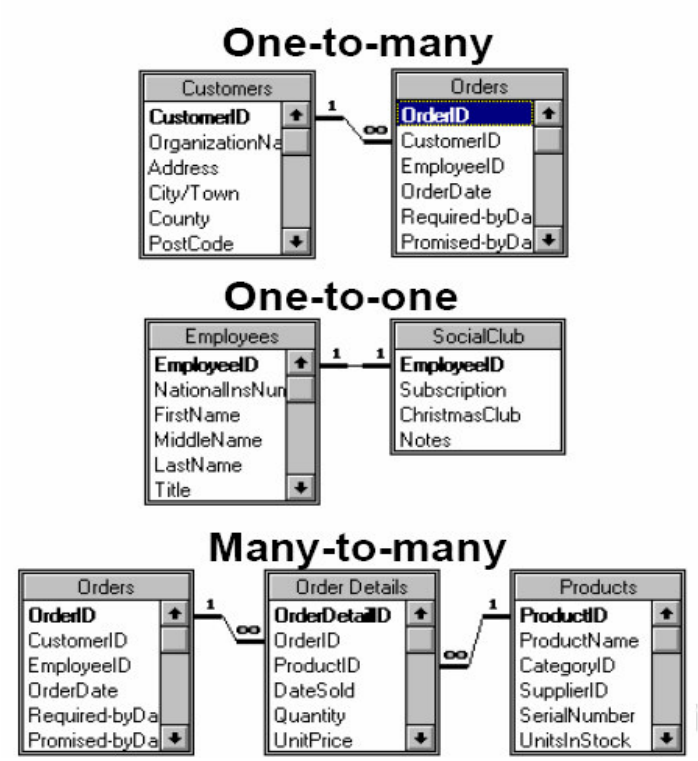


Vënia e integritetit referencial (Referential Integrity)

Me krijimin e lidhjes në mes të dy tabelave mund të vejmë integritetitn referencial.Integriteti referencial është një bashkësi e rregullave të cilin Access-i e përdorë për vërtetimin e vlefshmërisë së lidhjes.Integriteti referencial poashtu ndalon fshirjen ose ndryshimin aksidental të të dhënave.Për të shfrytëzuar Integritetin referencial duhet të plotësohen kushtet vijuese:fusha e lidhur nga tabela e parë duhet të jetë çelës primar, fushat në të dy tabelat duhet të jenë të të njëjtit tip të të dhënave si dhe ti takojnë të së njëjtës bazë.Me vënien e Integritetit referencial nuk mund të fusim të dhëna në tabelën e dytë nëse nuk përkon me fushën në tabelën e parë.Nuk mund të fshihen të dhënat prej tabelës së parë nëse ndonjë e dhënë në tabelën e dytë përkon me atë dhe së fundi nuk mund të ndryshoni vlerën çelësit primar në tabelën e parë nëse ekzistojnë të dhëna të lidhura.
Megjithatë nëse dojmë të ndryshojmë të dhënat dhe sërish të kemi integritetin referencial atëherë duhet që të zgjedhim opcionet edhe **Cascade Update Related Fields** dhe **Cascade Delete Related Records.**Kur të zgjedhen këto dy opcione atëherë Access-i automatikisht mirëmbanë integritetin referencial.

USHTRIME

Krijoni lidhjen e tabelës KLIENTËT me tabelën KREDITË.



PYETËSORËT

Fjala QUERY (PYETËSORË) rrjedh prej fjalës latine quærere, që do të thotë pyetje ose kërkesë.Pyetësori në Access është pyetja që bëhet për informacionin që gjendet në tabelë.Pyetësori mund të jetë i thjeshtë, të kërkoj të dhëna vetëm nga një tabelë dhe mund të jetë i përbërë të kërkoj të dhëna në shumë tabela në dallim nga filterët që u përdorën te tabelat, ku të dhënat mund të nxirren nga vetëm një tabelë.
Pyetësori nuk përmban të dhëna, ai përmban instruksione të cilat Access-i i përdorë për të nxjerr të dhënat që u përgjigjen atyre instruksioneve.Prandaj kur të shtojmë ndonjë të dhënë në tabelë nuk kemi nevojë që të bëjmë ndryshime edhe në pyetësor.Të dhënat që paraqiten pas ekzekutimit të pyetësorit quhen **bashkësi e të dhënave** (**Recordset**).

Mundësitë me pyetësorë

- **Zgjedhja e tabelës.** Mund të nxjerrim informacion prej një tabele ose shumë tabelave.
- **Zgjedhja e fushës.** Mund të caktojmë se cilën fushë nga tabela dëshirojmë ta shohim në bashkësinë e të dhënave.
- **Sortimi i të dhënave.** Me këtë na mundësohet që të dhënat t'i shohim në një renditje të caktuar.
- **Kalkulimet.** Mund të shfrytëzojmë pyetësorët për të bërë kalkulime në të dhëna.
- **Krijimin e tabelave.** Mund të shfrytëzojmë për krijimin e tabelave nga dhënat si rezultat i një pyetësori.
- **Krijimi i formave dhe rapoteve.** Mund të krijojmë forma dhe raporte, ku rëndësia e kësaj është se sa herë që hapet forma ose printohet raporti, pyetësori do të kthejë të dhënat më të freskëta nga tabelat.
- **Krijimi i grafeve.** Mund të krijojmë grafe nga rezultati i pyetësorit, të cilin rezultat mund ta shfrytëzojmë në forma dhe raporte.
- **Krijimin e nënpyetësorëve.** Mund të krijojmë pyetësor duke shfrytëzuar rezultatin nga një pyetësor tjetër.

Tipet e pyetësorëve

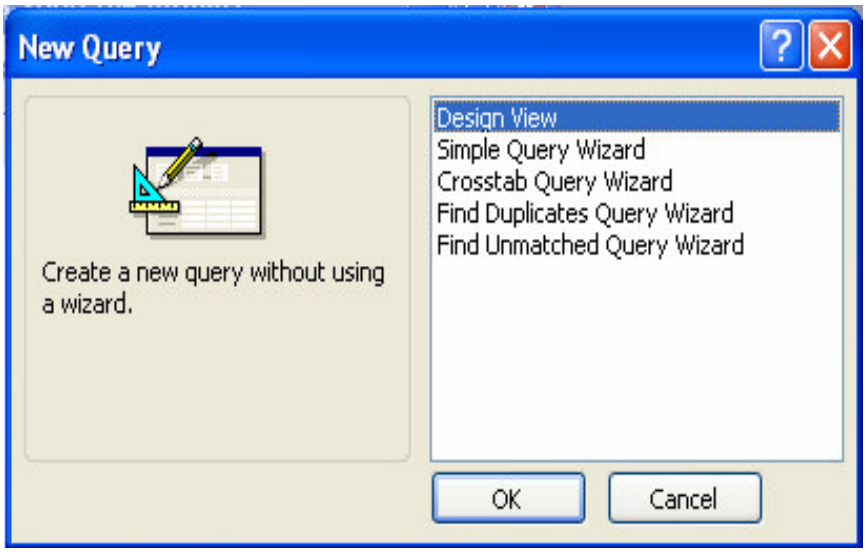
- **Select.** Ky është tipi më i zakonshëm i pyetësorëve.Sikurse tregon edhe vet emri ky pyetësor selekton të dhënat nga një ose më shumë tabela (bazuar në ndonjë kriter), duke krijuar bashkësinë e të dhënave dhe paraqitjen e saj në Datasheet.
- **Total.** Këto janë versione speciale të pyetësorëve **Select.** Pyetësorët **Total** mundësojnë për të mbledhur të dhënat sipas një kriteri (si p.sh. group by, count etj).
- **Action.** Ky tip i pyetësorëve na mundëson të krijojmë tabela te reja (make table query) ose ndryshimin e të dhënave (delete, update dhe append) në tabelat ekzistuese.

- **Crosstab.** Ky tip i pyetësorëve grupon të dhënat me metoda të ndryshme përllogaritjeje.
- **SQL.** Pyetësorët Structured Query Language krijohen me shkruarjen e komandave specifike SQL.
- **Top(n).** Ky pyetësor mund të shfrytëzohet vetëm së bashku me pesë tipet e pyetësorëve të mëparshëm.Na mundëson të specifikojmë numrin e përqindjes të të dhënave që dëshirojmë ti shohim.

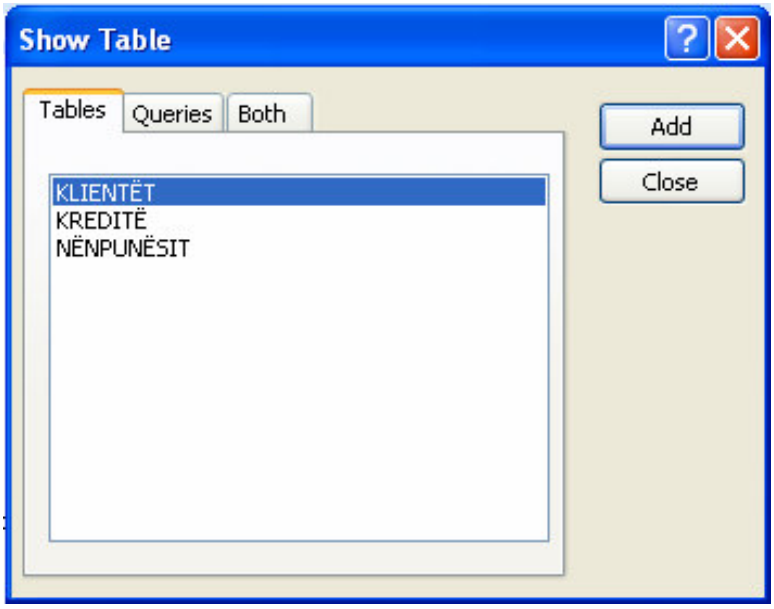
Krijimi i pyetësorit

Pas krijimit të tabelave dhe vendosjes së të dhënave në to, atëherë mund të punojmë me pyetësor.Për krijimin e një pyetësori duhet përcjell këta hapa:

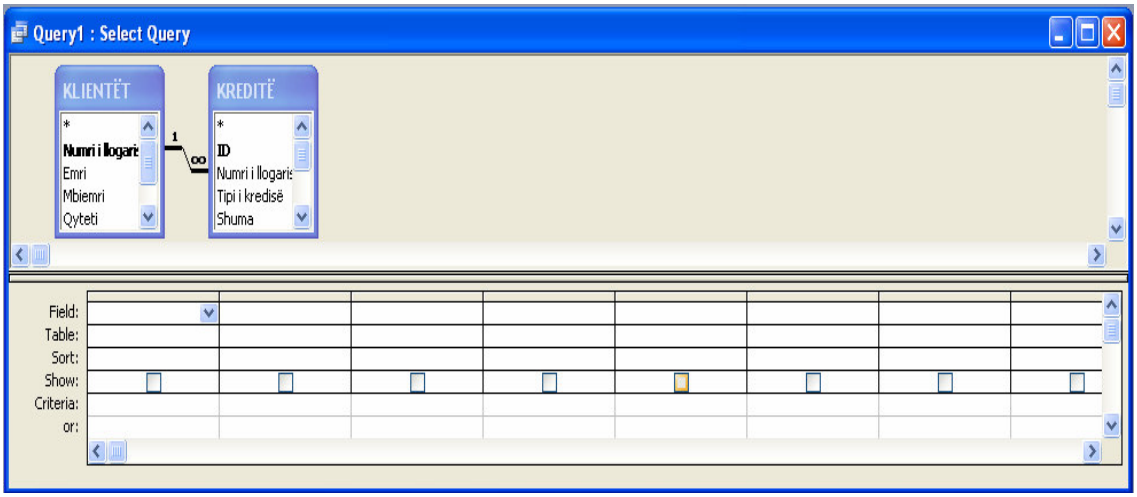
1. Zgjedhim objektin **Queries** në dritaren e bazës së të dhënave
2. Klikojmë në pullën **New** dhe hapet dritarja, ku mund të zgjedhim pesë mënyra të krijimit të pyetësorit.Zgjedhja e parë është **Design View**



3. Zgjedhim **Design View** dhe klikojmë në pullën **OK.**
4. Pastaj na paraqitet dritarja **Show Table** ku janë të gjitha tabelat dhe pyetësorët (nëse ka) të krijuar më parë.

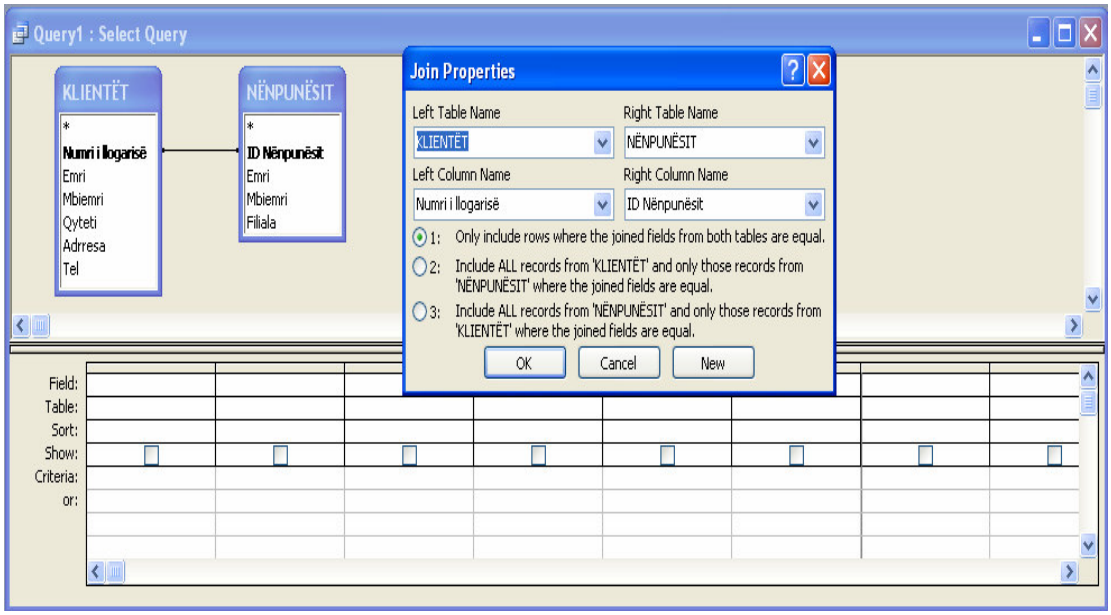


5. Zgjedhim tabelën dhe klikojmë në pullën **Add** (ose vetëm klikojmë dy herë). Këtë mund ta përsërisim edhe me zgjedhjen e tabelave tjera.
6. Në fund klikojmë në pullën **Close** dhe hapet dritarja e pyetësorit e cila përbëhet prej dy pjesëve.Pjesa e parë ku janë tabelat dhe lidhja e tyre dhe pjesa e dytë përfshin fushat, tabelën, sortimin, kriterin e pyetësorit që quhet zona e dizajnimit të pyetësorit.



Nëse kemi zgjedhur më shumë se një tabelë atëherë ato duhet të jenë të lidhura ashtu që të na japin rezultat të saktë dhe domethënës.Nëse tabelat nuk janë të lidhura atëherë si rezultat do të paraqiten të gjitha kombinimet e të dhënave në ato dy tabela.P.SH. nëse tabela e parë ka 20 të dhëna ndërsa tabela e dytë ka 5 të dhëna atëherë rezultati do të ketë 100 të dhëna, që nuk kanë ndonjë domethënie.
Nëse tabelat janë të lidhura në mes veti në dritaren e lidhjeve (Relationship window) atëherë lidhja në mes tyre do të shihet automatikisht edhe në Design View të pyetësorit (si në figurën më lartë).

Nëse tabelat nuk janë të lidhura në dritaren e lidhjeve (Relationship window) atëherë duhet që ato tabela t'i lidhim në dritaren e pyetësorit.Nga tabela e parë zgjedhim fushën dhe e tërheqim deri te fusha e tabelës së dytë me të cilën do të krijojmë lidhjen.Atëherë do të paraqitet një vijë lidhëse në mes të këtyre fushave, nëse klikojmë dy herë në atë vijë do të hapet një dritare ku mund të përcaktojmë edhe lidhjen në mes të dy tabelave.



Përcaktimi i lidhjes në mes të dy tabelave mund të bëhet në tri mënyra:

1. Relacioni përjashtues (ekskluzion)	Përfshin të gjitha të dhënat te të cilat fushat e lidhura janë të barabarta
2. Relacion-majtas përfshirës (inkluzion)	Përfshin të gjitha të dhënat nga tabela e parë edhe në qoftë se në tabelën e dytë për ato të dhëna nuk ndonjë të dhënë.
3. Relacion-djathtas përfshirës (inkluzion)	Përfshin të gjitha të dhënat nga tabela e dytë edhe në qoftë se në tabelën e parë për ato të dhëna nuk ndonjë të dhënë.

Lidhja që definohet në dritaren e pyetësorit nuk do të paraqitet në dritaren e lidhjeve (Relationship window).
Access-i bën lidhjen në mes të dy tabelave automatikisht nëse në të dy tabelat ekzistojnë fusha me emër të njëjtë.
Pjesa e dytë e dritares së pyetësorit përmban në vete emrat e fushave,emrin e tabelës, sortimin si dhe kriteret me anë të të cilave do të kërkohen të dhënat në tabela ose pyetësorë.

Përfshirja e fushave në pyetësor

Përfshirja e fushave në pyetësor bëhet duke klikuar dy herë në atë fushë ose tërheqjen e fushës nga tabela deri te zona e dizajnimit të pyetësorit. Nëse dëshirojmë që t'i bartim vetëm disa fusha në pyetësor atë mund ta bëjmë duke i zgjedhur fushat në tabelë duke mbajt të shtypur tastin **CTRL** dhe duke klikuar në fushat e dëshiruara dhe pastaj i tërheqim deri te zona e dizajnimit të pyetësorit.Nëse dëshirojmë që të gjitha fushat e tabelës të jenë në pyetësor atëherë klikojmë dy herë në * në tabelë ose e tërheqim * deri te zona e dizajnimit të pyetësorit.

Pas selektimit të fushave mund të bëjmë ndërrimin e renditjes, fshirjen, riemërimin (vetëm në pyetësorë), sortimin,fshehjen dhe vënien e kriterit të tyre.

- Ndërrimi i renditjes në zonën e dizajnimit të pyetësorit bëhet duke selektuar fushën dhe tërheqjen e saj në pozitën e dëshiruar.
- Fshirja e fushës në zonën e dizajnimit të pyetësorit bëhet duke selektuar fushën e dëshiruar dhe shtypim tastin **Delete**.
- Riemërimi i fushës në zonën e dizajnimit të pyetësorit bëhet në rreshtin **Field** në zonën e dizajnimit të pyetësorit duke shkruar te fusha e dëshiruar emrin e ri pastaj **:** dhe emrin e fushës.P.SH. Dëshrojmë të ndërrojmë emrin e fushës F1 me F2, **F2:F1**
- Kur të shikojmë rezultatin e pyetësorit mund të dëshirojmë që të dhënat të jenë të renditura (sortuara) sipas një rregulle për analizim sa më të lehtë të tyre.Sortimi bëhet sipas rregullit alfabetik ose numerik (në rritje A deri te Z dhe 0 deri te 9, në zbritje Z deri te A dhe 9 deri te 0).Sortimi mund të bëhet edhe me disa fusha, në atë rast sortimi do të bëhet nga e majta në të djathtë.
- Fshehja e fushave bëhet në zonën e dizajnimit të pyetësorit në rreshtin **Show** duke çaktivuar kutizën e kontrollit.
- Kriteret e një pyetësori formulohen në zonën e dizajnimit të pyetësorit në rreshtin **Criteria.** Kriteret thjeshtë janë rregulla të cilat Access-it i tregojnë se cilat të dhëna dëshirojmë t'i shohim.

Ruajtja e pyetësorit

Pyetësori ruhet duke klikuar në  në toolbar ose File dhe Save, pastaj shkruajmë emrin e dëshiruar për pyetësorin në dritare dhe shtypim OK.

Ekzekutimi i pyetësorit

Ekzekutimi i pyetësorit bëhet duke shtypur në toolbar ikonën  ose Query dhe Run.

Detyrë:

1. Krijoni një pyetësor me anë të tabelës **KLIENTËT** ku do të përfshihen të gjitha fushat e asaj tabele.Bëni ndërrimin e renditjes së fushave **Emri** dhe **Mbiemri**, riemëroni fushën **Tel** me **Telefoni,** sortoni fushën **Emri** sipas renditjes alfabetike **A** deri te **Z,** sortoni fushën **Emri** sipas renditjes alfabetike **A** deri te **Z** dhe fushën **Mbiemri** sipas **Z** deri te **A,** fshehni fushat **Mbiemri** dhe **Adresa,** fshini fushën **Adresa.**Në fund ruani pyetësorin me emrin **Test**.

2. Krijoni një pyetësor nga tabelat **KLIENTËT** dhe **KREDITË** ku do të përfshihen fushat **Numri i llogarisë, Emri, Mbiemri** nga tabela **KLIENTËT** dhe **Tipi i kredisë, Shuma** nga tabela **KREDITË**.Në fund ruani pyetësorin me emrin **Kreditë e klientëve**.

3. Krijoni një pyetësor ku do të përfshihet emri i klientit, mbiemri i klientit si dhe numri i telefonit të atij klienti.Në fund e ruani pyetësorin me emrin **Tel i klientit**.
4. Krijoni një pyetësor nga tabela **KREDITË** ku do të përfshihen të gjtha fushat.Pyetësorin e ruani me emrin **Test1**.

Kriteret me përfshirje të emrit të objektit, datës, orës dhe teksteve

Kur duhet të përdorim emrat e fushave apo vlerat e datës ose orës për formulimin e kushteve duhet të respektojmë sintaksën e mënyrës së shkrimit.Për këtë duhet të kemi parasysh këto rregulla:

	Sintaksa	Shpjegimi	Shembull
Objektet	[Emri]	Emrat e fushave, raporteve apo formularëve duhet të shkruhen në kllapa katrore.	[Çmimi]*1.6
Data, Ora	#Data#	Data dhe ora duhet të shkruhen brenda shenjës #. Si rregull Access-i është në gjendje të dallojë formatin e datës dhe, mbas shkrimit të një date, ai e vendos atë automatikisht midis shenjave #.	#12.04.05#
Teksti	"Tekst"	Tekstet duhet t'i shkruajmë gjithmonë midis thonjëzave.Edhe në këtë rast Access-i ka aftësinë t'i dallojë tekstet dhe i fut ato automatikisht midis thonjëzave.	"Ushtrimet laboratorike"

Operatori i krahasimit LIKE

Me ndihmën e operatorit LIKE mund të gjejmë nëse fushat e tipit tekst kanë në përmbajtjen e tyre shenja të caktuara.Për shembull ju kërkohet të hartoni listën e të gjithë klientëve emri i të cilëve fillon me shkronjat A deri në D. Kriteri i përzgjedhjes formulohet duke përdor shenjat zëvendësuese.

Shenja	Efekti	Shembull
*	I përgjigjet një numri të çfarëdoshëm shenjash	LIKE "K*" të gjithë personat emri i të cilëve fillon me shkronjën K. LIKE "[A-D]*" të gjithë klientët emri i të cilëve fillon me shkronjën A deri D. LIKE "[AZ]*" të gjithë klientët emri i të cilëve fillon me shkronjën A ose Z. LIKE "[!A]*" të gjithë klientët emri i të cilëve fillon me të gjitha shkronjat pos shkronjës A.
?	Zëvendëson vetëm një shenjë të çfarëdoshme	LIKE "???M" të gjithë emrat që përbëhen nga katër shkronja dhe ku shkronja e katërtë është M. LIKE "??S*" shkronja e tretë duhet të jetë S.
#	Zëvendëson një shifër të çfarëdoshme	LIKE "#*" përmbajtja e fushës duhet të filloj me një shifër dhe mund të ketë çfarëdo gjatësie. LIKE "####" të gjitha përmbajtjet e fushës që përbëhen nga katër shifra.
&	Lidh dy stringje mes veti si një	[Emri]&" "&[Mbiemri]

Operatorët logjikë	Sintaksa	Efekti
AND	Kushti1 AND Kushti2	Të dyja kushtet duhet të plotësohen
OR	Kushti1 OR Kushti2	Të paktën njëri kusht duhet të plotësohet
NOT	NOT Kushti	Ky kusht nuk lejohet të plotësohet
BETWEEN	BETWEEN Vlera1 AND Vlera2	Vlera e kësaj fushe duhet të jetë midis vlerës1 dhe vlerës2

Kriteri për fushën e tipit të të dhënave Yes/No

Një vlerë –Jo mund ta gjejmë duke përcaktuar si kusht No, False, Off ose 0.Për kërkimin e një vlere –Po përcaktojmë si kusht Yes, True, On ose -1.

Operatorët matematikë

Operatori	Përshkrimi	Shembull
*	Realizon shumëzimin e numrave	[Çmimi]*[Sasia] ose 2*[Çmimi]
+	Realizon mbledhjen e numrave	[Çmimi]+[Shpenzime_Transport] ose 11+45
-	Realizon zbritjen e numrave ose shërben si parashenjë e një numri	[Çmimi]-[Tatimi] ose [Çmimi]-([Çmimi]*0.1)
/	Realizon pjestimin e numrave	[Sasia]/[Pesha] ose [Shuma]/5
^	Realizon ngritjen në fuqi të një numri me eksponentin e dhënë	[Gjatësia]^2
Mod	Mbetja në numër të plotë e pjestimit të dy numrave	5 Mod 2 rezultati 1

Operatorët relacional

Operatori	Përshkrimi
=	Operatori: i barazimit [Tipi i kredisë]=”Veturë”
<>	I ndryshëm [Tipi i kredisë]<>”Veturë”
>	Operatori: më i madh [Shuma]>20000
>=	Operatori: më i madh ose baraz [Shuma]>=20000
<	Operatori: më i vogël [Shuma]<20000
<=	Operatori: më i vogël ose baraz [Shuma]<=20000

Operatorët me përparësi

Kur kemi të bëjmë me shprehje komplekse që kanë shumë operatorë,Access-i duhet të përcaktoj se cili operator ka përparësi.Për atë Access-i e ka të paradefinuar prioritetin e kategorive të operatorëve sipas kësaj renditje:

1. Operatorët matematikë
2. Operatorët relacional
3. Operatorët logjikë

Secila kategori ka rendtitjen e përparësisë të operatorëve.

Prioriteti te operatorët matematikë

1. Fuqia
2. Negacioni
3. Prodhimi dhe/ose Pjestimi (nga e majta në të djathtë)
4. Pjestimi i numrave të plotë
5. Moduli
6. Mbledhja dhe/ose zbritja (nga e majta në të djathtë)
7. Lidhja e Stringjeve

Prioriteti te operatorët relacional

1. Baras
2. Jo Baras
3. Më e vogël
4. Më e madhe
5. Më e vogël ose baras
6. Më e madhe ose baras
7. LIKE

Prioriteti te operatorët logjikë

1. NOT
2. AND
3. OR

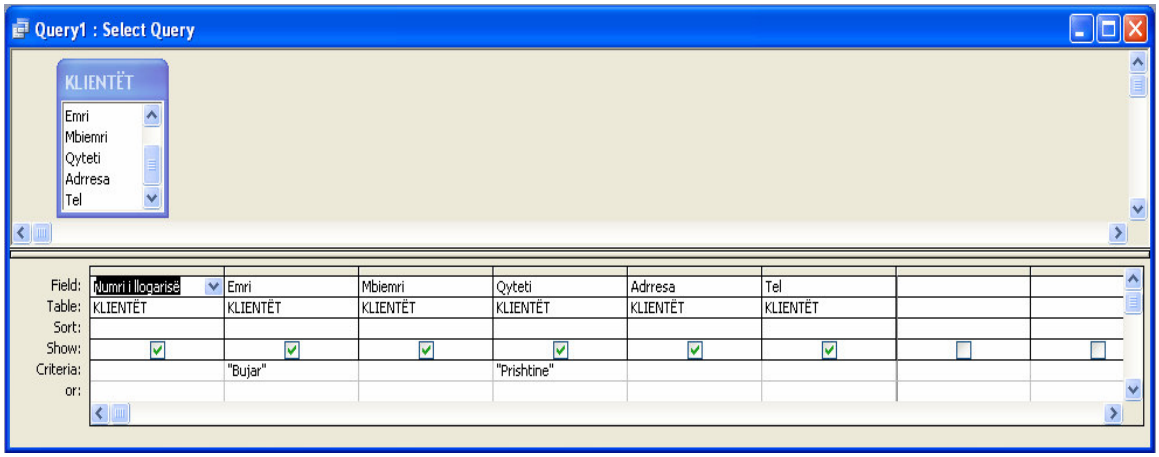
Shembuj:

LIKE “M[A]*” Kthen vlerën e saktë nëse shkronja e parë është M e dyta A.
LIKE “[!e-zE-Z]” Kthen vlerën e saktë nëse shkronjat janë A,B,C,D,a,b,c ose d.
LIKE “AB####” Kthen vlerën e saktë nëse fillon me shkronjat AB e shoqëruar me katër shifra.
LIKE “[#]*A” Kthen vlerën e saktë nëse fillon me # dhe përfundon me A.
LIKE NOT “M[A]*” Kthen vlerën e pasaktë nëse shkronja e parë është M dhe e dyta A.

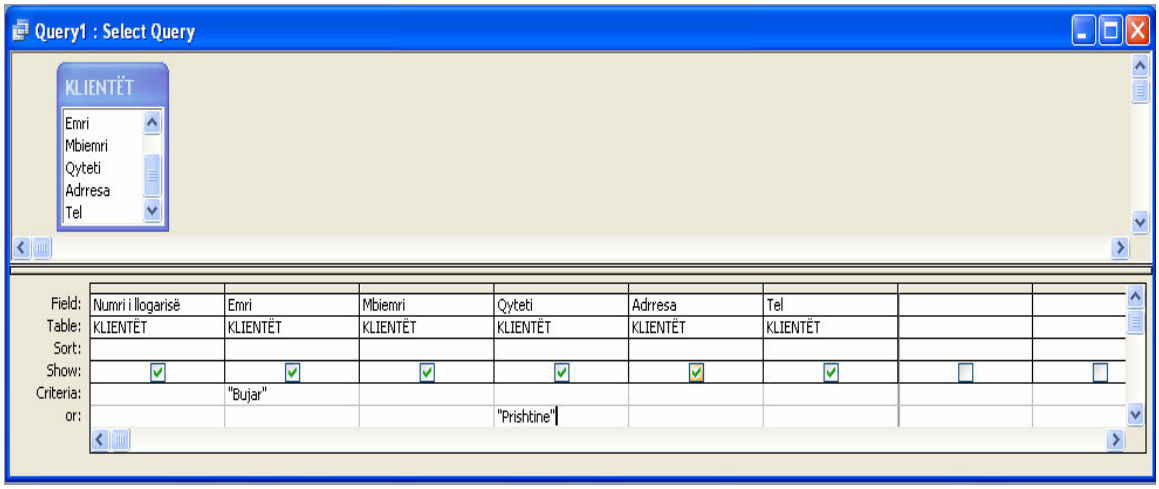
Pyetësor me më shumë se një kriter

Në Access kemi mundësi të formulojmë edhe më shumë se një kriter në fusha dhe rreshta të ndyshëm.Në këtë rast duhet të kemi parasysh këto veçori:

1. Lidhjet **AND**
   Kushtet e një rreshti lidhen ndërmjet veti me ndihmën e operatorit **AND.**Në këtë rast të dhënat përfshihen në rezultatin e ekzekutimit të pyetësorit vetëm kur janë të plotësuara të gjitha kushtet e formuluara.
   Shembull: Të gjithë personat me emrin Bujar **dhe** qyteti Prishtinë do të shfaqen në rezultatin e pyetësorit.



2. Lidhjet **OR**
   Kur të formulojmë kushte në dy rreshta ato lidhen në mes veti me anë të operatorit **OR.** Në këtë rast të dhënat përfshihen në rezultatin e ekzekutimit të pyetësorit atëherë kur plotësohet të paktën njëri prej kushteve të formuluara.
   Shembull: Të gjithë personat me emrin Bujar **ose** qytetin Prishtinë do të shfaqen në rezultatin e pyetësorit.



Detyrë:

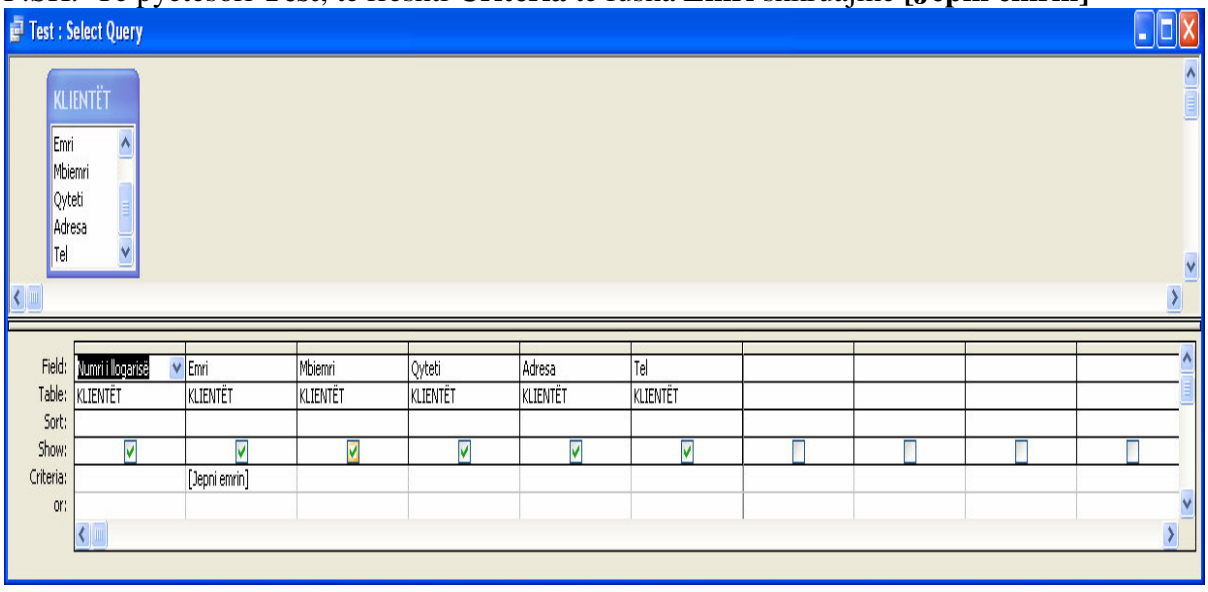
1. Modifikoni pyetësorin **Test1** ashtu që si rezultat të fitojmë listën e tipit të kredive **Veturë ose Shtëpi** dhe që janë të aprovuara.
2. Në pyetësorin **Test1** nxirrni si rezultat shumën e kredisë që kalon **20000**.
3. Modifikoni pyetësorin **Test** ashtu që si rezultat të paraqiten emrat e personave që fillojnë me **A** dhe përfundojnë me **A** ose **I** .
4. Me modifikimin e pyetësorit **Test1** të nxirren nga tabela **KREDITË** të gjitha kreditë që janë dhënë në vitin 2005.
5. Me modifikimin e pyetësorit **Test** nga tabela **KLIENTËT** të nxirren të dhënat për personat që jetojnë në **Prishtinë**.
6. Të krijohet një pyetësor me anë të cilit do të fitojmë si rezultat emrin dhe mbiemrin e klientëve që kanë marrë në kredi kompjuter si dhe emrin,mbiemrin dhë ID e nënpunësit që e ka lejuar atë kredi.

Mundësitë e pyetësorëve me parametra

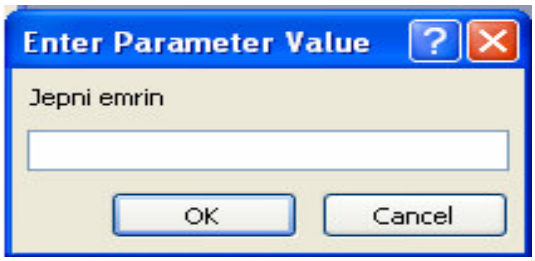
Shpesh ndodh që një pyetësor duhet formuluar në një mënyrë të tillë që vlera bazë e kërkimit të jetë fleksibile, pra që ajo të përcaktohet me fillimin e ekzekutimit të pyetësorit.Gjatë ekzekutimit të një pyetësori me parametër shfaqet në ekran një dritare dialogimi ku mund të japim vlerën ose kriterin e pyetësorit.Access-i pastaj e përdor këtë kriter (parametër) dhe ekzekuton pyetësorin.Në bashkësinë e të dhënave (Recordset) do të paraqiten të gjitha të dhënat që plotësojnë kriterin e dhënë në fillim.

Krijimi pyetësorit me një parametër

Pas hapjes së pyetësorit në **Design View** dhe përfshirjes së fushave në pyetësor, në rreshtin **Criteria** te ndonjë fushë e caktuar shkruajmë tekstin i cili do të paraqitet në dritaren e dialogimit i futur në kllapa të mesme **[teksti]**.
P.SH. Te pyetësori **Test**, te rreshti **Criteria** te fusha **Emri** shkruajmë **[Jepni emrin]**

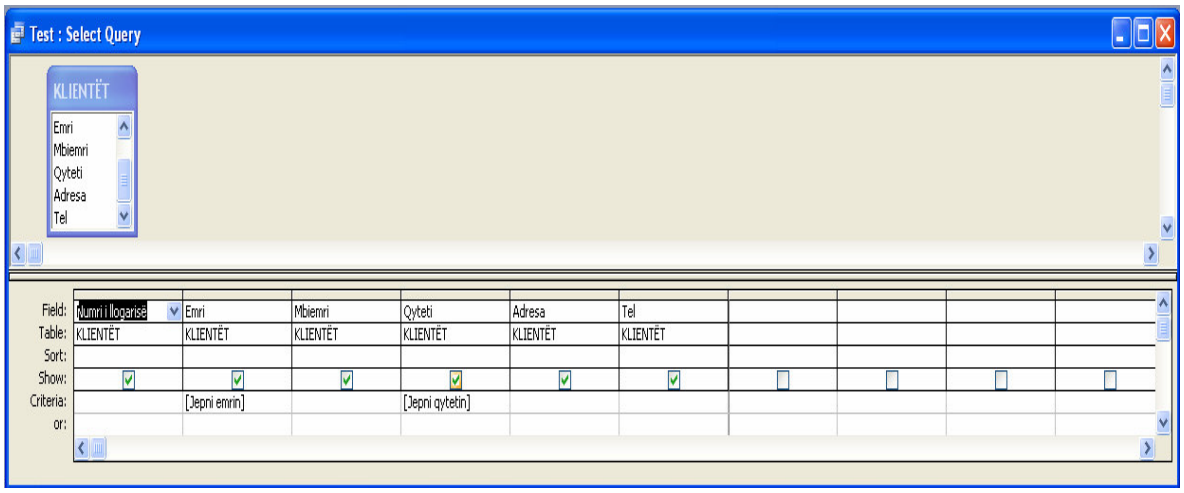


Pas ekzekutimit të pyetësorit na paraqitet dritarja e dialogimit, ku do të shkruajmë emrin e ndonjë klienti që e kërkojmë.



Krijimi pyetësorit me më shumë se një parametër

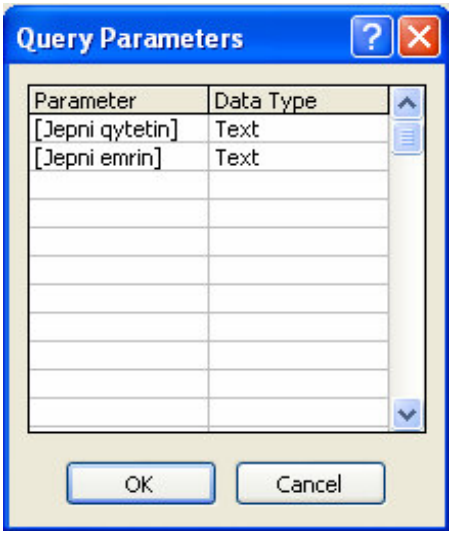
Nuk jemi të kufizuar në krijimin e pyetësorit me vetëm një parametër.Mund të krijojmë pyetësor me më shumë se një parametër.P.SH. dëshirojmë që nga tabela KLIENTËT të marrim si rezultat të gjithë klientët me emrin Bujar dhe qytetin Prishtinë.Pyetësori me më shumë se një parametër krijohet njësoj si me një parametër vetëm se te rreshti **Criteria** shkruajmë më shumë se një kriter.



Kur të ekzekutohet pyetësori së pari Access-i kërkon për kriteret në këtë renditje:

1. Jepni emrin
2. Jepni qytetin

Pra Access-i kërkon për parametrat nga e majta në të djathtë.Por këtë renditje mund t'a ndryshojmë, në Design View te pyetësori shkojmë te Query dhe Parameters… dhe hapet një dritare.Në atë dritare mund të rregullojmë renditjen.



Krijimi i një fushe kalkuluese

Fushat në pyetësor nuk janë të kufizuara në fusha vetëm nga tabela.Ne poashtu mund të krijojmë fusha ku mund të bëjmë kalkulime.P.SH. dëshirojmë që nga fusha Çmimi të kalkulojmë në një fushë tjetër tatimin 15%(TVSH).Tatimi:[Çmimi]/7.6666
Ose

Fusha1	Fusha2	Fusha3
23	45	7

Fusha4:[Fusha1] + [Fusha2] + [Fusha3]
Rezulati do të jetë 75.

Shtimi dhe ndryshimi i të dhënave me anë të pyetësorit

Pyetësori mund të përdoret edhe për shtimin dhe ndryshimin e të dhënave në tabela.Gjithmonë mund të ndryshojmë të dhënat në një tabelë dhe te tabelat e lidhura mes veti me lidhjen një me një, ndërsa te tabelat e lidhura mes veti me lidhjen një me shumë nuk mundemi gjithnjë të shtojmë ose të bëjmë ndryshime.

Si bëhet shtimi ose ndryshimi i të dhënave

Hapim pyetësorin në Datasheet View dhe klikojmë në pullën  që gjendet në fund të pyetësorit.Pastaj shtypim të dhënat që dëshirojmë ose bëjmë ndryshimin e ndonjë të dhëne dhe në fund i ruajmë ndryshimet e bëra.

Filtrimi i të dhënave në pyetësor

Filtrimi bëhet njësoj si te tabelat, pra hapet pyetësori në Datasheet View dhe shkohet njësoj si te tabelat.

Tipet e pyetësorëve

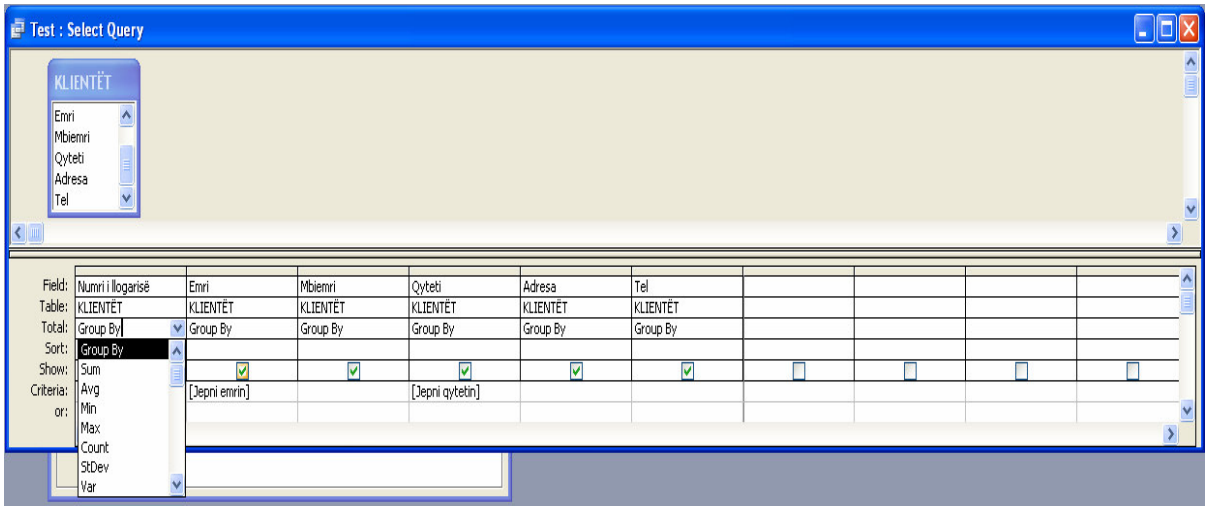
Siç u cek më herët tipet e pyetësorëve janë:**Select, Total, Action, Crosstab, SQL dhe Top(n)**.Tipi i pyetësorit **Select** është tip i zakonshëm i pyetësorëve.Me krijimin e pyetësorit në Design View si tip i parazgjedhur nga Access-i është tipi **Select.**Pra e gjitha çka u tha më lartë për krijimin pyetësorit në Design View është krijimi i pyetësorit **Select**. Tip tjetër me radhë është tipi Total.

Tipi Total

Shumë herë dëshirojmë që informacionin e nxjerrur nga tabelat të jetë i grupuar sipas një fushe apo fushave të asaj tabele.

Krijimi i pyetësorit Total

Në Design View të pyetësorit në menynë kryesore shkojmë te View dhe zgjedhim Totals ose në toolbar shtypim pullën  .Do të shohim se në zonën e dizajnimit të pyetësorit do të paraqitet rreshti Total.Nëse klikojmë në rreshtin Total do të shohim një listë të funksioneve të cilat përdoren për grupimin e informacioneve dhe përmisimin e tyre.



Ky përmisim i rezultateve të pyetësorit arrihet me ndihmën e funksioneve.Access-i i përdor katër kategori të funksioneve ato janë:

Kategoritë e funksioneve		
Kategoritë	**Numri i funksioneve**	**Qëllimi i funksionit**
Group by	1	Grupon të dhënat e përbashkëta.
Funksionet (Aggregate)	9	Funksione matematikore
Shprehje (Expression)	1	Grupon një numër të funksioneve dhe i ekzekuton si një
Kufizimi total i të dhënave në fushë	1	Kufizon të dhënat nga kalkulimet e bëra nga operatorët tjerë

Kategoria Group by

Ka një opcion, opcionin Group by.Përdoret për caktuar grupimin e të dhënave, të një fushe.P.SH. Nëse zgjedhim që të dhënat nga tabela KREDITË të grupohen sipas tipit të kredisë.Ky opcion automatikisht është i parazgjedhur kur të përfshihen fushat në pyetësor.

Kategoria Expression (Shprehja)

Edhe kjo kategori si kategoria Group by ka një opcion: **Expression.**Ky opcion përdoret për t'i treguar Access-it që të krijojë një fushë kalkuluese.

Kategoria Kufizimi total i të dhënave në fushë (Total field record limit category)

Edhe kjo kategori ka vetëm një opcion, opcionin **Where.**Kur të zgjedhim këtë opcion ne i tregojmë Access-it se dëshirojmë të caktojmë një kufizim në atë fushë.

Kategoria Funksionet (Aggregate)

Për dallim nga kategoritë tjera kjo kategori ka nëntë opcione:**Sum, Avg, Min, Max, Count, StDev, Var, First** dhe **Last.** Secili prej funksioneve përdoret për kalkulime në fusha dhe kthimin e rezultatit në bashkësinë e të dhënave.P.SH. dëshirojmë të definojmë maksimumin dhe minimumin e shumës së kredisë në tabelën KREDITË.
Për dallim nga kategoritë e përmendura më parë të cilat mund të shfrytëzohen për çfarëdo tipi të të dhënave, kategoria Funksionet mund të përdoret vetëm në disa tipe të të dhënave të paraqitura në tabelën e mëposhtme:

Kategoria fuksionet		
Funksioni	**Gjen**	**Tipi i të dhënave që përkrah**
Count	Numrin e vlerave joboshe në një fushë	AutoNumber,Number,Text,Currency,Date/Time, Yes/No,Memo,OLE
Sum	Total i vlerave në një fushë	AutoNumber,Number,Currency,Date/Time, Yes/No
Avg	Vlera mesatare në një fushë	AutoNumber,Number,Currency,Date/Time, Yes/No
Max	Vlerën më të madhe në një fushë	AutoNumber,Number,Currency,Date/Time, Yes/No,Text
Min	Vlera më e vogël në një fushë	AutoNumber,Number,Currency,Date/Time, Yes/No,Text
StDev	Devijimi standard i vlerave të një fushe	AutoNumber,Number,Currency,Date/Time, Yes/No
Var	Variacioni i vlerave të një fushe	AutoNumber,Number,Currency,Date/Time, Yes/No
First	Vlerën e të dhënës së parë në tabelë ose pyetësorë	AutoNumber, Currency,Date/Time,Yes/No, Text,Memo,OLE
Last	Vlerën e fundit nga e dhëna e fundit në tabelë ose pyetësorë	AutoNumber, Currency,Date/Time,Yes/No, Text,Memo,OLE

Me pyetësorin **Total** mund të bëjmë kalkulime me të gjitha fushat në tabelë ose pyetësor.P.SH. dëshirojmë të gjejmë numrin e kredive të aprovuara nga **BANKA**.
Shkojmë te **Queries** dhe zgjedhim **Create query in Design View**, zgjedhim tabelën **KREDITË** dhe i bartim fushat **Tipi i kredisë** dhe **Aprovimi** në zonën e dizajnimit të pyetësorit.Shtypim pullën  në toolbar.Te rreshti Total te fusha **Tipi i kredisë** zgjedhim funksionin **Count** ndërsa te fusha **Aprovimi** zgjedhim **Group by.**Këtë pyetësor ruajeni me emrin **Kreditë e aprovuara**.Siç shihet si rezultat fitojmë numrin e kredive të aprovuara dhe të pa aprovuara.Për të fituar vetëm kreditë e aprovuara duhet të vihet kriteri në rreshtin **Criteria** që do të shohim pak më vonë.

Ndonjë herë duhet që kalkulimet të bëhen në të dhënat e një fushe jo në tërë fushën si më parë.P.SH. dëshirojmë të gjejmë emrin dhe mbiemrin e të gjithë klientëve që kanë marrë kredi në bankën tonë.Shkojmë te **Queries** dhe zgjedhim **Create query in Design View**, zgjedhim tabelat **KLIENTËT** dhe **KREDITË** dhe i bartim fushat **Numri i llogarisë** nga tabela **KREDITË** dhe krijojmë një fushë me emrin dhe mbiemrin e klientit pra:

Emri dhe mbiemri:KLIENTËT.Emri&" "&KLIENTËT.Mbiemri

Shtypim pullën  në toolbar.Te rreshti Total te fusha **Numri i llogarisë** zgjedhim funksionin **Count** ndërsa te fusha **Emri dhe Mbiemri** zgjedhim **Group by.**Këtë pyetësor ruajeni me emrin **Kreditë e klientëve.** Siç po shihni me ekzekutimin e këtij pyetësori do

të na paraqiten emrat dhe mbiemrat e personave që kanë marrë kredi në bankë si dhe numri i kredive të tyre.Pra kemi bërë grupimin sipas emrave dhe mbiemrave të klientëve.

Grupimi i të dhënave mund të bëhet edhe në më shumë se një fushë.P.SH. dëshirojmë që nga pyetësori **Kreditë e klientëve** të bëjmë grupimin edhe sipas tipit të kredisë.Pra bartim fushën **Tipi i kredisë** në këtë pyetësor dhe te rreshti Total zgjedhim **Group by.**E ruajmë këtë pyetësor me emrin **Kreditë e klientëve-shtesë**.

Definimi i kriterit për pyetësorin Total

Në mënyrë që grupimi i të dhënave për pyetësorin Total të mund të limitohet përdoret kriteri në rreshtin Criteria.Ky kriter mund të përdoret në të tri fusha:

1. Group by
2. Funksionet (Aggregate)
3. Jo funksionet (Non aggregate)

Definimi i kriterit për fushën Group by

Për të kufizuar të dhënat në grupim definojmë kriterin në fushën Group by.P.SH. dëshirojmë që nga pyetësori **Kreditë e klientëve-shtesë** të marrim si rezultat vetëm kreditë si Veturë dhe Shtëpi.Shkojmë te **Queries** dhe hapim pyetësorin **Kreditë e klientëve-shtesë** në Design View.Te fusha **Tipi i kredisë** dhe rreshti **Criteria** shkruajmë: **In ("Veturë","Shtëpi")** dhe e ruajmë pyetësorin.Kur të ekzekutohet pyetësori tani si rezultat do të paraqiten vetëm kreditë e dhëna si Veturë dhe Shtëpi.

