Оператори

Оператори омогућавају додавање бројева, упоређивање вредности, спајање стрингова (група текстуалних карактера) и креирање комплексних израза. Оператори се користе да се информише Access да је одређену операцију потребно извршити над једним или више операнада.
Постоје:

1. Математички оператори
2. Стринг оператори и
3. Логички оператори

У математичке операторе спадају:

*	Множење
+	Сабирање
-	Одузимање
/	Дељење
\	Дељење целим бројем
^	Степеновање
Mod	Модуо

Ако се јави потреба да се два броја заокруже на целе бројеве и након тога поделе па да се тако добијен количник преведе у цео број, оператор \ то обавља у једном кораку.
Модуо оператор или оператор остатка дели први број другим и враћа само остатак. Сви бројеви, ако нису цели, се пре примене овог оператора заокружују на целе.
У Access-у постоји један стринг оператор, а то је оператор конкатенације (&), односно спајања стрингова.

Логички оператори

WHERE клаузула се може користити са SELECT, UPDATE и DELETE наредбама. У SELECT-у она дефинише услове које мора да задовоље све врсте у резултујућој табели.
Логички израз у WHERE клаузули се састоји од имена поља (први операнд), оператора за поређење и имена поља, константне вредности или листе вредности (други операнд). Сложени услов се формира повезивањем простих или других сложених услова логичким операторима **И** (**AND**), **ИЛИ** (**OR**) и **НЕ** (**NOT**).
Операнди у WHERE клаузули могу бити: бројчана вредност, знаковна (карактер) или датумска вредност.
Оператори за поређење се деле у две групе

1. Логички оператори поређења
2. SQL оператори поређења

Логички оператори поређења су:

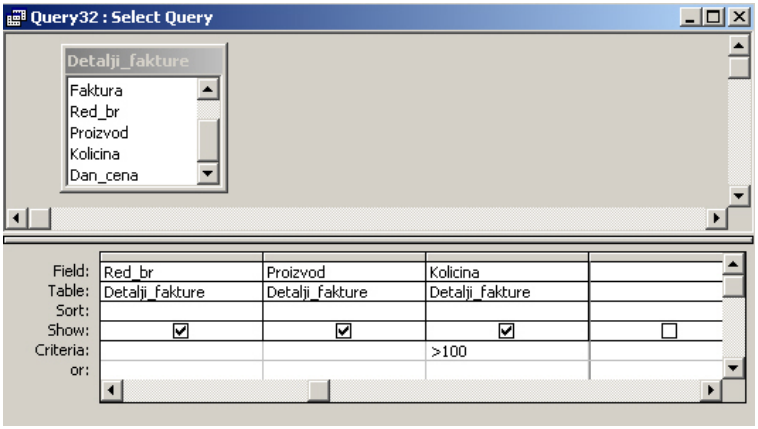
1. Једнако (=)
2. Није једнако (<>)
3. Веће (>)
4. Веће или једнако (>=)
5. Мање (<)
6. Мање или једнако (<=)

SQL оператори поређења или предикати су:

1. Између две вредности (BETWEEN ..... AND .....)
2. Листа вредности (IN (листа))
3. Поређење знаковних променљивих (карактера) (LIKE)
4. Недефинасана (нул) вредност (IS NULL)

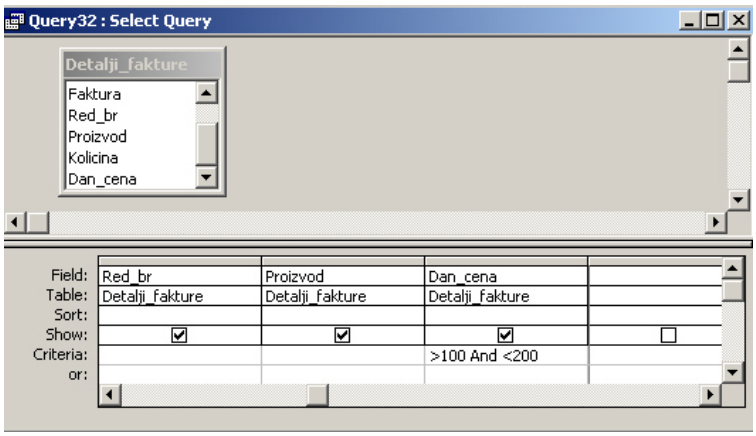
За негацију четири предиката употребљава се оператор NOT (није), тако да се добија NOT BETWEEN (није између), NOT IN (није у ...), NOT LIKE (није као...), и IS NOT NULL (није нул вредност или недефинисана вредност).

- Креирати упит који приказује све ставке на фактури (табела Detalji_fakture) такве да је количина већа од 100



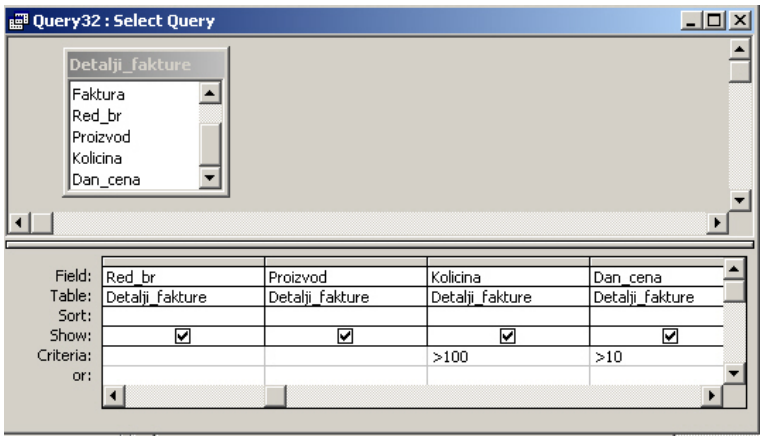

```
SELECT Detalji_fakture.Faktura, Detalji_fakture.Red_br, Detalji_fakture.Proizvod,
Detalji_fakture.Kolicina
FROM Detalji_fakture
WHERE (((Detalji_fakture.Kolicina)>100));
```

- Креирати упит који издваја све ставке са фактура за које је цена по јединици мере између 100 и 200 дин. (већа од 100 **И** мања од 200 дин.)



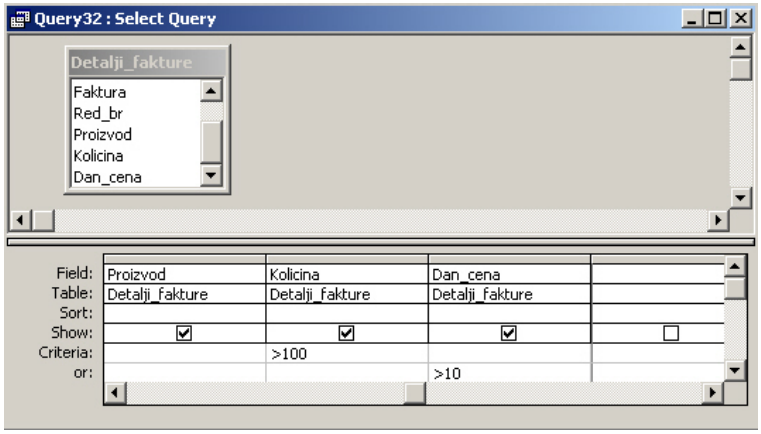

```
SELECT Detalji_fakture.Faktura, Detalji_fakture.Red_br, Detalji_fakture.Proizvod,
Detalji_fakture.Dan_cena
FROM Detalji_fakture
WHERE (((Detalji_fakture.Dan_cena)>100 And (Detalji_fakture.Dan_cena)<200));
```

- Приказати све ставке из фактуре код којих је количина већа од 100 **И** цена већа од 10



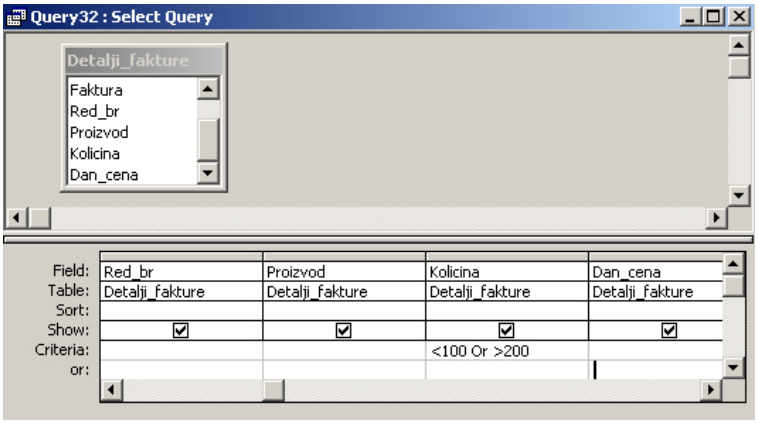

```
SELECT Detalji_fakture.Faktura, Detalji_fakture.Red_br, Detalji_fakture.Proizvod,
Detalji_fakture.Kolicina, Detalji_fakture.Dan_cena FROM Detalji_fakture
WHERE (((Detalji_fakture.Kolicina)>100) AND ((Detalji_fakture.Dan_cena)>10));
```

