

**Data Technician**

|  |
| --- |
|  |

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

**Table of contents**

[Day 1: Task 1 3](#_Toc193099469)

[Day 1: Task 2 4](#_Toc193099470)

[Day 3: Task 1 4](#_Toc193099471)

[Day 4: Task 1: SQL Practical 6](#_Toc193099472)

[Day 4: Task 2: Written (Optional) 14](#_Toc193099473)

[**Course Notes** 18](#_Toc193099474)

[**Additional Information** 19](#_Toc193099475)

# 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 column in a database table that uniquely identifies each row in that table. It ensures that no two rows have the same value in the primary key column and prevents NULL values. It is used to link tables in relational database. |
| How does this differ from a secondary key? | A secondary key is a column in a database table that is not the primary key but can still be used to identify records uniquely or for searching purposes. Unlike a primary key, a secondary key can have duplicate values and used for indexing.  Key feature of primary keys is   * Uniqueness: - Each value in the primary key column must be unique. * Not Null: - A primary key cannot have NULL values. * One per Table: - Each table can have only one primary key. * Used for Relationships: - It is often used to link tables in a relational database. |
| How are primary and foreign keys related? | A primary key and a foreign key are related in a way that helps establish and enforce relationships between tables in a relational database.  Primary Key: - Uniquely identifies each record in a table and ensures data integrity within its own table.  Foreign Key: a column in one table that references the primary key of another table. It's used to establishes a relationship between two tables and ensures referential integrity. |
| Provide a real-world example of a one-to-one relationship | A one-to-one relationship means that each record in Table A corresponds to exactly one record in Table B, and vice versa.  Examples: - Citizen and National ID, Person and Passport, Person and NHS Number, Car and Vehicle registration, Person and Driving license Number. |
| Provide a real-world example of a one-to-many relationship | A one-to-many relationship means that one record in Table A can be related to multiple records in Table B, but each record in Table B is related to only one record in Table A.  Examples: - Department and Employees, Company and Branches of company, Customer and Orders. |
| Provide a real-world example of a many-to-many relationship | A many-to-many relationship occurs when multiple records in Table A can be associated with multiple records in Table B and vice versa.  Examples: - Students and courses, Doctors and Patients, Actors and Films. |

# 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 is structured and organizes data into tables with rows and columns, enforcing relationships between them through keys. It follows a fixed schema, meaning data must adhere to a predefined structure. They typically scale vertically by adding more power to a single server and use SQL (Structured Query Language) for querying. Examples include MySQL, PostgreSQL, SQL Server, and Oracle, making them ideal for structured data like financial transactions, customer management systems, and enterprise applications.  A non-relational database (NoSQL) is more flexible, allowing data to be stored in various formats such as documents, key-value pairs, graphs, or wide-column stores. It does not require a fixed schema, making it well-suited for handling large, unstructured, or semi-structured data. They scale horizontally by distributing data across multiple servers. Examples include MongoDB (document-based), Cosmos DB, Couch DB, Amazon Dynamo DB, which are commonly used in big data applications, real-time analytics, and content management systems. |
| What type of data would benefit off the non-relational model?  Why? | The non-relational (NoSQL) model is ideal for handling large, complex, and rapidly changing data that does not fit neatly into structured tables. It benefits unstructured or semi-structured data, such as social media posts, multimedia content, and logs, because document-based databases like MongoDB allow flexible schemas that adapt to evolving data structures. NoSQL is also well-suited for big data and real-time analytics, such as IoT sensor data and stock market trends. |

# 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 join where a table is joined with itself. It is useful when working with hierarchical or relational data within the same table. The table is treated as two different tables using aliases.  Example Use Case: A company’s employee database where each employee has a manager (who is also an employee). |
| Right join | A right join returns all records from the right table and only matching records from the left table. If there is no match, NULL is returned for the left table’s columns.  Example Use Case: Retrieving all orders and their corresponding customers, including orders with no associated customer records (e.g., deleted customers). |
| Full join | A full join returns all records from both tables, filling in NULLs where there is no match. It combines left join and right join results.  Example Use Case: A customer and orders database where we want to see all customers and all orders, even if a customer has no orders or an order has no associated customer. |
| Inner join | An inner join returns only the records that have matching values in both tables. It excludes unmatched rows from both tables.  Example Use Case: Finding only customers who have placed an order. |
| Cross join | A cross join returns the Cartesian product of both tables, meaning every row in the first table is paired with every row in the second table.  Example Use Case: Generating all possible product-customer combinations for marketing analysis. |
| Left join | A left join returns all records from the left table and only matching records from the right table. If no match is found, NULL is returned for the right table’s columns.  Example Use Case: Retrieving all customers and their orders, including those who have not placed an order. |

# 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. **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.

|  |
| --- |
| -- 1 count cities in 'USA'  select count(\*) as number\_of\_Cities from city where city.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.

|  |
| --- |
| -- 2 Country with Highest Life Expectancy  select code, 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 city.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 city.name, Population  from city  order by Population desc  limit 10; |

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 city.name, Population  from city  where Population >= 2000000  order by Population asc; |

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 city.name  from city  where name like 'Be%'; |

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 city.name, Population  from city  where Population between 500000 and 1000000  order by Population asc; |

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 city.name, count(city.name) AS city\_count  FROM city  GROUP BY city.name  ORDER BY city.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 city.CountryCode, city.name, city.District, city.Population  from city  join country  on City.countryCode = country.Code  where country.Name like 'Spain'; |

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 city.countrycode, city.name, continent  from city  join country  on city.countrycode = country.code  where country.continent like 'Europe'  group by city.countrycode, city.name, continent; |

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 name, AVG(country.population) AS average\_population  FROM country  group by name, localname, population  ORDER BY average\_population asc; |

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 country.code, country.name as countriesName, city.name as citiesName, avg(city.population) as Averagepopulation  FROM country  JOIN city  ON country.code = City.countrycode  group by country.code, countriesName, citiesName  ORDER BY AveragePopulation 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 code, name, Population  from country  where population > 0  order by Population asc; |

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 country.Code, city.name as citiesName, country.name AS countriesName, country.population, GNP,  (GNP / NULLIF(city.population, 0)) AS GDP, ((GNP / NULLIF(city.population, 0))/ country.Capital) as GDPCapital  FROM country  join city  on country.code = city.countrycode  WHERE (GNP / NULLIF(City.population, 0)) >  (SELECT AVG(GNP / NULLIF(country.population, 0))  FROM country)  group by country.code, citiesName, countriesName, country.Population, GNP, GDP, GDPCapital  ORDER BY GDP desc; |

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 \*  from city  order by population desc  limit 10 offset 30; |

# Day 4: Task 2: Written (Optional)

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 |  |

|  |
| --- |
| **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:

* Basics of Databases and SQL
* Relational and non – relational database
* Database structure
* Types of relationships
* Database management systems
* Purpose of SQL

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