Definimi i kriterit për fushën e Funksioneve

Ndonjë herë është e nevojshme që pyetësori të kalkulon së pari me **Funksionet** mbi ndonjë fushë të caktuar dhe pastaj të paraqes si rezultat vetëm ato të dhëna që e plotësojnë kriterin e dhënë.P.SH. dëshirojmë të shohim kreditë ashtu që vlera mesatare e kredisë të jetë më e madhë se 1000.Shkojmë te **Queries** klikojmë dy herë në **Create query in Design View** dhe zgjedhim tabelën **KREDITË**, pastaj bartim fushat **Tipi i kredisë** dhe **Shuma** në zonën e dizajnimit të pyetësorit.Shtypim pullën  në toolbar.Te rreshti Total te fusha **Tipi i kredisë** zgjedhim opcionin **Group by** ndërsa te fusha **Shuma** zgjedhim funksionin **Avg** (funksioni për vlerën mesatare) dhe te rreshti **Criteria** shkruajmë **>1000**.Si rezultat do të paraqiten të dhënat kreditë vlera mesatare e të cilave kalon 1000.Ruajeni këtë pyetësor me emrin **Vlera mesatare e kredive.**

Definimi i kriterit për Where (Non aggregate)

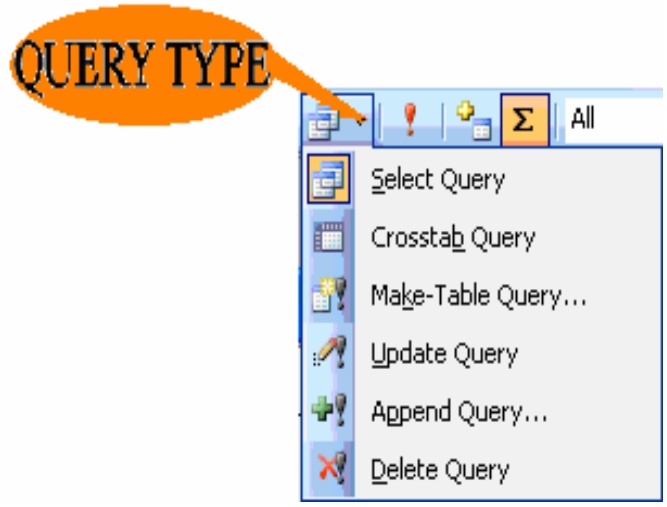
Më herët pamë se si kufizuam të dhënat pasi bëmë kalkulimet, por ne poashtu mund të kufizojmë të dhënat para se të bëjmë kalkulimet.P.SH. dëshirojmë të dijmë kreditë që janë lëshuar në një periudhë të caktuar kohore në tabelën KREDITË.Krijojmë një pyetësor nga tabelat KREDITË dhe KLIENTËT.Bartim fushat Numri i llogarisë, Emri nga tabela KLIENTËT dhe fushën Fillimi nga KREDITË.Shtypim pullën  në toolbar dhe te rreshti Total te fusha Numri i llogarisë zgjedhim funksionin Count, te fusha Emri zgjedhim Group by.Dhe në fund te fusha Fillimi rreshti Total zgjedhim Where ndërsa te rreshti Criteria shkruajmë kriterin BETWEEN #12/8/2005# AND #10/4/2006#.Si rezultat do të na paraqiten të gjitha kreditë e dhëna në këtë periudhë kohore.Ruajeni pyetësorin më emrin Kreditë.

Krijimi i Shprehjes (Expression) në pyetësorin Total

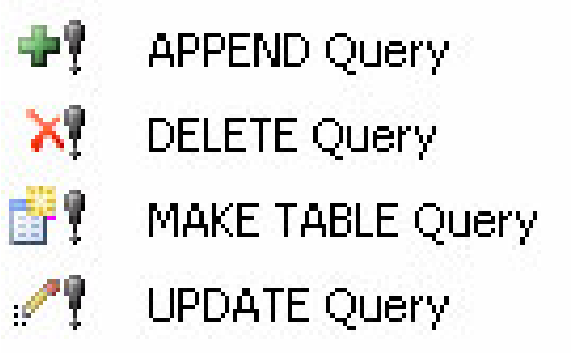
Në mënyrë që të dhënat të paraqiten sipas nevojës që kemi Access-i lejon krijimin e Shprehjeve (Expression) të bazuara në disa tipe të pyetësorit Total siç janë përdorimi i funksioneve Avg, Sum etj.P.SH. Shkruajmë një shprehje
Fusha: Sum(KREDITË.Shuma/KREDITË.[Kohëzgjatja e kredisë])
Te rreshti Total zgjedhet Expression dhe ruhet pyetësori.

Tipi i pyetësorëve Action

Ky tip i pyetësorëve është special, quhen pyetësorë të aksionit (Action) që na mundësojnë të ndryshojmë vlerat në të dhënat tona.Vet termi Action (Aksion) definon një pyetësor që thjesht bën diçka më shumë se sa selektimin e të dhënave dhe paraqitjen e tyre në bashkësinë e të dhënave (Recorset).Pyetësori Action mund të konsiderohet si një pyetësor Select që i është dhënë një detyrë ta kryej mbi një grup të caktuar të të dhënash.
Me krijimin e një pyetësori ne automatikisht krijojmë një pyetësor Select, ku pastaj ne mund të zgjedhim tipet e pyetësorit në pullën Query type që gjendet më toolbar.



Nga kjo meny mund të zgjedhim disa tipe të pyetësorëve Action.Zgjedhjet në meny janë : Make Table, Update, Append dhe Delete.Në dallim nga pyetësorët Select, pyetësorët Action identifikohen me një pikëçuditëse për skaj simbolit të pyetësorit.



Janë katër tipe të ndryshme që kanë ikona të ndryshme.Me anë të pyetësorëve Action mund të kryejmë:

- Fshirjen e të dhënave të caktuara nga një tabelë ose grup i tabelave
- Shtojmë të dhënat nga një tabelë te tjetra
- Ndryshojmë të dhënat
- Krijojmë një tabelë të re nga të dhënat e caktuara nga ndonjë pyetësor

Pasi që pyetësorët Action janë të pakthyeshëm (ireversibël) duhet të ndiqen këta katër hapa për krijimin e tyre:

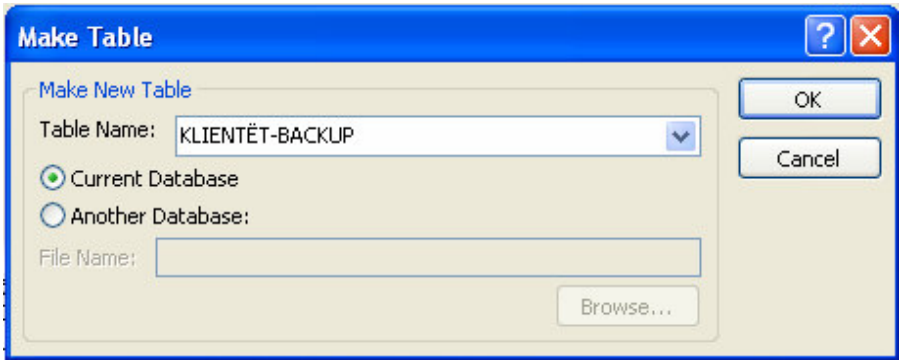
1. Krijimi i pyetësorit Action duke caktuar fushat dhe kriterin
2. Shikimi i të dhënave (në Datasheet View) të ndikuara nga pyetësori Action
3. Ekzekutimi i pyetësorit
4. Kontrollimi i ndryshimeve të bëra (në Datasheet View)

Shikimi i të dhënave para ekzekutimit të pyetësorëve DELETE dhe UPDATE

Para ekzekutimit të pyetësorëve **DELETE** dhe **UPDATE** klikojmë në pullën Datasheet View për shikimin e të dhënave që do të ndikohen nga pyetësori.Kjo bëhet shkaku se pyetësori ndikon në tabelën me anë të së cilës është krijuar dhe ndryshimet e bëra janë të pakthyeshme.
P.SH. Krijojmë një pyetësor nga tabela **KLIENTËT**.Bartim të gjitha fushat në zonën e dizajnimit dhe në toolbar shtypim pullën Query Type, aty zgjedhim **DELETE QUERY.** Në zonën e dizajnimit shihet rreshti **Delete** dhe në çdo fushë opcioni **Where**.Te rreshti **Criteria** shkruajmë kriterin mbi bazë të së cilit do të bëhet fshirja e të dhënave. Shkruajmë te fusha **Emri** dhe rreshti **Criteria** kriterin **"Bujar".**Shtypim pullën Datasheet View për të shikuar të dhënat që do të ndikohen nga ky pyetësor, pastaj bëjmë ekzekutimin e tij.Do të shikojmë se e dhëna është fshirë nga tabela **KLIENTËT.**

Shikimi i rezultatit të pyetësorëve MAKE-TABLE dhe APPEND

Në dallim nga pyetësorët **DELETE** dhe **UPDATE** pyetësorët **MAKE-TABLE** dhe **APPEND** kopjojnë të dhënat nga një tabelë në tabelën tjetër.Pra tabela prej ku krijohet pyetësori **MAKE-TABLE** ose **APPEND** nuk ndikohet.Me ekzekutimin e këtyre pyetësorëve krijohet një tabelë e re.
P.SH. Krijojmë një pyetësor nga tabela **KLIENTËT**, bartim fushat **Numri i llogarisë**, **Emri, Mbiemri, Qyteti, Adresa** dhe **Tel**.Te pulla **QUERY TYPE** zgjedhim **MAKE-TABLE** dhe na paraqitet një dritare



Te **Table Name** shkruajmë **KLIENTËT-BACKUP** dhe shtypim **OK**.Ekzekutojmë këtë pyetësor, do të shohim se kemi krijuar një tabelë të re **KLIENTËT-BACKUP** nga të dhënat e tabelës **KLIENTËT**.Siç shihet pyetësori **MAKE-TABLE** është i përshtatshëm për krijimin e **BACKUP** për tabelat e bazës.

KRIJIMI I PYETËSORËVE ACTION

Krijimi i pyetësorit UPDATE për ndryshimin e të dhënave

Më herët kemi mësuar për ndryshimi e të dhënave me anë të pyetësorit duke shtypur pullën **NEW RECORD**, por aty duhet që të dhënat të ndryshohen një nga një, pra mundësia e gabimit është e madhe.Që të ndryshohet një grup i të dhënash krijojmë pyetësorin **UPDATE**.Hapat e krijimit të pyetësorit UPDATE janë:

1. Krijimi i pyetësorit në Design View
2. Zgjedhja e tabelës si dhe bartja e fushave në zonën e dizajnimit
3. Shtypim pullën **QUERY TYPE** dhe zgjedhim **UPDATE QUERY**

Pastaj te rreshti **Criteria** shkrujamë kriterin në bazë të të cilit do t'i ndryshojmë të dhënat, ndërsa te rreshti **Update to** shkruajmë vlerën ndryshuese për ato të dhëna.
Në fund ekzekutjmë pyetësorin, ku do të hapet një dritare që do të informojë për numrin e të dhënave që do të afektohen nga ky ndryshim.Si dhe pyet se a dëshirojmë që të bëjmë ndryshimin e të dhënave sepse nëse po atëherë nuk mund të përdorim **Undo** komandën (pra pyetësorët Action siç u cek më herët janë ireversibël) .

P.SH. Dëshirojmë që në të dhënat e tabelës **KLIENTËT** ku mbiemri fillon me shkronjën K të ndryshohet në Gashi.
Krijojmë një pyetësor në Design View nga tabela **KLIENTËT**, bartim të gjitha fushat.Zgjedhim **UPDATE QUERY** nga pulla **QUERY TYPE**.Te fusha **Mbiemri**, te rreshti **Criteria** shkruajmë **LIKE "K*",** ndërsa te rreshti **Update to** shkruajmë **"Gashi".**
Ekzekutojmë pyetësorin dhe shkojmë te tabelat hapim tabelën **KLIENTËT** dhe shikojmë ndryshimet e bëra.

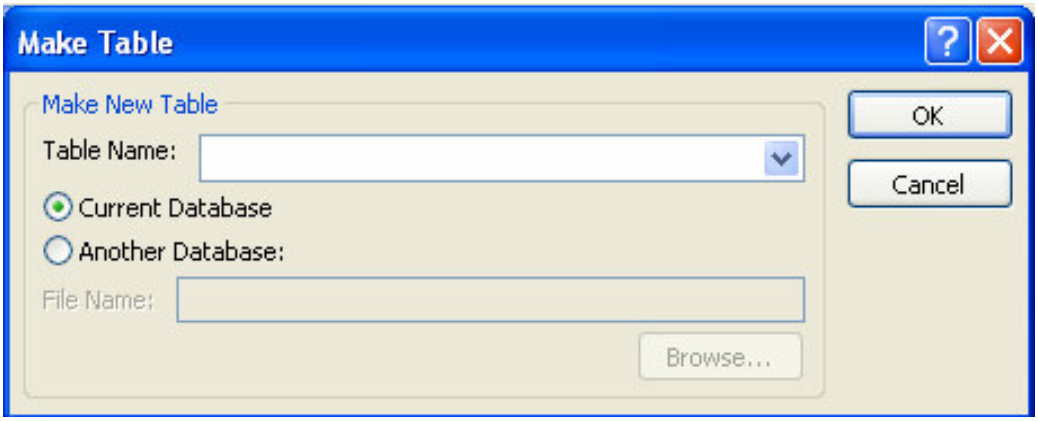
Krijimi i një tabele të re me anë të pyetësorit MAKE-TABLE

Për krijimin e një tabele të re mund të përdorim pyetësorët Action përkatësisht **MAKE-TABLE**.Me anë të këtyre pyetësorëve mund të krijojmë tabela nga rezultatet e pyetësorit, mund të bëjmë **BACKUP** për tabelat e bazës.Hapat për krijimin e pyetësorit **MAKE-TABLE** janë:

1. Krijimi i pyetësorit në Design View
2. Zgjedhja e tabelës si dhe bartja e fushave në zonën e dizajnimit
3. Shtypim pullën **QUERY TYPE** dhe zgjedhim **MAKE-TABLE QUERY**
4. Zgjedhim emrin e tabelës së re si dhe bazën e të dhënave ku do ta ruajmë

Ekzekutojmë pyetësorin dhe hapet një dritare ku tregon se sa të dhëna do të barten në tabelën e re.Si dhe pyet se a dëshirojmë që të krijojmë tabelë të re me të dhënat sepse nëse po atëherë nuk mund të përdorim Undo komandën (pra pyetësorët Action siç u cek më herët janë ireversibël) .

P.SH. Dëshirojmë të krijojmë një tabelë të re me numrin e llogarisë, emrin, mbiemrin, tipin e kredisë dhe shumën e kredisë nga baza **BANKA**.
Krijojmë një pyetësor në Design View nga tabelat **KLIENTËT** dhe **KREDITË**. Bartim fushat **Numri i llogarisë, Emri, Mbiemri** nga tabela **KLIENTËT** si dhe fushat **Tipi i kredisë** dhe **Shuma** nga tabela **KREDITË**.Te pulla **QUERY TYPE** zgjedhim **MAKE-TABLE QUERY** dhe hapet një dritare:



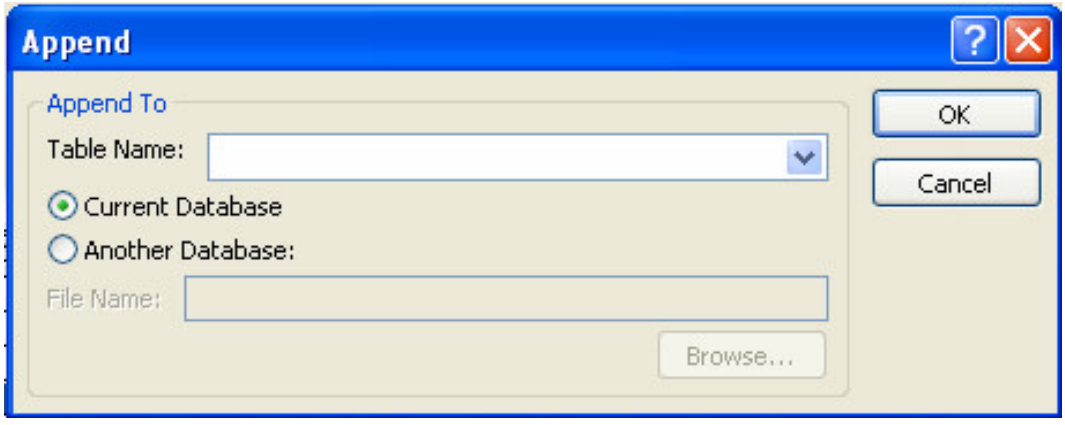
ku te **Table Name**: shkruajmë emrin e tabelës së re, nëse dëshirojmë që atë tabelë ta ruajmë në bazën ekzistuese e zgjedhim **Current Database** por nëse dëshirojmë që ta ruajmë në një bazë tjetër zgjedhim **Another Database** dhe shtypim pullën **Browse...**që të japim shtegun e bazës tjetër.Në fund shtypim **OK** dhe ekzekutojmë pyetësorin. Pas ekzekutimit hapet një dritare ku tregon se sa të dhëna do të barten në tabelën e re.Si dhe pyet se a dëshirojmë që të krijojmë tabelë të re me të dhënat sepse nëse po atëherë nuk mund të përdorim **Undo** komandën (pra pyetësorët Action siç u cek më herët janë ireversibël) .Shtypim **YES** dhe shikojmë tabelën e re të krijuar te objektet **Tables**.

Krijimi i pyetësorit APPEND për kopjimin e të dhënave

Fjala *append* nënkupton bashkangjit, pra pyetësori **APPEND** bashkangjet të dhënat te një tabelë e caktuar.Pyetësori **APPEND** bashkangjet të dhënat nga tabela të cilën e përdorim te ndonjë tabelë tjetër.Tabela në të cilën dëshirojmë që këto të dhëna t'i shtojmë duhet që të jetë e krijuar më parë.Të dhënat mund t'i shtojmë në bazën ku punojmë por edhe në ndonjë bazë tjetër.Me anë të pyetësorit **APPEND** mund të kopjojmë të dhënat nga një tabelë ose pyetësor dhe ti shtojmë në një tabelë tjetër.
Gjatë punës me pyetësorët **APPEND** duhet të kemi kujdes në këto rregulla:

1. Nëse tabela që po i bashkangjesim të dhëna ka fushë me çelës primar, atëherë nuk mund të shtojmë të dhëna boshe ose vlera të dyfishta në atë fushë.
2. Nëse shtojmë të dhëna në një bazë të të dhënash tjetër duhet që të dijmë vendin dhe emrin e bazës së të dhënave.
3. Nëse shfrytëzojmë * për bartjen e fushave në zonën e dizajnimit, nuk mund të përdorim më fushat individualisht në të njëjtën tabelë.Sepse Access-i do të kuptojë se po provojmë që të shtojmë të dhënat e asaj fushe dyherë.
4. Nuk duhet që të dhënat që i bashkangjesim të kenë ndonjë fushë Autonumber.

Me respektimin e këtyre rregullave të thjeshta, pyetësori **APPEND** duhet që të punojë mirë dhe do të jetë një vegël e dobishme.
P.SH. Nëse dëshirojmë që nga tabela **KLIENTËT** t'i kopjojmë të dhënat në tabelën **ISH KLIENTËT**.Te **Queries** krijojmë një pyetësor në **Design View**, zgjedhim tabelën **KLIENTËT** dhe i bartim të gjitha fushat në zonën e dizajnimit.Te **QUERY TYPE** zgjedhim **APPEND QUERY** dhe hapet një dritare



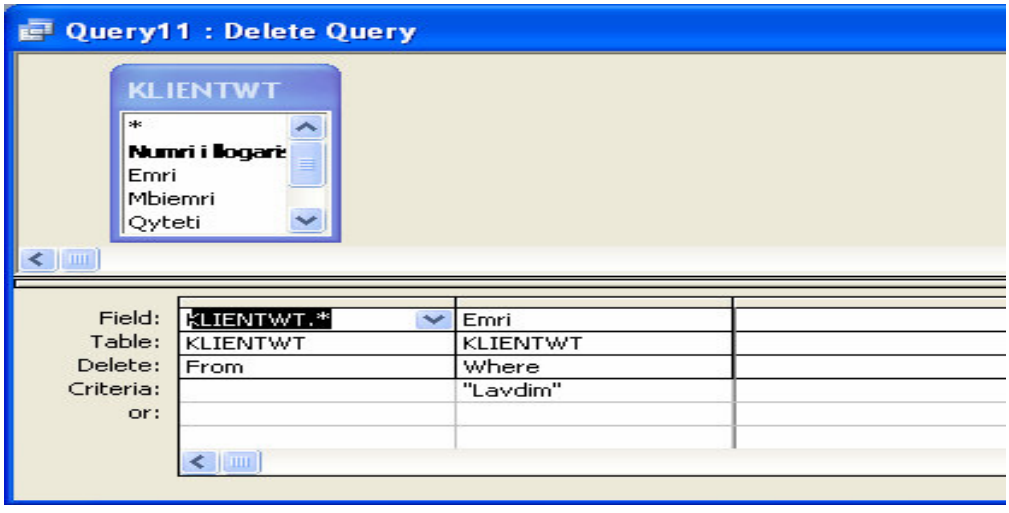
te **Table Name** shkruajmë emrin e tabelës së cilës dëshirojmë t'i bashkangjesim të dhënat, pra **ISH KLIENTËT** dhe shtypim **OK.**Nëse fushat në të dy tabelat kanë emra të njëjtë atëherë të dhënat shtohen automatikisht.Në fushat me emra të ndryshëm duhet të caktojmë vetë cilat të dhëna do t'i shtojmë.Kjo bëhet në rreshtin **Append to** ndërsa ka mundësi që të shfrytëzohet edhe **Criteria** për të zgjedhur të dhënat që dëshirojmë t'i shtojmë, pra vetëm të dhënat që plotësojnë kriterin do t'i shtohen tabelës.

Krijimi i pyetësorit DELETE për fshirjen e të dhënave

Nga të gjithë pyetësorët, pyetësorët **DELETE** janë më të pa përshtatshëm.Kjo vjen nga shkaku se bëjnë fshirjen e të dhënave në mënyrë të përhershmë dhe të pakthyeshme.Sikurse edhe pyetësorët tjerë edhe pyetësorët **DELETE** veprojnë në një grup të të dhënash në bazë të kriterit.Nëse paraqitet nevoja për fshirjen e të dhënave në më shumë se një tabelë përnjëherë atëherë duhet të shkojmë sipas hapave të mëposhtëm:

- Definimi i relacionit të tabelave në dritaren e lidhjeve (Relationship window)
- Kontrollimi i Integritetit Referencial
- Kontrollimi i Cascade Delete Related Records

Nëse lidhja e tabelave është një me shumë, pa definimin e relacionit të tabelave dhe nëse nuk është zgjedhur Cascade Delete atëherë do të fshihen vetëm të dhënat nga tabela e parë.Pyetësori **DELETE** nuk fshin të dhënat në fusha të caktuara por fshin tërë të dhënën.
P.SH. Krijojmë një pyetësor me tabelën **KLIENTËT**, në **QUERY TYPE** zgjedhim **DELETE QUERY**.Do të shohim se në zonën e dizajnimit shtohet rreshti **Delete**. Bartim ***** dhe fushën **Emri** në zonën e dizajnimit.Te fusha **KLIENTËT.*** dhe rreshti **Delete** zgjedhim **From** ndërsa te fusha **Emri** rreshti **Delete** zgjedhim **Where** dhe te rreshti **Criteria** shkruajmë **"Valbona".**E bëjmë ekzekutimin e pyetësorit, do të shohim se klienti me emrin Valbona është fshirë nga tabela **KLIENTËT** si dhe janë fshirë edhe të dhënat mbi kreditë e klientit në tabelën **KREDITË.**Kjo ka ndodh shkaku i relacionit të këtyre dy tabelave një me shumë si dhe **Cascade Delete** opcionit.Po të largohet opcioni **Cascade Delete** fshirja e kësaj të dhëne mbi këtë klientë nuk do të bëhet.



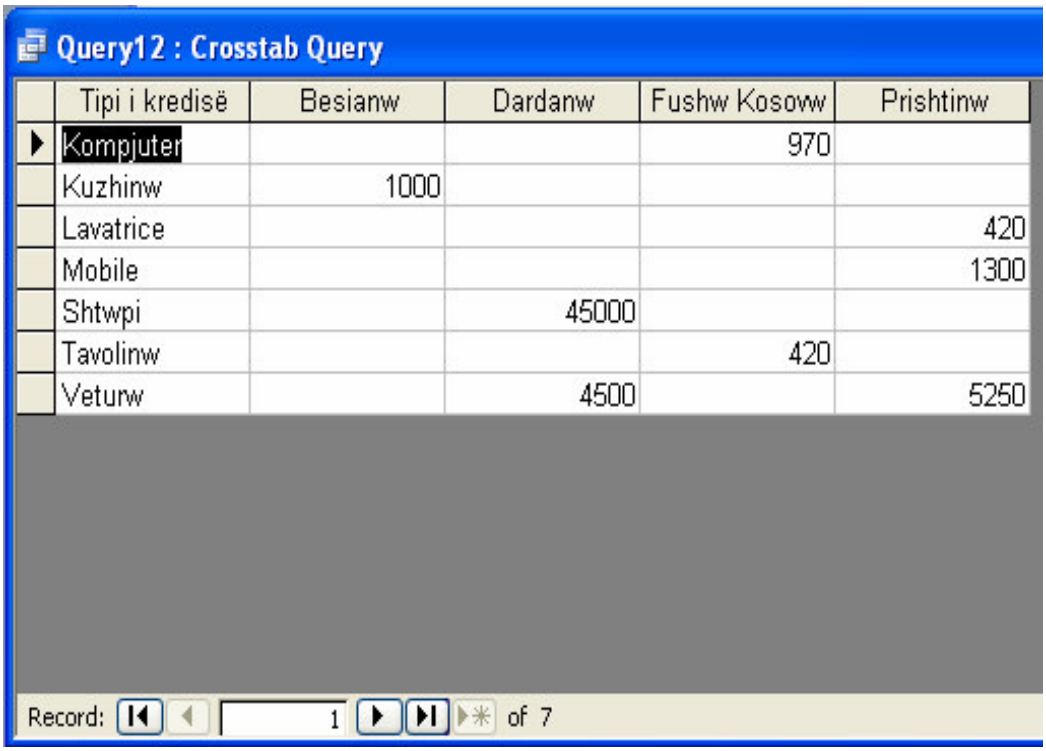
KRIJIMI I PYETËSORËVE CROSSTAB

Access-i lejon krijimin e një tipi të përshatshëm të pyetësorit Total ai është pyetësori Crosstab.Pyetësori Crosstab paraqet të dhënat në formatin rresht-kolonë.Thjesht thënë pyetësori Crosstab është një përmbledhje e të dhënave të caktuara nga titulli i rreshtit (row heading) dhe titulli i kolonës (column heading).Në këtë pyetësor rreshti Total gjithmonë është aktiv dhe shërben për përcaktimin e opcionit Group by për titullin e rreshtit dhe titullin e kolonës.

Krijimi i pyetësorit Crosstab bëhet duke zgjedhur CROSSTAB QUERY në pullën QUERY TYPE në toolbar.Pastaj duhet që së paku tri fusha të jenë në zonën e dizajnimit:

- Fusha e titullit të rreshtit (row heading)
- Fusha e titullit të kolonës (column heading)
- Fusha Value

P.SH. Krijojmë një pyetësor në Design View nga tabelat **KLIENTËT** dhe **KREDITË**. Bartim fushat **Tipi i kredisë**, **Shuma** nga tabela **KREDITË** si dhe fushën **Qyteti** nga tabela **KLIENTËT.**Zgjedhim Crosstab Query në pullën Query type në toolbar.Në zonën e dizajnimit paraqiten rreshtat Total dhe Crosstab.Te fusha Tipi i kredisë, rreshti Total zgjedhim Group by ndërsa te rreshti Crosstab zgjedhim **Row Heading**.Pastaj te fusha Qyteti, rreshti Total zgjedhim Group by ndërsa te rreshti Crosstab zgjedhim **Column Heading**.Te fusha Shuma, rreshti Total zgjedhim **Sum** ndërsa te rreshti Crosstab zgjedhim**Value**.Ekzekutojmë pyetësorin dhe do të shohim një tabelë



Provojeni që në fushën **Shuma** në vend të funksionit Sum të vëhet funksioni Count dhe ekzekutoni pyetësorin.

Pyetësorët Crosstab mund të kenë më shumë se një titull të rreshtit (Row Heading), por vetëm një titull të kolonës (Column Heading).
P.SH. Në pyetësorin e mëparshëm barti në zonën e dizajnimit edhe fushën Emri nga tabela KLIENËT dhe te rreshti Total zgjedhni Group by ndërsa te rreshti Crosstab zgjedhni Row Heading.

Caktimi i kriterit për pyetësorët Crosstab

Kriteri për pyetësorët Crosstab mund të caktohet në:

1. Fushë të re
2. Fushën e titullit të rreshtit (Row Heading)
3. Fushën e titullit të kolonës (Column Heading)

Caktimi i kriterit në fushën e re

Së pari bartim fushën në zonën e dizajnimit, te rreshti Total zgjedhim Group by ndërsa te rreshti Criteria shkruajmë kriterin.
P.SH. Te pyetësori i mëparshëm shtojmë fushën Mbiemri nga tabela KLIENTËT, te rreshti Total zgjedhim Group by ndërsa te rreshti Criteria zgjedhim "Krasniqi".Pra po kërkojmë klientët me mbiemrin Krasniqi.

Caktimi i kriterit për titullin e rreshtit (Row Heading)

Caktimi i kriterit bëhet në rreshtin Criteria.P.SH. Te fusha Tipi i kredisë te rreshti Criteria shkruajmë "Veturë", dhe si rezultat do të fitojmë të dhënat për kreditë Veturë.

Caktimi i kriterit për titullin e kolonës (Column Heading)

Njësoj sikur te titulli i rreshtit.

Tipi i pyetësorëve SQL

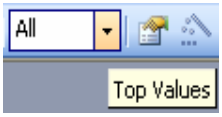
Ky tip i pyetësorëve krijohet me gjuhën specifike të bazës së të dhënave SQL (Structured Query Language).Me SQL mund të krijojmë të gjithë pyetësorët e përfytyrueshëm.Më gjërësisht për këtë do të mësojmë në kuadër të MYSQL-it.
P.SH. Krijimi i pyetësorit nga tabela KLIENTËT me të gjitha fushat e asaj tabele në SQL bëhet:

Select * from KLIENTËT

Ku * paraqet të gjitha fushat e tabelës sikurse edhe te Access-i.

Tipi i pyetësorëve TOP(n)

Me anë të këtyre pyetësorëve Access-i na mundëson t'i gjejmë (n) të dhënat e para (ku n është numër ose përqindje).Ky pyetësor gjithmonë përdoret me tipet e pyetësorëve që i mësuam më parë.Nëse krijojmë një pyetësor në Design Vieë nga tabela KREDITË dhe bartim të gjitha fushat në zonën e dizajnimit.Pastaj shkojmë te pulla Top Values në toolbar



dhe zgjedhim numrin e vlerave që dëshirojmë të paraqiten në bashkësinë e të dhënave ose përqindjen.P.SH. zgjedhim numrin 5, atëherë do të na paraqiten vetëm pesë të dhënat e para të bashkësisë së të dhënave.

FORMA (FORMULARI)

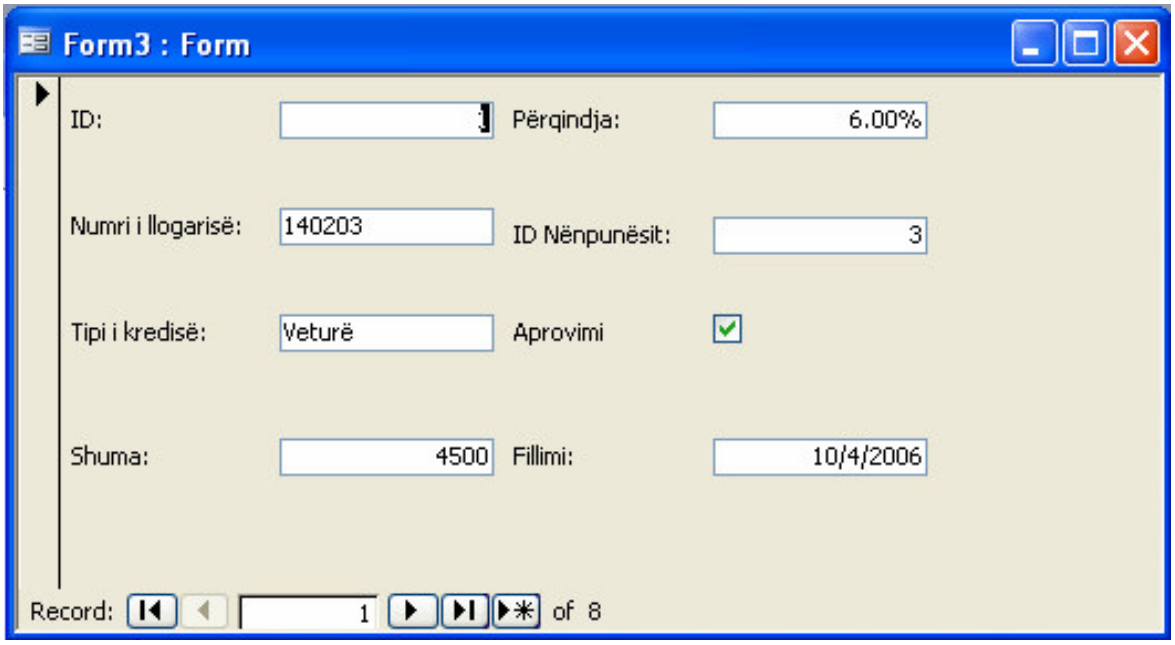
Format (Formularët) paraqesin mënyrën më fleksibile për shikimin, shtimin, ndryshimin dhe fshirjen e të dhënave. Me anë të formës mund të shikojmë më shumë se një të dhënë me të gjitha fushat në të njëjtën kohë. Poashtu mund të shfrytëzojmë format për krijimin e dritareve komanduese të njohura si **switchboards.**

Tipet e paraqitjes së formularëve

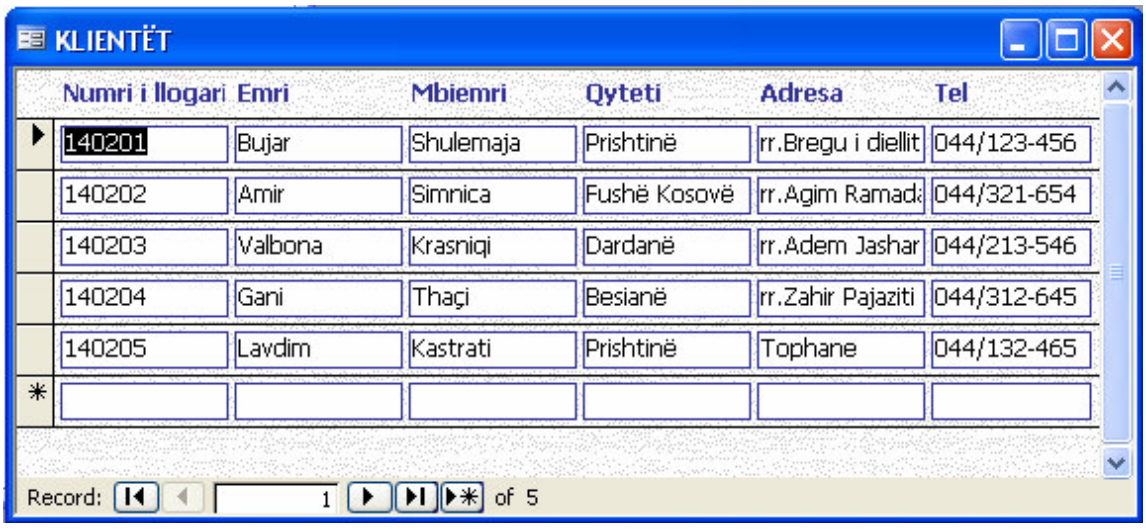
Tipet bazë për paraqitjen e formularëve janë:

- **Columnar** – paraqitja e formularëve në një kolonë
- **Tabular** – paraqitja tabelare e të dhënave
- **Datasheet View** – paraqitja e të dhënave në fletën e të dhënave
- **Justified** – paraqitja e të dhënave në bllok
- **Main/subforms** – paraqitja e formularit kryesor dhe një formulari të ndërvarur
- **Pivot table** – sikur në Microsoft Excel
- **Pivot Chart** – paraqitja e të dhënave me anë të grafeve etj.

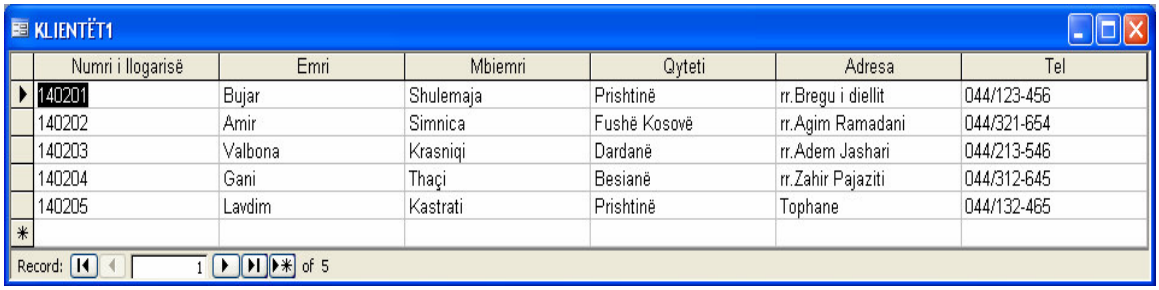
Figura e mëposhtme tregon paraqitjen e të dhënave në formën Columnar.Në formular shfaqet vetëm një e dhënë, fushat mund të rregullohen sipas dëshirës.Pastaj është mundësia e rregullimit të formularit për një pamje më të mirë me anë të vijave, ngjyrave dhe efekteve speciale (hija, pamja 3D).



Paraqitja e disa të dhënave në të njëjtën kohë në formular është tipi **Tabular** që tregohet në figurën e mëposhtme.Në këtë rast të dhënat paraqiten sipas rreshtave.

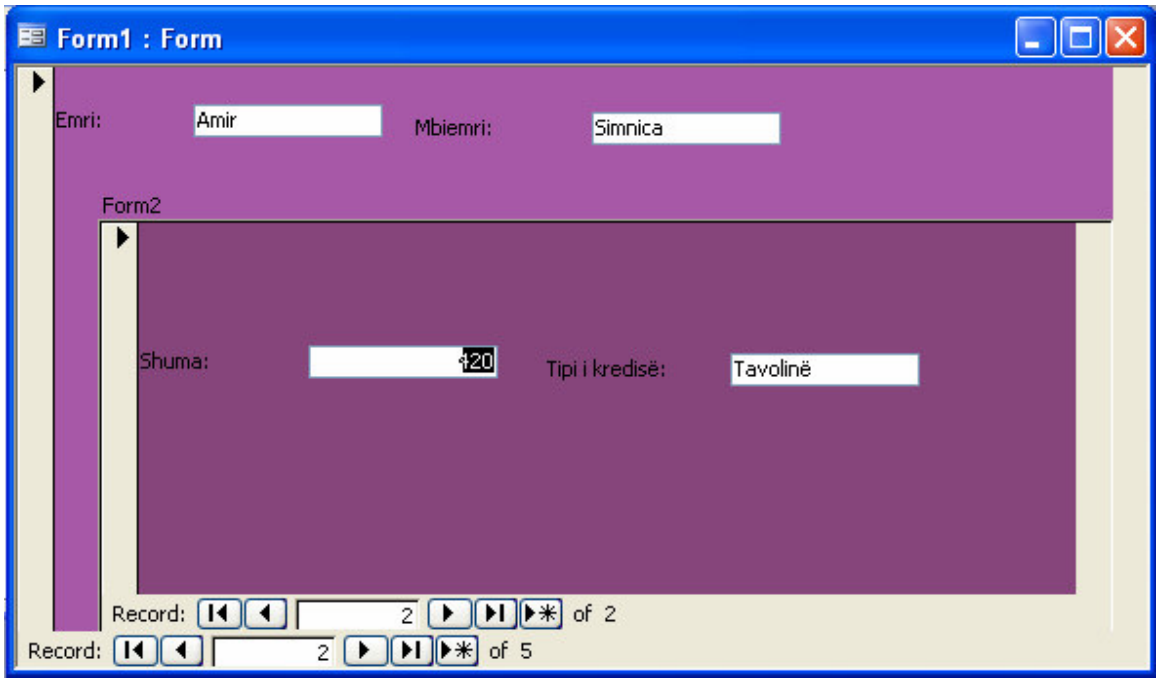


Paraqitja e të dhënave në formën Datasheet view është në mënyrë të ngjashme me pamjen e fletës së të dhënave të tabelave siç shihet nga figura e mëposhtme.

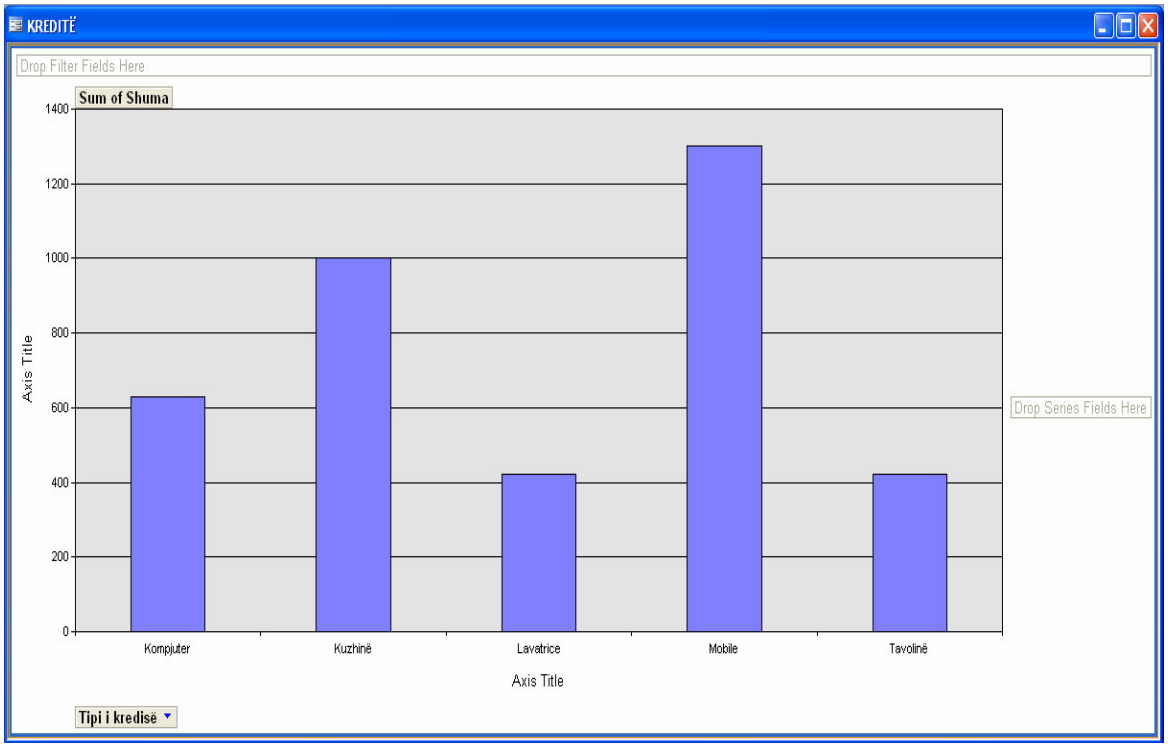


Paraqitja e të dhënave në bllok (**Justified**) bëhet ashtu që çdo e dhënë paraqitet në një bllok unik siç shihet nga figura e mëposhtme.

Paraqitja e formularit në formën **main/subforms** është një formular kryesor dhe një formular i ndërvarur.Kjo do të ishte një metodë e mirë për të paraqitur klientët në formularin kryesor si dhe tipin e kredisë në formularin e ndërvarur si në figurën e mëposhtme.



Paraqitja e formularëve në formë **Pivot Table** dhe **Pivot Chart** është paraqitje e njëjtë sikur te Microsoft Excel.



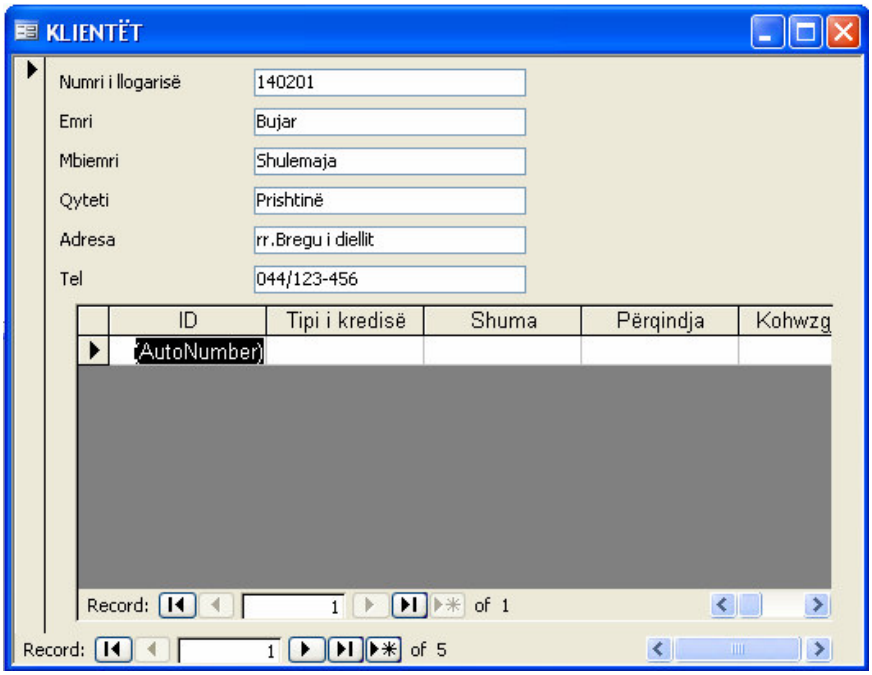
Krijimi i formularit me AutoForm

Mënyra më e shpejtë për krijimin e formularit është me përdorimin e AutoForm. Por me këtë mënyrë nuk na jepet kontroll mbi dizajnin e formularit siç do të shohim më poshtë.

Një ndër mënyrat e krijimit të formularit me AutoForm është:

Te objektet e bazës zgjedhim Tables (Tabelat) dhe zgjedhim tabelën me anë të së cilës dëshirojmë të krijojmë formularin, pastaj shkojmë në toolbar te pulla New Object dhe zgjedhim AutoForm. Do të shohim se formulari do të krijohet automatikisht.

Shembull:
Në bazën e të dhënave Banka te objektet e bazës Tables zgjedhim tabelën KLIENTËT, në toolbar te pulla New Object zgjedhim AutoForm dhe do të krijohet formulari si në figurën e mëposhtme.



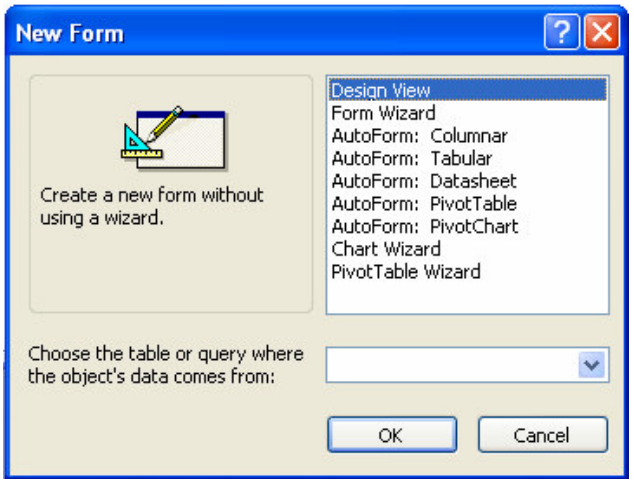
Siç shihet në formular u paraqit edhe tabela KREDITË kjo vjen ngase këto dy tabela janë të lidhura në dritaren e lidhjeve (Relationship window).

Krijimi i formularit me ndihmën Asistentit të formularit (Form Wizard)

Me anë të Asistentit të formularit e thjeshtojmë renditjen e fushave në formular. Asistenti i formularit na udhëheq nëpër një varg pyetjesh për formularin të cilin dëshirojmë ta krijojmë.Ekzistojnë disa mënyra për ta aktivizuar Asistentin e formularit:

- Nga dritarja e bazës së të dhënave zgjedhim **Insert** dhe **Form**
- Nga objektet e bazës zgjedhim **Forms** dhe klikojmë në pullën **New** në dritaren e bazës së të dhënave
- Në toolbar zgjedhim pullën **New Object** dhe **Form**

Pamarrë parasysh se cilën mënyrë e zgjedhim për krijimin e formularit neve do të na paraqitet dritarja e mëposhtme.

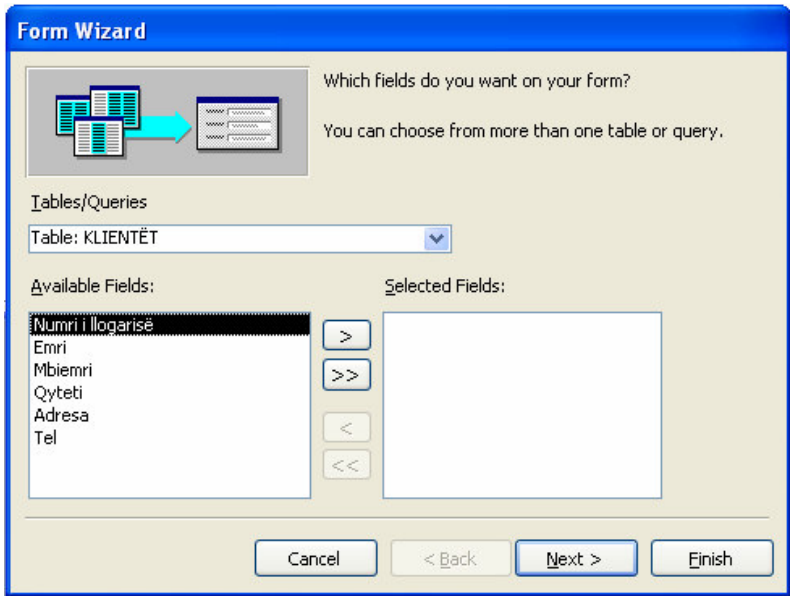


Ku aty mund të zgjedhim tipin e krijimit të formularit dhe tabelën ose pyetësorin nga i cili do të krijojmë formularin. Dritarja **New Form** ka nëntë zgjedhje për krijimin e formularit:

1. **Design View** - paraqet një formular plotësisht të zbrazët
2. **Form Wizard** - krijon formularin nga gjashtë mundësi të parazgjedhura: Columnar, Tabular, Datasheet, Justified, PivotTable dhe PivotChart.
3. **AutoForm: Columnar** - krijon formularin në bazë të tipit Columnar
4. **AutoForm: Tabular** - krijon formularin në bazë të tipit Tabular
5. **AutoForm: Datasheet** - krijon formularin në bazë të tipit Datasheet
6. **AutoForm: PivotTable** - krijon formularin në bazë të tipit Pivot Table
7. **AutoForm: PivotChart** - krijon formularin në bazë të tipit Pivot Chart

8. **Chart Wizard:** krijon formularin me anë të grafeve të ndryshme
9. **PivotTable Wizard:** krijon formularin si Excel Pivot Table

Për startimin e Asistentit për krijimin e formularit zgjedhim **Form Wizard**.
Në pjesën e poshtme të dritares mund të zgjedhim tabelën ose pyetësorin (të krijuar më parë) për krijimin e formularit. Pra te *Choose the table or query where the object's data comes from:* zgjedhim objektin e dëshiruar dhe klikojmë OK. Pastaj na paraqitet dritarja për zgjedhjen e fushave nga objektet e bazës. Pra mund të zgjedhim fusha nga një ose më shumë tabela ose pyetësorë.



