Test Documentation

# Unit Tests

NOTE: All tests are carried out on a clean database, unless stated otherwise

## Module Name: organise\_booking

### Inputs

|  |  |
| --- | --- |
| Variable | Description |
| booking\_name | Name of the current customer being processed |
| Pas\_in\_booking | Number of passengers in current booking being processed |
| empty\_seats\_per\_row | A dictionary showing row numbers and their associated number of empty seats. |
| empty\_seats | A list of tuples which show the layout of the airplane. Each tuple contains a row number, seat letter and a blank string for which a passenger name will be assigned. |
| cols | A string containing the columns names for each row |
| engine | The database engine which is created as part of the main function |

### Outputs

This function writes the booking name directly to the database.

### Unit Tests

*Test Description:* No direct unit tests can be applied to this function. The functionality however will be tested in the overall functional testing section of this document.

## Module Name: retrieve\_data

### Inputs

|  |  |
| --- | --- |
| Variable | Description |
| engine | The database engine which is created as part of the main function |
| rows | An integer value showing the number of rows in the airline |
| cols | A string containing the columns names for each row |

### Outputs

|  |  |
| --- | --- |
| Variable | Description |
| empty\_seats | A list of tuples which show the layout of the airplane. Each tuple contains a row number, seat letter and a blank string for which a passenger name will be assigned. |
| empty\_seats\_per\_row | A dictionary showing row numbers and their associated number of empty seats. |
| num\_pas\_refused | The current number of passengers refused a booking. |
| num\_pas\_split | The current number of passengers split from their booking group. |

### Unit Tests

*Test Description:* Unit test 1 is run on a cleaned airline\_seating.db database. It tests the first instance of this function being run when the database is empty and the first booking is about to be assigned.

*Unit test status:* **PASSED**

*Test Description:* Unit test 2 is run on the test\_airline\_seating.db database. This database has already received a number of bookings so the outputs will be different to unit test 1.

*Unit test status:* **PASSED**

## Module Name: test\_find\_row\_with\_n\_empty\_seats

### Inputs

|  |  |
| --- | --- |
| Variable | Description |
| Empty\_seats\_per\_row | A dictionary showing row numbers and their associated number of empty seats. |
| Number\_of\_pas | Number of passages to be booked |
| e | A constant number ranging between 1 and the width of the airplane |

### Outputs

|  |  |
| --- | --- |
| Variable | Description |
| Boolean | True or False |
| Key | The row number returned if a suitable row matching the number of passengers plus the constant 3 is found. If no value is found, zero is returned |

### Unit Tests

The number of passengers to be booked for all unit tests is three.

**Unit Test 1**

*Test Description:* UT1 looks through a completely empty airline (i.e. all rows have a dictionary value of 4) and attempts to find a row when the corresponding e constant value is 2.

*Expected Output:* (False, 0) – As no rows containing 5 (number of passengers (3) plus e (2) are available.

*Unit Test Status:* **PASSED**

**Unit Test 2**

*Test Description:* UT2 again looks through a completely empty airline (i.e. all rows have a dictionary value of 4) and attempts to find a row when the corresponding e constant value is 1.

*Expected Output:* (True, 1)

*Unit Test Status:* **PASSED**

**Unit Test 3**

*Test Description:* UT3 looks through a partially empty airline and attempts to find a row when the corresponding e constant value is 1.

*Expected Output:* (True, 4)

*Unit Test Status:* **PASSED**

## Module Name:

### Inputs

|  |  |
| --- | --- |
| Variable | Description |
|  |  |
|  |  |
|  |  |

### Outputs

|  |  |
| --- | --- |
| Variable | Description |
|  |  |
|  |  |

### Unit Tests