

# WinRDBI Educational Tool

WinRDBI is an educational tool that takes advantage of the connection between logic programming and databases to provide an evaluator for relational query languages. The [user interface](#) is implemented in Java™ [Swing](#) and the database and query languages are implemented in [Amzi! Prolog](#). The tool utilizes a common data definition facility and recognizes the following query languages:

- [relational algebra](#),
- [domain relational calculus](#),
- [tuple relational calculus](#), and
- [SQL](#).

---

## WinRDBI Links



[History](#)



[User Guide](#)



[Examples](#)



[Using WinRDBI on campus](#)



## Installing your own copy of WinRDBI

- **Before downloading and installing WinRDBI, you must have the appropriate version (1.3) of the Java Runtime Environment (JRE) installed on your machine.**
- **You must complete the registration form and acknowledge the license agreement to download a copy of WinRDBI.**

**NEW!** 22 August 2000: WinRDBI 2.0.5.130



## FAQs

- **What versions of the JRE can be used with WinRDBI?**  
WinRDBI 2.0.5.130 is designed for use with JRE 1.3.  
Note that the install program for WinRDBI assumes that the JRE is installed first.
- **Can I have multiple JREs installed on my machine at the same time?**  
Yes. Obviously, each JRE is in its own directory.  
You can edit the shortcut for WinRDBI to reference the appropriate JRE.
- **How can I edit the target of the WinRDBI shortcut?**  
The answer here is described in the context of a Windows NT machine.  
Locate the shortcut associated with the WinRDBI program in the following path:  
  
C:/Winnt/Profiles/All Users/Start Menu/Programs/WinRDBI  
  
Highlight the WinRDBI executable file.  
Open its Properties by choosing Properties off of the File menu or the right-mouse click menu.  
Choose the Shortcut Tab.  
The source of the target is displayed in a small one-line text window.  
You will probably need to scroll in the target window to see its entire contents.
- **What should the target of the WinRDBI shortcut look like?**  
DRIVE:\JREDIR\bin\javaw.exe -cp  
"DRIVE:\WinRDBIDIR;DRIVE:\WinRDBIDIR\WINRDBI.jar;"WinRDBI  
where  
  
DRIVE represents a drive specification  
JREDIR represents the path to the JRE that you want WinRDBI to use  
WinRDBIDIR represents a path to where you installed WinRDBI (e.g. Program Files\WinRDBI)
- **What are the differences between WinRDBI 2.0.2 and the latest version 2.0.5.130?**  
WinRDBI 2.0.5.130 is the first version of WinRDBI on the Java 2 Platform, which required a rewrite of the printing capabilities. A minor

syntax change was incorporated in the SQL recognized by WinRDBI. WinRDBI recognizes SQL-89, since it was originally written in 1990. At that time, the "except" operator was not part of the standard. However, WinRDBI did incorporate a "minus" operator for SQL, which has been changed to "except" in version 2.0.5.130.

- **What naming scheme is being used for the versions of WinRDBI?**  
The naming scheme for the versions of WinRDBI now includes a fourth component to document the version of the Java Runtime Environment with which WinRDBI has been designed and tested.



