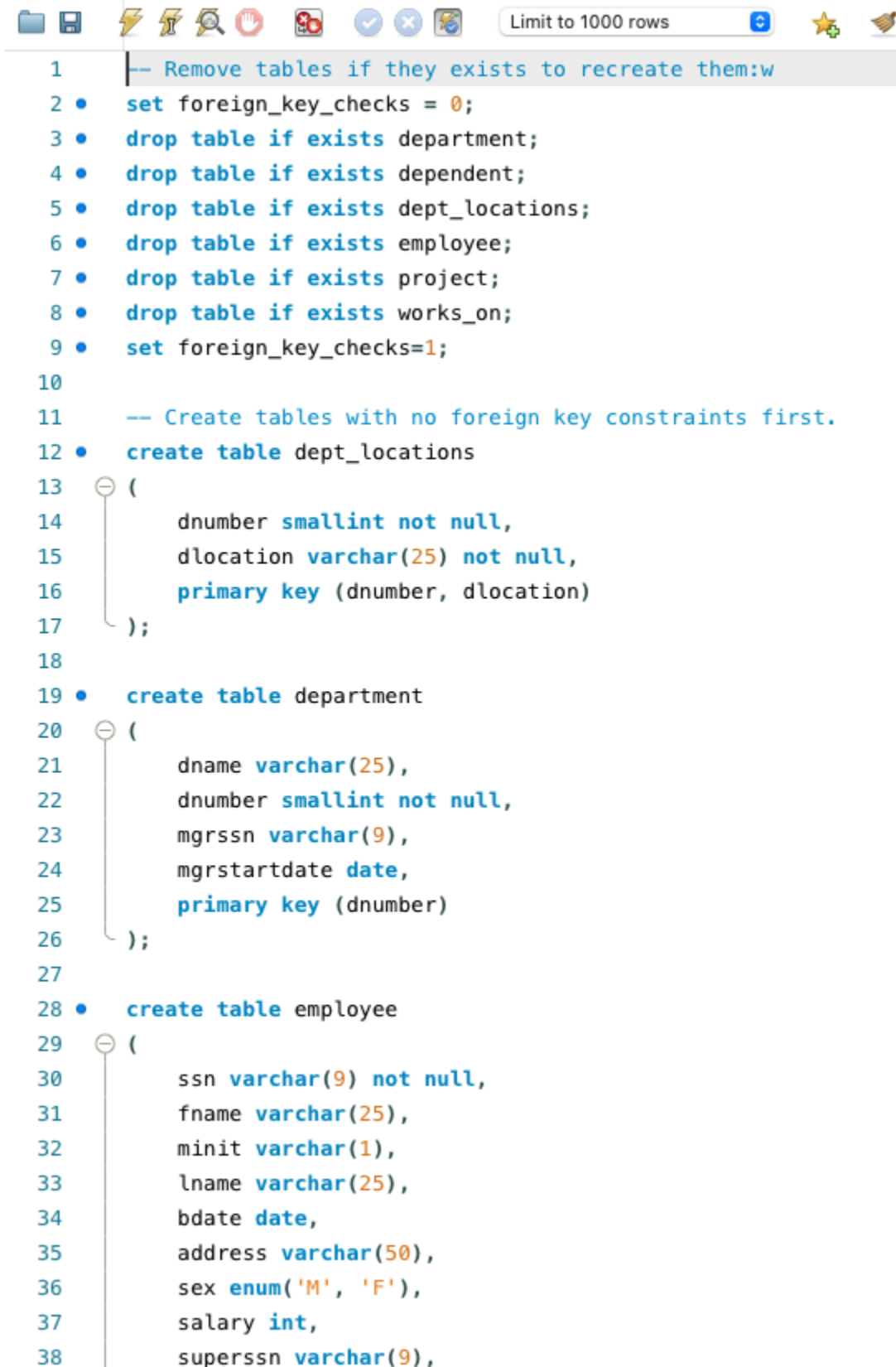





Build Database:


```
1  -- Remove tables if they exists to recreate them:w
2  • set foreign_key_checks = 0;
3  • drop table if exists department;
4  • drop table if exists dependent;
5  • drop table if exists dept_locations;
6  • drop table if exists employee;
7  • drop table if exists project;
8  • drop table if exists works_on;
9  • set foreign_key_checks=1;
10
11  -- Create tables with no foreign key constraints first.
12  • create table dept_locations
13  (
14      dnumber smallint not null,
15      dlocation varchar(25) not null,
16      primary key (dnumber, dlocation)
17  );
18
19  • create table department
20  (
21      dname varchar(25),
22      dnumber smallint not null,
23      mgrssn varchar(9),
24      mgrstartdate date,
25      primary key (dnumber)
26  );
27
28  • create table employee
29  (
30      ssn varchar(9) not null,
31      fname varchar(25),
32      minit varchar(1),
33      lname varchar(25),
34      bdate date,
35      address varchar(50),
36      sex enum('M', 'F'),
37      salary int,
38      superssn varchar(9),
```

39  Limit to 1000 rows  

```

40     primary key (ssn)
41 );
42
43 • create table works_on
44 (
45     essn varchar(9) not null,
46     pno smallint not null,
47     hours decimal(3, 1),
48     primary key (essn, pno)
49 );
50
51 • create table project
52 (
53     pname varchar(25),
54     pnumber smallint not null,
55     plocation varchar(25),
56     dnum smallint,
57     primary key (pnumber)
58 );
59
60 • create table dependent
61 (
62     essn varchar(9),
63     dependent_name varchar(25),
64     sex enum('M', 'F'),
65     bdate date,
66     relationship varchar(25),
67     primary key (essn, dependent_name)
68 );
69
70 -- Populate DEPT_LOCATIONS
71 • insert into dept_locations (dnumber, dlocation)
72 values
73     (1, 'Houston'),
74     (4, 'Stafford'),
75     (5, 'Bellaire'),
76     (5, 'Sugarland'),
77     (5, 'Houston');

```



```


73      (1, 'Houston'),
74      (4, 'Stafford'),
75      (5, 'Bellaire'),
76      (5, 'Sugarland'),
77      (5, 'Houston');
78
79  -- Populate DEPARTMENT
80 • insert into department (dname, dnumber, mgrssn, mgrstartdate
81  values
82      ('Research', 5, '333445555', '1988-05-22'),
83      ('Administration', 4, '987654321', '1995-01-01'),
84      ('Headquarters', 1, '888665555', '1981-06-19');
85
86  -- Populate PROJECT
87 • insert into project (pname, pnumber, plocation, dnum)
88  values
89      ('ProductX', 1, 'Bellaire', 5),
90      ('ProductY', 2, 'Sugarland', 5),
91      ('ProductZ', 3, 'Houston', 5),
92      ('Computerization', 10, 'Stafford', 4),
93      ('Reorganization', 20, 'Houston', 1),
94      ('Newbenefits', 30, 'Stafford', 4);
95
96
97  -- Populate WORKS_ON
98 • insert into works_on (essn, pno, hours)
99  values
100      ('123456789', 1, 32.5),
101      ('123456789', 2, 7.5),
102      ('666884444', 3, 40.0),
103      ('453453453', 1, 20.0),
104      ('453453453', 2, 20.0),
105      ('333445555', 2, 10.0),
106      ('333445555', 3, 10.0),
107      ('333445555', 10, 10.0),
108      ('333445555', 20, 10.0),
109      ('999887777', 30, 30.0),
110      ('999887777', 10, 10.0),
111  ...

