Databases - Lab Sheet 3

Gordon Royle

School of Mathematics & Statistics
University of Western Australia
Student ids

Getting the ids is a *projection* of the table $S$ onto the column $sid$.

$$\rho_{sid}(S)$$
Names of the male students

This needs a selection to extract the rows corresponding to the male students, and then a projection onto the name column, but all the information comes from the table $S$.

$$\rho_{\text{name}}(\sigma_{\text{gender}='M'}(S))$$
This expressions needs all three tables to get the student name, the unit name and the grades.
They can be joined using the natural join (⋈) but then the rows corresponding to Databases must be extracted.
The column called name needs to be qualified because S.name is the student’s name, while U.name is the unit name.

\[ \pi_{S\text{.name},G\text{.grade}}(\sigma_{U\text{.name}=’Databases’}(S \bowtie G \bowtie U)) \]
Translating SQL to RA

This is easy as it just uses a join, a selection and a projection. As $S$ and $G$ have just one column name in common - that is, sid, the natural join will be fine.

$$
\pi_{\text{sid}}(\sigma_{\text{grade}<50}(S \bowtie G))
$$

The query gets the IDs of anyone who has failed a unit (any unit).
The expression involves those rows of the natural join (i.e. \( \Join \)) relating to the unit CITS1402,

```sql
SELECT *
FROM S NATURAL JOIN G
WHERE G.uid = 'CITS1402';
```

As there is no projection onto a particular set of columns at the end, all the columns are selected and so `SELECT *` is the easiest way to do that (or give the names of all the columns).
One aggregate function applied to every row of one table:

```
SELECT AVG(Population) 
FROM Country;
```

```
+-----------------+  
| AVG(Population) |  
+-----------------+  
| 25434098.1172   |  
+-----------------+  
1 row in set (0.40 sec)
```
Total population

One aggregate function applied to every row of one table, and then renamed

```sql
SELECT SUM(Population)
AS worldPopulation
FROM Country;
```

```
+-----------------+
| worldPopulation |
+-----------------+
| 6078749450 |
+-----------------+
```
We need to join \texttt{Country} and \texttt{CountryLanguage} and then pull out the English-speaking countries, before applying the aggregate operators.

\begin{verbatim}
SELECT MIN(Population),
       MAX(Population),
       AVG(Population)
FROM CountryLanguage L
JOIN Country C
  ON L.CountryCode = C.Code
WHERE L.Language = 'English'
  AND L.IsOfficial = 'T';
\end{verbatim}

+-----------------+-----------------+-----------------+
| MIN(Population) | MAX(Population) | AVG(Population) |
+-----------------+-----------------+-----------------+
|     0           | 278357000       | 10435427.2727   |
+-----------------+-----------------+-----------------+
Life expectancy

This is a straightforward selection of rows from `Country` where the `Continent` is Asia, and then finding the minimum value of one of the columns.

```
SELECT MIN(LifeExpectancy)
FROM   Country
WHERE  Continent = 'ASIA';
```

```
+---------------------+
| MIN(LifeExpectancy) |
+---------------------+
| 45.9                |
+---------------------+
```
How many aggregate functions?

There are 17 different function names, (but 14 different functions)

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AVG()</td>
<td>Return the average value of the argument</td>
</tr>
<tr>
<td>BIT_AND()</td>
<td>Return bitwise and</td>
</tr>
<tr>
<td>BIT_OR()</td>
<td>Return bitwise or</td>
</tr>
<tr>
<td>BIT_XOR()</td>
<td>Return bitwise xor</td>
</tr>
<tr>
<td>COUNT(DISTINCT)</td>
<td>Return the count of a number of different values</td>
</tr>
<tr>
<td>COUNT()</td>
<td>Return a count of the number of rows returned</td>
</tr>
<tr>
<td>GROUP_CONCAT()</td>
<td>Return a concatenated string</td>
</tr>
<tr>
<td>MAX()</td>
<td>Return the maximum value</td>
</tr>
<tr>
<td>MIN()</td>
<td>Return the minimum value</td>
</tr>
<tr>
<td>STD()</td>
<td>Return the population standard deviation</td>
</tr>
<tr>
<td>STDDEV_POP()</td>
<td>Return the population standard deviation</td>
</tr>
<tr>
<td>STDDEV_SAMP()</td>
<td>Return the sample standard deviation</td>
</tr>
<tr>
<td>STDDEV()</td>
<td>Return the population standard deviation</td>
</tr>
<tr>
<td>SUM()</td>
<td>Return the sum</td>
</tr>
<tr>
<td>VAR_POP()</td>
<td>Return the population standard variance</td>
</tr>
<tr>
<td>VAR_SAMP()</td>
<td>Return the sample variance</td>
</tr>
<tr>
<td>VARIANCE()</td>
<td>Return the population standard variance</td>
</tr>
</tbody>
</table>