- Приказати све ставке из фактуре код којих је количина већа од 100 **ИЛИ** цена већа од 10




```
SELECT Detalji_fakture.Faktura, Detalji_fakture.Red_br, Detalji_fakture.Proizvod,
Detalji_fakture.Kolicina, Detalji_fakture.Dan_cena FROM Detalji_fakture
WHERE (((Detalji_fakture.Kolicina)>100)) OR (((Detalji_fakture.Dan_cena)>10));
```

- Креирати упит који издваја све ставке са фактура за које је количина мања од 100 **ИЛИ** већа од 200



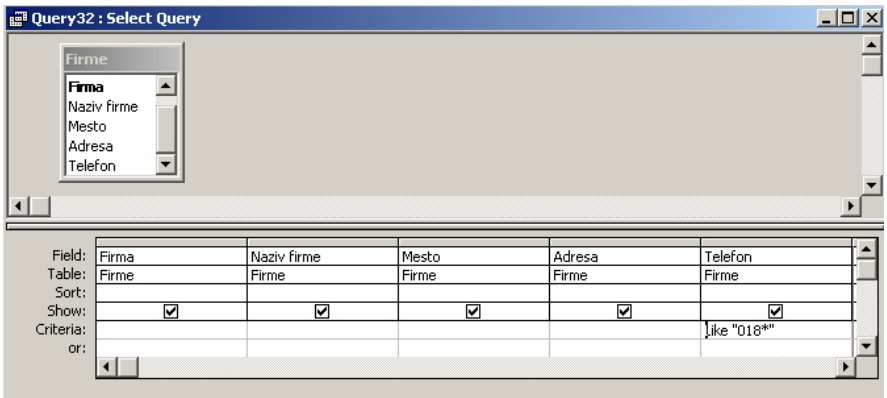

```
SELECT Detalji_fakture.Faktura, Detalji_fakture.Red_br, Detalji_fakture.Proizvod,
Detalji_fakture.Kolicina, Detalji_fakture.Dan_cena
FROM Detalji_fakture
WHERE (((Detalji_fakture.Kolicina)<100 Or (Detalji_fakture.Kolicina)>200));
```

LIKE

Access омогућава коришћење специјалних знакова када се користи LIKE оператор и то су:

?	Замењује било који карактер (A до Z, 0 до 9)
*	Било који низ карактера
#	Било која цифра (0 до 9)
[lista]	Било који карактер који је у овој листи
[!lista]	Било који карактер који није у овој листи

- Креирати упит који проналази све фирме којима број телефона почиње на 018




```
SELECT Firme.Firma, Firme.Naziv_firme, Firme.Mesto, Firme.Adresa, Firme.Telefon
FROM Firme
WHERE (((Firme.Telefon) Like "018*"));
```

Query32 : Select Query

	Firma	Naziv firme	Mesto	Adresa	Telefon
▶	1	BALKAN	Niš	Mokranjčeva 13	018-522-854
	3	KOKOMAX	Niš	Dušanova 33	018-352-666
*	0				

IS NULL

IS NULL је оператор који одређује да ли у неком пољу постоји вредност или не.

Агрегатне функције

Агрегатне фукције се извршавају над једним пољем (колоном) и враћају једну вредност.
У табели су приказане агрегатне функције које су подржане у Access-у. Првих 5 су стандарне, а последње две су специфичне за Access.

Функција	**Резултат**
Sum	Збир вредности у пољу (колони)
Avg	Средња вредност у пољу (колони)
Count	Број вредности у пољу (колони), не рачунајући Null (празне) вредности
Min	Најмања вредност у пољу (колони)
Max	Највећа вредност у пољу (колони)
StDev	Стандардна девијација вредности у пољу (колони)
Var	Варијанса вредности у пољу (колони)

Агрегане функције се могу користити у SELECT листи и у HAVING клаузули.
Вежба. Написати упит који одређује колико има фактура

- Отворити базу података **Poslovanje**.
- На траци *Objects*, притиснути *Queries*, а затим два пута брзо притиснути левим тастером миша на *Create Query in Design View*. Најпре ће се отворити прозор за упит у приказу *Design*, а затим се приказује оквир за дијалог *Show Table*.
- Затворити оквир за дијалог *Show Table* и прећи у *SQL* приказ
- Укуцати следећу SQL наредбу:

```
SELECT Count(Sifra_fakture) FROM Fakture;
```

- Извршити упит ()
- Резултат упита је само један запис који даје укупан број фактура
- Запамтити упит као **CountFakture1**

Овај упит је било могуће направити и у графичком моду, односно у *Design* приказу, на следећи начин;

- На траци *Objects*, притиснути *Queries*, а затим два пута брзо притиснути левим тастером миша на *Create Query in Design View*. Када се отвори оквир за дијалог

Show Table два пута кликнути на табелу Fakture и након тога притиснути дугме Close.

- Пребацити колону **Sifra_fakture** у решетку за пројектовање
- На палети алатки притиснути дугме *Totals* (Σ) и појавиће се додатна врста (*Total*) у решетки за пројектовање
- Кликнути у ћелију *Total* колоне **Sifra_fakture** и након тога кликнути на стрелицу усмерену на доле и из падајуће листе изабрати функцију *Count*
- Извршити упит (!)

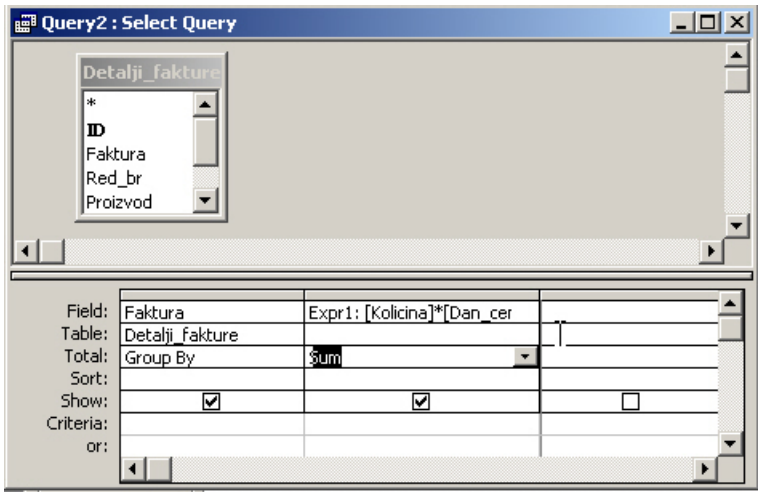
GROUP BY клаузула

Користи се за груписање резултата упита у мање скупине, односно за добијање збирних информација за сваку групу. Увек се користи у комбинацији са агрегатним функцијама.

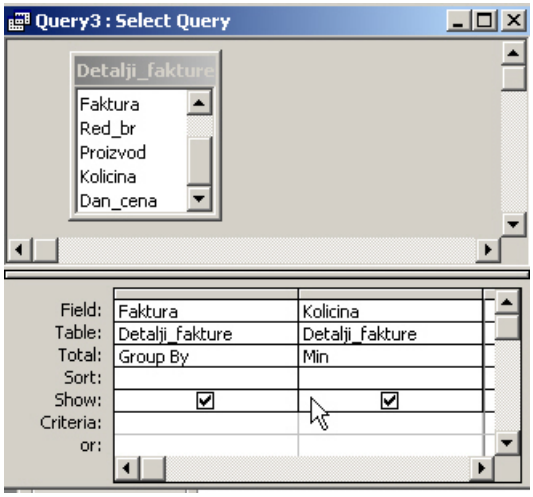
Биће урађено још неколико примера употребе агрегатних функција:

1. Израчунати укупне износе на свим фактурама
2. Пронаћи минималне количине на свакој фактури
3. Пронаћи ставку на свакој фактури која има максимални износ

