**Online Game Store Database:**

**Purpose:**

The objective of this application is to maintain records of the different products in the Video Game store as an inventory system, the records of the customers of the store and keeping track of their order history including the order status. Thus this application serves the purpose of an inventory and a transactional system that can be used by the staff, inventory administrator and the customers.

**Scope of the application**:

This application shows the list of different products available in the game store. This allows for a systematic organization of the store’s inventory, order history and products purchased. The application enables the customers to view the status of the orders they placed. As the products are ordered/ when any order is aborted, the inventory status is updated with the current count of products present.

Whenever an item is requested for purchase, after a lookup in the inventory for the availability, the request is approved and the order shows up in the “Customer Orders” table with status “ordered”. If there is no availability, the request is simply be denied with a message displayed to the user regarding the amount of available items of that product.

**Database Tables:**

1. **Product Table:**

**Product\_id *PRIMARY KEY* : An auto-generated field as the primary key**

**Description : Product description of whether the product is a “game” , “accessory” or “console”**

**Pro\_name: Product name/ title**

**Price : Price of the product**

1. **Product inventory Table:**

**Product\_id *PRIMARY KEY* : Primary key for product inventory & a *foreign key* for the product-id from the Product table**

**Product count : Count of the items in the inventory**

1. **Games Table:**

**Games\_id *PRIMARY KEY* : Primary key for games & a *foreign key* for the product-id from the product table**

**Games\_name : Name of the games**

**min\_memory\_size : Minimum memory size required to play the game**

**max\_no\_players : Maximum players the game supports**

**details : other details like what version of the game etc.**

**console\_fk : a *foreign key* for the console id from the consoles table.**

1. **Consoles Table:**

**Console\_id *PRIMARY KEY* : Primary key for consoles & a *foreign key* for the product-id from the product table**

**Drive\_type: Whether the console is an Xbox 360 or Play station 1/2/3/4**

**Size: Memory Size (in GB) the console supports**

**Details: Other details like if the device is upgradable or not and to what level or memory size**

1. **Accessories Table:**

**Accessory\_id *PRIMARY KEY* : Primary key for accessories & a *foreign key* for the product-id from the product table**