Fushat që dëshirojmë të jenë në formularë i zgjedhim dhe pastaj i shtojmë në anën e djathtë më anë të këtyre pullave:

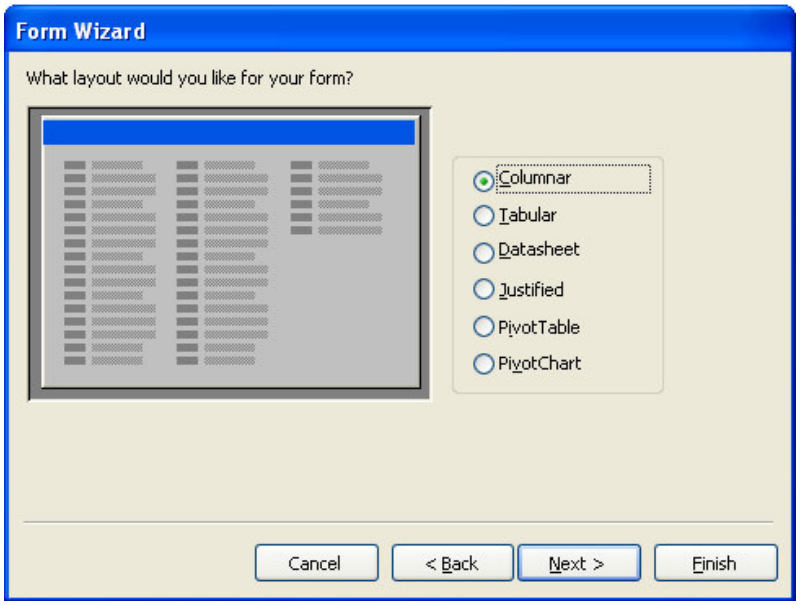
Shto fushën e zgjedhur

Shto tërë fushat

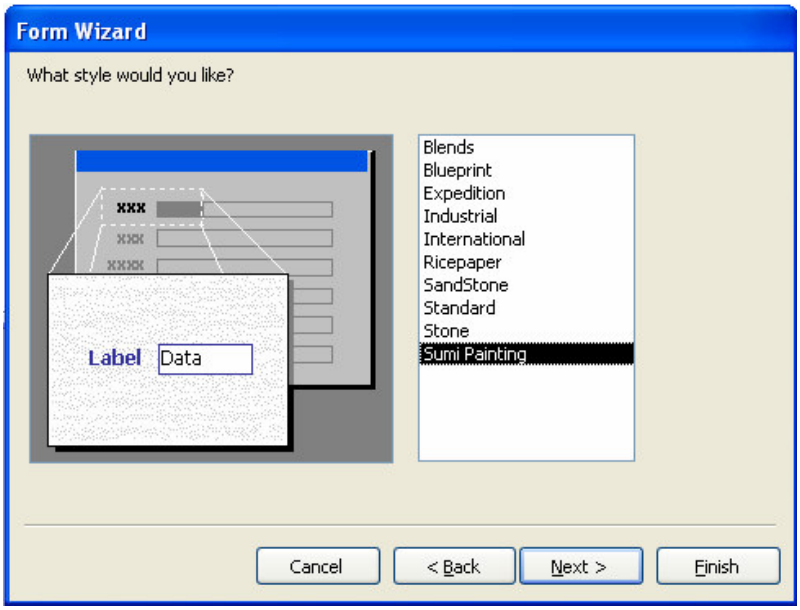
Largo fushën e zgjedhur

Largo tërë fushat

Pas zgjedhjes së fushave shtypim pullën Next dhe hapet dritarja për renditjen e fushave në formular.

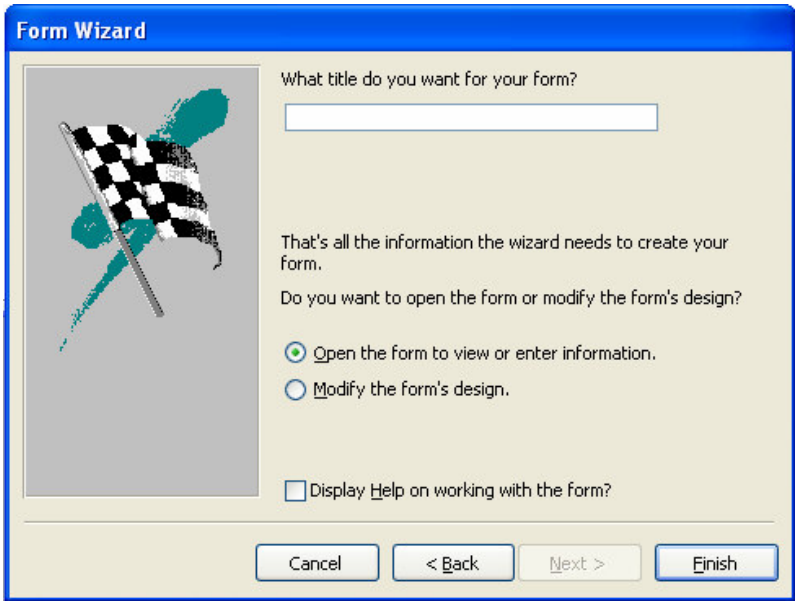


Pasi të jetë zgjedhur forma e renditjes së fushave në formular shtypim pullën Next dhe hapet dritarja për zgjedhjen e pamjes (stilit) së formularit. Janë në dispozicion dhjetë opcione, ku pamja e secilit opcion mund të shihet në anën e majtë të dritares vetëm me klikim të opcionit. Zgjedhet një opcion dhe shtypet pulla Next.



Pastaj paraqitet dritarja për emërimin e formularit si dhe opcionin për hapjen e formularit. Varësisht se a dëshirojmë që formulari të hapet në pamjen e tij përfundimtare zgjedhim

Open the form to view or enter information apo dëshirojmë që të ndërrojmë diçka në dizajnin e formularit me anë të Design View zgjedhim *Modify the form's design.* Pastaj shtypim pullën **Finish**.



Hapja e formularit bëhet duke klikuar dyherë në formular dhe do të shohim të dhënat. Navigimi në formular bëhet me anë të pullave në fund të formularit

 Kthen te e dhëna e parë

 Kthen te një e dhënë pas

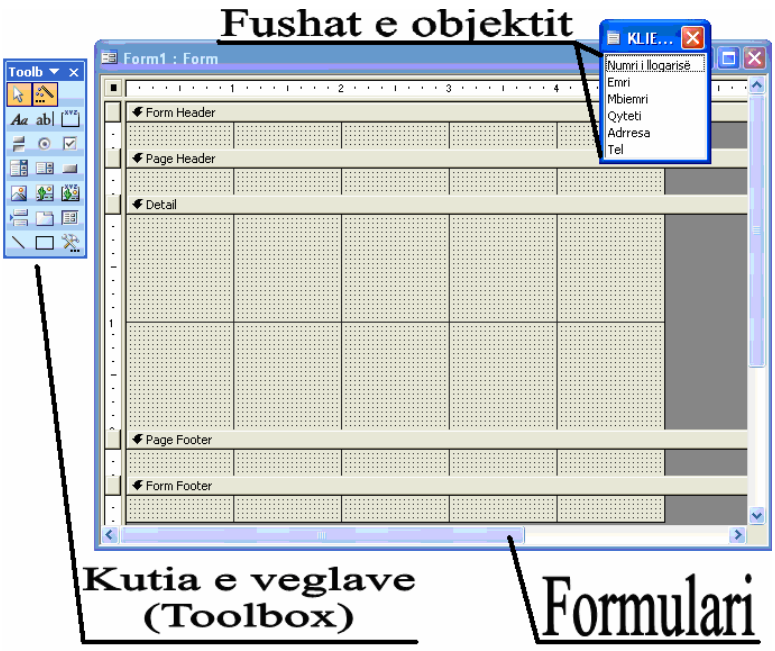
 E dhëna me radhë

 E dhëna e fundit

 Pulla për shtimin e të dhënave në formular (njësoj si te tabelat dhe pyetësorët)

Siç u pa me krijimin e formularit me anë të Asistentit të formularit (Form Wizard) nuk kemi shumë kontroll mbi krijimin e formularit. Ashtu si edhe te tabelat dhe pyetësorët kontroll më të madhe në krijimin e formularit do të kemi me anë të Design View.
Krijimi i formularit në Design View bëhet me klikimin në pullën **New** dhe duke zgjedhur **Design View** ndërsa te *Choose the table or query where the object's data comes from* shkruajmë tabelën ose pyetësorin që do t'a shfrytëzojmë për krijimin e formularit. Pastaj

do të paraqitet një formular bosh, fushat e objektit të zgjedhur (tabelë ose pyetësor) dhe kutia e veglave(toolbox).



Elementet kryesore përbërëse të pamjes Design View të formularit janë: zona e formularit dhe elementet e kontrollit. Formulari përbëhet nga Kreu i formularit (Form Header), Kreu i faqes (Page Header), Detalet (Detail), Fundi i faqes (Page Footer) dhe Fundi i formularit (Form Footer). Që të shihen këto pjesë përbërëse të formularit shkojmë në menynë kryesore te View dhe zgjedhim Page Header/Footer dhe Form Header/Footer.
Ndërsa elementet e kontrollit përcaktojnë përmbajtjen e formularit në pjesë të veçanta të tij.
Kreu i formularit përban mbishkrimin e formularit që shfaqet në pjesën e sipërme, ndërsa Fundi i formularit pjesën e poshtme të tij. Pjesa e Detaleve përmban fushat me të dhëna nga tabela ose pyetësori.

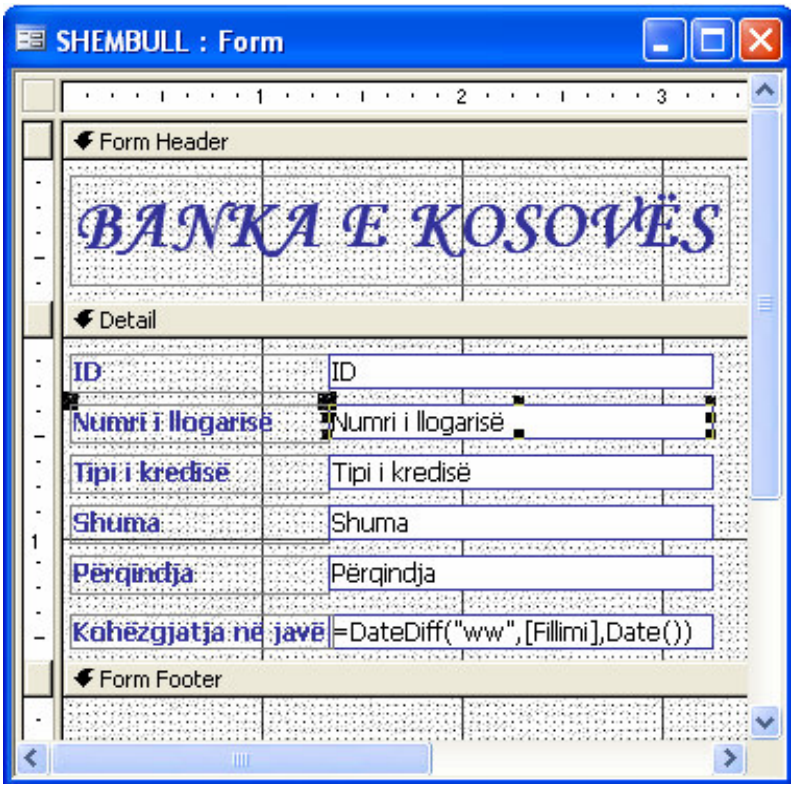
Elementet e kontrollit

Janë tri lloje të elementeve të kontrollit:

1. Elementet e lidhura
2. Elementet e palidhura
3. Elementet e kontrollit për kryerjen e përllogaritjeve

Elementet e lidhura të kontrollit paraqesin të dhënat nga fushat në tabelë ose pyetësor me anë të të cilëve është krijuar formulari. Poashtu elementet e lidhura përdoren edhe për futjen e të dhënave në formular.
Elementet e palidhura të kontrollit paraqesin informacionin që nuk është në tabelë. Elementet e palidhura mund të jenë tekst, fotografi, vija ose drejtëkëndësha të vizatuar në formular. Zakonisht elementet e palidhuara shoqërohen me elementet e lidhura si

emërtues dhe poashtu përdoren për të ofruar instruksione për mënyrën e shtimit të të dhënave.
Elementet e kontrollit për kryerjen e përllogaritjeve përmbajnë rezultatin nga llogaritja me fushat e tabelës.
Në figurën e mëposhtme shihen të tri llojet e elementeve të kontrollit. Emri i bankës pra *BANKA E KOSOVËS* është një element i palidhur i kontrollit. Ndërsa fushat në pjesën e detaleve janë elemente të lidhura sepse janë të lidhura me emrat e fushave. Fusha *Kohëzgjatja në javë* është një shembull i elementit të kontrollit për kryerjen e përllogaritjeve, ku funksioni *DateDiff* është shfrytëzuar për të llogaritur diferencën e kohës (në javë) së lëshimit të kredisë dhe datës së sotme.

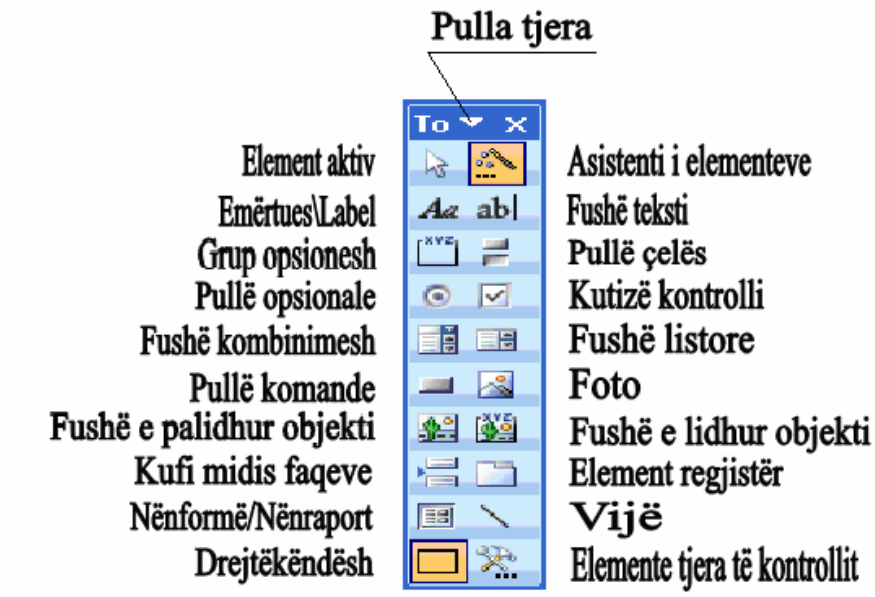


Veprimet themelore me elementet e kontrollit

Me klikimin në elementin e kontrollit bëjmë selektimin e tij, ndërsa selektimi i disa elementeve bëhet duke mbajtur shtypur pullën SHIFT dhe duke klikuar mbi elementet e kontrollit. Heqja e selektimit bëhet duke klikuar në një sipërfaqe të lirë të zonës së formularit. Ndryshimi i dimensionit bëhet duke selektuar elementin përkatës dhe vendosjen e miut në njërën nga pikat e vogla të tërheqjes derisa treguesi të marrë formën e një shigjete të dyfishtë. Pastaj bëjmë tërheqjen deri sa elementi të marrë dimensionin e dëshiruar. Zhvendosja e elementit të kontrollit bëhet me selektimin e tij dhe vendosjen e miut ndërmjet pikave të tërheqjes derisa treguesi të marrë formën e dorës. Pastaj bëhet tërheqja deri te pozita e dëshiruar. Fshirja e elemnetit të kontrollit bëhet duke selektuar atë dhe duke shtyp pullën Delete.

Përdorimi i kutisë së veglave (Toolbox)

Kutia e veglave përmban vegla për integrimin e elementeve të ndryshme të kontrollit në formë ose raport. Access-i hap dritaren e kutisë së veglave me hapjen e formularit në Design View, por nëse kutia e veglave nuk është aktivizuar mund ta aktivizojmë atë duke klikuar në toolbar në pullën  . Integrimi i elementeve të kontrollit nga kutia e veglave bëhet me selektimin e simbolit të dëshiruar dhe pastaj klikojmë në formular në vendin ku dëshirojmë ta integrojmë atë element të kontrollit. Mbajmë të shtypur pullën e miut dhe e tërheqim elementin e kontrollit deri në madhësinë e dëshiruar. Figura më poshtë është kutia e veglave (elementeve të kontrollit) dhe emërtimet e tyre.



Tani do të mundohemi që për secilën të japim shpjegim për përdorimin në dizajnimin e formularit.

Selektimi i objekteve

Ky element aktivizohet që në fillim. Ai shërben për selektimin, zhvendosjen, zmadhimin e elementeve të kontrollit.

Asistenti i elementeve

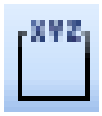
Në qoftëse është i aktivizuar ky element gjatë integrimit të elementeve tjerë të kontrollit atëherë do të paraqitet asistenti i elementeve në një dritare si ndihmë për secilin element përkatës.

Emërtues/Label

Me ndihmën e elementit të kontrollit Emërtues/Label paraqesim një tekst përshkrues në formular ose raport.

Fushë teksti

Elementi Fushë teksti shërben për paraqitjen e të dhënave dhe i ofron shfrytëzuesit të shtoj të dhëna, të ndryshojë të dhëna ose të fshijë. Secila Fushë teksti shoqërohet me një Emërtues për të identifikuar veten.



Grup opsionesh

Me ndihmën e elementit të kontrollit Grup opsionesh mund të bëjmë bashkimin e elementeve të kontrollit Pullë çelës, Pullë opsionale dhe Kutizë kontrolli në një pjesë të tërë.

Pullë çelës,

Pullë opsionale dhe

Kutizë kontrolli

Këto tri tipe të pullave shërbejnë që shfrytëzuesi të bëjë një zgjedhje. Të tri pullat veprojnë njësoj vetëm se pamja e tyre është e ndryshme. Këto elemente të kontrollit përdoren te tipet e të dhënave Yes/No. Këto kthejnë vlerën -1 nëse vlera e pullës është Yes, On ose True ndërsa 0 nëse vlera e pullës është No, Off ose False.

Fushë kombinimesh

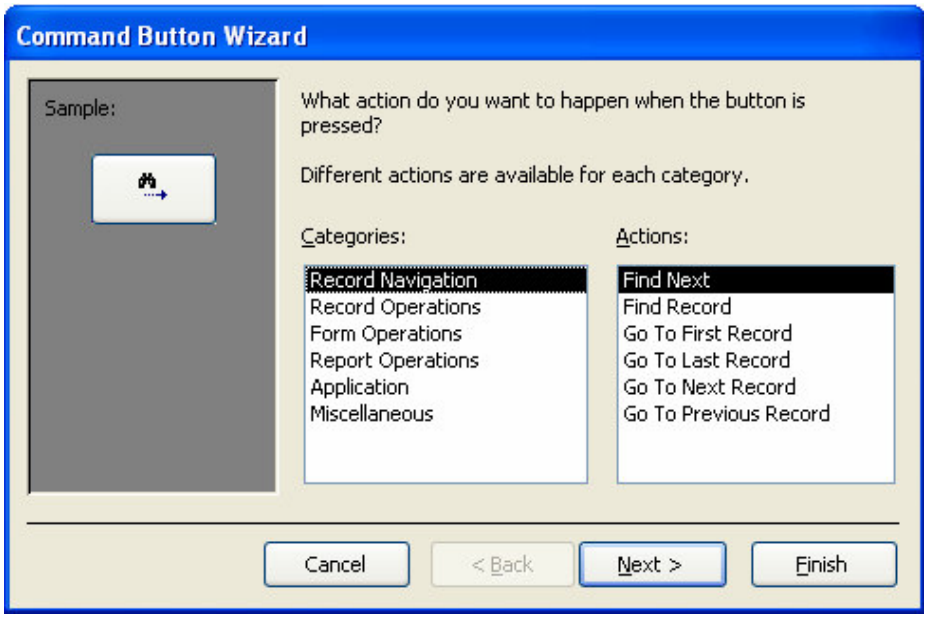
Elementi Fushë kombinimesh paraqitet në një rresht, kështu që mund të zgjedhim ose nga lista e shkruar më parë ose mund të shkruajmë tekstin e dëshiruar. Përparësi e fushës së kombinimeve është nga fakti se lista shpaloset vetëm atëherë kur hapet Fusha e kombinimeve duke zënë kështu pak vend në sipërfaqen e formularit.

Fushë listore

Ky element përmban listën e të dhënave nga të cilat shfrytëzuesi mund të bëjë zgjedhjen. Fusha listore është gjithnjë e hapur në dallim nga elementi i kontrollit Fushë kombinimesh dhe në të nuk mund të shkruajmë vlerat që dëshirojmë. Ky element i kontrollit përdoret kur ka mjaft hapësirë në formular.

Pullë komande

Në varësi ky element i kontrollit përdoret për ekzekutimin e makrove, ekzekutimin e ndonjë programi në Basic, mbylljen e formularit etj. Të gjitha mundësitë me pullën komandë do t'i paraqesim në tabelat e mëposhtme. Siç shihet nga figura e mëposhtme aksionet me Pullën komandë ndahen në disa kategori ato janë:



- Pullat e navigimit (Record navigation)
- Veprimet me të dhëna (Record Operations)
- Veprimet me formularë (Form Operations)
- Veprimet me raporte (Report Operations)
- Programet (Applications)
- Të ndryshme (Miscellaneous)

Ku secila kategori përmban veprimet (aksionet) përkatëse.

Pullat e navigimit	
Find Next	Gjeje tjetrin
Find Record	Gjeje të dhënën
Go To First Record	Shko te e dhëna e parë
Go To Last Record	Shko te e dhëna e fundit
Go To Next Record	Shko te e dhëna e radhës
Go To Previous Record	Shko te e dhëna paraprake

Veprimet me të dhëna	
Add New Record	Shto një të dhënë
Delete Record	Fshije të dhënën
Duplicate Record	Dyfisho të dhënën
Print Record	Shtyp të dhënën
Save Record	Ruaje të dhënën
Undo Record	Moho të dhënën

Veprimet me formularë	
Apply Form Filter	Zbatoje filtrin
Close Form	Mbyll formularin
Edit Form Filter	Ndrysho filtrin
Open Form	Hape formularin
Open Page	Hape faqen
Print a Form	Shtyp formularin
Print Current Form	Shtyp formularin actual
Refresh Form Data	Rifresko të dhënat në formular

Veprimet me raporte	
Mail Report	Dërgo raportin me e-mail
Preview Report	Pamja e raportit para shtypjes
Print Report	Shtyp raportin
Send Report to File	Dërgo raportin në një datotekë

Programet	
Quit Application	Dalja nga programi
Run Application	Ekzekuto programin
Run MS Excel	Ekzekuto MS Excel
Run MS Word	Ekzekuto MS Word
Run Notepad	Ekzekuto Notepad

Të ndryshme	
Auto dialer	Lidhje automatike me telefon
Print Table	Shtyp tabelën
Run Macro	Ekzekuto një makro
Run Query	Ekzekuto një pyetësor

Foto

Ky element integron një figurë statike në sipërfaqen e formularit.

Fushë e palidhur objekti

Ky element i kontrollit integron në formular një object OLE për të cilin nuk ekziston ndonjë lidhje me tabela ose pyetësorë.

Fushë e lidhur objekti

Ky element integron në formular një fushë të lidhur objekti OLE por që ruhet në fushat përkatëse të tabelave.

Kufi midis faqeve

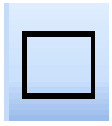
Me ndihmën e këtij elementi të kontrollit caktojmë në një formular kufirin midis faqeve të tij. Pikërisht në atë kufi bëhet ndarja e fletëve në printim.

Element për paraqitjen e formularëve me më shumë faqe

Ky element përdoret për krijimin e formularëve me shumë faqe.

Nënformular/Nënraport

Ky element shërben për integrimin e një formulari brenda formularit tjetër i cili është i lidhur sipas një relacioni 1:n me formularin kryesor.

 Vijë dhe  Drejtëkëndësh

Këto dy elemente më shumë përdoren për nga ana estetike e një formulari.

 Elemente tjera të kontrollit

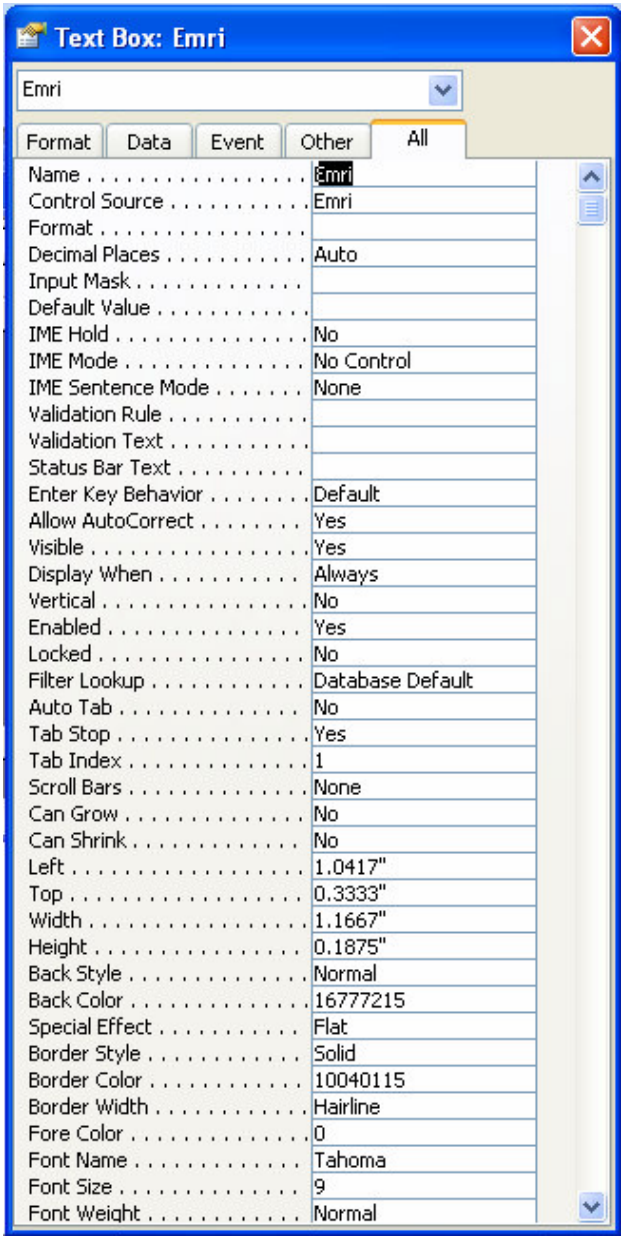
Me ndihmën e kësaj pulle hapet një meny e cila i përmban të gjithë elementet ActiveX . Një nga ato është integrimi i kalendarit me anë të Calendar Control 11.0.

Karakteristikat e fushave të formularit

Karakteristikat e fushave janë atribute që mundësojnë modifikimin e elementeve të kontrollit në formular dhe raport. Pra përdoren për ndryshimin se si të paraqiten të dhënat. Janë disa mënyra për shikimin e karakteristikave të fushave:

- Selektojmë një element të kontrollit shkojmë te View dhe Properties
- Selektojmë një element të kontrollit dhe klikojmë në pullën Properties në toolbar
- Me klikimin dy herë të elementit të kontrollit
- Klikojmë me të djathtën mbi elementin e kontrollit dhe zgjedhim Properties

Ja edhe dritarja e karakteristikave të fushave:

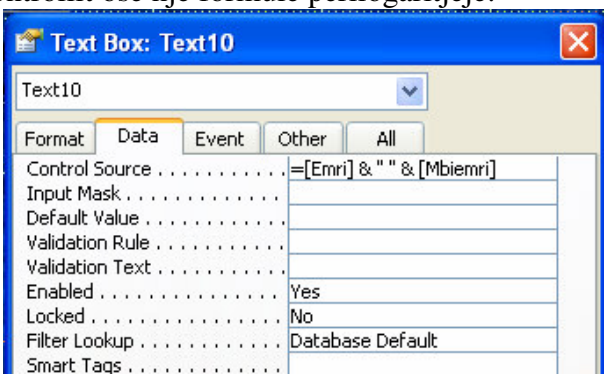


Dritarja e karakteristikave ka pullën **All** ku na mundëson të shikojmë të gjitha karakteristikat. Ose ne mund të zgjedhim ndonjë pullë tjetër për të shikuar karakteristika të kufizuara me një grup të karakteristikave, ato janë:

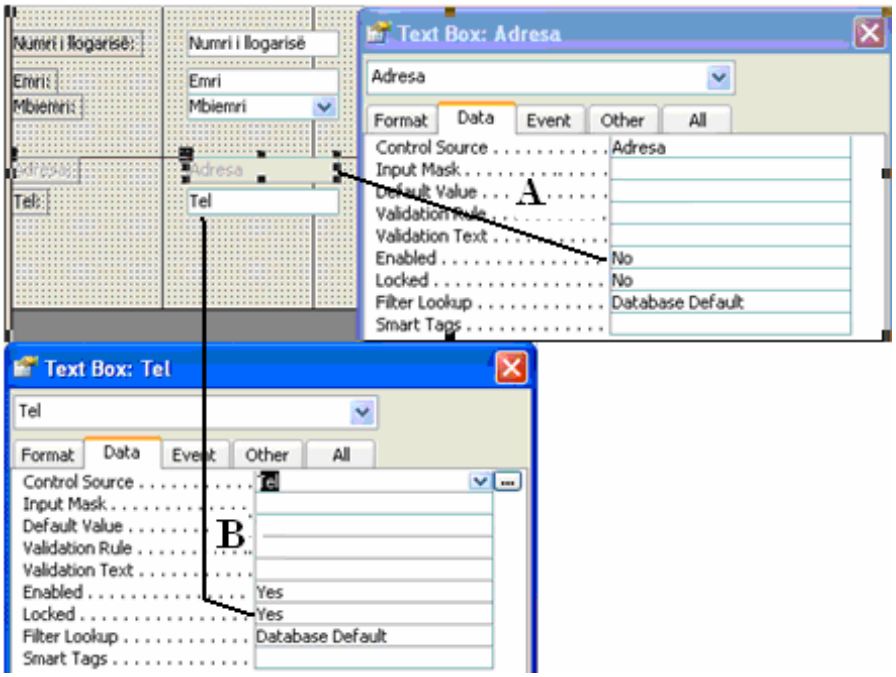
- **Format** – Këto karakteristika përcaktojnë se si një emërtues ose vlerë do të shihet: germat, madhësia, ngjyra, efektet, kufijtë dhe scrollbari.
- **Data** – Këto karakteristika ndikojnë se si do të paraqiten të dhënat dhe me cilin burim të të dhënash do të lidhen: control source, maska hyrëse, vlefshmëria, vlerat e parazgjedhura dhe tipet tjera.
- **Event** – Karakteristikat në këtë grup janë: klikimi, shtimi it ë dhënave, shtypja e ndonjë pulle pastaj për të cilën definohet një përgjigje në formë të ndonjë makro ose VBA procedurë.
- **Other** – Karakteristikat në këtë grup paraqesin karakteristikat shtesë të kontrollit si emri, përshkrimi në status bar.

Disa karakteristika të veçanta të elementeve të kontrollit

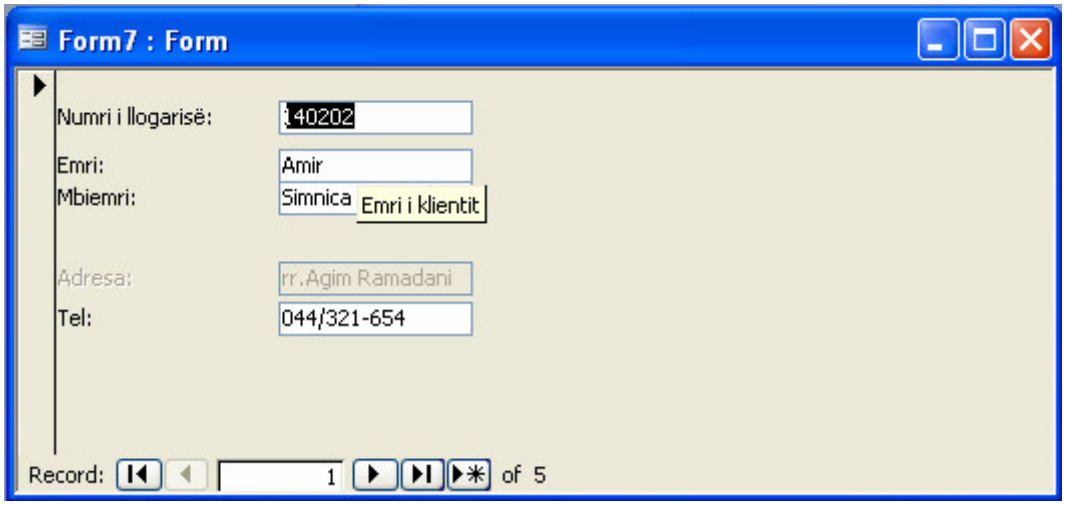
Control Source – në pullën Data te Control Source përcaktohet fusha e të dhënave që i përket elementit të kontrollit ose një formulë përllogaritjeje.



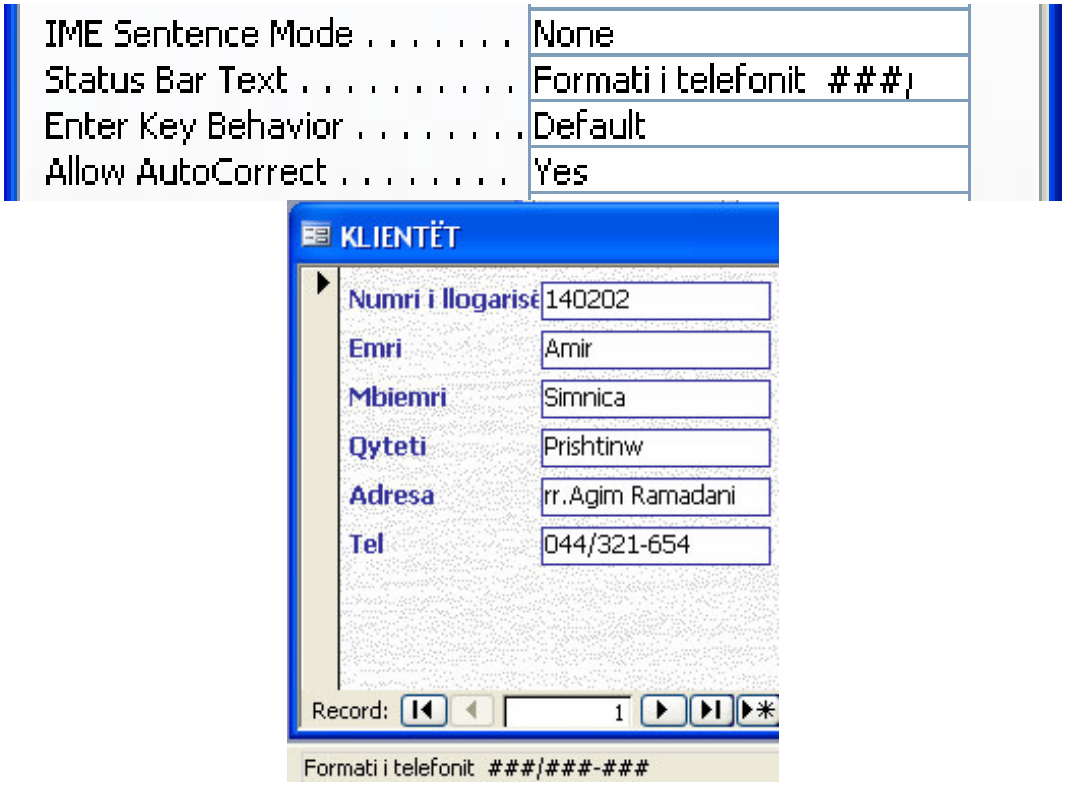
Enabled – në pullën Data te Enabled e bëjmë No dhe e dhëna do të paraqitet e hijëzuar (pamja A).
Locked – në pullën Data te Locked zgjedhim Yes dhe në atë fushë nuk mund të shkruajmë të dhëna (pamja B).



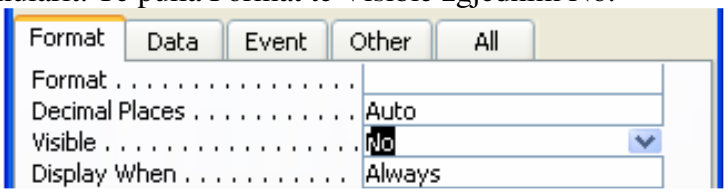
Control Tip Text – përdoret për drejtëkëndëshin e informacionit (QuickInfo), në pullën Other te Control Tip Text shkruajmë tekstin përshkrues informativ.



Status Bar Text – edhe me këtë karakteristikë ofrojmë më shumë informacion për fushat i cili informacion do të paraqitet në status bar. Në pullën Other te Staus Bar Text shkruajmë informacionin e dëshiruar dhe e ruajmë formularin.

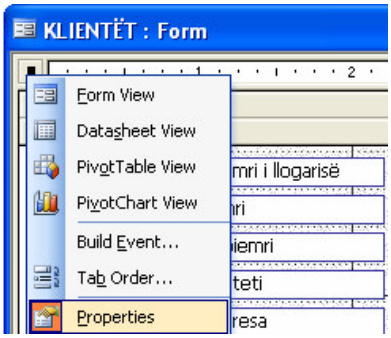


Visible – disa fusha me anë të kësaj karakteristike mund t'i bëjmë të padukshme në sipërfaqen e formularit. Te pulla Format te Visible zgjedhim No.



Allow Edits, Alow Deletions, Allow Additions dhe Data Entry

Së pari shkojmë në anën e majtë lartë të formularit dhe klikojmë me të djathtën dhe zgjedhim Properties.



Pastaj shkojmë te pulla Data dhe aty do t'i shohim këto karakteristika:

Karakteristika	Përshkrimi	E parazgjedhur
Allow Edits	Nëse zgjedhim No atëherë të dhënat në formular nuk do të mund të ndryshohen.	Yes
Allow Deletions	Nëse zgjedhim No atëherë nuk lejohet fshirja e të dhënave në formular.	Yes
Allow Additions	Nëse zgjedhim No nuk lejohet shtimi it ë dhënave	Yes
Data Entry	Nëse zgjedhim Yes atëherë do të mund të shtojmë të dhëna në formular por ato ekzistuese nuk do t'i shohim.	No

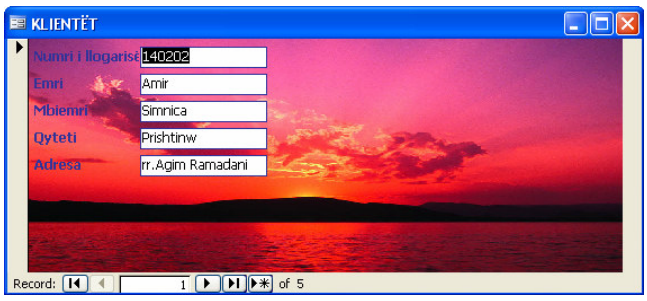
Ndërrimi i prapavisë së formularit me foto

Shkojmë në anën e majtë lartë klikojmë me të djathtën dhe zgjedhim Properties dhe hapet dritarja e karakteristikave. Në atë dritare te pulla Format zgjedhim karakteristikën Picture dhe klikojmë në pullën djathtas  dhe zgjedhim foton që dëshirojmë dhe e ruajmë formularin. Kur të hapim formularin në Form View do të shohim se në prapavijë është foto që kemi zgjedhur.

Fshehja e pullave të navigimit

Njësoj si më lartë shkojmë te Properties, te Navigation Buttons zgjedhim No. Kur të ekzekutohet formulari pullat e navigimit nuk do të shihen.

Që formulari



të merr pamjen

duhet që në Properties si më lartë të veprohet kështu: te Scroll Bars zgjedhim Neither, te Record Selectors zgjedhim No, te Navigation Buttons zgjedhim No, te Border Style zgjedhim None dhe te Control Box zgjedhim No.

Krijimi i nënformularëve

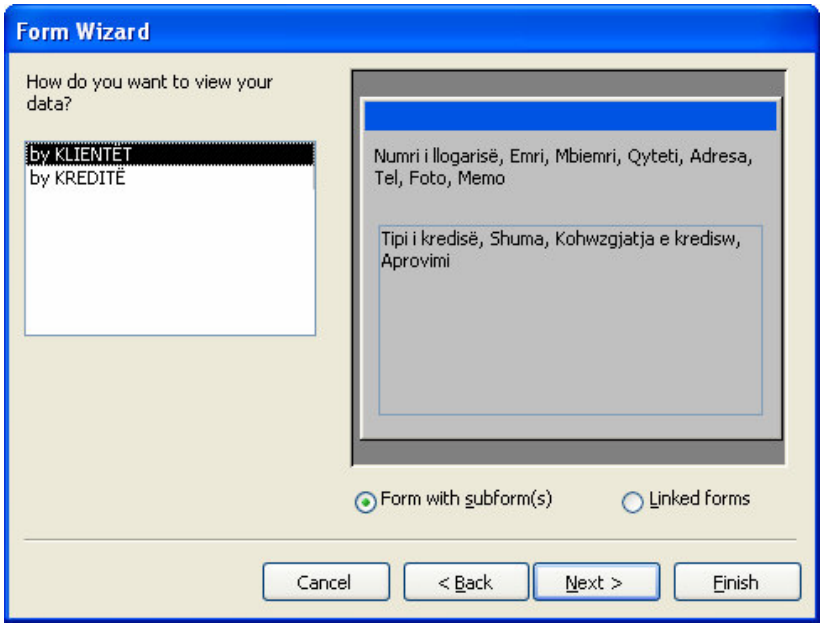
Krijimi i nënformularëve bëhet në tri mënyra:

- Me Asistentin e formularit (Form Wizard)
- Me kutinë e veglave (toolbox)
- Me tërheqjen e nënformularit në formular në dritaren e bazës së të dhënave

Krijimi i nënformularit me Asistentin e formularit (Form Wizard)

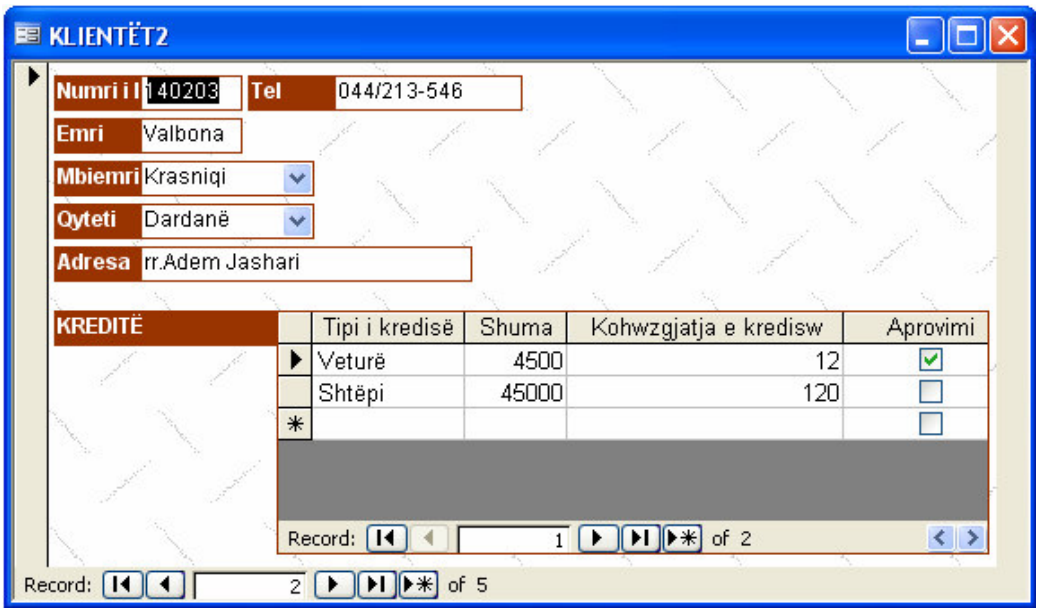
Në dritaren e bazës së të dhënave zgjedhim Forms dhe pastaj shtypim pullën New, hapet dritarja New Form, aty zgjedhim Form Wizard (Asistenti i formularit) ndërsa më poshtë zgjedhim tabelën KLIENTËT dhe shtypim OK. I bartim të gjitha fushat në anën e djathtë me pullën >> . Pastaj zgjedhim tabelën KREDITË te Tables/Queries dhe i bartim fushat Tipi i Kredisë, Shuma, Kohëzgjatja e kredisë dhe Aprovimi në fund shtypim Next. Në rastin tonë tabelat KLIENTËT dhe KREDITË janë të lidhura me lidhjen një me shumë prandaj nuk do të kemi problem për krijimin e këtij formulari me nënformular brenda, përndryshe Access-i do të lajmëroj për gabim.

Pasi të jetë shtypur pulla Next na hapet dritarja ku do të zgjedhim se si do t'i shohim të dhënat.



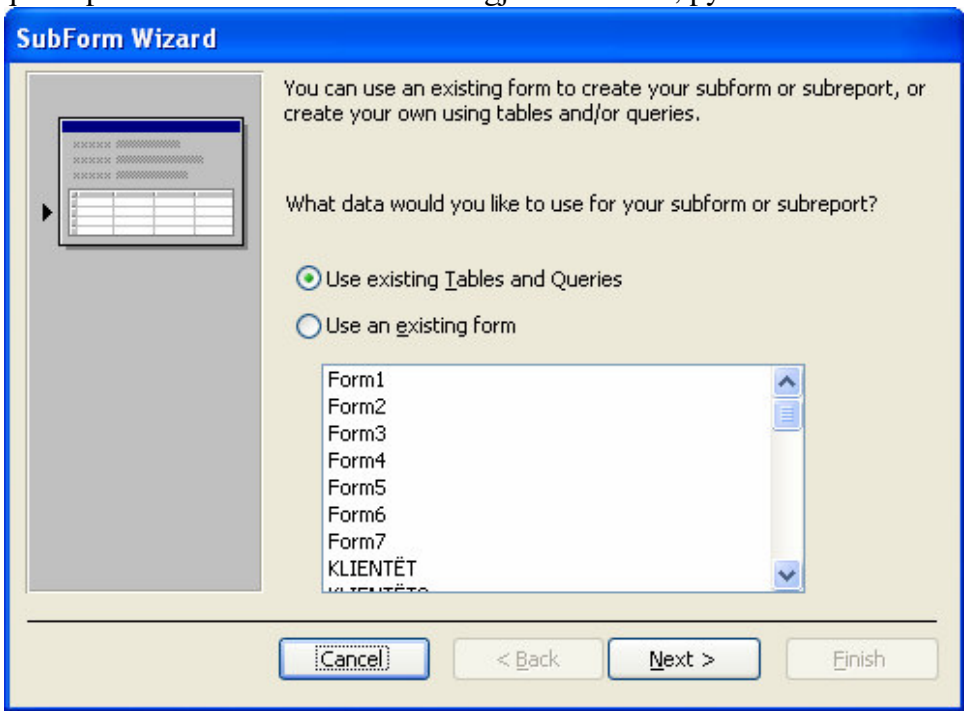
Tani mund të zgjedhim Formular me nënformular ose formular të lidhur (paraqiten ndamas). Zgjedhim Form with subform(s) dhe shtypim Next. Hapet dritarja ku kemi mundësi të zgjedhim renditjen e fushave në formular, zgjedhim njërën prej tyre P.SH.

Datasheet dhe shtypim Next. Nga dritarja tjetër zgjedhim stilin (pamjen) e formularit P.SH. Industrial dhe shtypim Next. Në fund i vejmë emrin formularit dhe nënformularit dhe shtypim Finish. Do të shohim dritaren



Krijimi i nënformularit me anë të kutisë së veglave (toolbox)

Hapim formularin e dëshiruar në Design View dhe nga kutia e veglave zgjedhim pullën Subform/Subreport dhe e vendosim në formular duke caktuar edhe madhësinë e nënformularit dhe hapet dritarja ku do të zgjedhim nënformularin i cili duhet të jetë i krijuar që më parë. Si nënformular mund të zgjedhim tabelë, pyetësor ose formular.



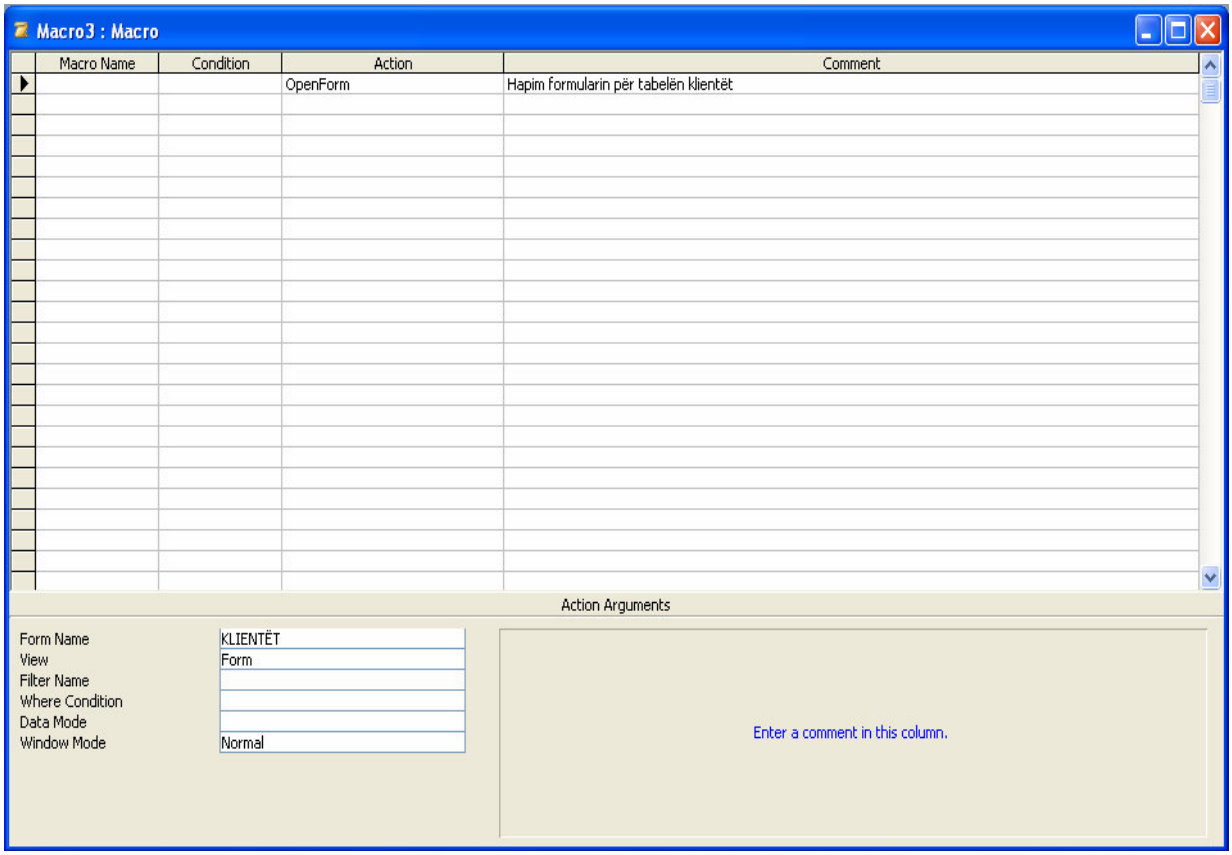
Zgjedhim objektin e dëshiruar pastaj shtypim Next dhe hapet dritarja ku mund të definojmë se cilat fusha nga formulari do të jenë të lidhura me fushat në nënformular. I kemi dy opsione *Choose from a list* pra të zgjedhim nga lista e mëposhtme ose *Define my own* të definojmë vetë lidhjet e fushave. Zgjedhim *Choose from a list* dhe shtypim Next. Emërojmë formularin dhe shtypim Finish. Në fund do të shohim se krijuam një formular me nënformular.

Krijimi i nënformularit me tërheqjen nga dritarja e bazës së të dhënave

Hapim një formular në Design View, atëherë nga dritarja e bazës së të dhënave tërheqim një formular që dëshirojmë të jetë nënformualr dhe e bartim deri te formulari i hapur në fillim. Access-i do të mundohet të bëjë lidhjen në mes këtyre dy formularëve.

Makrot

Makro është një bashkësi e urdhërave në varg që ekzekutohet si një. Makrot në Access automatizojnë aksionet si hapjen e një tabele, shtypjes së një formulari, gjetjes së ndonjë të dhëne ose zbatimin e filterit. Madje makrot mund të shërbejnë edhe për pulla komanduese, krijimin e menyve dhe toolbareve. Makro përbëhet nga aksioni dhe argumentet. Aksioni është puna që duhet kryer si hapja e një formulari ndërsa argumentet përcaktojnë specifikat e aksionit, si cili formular duhet të hapet. Makrot në Access krijohen në dritaren Macro Design. Kjo dritare ka dy pjesë, pjesa e sipërme quhet paneli i aksionit, i cili përmban deri katër kolona. Ku dy prej tyre janë të parazgjedhura: kolona Action dhe kolona Comment ndërsa dy kolonat tjera mund t'i bëjmë të dukshme duke shkuar te View dhe zgjedhur Macro Name dhe Conditions. Kolona Action përmban një nga shumë aksionet në dispozicion, kolona Comment shërben për përshkrimin e aksionit. Përshkrimi i aksionit duket të jetë jo i rëndësishëm kur të krijohet makroja por më vonë është treguar se është shumë i dobishëm në rastet kur dëshirojmë që të ndryshojmë makron. Kolona Macro Name përmban emrin e makros me të cilin do t'i referohemi, kolona Conditions përmban kushtin që duhet plotësuar në mënyrë që të ekzekutohet makroja.
Pjesa tjetër, pjesa e poshtme quhet Action Arguments dhe ndryshon nga pamja në varësi se cilin aksion kemi zgjedhur. Shumica e aksioneve kërkojnë më shumë se një argument.



