**Solent University**

**Coursework Assessment Brief**

**Assessment Details**

|  |  |
| --- | --- |
| Unit Title: | Introduction to Databases |
| Unit Code: | SWD406 |
| Unit Leader: | Kenton Wheeler |
| Level: | 4 (FHEQ) |
| Assessment Title: | Database development and evaluation |
| Assessment Number: | AE1 |
| Assessment Type: | Report containing code |
| Guidelines on Word Count: | Max. 1500 words for the evaluation report. Any SQL, Python code, diagrams or explanatory text for parts 1, 2 and 3 are not included in this word count. |
| Consequence of not meeting  time/word count limit: | There is no penalty for submitting above the word count guideline but students should be aware that there is a risk they may not maximise their potential mark if they submit substantially less. |
| Individual/Group: | Individual |
| Assessment Weighting: | 100% |
| Issue Date: | 24th February 2020 |
| Hand In Date: | Wed 6th May 2020 |
| Planned Feedback Date: | Wed 3rd June 2020 |
| Mode of Submission: | Online via SOL/Turnitin. |
| Format of Submission | Report in MS Word or PDF Format.  Code in zip file |

**Assessment Task**

**1. Introduction**

This assignment requires you to undertake practical database application development work

to meet specified requirements and write an evaluation report that discusses the outcomes of the work completed and the processes used.

This assignment is designed to assess the learning outcomes of the unit which are as follows:

1. Explain key issues in the development of relational databases and their role in modern IT systems.

2. Discuss the use of SQL functionality to create information from data.

3. Apply conceptual modelling techniques to the design and implementation of a simple database.

4. Apply enterprise-level database software tools in the development, implementation and testing of SQL-based database solutions.

5. Demonstrate an understanding of and the ability to use SQL for writing queries, creating and populating databases.

6. Demonstrate the ability to write simple Python code to interact with a relational database.

**2. The Practical Development Work**

The practical development work is based on an online electronics shopping company where you work as a Database Analyst/Developer. The entity-relationship diagram and SQL script for creating and populating the database are provided on SOL.

**Part 1 - Retrieving Data using SQL**

You have been asked to write the following SQL queries for management information purposes. **All students should complete questions a, b and c below and to achieve a higher grade, also complete question d.**

1. The company want to do a marketing campaign to new shoppers and those aged under 30. Retrieve the first name, surname, email address, date joined, and the age in years of all shoppers who joined on or after 1st Jan 2020 or those aged 29 or less on the 1st Jan 2020. Print date columns in the format DD-MM-YYYY. Order results by age (highest first) and then surname (A-Z).

Refer to the SQLite Built-in Functions reference on SOL for how to calculate the age and format the dates.

1. The website requires a customer account history page which will accept the shopper id as a parameter. Write a query to retrieve the first name and surname for a specific shopper along with details of all the orders they’ve placed, displaying the order no, order date, product description, seller name, quantity ordered, price (right-justified with two decimal places and prefixed by a £ sign) and ordered product status. Print date columns in the format DD-MM-YYYY. Sort the results by order date showing the most recent order first. Test your query by prompting for the user to input a shopper account ref and produce results for shopper ids 10000 and 10019.
2. The business relationship manager has asked you to write a summary report on the sellers and products that they have had sold since 1st June 2019. Display the seller account ref, seller name, product code, product description, number of orders, total quantity sold and total value of all sales (right-justified with two decimal places and prefixed by a £ sign) for each product they sell. You should also include products that a seller sells but has had no orders for and show any NULL values as 0. Sort results by seller name and then product description.
3. The head of sales wants a report showing the products that have an average quantity sold that is less than the average quantity sold for the category that the product is in. Any products that haven’t sold at all should also be displayed with an average quantity of 0. Display the category description, product code, product description and average quantity sold and show results in category description, then product description order.

