Vi D, Scotty B, Aleu A, Samuel H, Alix S

December 5, 2024

Professor Jones Yeboah  
**LockHeed Martin Database Report**

**Introduction**  
 Our group decided to take a deep dive into the aerospace/defense sector by exploring Lockheed Martin with a simple overview. The United States and other countries spend a lot of money and resources on these large defense companies. Our stated objective being such that we can demonstrate our ability to make a small scaled down database of what they may have at Lockheed Martin.

Lockheed Martin is an example of the many organizations that develop objects for military use and defense. These technologies are also used for aerospace purposes such as satellites. Therefore, as an organization, Lockheed Martin must have a database to manage all sorts of data throughout the company for matters such as the buyers of the products through contracts, the staff members, the products, etc.

**Mid-term Recap**

We decided upon deep deliberation to settle on the following entities to help shape a clear and coherent database. They are as follows: Organization, Staff, Departments, Products, Contracts, orders and lastly projects which was a last minute addition to the database due the consoul we sought from our professor.

The attributes of each entity have been shaped after careful arbitration based on what we believe would make the most sense for our scaled down model of a hypothetical database that an aerospace giant like Lockheed Martin would have. Provided below in table form are the attributes.

| **Entities** | **Attributes** |
| --- | --- |
| Organization. | OrganizationID, Organization Name, Country, OrderID, Address |
| Staff | Staff ID, Fullname, Title, Role, Department ID, Email, Phone |
| Department | Department ID, Department Name, Location, Budget |
| Product | ProductID, ProductName, Price, Product Type |
| Order Details | OrderID, Country, Organization Name, ProductID, Organization ID, Contract Length, Date Established, Active Status |
| Project | Project ID, Staff ID, Product ID, Organization ID |

We had spent a lot of time working sorting out the best methodology to get the more important information into the database. An example of things related to each other would be Projects having a one to many with staff, product, organizations.

**Database Design Process**

In this section, we begin by reviewing the planned table relationships established during the design phase. Specifically, we will examine the primary relationships between the tables, categorizing them as either independent or dependent.

**Relationships:**

**Independent Tables**

**Organization -** Data on the Organizations that Lockheed may sell to

**Product -** Data on the products they produce and sell **Department -** Data for the departments that will relate to staff

**Dependent Tables**  
 **Project** is dependent on the **Organization, Product, Staff**

**Staff** is dependent on the **Organization**

**Order Details** are dependent on **Organization** and **Product**

Now that these relationships have been established, we will outline the entities, primary keys, foreign keys, and attributes associated with each table:

**Table: Entities, Attributes, Primary Keys, and Foreign Keys (if any)**

| **Entities** | **Primary Key** | **Foreign Key(s)** | **Attribute** |
| --- | --- | --- | --- |
| **Organizations** | OrganizationID | None | OrganizationID, Organization Name, Country, Address |
| **Staff** | StaffID | DepartmentID | StaffID, Fullname, Title, Role, DepartmentID, Email, Phone |
| **Departments** | DepartmentID | None | DepartmentID, Department Name, Location, Budget |
| **Products** | ProductID | None | Product Name, Price, Product Type |
| **Order Details** | OrderID | ProductID, OrganizationID | OrderID, Country, Organization Name, ProductID, OrganizationID, Contract Length, Date Establish, Active Status |
| **Projects** | ProjectID | StaffID, ProductID, OrganizationID | Project ID, Staff ID, Product ID, Org ID |

**Physical Database Design**

Our database includes six entities, designed with a streamlined schema to simplify its construction. While all data is valuable, we prioritized the most critical attributes by setting them to NOT NULL, ensuring that essential fields are never left blank. For simplicity, we used the NVARCHAR data type for all attributes to reduce decision-making complexity. The NOT NULL constraint was applied most frequently to key fields that the database relies on, such as FullName in the 'Staff' section. After all, it would be superfluous to have a staff member without a name  
 We decided not to implement indexes in our database because the scope of our project is relatively small. While indexes are a powerful tool for quickly retrieving data by avoiding a full table scan, the size of our dataset does not warrant their use. Given the limited scale of our database, we determined that the performance benefits of indexing would be negligible and outside the requirements of our design.  
 Foreign keys play a crucial role in the functionality of our database tables. Understanding how keys link tables provides clarity and makes the structure of the database easier to reference and understand. The entity with the most foreign keys is 'Projects,' which has three: OrganizationID from the 'Organization' table, StaffID from the 'Staff' table, and—believe it or not—ProductID from the 'Product' table. This demonstrates a clear pattern with how our entities with our database are related.

**Inserting Data**

During the data insertion process, we encountered an issue with foreign key dependencies. Initially, our plan was to create the tables and populate them fully. However, some primary keys were missing at the start, which caused errors when inserting related foreign keys. To resolve this efficiently, we decided to insert the data in stages.

We began by inserting all data that did not rely on foreign keys, deferring any foreign key-related entries to a later step. For example, here is the data insertion for the **Organizations** table, which other tables (e.g., Order Details) depend on through its primary key:  
-- Data for Organizations Table

INSERT INTO Organizations (OrganizationID, OrganizationName, Country, Address)

VALUES

('C001', 'U.S. Department of Defense', 'United States', '1400 Defense Pentagon, Washington, DC 20301-1400'),

('C002', 'Battle Organization', 'Taiwan', '123 Maple Street, Taipei, Taiwan'),

('C003', 'Transport Organization', 'Poland', '124 Oak Avenue, Warsaw, Poland'),

('C004', 'Tech Organization', 'United Kingdom', '125 Pine Road, London, UK'),

('C005', 'Transport Organization', 'Mexico', '126 Cedar Lane, Mexico City, Mexico'),

This example demonstrates the insertion of the first 5 records into the **Organizations** table, which is a prerequisite for tables that use its primary key as a foreign key. The remaining records will be added subsequently to ensure all dependencies are satisfied.

**Challenges Faced**

Throughout the construction of our tables, we encountered several issues that disrupted the process. One significant problem arose during the creation of data tables due to improper handling of foreign keys. Our initial plan involved creating tables in a predetermined order, but this approach caused errors because it didn’t account for the dependencies between tables. In hindsight, we should have started with independent tables to make the script more efficient and avoid these conflicts.

To save time and proceed with our queries, we adjusted our approach by creating **INSERT** statements specifically for the foreign key fields. However, this issue wasn’t limited to table creation; it also extended to the data insertion phase. The most logical strategy for data insertion would have been to populate all primary keys in the independent tables first. Instead, we repeated our earlier mistake of inserting data in a disorganized order based on the initial table structure from the planning phase. This led to incomplete records where foreign key fields were left as NULL. As a result, we had to manually correct these records using **UPDATE** queries, which was time-consuming.

Additionally, another issue arose when we had outdated data that no longer aligned with our final design. This caused further errors because the old data was no longer valid or relevant. To address this, we inserted placeholder data or similar content where appropriate to ensure consistency and avoid leaving fields as NULL.

**Conclusion**

This database project showcases the combined effort of five team members, demonstrating not only our SQL skills but also our ability to analyze and design a functional database. While this project was primarily a thought experiment rather than an industry-changing endeavor, it provided valuable insights into the aerospace industry and allowed us to explore the operations of a major company like Lockheed Martin in great detail. The experience itself holds significance as it deepened our understanding of database design and industry analysis.

In the future, this database could be expanded to include more comprehensive data about Lockheed Martin's civilian sector, such as satellites and other space-related products. Lockheed Martin's contributions to NASA, including spacecraft that have ventured beyond Earth's orbit to destinations like Jupiter, highlight the potential scope of such additions. Another area for improvement is enriching the products section of the database.If you don’t have a product to sell you don’t really have a company.

This project demonstrates the importance of thoughtful database design and highlights opportunities to further refine and expand upon our work, potentially benefiting similar projects in the future.