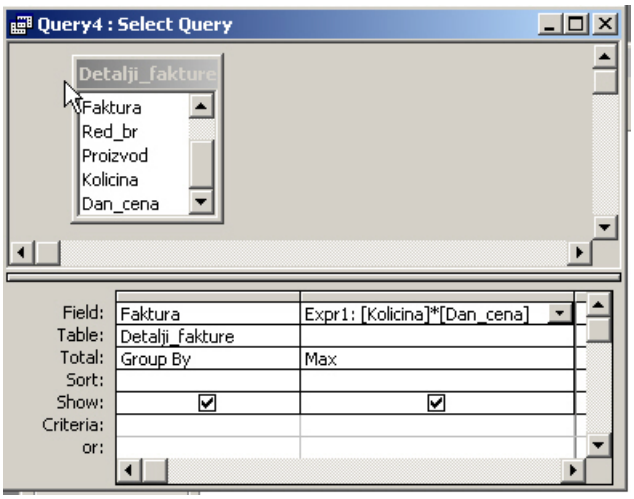
1. SELECT Faktura, Sum(Kolicina*Dan_cena)
   FROM Detalji_fakture GROUP BY Faktura;



2. SELECT Faktura, Min(Kolicina)
   FROM Detalji_fakture GROUP BY Faktura;



3. SELECT Faktura, Max(Kolicina*Dan_cena)
   FROM Detalji_fakture GROUP BY Faktura;



Ако SELECT листа садржи агрегатну функцију а нема GROUP BY клаузуле која би груписала податке, онда ниједан елемент SELECT листе не може садржавати било какву референцу на поље (колону), осим ако је поље аргумент у агрегатној функцији.

Пример погрешног упита:

SELECT Faktura, Max(Kolicina*Dan_Cena), Proizvod
FROM Detalji_fakture GROUP BY Faktura;

Изрази

Изрази се у Access-у користе за вршење израчунавања, манипулацију карактерима и тестирање података.

Access анализира израз сваки пут када се користи. Ако је израз садржан у извештају или обрасцу, Access израчунава вредност сваки пут када се извештај или образац користи. Овим се осигурава тачност података. Ако се израз користи као критеријум у упиту, он се израчунава сваки пут када се покреће упит.

Изрази могу бити једноставни или комплексни. Они могу садржати комбинацију оператора, имена објеката, функција, литерарних вредности и константи.
Изрази не морају саржати све наведене делове али је потребно разумети сваки од њих:
Оператори – Оператори указују на то који тип акције (или операције) ће се извршити над једним или више елемената израза.
Имена објеката – Имена објеката, позната и као идентификатори, су стварни објекти: табеле, обрасци, извештаји, контроле или поља. (Forms![Filmovi], [Naziv], [Kolicina]).
Функције – Функције увек враћају вредност. Резултујућа вредност може бити из израчунавања, конверзије података, итд. Могу се користити функције уграђене у сам Access или функције које дефинише сам корисник.
Литерарне вредности – Ово су стварне вредности које се прослеђују изразима. Ово могу бити бројеви, стрингови или датуми. Access користи ове вредности тачно онако како су унесене. (100, 10.12.2005, “kg”).
Константе – Константе представљају вредности које се не мењају (Yes, No, Null, True, False).

Креирање израза

Изрази се често уносе у поља за својства, аргументе акција и поља за критеријуме код упита. У току уношења израза Access може сам да додаје неке симболе када препозна неке типове података. Access стално проверава синтаксу и уноси следеће симболе:

- Заграде ([]) око имена контрола које у имену немају размаке или знаке интерпункције
- Знак # око датума које препозна
- Знаке навода (“ ”) око текста који нема размаке и знаке интерпункције у телу

Access уноси ове симболе након што корисник напусти поље у коме је вршио уношење израза. У овом случају је могуће да Access пријави грешку.

Уношење имена објеката

Имена објеката се идентификују постављањем угластих заграда око имена. Access захтева коришћење заграда када објекат у свом имену садржи размаке или знаке интерпункције. Ако размаци и знаци интерпункције нису присутни није потребно (али је препоручљиво) користити заграде – Access то ради аутоматски.

Уношење текста

Постављање знакова навода око унетих карактера их дефинише као текст. Ако се заборави да се ово уради Access то ради аутоматски.

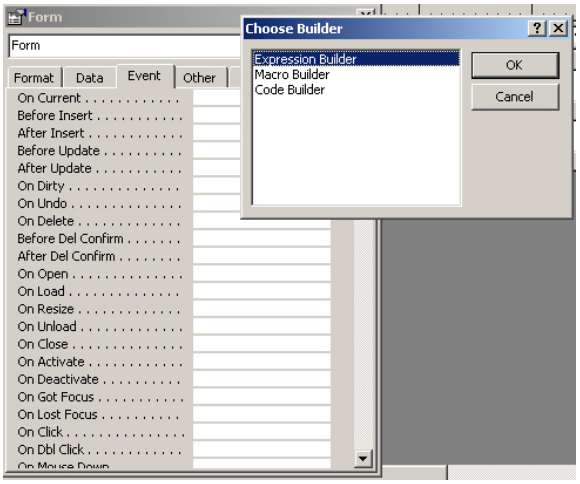
Уношење датума и времена

Око ових вредности се поставља знак #. Ако корисник не постави знакове Access то ради аутоматски.

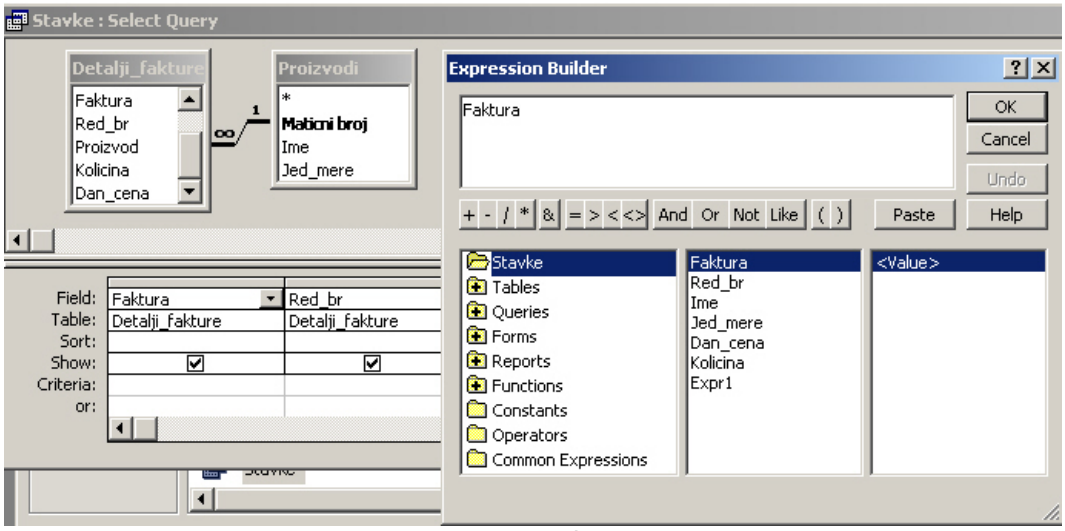
Expression Builder – алат за лакше креирање израза

Access има алат **Expression Builder** који помаже кориснику у прављењу комплексних израза. Може се користити свуда где је могуће правити израз и то:

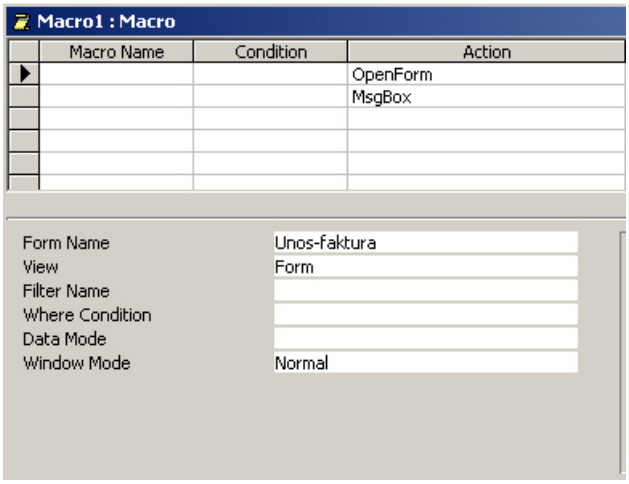
1. У табелама за дефинисање подразумеване вредности (*Default Value*) и дефинсање правила за проверу ваљаности (*Validation Rule*)
2. У обрасцима у својствима контрола која су груписана на картици Event оквира за дијалог Properties



3. У упитима у реду Field решетке за пројектовање када је поље које се жели да се види у резултату сложено и у реду Criteria за креирање сложених критеријума, односно израза