Krijimi i një makroje

Hapim bazën e të dhënave, te dritarja e bazës klikojmë te Macros dhe pastaj klikojmë pullën New. Do të hapet dritarja Macro Design. Te Action zgjedhim OpenForm, te Comments shkruajmë: Hapim formularin për tabelën klientët, ndërsa te Action Arguments, te Form Name zgjedhim formularin KLIENTËT. Në fund e ruajmë makron me emrin Test-makro. Këtë makro mund ta provojmë duke krijuar një pullë komande në ndonjë formular, aty zgjedhim kategorinë Miscellaneous dhe aksionin Run Macro. Aty zgjedhim pastaj makron me emrin Test-makro dhe emërojmë pullën Hape formularin. Kur të hapet formulari në Form View dhe shtypet pulla Hape formularin atëherë do të hapet formulari KLIENTËT.

Siç u pa me krijimin e makros kur klikuam në kolonën Action u hap një listë e aksioneve. Një makro mund të përmbajë 999 aksione, secili aksion në rresht të veçantë. Në tabelën e mëposhtme janë disa nga aksionet:

Aksioni	**Përshkrimi**
AddMenu	Shton meny në një meny bar
ApplyFilter	Zbaton një filter ose pyetësor
Close	Mbyll një dritare të specifikuar
CopyObject	Kopjon një object të bazës në një bazë tjetër ose në të njëjtën bazë me emër të ndryshëm
DeleteObject	Fshin objektin e specifikuar
HourGlass	Ndryshon pamjen e treguesit të miut në orë të rërës.
MsgBox	Hap një dritare që përmban tekst informative ose lajmërues
OpenForm	Hap një formular
OpenDiagram	Hap një diagram
OpenQurey	Hap një pyetësor
OpenReport	Hap një raport
OpenTable	Hap një tabelë
Quit	Mbyll Access-in
Rename	Riemeron objektin e caktuar
RunApp	Starton një program
RunCode	Ekzekuton një procedurë të Microsoft Access Basic
RunMacro	Ekzekuton një makro
Save	Ruan objektin e caktuar
SetWarnings	Kthen sistemin e porosive On dhe Off
StopAllMacros	Ndalon të gjitha makrot e ekzekutuara
TransferDatabase	Importon ose eksporton të dhëna në mes të dy bazëve të të dhënave
TransferText	Importon ose eksporton tekst në mes të bazës active dhe një tekst datoteke

Raportet

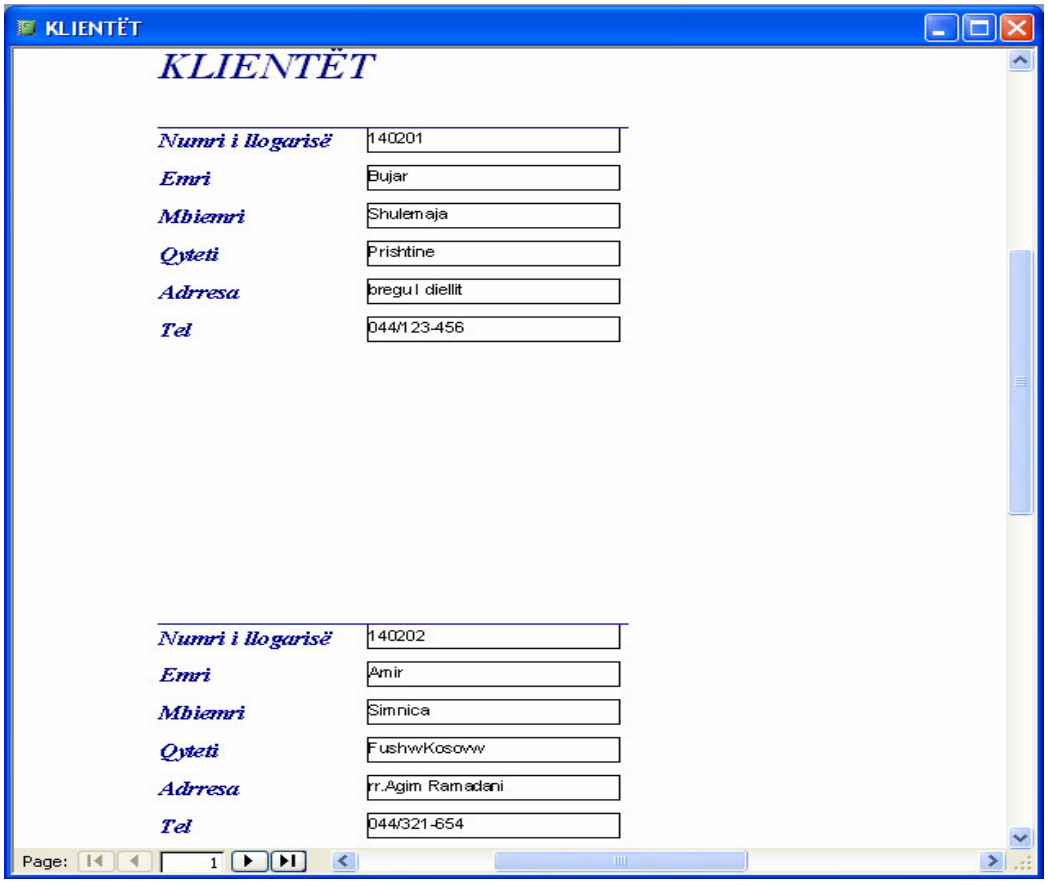
Raportet paraqesin një formë të përshtatshme për shikimin e të dhënave duke na mundësuar edhe shypjen e informacioneve në çfarëdo formati. Me anë të raporteve mund të bëjmë grupimin e të dhënave, kombinimin e tyre sipas dëshirës. Mund të krijojmë kalkulime si vlera mesatare ose kalkulime tjera statistikore dhe paraqitjen e tyre grafike. Raportet mund të përmbajnë të dhëna nga tabela ose tabela të lidhura dhe nga pyetësorët.

Tipet e paraqitjes së raporteve

Tipet bazë për paraqitjen e raporteve janë:

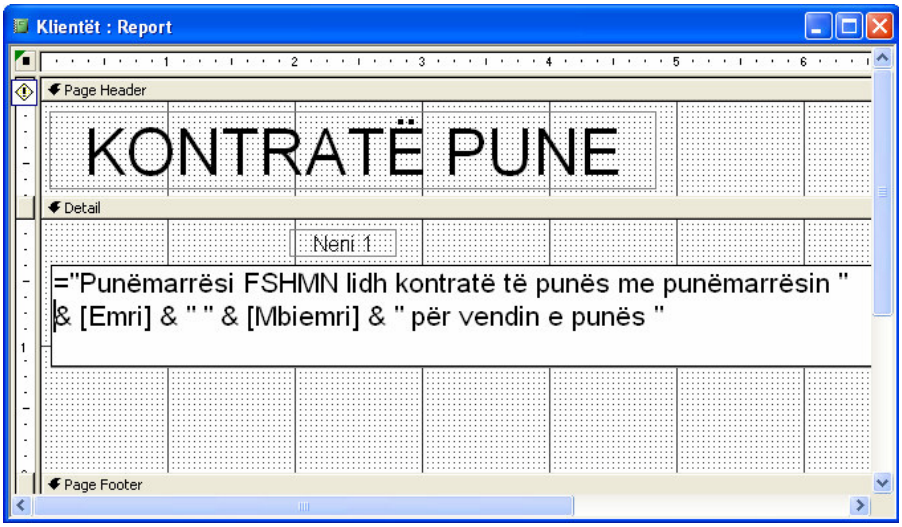
- **Columnar** – paraqitja e fushave në një kolonë, të dhënat paraqiten njëra mbi tjetrën.
- **Tabular** – paraqitja e të dhënave në rreshta dhe kolona me grupime.
- **Mail-merge reports** – krijimi i vërtetimeve, kontratave etj.
- **Mailing labels** – paraqitja e të dhënave në etiketa të madhësive të ndryshme.

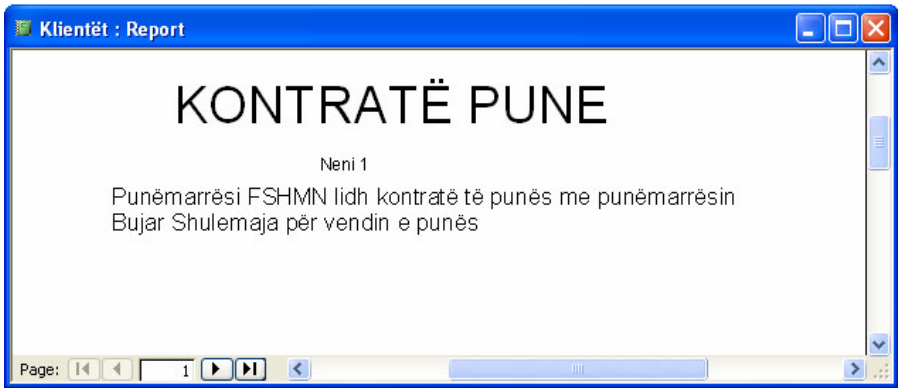
Figura e mëposhtme tregon paraqitjen e të dhënave me anë të tipit Columnar.



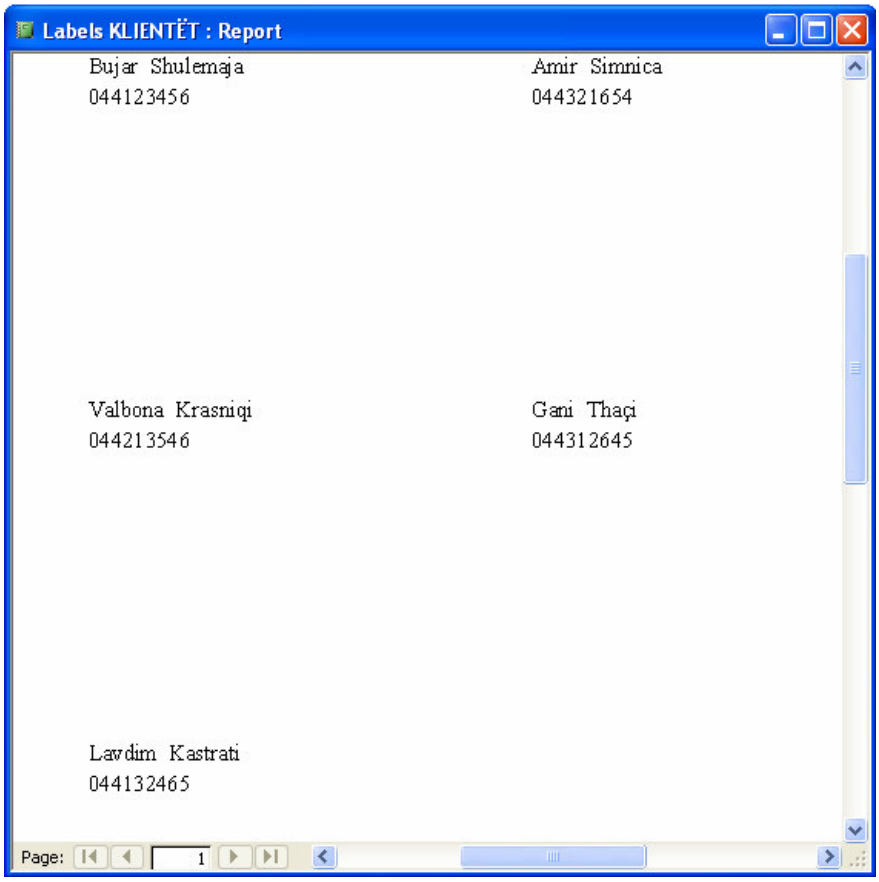
Paraqitja e të dhënave për printim në një fletë në rreshta dhe kolona është paraqitja me anë të tipit Tabular, si në figurën e mëposhtme.

Krijimi i vërtetimeve , kontratave etj bëhet me anë të tipi Mail-merge reports. Kjo bëhet në Design View si figurën e mëposhtme.





Mailing labels paraqet të dhënat në etiketa të madhësive të ndryshme, si në figurën e mëposhtme.



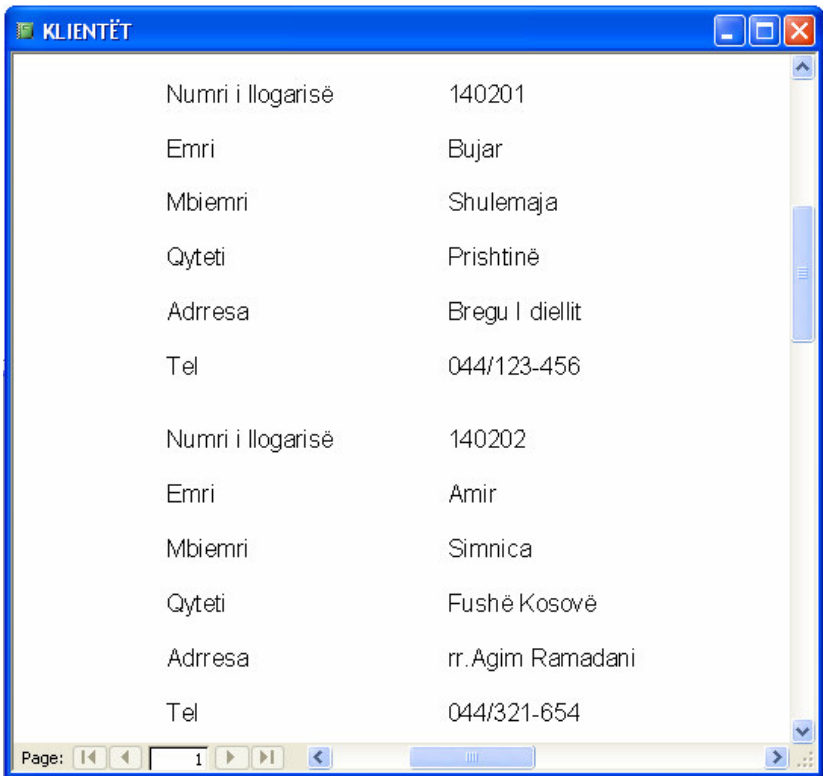
Këto janë tipet kryesore të krijimit të raporteve, ndërsa mënyrat për krijimin e tyre janë me anë të:

- AutoReport
- Design View
- Asistentit të raportit (Report Wizard)
- AutoReport: Columnar
- AutoReport: Tabular
- Chat Wizard
- Label Wizard

Krijimi i raportit me anë të AutoReport

Mënyra më e shpejtë për krijimin e raportit është me anë të AutoReport. Me anë të AutoReport mund të krijojmë raporte vetëm nga tabelat e krijuara më parë dhe nuk kemi kontroll mbi dizajnimin e atyre raporteve.
Në dritaren e bazës së të dhënave klikojmë te Tables dhe zgjedhim një tabelë me anë të së cilës dëshirojmë të krijojmë raportin. Pastaj shkojmë te pulla New Object zgjedhim AutoReport(ose Insert dhe AutoReport)dhe raporti do të krijohet automatikisht dhe do të shfaqet.

Shembull:
Në bazën e të dhënave Banka te objektet Tables zgjedhim tabelën KLIENTËT, pastaj në toolbar te pulla New Object zgjedhim AutoReport dhe do të krijohet raporti i mëposhtëm.

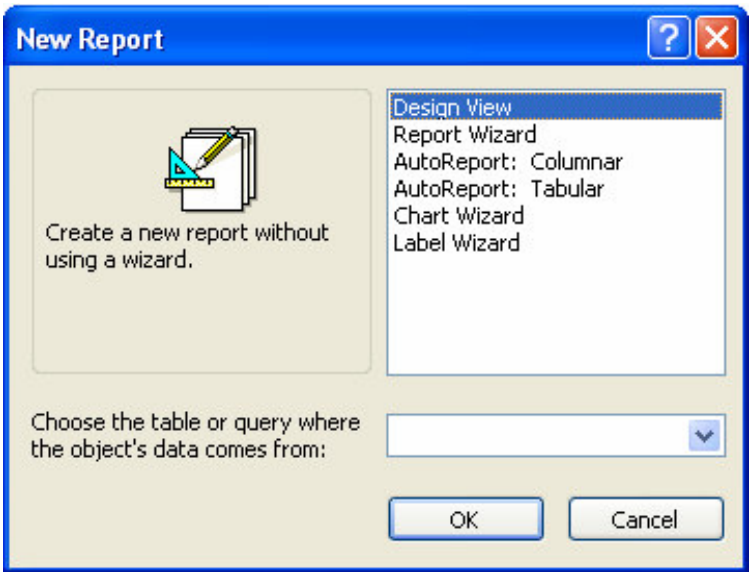


Krijimi i raportit me anë të Asistentit të raportit (Report Wizard)

Me anë të Asistentit të raportit e thjeshtësojmë krijimin e raportit. Asistenti i raportit na udhëheq nëpër një varg pyetjesh për krijimin e raportit. Sikurse Asistenti i formularit edhe Asistenti i raportit na mundëson pamje bazike për raportin të cilën pastaj mund ta ndryshojmë. Ekzistojnë disa mënyra për të aktivizuar Asistentin e raportit:

- Nga dritarja e bazës së të dhënave zgjedhim Insert dhe Report
- Nga objektet e bazës zgjedhim Reports dhe klikojmë në pullën New në dritaren e bazës së të dhënave
- Në toolbar zgjedhim pullën New Object dhe Report

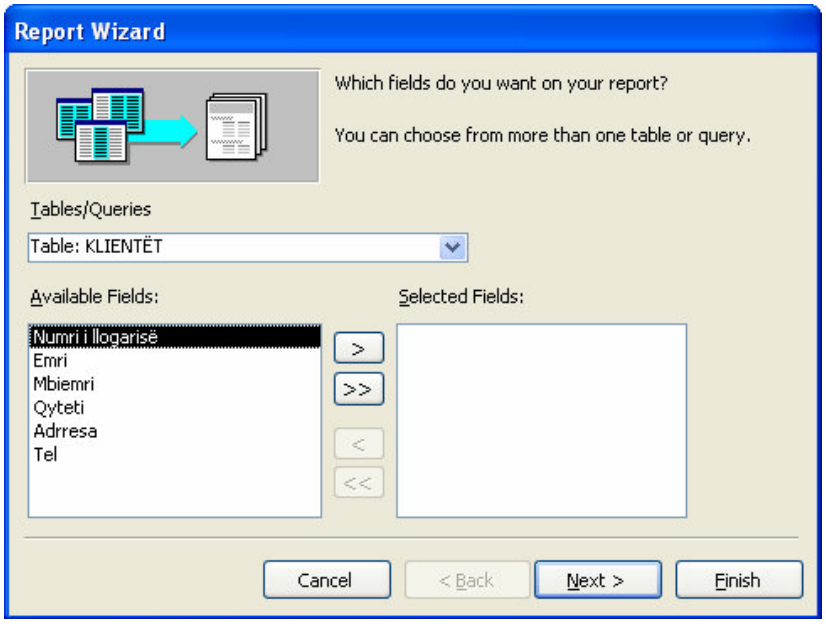
Pamarrë parasysh se cilën mënyrë e zgjedhim për krijimin e raportit do të hapet dritarja e mëposhtme.



Dritarja New Report përmban gjashtë zgjedhje për krijimin e raportit:

- Design View – paraqet një raport plotësisht të zbrazët
- Asistentit të raportit (Report Wizard) – krijon raportin nga tri mundësi: Columnar, Tabular dhe Justified
- AutoReport: Columnar – krijon raportin në bazë të tipit Columnar
- AutoReport: Tabular – krijon raportin në bazë të tipit Tabular
- Chat Wizard – krijon raportin në bazë të grafeve të ndryshme
- Label Wizard – Asistenti i krijimit të etiketave me madhësi të ndryshme

Për startimin e Asistentit të raportit zgjedhim Report Wizard.
Në pjesën e poshtme të dritares mund të zgjedhim tabelën ose pyetësorin (të krijuar më parë) për krijimin e formularit. Pra te *Choose the table or query ëhere the object's data comes from:* zgjedhim objektin e dëshiruar dhe klikojmë OK. Pastaj na paraqitet dritarja për zgjedhjen e fushave nga objektet e bazës. Pra mund të zgjedhim fusha nga një ose më shumë tabelave ose pyetësorëve.



Fushat që dëshirojmë të jenë në raport i zgjedhim dhe pastaj i shtojmë në anën e djathtë më anë të këtyre pullave:

Shto fushën e zgjedhur

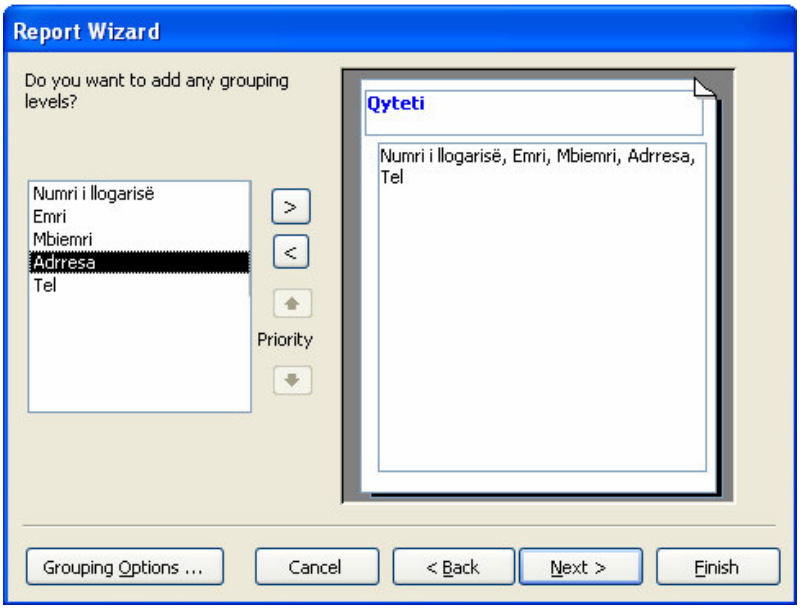
Shto tërë fushat

Largo fushën e zgjedhur

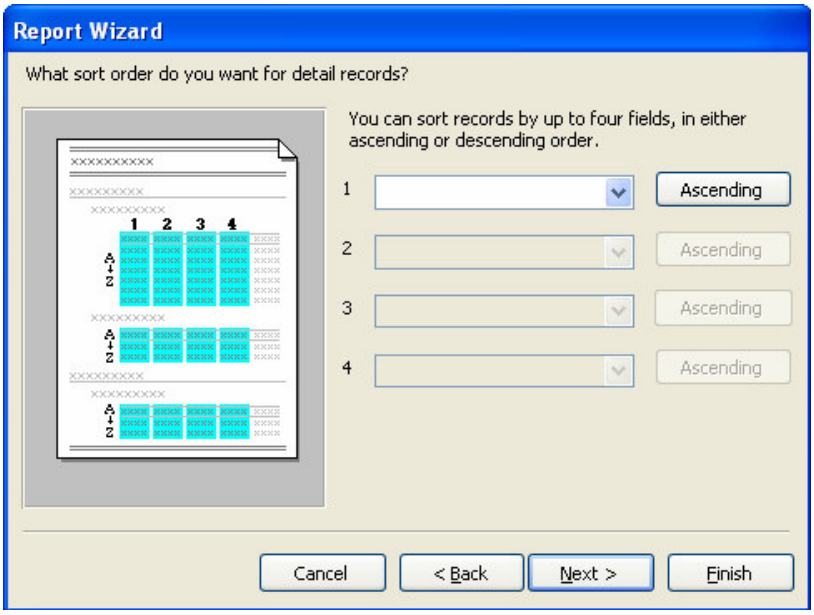
Largo tërë fushat

Pra mund të zgjedhim fusha nga më shumë se një tabelë ose pyetësor.
Pas zgjedhjes së fushave shtypim pullën Next dhe hapet dritarja për përcaktimin e paraqitjes së të dhënave në raport. Këtu përcaktojmë se në bazë të cilës fushë do të bëjmë grupimin e të dhënave. Kjo zgjedhje bëhet me anë të shigjetave. Zgjedhim fushën Qyteti për grupimin e të dhënave në rastin tonë. Mund të zgjedhim më shumë se një fushë për të bërë grupimin e të dhënave. Nëse nuk zgjedhim fushë për grupimin e të dhënave atëherë

gjatë krijimit të raportit do të paraqitet dritarja ku do të kemi mundësi të zgjedhim renditjen e fushave sipas tipit: Columnar, Tabular ose Justified.

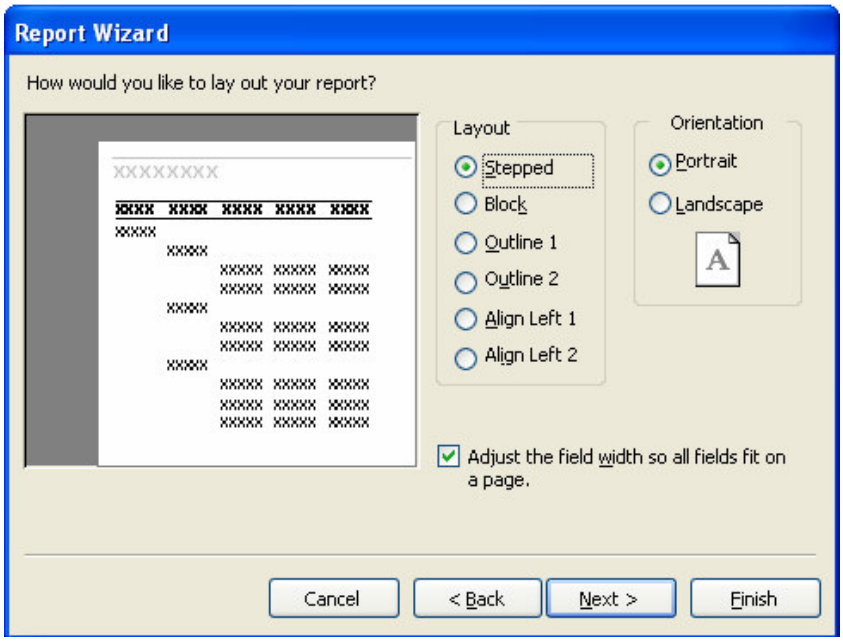


Shtypim pullën Next dhe hapet dritarja

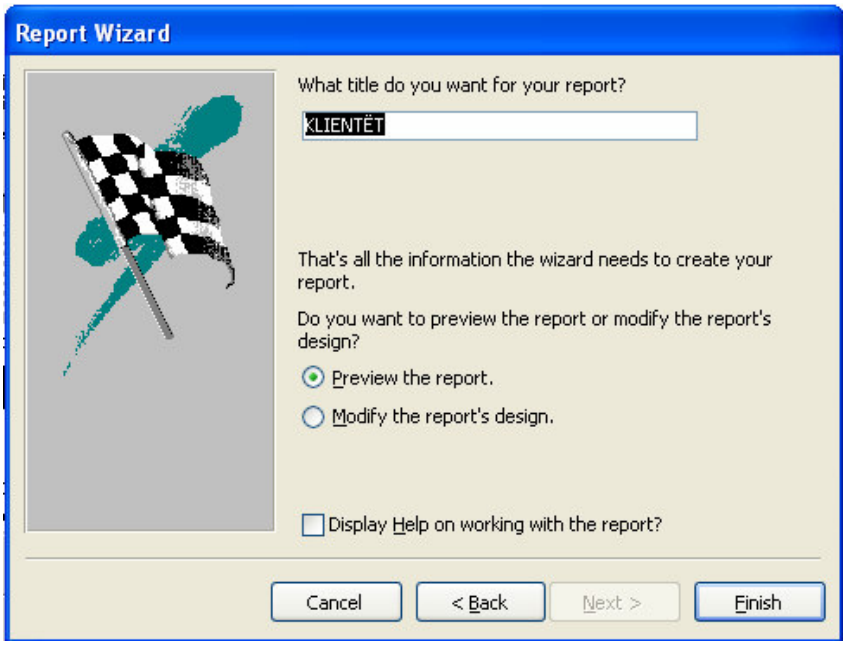


E cila mundëson që të bëjmë sortimin e të dhënave sipas rregullave: në rritje A deri në Z ose 0 deri në 9 dhe në zbritje Z deri në A ose 9 deri në 0. Bëjmë zgjedhjen e dëshiruar dhe shtypim pullën Next. Pastaj paraqitet dritarja e përzgjedhjes së pamjes së të dhënave në raport. Aty kemi gjashtë zgjedhje që mund t'i shohim në anën e majtë të dritares me klikimin në anën e djathtë të zgjedhjeve. Aty është edhe

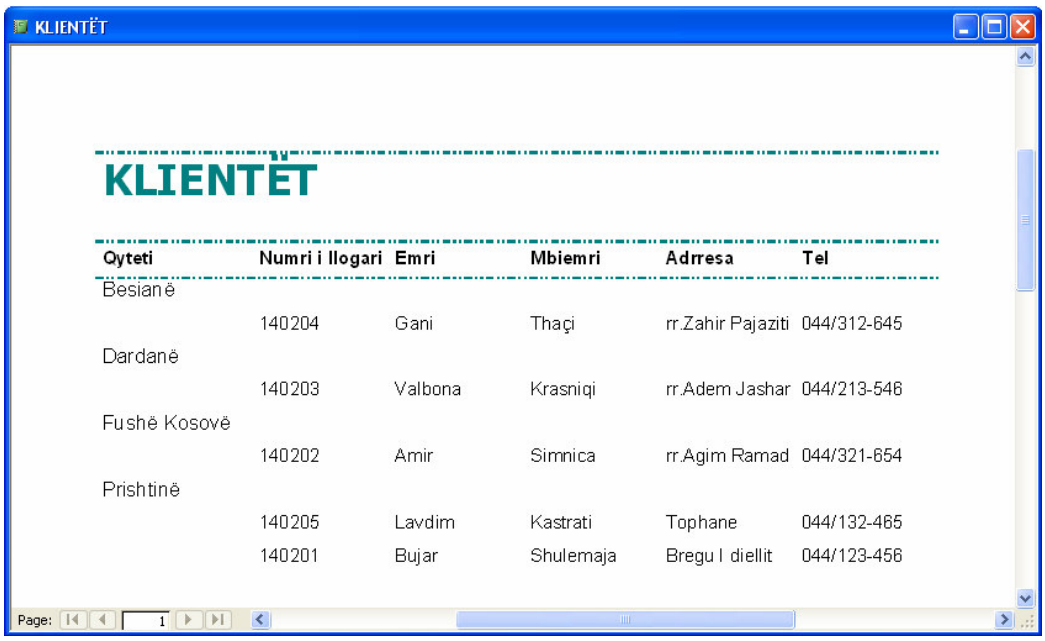
zgjedhja për fletën e raportit se si të ketë pamjen Portrait ose Landscape P.SH. zgjedhim Stepped dhe Portrait dhe shtypim pullën Next.



Pas zgjedhjes së pamjes, hapet dritarja për zgjedhjen e stilit të paraqitjes së të dhënave (si në figurën e mëposhtme). Secili stil ka prapavijë të ndryshme, shkronja, madhësi të ndryshme. Ku secili që klikohet, në anën e majtë mund të shihet pamja e tij. Zgjedhim Casual dhe shtypim pullën Next.



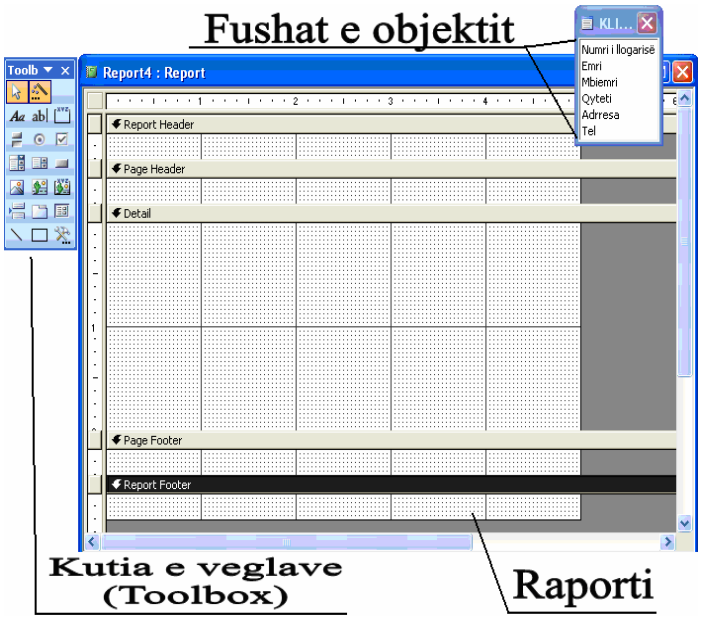
Në fund paraqitet dritarja për emërimin e raportit, si dhe zgjedhjes se a dëshirojmë të bëjmë modifikim në Design View apo raporti të hapet në formën e tij përfundimtare. Shtypim pullën Finish dhe raporti do të duket kësisoj:



Ky raport mund të printohet : File dhe Print dhe hapet dritarja për printim e Microsoft Windows , pra njësoj si te programet Microsoft Office. Ruajtja e raportit bëhet duke shkuar në File dhe Save ose File dhe Save As.

Krijimi i raportit në Design View

Krijimi i raportit me anë të Asistentit të raportit nuk na jep shumë kontroll mbi dizajnimin e tij, por siç do të shohim me krijimin e raportit në Design View ne kemi mundësi të shumta në dizajnimin e raportit. Në dritaren e bazës klikojmë Reports shtypim pullën New dhe zgjedhim Design View, ndërsa te *Choose the table or query where the object's data comes from:* zgjedhim tabelën ose pyetësorin me anë të cilit do të krijojmë raportin.



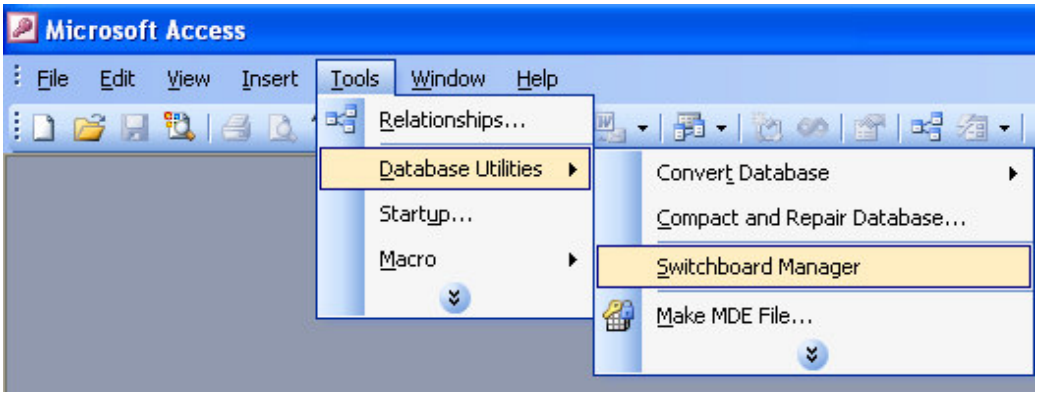
Elementet kryesore përbërëse të pamjes Design View të raportit janë: zona e raportit dhe elementet e kontrollit. Raporti përbëhet nga Kreu i raportit (Report Header), Kreu i faqes (Page Header), Detalet (Detail), Fundi i faqes (Page Footer) dhe Fundi i raportit (Report Footer). Që të shihen këto pjesë përbërëse të formularit shkojmë në menynë kryesore te View dhe zgjedhim Page Header/Footer dhe Report Header/Footer.
Veprimet me elementet e kontrollit dhe përdorimi i kutisë së veglave (toolbox) është i njëjtë si te formularët. Karakteristikat e fushave në raport janë njësoj si te formularët të gjitha vetitë që u paraqitën te formularët vlejnë edhe për raportet. Poashtu edhe krijimi i nënraporteve bëhet në mënyrë të njëjtë si te formularët.

Dritarja komanduese (Switchboard)

Më herët kemi mësuar për krijimin e objekteve të bazës së të dhënave si tabelat, pyetësorët, formularët, raportet dhe makrot. Pastaj kemi punuar me secilin object veçmas, ndërsa me anë të dritares komanduese do t'i bashkojmë këto objekte në një program pa pasë nevojë të shkruajmë ndonjë program kompleks për baza të të dhënave. Me anë të dritares komanduese do të bëjmë automatizimin e funksionimit të bazës së të dhënave.
Në Access dritarja komanduese (switchboard) është një formë që lejon shfrytëzuesin të ekzekutojë urdhëra nga një dritare kryesore. Në fakt dritarja komanduese është një formular i cili nuk është i lidhur me ndonjë tabelë ose pyetësorë. Dritarja komanduese përmban pulla komanduese ose objekte tjera që ekzekutojnë makrot. Krijimi i një dritare komanduese duhet të jetë hapi i fundit i krijimit të një baze të të dhënave. Janë disa mënyra për krijimin e driatres komanduese, mund të shfrytëzojmë Switchboard Manager-in ose me anë të një formulari të zbrazët. Nëse shfrytëzojmë Switchboard Manager-in për krijimin e dritares komanduese atëherë Access-i automatikisht do të krijojë një tabelë e cila përshkruan aksionet që kryejnë pullat në dritaren komanduese. Poashtu mund të krijojmë dritaren komanduese në Design View duke përdorë një formular të zbrazët. Nëse krijojmë një formular i cili do të shfrytëzohet për dritaren komanduese atëherë ai formular nuk duhet të jetë i lidhur me ndonjë tabelë ose pyetësorë. Që ky formular të ketë pamjen e një dritare komanduese duhet që në karakteristikat (properties) e formularit të bëjmë disa ndryshime si: fshehja e pullave të navigacionit, scrollbareve, kufirit të formularit etj.

Krijimi i dritares komanduese

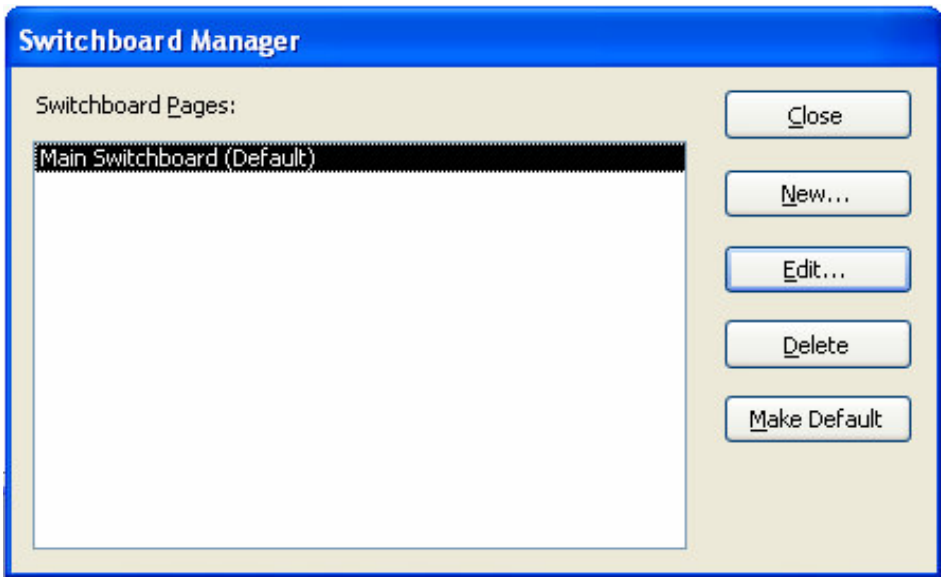
Nga menyja zgjedhim Tools pastaj Database Utilities dhe Switchboard Manager.



Do të hapet dritarja me porosinë:

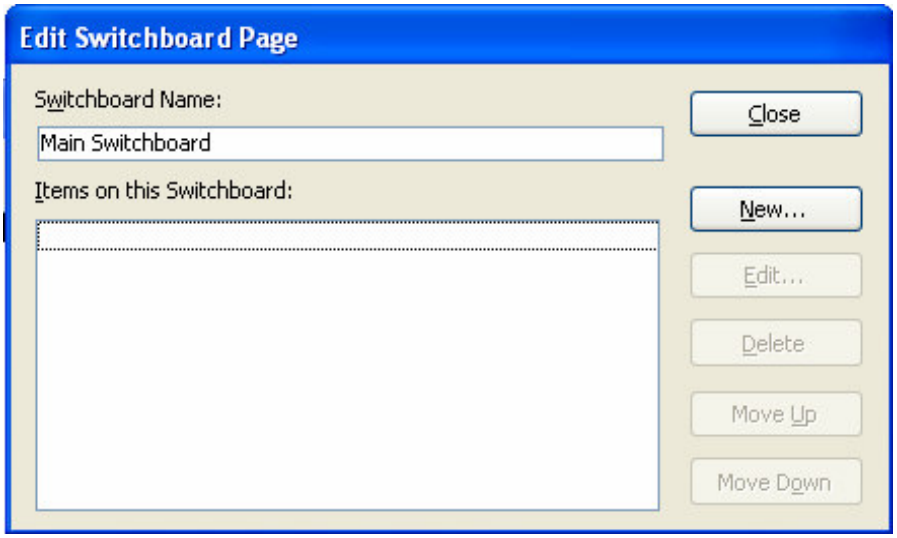
Switchboard Manager-i nuk ka gjetë ndonjë dritare komanduese. A dëshironi të krijoni një?

Klikojmë në Yes dhe hapet dritarja.

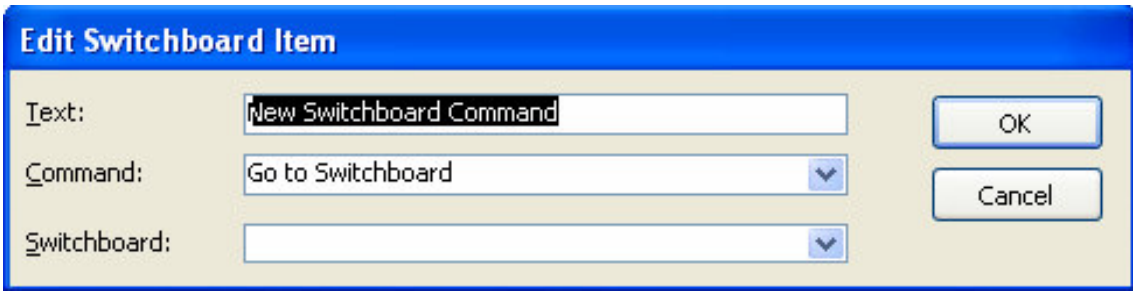


Close – e mbyll dritaren komanduese
New - krijimi i një dritare komanduese të re
Edit - ndryshimi i dritares komanduese të selektuar
Delete – fshirja e dritares komanduese të selektuar
Make Default – e bën dritaren komanduese të selektuar të parazgjedhur

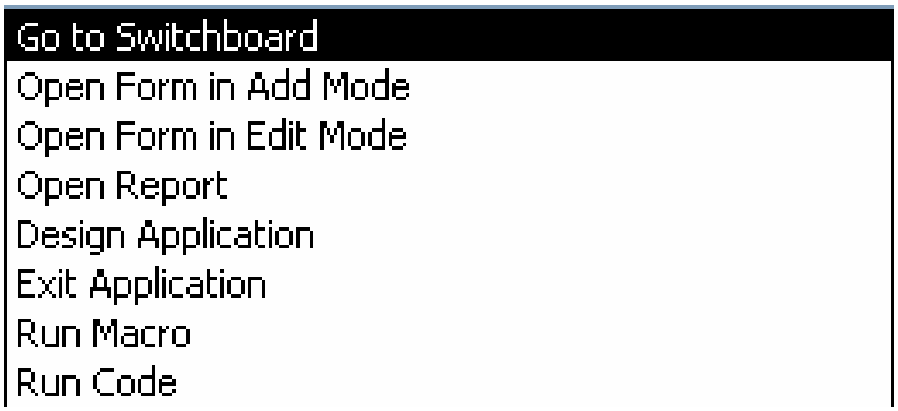
Shtypim Edit dhe hapet dritarja.



Në fushën Switchboard Name: shkruajmë emrin me të cilin dëshirojmë që ta ruajmë dritaren komanduese, ndërsa me anë të pullës New krijojmë pulla të reja në dritaren komanduese. Shtypim pullën New dhe hapet dritarja.

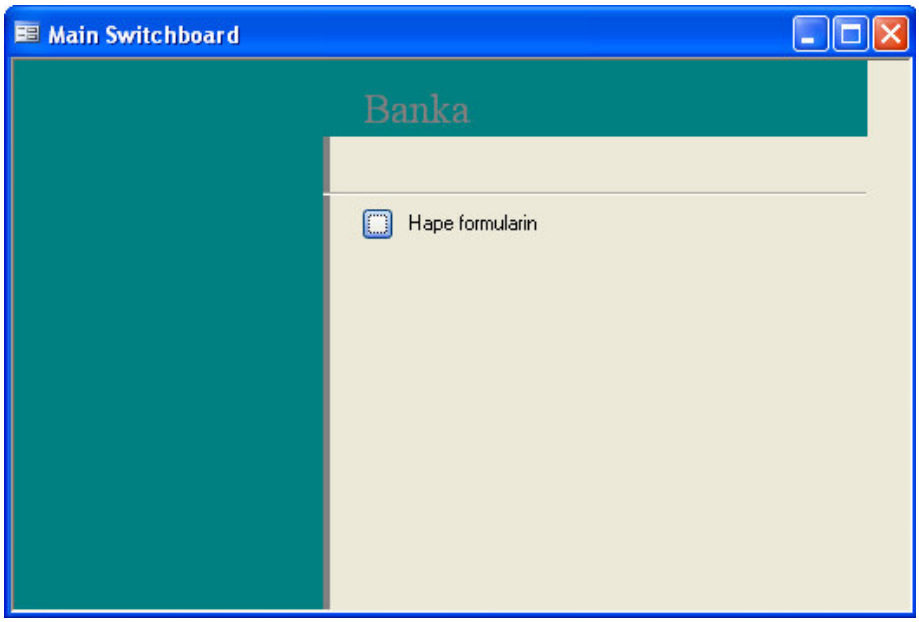


Te fusha Text: shkruajmë tekstin që do të paraqitet për pullën komanduese.
Te Command: zgjedhim urdhërin që do të kryej ajo pullë komanduese, ku aty janë disa mundësi të treguara më poshtë:



Urdhëri	Përshkrimi
Go to Switchboard	Hape dritaren komanduese
Open Form in Add Mode	Hape formularin për shtimin e të dhënave
Open Form in Edit Mode	Hape formularin për ndryshimin e të dhënave
Open Report	Hape raportin
Design Application	Hape dritaren komanduese për ndryshime në dizajn
Exit Application	Mbyll bazën e të dhënave
Run Macro	Ekzekuto makron
Run Code	Ekzekuto kodin

Zgjedhim njërën nga mundësitë dhe shtypim OK. Dritarja komanduese do të ketë pamjen.



Krijimi i dritares komanduese me Design View, bëhet si te krijimi i formularëve vetëm se aty i fshehim pullat e navigacionit, kufirin e formularit, pullat për minimizim, maksimizim dhe mbyllje.
Pasi të krijohet dritarja komanduese është mirë që ajo të jetë startuese. Pra sa herë që hapet baza e të dhënave të hapet dritarja komanduese. Për këtë shkojmë te Tools pastaj StartUp dhe te Display Form/Page zgjedhim dritaren komanduese.

Maksimizimi i dritares komanduese

Nëse dëshirojmë që kjo dritare të maksimizohet sa tërë monitori dhe nga dritarja të largohen : pullat e navigacionit, pullat për minimizim, maksimizim dhe mbyllje veprojmë kështu.
Hapim dritaren komanduese në Design View dhe shtypim pullën Properties në toolbar. Pastaj hapet dritarja e karakteristikave dhe aty i bëjmë këto ndryshime:

Scrollbar	Neither
Record Selectors	No
Navigation buttons	No
Dividing lines	No
Pop up	Yes
Auto resize	Yes
Border style	None
Control box	No
Min Max buttons	None
Close button	No
On Open	Event procedure dhe klikojmë në tri pikat dhe hapet dritarja ku e shkruajmë kodin: **Private Sub Form_Open(Cancel As Integer)** **DoCmd.Maximize** **End Sub**

Paraqitja e dritares përshëndetëse kur të starton dritarja komanduese

Së pari krijojmë një makro me emrin Përshëndetje, ku në kolonën Action do të zgjedhim MsgBox, ndërsa në pjesën e poshtme pra te Action Arguments te rreshti Message shkruajmë: Përshëndetjë, pastaj te rreshti Type zgjedhim Information, te rreshti Title shkruajmë titullin e dritares përshëndetëse. Në fund e ruajmë makron. Pastaj hapim dritaren komanduese në Design View dhe shtypim pullën Properties. Te Event gjejmë karakteristikën On Open dhe zgjedhim makron Përshëndetje, i ruajmë ndryshimet e bëra dhe e hapim dritaren komanduese. Do të shohim se do të paraqitet një dritare me tekstin Përshëndetje. Poashtu mund të bëjmë që dritarja të paraqitet edhe kur të mbyllet dritarja komanduese, shkojmë te Properties, Event dhe te karakteristika On Close zgjedhim makron që dëshirojmë.

MYSQL

Instalimi i MySQL në Windows

Kërkesat e sistemit operativ Windows

Për të punuar me MySQL në Windows duhet të plotësohen këto kushte:

- Sistemi operativ Windows 32 bitësh si: 9x, Me, NT, 2000, XP ose Windows Server 2003
- Përkrahje të protokollit TCP/IP
- Një kopje të MySQL binare për Windows
- Programi që do të lexoj .zip datotekat
- Hapësirë të mjaftueshme në hard disk (në përgjithësi rekomandohet 200 MB)

Zgjedhja e paketës instaluese

Për MySQL 5.0 janë tri paketa instaluese prej ku mund të bëjmë instalimin e MySQL në Windows, ato janë:

- Paketa esenciale (Essential package) : Kjo paketë e ka emrin mysql-essential-5.0.18-win32.msi dhe përmban minimumin e datotekave të nevojshme për instalimin e MySQL në Windows, duke përfshirë edhe Asistentin e konfigurimit (Configuration Wizard).
- Paketa e plotë (Complete package):Kjo paketë e ka emrin mysql-5.0.18-win32.zip dhe përmban datotekat për instalimin e plotë të MySQL në Windows, duke përfshirë edhe Asistentin e konfigurimit (Configuration Wizard). Kjo paketë përmban komponente opsionale si serverin.
- Paketa joinstaluese (The noninstall archive): Kjo paketë ka emrin mysql-noinstall-5.0.18-win32.zip dhe përmban të gjitha datotekat që janë në paketën e plotë duke përjashtuar Asistentin e konfigurimit. Kjo paketë nuk përmban instalimin e automatizuar prandaj duhet të instalohet dhe konfigurohet në mënyrë joautomatike.

Instalimi i MySQL-it me anë të Asistentit të instalimit (Installation Wizard)

Paketa esenciale ose e plotë

Shkarkimi dhe startimi i Asistentit për instalimin e MySQL-it

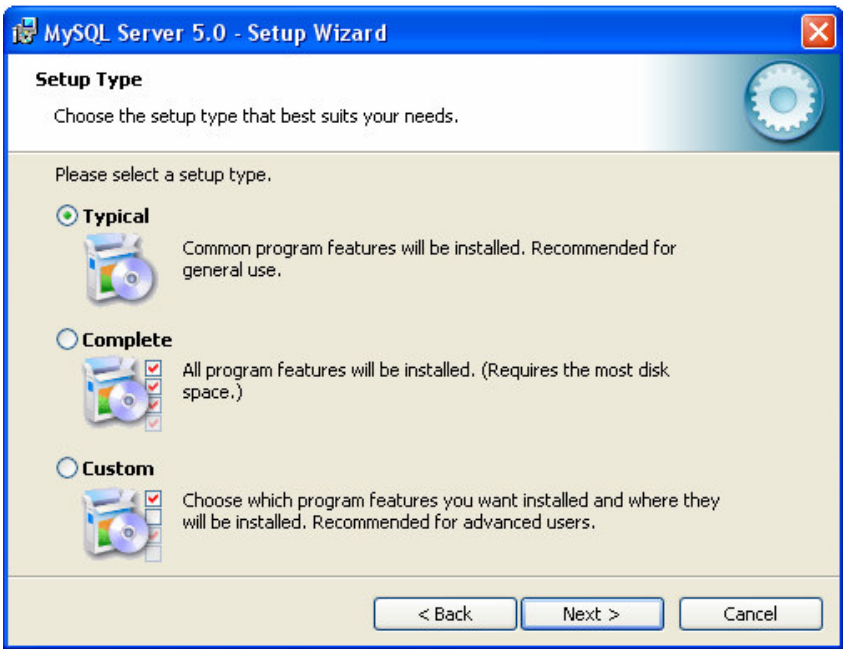
Paketat instaluese të MySQL mund të shkarkohen nga adresa :
http://dev.mysql.com/downloads/
Procesi i startimit të Asistentit të instalimit dallon nga përmbajtja e paketës së shkarkuar. Nëse është datoteka **Setup.exe** atëherë vetëm klikohet dy herë në të për të startuar instalimin me Asistentin e instalimit, e njëjta vlen nëse datoteka e shkarkuar është **.msi** .

