Технички факултет Битола

СУБП и развојни алатки

1 Архитектура на системи на БП

доц. д-р Митко Костов

2012/13

Содржина

- Вовед
- Класификација
- Интегрирани решенија
- Слоевита архитектура на апликации
- Паралелни СУБП
- Дистрибуирани СУБП
- Технологии кои овозможуваат размена на податоци помеѓу БП и апликациите

Класификација

- Постојат бројни архитектури на системи на бази на податоци кои се користат.
- Тие може да се класифицираат според различни критериуми:
  - Каде се наоѓаат податоците, а каде СУБП?
  - Каде се извршуваат апликациските програми (на пример на која CPU)
  - Каде се извршуваат работните правила (*business rules*)
  - Каде се наоѓаме во историскиот развој

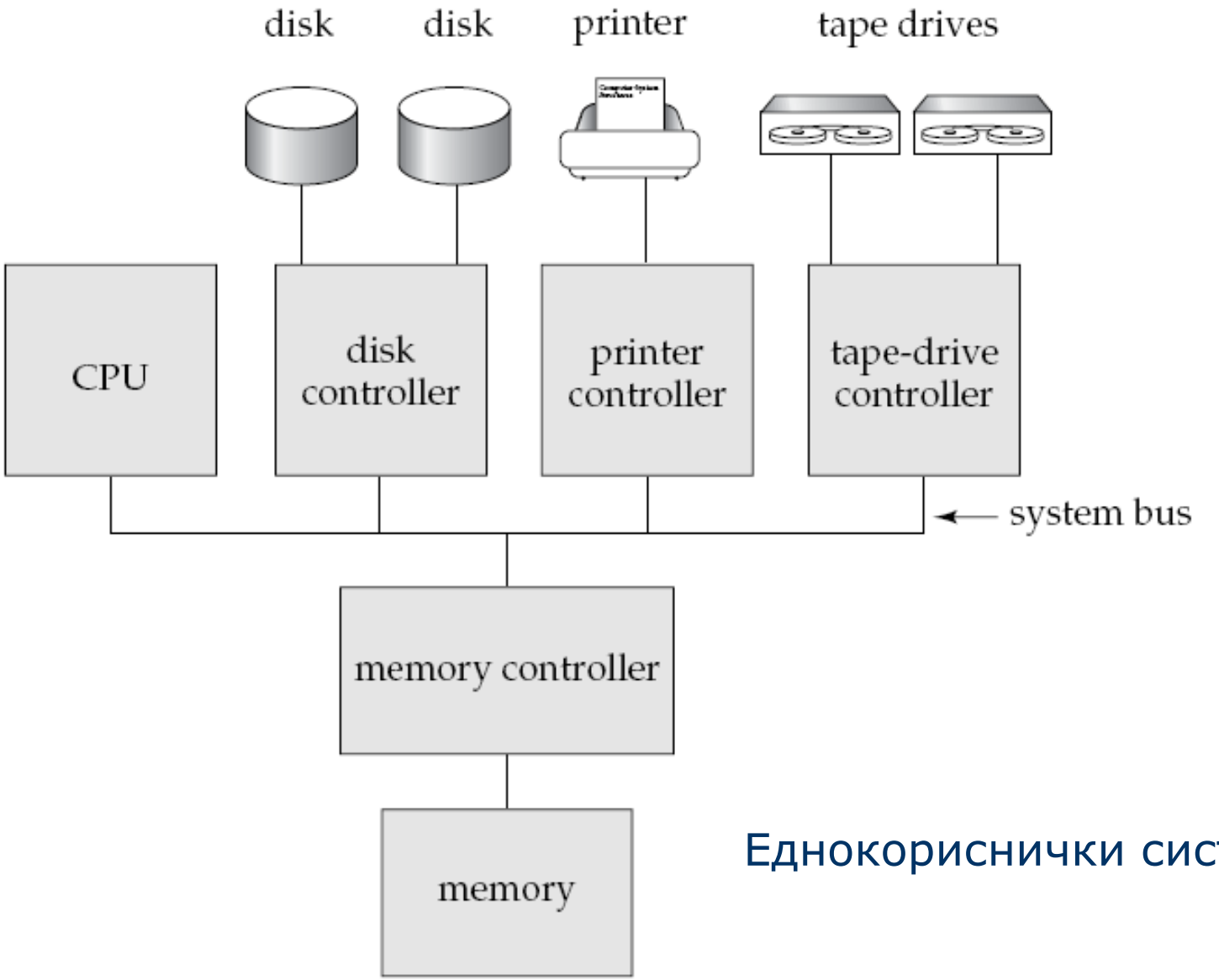
Класификација

- На архитекурата на СУБП влијае основниот компјутерски систем. Пр:
  - Вмрежувањето на компјутери
    - дозволува дел од задачите да бидат извршени на серверот, дел на станиците → клиент-сервер СУБП
  - Паралелно процесирање во рамките на еден систем
    - овозможува поголема брзина на обработката на податоците → паралелен СУБП
  - Дистрибуирани податоци
    - податоците се наоѓаат таму каде што се генерирани или потребни, но до нив може да се пристапи и од други сајтови → дистрибуиран СУБП

Класификација

1. Централизирани системи
   - Традиционална (mainframe) архитектура
   - PC самостојна база на податоци
2. Архитектура на делива датотека (file sharing architecture)
3. Клиент/сервер архитектури
4. Паралелни архитектури
5. Архитектура на дистрибуирани бази на податоци (distributed database)
   - Поделба на податоците (data partitioning)
   - Повторување на податоците (data replication)
6. Отворени архитектури (open database connectivity)
7. WEB ориентирани архитектури

Централизиран компјутерски систем



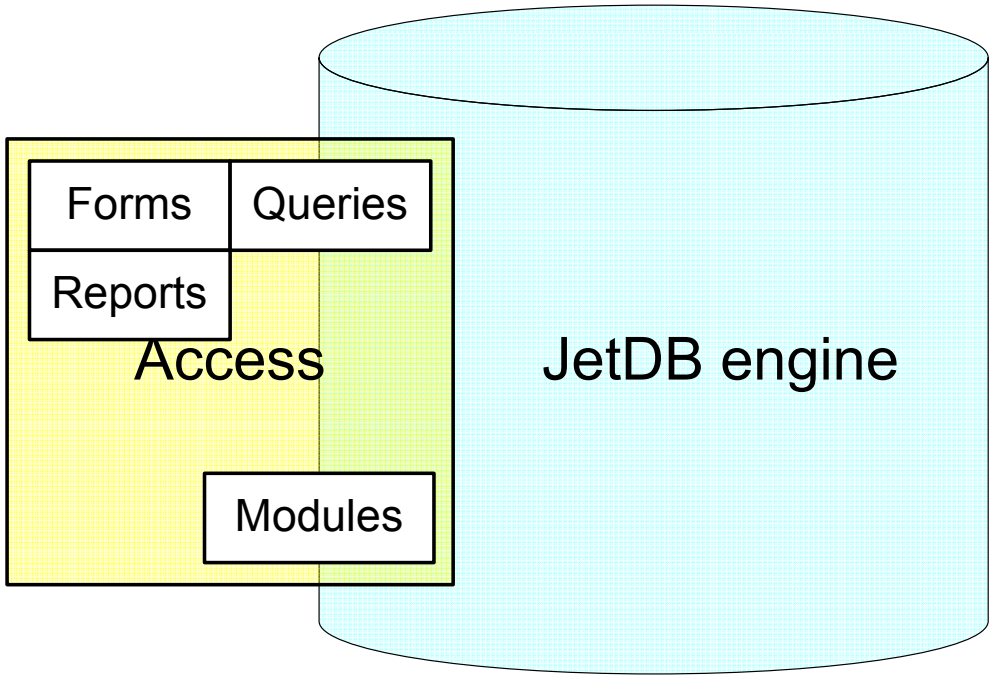
Еднокорисnички систем

Централизиран компјутерски систем

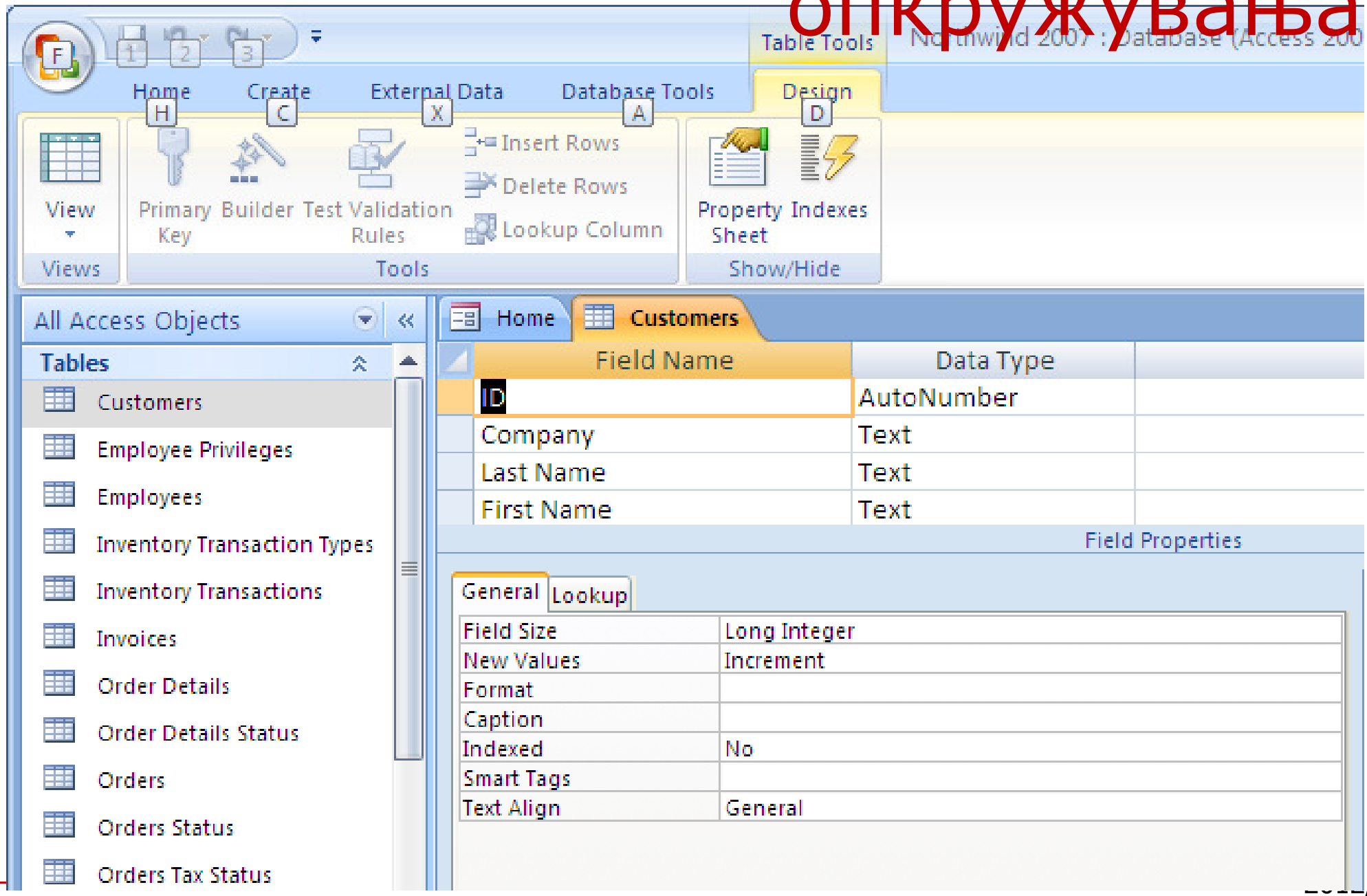
- Претставува еден компјутерски систем и нема интеракција со други компјутерски системи
- Еден до неколку CPU-и и одреден број на контролери поврзани на заедничка магистрала која овозможува пристап до делена меморија.
- Се разликуваат два начини на користење на компјут.:
  - Како еднокориснички систем
    - PC или работна станица.
    - Еден корисник, најчесто еден CPU, еден или два диска; ОС поддржува само еден корисник.
  - Како повеќекориснички систем
    - Опслужува поголем број на корисници поврзани преку терминали.
    - Повеќе дискови, повеќе меморија, повеќе CPU-и, повеќекориснички ОС.