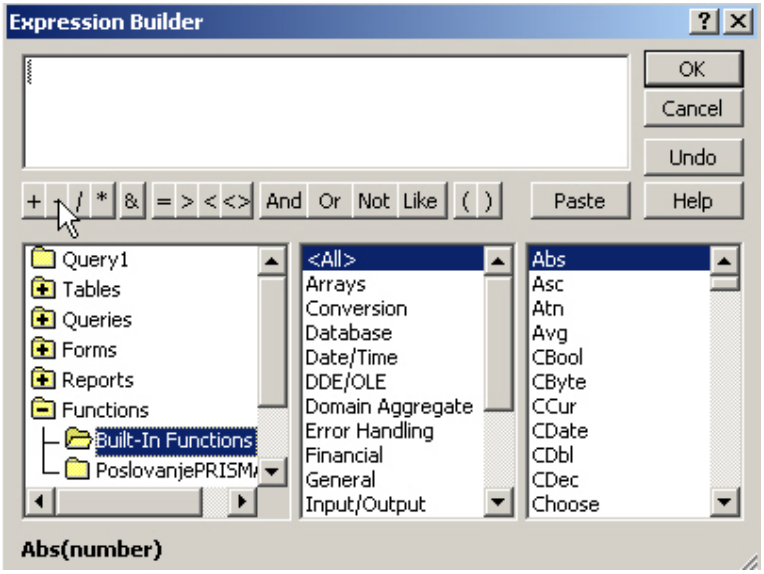


4. У извештајима (важи исто што и за обрасце)
5. У макроима – код свих макроа у колони Condition и код појединих макроа на местима где је потребно креирати изразе (на пример, акција OpenForm својство Where Condition)



Активирање генератора израза (**Expression Builder**) могуће је на два начина:

- Притиском на **Build** дугме ( ) на палети алатки Access -а
- Притиском десним тастером миша и избором **Build ...** команде из менија који се појављује



Access има два специјална идентификатор оператора: тачку (.) и знак узвика (!).

! идентификатор оператор (знак узвика)
Знак узвика (!) се користи у комбинацији са неколико резервисаних речи. Једна од њих је реч Forms. Када је ова реч праћена знаком ! Access -у се каже да је име објекта који следи у ствари име обрасца који се жели референцирати. Рецимо да постоји поље Datum rodjenja у два обрасца – [Studenti] и [Profesori] (имена образаца су у угластим заградама пошто су обрасци објекти базе података) и да се жели да се референцира поље Datum rodjenja у обрасцу [Profesori]. Начин да се ово уради јесте да се користи знак ! и резервисана реч Forms:
Forms![Profesori]
Сада када је образац специфициран треба само додати име поља Datum rodjenja.

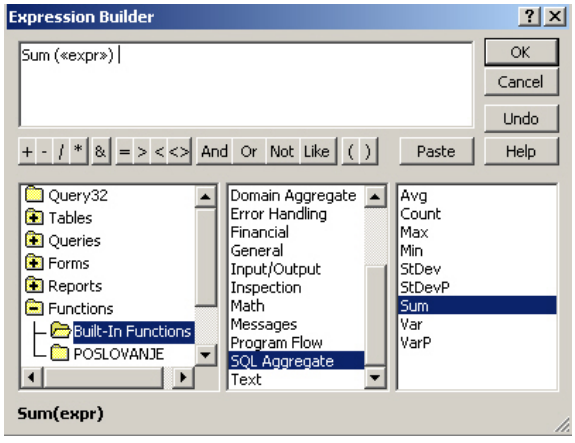
У ствари, овде се специфицира контрола на обрасцу. Контрола ће користити назначено поље (Datum rodjenja). Контрола има исти назив као и поље. Према томе, приступа се одређеном објекту коришћењем следећег израза:
Forms![Profesori]![Datum rodjenja]
Други знак узвика специфицира контролу на обрасцу (оном који је идентификован помоћу резервисане речи Forms).
Може се рећи да иза идентификатор оператора ! увек следи име објекта. Име објекта може бити назив обрасца, извештаја, поља или друге контроле која је направљена у бази.

. идентификатор оператор (тачка)

Обично се овај оператор смешта одмах иза објекта кога је дефинисао корисник. Овај оператор обично идентификује својство одређеног објекта. Тако, ако се жели да се одреди вредност својства Visible контроле која је коришћена у претходном примеру треба употребити следећи израз:
Forms![Profesori]![Datum rodjenja].Visible
Ово даје вредност својства Visible одређеног поља на одређеном обрасцу.
Понекад се овај оператор користи између назива табеле и њеног поља када се приступа вредности из одређеног поља одређене табеле као што је:
[Studenti].[Datum rodjenja].

Вежба. Написати упит који који приказује бројеве фактура и укупне износе ставки на тим фактурама, такве да је укупан износ већи од 10000 (пример рађен на претходном часу али без употребе **Expression Builder**-а)

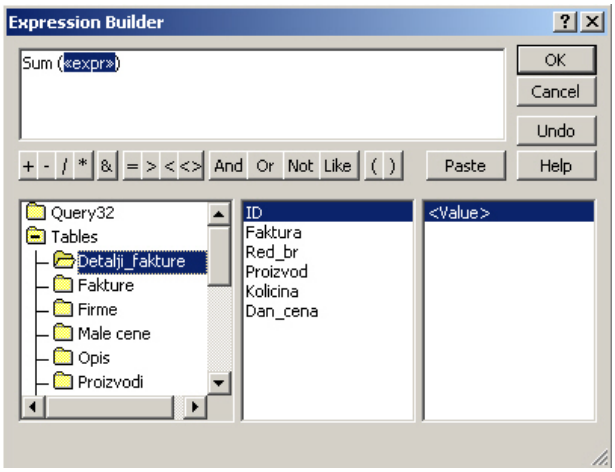
- Отворити базу података Poslovanje
- На траци са објектима изабрати Queries и након тога два пута брзо кликнути левим тастером миша на *Create query in Design view*
- Из листе са табелама изабрати табелу Detalji_fakture
- Пребацити колону Fakture у решетку за пројектовање да бу се у резултату приказао број фактуре
- Поставити показивач у другу колону решетке за пројектовање у ћелију која се налази реду Field и притиснути леви тастер миша. Након тога кликнути на дугме **Build** () на палети алатки



- У доњем левом прозору изабрати **Functions** (двоструки клик) → **Built-In Functions** (један клик)
- У средњем прозору изабрати SQL Aggregate (један клик) (пошто је потребан укупан износ, неопходно је употребити функцију сумирања која је агрегатна функција)
- Потом изабрати функцију **Sum** из десног прозора (двоструки клик)

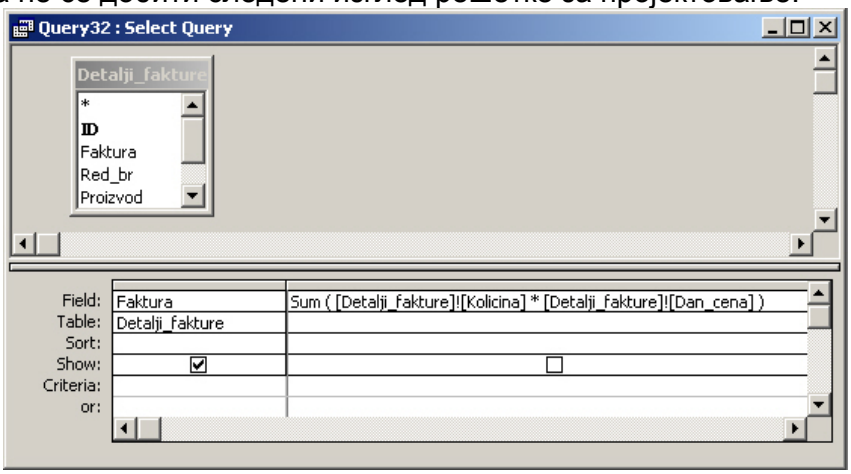
На место израза <<expr>> потребно је уписати израз који ће се сумирати, а то је у овом случају производ количине и цене. Искористиће се овај алат за бирање ових поља уместо куцања њихових назива

- Кликнути на израз <<expr>> и он ће бити селектован



- У доњем левом прозору изабрати **Tables** (двоструки клик) и из листе табела изабрати **Detalji_fakture** (један клик)
- Из средњег прозора изабрати поље **Kolicina** (двоструки клик) и потом кликнути на дугме * (операција множења), након чега ће се појавити израз *Sum ([Detalji_fakture]![Kolicina] * «Expr»)*
- Кликнути на израз <<Expr>> и он ће бити селектован
- Из средњег прозора изабрати поље **Dan_cena** (двоструки клик) и кликнути на дугме **OK**