```

```

100
107
108         ('333445555', 20, 10.0),
109         ('999887777', 30, 30.0),
110         ('999887777', 10, 10.0),
111         ('987987987', 10, 35.0),
112         ('987987987', 30, 5.0),
113         ('987654321', 30, 20.0),
114         ('987654321', 20, 15.0),
115         ('888665555', 20, null);
116
117 -- Populate EMPLOYEE
118 • insert into employee (ssn, fname, minit, lname, bdate, address1, address2)
119 values
120     ('123456789', 'John', 'B', 'Smith', '1965-01-09', '731 F', '123456789'),
121     ('333445555', 'Franklin', 'T', 'Wong', '1955-12-08', '631 F', '123456789'),
122     ('999887777', 'Alicia', 'J', 'Zelaya', '1968-07-19', '331 F', '123456789'),
123     ('987654321', 'Jennifer', 'S', 'Wallace', '1941-06-20', '123 F', '123456789'),
124     ('666884444', 'Ramesh', 'K', 'Narayan', '1962-09-15', '987 F', '123456789'),
125     ('453453453', 'Joyce', 'A', 'English', '1972-07-31', '567 F', '123456789'),
126     ('987987987', 'Ahmad', 'V', 'Jabbar', '1969-03-29', '980 F', '123456789'),
127     ('888665555', 'James', 'E', 'Borg', '1937-11-10', '450 S', '123456789');
128
129 -- Populate DEPENDENT
130 • insert into dependent (essn, dependent_name, sex, bdate, relationship)
131 values
132     ('333445555', 'Alice', 'F', '1986-04-05', 'Daughter'),
133     ('333445555', 'Theodore', 'M', '1983-10-25', 'Son'),
134     ('333445555', 'Joy', 'F', '1958-05-03', 'Spouse'),
135     ('987654321', 'Abner', 'M', '1942-02-28', 'Spouse'),
136     ('123456789', 'Michael', 'M', '1988-01-04', 'Son'),
137     ('123456789', 'Alice', 'F', '1988-12-30', 'Daughter'),
138     ('123456789', 'Elizabeth', 'F', '1967-05-05', 'Spouse');
139
140 -- Add the foreign key constraints on each of the tables created
141
142 -- EMPLOYEE.DNO is a foreign key to DEPARTMENT.DNUMBER
143 • alter table employee
144 add foreign key (dno) references department(dnumber);
145

```

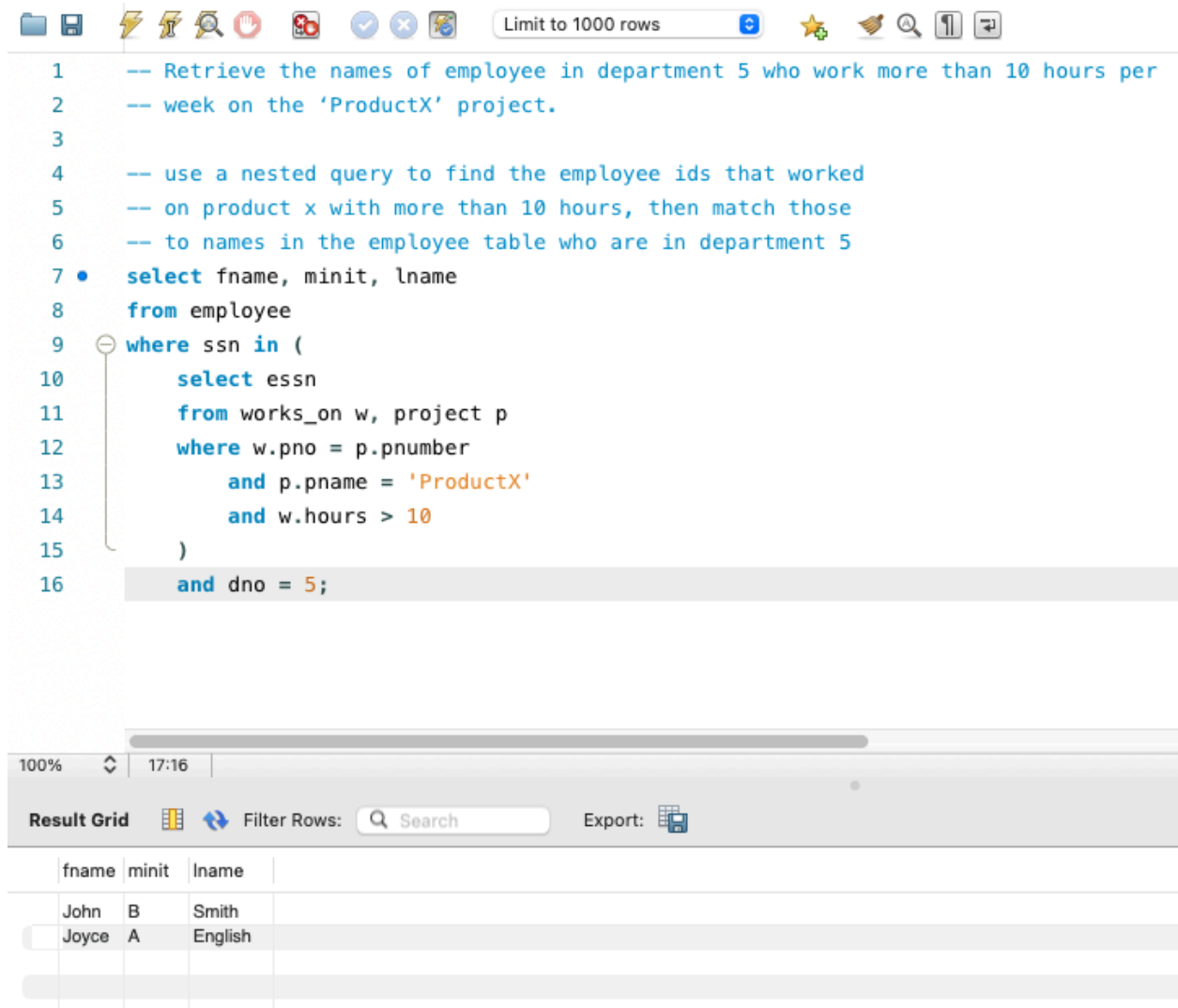


```

132      ('333445555', 'Alice', 'F', '1986-04-05', 'Daughter'),
133      ('333445555', 'Theodore', 'M', '1983-10-25', 'Son'),
134      ('333445555', 'Joy', 'F', '1958-05-03', 'Spouse'),
135      ('987654321', 'Abner', 'M', '1942-02-28', 'Spouse'),
136      ('123456789', 'Michael', 'M', '1988-01-04', 'Son'),
137      ('123456789', 'Alice', 'F', '1988-12-30', 'Daughter'),
138      ('123456789', 'Elizabeth', 'F', '1967-05-05', 'Spouse');
139
140  -- Add the foreign key constraints on each of the tables created
141
142  -- EMPLOYEE.DNO is a foreign key to DEPARTMENT.DNUMBER
143  • alter table employee
144    add foreign key (dno) references department(dnumber);
145
146  -- DEPARTMENT.MGRSSN is FK to EMPLOYEE.SSN
147  • alter table department
148    add foreign key (mgrssn) references employee(ssn);
149
150  -- DEPT_LOCATIONS.DNUMBER references DEPARTMENT.DNUMBER
151  • alter table dept_locations
152    add foreign key (dnumber) references department(dnumber);
153
154  -- WORKS_ON.ESSN references EMPLOYEE.SSN
155  • alter table works_on
156    add foreign key (essn) references employee (ssn);
157
158  -- WORKS_ON.PNO references PROJECT.PNUMBER
159  • alter table works_on
160    add foreign key (pno) references project (pnumber);
161
162  -- PROJECT.DNUM references DEPARTMENT.DNUMBER
163  • alter table project
164    add foreign key (dnum) references department(dnumber);
165
166  -- DEPENDENT.ESSN references EMPLOYEE.SSN
167  • alter table dependent
168    add foreign key (essn) references employee(ssn);

```

(a) Retrieve the names of employees in department 5 who work more than 10 hours per week on the 'ProductX' project.



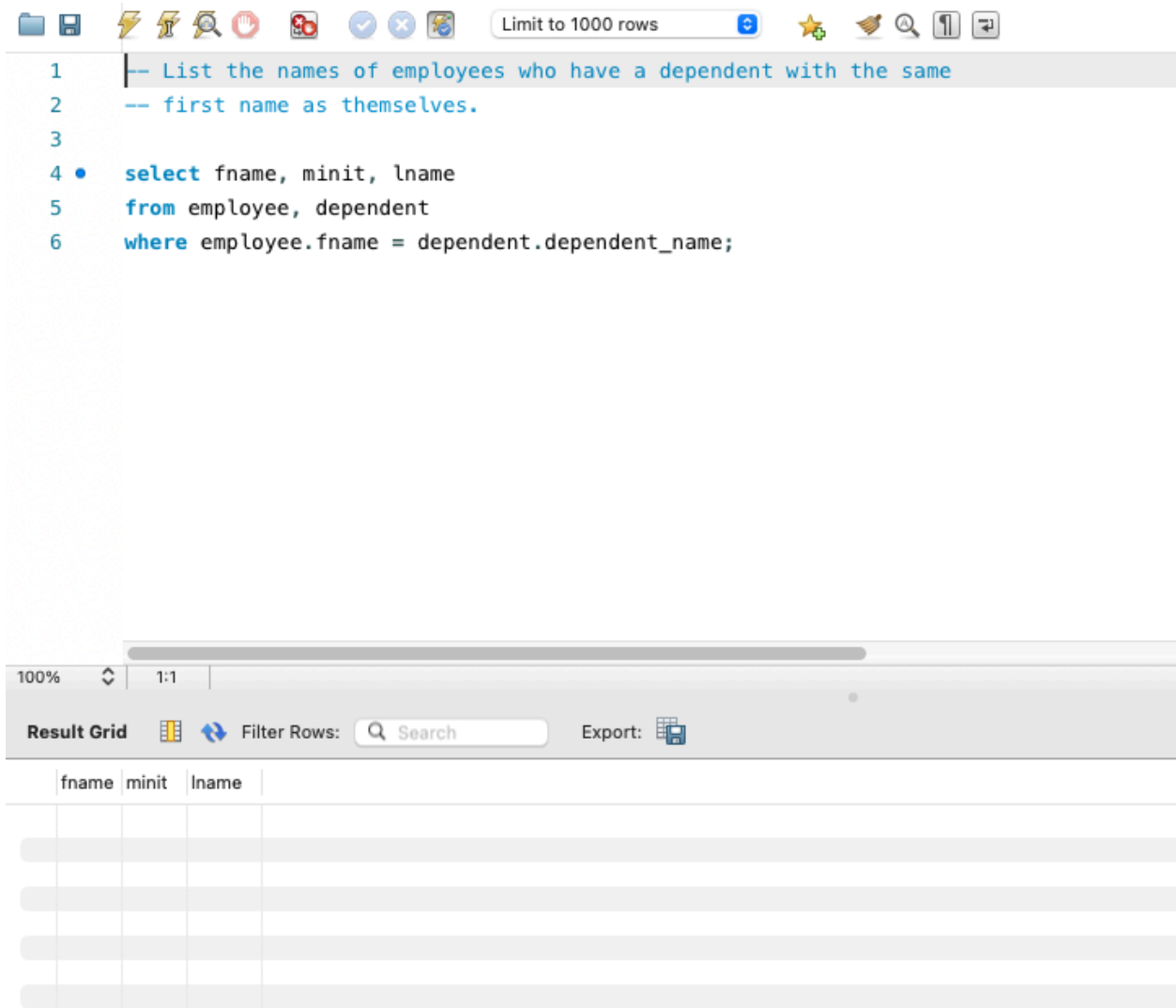
The screenshot shows a SQL IDE interface. The top toolbar includes icons for file operations, execution, and a 'Limit to 1000 rows' dropdown. The SQL editor contains the following query:

```
1  -- Retrieve the names of employee in department 5 who work more than 10 hours per
2  -- week on the 'ProductX' project.
3
4  -- use a nested query to find the employee ids that worked
5  -- on product x with more than 10 hours, then match those
6  -- to names in the employee table who are in department 5
7  • select fname, minit, lname
8     from employee
9     where ssn in (
10         select essn
11         from works_on w, project p
12         where w.pno = p.pnumber
13             and p.pname = 'ProductX'
14             and w.hours > 10
15     )
16     and dno = 5;
```

Below the editor is a 'Result Grid' section with a search bar and an 'Export' button. The results table has three columns: 'fname', 'minit', and 'lname'. It contains two rows of data:

fname	minit	lname
John	B	Smith
Joyce	A	English

(b) List the names of employees who have a dependent with the same first name as themselves.



The screenshot shows a SQL IDE interface. At the top, there is a toolbar with various icons and a dropdown menu set to "Limit to 1000 rows". Below the toolbar, a SQL query is entered in a text area:

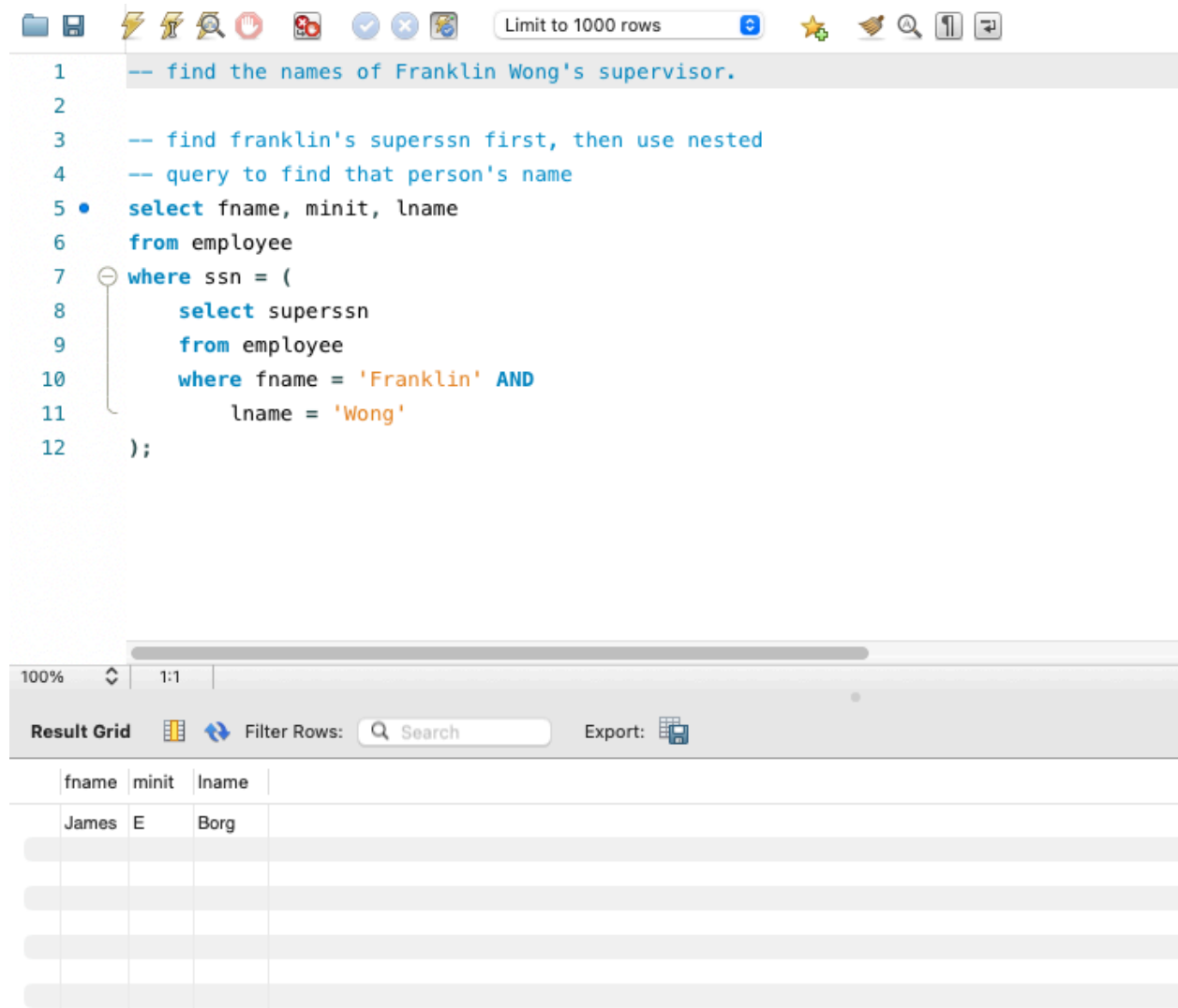
```
1  -- List the names of employees who have a dependent with the same
2  -- first name as themselves.
3
4  •  select fname, minit, lname
5     from employee, dependent
6     where employee.fname = dependent.dependent_name;
```

Below the query editor, there is a horizontal scrollbar. Underneath that is a control bar with "100%", a refresh icon, "1:1", and a "Result Grid" label. To the right of "Result Grid" are icons for "Filter Rows" (a funnel icon), a search input field with the placeholder "Search", and an "Export" button with a document icon.

The main area of the IDE displays a result grid with the following columns: `fname`, `minit`, and `lname`. The grid is currently empty, showing only the header row and several blank rows below it.

fname	minit	lname

(c) Find the names of Franklin Wong's supervisor.



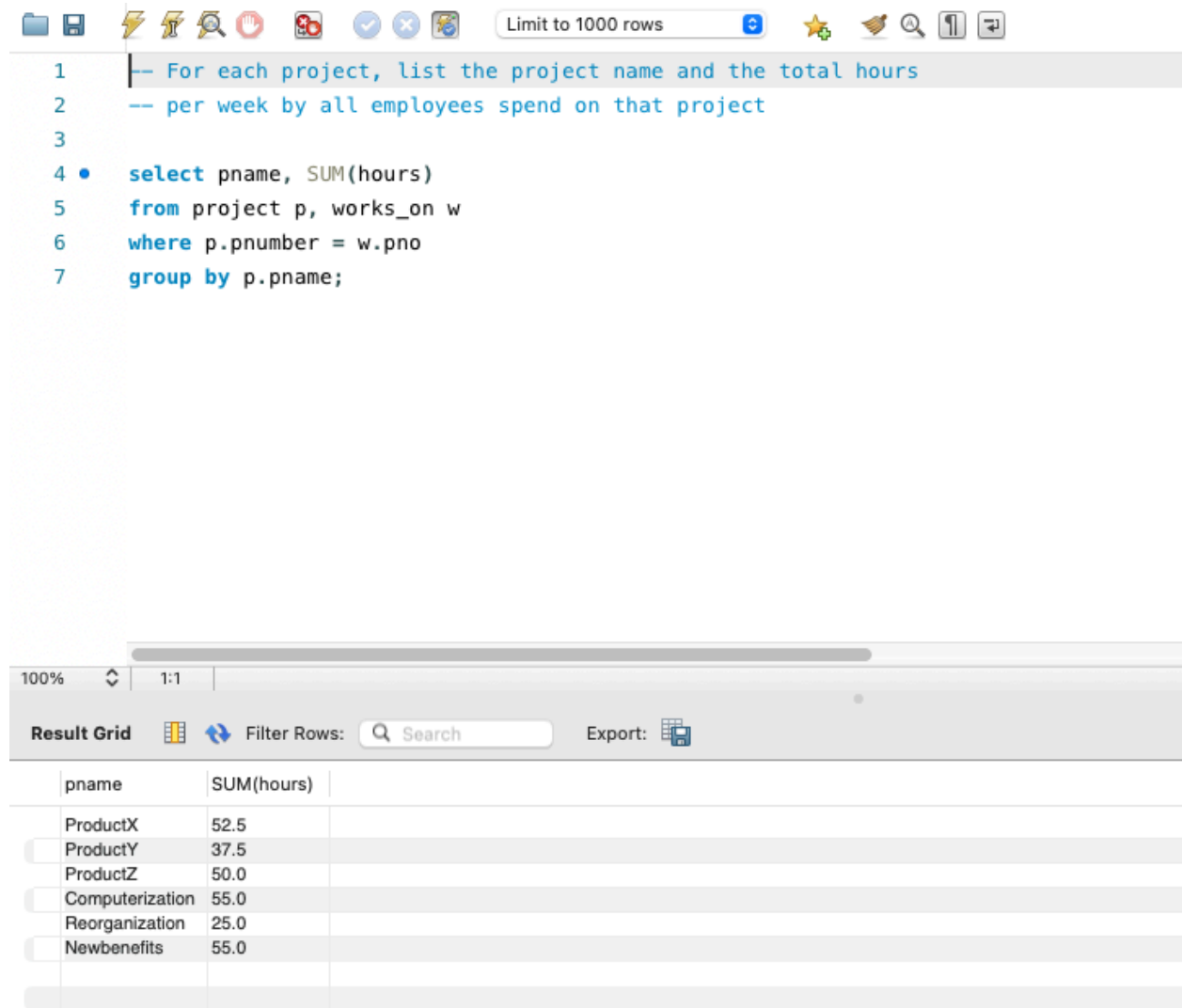
```
1  -- find the names of Franklin Wong's supervisor.
2
3  -- find franklin's superssn first, then use nested
4  -- query to find that person's name
5  • select fname, minit, lname
6     from employee
7  where ssn = (
8         select superssn
9         from employee
10        where fname = 'Franklin' AND
11              lname = 'Wong'
12    );
```

100% 1:1

Result Grid Filter Rows: Search Export:

fname	minit	lname
James	E	Borg

(d) For each project, list the project name and the total hours per week (by all employees) spent on that project.



