**Individual Project: Executive Summary:**

**1. Introduction:**

This submission presents the physical implementation of the logical database design shown in unit 6. Here, I present the final diagram of how the database is put together, consider in which database model this database is optimally implemented, e.g., considering SQL versus NoSQL options, and also examine compliance topics to ensure that the database does not only function well but is also protected against malicious actors.

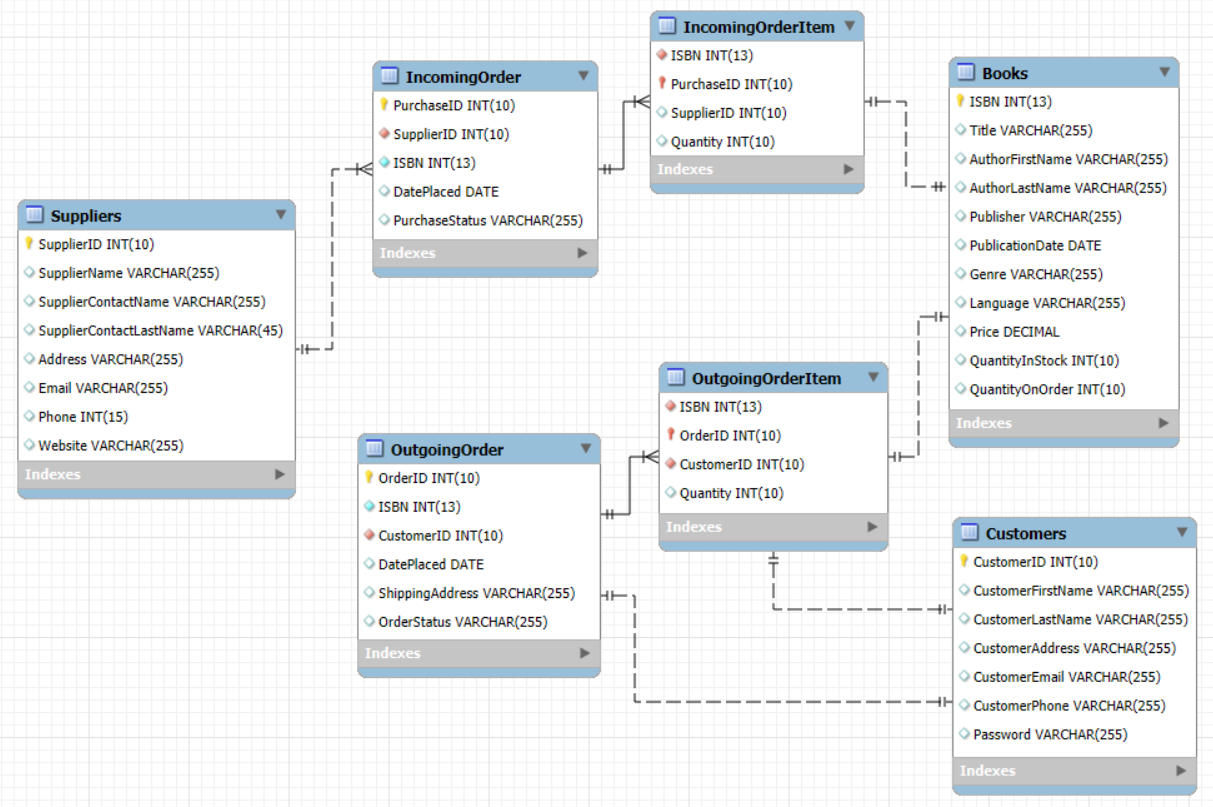
**2. The Physical Design View of the Database:**

Within the logical design phase, we have created the following Entity-Relationship-Diagram (ER-Diagram):

IncomingOrderItem 
ISBN (PK) (varchar (255)) 
Purchase ID (FK) (integer (10)) 
Supplier ID (FK) (Integer (10) 
Quantity (Integer (10)) 
Supplier 
Supplier ID (PK) (integer (10)) 
Name (varchar (255)) 
Supplier contact narne (varchar (255)) 
Address (varchar (255)) 
Email (varchar (255)) 
Phone (unique) (varchar) 
Website (varchar (255)) 
IncomingOrder 
Purchase ID (PK) (integer (10)) 
Supplier ID (FK) (integer (10)) 
ISBN (FK) (varchar (255)) 
Date placed (datetime) 
Purchase status (varchar (255)) 
Outgoingorder 
Order ID (PK) (integer (10)) 
ISBN (FK) (varchar (255)) 
Customer ID (FK) (Integer (10)) 
Date placed (datetime) 
Shipping address (varchar (255)) 
Order status (varchar (255)) 
ISBN (PR) (varchar (255)) 
Title (varchar (255)) 
Author (varchar (255)) 
Publisher (varchar (255)) 
Publication Date (datetime) 
Genre (varchar (255)) 
Language (varchar (255)) 
Price (numeric (19. O)) 
Quantity in stock (integer (10) 
Ouantity on order (integer (10) 
ISBN (PK) (varchar (255)) 
Order ID (FK) (integer (10)) 
Customer D (FK) (Integer 
(10) 
Quantity (Integer (10)) 
Customer D (PK) (integer (10)) 
Name (varchar (255)) 
Address (varchar (255)) 
Email (varchar (255)) 
Phone (unique) (varchar) 
Password (varchar (255)) 

