Word Count: 1121 Word

Date: 29/08/2023

**1. Introduction**

In this coursework, the aim is to design and develop a prototype database application for an IT Store that provides different product and services for its clients. This project involves creating a relational database system that includes entity structures, SQL queries, relationships, graphical user interface, and user-friendly features using the Microsoft Access Software. The project's primary focus is to address the needs of the client to improve the efficiency and usability of their system.

* 1. **Client Description**

The client operates a store that feed the needs of various customer that are interested in IT gadgets and services, such as Video games, PC’s, Play stations, Xboxes and Nintendo switch, besides other repairing services, charging cards and many other services.

**1.2 Current System Overview**

The existing system relies on manual processes for tracking inventory and managing sales and rental purchases for the video game store section. This approach is prone to errors, lacks efficient reporting mechanisms resulting a lack of information about the video game store section, as well as tracking the revenue.

**1.3 End-Users**

The end-users of the system include the store administrators, sales personnel, and customers who browse and purchase products through the online shop, in which allowing both the employees and the customers to use the system.

**2. Analysis**

**2.1 Current System Limitations**

The current system lacks real-time updates and reliant on manual processes, leading to inaccuracies in inventory control and management causing limitation in producing up-to-date reports and item stock regarding the products of video games, here are examples of the current system issues:

Sales and Rental Transaction recording: The lack of a systematic system for recording sales and rental transactions makes accurate revenue calculation and client involvement difficult.

Reporting: The store's inability to acquire information about its video game business performance and income production is complicated by the lack of an efficient reporting structure.

**2.2 Functional Requirements**

Our client requires a database application that can handle inventory management, order processing, customer data management, and comprehensive reporting. The application should provide a user-friendly interface for seamless user interaction.

Data Integrity: data security should be achieved through login and registration forms and entities for both employees and customers.

Game Management: Allowing store employees to add, edit, and delete games information, including details such as title, platform, genre, developer, supplier, rental price, and purchase price.

Sales and Rentals: facilitating the processing of both sales and rental transactions that can be done by customers, in addition to that capturing customer details, game titles, transaction dates, and transaction amounts in Orders entity.

User Accessibility: ensuring ease of use and navigation to both store employees and customers.

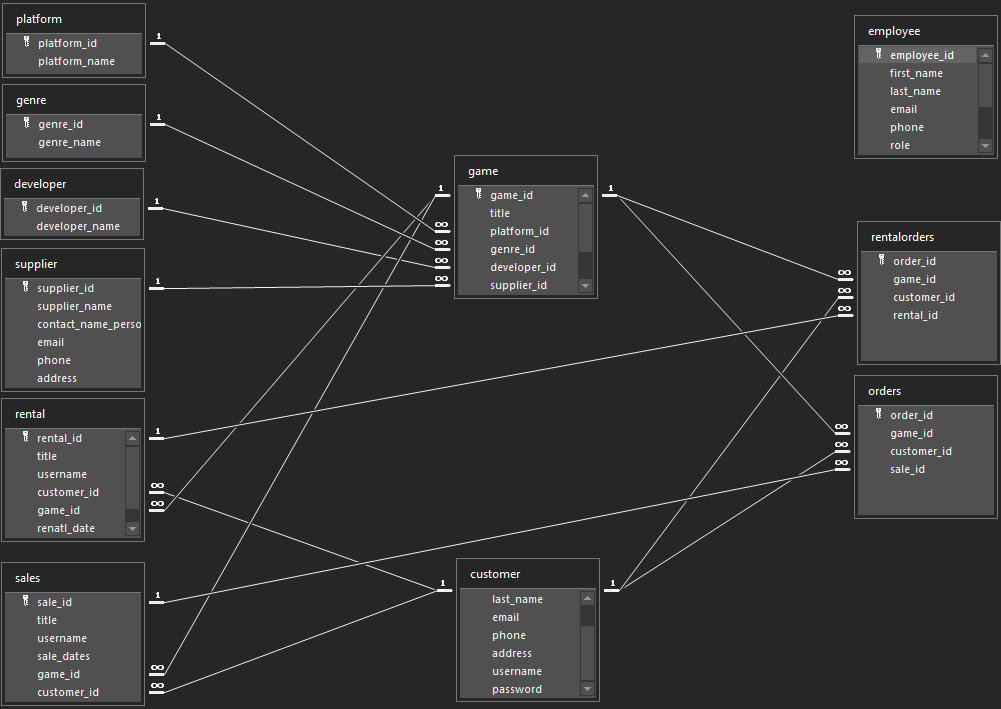
**2.3 Client Benefits**

The video game database application will provide several benefits to both employees and clients by providing efficient games management, enhanced sales and rentals tracking. Witch will Enhance the Customer Experience, improve decisions about stocking, pricing, and marketing strategies.