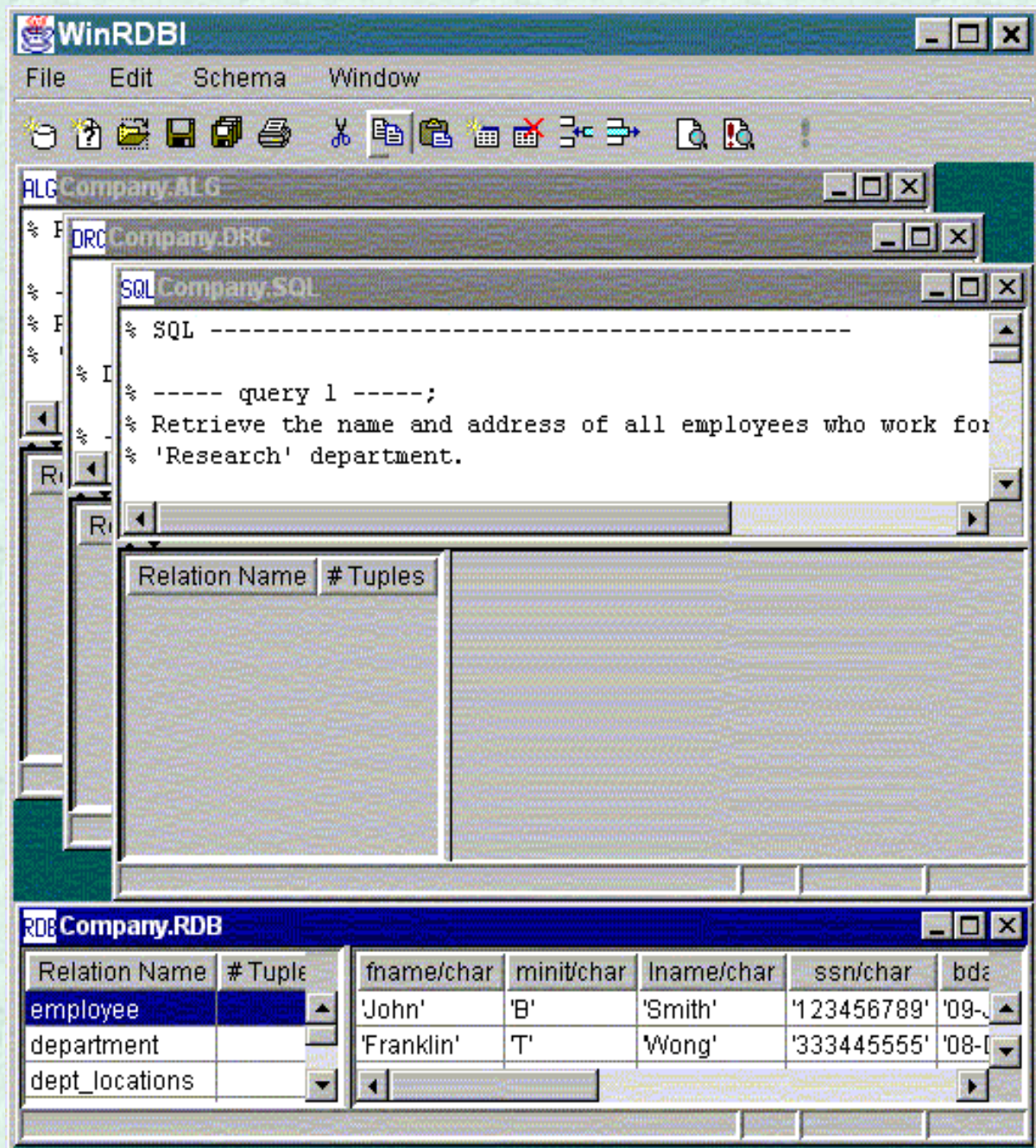
Send comments/suggestions regarding the educational tool to [WinRDBI@ASU.EDU](mailto:WinRDBI@ASU.EDU).

---

Home

# WinRDBI User Interface

The WinRDBI user interface is implemented in Java™ [Swing](#). WinRDBI 2.0 allows you to have multiple query panes open at the same time (a query pane is limited to one language: alg, drc, trc, sql) operating on one underlying database (rdb).



Home

```

% Fundamentals of Database Systems, 2nd ed., R. Elmasri and S. B. Navathe,
Benjamin/Cummings, 1994
% Schema (Chapter 6, Section 6.2, pp 145)
%   employee(fname, minit, lname, ssn, bdate, address, sex, salary, superssn, dno)
key:ssn
%   department(dname, dnumber, mgrssn, mgrstartdate) key: dnumber
%   dept_locations(dnumber, dlocation) key: dnumber,dlocation
%   project(pname, pnumber, plocation, dnum) key: pnumber
%   works_on(essn, pno, hours) key: essn, pno
%   dependent(essn, dependent_name, sex, bdate, relationship) key: essn,
dependent_name
% Queries (Chapter 6, Section 6.7, pp 170-172)

% RELATIONAL ALGEBRA-----

% ----- query 1 -----;
% Retrieve the name and address of all employees who work for the
% 'Research' department.

research_dept:=select dname='Research' (department);
research_dept_emps:=select dnumber=dno (research_dept product employee);
alg1:=project fname, lname, address (research_dept_emps);

% ----- query 2 -----;
% For every project located in 'Stafford', list the project number, the
% controlling department number, and the department manager's last name,
% address, and birthdate.

stafford_projs:=select plocation='Stafford' (projects);
contr_dept:=select dnum=dnumber (stafford_projs product department);
proj_dept_mgr:=select mgrssn=ssn (contr_dept product employee);
alg2:=project pnumber, dnum, lname, address, bdate (proj_dept_mgr);

% ----- query 3 -----;
% Find the names of employees who work on all the projects controlled
% by department number 5.

dept5_projs(pno):=project pnumber (select dnum=5 (projects));
emp_proj(ssn, pno):=project essn, pno (works_on);
emp_proj_ssns:=project ssn (emp_proj);
% All possibilities of employees working on dept5 projects.
poss_emps_dept5:=(dept5_projs product emp_proj_ssns);
% Employees that don't work on all dept5 projects..
emps_not_dept5:=project ssn (poss_emps_dept5 difference emp_proj);
result_emp_ssns:=emp_proj_ssns difference emps_not_dept5;
alg3:=project lname, fname (result_emp_ssns njoin employee);

% ----- query 4 -----;
% Make a list of project numbers for projects that involve an employee
% whose last name is 'Smith', either as a worker or as a manager of the
% department that controls the project.

smiths(essn):=project ssn (select lname='Smith' (employee));
smith_worker_projs:=project pno (works_on njoin smiths);
mgrs:=project lname, dnumber (select ssn=mgrssn (employee product department));
smith_mgrs:=select lname='Smith' (mgrs);
smith_managed_depts(dnum):=project dnumber (smith_mgrs);
smith_mgr_projs(pno):=project pnumber (smith_managed_depts njoin projects);