The screenshot shows a SQL IDE interface. At the top, there is a toolbar with various icons and a dropdown menu set to "Limit to 1000 rows". Below the toolbar, a SQL query is entered in a text area:

```
1  -- For each project, list the project name and the total hours
2  -- per week by all employees spend on that project
3
4  •  select pname, SUM(hours)
5     from project p, works_on w
6     where p.pnumber = w.pno
7     group by p.pname;
```

Below the query editor, there is a "Result Grid" section. It includes a "Filter Rows:" search bar and an "Export:" button. The results are displayed in a table with two columns: "pname" and "SUM(hours)".

pname	SUM(hours)
ProductX	52.5
ProductY	37.5
ProductZ	50.0
Computerization	55.0
Reorganization	25.0
Newbenefits	55.0

(f) Retrieve the names of employees who do not work on any project.

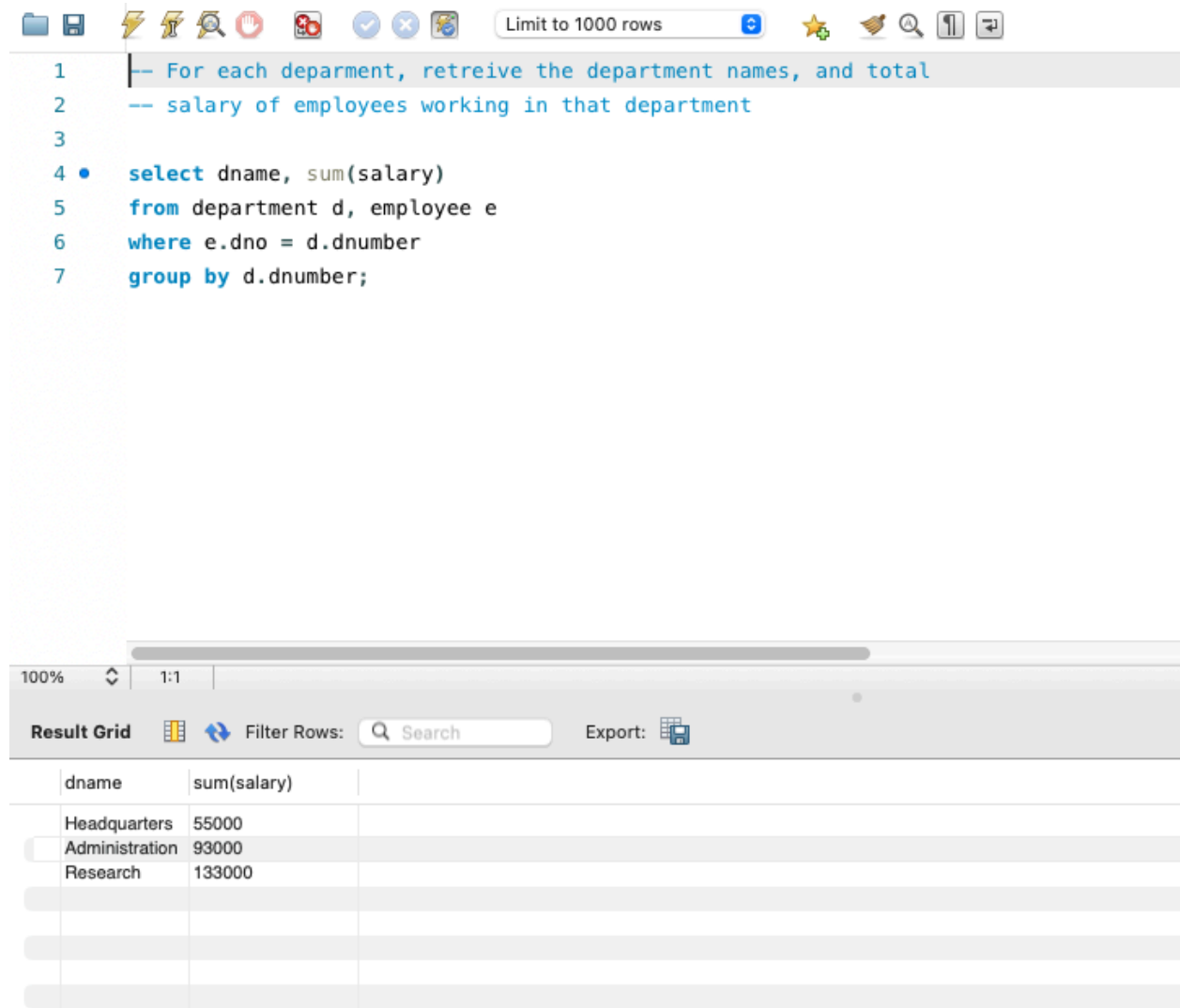
The screenshot shows a SQL IDE interface. The top toolbar includes icons for file operations, execution, and search, along with a 'Limit to 1000 rows' dropdown. The SQL editor contains the following query:

```
1  -- Retrieve the name of employees who do not work on any project
2
3  -- find the ESSN of all employees who worked on a project
4  • select fname, minit, lname
5     from employee
6  where ssn not in (
7      select essn
8      from works_on
9  );
```

Below the editor is a 'Result Grid' section. It includes a 'Filter Rows' search bar and an 'Export' button. The grid has columns for 'fname', 'minit', and 'lname', and it is currently empty.

fname	minit	lname

(g) For each department, retrieve the department name, and the total salary of employees working in that department.



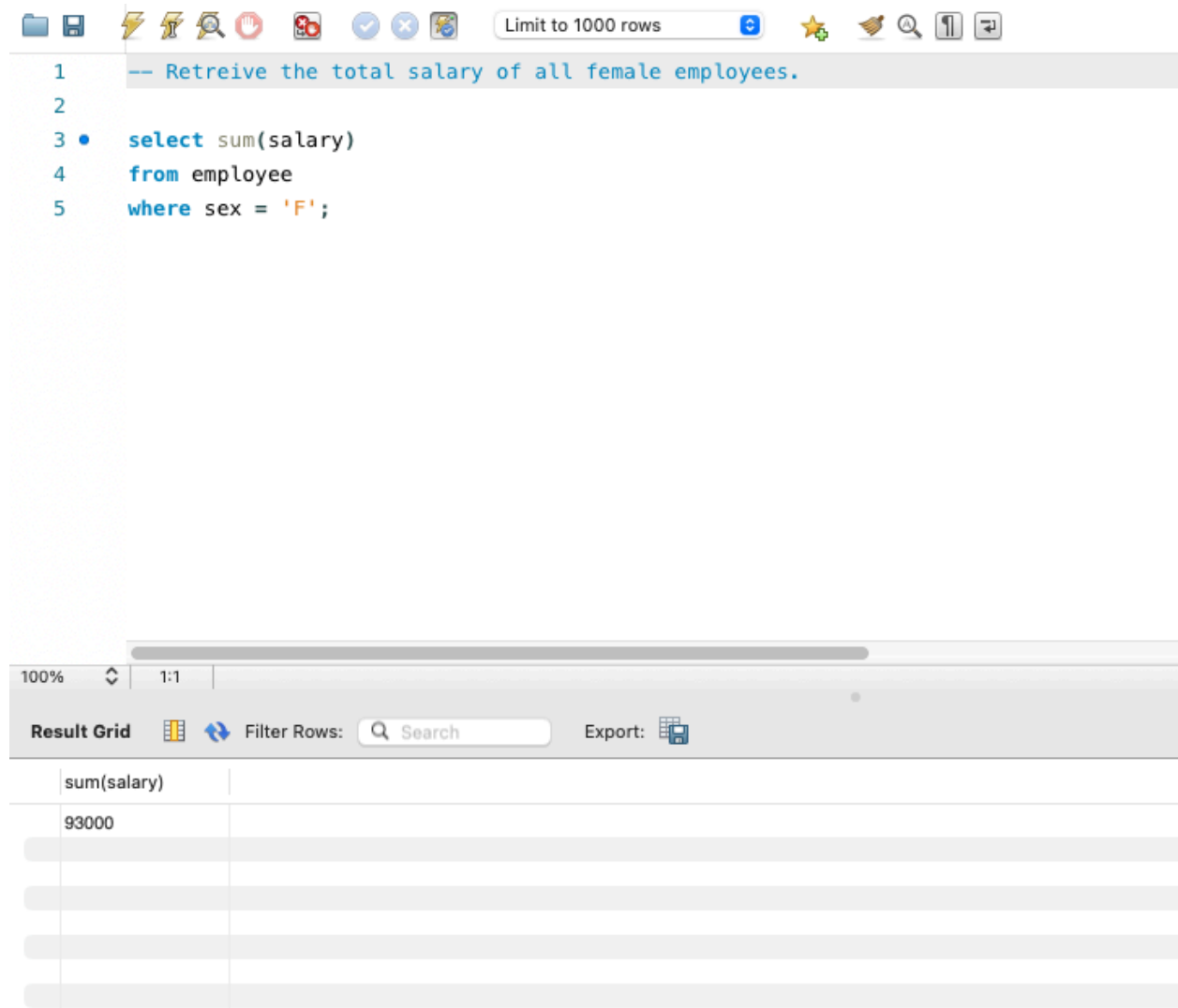
The screenshot shows a SQL IDE interface. At the top, there is a toolbar with various icons and a text input field that says "Limit to 1000 rows". Below the toolbar, a SQL query is entered in a text area:

```
1  -- For each department, retrieve the department names, and total
2  -- salary of employees working in that department
3
4  •  select dname, sum(salary)
5     from department d, employee e
6     where e.dno = d.dnumber
7     group by d.dnumber;
```