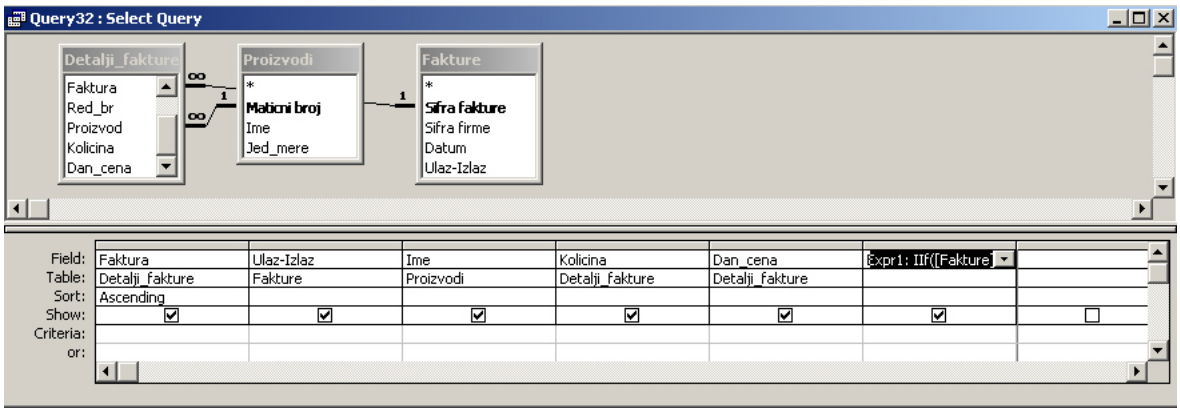
Након овога ће се добити следећи изглед решетке за пројектовање:



- На палети алатки Access-а притиснути дугме **Totals** (Σ). У решетки за пројектовање се појављује ред Totals.
- У реду Totals за колону Faktura оставити подразумевану вредност Group By. У истом реду за новоформирану колону из падајуће листе изабрати Expression.
- Потребно је још у колони у којој се врши сумирање у ћелији за критеријум (Criteria) уписати израз >10000

Вежба. Направити упит који приказује све ставке на фактури. Потребно је приказати број фактуре (уређено у растућем редоследу), тип фактуре (да ли је улаз или излаз), име производа, количину, цену и износ, при чему износе

приказивати као позитивне бројеве уколико је фактура излазна (нешто је продато и очекује се прилив новца за то), односно негативне за случај улазних фактуре (износ се дугује).
Изглед Design приказа упита:

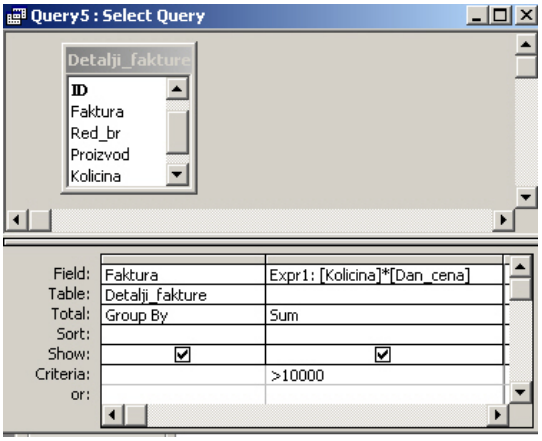


Функција IIf се налази у групи функција Program Flow. Синтакса ове функције је IIf(expr; truepart; falsepart). Ова функција прима три аргумента. Први аргумент је израз који се тестира. Други аргумент је вредност или израз који се враћају ако је израз који се тестира тачан. Трећи аргумент је вредност или израз који се враћају ако је израз који се тестира нетачан. Аргументи су раздвојени знаком ;

Изглед израза:
Expr1: IIf([Fakture]![Ulaz-Izlaz]="2";
[Detalji_fakture]![Dan_cena]*[Detalji_fakture]![Kolicina];
-[Detalji_fakture]![Dan_cena]*[Detalji_fakture]![Kolicina])
Вежба. Уместо бројева 1 и 2 за поље **Ulaz-Izlaz** треба да се појављују речи Ulaz односно Izlaz.
Изглед израза:
Expr2: IIf([Fakture]![Ulaz-Izlaz]="1";"Ulaz";"Izlaz")

HAVING клаузула

HAVING клаузула се користи када се жели да се специфицира које групе треба приказати, односно, користи се за рестрикцију група које се приказују, тј. за испитивање вредности агрегатних функција.
У услову за HAVING клаузулу може да се укључи нека агрегатна фукција или колона из Group By. HAVING може да се користи и без Group By, односно без употребе агрегатних функција, али је то неубичајено.
Вежба. Нађи све фактуре чији је укупан износ већи од 10000




```
SELECT Detalji_fakture.Faktura,
Sum([Detalji_fakture]![Kolicina]*[Detalji_fakture]![Dan_cena]) AS Expr1
FROM Detalji_fakture
GROUP BY Detalji_fakture.Faktura
HAVING (((Sum([Detalji_fakture]![Kolicina]*[Detalji_fakture]![Dan_cena]))>10000));
```

ORDER BY клаузула

Технике сортирања које се користе над табелом могу се користити и над упитима који су приказани у *Datasheet* приказу. Ако се жели да се креирају мало сложенији упити, SQL пружа могућност да се сортирају поља у упиту и користи то уређење као део упита

Сортирање у *Design* приказу

Вежба. Направити упит који приказује све фирме тако да су им називи уређени растуће по абецеди.

- отворити базу података Poslovanje
- на траци са објектима изабрати *Queries*
- два пута брзо кликнути левим тастером миша на *Create query in Design view*
- из оквира за дијалог Show Table изабрати табелу Firme
- пребацити колоне Firma и Naziv_firme у решетку за пројектовање
- кликнути на ћелију у колони Naziv_firme која се налази у реду ***Sort*** и потом изабрати опцију *Ascending* из падајуће листе (сортирање по ратућем редоследу). (Такође је могуће извршити сортирање по опадајућем редоследу избором опције *Descending*).
- Извршити упит ( )

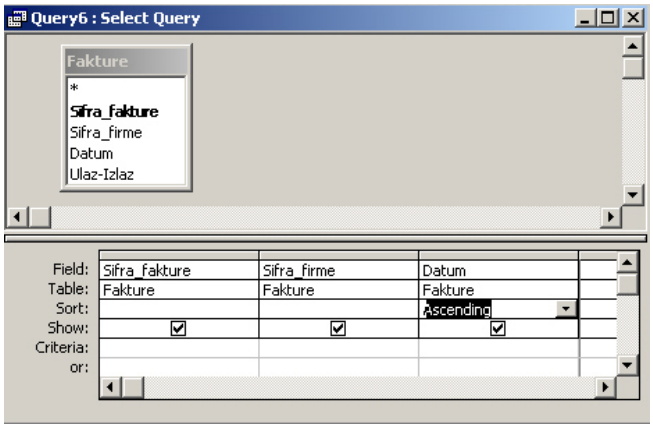
Вежба. Извршити сортирање у претходно креираном упиту по колони Ime.

Сортирање у SQL-у

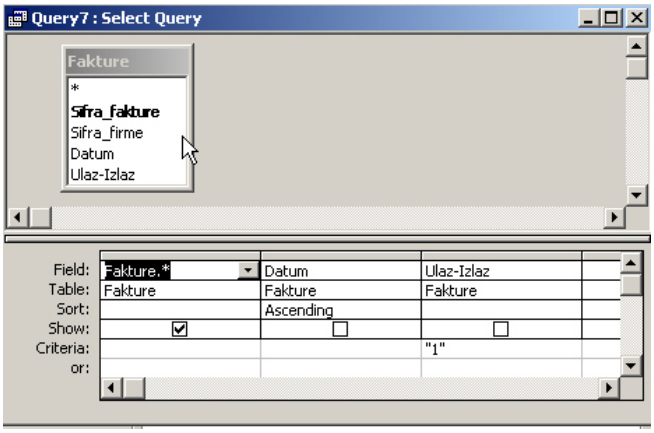
Сортирање у SQL-у се може извршити коришћењем **ORDER BY** клаузуле.