**3. Design**

**3.1 Entity-Relationship Diagram**

The employee table was created to store employee’s information and to ensure data integrity through hiding details from the end user using registration and login.



**3.2 Normalization**

1NF:

Order (order-id, rental-id, rent-date, sale-id, sale-date, employee-id, employee-name, employee-Phone, employee-username, employee-password, customer-id, customer-name, customer-Phone, customer-username, customer-password, game-id, title, rental-price, purchase-price, available, platform-id, platform-name, genre-id, genre-name, developer-id, developer-name, supplier-id, supplier-name, contact-name-person, phone, address)

The primary key was identified as order-id in which the rest of the attributes are related to it, at this point all the values were atomised.

**2NF:**

Partial dependences were removed as follows:

**Game** (game-id, title, rental-price, purchase-price, available,)

**Platform** (platform-id, platform-name)

**Genre** (genre-id, genre-name)

**developer** (developer-id, developer-name)

**supplier** (supplier-id, supplier-name, contact-name-person, phone, address)

**customer** (customer-id, customer-name, customer-Phone, customer-username, customer-password)

**employee** (employee-id, employee-name, employee-Phone, employee-username, employee-password)

**order** (order-id, rental-id, rent-date, sale-id, sale-date)

**3NF:**

Removing transitive dependencies:

**Rentorder**

|  |  |  |  |
| --- | --- | --- | --- |
| order-id | customer-id | game-id | rent-id |

**Saleorder**

|  |  |  |  |
| --- | --- | --- | --- |
| sale-id | customer-id | game-id | sale-id |

**Game**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| game-id | platform-id | genre-id | developer-id | supplier-id | title | rental-price | purchase-price | available |

**customer**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| customer-id | customer-name | customer-Phone | customer-username | customer-password |

**employee**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| employee-id | employee-name | employee-Phone | employee-username | employee-password | |
|  |  |  |  |  |

Platform

|  |  |
| --- | --- |
| platform-id | platform-name |
|  |  |

**Genre**

|  |  |
| --- | --- |
| genre-id | genre-name |
|  |  |

**developer**

|  |  |
| --- | --- |
| developer-id | developer-name |
|  |  |

**Sale**

|  |  |
| --- | --- |
| sale-id | sale-date |
|  |  |

**rent**

|  |  |
| --- | --- |
| rent-id | rent-date |
|  |  |

**supplier**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| supplier-id | supplier-name | contact-name-person | phone | address |
|  |  |  |  |  |

The unnormalized form have been normalised till the 3NF, however in the 3NF one transitive dependency haven’t been removed (see Game table in 3NF, the transitive dependency id highlighted)

**4. Testing and Validation**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Number** | **Purpose of Test** | **Test Data** | **Expected Result** | **Actual Result** | **Comment** |
| 1 | Interfaces work correctly, dataflow. | Entities-relationship. | Primary and foreign keys identified correctly. | All tables are related using 1:M relation using bridge tables. | Return to the (E-R) Diagram look for bridge table called orders which links customers with sales and rentals. |
| 2 | Data-values are stored correctly. | Datatypes | Assigning correct Data types to each attribute in the database to ensure data are stored correctly | All datatypes are correct but require data formatting | See Picture “Test 2” |
| 3 | Boundary-conditions | Relation data, not random data. | if a value in a field should be from 1 to 12, it should accept 1 till 12 and rejects 13 and 14. | All data are related | See Picture “Test 4” |
| 4 | Integration-testing, checking if sales and orders are communicating correctly. | Database Integration | After adding a sale an order should be placed automatically carrying the game-id and customer-d form the sale table. | Successfully, a query used | See Picture “Test 1” |
| 5 | System-testing User Registration | User-details for new registration | Enter existing username | User registration failed | Duplicate username detected |
| 6 | Data integrity | Login Functionality | Functional login | Successful login | See Picture “Test 3” |
| 7 | Sales Transaction | Invalid sales transaction details | Customer-id doesn’t much customer-username | Sales transaction failed | invalid data provided |
| 8 | Compatibility and performance | transactions and game amending performance issues. | Successful transactions | For instance, after placing a first sale just after opening the application it doesn’t save the record. | After the first time it start functioning. |
| 9 | Alert and amending screen messages. | Massage box | Displaying amending, adding, deleting message boxes. | Sufficient message boxes |  |
| 10 | Saving values that matches the data formatting | Data formatting | Saving formatted values | Absence of data formatting in the application such as the hashed password. | See Picture “Test 5” |

