

**Data Technician**

|  |
| --- |
|  |

|  |
| --- |
| **Name: Maduka Chinemelum Ogwueleka** |
| **Course Date: 06/01/2025** |
|  |

**Table of contents**

[Day 1: Task 1 3](#_Toc106466110)

[Day 1: Task 2 3](#_Toc1673105856)

[Day 3: Task 1 4](#_Toc1682646377)

[Day 4: Task 1: Written 6](#_Toc135095808)

[Day 4: Task 2: SQL Practical 9](#_Toc1228298484)

[Course Notes 19](#_Toc631049525)

[Additional Information 19](#_Toc1612679949)

# Day 1: Task 1

Please research and complete the below questions relating to key concepts of databases.

|  |  |
| --- | --- |
| **What is a primary key?** | A primary key or a primary keyword, is a column in a relational database table that is distinctive for each record. It is a unique identifier, such as a driver's license number, telephone number with area code or vehicle identification number (VIN). A relational database must have only one primary key. |
| **How does this differ from a secondary key?** | A secondary key is any index that is not the primary key of a table. Every MySQL table has one primary key and can have many secondary keys. It is important to note that primary and secondary keys are still related to each other. |
| **How are primary and foreign keys related?** | Primary keys serve as unique identifiers for each row in a database table. Foreign keys link data in one table to the data in another table. A foreign key column in a table points to a column with unique values in another table (often the primary key column) to create a way of cross-referencing the two tables. |
| **Provide a real-world example of a one-to-one relationship** | For example, in a "Patient" table and a "MedicalRecord" table, each patient can only have one medical record and each medical record belongs to only one patient, creating a one-to-one relationship between them. |
| **Provide a real-world example of a one-to-many relationship** | For example, each customer can have many sales orders. |
| **Provide a real-world example of a many-to-many relationship** | For example, an Order table can contain orders placed by many customers (who are listed in the Customers table), and a customer may place more than one order. |

# Day 1: Task 2

Please research and complete the below questions relating to key concepts of databases.

|  |  |
| --- | --- |
| **What is the difference between a relational and non-relational database?** | Relational databases are structured and use tables, while non-relational databases are more flexible and can store a variety of data types. |
| **What type of data would benefit off the non-relational model?**  **Why?** | Non-relational databases, or NoSQL databases, offer the flexibility and scalability required for managing diverse and fast-growing datasets. They are designed to handle varying data structures, making them ideal for applications with rapid development needs and large-scale data requirements. |

# Day 3: Task 1

Please research the below ‘JOIN’ types, explain what they are and provide an example of the types of data it would be used on.

|  |  |
| --- | --- |
| **Self-join** | A Self Join is a type of a JOIN query used to compare rows within the same table. Unlike other SQL JOIN queries that join two or more tables, a self join joins a table to itself.  SQL self join is most typically used to query hierarchical data structures.  If you have a table representing connections between users in a social network, a self join can help identify common connections, mutual friends, or indirect relationships between users. |
| **Right join** | Returns all records from the right table, and the matched records from the left table. |
| **Full join** | A Full join returns all rows from both tables, whether or not there is a match between them. Unmatched rows are filled with NULLs where data is missing from one of the tables.   * The **FULL JOIN** or **FULL OUTER JOIN** in **SQL** is used to retrieve all rows from both tables involved in the **join**, regardless of whether there is a match between the rows in the two tables. |
| **Inner join** | Returns records that have matching values in both tables. |
| **Cross join** | **Returns all matching records from both tables** whether the other table matches or not. |
| **Left join** | Returns all rows from the left table and matching rows from the right table. |

# Day 4: Task 1: SQL Practical

In your groups, work together to answer the below questions. It may be of benefit if one of you shares your screen with the group and as a team answer / take screen shots from there.

**Setting up the database:**

1. **Visit this page [SQLBolt - Learn SQL - SQL Lesson 1: SELECT queries 101](https://sqlbolt.com/lesson/select_queries_introduction)**

**Write your query using the table given. Copy paste your query in the box**

**Once written query is correct, green tick will appear next to the task.**

**Remove written query to view original table.**

1. **Find the title of each film**

|  |
| --- |
| SELECT title FROM movies; |

1. **Find the director of each film**

|  |
| --- |
| SELECT director FROM movies; |

1. **Find the title and director of each film**

|  |
| --- |
| SELECT title, director FROM movies; |

1. **Find the title and year of each film**

|  |
| --- |
| SELECT title, year FROM movies; |

1. **Find all the information about each film**

|  |
| --- |
| SELECT \* FROM movies; |

1. **Find the movie with a row id of 6**

|  |
| --- |
| SELECT id, title FROM movies  WHERE id = 6; |

1. **Find the movies released in the years between 2000 and 2010**

|  |
| --- |
| SELECT title, year FROM movies  WHERE year BETWEEN 2000 AND 2010; |

1. **Find the movies not released in the years between 2000 and 2010**

|  |
| --- |
| SELECT title, year FROM movies  WHERE year < 2000 OR year > 2010; |

1. **Find the first 5 Pixar movies and their release year**

|  |
| --- |
| SELECT title, year FROM movies  WHERE year <= 2003; |

1. **Find all the Toy Story movies**

|  |
| --- |
| SELECT title, director FROM movies  WHERE title LIKE "Toy Story%"; |

1. **Find all the movies directed by John Lasseter**

|  |
| --- |
| SELECT title, director FROM movies  WHERE director = "John Lasseter"; |

1. **Find all the movies (and director) not directed by John Lasseter**

|  |
| --- |
| SELECT title, director FROM movies  WHERE director != "John Lasseter"; |

1. **Find all the WALL-\* movies**

|  |
| --- |
| SELECT \* FROM movies  WHERE title LIKE "WALL-\_"; |

1. **List all directors of Pixar movies (alphabetically), without duplicates**

|  |
| --- |
| SELECT DISTINCT director FROM movies  ORDER BY director ASC; |

1. **List the last four Pixar movies released (ordered from most recent to least)**

|  |
| --- |
| SELECT title, year FROM movies  ORDER BY year DESC  LIMIT 4; |

1. **List the first five Pixar movies sorted alphabetically**

|  |
| --- |
| SELECT title FROM movies  ORDER BY title ASC  LIMIT 5; |

1. **List the next five Pixar movies sorted alphabetically**

|  |
| --- |
| SELECT title FROM movies  ORDER BY title ASC  LIMIT 5 OFFSET 5; |

1. **Find the domestic and international sales for each movie**

|  |
| --- |
| SELECT title, domestic\_sales, international\_sales FROM movies JOIN boxoffice ON movies.id = boxoffice.movie\_id; |

1. **Show the sales numbers for each movie that did better internationally rather than domestically**

|  |
| --- |
| SELECT title, domestic\_sales, international\_sales FROM movies JOIN boxoffice ON movies.id = boxoffice.movie\_id WHERE international\_sales > domestic\_sales; |

1. **List all the movies by their ratings in descending order**

|  |
| --- |
| SELECT title, rating  FROM movies  JOIN boxoffice  ON movies.id = boxoffice.movie\_id  ORDER BY rating DESC; |

|  |
| --- |
| **Course Notes** |

It is recommended to take notes from the course, use the space below to do so, or use the revision guide shared with the class:

|  |
| --- |
| **Additional Information** |

We have included a range of additional links to further resources and information that you may find useful, these can be found within your revision guide.

**END OF WORKBOOK**

**Please check through your work thoroughly before submitting and update the table of contents if required.**

**Please send your completed work booklet to your trainer.**