Вежба. Извршити следеће упите:

```
SELECT Fakture.Sifra_fakture, Fakture.Sifra_firme, Fakture.Datum
FROM Fakture
ORDER BY Fakture.Datum;
```




```
SELECT Fakture.*
FROM Fakture
WHERE (((Fakture.[Ulaz-Izlaz])="1"))
ORDER BY Fakture.Datum;
```



Подразумевани начин сортирања је у растућем редоследу, мада постоји службена реч која контролише сортирање у растућем редоследу. То је **ASC**.

```
SELECT * FROM Fakture ORDER BY Datum ASC
```

Ако се жели да се изврши сортирање у опадајућем редоследу онда се користи службена реч **DESC**.

```
SELECT * FROM Fakture ORDER BY Datum DESC
```

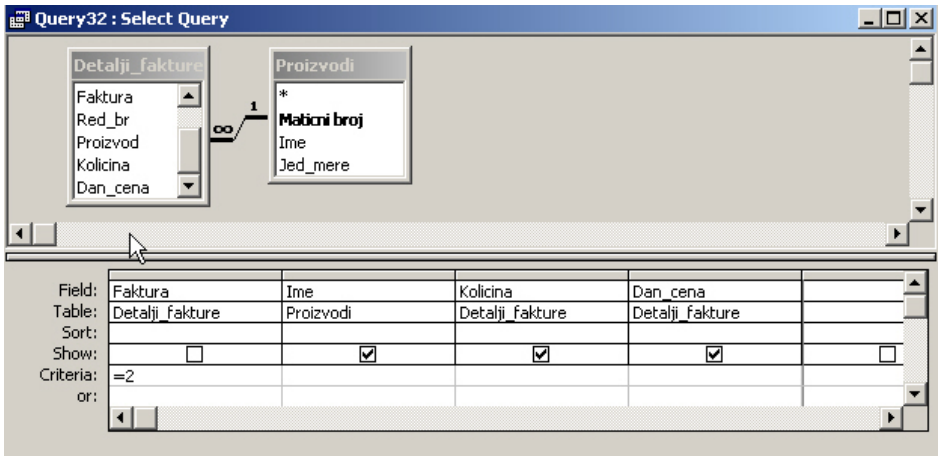
Сложени упити 1

Када се успоставља релација између табела, повезује се примарни кључ једне табеле са страним кључем друге табеле. Када је таква релација успостављена, може се креирати упит који комбинује податке те две табеле да би се добили жељени резултати. Укључивањем више табела у упит омогућено је да се подаци из више табела виде у формату једне.

Вежба. Написати упит који приказује све ставке на некој фактури (нпр. фактури број 2), тако да се виде: назив производа, количина и цена

- Отворити базу података **Poslovanje**
- На траци са објектима изабрати *Queries*
- Двапут брзо кликнути левим тастером миша на *Create query in Design view*
- Из оквира за дијалог *Show Table* изабрати табеле **Detalji_fakture** и **Proizvodi**

- Пребацити жељена поља у решетку за пројектовање и поставити критеријум над колоном **Faktura** (из текста задатка се види да је потребно пребацити поље **Ime** из табеле **Proizvodi** и поља **Kolicina** и **Dan_cena** из табеле **Detalji_fakture**; поље **Faktura** из табеле **Detalji_fakture** је пребачено у решетку за пројектовање да би било употребљено за постављање критеријума упита и може се видети да се вредности из овог поља не приказују у резултатима упита)




```
SELECT Proizvodi.Ime, Detalji_fakture.Kolicina, Detalji_fakture.Dan_cena
FROM Proizvodi INNER JOIN Detalji_fakture ON Proizvodi.Maticni_broj =
Detalji_fakture.Proizvod
WHERE (((Detalji_fakture.Faktura)=2));
```

Query32 : Select Query

Ime	Kolicina	Dan_cena
_ak	36.00	11.00
Stiropor	50.00	23.00
Gips	50.00	25.00

Када се изаберу табеле, Access приказује тзв. линију повезивања. То је графичка репрезентација релације између табела. Ова се линија приказује аутомтски пошто је релација већ успостављена. Уколико релација није успостављена, Access може да је сам успостави приликом прављења упита ако је задовољено следеће:

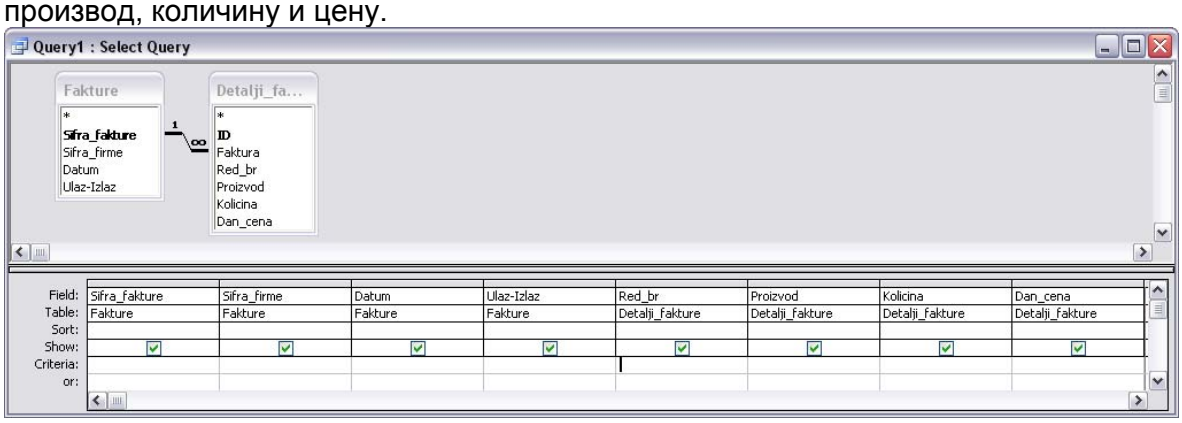
1. У табелама постоје поља која имају исти назив
2. Та поља садрже исти тип података и имају исту велчину поља
3. Једно од поља је примарни кључ у једној табели

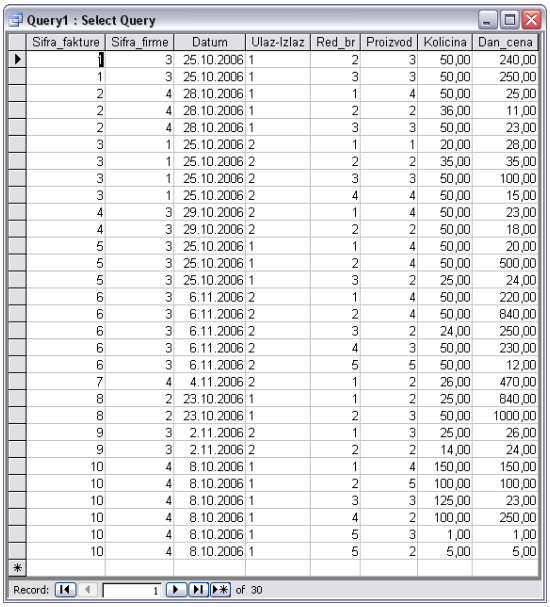
Типови спајања табела

1. INNER JOIN (унутрашњи спој)
2. FULL OUTER JOIN (пун спољашњи спој)
3. LEFT OUTER JOIN (леви спољашњи спој)
4. RIGHT OUTER JOIN (десни спољашњи спој)

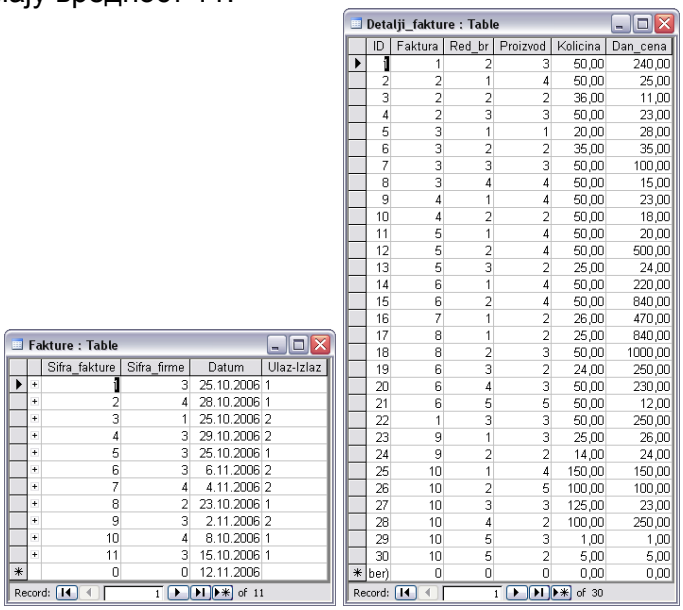
INNER JOIN – Ово је подразумевани спој у Access-у. Он каже Access-у да изабере све записе из обе табеле који имају исту вредност у пољима која су спојена. Ако постоје записи из било које табеле који немају записе са којима се преко вредности у заједничком пољу поклапају у другој табели они се искључују из резултујућег dynaset-a и неће бити приказани у листу са подацима.