Zgjedhja e tipit të instalimit
Janë tri lloje të instalimit në dispozicion: **Tipike (Typical)**, **E plotë (Complete)** dhe **Sipas dëshirës (Custom)**.

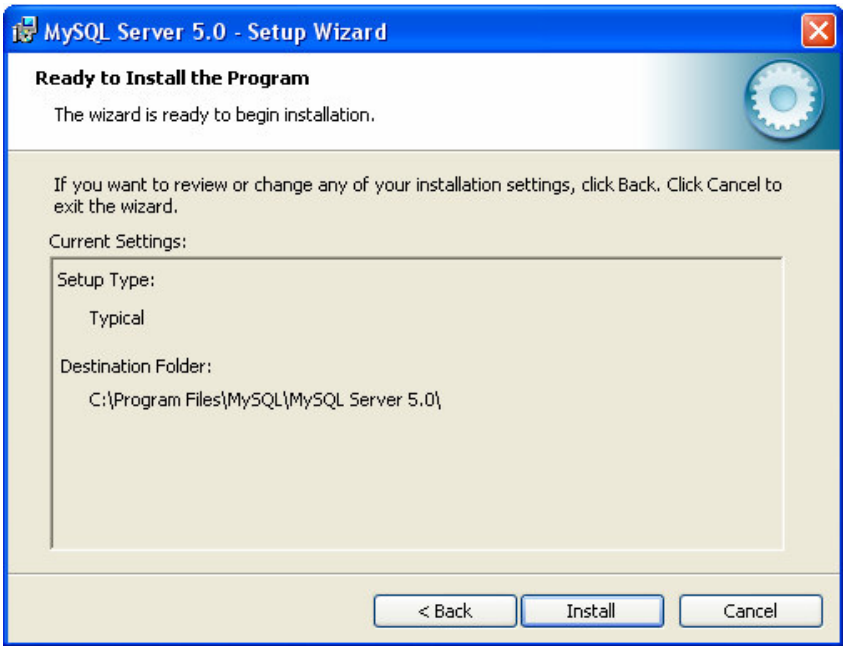
- Instalimi **Tipik** instalon MySQL serverin, komandë-rreshtin **mysql** për klientin dhe komandë-rreshtin me vegla. Komandë-rreshti për klientët dhe veglat përmbajnë **mysqldump**, **myisamchk** dhe vegla të tjera që ndihmojnë në menaxhimin e MySQL serverit.

- Instalimi **i plotë** i MySQL përmban të gjitha komponentat në paketën instaluese. Instalimi i plotë përmban komponente si librari serverin, përkrahja e skriptave dhe dokumentimit.

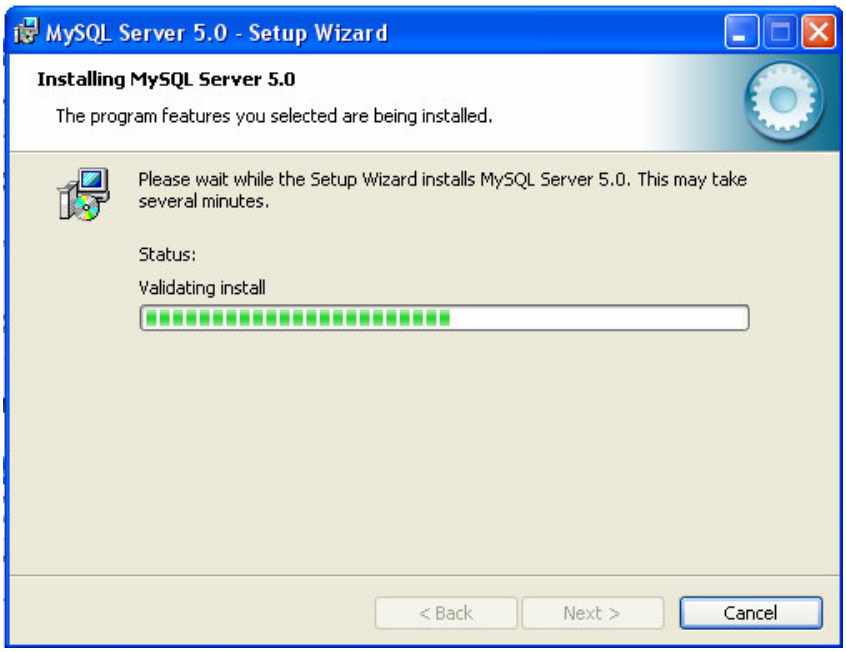
- Instalimi **sipas dëshirës** na ofron kontroll dhe më shumë mundësi për zgjedhjen e paketës instaluese.



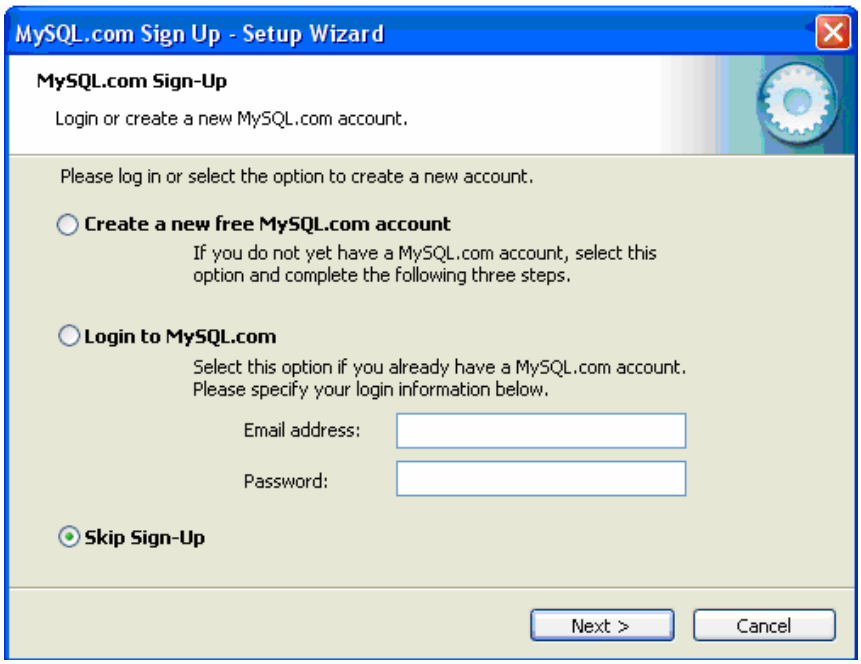
Nëse dëshirojmë të ndërrojmë shtegun e instalimit ose ndonjë komponenetë atëherë duhet të zgjedhim Custom (Sipas dëshirës). Përndryshe preferohet të zgjedhet Typical. Bëjmë zgjedhjen e duhur dhe shtypim Next. Hapet dritarja konfirrmuese me zgjedhjet e bëra më herët. Për të instaluar MySQL me këto zgjedhjet e bëra vetëm shtypim Install. Për të ndryshuar zgjedhjet e bëra vetëm shtypim pullën Back dhe bëjmë ndërrimet e nevojshme. Nëse dëshirojmë që të ndërpresim instalimin atëherë shtypim pullën Cancel.



Pas shtypjes së pullës Install paraqitet dritarja, ku do instalohen të gjitha komponentat e zgjedhura.

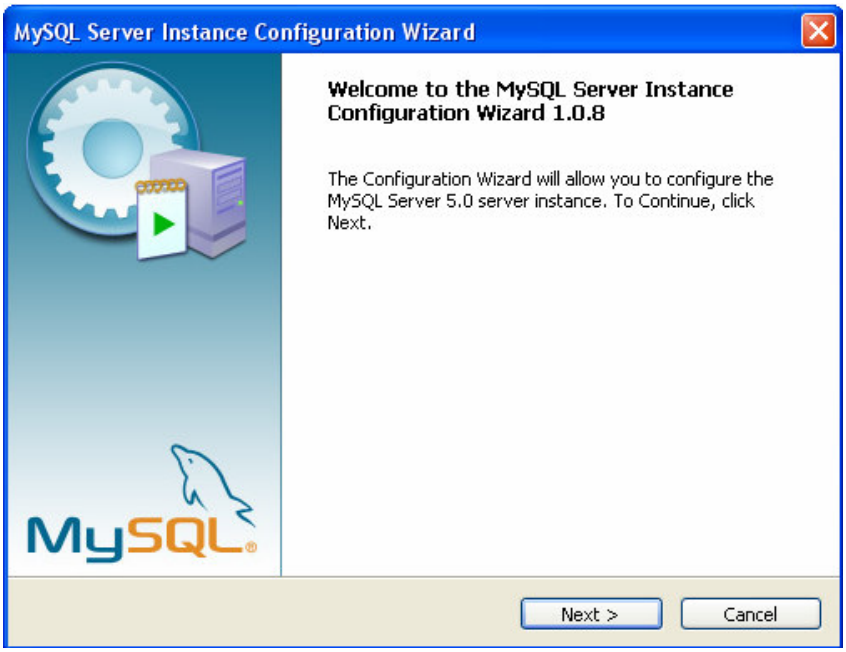


Pas instalimit na kërkohet të krijojmë një llogari në MySQL.com por këtu ne zgjedhim Skip-Sign Up dhe shtypim Next.



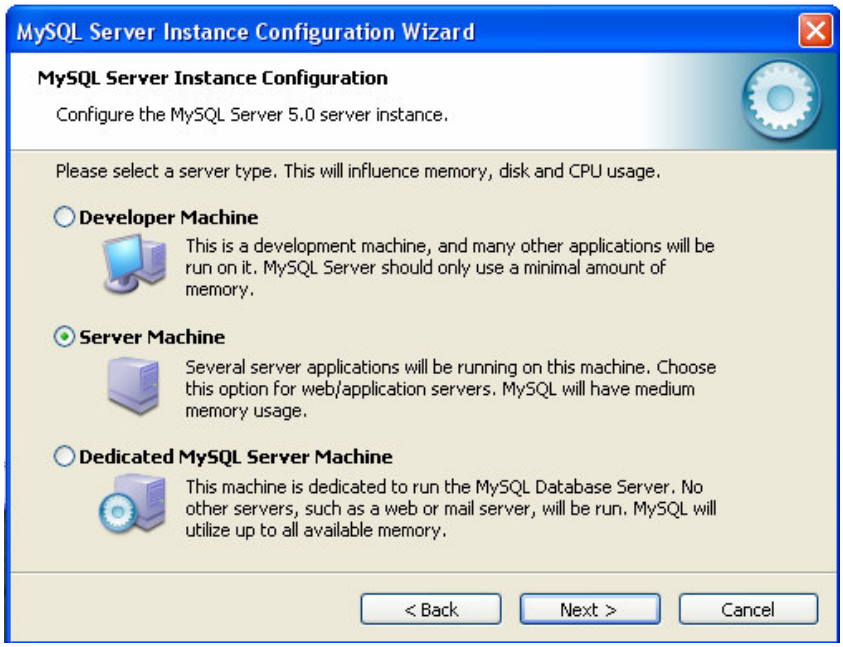
Dhe hapet dritarja që tregon se instalimi përfundoi. Në këtë dritare është mundësia e zgjedhjes për konfigurimin e MySQL serverit, e bëjmë zgjedhjen e saj dhe shtypim Finish.

Tani na paraqitet dritarja përshëndetëse për konfigurimin e MySQL serverit. Pra do të krijojmë një instancë të serverit. Shtypim pullën Next.

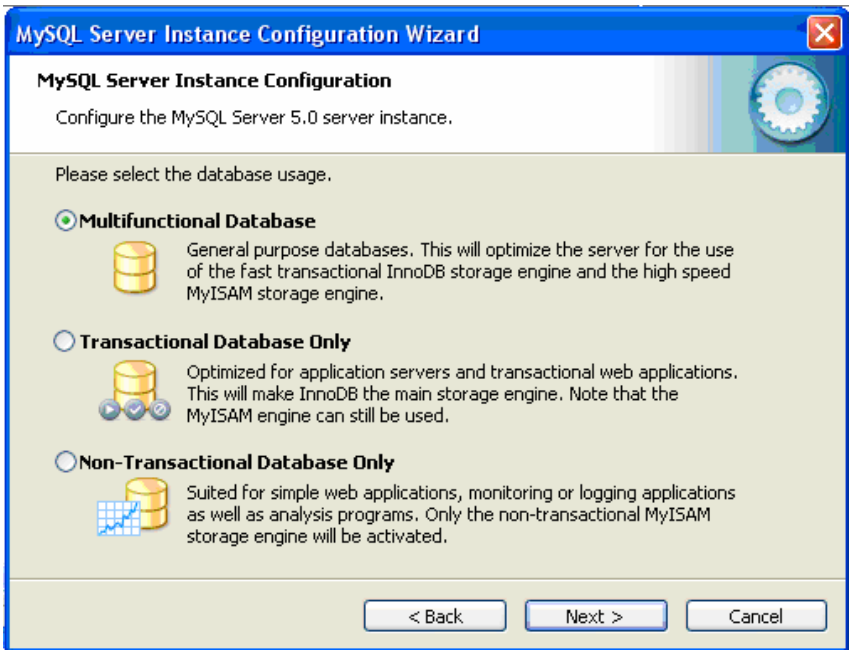


Do të hapet dritarja për zgjedhjen e tipit të serverit. Janë tri tipe nga të cilat mund të zgjedhim, dhe pasi të zgjedhim tipin e serverit kjo do të ndikojë në vendimet që i bën Asistenti i konfigurimit për sa i përket memories në disk dhe shfrytëzimin e procesorit.

- **Developer Machine** : Zgjedhni këtë opsion nëse keni ndërmend që MySQL ta shfrytëzoni për nevoja personale. Me këtë MySQL serveri do të shfrytëzojë minimumin e burimeve të sistemit.
- **Server Machine** : Zgjedhni këtë opsion për server aty ku MySQL serveri do të punojë së bashku me serverët tjerë siç janë FTP, E-mail dhe Web serverët. MySQL serveri është i konfiguruar ashtu që të shfrytëzojë një mesatare të burimeve të sistemit.
- **Dedicated MySQL Server Machine** : Zgjedhni këtë opsion nëse dëshironi që vetëm MySQL serveri të punojë në atë makinë . MySQL serveri është i konfiguruar ashtu që të shfrytëzojë të gjitha burimet që janë në dispozicion nga burimet e sistemit.

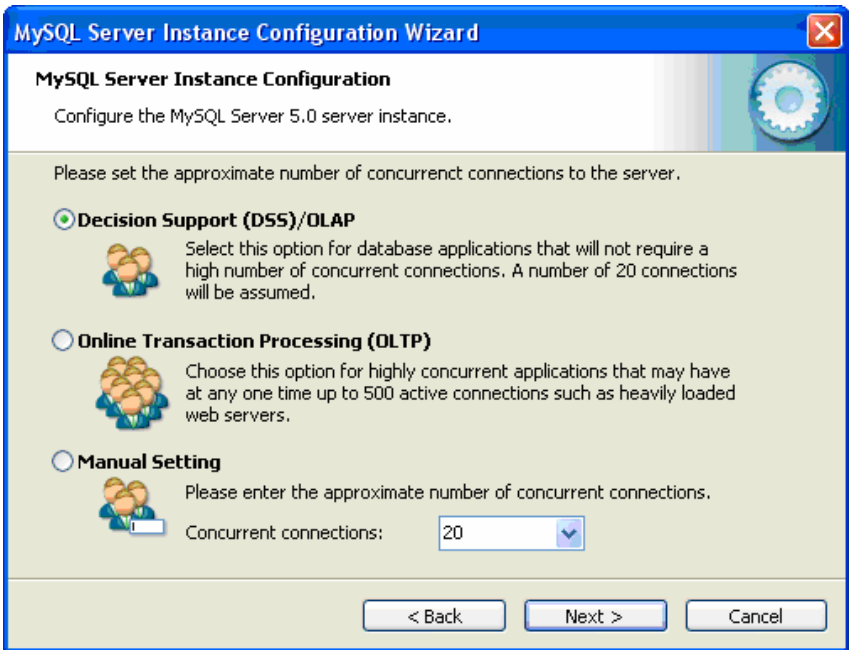


Shtypim Next dhe hapet dritarja për mënyrën e shfrytëzimit të bazës së të dhënave. Janë tri opsione ne do të zgjedhi opsionin e fundit **Non-Transactional Database Only**.

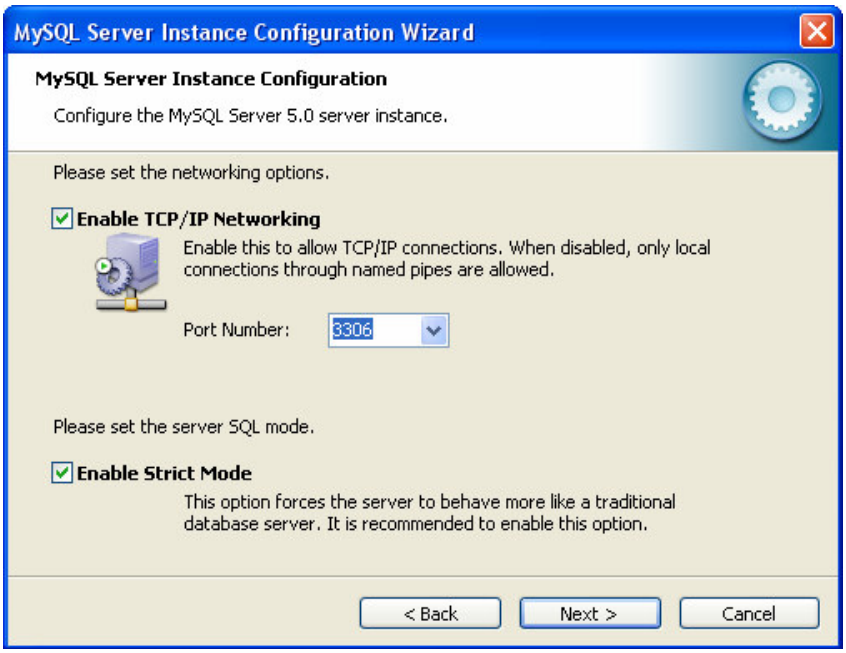


Në fund shtypim pullën Next. Do të paraqitet dritarja për kufizimin e kyçjeve në MySQL server. Jane tri opsione:

- Decision Support (DSS)/OLAP : Zgjedhni këtë opsion nëse serveri nuk do të kërkojë numër të madh të kyçjeve. Numri maksimal i kyçjeve është 100.
- Online Transaction Processing (OLTP) : Zgjedhni këtë opsion nëse serveri kërkon një numër të madh të kyçjeve. Numri maksimal i kyçjeve është 500.
- Manual Setting : Zgjedhni këtë opsion për të vënë numrin maksimal të kyçjeve.



Zgjedhim opsionin e parë dhe shtypim Next. Pastaj hapet dritarja për zgjedhjen e TCP/IP për komunikim në rrjetë ose jo dhe për konfigurimin e portit në të cilin do të kyçemi në MySQL serverin. TCP/IP është e parazgjedhur, për të mos zgjedhur TCP/IP vetëm e çaktivojmë kutizën. Porti 3306 është port i parazgjedhur nga Asistenti i konfigurimit, për të ndërruar numrin e portit vetëm zgjedhim një numër nga lista ose vetëm e shkruajmë atë. Nëse porti që keni zgjedhur është veç në shfrytëzim atëherë do të kërkohet nga ju për një numër tjetër.



Bëjmë zgjedhjet e duhura dhe shtypim pullën Next. Do të hapet dritarja për zgjedhjen e bashkësisë së karakterëve. Janë tri opsione zgjedhim opsionin e parë **Standard Character Set**.

Dhe shtypim pullën Next. Do të hapet dritarja për zgjedhjen e shërbimit (Service) të MySQL. Në platformën Windows MySQL mund të instalohet si shërbim (Service). Asistenti i konfigurimit e ka të parazgjedhur instalimin e MySQL serverit si shërbim (Service). Nëse dëshirojmë që të mos instalojmë MySQL si shërbim atëherë vetëm e çaktivojmë kutinë. Për instalimin e MySQL serverit si shërbim por që ai të mos startoj automatikisht duhet të çaktivojmë kutinë te **Launch the MySQL Server automatically.**

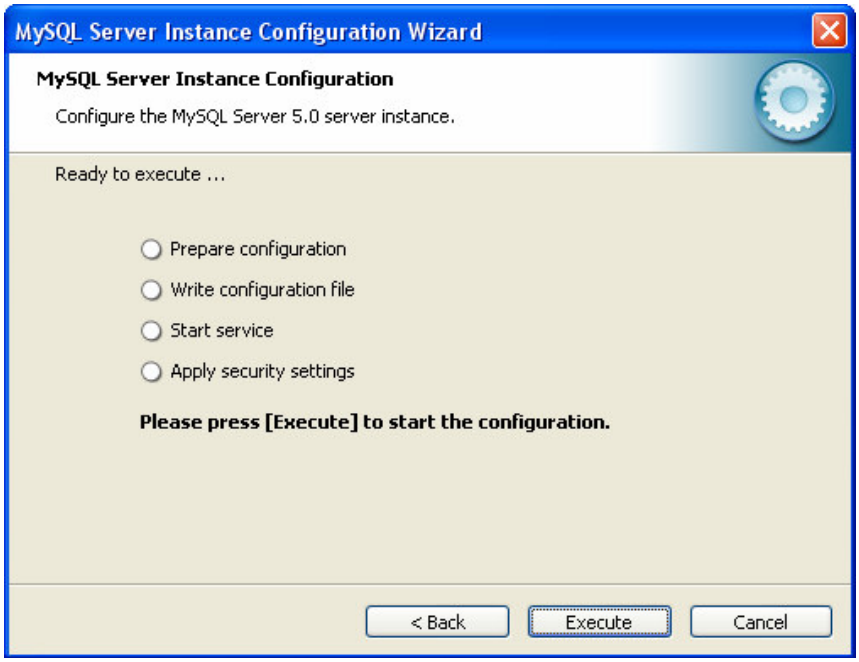
Shtypim pullën Next. Në dritaren për sigurinë është e preferuar të vëhet fjalëkalimi (root passëord) për MySQL serverin.Asistenti i konfigurimit kërkon vënien e një fjalëkalimi. Nëse nuk dëshirojmë që të vëjmë fjalëkalim atëherë e çaktivojmë kutinë te **Modify Security Settings.**

Për vënien e fjalëkalimit shkruajmë fjalëkalimin në **New root password** kutinë dhe në **Confirm** kutinë. Nëse jemi duke bërë rikonfigurimin e instancës së MySQL serverit atëherë do të na kërkohet që të shkruajmë edhe fjalëkalimin ekzistues në **Current root password** kutinë.

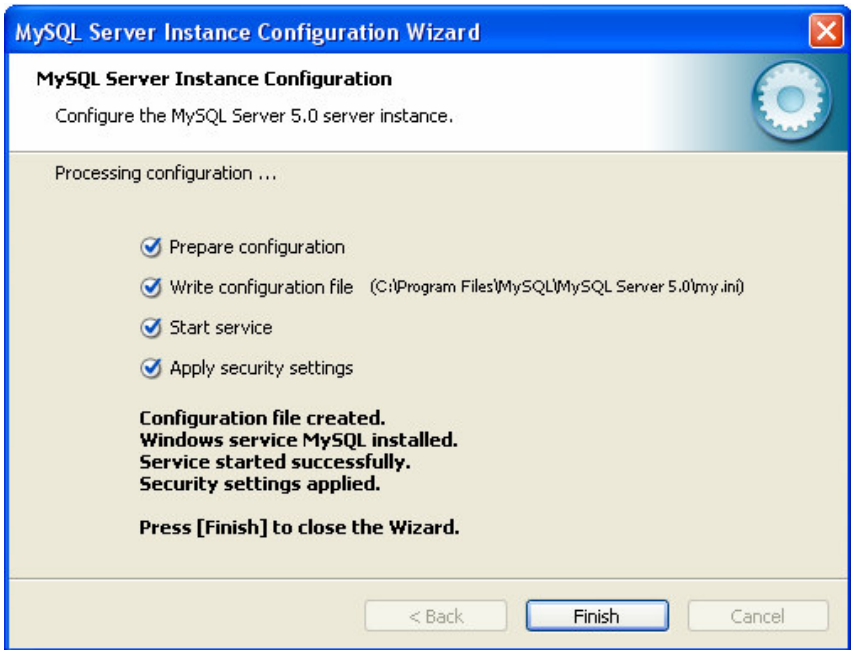
Për të ndaluar kyçjen në MySQL server përmes rrjetës vetëm e çaktivojmë kutinë te **Enable root access from remote machines.** Kjo rritë sigurinë e fjalëkalimit.

Për krijimin e një shfrytëzuesi anonym e çaktivojmë kutinë afër **Create An Anonymous Account**. Me krijimin e një shfrytëzuesi anonym do të zvogëlojmë sigurinë në server dhe do të shkaktojmë vështirësira në kyçje dhe leje për atë shfrytëzues. Për këtë arsye nuk preferohet krijimi i shfrytëzuesit anonim.

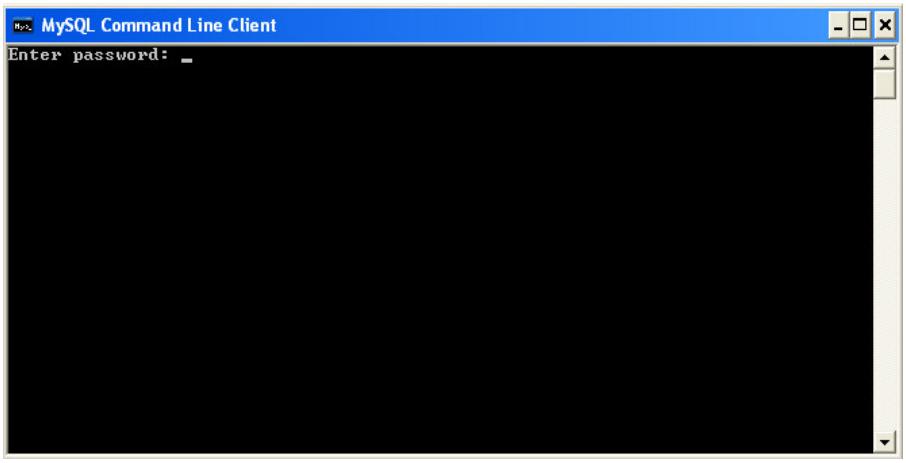
Shkruajmë fjalëkalimin : **fshmn** dhe e konfirmojmë atë. Shtypim pullën Next. Në fund paraqitet dritarja konfirmuese. Për startimin e konfigurimit të MySQL serverit shtypim **Execute**. Pasi të shtypet **Execute**, Asistenti për konfigurimin e MySQL serverit kryen detyrat e përcaktuara më parë duke treguar progresin në dritaren konfirmuese. Asistenti i konfigurimit të MySQL serverit i shkruan të gjitha konfigurimet e serverit në datotekën *my.ini* i cili gjendet në C:\Program Files\MySQL\MySQL 5.0\my.ini .



Dhe në fund shtypim Finish për përfundimin e konfigurimit të serverit.



Shkojmë pastaj te *Start → Programs → MySQL → MySQL Server 5.0 → MySQL* Command Line Client dhe hapet dritarja:



Shkruajmë fjalëkalimin *fshmn* dhe kyçemi në server.

Çka është MySQL

MySql është një system i menaxhimit të relacioneve të bazave të të dhënave (RDBMS) . Nuk përmban në vehte vetëm pjesën e serverit për menaxhim të bazës së të dhënave, por ai ka edhe vegla për qasjen e bazës së të dhënave dhe ndërtimin e programeve përkundrejt atyre bazave të të dhënave. Disa prej veglave janë:

- mysql – Ekzekuton SQL komandat përkundrejt MySQL. Poashtu mund të përdoret për ekzekutimin e SQL komandave të ruajtura në datoteka.
- mysqlaccess – menaxhimi i shfrytëzuesve
- mysqladmin – mundëson menaxhimin e serverit të bazës së të dhënave, duke përfshirë krijimin dhe fshirjen e bazës së të dhënave.
- mysqldump – hedh përmbajtjen e një baze të të dhënave MySQL ose tabele në një datotekë.
- mysqlhotcopy – bën ruajtjen e një kopje rezervë (BackUp) të bazës së të dhënave MySQL.
- mysqlimport – bën importimin e të dhënave në datoteka me formate të ndryshme në bazën e të dhënave.
- mysqlshow – shfaq informacionin për MySQL serverin dhe për çdo objekt si baza e të dhënave dhe tabelat në atë server.
- Safe_mysqld – starton mysqld në UNIX me siguri të madhe.

MySQL është një software që mundëson shfrytëzuesit të krijojë, mirëmbajë dhe të menaxhojë bazat e të dhënave. Kjo kategori e software-it është e njohur si sistemi i menaxhimit të bazave të të dhënave (DBMS). Sipas definicionit baza e të dhënave është një grumbull i organizuar i të dhënave. Siç u cek më herët MySql është një system i menaxhimit të relacioneve të bazave të të dhënave (RDBMS). Një bazë relacionale organizon të dhënat nëpër tabela dhe paraqet lidhjet në mes atyre tabelave.
Këto lidhje na mundësojnë bashkimin e të dhënave nga shumë tabela që na jep një paraqitje të ndryshme të të dhënave. Natyrisht se më lehtë është të bëhet ilustrimi me një shembull për të kuptuar më mirë konceptin e tabelave dhe relacioneve të tyre.

Shembull:
Një tabelë ka emrin, disa kolona dhe rreshta që përmbajnë të dhëna. Në bazën e të dhënave relacionale të dhënat paraqiten në formë të tabelave.
Në shembullin e mëposhtëm shohim se në të dy tabelat është paraqitur kolona *Numri i llogarisë*. Me anë të atij numri mund të gjejmë se cili klient ka marrë kredinë e caktuar. Thënë shkurt sistemi i menaxhimit të bazës së të dhënave (DBMS) për një sistem relacional quhet sistemi i menaxhimit të bazës relacionale (RDBMS). Ku MySQL është një shembull i një RDBMS.

Tabela : Klientët

Numri i llogarisë	*Emri*	*Mbiemri*	*Qyteti*	*Adresa*	*Tel*
140201	Bujar	Shulemaja	Prishtinë	Bregu i Diellit	044/123-456
140202	Amir	Simnica	Fushë Kosovë	rr. Agim Ramadani	044/321-654
140203	Valbona	Krasniqi	Dardanë	rr. Adem Jashari	044/213-546
140204	Gani	Thaçi	Besianë	rr. Zahir Pajaziti	044/312-645
140205	Lavdim	Kastrati	Prishtinë	Tophane	044/132-465

Tabela : Kreditë

ID	*Numri i llogarisë*	*Tipi i kredisë*	*Shuma*	*Përqindja*	*Kohëzgjatja e kredisë*	*Aprovimi*	*Fillimi*
1	140203	Veturë	4500	6	12	po	10/04/ 2006
2	140205	Mobile	1300	7.5	6	po	23/08/ 2006
3	140201	Veturë	7000	5.5	36	po	15/02/ 2005
4	140203	Shtëpi	45000	4.7	120	jo	20/05/ 2005
5	140202	Kompjuter	630	7	12	po	12/07/ 2006
6	140204	Kuzhinë	1000	4	12	po	04/09/ 2005
7	140205	Lavatriçe	420	3	6	po	05/06/ 2006
8	140201	Mobile	1730	5.8	12	po	02/10/ 2006
9	140205	Veturë	5250	7.3	18	po	12/04/ 2005
10	140202	Tavolinë	420	8.3	12	jo	12/08/ 2005

Por si bëhet krijimi i bazës së të dhënave, si bëhet definimi i tabelave, ndryshimi, shtimi dhe fshirja e të dhënave. **SQL** (**S**tructured **Q**uery **L**anguage) është një gjuhë që shërben për operacionet në një bazë të të dhënave. **SQL** është një gjuhë standarde që shumica e programerëve të bazave të të dhënave e njohin.

Mirëmbajtja e një baze të të dhënave në një tabelë nuk kërkon shumë njohuri në teorinë e bazës së të dhënave. Në jetën reale bazat e të dhënave kanë numër të madh të dhënash (qindra, mijëra madje edhe miliona). Këtu është edhe rëndësia e bazave të të dhënave relacionale. Konsiderojmë për shembull se libraria e Kongresit në SHBA ka afër 16 milion libra. Për arsye që do t'i sqarojmë më poshtë është e pamundur të mbahen të gjitha të dhënat në një tabelë.

Përsëritja e të dhënave (Data redundancy)

Problemi kryesor që lidhet me përdorimin e një tabele të vetme për bazë të të dhënave është përsëritja e panevojshme e të dhënave. Ndonjëherë përsëritja e të dhënave është e nevojshme, por ideja është që të largohen sa më shumë përsëritjet e të dhënave.

ID	Numri i llogarisë	Emri	Mbiemri	Qyteti	Tipi i kredisë	Shuma	ID Nënpunësi	Aprovimi
1	140203	Valbona	Krasniqi	Dardanë	Veturë	4500	Faton Gashi	po
2	140205	Lavdim	Kastrati	Pishtinë	Mobile	1300	Astrit Kabashi	Po
3	140201	Bujar	Shulemaja	Prishtinë	Veturë	7000	Astrit Kabashi	po
4	140203	Valbona	Krasniqi	Dardanë	Shtëpi	45000	Faton Gashi	jo
5	140202	Amir	Simnica	Fushë Kosovë	Kompjuter	630	Albana Gashi	po
6	140204	Gani	Thaqi	Besianë	Kuzhinë	1000	Valon Ramadani	po
7	140205	Lavdim	Kastrati	Prishtinë	Lavatriçe	420	Astrit Kabashi	po
8	140201	Bujar	Shulemaja	Prishtinë	Mobile	1730	Astrit Kabashi	po

Përsëritja e të dhënave në tabelën e mësipërme është e qartë. P.SH. *Numri i llogarisë, Emri, Qyteti, Tipi i kredisë, Nënpunësi* përsëriten disa herë. Në përpjekje për reduktimin e përsëritjes së të dhënave duhet që këtë tabelë ta ndajmë në disa tabela. Një zgjidhje e mundshme mund të jetë ndarja në pesë tabela të reja.

- Klientët ku secili klientë ka numrin e llogarisë,emrin, mbiemrin, adresën.
- Qyteti , secili qytet me shifër
- Kreditë ku secila kredi do të ketë shifrën e vet si dhe nënpunësi i bankës që e ka lëshuar atë kredi
- Shifra e kredisë , secila kredi do të ketë shifrën e vet
- Nënpunësit ku secili nënpunës do të ketë shifrën e vet.

Tabela : Klientët

Numri i llogarisë	*Emri*	*Mbiemri*	*ID Qyteti*	*Adresa*	*Tel*
140201	Bujar	Shulemaja	1	Bregu i Diellit	044/123-456
140202	Amir	Simnica	3	rr. Agim Ramadani	044/321-654
140203	Valbona	Krasniqi	2	rr. Adem Jashari	044/213-546
140204	Gani	Thaçi	4	rr. Zahir Pajaziti	044/312-645
140205	Lavdim	Kastrati	1	Tophane	044/132-465

Tabela : Qyteti

ID Qyteti	*Qyteti*
1	Prishtinë
2	Dardanë
3	Fushë Kosovë
4	Besianë

Tabela : Kreditë

ID	*Numri i llogarisë*	*Shifra e kredisë*	*Shuma*	*Përqindja*	*Aprovimi*	*ID Nën*
1	140203	K1	4500	6	po	3
2	140205	K2	1300	7.5	po	1
3	140201	K1	7000	5.5	po	1
4	140203	K4	45000	4.7	jo	3
5	140202	K3	630	7	po	4
6	140204	K5	1000	4	po	2
7	140205	K6	420	3	po	1
8	140201	K2	1730	5.8	po	1
9	140202	K7	420	8.3	jo	4

Tabela : Shifra e kredisë

Shifra e kredisë	*Tipi i kredisë*
K1	Veturë
K2	Mobile
K3	Kompjuter
K4	Shtëpi
K5	Kuzhinë
K6	Lavatriçe
K7	Tavolinë

Tabela : Nënpunësit

ID Nënpunësit	*Emri*	*Mbiemri*	*Filiala*
1	Astrit	Kabashi	Prishtinë
2	Valon	Ramadani	Besianë
3	Faton	Gashi	Dardanë
4	Albana	Gashi	Fushë Kosovë

Tani qyteti, tipi i kredisë, emri dhe mbiemri i klientit si dhe emri dhe mbiemri i nënpunësit paraqitet vetëm nga një herë. Natyrisht se ka ende të dhëna të dyfishta në bazë të të dhënave. P.SH. Shifra e kredisë, ID qyteti paraqiten më shumë se një herë në këto tabela. Si u tha më herët nuk mund të eliminojmë të gjitha duplikatet dhe të mbajmë lidhjen në mes tabelave, këto janë të dhëna që përsëriten por të nevojshme. Rëndësia e eliminimit të përsëritjes së të dhënave mund të shpjegohet edhe me një shembull të thjeshtë.
Shembull: Konsiderojmë bazën e të dhënave për një bibliotekë me një fond të librave 16 milionë. Supozojmë se baza e të dhënave përmban 10000 botues të ndryshëm. Adresa e botuesit në bazën e të dhënave në një tabelë do të paraqitej 16 milionë herë, ku 10 000 adresa janë të ndryshme mes veti. Tani nëse një adresë ka mesatarisht 50 karakterë atëherë:
(16 000 000 – 10 000)* 50 = 799,5 milionë karakterë
Nëse secili karakterë zë vend 2 byte atëherë në bazën e të dhënave me një tabelë është zënë vend në hapësirën e diskut përafërsisht 1.49 GB vetëm për adresën e botuesit.
Nga kjo që u tha për përsëritjen e panevojshme të të dhënave konkludohet se ndërtimi i një baze të të dhënave me anë të një tabele nuk është praktike.
Disa prej arsyeve tjera për mos përdorimin e vetëm një tabele për ndërtimin e bazës së të dhënave janë:

- Ndryshimi i të dhënave
- Shtimi i të dhënave
- Fshirja e të dhënave
- Mirëmbajtja e integritetit referencial etj

Tipet e të dhënave

Sikurse edhe të gjitha sistemet e menaxhimit të bazave relacionale edhe MySQL ka tipet e të dhënave specifike:

- Tipet numerike
- Tipet string ose tipet karakterë
- Tipet e renditjes (Enumerations) dhe bashkësitë (Set)
- Tipet për datë

MySQL përkrah tipe të ndryshme të të dhënave me funksion të ndyshëm. Tip i të dhënave është, tipi i të dhënës që do të ruhet në fushë (kolonë). Në një tabelë mund të jenë tipe të ndryshme të të dhënave, por secila fushë do të ketë tipin e të dhënave specifik. Një fushë e definuar si numerike do të pranojë vetëm të dhëna numerike, një fushë e definuar si tip i karakterëve CHAR(10) do të pranojë vetëm deri në 10 karakterë. Këto definime janë çelës për një bazë të të dhënave të shpejtë dhe efikase.
Në parim janë tri grupe të të dhënave. Grupi i parë është numerik. Të dhënat numerike janë ato të dhëna që janë numra pozitiv ose negative si 4 ose -32. Të dhënat numerike poashtu mund të jenë në formën : heksadecimale si 2EE250CC, shkencore si 2 x 10^23, decimale. Grupi i dytë i tipit të të dhënave është tipi string ose tipi karakter. Në këtë grup hyjnë të gjitha të dhënat që janë kombinim i shkronjave dhe numrave si : shkronjat, numrat, fjalitë, adresa, numrat e telefonit etj. Në fund grupi i ndryshëm, ai përmban të dhënat që nuk mund të përfshihen në dy grupet e para si: data, koha. MySQL përkrah tipeve të të të dhënave ofron edhe modifikuesit e kolonave. Ata janë AUTO_INCREMENT, UNSINGNED, PRIMARY KEY, NULL, NOT NULL dhe BINARY.

Tipet numerike

Tipet numerike janë të paracaktuara që të përfshijnë vetëm numra. Nuk mund të ruajmë shkronjë ose një varg të karakterëve në një fushë të definuar mëparë si numerike.Tipe të ndryshme numerike zënë hapësirë të ndryshme në memorie. Kjo është arësye pse secili tip numerik ka intervalin përkufizues të ndryshëm.

Tipi i të dhënave	Hapësira në memorie (byte)	Intervali përkufizues	Pa shenjë
TINYINT	1	-128 deri 127	0-255
SMALLINT	2	-32768 deri 32767	0-65535
MEDIUMINT	3	-8388608 deri 8388607	0-16777215
INT	4	-2147483648 deri 2147483647	0-4294967295
BIGINT	8	-9223372036854775808	0-

		deri 9223372036854775807	18446744073709550615
FLOAT(M,D)	4	Varësisht nga vlera	
DOUBLE(M,D)	8	Varësisht nga vlera	
DECIMAL(M,D)	M+2	Varësisht nga vlera	

Nëse një fushë është e definuar si numerike dhe UNSIGNED atëherë intervali përkufizues dyfishohet për atë tip të dhënash. P.SH. nëse deklarojmë fushën si UNSIGNED TINYINT intervali përkufizues do të jetë prej 0 deri në 255. Me deklarimin e një fushe si UNSIGNED bëjmë që ajo fushë të ketë vetëm numra pozitiv. Madhësia e tipit të të dhënave nuk ndryshon por vetëm intervali përkufizues i tij. Tipet FLOAT, DOUBLE dhe DECIMAL janë numerike dhe mund të përfshijnë numra pas presjes dhjetore. Tipet e tjera nuk e kanë këtë veti. MySQL na mundëson kufizimin e shifrave pas presjes dhjetore. P.SH. numri 5.6876 me anë të FLOAT(4,2) do të ruhet si 5.69 .
Duhet të cekim se MySQL proceson të dhënat numerike më shpejtë se çdo tip të të dhënash. Për atë nëse dëshirojmë pyetësorë të shpejtë përdorim si kriter të dhëna numerike. Poashtu edhe indeksat numerik në përgjithësi janë më të shpejtë se indeksat të bazuar në karakterë. Kur të bëjmë definimin e tipit të të dhënave për një fushë duhet të kemi kujdes dhe të zgjedhim vlerën më të madhe që do të na duhet. Nëse nuk e bëjmë këtë atëherë do të kemi probleme më vonë. MySQL për vlerat që janë jashtë intervalit përkufizues të tipit të të dhënave e zëvendëson me vlerën më të madhe (skajin e djathtë të intervalit). P.SH. kemi tipin e të dhënave UNSIGNED TINYINT dhe dëshirojmë të fusim vlerën 1000 në atë fushë. MySQL do të ruaj atë vlerë si 255, pra si vlerën më të madhe të intervalit përkufizues.
Modifikuesit e kolonave AUTO_INCREMENT,UNSINGNED dhe ZEROFILL mund të përdoren vetëm për të dhënat numerike. Këta modifikues kryejnë veprime që mund të bëhen vetëm me numra. Modifikuesi UNSIGNED bën numrat në fushë të jenë pozitiv.

AUTO_INCREMENT
Modifikuesi i fushës AUTO_INCREMENT rrit vlerën e fushës duke shtuar 1 në vlerën maksimale. Është i dobishëm për krijimin e vlerave unike. Vlera e fushës me modifikuesin AUTO_INCREMENT fillon prej numrit 1 dhe shton nga një për çdo të dhënë që shtohet në tabelë. P.SH. krijojmë një tabelë me një fushë AUTO_INCREMENT. Shtojmë një të dhënë dhe vlera në fushën AUTO_INCREMENT është 1, shtojmë një të dhënë tjetër AUTO_INCREMENT bëhet 2. Tani fshijmë të dhënën e parë, dhe shtojmë një të dhënë tjetër.Çfarë vlere do të merr tani fusha AUTO_INCREMENT? Nëse jeni përgjigjur 3, është e saktë. Pra modifikuesi AUTO_INCREMENT nuk e ripërdorë vlerën që kemi fshirë më herët.

```
CREATE TABLE Test (Auto_Test int NOT NULL AUTO_INCREMENT);
INSERT INTO Test (Auto_Test) values(NULL);
INSERT INTO TEST (Auto_Test) values(0);
INSERT INTO Test (Auto_Test) values();
```

Poashtu mund të shtojmë edhe ndonjë numër, nëse ai numër ekziston në tabelë atëherë do të paraqitet gabim. Nëse jo atëherë ai numër do të futet në tabelë.
Mundësi tjetër është caktimi i numrit startues të fushës AUTO_INCREMENT. Nëse dëshirojmë që modifikuesi i fushës të fillojë nga numri 50 atëherë shkruajmë:

```
CREATE TABLE Test
(Test_ID INT NOT NULL AUTO_INCREMENT
AUTO_INCREMENT = 50,
Një_fushë  INT)
```

Nuk ka rëndësi ku vendoset AUTO_INCREMENT=nnn në sintaksën e krijimit të tabelës. Në një tabelë mund të jetë vetëm një fushë AUTO_INCREMENT.
Modifikuesi i fushës kur të arrin vlerën maksimale të tipit të të dhënave na lajmërohet gabim. P.SH. nëse kemi zgjedhur TINYINT për AUTO_INCREMENT vlera maksimale është 255. Pasi te kemi arritur vlerën 255 dhe nëse dëshirojmë të shtojmë edhe një të dhënë atëherë MySQL do të lajmëroj për gabim. Për të evituar këtë preferohet të përdoret INT për modifikuesin AUTO_INCREMENT.
AUTO_INCREMENT fusha punon vetëm me numra të plotë, ndërsa për FLOAT, DOUBLE dhe DECIMAL nuk mund të shfrytëzohet.

ZEROFILL
Nëse deklarohet një fushë INT(8) ZEROFILL dhe fusim vlerën 23 atëherë ai paraqitet në formën 00000023.

Tipet e të dhënave string ose karakterë

Grup tjetër i tipit të të dhënave janë tipet string ose karakterë. String është një bashkësi e karakterëve (varg i karakterëve). Tipi i të dhënave string mund të ruaj të dhënat si: Bregu i diellit, rruga 2 , 992 Prishtinë. Edhe këtu si te tipet numerike varet nga madhësia se çfarë tipi të të dhënash do të përdorim. Madhësia maksimale është e dhënë në tabelën e mëposhtme.

Tipi i të dhënave	Max (byte)	Madhësia (byte)
CHAR(X)	255	X
VARCHAR(X)	255	X+1
TINYTEXT	255	X+1
TINYBLOB	255	X+2
TEXT	65535	X+2
BLOB	65535	X+2
MEDIUMTEXT	1.6 MB	X+3
MEDIUMBLOB	1.6 MB	X+3
LONGTEXT	4.2 GB	X+4
LONGBLOB	4.2 GB	X+4

CHAR dhe VARCHAR
Prej të gjithë tipeve string, tipet VARCHAR dhe CHAR përdoren më së shumti. Dallimi në mes tyre është se nëse deklarojmë një fushë CHAR(10) atëherë të gjitha të dhënat të

ruajtura në këtë fushë do të zënë madhësi prej 10 byte edhe pse kanë gjatësi 3 karakterë. Ndërsa nëse deklarojmë një fushë VARCHAR(10) dhe ruajmë të dhënë në atë fushë që kanë gjatësi 3 karakterë, madhësia do të jetë 4 byte (gjatësia + 1). Përparësitë e CHAR ndaj VARCHAR qëndrojnë se procesimi i të dhënave bëhet më shpejtë me tipin CHAR. Përdorimi i tipit CHAR është vetëm madhësia e hapësirës që zë në memorie. Si rregull CHAR dhe VARCHAR nuk përdoren në një tabelë. MySQL automatikisht konverton tipet në VARCHAR.

Konvertimi i CHAR në VARCHAR
Kur një tabelë ka së paku një fushë të tipit VARCHAR dhe të gjitha fushat tjera të tipit CHAR me më shumë se 3 karakterë, atëherë të gjitha fushat e tipi CHAR do të konvertohen në fusha të tipit VARCHAR.
CREATE TABLE Test (Fusha_Char CHAR(5), Fusha_Var VARCHAR(15))
MySQL automatikisht do të konvertojë fushën Fusha_Char në VARCHAR.

Konvertimi i VARCHAR në CHAR
Kur një fushë e tipit VARCHAR ka gjatësi më të vogël se 4 atëherë ajo konvertohet në fushë të tipit CHAR.
CREATE TABLE Test (Fusha_Var VARCHAR(3))
MySQL do të konvertojë fushën Fusha_Var në fushë të tipit CHAR.

TEXT dhe BLOB
Me anë të TEXT dhe BLOB mund të ruajnë një numër të madh të të dhënave. Këto tipe të të dhënave përdoren për ruajtjen e fotografive, audiove si dhe shumë tekst, si Web faqet ose dokumente. Këto tipe janë të përshtatshme edhe për të dhënat që zënë madhësi të ndryshme varësisht nga rreshti. Përparësitë e TEXT dhe BLOB në krahasim me tipet tjera është mundësia e ruajtjes të dhënave të shumta. Mund të ruajmë një datotekë të tërë. Mangësi e këtyre dy tipeve është procesimi i ngadaltë i të dhënave dhe madhësia e madhe që zënë në memorie. Në versionet e reja të MySQL mund të vejmë indeks në këto dy tipe, por nuk rekomandohet sepse bën degradimin e performancës së bazës së të dhënave. Indekset në këto dy tipe vetëm se i ngadalsojnë gjërat.

Tipet e renditjes (Enumerations) dhe bashkësitë (Set)

Janë dy tipe: ENUM dhe SET.

ENUM
Tipi i të dhënave ENUM është një listë e renditur. Duke pasë parasysh se ky tip mund të ruajë vetëm vlera që janë të deklaruara në listën e renditur. Fusha ENUM mund të ketë vetëm njërën nga vlerat e listës. Sintaksa e deklarimit të tipit të të dhënave ENUM është:

```
CREATE TABLE Test
(
    Return ENUM('P','J') DEFAULT 'J',
    Madhësia ENUM('S','M','L','XL','XXL'),
    Ngjyra ENUM('E zezë','e kuqe','e bardhë')
)
```

Mund të kemi deri 65535 vlera në listën e renditur. Tipi i të dhënave ENUM është një zgjedhje e mirë për fushë kombinimesh (Combo box) në Web faqe ose kudo që shfrytëzuesi duhet të zgjedh vlerat nga lista. ENUM mund të përmbajë ose ndonjë vlerë nga lista ose vlerën NULL. Nëse mundohemi të fusim ndonjë të dhënë që nuk është në listën e vlerave atëherë do të insertohet vlerë e zbrazët.

SET
Tipi it ë dhënave SET është i ngjashëm me ENUM. Njësoj si ENUM tipi SET përmban listë të vlerave. Ndryshimi mes tyre është se te SET mund të zgjedhim më shumë se një vlerë. Tipi SET mund të përmbajë deri 64 vlera të ndryshme. Sintaksa për krijimin e tipit SET është:

```
CREATE TABLE Test
 (
    Reklama SET('Web faqe','Televizion','Gazetë')
 )
```

Kolona e krijuar Reklama mund të përmbajë këto vlera:
"Web faqe"
"Televizion,Gazetë"
""
Kur të bëjmë shtimin e të dhenave në kolonën me tipin SET duhet që vlerat të jenë të futura brenda kuotave të ndara me presje. P.SH. për të shtuar një të dhënë me dy vlera nga shembulli i mëparshëm veprojmë:
INSERT INTO Test(Reklama) VALUES ('Web faqe,Televizion')
Për këtë arsye nuk duhet që brenda në listën e vlerave të ketë ndonjë vlerë që përmban presje. Arsyeja që këto dy tipe të të dhënave janë të ndara nga tipet tjera është se këto janë si bashkësi e karakterëve por MySQL i ruan në memorie si numra. Për këtë arsye edhe të dhënat procesohen më shpejtë. Poashtu mund të bëhen manipulime duke përdorë operacione numerike. P.SH.
SELECT * FROM Test WHERE Reklama=1;
Ky urdhër do të kthejë si rezultat të gjitha vlerat nga tabela Test ku Reklama është Web faqe.

Modifikuesit e kolonave (fushave)
MySQL ka disa fjalë të rezervuara që modifikojnë kolonat. P.SH. më herët i pamë modifikuesit AUTO_INCREMENT, UNSIGNED dhe ZEROFILL. Disa nga modifikuesit mund të zbatohen vetëm në disa tipe të të dhënave, kjo mund të shihet në tabelën e mëposhtme:

Emri i modifikuesit	Tipi mbi të cilin zbatohet
AUTO_INCREMENT	Në të gjithë tipet INT
BINARY	CHAR, VARCHAR
DEFAULT	Të gjithë tipet përveç BLOB dhe TEXT
NOT NULL	Të gjitha tipet
NULL	Të gjitha tipet

PRIMARY KEY	Të gjitha tipet
UNIQUE	Të gjitha tipet
UNSIGNED	Tipet numerike
ZEROFILL	Tipet numerike

Tipet për datë dhe kohë

MySQL ka disa tipe të të dhënave për datë dhe kohë që na lehtësojnë punën për ruajtjen e informative. Ato janë: DATETIME, DATE, TIME, YEAR dhe TIMESTAMP.
Tabela e mëposhtme i përshkruan këto tipe të të dhënave.

Tipi	Përshkrimi
DATETIME	YYYY-MM-DD HH:MM:SS prej 1000-01-01 00:00:00 deri 9999-12-31 23:59:59
DATE	YYYY-MM-DD prej 1000-01-01 deri 9999-12-31
TIME	HH:MM:SS
YEAR	YYYY
TIMESTAMP	YYYYMMDDHHMMSS

Tipi TIMESTAMP mund të paraqitet edhe në mënyra të ndryshme, si mëposhtë:

Tipi	Përshkrimi
TIMESTAMP (14)	YYYYMMDDHHMMSS
TIMESTAMP (12)	YYMMDDHHMMSS
TIMESTAMP (10)	YYMMDDHHMM
TIMESTAMP (8)	YYYYMMDD
TIMESTAMP (6)	YYMMDD
TIMESTAMP (4)	YYMM
TIMESTAMP (2)	YY

Data në MySQL paraqitet së pari viti pastaj muaji dhe në fund dita.

SQL (Structured Query Language)

SQL është një gjuhë që përdoret për leximin dhe shtimin e të dhënave në pothuaj se në të gjitha bazat e të dhënave. Me SQL mund të bëjmë kërkimin e të dhënave, shtimin e të dhënave të reja, ndryshimin (modifikimin) e të dhënave ose fshirjen e tyre. SQL është thjeshtë një vegël thelbësore për bashkëveprim me MySQL. Edhe nëse përdorim ndonjë program ose interfejs për baza të të dhënave, diku në prapavijë të atij programi është i gjeneruar kodi SQL. Urdhërat në SQL lexohen si një fjali në gjuhën angleze. Kjo qasje ka disa përparësi edhe mangësi sepse në fund SQL dallon nga gjuhët programuese tradicionale si C, Java ose Perl.

Gjuhën e strukturuar SQL e paraqiti së pari kompania IBM në vitin 1970 menjëherë pas zbulimit të teorisë të bazave relacionale. SQL ishte i popullarizuar sa që në vitin 1980 korporata Oracle lansoi sistemin e parë publik SQL. Në atë kohë SQL nuk ishte i standardizuar, dhe kjo nuk u bë deri në vitin 1989 ku ai standard u quajt SQL89. Mirëpo në këtë standard nuk është definuar sa duhet struktura e gjuhës SQL. Edhe pse në treg u paraqiten disa versione komerciale të gjuhës SQL, diferenca në sintaksë e bëri të pamundshme kyqjen e tyre në ndonjë implementim. Kjo zgjati deri në vitin 1992 kur ANSI vuri standardin SQL92 ose SQL2. Shumica e sistemeve të menaxhimit të bazave të të dhënave kanë pranuar SQL92 si standard, megjithatë bazat relacionale që implementojnë standardin SQL92 janë komplekse. Standardi SQL92 nuk është fjala e fundit e SQL standardeve. Me rritjen e popullaritetit të sistemit të menaxhimit të bazave të orientuara kah objektet (Object Oriented Database Managament System OODBMS) dhe sistemit të menaxhimit të bazave të lidhura me objekte (Object Relational Database Managament System ORDBMS) është vënë re një tendencë për mbështetjen e SQL standardit për këto baza të të dhënave. Pra SQL3 do të ishte një përgjigje për këtë.

Dizajni i SQL

Siç u cek më herët SQL më shumë i ngjan një gjuhe që flasim ne se sa një gjuhe të kompjuterit. Të shohim shembujt e mëposhtëm:

CREATE	TABLE	Klientët (emri varchar(10))	
folje	objekt	mbiemër	
INSERT	INTO Klientët	VALUES (‘Arben’)	
folje	një objekt	një objekt	
SELECT	emri	FROM Klientët	ËHERE emri LIKE ‘%n’
folje	objekt	objekt	mbiemër

Shumica e implementimeve të SQL duke përfshirë edhe MySQL janë jo të ndieshme në shkronja (case-insensitive). Pra nuk ka rëndësi se si i shkruani fjalët e rezervuara me shkronja të mëdha apo të vogla, e rëndësishme është që të shkruhen shkronjat e duhura. P.SH. nga shembulli më lartë, rreshti i parë mund të shkruhet edhe kësisoj:

```
cREatE TabLe Klientët (emri varchar(10))
```

Jo ndieshmëria e shkronjave vlen vetëm për fjalët e rezervuara të SQL. Në MySQL emri i bazës së të dhënave, tabelës dhe fushave është i ndieshëm në shkronja. Për shkak të leximit më të mirë të komandave të SQL fjalët e rezervuara do ti shkruajmë me shkronja të mëdha.
Element i parë në një SQL pyetësor është gjithnjë një folje. Folja shpreh aksionin që dëshirojmë që baza e të dhënave t'a kryejë. Pastaj pjesa tjetër e komandës vepron varësisht nga folja që kemi përdorë. P.SH. nga shembulli më lartë, folja CREATE (krijo) shoqërohet me objektin TABLE dhe pjesa tjetër është përshkrimi i tabelës që do të krijojmë.

Kyçja në server
Ka disa mënyra për kyçjen në server, më e shpeshta është ajo përmes ndonjë gjuhe programuese. Por këtu ne do të përdorim veglën *mysql*. Pra
Start → Programs → MySQL → MySQL Server 5.1 → MySQL Command Line Client
Shkruajmë fjalëkalimin dhe kyqemi në server.

*Enter password: **
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 5
Server version: 5.1.12-beta-community-nt MySQL Community Server (GPL)
Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql>

Komanda më lartë tregon lidhjen në server në makinë lokale si shfrytëzues. Një opcion tjetër është opcioni –h i cili na mundëson lidhjen në MySQL server nga distanca (*mysql –u root –h db.Banka –p*).
Nuk ka ndonjë lidhje në mes të shfrytëzuesve të Windows-it dhe shfrytëzuesve të MySQL. Shfrytëzuesit për MySQL duhet krijuar në mënyrë të pavarur nga shfrytëzuesit e Windows –it. Të gjitha komandat në MySQL duhet të përfundojnë me shenjën ; ose \g.

Sintaksa e komandave
Shprehja në kllapat e mesme është opsionale,shprehja brenda < > është emri i objektit.

Krijimi i bazës së të dhënave

Hapi i parë në krijimin e një baze MySQL është krijimi i objektit të bazës së të dhënave, i cili shërben si vend ku do të vendosen tabelat e asaj baze. Krijimi i një baze të të dhënave në MySQL është një ndër punët më të lehta, kërkohet fjala e rezervuar CREATE DATABASE duke shkruar pastaj emrin e bazës së të dhënave, ja edhe sintaksa:

CREATE DATABASE [IF NOT EXISTS] <emri i bazës së të dhënave>
[[DEFAULT] CHARACTER SET <emri i bashkësisë së karakterëve>]
[[DEFAULT] COLLATE <mënyra e krahasimit apo rendtjes>]

Në komandën për krijimin e bazës së të dhënave përfshihen edhe disa opsione. I pari është opsioni IF NOT EXISTS, i cili nëse përdoret dhe ekziston emri i bazës së të

dhënave na shfaqet një paralajmërim, por nëse nuk përdoret atëherë do të lajmëroj një gabim. Nëse ekziston një bazë e të dhënave me emër të njëjtë atëherë ajo nuk do të krijohet. Opsioni tjetër është CHARACTER SET, ku kjo është bashkësi e shkronjave, numrave dhe simboleve në bazë të të dhënave. P.SH. A,B,C,a,b,c,1,2,3,>,+ etj. Opsioni tjetër COLLATE përcakton renditjen, krahasimin, grupimin e bashkësisë së karakterëve (CHARACTER SET). Ndërsa opsioni DEFAULT vlerë e parazgjedhur gjatë instalimit të MySQL serverit.
Marrim disa shembuj:
Krijojmë bazën e të dhënave me emrin BANKA:

CREATE DATABASE BANKA;

Ose

CREATE DATABASE IF NOT EXISTS BANKA;

Ose

CREATE DATABASE IF NOT EXISTS BANKA;
DEFAULT CHARACTER SET latin1
DEFAULT COLLATE latin1_bin;

Krijimi i tabelës

Hap tjetër i rëndësishëm për krijimin e një baze të të dhënave është krijimi i tabelave. Tabelat janë objekte të bazës për ruajtjen e të dhënave dhe sigurinë e tyre. Për krijimin e tabelës në MySQL përdoret komanda CREATE TABLE. Kjo komandë është njëra ndër SQL komandat më komplekse në MySQL.
Sintaksa:

<definicioni>::=
CREATE [TEMPORARY] TABLE [IF NOT EXISTS] < emri i tabelës >
(< element i tabelës > [{, < element i tabelës > } …])
[< tipi i tabelës > [< tipi i tabelës > …]]

< element i tabelës >::=
< definimi i fushës >
| { [CONSTRAINT < emri i rregullës (constraint) > PRIMARY KEY
(< emri i fushës > [{, < emri i fushës > } …]) }
| { [CONSTRAINT < emri i rregullës >] FOREIGN KEY [< emri i indeksit>]
(< emri i fushës > [{ , < emri i fushës > } …]) < definimi i referencës > }
| { [CONSTRAINT < emri i rregullës >] UNIQUE [INDEX] [< emri i indeksit>]
(< emri i fushës > [{, < emri i fushës > } …]) }
| { { INDEX | KEY } [<emri i indeksit> (<emri i fushës> [{ ,<emri i fushës> } …]) }
|{FULLTEXT[INDEX] [<emri i indeksit>] (<emri i fushës> [{,<emri i fushës>}…]) }

< definimi i fushës>::=
<emri i fushës> <tipi i të dhënave>[NOT NULL|NULL] [DEFAULT <ndonjë vlerë>]
[PRIMARY KEY] [COMMENT '<ndonjë koment>'] [<definimi i referencës>]

< tipi i të dhënave>::=
< tip i të dhënave numerike>
| < tip i të dhënave sting ose karakter>
| <tip i të dhënave për datë>

< definimi i referencës >::=
REFERENCES < emri i tabelës > [(<emri i fushës> [{,<emri i fushës>} …])]
[ON DELETE {RESTRICT |CASCADE |SET NULL |NO ACTION |SET DEFAULT }]
[ON UPDATE {RESTRICT |CASCADE |SET NULL |NO ACTION |SET DEFAULT }]
[MATCH FULL | MATCH PARTIAL]

< tipi i tabelës >::=
{ ENGINE = {BDB | MEMORY | ISAM | INNODB | MERGE | MYISAM }}
| < tipet shtesë të tabelave>

Tash le të komentojmë sintaksën për krijimin e tabelës. Në rreshtin e parë kërkohet fjala e rezervuar CREATE TABLE e shoqëruar me emrin e tabelës që dëshirojmë të krijojmë. Ky rresht përmban dy opsione, opsioni i parë TEMPORARY tregon se kjo është një tabelë e përkohshme që është aktive vetëm gjatë kyqjes së shfrytëzuesit. Tabela e përkohshme ekziston deri sa shfrytëzuesi nuk ç'kyqet nga serveri. Opsioni i dytë IF NOT EXISTS është njësoj sikur te krijimi i bazës së të dhënave. Rreshti i dytë na lejon përcaktimin e elementeve të tabelës me anë të < element i tabelës >. Element i tabelës është çdo objekt që është i definuar në tabelë si: fusha ose çelësi primar. Në secilën tabelë mund të përfshihen më shumë se një element të ndarë mes veti me presje. Ndërsa rreshti i fundit na ofron përcaktimin e tipit të tabelës. Të gjithë tipet e tabelës janë opsionale. Siç mund të shihet sintaksa e krijimit të tabelës mund të thjeshtësohet ose të komplikohet. Elementet e domosdoshme në sintaksë për krijimin e tabelës janë:

CREATE TABLE < emri i tabelës > (<element i tabelës >)

Pasi elementi i tabelës është komponentë e domosdoshme shikojmë sintaksën për
< element i tabelës >. Me elementin e tabelës kemi disa opsione por më së shpeshti përdoret < definimi i fushës >. Ku me anë të tij bëjmë definimin e fushës në tabelë. Definimi i fushës duhet të bëhet për çdo fushë që do të jetë pjesë e tabelës. Nga sintaksa vërejmë se vetëm dy elemente kërkohen <emri i fushës> dhe <tipi it ë dhënave>. Të gjithë elementet shtesë për definimin e fushës janë opsionale. Definimi i tipit të të dhënave bëhet me anë të tipeve numerike, string dhe datë. Tipet e të dhënave janë të sqaruara më herët.

Deri tani shqyrtuam vetëm elementet e nevojshme (domosdoshme) për krijimin e tabelës dhe definimin e fushës. Si rezultat definimi i fushës përfshinte emrin e fushës dhe tipin e të dhënave për atë fushë. Komponentë tjetër nga definimi i fushës është edhe vlera boshe e fushës e cila caktohet me anë të fjalëve të rezervuara NULL dhe NOT NULL. Me anë të këtyre fjalëve të rezervuara mund të definojmë fushën të pranojë ose jo vlera

boshe. Nëse nuk është definuar as njëra as tjetra prej opsioneve atëherë nënkuptohet se ajo fushë mund të pranojë vlerë boshe.

```
CREATE TABLE Test
(
   ID SMALLINT UNSIGNED NOT NULL,
   Emri VARCHAR(20) NOT NULL
);
```

Definimi i vlerës së parazgjedhur (nënkuptuar) bëhet me anë të fjalës së rezervuar DEFAULT. Kjo është e dobishme për faktin kur në një fushë përsëritet një vlerë.

```
CREATE TABLE Autori
(
ID SMALLINT UNSIGNED NOT NULL,
Viti_i_lindjes YEAR NOT NULL,
Vendlindja VARCHAR(40) NOT NULL DEFAULT 'E panjohur'
Sasia SMALLINT NOT NULL DEFAULT 1
);
```

Për tipet e të dhënave string vlera e DEFAULT shkruhet në thonjëza ndërsa te tipet numerike shkruhet vetëm numri. Nëse nuk caktojmë vlerën e parazgjedhur atëherë MySQL automatikisht i ndan një vlerë asaj fushe varësisht nga tipi i të dhënave. Nëse fusha mund të marrë si vlerën boshe atëherë vlera e parazgjedhur është NULL. Nëse fusha nuk merr vlerën boshe atëherë vlera e parazgjedhur varet nga definimi i fushës:

- Për fushat me tip të të dhënave TIMESTAMP vlera e parazgjedhur është data dhe ora aktuale.
- Për fushat me tip të të dhënave DATE/TIME vlera e parazgjedhur është zero.
- Për fushat me modifikuesin AUTO_INCREMENT vlera e parazgjedhur është numri vijues në vargun rritës.
- Për fushat me tip të të dhënave numerike që nuk e kanë të definuar AUTO_INCREMENT vlera e parazgjedhur është 0.
- Për fushat me tipin ENUM vlera e parazgjedhur është vlera e parë në definimin e fushës.
- Për fushat me tip të të dhënave string vlera e parazgjedhur është një string i zbrazët (string – një varg i karakterëve).

Siç shihet në MySQL të gjitha fushave u është ndarë vlera e parazgjedhur.

Definimi i çelësit primar

Mënyra më e thjeshtë për definimin e çelësit primar është caktimi i opsionit PRIMARY KEY në definimin e fushës, si shembullin e mëposhtëm:

```
CREATE TABLE Porositë
(
ID SMALLINT UNSIGNED NOT NULL PRIMARY KEY,
Modeli SMALLINT UNSIGNED NOT NULL,
Përshkrimi VARCHAR(40)
);
```

SHEMBUJ:
Krijojmë tabelat KLIENTËT dhe KREDITË në bazëne të dhënave BANKA:
Së pari përdorim komandën USE < emri i bazës së të dhënave >, pra USE BANKA;

KLIENTËT:	Tipi i të dhënave
Numri i llogarisë	**Varchar(10) PRIMARY KEY**
Emri	**Varchar(20)**
Mbiemri	**Varchar(20)**
Qyteti	**Varchar(20)**
Adresa	**Varchar(20)**
Tel	**Varchar(20) DEFAULT 'I panjohur'**
KREDITË:	
ID	**INT AUTO_INCREMENT PRIMARY KEY**
Numri I llogarisë	**Varchar(10)**
Tipi i kredisë	**Varchar(20)**
Shuma	**INT**
Përqindja	**INT**
Kohëzgjatja e kredisë	**INT**
Aprovimi	**ENUM('Po','Jo')**
Fillimi	**DATE**

```
CREATE TABLE KLIENTËT
(
Numri_i_llogarisë Varchar(10) NOT NULL PRIMARY KEY,
Emri Varchar(20) NOT NULL,
Mbiemri Varchar(20) NOT NULL,
Qyteti Varchar(20) NOT NULL,
Adresa Varchar(20) NOT NULL,
Tel Varchar(20) NOT NULL DEFAULT 'I panjohur'
);

CREATE TABLE KREDITË
(
ID INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
Numri_i_llogarisë Varchar(10) NOT NULL,
Tipi_i_kredisë Varchar(20) NOT NULL,
Shuma INT NOT NULL,
Përqindja INT NOT NULL,
Kohëzgjatja_e_kredisë INT NOT NULL,
Aprovimi ENUM('Po','Jo') DEFAULT 'Jo',
Fillimi DATE,
FOREIGN KEY (Numri_i_llogarisë) REFERENCES KLIENTËT (Numri_i_llogarisë)
ON DELETE CASCADE ON UPDATE CASCADE
);
```

Në dy rreshtat e fundit (me shkronja të nxira (Bold)) kemi bërë lidhjen e tabelave *KLIENTËT* dhe *KREDITË*. Ku tabela *KLIENTËT* është tabela prind ndërsa tabela *KREDITË* tabela fëmijë. Lidhja në mes këtyre tabelave është një me shumë (1 - ∞) pra është vënë integriteti referncial (sikur në ACCESS). Ndërsa ON DELETE CASCADE dhe ON UPDATE CASCADE ruajnë që të dhënat në tabelën *KREDITË* të mos mbesin jetimë. Pra nëse fshihet (ndryshohet) ndonjë e dhënë në tabelën *KLIENTËT* atëherë të gjitha të dhënat në tabelën *KREDITË* që janë të lidhura me të dhënën përkatëse do të fshihen (ndryshohen).

Përdorim komandën *DESCRIBE < emri i tabelës >* për të shikuar të dhënat e përgjithshme mbi tabelat e krijuara.
mysql> DESCRIBE KLIENTËT;
mysql> DESCRIBE KREDITË;

Manipulimi i të dhënave
Tani pas krijimit të tabelave gjëja e parë që duhet bërë është futja e të dhënave në ato tabela. Për këtë arsye do të mësojmë komandat për futjen e të dhënave INSERT dhe REPLACE. MySQL përkrah edhe metoda tjera për futjen e të dhënave si kopjimi ose importimi i të dhënave. Urdhëri që më së shpeshti përdoret për futjen (insertimin) e të dhënave është INSERT. Ja edhe sintaksa e urdhërit INSERT:

INSERT [LOW_PRIORITY | DELAYED] [IGNORE] [INTO]
{ < opsioni VALUES > | < opsioni SET > | < opsioni SELECT > }

< opsioni VALUES >::=
< emri i tabelës > [(emri i fushës > [{ , < emri i fushës > } …])]
VALUES ({ < shprehja > | DEFAULT } [{ , { < shprehja > | DEFAULT } } …])
[{ , ({ < shprehja > | DEFAULT } [{ , { < shprehja > | DEFAULT } } …]) } …]

< opsioni SET > ::=
< emri i tabelës >
SET < emri i fushës > = { < shprehja > | DEFAULT }
[{ , < emri i fushës > = { < shprehja > | DEFAULT } } …]

< opsioni SELECT > ::=
< emri i tabelës > [(< emri i fushës > [{ , < emri i fushës > } …])]
< urdhëri SELECT >

Komentojmë sintaksën për urdhërin *INSERT*. Në rreshtin e parë është fjala e rezervuar *INSERT* pastaj opsionet.

Opsioni i parë nga këto opsione është *LOW_PRIORITY* dhe *DELAYED*, mund të zgjedhet njëri, asnjërin nga këto dy opsione por zgjedhja e të dyve nuk mund të bëhet. Nëse zgjedhim opsionin *LOW_PRIORITY* atëherë urdhëri nuk do të ekzekutohet derisa asnjë klient të mos jetë duke shfrytëzuar tabelën në të cilën do të ekzekutohet urdhëri. Deri sa nuk bëhet ekzekutimi i urdhërit nuk mund të kryhen veprime tjera. Nëse zgjedhim opsionin *DELAYED* ekzekutimi i urdhërit do të shtyhet por në dallim nga opsioni i parë

ndërkohë mund të kryhen veprime tjera. Këto opsione përdoren vetëm te tabelat e tipit *MyISAM* dhe *ISAM*.

Opsioni tjetër është *IGNORE*, me zgjedhjen e këtij opsioni do të injorohen të gjithë rreshtat që përmbajnë vlera që bëjnë dyfishimin e çelësit primar ose dyfishimin e vlerës së indeksit unik.

Opsioni i fundit është fjala kyçe *INTO*, përdoret për të treguar se në cilën tabelë do të bëhet insertimi i të dhënave.

Në rreshtin e dytë janë tri opsione prej të cilave duhet të zgjedhim.

Opsioni *VALUES* na mundëson futjen e më shumë se një rreshti në tabelë.

Nga sintaksa vërejmë se duhet të ipet emri i tabelës pastaj opsioni *VALUES*. Poashtu mund të caktojmë fushat e tabelës të cilët duhet të jenë të ndarë mes veti me presje dhe të futura në kllapa të vogla. Pas opsionit *VALUES* duhet të ipen vlerat që do të futen në tabelë për secilën fushë të deklaruar. Nëse nuk caktojmë emrat e fushave atëherë vlerat që do të ipen për secilën fushë duhet të jenë të renditura si në tabelë.Nëse keni harruar renditjen e fushave në tabelë, përdorni urdhërin DESCRIBE.

Të gjitha vlerat duhet të jenë brenda kllapave të vogla dhe të ndara mes veti me presje. Për fushat të definuara si *AUTO_INCREMENT* ose *TIMESTAMP* është e preferuar vlera *DEFAULT*.

Shembuj:

Fusim të dhënat në tabelën *KLIENTËT*:

INSERT INTO KLIENTËT (Numri_i_llogarisë, Emri, Mbiemri, Qyteti, Adresa, Tel)
VALUES (‘140201’,’Bujar’,’Shulemaja’,’Prishtinë’,’Bregu i diellit’,’044/123-456’);

INSERT INTO KLIENTËT (Numri_i_llogarisë, Emri, Mbiemri, Qyteti, Adresa, Tel)
VALUES (‘140202’,’Amir’,’Simnica’,’Fushë Kosovë’,’rr.Agim Ramadani’,’044/321-654’);

INSERT INTO KLIENTËT (Numri_i_llogarisë, Emri, Mbiemri, Qyteti, Adresa, Tel)
VALUES (‘140203’,’Valbona’,’Krasniqi’,’Dardanë’,’rr.Adem Jashari’,’044/213-546’);

INSERT INTO KLIENTËT (Numri_i_llogarisë, Emri, Mbiemri, Qyteti, Adresa, Tel)
VALUES (‘140204’,’Gani’,’Thaqi’,’Besianë’,’rr.Zahir Pajaziti’,’044/312-645’);

INSERT INTO KLIENTËT (Numri_i_llogarisë, Emri, Mbiemri, Qyteti, Adresa, Tel)
VALUES (‘140205’,’Lavdim’,’Kastrati’,’Prishtinë’,’Tophane’,’044/132-465’);

Kjo mund të shkruhet edhe si:

*INSERT INTO KLIENTËT (**Numri_i_llogarisë, Emri, Mbiemri,Qyteti, Adresa,Tel)***
VALUES
(‘140201’,’Bujar’,’Shulemaja’,’Prishtinë’,’Bregu i diellit’,’044/123-456’),
(‘140202’,’Amir’,’Simnica’,’Fushë Kosovë’,’rr.Agim Ramadani’,’044/321-654’),
(‘140203’,’Valbona’,’Krasniqi’,’Dardanë’,’rr.Adem Jashari’,’044/213-546’),
(‘140204’,’Gani’,’Thaqi’,’Besianë’,’rr.Zahir Pajaziti’,’044/312-645’),
(‘140205’,’Lavdim’,’Kastrati’,’Prishtinë’,’Tophane’,’044/132-465’);

Nëse e dimë renditjen e fushave atëherë nuk kemi nevojë të shkruajmë pjesën me shkronja të nxira (bold).

Opsioni *SET* është i përshtatshëm nëse dëshirojmë të fusim të dhëna vetëm në disa fusha. Nga sintaksa shihet se së pari duhet të shkruhet emri i tabelës pastaj *SET* dhe fushat që dëshirojmë të ju ndajmë vlerë.

Shembuj:
Fusim të dhëna në tabelën *KREDITË*:

INSERT INTO KREDITË SET Numri_i_llogarisë='140203',Tipi_i_kredisë='Veturë', Shuma=4500,Përqindja=6,Kohëzgjatja_e_kredisë=12,Aprovimi=1,Fillimi=20060412;

INSERT INTO KREDITË SET Numri_i_llogarisë='140205',Tipi_i_kredisë='Mobile', Shuma=1300,Përqindja=7.5,Kohëzgjatja_e_kredisë=6,Aprovimi=1,Fillimi=20060823;

INSERT INTO KREDITË SET Numri_i_llogarisë='140201',Tipi_i_kredisë='Veturë', Shuma=7000,Përqindja=5.5,Kohëzgjatja_e_kredisë=36,Aprovimi=1, Fillimi=20050215;

INSERT INTO KREDITË SET Numri_i_llogarisë='140203',Tipi_i_kredisë='Shtëpi', Shuma=45000,Përqindja=4.7,Kohëzgjatja_e_kredisë=120,Aprovimi=2, Fillimi=20050520;

INSERT INTO KREDITË SET Numri_i_llogarisë='140202',Tipi_i_kredisë='Kompjuter', Shuma=630,Përqindja=7,Kohëzgjatja_e_kredisë=12,Aprovimi=1,Fillimi=20060712;

INSERT INTO KREDITË SET Numri_i_llogarisë='140204,Tipi_i_kredisë='Kuzhinë', Shuma=1000,Përqindja=4,Kohëzgjatja_e_kredisë=12,Aprovimi=1,Fillimi=20050904;

INSERT INTO KREDITË SET Numri_i_llogarisë='140205',Tipi_i_kredisë='Lavatriçe', Shuma=420,Përqindja=3,Kohëzgjatja_e_kredisë=6,Aprovimi=1,Fillimi=20060605;

INSERT INTO KREDITË SET Numri_i_llogarisë='140201',Tipi_i_kredisë='Mobile', Shuma=1730,Përqindja=5.8,Kohëzgjatja_e_kredisë=12,Aprovimi=1, Fillimi=20061002;

INSERT INTO KREDITË SET Numri_i_llogarisë='140205',Tipi_i_kredisë='Veturë', Shuma=5250,Përqindja=7.3,Kohëzgjatja_e_kredisë=18,Aprovimi=1, Fillimi=20050412;

INSERT INTO KREDITË SET Numri_i_llogarisë='140202',Tipi_i_kredisë='Tavolinë', Shuma=420,Përqindja=8.3,Kohëzgjatja_e_kredisë=12,Aprovimi=2,Fillimi=20050812;

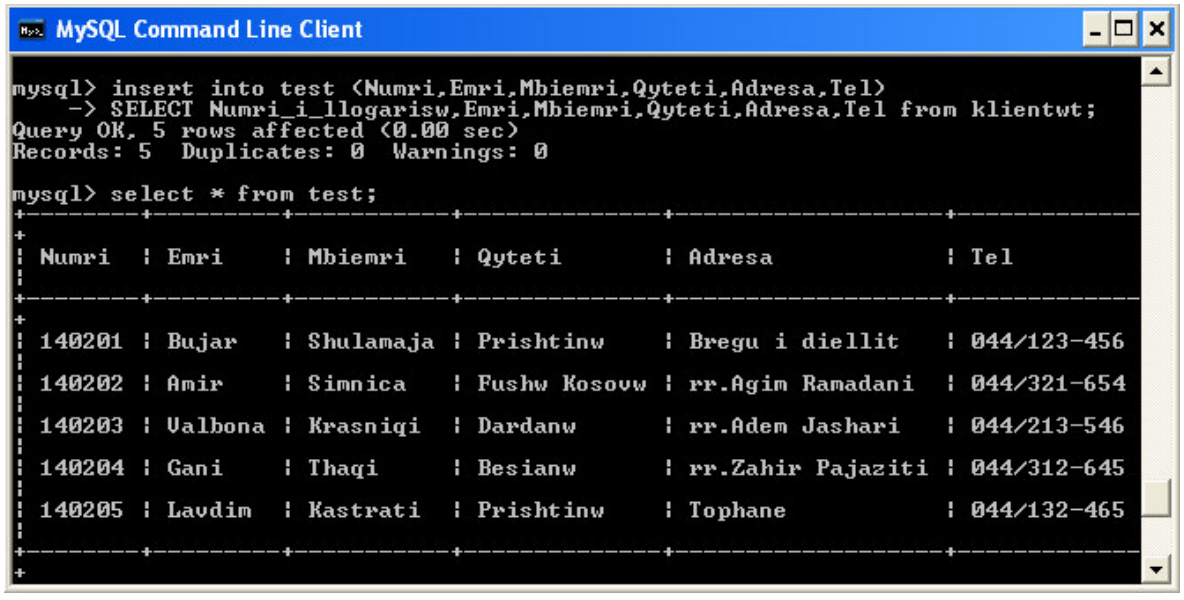
Opsioni *SELECT* më së shpeshti përdoret për kopjimin e të dhënave të një tabele në një tjetër, si shembull:
Krijojmë tabelën *TEST* në bazën e të dhënave *BANKA*.

```
CREATE TABLE TEST
(
Numri VARCHAR(10) NOT NULL PRIMARY KEY,
Emri VARCHAR(20) NOT NULL,
Mbiemri VARCHAR(20) NOT NULL,
Qyteti VARCHAR(20) NOT NULL,
Adresa VARCHAR(20) NOT NULL,
Tel VARCHAR(20) NOT NULL
);
```

Pastaj shkruajmë

```
INSERT INTO TEST (Numri, Emri, Mbiemri, Qyteti, Adresa, Tel)
SELECT Numri_i_llogarisë,Emri,Mbiemri,Qyteti,Adresa,Tel FROM KLIENTËT;
```

Ekzekutohet, dhe të gjitha të dhënat e tabelës *KLIENTËT* do të gjinden edhe në tabelën *TEST*.



Njësoj si urdhëri *INSERT* përdoret urdhëri *REPLACE,* sintaksa e të cilit ndryshon me atë të urdhërit *INSERT* vetëm se nuk e përkrah opsionin *IGNORE*. Ndërsa të gjitha opsionet tjera *VALUES, SET* dhe *SELECT* i përkrah.

Modifikimi i tabelës

Nuk është e pazakontë, që pas krijimit të tabelës të kemi nevojë për ndonjë ndryshim në definimin e saj. Për fat të mirë MySQL na mundëson ndryshimin e elementeve të tabelës pas krijimit të saj. P.SH. mund të shtojmë fusha, të ndryshojmë definimin e fushave në tabelë, shtojmë çelësin primar dhe çelësin e jashtëm ose fshirjen e fushave. Për modifikimin e një tabele përdorim urdhërin *ALTER TABLE*. Ja edhe sintaksa e urdhërit *ALTER TABLE:*

ALTER TABLE <emri i tabelës>
<opsioni alter> [{, <opsioni alter>}...]

<opsioni alter>::=
{ADD [COLUMN] <definimi i fushës> [FIRST | AFTER <emri i fushës>]}
| {ADD [COLUMN] (<element i tabelës> [{, <element i tabelës>}...])}
| {ADD [CONSTRAINT <emri i rregullës>] PRIMARY KEY
(<emri i fushës> [{, <emri i fushës>}...])}
| {ADD [CONSTRAINT <emri i fushës>] FOREIGN KEY [<emri i indeksit>]
(<emri i fushës> [{, < emri i fushës>}...]) <definimi i referencës>}
| {ADD [CONSTRAINT <emri i rregullës>] UNIQUE [<emri i indeksit>]
(<emri i fushës> [{, <emri i fushës>}...])}
| {ADD INDEX [<emri i indeksit>] (<emri i fushës> [{, <emri i fushës>}...])}
| {ADD FULLTEXT [<emri i indeksit>] (<emri i fushës> [{, <emri i fushës>}...])}
| {ALTER [COLUMN] <emri i fushës> {SET DEFAULT <vlerë> | DROP
DEFAULT}}
| {CHANGE [COLUMN] <emri i fushës> <definimi i fushës> [FIRST | AFTER
< emri i fushës>]}
| {MODIFY [COLUMN] <definimi i fushës> [FIRST | AFTER <emri i fushës>]}
| {DROP [COLUMN] <emri i fushës>}
| {DROP PRIMARY KEY}
| {DROP INDEX <emri i indeksit>}
| {DROP FOREIGN KEY <emri i rregullës>}
| {RENAME [TO] <emri i tabelës së re>}
| {ORDER BY <emri i fushës> [{, <emri i fushës>}...]}
| {<tip i tabelës> [<tip i tabelës>...]}

Elementet e domosdoshme në sintaksë për modifikimin e tabelës janë: ALTER TABLE, emri i tabelës që do të modifikojmë dhe një ose më shumë opsione për ndryshim (opsionet alter). Nëse zgjedhim më shumë se një opsion për ndryshim atëherë ato duhet të ndahen me presje. Supozojmë se krijojmë një tabelë:

```
CREATE TABLE Librat
  (
    ID SMALLINT NOT NULL,
    Titulli_i_librit VARCHAR(40) NOT NULL,
    IDBotuesi SMALLINT NOT NULL DEFAULT 'I panjohur'
```

)
ENGINE=INNODB;

Siç mund të shihet krijuam një tabelë me emrin Librat që përmban tri fusha. Tani supozojmë që dëshirojmë të bëjmë ndryshime në definimin e këtyre fushave. Dëshirojmë që në tabelë të ketë çelës primar, çelës të jashtëm dhe fusha shtesë. Urdhëri vijues *ALTER TABLE* e bën atë:

ALTER TABLE Librat
ADD PRIMARY KEY (ID),
ADD CONSTRAINT emri_i_rregullës FOREIGN KEY (IDBotuesi) REFERENCES Botuesi (IDBotuesi),
ADD COLUMN Zhanri ENUM('Dramë','Roman','Poezi') NOT NULL AFTER Titulli_i_librit;

Urdhëri fillon me *ALTER TABLE* e cila identifikon emrin e tabelës që do të modifikojmë, që në rastin tonë është tabela *Librat*. Rreshti tjetër shton çelësin primar në tabelë duke u bazuar në fushën *ID*. Rreshti i tretë shton një çelës të jashtëm (foreign key) në tabelë dhe në fund shtuam një fushë me emrin *Zhanri* me tipin e të dhënave *ENUM* dhe të paraqitet në renditje pas fushës *Titulli_i_librit*. Nëse gjatë krijimit të tabelës gabojmë në definimin e fushës atëherë përmisimet mund të bëhen kështu, shembull: nga tabela Librat dëshirojmë që emri i fushës *Titulli_i_librit* të jetë *Titulli* dhe tipi i të dhënave *VARCHAR(10)* dhe të paraqitet pas fushës *IDBotuesi*

ALTER TABLE Librat
CHANGE Titulli_i_librit Titulli VARCHAR(10) NOT NULL AFTER IDBotuesi;

Të njëjtën gjë mund t'a bëjmë edhe me *MODIFY* por aty nuk mund të bëjmë ndryshimin e emrit të fushës.

Me anë të *ALTER TABLE* mund të bëjmë riemërimin e tabelës:

ALTER TABLE Librat RENAME TO Romanet;

Urdhëri *ALTER TABLE* shfrytëzohet edhe për largimin e fushave nga tabela,

ALTER TABLE Librat
DROP COLUMN Zhanri,
DROP PRIMARY KEY;

Fshirja e tabelës

Fshirja e tabelës bëhet me anë të urdhërit DROP TABLE dhe sintaksa e tij është:

DROP [TEMPORARY] TABLE [IF EXISTS] <emri i tabelës> [{,<emri i tabelës>} …]

Urdhëri *DROP TABLE* përfshin opsionin *TEMPORARY* i cili përdoret për fshirjen e tabelave të përkohshme duke mos bërë fshirjen aksidentale të ndonjë tabele të përhershme. Tjetër opsion është *IF EXISTS*, ashtu si te tabelat edhe këtu është më mirë të përdoret kjo fjalë e rezervuar. Më mirë të marrim një vërejtje se ndonjë gabim kur dëshirojmë të fshijmë ndonjë tabelë që nuk ekziston. Poashtu këtë urdhër mund të përdorim për fshirjen e më shumë se një tabele në të njëjtën kohë, duke bërë ndarjen e tyre me presje.

Shembuj:

DROP TABLE IF EXISTS Librat;
DROP TEMPORARY TABLE Test, Test1, Test2;
DROP TEMPORARY TABLE IF EXISTS Test, Test1, Test2;

Urdhëri *SHOW*

Me anë të këtij urdhëri mund të shohim të dhëna specifike për çdo bazë të të dhënave dhe tabelave. Urdhëri

SHOW CREATE DATABASE <emri i bazës së të dhënave>

Pra ky urdhër kërkon vetëm emrin e bazës së të dhënave, p.sh:

SHOW CREATE DATABASE Banka;

Urdhëri ***SHOW DATABASES***; përdoret për shikimin e të gjitha bazave të të dhënave. Sintaksa është

SHOW DATABASES;

Urdhëri SHOW COLUMNS liston fushat në tabelë me të gjitha të dhënat për ato fusha. Sintaksa është

SHOW [FULL] COLUMNS FROM <emri i tabelës> FROM <emri i bazës së të dhënave] [LIKE <shprehje>]

Një shembull:
SHOW COLUMNS FROM user FROM mysql LIKE '%priv';

Poashtu urdhëri *SHOW* mund të përdoret edhe për të dhënat mbi tabelat, ja edhe sintaksa:

SHOW TABLES [FROM <emri i bazës së të dhënave>] [LIKE <shprehje>]
Shembull:

SHOW TABLES FROM mysql LIKE 'help%';

Urdhëri ***SHOW TABLES*** tregon të gjitha tabelat në bazë të të dhënave.

Përdorimi i urdhërit *DESCRIBE*

Një urdhër tjetër i përshtatshëm për shikimin e informacioneve mbi tabela është *DESCRIBE*.
Sintaksa:
DESCRIBE <emri i tabelës> [<emri i fushës> | <shprehje>];
Shembull:
DESCRIBE user '%priv';

Editimi (ndryshimi) i të dhënave në MySQL

Pasi kemi mësuar insertimin e të dhënave në një bazë të të dhënave tani jemi gati për të mësuar editimin e tyre. Editimi bëhet me anë të urdhërit UPDATE, ja edhe sintaksa:

UPDATE [LOW_PRIORITY] [IGNORE]
<editimi i një tabele> | <editimi i tabelës së lidhur>

<editimi i një tabele>::=
<emri i tabelës>
SET <emri i fushës>=<shprehje> [{, <emri i fushës>=<shprehje>}...]
[WHERE <kushti>]
[ORDER BY <emri i fushës> [ASC | DESC] [{, <emri i fushës > [ASC | DESC]}...]]
[LIMIT <numri i rreshtave>]

<editimi i tabelës së lidhur>::=
<emri i tabelës> [{, <emri i tabelës>}...]
SET <emri i fushës>=<shprehje> [{, <emri i fushës>=<shprehje>}...]
[WHERE <kushti>]

Rreshti i parë i sintaksës përmban fjalën e detyrueshme *UPDATE* bashkë me opsionet *LOW_PRIORITY* dhe *IGNORE* të cilat i pamë edhe te urdhëri *INSERT*. Opsionin *LOW_PRIORITY* e zgjedhim kur dëshirojmë të shtyjmë ekzekutimin e urdhërit *UPDATE* derisa asnjë klient të mos jetë i kyçur në tabelën që do të bëjmë editimin. Opsionin *IGNORE* e përdorim nëse dojmë të bëjmë editimin edhe pse ka çelës primar apo indeks unik të dyfishtë (rreshti me vlerë të dyfishtë nuk do të editohet).
Në rreshtin e dytë janë dy alternativa *<editimi i një tabele>, <editimi i tabelës së lidhur>*. Editimi i tabelës së lidhur i referohet tabelës që është e lidhur me një tabelë tjetër me anë të urdhërave SQL. Kjo lidhje është e bazuar në *FOREIGN KEY*, pra me lidhjen në mes të dy tabelave.
Ndërsa alternativa e parë editimi i një tabele, nuk ka kushte specifike për lidhjen në mes të tabelave, duhet të krijojmë urdhërin *UPDATE* si më lartë. Siç shihet nga sintaksa së pari duhet të specifikojmë emrin e tabelës pastaj fjalën e rezervuar SET. Fjala *SET* përmban së paku një emër të një fushe të shoqëruar me ndonjë vlerë. Nëse dëshirojmë që të bëjmë editimin e më shumë se një fushe atëherë këtë e bëjmë duke i ndarë çiftin emrin

e fushës/shprehjen me presje.Veç kësaj në alternativën e parë mund të bëjmë edhe zgjedhjen e disa opsioneve. I pari nga ata është *WHERE* i cili përcakton cilët rreshta do të editohen. Pasi që *WHERE* është si pjesë integrale e urdhërit *SELECT* ky do të mësohet në detale më vonë.
Opsioni i dytë *ORDER BY* mundëson editimin e rreshtave sipas një renditje të caktuar duke u bazuar në vlerat në fushë ose fusha. Opsioni *ORDER BY* mund të vëhet për më shumë se një fushë. Veç kësaj për secilën fushë të përfshirë në fjalën e rezervuar *ORDER BY* mund të zgjedhen opsionet *ASC* dhe *DESC*. *ASC* do të thotë editimi i rreshtave sipas renditjes rritëse, ndërsa *DESC* editimi i rreshtave sipas renditjes zbritëse.
Opsion tjetër është fjala e rezervuar *LIMIT*. Me anë të këtij opsioni mund të kufizojmë numrin e rreshtave që do të editohen. P.SH. Nëse me urdhërin *UPDATE* do të editoheshin 10 rreshta, atëherë me urdhërin *UPDATE* por me *LIMIT* 5 do të editohen vetëm 5 rreshtat e parë.

Shembuj:

```
UPDATE Kreditë
SET Shuma=Shuma+10;
```

```
UPDATE Kreditë
SET Shuma=Shuma+15
WHERE Numri_i_llogarisë='140204';
```

```
UPDATE Kreditë
SET Shuma=Shuma+15
WHERE Përqindja=5
ORDER BY Fillimi DESC
LIMIT 3;
```

Në urdhërin e fundit editohet tabela Kreditë, fusha Shuma rritet për 15 ku fusha Përqindja ka vlerën 5, në renditjen DESC sipas fushës Fillimi e cila pasi që është e definuar si TIMESTAMP ateherë rreshtat e edituar do të jenë ato të datës më të vonshme, dhe në fund editimi është i kufizuar në 3 rreshtat e parë.

Fshirja e të dhënave nga tabela

Është e pashmangshme që pas insertimit të të dhënave disa duhet të fshihen. Kur të fshijmë të dhëna nga një tabelë atëherë fshihet një ose më shumë rreshta. Nuk mund të fshijmë vetëm një pjesë të një rreshti. MySQL përkrah dy urdhëra që përdoren për fshirjen e të dhënave: *DELETE* dhe *TRUNCATE.*

Urdhëri *DELETE*

Urdhëri *DELETE* duhet të jetë urdhëri primar për fshirjen e të dhënave nga tabela.
Sintaksa:
DELETE [LOW_PRIORITY] [QUICK] [IGNORE]

{<fshirja në një tabelë> | <fshirja from> | <fshirja using>}

< fshirja në një tabelë>::=
FROM <emri i tabelës>
[WHERE <kushti>]
[ORDER BY <emri i fushës> [ASC | DESC] [{, <emri i fushës> [ASC | DESC]}...]]
[LIMIT <numri i rreshtave>]
<fshirja from>::=
<emri i tabelës>[.] [{, <emri i tabelës>[.*]}...]*
FROM <emri i tabelës> [{, <emri i tabelës>}...]
[WHERE <kushti>]

<fshirja using>::=
FROM <emri i tabelës>[.] [{, <emri i tabelës>[.*]}...]*
USING <emri i tabelës> [{, <emri i tabelës>}...]
[WHERE <kushti>]

Në rreshtin e parë është fjala kyçe *DELETE* së bashku me opsionet *LOW_PRIORITY*, *QUICK* dhe *IGNORE*. Opsionet *LOW_PRIORITY* dhe *IGNORE* i pamë më herët te urdhërat *INSERT* dhe *UPDATE*. Nëse zgjedhim opsionin *LOW_PRIORITY* atëherë urdhëri *DELETE* nuk do të ekzekutohet derisa të ç'kyçen të gjithë shfrytëzuesit nga tabela ku do të bëhet fshirja e të dhënave. Opsioni *IGNORE* nuk kthen gabime, vetëm lajmërime kur nuk mund të fshihet ndonjë rresht. Opsioni tjetër *QUICK* zbatohet vetëm te tabelat e tipit MyISAM.

Fshirja e të dhënave në një tabelë

Si alternativë e parë për fshirjen e të dhënave përdoret fshirja nga një tabelë. Nga sintaksa shihet se kërkohet fjala kyçe *FROM* që definon emrin e tabelës nga e cila do të fshihen të dhënat. Poashtu ka mundësi opsionesh si fjala *WHERE, ORDER BY* dhe *LIMIT*. Opsioni *WHERE* përmban kushtin me anë të cilit do të fshihen të dhënat. *ORDER BY* bën sortimin e fushave. *LIMIT* kufizon numrin e rreshtave që do të fshihen.

Shembuj:

DELETE FROM Test;

Nga ky urdhër kuptojmë, janë shkruar fjalët kyçe DELETE FROM dhe emri i tabelës. Pasi nuk ka ndonjë kusht tjetër atëherë të gjitha të dhënat nga tabela Test do të fshihen.

DELETE FROM Test
WHERE ID=12;

Fjala *WHERE* bën që të fshihen vetëm të dhënat që kanë fushën ID të barabartë me 12. Rreshtat që nuk plotësojnë këtë kusht nuk fshihen.

DELETE FROM Test
WHERE ID=13
ORDER BY Data DESC
LIMIT 1;

Në këtë urdhër rreshtat që do të fshihen janë: me ID=13, të sortuar duke u bazuar në fushën Data me renditje zbritëse (duke kuptuar se rreshti me datë më të vonë do të fshihet i pari). Ndërsa LIMIT kufizon numrin e rreshtave që do të fshihen, në rastin tonë vetëm një.

Fshirja e të dhënave nga tabelat e lidhura mes veti

Urdhëri DELETE përkrah dy alternativa për fshirjen e të dhënave nga tabelat e lidhura mes veti: <fshirjen from> dhe <fshirjen using>.

Alternativa <fshirja from>

Kjo alternativë është e ngjashme me atë të fshirjes nga një tabelë në atë se përmban fjalët *FROM* dhe *WHERE*. Por nga sintaksa mund të vërejmë se nuk përfshihet fjala *ORDER BY* dhe *LIMIT*. Nga sintaksa vërejmë pikën (.) dhe yllin (*) që shoqërojnë emrin e tabelës. Pika dhe ylli janë opsionale. Me anë të tyre tregojmë se të gjitha fushat e tabelës janë të përfshira.

Shembull:

*DELETE Test.**
FROM Test1, Test
WHERE Test1.ID=Test.ID
AND Test1.Emri='AAAA';

Nga ky urdhër kuptojmë: tabela Test është tabela nga e cila do të fshihen të dhënat, fjala *FROM* përcakton tabelat e lidhura që në rastin tonë janë Test dhe Test1. Fjala *WHERE* tregon lidhjen në mes atyre tabelave dhe përfshin edhe kushtin e dytë.

Alternativa <fshirja using>

Dallimi primar në mes <fshijes using> dhe <fshirjes from> është se në alternativën <fshirja using> tabela nga e cila do të fshihen të dhënat caktohet në fjalën *FROM* dhe lidhja e tabelave caktohet me fjalën USING. Të gjitha aspektet tjera të këtyre dy alternativave janë të njëjta.

Shembull:
DELETE FROM Test
USING Test, Test1
WHERE Test.ID=Test1.ID
AND Test.Emri='AAAA';

Siç mund të shihni fjala *DELETE* nuk e cakton emrin e tabelës dhe përdorimi i fjalës *WHERE* është identik si te alternativa <fshirja from>.

Përdormi i urdhërit TRUNCATE për fshirjen e të dhënave

Me anë të këtij urdhëri bëjmë fshirjen e të gjitha të dhënave nga tabela. Sintaksa

TRUNCATE [TABLE] <emri i tabelës>

Siç mund të shihet duhet të shkruhet fjala kyçe *TRUNCATE* dhe emri i tabelës. Mund të shkruajmë edhe opsionin *TABLE* por nuk ka ndonjë efekt. P.SH. urdhëri i mëposhtëm bën fshirjen e të dhënave nga tabela Test:

TRUNCATE TABLE Test;

Pra urdhëri *TRUNCATE* përmban opsionin *TABLE* dhe emrin e tabelës. Ekzekutimi i urdhërit *TRUNCATE* është i njëjtë me ekzekutimin e urdhërit vijues *DELETE*:

DELETE FROM Test;

Dallimi thelbësor ndërmjet urdhërave *TRUNCATE* dhe *DELETE* është se urdhëri *TRUNCATE* nuk ka siguri gjatë kryerjes së transaksioneve. Dallim tjetër është se urdhëri *TRUNCATE* e starton numrin e *AUTO_INCREMENT* nga fillimi për dallim nga urdhëri *DELETE.* Në përgjithësi urdhëri *TRUNCATE* është më i shpejtë se urdhëri *DELETE*.

Shikimi i të dhënave në MySQL

Një nga funksionet kryesore që një system i menaxhimit të bazave relacionale duhet të përkrah është qasja e të dhënave. Kjo qasje duhet të mundësoj shikimin e të dhënave të dëshiruara si dhe mënyrën se si ato do të paraqiten. Për përkrahjen e këtyre veprimeve MySQL është i pajisur me SQL urdhërin *SELECT*. Ky urdhër përdoret për shikimin e të dhënave në gati të të gjitha sistemet e menaxhimit të bazave relacionale. Me anë të urdhërit SELECT mund të zgjedhim cilat fusha dhe cilët rreshta nga tabela ose tabelat dëshirojmë t'i shohim.

Urdhëri *SELECT*

Sa herë që dëshirojmë të shohim të dhëna nga ndonjë bazë e të dhënave përdorim urdhërin *SELECT*, me anë të të cilit përcaktojmë të dhënat që dëshirojmë t'i shohim. Urdhëri *SELECT* është urdhëri më i fuqishëm në MySQL. Ka një fleksibilitet të madh dhe mundëson krijimin e pyetësorëve prej atyre më të thjeshtë deri te ata më kompleks. Sintaksa e urdhërit *SELECT* është e përbërë nga një numër i fjalëve dhe elementeve të ndryshme, ku shumica nga to janë opsionale. Ja edhe sintaksa:

SELECT
[<opsionet select> [<opsionet select>...]]
{ | <lista e elementeve>}*
[<definimi i ekportit>]
[
FROM <tabela referuese> [{, <tabela referuese>}...]
[WHERE <shprehje> [{<operator> <shprehje>}...]]
[GROUP BY <definimi i group by>]
[HAVING <shprehje> [{<operator> <shprehje>}...]]
[ORDER BY <definimi për order by>]
[LIMIT [<fillimi>,] <numri rrestave>]
[PROCEDURE <emri i procedurës> [(<argumenti> [{, <argumenti>}...])]]
[{FOR UPDATE} | {LOCK IN SHARE MODE}]
]

<opsionet select>::=
{ALL | DISTINCT | DISTINCTROË}
| HIGH_PRIORITY
| {SQL_BIG_RESULT | SQL_SMALL_RESULT}
| SQL_BUFFER_RESULT
| {SQL_CACHE | SQL_NO_CACHE}
| SQL_CALC_FOUND_ROËS
| STRAIGHT_JOIN

<lista e elementeve>::=
{<emri i fushës> | <shprehje>} [[AS] <pseudonim>]
[{, {<emri i fushës> | <shprehje>} [[AS] <pseudonim>]}...]

<definimi i ekportit>::=
INTO OUTFILE '<emri i datotekës>' [<opsionet e ekportimit>
[<opsionet e ekportimit >]]
| INTO DUMPFILE '<emri i datotekës >'

< opsionet e ekportimit >::=
{FIELDS
[TERMINATED BY '<vlerë>']
[[OPTIONALLY] ENCLOSED BY '<vlerë>']
[ESCAPED BY '<vlerë>']}
| {LINES
[STARTING BY '<vlerë>']
[TERMINATED BY '<vlerë>']}

< tabela referuese >::=
<emri i tabelës> [[AS] <pseudonim>]
[{USE | IGNORE | FORCE} INDEX <emri i indeksit> [{, <emri i indeksit>}...]]

<definimi i group by>::=
<emri i fushës> [ASC | DESC]
[{, <emri i fushës> [ASC | DESC]}...]
[WITH ROLLUP]

<definimi për order by>::=
<emri i fushës> [ASC | DESC]
[{, <emri i fushës> [ASC | DESC]}...]

Siç mund të shihet nga sintaksa urdhëri *SELECT* përmban një numër të madh të elementeve. Duke ju referuar sintaksës vërejmë se elementet e domosdoshme janë:

*SELECT { * | <lista e elementeve> }*

Pra sintaksa është e përbërë nga fjala kyçe *SELECT* dhe * ose lista e elementeve, e cila është e përbërë nga fushat ose shprehjet si më poshtë:

<lista>::=
{<emri i fushës> | <shprehje>} [[AS] <pseudonim>]
[{, {<emri i fushës> | <shprehje>} [[AS] <pseudonim>]}...]

Pra, sintaksa duhet të ketë së paku një fushë ose shprehje. Nëse përfshihet më shumë se një fushë ose shprehje atëherë ato ndahen me presje. Si shtesë apo si opsion është ndarja e një pseudonimi për fushat ose shprehjet duke përdor fjalën *AS*.
Për shikimin e të dhënave nga ndonjë tabelë duhet të përdorim fjalën *FROM* dhe tabelën referuese. Fjala *FROM* kërkon një ose më shumë tabela referuese të ndara mes veti me presje si më poshtë:

FROM <tabela referuese> [{, <tabela referuese>}...]

< tabela referuese >::=
<emri i tabelës> [[AS] <pseudonim>]
[{USE | IGNORE | FORCE} INDEX <emri i indeksit> [{, <emri i indeksit>}...]]

Secila tabelë referuese është emri i tabelës, zgjedhja *AS* mundëson ndarjen e një pseudonimi për atë tabelë. Fjala *FROM* mund të përmbajë më shumë se një tabelë referuese. Për krijimin e urdhërit *SELECT* për shikimin e të dhënave nga një tabelë kërkon një numër të vogël elementesh. Se si funksionon kjo do ta shohim me disa shembuj:

*SELECT * FROM Klientet;*

*SELECT * FROM Kredite;*

SELECT Emri, Mbiemri FROM Kredite;

Në urdhërin e parë përfshihet fjala kyçe *SELECT* pastaj *, me anë të * kërkojmë nga MySQL që të kthejë si rezultat të gjitha fushat. Ndërsa me *FROM* përcaktojmë se nga cila tabelë dëshirojmë t'i shohim këto të dhëna. Kur të ekzekutojmë këtë urdhër do të shohim të gjitha të dhënat nga tabela Klientet. Përdorimi i * është një mënyrë e lehtë për shikimin e rezultatit për secilën fushë. Nëse dëshirojmë të shohim të dhëna nga disa fusha të caktuara atëherë shkruajmë fjalën kyçe *SELECT* dhe emrat e fushave nga të cilat dëshirojmë të shohim të dhënat të ndara mes veti me presje dhe *FROM* emri i tabelës, ashtu si më lartë te shembulli në rreshtin e tretë. Duke ju referuar sërish sintaksës për urdhërin *SELECT* shohim mundësinë e ndarjes së pseudonimit për secilin element të listës. Kjo bëhet si më poshtë:

SELECT Emri AS Filan, Mbiemri AS Fisteku FROM Klientet;

SELECT Emri AS 'Emri i klientit', Mbiemri AS 'Mbiemri i klientit' FROM Klientet;

Në shembullin e mësipërm në rreshtin e dytë shohim se në të dy rastet pseudonimet janë futur në thonjëza, kjo është pë shkak se ato përbëhen nga më shumë se një fjalë dhe futja në thonjëza i tregon MySQL se ato fjalë janë pjesë e pseudonimit.

Përdorimi i shprehjeve në urdhërin SELECT

Në sintaksën e urdhërit *SELECT <lista e elementeve>* përveç emrit të fushës përfshinte edhe shprehjen. Le të shohim një urdhër SELECT që përmban një shprehje:

SELECT Emertimi AS 'Emri i mallit', NeDepo+TeShitura AS Total
FROM Dyqani;

Elementi i parë i listës së elementeve është i njëjtë si te shembujt e mëparshëm, ndërsa elementi i dytë është shprehje. Shprehja (NeDepo+TeShitura) mbledh vlerat nga fusha NeDepo dhe TeShitura. Kësaj shprehje i është ndarë emri Total, i cili emër do të paraqitet në rezultatin e ekzekutimit të këtij pyetësori. Mund të krijohen shprehje edhe më të komplikuara varësisht nga llogaritja që bëhet.

Përdorimi dhe definimi i variablave në urdhërin *SELECT*

Një lehtësim tjetër te shprehjet është mundësia e krijimit dhe definimit të variablave. Variabla duhet të filloj me shenjën @:

@Emri, @Mbiemri etj.

Ndarja e vlerës për variablat bëhet me anë të urdhërit *SET*, P.SH.

SET @Numri='140201';
SET @Emri='Bujar';
SET @Shuma=550;

Shembull:
SELECT 1+3, 'Klientet', NOW( ) AS 'Data dhe ora';

Me ekzekutimin e këtij urdhëri marrim si rezultat:

```
+-----+-------------+-------------------------+
| 1+3 |   Klientet  | Data dhe ora            |
+-----+-------------+-------------------------+
|  4  |   Klientet  | 2004-08-24 11:39:40 |
+-----+-------------+-------------------------+
1 row in set (0.00 sec)
```

Elementi i parë tregon se si me urdhërin *SELECT* mund të bëjmë llogaritje, elementi i dytë 'Klientet' është një string i cili paraqitet edhe si rezultat, në fund elementi i tretë NOW( ) AS 'Data dhe ora' kthen datën dhe orën aktuale.

Opsionet te urdhëri *SELECT*

Opsioni *ALL* përcakton që pyetësori mund të kthejë si rezultat të gjithë rreshtat edhe pse ka rreshta duplikat. Opsioni *DISTINCT* dhe *DISTINCTROW* kanë të njëjtën domethënie, përcaktojnë që në rezultatin e pyetësorit të mos paraqiten rreshtat me të dhëna të dyfishta. Nëse nuk zgjedhim asnjërin nga këto opsione nënkuptohet se është zgjedhur ALL.
Në tabelën Klientet nga shembujt e mëparshëm shtojmë këto të dhëna:

INSERT INTO Klientet VALUES
('140206','Bujar','Krasniqi','Pejë,'rr.Gjimnazi','044/123-678'),
('23','Gani','Thaqi','Besianë','rr.Zahir Pajaziti','044/312-645');

Ekzekutojmë këtë urdhër dhe pastaj shkruajmë:

SELECT ALL Emri FROM Klientet;

Do të paraqiten të gjithë rreshtat nga kjo tabelë, ku te fusha Emri paraqitet Gani dhe Bujar dyherë, ndërsa me ekzekutimin e urdhërit:

SELECT DISTINCT Emri FROM Klientet;

Emrat nuk janë duplikat.

SELECT ALL Emri, Mbiemri FROM Klientet;

Tani Gani Thaqi përsëritet dyherë, ndërsa me

SELECT DISTINCT Emri, Mbiemri FROM Klientet;

Të gjithë rreshtat janë të ndryshëm.

WHERE

Fjala kyçe *WHERE* na mundëson përcaktimin e rreshtave që dëshirojmë t'i shohim si rezultat i ekzekutimit të një pyetësori. Fjala kyçe *WHERE* është e përbërë nga një ose më shumë kushte që definojnë parametrat të urdhërit *SELECT*. Secili kusht është një shprehje që mund të jetë emër i një fushe, operator ose funksion. Sintaksa në vijim përshkruan se si është e definuar fjala kyçe *WHERE*:

WHERE < shprehja > [{ < operator > < shprehja > } …]

Siç mund të shihet fjala kyçe *WHERE* duhet të përmbajë së paku një kusht që definon rreshtat që do të paraqiten në rezultat. Kur të definojmë më shumë se një kusht atëherë ato kushte lidhen mes veti me operatorët logjikë *AND* ose *OR*. Tani le të shohim një shembull:

SELECT Numri, Tipi_i_kredise AS Kredia FROM Kredite
WHERE Tipi_i_kredise = 'Mobile';

Fjala kyçe *WHERE* kushtëzon që vetëm rreshtat në fushën Tipi_i_kredise me vlerën 'Mobile' të paraqiten si rezultat i këtij pyetësori.

SELECT Numri, Tipi_i_kredise AS Kredia FROM Kredite
WHERE Tipi_i_kredise='Veturë' AND Shuma>5000;

Këtu fjala kyçe *WHERE* përmban dy kushte të lidhura mes veti me operatorin *AND*. Kjo do të thotë që duhet të plotësohen dy kushtet në mënyrë që rreshti të paraqitet si rezultat. Kushti i parë kërkon që tipi i kredise të jetë veturë dhe çmimi i saj të jetë më i madh se 5000. Nga urdhëri i mësipërm mund të themi se pse nuk e shfrytëzuam pseudonimin Kredia për fushën Tipi_i_kredise te fjala kyçe *WHERE*. Kjo është për shkak se si MySQL i proceson urdhërat *SELECT* nuk mund të shfrytëzohen pseudonimet në fjalën kyçe *WHERE*. Një mundësi e tillë është e lejuar te fjala kyçe *HAVING*.

GROUP BY

Deri tani mësuam disa pjesë përbërëse të urdhërit *SELECT* të cilat kthenin si rezultat vlera nga fushat dhe rreshtat. Fjala kyçe *GROUP BY* ndryshon pak nga elementet tjera të urdhërit *SELECT* nga ajo se ai bën grupimin e vlerave. Ja edhe sintaksa:

GROUP BY <definimi i group by>

<definimi i group by>::=
<emri i fushës> [ASC | DESC]
[{, <emri i fushës> [ASC | DESC]}...]
[WITH ROLLUP]

Fjala kyçe *GROUP BY* përmban së paku një fushë edhe pse mund të përmbajë më shumë se një. Nëse janë zgjedhur më shumë se një fushë atëherë ato ndahen me presje në mes veti. Me përcaktimin e fjalës kyçe *GROUP BY* rreshtat grupohen sipas vlerave në fusha. Si rezultat grupimi bëhet për ato fusha që kanë vlera që përsëriten. P.SH. nuk mund të bëjmë grupimin për fushën që është çelës primar, sepse çdo vlerë në atë fushë është unike.

Shembuj:

*SELECT * FROM Klientet*
WHERE Qyteti='Prishtinë'
GROUP BY Emri;

Si rezultat do të paraqiten të gjithë klientët në qytetin e Prishtinës të sortuar sipas emrit. Sortimi mund të bëhet me renditje rritëse *ASC* dhe renditje zbritëse *DESC*, nëse nuk bëhet zgjedhja e asnjërës nga këto atëherë nënkuptohet vlera *ASC*.

SELECT Numri, COUNT(Shuma) AS 'Numri i kredive'
FROM Kredite
WHERE ID<10
GROUP BY Numri DESC;

Si rezultat paraqitet numri i kredive për secilin numër llogarie.

Fjala kyçe *HAVING*

Fjala kyçe *HAVING* është e ngjashme me *WHERE* në atë se përmban një ose më shumë kushte, të cilët definojnë rreshtat që do të paraqiten në rezultat. Megjithatë *HAVING* ka disa përparësi nga *WHERE*. P.SH. në *HAVING* mund të përfshijmë funksionet aggregate sikur funksioni COUNT( ). Pastaj mund të shfrytëzohen pseudonimet në *HAVING* të cilën gjë nuk mund t'a bëjmë te fjala kyçe *WHERE*.
Sintaksa e *HAVING*:

HAVING < shprehje > [{ < operator > < shprehje > } …]

Siç shihet nga sintaksa duhet të shkruhet së paku një shprehje. *HAVING* është e konstruktuar njësoj si *WHERE* në kuptimin e kushteve dhe lidhjen e atyre kushteve mes veti me operator.

SELECT Numri, Tipi_i_kredive, COUNT(Shuma) AS Nkredive FROM Kredite
WHERE ID<20
GROUP BY Numri
HAVING Nkredive<3;

Si rezultat do të paraqiten të gjithë numrat e llogarisë së klientëve që kanë marrë më pak se tri kredi.

Fjala kyçe *ORDER BY*

Në sintaksën e urdhërave *UPDATE* dhe *DELETE* është përmend edhe fjala kyçe *ORDER BY* e cila përdorej për sortimin e të dhënave që u modifikuan ose ato që u fshin. Poashtu edhe urdhëri *SELECT* përmban fjalën kyçe *ORDER BY* që na mundëson përcaktimin e renditjes së rreshtave në rezultat. Ja edhe sintaksa:

ORDER BY < definimi i order by >

< definimi i order by >::=
< emri i fushës > [ASC | DESC]
[{ , < emri i fushës > [ASC | DESC] } ...]

Siç shihet nga sintaksa këtu duhet të përfshihet përveq fjalës kyçe *ORDER BY* edhe së paku një emër i një fushe. Mund të shkruajmë edhe pseudonim të ndonjë fushe ose emrin e asaj fushe. Nëse dëshirojmë që të përfshihen më shumë se një fushë atëherë ato shkruhen duke i ndarë me presje. Për secilën fushë mund të shkruajmë mënyrën e sortimit: sipas renditjes rritëse *ASC* ose zbritëse *DESC*. Nëse nuk zgjedhet asnjëri nga opsionet atëherë nënkuptohet se është zgjedh mënyra e renditjes *ASC*.

Shembuj:

*SELECT * FROM Klientet*
ORDER BY Emri;

Si rezultat paraqiten të gjitha shënimet e klientëve nga tabela Klientet të sortuara sipas emrit me renditje rritëse. Në renditjet me më shumë se një fushë, është rregull që së pari renditja bëhet sipas fushës së parë dhe pastaj fushës së dytë.

*SELECT * FROM Klientet*
ORDER BY Emri,Qyteti;

Fjala kyçe *LIMIT*

Fjala kyçe *LIMIT* mund të merr dy argumente:

LIMIT [<fillimi>,] <numri rrestave>

Argumenti i parë është opsional dhe përcakton prej ku të fillojë numrimi i rreshtave. Nëse nuk është zgjedhur ndonjë vlerë atëherë nënkuptohet vlera 0. Argumenti i dytë me radhë *< numri i rreshtave >* në fjalën kyçe *LIMIT* tregon numrin e rreshtave që do të paraqiten në rezultat.

Shembuj:

*SELECT * FROM Kredite*
WHERE ID<20
ORDER BY ID
LIMIT 5;

Si rezultat paraqiten 5 të dhënat e fundit që plotësojnë kushtin *WHERE.*

*SELECT * FROM Kredite*
WHERE ID<20
ORDER BY ID
LIMIT 4, 5;

Si rezultat paraqiten 5 të dhëna duke filluar nga e dhëna e katërtë që plotësojnë kushtin *WHERE.*

Përdorimi i operatorëve në urdhërat SQL

Gjatë mësimit të SQL urdhërave kemi parë se në sintaksën e tyre janë përdorë një numër i shprehjeve. P.SH. kemi përdorë shprehjet te fjala kyçe WHERE, te urdhërat SELECT, UPDATE dhe DELETE. Shprehja është një formulë e përbërë nga emra të fushave, operatorëve dhe funksioneve. Këto elemente së bashku krijojnë shprehje më efektive për modifikimin e të dhënave në MySQL. Në mënyrë që çdo shprehje të jetë efikase shfrytëzohen operatorë të ndryshëm. Operatori është një symbol ose fjalë kyçe që përcakton një aksion të caktuar ose kushtet ndërmjet elementeve të shprehjes ose ndërmjet shprehjeve.

Prioriteti i operatorëve

Kur një shprehje të procesohet ajo përllogaritet sipas renditjes së elementeve të shprehjes dhe prioritetit të operatorëve. MySQL proceson shprehjet sipas një rregulle specifike të prioritetit të operatorëve. Kur kemi të bëjmë me shprehje komplekse që kanë shumë operatorë,MySQL duhet të përcaktoj se cili operator ka përparësi.Për atë MySQL e ka të paradefinuar prioritetin e kategorive të operatorëve sipas kësaj renditje:

- BINARY, COLLATE
- NOT (negacioni), ! (negacioni)
- - (minusi unar)
- * (shumëzimi), / (pjestimi), % (pjestimi me mbetje)
- - (zbritja), + (mbledhja)
- Të gjithë operatorët e krahasimit përveq BETËEEN dhe NOT BETËEEN
- BETWEEN, NOT BETWEEN
- AND, &&
- OR, | |, XOR

Operatorët aritmetik

Operatorët aritmetik përdoren për llogaritje në shprehje. Janë të ngjashëm me simbolet në algjebër mbledhja, zbritja, shumëzimi, pjestimi. Në tabelën e mëposhtme janë përshkrimet për secilin operatorë:

Operatori	Përshkrimi
+ (mbledhja)	Mbledh dy argumente
- (zbritja)	Zbret argumentin e dytë nga argumenti i parë
- (unar)	Ndërron shenjën e argumentit
* (shumëzimi)	Shumëzon dy numra
/ (pjestimi)	Pjeston argumentin e parë me argumentin e dytë
% (pjestimi me mbetje)	Pjeston argumentin e parë me argumentin e dytë dhe si rezultat jep mbetjen

Më herët kemi parë shembuj ku kemi përdorë operatorët aritmetik në urdhërat SQL.

SELECT Emertimi AS 'Emri i mallit', NeDepo+TeShitura AS Total
FROM Dyqani;

UPDATE Kreditë
SET Shuma=Shuma+10;

Operatorët e krahasimit

Operatorët e krahasimit përdoren për krahasimin ndërmjet dy argumenteve dhe përcaktimin e vlerës ***saktë, pasaktë*** ose ***NULL*** të atij kushti. Nëse njëri ose të dy argumentet janë *NULL* atëherë vlera e kushtit është *NULL*, përveq te operatori < = > i cili kthen vlerën e saktë kur dy argumentet janë të njëjtë. Që kushti të pranohet duhet të kthejë vlerë të saktë. P.SH. me anë të urdhërit *SELECT* i cili përmban fjaën kyçe *WHERE:*

*SELECT * FROM Klientet*
WHERE Qyteti='Prishtinë';

Fjala kyçe *WHERE* përmban shprehjen *Qyteti='Prishtinë'* . Kur të ekzekutohet urdhëri i mësipërm vlera në fushën *Qyteti* krahasohet me vlerën *Prishtinë*. Nëse vlera është *Prishtinë* atëherë kushti është i saktë. Nëse vlera nuk është *Prishtinë* kushti do të jetë i pasaktë. Nëse në fushën *Qyteti* kemi vlera *NULL* atëherë kushti është *NULL*. Si rezultat do të paraqiten vetëm rreshtat ku në fushën *Qyteti* është vlera *Prishtinë.*

MySQL përkrah një numër të madh të operatorëve të krahasimit, që mundësojnë përcaktimin e kushteve të ndryshme në urdhërat SQL. Tabela e mëposhtme përshkruan këta operatorë:

Operatori	Përshkrimi
=	Kthen vlerën e saktë nëse të dy argumentet janë të barabartë
< = >	Kthen vlerën e saktë nëse të dy argumentet janë të barabartë, edhe në rastin kur kanë vlerë NULL
<>, !=	Kthen vlerën e saktë nëse argumentet nuk janë të barabartë
<	Kthen vlerën e saktë nëse argumenti i parë është më i vogël se argumenti i dytë
<=	Kthen vlerën e saktë nëse argumenti i parë është më i vogël ose i barabartë me argumentin i dytë
>	Kthen vlerën e saktë nëse argumenti i parë është më i madh se argumenti i dytë
>=	Kthen vlerën e saktë nëse argumenti i parë është më i madh ose i barabartë mse argumentin i dytë
IS NULL	Kthen vlerën e saktë nëse argumenti ka vlerën NULL
IS NOT NULL	Kthen vlerën e saktë nëse argumenti është i ndryshëm nga NULL
BETWEEN	Kthen vlerën e saktë nëse vlera e argumentit gjendet në intervalin e parcaktuar nga fjala kyçe BETWEEN
NOT BETWEEN	Kthen vlerën e saktë nëse vlera e argumentit nuk gjendet në intervalin e parcaktuar nga fjala kyçe BETWEEN
IN	Kthen vlerën e saktë nëse vlera e argumentit është e parcaktuar në fjalën kyçe IN
NOT IN	Kthen vlerën e saktë nëse vlera e argumentit nuk është e parcaktuar në fjalën kyçe NOT IN
LIKE	Kthen vlerën e saktë nëse vlera e argumentit është e përcaktuar nga LIKE
NOT LIKE	Kthen vlerën e saktë nëse vlera e argumentit nuk është e përcaktuar nga LIKE
REGEXP	Kthen vlerën e saktë nëse vlera e argumentit është e përcaktuar me REGEXP
NOT REGEXP	Kthen vlerën e saktë nëse vlera e argumentit nuk është e përcaktuar me NOT REGEXP

Siç shihet ka një numër të madh operatorësh të krahasimit dhe mënyra më e mirë për kuptimin e tyre është me anë të shembujve:

```
SELECT ID, Numri, Tipi_i_kredise
FROM Kredite
WHERE Shuma=420
ORDER BY Tipi_i_kredise;
```

Siç shihet operatori i krahasimit (=) përcakton se si rezultat janë vetëm rreshtat te të cilët vlera në fushën Shuma është 420.

SELECT ID, Numri
FROM Kredite
WHERE Tipi_i_kredise < = > NULL

Nga ky shembull si rezultat kthehen vetëm ata rreshta që në fushën *Tipi_i_kredise* kanë vlerën *NULL.*

SELECT ID, Numri
FROM Kredite
WHERE Tipi_i_kredise IS NOT NULL;

Nga ky shembull si rezultat kthehen vetëm ata rreshta që në fushën *Tipi_i_kredise* kanë vlerë të ndryshme nga *NULL.*

SELECT ID, Numri, Tipi_i_kredise
FROM Kredite
WHERE Shuma>1000
ORDER BY Tipi_i_kredise;

Si rezultat kthehen të gjitha të dhënat që në fushën *Shuma* kanë vlerën më të madhe se 1000. Të renditura sipas *Tipi_i_kredise.*

SELECT ID, Numri, Tipi_i_kredise
FROM Kredite
WHERE Shuma BETWEEN 420 AND 1800
ORDER BY Tipi_i_kredise;

Si rezultat kthehen të gjitha të dhënat që vlera në fushën *Shuma* është ndërmjet vlerës 420 dhe 1800. Të renditura sipas *Tipi_i_kredise*.

SELECT ID, Numri, Tipi_i_kredise
FROM Kredite
WHERE Shuma NOT BETWEEN 420 AND 1800
ORDER BY Tipi_i_kredise;

Si rezultat kthehen të gjitha të dhënat që vlera në fushën *Shuma* është më e vogël se 420 dhe më e madhe se 1800. Të renditura sipas *Tipi_i_kredise.*

SELECT Emri, Mbiemri, Adresa
FROM Klientet
WHERE QYTETI IN ('Prishtinë','Besianë')
ORDER BY Emri;

Nga ky shembull fitojmë si rezultat të dhënat që në fushën *Qyteti* përmbajnë vlerën *'Prishtinë'* ose *'Besianë'*. Pra lista e vlerave e cila është brenda kllapave të vogla të ndara

mes veti me presje. Kushti merr vlerën e saktë vetëm për vlerat brenda kllapave të vogla, të dhënat që përmbajnë vlera tjera ose *NULL* në fushën *Qyteti* nuk do të paraqiten në rezultat. Lista e vlerave Brenda kllapave të vogla në këtë shembull janë të future në thonjëza sepse janë të tipit sting, në rastin kur vlerat janë të tipit numeric atëherë vlerat nuk futen në thonjëza.

SELECT Numri,Tipi_i_kredise
FROM Kredite
WHERE Shuma IN (420,7000,390);

Një tjetër operator i krahasimit mjaft i dobishëm është operatori *LIKE*, i cili mundëson kërkimin e vlerave të ngjashme me vlerën e caktuar. Operatori *LIKE* përkrah dy karakterë gjithëpërfshirës:

- **%** (përqindja) përfaqëson zero ose një numër të vlerave
- _ (underline) përfaqëson saktësisht një vlerë.

SELECT Numri, Emri, Mbiemri
FROM Klientet
WHERE Qyteti LIKE 'P%'
ORDER BY Emri;

Si rezultat paraqiten të gjitha të dhënat që në fushën *Qyteti* kanë vlera që fillojnë me shkronjën *P* (të madhe). Të renditura sipas fushës *Emri*.

SELECT Numri, Emri, Mbiemri
FROM Klientet
WHERE Qyteti LIKE '%P%'
ORDER BY Emri;

Rezultati i këtij shembulli është i njëjtë me atë të shembullit të mëparshëm. Kjo është ngase % mund të përfaqësojë zero ose një numër të vlerave.

SELECT Numri, Emri, Mbiemri
FROM Klientet
WHERE Qyteti LIKE '_r%n'
ORDER BY Emri;

Si rezultat fitojmë të gjitha të dhënat që në fushën Qyteti kanë vlerat: shkronja e dytë duhet të jetë *r* dhe duhet të përfundojë me shkronjën *n.* Sepse _ (underline) pëfaqëson saktësisht një vlerë, pra shkronja e parë mund të jetë çfarëdo. Nuk duhet të harrojmë që vlerat në fusha janë *CASE SENSITIV.*

MySQL përkrah edhe një operator që mundëson gjetjen e vlerave të ngjashme me vlerën e caktuar. Operatori REGEXP mundëson një numër të ndryshëm të opsioneve dhe trajtash në mënyrë të gjetjes së vlerave të ngjashme me vlerën e caktuar. Tabela e

mëposhtme liston një numër të opsioneve që mund t'i përdorim me operatorin REGEXP në krijimin e SQL urdhërave.

Opsioni	Përshkrimi	Shembull	Vlerat e pranueshme
< vlera >	Vlera e testuar duhet të përmbajë vlerën e caktuar	'bo'	bota, boshe, karboni
< ^ >	Vlera e testuar nuk duhet të përmbajë vlerën e caktuar	'^bo'	shuma, emri, mbiemri
.	Vlera e testuar mund të përmbajë njërin nga karakterët	'b.'	mbiemri, bora, baza
[<karakterët>]	Vlera e testuar përmban së paku një nga karakterët e listuar brenda kllapave të mesme	'[df]'	dritarja, formulari, mundësia
[<intervali>]	Vlera e testuar duhet të përmbajë së paku njërin nga karakterët e listuar në intervalin në kllapat e mësme	'[1-5]'	15,3,346,50,22,791
^	Vlera e testuar duhet të fillojë me vlerën pas shenjës ^	'^s'	start, supë, sqarim
$	Vlera e testuar duhet të përfundojë me vlerën para shenjës $	'ar$'	sqaruar, drejtuar,formular
*	Vlera e testuar duhet të përmbajë zero ose më shumë karakterë për shenjën *	'^s.*i$'	sqarimi, startimi, syri

Operatori REGEXP mund të shihet si jo i përshtatshëm por pas shembujve që do t'i punojmë do të kemi të qartë rëndësinë dhe dobinë e këtij operatori.

SELECT Emri, Mbiemri, Adresa
FROM Klientet
WHERE Emri REGEXP '^[a-f]'
ORDER BY Emri;

Shprehja *Emri REGEXP '^[a-f]'* së pari përcakton fushën *Emri*, fjalën kyçe *REGEXP* dhe vlerën që duhet të plotësohet. Vlera është brenda kuotave (thonjëzave të njëfishta) dhe përmban shenjën ^ dhe intrevalin që përcakton shkronjat prej ***a*** deri në ***f***. Pasi që është përdorë shenja ^ atëherë vlera e ardhshme e përcaktuar duhet të paraqitet në fillim. Në rastin tonë është intervali brenda kllapave të mesme. Pra që kushti të merr vlerën e saktë për një të dhënë duhet që në fushën *Emri* vlera të fillojë me shkronjat nga ***a*** deri në ***f***.

SELECT Emri, Mbiemri, Adresa
FROM Klientet
WHERE Emri REGEXP '^[mn].[ai]$'*
ORDER BY Emri;

Në këtë urdhër sërish *REGEXP* fillon me shenjën ^ duke treguar që vlera e ardhshme duhet të paraqitet në fillim të vlerës së fushës *Emri*. Por në këtë rast nuk kemi interval por dy karakterë ***m*** dhe ***n***. Si rezultat vlera në fushën *Emri* duhet të fillojë me ***m*** ose ***n***. Pastaj është pika (**.**) dhe ylli (*****). Pika tregon se vetëm një karakter mund të përfshihet dhe ylli tregon se mund të përfshihen zero ose më shumë karakterë të përsëritur. Karakterët mund të përsëriten. Pastaj *[ai]$* tregon se vlera duhet të përfundojë me karakterët ***a*** ose ***i***.

Operatorët logjikë

Operatorët logjikë mundësojnë testimin e vlefshmërisë së shprehjeve. Për një kusht ose disa kushte që të jenë të pranueshme duhet të marrin vlerë të saktë. Tabela e mëposhtme përshkruan operatorët logjikë në MySQL:

Operatori	Përshkrimi
AND	Kthen vlerën e saktë nëse të dy argumentet kanë vlerë të saktë. Poashtu mund të shfrytëzojmë edhe && në vend të AND
OR	Kthen vlerën e saktë nëse njëri nga argumentet është i saktë. Mund të shfrytëzojmë || në vend të OR
XOR	Kthen vlerën e saktë nëse saktësisht njëri nga argumentet është i saktë
NOT, !	Kthen vlerën e saktë nëse argumenti ose shprehja ka vlerë të pasaktë. Mund të shfrytëzojmë ! në vend të operatorit NOT

```
SELECT ID, Tipi_i_kredise
FROM Kredite
WHERE Tipi_i_kredise='Veturë' AND Shuma>3500
ORDER BY Shuma;
```

Shprehja e parë te fjala kyçe *ËHERE* përcakton që fusha *Tipi_i_kredise* duhet të ketë vlerën *Veturë*, shprehja e dytë kërkon që vlera në fushën *Shuma* të jetë më e madhe se 3500. Këto dy shprehje janë të lidhura mes veti me operatorin logjik *AND*, d.m.th. si rezultat të dy shprehjet duhet të marrin vlerë të saktë në mënyrë që i tërë kushti të merr vlerë të saktë. Me fjalë të tjera të gjitha të dhënat në rezultat te fusha *Tipi_i_kredise* do të kenë vlerën *Veturë* dhe në fushën *Shuma* vlera do të jetë më e madhe se 3500.

```
SELECT ID, Tipi_i_kredise
FROM Kredite
WHERE Tipi_i_kredise='Veturë' OR Shuma>3500
ORDER BY Shuma;
```

Shprehja e parë te fjala kyçe *WHERE* përcakton që fusha *Tipi_i_kredise* duhet të ketë vlerën *Veturë*, shprehja e dytë kërkon që vlera në fushën *Shuma* të jetë më e madhe se 3500. Këto dy shprehje janë të lidhura mes veti me operatorin logjik *OR*, d.m.th. si rezultat njëra nga dy shprehjet duhet të merr vlerë të saktë në mënyrë që i tërë kushti të

merr vlerë të saktë. Me fjalë të tjera të gjitha të dhënat në rezultat te fusha *Tipi_i_kredise* do të kenë vlerën *Veturë* ose në fushën *Shuma* vlera do të jetë më e madhe se 3500.

SELECT Emri, Mbiemri
FROM Klientet
WHERE Emri='Amir' XOR Qyteti='Prishtinë';

Shprehja e parë te fjala kyçe *ËHERE* përcakton që fusha *Emri* duhet të ketë vlerën *Amir*, shprehja e dytë kërkon që vlera në fushën *Qyteti* të jetë vlera *Prishtinë*. Këto dy shprehje janë të lidhura mes veti me operatorin logjik *XOR*, d.m.th. si rezultat saktësisht njëra nga dy shprehjet duhet të merr vlerë të saktë në mënyrë që i tërë kushti të merr vlerë të saktë. Me fjalë të tjera si rezultat do të paraqiten të gjitha të dhënat që në fushën *Emri* kanë vlerën *Amir* dhe te fusha *Qyteti* kanë vlerë të ndryshme nga *Prishtinë*, ose te fusha *Emri* kanë vlerë të ndryshme nga vlera *Amir* dhe te fusha *Qyteti* kanë vlerë *Prishtinë*.

Mund të ndërtojmë edhe shprehje më të komplikuara, ku përfshihen më shumë se një operator logjik, sin ë shembullin e mëposhtëm:

SELECT Emri, Mbiemri
FROM Klientet
WHERE Emri='Lavdim' AND (Qyteti IS NULL OR NOT (Qyteti='Prishtinë'))
ORDER BY Emri;

Shprehja e parë te fjala kyçe *ËHERE* përcakton që fusha *Emri* duhet të ketë vlerën *Lavdim*, dy shprehjet tjera janë të lidhura me operatorin logjik *OR*. E para nga këto dy shprehje kërkon që fusha *Qyteti* të ketë vlerë *NULL*, ndërsa shprehja tjetër kërkon që vlera në fushën *Qyteti* të mos jetë *Prishtinë*. Pasi që këto dy shprehje janë të lidhura me operatorin logjik *OR* atëherë që të kthehet vlerë e saktë duhet që njëra nga shprehjet të merr vlerë të saktë. Ndërsa që e tërë shprehja të merr vlerën e saktë duhet që vlera në fushën *Emri* të jetë *Lavdim*.

Opertaorët logjik mund t'i përdorim edhe në urdhërat *UPDATE* dhe *DELETE* për definimin e rreshtave që do të modifikohen.

.

Operatorët e sortimit

Tip tjetër i operatorëve janë operatorëte sortimit , të cilët përdoren për definimin e kushtit për krahasim me vlerat nëpër fusha. Në tabelën e mëposhtme janë operatorët logjik:

Operatori	Përshkrimi
BINARY	Konverton një string në string binar ashtu që sortimi i të dhënave do të jetë CASE SENSITIV
COLLATE	Përcakton një sistemim të veçantë që përdoret për krahasimin dhe sortimin e të dhënave

Mënyra më e mirë për të kuptuar secilin nga këta operatorë është përmes shembujve:

CREATE TABLE Produkti
(
ProdID SMALLINT NOT NULL PRIMARY KEY,
ProdColor VARCHAR(15) NOT NULL
);

INSERT INTO Produkti
VALUES (101, 'Red'), (102, 'red'), (103, 'RED'), (104, 'REd'), (105, 'reD'),
(106, 'Blue'), (107, 'blue'), (108, 'BLUE'), (109, 'BLue'), (110, 'blUE');

Para se të shohim operatorin BINARY shohim një shembull:

*SELECT * FROM Produkti*
WHERE ProdColor='red';

Do të kthehen si rezultat të gjitha të dhënat që në fushën ProdColor kanë vlerën *red*

```
+--------+-------------+
| ProdID | ProdColor   |
+--------+-------------+
|   101  |    Red      |
|   102  |    red      |
|   103  |    RED      |
|   104  |    Red      |
|   105  |    reD      |
+--------+-------------+
```
5 roës in set (0.00 sec)

*SELECT * FROM Produkti*
ËHERE ProdColor=BINARY 'red';

```
+--------+-------------+
| ProdID | ProdColor   |
+--------+-------------+
|  102   |    red      |
+--------+-------------+
```
1 roë in set (0.01 sec)

Siç shihet rezultati përfshinë në fushën ProdColor saktësisht vlerën red, pra kërkimi është bërë CASE SENSITIV.

*SELECT * FROM ProductColors*
ËHERE ProdColor COLLATE latin1_german2_ci = 'red';

Nga shembulli më lartë shohim se është përdorë sistemi i krahasimit *latin1_german2_ci.*

Lidhja e tabelave në urdhërin SELECT

Më herët mësuam përdorimin e urdhërit SELECT për shikimin e të dhënave nga ndonjë bazë e të dhënave. Ku kemi ekstraktuar rreshta dhe fusha të caktuara, të grupuara sipas nevojës. Këto të dhëna i kemi nxjerr nga një tabelë, por MySQL mundëson shikimin e të dhënave nga shumë tabela dhe kthen si rezultat të dhëna sikur nga një tabelë. Ç'është e vërtetë ne mund të qasemi shumë tabelave edhe me urdhërat UPDATE dhe DELETE. Do të mësojmë si të krijojmë lidhjet ndërmjet dy apo më shumë tabelave dhe shumë nga elementet që përdoren te urdhëri SELECT janë të ngjashme me ato të urdhërat UPDATE dhe DELETE.
Me krijimin e një urdhëri SELECT dëshirojmë që rezultati të mirret nga të dhënat që ruhen në tabela të ndryshme. Në rezultat nuk duhet të ketë të dhëna që paraqiten arbitarisht (pa arsye), me fjalë të tjerë të dhënat duhen të paraqiten sikur të ishin nga një tabelë. Të dhënat duhet të jenë të harmonizuara dhe logjike pavarësisht faktit se ato janë nga tabela të ndryshme.
Për të arritur këtë harmonizim MySQL mundëson lidhjet te urdhëri SELECT. Ja edhe sintaksa:

SELECT
[<opsioni select> [<opsioni select>...]]
{ | <lista select>}*
[
FROM {<tabela referuese> | <definimi i lidhjes>}
[WHERE <shprehje> [{<operator> <shprehje>}...]]
[GROUP BY <definimi i group by>]
[HAVING <shprehje> [{<operator> <shprehje>}...]]
[ORDER BY <definimi i order by>]
[LIMIT [<fillimi>,] <numri i rreshtave>]
]

<definimi i lidhjes>::=
{<tabela referuese>, <tabela referuese> [{, <tabela referuese>}...]}
| {<tabela referuese> [INNER | CROSS] JOIN <tabela referuese> [<kushtet e lidhjes>]}
| {< tabela referuese > STRAIGHT_JOIN < tabela referuese >}
| {< tabela referuese > LEFT [OUTER] JOIN < tabela referuese > [<kushtet e lidhjes >]}
| {<tabela referuese> RIGHT [OUTER] JOIN < tabela referuese > [<kushtet e lidhjes >]}
| {<tabela referuese> NATURAL [{LEFT | RIGHT} [OUTER]] JOIN<tabela referuese>}

< tabela referuese >::=
<emri i tabelës> [[AS] <pseudonimi>]
[{USE | IGNORE | FORCE} INDEX <emri i indeksit> [{, <emri i indeksit>}...]]

<kushtet e lidhjes>::=
ON <shprehje> [{<operator> <shprehje>}...]
| USING (<fushë> [{, <fushë>}...])

Siç mund të shihet nga sintaksa fjala kyçe FROM është:

FROM {<tabela referuese> | <definimi i lidhjes>}

Ndërsa nga sintaksa e urdhërit SELECT më herët te fjala kyçe FROM ka qenë më ndryshe:

FROM <tabela referuese> [{ , <tabela referuese> } …]

Kjo është bërë për paraqitje më të shkurtër të sintaks e lidhjes ës së urdhërit SELECT. Pra tani fjala kyçe FROM *përmban edhe definimin*. Definimi i lidhjes i referohet tipeve të ndryshme të lidhjes, si mëposhtë:

<definimi i lidhjes>::=
{<tabela referuese>, <tabela referuese> [{, <tabela referuese>}...]}
| {<tabela referuese> [INNER | CROSS] JOIN <tabela referuese> [<kushtet e lidhjes>]}
| {< tabela referuese > STRAIGHT_JOIN < tabela referuese >}
| {< tabela referuese > LEFT [OUTER] JOIN < tabela referuese > [<kushtet e lidhjes >]}
| {<tabela referuese> RIGHT [OUTER] JOIN < tabela referuese > [<kushtet e lidhjes >]}
| {<tabela referuese> NATURAL [{LEFT | RIGHT} [OUTER]] JOIN<tabela referuese>}

E para nga këto lidhje është lidhja themelore (bazike) e cila përbëhet vetëm nga tabelat referuese të ndara mes veti me presje. Lidhjet tjera janë INNER, CROSS, STRAIGHT_JOIN, LEFT, RIGHT dhe NATURAL.

Krijimi i lidhjes themelore

Në disa nga shembujt deri tani kemi parë përdorimin e urdhërit SELECT ku te tabela referuese kemi zgjedhur vetëm një tabelë. Te lidhja themelore tani në vend të një tabele te fjala kyçe FROM përdorim më shumë se një tabelë (siç shihet edhe nga sintaksa).
Këtë më mirë do t'a kuptojmë me një shembull:

SELECT Emri, Mbiemri, Shuma
FROM Klientet, Kredite;
Ky urdhër është i ngjashëm me urdhërat SELECT të mësuar më herët, dallimi është te fjala kyçe FROM e cila përmban dy tabela: Klientet dhe Kredite. Te urdhëi SELECT kemi zgjedh dy fushat e para nga tabela Klientet dhe fushën Shuma nga tabela Kredite.

Nga ky shembull lidhja është definuar në dy tabela por vetë lidhja nuk është e kufizuar. Fushat e zgjedhura në fjalën kyçe SELECT nuk kanë ndonjë efekt se si do të definohet vetë lidhja. Kur një lidhje nuk është e kufizuar në çdo mënyrë kthen si rezultat çdo rresht të një tabele me çdo rresht të tabelës tjetër. Ky tip i rezultatit quhet Prodhimi Kartezian. Nga ky shembull tabela e parë ka 7 të dhëna, ndërsa tabela e dytë ka 9 të dhëna, dhe si rezultat paraqiten 7x9 të dhëna pra 63 të dhëna. Shfrytëzimi i lidhjes për kthimin e rezultatit si prodhim kartezian nuk është e dobishme. Kjo është shkaku se nëse bëjmë lidhjen e tri tabelave që kanë nga 100 të dhëna atëherë prodhimi kartezian i tyre do të jetë një million të dhëna. Është e dobishme që me përdorimin e lidhjes themelore të shtohen kushtet e nevojshme në fjalën kyçe ËHERE në mënyrë që të kufizohet lidhja. P.SH. urdhëri i mëposhtëm është i ngjashëm me atë më parë, me përjashtim të fjalës kyçe WHERE:

SELECT Emri, Mbiemri, Shuma
FROM Klientet, Kredite
WHERE Klientet.Numri=Kredite.Numri;

Tani me fjalën WHERE kemi vënë një kusht që definon kufizimin e lidhjes. Pra të dënat që do të paraqiten si rezultat i ekzekutimit të këtij urdhëri janë : të gjitha të dhënat ku vlera në fushën Numri është e barabartë në të dy tabelat. Në shembullin tonë do të kthehen 9 të dhëna. Mund të përdorim edhe pseudonimet si mëposhtë:

SELECT Emri, Mbiemri, Shuma
FROM Klientet AS A, Kredite AS B
WHERE A.Numri=B.Numri;

Te fjala kyçe ËHERE mund të vejmë edhe kushte tjera, operatorë logjik, shprehje që kufizojnë lidhjen. P.SH. urdhëri i mëposhtëm kthen si rezultat vetëm ato të dhëna që në fushën Shuma kanë vlerë më të madhe se 1400.

SELECT Emri, Mbiemri, Shuma
FROM Klientet, Kredite
WHERE Klientet.Numri=Kredite.Numri AND Shuma >1400;

Tani si rezultat kthehen vetëm 4 të dhëna që plotësojnë kushtet (Shembujt gjenden në ftp://comp-253 → BAZAT E TË DHËNAVE).

Lidhja INNER dhe CROSS

Kur është fjala te lidhjet MySQL është i pajisur me mundësi të shumta që japin të njëjtin rezultat. Për shembull mund të krijojmë INNER dhe CROSS lidhje që japin të njëjtin rezultat si lidhja themelore. Shohim sintaksën për INNER dhe CROSS lidhjen:

{<tabela referuese> [INNER | CROSS] JOIN <tabela referuese> [<kushtet e lidhjes>]}

<kushtet e lidhjes>::=
ON <shprehje> [{<operator> <shprehje>}...]
| USING (<fushë> [{, <fushë>}...])

Siç shihet nga sintaksa duhet të shkruajmë tabelën referuese pastaj opsionet INNER ose CROSS pastaj fjalën kyçe JOIN dhe tabelën e dytë referuese dhe në fund opsioni ndonjë kusht. Ky kusht përmban fjalët kyçe ON dhe USING. Me anë të këtyre fjaëve kyçe defnojmë kushtin. Kushti te lidhja themelore definohej me fjalën kyçe ËHERE, ndërsa te INNER dhe CROSS lidhjet definohet me anë të < kushtet e lidhjes >.
Në përgjithësi caktimi i fjalëve INNER dhe CROSS jep rezultat të njëjtë sikur të mos ishin shkruar fare. Kur nuk është shkruar as INNER dhe CROSS atëherë nënkuptohet të jetë lidhje CROSS. Nëse nuk vejmë <kushtet e lidhjes> atëherë rezultati i fituar do të jetë i njëjtë me atë të lidhjes themelore, pra prodhim kartezian. Ja edhe disa shembuj:

SELECT Emri, Mbiemri, Shuma FROM Klientet, Kredite;

SELECT Emri, Mbiemri, Shuma FROM Klientet JOIN Kredite;

SELECT Emri, Mbiemri, Shuma FROM Klientet INNER JOIN Kredite;

SELECT Emri, Mbiemri, Shuma FROM Klientet CROSS JOIN Kredite;

Secili urdhër kthen si rezultat 63 të dhëna (rreshta), pra si prodhimi kartezian.
Për kufizimin e lidhjes *INNER* dhe *CROSS* e bëjmë me anë të fjalëve *ON* ose *USING*. Ja edhe një shembull me fjalën *ON*:

SELECT Emri, Mbiemri, Shuma
FROM Klientet AS A JOIN Kredite AS B
ON A.Numri=B.Numri;
Pra tani me fjalën *FROM* definuam lidhjen. Më mirë së ndarjen e tabelave të lidhura me presje përdorëm fjalën kyçe *JOIN*. Pastaj me anë të fjalës *ON* caktuam se vlerat në fushat Numri te të dy tabelat duhet të jetë i barabartë në mënyrë që të paraqiten në rezultat. Me definimin e kushtit te fjala kyçe *ON* rezultati nga shembulli kufizohet në 9 të dhëna.
Mund të fitojmë të njëjtin rezultat edhe me fjalën kyçe *USING* për kufizimin e lidhjes.

SELECT Emri, Mbiemri, Shuma
FROM Klientet AS A JOIN Kredite AS B
USING(Numri);

Siç mund të shihet te fjala kyçe *USING* është e nevojshme caktimi i fushave të lidhura (në kllapa të vogla). Kjo mund të shfrytëzohet vetëm kur fushat e lidhura në të dy tabelat kanë emër të njëjtë. Nëse bëjmë lidhjen në më shumë se një fushë atëherë ato shkruhen të ndara me presje mes veti.
Njësoj sit e lidhja themelore edhe këtu mund të kufizojmë lidhjen me anë të kushteve *WHERE* si mëposhtë:

SELECT Emri, Mbiemri, Shuma
FROM Klientet JOIN Kredite
ON Klientet.Numri=Kredite.Numri
WHERE Shuma >1400;

Kur të bëjmë definimin e lidhjeve *INNER* dhe *CROSS* duhet të përcaktojmë fushat e lidhura duke shfrytëzuar fjalët kyçe *ON* ose *USING* dhe kushte tjera te fjala kyçe *WHERE*. Poashtu mund të vejmë kushte te *WHERE* sit e lidhja themelore duke mos shfrytëzuar fjalët kyçe *ON* ose *USING*. Por urdhërat janë më lehtë të lexueshëm nëse përdorim fjalët kyçe *ON* ose *USING*.

Lidhja STRAIGHT_JOIN

Një tip tjetër i lidhjes është lidhja direkte (straight join) e cila është e ngjashme me lidhjen themelore në disa aspekte. Dallimi kryesor është se lidhja direkte mundëson zgjedhjen e optimizuesit të lidhjes i cili së pari bën leximin e tabelës në të majtë e pastaj asaj në të djathtë (Optimizuesi i lidhjes është një komponentë e MySQL serverit i cili mundohet të gjejë mënyrën më të mirë për procesimin e lidhjes). Pra lidhja direkte shfrytëzohet atëherë kur mendojmë se optimizuesi i lidhjes nuk është duke bërë mirë procesimin e e urdhërit *SELECT*. Në të shumtën e rasteve duhet të mbështetemi te optimizuesi i lidhjes.
Për krijimin e një lidhje direkte shfrytëzojmë sintaksën:

<tabela referuese> STRAIGHT_JOIN <tabela referuese>

Pra nga sintaksa shihet që së pari duhet të caktohet tabela referuese e përcjellur nga fjala kyçe STRAIGHT_JOIN pastaj nga një tabelë tjetër referuese.
Shembull:

SELECT Emri, Mbiemri, Shuma
FROM Klientet STRAIGHT_JOIN Kredite
WHERE Klientet.Numri=Kredite.Numri AND Shuma>1400;

Pas ekzekutimit të këtij urdhëri marrim rezultat të njëjtë sit e lidhjet e tjera. Një gjë vërehet, te lidhja direkte sikur edhe te lidhja themelore lidhja është kufizuar me fjalën kyçe *WHERE* e jo me fjalët kyçe *ON* ose *USING*. Në fakt i vetmi dallim ndërmjet lidhjes direkte me atë themelore është në sintaksë, te lidhja direkte shfrytëzohet fjala kyçe *STRAIGHT_JOIN* për ndarjen e tabelave referuese ndërsa te lidhja themelore kjo gjë bëhet me anë të presjes.