**For each query, include the SQL code you have written (in a format that can be copied and pasted i.e. not as a screenshot) along with a brief explanation of the SQL. Supply screenshots of the query results and user input (if any) and provide proof that the results are correct by doing thorough testing. Remember to display meaningful and user-friendly column headings on all queries.**

**Part 2 – Database Design, Implementation and Integrity**

The online electronics shopping database needs to be extended to store the data required to implement shopper reviews about sellers and products and questions and answers about products.

Seller reviews are just about the seller not about the product they sold and product reviews are about the product and not the seller that sold it. Each review must be star-rated as \* (Poor), \*\* (Fair), \*\*\* (Good), \*\*\*\* (Very Good) and \*\*\*\*\* (Excellent) and hold a brief textual comment from the shopper. The date and time that the feedback was submitted should also be stored.

Questions about products are anonymously asked by shoppers and can be answered by other shoppers or by sellers. A question can be answered many times. The date and time that the question is asked and any answers posted also need to be stored.

**All students should complete questions a, b and c below and to achieve a higher grade, also complete question d.**

1. Produce a table design to support this additional functionality explaining the process you used to arrive at your design, how you ensured the database integrity would be maintained and any design assumptions that you have made. **Your design should consist of at least two new tables and you must link to at least one of the existing tables.**
2. Modify the provided Entity Relationship diagram to show your new tables, their primary and foreign keys and how they relate to each other and to the existing tables.
3. Implement your design by creating the new tables, insert enough rows into your new tables to facilitate testing and prove that your integrity constraints work correctly through testing. Include the SQL that you used to create, populate and test the new tables in your submission.
4. Create a view that joins your newly created tables together with existing tables and provide at least two SQL queries that select from this view.

**Part 3 – Programming for Databases**

Develop Python code to implement some basic text-based functionality to allow the user to interact with the online electronics shopping database as outlined below. **All students should complete questions a, b and c below and to achieve a higher grade, also complete question d.**

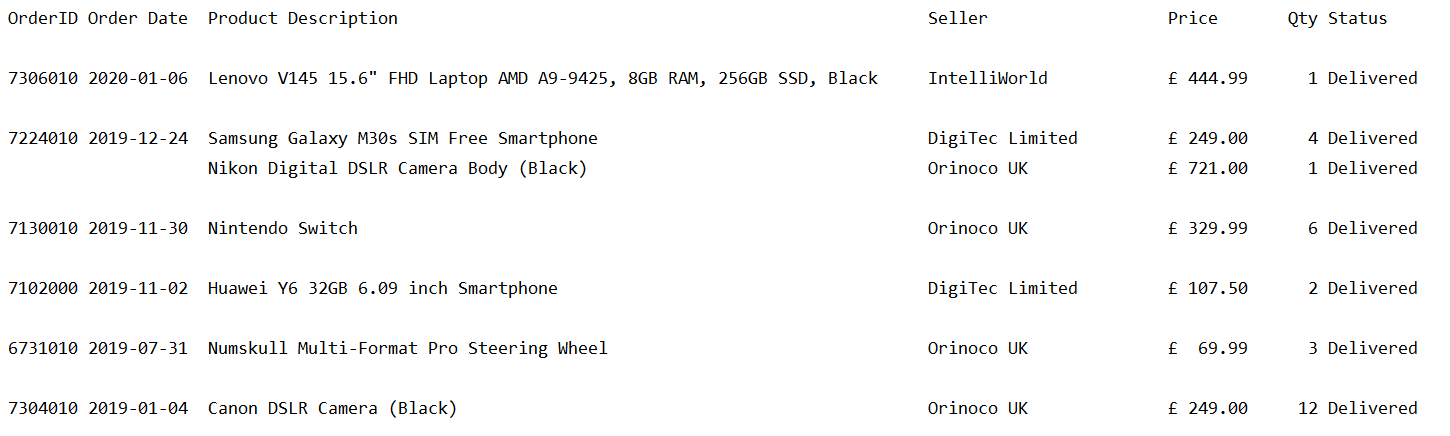
1. Firstly, prompt for the entry of a shopper\_id which will be used to test all the menu options. If the shopper\_id entered is not found on the database, print an error message and exit the program otherwise print the main menu below.
2. Provide a text-based menu as follows:

ORINOCO – SHOPPER MAIN MENU

1. Display your order history
2. Add an item to your basket
3. View your basket
4. Checkout
5. Exit
6. Implement menu options 1, 2, 3 and 5 as follows:

Option 1 – Display your order history

1. For each order that the customer has placed, display the order id and order date together with the product description, seller name, price, quantity ordered and status of each product on that order. You can use your query from Part 1b as a basis for the SQL query in your Python code.
2. Sort orders by order date (most recent first)
3. If no orders are found for the shopper\_id that you are testing with, print the message “No orders placed by this customer”
4. Display the data in the format shown below (which is for shopper\_id 10010)

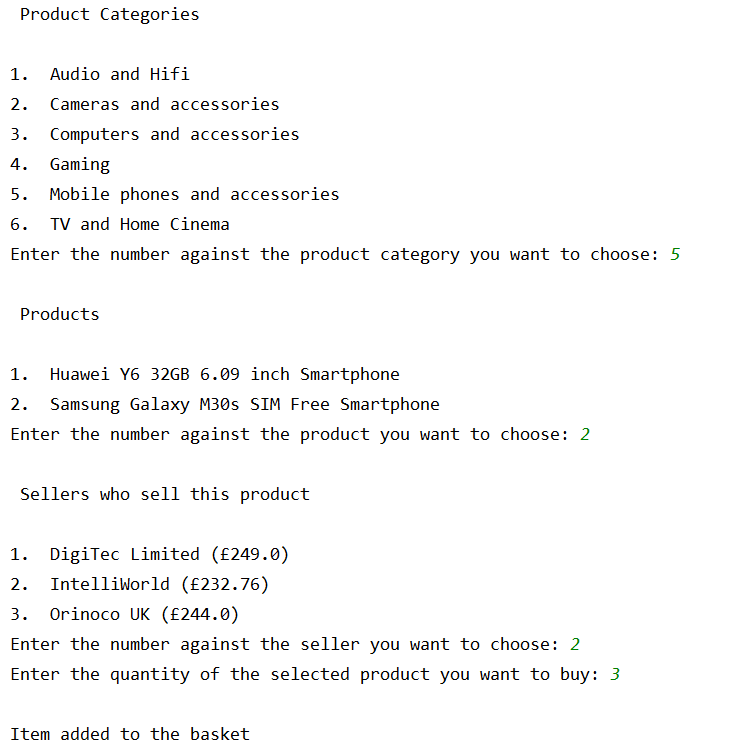


1. Return to the main menu

Option 2 – Add an item to your basket

1. Display a numbered list of product categories
2. Prompt the user to enter the number of the product category they want to choose from and store the category\_id for the selected category
3. Display a numbered list of the available products in the category selected
4. Prompt the user to enter the number of the product they want to purchase and store the product\_id for the selected product
5. Display a numbered list of sellers who sell the product they have selected and the price they are selling that product at
6. Prompt the user to enter the seller they wish to buy the product from and store the seller\_id for the selected seller
7. Prompt the user to enter the quantity of the selected product they want to order
8. Get the price of the selected product from the selected supplier
9. Get the next basket id by selecting from the sqlite\_sequence table
10. If the basket is empty, insert a new row into the shopper\_basket table using the next basket \_id selected in stage ix.
11. Insert a new row into the basket\_contents table for the product they want to purchase again using the next basket id selected in stage ix.
12. Commit the transaction
13. Print “Item added to your basket”
14. Return to the main menu

Below is an example of what should be displayed and what should be prompted for:



To simplify your code, the following function must be included at the top of your program to display a numbered list of options and return the id of the selected option. You can amend this function accordingly if you wish but a function must be included.

