Databases

ASSESSMENT 2 : Web application with supporting documentation

Marker: Edward Longford

|  |  |
| --- | --- |
| Contribution towards overall module mark | 70% |
| Date set | 5/10/2023 |
| Marked work returned by | 9/02/2024 |
| DEADLINE DATE | 19/01/2024 |

Assessment 2: Web application with supporting documentation

Word Count: 2,000 (+/-10%)

**The Brief**

The task is to calculate the number of seats for each of the electoral systems listed below. This task uses a mixture of data science/data analysis skills - data preparation (by casting the data into a standard easy to read format), data manipulation (by calculating electoral results), and result analysis (by comparing the results).

The formulas for the electoral system is available in the supporting document within the assignment 2 folder on Minerva.

**Deliverables**

* Database in SQL containing vote data - 15%
* Code to calculate seat allocations - 40%
* Website - 10%
* Supporting documentation - 35%

A fully implemented database in SQL

For this you will need to provide:

* Database creation script for all tables with insert scripts
* Any stored procedures you have created

These scripts should be in a single **.sql or .txt file** for running and can be provided in **1 file** (creation script and insertion of data script).

The code should contain comments throughout the script explaining what each section of code does.

The database should be set up to store the vote information for each UK constituencies in a 3rd Normal Form database using the data provided in the HoC-GE2019-results-by-candidate-csv.csv provided.

Code to calculate Seat allocations

With a language of your choice (python, c++, ruby, R, etc…) you will need to extract data from the database you have created (either via stored procedures or SQL embedded in your chosen language) and manipulate it to produce seat results by party using a series of electoral systems.

You can use **ANY** combination of SQL and the code of your choice to calculate the results. For example you can use aggregate functions in SQL to return partially prepared data to be manipulated in your other code of choice rather than just returning all the data and doing all the manipulation and calculations in the code of your choice.

After all the calculations **ALL RESULTS** should be stored on a single table within your SQL database with the following information: Year, System, Party, Number of seats, Percentage of Seats, Percentage of Popular Votes, Difference Between Percentage Of Popular Votes And Percent Of seats, Party

With Most Seats, Different From Actual Winner.

You will be calculating the number of parliamentary seats allocations for the entire United Kingdom for the 2019 General Elections. Details of these electoral systems can be found in the supporting documentation.

1. First past the post by Constituency (this is the same as the current UK method of election)
2. General Election seats based on Simple Proportional Representation (All Seats)
3. General Election seats based on Simple Proportional Representation (All Seats) with threshold of 5%
4. General Election seats based on Proportional Representation (By County)
5. General Election seats based on Proportional Representation (By Region)
6. General Election seats based on Proportional Representation (By Country)
7. General Election seats allocations based on Largest Remainder (By County)
8. General Election seats allocations based on Largest Remainder (By Region)
9. General Election seats allocations based on Largest Remainder (By Country)
10. General Election seats allocations based on D’Hondt (By County)
11. General Election seats allocations based on D’Hondt (By Region)
12. General Election seats allocations based on D’Hondt (By Country)
13. A system of your own design (with explanation of the system). This can be a hybrid system combining multiple systems above or modifications to existing system.

Website

The results for each election should be published to a website, using a mixture of html, css, and javascript (where appropriate). Each result should be displayed separately in a table. These results should be ordered by year, and then by the electoral system used.

Report

In the report you will need to outline

* How you implemented your database system
* How you implemented your program to calculate results
* Snippets of SQL and the program you chose showing how various tasks are performed
* Reflection on the process that you used when creating both the SQL and the program to calculate the results and what changes you made during the process and after the process
* Evidence of testing
* A table output of results for each electoral system with the following structure:
  + System,
  + Party,
  + Seat Allocation,
  + Percentage of Seats,
  + Percentage of Popular Votes,
  + Difference Between Percentage Of Popular Votes And Percent Of Seats, Winning Party,
  + Different From Actual Winner.
* A short analysis of the different electoral systems from a mathematical perspective

**Submission**

Please follow the submission instructions below. Work that is submitted incorrectly may not be accepted or could incur a points penalty.

