

**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 a unique identifier for a record in a database table. It ensures each record can be accessed and prevents duplicate entries and allows for efficient data retrieval. |
| How does this differ from a secondary key? | A secondary key is another unique identifier in a database table, it isn’t necessarily uniquely identifying every single record. |
| How are primary and foreign keys related? | A foreign key is a field in one of the tables that uniquely identifies a row in another, which establishes a relationship between the two tables.  The foreign key links to the primary key in the related table, and this relationship allows the data to be connected across tables. |
| Provide a real-world example of a one-to-one relationship | A one-to-one relationship could be seen in a scenario where each person has a single passport, and each passport is only associated with only one individual. |
| Provide a real-world example of a one-to-many relationship | A one-to-many relationship exists between a parent company and its subsidiaries; one parent company can have multiple subsidiaries, but each subsidiary has only one parent company. |
| Provide a real-world example of a many-to-many relationship | A many-to-many relationship can be seen in a university setting where students enrol in multiple courses, and each course has multiple students. |

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? | A relational database organises data into tables with rows columns. It uses a structured schema and SQL queries for data manipulation  Whereas, non-relational databases, also called NoSQL databases, store data in formats like documents, more visual formats like graphs, or key-value pairs, often without a fixed schema. |
| What type of data would benefit off the non-relational model?  Why? | Non-relational databases are very much suited for unstructured data, such as social media content or product catalogues  Non-relational models are more flexible, allowing adaptability and quick adjustments, since the data doesn't fit neatly into tables, so NoSQL databases allow for enabling adapting to evolving 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 when it joins a table with itself. It is used for hierarchical data, for example, employees within an organisation, and the ‘Employee’ table can join to itself to link an employee with their manager. |
| Right join | For example, customers (left table) and orders (right table) would be the tables. A right join would show every customer (even those who didn't buy anything) but include only the orders for the customers that bought something. If a customer didn’t buy anything, their name will still show up, but the order information will be blank. |
| Full join | A full join combines everything from both tables, even if there’s no match. For example, if there's a table for inventory (products) and another for sales (products sold), all products, whether sold or not, and all sales, even if the product isn't listed in inventory will be shown |
| Inner join | An inner join shows only the matching records from both tables. If you have a products table and an orders table, you’ll see only the products that have been ordered (no extras from either side). |
| Cross join | A cross join combines every record from one table with every record from the other. If you have a table of fruits and a table of colours, a cross join will list every possible fruit-colour combination (e.g., apple-red, apple-yellow, banana-red, banana-yellow). |
| Left join | A left join takes everything from the first table ("left one") and adds matching records from the second table ("right one"). A left join would show every customer (even those who didn't buy anything) but include only the orders for the customers that bought something. If a customer didn’t buy anything, their name will still show up, but the order information will be blank. |

# 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 | 1. Building a Corner Shop Database:  Setting up a database for a small grocery shop means figuring out what the business needs, designing a clear structure, setting it up with SQL, adding starting data, and keeping it running smoothly.  Figuring Out What's needed:  Data to Track: The database should handle inventory (products, stock levels, suppliers), sales (transactions, payment methods, dates), and customers (contact details, loyalty points, and purchase history).  How It’ll Be Used: Staff will use it to check stock, log sales, and access customer details. Managers will analyze sales trends and run the loyalty program. Security features need to block unauthorized access.  2. Designing the Database Schema  Table Structure:  Products Table: Contains product\_id, name, category, price, supplier\_id, and stock\_quantity.  Suppliers Table: Stores supplier\_id, name, and contact\_info.  Sales Table: Tracks sale\_id, date, total\_amount, and customer\_id.  Sales\_Details Table: Stores line-item details like sale\_id, product\_id, quantity\_sold, and total\_price.  Customers Table: Includes customer\_id, name, email, phone\_number, and loyalty\_points.  Relationships:  Products and sales are linked through the Sales\_Details table (many-to-many relationship).  Sales are tied to customers (one-to-many relationship).  Products are connected to suppliers (many-to-one relationship).  3. Setting Up the Database  -- Create the database and select it  -- Create the database and select it  CREATE DATABASE CornerShopDB;  USE CornerShopDB;  -- Products table: Tracks inventory details  CREATE TABLE Products (  ProductID INT AUTO\_INCREMENT PRIMARY KEY,  Name VARCHAR(100) NOT NULL,  Category VARCHAR(50),  Price DECIMAL(10, 2) NOT NULL,  SupplierID INT,  StockQuantity INT DEFAULT 0,  FOREIGN KEY (SupplierID) REFERENCES Suppliers(SupplierID)  );  -- Suppliers table: Stores supplier info  CREATE TABLE Suppliers (  SupplierID INT AUTO\_INCREMENT PRIMARY KEY,  Name VARCHAR(100) NOT NULL,  ContactInfo VARCHAR(255)  );  -- Customers table: Handles customer details and loyalty points  CREATE TABLE Customers (  CustomerID INT AUTO\_INCREMENT PRIMARY KEY,  Name VARCHAR(100),  Email VARCHAR(100) UNIQUE,e)  PhoneNumber VARCHAR(15),  LoyaltyPoints INT DEFAULT 0  );  -- Sales table:  CREATE TABLE Sales (  SaleID INT AUTO\_INCREMENT PRIMARY KEY,  Date DATETIME DEFAULT CURRENT\_TIMESTAMP,  TotalAmount DECIMAL(10, 2),  CustomerID INT,  FOREIGN KEY (CustomerID) REFERENCES Customers(CustomerID)  );  -- SalesDetails table:  CREATE TABLE SalesDetails (  SaleID INT,  ProductID INT,  QuantitySold INT,  TotalPrice DECIMAL(10, 2),  PRIMARY KEY (SaleID, ProductID),  FOREIGN KEY (SaleID) REFERENCES Sales(SaleID),  FOREIGN KEY (ProductID) REFERENCES Products(ProductID)  );  4. Populating the Database  -- Insert products  INSERT INTO Products (name, category, price, supplier\_id, stock\_quantity)  VALUES ('Milk', 'Dairy', 1.50, 1, 100),  ('Bread', 'Bakery', 1.00, 2, 50);  -- Insert customers  INSERT INTO Customers (name, email, phone\_number, loyalty\_points)  VALUES ('John Doe', 'john@example.com', '123-456-7890', 50);  -- Insert sales  INSERT INTO Sales (total\_amount, customer\_id)  VALUES (15.00, 1);  -- Insert sales details  INSERT INTO Sales\_Details (sale\_id, product\_id, quantity\_sold, total\_price)  VALUES (1, 1, 2, 3.00), (1, 2, 3, 9.00);  5. Maintaining the Database  Data Accuracy: Implement constraints like NOT NULL, CHECK, and triggers to validate data integrity. For example, ensure stock\_quantity cannot be negative.  Backups and Security:  Schedule regular backups using tools like MySQL Workbench or scripts.  Secure data with user roles, permissions (GRANT and REVOKE commands), and encrypted connections.  Use audit logs to track changes and prevent unauthorized modifications. |

# 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)** [**here**](https://justit831-my.sharepoint.com/:u:/g/personal/danpe_justit_co_uk/Ef6vAaaYVi5FhHqKGxqnn60B9g2khoYekEIO3Y7J00UcJQ?e=pv9NNE)
2. **Follow each step to create your database** [**here**](https://justit831-my.sharepoint.com/:b:/g/personal/danpe_justit_co_uk/EdeCKl2Sas1Hl7u9amDy0fIB9jGVCKxSR0u2-lFOvS5rXw?e=xKv1U7)

**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.

|  |
| --- |
| SELECT COUNT(\*) AS TotalCities  FROM city  WHERE CountryCode = 'USA'; |

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.

|  |
| --- |
| SELECT Name, LifeExpectancy  FROM country  ORDER BY LifeExpectancy DESC  LIMIT 1; |

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.

|  |
| --- |
| SELECT Name  FROM city  WHERE Name LIKE '%New%'; |

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.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| SELECT Name, Population  FROM city  ORDER BY Population DESC  LIMIT 10;  OUTPUT:   |  | | --- | | {'Name': 'London', 'Population': 7285000} | | {'Name': 'Sydney', 'Population': 3276207} | | {'Name': 'Buenos Aires', 'Population': 2982146} | | {'Name': 'Madrid', 'Population': 2879052} | | {'Name': 'Kabul', 'Population': 1780000} | | {'Name': 'Barcelona', 'Population': 1503451} | | {'Name': 'Qandahar', 'Population': 237500} | | {'Name': 'Herat', 'Population': 186800} | |

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.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| SELECT Name, Population  FROM city  WHERE Population > 2000000;  OUTPUT:   |  | | --- | | {'Name': 'Buenos Aires', 'Population': 2982146} | | {'Name': 'London', 'Population': 7285000} | | {'Name': 'Madrid', 'Population': 2879052} | | {'Name': 'Sydney', 'Population': 3276207} | |  | |

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.

|  |
| --- |
| SELECT Name  FROM city  WHERE Name LIKE 'Be%';  OUTPUT:  +------------------+----------------+------------+  | Name | CountryCode | Population |  +------------------+----------------+------------+  | Beijing | CHN | 7472000 |  | Belize City | BLZ | 55810 |  | Belmopan | BLZ | 7105 |  | Belfast | GBR | 287500 |  | Belo Horizonte | BRA | 2139125 |  +------------------+----------------+------------+ |

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.

|  |
| --- |
| SELECT Name, Population  FROM city  WHERE Population BETWEEN 500000 AND 1000000;  OUTPUT:  +------+----------------+------------+  | Name | CountryCode | Population |  +------+----------------+------------+  | None | None | None |  +------+----------------+------------+ |

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.

|  |
| --- |
| SELECT Name  FROM city  ORDER BY Name ASC; |

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.

|  |
| --- |
| SELECT Name, Population  FROM city  ORDER BY Population DESC  LIMIT 1; |

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.

|  |
| --- |
| SELECT Name, COUNT(\*) AS Frequency  FROM city  GROUP BY Name  ORDER BY Name ASC; |

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.

|  |
| --- |
| SELECT Name, Population  FROM city  ORDER BY Population ASC  LIMIT 1; |

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.

|  |
| --- |
| SELECT Name, Population  FROM country  ORDER BY Population DESC  LIMIT 1; |

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.

|  |
| --- |
| SELECT Name  FROM city  WHERE CountryCode = 'ESP' AND IsCapital = 1; |

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.

|  |
| --- |
| SELECT Name, LifeExpectancy  FROM country  ORDER BY LifeExpectancy DESC  LIMIT 1; |

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.

|  |
| --- |
| SELECT Name  FROM city  WHERE CountryCode IN (  SELECT Code  FROM country  WHERE Continent = 'Europe'  ); |

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.

|  |
| --- |
| SELECT CountryCode, AVG(Population) AS AvgPopulation  FROM city  GROUP BY CountryCode; |

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.

|  |
| --- |
| SELECT Name, Population  FROM city  WHERE IsCapital = 1  ORDER BY Population DESC; |

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.

|  |
| --- |
| SELECT Name, Population / SurfaceArea AS PopulationDensity  FROM country  WHERE (Population / SurfaceArea) < 50; |

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.

|  |
| --- |
| SELECT Name, (GDP / Population) AS GDPPerCapita  FROM city  WHERE (GDP / Population) > (SELECT AVG(GDP / Population) FROM city); |

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.

|  |
| --- |
| SELECT Name, Population  FROM city  ORDER BY Population DESC  LIMIT 10 OFFSET 30; |

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