Below the query editor, there is a horizontal scrollbar. Underneath that, a "Result Grid" section is visible. It includes a "Filter Rows:" label, a search input field, and an "Export:" button. The result grid itself is a table with two columns: "dname" and "sum(salary)". The first three rows of the table are populated with data:

dname	sum(salary)
Headquarters	55000
Administration	93000
Research	133000

(h) Retrieve the total salary of all female employees.



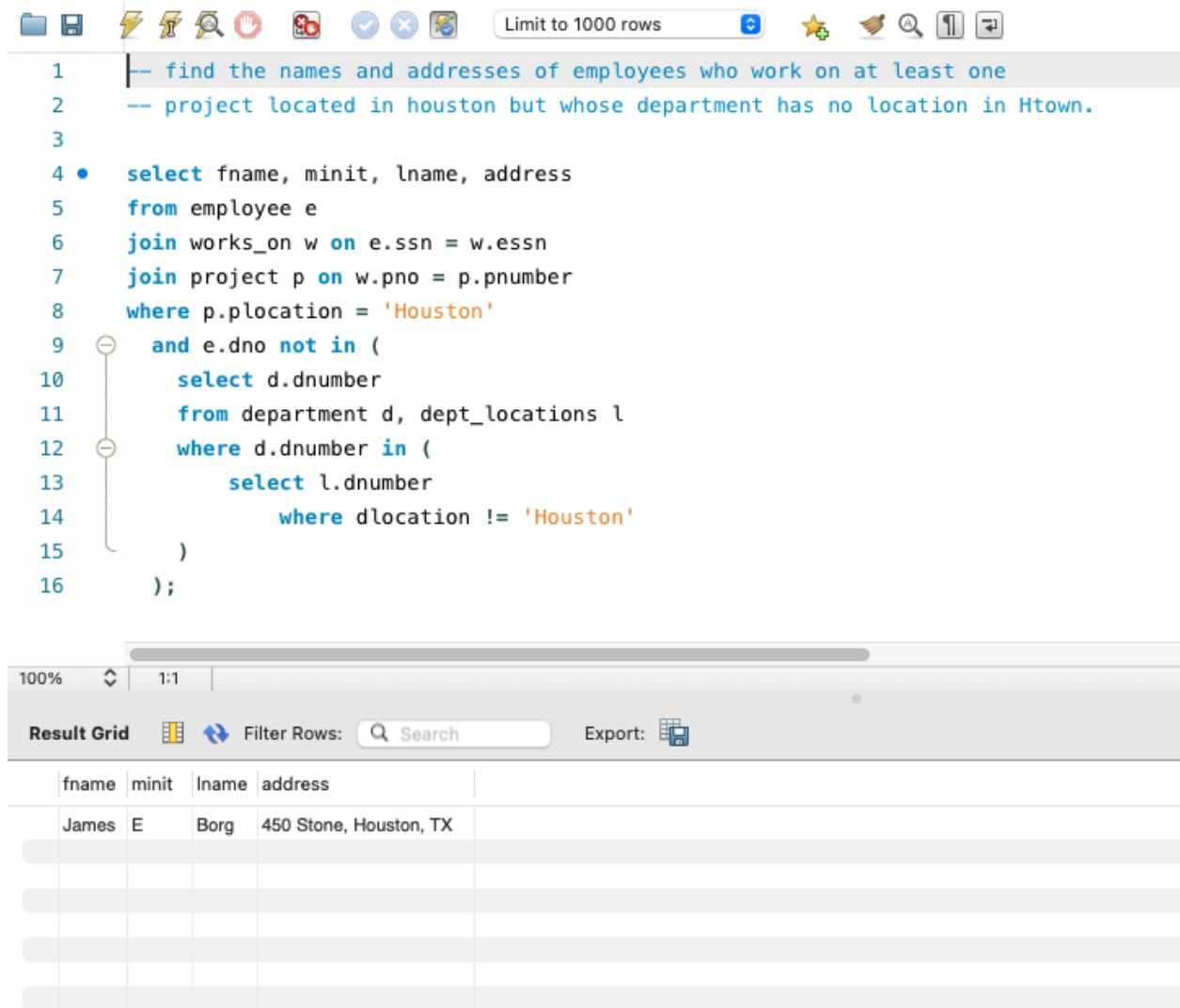
The screenshot shows a SQL IDE interface. At the top, there is a toolbar with various icons and a dropdown menu set to "Limit to 1000 rows". Below the toolbar, a SQL query is entered in a text area:

```
1 -- Retrieve the total salary of all female employees.  
2  
3 • select sum(salary)  
4 from employee  
5 where sex = 'F';
```

Below the query editor, there is a "Result Grid" section. It includes a "Filter Rows:" search bar and an "Export:" button. The result grid displays the following data:

sum(salary)
93000

(i) Find the names and addresses of employees who work on at least one project located in Houston but whose department has no location in Houston.