```

```
alg4:=smith_worker_projs union smith_mgr_projs;

% ----- query 5 -----;
% List the names of all employees with two or more dependents.

% Make two copies of employees with dependents.
empdep1(essn1, depname1):=project essn, dependent_name (dependent);
empdep2(essn2, depname2):=empdep1;
% Employees with more than one dependent.
emps_gtone_dep(ssn):=project essn1 (select (essn1=essn2) and (depname1<>depname2)
(empdep1 product empdep2));
alg5:=project lname, fname (employee njoin emps_gtone_dep);

% ----- query 6 -----;
% Retrieve the names of employees who have no dependents.

all_emps:=project ssn (employee);
emps_with_deps(ssn):=project essn (dependent);
emps_without_deps:=(all_emps difference emps_with_deps);
alg6:=project lname, fname (emps_without_deps njoin employee);

% ----- query 7 -----;
% List the names of managers who have at least one dependent.

mgrssn(ssn):=project mgrssn (department);
% Following line commented out since already defined in query 6.
% emps_with_deps(ssn):=project essn (dependent);
mgrs_with_deps:=(mgrssn intersect emps_with_deps);
alg7:=project lname, fname (mgrs_with_deps njoin employee);
```

```
% Fundamentals of Database Systems, 2nd ed., R. Elmasri and S. B. Navathe,
Benjamin/Cummings, 1994
% Schema (Chapter 6, Section 6.2, pp 145)
% employee(fname, minit, lname, ssn, bdate, address, sex, salary, superssn, dno)
key:ssn
% department(dname, dnumber, mgrssn, mgrstartdate) key: dnumber
% dept_locations(dnumber, dlocation) key: dnumber,dlocation
% project(pname, pnumber, plocation, dnum) key: pnumber
% works_on(essn, pno, hours) key: essn, pno
% dependent(essn, dependent_name, sex, bdate, relationship) key: essn,
dependent_name
% Queries (Chapter 6, Section 6.7, pp 170-172)
```

```
% DOMAIN RELATIONAL CALCULUS
```

```
% ----- query 1 -----;
% Retrieve the name and address of all employees who work for the
% 'Research' department.
```

```
drc1:=
{FNAME,LNAME,ADDRESS|
  (exists DNO)
  (department('Research',DNO,_,_) and
   employee(FNAME,_,LNAME,_,_,ADDRESS,_,_,_,DNO))};
```

```
% ----- query 2 -----;
% For every project located in 'Stafford', list the project number, the
% controlling department number, and the department manager's last name,
% address, and birthdate.
```

```
drc2:=
{PNUMBER,DNUM,LNAME,ADDRESS,BDATE|
  (exists MGRSSN)
  (projects(_,PNUMBER,'Stafford',DNUM) and
   department(_,DNUM,MGRSSN,_) and
   employee(_,_,LNAME,MGRSSN,BDATE,ADDRESS,_,_,_,_))};
```

```
% ----- query 3 -----;
% Find the names of employees who work on all the projects controlled
% by department number 5.
```

```
drc3:=
{LNAME,FNAME|
  (exists SSN)
  (employee(FNAME,_,LNAME,SSN,_,_,_,_,_,_) and
   not(exists PNUMBER)
   (projects(_,PNUMBER,_,5) and
    not(works_on(SSN,PNUMBER,_))))};
```

```
% ----- query 4 -----;
% Make a list of project numbers for projects that involve an employee
% whose last name is 'Smith', either as a worker or as a manager of the
% department that controls the project.
```

```
drc4:=
{PNO|
  (exists SSN)
  (employee(_,_, 'Smith', SSN, _, _, _, _, _)) and
```

```
(works_on(SSN,PNO,_)
  or
  (exists DNO)
    (department(_,DNO,SSN,_) and
     projects(_,PNO,_,DNO))))};
```

```
% ----- query 5 -----;
```

```
% List the names of all employees with two or more dependents.
```

```
drc5:=
```

```
{LNAME,FNAME |
  (exists SSN)
    (employee(FNAME,_,LNAME,SSN,_,_,_,_,_,_) and
     (exists DEP1,DEP2)
       (dependent(SSN,DEP1,_,_,_) and
        dependent(SSN,DEP2,_,_,_) and
         DEP1 <> DEP2))};
```

```
% ----- query 6 -----;
```

```
% Retrieve the names of employees who have no dependents.
```

```
drc6:=
```

```
{LNAME,FNAME |
  (exists SSN)
    (employee(FNAME,_,LNAME,SSN,_,_,_,_,_,_) and
     not(dependent(SSN,_,_,_,_)))};
```

```
% ----- query 7 -----;
```

```
% List the names of managers who have at least one dependent.
```

```
drc7:=
```

```
{LNAME,FNAME |
  (exists MGRSSN)
    (employee(FNAME,_,LNAME,MGRSSN,_,_,_,_,_,_) and
     department(,_,MGRSSN,_) and
     dependent(MGRSSN,_,_,_,_))};
```



```
% Fundamentals of Database Systems, 2nd ed., R. Elmasri and S. B. Navathe,
Benjamin/Cummings, 1994
% Schema (Chapter 6, Section 6.2, pp 145)
% employee(fname, minit, lname, ssn, bdate, address, sex, salary, superssn, dno)
key:ssn
% department(dname, dnumber, mgrssn, mgrstartdate) key: dnumber
% dept_locations(dnumber, dlocation) key: dnumber,dlocation
% project(pname, pnumber, plocation, dnum) key: pnumber
% works_on(essn, pno, hours) key: essn, pno
% dependent(essn, dependent_name, sex, bdate, relationship) key: essn,
dependent_name
% Queries (Chapter 6, Section 6.7, pp 170-172)
```

```
% TUPLE RELATIONAL CALCULUS-----
```

```
% ----- query 1 -----;
% Retrieve the name and address of all employees who work for the
% 'Research' department.
```

```
trc1:=
  {T.fname, T.lname, T.address |
   employee(T) and
   (exists D)
   (department (D) and D.dname='Research' and D.dnumber=T.dno)};
```

```
% ----- query 2 -----;
% For every project located in 'Stafford', list the project number, the
% controlling department number, and the department manager's last name,
% address, and birthdate.
```

```
trc2:=
  {P.pnumber, P.dnum, M.lname, M.bdate, M.address |
   projects(P) and employee(M) and P.plocation='Stafford' and
   (exists D)
   (department(D) and P.dnum=D.dnumber and D.mgrssn=M.ssn)};
```

```
% ----- query 3 -----;
% Find the names of employees who work on all the projects controlled
% by department number 5.
```

```
trc3:=
  {E.lname, E.fname |
   employee(E) and
   (forall X)
   (not (projects(X)) or (not (X.dnum=5) or
   (exists W)
   (works_on(W) and W.essn=E.ssn and X.pnumber=W.pno))))};
```

```
% ----- query 4 -----;
% Make a list of project numbers for projects that involve an employee
% whose last name is 'Smith', either as a worker or as a manager of the
% department that controls the project.
```

```
trc4:=
  {P.pnumber |
   projects(P) and
   ((exists E,W)
   (employee(E) and works_on(W) and W.pno=P.pnumber and
```

```

    E.lname='Smith' and E.ssn=W.essn)
or
(exists M,D)
(employee(M) and department(D) and P.dnum=D.dnumber and
D.mgrssn=M.ssn and M.lname='Smith'))};

