

**Data Technician**

|  |
| --- |
|  |

|  |
| --- |
| Name: |
| Course Date: |
|  |

**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 is the column or columns that contain values that uniquely identify each row in a table |
| How does this differ from a secondary key? | Primary keys are defined on tables objects only. In SQL, Table extension objects inherit the primary key of the table object they extend, so any key that define in a table extension object is considered a secondary key. |
| 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. |
| Provide a real-world example of a one-to-one relationship | One to One relationship:  One Employee has one Contract.  One driver has one car. |
| Provide a real-world example of a one-to-many relationship | Parent child relationship  Teachers and students' relationship  Book authors and their books |
| Provide a real-world example of a many-to-many relationship | Student and courses.  Actor and films- An actor can act in multiple movies. |

# 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? | Non-relational databases, often referred to as NoSQL databases, store data in a non-tabular form, making them more flexible than traditional relational databases. Unlike relational databases that use tables with rows and columns, non-relational databases use various data models such as documents, key-value pairs, graphs, and wide-column stores. |
| What type of data would benefit off the non-relational model?  Why? | A [non-relational database](https://www.geeksforgeeks.org/non-relational-databases-and-their-types/) is a type of database that does not rely on the traditional tabular structure of rows and columns found in relational databases. Instead, it uses flexible data models such as key-value pairs, documents, graphs, and wide-column stores.  This flexibility allows non-relational databases to manage unstructured, semi-structured, and structured data efficiently. They were designed when data was expected to be partitioned across multiple machines to scale, in contrast to relational databases, which assumed the data would stay on a single machine |

# 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 | **What it is**: A SELF JOIN is used to join a table to itself. This means you are comparing rows within the same table. To do this, you must use table aliases to treat the table as two separate entities within the query. The join condition compares columns within that single table.  **Example Data Usage**: Imagine an Employees table that includes EmployeeID, EmployeeName, and ManagerID. The ManagerID refers back to the EmployeeID of the employee's manager (who is also in the same table). A SELF JOIN could be used to list each employee alongside the name of their manager by joining the Employees table to itself on the condition Employee.ManagerID = Manager.EmployeeID (where Employee and Manager are aliases for the same Employees table). |
| Right join | **What it is**: A RIGHT JOIN returns all rows from the right table and the matched rows from the left table. If there's no match for a row from the right table in the left table, the result will include the row from the right table, but with NULL values for the columns selected from the left table.  **Example Data Usage**: With Customers (left) and Orders (right), a RIGHT JOIN would return all orders. If an order has a corresponding customer, the customer details are shown. If an order somehow exists without a matching customer that order would still appear, but the customer columns would be NULL. |
| Full join | **What it is**: A FULL JOIN combines the results of both LEFT JOIN and RIGHT JOIN. It returns all rows from both the left and the right tables. If there's a match between tables based on the join condition, the corresponding data from both tables is returned. If a row in the left table has no match in the right table, it's included with NULLs for the right table's columns. Conversely, if a row in the right table has no match in the left table, it's included with NULLs for the left table's columns. This is useful when you need a complete picture of all records in both tables, regardless of whether they have a match in the other.  **Example Data Usage**: Again, using Customers and Orders. A FULL JOIN would show:   * Customers who have placed orders (matched data from both tables). * Customers who have not placed orders (Customer data + NULLs for Order data). * Orders that do not have a matching customer (Order data + NULLs for Customer data). |
| Inner join | **What it is**: An INNER JOIN returns only the rows where the join condition is met in both tables. If a row in one table doesn't have a matching row in the other table (based on the specified join column), it is excluded from the result set. Think of it as the intersection of the two tables based on the join criteria.  **Example Data Usage**: Commonly used to link tables with a primary key-foreign key relationship. For instance, you might have a Customers table (with CustomerID as the primary key) and an Orders table (with CustomerID as a foreign key). An INNER JOIN on CustomerID would retrieve a list of customers who have placed orders, combining customer details with their corresponding order information. Customers without orders would not appear. |
| Cross join | **What it is**: A CROSS JOIN produces the Cartesian product of the two tables involved. This means it combines every row from the first table with every row from the second table. If the first table has N rows and the second table has M rows, the result will have N \* M rows. Useful when you need to generate all possible combinations between items in two distinct lists.  **Example Data Usage**: If you have a Colours table (Red, Blue) and a Sizes table (Small, Medium, Large), a CROSS JOIN would produce all possible colour/size combinations: (Red, Small), (Red, Medium), (Red, Large), (Blue, Small), (Blue, Medium), (Blue, Large). |
| Left join | **What it is**: A LEFT JOIN returns all rows from the left table (the first table mentioned in the JOIN clause) and the matched rows from the right table (the second table mentioned). If there is no match for a row from the left table in the right table, the result will still include the row from the left table, but with NULL values for all columns selected from the right table.  **Example Data Usage**: Using the Customers and Orders tables again, a LEFT JOIN with Customers as the left table would return all customers. If a customer has placed orders, their details will be shown alongside the order information. If a customer has not placed any orders, their details will still be shown, but the columns for order information will contain NULL. |

# Day 4: Task 1: Written

In your groups, discuss and complete the below activity. You can either nominate one writer or split the elements between you. Everyone however must have the completed work below:

*Imagine you have been hired by a small retail business that wants to streamline its operations by creating a new database system. This database will be used to manage inventory, sales, and customer information. The business is a small corner shop that sells a range of groceries and domestic products. It might help to picture your local convenience store and think of what they sell. They also have a loyalty program, which you will need to consider when deciding what tables to create.*

*Write a 500-word essay explaining the steps you would take to set up and create this database. Your essay should cover the following points:*

1. ***Understanding the Business Requirements****:*
   1. *What kind of data will the database need to store?*
   2. *Who will be the users of the database, and what will they need to accomplish?*
2. ***Designing the Database Schema****:*
   1. *How would you structure the database tables to efficiently store inventory, sales, and customer information?*
   2. *What relationships between tables are necessary (e.g., how sales relate to inventory and customers)?*
3. ***Implementing the Database****:*
   1. *What SQL commands would you use to create the database and its tables?*
   2. *Provide examples of SQL statements for creating tables and defining relationships between them.*
4. ***Populating the Database****:*
   1. *How would you input initial data into the database? Give examples of SQL INSERT statements.*
5. ***Maintaining the Database****:*
   1. *What measures would you take to ensure the database remains accurate and up to date?*
   2. *How would you handle backups and data security?*

*Your essay should include specific examples of SQL commands and explain why each step is necessary for creating a functional and efficient database for the retail business.*

|  |  |
| --- | --- |
| Please write your 500-word essay here | Creating an Efficient Database System for a Small Retail Business  Implementing a database system is a crucial step for streamlining operations in a small retail business, like a local convenience store. This system will manage inventory, track sales, and maintain customer information, including a loyalty program, ultimately leading to greater efficiency and better decision-making. The process involves several key stages, from understanding the business needs to ongoing maintenance.  1. Understanding the Business Requirements  The first step is a thorough analysis of the business's needs. We must identify precisely what data needs storing. For a corner shop selling groceries and domestic products, this includes:   * Inventory Data: Product name, description (e.g., size, brand), category (e.g., Dairy, Cleaning Supplies, Snacks), supplier details, cost price, selling price, and current stock quantity. * Sales Data: Transaction details (date, time, unique sale ID), items sold in each transaction, quantity of each item, price at the time of sale, total transaction amount, and payment method. * Customer Data: Customer name, contact details (phone number or email), and crucially, a loyalty program identifier and their current points balance.   We also need to identify the users and their tasks. The shop owner/manager will need broad access: viewing sales reports, managing inventory levels (adding stock, updating prices), adding new products, managing customer accounts, and overseeing the loyalty scheme. Till operators (cashiers) will primarily need to record sales transactions quickly, look up product prices, add customers to the loyalty scheme, and update points.  2. Designing the Database Schema  Based on the requirements, we design the structure (schema). We'll use separate tables for distinct entities to avoid redundancy and ensure data integrity (a concept called normalization). Key tables would include:   * Products: Stores inventory details (ProductID [Primary Key], Name, Category, Price, StockLevel). * Customers: Holds customer information (CustomerID [Primary Key], FirstName, LastName, ContactInfo, LoyaltyPoints). * Sales: Records overall sales transactions (SaleID [Primary Key], CustomerID [Foreign Key linking to Customers], SaleDate, TotalAmount). * SaleItems: A crucial linking table detailing *which* products were in *which* sale (SaleItemID [Primary Key], SaleID [Foreign Key linking to Sales], ProductID [Foreign Key linking to Products], QuantitySold, PriceAtSale). This handles the scenario where one sale includes multiple products.   Relationships are vital: A customer can have multiple sales (Customers to Sales - one-to-many). A sale can involve multiple products, and a product can be part of multiple sales (Sales to Products - many-to-many, managed via the SaleItems table). These relationships are enforced using primary and foreign keys.  3. Implementing the Database  With the design finalised, we implement it using SQL (Structured Query Language). First, we create the database itself (the command varies slightly by system, e.g., CREATE DATABASE CornerShopDB;). Then, we create the tables:  SQL  CREATE TABLE Customers (  CustomerID INT PRIMARY KEY AUTO\_INCREMENT, -- Or SERIAL depending on SQL flavour  FirstName VARCHAR(50),  LastName VARCHAR(50),  ContactInfo VARCHAR(100) UNIQUE, -- Ensures unique contact  LoyaltyPoints INT DEFAULT 0  );  CREATE TABLE Products (  ProductID INT PRIMARY KEY AUTO\_INCREMENT,  Name VARCHAR(100) NOT NULL,  Category VARCHAR(50),  Price DECIMAL(10, 2) NOT NULL,  StockLevel INT DEFAULT 0  );  CREATE TABLE Sales (  SaleID INT PRIMARY KEY AUTO\_INCREMENT,  CustomerID INT, -- Allow NULL if sale is not linked to loyalty customer  SaleDate DATETIME DEFAULT CURRENT\_TIMESTAMP,  TotalAmount DECIMAL(10, 2),  FOREIGN KEY (CustomerID) REFERENCES Customers(CustomerID)  );  CREATE TABLE SaleItems (  SaleItemID INT PRIMARY KEY AUTO\_INCREMENT,  SaleID INT NOT NULL,  ProductID INT NOT NULL,  QuantitySold INT NOT NULL,  PriceAtSale DECIMAL(10, 2) NOT NULL, -- Price when sold  FOREIGN KEY (SaleID) REFERENCES Sales(SaleID),  FOREIGN KEY (ProductID) REFERENCES Products(ProductID)  );  These commands define table structures, data types (like INT, VARCHAR, DECIMAL, DATETIME), constraints (PRIMARY KEY, FOREIGN KEY, NOT NULL, UNIQUE, DEFAULT), and establish the relationships.  4. Populating the Database  Once structured, the database needs initial data. This involves a physical stocktake for inventory and compiling the existing customer list. SQL INSERT statements are used:  SQL  -- Example: Adding a product  INSERT INTO Products (Name, Category, Price, StockLevel)  VALUES ('Digestive Biscuits', 'Biscuits', 1.20, 50);  -- Example: Adding a customer  INSERT INTO Customers (FirstName, LastName, ContactInfo)  VALUES ('Jane', 'Doe', 'jane.doe@email.com');  Bulk import tools or scripts are often used for larger initial data loads.  5. Maintaining the Database  Ongoing maintenance ensures data accuracy and system reliability. Key practices include:   * Accuracy: Implement procedures for regular stock updates (daily or weekly counts for key items), promptly updating customer details, and potentially adding data validation checks within the application used to access the database. * Backups: Schedule regular automated backups (e.g., daily). Store backups securely, ideally off-site or using a cloud service, to protect against hardware failure, accidental deletion, or disasters. Periodically test restoring from a backup. * Security: Use strong passwords for database access. Create different user accounts with permissions tailored to roles (e.g., cashier cannot delete sales records). Keep the database software updated.   By following these steps – understanding requirements, careful design, structured implementation, accurate population, and diligent maintenance – the corner shop can establish a robust database system. This foundation will streamline inventory management, speed up checkouts, enhance the loyalty program, and provide valuable insights into sales patterns, contributing significantly to the business's success. |

# Day 4: Task 2: 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. **Download world\_db(1)**
2. **Follow each step to create your database**

**For each question I would like to see both the syntax used and the output.**

1. **Count Cities in USA:** *Scenario:* You've been tasked with conducting a demographic analysis of cities in the United States. Your first step is to determine the total number of cities within the country to provide a baseline for further analysis.

|  |
| --- |
|  |

1. **Country with Highest Life Expectancy:** *Scenario:* As part of a global health initiative, you've been assigned to identify the country with the highest life expectancy. This information will be crucial for prioritising healthcare resources and interventions.

|  |
| --- |
|  |

1. **"New Year Promotion: Featuring Cities with 'New :** *Scenario:* In anticipation of the upcoming New Year, your travel agency is gearing up for a special promotion featuring cities with names including the word 'New'. You're tasked with swiftly compiling a list of all cities from around the world. This curated selection will be essential in creating promotional materials and enticing travellers with exciting destinations to kick off the New Year in style.

|  |
| --- |
|  |

1. **Display Columns with Limit (First 10 Rows):** *Scenario:* You're tasked with providing a brief overview of the most populous cities in the world. To keep the report concise, you're instructed to list only the first 10 cities by population from the database.

|  |
| --- |
|  |

1. **Cities with Population Larger than 2,000,000:** *Scenario:* A real estate developer is interested in cities with substantial population sizes for potential investment opportunities. You're tasked with identifying cities from the database with populations exceeding 2 million to focus their research efforts.

|  |
| --- |
|  |

1. **Cities Beginning with 'Be' Prefix:** *Scenario:* A travel blogger is planning a series of articles featuring cities with unique names. You're tasked with compiling a list of cities from the database that start with the prefix 'Be' to assist in the blogger's content creation process.

|  |
| --- |
|  |

1. **Cities with Population Between 500,000-1,000,000:** *Scenario:* An urban planning committee needs to identify mid-sized cities suitable for infrastructure development projects. You're tasked with identifying cities with populations ranging between 500,000 and 1 million to inform their decision-making process.

|  |
| --- |
|  |

1. **Display Cities Sorted by Name in Ascending Order:** *Scenario:* A geography teacher is preparing a lesson on alphabetical order using city names. You're tasked with providing a sorted list of cities from the database in ascending order by name to support the lesson plan.

|  |
| --- |
|  |

1. **Most Populated City:** *Scenario:* A real estate investment firm is interested in cities with significant population densities for potential development projects. You're tasked with identifying the most populated city from the database to guide their investment decisions and strategic planning.

|  |
| --- |
|  |

1. **City Name Frequency Analysis: Supporting Geography Education** *Scenario*: In a geography class, students are learning about the distribution of city names around the world. The teacher, in preparation for a lesson on city name frequencies, wants to provide students with a list of unique city names sorted alphabetically, along with their respective counts of occurrences in the database. You're tasked with this sorted list to support the geography teacher.

|  |
| --- |
|  |

1. **City with the Lowest Population:** *Scenario:* A census bureau is conducting an analysis of urban population distribution. You're tasked with identifying the city with the lowest population from the database to provide a comprehensive overview of demographic trends.

|  |
| --- |
|  |

1. **Country with Largest Population:** *Scenario:* A global economic research institute requires data on countries with the largest populations for a comprehensive analysis. You're tasked with identifying the country with the highest population from the database to provide valuable insights into demographic trends.

|  |
| --- |
|  |

1. **Capital of Spain:** *Scenario:* A travel agency is organising tours across Europe and needs accurate information on capital cities. You're tasked with identifying the capital of Spain from the database to ensure itinerary accuracy and provide travellers with essential destination information.

|  |
| --- |
|  |

1. **Country with Highest Life Expectancy:** *Scenario:* A healthcare foundation is conducting research on global health indicators. You're tasked with identifying the country with the highest life expectancy from the database to inform their efforts in improving healthcare systems and policies.

|  |
| --- |
|  |

1. **Cities in Europe:** *Scenario:* A European cultural exchange program is seeking to connect students with cities across the continent. You're tasked with compiling a list of cities located in Europe from the database to facilitate program planning and student engagement.

|  |
| --- |
|  |

1. **Average Population by Country:** *Scenario:* A demographic research team is conducting a comparative analysis of population distributions across countries. You're tasked with calculating the average population for each country from the database to provide valuable insights into global population trends.

|  |
| --- |
|  |

1. **Capital Cities Population Comparison:** *Scenario:* A statistical analysis firm is examining population distributions between capital cities worldwide. You're tasked with comparing the populations of capital cities from different countries to identify trends and patterns in urban demographics.

|  |
| --- |
|  |

1. **Countries with Low Population Density:** *Scenario:* An agricultural research institute is studying countries with low population densities for potential agricultural development projects. You're tasked with identifying countries with sparse populations from the database to support the institute's research efforts.

|  |
| --- |
|  |

1. **Cities with High GDP per Capita:** *Scenario:* An economic consulting firm is analysing cities with high GDP per capita for investment opportunities. You're tasked with identifying cities with above-average GDP per capita from the database to assist the firm in identifying potential investment destinations.

|  |
| --- |
|  |

1. **Display Columns with Limit (Rows 31-40):** *Scenario:* A market research firm requires detailed information on cities beyond the top rankings for a comprehensive analysis. You're tasked with providing data on cities ranked between 31st and 40th by population to ensure a thorough understanding of urban demographics.

|  |
| --- |
|  |

|  |
| --- |
| **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.**