# DB Diagram



# Self-evaluation

In order to gather correct information, we have built the Order Management System to handle data by crude operations and filtering. We attempted to embed the picture in our project, but it failed, so we had to recreate our application to correct any mistake. For crude operations we used stored procedure to satisfy the demand but mostly because of the incorrect declaration of parameters and the GetById approach we have got a null reference exception. It was not so straightforward to keep the development stable due to errors in ID and controller transfer. In addition, our lecturer and Shamim Uddin had tutorials to solve this situation. (Uddin, 2019) This program has overall been built to handle retail shop orders, where the consumer can add and monitor orders quickly. Further improvement can be made with the use of values from other tables, but due to lack of experience using SQL it was not introduced. CRUD acronym means computer programming to be developed, interpreted, updated and deleted. That are the four basic characteristics of continuous storage. In addition, any character in connection database apps mapped in standard HTTP, SQL, DDS or DDS type can be matched by each letter in the acronym. (Altvater, 2017) Instead of using Ad-hoc SQL statements, many programmers use CRUD due to their efficiency. When a saved operation is first executed and reuse for all storage process applications, the execution plan will be saved in the SQL Server database cache. The link engine will search the method cache before a SQL statement is executed on SQL Server to ensure an active SQL statement execution plan is available and use the existing plan to decrease the need to refine, analyze and recompile SQL statement action. We used DML triggers at the database level to perform log auditing and monitor crud operations in the database. Another DML trigger was used to protect data in the database by removing erase functions without any conditions. Causes in SQL Server are essentially database objects that are a kind of mechanism that "reacts" to such database operations that we perform. The basic premise behind the stimulation is that when anything happens, they also do an action. Although we are concerned about DML triggers, these updates are changes to our records. (Drkusic, 2020) We learned a lot in practices and seminars, which allowed us to apply what we learned to add basic features to an application. However, since photos and checkboxes were not needed in our business case, we left them alone.

We also learned how to export and import data from and from databases, which helped us to apply it in our case by importing data into the order table and retrieving it from the database.

|  |  |  |  |
| --- | --- | --- | --- |
| Test description | Expected results | Actual results | Comments |
| GetOrders() | All the data is selected and mapped correctly | All the data is selected and mapped correctly | Used Dapper nugget package for using this template |
| Insert(Order order) | All the data is taken from corresponding fields and inserted to database using stored procedures | Data is taken, but not inserted | After adding parameter with wrong spelling, it couldn’t find and approach this parameter from stored procedure |
| Insert(Order order) | All the data is taken from corresponding fields and inserted to database using stored procedures | Data is taken, but not inserted | When nullable column was lived as null, got the null exception |
| Insert(Order order) | All the data is taken from corresponding fields and inserted to database using stored procedures | All the data is taken from corresponding fields and inserted to database using stored procedures | Data was taken and written to the database. |
| UpdateSP(Order order) | All the data is taken from corresponding fields and order is updated according to its id using stored procedures | Edit getting method didn’t work properly, because GetOrderByIdSP couldn’t return id of the order from db | Stored procedure didn’t pass id because we forgot to assign order.OrderId=id; |
| UpdateSP(Order order) | All the data is taken from corresponding fields and order is updated according to its id using stored procedures | All the data is taken from corresponding fields and order is updated according to its id using stored procedures | Old data is taken to the view edit, and editing worked properly |
| Delete(int id) | Delete one record from the table by using its id | Delete one record from the table by using its id | Delete worked properly |
| List<Order> Filter(  string StoreName,  int ManagerOrderId,  int CustomerOrderId,  int SellerOrderId,  int ItemOrderId,  out int totalCount,  string sortColumn,  bool sortDesc = false,  int? page = 1,  int pageSize = 2  ); | Data paging, sorting and filtering by corresponding fields implemented. Data should take StoreName, ManagerOrderId, CustomerOrderId, SellerOrderId and ItemOrderId | Data paging, sorting and filtering by corresponding fields implemented. Data should take StoreName, ManagerOrderId, CustomerOrderId, SellerOrderId and ItemOrderId  Thus was implemented another model, OrderFilterViewModel.  And worked with paging filtering and sorting columns. | Filter view worked properly |
| ActionResult ExportJson(OrderFilterViewModel model) | Export Json Data from Filter | Exported Json Data from Filter | Worked properly |
| ActionResult ExportXml(OrderFilterViewModel model) | Export Xml Data from Filter | Exported Xml Data from Filter | Worked properly |
| public ActionResult ExportCsv(OrderFilterViewModel model) | Export Csv Data from Filter | Exported Csv Data from Filter | Worked properly |
| ActionResult ImportJson(IFormFile importFile) | Import Json Data to table | Imported Json Data to table | Worked properly |
| ActionResult ImportXml(IFormFile importFile) | Import Xml Data to table | Imported Xml Data to table | Worked properly |
| ImportCsv(IFormFile importFile) | Import Csv Data to table | Import Csv Data to table | Worked properly |
| ActionResult Index() | All data is shown in Index view | All data is shown in Index view | Worked properly |
| ActionResult Details(int id) | All data is shown in Details view | All data is shown in Details view | Worked properly |
| ActionResult Create() | All data is shown in Create view | All data is shown in Create view | Worked properly |
| ActionResult Edit(int id) | All data is shown in Edit view | All data is shown in Edit view | Worked properly |
| ActionResult Delete(int id) | All data is shown in Delete view | All data is shown in Delete view | Worked properly |

# Reference list

Drkusic, E., 2020. *Learn SQL: SQL Triggers*. [online] SQL Shack - articles about database auditing, server performance, data recovery, and more. Available at: <https://www.sqlshack.com/learn-sql-sql-triggers/> [Accessed 12 April 2021].

Altvater, A., 2017. *What are CRUD Operations? Examples, Tutorials & More*. [online] Stackify. Available at: <https://stackify.com/what-are-crud-operations/> [Accessed 12 April 2021].

Uddin, S., 2019. *CRUD Operations Using ASP.NET Core And ADO.NET*. [online] C-sharpcorner.com. Available at: <https://www.c-sharpcorner.com/article/crud-operations-using-asp-net-core-and-ado-net/> [Accessed 8 April 2021].