**accessory\_name : Name of the accessories**

**details : other details like what size & color of the accessory is available etc.**

1. **Customer Table:**

**customer\_id *PRIMARY KEY* : An auto-generated field as the primary key**

**customer\_name : Name of the customer**

**customer\_phone: Customer’s phone number**

**address: Customer’s address**

1. **Customer Orders:**

**order\_id *PRIMARY KEY*: An auto-generated field as the primary key**

**date\_order: Date when the order is placed**

**product : a *foreign key* for the product-id from the product table indicating which product is ordered**

**product\_count: Count of the items of the product ordered**

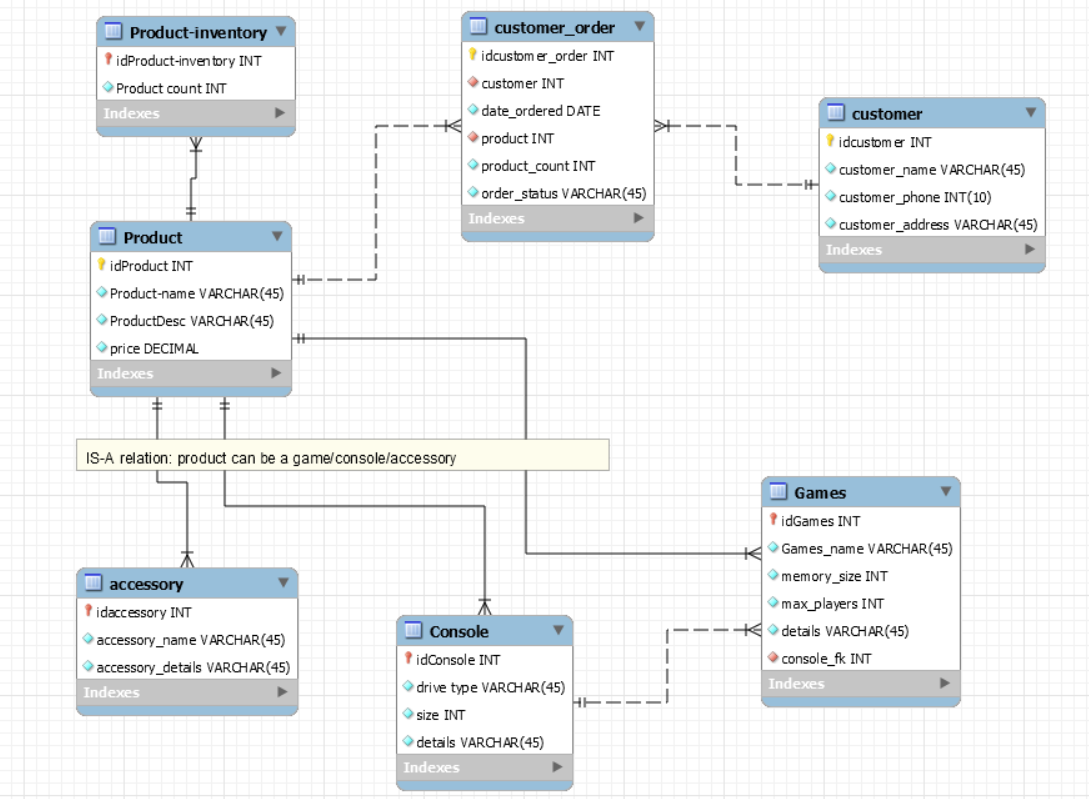
**customer: a *foreign key* for the customer-id from the customer table indicating which customer has ordered**

**order\_status: Status of the order placed whether “ordered”, “delivered” or “aborted”**

**Triggers:**

1. **Prod\_Order\_Insert: Trigger BEFORE INSERT ON customer\_orders that decrements the requested amount from the product inventory for orders with status as ‘ordered’ OR ‘delivered’.**
2. **Prod\_Order\_Update: Trigger BEFORE UPDATE ON customer\_orders that decrements the requested amount from the product inventory when order status changes from ‘ordered’ to ‘aborted’ or from ‘aborted’ to ‘ordered’/’delivered’. And even when there is a change in the order for count of items requested.**

**ER- Diagram:**

****

**NOTE:**

**There is an IS-A relationship (specialization) between products and the three entities(accessory, console, games) using an inverted triangle or as shown in this link:** <http://www.databaseanswers.org/data_models/game_shop/index.htm>

**Functions & Views:**

1. **Products & Product Inventory:**

**The first view lists all products and allows you to add a new one. On the view page, there are options for edit & delete individual records. Sorting based on highlighted titles is also possible – Description, Product Names and Price. Besides one can also view the availability and the orders for a particular product (Orders filtered by each product). Viewing availability shows the count in the inventory and allows for updating the same. Viewing orders shows the orders placed against that product and allows for update/delete of individual orders. There is also a link to add a new order.**

**Apart from this the view products page has the links to views of the different product categories- Games, Consoles & Accessories. The link for add functionality for each of these tables on their respective view pages shows up only if there are no pending products from the Product Tables that are to be added in these tables.**

**The dropdown menu on the add page for each of these tables also lists only those products from the Product table that are pending to be added into these tables, thus maintaining the one-to-one relationship between product and each of the product categories. Filtering from the Product Table into these categories’ tables is done based on the description attribute of the Product Table.**

1. **Customers:**

**Lists all the customers and allows you to add a new customer to the database. Similar options are there for edit/ delete individual customer records and sorting based on highlighted titles – Customer id and customer name. This section also allows one to view/edit/delete orders made by each customer (Orders filtered by each customer). There is also a link to add a new order.**

**There is another link that allows one to select different combinations for Customers against Products and view the orders for the same combination. This query might be helpful if the organization wants to study the buying patterns/ preferences of each customer.**

1. **Customer Orders:**

**Lists all the customer orders and allows one to place a new order. View page of orders allows one to edit/ delete individual orders. Sorting based on highlighted titles is also enabled- Order-id, Order Date, Product Count, Customer Name, Product Description and Order Status. There is also a link on the view page which redirects the user to a page where they can view the orders within different date ranges in the timeline.**

**Major Challenges:**

1. **Updating the inventory based on Order Status:**

**One of the major challenges in this application was to keep the Product Inventory consistent with the orders placed, their quantity and their status. For this I used two triggers – one before insert on customer\_orders and one before update on customer\_orders. I had to keep these cases in mind while designing the triggers:**

* **New order with aborted status simply means no change to database**
* **New order with ‘delivered’/’ordered’ should decrement the inventory by respective count.**
* **Update order with new product count or new order status and reflect that in the inventory accordingly.**
* **Inventory changes should be made only when order status changes from ordered to aborted, or from aborted to ordered or delivered. No changes will be made if status changes from delivered to ordered/aborted, but logically no one will do this in a real-time application.**

1. **Check for availability**

**Check for availability in the inventory before placing the order or editing the order for increased product counts. An error message is displayed to the user if it demand is more than availability.**

1. **Creation of product records in the inventory for every new product in the database.**
2. **Product Categories- Accessories/Games/Consoles**

**I had to figure the logical view page for show/add/update the product categories- accessories, games & consoles. Delete is not allowed on these tables, as delete will automatically be done when the user deletes the main product.**

**I have also prompted the user to add any pending records from the Product Tables into the individual categories’ tables on the latter’s respective view pages. The add link being displayed only when there are pending records. Even the dropdown menu for the product on each page is loaded with only the pending items. This allows for ensured one-to-one relationship between tables, maintaining integrity across the table records and preventing users to run into unnecessary errors while using the application.**

**Create-Read-Update-Delete Functions:**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Tables | Product | Product Inventory | Customer | Customer Order | Games | Accessories | Consoles |
| Create | add\_product.php | Inside add\_product.php | add\_customer.php | add\_order.php | add\_games.php | add\_accessories.php | add\_consoles.php |
| Read | view\_products.php | view\_inventory.php | view\_customers.php | view\_customer\_orders.php | view\_games.php | view\_accessories.php | view\_consoles.php |
| Update | edit\_product.php | update\_inventory.php | edit\_customer.php | edit\_orders.php | edit\_games.php | edit\_accessories.php | edit\_consoles.php |
| Delete | delete\_product.php |  | delete\_customer.php | delete\_order.php |  |  |  |

**Other scripts:**

**view\_cust\_orders.php : View orders for each customer**

**view\_orders.php: View orders for each product**

**cust\_product.php: View orders for different combinations of customers & products**

**order\_timeline.php: View orders between the selected date range**

**index.php: Index page**

**mysqli\_connect.php: Connection to database**

**Styles:**

**style.css : only used for table formatting of results displayed**

**Major Development Changes Since Proposal:**

**The login/logout functionality I proposed that allows for different permissions for various tables depending on the different potential users - Customers/Sales Person/Admin can definitely be the next advanced feature for the application. I have included all the required basic and quite a few useful functionalities as explained above in this application.**

**The major changes being, the way I have shown the tables for different product categories, and allowing edit on all the category records, allowing addition of only the pending records and not allowing direct delete as delete would only be when the main product is deleted from the product table.**

**Next one, is the decision to make a record for new product in the product inventory on insertion of a new product in the product table, allowing user to update the inventory later and prompting the user to do the same.**

**Also the triggers and validations for product availability on the scripts have helped me a long way in keeping the product inventory count consistent with the actual transactions during the addition/ update of new orders. I have tried to make this functionality work for all the possible cases that I could think of.**