Before submitting have you…

* Checked that any digital work is functioning as expected?
* Spell-checked and grammar-checked any written work that accompanies your digital work? Please make an appointment with the [Writing and Learning Centre](https://www.bathspa.ac.uk/library/writing-and-learning-centre/) or speak to your tutor if you are experiencing challenges in this area.
* Formatted your written work to the specification below?
* Referenced all sources of information accurately? Please refer to [www.citethemrightonline.com](http://www.citethemrightonline.com) (Harvard) for guidance.

Your work must be submitted via Turnitin. Please adhere to the following method:

* Log into your university OneDrive account.
* Create a folder for your project. Call it something meaningful.
* Upload your work to your new folder.
* Right click your folder, select ‘Get Shareable Link’, turn on ‘Link Sharing’ and then copy the URL provided.
* Paste the URL into a Word document (please do not include your name for the purposes of anonymous marking).
* Attach any required written work (e.g. development documents) to your Word document.
* Log into Minerva, go to the Assessment folder and submit your Word document via the appropriate Turnitin link.

Before submitting have you…

* Spell-checked and grammar-checked your work? Please make an appointment with the [Writing and Learning Centre](https://www.bathspa.ac.uk/library/writing-and-learning-centre/) or speak to your tutor if you are experiencing challenges in this area.
* Formatted your written work to the specification below?
* Referenced all sources of information accurately? Please refer to [www.citethemrightonline.com](http://www.citethemrightonline.com) (Harvard) for guidance.

Your work must be submitted via Turnitin. Please adhere to the following method:

* Save your work as a Word document (please do not include your name for the purposes of anonymous marking).
* Log into Minerva, go to the Assessment folder and submit your Word document via the appropriate Turnitin Link.

**Format**

All written work must conform to university styling and submission guidelines. They must:

* Contain appropriate in-text citation that supplies an accurate list of references.
* Be accurate in referencing. See [Bath Spa guidelines](https://www.bathspa.ac.uk/library/researching-and-referencing/) and the Harvard system described at [www.citethemrightonline.com](http://www.citethemrightonline.com).
* Be accurate in spelling and paragraphing.

Word counts

Written assessments must be within a tolerance of +/-10% of the word count stated on briefs. For example, an assessment with a specified word count of 2,000 words must be in the range of 1,800 to 2,200 words.

Submissions that do not fall within word count limits of +/-10% may be penalised for not meeting the constraints of the assessment. Furthermore, written pieces that fall below the minimum word count (-10%) may lack the detail required by the assessment.

**Marking Criteria**

Assessment 2: Web application with supporting documentation will be marked against the following criteria:

1. SQL and database structure
2. Data manipulation
3. Website
4. Supporting documentation

|  |  |  |  |
| --- | --- | --- | --- |
| Criteria | Weighting | Mark Range Description | Mark Range |
| SQL and database structure | 15% | Very limited database implementation, most relations missing or inappropriate. Database is not in 1st Normal Form. No requirements have been met. No keys or appropriate data types present. | 0 - 19  (Low Fail) |
| Poor database implementation, most relations missing or inappropriate. Database is in 1st Normal Form. Some requirements have been met. No keys or appropriate data types present. | 20 - 39  (Fail) |
| Basic database implementation, some relations missing. Database is in 1st Normal Form. Some requirements have been met. A few keys or data types present and appropriate. | 40 - 49  (Third) |
| Fair database implementation. Database is in 2nd Normal Form. Most requirements have been met. Most keys or data types present and appropriate. | 50 - 59  (2:2) |
| Good database implementation. Database is in 2nd Normal Form. All requirements have been met. All keys or data types present and appropriate. | 60 - 69  (2:1) |
| Very good database implementation. Database is in 3rd or 2.5 Normal Form. All requirements have been met. All keys or data types present and appropriate. | 70 - 79  (First) |
| Excellent database implementation. Database is in 3rd or 2.5 Normal Form. All requirements have been met. All keys or data types present and appropriate. | 80 - 89  (High First) |
| Beyond expectations for this level of study. | 90 - 100  (Outstanding) |
| Code to calculate Seat allocations | 40% | Very limited attempt to extract data from the database. Limited attempt at calculation of Seat allocations. | 0 - 19  (Low Fail) |
| Poor attempt to extract data from the database. Poor attempt at calculation of Seat allocations. | 20 - 39  (Fail) |
| Basic attempt to extract data from the database. Basic attempt at calculation of Seat allocations. | 40 - 49  (Third) |
| Fair attempt to extract data from the database. Fair attempt at calculation of Seat allocations. | 50 - 59  (2:2) |
| Good attempt to extract data from the database. Good attempt at calculation of Seat allocations. | 60 - 69  (2:1) |
| Very good attempt to extract data from the database. Very good attempt at calculation of Seat allocations. | 70 - 79  (First) |
| Excellent attempt to extract data from the database. Excellent attempt at calculation of Seat allocations. | 80 - 89  (High First) |
| Beyond expectations for this level of study. | 90 - 100  (Outstanding) |
| Website | 10% | Very limited functionality of front end website. HTML is hard coded | 0 - 19  (Low Fail) |
| Poor functionality of front end website. 1 electoral system. HTML is hard coded | 20 - 39  (Fail) |
| Basic functionality of front end website. Some electoral systems presented. HTML is hard coded | 40 - 49  (Third) |
| Fair functionality of front end website.  Multiple electoral systems. HTML is not hard coded. | 50 - 59  (2:2) |
| Good functionality of front end website. Most electoral system. HTML is not hard coded. Included CSS files that improve usability and visuals. | 60 - 69  (2:1) |
| Very good functionality of front end website. All electoral system. HTML is not hard coded. Included CSS files that improve usability and visuals. | 70 - 79  (First) |
| Excellent functionality of front end website. All electoral system with changes made for states that don’t use a winner takes all Electors system. HTML is not hard coded. Included CSS files that improve usability and visuals. | 80 - 89  (High First) |
| Beyond expectations for this level of study. | 90 - 100  (Outstanding) |
| Supporting documentation | 35% | Very limited documentation or unrelated to the implemented system. issues. Very limited SQL/code resented. Very limited user guide. Very limited reflection. Very limited evidence of testing. | 0 - 19  (Low Fail) |
| Poor documentation or unrelated to the implemented system. Poor discussion of security issues. Poor SQL/code presented. Poor user guide. Poor reflection. Poor evidence of testing. | 20 - 39  (Fail) |
| Basic documentation on the implemented system. Basic discussion of security issues. Basic SQL presented. Basic user guide. Basic reflection. Basic evidence of testing. | 40 - 49  (Third) |
| Fair documentation on the implemented system. Fair discussion of security issues. Fair SQL presented. Fair user guide. Fair reflection. Fair evidence of testing. | 50 - 59  (2:2) |
| Good documentation on the implemented system. Good discussion of security issues. Good SQL presented. Good user guide. Good reflection. Good evidence of testing. | 60 - 69  (2:1) |
| Very good documentation on the implemented system. Very good discussion of security issues. Very good SQL presented. Very good user guide. Very good reflection. Very good evidence of testing. | 70 - 79  (First) |
| Excellent documentation on the implemented system. Excellent discussion of security issues. Excellent SQL presented. Excellent user guide. Excellent reflection throughout. Excellent evidence of testing. | 80 - 89  (High First) |
| Beyond expectations for this level of study. | 90 - 100  (Outstanding) |

**Intended Learning Outcomes (ILOs)**

|  |  |
| --- | --- |
| ILO | Assessed |
| An ability to implement a database design that addresses a defined requirement specification. | ✓ |
| Knowledge and critical understanding of data modelling approaches, and the formal methods used to express relationships between entities. | ✓ |
| An ability to select and implement tools and procedures for querying and manipulating databases, and serving content to a user. | ✓ |
| Critical understanding of database management issues, and an ability to mitigate them. | ✓ |

Mark penalties may be applied to late submissions without prior approval of an extension. Please ensure that you prepare and submit your work in good time to allow for any issues that may arise.