**def** \_display\_options(all\_options,title,type):  
 option\_num = 1  
 option\_list = []  
 print(**"\n"**,title,**"\n"**)  
 **for** option **in** all\_options:  
 code = option[0]  
 desc = option[1]  
 print(**"{0}.\t{1}"**.format(option\_num, desc))  
 option\_num = option\_num + 1  
 option\_list.append(code)  
 selected\_option = 0  
 **while** selected\_option > len(option\_list) or selected\_option == 0:  
 prompt = **"Enter the number against the "**+type+**" you want to choose: "** selected\_option = int(input(prompt))  
 **return** option\_list[selected\_option - 1]

This function should be called in steps i, iii and v above using a command of the following format:

id\_of\_selected\_option = \_display\_options(query\_rows,**title**,**type**)

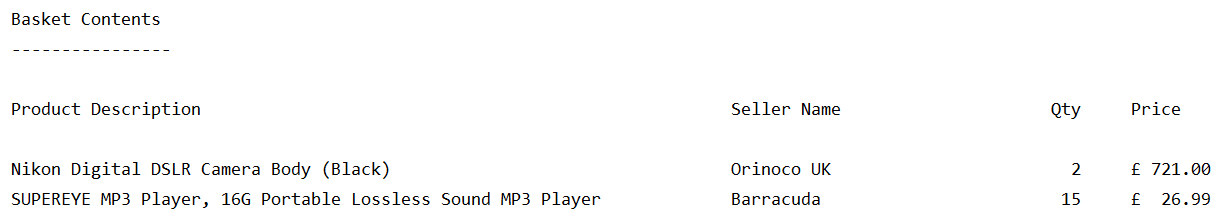
query\_rows must consist of two values - id and description i.e. the category\_id and category\_description

title is some text to put above the list of options to act as a title

type is used to customise the prompt to make it appropriate for what you want the user to select

Option 3 – Display your basket

1. If the basket is empty, display an error message otherwise display all rows from the basket\_contents table for the current basket as per the example below:

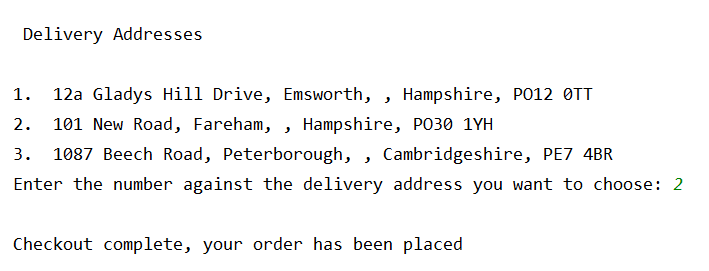


1. Return to the main menu

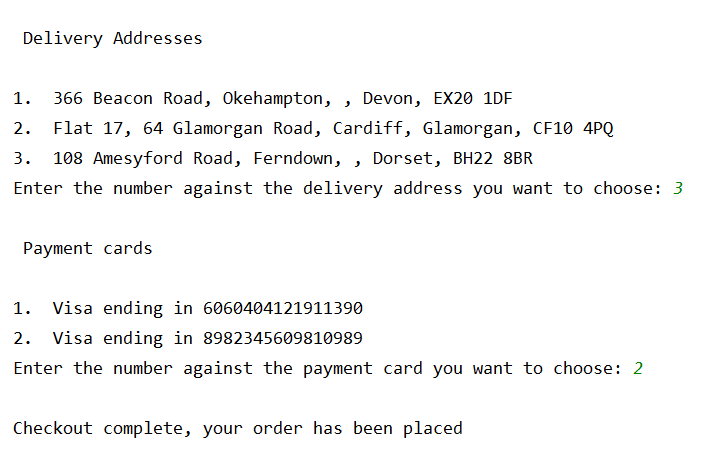
Option 5 – Exit

1. Exit the program
2. Option 4 – Checkout your basket
3. If the basket is empty, display an error message otherwise display the current basket and total amount to pay (which you will need to calculate)
4. If the shopper has more than one delivery address stored for them, display a numbered list of the delivery addresses (most recently used first) and prompt the shopper to choose the address they want to deliver the items to. If the shopper has one delivery address, display and use this address for the checkout. If they have no delivery addresses, prompt the shopper to enter a new delivery address, use this address for the checkout process and insert a new row into the shopper\_delivery\_addresses table.
5. If the shopper has more than one payment card stored for them, display a numbered list of the payment cards (most recently used first) and prompt the shopper to choose the card they want to pay with. If the shopper only has one payment card, display the card number and use it for the checkout. If they have no payment card stored for them, prompt the shopper to enter new card details, use this card for the checkout process and insert a new row into the shopper\_payment\_cards table.
6. Insert a new row into the shopper\_order table for the basket with a status of ‘Placed’
7. Insert a new row into the ordered\_product table for each item in the basket with a status of ‘Placed’
8. Delete the rows from the shopper\_basket and basket\_contents tables for this basket
9. Return to the main menu

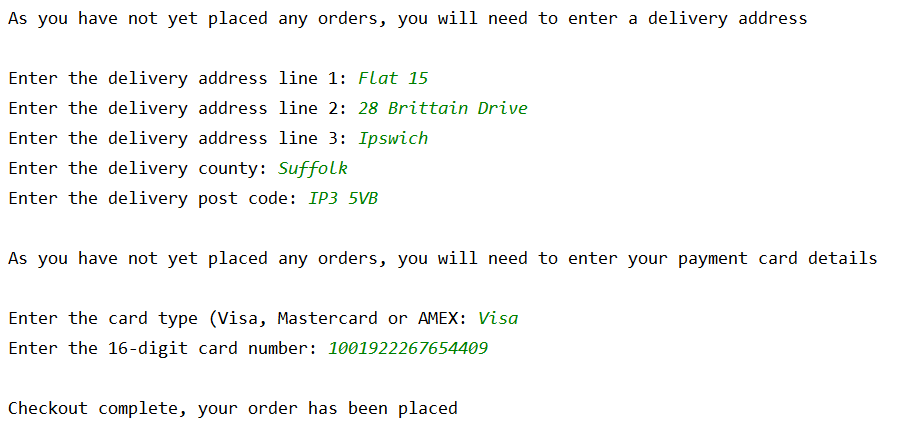
Below is an example of what should be displayed for shopper\_id 10000 who has more than one delivery address and only one payment card stored on the system:



Below is an example of what should be displayed for shopper\_id 10005 with more than one delivery address and more than one payment card stored on the system:



Below is an example of what should be displayed for shopper\_id 10023 with no delivery addresses or payment cards stored on the system:



You should inserts comments throughout your code to make it easier for someone else to understand.

**With your submission, you must include all your Python code, screenshots of the output and any user interaction together with evidence that the requirements outlined under each menu option have been met and thoroughly tested by including screenshots of data successfully inserted, updated and deleted from the database and errors/exceptions being correctly handled.**

**3. The Evaluation Report (max 1500 words)**

**All students must include an evaluation report in their submission which should cover the following:**

a) An evaluation of the quality of the work you have produced with reference to both the specified requirements and the assessment criteria. You should identify:

* any deliverables that are not complete or fully tested, and
* elements of the practical work that you feel contribute to higher grade achievement

b) Your approach to design and testing of the SQL and Python elements of the assessment and what more you would have done if you’d had more time.

c) What you had changed or improved from any pre-submitted versions of parts 1, 2 or 3 based on feedback from your lecturer.

d) Bibliography and references

**4. Submitting Your Work**

The online submission (which should be well structured and clearly written) must be in a single document in MS Word or PDF format and should include:

**1.** A title page

**2.** Contents page and page numbers.

**3.** Copies of all your **final (versions** of the deliverables for parts 1, 2 and 3.

**4.** Your evaluation report (see section 3 above)

**Assessment criteria**

The summary grid below is the basis for grading achievement. Higher levels of achievement are described towards the right-hand side of the grid. Each **level** subsumes the previous level. An indication of the % contribution of each part and the report is indicated. Normally, an attempt must be made for each block and the Evaluation Report.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S, F3-F1** | **D3-D1** | **C3->C1** | **B3->B1** | **A4->A1** |
| Part 1 - Retrieving Data using SQL – Learning Outcomes K1, C1, P1 – 40% | | | | |
| Does not reach required threshold. | At least two queries were submitted that produce the correct results.  The queries meet the basic requirements but no attempt has been made to improve the presentation.  No evidence of testing has been provided to ensure the results for each query are correct.  No explanation has been provided for each query. | Has submitted queries a, b and c and they all produce the correct results.  The queries meet the basic requirements and some attempt has been made to improve the presentation.  Evidence of testing has been provided for some of the queries.  Some explanation has been provided for each query demonstrating a basic understanding of the concepts. | Has submitted queries a, b and c and they all produce the correct results  The queries meet the majority of the requirements and the output is well presented.  Some evidence of testing has been provided to ensure the results for each query are correct.  Some explanation has been provided for each query demonstrating a good understanding of the concepts. | Has submitted queries a, b, c and they all produce the correct results  The queries meet or exceed all the requirements and the output is professionally presented.  Evidence of through testing has been provided to ensure the results for each query are correct.  A detailed explanation has been provided for each query demonstrating a full understanding of the concepts. |
| Part 2 – Database Design Implementation and Integrity - Learning Outcomes K1, C1, P1 – 25% | | | | |
| Does not reach required threshold. | The design has some flaws but meets the requirements or does not have a minimum of 3 new tables.  The rationale behind the design has not been provided and any design assumptions have not been outlined.  The newly-created tables have not been correctly implemented or the primary and foreign keys are missing or incorrect.  Each new table has not been populated with sufficient test data.  No evidence has been provided that the new constraints have been tested to ensure they work correctly.  No amended ERD has been provided or the one provided is incorrect.  No view has been created or used. | The design has some flaws but meets the requirements and has a minimum of 3 new tables.  A basic rationale behind the design has been provided but no design assumptions have been outlined.  The newly-created tables have been correctly implemented with the right primary and foreign keys. No other constraints have been used  Each new table has been populated with basic test data.  Evidence has been provided that some of the new constraints have been tested to ensure they work correctly.  A amended ERD diagram has been provided but it is incorrect.  No view has been created or used. | The design is correct, meets the requirements and has a minimum of 3 new tables.  The rationale behind the design has been explained and any design assumptions have been outlined.  The newly-created tables have been correctly implemented with the right primary and foreign keys. Some other constraints have been used.  Each new table has been populated with a reasonable amount of test data.  Evidence has been provided that some of the new constraints have been tested to ensure they work correctly.  A correctly amended ERD diagram has been provided.  If a view was created, it was not fully tested with at least two correct queries. | The design is correct, meets or exceeds the requirements and has a minimum of 3 new tables.  The rationale behind the design has been fully explained and any design assumptions have been outlined.  The newly-created tables have been correctly implemented with the right primary and foreign keys. Other constraints have been widely used and the constraints have been named.  Each new table has been populated with a good amount of test data.  Full evidence has been provided that all new constraints have been tested to ensure they work correctly.  A correctly amended ERD diagram has been provided.  A view has been created based on the new and existing tables and at least two queries correctly written using the view. |
| Part 3 - Programming Learning Outcomes K1 and T1 – 25% | | | | |
| Does not reach required threshold. | Menu option 1 has been implemented, works correctly and mostly meets the requirements.  The user interface is basic or does not function correctly.  The code retrieves data from the database  There are some major issues with how the code has been written.  There is little or no evidence that the code has been tested  Comments are not used in the code  There is no evidence of error handling. | Menu option 1 has been implemented and is working correctly and menu options 2 and 3 are partially implemented or do not function correctly  The user interface is basic but functional.  The code retrieves data from the database  There are some minor issues with how the code has been written.  There is some evidence that the code has been tested  Comments are sparsely used in the code  There is some evidence of error handling. | Menu options 1,2 and 3 have been fully implemented, work correctly and mostly meet the requirements  The user interface is good with clear prompts and helpful error messages.  The code retrieves data from the database and inserts rows in the database.  The code is correctly structured and well-written.  There is evidence that the code has been tested  Clear comments are used throughout the code.  There is some evidence of error handling. | Menu options 1-4 have been fully implemented, they all work correctly and fully meet the requirements.  The user interface is good with clear prompts and helpful error messages.  The code retrieves data from the database and inserts, updates and deletes rows in the database.  The code is correctly structured and well-written.  There is evidence that the code has been thoroughly tested  Extensive error handling has been used. |
| Evaluation Report = 10% | | | | |
| Does not reach required threshold. | A basic description of the approach you used to produce your deliverables and evaluation of the work you have produced.  You made no pre-submissions or you made no attempt to improve your work based on feedback from any pre-submissions that you did make. | Some reflection on the approach you used to produce your deliverables and the quality of the work you have produced.  Some attempt was made to improve your work based on feedback from pre-submissions. | An honest self-assessment of the approach you used to produce the deliverables and the quality of the work you have produced stating what you did well and not so well.  You addressed many of the issues identified in the feedback from pre-submissions. | An honest, reflective and insightful evaluation of the approach you used to produce the deliverables and the quality of the work you have produced stating what you did well and not so well and what more you would have done if you’d had more time.  You fully took on- board the feedback from your pre-submissions and addressed all the issues identified. |

