Load the COMPANY database tables for the user CSUPERSON/EUCLID. Write and test a PL/SQL function [AssignEmployee] to act on the database according to the following specifications

1. The function takes three arguments: SocialSecurityNumber, ProjectNumber, HoursWeek
2. If the employee already works in the given ProjectNumber, the Hours value held in the database should be changed to the new value provided in the HoursWeek parameter.
3. If the employee does not work in the project then insert (if possible!) the employee in that project for the requested number of hours.

Operational and Business Rules to be Enforced

1. The SocialSecurityNumber and ProjectNumber must be valid.
2. The sum of hours worked on all of his/her projects (including the new assignment) should not exceed a total of 40 for any employee.
3. (From now on) New assignments must be controlled according to the policy
   a. The SALARY of employees working on five or more projects should be increased by $1000 for each new assignment.
   b. However, employees who are MARRIED and have CHILDREN cannot be placed on more than five projects.

The function AssignEmployee returns the following values

0  Operation successfully executed
1  Violation to rule 1
2  Violation to rule 2
3  Violation to rule 3b

What to turn in?
1. Print the (nicely documented, organized) listing of your code.
2. Begin the test with a fresh image of the database (similar to the data in the book).
3. Insert yourself in the database (use SSN 111222333). Introduce some family data (spouse and children).
4. Test the program with the following sequence of values (corresponding to SSN, ProjectNumber and Hours respectively). Annotate your observations, indicating what happens in each case (total hours, total projects, salary). Provide relevant screen-shots of each test.

<table>
<thead>
<tr>
<th>Soc. Sec.No</th>
<th>Proj. No.</th>
<th>NewHours</th>
<th>Accepted [yes/no?]</th>
<th>Total Hours</th>
<th>Total Projects</th>
<th>Salary</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>111222333</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>111222333</td>
<td>2</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>111222333</td>
<td>3</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>111222333</td>
<td>10</td>
<td>10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>111222333</td>
<td>20</td>
<td>20</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>111222333</td>
<td>39</td>
<td>10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>777888999</td>
<td>1</td>
<td>40</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>123456789</td>
<td>5</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>123456789</td>
<td>10</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Note: Test each aspect of the program. If needed add new data to make the test possible.