**Figure 1: ER-Diagram from the logical design phase**

While the interrelations of the different entities in the ER-Diagram are represented correctly, there are some minor mistakes which would cause problems in the physical implementation of the database and thus need to be corrected. Figure 2 shows the corrected ER-diagram.

****

**Figure 2: Final ER-Diagram**

Most importantly, some of the keys were changed. I furthermore changed the data type for the phone number from VarChar to integer with a limitation of 15 digits, the maximum number of digits a phone number can have in Germany due to Iam Expat Media (n. d.) , which the online book store for which I create the database design is located in. For normalization purposes, each natural person name has been split into a first name and a last name. As I expect suppliers to be companies the supplier name can remain as it is. However, the customers are typically natural persons and are thus noted with first and last names. In the books table, I changed the datatype for the price to decimal. The ISBN is changed to an integer with maximum 13 digits in line with its definition. I also changed some of the column names, especially for the customers and suppliers tables, to not have the same names for columns in both tables, e.g., for the email.

In the final ER-diagram, I added the type of relationships. The graphic now shows that I expect one supplier to have multiple incoming orders, one incoming order to have multiple incoming order items, one incoming order item is mapped to one book, one book related to one outgoing order item which is mapped to one outgoing order and one outgoing order item as well as one outgoing order are each mapped to only one customer. This helps get a better understanding of how the tables are interrelated. In the graphic, the continuous line stands for identifying relationships and the dashed line represents non-identifying relationships. Due to Geeks for Geeks (2021) identifying relationships mean that the child entity cannot be uniquely identified without the parent entity while in non-identifying relationships the child entity can be uniquely identified without the parent entity.

**3. Evaluation of the Choice of Data Model:**

As underlying data model I choose the relational data model over, e.g., the hierarchical model and the network approach. The hierarchical model presents data in a tree-structure with a clear hierarchical structure. While it is easy to understand and follow, it is inflexible when trying to rearrange the data and a representation of complex data structures can be challenging. Also, it presents data in a redundant form which might lead to problems. For the network approach, the tree-structure is changed for directed arrows from higher roots downwards which allows one child object to have multiple parent nodes. This enables more flexibility than the hierarchical structure and is often preferrable when many-to-many relationships need to be represented. However, it introduces additional complexity into the database design and is hard to maintain. Thus, I decided to use the relational data model which, for the given use case is superior to the two other options. [Geeks for Geeks, 2024]

The relational data model presents data in tables where at the intersection of one row and one column only one data object can be accessed, creating a clear structure that is easy to understand and follow. Tables are split into sub-tables that are linked via keys and can be joined together. This achieves a higher level of flexibility than with any of the other options. Redundancy and mistakes that might exist due to duplication of data entries are mitigated. Querying of data that is presented using the relational data model is easy and straight-forward. In the given use case, the database can be used by the customer’s logistic and sales departments, its customers and suppliers. The different information requirements of these different user groups can be served by a relational database which enables the creation of different views dependent on information requirements. [IBM, n.d.]

However, relational data model also has some drawbacks that need to be kept in mind. Following Datamation (2023), when the size of the database increases, this might lead to performance issues when relying primarily on a relational database. Furthermore, the database design for larger relational database can become complicated due to the vast number of different tables with interrelations that would need to be introduced and understood. Additionally, it must be kept in mind that relational databases do not work well with unstructured data. Wrath (2023) adds that especially the necessity to join multiple tables together for analytic purposes can result in slow queries and associated performance issues. Furthermore, the normalization process associated with relational databases can introduce an additional complexity to the database, especially with large databases with many different tables and data entries.