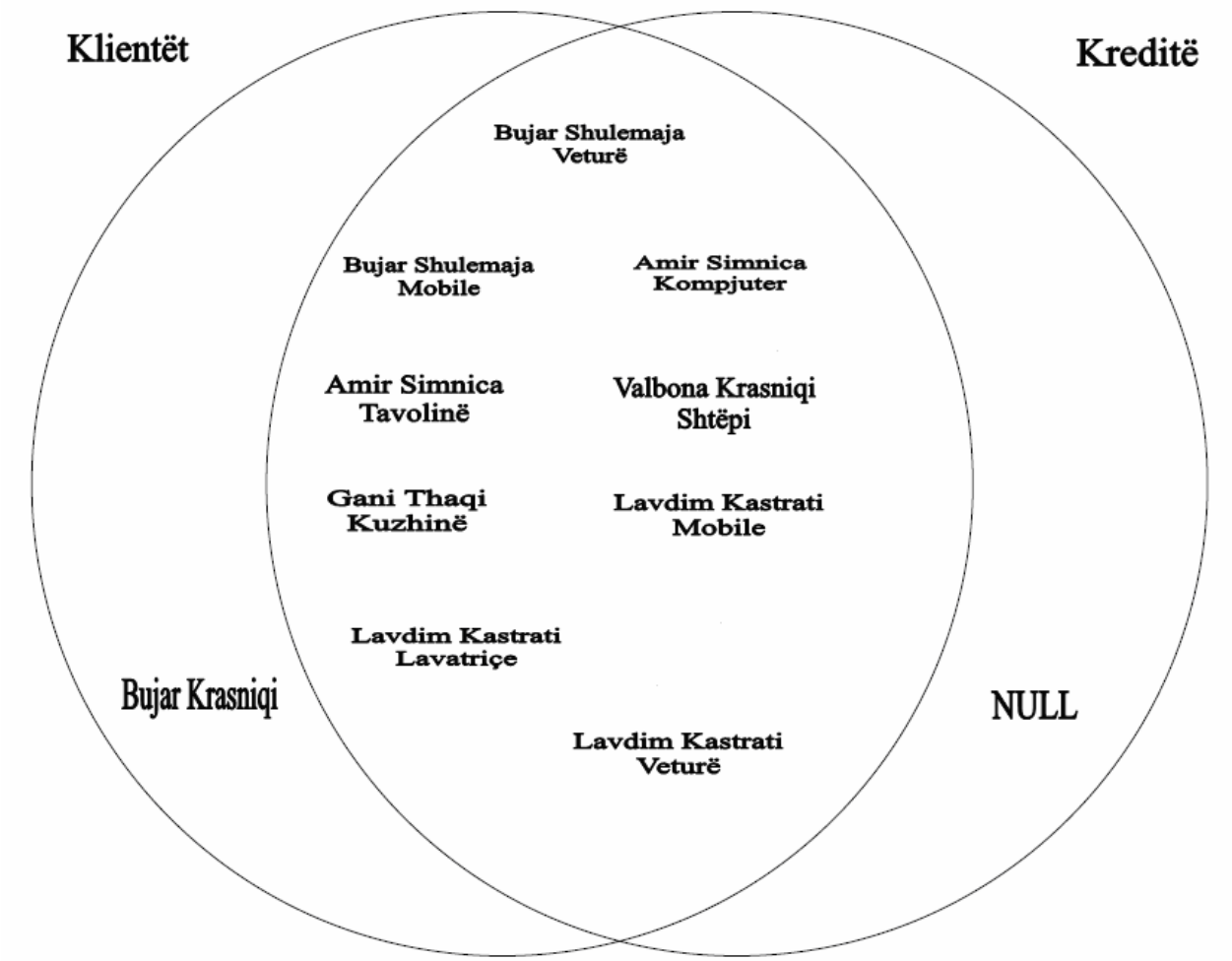
MySQL ofron edhe një metodë për krijimin e lidhjes direkte. Në vend të përdorimit të lidhjes direkte në fjalën kyçe *FROM* përdorim sintaksën e lidhjes direkte dhe caktojmë opsionin *STRAIGHT_JOIN* te fjala kyçe *SELECT* sin ë shembullin e mëposhtëm:

```
SELECT STRAIGHT_JOIN Emri, Mbiemri, Shuma
FROM Klientet, Kredite
WHERE Klientet.Numri=Kredite.Numri AND Shuma>1400;
```

Shihet se te fjala kyçe *FROM* tabelat referuese janë të ndara me presje që është tipike për lidhjen themelore. Por urdhëri *SELECT* përmban opsionin *STRAIGHT_JOIN*. Ekzekutimi i këtij urdhëri jep rezultat të njëjtë me atë më parë.

Krijimi i lidhjeve të jashtme (Outer joins)

Përveq lidhjeve që kthehin si rezultat prodhimin kartezian, lidhjet tjera që i pamë deri tani ishin ato që kthehin si rezultat vetëm të dhënat që përmbanin vlera të njëjta në fusha të caktuara. Supozojmë që dëshirojmë të shohim të dhëna që nuk i plotësojnë këto kushte, por jo që rezultati të mos jetë numër i madh i rreshtave. P.SH. më herët shikuam të dhëna nga tabela *Klientët* dhe tabela *Kreditë*, ku kemi kërkuar që si rezultat të paraqiten vetëm të dhënat që plotësojnë kushtin që: fusha Numri në tabelën *Klientet* dhe fusha *Numri* në tabelën *Kreditë* të jenë të barabartë. Por nëse tabela *Klientët* do të përmbante ndonjë klientë që nuk ka marrë asnjë kredi.
MySQL ofron mundësinë e shikimit të këtyre klientëve duke përdorë lidhjet e jashtme (outer joins). Mund të krijojmë dy tipe të lidhjeve të jashtme: lidhja e majtë dhe lidhja e djathtë. Lidhja e majtë nxjerr të gjitha të dhënat nga tabela e majtë, ndërsa lidhja e djathtë nxjerr të gjitha të dhënat nga tabela e djathtë. Figura e mëposhtme tregon diagramin e Venn-it për lidhjen e majtë. Rrethi në të majtë është tabela Klientët dhe rrethi në të djathtë paraqet tabelën Kreditë. Diagrami i Venn-it ilustron lidhjen e këtyre dy tabelave. Vlerat në të majtë janë nga tabela Klientët ndërsa vlerat në të djathtë janë nga tabela Kreditë.Në lidhjen *[INNER|CROSS] JOIN* si rezultat janë paraqitur vetëm të dhënat që kanë plotësuar kushtin Klientet.Numri=Kredite.Numri, këto të dhëna janë paraqitur në prerjen e dy rrathëve. Siç mund të shihet çdo klienti i korrespondon një kredi ose më shumë kredi. Por në figurë shihet se klienti Bujar Krasniqi nuk është në rezultat për shkak të lidhjes *JOIN*. Pasi që kjo është lidhje e jashtme e majtë (left outer join) klienti Bujar Krasniqi do të jetë në rezultat por vlera që korrespondon është *NULL*.



Lidhja e jashtme e majtë (Left outer join)

Sintaksa për lidhjen e jashtme të majtë është e njëjtë me atë të *INNER* dhe *CROSS JOIN*, vetëm se duhet të përfshihet fjala kyçe *LEFT*, siç është treguar në sintaksën e mëposhtme:

<tabela referuese> LEFT [OUTER] JOIN <tabela referuese> [<kushtet e lidhjes>]

<kushtet e lidhjes>::=
ON <shprehje> [{<operator> <shprehje>}...]
| USING (<fushë> [{, <fushë>}...])

Pra së pari specifikojmë tabelën referuese të përcjellur nga fjala kyçe *LEFT* dhe opsioni *OUTER* pastaj fjala kyçe *JOIN*. Kjo pastaj përcjellet nga një tabelë tjetër referuese dhe kushtet e lidhjes, të cilat mund të jenë fjalët kyçe *ON* dhe *USING*.
Shembull: Urdhëri i mëposhtëm definon lidhjen e jashtme të majtë në tabelat Klientët dhe Kreditë.

SELECT Emri, Mbiemri, Tipi_i_Kredise
FROM Klientet LEFT JOIN Kredite
ON Klientet.Numri=Kredite.Numri
ORDER BY Emri;

Mund të shihet që fjala kyçe *FROM* së pari i referohet tabelës Klientet dhe pastaj i referohet tabelës Kredite. Si rezultat paraqiten të gjitha të dhënat nga tabela Klientet (tabela e majtë) dhe vetëm të dhënat nga tabela Kredite (tabela e djathtë) që e plotësojnë kushtin e lidhjes te fjala kyçe *ON*. Sikur në tabelën e mëposhtme

```
mysql> SELECT Emri, Mbiemri, Tipi_i_kredise
    -> FROM Klientet LEFT JOIN Kredite
    -> ON Klientet.Numri=Kredite.Numri
    -> ORDER BY Emri;
+---------+-----------+-----------------+
| Emri    | Mbiemri   | Tipi_i_kredise  |
+---------+-----------+-----------------+
| Amir    | Simnica   | Kompjuter       |
| Amir    | Simnica   | Tavolinw        |
| Bujar   | Shulemaja | Veturw          |
| Bujar   | Shulemaja | Mobile          |
| Bujar   | Krasniqi  | NULL            |
| Gani    | Thaqi     | Kuzhinw         |
| Lavdim  | Kastrati  | Mobile          |
| Lavdim  | Kastrati  | Lavatriçe       |
| Lavdim  | Kastrati  | Veturw          |
| Valbona | Krasniqi  | Shtwpi          |
+---------+-----------+-----------------+
10 rows in set (1.09 sec)
```

Nëse i referohemi rreshtit të pestë shohim se është kthyer vlera *NULL* për fushën Tipi_i_kredise. Është kthyer vlera *NULL* sepse klienti Bujar Krasniqi nuk ka marrë asnjë kredi. Pra nëse vlera në tabelën e majtë nuk ka ndonjë vlerë lidhëse në tabelën e djathtë atëherë i ndahet vlera *NULL* vlerës së tabelës së djathtë.
Mund të fitojmë rezultat të njëjtë edhe me fjalën kyçe *USING*, si mëposhtë:

SELECT Emri, Mbiemri, Tipi_i_Kredise
FROM Klientet LEFT JOIN Kredite
USING(Numri)
ORDER BY Emri;

Poashtu mund të bëjmë lidhje të jashtme të majtë edhe për më shumë se dy tabela.
Shembull:

SELECT BookTitle, Copyright, AuthFN, AuthMN, AuthLN
FROM Books AS b LEFT JOIN AuthorBook AS ab ON b.BookID=ab.BookID
LEFT JOIN Authors AS a ON ab.AuthID=a.AuthID
ORDER BY BookTitle;

Lidhja e jashtme e djathtë (Right outer join)

Lidhja e jashtme e djathtë është e ngjashme me lidhjen e jashtme të djathtë, vetëm se dallimi qëndron që të dhënat kthehen nga tabela e djathtë. Ja edhe sintaksa që tregon definimin e lidhjes së djathtë të jashtme:

<tabela referuese> RIGHT [OUTER] JOIN <tabela referuese> [<kushtet e lidhjes>]

<kushtet e lidhjes>::=
ON <shprehje> [{<operator> <shprehje>}...]
| USING (<fushë> [{, <fushë>}...])

Siç mund të shihet sintaksa është gati e njëjtë me atë të lidhjes së majtë, përveq fjalës kyçe *RIGHT*. Tani le të shohim një shembull:

Tab1

Numri	Emri	Qyteti
128	Gezim	Prishtine
234	Driton	Prizren
432	Festim	Mitrovice
534	Artan	Gjilan
567	Ilir	Peje

Tab2

Numri	Profesioni	Institucioni
128	Student	Mjeksi
234	Puntore	PTT
432	Mjek	Ambulance
433	Rojtare	Shkolle
567	Inxhinier	KEK

Si fitohet ky rezultat (Tab3) nga tabelat Tab1 dhe Tab2:

Tab3

Numri	Profesioni	Institucioni	Emri	Qyteti
128	Student	Mjeksi	Gezim	Prishtine
234	Puntore	PTT	Driton	Prizren
432	Mjek	Ambulance	Festim	Mitrovice
433	Rojtare	Shkolle	NULL	NULL
567	Inxhinier	KEK	Ilir	Peje

SELECT Tab2.Numri, Tab2.Profesioni, Tab2.Institucioni, Tab1.Emri, Tab1.Qyteti
FROM Tab1 RIGHT JOIN Tab2
ON Tab1.Numri=Tab2.Numri;

Në shembullin e mësipërm do të fitohet rezultat i njëjtë edhe në rastin e lidhjes së jashtme të majtë vetëm se tabela e parë referuese duhet të jetë Tab2 pastaj tabela e dytë referuese do të jetë Tab1.

Tab2 LEFT JOIN Tab1

Si fitohet rezultati (Tab4) i mëposhtëm nga tabelat Tab1 dhe Tab2

Tab4

Numri	Profesioni	Institucioni	Emri	Qyteti
128	Student	Mjeksi	Gezim	Prishtine
234	Puntore	PTT	Driton	Prizren
432	Mjek	Ambulance	Festim	Mitrovice
NULL	NULL	NULL	Artan	Gjilan
567	Inxhinier	KEK	Ilir	Peje

SELECT Tab2.Numri, Tab2.Profesioni, Tab2.Institucioni, Tab1.Emri, Tab1.Qyteti
FROM Tab1 LEFT JOIN Tab2
ON Tab1.Numri=Tab2.Numri;

Edhe në këtë shembull si më lartë fitohet rezultat i njëjtë edhe me lidhjen e jashtme të djathtë:

Tab2 RIGHT Tab1

Krijimi i lidhjes NATURAL

Një lidhje tjetër që mund të përcaktohen në urdhërin SELECT është lidhja NATURAL. Kjo lidhje mund të përcaktohet si lidhje INNER ose CROSS, ose lidhje e jashtme e majtë, ose lidhje e jashtme e djathtë. Sintaksa në vijim tregon definimin e lidhjes NATURAL:

<tabela referuese> NATURAL [{LEFT | RIGHT} [OUTER]] JOIN <tabela referuese>

Nga sintaksa shohim se nuk bëhet definimi i kushteve të lidhjes. Lidhja NATURAL automatikisht bën lidhjen e tabelave përmes fushave me emër të njëjtë. P.SH. nëse bëjmë lidhjen e tabelës Klientët dhe tabelës Kreditë, atëherë lidhja NATURAL do të definojë automatikisht lidhjen me fushat Numri të të dy tabelave.

Për krijimin e një lidhje *NATURAL* duhet të shkruhet së pari tabela referuese pastaj fjala kyçe *NATURAL JOIN* dhe tabela e dytë referuese. Nëse i fusim vetëm këto elemente do të krijojmë lidhje *[INNER | CROSS] JOIN*. Nëse përcaktojmë fjalët kyçe *LEFT* ose *LEFT OUTER* atëherë jemi duke krijuar lidhje të majtë, ndërsa nëse përcaktojmë fjalët kyçe *RIGHT* ose *RIGHT OUTER* jemi duke krijuar lidhje të djathtë.
Shembull:

Me ekzekutimin e këtij urdhëëri

SELECT Emri, Mbiemri, Tipi_i_kredise
FROM Klientet NATURAL JOIN Kredite;

do të fitojmë rezultat të njëjtë sikurse me lidhjen *[INNER | CROSS] JOIN*, pra urdhëri i mësipërm jep rezultat të njëjtë sikur urdhëri:

SELECT Emri, Mbiemri, Tipi_i_kredise
FROM Klientet INNER JOIN Kredite
ON Klientet.Numri=Kredite.Numri;

Shihet që te urdhëri *NATURAL JOIN* nuk është bërë definimi i kushteve të lidhjes sit e lidhja e *INNER JOIN*, nuk ka fjalë kyçe *ON* ose *USING*. Pra lidhja *NATURAL* automatikisht e bën lidhjen ndërmjet tabelave në fushat me emër të njëjtë. Nëse dëshirojmë lidhje të majtë të jashtme vetëm duhet të përcaktohet fjala *LEFT* ose *LEFT OUTER:*

SELECT Emri, Mbiemri, Tipi_i_kredise
FROM Klientet NATURAL LEFT JOIN Kredite;

Nëse dëshirojmë lidhje të djathtë të jashtme vetëm duhet të përcaktohet fjala *RIGHT* ose *RIGHT OUTER*:

SELECT Emri, Mbiemri, Tipi_i_kredise
FROM Klientet NATURAL RIGHT JOIN Kredite;

Rezultatet janë të njëjtë sikur te shembujt te lidhjet *LEFT* dhe *RIGHT*.
Por nëse në dy tabela që dëshirojmë të bëjmë lidhje *NATURAL* dhe në ato dy tabela nuk ekzistojnë fusha me emër të njëjtë atëherë si rezultat do të fitojmë prodhimin kartezian të të dhënave të atyre tabelave.

Bashkimi i tabelave në urdhërin SELECT

Nëse dëshirojmë të bashkojmë rezultatet që dalin nga urdhërat *SELECT* në një rezultat atëherë një mënyrë e lehtë për të bërë këtë është me anë të operatorit *UNION*, ja edhe sintaksa:

<urdhëri select> UNION <urdhëri select>

Me bashkimin e dy urdhërave në këtë mënyrë kthen si rezultat numrin e njëjtë të fushave që i korrespondojnë secilit urdhër dhe tipin e të dhënave të njëjtë. P.SH. nëse me urdhërin e parë marrim si rezultat dy fusha Fusha1 dhe Fusha2. Ku Fusha1 e ka tipin e të dhënave *INT* ndërsa Fusha2 tipin e të dhënave *CHAR*. Atëherë si rezultat duhet që urdhëri i dytë të kthej dy fusha me tipe të të dhënavë *INT* dhe *CHAR* përkatësisht.

SELECT Emri, Mbiemri FROM Klientet
UNION
SELECT Numri, Tipi_i_kredise FROM Kredite;

Funksionet

Më herët mësuam për shprehjet në *SQL* urdhërat. Që shprehjet të jenë më të fuqishme dhe specifike kemi përdor elemente si funksionet. Secili funksion kryen një punë të caktuar dhe kthen një vlerë që është rezultat i punës së caktuar. Për shumë funksione duhet të japim më shumë se një argument që i ndahen si vlerë çdo parametri që përdoret në funksion për kryerjen e një pune të caktuar. Këto punë mund të jenë kalkulime të të dhënave numerike, manipulimi it ë dhënave string, manipulime me data e shumë operacione tjera. Funksionet ndahen në funksione për datë dhe kohë, funksione për stringje, funksione numerike dhe funksionet aggregate.

Funksionet për datë dhe kohë

Funksionet për datë përdoren kur kemi të bëjmë me datë dhe kohë, si kthimin e një date të caktuar.

CURRENT_DATE( )

Kthen si rezultat datën aktuale në formën si string YYYY-MM-DD ose si vlerë numerike YYYYMMDD.
Shembull:

mysql> **SELECT CURRENT_DATE();**

```
+----------------+
| CURRENT_DATE() |
+----------------+
| 2007-01-11     |
+----------------+
```

mysql> **SELECT CURRENT_DATE()+1;**

```
+------------------+
| CURRENT_DATE()+1 |
+------------------+
|         20070112 |
+------------------+
```

CURRENT_TIME( )

Kthen si rezultat kohën aktuale, si string hh:mm:ss ose si numër hhmmss.

mysql> **SELECT CURRENT_TIME();**

```
+----------------+
| CURRENT_TIME() |
+----------------+
| 23:53:15       |
+----------------+
```

mysql> **SELECT CURRENT_TIME() + 1;**

```
+--------------------+
| CURRENT_TIME() + 1 |
+--------------------+
|             235434 |
+--------------------+
```

DAYNAME( )

Kthen si rezultat emrin e ditës në datën e caktuar.

mysql> **SELECT DAYNAME('2007-01-11');**

```
+-----------------------+
| DAYNAME('2007-01-11') |
+-----------------------+
| Thursday              |
+-----------------------+
```

Pastaj janë funksionet DAYOFMONTH( ), DAYOFWEEK( ) dhe DAYOFYEAR( ) që kthejnë si rezultat ditën e muajit, ditën e javës dhe ditën e vitit të datës së caktuar përkatësisht.

Funksioni	Përshkrimi	Shembull	Rezultati
HOUR( )	Kthen orën për kohën e caktuar	SELECT HOUR('06:59:03');	6
MINUTE ( )	Kthen minutat për kohën e caktuar	SELECT MINUTE('00:01:03');	1
MONTH( )	Kthen muajin për datën e caktuar	SELECT MONTH('2007-12-21');	12

MONTHNAME( )	Kthen emrin e muajit për datën e caktuar	SELECT MONTHNAME('2006-12-21');	December
NOW( )	Kthen datën dhe kohën aktuale	SELECT NOW( )	2007-01-11 12:00:00

Këto janë disa nga funksionet e shumta për datë dhe kohë.

Funksionet për string

Të gjitha funksionet string marrin si argument string dhe kthejnë si rezultat një string.

ASCII( )

Kthen si rezultat vlerën ASCII të karakterit të parë nga e majta, kthen 0 kur stringu është i zbrazët dhe NULL kur stringu ka vlerën NULL.

mysql> **SELECT ASCII('a');**
```
+------------+
| ASCII('a') |
+------------+
|         97 |
+------------+
```
mysql> **SELECT ASCII('aza');**
```
+-------------+
| ASCII('az') |
+-------------+
|          97 |
+-------------+
```
mysql> **SELECT ASCII('');**
```
+------------+
| ASCII('')  |
+------------+
|          0 |
+------------+
```
mysql> **SELECT ASCII(NULL);**
```
+------------+
| ASCII('a') |
+------------+
|       NULL|
+------------+
```

BIN( )
Kthen si rezultat vlerën binare për ndonjë numër (BIGINT), kthen vlerën 0 nëse numri nuk mund të konvertohet (funksioni do të konvertojë derisa mundet nga e majta) dhe NULL nëse është NULL.

mysql> **SELECT BIN(15);**
```
+---------+
| BIN(15) |
+---------+
| 1111    |
+---------+
```
mysql> **SELECT BIN('8');**
```
+----------+
| BIN('8') |
+----------+
| 1000     |
+----------+
```
mysql> **SELECT BIN('2w');**
```
+-----------+
| BIN('2w') |
+-----------+
| 10        |
+-----------+
```
mysql> **SELECT BIN('ë2');**
```
+-----------+
| BIN('w2') |
+-----------+
| 0         |
+-----------+
```
Ky funksion është ekuivalent me funksionin CONV(numri,10,2).

CHAR( )
Kthen si rezultat karakteret sikur secili numër të jetë i konvertuar nga ASCII kodi.
Numrat decimal rrumbullaksohen me vlerën më të afërt.

mysql> **SELECT CHAR(97,101,105,111,117);**
```
+-------------------------+
| CHAR(97,101,105,111,117) |
+-------------------------+
| aeiou                   |
+-------------------------+
```
mysql> **SELECT CHAR(97.6,101,105,111,117);**
```
+--------------------------+
| CHAR(0.97,101,105,111,117) |
+--------------------------+
| beiou                    |
+--------------------------+
```

CONCAT( )

Lidh dy argumentet stringje dhe kthen si rezultat, *NULL* nëse argumenti është *NULL*. Argumentet që nuk janë stringje kthehen në stringje.

mysql> **SELECT CONCAT('a','b');**

```
+-----------------+
| CONCAT('a','b') |
+-----------------+
| ab              |
+-----------------+
```

mysql> **SELECT CONCAT('a',12);**

```
+----------------+
| CONCAT('a',12) |
+----------------+
| a12            |
+----------------+
```

CONCAT_WS( )

Është e ngjashme me CONCAT( ) vetëm se argumenti i parë është separator që vëhet në mes secilit argument.

mysql> **SELECT CONCAT_WS('-','a','b');**

```
+-----------------------+
| CONCAT_ËS('-','a','b') |
+-----------------------+
| a-b                   |
+-----------------------+
```

mysql> **SELECT CONCAT_WS('-','a','b','c');**

```
+--------------------------+
| CONCAT_ËS('-','a','b','c') |
+--------------------------+
| a-b-c                    |
+--------------------------+
```

CONV( )

Është funksion që bën konvertimin e numrave nga një bazë te tjetra.
CONV(numri, prej cilës bazë, te cila bazë)

CONV(10,10,2) d.m.th Numrin 10 konverto nga baza 10 në bazën 2.
CONV('3f',16,10)d.m.th. konverto numrin 3f nga baza heksadecimale në atë dhjetëshe.
CONV('a',16,2) d,m,th. Konverto numrin a nga baza 16 në bazën 2 etj.

HEX( )
Kthen vlerën heksadecimale për një numër, kthen vlerën 0 nëse nuk mund të konvertojë numrin dhe *NULL* për vlerën *NULL*.

mysql> **SELECT HEX(13);**
```
+---------+
| HEX(13) |
+---------+
| D       |
+---------+
```

LENGTH( )

Kthen gjatësinë në karakterë të stringut.

mysql> **SELECT LENGTH('MySQL');**
```
+-----------------+
| LENGTH('MySQL') |
+-----------------+
|               5 |
+-----------------+
```
mysql> **SELECT LENGTH(99);**
```
+------------+
| LENGTH(99) |
+------------+
|          2 |
+------------+
```

LOËER( )
Kthen si rezultat të gjithë karakterët në shkronja të vogla.

mysql> **SELECT LOWER('AbC');**
```
+--------------+
| LOËER('AbC') |
+--------------+
| abc          |
+--------------+
```

UPPER( )
Kthen si rezultat të gjithë karakterët në shkronja të mëdha.
mysql> **SELECT UPPER('aBc');**
```
+--------------+
| UPPER('aBc') |
+--------------+
| ABC          |
+--------------+
```
Këto janë disa nga funksionet për stringje.

Funksionet numerike

Funksionet numerike përdoren për kalkulime, ku si argument marrin më së shpeshti numër. Në rast të ndonjë gabimi ato lajmërojnë për gabim. Duhet të kemi kujdes që të mos e kalojmë intervalin e tipit të të dhënave. Shumica e funksioneve punojnë me tipin e të dhënave BIGINT (2^{63} SIGNED ose 2^{64} UNSIGNED).

ABS( )

Kthen vlerën absolute të një numri.

mysql> **SELECT ABS(24-26);**

```
+------------+
| ABS(24-26) |
+------------+
|          2 |
+------------+
```

DEGREES( )

Konverton radianët në shkallë.

mysql> **SELECT DEGREES(2);**

```
+-----------------+
| DEGREES(2)      |
+-----------------+
| 114.59155902616 |
+-----------------+
```

mysql> **SELECT DEGREES(PI()/2);**

```
+-----------------+
| DEGREES(PI()/2) |
+-----------------+
|              90 |
+-----------------+
```

EXP( )

Kthen si rezultat numrin e të ngritur në fuqi me numrin e caktuar.

mysql> **SELECT EXP(1);**

```
+----------+
| EXP(1)   |
+----------+
| 2.718282 |
+----------+
```

LN( )

Kthen si rezultat logaritmin natyral të numrit të caktuar.

```
mysql> SELECT LN(2.3);
+-----------------+
| LN(2.3)         |
+-----------------+
| 0.8329091229351 |
+-----------------+
```

LOG10( )

Kthen si rezultat logaritmin me bazë dhjetë të numrit të caktuar.

mysql> **SELECT LOG10(100);**

```
+-------------+
| LOG10(100)  |
+-------------+
|   2.000000  |
+-------------+
```

MOD( )

Kthen modulin e dy numrave të caktuar (pra mbetjen e pjestimit të atyre numrave).

mysql> **SELECT MOD(15,4);**

```
+------------+
| MOD(15,4)  |
+------------+
|          3 |
+------------+
```

PI( )

Kthen vlerën e pi-së.

mysql> **SELECT PI();**

```
+----------+
| PI()     |
+----------+
| 3.141593 |
+----------+
```

POWER( )

Kthen si rezultat argumentin e parë të ngritur në fuqi në argumentin e dytë.

mysql> **SELECT POËER(2,3);**

```
+------------+
| POËER(2,3) |
+------------+
|   8.000000 |
+------------+
```

RADIANS( )

Konverton numrin e shkallëve në radian.

mysql> **SELECT RADIANS(180);**

```
+-----------------+
| RADIANS(180)    |
+-----------------+
| 3.1415926535898 |
+-----------------+
```

SQRT( )
Kthen rrënjën katrore të numrit të caktuar.

mysql> **SELECT SQRT(81);**

```
+----------+
| SQRT(81) |
+----------+
| 9.000000 |
+----------+
```

TRUNCATE( )

Cungon numrin e caktuar deri te numri i caktuar i decimaleve.
mysql> **SELECT TRUNCATE(2.234,2);**

```
+------------------+
| TRUNCATE(2.234,2) |
+------------------+
|              2.23 |
+------------------+
```

mysql> **SELECT TRUNCATE(2.998,0);**

```
+------------------+
| TRUNCATE(2.998,0) |
+------------------+
|                 2 |
+------------------+
```

Funksioni	Përshkrimi	Shembull	Rezultati
SIN( )	Kthen sinusin e numrit të caktuar në radian	SELECT SIN(5.14);	0.850904
COS( )	Kthen kosinusin e numrit të caktuar në radian	SELECT COS(5.14);	0.525322
TAN( )	Kthen tangjentin për numrin e caktuar në radian	SELECT TAN(5.14);	1.619775
COT( )	Kthen kotangjentin për numrin e caktuar në radian	SELECT COT(5.14);	0.617370
ASIN( )	Kthen vlerë e arkus sinusit të numrit të caktuar	SELECT ASIN(0.5);	0.523599
ACOS( )	Kthen vlerë e arkus kosinusit të numrit të caktuar	SELECT ACOS(0.5);	1.047198
ATAN( )	Kthen arkus tangjentin për numrin e caktuar në radian	SELECT ATAN(5.14);	1.548578

GREATEST( )

Kthen si rezultat argumentin me vlerë më të madhe.

mysql> **SELECT GREATEST(-3,-4,5);**
```
+-------------------+
| GREATEST(-3,-4,5) |
+-------------------+
|                 5 |
+-------------------+
```
mysql> **SELECT GREATEST('Pa','Ma','Ca');**
```
+--------------------------+
| GREATEST('Pa','Ma','Ca') |
+--------------------------+
| Pa                       |
+--------------------------+
```

LEAST( )

Kthen si rezultat argumentin me vlerë më të vogël.

mysql> **SELECT LEAST(-3,-4,5);**
```
+----------------+
| LEAST(-3,-4,5) |
+----------------+
|             -4 |
+----------------+
```

mysql> **SELECT LEAST('Pa','Ma','Ca');**

```
+----------------------+
| LEAST('Pa','Ma','Ca') |
+----------------------+
| Ca                   |
+----------------------+
```

Funksionet aggregate

Funksionet aggregate janë funksione që bëjnë grupimin e të dhënave(duke supozuar që është përdorë fjala kyçe *GROUP BY*).

AVG( )

Kthen vlerën mesatare të ndonjë shprehje. Kthen vlerën 0 nëse shprehja nuk është tip i të dhënave numerik.

mysql> **SELECT AVG(Shuma) FROM Kredite;**

```
+-------------+
| AVG(Shuma)  |
+-------------+
|   6948.8889 |
+-------------+
```

mysql> **SELECT AVG(Shuma) FROM Kredite WHERE Shuma>1000;**

```
+-------------+
| AVG(Shuma)  |
+-------------+
|   12056.0000|
+-------------+
```

COUNT( )

Kthen numrin e vlerave jo-boshe në një fushë.

mysql> **SELECT COUNT(Emri) FROM Klientet;**

```
+-------------+
| COUNT(Emri) |
+-------------+
|           6 |
+-------------+
```

mysql> **SELECT COUNT(Emri) FROM Klientet WHERE Emri LIKE 'B%';**

```
+-------------+
| COUNT(Emri) |
+-------------+
|           2 |
+-------------+
```

MAX( )

Kthen vlerën maksimale të një shprehje. Shprehja mund të jetë numerike dhe string.

mysql> **SELECT MAX(Shuma) FROM Kredite;**

```
+-------------+
| MAX(Shuma)  |
+-------------+
|       45000 |
+-------------+
```

mysql> **SELECT MAX(Emri) FROM Klientet;**

```
+-------------+
| MAX(Emri)   |
+-------------+
|     Valbona |
+-------------+
```

MIN( )

Kthen vlerën më të vogël të një shprehje, shprehja mund të jetë numerike dhe string.

mysql> **SELECT MIN(Shuma) FROM Kredite;**

```
+-------------+
| MIN(Shuma)  |
+-------------+
|         420 |
+-------------+
```

mysql> **SELECT MIN(Perqindja) FROM Kredite;**

```
+--------------+
|MIN(Perqindja)|
+--------------+
|            3 |
+--------------+
```

STD( )

Kthen devijimin standard për vlerat në shprehje.

mysql> **SELECT STD(Shuma) FROM Kredite;**

```
+-------------+
| STD(Shuma)  |
+-------------+
|  13634.8862 |
+-------------+
```

mysql> **SELECT STD(Perqindja) FROM Kredite;**

```
+--------------+
|STD(Perqindja)|
+--------------+
|      1.6330  |
+--------------+
```

SUM( )

Kthen si rezultat shumën e vlerave të një fushe ose shprehje.

mysql> **SELECT SUM(Shuma) FROM Kredite;**

```
+-------------+
| SUM(Shuma)  |
+-------------+
|      62540  |
+-------------+
```

mysql> **SELECT SUM(Shuma) FROM Kredite WHERE Perqindja>6;**

```
+-------------+
| SUM(Shuma)  |
+-------------+
|       7390  |
+-------------+
```

PROCEDURAT E RUAJTURA (STORED PROCEDURES)

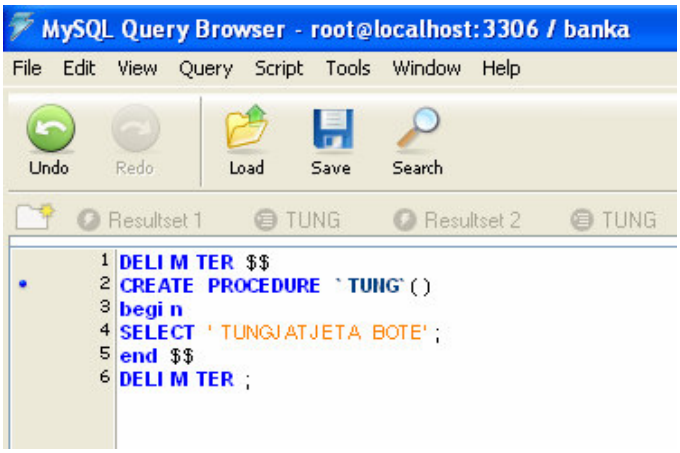
Procedurat e ruajtura janë një tip i programit të ruajtur, të cilat janë të përgjithshme që ekzekutohen në kërkesë të klientit dhe mund të marrin vlera hyrëse dhe të japin vlera dalëse. Përdorimi i procedurave të ruajtura e bën një bazë të të dhënave më të sigurtë, ofrojnë mekanizma për mirëmbajtjen e bazës së të dhënave, bëjnë zvogëlimin e trafikut në rrjetë sepse programi punon me të dhëna në server se sa transferi i tyre nëpër rrjetë etj. Kodi për të gjitha procedurat e ruajtura shkruhet në MySQL Query Browser i cili mund të mirret nga http://dev.mysql.com .

Krijimi i procedurës së ruajtur

Hapim MySQL Query Browser shkojmë të File → New Script tab dhe shkruajmë kodin për atë procedurë. Për krijimin e një procedure përdoret urdhëri

CREATE PROCEDURE< emri_i_procedurës> ([<argumenti> <tipi_i_të_dhënave [{, <argumenti> <tipi_i_të_dhënave }...]])

Pastaj fillojmë me bllokun *BEGIN* , brenda këtij blloku shkruajmë kodin dhe e mbyllim me *END*. Kjo ka një ngjashmëri me gjuhën programuese *PASCAL*.



Shembull:
Shkruajmë një procedurë që si rezultat do të jap shprehjen ***Tungjatjeta botë .***

DELIMITER $$
CREATE PROCEDURE `TUNG`()
begin
SELECT 'TUNGJATJETA BOTË';
end $$
DELIMITER;

E ruajmë këtë procedurë me TUNG.sql pra emri i njëjtë si i procedurës, dhe e ekzekutojmë. Ekzekutimi i procedurës së ruajtur bëhet kështu:

CALL <emri_i_procedurës>([parametrat]);

Pra CALL TUNG( ); dhe rezultati do të jetë:

```
mysql> CALL TUNG( );
+-----------------+
|TUNGJATJETA BOTË |
+-----------------+
|TUNGJATJETA BOTË |
+-----------------+
1 row in set (0.01 sec)
```

Tani krijojmë një procedurë me anë të cilës do t'i shohim të dhënat nga tabela *Klientët*.

```
DELIMITER $$
CREATE  PROCEDURE `TE_DHENAT`()
begin
SELECT * FROM klientet;
end $$
DELIMITER;
```

E ruajmë gjithmonë me emrin e procedurës *TE_DHENAT.sql* dhe e ekzekutojmë. Do të fitojmë të gjitha të dhënat e tabelës Klientët.

Procedurat siç u cek në fillim mund të marrin parametra, kjo gjë sigurohet gjatë shkruarjes së kodit. Krijojmë një procedurë që merr si parametra vlera që do të shtohen në një tabelë.
Tani krijojmë një procedurë për shtimin e të dhënave në tabelën *Klientët.*

```
DELIMITER $$
CREATE  PROCEDURE `SHTO_TE_DHENAT`(Numri VARCHAR(10),
Emri VARCHAR(20),Mbiemri VARCHAR(20),Qyteti VARCHAR(20),
Adresa VARCHAR(20),Tel VARCHAR(15) )
begin
INSERT INTO Klientet VALUES (Numri,Emri,Mbiemri,Qyteti,Adresa,Tel);
end $$
DELIMITER ;
```

E ruajmë me emrin SHTO_TE_DHENAT.sql dhe e ekzekutojmë. Tani ekzekutimi i kësaj procedure bëhet kështu:

```
CALL SHTO_TE_DHENAT('140207','ARSIM','BAJRAMAJ','BURIM','rr.A',
'044/123-789');
```

Pas ekzekutimit të kësaj procedure do të shohim se në tabelën *Klientët* është shtuar një e dhënë, pikërisht parametrat e procedurës.

Brenda kodit të procedurës mund të thërrasim procedurë tjetër. Ja një shembull si kombinim i shembujve të mëparshëm.

Shembull:

Krijojmë procedurën që do të shtoj të dhëna në tabelën Klientët por edhe do të thërret procedurën *TE_DHENAT* me anë të së cilës do t'i shohim të dhënat nga tabela *Klientët.*

```
DELIMITER $$
CREATE PROCEDURE `SHTO_SHIKO_TE_DHENAT`(Numri VARCHAR(10),
Emri VARCHAR(20),Mbiemri VARCHAR(20),Qyteti VARCHAR(20),
Adresa VARCHAR(20),Tel VARCHAR(15) )
begin
INSERT INTO Klientet VALUES (Numri,Emri,Mbiemri,Qyteti,Adresa,Tel);
CALL TE_DHENAT();
end $$
DELIMITER ;
```

Pra brenda kodit të procedurës u thirr procedura *TE_DHENAT( ).*

```
CALL TE_DHENAT();
```

Tani pas ekzekutimit të kësaj procedure do të shtohet e dhëna që procedura e merr si parametër dhe ekzektohet procedura *TE_DHENAT( )* që mundëson shikimin e të dhënave nga tabela Klientët.

PHP, APACHE, MySQL në WEB

PHP, APACHE dhe MySQL janë pjesë e grupit të programeve *open source.*

Kuptimi i rolit të secilit prej këtyre programeve në krijimin e një WEB faqeje

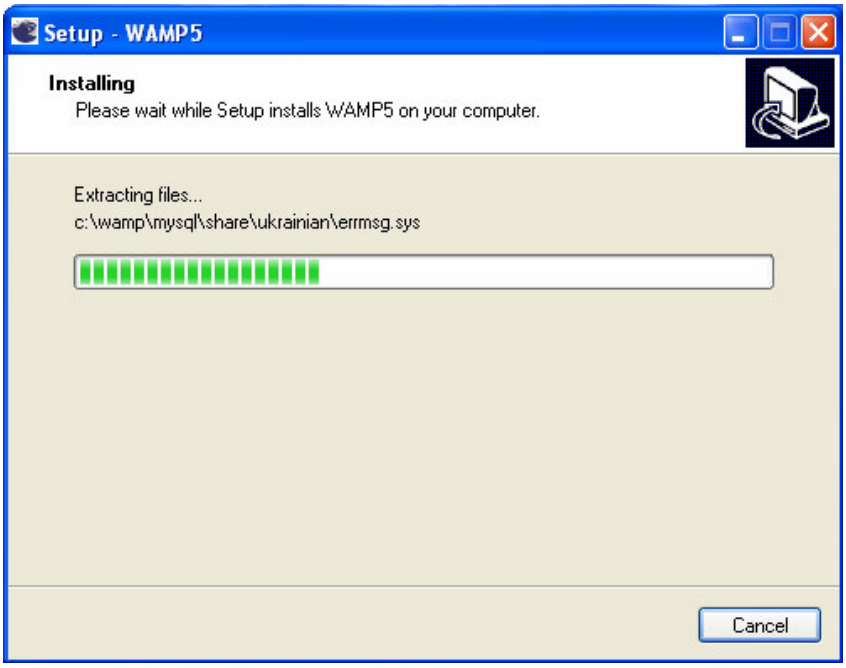
Apache është një WEB server, puna e tij kryesore është të analizojë çdo datotekë që është kërkuar nga shfrytëzuesi dhe të paraqes rezultatin duke u bazuar në kodin brenda asaj datoteke. Apache është mjaft i fuqishëm dhe mund të përmbush çdo kërkesë që i shtroni. Mund t'a shkarkoni në adresën http://httpd.apache.org .
PHP është gjuhë skriptuese, pjesë e serverit që mundëson WEB faqja të jetë dinamike. PHP (Hypertext Preprocessor) është fleksibil dhe i lehtë për tu mësuar sidomos për ata që kanë njohuri në C, Java ose Perl. Është një nga gjuhët skriptuese më të popullarizuara. Mund t'a shkarkoni në adresën http://www.php.net .
MySQL është baza e të dhënave e ndërtuar në atë mënyrë që mundëson PHP-në dhe Apache-in të punojnë së bashku për qasjen dhe shikimin e të dhënave në një WEB browser. Mund t'a shkarkoni në adresën http://www.mysql.com .

Instalimi dhe konfigurimi i PHP, Apache dhe MySQL

Instaluesit AMP

Nëse nuk dëshirojmë që të bëjmë instalimin e të tri komponenteve një nga një atehëre mund të bëjmë instalimin e këtyre komponenteve me një datotekë instaluese. Datoteka instaluese janë disa por ne do të shkarkojmë instaluesin WAMP5 nga adresa http://www.wampserver.com/en ose në ftp://comp-253 . Me anë të WAMP5 bëjmë instalimin e të tri komponenteve dhe poashtu bëhet konfigurimi i tyre.
Klikojmë në datotekën instaluese dhe hapet dritarja:

Shtypim pullën *Next* dhe në dritaren vijuese zgjedhim opsionin: *I accept the agreement* pastaj shtypim sërish pullën *Next*. Në dritaren e re mund të ndryshojmë shtegun e instalimit të komponenteve por ne e lëmë ashtu siç është *C:\wamp* dhe shtypim pullën *Next*. Në dritaren tjetër veprojmë njësoj vetëm shtypim pullën *Next* dhe në dritaren vijuese zgjedhim opsionin për startim automatik të *WAMP5* dhe shtypim *Next* dhe në fund *Install*.

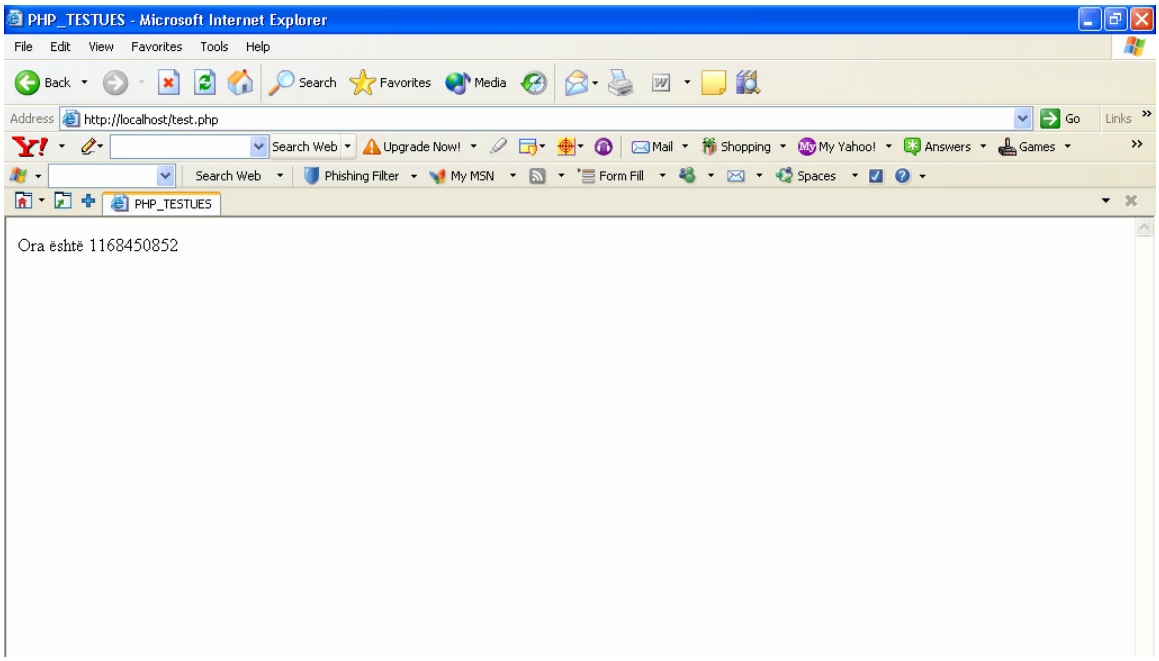


Gjatë instalimit paraqitet dritarja:

ku duhet të zgjedhim vendin ku do të ruhen datotekat që do të kërkohen nga *serveri* (gjuhë skriptuese). Pastaj zgjedhim *SMTP* serverin ose e lëmë *localhost* dhe shtypim *Next*. Pastaj shkruajmë *e-mail* adresën p.sh. *admin@localhost* dhe shtypim *Next*. Pastaj përfundon instalimi i këtyre komponenteve shtypim *Finish* dhe do të startojë *WAMP* serveri.
Që këto tri komponenete të punojnë së bashku duhet bërë konfigurimi i tyre, i cili bëhet në datotekat *httpd.conf* për Apache, *php.ini* për PHP dhe *my.ini* për MySQL.
Konfigurimet e tyre mund t'i gjeni në ftp://comp-253 në datotekat përkatëse.
Tani shkruajmë datotekat për krijmin e bazës së të dhënave, krijimin e tabelave dhe shtimin e të dhënave në ato tabela, e gjitha kjo bëhet me anë të gjuhës skriptuese PHP.
Hapim *NOTEPAD-in* dhe shkruajmë kodin e mëposhtëm:

```
<html>
<head>
<title>PHP_TESTUES</title>
</head>
<body>
<?php
echo "Ora është ".time();
?>
</body>
</html>
```

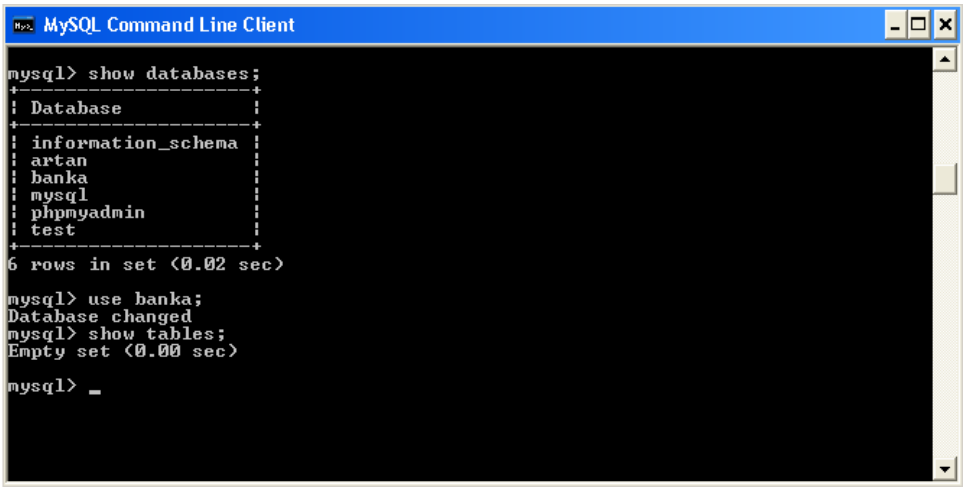
E ruajmë këtë me emrin *test.php* në shtegun *C:/wamp/www*, pra në vendin ku serveri do të kërkojë datotekat. Hapim *INTERNET EXPLORER* dhe kërkojmë adresën *http://localhost/test.php* dhe paraqitet:



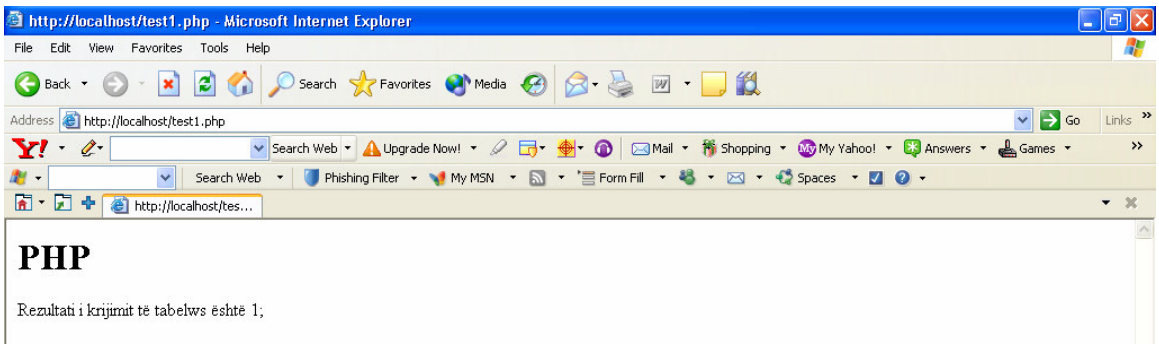
Krijojmë tani tabelën *Klientet* në bazën e të dhënave *Banka*.

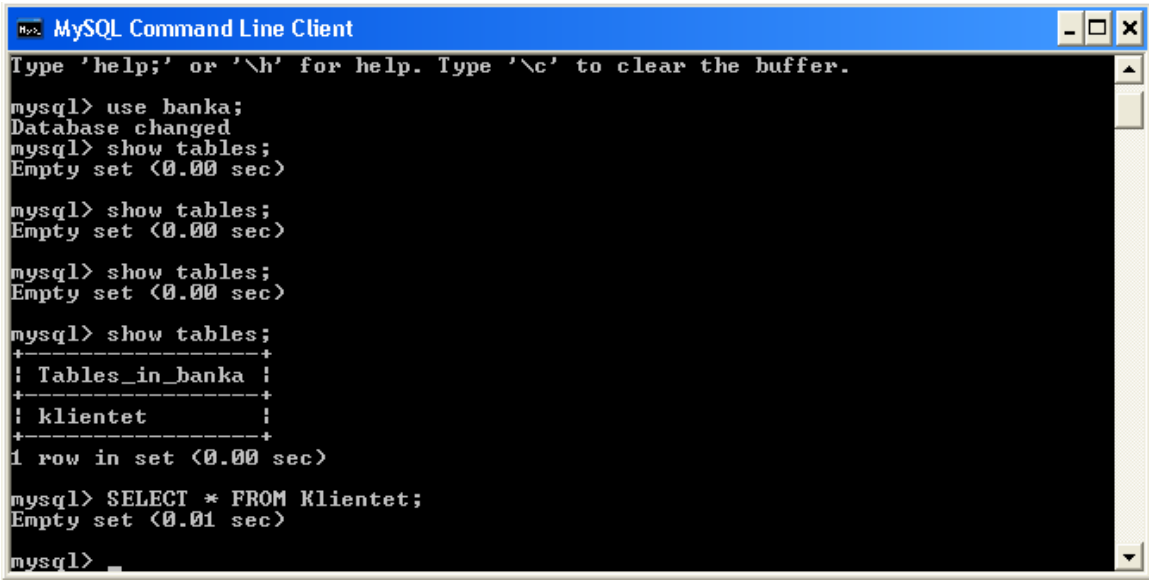
```
<html>
<body>
<h1>PHP</h1>
<?php
$lidhja = mysql_connect("localhost","root","");
mysql_select_db("banka");
$pyetesor="CREATE TABLE Klientet
      (
          Numri INT NOT NULL PRIMARY KEY,
          Emri VARCHAR(20) NOT NULL,
          Mbiemri VARCHAR(20) NOT NULL,
          Qyteti VARCHAR(15) NOT NULL,
          Adresa VARCHAR(20) NOT NULL,
          Tel VARCHAR(20) NOT NULL
      )";
$rezultati=mysql_query($pyetesor);
echo "Rezultati i krijimit të tabelës është $rezultati<br>\n";
?>
</body>
</html>
```

E ruajmë këtë me emrin *Test1.php* në *C:/wamp/www*. Kyçemi tani në server, shohim se aty është baza Banka por ne të nuk ka të krijuar asnjë tabelë.



Kur të kërkojmë në adresën http://localhost/Test1.php do të krijojmë tabelën *Klientet* në bazën e të dhënave *Banka*.





Por në tabelën *Klientet* nuk asnjë të dhënë, bëjmë shtimin e të dhënave në tabelën *Klientët*.

```
<html>
<body>
<h1>PROVA</h1>
<?php
$lidhja = mysql_connect("localhost","root","");
$Baza = mysql_select_db ("banka");
$pyetesor = "INSERT INTO Klientet VALUES
('140201','Bujar','Shulemaja','Prishtinë','Bregu i diellit','044/123-456'),
('140202','Amir','Simnica','Fushë Kosovë','rr.Agim Ramadani','044/321-654'),
('140203','Valbona','Krasniqi','Dardanë','rr.Adem Jashari','044/213-546'),
('140204','Gani','Thaqi','Besianë','rr.Zahir Pajaziti','044/312-645'),
('140205','Lavdim','Kastrati','Prishtinë','Tophane','044/132-465')";
$rezultati = mysql_query ($pyetesor);
echo "Rezultati i shtimit te te dhenave eshte $rezultati<br>\n";
?>
</body>
</html>
```

E ruajmë këtë me emrin *Test2.php* në *C:/wamp/www* dhe kur të kërkojmë në adresën http://localhost/Test2.php do të bëjmë shtimin e të dhënave në tabelën *Klientet*.