Интегрирани опкружувања

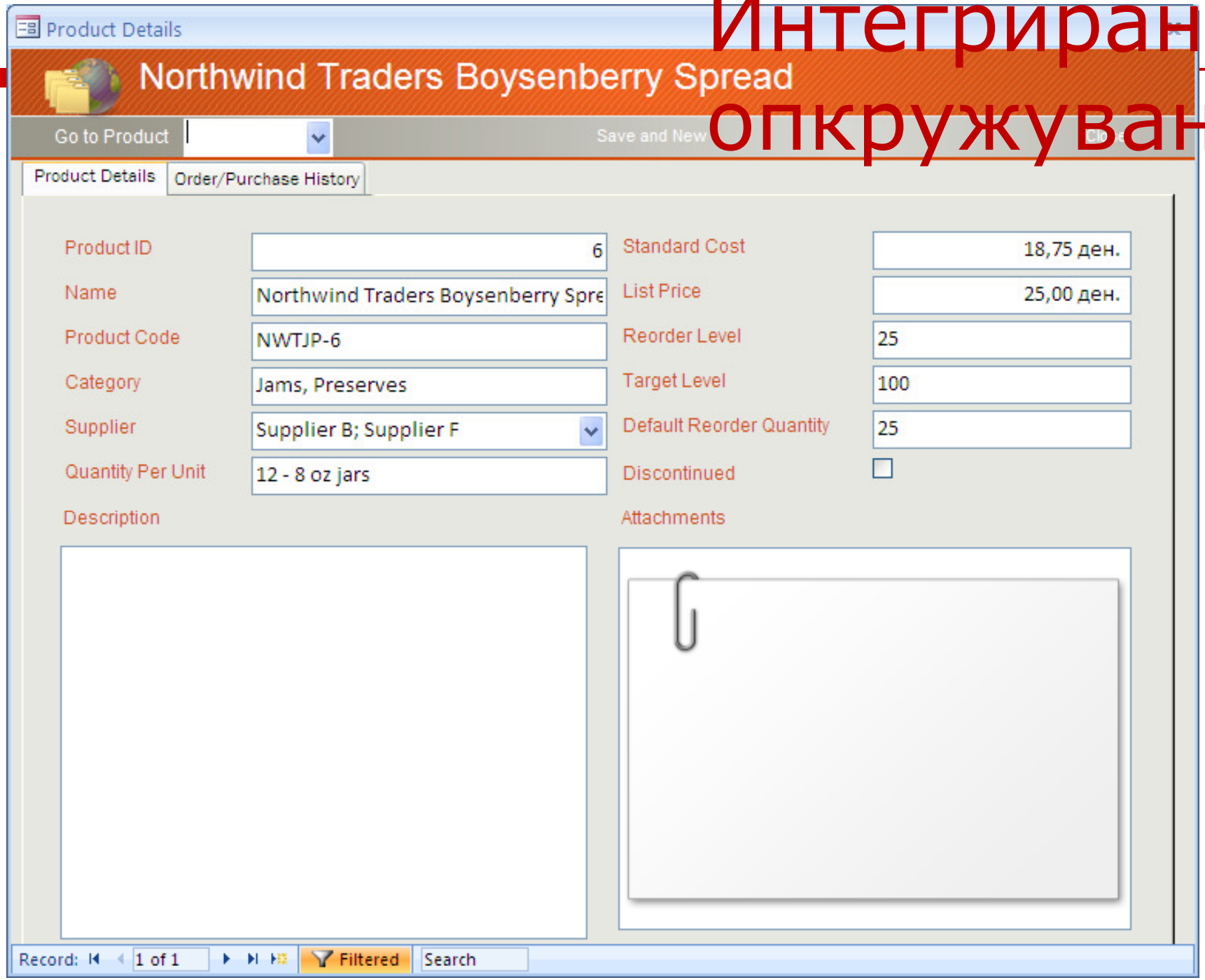
- Платформска зависност
- Пример Access и MS JetDB
- Сè е во една датотека