While these disadvantages present some limitations for the usage of relational databases, they do not limit the suitability of the relational data model for the given use case. The database I designed for the online book store is rather small, with clearly defined interrelations. Querying and merging tables should be sufficiently easy as not an excessive amount of different tables and interrelations is introduced. Still, the drawbacks of the relational data model should be kept in mind by the customer whenever they aim at expanding the database in the future and it must be re-assessed constantly whether the relational model remains the optimal choice.

**4. Choice of Database Management System:**

I firstly compare the advantages and drawbacks of SQL and NoSQL to inform the decision about the best-suited database management system. Following Agrawal (2023), SQL databases profit from consistency and integrity due to the structured approach and clearly defined schemas, making them well-suited to present complex data structures as long as the interrelations are clearly defined. This enables easy querying and gives the developer a high degree of flexibility for database manipulation. DbVisualizer (2023) adds that the long existence of SQL databases results in established standards, case studies and community support structures.

However, SQL databases also face some important drawbacks. Agrawal (2023) names as challenge that the clearly-defined schema can cause problems when data structures change or any unexpected datatype is faced. Changing the schema requires extensive planning and might make a significant downtime necessary. Furthermore, SQL databases scale vertically, by adding more computational power to the existing machine. This makes downtimes inevitable when scaling. This is the reason why extensive scaling fits better to NoSQL databases.

Following Agrawal (2023), NoSQL databases have more flexibility than SQL databases by allowing dynamic schemas, making it unnecessary to modify schemas whenever data structures change. They provide easier scalability than SQL databases because they are horizontally scalable, meaning that they are scalable by adding computational nodes. This also means that the database does not need to be put offline when scaling up. Also, NoSQL databases are well-suited to handle unstructured or semi-structured data.

The drawbacks of NoSQL databases are also well documented. Agrawal (2023) notes that NoSQL databases lack the consistency of their SQL counterparts. They also offer less complex querying functionalities, meaning that the focus on computational speed also comes at a cost. Also, as data-denormalization may become necessary for NoSQL databases, this re-introduces redundancy and thus causes higher storage costs. DbVisualizer (2023) adds that as NoSQL is rather new compared to SQL, there is no standard querying approach yet and there are weaker community support structures than for SQL.

I choose to implement an SQL database management system because it fits the given case better. The online bookstore will prefer query accuracy over query speed because they highly depend on the correctness of the queries. They use the database to determine the stock of books and keep their online information up to date. Incorrect query results would thus have major consequences on their operations. While these queries thus have to be accurate and this creates a demand for the SQL database feature of data consistency, query speed will with SQL still be sufficiently fast. While the scaling of the database is also an important feature, for an online bookstore it typically is not expected to become rapid and should thus still be manageable via an SQL database. Furthermore, the data that is managed in the database is structured and well-defined. The schema that is presented in figure 2 is expected to remain the same and contains clearly defined interrelations among the different tables. Thus, overall SQL for the use case of the online bookstore is preferable over NoSQL.

However, it is important for the online bookstore to understand the considerations behind this choice of database management system because they will have to keep track of how this database develops and whether at some point there are enough reasons to switch to a NoSQL database.

The code for the schema creation of the SQL database using the forward engineering feature of MySQL and the ER-diagram shown in chapter 2 is uploaded with this assignment.

**5. Considerations on Regulatory Compliance:**

Regulatory requirements also need to be considered, especially because the database holds sensitive data like names, phone numbers and addresses. Ot thus must be ensured that the database and its usage complies to the General Data Protection Regulation (GDPR) defined by the European Parliament (2016).

For databases, unauthorized access to sensitive data shall be prohibited by defining roles and providing access only to the roles that need access to the database. Furthermore, as data will typically pass multiple individuals, e.g., while testing the database, it is important to use encryption or pseudonymization techniques to mask the data and still provide a high level of protection for sensitive data even in the event of a breach. Also, it must be ensured that it is known at all times who made which changes to the database. For that, a complete audit trail must be set up to comply with the GDPR. Then, alerts must be set up for caught breach attempts as these alerts allow the company to comply with notification obligations. Additionally, it must be made sure that at all times personally identifiable information are not only protected against breaches but also against accidental destruction, e.g., by configuration drifts. Finally, the logical database design explained the advantages of storing the database in the cloud. While these advantages remain true and I still advise to store the database in the cloud, this also means additional compliance considerations as some of the security obligations now fall on the cloud service provider. Here, it must be ensured that the servers on which the database are stored reside in the European Union or a country with equivalent data protection regulations like the United Kingdom as this makes sure that the regulatory obligation on the side of the service provider remains the same as on the company’s side. Should all these obligations be followed as explained, this ensures a high level of compliance which is seen as a starting point of the company’s security measures. [Yehuda, 2018 & Lyon, 2018]

**6. Conclusions and Recommendations:**

Following an initial logical database design, I have corrected some of the tables’ interrelations and created a schema in SQL for the physical database design. The relational data model was chosen for this design because it eliminates redundancy and introduces a reliable representation of the data that is easy to grasp and query. Because data is presented in a structured manner and query accuracy is considered to be more important than query speed, I choose to implement the database using SQL. Access policies dependent on different roles and permissions need to be set up and followed to ensure that everyone only has access to objects to which access is needed and compliance with the GDPR is achieved. Sensitive data must be encrypted or pseudonymized at all times to ensure compliance even in the event of a breach and it must be ensured that the company always has knowledge about who made which changes to the database via audit logging.

For future use, it is recommended for the online bookstore to keep the database up to date and recurringly check whether the options chosen in this proposal are still optimal. E.g., if the amount of data that runs through the database increases significantly it might make sense to switch to a NoSQL solution with a network approach data model. Thus, overall, this proposal of a physical database must be seen as a current status that must be kept up to date. The given argumentation presents a starting point for the customer and enough arguments for different solutions to re-evaluate the solution whenever the customer’s business situation changes.

**References:**

Agrawal, A. (2023) *SQL vs. NoSQL: Making the Right Choice for Your Database Needs.* Medium. Available from: <https://medium.com/cloud-native-daily/sql-vs-nosql-making-the-right-choice-for-your-database-needs-fc849d1b0e00#:~:text=Jun%2017,%202023.%20--%20When%20it%20comes%20to%20storing%20and> [Accessed 06. October 2024]

Datamation (2023) Relational Data Model 101: Key Components & Benefits. Available from: <https://www.datamation.com/big-data/relational-data-model/#:~:text=Disadvantages%20of%20a%20Relational%20Data%20Model%201%20Performance,...%203%20Limited%20support%20for%20unstructured%20data.%20> [Accessed 05. October 2024]

DbVisualizer (2023) *SQL vs NoSQL Databases: A Guide.* Medium. Available from: <https://medium.com/the-table-sql-and-devtalk/sql-vs-nosql-databases-a-guide-b0a4033dbabf#:~:text=In%20this%20article,%20we%20will%20look%20at%20the%20key%20differences> [Accessed 06. October 2024]

European Parliament (2016) General Data Protection Regulation. Available from: <https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX%3A32016R0679> [Accessed 13. October 2024]

Geeks for Geeks (2024) Difference Between Hierarchical, Network and Relational Data Model. Available from: <https://www.geeksforgeeks.org/difference-between-hierarchical-network-and-relational-data-model/> [Accessed 05. October 2024]

Geeks for Geeks (2021) Difference between identifying and non-identifying relationships. Available from: <https://www.geeksforgeeks.org/difference-between-identifying-and-non-identifying-relationships/> [Accessed 13. October 2024]

Iam Expat Media (n. d.) Phone numbers in Germany. Available from: <https://www.iamexpat.de/expat-info/communication/phone-number-germany#:~:text=According%20to%20the%20Bundesnetzagentur,%20the%20maximum%20length%20of%20a%20complete> [Accessed 05. October 2024]

IBM (n. d.) What is a relational database? Available from: <https://www.ibm.com/topics/relational-databases#:~:text=A%20relational%20database%20is%20a%20type%20of%20database%20that%20organizes> [Accessed 05. October 2024]

Lyon, L. (2018) *GDPR for the DBA.* Database Journal. Available from: <https://www.databasejournal.com/db2/gdpr-for-the-dba/> [Accessed 13. October 2024]

Wrath (2023) *Advantages and Disadvantages of Using a Relational Model in Database Management.* Medium. Available from: <https://medium.com/@praveenelaiyappan/advantages-and-disadvantages-of-using-a-relational-model-in-database-management-53b3c95a5389#:~:text=Disadvantages%20of%20the%20Relational%20Model:%20Performance%20and%20Scalability%20Challenges:%20Scaling> [Accessed 05. October 2024]

Yehuda, Y. (2018) 5 Ways to Make Your Database GDPR-Compliant. DZone. Available from: <https://dzone.com/articles/5-ways-to-make-your-database-gdpr-compliant#:~:text=GDPR%20in%20the%20Database%20%E2%80%93%20Does%20it%20Have,5.%20Prevent%20Configuration%20Drift%20and%20Data%20Loss%20> [Accessed 13. October 2024]