(VLEDMU, 2023)

**5. Evaluation**

The new database solution successfully addresses the client's requirements for video games inventory management, order processing, and reporting. As well as facilitating user-friendly interfaces. While most user requirements have been met, we acknowledge certain limitations and suggest improvements for future iterations. For instance, data formatting, transactions and game amending performance issues.

**6. Conclusion**

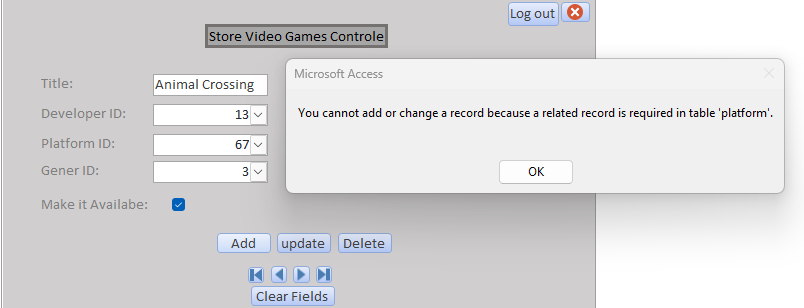
This coursework provided valuable insights into the intricacies of designing and developing a functional database application. The project management and development cycle were essential in successfully delivering a prototype that meets most of the client's requirements. As future iterations of the system are considered, addressing the minor bugs highlighted earlier will undoubtedly contribute to refining the overall application.

**7. Referencing**

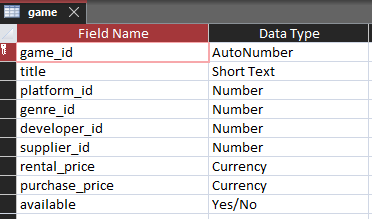
Meets DMU (no date) Testing and Validation, Blackboard-DM&R. Available at: https://vle.dmu.ac.uk/webapps/blackboard/content/listContent.jsp?course\_id=\_613443\_1&amp;content\_id=\_6349124\_1 (Accessed: 29 August 2023).

**8. Appendices**

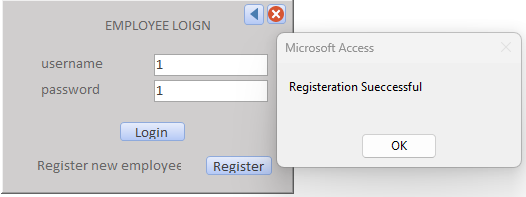
**Test 1:**



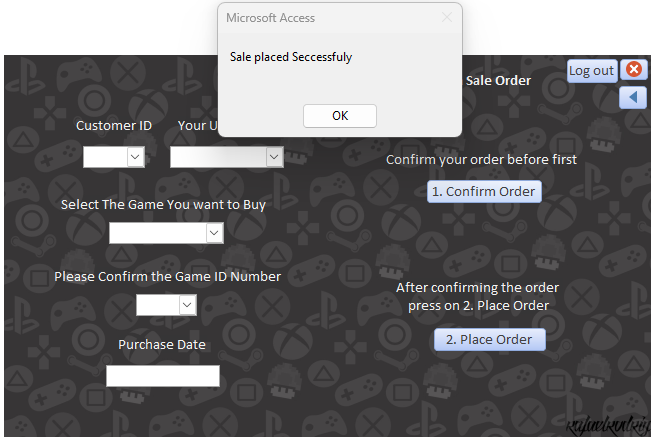
Test 2:



Test 3:



Test 4:



Test 5:

