# **Exp19\_Access\_Ch05\_Cap - Varmel Studio Toys 1.0**

## **Project Description:**

You work as the database manager at Varmel Studio Toys, located in Lansing, Michigan. Varmel Studio Toys has the world's largest selection of collectible superhero posters and action figures. Your task is to add validation rules, create a form that data entry associates can use to add new items to the inventory, and create a new table to classify items as a poster or action figure. You will also use queries to analyze existing data.

## **Steps to Perform:**

| **Step** | **Instructions** | **Points Possible** |
| --- | --- | --- |
| 1 | Start Access. Open the file named *Exp19\_Access\_Ch05\_Cap\_Varmel\_Studio\_Toys.accdb*. Grader has automatically added your last name to the beginning of the filename. | 0 |
| 2 | You will save the database file with a new name and create a new table that will be the source for a lookup.  Save the database file as Exp19\_Access\_Ch05\_Cap\_Varmel\_Studio\_Toys\_LastFirst.accdb. Use Design view to create a new table. Add **ItemTypeID** as the first field name, with data type AutoNumber; add **ItemTypeDescription** as the second field name, with data type Short Text and field size **15**. | 8 |
| 3 | Ensure ItemTypeID is set as the primary key. Save the table and name it **Item Types**. Add two records: **Action Figure** and **Poster**. Close the table. | 8 |
| 4 | You will edit the Inventory table design to validate data. You will make two fields required and add a validation rule to a field. You will also make sure you test the validation to make sure the rules work as intended.  Open the Inventory table in Design view. Set the InventoryQty and ItemTypeID fields to Required. | 6 |
| 5 | Establish a validation rule for the InventoryQty field that requires the value to be greater than or equal to zero. Create validation text for the InventoryQty: **The value of this field must be 0 or greater.** (include the period). Save the table. Switch to Datasheet view. Change the InventoryQty in the first record to **-2**. The validation text appears. Close the message box, then press Esc to restore the original InventoryQty value. Close the Inventory table. | 6 |
| 6 | To help keep data input consistent, you will add input masks to the Phone fields in the Employees table.  Open the Employees table in Design view. Add a phone number input mask for the Phone field, storing with the symbols. Save and close the table. | 16 |
| 7 | You will convert the ItemTypeID field in the Inventory table to a lookup field, using the new table you created previously as the source for the values in the lookup field.  Open the Inventory table in Design view. Change the Data Type of the ItemTypeID field to Lookup Wizard. Use the Item Types table for the values in the lookup field, select both fields in the table, accept the default sort, hide the key field from the user, and then accept the default name *ItemTypeID*. Save the table. Switch to Datasheet view. Change the item type to *Poster* in the first record and click the second record. If the record updates successfully, change the first record back to *Action Figure*. Close the table. | 8 |
| 8 | You will modify an existing query to add a parameter so employees doing data entry can quickly get a list of inventories below a certain level.  Open the Find Low Inventory query in Design view. Add criteria for the InventoryQty field. The user should be prompted to **Enter Threshold**. The query should display all results that are less than or equal to the threshold but greater than or equal to 1. Run the query. Enter **2** when prompted to Enter Threshold. You should have two results. Save and close the query. | 12 |
| 9 | You will modify the Rounded Item Prices query to round retail values for items in the inventory.  Open the Rounded Item Prices query in Design view. Create a new column to round the Retail price of each item to the nearest dollar. Name the field **RoundedRetail**. Save the query. Create a new column to display **Premium** for all items that have a RoundedRetail value of $35 or more and **Standard** for items that are less than $35. Name the field **Class**. Run the query. Save and close the query. | 12 |
| 10 | You will also create a query to display employees who are in line for a performance review.  Open the Overdue Employee Reviews query in Design view. Add a new column to determine if an employee’s performance review is overdue. If the employee’s DateOfLastReview is null, it should display **Overdue**. If not, it should display nothing. Name the column **ReviewStatus**. Add criteria of **Overdue** to the column you just created, so only the employees who are Overdue display. Run the query. Ensure only employees with null DateOfLastReview display. Save and close the query. | 12 |
| 11 | You will modify an existing query displaying daily totals to instead display monthly totals.  Open the Order Totals By Month query in Design view. Change the first column so that instead of grouping by the order date, you group by the month. Use the DatePart function to extract the month from the date. Name the column **MonthNumber**. Run the query. The first line should read 5 (as the month, representing May), with a total of *$658.72*. Save and close the query. | 12 |
| 12 | Save the database. Close the database, and then exit Access. Submit the database as directed. | 0 |

|  |  |
| --- | --- |
| **Total Points** | **100** |