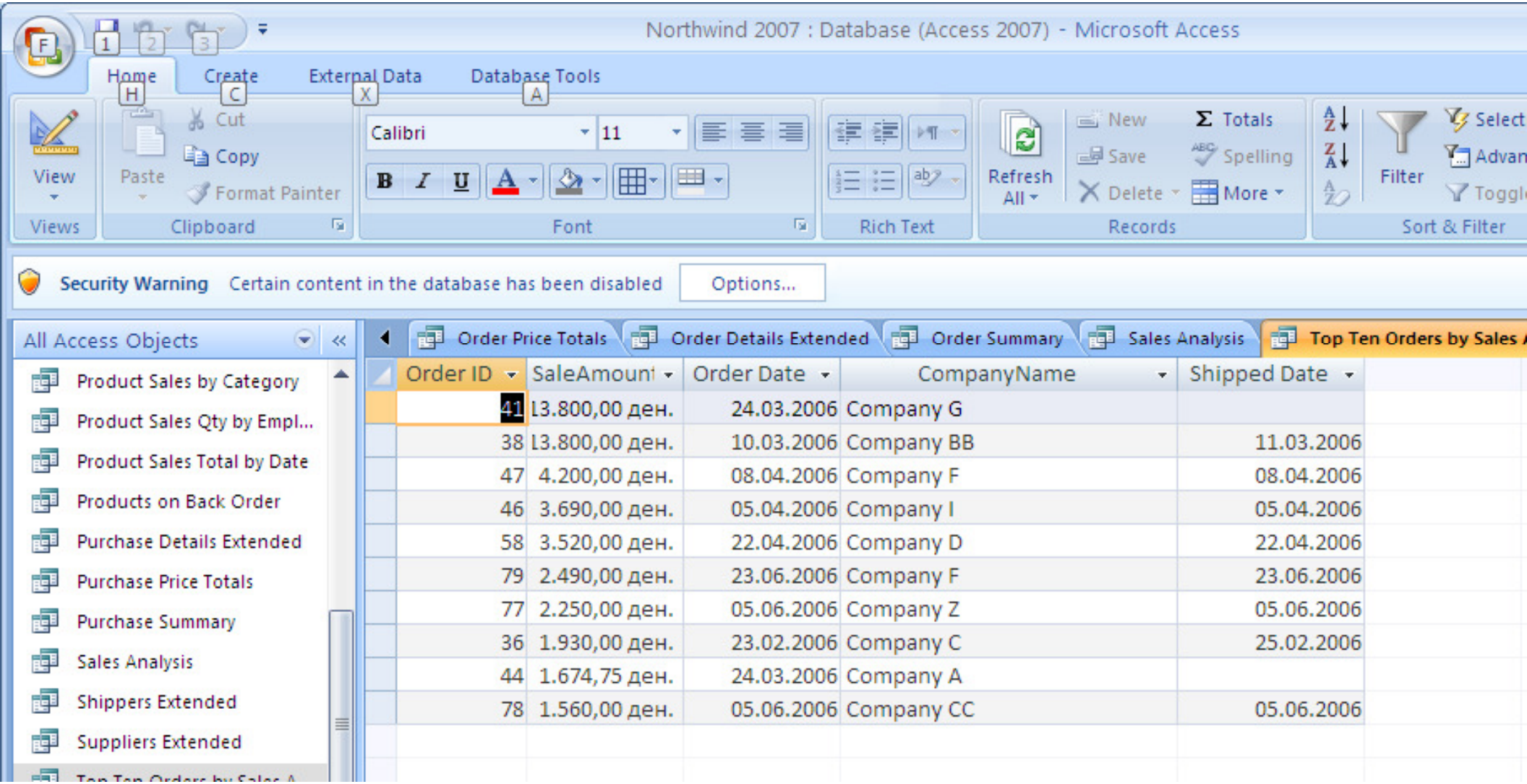
Интегрирани опкружувања



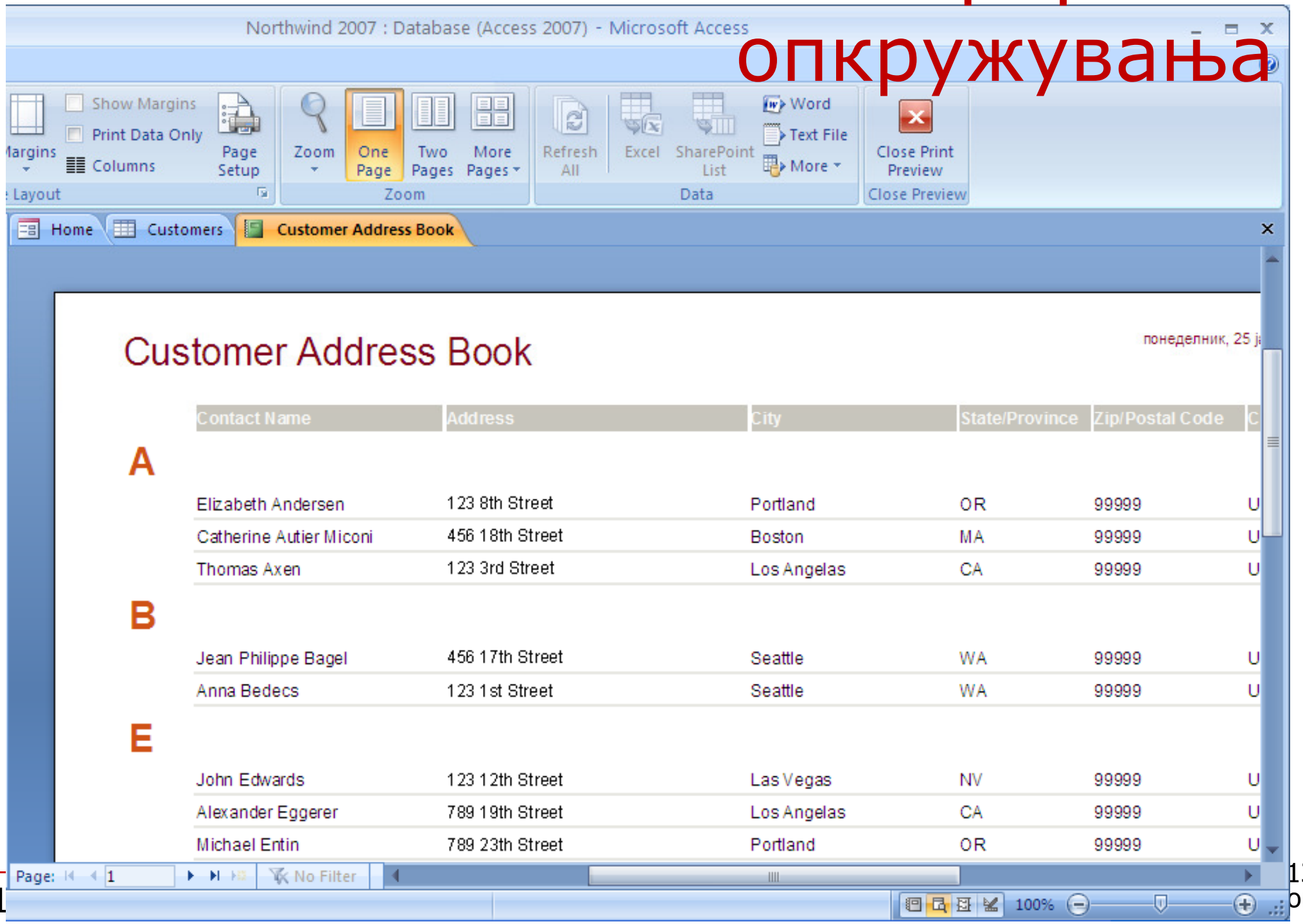
Интегрирани опкружувања



Интегрирани опкружувања

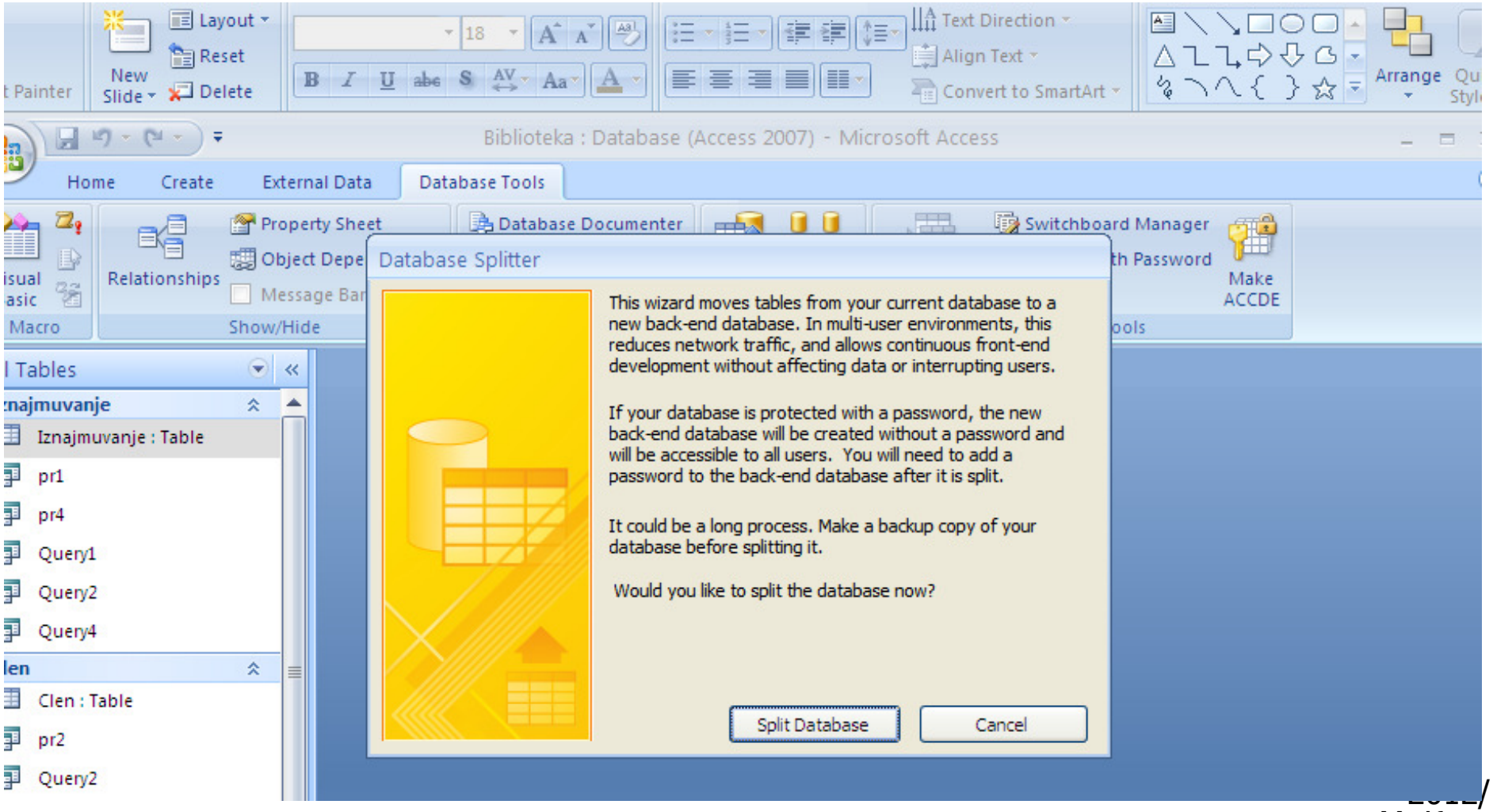


Интегрирани опкружувања



Интегрирани опкружувања

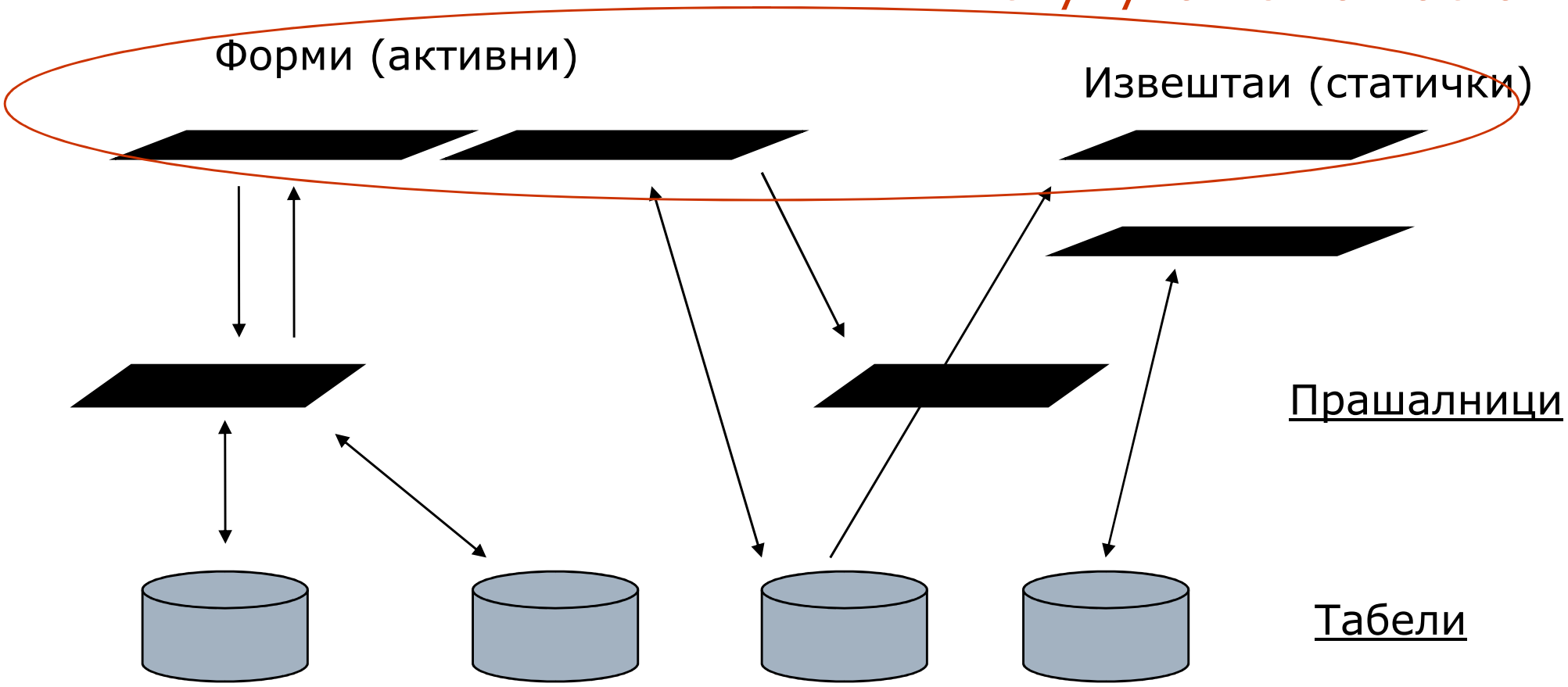
- Раслојување по потреба



Интегрирани опкружувања

Напредно – Делење

VB + Макроа – Автоматизација поврзана со случување на настан



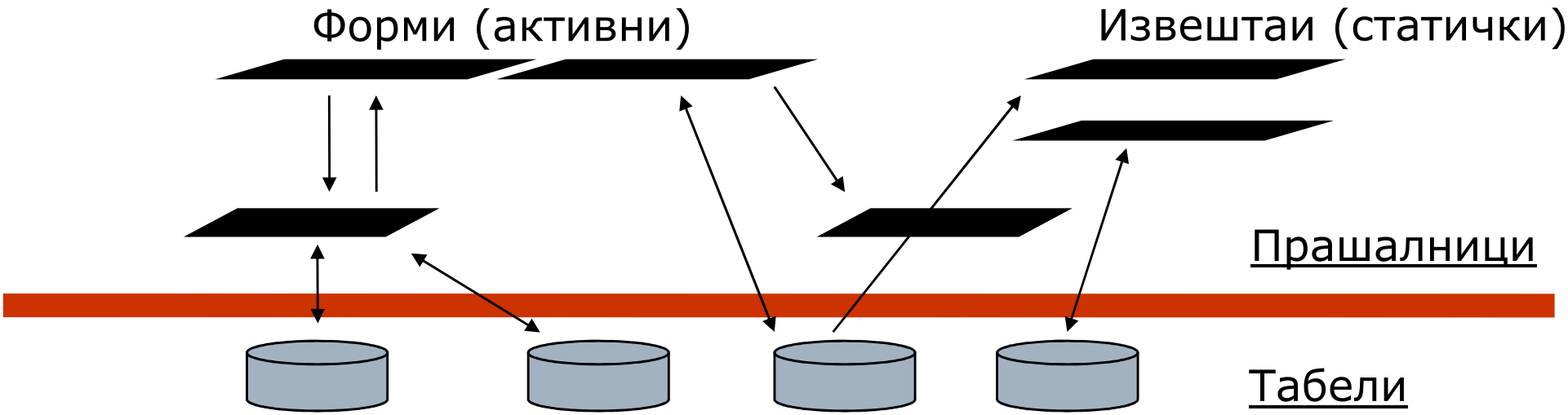
- Повеќе предности/backup/update/…

Интегрирани опкружувања

Напредно – Делење

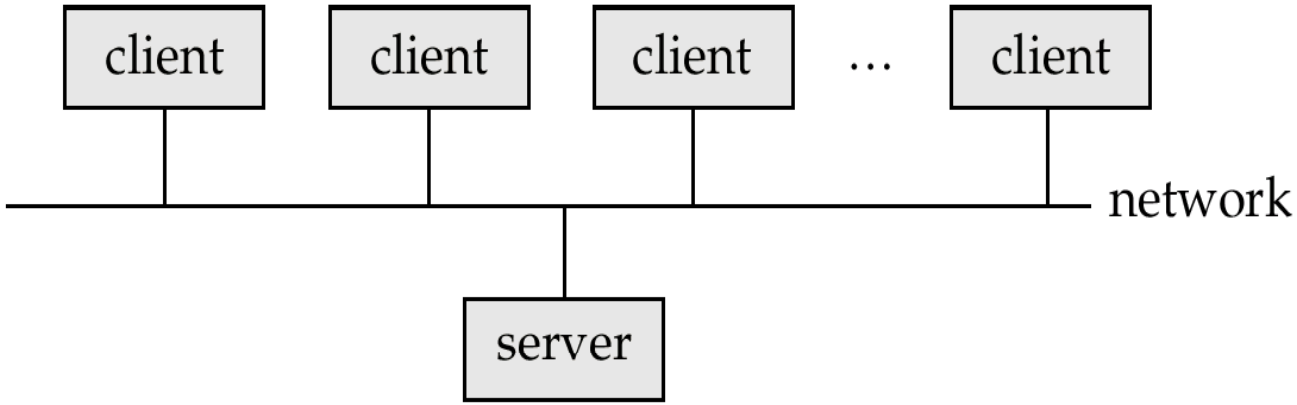
Датотека во преден план (Front-End) –
ги содржи сите објекти освен табелите од појдовната датотека (форми, прашалници, итн.) и линкови кон табелите во позадинската датотека (back-end).

VB + Macros – Event Driven Automation.



Позадинска датотека (Back-End) –
ги содржи сите табели од појдовната датотека

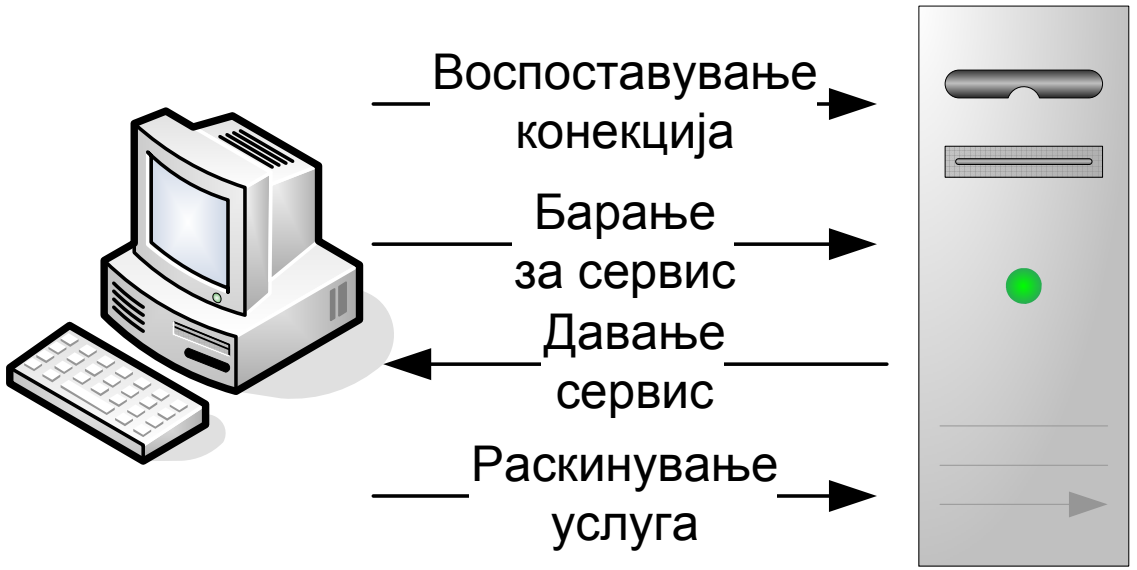
Класична клиент/сервер архитектура



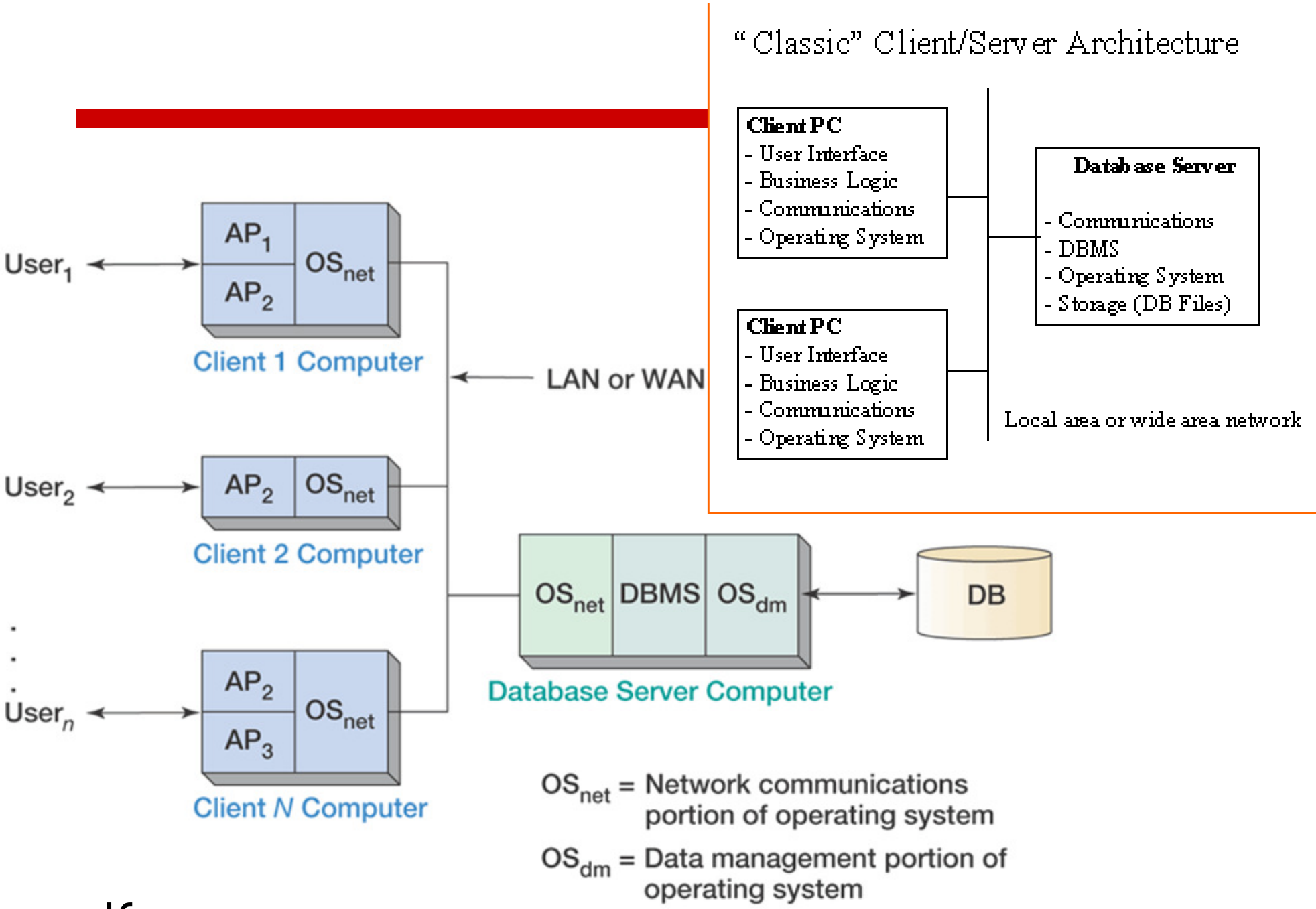
- Се состои од мрежа од сметачи поврзани во LAN
- Клиентите се PC-и (или работни станици) на кои се извршуваат апликации
- Серверите се PC-и или серверски машини, на кои се наоѓа СУБП и дел од ОС кој управува со податоците