```

```

% ----- query 5 -----;
% List the names of all employees with two or more dependents.

```

```

trc5:=
{E.lname, E.fname |
employee(E) and
(exists D1,D2)
(dependent(D1) and dependent(D2) and
D1.essn=E.ssn and D2.essn=E.ssn and
D1.dependent_name<>D2.dependent_name)};

```

```

% ----- query 6 -----;
% Retrieve the names of employees who have no dependents.

```

```

trc6:=
{E.lname, E.fname |
employee(E) and
not(exists D)
(dependent(D) and E.ssn=D.essn)};

```

```

% ----- query 7 -----;
% List the names of managers who have at least one dependent.

```

```

trc7:=
{E.lname, E.fname |
employee(E) and
(exists D,P)
(department(D) and dependent(P) and E.ssn=D.mgrssn and P.essn=E.ssn)};

```

## SQL

```
% Fundamentals of Database Systems, 2nd ed., R. Elmasri and S. B. Navathe,
Benjamin/Cummings, 1994
% Schema (Chapter 6, Section 6.2, pp 145)
%   employee(fname, minit, lname, ssn, bdate, address, sex, salary, superssn, dno)
key:ssn
%   department(dname, dnumber, mgrssn, mgrstartdate) key: dnumber
%   dept_locations(dnumber, dlocation) key: dnumber,dlocation
%   project(pname, pnumber, plocation, dnum) key: pnumber
%   works_on(essn, pno, hours) key: essn, pno
%   dependent(essn, dependent_name, sex, bdate, relationship) key: essn,
dependent_name
% Queries (Chapter 6, Section 6.7, pp 170-172)

% SQL -----

% ----- query 1 -----;
% Retrieve the name and address of all employees who work for the
% 'Research' department.

sql1 := select fname,lname,address
        from employee, department
        where dname = 'Research' and
              dnumber = dno;

% ----- query 2 -----;
% For every project located in 'Stafford', list the project number, the
% controlling department number, and the department manager's last name,
% address, and birthdate.

sql2 := select pnumber,dnum,lname,address,bdate
        from projects,department,employee
        where dnum = dnumber and
              mgrssn = ssn and
              plocation = 'Stafford';

% ----- query 3 -----;
% Find the names of employees who work on all the projects controlled
% by department number 5.

sql3 := select lname, fname
        from employee E
        where not exists (select *
                          from projects P
                          where P.dnum = 5 and not exists (select *
                                                            from works_on W
                                                            where W.essn = E.ssn and
                                                                  W.pno = P.pnumber));

% ----- query 4 -----;
% Make a list of project numbers for projects that involve an employee
% whose last name is 'Smith', either as a worker or as a manager of the
% department that controls the project.

sql4 := (select pnumber
        from projects, department, employee
        where dnum = dnumber and
              mgrssn = ssn and
```

```
        lname = 'Smith')
union
(select pnumber
 from projects,works_on, employee
 where pnumber = pno and
       essn = ssn and
       lname = 'Smith');
```

```
% ----- query 5 -----;
```

```
% List the names of all employees with two or more dependents.
```

```
dep_num(ssn,depend) := select ssn, count(*)
                       from employee,dependent
                       where ssn = essn
                       group by ssn;
```

```
sql5 := select lname, fname
         from employee e, dep_num d
         where e.ssn = d.ssn and
               depend >= 2;
```

```
% ----- query 6 -----;
```

```
% Retrieve the names of employees who have no dependents.
```

```
sql6 := select lname, fname
         from employee
         where not exists (select *
                          from dependent
                          where ssn = essn);
```

```
% ----- query 7 -----;
```

```
% List the names of managers who have at least one dependent.
```

```
sql7 := select lname, fname
         from employee
         where exists (select *
                      from dependent
                      where ssn = essn) and
                exists (select *
                      from department
                      where ssn = mgrssn);
```

# History of WinRDBI

The original version of the educational tool, called [RDBI](#), was implemented in Quintus Prolog and had only a command line interface. The following paper describes the motivation for RDBI and the syntax of the relational query languages recognized by the tool:

[An Educational Tool for Formal Relational Database Query Languages](#)

Suzanne W. Dietrich

Computer Science Education, Vol. 4, pp.157-184, 1993

To increase the availability of the educational tool and to provide a graphical user interface, a windows-based version, WinRDBI, was made available in Fall 1996. The following paper describes the features of WinRDBI 1.0 and provides nontrivial examples, illustrating how to write queries in the formal languages to support (a limited form of) counting and minimum/maximum queries:

[WinRDBI: A Windows-based Relational Database Educational Tool](#)

Suzanne W. Dietrich, Eric Eckert and Kevin Piscator

Proceedings of the 28th ACM SIGCSE Technical Symposium on Computer Science Education, San Jose, California, February 27 - March 1, 1997, pp. 126-130

Unfortunately, there were several limitations in the design of the graphical user interface for WinRDBI 1.0 that motivated a new version of the tool. For example, WinRDBI 1.0 allowed only one query in one language to be specified at any time. WinRDBI 2.0 is a redesign and rewrite of the graphical user interface, using the capabilities provided by Java Swing. WinRDBI 2.0 allows you to have multiple query files open at the same time operating on one underlying database. Since all versions of the educational tool recognize the same syntax for the four query languages, the publications on the previous versions of the tool provide useful examples.