**Learning Outcomes**

This assessment will enable students to demonstrate in full or in part the learning outcomes identified in the unit descriptors.

**Late Submissions**

Students are reminded that:

i. If this assessment is submitted late i.e. within 5 working days of the submission deadline, the mark will be capped at 40% if a pass mark is achieved;

ii. If this assessment is submitted later than 5 working days after the submission deadline, the work will be regarded as a non-submission and will be awarded a zero;

iii. If this assessment is being submitted as a referred piece of work (second or third attempt) then it must be submitted by the deadline date; any Refer assessment submitted late will be regarded as a non-submission and will be awarded a zero.

http://portal.solent.ac.uk/documents/academic-services/academic-handbook/section-

2/2o-assessment-policy-annex-1-assessment-regulations.pdf?t=1411116004479

**Extenuating Circumstances**

The University’s Extenuating Circumstances procedure is in place if there are genuine circumstances that may prevent a student submitting an assessment. If students are not 'fit to study’, they can either request an extension to the submission deadline of 5 working days or they can request to submit the assessment at the next opportunity (Defer). In both instances students must submit an EC application with relevant evidence. If accepted by the EC Panel there will be no academic penalty for late submission or non-submission dependent on what is requested. Students are reminded that EC covers only short term issues (20 working days) and that if they experience longer term matters that impact on learning then they must contact a Student Achievement Officer for advice.

A summary of guidance notes for students is given below:

http://portal.solent.ac.uk/documents/academic-services/academic-handbook/section-

4/4p-extenuating-circumstances-procedures-for-students.pdf?t=1472716668952

**Academic Misconduct**

Any submission must be students’ own work and, where facts or ideas have been used from other sources, these sources must be appropriately referenced. The University’s Academic

Handbook includes the definitions of all practices that will be deemed to constitute academic misconduct. Students should check this link before submitting their work.

Procedures relating to student academic misconduct are given below:

http://portal.solent.ac.uk/support/official-documents/information-forstudents/

complaints-conduct/student-academic-misconduct.aspx

**Ethics Policy**

The work being carried out by students must be in compliance with the Ethics Policy. Where there is an ethical issue, as specified within the Ethics Policy, then students will need an ethics release or an ethical approval prior to the start of the project. The Ethics Policy is contained within Section 2S of the Academic Handbook:

http://portal.solent.ac.uk/documents/academic-services/academic-handbook/section-

2/2s-university-ethics-policy.pdf

**Grade marking**

The University uses a letter grade scale for the marking of assessments. Unless students have been specifically informed otherwise their marked assignment will be awarded a letter grade. More detailed information on grade marking and the grade scale can be found on the portal and in the Student Handbook.

Policy: http://portal.solent.ac.uk/documents/academic-services/academichandbook/

section-2/2o-assessment-policy.pdf

**Guidance for online submission through Solent Online Learning (SOL)**

http://learn.solent.ac.uk/onlinesubmission