Класична клиент/сервер архитектура

- Клиент-сервер архитектура
  - СУБП – серверска страна
  - Апликацијата која ја користи БП – клиентска страна



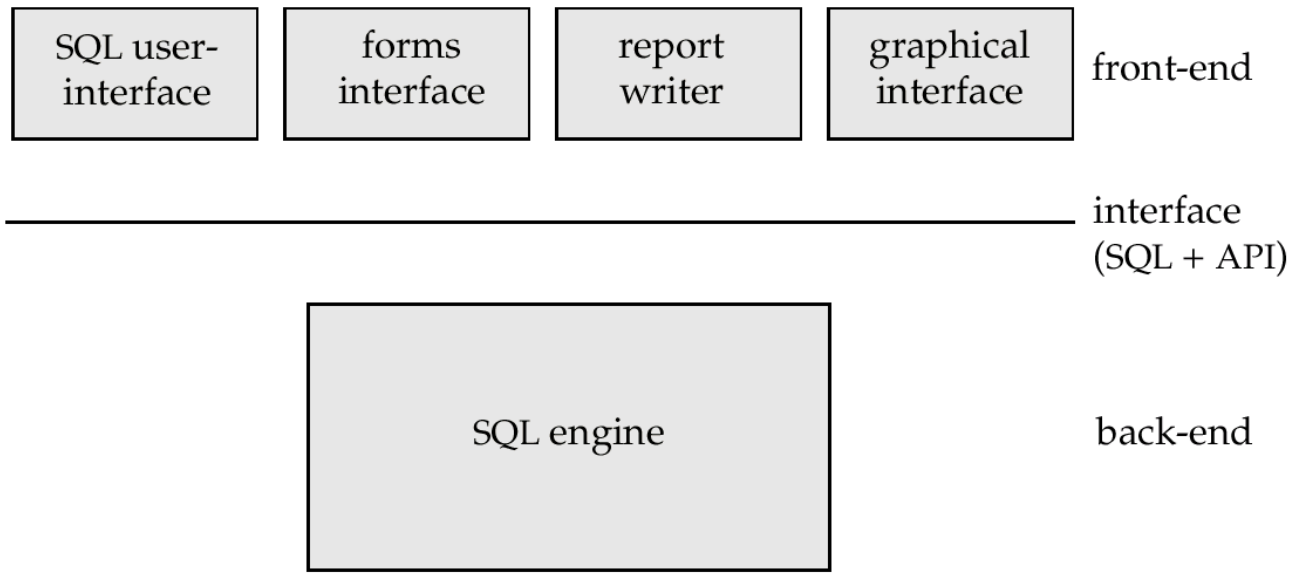
(други примери за клиент сервер – емаил сервер, веб сервер, DNS сервер, ...)



Клиент-сервер архитектура

Клиент/сервер архитектура

- Функционалноста на БП може да се подели на два дела:
  - **Во преден план (*front-end*)**
    - содржи алатки како што се форми, извештаи, графички кориснички интерфејси.
  - **Позадински дел (*back-end*)**
    - управува со пристапот, евалуација и оптимизација на прашалници, контрола на конкурентност и обновување (recovery).
  - Посредник помеѓу двете е SQL или преку апликациска програма



Клиент/сервер архитектура

Клиенти

- Работат под сопствен ОС.
- Извршуваат една или повеќе апликации кои ги користат меморијата и процесорот на клиентот.
- Апликациите со СУБП комуницираат преку драјвер за бази на податоци.
- Драјверот за бази на податоци (*middleware*) се поврзува со СУБП серверот преку мрежата.
- Примери на клиенти:
  - PC со MS Windows оперативен систем
  - Форми и извештаи креирани во MS Access, Borland Delphi, Oracle Developer/2000, MS Visual Basic, C, C++, итн.

Клиент/сервер архитектура

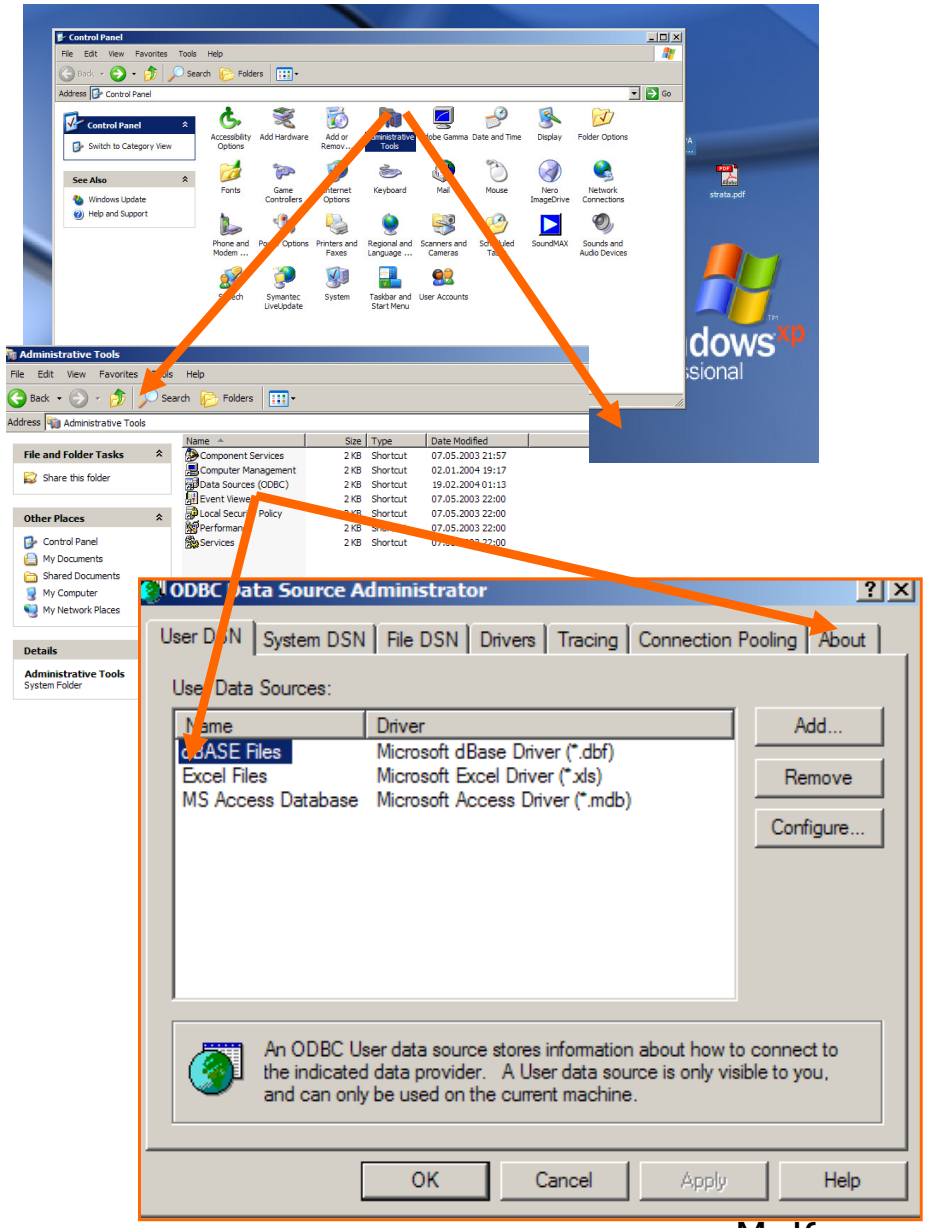
Сервери

- Работат под сопствен ОС.
- СУБП управува со базата на податоци.
- Обезбедува "демон" програма (*listening daemon*) што ги наслушнува и прифаќа поврзувањата од клиентите и ги пренесува до СУБП.
- Примери на сервери:
  - Sun Spark сервер под UNIX оперативен систем. RDBMS како Oracle Server, Sybase, Informix, DB2 итн. PC со Windows NT оперативен систем.

Клиент/сервер архитектура

Middleware

- Програм кој посредува помеѓу клиентот и серверот.
- Воспоставува конекција од клиентот до серверот и ги пренесува командите (како на пример SQL) помеѓу нив.
  - Пример ODBC



Клиент/сервер архитектура

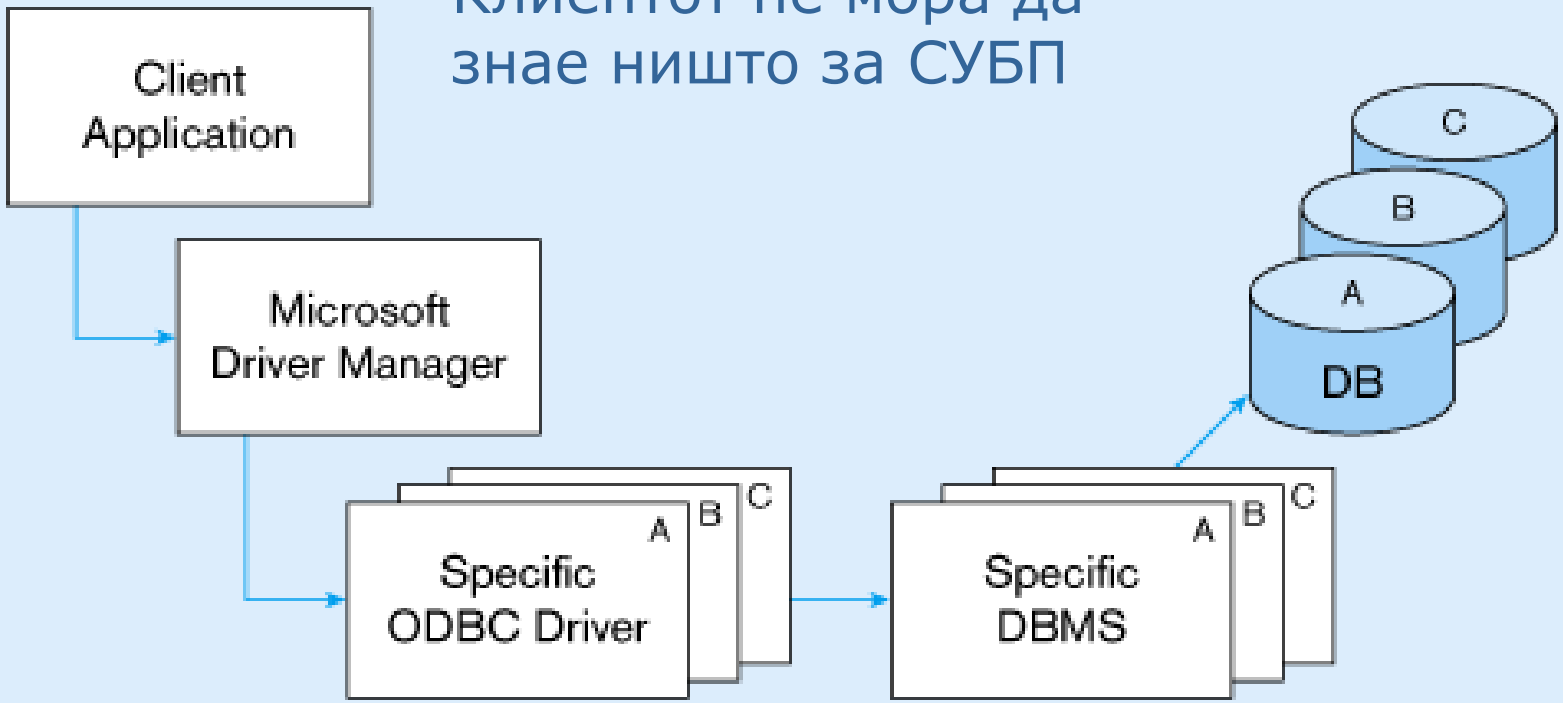
Middleware

- *ODBC* и *JDBC*
  - Се појавуваат стандарди како *ODBC* и *JDBC*, како интерфејс помеѓу клиентите и серверите.
  - Секој клиент кој користи ODBC или JDBC интерфејс може да се поврзе на било кој сервер со податоци кој го обезбедува тој интерфејс.
  - Целта е да се овозможи со креирање на една единствена апликација да може да се пристапи до БП-и поддржани од различни СУБП без потреба од менување или рекомпајлирање на апликацијата

Клиент/сервер архитектура

Middleware

Клиентот не мора да знае ништо за СУБП



Секој СУБП има свој сопствен ODBC драјвер

Клиент/сервер архитектура

Работни правила

- За дефинирање или ограничување на вредноста на податоците
  - Пр. на професорите им е дозволено да внесуваат и менуваат оценки на студенти по предмети кои ги предаваат, но не и на другите предмети
- Може да се изведуваат:
  - Со апликациите на клиентската страна **“Fat Clients”**.
  - Целосно на серверот за бази на податоци **“Thin Clients”**.
  - Комбинација на двете.

