Database Assignment

### Purpose and End User of my database

| I want customers to be able to use this database so that they may make an order for a pizza, taking their name, the date, and the pizza size. |
| --- |

### Describe at least 3 implications that are relevant to your database and its use by the end user and why they are important

| Functionality - Functionality means if it is working as intended. I need my outcome to function properly or else it will not be useful to anyone. For example, if my dog did not function properly, it would most likely die. So it is important to have your product (database) function properly so that people will use it. My goal is to have people use my database.  Usability - Usability closely relates to functionality, but refers to how easy clients and people can use my outcome / product / database. My database needs to be usable because if it is not, people will not use it (my goal is for people to use it). For example, the inverted mouse option. What if it was always on and couldn't be turned off? You probably wouldn't use it (my goal is for people to use it). Therefore I need my database to be usable and easy to understand so people will actually use it.  Privacy - Privacy is security of user and outcome data. The database needs to not give away user data or the actual database data to strangers without permission. For example, in this context (database), I don't want anyone editing or looking at my pizza data. I want only the employees to look and edit the data, so I would make a login system. I also don't want my database to find too much information out about who is using it, for it is unnecessary and can endanger my clients if unwanted database users get this information. That is why my database needs to not gain too much information about who is using it and keep unwanted users out, otherwise it could be dangerous or prove not to be useful (my goal is for people to use it). |
| --- |

### Database Design- Your Entity Relationship Diagram.

| Pizza - shows information about the pizza  Pizza\_orders - shows what pizzas are in what order  Orders - shows who is getting the order  Pizza (1 to many) Pizza\_orders (many to 1) Orders  (slight change: I moved size from Pizza to Orders, because customers should be able to pick their size of the pizza and it should not be set) |
| --- |

### Database Testing Table: SQL Statements

| **Purpose** | **SQL Statement** | **Result Success?** |
| --- | --- | --- |
| join the pizza\_name to the pizza\_id (meat lover = 5, plain = 1 for example) | SELECT pizza.pizza\_id, pizza.name FROM pizza  JOIN pizza\_orders ON pizza.pizza\_id = pizza\_orders.pizza\_id | ✔ |
| shows the who ordered what order (kenan = 3, jack = 1) | SELECT orders.orders\_id, orders.orders\_fname FROM orders  JOIN pizza\_orders ON pizza\_orders.orders\_id = orders.orders\_id | ✔ |
| (my database corrupted so the data + table will be a little different, although relatively the same)  joins orders onto pizza\_orders where the order\_id's line up (order\_id = whos order is whos) | SELECT \* FROM pizza\_orders  JOIN orders ON pizza\_orders.order\_id = orders.order\_id; | ✔ |
| the opposite; joins pizza onto pizza\_orders where pizza\_ids line up. This query also shows everything from the tables because of the \*, so you can see everything in pizza that lines up with that specific id and the same for pizza\_orders. | SELECT \* FROM pizza\_orders  JOIN pizza ON pizza\_orders.pizza\_id = pizza.id; | ✔ |
| I really just wanted to see who was ordering what pizza, but this works as well | SELECT \* FROM orders  JOIN pizza\_orders ON orders.order\_id = pizza\_orders.order\_id; | ✔ |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

### Relevant Implications- Explain how your database addresses the relevant implications that you identified at the start.

| Functionality - My database is fully functional and meets my purpose and end user, working for the customer.  Usability - I think my database is relatively easy to use and understand, accessible to most.  Privacy - The only data the database takes in from the user is their pizza order and name, which is non threatening information. |
| --- |

### Showcase:

Give evidence of your database and the Python code that interfaces with it. Use screenshots or a short video. Explain how it improved, how it functions, how it was tested etc.

| [Video](https://www.youtube.com/watch?v=CiDe-BaMo9g) sorry that the audio is a bit quiet, you can still see it works though |
| --- |

**Teacher Checklist:**

**AS91879- Develop a digital outcome to manage data**

| Achieved- Develop a digital outcome to manage data | **Evidence** |  |
| --- | --- | --- |
| using appropriate tools and techniques to structure, organise, query and present data for a purpose and end user |  | ✓ |
| applying appropriate data integrity and testing procedures |  | ✓ |
| describing relevant implications. |  | ✓ |
| Merit- Develop an informed digital outcome to manage data |  |  |
| using information from testing procedures to improve the quality and functionality of the outcome |  | ✓ |
| structuring, organising and querying the data logically |  | ✓ |
| addressing relevant implications. | Functionality and Usability not met- crashes when exiting option 4 and forces users to type in things rather than menu options (even case sensitive!) | - |
| Excellence- Develop a refined digital outcome to manage data |  |  |
| iterative improvement throughout the development and testing process |  |  |
| presenting the data effectively for the purpose and to meet end-user requirements. |  |  |

Final grades will be decided using professional judgement based on a holistic examination of the evidence provided against the criteria in the Achievement Standard.

**Develop a computer program**

**Credits:** 4 (Internal)

**NZQA:** <http://www.nzqa.govt.nz/nqfdocs/ncea-resource/achievements/2018/as91883.pdf>

| **Achieved**  **Develop a computer program** | **Evidence** |  |
| --- | --- | --- |
| Wrote a program that performs a specific task using a suitable programming language |  | ✓ |
| Set out the program code clearly |  | ✓ |
| Documented the program with comments |  | ✓ |
| Tested and debugged to ensure that it works on a sample of expected cases | In general;- other than random crash on exit. | ✓ |
| **Merit**  **Develop an informed computer program** |  |  |
| Documented the program with variable names and comments that describe code function and behaviour |  | ✓ |
| Following conventions of the chosen programming language |  | ✓ |
| Tested and debugged the program in an organised way to ensure it works on expected and relevant boundary cases | Not enough. |  |
| **Excellence**  **Develop a refined computer program** |  |  |
| Ensured the program is a well structured logical solution to the task |  |  |
| Making the program flexible and robust |  |  |
| Comprehensively tested and debugged the program |  |  |

Comments: