

**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? | A Primary Key uniquely identifies each record in a table, while a Secondary Key is an additional key used for searching or indexing but does not enforce uniqueness. |
| How are primary and foreign keys related? | A Primary Key uniquely identifies each record in a table, while a Secondary Key is an additional key used for searching or indexing but does not enforce uniqueness. |
| Provide a real-world example of a one-to-one relationship | A passport and a citizen – each citizen has only one passport, and each passport belongs to only one citizen. |
| Provide a real-world example of a one-to-many relationship | A teacher and students – one teacher can teach many students, but each student has only one main teacher.  Mother and children- one mother can have multiple children. |
| Provide a real-world example of a many-to-many relationship | A student and courses – a student can enroll in many courses, and each course can have many students.  An actor can have many 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 stores data in structured tables with relationships, while a non-relational database (NoSQL) stores data in documents, key-value pairs, or graphs without strict relationships. |
| What type of data would benefit off the non-relational model?  Why? | Unstructured or semi-structured data, such as social media posts, videos, IoT sensor data, and real-time analytics.  Non-relational databases handle large volumes of data efficiently, offer flexibility, and are scalable for dynamic applications like e-commerce, big data, and cloud-based services. |

# 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 JOIN where a table is joined with itself. Used to compare rows within the same table. |
| Right join | Returns all records from the right table and matching records from the left table. |
| Full join | Returns all records when there is a match in either left or right table. |
| Inner join | Returns only the matching rows between two tables. |
| Cross join | Each row from Table A is combined with each row from Table B. |
| Left join | Returns all records from the left table and matching records from the right table. |

# 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. Inventory Data    * Product details-    * Name of product - Milk, Bread, Cake, Detergent, Egg    * Price – Milk = £2.00, Bread- £1.00, Detergent £5.99  , Eggs  £2.00,  Cake £20   name, category, price   * Stock levels (quantity in stock). Calculated field ( starting quantity- current quantity) * Quantity – Milk =  , Bread- , Detergent , Eggs  , Cake * Supplier information (who provides each product).  1. Sales Data    * Date and time of each sale.    * List of products sold in each sale.    * Quantity of Products left.    * Total amount spent by customers. 2. Customer Data    * Customer details (name, contact information).         CustomerFrist-name , Customerlast-name ,Customer\_id, Email, Phone   * Loyalty program points (if applicable).      Code to create the database   CREATE DATABASE SmallRetailBusiness; use SmallRetailBusiness; show tables;  Code to create the tables CREATE TABLE Products (     ProductID INT PRIMARY KEY AUTO\_INCREMENT,     Name VARCHAR(255) NOT NULL,     StockQuantity INT NOT NULL,     Price DECIMAL(5,2) NOT NULL );    CREATE TABLE Customers (     CustomerID INT PRIMARY KEY AUTO\_INCREMENT,     Name VARCHAR(255) NOT NULL,     Email VARCHAR(255) UNIQUE NOT NULL,     LoyaltyPoints INT DEFAULT 0 );    CREATE TABLE Sales (     SaleID INT PRIMARY KEY AUTO\_INCREMENT,     SaleDate DATE NOT NULL,     CustomerID INT,     ProductID INT,     Quantity INT NOT NULL,     TotalPrice DECIMAL(5,2) NOT NULL,     FOREIGN KEY (CustomerID) REFERENCES Customers(CustomerID),     FOREIGN KEY (ProductID) REFERENCES Products(ProductID) );    CREATE TABLE Employees (     EmployeeID INT PRIMARY KEY AUTO\_INCREMENT,     Name VARCHAR(255) NOT NULL,     Role VARCHAR(255) NOT NULL,     Permissions TEXT NOT NULL );    CREATE TABLE Employees (     EmployeeID INT PRIMARY KEY AUTO\_INCREMENT,     Name VARCHAR(255) NOT NULL,     Role VARCHAR(255) NOT NULL,     Permissions TEXT NOT NULL );    DELIMITER $$   CREATE TRIGGER UpdateLoyaltyPoints   AFTER INSERT ON Sales   FOR EACH ROW   BEGIN       UPDATE Customers       SET LoyaltyPoints = LoyaltyPoints + NEW.TotalPrice       WHERE CustomerID = NEW.CustomerID;   END$$   DELIMITER ;     1. Shop Manager    * Track inventory levels -need to track stock and sales .    * Analyse sales trends to make informed decisions.    * Check customer loyalty points for rewards and discounts 2. Shop Staff    * Record transactions at the point of sale .    * Check stock availability and update inventory.    * Check customer loyalty points for rewards and discounts. 3. IT Support Team    * Support and secure the database to ensure smooth operation.    * Perform backups to prevent data loss.    * Optimize database performance.     Step 1: Creating the Database    The first step in implementing the system is to create the database itself. This database will serve as a container for all the tables and information related to the business.   * using MySQL, open a database management tool like MySQL Workbench and enter a command to create the database.       After creating the database, the next step is to define tables to store different types of  data.  Inventory Products: Milk, Bread, Egg, Cake, Detergent.  Milk                     1.50           100quantity  Bread                1.00              200quantity  Detergent        5.99                500quantity  Egg                     2.00               300quantity  Cake                  20.00             15.00quantity   1. Inventory Data    * Product details name,    * Product ID    * Product Name    * Starting Quantity    * Current Quantity    * Customer ID    * Price    * Supplier information (who provides each product). 2. Sales Data    * Date and time of each sale.    * List of products sold in each sale.    * Total amount spent by customers. 3. Customer Data    * Customer details (name, contact information).    * Loyalty program points (if applicable). 4. Supplier Data    * Supplier names and contact details.    * Products they supply to the store.      1. Inventory Data    * Product details-    * Name of product - Milk, Bread, Cake, Detergent, Egg    * Price – Milk = £2.00 , Bread- £1.00, Detergent £5.99  , Eggs  £2.00,  Cake £20   name, category, price   * Stock levels (quantity in stock). Calculated field ( starting quantity- current quantity) * Quantity – Milk =  , Bread- , Detergent , Eggs  , Cake * Supplier information (who provides each product).  1. Sales Data    * Date and time of each sale.    * List of products sold in each sale.    * Quantity of Products left.    * Total amount spent by customers. 2. Customer Data    * Customer details (name, contact information).        CustomerFrist-name , Customerlast-name ,Customer\_id, Email, Phone   * Loyalty program points (if applicable). * The shop staff record transactions and check loyalty card. * IT Support Maintain the data base and make sure everything run smoothly.       3.**Implementing the database.**  To Create a data base with the MYSQL as script   CREATE DATABASE SmallRetailBusiness; use SmallRetailBusiness; show tables;    CREATE TABLE Products (     ProductID INT PRIMARY KEY AUTO\_INCREMENT,     Name VARCHAR(255) NOT NULL,     StockQuantity INT NOT NULL,     Price DECIMAL(5,2) NOT NULL );    CREATE TABLE Customers (     CustomerID INT PRIMARY KEY AUTO\_INCREMENT,     Name VARCHAR(255) NOT NULL,     Email VARCHAR(255) UNIQUE NOT NULL,     LoyaltyPoints INT DEFAULT 0 );    CREATE TABLE Sales (     SaleID INT PRIMARY KEY AUTO\_INCREMENT,     SaleDate DATE NOT NULL,     CustomerID INT,     ProductID INT,     Quantity INT NOT NULL,     TotalPrice DECIMAL(5,2) NOT NULL,     FOREIGN KEY (CustomerID) REFERENCES Customers(CustomerID),     FOREIGN KEY (ProductID) REFERENCES Products(ProductID) );    CREATE TABLE Employees (     EmployeeID INT PRIMARY KEY AUTO\_INCREMENT,     Name VARCHAR(255) NOT NULL,     Role VARCHAR(255) NOT NULL,     Permissions TEXT NOT NULL );    CREATE TABLE Employees (     EmployeeID INT PRIMARY KEY AUTO\_INCREMENT,     Name VARCHAR(255) NOT NULL,     Role VARCHAR(255) NOT NULL,     Permissions TEXT NOT NULL );    DELIMITER $$   CREATE TRIGGER UpdateLoyaltyPoints   AFTER INSERT ON Sales   FOR EACH ROW   BEGIN       UPDATE Customers       SET LoyaltyPoints = LoyaltyPoints + NEW.TotalPrice       WHERE CustomerID = NEW.CustomerID;   END$$   DELIMITER ;  5. Maintaining the Database   1. Regular Updates  * Updating Inventory After a Sale: Whenever a product is sold, the inventory must be adjusted to reflect the new stock levels.   + We need a trigger to calculate new stock and automatically reduce the stock levels (quantity).   + Updating the records by staff should make this activity happen. * Updating Customer Loyalty Points: If a store has a loyalty program, customers should earn points after every purchase. The database must update these points accordingly.   Monitoring Data Quality     * Checking for Duplicate Customers * Ensuring Product Information is Complete and correct.   + This is responsibility IT support to check that this automatic “trigger” is functioning.   Maintaining high data quality improves business efficiency and prevents errors that could impact customers or sales.    3. Backing Up the Database     * Creating Automatic Backups By maintaining regular backups, the business can recover any lost data quickly and continue operations without disruption. * There needs to be redundancy in the backups. Cloud back up can be done daily in the morning and an on-site backup storage can be in the manager’s office.     4.   Data Security     * Restricting Access to Sensitive Data: Only managers should have the ability to change product prices, while cashiers should only be able to process sales transactions. * IT Support will have full access but authority level approval should be in place to control the change of any sensitive data such as price, staff information, customer information and more.      * Securing Customer Information:      Customer data, including email addresses and phone numbers, should be protected using Data protection act and GDPR as a policy.  Only authorized personnel should be able to access or update this information.  And these authorised personnel should be the manager and the data owners.  For regular updates of new stock, there will be a mechanism  Example: After selling 2 bottles of milk, reduce the stock level from 10 to 8 in the Products Table.  2. Monitor Data Quality: o Check for errors like duplicate customer entries or missing product  details.  3. Backups:  o Export the database weekly as a backup file (e.g., save it as an Excel file or database file).  4. Data Security:  o Restrict access so only authorised staff can edit tables. For example, the manager can update product prices, but staff can only record sales.  Appendix  /\* Create and use a database\*/ CREATE DATABASE SmallRetailBusiness; use SmallRetailBusiness;    /\* Create tables \*/ CREATE TABLE Products (     ProductID INT PRIMARY KEY AUTO\_INCREMENT,     Name VARCHAR(255) NOT NULL,     StockQuantity INT NOT NULL,     Price DECIMAL(5,2) NOT NULL );    CREATE TABLE Customers (     CustomerID INT PRIMARY KEY AUTO\_INCREMENT,     Name VARCHAR(255) NOT NULL,     Email VARCHAR(255) UNIQUE NOT NULL,     LoyaltyPoints INT DEFAULT 0 );    CREATE TABLE Sales (     SaleID INT PRIMARY KEY AUTO\_INCREMENT,     SaleDate DATE NOT NULL,     CustomerID INT,     ProductID INT,     Quantity INT NOT NULL,     TotalPrice DECIMAL(5,2) NOT NULL,     FOREIGN KEY (CustomerID) REFERENCES Customers(CustomerID),     FOREIGN KEY (ProductID) REFERENCES Products(ProductID) );    CREATE TABLE Employees (     EmployeeID INT PRIMARY KEY AUTO\_INCREMENT,     Name VARCHAR(255) NOT NULL,     Role VARCHAR(255) NOT NULL,     Permissions TEXT NOT NULL );    CREATE TABLE Employees (     EmployeeID INT PRIMARY KEY AUTO\_INCREMENT,     Name VARCHAR(255) NOT NULL,     Role VARCHAR(255) NOT NULL,     Permissions TEXT NOT NULL );    /\* Create triggers \*/    DELIMITER $$   CREATE TRIGGER UpdateLoyaltyPoints   AFTER INSERT ON Sales   FOR EACH ROW   BEGIN       UPDATE Customers       SET LoyaltyPoints = LoyaltyPoints + NEW.TotalPrice       WHERE CustomerID = NEW.CustomerID;   END$$   DELIMITER ;    DELIMITER $$ CREATE TRIGGER UpdateStockAfterSale AFTER INSERT ON Sales FOR EACH ROW BEGIN     UPDATE Products     SET StockQuantity = StockQuantity - NEW.Quantity     WHERE ProductID = NEW.ProductID; END$$ DELIMITER ;    DELIMITER $$ CREATE TRIGGER PreventIncompleteProduct BEFORE INSERT ON Products FOR EACH ROW BEGIN     IF NEW.ProductName IS NULL OR NEW.Price IS NULL OR NEW.StockQuantity IS NULL THEN         SIGNAL SQLSTATE '45000'         SET MESSAGE\_TEXT = 'Erro: Produto com informações faltando!';     END IF; END$$ DELIMITER ;    /\* Create events\*/ DELIMITER $$ CREATE EVENT CheckDataQuality ON SCHEDULE EVERY 1 DAY DO BEGIN     -- Registrar problemas encontrados na tabela de auditoria     INSERT INTO DataQualityLog (IssueType, IssueDetails, CheckDate)     SELECT 'Duplicate Customer', Email, NOW()     FROM DuplicateCustomers;      INSERT INTO DataQualityLog (IssueType, IssueDetails, CheckDate)     SELECT 'Incomplete Product', ProductID, NOW()     FROM IncompleteProducts; END$$ DELIMITER ; |

# 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.

|  |
| --- |
| SELECT COUNT(\*) FROM city  WHERE CountryCode = 'USA';  A screenshot of a computer  AI-generated content may be incorrect. |

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 FROM country WHERE LifeExpectancy IS NOT NULL ORDER BY LifeExpectancy DESC LIMIT 1;  A screenshot of a computer  AI-generated content may be incorrect. |

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%';  A screenshot of a computer  AI-generated content may be incorrect. |

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 \* FROM city ORDER BY Population DESC LIMIT 10;  A screenshot of a computer  AI-generated content may be incorrect. |

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 \* FROM city WHERE Population > 2000000;  A computer screen shot of a computer screen  AI-generated content may be incorrect. |

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%';  A screenshot of a computer  AI-generated content may be incorrect. |

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 \* FROM city WHERE Population >= 500000 AND Population <= 1000000  A screenshot of a computer  AI-generated content may be incorrect. |

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;  A screenshot of a computer  AI-generated content may be incorrect. |

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;  A screenshot of a computer  AI-generated content may be incorrect. |

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;  A screenshot of a computer  AI-generated content may be incorrect. |

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 FROM city WHERE population = (SELECT MIN(population) FROM city WHERE population > 0);  A screenshot of a computer  AI-generated content may be incorrect. |

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;  A screenshot of a computer  AI-generated content may be incorrect. |

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 c.name from city c  join country co on c.id = co.capital  where co.name = 'spain';  A screenshot of a computer  AI-generated content may be incorrect. |

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 from country  where lifeexpectancy is not null  order by lifeexpectancy desc limit 1;  A screenshot of a computer  AI-generated content may be incorrect. |

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');  A screenshot of a computer  AI-generated content may be incorrect. |

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) from city group by countrycode;  A screenshot of a computer  AI-generated content may be incorrect. |

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 c.name, c.population  from city c join country co on c.id = co.capital; |

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 avg(population / surfacearea) from country; select name from country  where population / surfacearea < (select avg(population / surfacearea) from country);  A screenshot of a computer  AI-generated content may be incorrect. |

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 city.Name AS CityName,         country.Name AS CountryName,        city.Population AS CityPopulation,        country.GNP AS CountryGNP,        (country.GNP \* 1000000000 / country.Population) AS GDPPerCapita FROM city JOIN country ON city.CountryCode = country.Code WHERE (country.GNP \* 1000000000 / country.Population) >        (SELECT AVG(GNP \* 1000000000 / Population)         FROM country         WHERE Population > 0 AND GNP > 0) ORDER BY GDPPerCapita DESC;  A screenshot of a computer  AI-generated content may be incorrect. |

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 city.ID, city.Name, city.CountryCode, city.District, city.Population,        country.Name AS CountryName, country.Continent, country.Region FROM city JOIN country ON city.CountryCode = country.Code ORDER BY city.Population DESC LIMIT 30, 10;  A screenshot of a computer  AI-generated content may be incorrect. |

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