Клиент/сервер архитектура

- **Предности** од замената на mainframe систем со клиент-сервер систем (мрежа од работни станици или PC-и поврзани на back-end серверски машини):
  - Подобра функционалност
    - Процесирањето на податоците се извршува и на клиентите и на серверот.
    - СУБП може да постигне добри перформанси затоа што се грижи само за процесите на трансакциите (не и за извршување на апликациите).
  - Флексибилност во лоцирањето на ресурси
  - Подобри кориснички интерфејси (GUI)
  - Полесно одржување

Клиент/сервер архитектура

- **Недостатоци** на клиент-сервер архит.
  - Имплементацијата е покомплексна
    - потребно е да се земат предвид посредничката (*middleware*) програма и мрежата.
  - Можно е мрежата да не е погодна за клиент/сервер комуникации и може да стане преоптоварена.
  - Дополнителна тешкотија за СУБП серверот е контрола на конкурентноста и сл.
  - Колку повеќе логички правила се вградуваат во работните правила кај клиентот, тие стануваат се понеспретни.
    - Во овој случај помагаат тригерите и вградените процедури.

Апликациска логика во клиент-сервер архитектури

- Презентациска логика
  - Влез - тастатура/глувче
  - Излез – монитор/принтер
- Процесирачка логика
  - I/O процесирање
  - Бизнис правила
  - Управување со податоци
- Логика за складирање податоци
  - Складирање/пребарување податоци

GUI Interface

Procedures, functions, programs

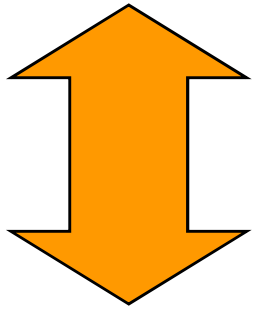
DBMS activities

Во зависност каде ќе бидат сместени овие логики, клиент-сервер архитектурата се дели на еднослојна, двослојна, трослојна, n-слојна

Споредба на клиент-сервер архитектури

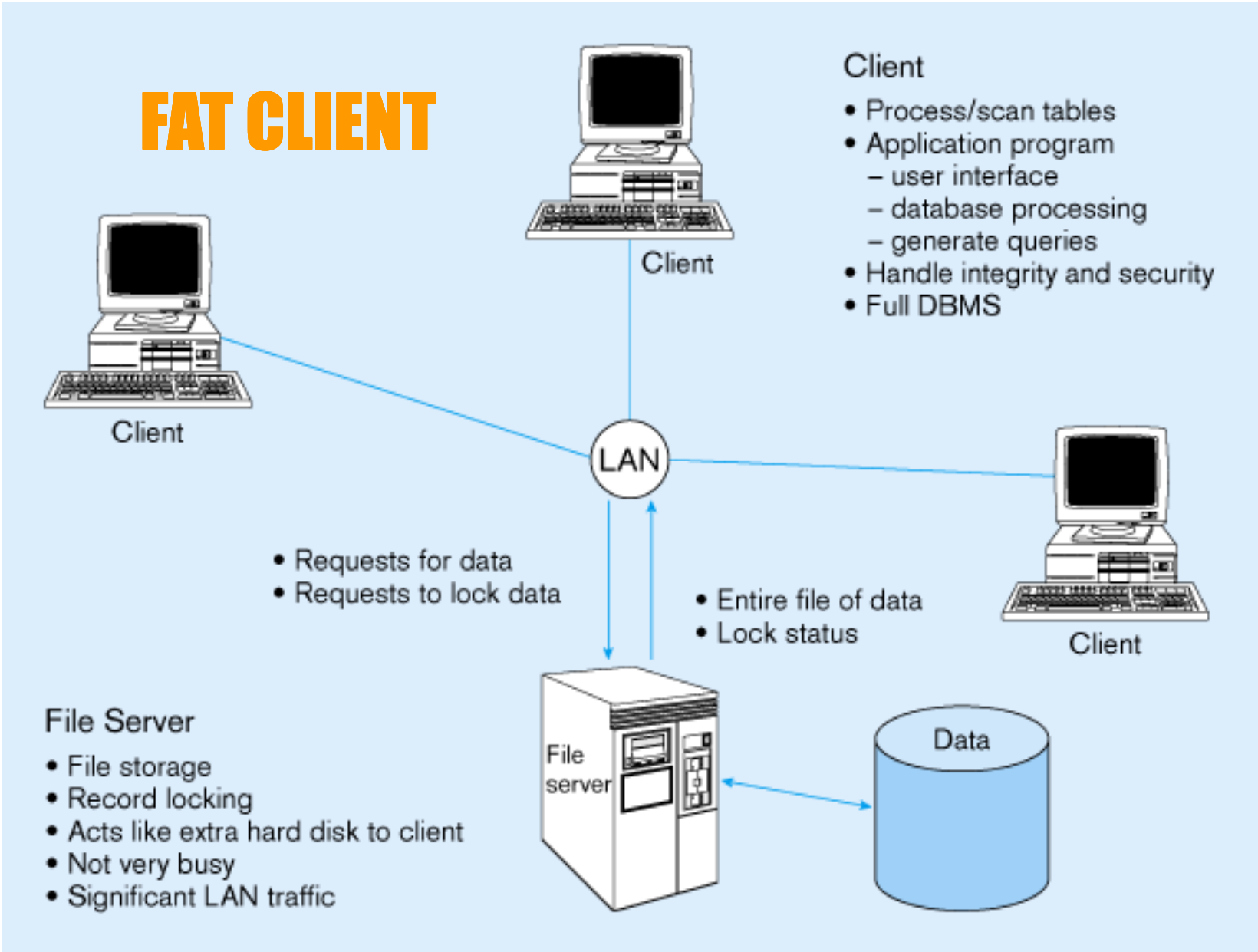
- Податочен сервер (File server)
- Двослојна (Two-Tier) архитектура
- Трислојна (Three-Tier) архитектура
- n – слојна (n-Tier) архитектура

Клиентите извршуваат многу процесирање



Клиентите извршуваат малку процесирање

File server architecture



Архитектура
Податочен сервер

- Единствен, датотечен (file) сервер се користи за сместување на единствена копија од БП.
- Податочниот сервер се однесува како дополнителен диск за секој од клиентите
- Секое PC претставува FAT CLIENT
- Целото процесирање се врши на PC-то кое ги бара податоците
  - клиентот се справува со презентациската логика, процесирачката логика и во голема мера со логиката за складирање податоци
- Се пренесуваат цели датотеки за да се обработат од клиентот.
  - Пр. Делење на MS Access датотека поставена на податочен сервер.

Архитектура
Податочен сервер

Предности:

1. Можност да се делат податоците меѓу неколку корисници
2. Цената на складирање на податоци е поделена помеѓу повеќе корисници.

Проблеми:

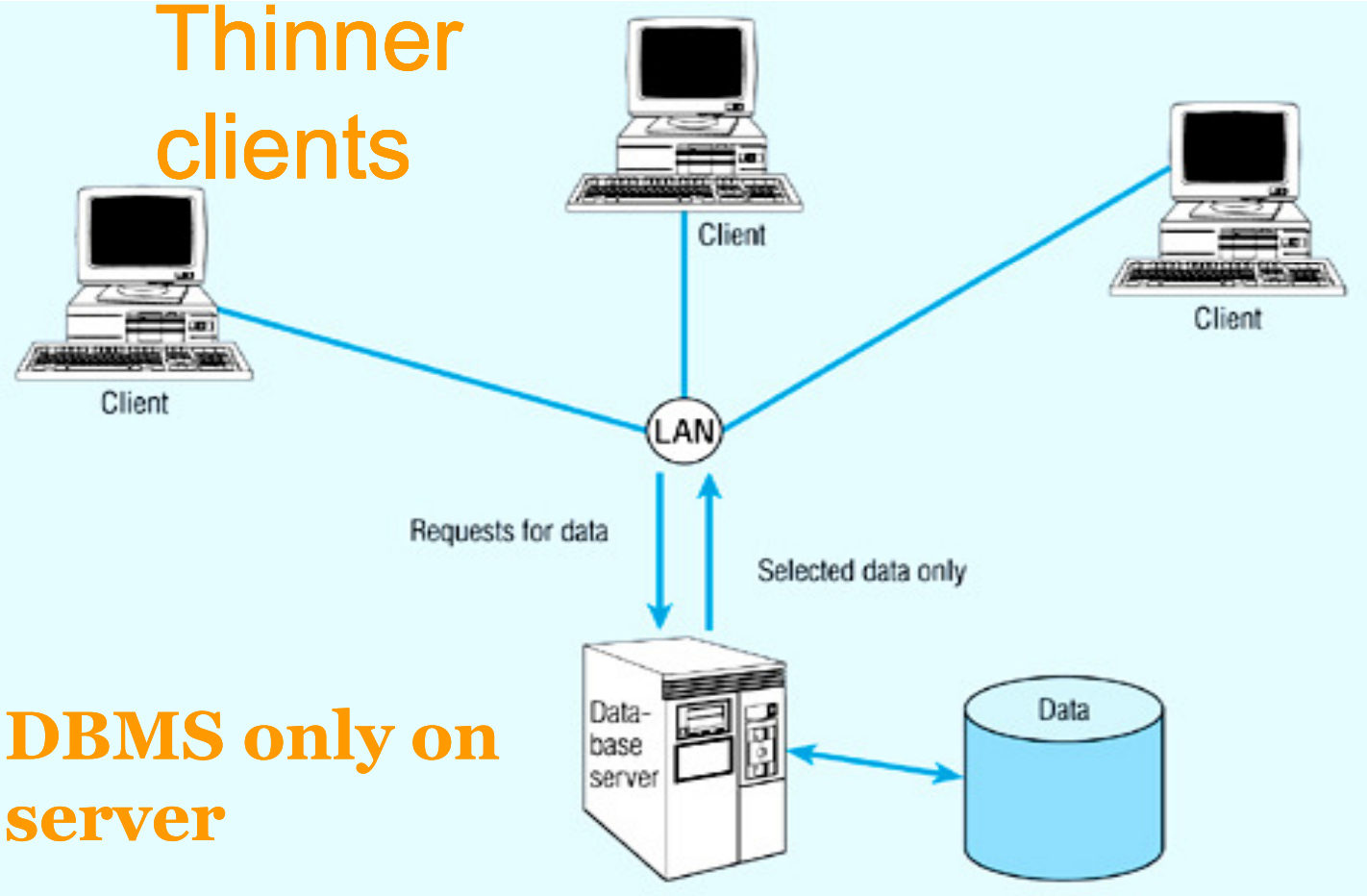
1. Голем сообраќај преку мрежата
2. Секој клиент е авторизиран да го користи СУБП
3. Делен интегритет на базата
   - Заклучувања, проверка на интегритет

Слоевита структура

- Со појавата на ООП е овозможено:
  - Раздвојување на податоците од логиката на нивната обработка
  - Раздвојување на податоците од интерфејсот кој го користат корисниците.
- Апликациите се градат од објекти
- Примери:
  - Група објекти од кои се градат кориснички интерфејси
  - Група објекти кои остваруваат конекција со БП, извршуваат прашалници и прифаќаат резултати од прашалници

Слоевита структура

- **Двослоен модел**



Слоевита структура

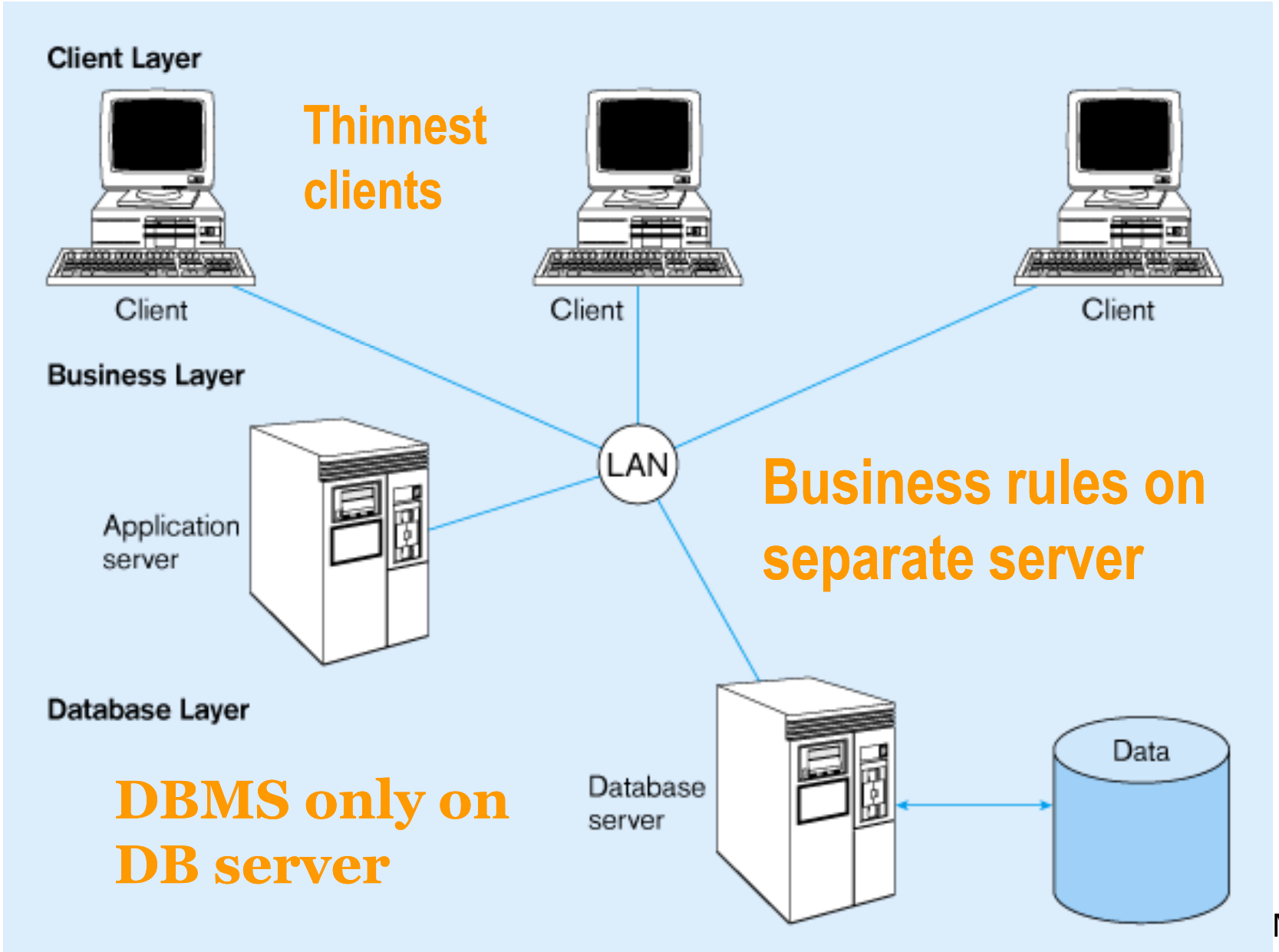
- **Двослоен модел**
  - Корисничкиот интерфејс обично се извршува на клиентот (десктоп машината)
    - Клиентот е одговорен за И/О логиката, дел од правилата на бизнис логиката
  - Апликациските сервиси се обезбедени од сервер (пример за податочна база)
    - Серверот ги извршува сите процесирања за складирање и пристапи до податоците
    - СУБП се наоѓа на сервротот
    - Вгнездени процедури (stored procedures / објаснети понатаму)