FULL OUTER JOIN – Овим спајањем се селектују сви записи који немају одговарајуће записе са којима су повезани. За ову врсту спајања не постоји имплентација у Access-у.
LEFT OUTER JOIN и RIGHT OUTER JOIN – За разлику од унутрашњег споја, спољашњи спојеви се користе за приказивање свих записа из једне табеле и само повезаних записа из друге табеле. Тамо где неки запис нема повезани запис ће једноставно бити празних поља у листу са подацима. Када се формира спољашњи спој линија споја ће графички показивати на једну од табела (стрелица). Биће приказани сви записи из табеле која нема стрелицу и само повезани записи из табеле која има стрелицу. У зависности да ли стрелица показује на десну или леву страну говоримо о десном или левом спољашњем споју.
Својства споја се могу изменити десним притиском мишем на линију споја у Design приказу упита и избором **Join Properties** из shortcut менија. Тада се отвара **Join Properties** дијалог прозор.
Овај дијалог прозор има два дела: четири combo box-а и три опциона дугмета. Прва опција је обично позната као inner-join (унутрашњи спој), а друге две као outer-joins (спољашњи спојеви). Ови спојеви контролишу понашање Access-а када гради dynaset након извршавања упита.
Вежба. Приказати ставке за све фактуре. У резултатима упита је потребно приказати шифру фактуре, шифру фирме, датум, тип фактуре, редни број ставке, производ, количину и цену.





На овај начин се приказују сви записи из табела **Fakture** и **Detalji_fakture** који имају исту вредност у пољима **Sifra_fakture** и **Faktura**. Међутим, ако се погледају у **Datasheet** приказу табеле **Fakture** и **Detalji_fakture** видеће се да табела **Fakture** садржи податке за фактуру са шифром 11, а да у табели **Detalji_fakture** нема ставки за ту фактуру, односно нема записа који у пољу **Faktura** табеле **Detalji_fakture** имају вредност 11.



Fakture : Table

Sifra_fakture	Sifra_firme	Datum	Ulaz-Izlaz
1	3	25.10.2006	1
2	4	28.10.2006	1
3	1	25.10.2006	2
4	3	29.10.2006	2
5	3	25.10.2006	1
6	3	6.11.2006	2
7	4	4.11.2006	2
8	2	23.10.2006	1
9	3	2.11.2006	2
10	4	8.10.2006	1
11	3	15.10.2006	1
0	0	12.11.2006	

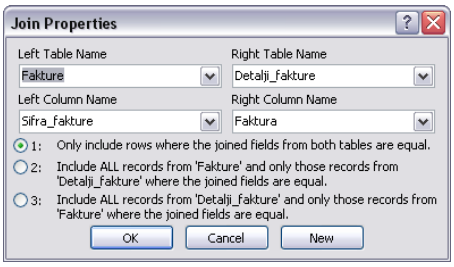
Record: 1 of 11

Detalji_fakture : Table

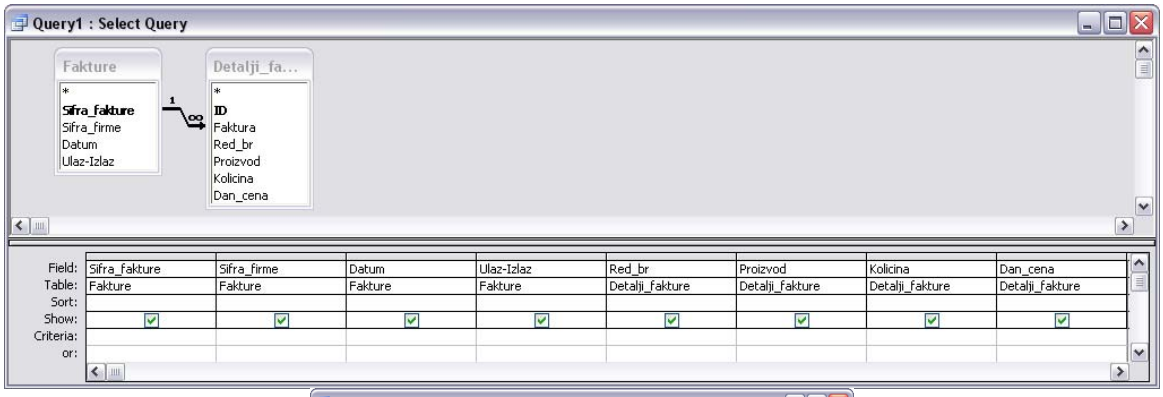
ID	Faktura	Red_br	Proizvod	Kolicina	Dan_cena
1	1	2	3	50,00	240,00
2	2	1	4	50,00	25,00
3	2	2	2	36,00	11,00
4	2	3	3	50,00	23,00
5	3	1	1	20,00	28,00
6	3	2	2	35,00	35,00
7	3	3	3	50,00	100,00
8	3	4	4	50,00	15,00
9	4	1	4	50,00	23,00
10	4	2	2	50,00	18,00
11	5	1	4	50,00	20,00
12	5	2	4	50,00	500,00
13	5	3	2	25,00	24,00
14	6	1	4	50,00	220,00
15	6	2	4	50,00	840,00
16	7	1	2	26,00	470,00
17	8	1	2	25,00	840,00
18	8	2	3	50,00	1000,00
19	6	3	2	24,00	250,00
20	6	4	3	50,00	230,00
21	6	5	5	50,00	12,00
22	1	3	3	50,00	250,00
23	9	1	3	25,00	26,00
24	9	2	2	14,00	24,00
25	10	1	4	150,00	150,00
26	10	2	5	100,00	100,00
27	10	3	3	125,00	23,00
28	10	4	2	100,00	250,00
29	10	5	3	1,00	1,00
30	10	5	2	5,00	5,00
ber)	0	0	0	0,00	0,00

Record: 1 of 30

Ако се жели да се у резултатима упита прикажу сви записи из табеле **Fakture** и само повезани записи из табеле **Detalji_fakture** потребно је променити својства споја између ових табела. Треба кликнути десним тастером миша на линију спајања и изабрати *Join Properties* из shortcut менија. Добија се следећи прозор:



Прва опција у доњем делу прозора дефинише INNER JOIN, друга RIGHT JOIN, а трећа LEFT JOIN. Треба изабрати другу опцију и након тога притиснути дугме OK. Сада Design приказ упита и резултати упита изгледају овако:



Query1 : Select Query

Sifra_fakture	Sifra_firme	Datum	Ulaz-Izlaz	Red_br	Proizvod	Kolicina	Dan_cena
1	3	25.10.2006	1	2	3	50,00	240,00
1	3	25.10.2006	1	3	3	50,00	250,00
2	4	28.10.2006	1	1	4	50,00	25,00
2	4	28.10.2006	1	2	2	36,00	11,00
2	4	28.10.2006	1	3	3	50,00	23,00
3	1	25.10.2006	2	1	1	20,00	28,00
3	1	25.10.2006	2	2	2	35,00	35,00
3	1	25.10.2006	2	3	3	50,00	100,00
3	1	25.10.2006	2	4	4	50,00	15,00
4	3	29.10.2006	2	1	4	50,00	23,00
4	3	29.10.2006	2	2	2	50,00	18,00
5	3	25.10.2006	1	1	4	50,00	20,00
5	3	25.10.2006	1	2	4	50,00	500,00
5	3	25.10.2006	1	3	2	25,00	24,00
6	3	6.11.2006	2	1	4	50,00	220,00
6	3	6.11.2006	2	2	4	50,00	840,00
6	3	6.11.2006	2	3	2	24,00	250,00
6	3	6.11.2006	2	4	3	50,00	230,00
6	3	6.11.2006	2	5	5	50,00	12,00
7	4	4.11.2006	2	1	2	26,00	470,00
8	2	23.10.2006	1	1	2	25,00	840,00
8	2	23.10.2006	1	2	3	50,00	1000,00
9	3	2.11.2006	2	1	3	25,00	26,00
9	3	2.11.2006	2	2	2	14,00	24,00
10	4	8.10.2006	1	1	4	150,00	150,00
10	4	8.10.2006	1	2	5	100,00	100,00
10	4	8.10.2006	1	3	3	125,00	23,00
10	4	8.10.2006	1	4	2	100,00	250,00
10	4	8.10.2006	1	5	3	1,00	1,00
10	4	8.10.2006	1	5	2	5,00	5,00
11	3	15.10.2006	1				

Record: 1 of 31

Примећује се да задњи ред резултата упита садржи податке из табеле **Fakture** за фактуру са шифром 11, али да нема података из табеле **Detalji_fakture**, пошто у тој табели нема ставки за фактуру са шифром 11.

```
SELECT Fakture.Sifra_fakture, Fakture.Sifra_firme, Fakture.Datum, Fakture.[Ulaz-Izlaz], Detalji_fakture.Red_br, Detalji_fakture.Proizvod, Detalji_fakture.Kolicina, Detalji_fakture.Dan_cena
FROM Fakture LEFT JOIN Detalji_fakture ON Fakture.Sifra_fakture = Detalji_fakture.Faktura;
```

Вежба. Пронаћи ставке на свим фактурама код којих је количина * цена > 10000. Приказати број фактуре, датум, тип фактуре, количину, цену и укупан износ.