We expect future releases of WinRDBI based on changes to the Java Foundation Classes (JFC). We also hope to incorporate the checking of key constraints and the inclusion of referential integrity in the future.

[Home](#)

# CSE 412/598

## Class Related Publications

*These documents are provided by the contributing authors as a means to ensure timely dissemination of scholarly and technical work on a non-commercial basis. Copyright and all rights therein are maintained by the authors or by other copyright holders, notwithstanding that they have offered their works here electronically. It is understood that all persons copying this information will adhere to the terms and constraints invoked by each author's copyright. These works may not be reposted without the explicit permission of the copyright holder.*

- [Database Theory in Practice: Learning from Cooperative Group Projects](#)  
Suzanne W. Dietrich and Susan D. Urban,  
Proceedings of the 27th ACM SIGCSE Technical Symposium on Computer Science Education  
Philadelphia, Pennsylvania, February 15-18, 1996, pp. 112-116
- [Integrating the Practical Use of a Database Product into a Theoretical Curriculum](#)  
Susan D. Urban and Suzanne W. Dietrich,  
Proceedings of the 28th ACM SIGCSE Technical Symposium on Computer Science Education,  
San Jose, California, February 27 - March 1, 1997, pp. 121-125
- [WinRDBI: A Windows-based Relational Database Educational Tool](#)  
Suzanne W. Dietrich, Eric Eckert and Kevin Piscator  
Proceedings of the 28th ACM SIGCSE Technical Symposium on Computer Science Education,  
San Jose, California, February 27 - March 1, 1997, pp. 126-130
- [A Cooperative Learning Approach to Database Group Projects:  
Integrating Theory and Practice](#)   
Suzanne W. Dietrich and Susan D. Urban,  
IEEE Transactions on Education, November 1998, CD-ROM Directory 06, 11 p.
- [An Educational Tool for Formal Relational Database Query Languages](#)   
Suzanne W. Dietrich,  
Computer Science Education, Vol. 4, pp. 157-184, 1993.

Home

# WinRDBI Examples

Download Example	Reference
<a href="#">Company</a>	Fundamentals of Database Systems R. Elmasri and S. Navathe Benjamin/Cummings Publishing, 2nd edition, 1994.
<a href="#">University</a> <a href="#">Cars</a>	<a href="#">An Educational Tool for Formal Relational Database Query Languages</a> Suzanne W. Dietrich, Computer Science Education, Vol. 4, pp. 157-184, 1993.
<a href="#">Max Dept Salary</a>	<a href="#">WinRDBI: A Windows-based Relational Database Educational Tool</a> Suzanne W. Dietrich, Eric Eckert and Kevin Piscator Proceedings of the 28th ACM SIGCSE Technical Symposium on Computer Science Education, San Jose, California, February 27 - March 1, 1997, pp. 126-130

[Home](#)

# WinRDBI Use at ASU

## GWC 242/250

- Login on any Windows NT workstation. Since WinRDBI is a Windows application, you will not find WinRDBI on the Solaris or Indy workstations.
  - Click on the Start menu and go to Programs.
  - Point to WinRDBI in the Programs menu, and click on WinRDBI 2.0
  - Note: If you print from GWC 242/250, collect your printouts downstairs (GWC 181).
- 

## IT Sites

### To Install:

1. You need to get to the CSE412 Instructor Volume, either by an IV mount shortcut on the desktop or through Start->Programs->Instructor Volume.
2. Double click on CSE412. This will create a shortcut to the U drive on the desktop (you will get no message stating that the shortcut has been created. After double clicking, immediately look for it on the desktop).
3. Double click on the U drive shortcut.
4. Click on the WinRDBI icon to install WinRDBI.
5. An icon for WinRDBI will be created on the desktop.

### To Run WinRDBI:

- Double click the WinRDBI icon on the desktop  
or
- Click on the Start menu and go to Programs.  
Point to WinRDBI in the Programs menu, and click on WinRDBI

It may take 2-3 minutes for WinRDBI to load. Be patient! You will first see an ASU splash screen before the main WinRDBI screen.



Home