Слоевита структура

- **Двослоен модел**
  - **Предност**
    - Клиентите може да бидат и послаби машини
    - Намален сообраќајот низ мрежата
    - Подобрен интегритет на податоци
    - Вгнездени процедури (stored procedures)
  - **Недостаток**
    - Има **намалување на перформансите** заради голем број на корисници
    - **Серверот губи многу време за управување со конекциите** и нема доволно процесорски циклуси за завршување на задачите во одреден временски рок

Слоевита структура

- **Трослоен модел**



Слоевита структура

- **Трослоен модел**

Client	**GUI interface (I/O processing)**	***Browser***
Application server	**Business rules**	***Web Server***
Database server	**Data storage**	***DBMS***

Thin Client

- PC само како кориснички интерфејс и малку процесирање
- Ограничено или нема сместување податоци

Слоевита структура

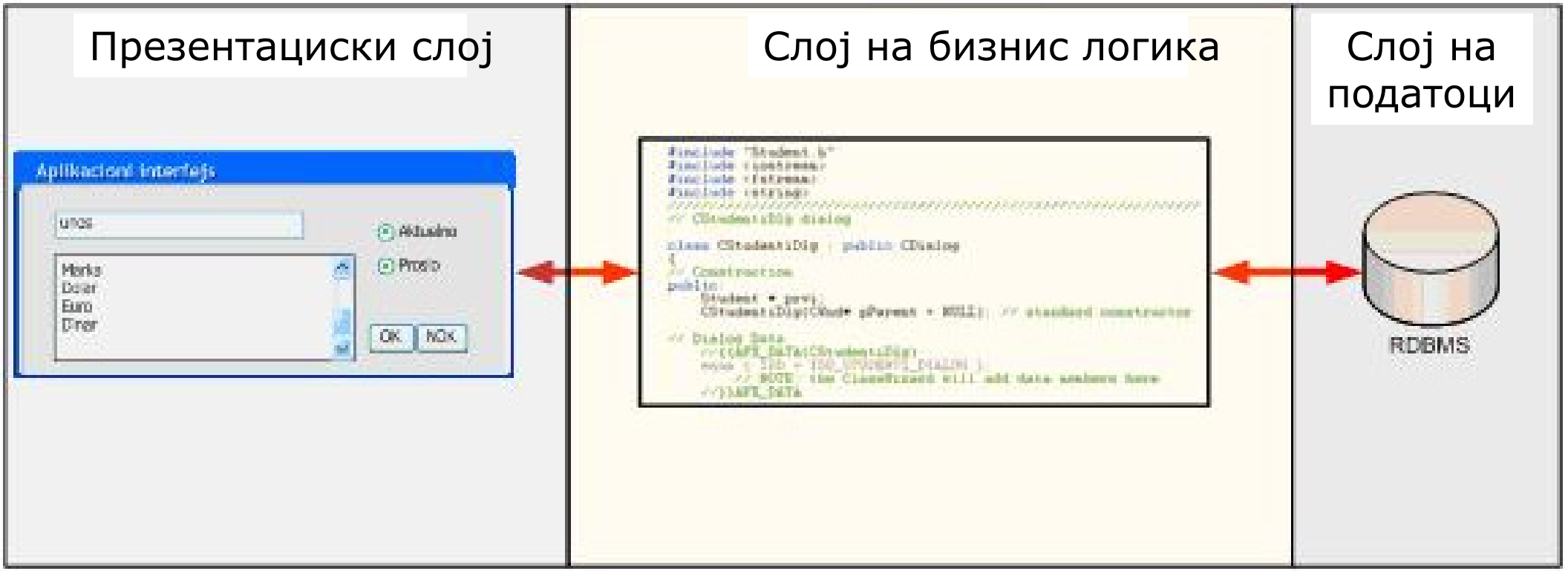
- **Трослоен модел**
  - **Предности**
    - Скалабилност
      - преку средното ниво, ефикасно работи и при зголемување на бројот на корисници или ресурси
    - Технолошка флексибилност
      - различни платформи
    - Намалена цена при долгорочно планирање
    - Предност при конкурентен пристап
  - **Предизвици / Недостатоци**
    - Високи почетни вложувања
    - Знаење / искуство
    - Некомпатибилни стандарди и алатки на крајни корисници

Слоевита структура

- **Трослоен модел**
  - Основниот апликациски модел е трослојниот модел
    - Презентациски слој (*presentation layer*)
      - Објекти на GUI (форми за преглед, внесување, измена, бришење податоци)
    - Слој на бизнис логика (*business logic layer*)
      - Обработка на податоци и објекти за синхронизација на процесите на презентацискиот и слојот на податоци
    - Слој на податоци (*data layer*)
      - Објекти за комуникација со БП (СУБП)

Слоевита структура

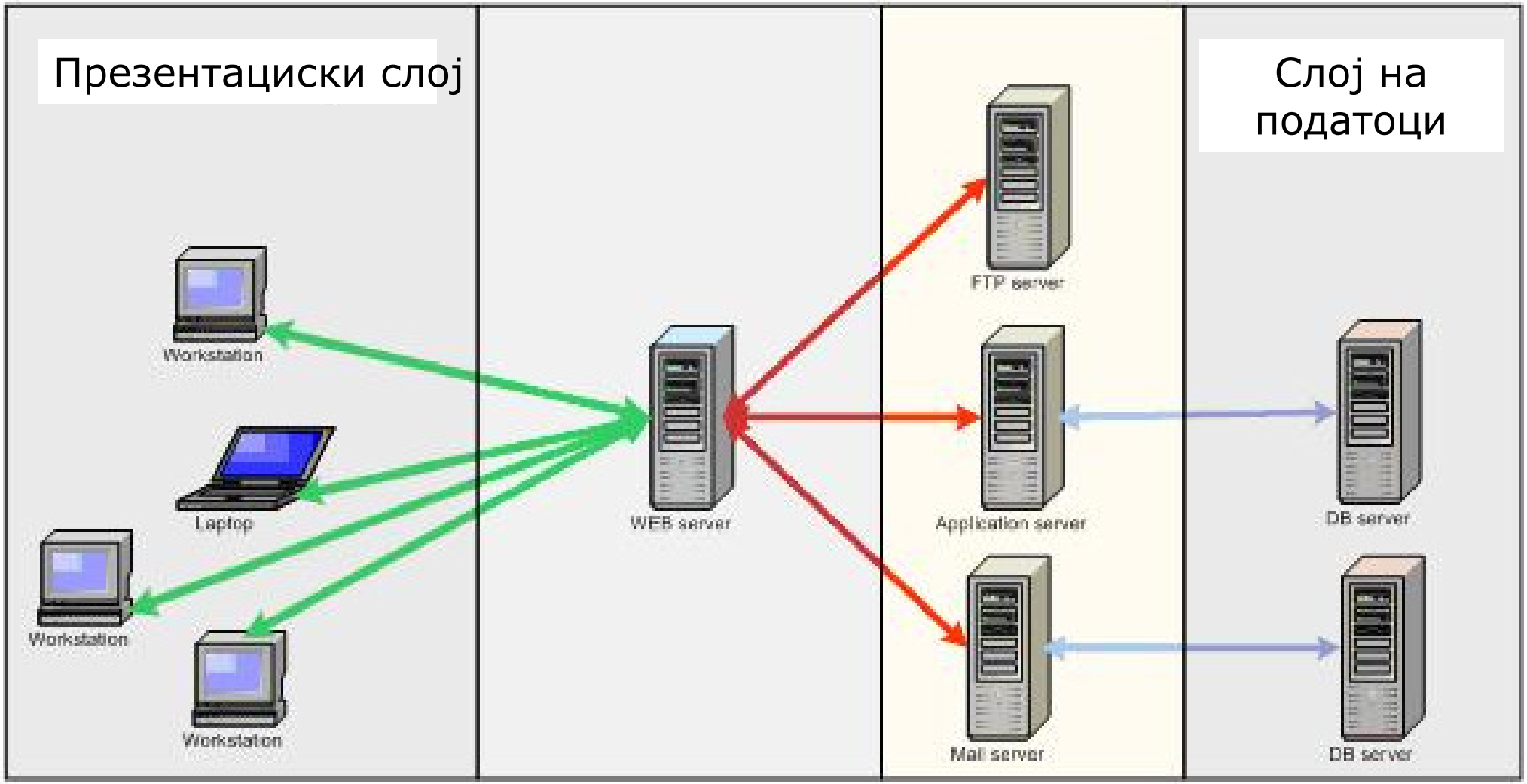
- **Трослоен модел**



Слоевита структура

- Апликациите може да имаат повеќе од три слоја
- Податоците можат да бидат раздвоени на повеќе различни места
  - Растоварување на хардверските (серверските) платформи и врските (линковите) помеѓу корисниците и апликациите
- Повеќе нивоа на обработка
  - Пр. Web апликации

Слоевита структура



Процесирање на различни слоеви

- Секој слој има различна функција и може да има различен оперативен систем (Macintosh, Windows, UNIX)
- Различен апликациски софтвер (Microsoft IIS, Apache)
- Различни СУБП (Oracle, SQL Server, Access)

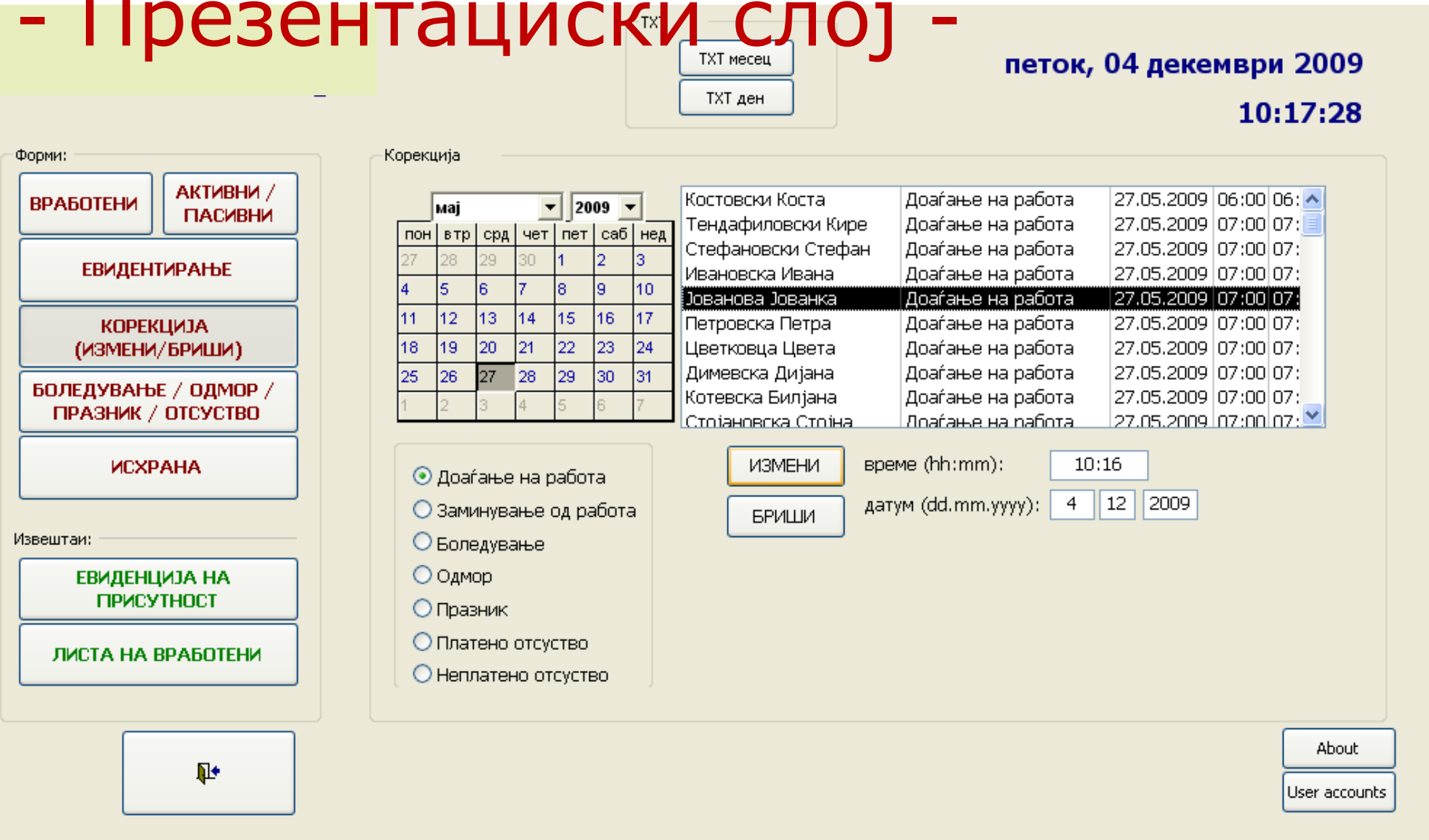
Пристап до БП

- До БП-и може да се пристапи на три начини:
  - Пристап до податоците од презентацискиот слој
  - Пристап до податоците од слојот на бизнис логика
  - Пристап од слојот на податоци