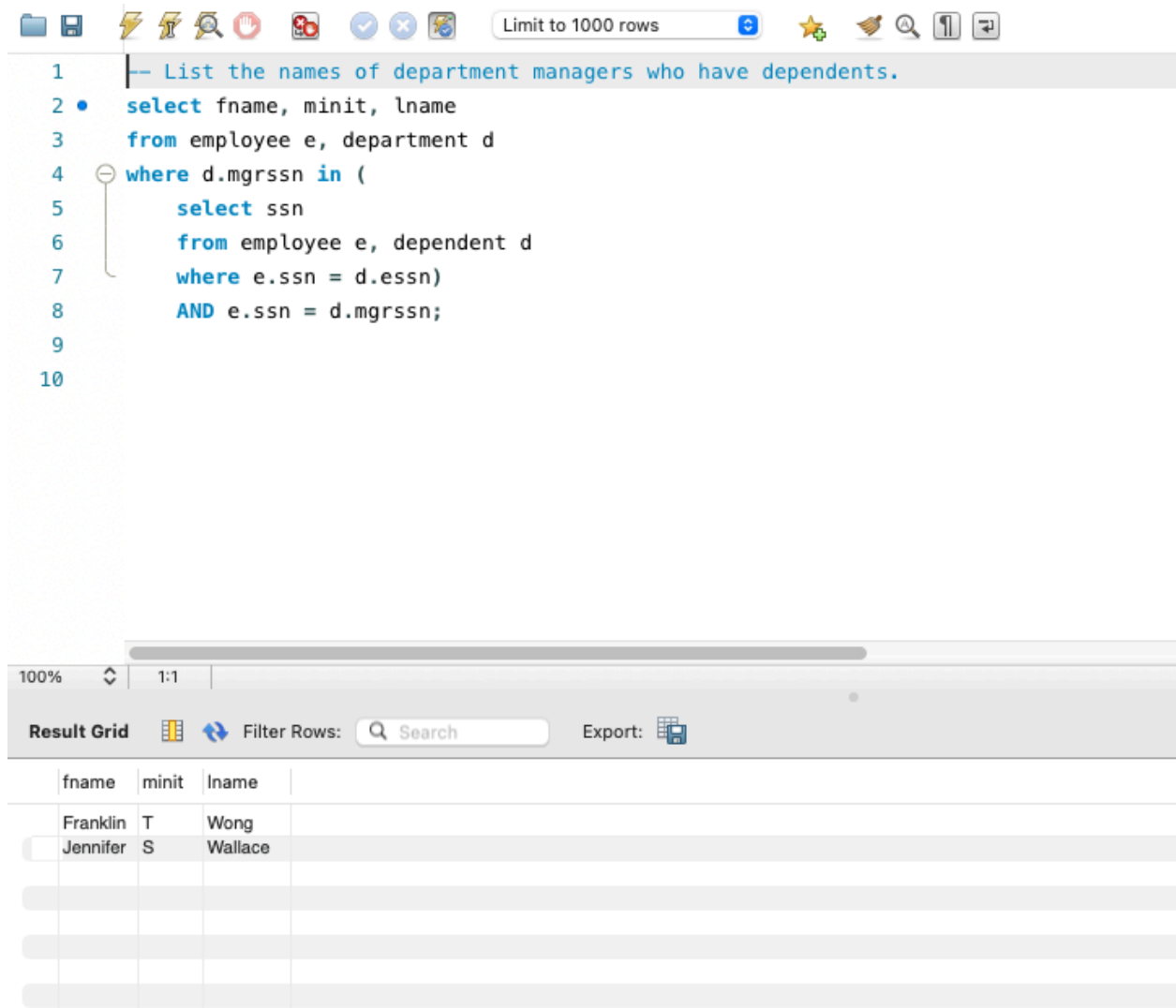
The screenshot shows a SQL IDE interface. At the top, there is a toolbar with various icons and a dropdown menu set to "Limit to 1000 rows". Below the toolbar, a SQL query is entered in a text area. The query is as follows:

```
1  -- find the names and addresses of employees who work on at least one
2  -- project located in houston but whose department has no location in Htown.
3
4  •  select fname, minit, lname, address
5     from employee e
6     join works_on w on e.ssn = w.essn
7     join project p on w.pno = p.pnumber
8     where p.plocation = 'Houston'
9     and e.dno not in (
10         select d.dnumber
11         from department d, dept_locations l
12         where d.dnumber in (
13             select l.dnumber
14             where dlocation != 'Houston'
15         )
16     );
```

Below the query editor, there is a "Result Grid" section. It includes a "Filter Rows:" search bar and an "Export:" button. The result grid displays the following data:

fname	minit	lname	address
James	E	Borg	450 Stone, Houston, TX

(j) List the last names of department managers who have dependents.



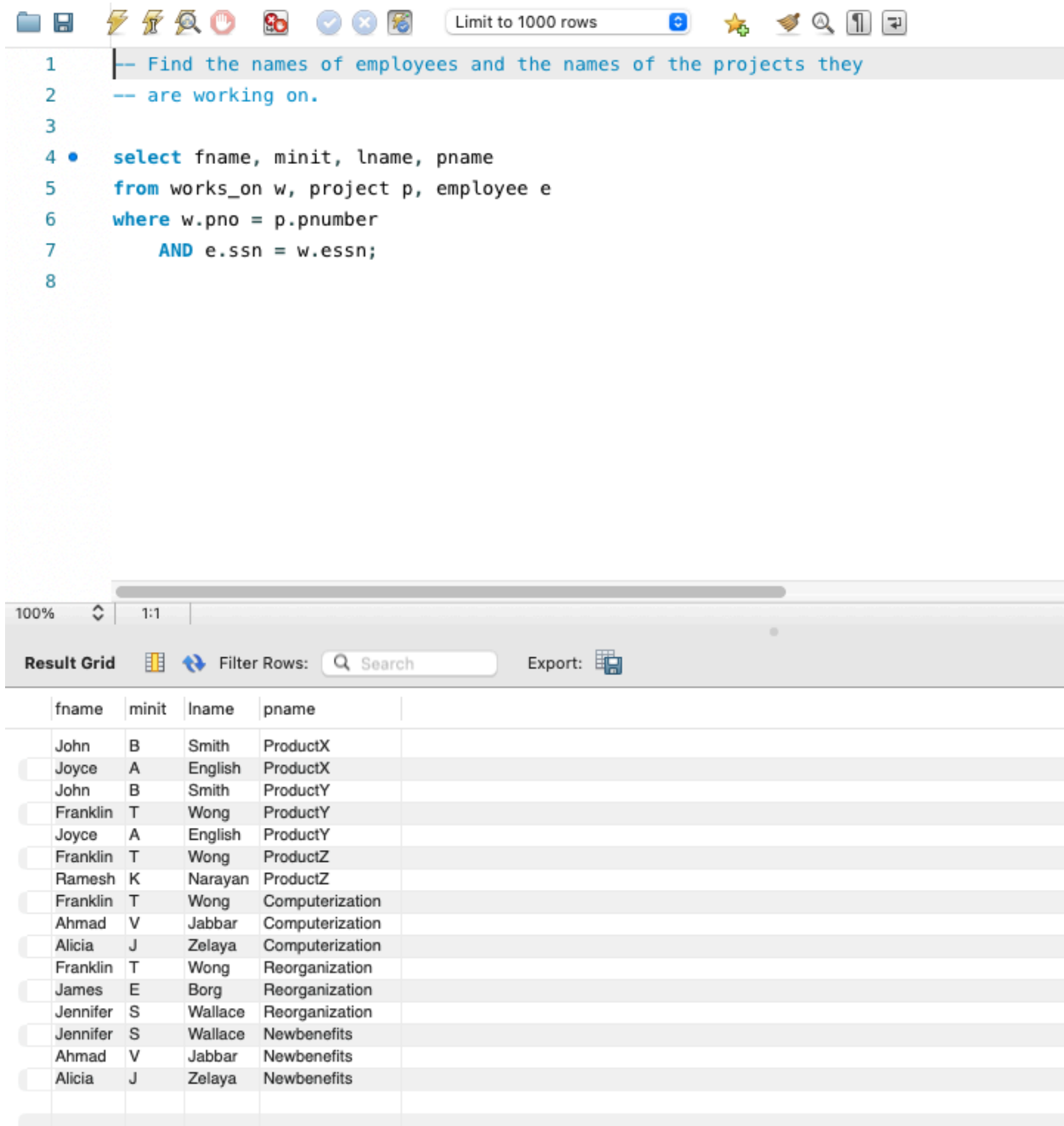
The screenshot shows a SQL IDE interface. At the top, there is a toolbar with various icons and a text box that says "Limit to 1000 rows". Below the toolbar, a SQL query is entered in a text area. The query is as follows:

```
1  -- List the names of department managers who have dependents.
2  • select fname, minit, lname
3     from employee e, department d
4  where d.mgrssn in (
5      select ss
6      from employee e, dependent d
7      where e.ssn = d.essn)
8      AND e.ssn = d.mgrssn;
9
10
```

Below the query editor, there is a "Result Grid" section. It includes a "Filter Rows:" search bar and an "Export:" button. The result grid displays the following data:

fname	minit	lname
Franklin	T	Wong
Jennifer	S	Wallace

(k) Find the names of employees and the names of the projects that the employees are working on.



The screenshot shows a SQL IDE interface. At the top, there is a toolbar with various icons and a dropdown menu set to "Limit to 1000 rows". Below the toolbar, a SQL query is entered in a text area:

```

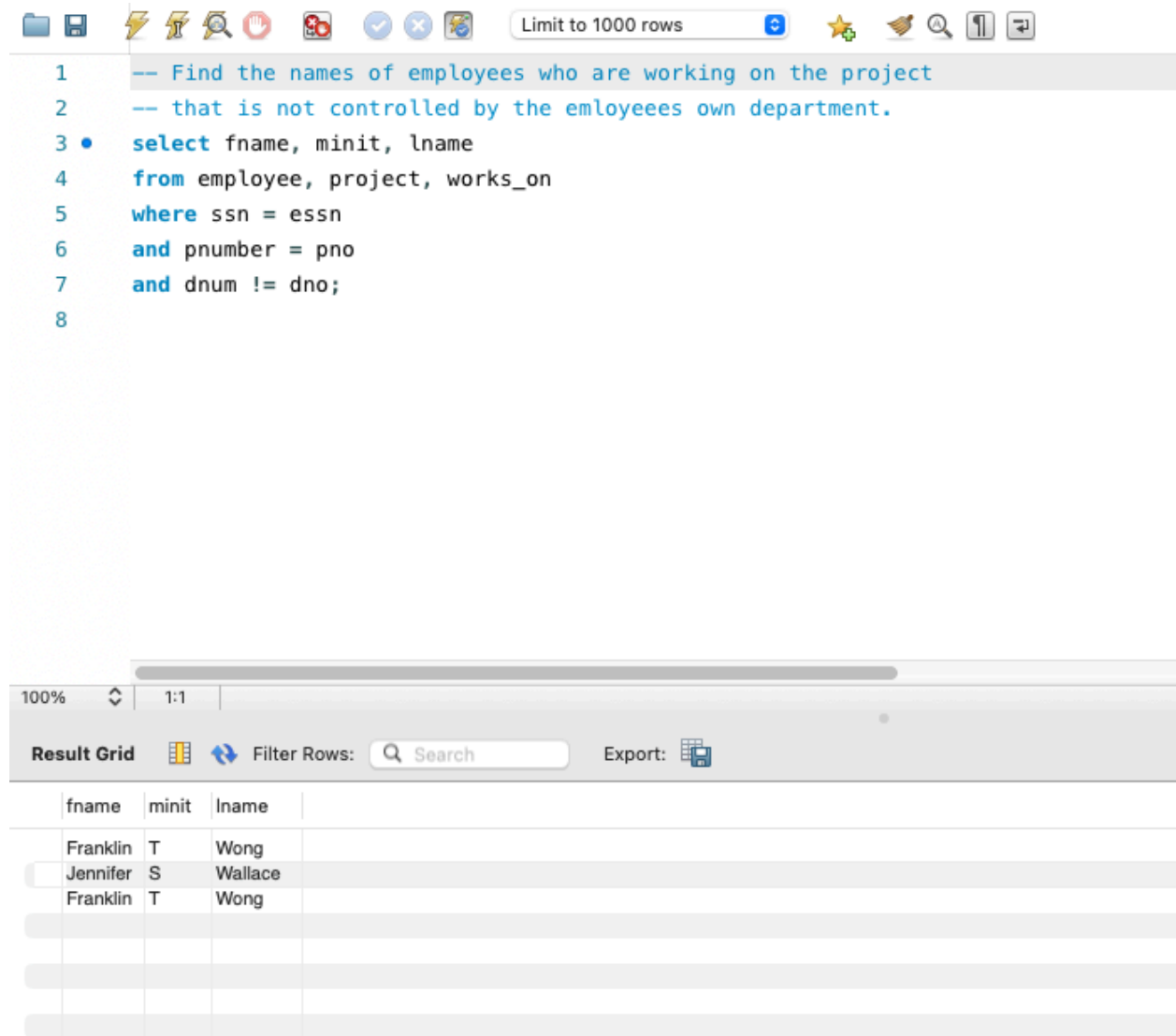
1  -- Find the names of employees and the names of the projects they
2  -- are working on.
3
4  •  select fname, minit, lname, pname
5     from works_on w, project p, employee e
6     where w.pno = p.pnumber
7         AND e.ssn = w.essn;
8

```

Below the query editor, there is a "Result Grid" section. It includes a "Filter Rows:" search bar and an "Export:" button. The result grid displays the following data:

fname	minit	lname	pname
John	B	Smith	ProductX
Joyce	A	English	ProductX
John	B	Smith	ProductY
Franklin	T	Wong	ProductY
Joyce	A	English	ProductY
Franklin	T	Wong	ProductZ
Ramesh	K	Narayan	ProductZ
Franklin	T	Wong	Computerization
Ahmad	V	Jabbar	Computerization
Alicia	J	Zelaya	Computerization
Franklin	T	Wong	Reorganization
James	E	Borg	Reorganization
Jennifer	S	Wallace	Reorganization
Jennifer	S	Wallace	Newbenefits
Ahmad	V	Jabbar	Newbenefits
Alicia	J	Zelaya	Newbenefits

(l) Find the names of employees who are working on the project that is not controlled by the employee's own department.



The screenshot shows a SQL IDE interface. At the top, there is a toolbar with various icons and a dropdown menu set to "Limit to 1000 rows". Below the toolbar, a SQL query is entered in a text area:

```
1  -- Find the names of employees who are working on the project
2  -- that is not controlled by the employees own department.
3  • select fname, minit, lname
4     from employee, project, works_on
5  where ssn = essn
6     and pnumber = pno
7     and dnum != dno;
8
```

Below the query editor, there is a horizontal scrollbar. Underneath that, a "Result Grid" section is visible, showing the results of the query. The grid has four columns: "fname", "minit", "lname", and an empty column. The results are as follows:

fname	minit	lname	
Franklin	T	Wong	
Jennifer	S	Wallace	
Franklin	T	Wong	