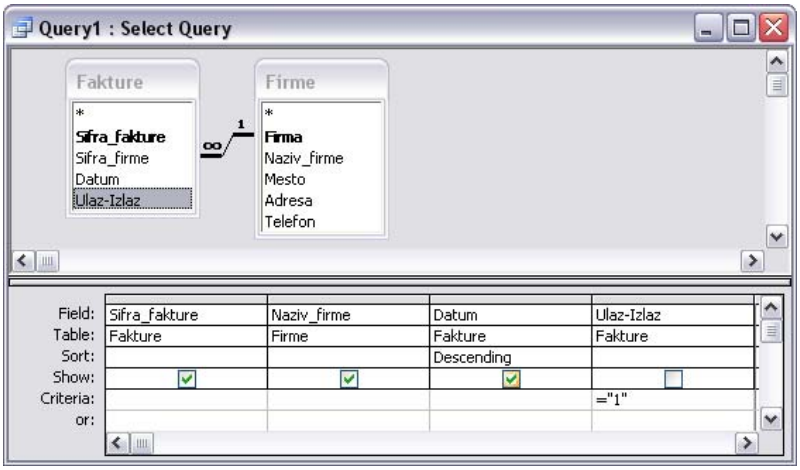
Dynaset

Access записе који представљају резултат упита приказује у листу са подацима, у коме се записи називају *Dynaset*. Физички *Dynaset* личи на табелу; у ствари он није табела. *Dynaset* је динамички (или виртуелан) скуп записа. Динамички скуп записа се не смешта у базу података.

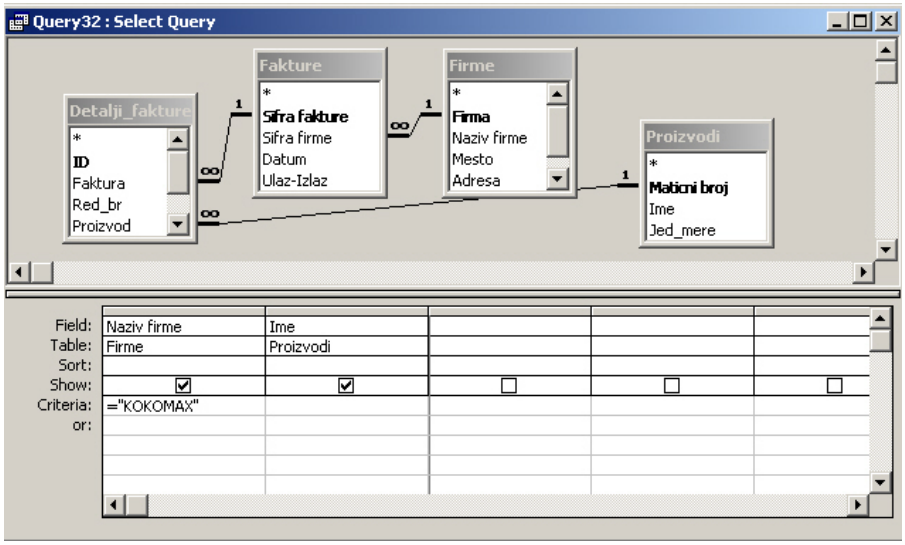
Када се упит затвори његов *Dynaset* више не постоји. Иако *Dynaset* више не постоји, подаци који су га сачињавали остају смештени у табелама базе података. Када се изврши упит Access смешта резултујуће записе у *Dynaset*. Када се упит сними, подаци се не снимају; само се структура упита (правила за извршавање упита) снима – табеле, поља, редослед сортирања, критеријуми, тип упита, итд.

Сложени упити 2

Вежба. Написати упит који приказује датуме свих **улазних** фактура заједно са бројевима фактура и називима фирми. Приказати резултате сортиране опадајуће по датуму.



Вежба. Пронаћи све производе које је купила нека фирма.



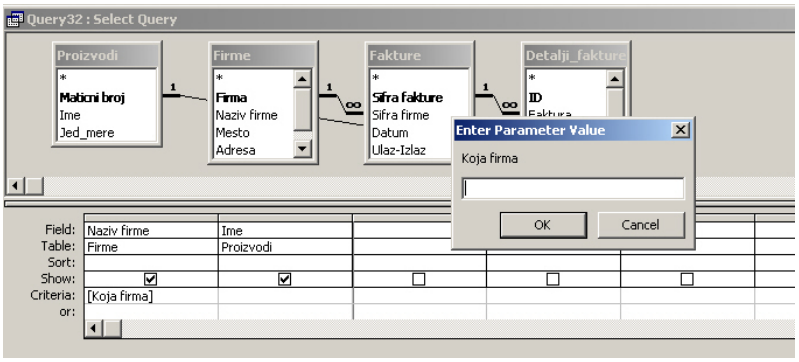
Резултат оваквог упита је веома непрегледан јер се производи понављају Преласком у SQL приказ, може да се дода DISTINCT предикат којим се елиминишу дупликати у резултатима упита

```
SELECT DISTINCT Firme.Naziv_firme, Proizvodi.Ime
FROM Proizvodi INNER JOIN (Firme INNER JOIN (Fakture INNER JOIN
Detalji_fakture ON Fakture.Sifra_fakture = Detalji_fakture.Faktura) ON Firme.Firma =
Fakture.Sifra_firme) ON Proizvodi.Maticni_broj = Detalji_fakture.Proizvod
WHERE (((Firme.Naziv_firme)="KOKOMAX"));
```

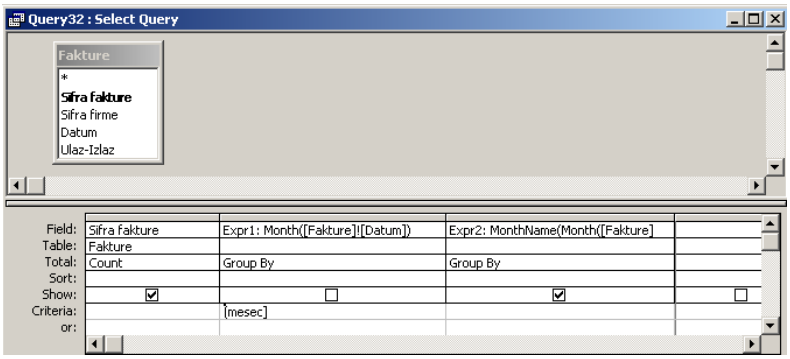
Параметарски упит

Уместо конкретног назива фирме могуће је уписати неко симболичко име, односно параметар упита. У том случају, приликом извршења упита, појављује се оквир за дијалог који очекује да се упише назив фирме, односно вредност параметра. На

тај начин је могуће искористи упит не само за проналажење производа за неку конкретну фирму, него за било коју фирму. Назив параметра увек стоји у угластим заградама. Назив параметра не сме бити исти као назив поља за које се примењује параметар. Упит може садржати више параметара.



Самостална вежба. Написати параметраски упит који приказује све фактуре неке фирме за жељени датум (два параметра: назив фирме и датум).
Вежба. Написати упит који приказује укупан број фактура за неки месец.



Напомене: Критеријум је постављен само за месец ради једноставности упита. Могуће је додати још један критеријум за годину. Вредност параметра је број жељеног месеца.
Функција **Month** као аргумент прима датум и враћа број месеца у години (1 до 12). Дефинисана функција: Month([Fakture]![Datum])
Функција **MonthName** као први аргумент прима број месеца у години (1 до 12) и враћа назив месеца. Не треба дефинисати други аргумент ове функције (abbreviate), тј. треба га обрисати, пошто је опциони аргумент. Дефинисана функција: MonthName(Month([Fakture]![Datum]))
Самостална вежба. Написати упит који приказује производе продате у неком граду. Приказати називе производа и укупне количине, уређене по количини од највеће ка најмањој.