Пристап до БП
- Презентациски слој -

- Презентацискиот слој содржи објекти на корисничкиот интерфејс
- Прозорци кои содржат контроли за интеракција со корисникот

Пристап до БП
- Презентациски слој -



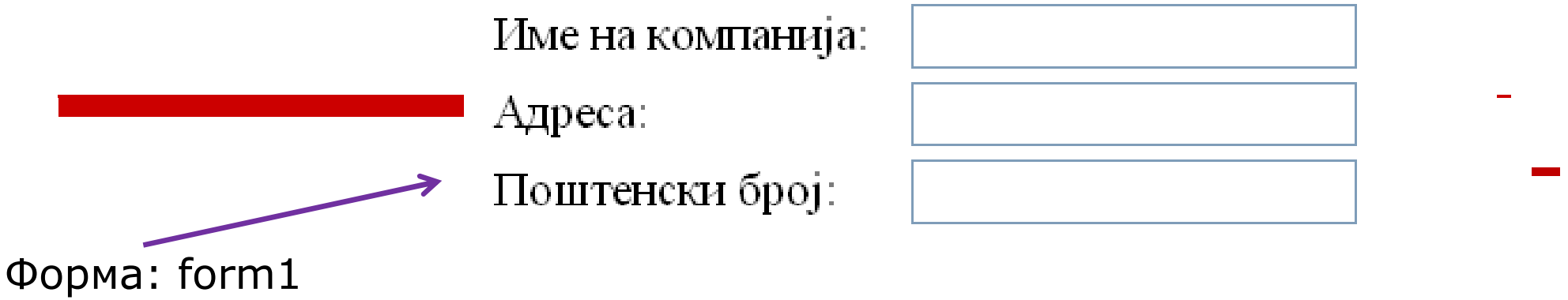
Пристап до БП
- Презентациски слој -

```
1:Private Sub Zapisi_Click()
2:DoCmd.RunSQL "UPDATE tbl_evid SET [vreme] = " &
3:Forms![frm_korek]![Text27].Value
4:st = Me.Text33.Value & "/" & Me.Text35.Value & "/" &
5:Me.Text37.Value
6:DoCmd.RunSQL "UPDATE tbl_evid SET [datum] = " &
7:DateValue(st)
8:End Sub
```

VBA скрипта која содржи SQL наредба

Пристап до БП
- Презентациски слој – Пр. web апликација

- Форми со контроли за интеракција со корисниците кодирани во HTML јазикот.
  - После внесувањето податоци, корисникот со притиснување на копче започнува соодветна акција.
- *Web browser-от* на клиентската машина го интерпретира HTML кодот и прикажува соодветна web страница во својот прозорец.
- Внесените податоци во вид на HTTP барање се пренесуваат до Web сервер, каде се извршува соодветниот код (пр. ASP, PHP).
  - До колку акцијата се однесува на пристап до БП, се воспоставува конекција со БП, се составува SQL наредба и се извршува.




```
<body>
<form name="form1" method="post" action=“add_cust.asp">
 <table width="300" border="0">
  <tr>
   <td>Име на компанија:</td>
   <td><input type="text" name=“firma”></td>
  </tr>   <tr>
   <td>Адреса:</td>
   <td><input type="text" name=“adresa"></td>
  </tr>   <tr>
   <td>Поштенски број:</td>
   <td><input type="text" name=“postkod"></td>
  </tr>   <tr>
   <td><input type="submit" name="Submit“></td>
   <td><input type="reset" name="Submit2” ></td>
  </tr>
 </table></form></body></html>
```

Пристап до БП - Презентациски слој -

```
1:  <html>
2:  <body>
3:  <%
4:  set conn=Server.CreateObject("ADODB.Connection")
5:  conn.Provider="Microsoft.Jet.OLEDB.4.0"6:  conn.Open
"d:/webdata/partneri.mdb“
7:  sql="INSERT INTO kupuvaci (ime_firma, adresa, postbroj)“
8:  sql=sql & " VALUES “
9:  sql=sql & "('" & Request.Form("firma") & "',“
10: sql=sql & "'" & Request.Form("adresa") & "',“
11: sql=sql & "'" & Request.Form("postkod") & "')“
12: on error resume next
13: conn.Execute sql,recaffected
14: if err<>0 then
15:   Response.Write("Nemate pravo na dodavanje podatoci!")
16: else
17:   Response.Write("<h3>Klientot " & Request.Form("firma")
18:  & “ e dodaden</h3>")
19: end if
20: conn.close
21: %>
22: </body>
23: </html>
```

Пристап до БП преку ASP (*Active Server Pages*) страница

Пристап до БП
- Презентациски слој -

Предности на пристапот од презентацискиот слој:

- **едноставност и брзина** на имплементација
- погоден за едноставни апликации (тестирање) бидејќи **сè се наоѓа на едно место**
- погоден кога се користат едноставни SQL наредби и кога целниот СУБП е **однапред познат и непроменлив**

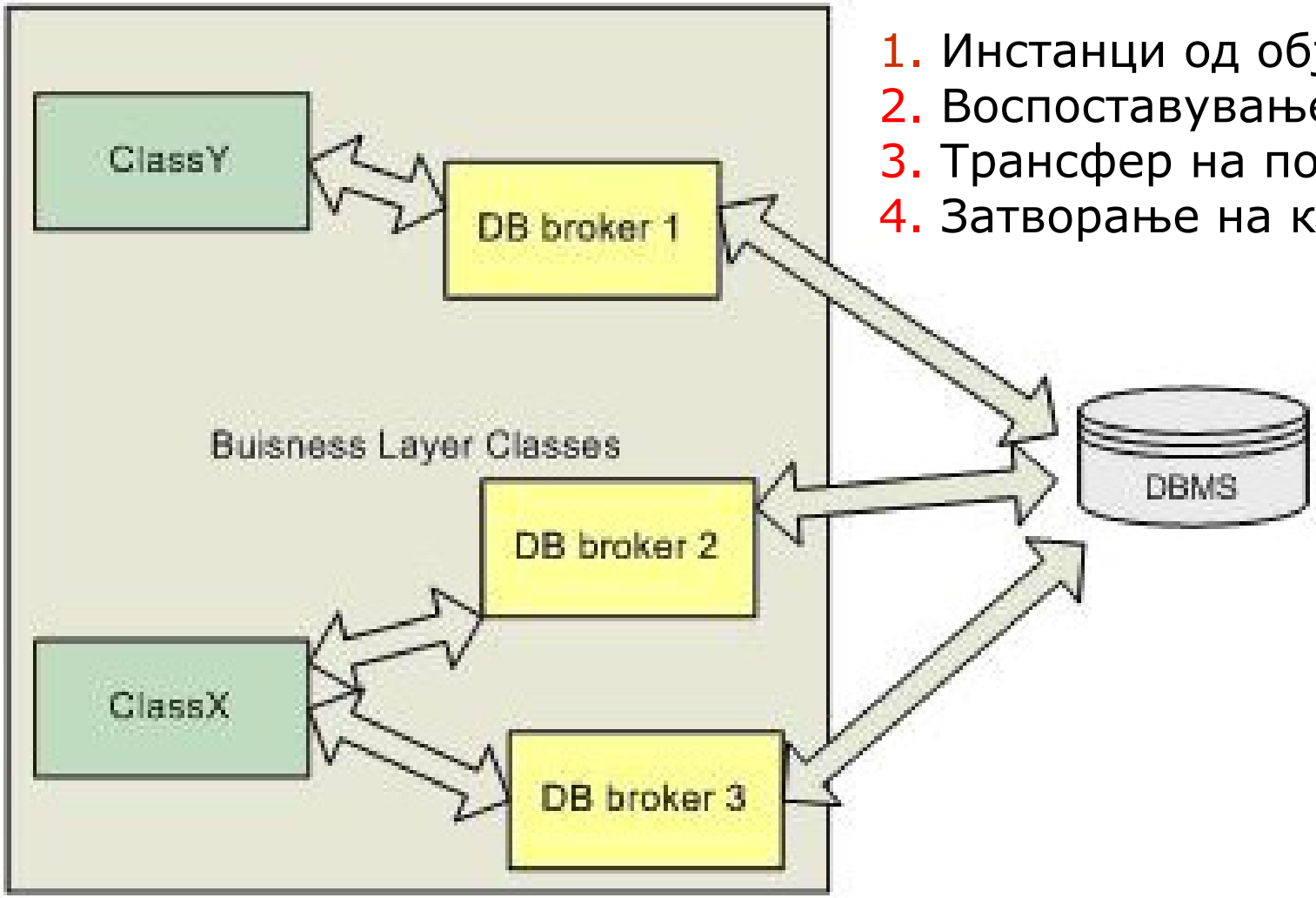
Недостатоци на пристапот од презентацискиот слој:

- обично е потребно **менување на SQL кодот** кој се наоѓа во објектите на корисничкиот интерфејс во случај на промена или инсталација на нова верзија на СУБП
- **преклопување на работите** на дизајнерите и програмерите што создава конфузија со ваквиот пристап кај посложените апликации
- **макотрпно одржување и управување** со нераслоениот софтвер што често дава лоши резултати

Пристап до БП - Слој на бизнис логика

- Пр. SQL кодот директно во кодот на самата апликација
- Најчесто користен пристап кај повеќеслојните апликации
- Овој слој содржи ентитети (класи или модули) се задолжени за комуникација со БП
- Услужни класи кои овозможуваат интеракција со БП
  - *CDatabase, CRecordset* класи од Microsoft (MFC)
  - *ResultSet, Connection* класи во Java пакетот *java.sql.**

Пристап до БП
- Слој на бизнис логика



1. Инстанци од објектите.
2. Воспоставување конекција со БП.
3. Трансфер на податоци.
4. Затворање на конекцијата со БП.

Пристап до БП - Слој на бизнис логика

```
try{
   java.sql.Connection conn;
   java.sql.Statement stmt;
   java.sql.ResultSet rset;
   java.sql.ResultSetMetaData rsm;
   conn=this.cpool.checkOut();
   stmt=conn.createStatement();
   stmt.executeUpdate("INSERT INTO " +
            "t_mtutor_groups" +
            " VALUES ( 0, '"+group_name+"')");
   stmt.close();
   this.cpool.checkIn(conn);
}catch(Exception ex){
   ex.printStackTrace();
}
```

Java код кој додава нов запис во табелата *t_mtutor_groups* во БП

Пристап до БП - Слој на бизнис логика

Предности на пристапот од слојот на бизнис логика:

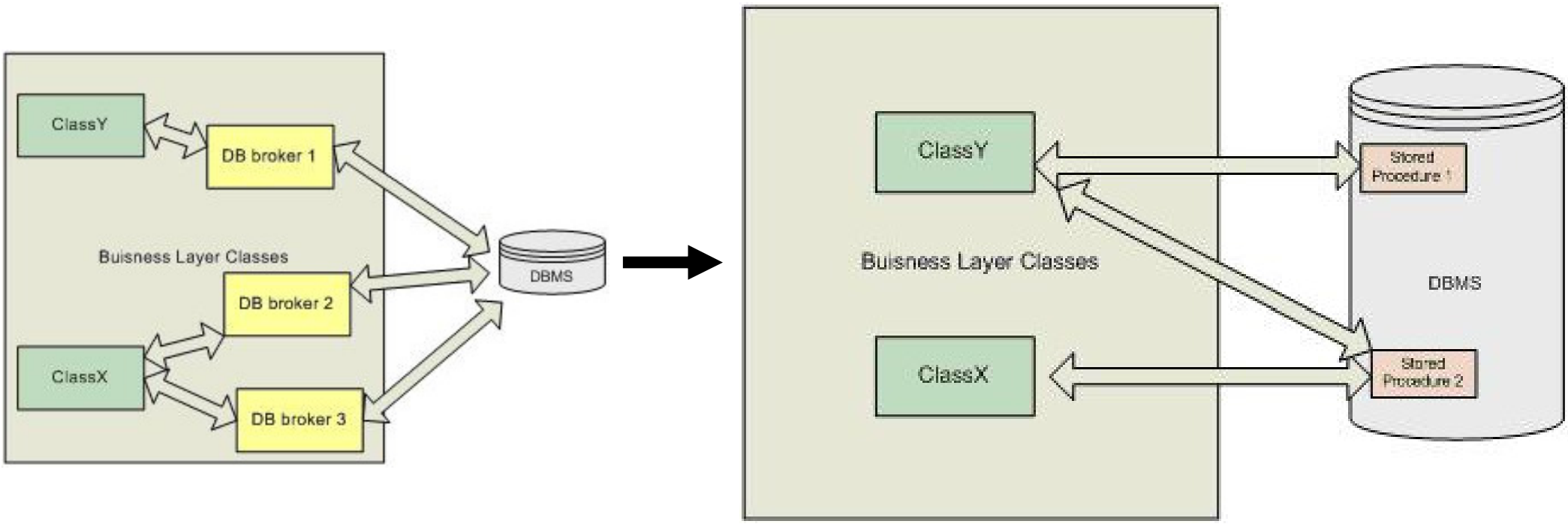
- Објектите за размена на податоци со БП се дизајнираат потполно **независно** од презентацискиот слој
- Објектите се **посредници** меѓу БП и остатокот од апликацијата

Недостатоци на пристапот од слојот на бизнис логика:

- SQL наредбите **се директно внесени во изворниот код** на апликацијата.
- **Нарушена оптимизираност** на код и на цела апликација.
- **Зголемен обем на код → отежнато одржување**.
  - На пр. ако е потребно да се промени името и структурата на некоја табела во базата, соодветните измени мора да се направат над сите SQL наредби кои референцираат на таа табела.
  - Исто така, апликацијата би морала повторно да се генерира, инсталира и подесува.

Пристап до БП - Слој на податоци

- Преместување на SQL наредбите од изворниот код на апликацијата во СУБП



Пристап до БП - Слој на податоци

- Вгнездени процедури (*stored procedures*)
  - Множества инструкции кои често се користат
  - Програмерите се ослободуваат од **повеќекратно повикување** на исти команди
  - **Најбрзо се извршуваат** бидејќи процесот на преведување инструкции се врши на SQL серверот
  - Процедурите се наоѓаат **на едно место**, а не на повеќе места во апликациите (пр. на *front-end* крајот – презентациски слој), па нивната измена и ажурирање е многу полесно

Пристап до БП - Слој на податоци

- Вгнездени процедури (*stored procedures*)
  - **Зголемена контрола на безбедност**
  - При извршување, само резултатот од обработката се испраќа на следниот слој
    - Корисни се кога голем број на пристапи до БП треба да се извршат, а мал резултат да се врати до клиентот
    - **Најмало оптоварување на мрежата**, подобрени перформанси

Пристап до БП - Слој на податоци

- Вгнездени процедури (*stored procedures*)
  - Може да се проследуваат параметри и променливи
  - Може да се повикаат и од други процедури
  - Најчесто се пишуваат во некој од проширените SQL јазици кои ги дефинира производителот на конкретниот СУБП
    - Oracle – PL/SQL
    - Microsoft SQL Server – Tansact-SQL

Пристап до БП - Слој на податоци

- Предуслов – СУБП мора да поседува можност за креирање процедури
- SQL наредбите се *вгнездуваат* како процедури (*stored procedure*) во целната БП

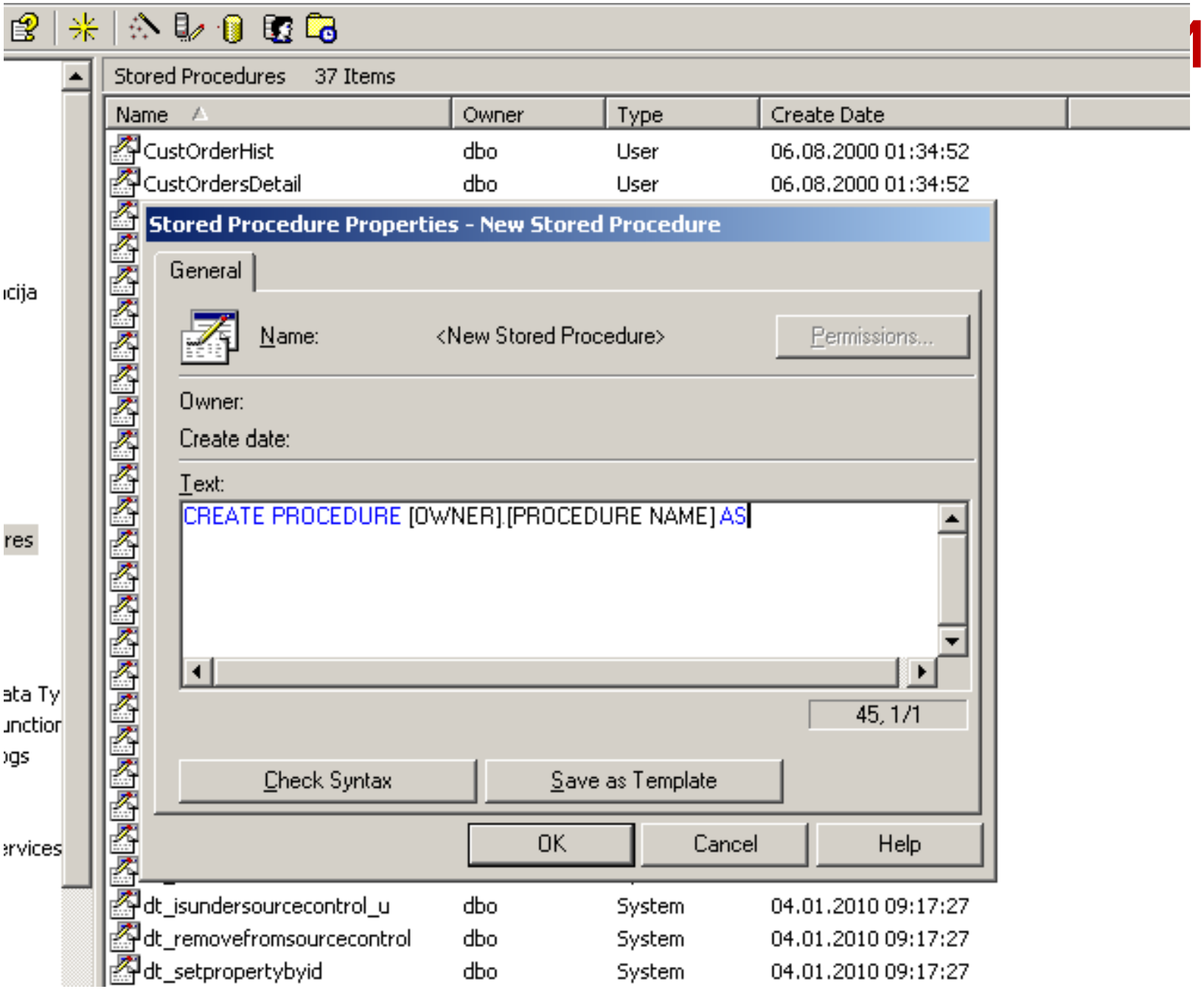
```
1: CREATE PROCEDURE `spUsedTestSets`(IN u_id INTEGER(11))
2: BEGIN
3: SELECT * FROM `t_mtutor_used_test_sets` WHERE (user_id=u_id);
4: END;
```

Пристап до БП - Слој на податоци

- Повикување на вгнездените процедури

```
1: cs = conn.prepareCall("{call spUsedTestSets(?)}");
2: cs.setInt("user_id", u_id);
3: rs = cs.executeQuery();
4: while( rs.next() ){
5:     int test_id = rs.getInt("test_set_id");
6:     Date test_dat = rs.getDate("date");
7: }
```

Пристап до БП



Пристап до БП - Слој на податоци

- Тригери, окидачи (*triggers*)
  - Специјален вид на програмска (вгнездена) процедура во рамките на СУБП, која се активира со случување на одреден настан во СУБП:
    - Внесување на нов запис во табела
    - Бришење на одреден запис
    - Модификација на постоечки запис
- Тригерот претставува еден од механизмите за проверка на услов на интегритет на БП

Пристап до БП - Слој на податоци

- Тригери, окидачи (*triggers*)
  - **DML тригери** се извршуваат кога еден корисник се обидува да промени податоци
    - DML настани се INSERT, UPDATE или DELETE наредби врз табела или поглед
  - **DDL тригери** се извршуваат како одговор на DDL настани
    - CREATE, ALTER или DROP наредби, или системски вгнездени процедури кои вршат DDL операции.
  - **Logon triggers** се извршуваат при LOGON настан при воспоставување на корисничка сесија.

Пристап до БП - Слој на податоци

- Тригери, окидачи (*triggers*)
  - Тригерот започнува одредена активност над базата, секогаш кога ќе се случи еден од наведените настани
    - Поточно, СУБП иницира извршување на тригери, веднаш по случувањето на настаните
  - Можност за повеќе тригери за еден конкретен настан
  - Работата со тригери е надвор од контролата на апликацијата која е врзана на СУБП, а задолжително се извршува

Пристап до БП - Слој на податоци

- Тригери, окидачи (*triggers*)
  - Синтаксата на тригерот е следна:
  - CREATE [OR REPLACE] TRIGGER /ime_na_triger/ /tip triger/ ON /ime tabela/
    BEGIN
    /izvrsni_instrukcii_na_trigerot/
    END
  - Типот може да биде: BEFORE, AFTER и INSTEAD OF

Пристап до БП - Слој на податоци

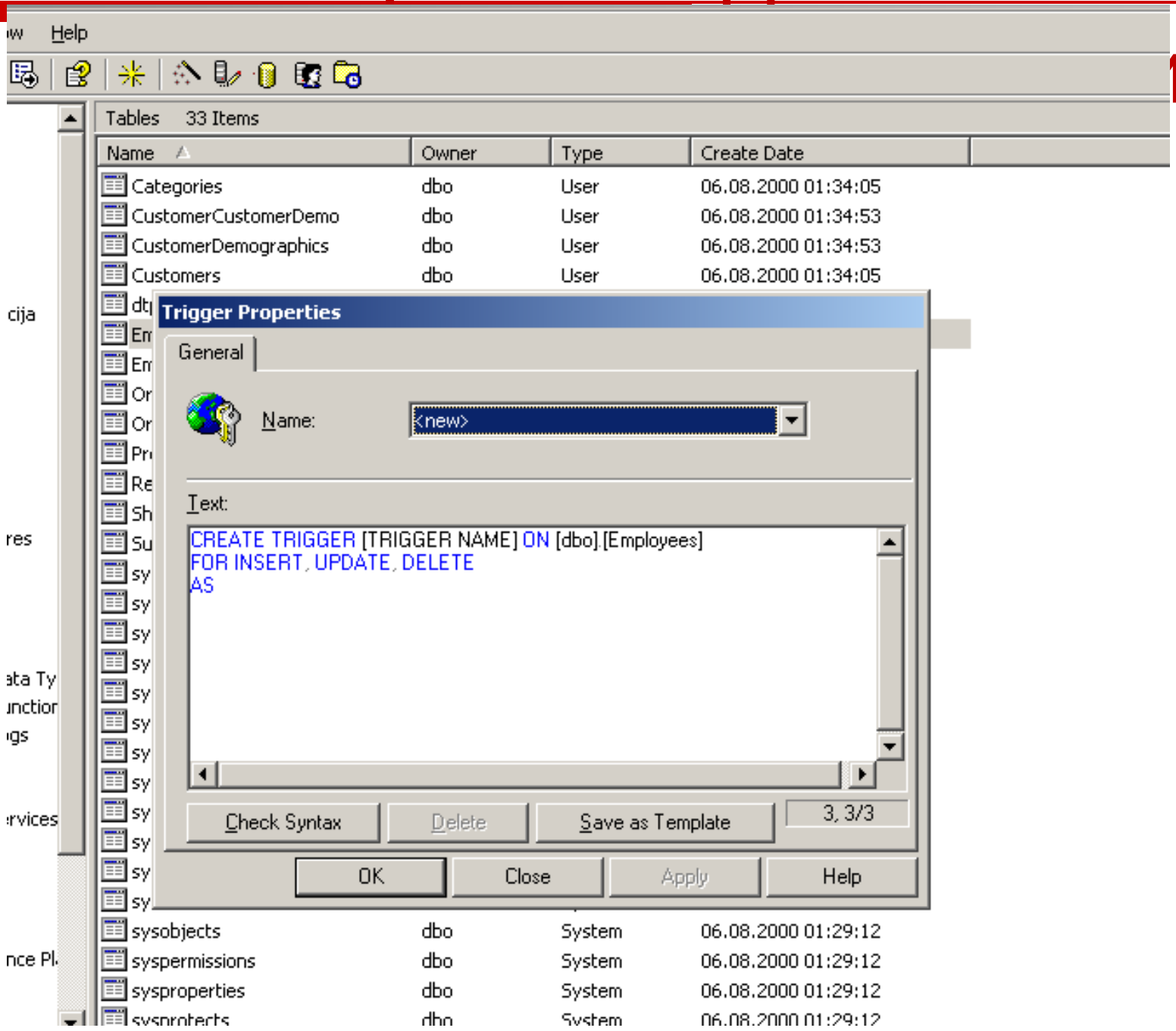
- Тригери, окидачи (*triggers*)
  - Пример за тригер кој прикажува порака пред внесувањето запис во табелата emp.

```
CREATE OR REPLACE TRIGGER emp_alert_trig
        BEFORE INSERT ON emp
BEGIN
        DBMS_OUTPUT.PUT_LINE('New employees are
                                about to be added');
END;
```

Пристап до БП - Слој на податоци

- Тригери, окидачи (*triggers*)
  - Oracle има тригери кои се активираат и кога се менува шемата на РБП
  - Тригери на ниво на шема (schema-level)
    - After Creation
    - Before Alter
    - After Alter
    - Before Drop
    - After Drop
    - Before Logoff
    - After Logon

Пристап до БП



Пристап до БП - Слој на податоци

Предности на пристапот од слојот на податоци:

- Со користењето на вгнездените процедури **се намалува комплексноста** на слојот на бизнис логика.
- Вгнездените процедури се прават за целниот СУБП, така да **се тестираат независно** од апликацијата (базата не мора да биде поврзана со апликацијата).
  - На овај начин е многу олеснето одржувањето и проширувањето на сложените системи на ниво на податоци.

Недостатоци на пристапот од слојот на податоци:

- Користењето вгнездени процедури **ја зголемува комплексноста** на БП и го оптоварува СУБП, бидејќи дел од програмерските работи се префрлуваат на администраторите на БП.